POLITECNICO DI TORINO
Corso di Laurea Magistrale in Ingegneria Gestionale

Agile Development e crunch nell'industria videoludica: riduzione o mascheramento?

Relatore

Filippo Maria Ottaviani

Candidato

Giacomo Vivarelli

Anno accademico 2025-2026

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

1.1 Esistenza del problema

Negli ultimi decenni il settore dei videogiochi è diventato uno dei più importanti nell'ambito dell'intrattenimento digitale tanto da «essere uno dei segmenti in più rapida crescita del settore E&M, con ricavi totali pari a 227,6 miliardi di dollari nel 2023. I ricavi sono destinati a superare i 300 miliardi di dollari nel 2027» (Capobianco, 2024) superando il fatturato del settore cinematografico e musicale messi assieme. Anche in Italia, il settore sta registrando una costante crescita sia occupazionale che economica. Parallelamente alla crescita del settore videoludico, si è intensificata anche la presenza del fenomeno del crunch durante lo sviluppo e produzione del videogioco.

La definizione di crunch si può riassumere come un periodo di lavoro estremamente intenso, solitamente durante la fase di produzione o prima di importanti milestone di progetto. Il crunch è un fenomeno che esiste sin dagli albori dello sviluppo di videogiochi, già nel 2004 la pubblicazione della lettera anonima di EA_spouse (2004) – moglie di uno sviluppatore di Electronic Arts – portò per la prima volta all'attenzione pubblica le condizioni di lavoro nell'industria: settimane da 60-90 ore, weekend sacrificati, burnout sistematici e l'assenza di compensi economiche proporzionati. A distanza di vent'anni, la situazione non appare sostanzialmente mutata. Durante i periodi di crunch gli sviluppatori sono spesso costretti a fare ore di straordinari, spesso non retribuiti, per cercare di rientrare nei tempi prefissati e obiettivi di produzione.

Le prove dell'esistenza e della persistenza del crunch sono numerose e provengono da fonti

eterogenee: il *Game Developers Conference State of the Industry* del 2019 rivelò che circa il 59% degli sviluppatori intervistati aveva sperimentato crunch nell'anno precedente; tra questi, il 68% non aveva ricevuto alcuna compenso economica aggiuntivo. Il report del 2023 mostra segnali di miglioramento solo parziali, con il crunch che rimane endemico soprattutto negli studi AAA.

Tra gli episodi più noti ci sono lo sviluppo di *Red Dead Redemption 2* (Rockstar Games, 2018), durante il quale il co-fondatore Dan Houser dichiarò settimane da 100 ore, salvo poi ritrattare parzialmente; il crunch massiccio riportato durante la produzione di *Cyberpunk 2077* (CD Projekt Red, 2020), culminato in un lancio disastroso e una class action degli azionisti; infine i licenziamenti post-lancio e successivamente la chiusura di *Anthem* (BioWare, 2019), descritti nel dettaglio da Bloomberg come conseguenza di una cultura aziendale tossica fondata sul crunch cronico. Ricerche come quella di Weststar & Legault (2017b) e di O'Donnell (2014b) documentano correlazioni dirette tra crunch e burnout, ansia, depressione e abbandono del settore, con tassi di turnover particolarmente elevati tra sviluppatori junior e donne.

Le cause del crunch sono molteplici e includono fattori organizzativi, economici e culturali. In molti casi il cruch viene utilizzato come strumento per ridurre i costi di sviluppo o per evitare il rinvio della data di uscita del gioco, soprattutto nei progetti ad alto budget dove ritardi e cambiamenti possono avere impatti economici significativi. Le conseguenze del fenomeno sono rilevanti sia sul piano individuale sia su quello organizzativo. Diversi studi e testimonianze di sviluppatori evidenziano effetti negativi sul benessere fisico e psicologico dei lavoratori, come stress, burnout e aumento del turnover. Inoltre, periodi prolungati di lavoro intensivo possono compromettere la qualità del prodotto finale e la sostenibilità dei team di sviluppo nel lungo periodo.

Negli ultimi anni, per affrontare questi problemi, molte aziende del settore hanno iniziato a introdurre metodologie di sviluppo più flessibili, tra cui le pratiche Agile, con l'obiettivo di migliorare la pianificazione dei progetti, favorire cicli iterativi più brevi e ridurre il rischio di sovraccarico lavorativo nelle fasi finali dello sviluppo. Secondo alcune ricerche sui processi di sviluppo dei videogiochi, circa il 45% dei progetti analizzati utilizza esplicitamente pratiche Agile, mentre modelli più tradizionali come il waterfall continuano comunque a essere presenti. Nonostante questa crescente diffusione, rimane aperta una questione centrale:

L'adozione dell'Agile contribuisce realmente a ridurre il fenomeno del crunch oppure tende semplicemente a redistribuirlo e renderlo meno visibile nel processo di sviluppo?

Come osservano Peticca-Harris et al. (2015), l'Agile può innescare meccanismi di controllo normativo: i lavoratori, avendo negoziato autonomamente gli sprint e le stime, si sentono moralmente obbligati a rispettare impegni autoimposti, riproducendo il crunch per pressione interna piuttosto che per coercizione esplicita. Il problema smette di essere visibile come tale, ma non smette di esistere, ed è mia intenzione in questo lavoro affrontarlo sia da un punto di vista teorico che da un punto di vista pratico: sul piano teorico, ci interrogheremo sulla natura dell'Agile come framework di project management, verificando se i suoi assunti fondamentali siano compatibili con le specificità dell'industria videoludica (creatività non lineare, scope creep, dipendenze tecnologiche imprevedibili). Sul piano pratico, valuteremo se l'adozione dell'Agile abbia prodotto miglioramenti misurabili nelle condizioni lavorative degli sviluppatori o se abbia invece generato nuove forme di auto-sfruttamento difficili da riconoscere e contrastare.

1.2 Importanza del problema

Non è possibile ricondurre le situazioni relative al crunch a problematiche interne di un solo settore: le sue conseguenze infatti attraversano diversi ambiti che variano dalla sociologia del lavoro fino alla parte economica-creativa, dall'organizzazione aziendali alle diverse politiche pubbliche sui lavori digitali. Questo fenomeno all'interno del settore videoludico rappresenta una problematica di crescente importanza sia a livello professionale che accademico, in quanto si colloca all'intersezione tra organizzazione del lavoro, sviluppo software e sostenibilità delle pratiche produttive.

Dal punto di vista degli studiosi, il crunch costituisce un caso emblematico per analizzare i limiti delle metodologie di sviluppo software in contesti altamente creativi e incerti come quello videoludico. In particolare, il dibattito sull'efficacia delle metodologie Agile si intreccia con la necessità di comprendere se tali approcci siano realmente in grado di migliorare la sostenibilità dei processi lavorativi o se, al contrario, contribuiscano a ridefinire forme di pressione temporale meno visibili ma comunque pervasive.

Inoltre, l'industria videoludica costituisce un laboratorio unico per lo studio delle trasformazioni del lavoro creativo nell'economia digitale. Come argomentano Dyer-Witheford e De Peuter nel loro *Games of Empire* (Dyer-Witheford & De Peuter, 2009), il settore dei videogiochi anticipa dinamiche presenti in molte industrie creative come la flessibilizzazione del lavoro e la fusione tra passione e sfruttamento. La letteratura organizzativa ha dedicato attenzione crescente al concetto di "passionate work exploitation" (Tokumitsu, 2015): lavoratori che, proprio perché amano il proprio lavoro, accettano condizioni che non accetterebbero in settori percepiti come meno vocazionali. Sul versante della ricerca sull'Agile, il contesto videoludico offre un caso limite particolarmente illuminante. La maggior parte degli studi sull'Agile si concentra su software enterprise o startup tecnologiche, ambienti in cui i requisiti, pur mutevoli, sono prevalentemente funzionali. Lo sviluppo di videogiochi introduce variabili qualitativamente diverse che mettono sotto stress i presupposti fondamentali dei framework Agile. Studiare questo caso consente di testare i limiti di validità di teorie organizzative che tendono a presentarsi come universali.

A livello organizzativo la posta in gioco per i game developer e i responsabili degli studi è concreta e urgente. Il crunch ha costi diretti che l'industria fatica ancora a riconoscere pienamente, e che si articolano su tre dimensioni strettamente interconnesse: i tempi di sviluppo, i costi produttivi e la disponibilità del talento.

- Sul piano dei tempi, il crunch genera un accumulo progressivo di debito tecnico e bug sistemici che, come mostra l'analisi di Schreier in *Blood, Sweat, and Pixels* (Schreier, 2017), si traduce inevitabilmente in patch post-lancio costose e in slittamenti che compromettono i cicli produttivi successivi. Paradossalmente, i periodi di intensificazione lavorativa non garantiscono prodotti migliori né tempi più rapidi: la qualità del codice prodotto sotto pressione è strutturalmente inferiore, e il tempo guadagnato nel breve periodo viene spesso perso – e moltiplicato – nelle fasi di correzione e consolidamento. Per i project manager, questa dinamica ha implicazioni metodologiche immediate: adottare un framework senza comprenderne i limiti strutturali rischia di produrre un falso senso di controllo, ritardando interventi necessari e scaricando la responsabilità del crunch sui team stessi, che vengono così percepiti come il problema anziché come le vittime di un sistema mal configurato.

- Sul piano dei costi, le conseguenze economiche del crunch sono tanto reali quanto si-

stematicamente sottovalutate. Ricerche come quelle di Weststar, Legault & O'Meara (2017) mostrano che una quota significativa di sviluppatori abbandona il settore entro cinque anni dall'ingresso, portando con sé competenze difficilmente sostituibili e know-how accumulato nel tempo. La sostituzione di figure senior non è mai neutrale: richiede tempo, risorse di onboarding e un periodo di adattamento durante il quale la produttività del team risulta inevitabilmente ridotta. A questo si aggiungono i costi diretti legati alle patch post-lancio, alle revisioni straordinarie e alla gestione delle crisi reputazionali che spesso seguono i lanci affrettati. La perdita continua di talenti senior costringe gli studi ad affidarsi a sviluppatori junior meno esperti, aumentando il rischio di errori e ritardi che a loro volta generano nuovo crunch, alimentando un circolo vizioso che tende ad auto-rafforzarsi in assenza di interventi strutturali (Kerr, 2006).

- Sul piano della produttività e della disponibilità del talento, un settore in crunch cronico tende a selezionare i lavoratori sulla base della resistenza fisica e della disponibilità a sacrificare la vita privata, piuttosto che per la creatività o per le competenze tecniche. Questo meccanismo di selezione implicita produce un bias sistemico che colpisce in misura sproporzionata donne, genitori, persone con disabilità o condizioni croniche (Gray & Leonard, 2018), restringendo progressivamente il bacino di profili disponibili e riducendo la diversità creativa all'interno dei team. In un settore in cui l'innovazione e la qualità narrativa ed estetica rappresentano fattori competitivi fondamentali, l'omogenizzazione del talento si traduce direttamente in una riduzione della qualità dei prodotti e in una minore capacità di rispondere alle aspettative di un pubblico sempre più eterogeneo.

A livello individuale, il crunch cronico è associato a conseguenze documentate sulla salute mentale e fisica: esaurimento, disturbi del sonno, deterioramento delle relazioni personali, fino a episodi di depressione clinica. Come evidenzia il report IGDA del 2021, circa il 40% degli sviluppatori considera di abbandonare il settore entro due anni, e tra le motivazioni principali figura il sovraccarico lavorativo non sostenibile. Questa intenzione di abbandono non riguarda soltanto i profili più fragili o meno motivati, ma colpisce trasversalmente anche sviluppatori esperti e altamente qualificati, che dispongono di alternative occupazionali e scelgono di esercitarle quando le condizioni lavorative diventano insostenibili. La dimensione individuale del crunch è quindi inseparabile da quella organizzativa: ogni sviluppatore che lascia il settore rappresenta una perdita concreta di capitale umano, relazionale e creativo che nessun processo di recruiting è in grado di compensare pienamente nel breve periodo. Come mostra Kerr in *The*

Business and Culture of Digital Games (Kerr, 2006), questo ciclo tende ad auto-rafforzarsi in assenza di interventi strutturali.
A livello culturale e sociale, la normalizzazione del crunch contribuisce a riprodurre e legittimare una cultura del lavoro basata sul sacrificio come prova di dedizione professionale, una logica che, come argomentano diversi teorici del lavoro immateriale (Lazzarato, 1996; Hardt & Negri, 2000), è tutt'altro che limitata all'industria videoludica, ma trova in essa una delle espressioni più visibili e analizzabili.

In questo contesto, l'adozione di metodologie Agile viene spesso proposta come soluzione, in quanto promette maggiore flessibilità, iterazione continua e migliore gestione dei carichi di lavoro. Tuttavia, alcuni studi suggeriscono che, se implementato in modo superficiale o adattato a contesti organizzativi già problematici, l'Agile potrebbe non eliminare il crunch, ma piuttosto distribuirlo lungo tutto il ciclo di sviluppo, trasformandolo in una pressione costante anziché concentrata. Pertanto, comprendere se l'Agile rappresenti una reale soluzione o una riformulazione del problema è cruciale non solo per migliorare le pratiche di sviluppo, ma anche per garantire la sostenibilità a lungo termine dell'industria videoludica.

1.3 Riepilogo della letteratura classica e contemporanea

Nel corso del tempo, il fenomeno del crunch nell'industria dei videogiochi è stato affrontato attraverso diversi approcci, che riflettono l'evoluzione delle pratiche di sviluppo software e della cultura organizzativa del settore. La letteratura sul crunch nell'industria videoludica si è sviluppata in tre fasi relativamente distinte, riflettendo l'evoluzione del settore stesso.

Prima fase (2004-2010): prendere atto del problema
In una fase iniziale, il crunch veniva considerato una conseguenza inevitabile dei modelli di sviluppo tradizionali, in particolare del paradigma waterfall, caratterizzato da una pianificazione sequenziale e rigida. I primi contributi accademici e giornalistici si concentrarono sulla documentazione empirica del fenomeno. Il già citato caso *ea_spouse* del 2004 innescò una prima ondata di indagini, tra cui i report annuali dell'IGDA (*Developer Satisfaction Survey*), che cominciarono a quantificare la diffusione del crunch e i suoi effetti sui lavoratori. In questo contesto, eventuali ritardi accumulati nelle fasi intermedie del progetto tendevano a concentrar-

si nella fase finale, generando periodi intensivi di lavoro straordinario.
In questi anni la ricerca è prevalentemente descrittiva: si tratta di riconoscere l'esistenza del problema, la sua diffusione e i soggetti coinviloti. Contributi come quelli di Dyer-Witheford & De Peuter (2009) fornirono le prime cornici teoriche, inquadrando il crunch all'interno delle dinamiche più ampie del lavoro immateriale e dell'economia creativa. L'Agile, in questa fase, pur essendo già presente nell'industria videoludica, era considerato nella letteratura quasi esclusivamente come questione tecnica di project management, senza connessione esplicita al tema del benessere lavorativo.

Seconda fase (2010-2018): l'Agile come soluzione emergente

Con l'emergere di metodologie iterative e flessibili, formalizzate nel *Manifesto for Agile Software Development* (Beck et al., 2001), si è sviluppato un nuovo approccio orientato alla riduzione dei rischi legati alla pianificazione. Con la diffusione massiccia di Scrum e metodologie iterative negli studi di medie dimensioni, la letteratura di settore iniziò a esplorare l'applicabilità nella produzione videoludica dell'Agile, il quale introduce cicli di sviluppo brevi, feedback continui e una maggiore adattabilità ai cambiamenti, elementi che teoricamente dovrebbero prevenire l'accumulo di lavoro nelle fasi finali.
I contributi rappresentativi di questa fase sono testi prevalentemente prescrittivi, scritti da practitioner, che adattano i framework Agile alle specificità della produzione videoludica. Tra questi citiamo *Agile Game Development with Scrum* (Keith, 2010). Parallelamente, ricercatori come Peticca-Harris et al. (2015) cominciarono a sollevare dubbi teorici più profondi, introducendo la questione del controllo normativo nei team Agile ovvero la possibilità che l'autoorganizzazione dei team non riducesse la pressione lavorativa ma la redistribuisse, rendendola meno visibile. Questa tensione – Agile come soluzione versus Agile come nuovo vettore di crunch – comincia a strutturare il dibattito, ma rimane ancora ai margini della letteratura dominante.

Terza fase (2018-oggi): critica, complessità e nuove domande

Il periodo post 2018 è segnato da una svolta critica. Il giornalismo d'inchiesta – in particolare il lavoro di Jason Schreier su Kotaku e in *Blood, Sweat, and Pixels* (Schreier, 2017) e *Press Reset* (Schreier, 2021) – portò all'attenzione pubblica una serie di casi concreti in cui studi che adottavano metodologie Agile continuavano a produrre crunch sistemico. Questo aprì una fase

di ricerca più matura, caratterizzata da diverse direttrici principali:

Una parte della letteratura distingue tra un'adozione in toto dei principi Agile – che include autonomia dei team, riduzione dello scope e cultura della sostenibilità – e un'adozione superficiale, che si limita ad adottare il vocabolario (sprint, backlog, velocity) senza modificare le strutture di potere e gli incentivi economici. Ricercatori come Hodgson & Briand (2013) avevano già anticipato questo problema in contesti software non videoludici, mostrando come l'Agile possa essere reintegrato in logiche di controllo manageriale tradizionali.
Inoltre, studi più recenti – tra cui *Developer's Dilemma* di O'Donnell (2014a) e *The Business and Culture of Digital Games* di Kerr (2006) – mettono in discussione l'assunto che i framework nati per il software enterprise siano direttamente trasferibili allo sviluppo videoludico, in quanto componenti come il game feel, l'estetica e la narrativa resistono per natura alla pianificazione a sprint.
Infine, è possibile trovare una linea di ricerca emergente, all'interno della quale si collocano i contributi di Hutchinson (2021) e Weststar & Legault (2022), che sposta il fuoco dall'organizzazione interna agli studi alle pressioni strutturali esterne: i publisher, i cicli delle console, le finestre di lancio commerciale e i modelli di finanziamento. In questa prospettiva, nessuna metodologia interna – Agile incluso – può risolvere il crunch finché le sue cause basilari rimangono nei contratti con i publisher e nelle aspettative del mercato.

Il dibattito è attualmente aperto e non risolto. Esiste un consenso crescente sul fatto che il crunch non sia eliminabile attraverso la sola adozione di metodologie Agile, ma permane disaccordo su quanto l'Agile possa comunque mitigarlo e in quali condizioni. Mancano studi longitudinali su larga scala che misurino l'effetto dell'adozione Agile sulle ore lavorate nel tempo; la ricerca qualitativa è abbondante ma difficilmente generalizzabile; e il settore stesso è in rapida trasformazione, con l'ascesa dei modelli games-as-a-service che introducono nuove forme di crunch post-lancio ancora poco studiate.
Negli ultimi anni, lo stato dell'arte si è quindi evoluto verso una visione più articolata del problema. Il crunch non è più interpretato esclusivamente come un fallimento della pianificazione, ma come un fenomeno complesso che coinvolge metodologie di sviluppo, cultura organizzativa e condizioni del mercato del lavoro. Le ricerche attuali tendono a concentrarsi sull'interazione tra questi fattori piuttosto che su soluzioni puramente tecniche. È in questo spazio aperto che la

presente ricerca si inserisce, concentrandosi sugli interrogativi che la letteratura non ha ancora risolto con sistematicità: l'Agile riduce il crunch o ne sposta e maschera le dinamiche?
Nel Capitolo 2 si offrirà una panoramica nelle sue due componenti fondamentali: da un lato si affronterà cosa sia effettivamente l'Agile – origini, principi, framework principali e adattamenti al mondo videoludico – e dall'altro, come si struttura lo sviluppo di un videogioco nelle sue fasi produttive, per comprendere dove e perché il crunch tende a manifestarsi e in quali punti l'Agile interviene (o promette di intervenire).

1.4 Gap

Nonostante i contributi significativi prodotti negli ultimi vent'anni – dalla documentazione empirica del crunch alle prime teorizzazioni critiche sull'Agile, fino alle indagini giornalistiche di Schreier e agli studi organizzativi di O'Donnell, Peticca-Harris e Legault & Weststar – la letteratura esistente presenta una lacuna strutturale che rende necessaria un'indagine più mirata.

Ad oggi non esiste un'analisi sistematica delle implicazioni che le modalità di gestione di progetto esercitano sul fenomeno del crunch, inteso come deriva della performance di progetto. Questa assenza non è accidentale: riflette una frammentazione disciplinare profonda, in cui gli studi sul crunch e quelli sulle metodologie di sviluppo hanno percorso strade parallele che raramente si intersecano in modo organico. Da un lato, la ricerca sul crunch – prevalentemente sociologica ed etnografica – documenta con precisione le condizioni lavorative degli sviluppatori, ma tende a trattare le metodologie di produzione come variabili di contorno, analizzandole al più in senso descrittivo senza mai interrogarne sistematicamente il ruolo causale. Dall'altro, la letteratura sull'Agile applicato ai videogiochi – prevalentemente prodotta da practitioner come Keith (2010) – è di natura prescrittiva e si concentra sull'efficienza produttiva, ignorando quasi sistematicamente le implicazioni per il benessere lavorativo e per la sostenibilità dei processi nel medio e lungo periodo. Manca, in altri termini, uno studio che analizzi in modo integrato come specifiche scelte di project management interagiscano con le dinamiche di crunch nelle diverse fasi dello sviluppo videoludico, mettendo in relazione le variabili organizzative con quelle relative alle condizioni di lavoro. La presente tesi si propone di colmare precisamente questo spazio.

- Una prima conseguenza di questa frammentazione è che la domanda centrale – se speci-

fiche scelte di gestione di progetto producano, riducano o trasformino il crunch – non è mai stata posta in questi termini. La questione se l'Agile riduca il crunch o lo mascheri in forme meno visibili è stata al più accennata dalla letteratura esistente, ma non sviluppata in modo teoricamente fondato. Peticca-Harris et al. (2015) introducono il concetto di controllo normativo nei team Agile, suggerendo che l'auto-organizzazione possa generare pressione interna sostitutiva della coercizione esplicita. Hodgson & Briand (2013) mostrano, in contesti software non videoludici, come i framework Agile possano essere reintegrati in logiche manageriali tradizionali. Tuttavia, nessuno di questi contributi sviluppa una cornice analitica coerente che permetta di distinguere operativamente tra riduzione autentica del crunch e sua trasformazione in forme meno visibili. Questa mancanza ha conseguenze concrete: senza strumenti concettuali adeguati, studiosi e ricercatori non sono in grado di valutare criticamente le proprie pratiche, né di distinguere tra un'adozione Agile genuinamente trasformativa e una che riproduce il problema sotto nuove vesti. Ciò che questo lavoro si propone è articolare questa distinzione in modo teoricamente fondato, attingendo alla letteratura sul controllo organizzativo, sul lavoro affettivo e sulla gestione dei knowledge worker.

- Una seconda conseguenza riguarda la dimensione temporale del fenomeno. La quasi totalità della ricerca esistente concettualizza il crunch come fenomeno legato alla fase finale di sviluppo di un titolo, il cosiddetto gold rush verso la data di lancio. Se questo inquadramento era adeguato per l'era dei giochi retail con cicli di sviluppo discreti e definiti, risulta progressivamente inadeguato di fronte alla trasformazione dell'industria verso i modelli games-as-a-service (GaaS). Titoli come *Fortnite*, *Destiny 2* o *Final Fantasy XIV* hanno sostituito il ciclo sviluppo-lancio-fine con un flusso continuo di aggiornamenti, stagioni e contenuti che non prevede, per definizione, una conclusione. In questo contesto, il crunch non si concentra più in un picco prevedibile ma si distribuisce in modo potenzialmente permanente lungo l'intero ciclo di vita del prodotto – una condizione che Weststar & Legault (2022) nominano ma non analizzano sistematicamente. L'Agile, con i suoi sprint ricorrenti, sembrerebbe strutturalmente compatibile con questi modelli; ma se ogni sprint porta con sé pressione da consegna, il risultato potrebbe essere un crunch non più episodico ma cronico e normalizzato per design. Questo scenario, quasi completamente assente nella letteratura accademica, rappresenta una delle aree in cui il presente lavoro intende offrire un contributo esplorativo, interrogandosi su

come le logiche di gestione iterativa interagiscano con la natura potenzialmente infinita dei prodotti live-service.

- Una terza conseguenza riguarda i soggetti che la ricerca ha sinora privilegiato. La letteratura esistente tende a polarizzarsi su due categorie: i senior developer e lead – spesso intervistati come voci autorevoli sulle dinamiche di produzione – e i developer junior, citati come vittime principali del crunch nelle prime fasi di carriera. La prospettiva dei middle manager, degli Scrum master e dei producer – coloro che si trovano a implementare l'Agile nella pratica quotidiana, mediando tra le pressioni dei publisher e le esigenze dei team – è sistematicamente sottorappresentata. Eppure è precisamente in questa fascia intermedia che si gioca buona parte della partita: sono queste figure a decidere se e come applicare i principi Agile, a gestire le tensioni tra pianificazione e imprevedibilità creativa, e a tradurre – o non tradurre – i valori dichiarati dello studio in pratiche concrete. La loro assenza nella letteratura costituisce un punto cieco significativo per chiunque voglia capire perché specifiche modalità di gestione di progetto producano crunch in certi contesti e riescano invece ad arginarlo in altri. Interrogare questa fascia intermedia significa, in ultima analisi, interrogare il meccanismo attraverso cui le scelte organizzative si trasformano in condizioni di lavoro vissute.

L'insieme di queste considerazioni converge verso un'unica lacuna di fondo: la mancanza di un quadro interpretativo che analizzi il crunch non come fenomeno culturale autonomo né come effetto collaterale di scelte tecniche, ma come esito – prevedibile o evitabile – di precise modalità di organizzazione e gestione del lavoro di progetto. La motivazione di questa ricerca è, in ultima analisi, sia critica che costruttiva. Critica, perché si propone di smontare una narrazione manageriale che tende a presentare l'Agile come soluzione tecnica a un problema che è invece strutturale; costruttiva, perché l'obiettivo non è delegittimare le metodologie iterative – che possono offrire strumenti genuinamente utili – ma identificare con maggiore precisione le condizioni in cui esse producono miglioramenti reali, e quelle in cui rischiano invece di diventare uno strumento di legittimazione che lascia invariate le cause profonde del problema.

1.5 Obiettivi

La ricerca si propone di rispondere a una domanda centrale, articolata in tre quesiti che corrispondono alle lacune identificate nella letteratura. La domanda di ricerca principale risulta

essere la seguente:

L'adozione di metodologie Agile negli studi di sviluppo videoludico costituisce una risposta strutturalmente efficace al problema del crunch, oppure ne trasforma le manifestazioni senza eliminarne le cause?

Questa domanda non presuppone una risposta binaria. L'ipotesi di lavoro è che l'Agile possa simultaneamente ridurre alcune forme di crunch – in particolare quelle legate alla disorganizzazione progettuale e alla mancanza di trasparenza sullo stato di avanzamento – e riprodurne o mascherarne altre, specialmente quelle radicate nelle strutture economiche dell'industria, nella cultura organizzativa degli studi e nei meccanismi di controllo normativo che i framework iterativi possono involontariamente generare. A partire da questo obiettivo generale, la ricerca si articola nei seguenti obiettivi specifici:

1. **Obiettivo 1** - Ricostruire in modo sistematico le fasi dello sviluppo videoludico e identificare i momenti e le condizioni in cui il crunch tende a manifestarsi.

2. **Obiettivo 2** - Esaminare i principi fondamentali dell'Agile – così come i principali framework applicati all'industria videoludica – valutandone la compatibilità teorica con le specificità del processo creativo di sviluppo, con particolare attenzione ai punti di tensione e alle aree di adattamento necessario.

3. **Obiettivo 3** - Attraverso l'analisi di casi studio selezionati, verificare se e come l'adozione dell'Agile abbia prodotto cambiamenti misurabili nelle condizioni lavorative degli sviluppatori, prestando attenzione alle voci dei livelli intermedi della gerarchia organizzativa sistematicamente assenti nella letteratura.

4. **Obiettivo 4** -Identificare le condizioni organizzative, culturali ed economiche in cui l'Agile può effettivamente contribuire a ridurre il crunch, formulando indicazioni operative per studi, ricercatori e, in prospettiva, per chi si occupa di regolazione del lavoro nel settore creativo digitale.

Sulla base di tali obiettivi, la ricerca è guidata dai seguenti quesiti:

- *Sul piano organizzativo, in quali fasi del ciclo di sviluppo videoludico l'Agile dimostra maggiore efficacia nel prevenire il crunch, e in quali tende invece a fallire o a produrre effetti controproducenti?*

- *Sul piano dei meccanismi di controllo, attraverso quali meccanismi l'Agile può trasformare il crunch da fenomeno imposto esternamente a fenomeno auto-generato dai team, e in che misura questo cambiamento rappresenta un miglioramento reale delle condizioni lavorative?*
- *Sul piano strutturale, in che misura le pressioni esterne agli studi – publisher, modelli di finanziamento, aspettative del mercato, logiche dei modelli games–as-a-service – limitano o annullano la capacità dell'Agile di incidere strutturalmente sul crunch?*

Il primo quesito risponde alla prima lacuna identificata – l'assenza di un'analisi integrata tra pratiche Agile e dinamiche di crunch – e sarà affrontata nel Capitolo 2 attraverso la mappatura delle fasi di sviluppo e nel Capitolo 3 attraverso l'analisi dei casi studio. Con il secondo si mira a sviluppare teoricamente la questione del mascheramento, attingendo alla letteratura sul controllo normativo e sul lavoro affettivo per costruire la cornice analitica assente nella ricerca esistente. Infine, l'ultimo quesito mira a collocare la questione metodologica all'interno del più ampio sistema economico dell'industria, evitando la trappola di analizzare l'Agile come se operasse in un vuoto organizzativo.

Questi obiettivi e quesiti permettono di affrontare direttamente la lacuna individuata, offrendo un'analisi più integrata e contestualizzata del rapporto tra metodologie di sviluppo e sostenibilità del lavoro nell'industria videoludica.

Sia chiaro che il presente lavoro non intende formulare un giudizio definitivo sull'Agile come metodologia, né produrre una guida operativa alla sua implementazione. Non pretende di coprire l'intera varietà dell'industria videoludica globale – che comprende realtà radicalmente diverse, dagli studi indie a team singolo ai publisher AAA con migliaia di dipendenti. I risultati saranno necessariamente contestuali e le conclusioni formulate con il grado di cautela che la complessità del fenomeno richiede.

L'obiettivo, più modesto ma più onesto, è offrire strumenti analitici più precisi per leggere una dinamica che l'industria e la ricerca tendono ancora a semplificare eccessivamente, in direzioni opposte.

1.6 Struttura

La tesi è organizzata in sei capitoli, ciascuno dei quali assolve una funzione specifica nel percorso argomentativo complessivo.

Il capitolo 1 inquadra il problema del crunch nell'industria videoludica, ne documenta la rilevanza su più livelli – individuale, organizzativo, culturale ed economico – e identifica la lacuna nella letteratura esistente, che questa ricerca si propone di colmare, ovvero l'assenza di un'analisi integrata tra pratiche di project management e dinamiche di pressione lavorativa nel contesto dello sviluppo di videogiochi.

Il capitolo 2 compone il quadro teorico di riferimento attraverso una revisione della letteratura articolata in due sezioni complementari. La prima ricostruisce le origini, i valori fondativi e i principali framework Agile – Scrum, Kanban e i modelli ibridi – con particolare attenzione alle tensioni strutturali che emergono quando metodologie nate per il software enterprise vengono applicate al contesto videoludico. La seconda mappa le fasi produttive dello sviluppo di un videogioco – dalla pre-produzione al post-lancio – identificando dove e perché il crunch tende a manifestarsi e in quali punti l'Agile promette (o fallisce) di intervenire.

Il capitolo 3 descrive le scelte metodologiche della ricerca. Viene giustificata l'adozione di un approccio qualitativo ed esplorativo, illustrata la composizione del campione – dieci professionisti afferenti a sette studi videoludici italiani – e presentato lo strumento di raccolta dei dati, ovvero l'intervista semi-strutturata articolata in cinque domande principali. Il capitolo si chiude con una discussione sulla procedura di analisi tematica adottata e sui criteri di validità, affidabilità ed etica.

Il capitolo 4 presenta i risultati dell'analisi tematica, organizzati in quattro sezioni: le pratiche metodologiche effettivamente adottate dagli studi del campione, il rapporto percepito tra Agile e crunch, i meccanismi di generazione e distribuzione della pressione lavorativa e le pressioni strutturali esterne. Particolare attenzione è dedicata a due pattern emersi induttivamente dal *corpus*: il crunch volontario e il ruolo del producer come mediatore organizzativo della pressione.

Il capitolo 5 interpreta i risultati alla luce della domanda di ricerca principale e dei tre quesiti secondari, discutendone le implicazioni teoriche – tra cui la conferma empirica del meccanismo di controllo normativo e la proposta di una revisione della dicotomia Agile autentico *versus*

Agile di facciata – e le implicazioni concrete per practitioner, ricercatori e per chi si occupa di regolazione del lavoro nel settore creativo digitale.

Infine, il capitolo 6 sintetizza i risultati principali, pre le delimitazioni intenzionali del disegno di ricerca, riconoscendo i limiti strutturali dello studio e indicando sei direzioni per la ricerca futura, tra cui studi longitudinali sull'efficacia dell'Agile, analisi comparative tra contesti normativi diversi e indagini dedicate al crunch nei modelli games-as-a-service.

Capitolo 2

Revisione della letteratura

Per affrontare la domanda centrale di questa ricerca – se l'Agile riduca il crunch o ne mascheri le dinamiche – è necessario costruire un quadro di riferimento solido su due fronti distinti ma strettamente interconnessi: da un lato, la natura e il funzionamento delle metodologie Agile; dall'altro, la struttura produttiva dell'industria videoludica e i meccanismi attraverso cui il crunch si genera e si riproduce al suo interno.

Questo capitolo assolve dunque una duplice funzione. Da un lato, ricognitiva: si tratta di mappare ciò che la letteratura esistente ha già stabilito su entrambi i fronti, identificando le teorie, i modelli e i risultati empirici che costituiscono il fondamento analitico del presente studio. Dall'altro, critica: non ci si limita a descrivere lo stato dell'arte, ma si evidenziano le tensioni, le incongruenze e le aree in cui la ricerca esistente si è arrestata prima di rispondere alle domande più rilevanti.

La prima sezione ripercorre le origini dell'Agile – a partire dal *Manifesto for Agile Software Development* (Beck et al., 2001) –, ne esamina i principi fondativi e analizza i principali framework – Scrum, Kanban e le loro varianti ibride –, con particolare attenzione agli adattamenti sviluppati specificamente per il contesto videoludico e alle tensioni che emergono quando metodologie nate per il software enterprise vengono applicate a un processo creativo strutturalmente diverso.

La seconda sezione ricostruisce le fasi dello sviluppo di un videogioco, dalla pre-produzione al post-lancio, con l'obiettivo di individuare dove e perché il crunch tende a concentrarsi, e in quali punti l'Agile promette (o fallisce) di intervenire.

Ciò che emerge da questa ricognizione non è un quadro unitario e coerente, ma un campo attra-

versato da quesiti irrisolti: tra i principi dichiarati dell'Agile e le condizioni reali in cui viene implementato; tra la promessa di sostenibilità iterativa e le pressioni economiche che strutturano l'industria videoludica dall'esterno; tra la visibilità organizzativa che i framework Agile producono e l'invisibilità che talvolta generano attorno alle forme più sottili di sfruttamento lavorativo. Comprendere queste quesiti è il presupposto indispensabile per valutare, nei capitoli successivi, cosa accade davvero quando uno studio adotta l'Agile e dichiara di aver risolto il problema del crunch.

2.1 Le metodologie Agile: origini, principi e framework principali

2.1.1 Origini e contesto di nascita

Per comprendere cosa l'Agile sia davvero – e cosa non sia – è necessario partire dal contesto storico e metodologico in cui è nato, poiché le condizioni che ne hanno determinato l'emergere ne delimitano anche l'ambito di applicabilità e ne chiariscono le finalità originarie. Agli inizi degli anni Novanta, lo sviluppo software era dominato dal modello Waterfall, formalizzato da Winston W. Royce (1970) in un celebre articolo che, paradossalmente, conteneva già una critica implicita alla rigidità del modello stesso. Nonostante ciò, nel decennio successivo il Waterfall si affermò come standard de facto nell'industria tecnologica e nei contesti ingegneristici più strutturati.

Il modello Waterfall concepiva lo sviluppo software come una sequenza lineare e rigidamente ordinata di fasi – analisi dei requisiti, progettazione, implementazione, testing e rilascio –, ciascuna delle quali doveva essere completata e formalmente approvata prima dell'avvio della successiva. Tale impostazione derivava direttamente dai paradigmi dell'ingegneria manifatturiera e civile, dove la pianificazione ex ante e il controllo rigoroso dei processi consentono una previsione relativamente affidabile dei risultati. In questo contesto, il software veniva trattato come un prodotto industriale, assimilabile a un manufatto fisico, piuttosto che come un sistema complesso, dinamico e fortemente dipendente da variabili cognitive e organizzative. Questa impostazione si rivelò progressivamente inadeguata man mano che la complessità dei sistemi software aumentava e i contesti di sviluppo diventavano più incerti e mutevoli. I requisiti dei clienti, lungi dall'essere stabili e completamente definibili *a priori*, evolvevano durante il ciclo

di vita del progetto; le stime temporali e di costo si dimostravano sistematicamente ottimistiche; e soprattutto, come espone Boehm (1981), i problemi più critici emergevano spesso solo nelle fasi finali del processo, quando il costo del cambiamento era massimo . Di conseguenza, molti progetti fallivano o venivano consegnati con ritardi significativi, budget sforati e funzionalità non allineate alle reali esigenze degli utenti.

La cosiddetta "crisi del software", già evidenziata negli anni Settanta e Ottanta, trovò ulteriore conferma negli studi empirici e nelle riflessioni teoriche di autori come Frederick P. Brooks, che in *The Mythical Man-Month* (Brooks, 1975) sottolineava l'inefficacia dell'aggiunta di risorse umane a progetti in ritardo, e come Tom DeMarco e Timothy Lister, che in *Peopleware* (DeMarco & Lister, 1987) mettevano in luce il ruolo cruciale dei fattori umani, organizzativi e ambientali nella produttività dei team di sviluppo. Questi contributi contribuirono a spostare l'attenzione dalla dimensione puramente tecnica a quella socio-cognitiva del lavoro software, evidenziando come lo sviluppo non fosse un processo meccanico, bensì un'attività creativa, collaborativa e intrinsecamente incerta.

In parallelo, durante gli anni Novanta iniziarono a emergere approcci alternativi, spesso definiti "lightweight methodologies", come Scrum (Schwaber & Sutherland, 1995), Extreme Programming (Beck, 1999) e Crystal (Cockburn, 2001), che proponevano cicli di sviluppo iterativi e incrementali, feedback continuo e una maggiore centralità del team. Queste pratiche, inizialmente marginali, rappresentavano tentativi concreti di rispondere ai limiti strutturali del modello sequenziale tradizionale.

Fu in questo clima di insoddisfazione diffusa e di sperimentazione metodologica che, nel febbraio del 2001, diciassette sviluppatori e teorici del software si riunirono a Snowbird, nello Utah, dando vita al documento destinato a ridefinire profondamente il campo: il *Manifesto per lo Sviluppo Agile del Software* (Beck et al., 2001). Il testo, volutamente conciso – quattro valori fondamentali e dodici principi – rappresenta una sintesi ad alto livello di una nuova visione dello sviluppo software, basata sull'adattabilità, sulla collaborazione e sull'apprendimento continuo. I quattro valori del *Manifesto* – individui e interazioni più che processi e strumenti, software funzionante più che documentazione esaustiva, collaborazione con il cliente più che negoziazione contrattuale, risposta al cambiamento più che adesione a un piano – non rifiutano completamente gli elementi tradizionali, ma ne ridimensionano il ruolo, ponendo l'accento su ciò che genera effettivamente valore in contesti complessi e incerti. Questa impostazione riflette un cambio di paradigma: da un modello predittivo e deterministico a uno empirico e

adattivo.

L'influenza del *Manifesto Agile* sull'industria tecnologica dei due decenni successivi è stata profonda e pervasiva. Non solo ha dato origine a una vasta gamma di framework e pratiche operative, ma ha anche contribuito a ridefinire il modo in cui le organizzazioni concepiscono il lavoro, la leadership e l'innovazione. Tuttavia, è importante sottolineare che Agile non costituisce una soluzione universale: la sua efficacia dipende dal contesto, dal tipo di progetto e dal livello di incertezza. Comprendere le condizioni storiche e teoriche della sua nascita è quindi essenziale per evitarne applicazioni superficiali o dogmatiche

2.1.2 I valori e i principi fondativi

Figura 2.1: Manifesto Agile.

I quattro valori del *Manifesto Agile* stabiliscono una gerarchia esplicita tra approcci alternativi allo sviluppo:

- Individui e interazioni più che processi e strumenti;
- Software funzionante più che documentazione esaustiva;
- Collaborazione con il cliente più che negoziazione dei contratti;

- Risposta al cambiamento più che adesione a un piano .

Questa formulazione è deliberatamente relazionale e non oppositiva: non nega il valore dei secondi elementi di ciascuna coppia, ma afferma che, in condizioni di tensione o conflitto, la priorità debba essere assegnata ai primi. In altri termini, il *Manifesto* non propone un rifiuto delle pratiche tradizionali, bensì una loro subordinazione rispetto a ciò che genera valore in contesti incerti e dinamici.

Questa impostazione può essere letta come una presa di posizione epistemologica prima ancora che metodologica. Essa riconosce implicitamente che lo sviluppo software non è un processo completamente prevedibile e deterministico, ma un'attività caratterizzata da complessità emergente, apprendimento progressivo e adattamento continuo. In tal senso, Highsmith (2002) pensa che Agile si collochi in continuità con approcci empirici alla gestione dei sistemi complessi, nei quali la conoscenza rilevante non può essere interamente anticipata nella fase di pianificazione, ma si costruisce iterativamente attraverso il feedback e l'esperienza.

I dodici principi che accompagnano i valori articolano questa visione in termini più operativi, fornendo una guida generale senza tuttavia tradursi in prescrizioni rigide. Tra questi, tre risultano particolarmente rilevanti in relazione al tema del crunch e delle dinamiche di lavoro sostenibile. Il principio della sostenibilità afferma che «i processi Agile promuovono uno sviluppo sostenibile. Gli sponsor, gli sviluppatori e gli utenti dovrebbero essere in grado di mantenere un ritmo costante a tempo indeterminato» (Beck et al., 2001). Questo principio introduce una concezione del lavoro diametralmente opposta alla logica del crunch, che si fonda invece su picchi intensivi di lavoro straordinario concentrati in fasi critiche del progetto. L'idea di un ritmo sostenibile implica non solo una distribuzione più equilibrata del carico di lavoro, ma anche una diversa concezione della produttività, non più legata all'intensità momentanea dello sforzo, bensì alla sua continuità nel tempo.

Il principio della riflessione continua stabilisce che «a intervalli regolari il team riflette su come diventare più efficace, dopodiché regola e adatta il proprio comportamento di conseguenza» (Beck et al., 2001). Questo introduce un meccanismo istituzionalizzato di apprendimento organizzativo, spesso implementato attraverso pratiche come le retrospective nei framework Agile. In teoria, tale meccanismo dovrebbe consentire ai team di identificare precocemente segnali di sovraccarico, inefficienza o deterioramento delle condizioni di lavoro, intervenendo prima che tali dinamiche si consolidino. In questo senso, la riflessione continua rappresenta uno strumento potenziale di prevenzione del crunch, anche se la sua efficacia dipende forte-

mente dal contesto organizzativo e dal grado di autonomia effettivamente concesso ai team.
Il principio dell'eccellenza tecnica, secondo cui «la continua attenzione all'eccellenza tecnica e alla buona progettazione migliora l'agilità» (Beck et al., 2001), evidenzia un ulteriore aspetto cruciale: la velocità iterativa non deve essere ottenuta a scapito della qualità del codice e dell'architettura del sistema. Questo principio rimarca il problema del debito tecnico scoperto da Cunningham (1992), ovvero l'accumulo di soluzioni subottimali che, sebbene consentano un progresso rapido nel breve termine, generano costi crescenti nel lungo periodo. Il debito tecnico è spesso uno dei principali fattori che contribuiscono al verificarsi del crunch nelle fasi finali di sviluppo, quando la necessità di correggere errori accumulati richiede uno sforzo intensivo e prolungato.
Nonostante la coerenza teorica di questi principi, è fondamentale sottolineare la distanza che li separa dalla loro implementazione pratica. Il *Manifesto Agile* è, per sua natura, un documento di valori e non una metodologia operativa. Esso non fornisce indicazioni dettagliate su come organizzare concretamente il lavoro: non prescrive la struttura dei team, le tecniche di stima, le modalità di gestione dei requisiti o le strategie di interazione con i clienti. Questa indeterminatezza è al tempo stesso una forza e una debolezza: da un lato consente un'ampia adattabilità a contesti diversi; dall'altro apre lo spazio a interpretazioni eterogenee e, talvolta, distorsive.
In particolare, molti dei cosiddetti approcci Agile adottati nell'industria – inclusa quella videoludica – si concretizzano attraverso framework specifici come Scrum o Kanban analizzati da Schwaber & Sutherland (2020) e Anderson (2010), che traducono i principi del *Manifesto* in pratiche operative definite. Tuttavia, tali framework non coincidono con Agile in senso stretto: rappresentano piuttosto interpretazioni situate, che possono enfatizzare alcuni aspetti a scapito di altri. Ne deriva che una parte significativa di ciò che viene etichettato come "Agile" condivide con il *Manifesto* il lessico e la terminologia, ma non necessariamente l'impianto valoriale sottostante. Questa discrepanza tra principi fondativi e pratiche implementative costituisce uno dei nodi teorici centrali per comprendere le contraddizioni dell'adozione dell'Agile, in particolare nei contesti ad alta pressione produttiva, dove il rischio di una sua strumentalizzazione – ad esempio per giustificare ritmi di lavoro intensificati – è particolarmente elevato.

2.1.3 I principali framework Agile

Tra i numerosi framework che si richiamano ai principi Agile, tre hanno trovato un'applicazione particolarmente significativa nell'industria videoludica e meritano un'analisi più appro-

fondita: Scrum, Kanban e le metodologie ibride sviluppate specificamente per il contesto del game development. La loro diffusione non è casuale, ma riflette il tentativo dell'industria di conciliare esigenze spesso in tensione tra loro: da un lato la prevedibilità e il controllo dei processi produttivi, dall'altro la natura creativa, iterativa e altamente incerta dello sviluppo di videogiochi.

Scrum

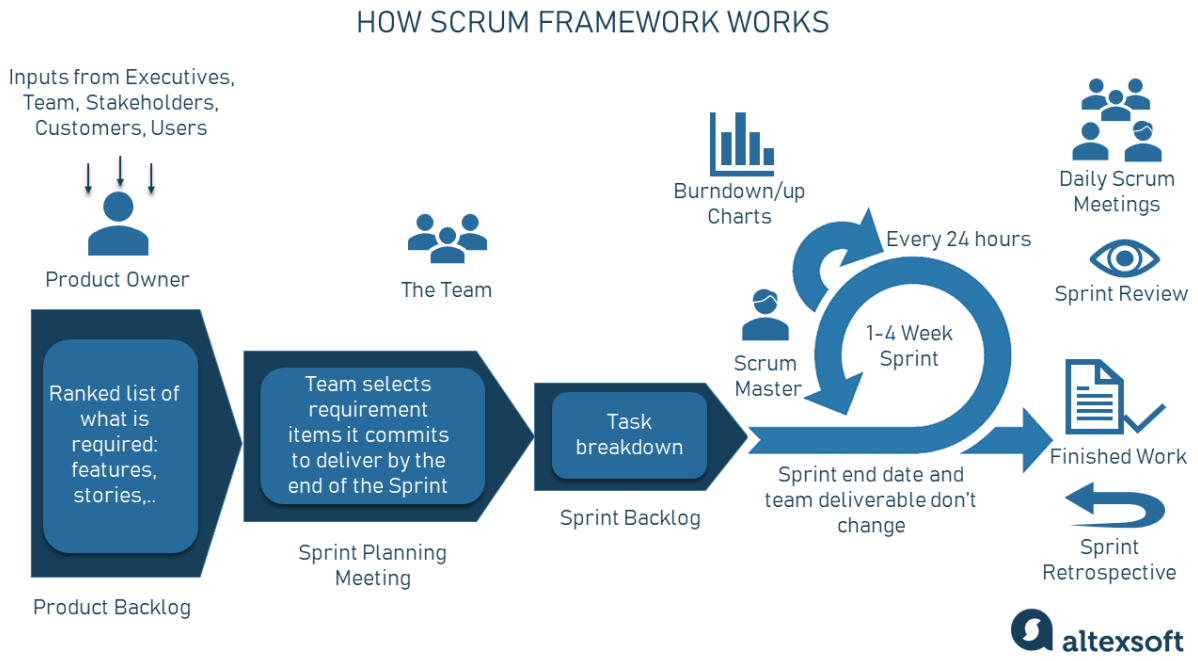

Figura 2.2: Esempio framework processo Scrum.

Scrum rappresenta il framework Agile più diffuso, sia nell'industria software in generale sia nello sviluppo videoludico in particolare. Formalizzato da Ken Schwaber e Jeff Sutherland nei primi anni Novanta e progressivamente codificato nella Scrum Guide – aggiornata più volte, con l'ultima versione nel 2020 (Schwaber & Sutherland, 2020) – Scrum propone una struttura relativamente semplice ma fortemente prescrittiva, basata su tre pilastri: ruoli, eventi e artefatti. I ruoli definiscono una chiara articolazione delle responsabilità all'interno del team.

Il Product Owner è responsabile della gestione e prioritarizzazione del Product Backlog e agisce come interfaccia tra il team di sviluppo e gli stakeholder esterni, che nel contesto videoludico possono includere publisher, investitori o divisioni marketing.

Lo Scrum Master svolge una funzione di facilitatore e "guardiano" del processo, assicurando che i principi Scrum vengano rispettati, rimuovendo impedimenti operativi e proteggendo il

team da interferenze esterne che potrebbero comprometterne il focus. Il Development Team, idealmente composto da un numero ristretto di membri (tradizionalmente tra cinque e nove), è auto-organizzato e cross-funzionale, e ha la responsabilità di trasformare gli elementi del backlog in incrementi di prodotto funzionanti.

Gli eventi strutturano il tempo e introducono una cadenza regolare nel lavoro. Lo Sprint costituisce l'unità temporale fondamentale – tipicamente da una a quattro settimane – al termine della quale deve essere prodotto un incremento potenzialmente rilasciabile. Attorno allo Sprint si articolano una serie di eventi ritualizzati: la Sprint Planning, in cui il team seleziona e pianifica il lavoro; il Daily Scrum, breve incontro quotidiano finalizzato alla sincronizzazione; la Sprint Review, momento di ispezione dell'incremento con gli stakeholder; e la Sprint Retrospective, dedicata alla riflessione e al miglioramento continuo del processo.

Gli artefatti – Product Backlog, Sprint Backlog e Increment – costituiscono gli strumenti attraverso cui si realizza la trasparenza del lavoro e si abilita il controllo empirico del processo.

Nel contesto videoludico, Scrum presenta una serie di vantaggi rilevanti. La cadenza fissa degli sprint introduce una disciplina di consegna che tende a ridurre la procrastinazione delle decisioni critiche, uno dei principali fattori che contribuiscono al crunch nelle fasi finali di sviluppo. Inoltre, la visibilità continua del backlog e degli incrementi consente di rendere esplicito il debito tecnico e le criticità emergenti, riducendo il rischio di scoprirle tardivamente.

Tuttavia, emergono anche limiti significativi. La rigidità temporale degli sprint può entrare in conflitto con la natura esplorativa e iterativa del game design: mentre una feature software può essere definita e completata in modo relativamente prevedibile, elementi come il game feel, il bilanciamento o l'esperienza utente richiedono cicli di iterazione difficilmente comprimibili in finestre temporali predefinite (Keith, 2010). Questo mismatch può generare pressioni artificiali che, anziché ridurre, rischiano di spostare o mascherare il crunch.

Kanban

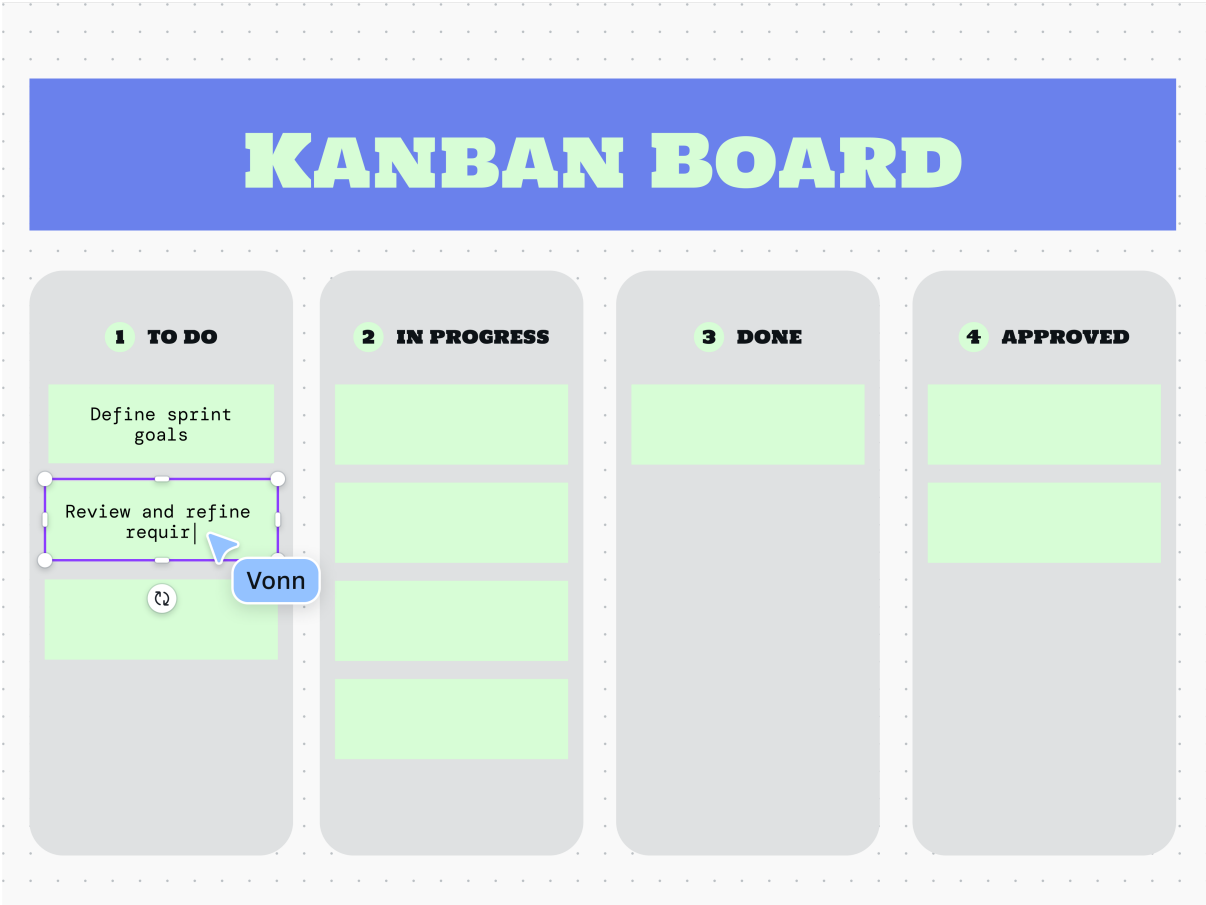

Figura 2.3: Kanban Board.

Kanban ha origini concettuali e storiche profondamente diverse. Sviluppato da Taiichi Ohno nel contesto del sistema produttivo Toyota negli anni Cinquanta e successivamente adattato allo sviluppo software da David J. Anderson (2010), Kanban si fonda su principi di gestione del flusso piuttosto che su iterazioni time-boxed. A differenza di Scrum, non prescrive ruoli specifici né eventi obbligatori: il suo fulcro è la visualizzazione del lavoro e la limitazione del work in progress (WIP).

Un sistema Kanban utilizza tipicamente una board – fisica o digitale – su cui le unità di lavoro sono rappresentate da card che attraversano colonne corrispondenti agli stati del processo. La configurazione delle colonne è adattabile al contesto, ma segue generalmente una struttura del tipo To Do, In Progress, Done, con possibili stati intermedi. Il limite al WIP, ovvero il numero massimo di attività simultaneamente presenti in ciascuna fase, costituisce il meccanismo centrale per evitare il sovraccarico, identificare i colli di bottiglia e mantenere un flusso di lavoro regolare e prevedibile.

Nel contesto videoludico, Kanban si dimostra particolarmente efficace nelle fasi di produzione continua e manutenzione, come gli aggiornamenti post-lancio, il bug fixing o la gestione di pipeline di asset artistici già stabilizzate. In queste situazioni, il lavoro tende a essere più modulare e le dipendenze più gestibili, rendendo il modello di flusso particolarmente adatto. Al contrario, nelle fasi di pre-produzione e prototipazione – caratterizzate da elevata incertezza e da frequenti cambi di direzione – la mancanza di strutture temporali e obiettivi intermedi può rendere più difficile mantenere allineamento e direzione. Dal punto di vista delle condizioni di lavoro, Kanban presenta un profilo ambivalente. L'assenza di scadenze rigide riduce la pressione psicologica associata alla fine dello sprint, rendendolo teoricamente meno incline a generare forme di crunch auto-imposto. Tuttavia, questa stessa flessibilità può tradursi in una minore percezione dell'urgenza e favorire fenomeni di scope creep o accumulo di lavoro incompiuto, che rischiano di essere "pagati" in fasi successive sotto forma di intensificazione del lavoro.

Framework ibridi e adattamenti videoludici

Nella pratica, l'adozione di framework Agile nell'industria videoludica raramente avviene in forma pura. La configurazione più diffusa è quella dei framework ibridi, ovvero combinazioni di pratiche tratte da Scrum, Kanban e altri approcci, adattate alle esigenze specifiche dello studio, del progetto e della fase di sviluppo. Questa ibridazione riflette la necessità di gestire contemporaneamente attività di natura molto diversa – programmazione, design, produzione artistica, testing – che difficilmente possono essere ricondotte a un unico modello operativo. Un contributo rilevante in questa direzione è quello di Clinton Keith, che in *Agile Game Development with Scrum* (Keith, 2010) propone una reinterpretazione di Scrum orientata al contesto videoludico. Tra gli adattamenti suggeriti vi sono sprint più lunghi nelle fasi iniziali di pre-produzione, per accomodare i tempi dell'esplorazione creativa, l'integrazione di pratiche di prototipazione rapida, spesso non formalizzate nei framework Agile tradizionali, e una gestione esplicita della tensione tra feature completeness e qualità dell'esperienza di gioco. Sebbene queste proposte risultino pragmaticamente utili, è importante notare come esse rimangano prevalentemente orientate all'efficienza del processo produttivo, senza affrontare in modo sistematico le implicazioni per le condizioni di lavoro e il benessere dei team.

Un'ulteriore configurazione ibrida ampiamente diffusa prevede l'utilizzo di Scrum per la gestione del backlog e della pianificazione iterativa, combinato con Kanban per la gestione dei flussi di lavoro più continui e meno prevedibili, come la produzione di asset artistici (grafica,

animazione, audio). Questa distinzione riflette una differenza strutturale tra attività: mentre la programmazione può essere più facilmente discretizzata in task relativamente autonomi, la produzione artistica segue spesso logiche iterative e cumulative che si adattano meglio a un modello di flusso continuo.

Ciò che emerge da questa analisi è che l'Agile, nel contesto videoludico, non si presenta come un insieme coerente e uniforme di pratiche, ma come un ecosistema di approcci adattivi, il cui esito dipende in larga misura dalle modalità concrete di implementazione. Questa variabilità costituisce un elemento cruciale per l'analisi del crunch: le stesse pratiche che, in teoria, dovrebbero promuovere sostenibilità e adattabilità possono, in determinati contesti organizzativi, essere reinterpretate in modo tale da legittimare o persino intensificare le pressioni produttive

2.1.4 Tensioni strutturali: quando l'Agile incontra il videogioco

Prima di procedere all'analisi delle fasi di sviluppo, è utile anticipare tre tensioni strutturali che attraverseranno l'intera tesi e che emergono già da questa ricognizione dei framework. Queste tensioni non sono difetti contingenti di implementazioni specifiche: sono strutturali ai framework Agile quando applicati a contesti in cui le condizioni originarie del *Manifesto* – team piccoli, requisiti negoziabili, assenza di pressioni contrattuali rigide – non sono soddisfatte. Comprenderle è il presupposto per valutare, nella seconda parte del capitolo, dove e come queste tensioni si manifestano nelle diverse fasi dello sviluppo di un videogioco.

1. **La prima tensione è quella tra pianificabilità e creatività.**

   I framework Agile presuppongono che il lavoro sia scomponibile in unità discrete, stimabili e completabili entro un'iterazione. Questa presupposizione, che trova la sua espressione più sistematica nel concetto di *definition of done* e nella pratica dello sprint planning, regge ragionevolmente per il software funzionale, dove un requisito può essere tradotto in un task con input e output definiti. La situazione si complica radicalmente di fronte alle componenti creative del game design – il bilanciamento, la narrativa, il feel – che sono per natura emergenti e resistono alla granularizzazione in task.

   Il problema non è nuovo nella letteratura sul design. Schön (1983), nel suo studio sul *reflective practitioner*, descrive il design come un processo di «conversation with the situation»: il progettista non applica soluzioni a problemi già definiti, ma scopre il problema mentre tenta di risolverlo. Questa logica si scontra frontalmente con la struttura dello sprint, che richiede che il problema sia già sufficientemente definito da poter essere

pianificato, stimato e completato in due settimane. Schell (2008), nel suo manuale *The Art of Game Design*, nota che il processo di game design è fondamentalmente iterativo in un senso che trascende la ciclicità agile: non si tratta di iterare su soluzioni a problemi noti, ma di ridefinire continuamente il problema stesso man mano che il gioco prende forma. In questo senso, l'iterazione del design di gioco è ontologicamente diversa dall'iterazione del software funzionale.

Questa tensione emerge con particolare chiarezza nelle discussioni sul playtesting e sul bilanciamento. Keith (2010), nel suo studio sullo sviluppo Agile specificamente applicato ai videogiochi, osserva che il bilanciamento non è un task completabile – è uno stato del sistema che deve essere continuamente monitorato e aggiustato. Inserire ≪bilancia il livello 3≫ in un backlog come se fosse un'attività con criterio di accettazione definito è, nella migliore delle ipotesi, una semplificazione operativa; nella peggiore, una fonte sistematica di technical e creative debt.

2. **La seconda tensione è quella tra autonomia del team e pressioni esterne.**

   Manifesto Agile presuppone una relazione collaborativa con il cliente, basata sulla fiducia e sulla disponibilità a rinegoziare lo scope in risposta al cambiamento. Il quarto valore del *Manifesto* – ≪rispondere al cambiamento più che seguire un piano≫ – e il principio relativo alla collaborazione continua con il cliente presuppongono che tale collaborazione sia possibile, ovvero che il cliente abbia interesse e capacità di partecipare attivamente al processo produttivo, e che le strutture contrattuali non cristallizzino aspettative non negoziabili in milestone irrevocabili.

   Nel contesto videoludico, questa presupposizione è sistematicamente violata. Il cliente è spesso un publisher con contratti a milestone rigide, aspettative di mercato non negoziabili e interessi economici che possono confliggere con la sostenibilità del processo produttivo. Tschang (2007), nella sua analisi economica dell'industria videoludica, documenta come i contratti tra sviluppatori indipendenti e publisher tendano a strutturarsi attorno a deliverable predefiniti – spesso descritti con anni di anticipo, quando l'incertezza sul prodotto finale è massima – con conseguenze finanziarie severe in caso di ritardo o deviazione. In questo contesto, l'autonomia promessa dall'Agile si riduce spesso a un'autonomia interna al team che non scalfisce le pressioni strutturali esterne.

   Questa dinamica è stata teorizzata da Dyer-Witheford & De Peuter (2009) in termini più ampi, come una delle contraddizioni fondamentali dell'industria creativa digitale:

il discorso sul creative freedom e sull'autonomia del developer convive con (e in parte maschera) strutture di controllo economico molto stringenti. L'adozione dell'Agile, in questo quadro, può paradossalmente rafforzare questa contraddizione: fornisce agli sviluppatori un vocabolario e una cultura dell'autonomia – la self-organization, la cross-functional team – che rimane confinata all'interno del team e non intacca il rapporto di potere con chi detiene il controllo finanziario del progetto.

3. **La terza tensione è quella tra trasparenza e controllo.**
   Uno degli effetti più documentati dell'adozione Agile è l'aumento della visibilità sul lavoro del team – attraverso backlog pubblici, velocity tracking, burndown chart. Questa trasparenza è presentata, nella letteratura normativa sull'Agile, come uno strumento di empowerment del team: rendere visibile il lavoro significa proteggere il team dal sovraccarico, rendere negoziabile lo scope e creare una base empirica per la pianificazione.
   Brechner (2015), descrivendo l'adozione del sistema Kanban in Microsoft, sostiene che la visualizzazione del flusso di lavoro ha ridotto il work in progress e migliorato la capacità del team di identificare i colli di bottiglia. Tuttavia, la stessa trasparenza può operare in modo radicalmente diverso in contesti organizzativi differenti. Ågerfalk & Fitzgerald (2006) osservano che l'introduzione di metriche agili in organizzazioni con strutture gerarchiche consolidate tende a trasformare strumenti pensati per il team in strumenti di monitoraggio manageriale.
   Nel settore videoludico, questa dinamica è stata documentata in relazione al fenomeno del crunch – le settimane o i mesi di lavoro straordinario non retribuito o scarsamente retribuito che caratterizzano la fase finale di molti sviluppi. Schreier (2021), nel suo studio giornalistico sulle condizioni di lavoro nell'industria videoludica, mostra come la visibilità prodotta dagli strumenti agili possa essere mobilizzata non per proteggere i lavoratori dal sovraccarico ma per identificare chi produce meno e aumentare la pressione su di loro. In questo scenario, la transparency board non è uno strumento di auto-organizzazione del team ma una forma di sorveglianza manageriale raffinata che sfrutta il linguaggio dell'empowerment per legittimare pratiche di intensificazione del lavoro.
   Questa lettura si inserisce in un quadro teorico più ampio: quello offerto da Gillespie et al. (2010) nell'analisi del controllo nelle organizzazioni creative, dove la sostituzione dei meccanismi di controllo diretto con meccanismi di controllo culturale e normativo – «lavoriamo così perché siamo un team agile» – tende a rendere meno visibile e meno

contestabile la struttura di potere sottostante.

2.2 Lo sviluppo di un videogioco: fasi produttive e dinamiche di crunch.

Prima di analizzare le singole fasi, è necessario chiarire un aspetto metodologico. Lo sviluppo di un videogioco non è un processo omogeneo: le competenze richieste, il grado di incertezza, la natura del lavoro e le pressioni temporali cambiano radicalmente da una fase all'altra. Questa eterogeneità ha una conseguenza diretta sull'argomento affrontato in questa tesi: il crunch non è un fenomeno uniforme che si manifesta sempre allo stesso modo e per le stesse ragioni. È invece un fenomeno contestuale, le cui cause e manifestazioni variano significativamente in funzione della fase produttiva in cui si verifica.
Questa premessa ha anche un'implicazione critica per la valutazione dell'Agile: un framework che si dimostra efficace nel contenere il crunch in una fase può rivelarsi inefficace o controproducente in un'altra. Valutare l'Agile come soluzione unitaria al crunch – come tende a fare buona parte della letteratura di settore – significa ignorare questa eterogeneità e rischiare di trarre conclusioni che reggono in alcuni contesti e non in altri.
Il modello di sviluppo che si adotta in questa analisi è quello prevalente nell'industria AAA e nei team di medie dimensioni, articolato in cinque fasi principali: concept e pre-produzione, prototipazione, produzione, alpha e beta, gold e post-lancio. È un modello approssimativo – nella pratica i confini tra fasi sono spesso porosi e le fasi stesse si sovrappongono – ma offre una struttura analitica sufficientemente precisa per i fini di questa ricerca.

2.2.1 Concept e pre-produzione

La fase di concept e pre-produzione è quella in cui il progetto prende forma come idea: si definiscono il genere, la piattaforma target, le meccaniche fondamentali, il tono narrativo e visivo, e si producono i primi documenti di design – il Game Design Document (GDD) e le sue varianti più agili. In studi AAA questa fase può durare da sei mesi a due anni; in team indie o di medie dimensioni si comprime spesso in modo significativo, con conseguenze che si manifestano nelle fasi successive. Questa asimmetria temporale non è neutrale: la pre-produzione è la fase in cui le decisioni hanno il costo di modifica più basso e l'impatto più alto sull'intero sviluppo.

Rinunciarvi per pressioni di budget o di calendario significa spostare il costo dell'esplorazione nelle fasi più avanzate, dove ogni cambiamento di direzione è enormemente più oneroso in termini di lavoro già prodotto da scartare. McConnell (1996), nel suo studio classico sull'impatto economico delle modifiche al software, stima che il costo di correzione di un requisito in fase di produzione avanzata sia da dieci a cento volte superiore a quello dello stesso intervento in fase di analisi e progettazione – una stima che, applicata al game design, suggerisce che la compressione della pre-produzione non risparmia tempo ma lo sposta, con interessi, nelle fasi successive.

Il GDD, documento centrale di questa fase, ha una storia travagliata nei framework Agile. Nella pratica tradizionale dello sviluppo videoludico degli anni Novanta e Duemila, il GDD era un documento monolitico, spesso di centinaia di pagine, che tentava di specificare in anticipo ogni aspetto del gioco – meccaniche, livelli, dialoghi, interfacce. Questo approccio, chiaramente waterfall nella sua logica, è stato progressivamente abbandonato o ridimensionato con l'adozione dei framework iterativi. Bethke (2003) descrive il passaggio da GDD monolitici a «living documents» che evolvono con il progetto, riducendo la prescrittività iniziale a favore di una maggiore flessibilità. Il problema, come osserva Fullerton (2014), è che un documento troppo vago non adempie alla sua funzione principale: creare allineamento tra i membri del team su cosa si sta costruendo. Il GDD Agile ideale dovrebbe essere sufficientemente definito da orientare le decisioni quotidiane e sufficientemente flessibile da non cristallizzare scelte che l'esperienza di gioco reale potrebbe rimettere in discussione – un equilibrio difficile da trovare e ancora più difficile da mantenere nel tempo.

Il rischio di crunch in questa fase è generalmente contenuto, ma non assente. Le pressioni più comuni derivano dalla necessità di presentare prototipi o pitch ai publisher entro scadenze non negoziabili – le cosiddette *greenlight meetings* – che possono concentrare un carico di lavoro elevato in finestre temporali ristrette. Schreier (2017) documenta come in diversi titoli AAA la pre-produzione abbia generato forme di crunch localizzato proprio attorno a queste scadenze di approvazione, spesso invisibili nelle statistiche aggregate perché coinvolgono team ridotti. La dinamica è strutturalmente analoga a quella che Perlow (1999) aveva descritto nel software engineering: le scadenze ravvicinate producono cicli di lavoro intensivo che vengono normalizzati come episodi eccezionali, ma che si ripetono con sufficiente regolarità da costituire un

pattern strutturale piuttosto che un'eccezione. Nel contesto della pre-produzione videoludica, questo pattern è amplificato dall'asimmetria di potere tra sviluppatore e publisher: il team non può negoziare la data del greenlight meeting nello stesso modo in cui, in un contesto Agile ideale, potrebbe negoziare lo scope di uno sprint con un product owner interno.
Dal punto di vista delle metodologie Agile, la pre-produzione rappresenta la fase in cui i principi iterativi e adattivi mostrano la maggiore compatibilità con la natura del lavoro. L'esplorazione di alternative progettuali, la prototipazione rapida e la disponibilità ad abbandonare soluzioni non soddisfacenti sono pratiche pienamente coerenti con l'enfasi Agile sulla risposta al cambiamento e sull'apprendimento continuo, come formulato da Beck et al. (2001). In questo senso, la pre-produzione può essere interpretata come un contesto intrinsecamente "agile", anche in assenza di un'adozione formale di framework come Scrum o Kanban. Il concetto di spike in Scrum – un'iterazione dedicata all'esplorazione di un'incognita tecnica o di design piuttosto che alla produzione di valore direttamente consegnabile – è particolarmente adatto a questa fase, e Keith (2010) lo identifica come uno degli strumenti Agile più utili nel contesto videoludico proprio perché legittima il lavoro esplorativo all'interno di una struttura iterativa altrimenti orientata alla consegna incrementale.

Tuttavia, come osserva O'Donnell (2014a), questa compatibilità apparente può essere ingannevole: in assenza di una visione sufficientemente definita, gli sprint di pre-produzione rischiano di produrre iterazioni che consumano tempo e risorse senza convergere verso una direzione produttiva, generando quello che in letteratura è noto come pivoting loop – un ciclo di ridefinizioni che ritarda l'ingresso in produzione e comprime le fasi successive. O'Donnell, che ha studiato il processo produttivo di *Halo* attraverso un'etnografia durata anni all'interno di Bungie, descrive come la difficoltà di stabilire criteri di valutazione condivisi per il lavoro creativo renda particolarmente difficile decidere quando una fase esplorativa ha prodotto risultati sufficienti per procedere. In assenza di tali criteri, la pre-produzione tende a prolungarsi finché non interviene una pressione esterna – una scadenza contrattuale, una decisione del management – che forza la chiusura indipendentemente dalla maturità delle soluzioni raggiunte. Questo meccanismo, che potremmo definire deadline-driven convergence, è l'opposto della convergenza organica che i framework Agile dovrebbero favorire, e rappresenta uno dei modi in cui le pressioni strutturali esterne neutralizzano i benefici teorici dell'approccio iterativo già nella primissima fase dello sviluppo.

2.2.2 Prototipazione

La prototipazione è la fase in cui le idee generate nella pre-produzione vengono tradotte in forme concrete e verificabili: si costruiscono versioni rudimentali – spesso incomplete e tecnicamente approssimative – delle meccaniche fondamentali del gioco, con l'obiettivo di valutarne la *fun*, la fattibilità tecnica e la coerenza con la visione complessiva del progetto. In questa fase, il valore del lavoro non risiede nella qualità dell'output in sé, ma nella quantità e nella qualità dell'informazione che esso produce. Un prototipo che dimostra che una meccanica non funziona, o che non genera l'esperienza desiderata, rappresenta un esito positivo, in quanto consente di evitare investimenti successivi su direzioni non promettenti.
Si tratta quindi di una fase ad alta incertezza ma anche ad alto valore informativo, in cui il progresso non può essere misurato esclusivamente in termini di funzionalità completate, ma deve essere interpretato come riduzione dell'incertezza progettuale. Questa caratteristica la distingue nettamente dalle fasi di produzione, in cui l'obiettivo è la costruzione sistematica di contenuti e sistemi già definiti. Nella prototipazione, al contrario, il processo è intrinsecamente esplorativo e iterativo, e spesso comporta la creazione e l'abbandono di molteplici soluzioni alternative prima di convergere su una direzione stabile.

Il rischio di crunch in questa fase è intermedio e assume caratteristiche qualitativamente diverse rispetto a quelle osservabili nelle fasi successive. Non si tratta, nella maggior parte dei casi, di un crunch legato a scadenze esterne rigide – il prototipo non è ancora vincolato a una data di rilascio – ma piuttosto di quello che Peticca-Harris et al. (2015) definiscono *crunch da commitment*. Questo tipo di pressione deriva dalla necessità percepita, spesso interiorizzata dai membri del team, di dimostrare rapidamente che il progetto è valido e merita ulteriori investimenti. In contesti organizzativi caratterizzati da forte competizione interna, da risorse limitate o da una cultura del lavoro intensivo, questa pressione può tradursi in ritmi di lavoro elevati anche in assenza di vincoli temporali formalizzati. A differenza del crunch da scadenza, che è tipicamente imposto dall'esterno, il crunch da commitment ha una dimensione più ambigua e diffusa: è al tempo stesso auto-imposto e socialmente rinforzato. I membri del team possono essere motivati a lavorare oltre i limiti sostenibili non solo per rispettare aspettative esplicite, ma anche per dimostrare dedizione, talento o allineamento agli obiettivi dello studio. La letteratura sul lavoro creativo evidenzia come questo tipo di dinamica sia particolarmente diffuso nei settori ad alta componente vocazionale, dove la passione per il prodotto tende a sovrapporsi

alle logiche di lavoro. Nel contesto videoludico, ciò si traduce spesso in una normalizzazione di pratiche di lavoro intensivo già nelle fasi iniziali del progetto.

È in questa fase che la tensione tra Agile e creatività videoludica si manifesta per la prima volta in modo acuto. Lo sprint di Scrum funziona bene quando il risultato atteso è definibile *a priori*; nel caso della prototipazione, il risultato non è una feature completa ma una risposta a una domanda – questa meccanica è divertente? – che non sempre si riesce a formulare e a rispondere entro due settimane. Keith (2010) propone sprint più lunghi in questa fase, ma questa soluzione introduce una tensione interna al framework: sprint da quattro o sei settimane riducono la frequenza dei cicli di feedback e si avvicinano pericolosamente alla logica sequenziale del Waterfall che l'Agile era nato per sostituire. Il problema non è solo tecnico ma epistemologico: la lunghezza dello sprint dovrebbe essere determinata dalla natura del lavoro da svolgere, ma i framework Agile standardizzati tendono a imporre una lunghezza uniforme per garantire la prevedibilità del processo. Questa uniformità, funzionale in contesti di sviluppo software convenzionale, diventa una fonte di rigidità in contesti dove la natura del lavoro cambia radicalmente da fase a fase.

Alcuni studi hanno risposto a questa sfida ibridando Scrum con approcci più flessibili. La combinazione di Scrum e Kanban – a volte denominata Scrumban – permette di mantenere i rituali di sincronizzazione dello sprint (planning, review, retrospective) allentando il vincolo sulla durata delle iterazioni e consentendo un flusso di lavoro più continuo e meno rigidamente pianificato. Questa ibridazione è particolarmente adatta alla prototipazione perché consente di gestire la variabilità intrinseca del lavoro esplorativo senza abbandonare del tutto la struttura di accountability che i framework iterativi forniscono. Tuttavia, come avverte Anderson (2010) nella sua analisi del metodo Kanban applicato allo sviluppo software, la flessibilità procedurale non risolve le tensioni strutturali sottostanti: se le pressioni esterne – un publisher che vuole vedere risultati, un management che chiede previsioni – rimangono invariate, aumentare la flessibilità del processo interno può semplicemente rendere più difficile resistere a richieste irragionevoli, perché viene meno la struttura formale dello sprint come dispositivo di negoziazione dello scope.

2.2.3 Produzione

La fase di produzione è il cuore dello sviluppo: l'intero team – programmatori, artisti, designer, sound designer, writer – lavora in parallelo per costruire il gioco nelle sue componenti complete. È la fase più lunga, tipicamente da uno a tre anni nei titoli AAA, e quella in cui il rischio di crunch inizia ad accumularsi in modo strutturale. La produzione è anche la fase in cui le decisioni prese (o evitate) nella pre-produzione e nella prototipazione presentano il conto: una visione di design insufficientemente definita genera ridefinizioni continue, un'architettura tecnica affrettata accumula debito, un contratto con milestone rigide riduce progressivamente lo spazio di manovra del team. In questo senso, il crunch di produzione è spesso il sintomo di problemi sistemici che affondano le radici nelle fasi precedenti, non una patologia autonoma.
Vale la pena inquadrare la fase di produzione dal punto di vista organizzativo prima di analizzarne le dinamiche specifiche. Un team di sviluppo AAA in produzione è un'organizzazione temporanea di notevole complessità: centinaia di persone con competenze eterogenee – spesso distribuite su più sedi geografiche – che devono coordinare il loro lavoro verso un obiettivo comune la cui definizione precisa continua a evolvere nel corso del progetto. Esistono quattro caratteristiche che contraddistinguono le organizzazioni permanenti come gli studi di sviluppo: la temporaneità, l'orientamento al task, il cambiamento come condizione normale e l'alta incertezza rispetto all'ambiente esterno. Tutte e quattro sono presenti in modo accentuato nello sviluppo videoludico AAA, il che rende questo contesto particolarmente sfidante per qualsiasi framework organizzativo, incluso l'Agile, che è stato originariamente concepito per team molto più piccoli e meno differenziati al loro interno.

Le cause del crunch in produzione sono molteplici e si alimentano reciprocamente. La prima è lo *scope creep*: l'aggiunta progressiva di feature, contenuti e meccaniche non previste nella pianificazione originale, spesso su impulso del management o del publisher, che espande il perimetro del progetto senza un corrispondente aumento delle risorse o un'estensione della timeline. Lo *scope creep* non è semplicemente il risultato di cattiva pianificazione: è in parte una conseguenza strutturale della natura del game design. Poiché il gioco diventa comprensibile come esperienza solo quando ha raggiunto uno stadio di sviluppo che ne consente la fruizione, le decisioni su cosa aggiungere o modificare vengono spesso prese tardi nel processo, quando il costo di implementazione è più alto.
La seconda causa è il debito tecnico: la tendenza a rimandare la correzione di problemi architet-

turali per rispettare le milestone intermedie, accumulando un passivo tecnico che dovrà essere saldato nelle fasi finali a costo di crunch intensivo. Il concetto di debito tecnico, introdotto da Cunningham (1992) in un contesto di sviluppo software generale, descrive la scelta deliberata di adottare soluzioni subottimali nel breve termine per raggiungere un obiettivo immediato, con la consapevolezza che il costo differito sarà superiore a quello che si sarebbe pagato intervenendo tempestivamente. Nel game development, il debito tecnico tende ad accumularsi in modo particolarmente rapido perché le pressioni a mostrare risultati visibili incentivano sistematicamente soluzioni rapide a scapito della robustezza architetturale.

La terza causa è la dipendenza tra discipline: in un progetto videoludico, il lavoro di programmatori, artisti e designer è profondamente interdipendente – un ritardo in una disciplina si propaga alle altre, comprimendo le timeline a valle. Questa interdipendenza è strutturalmente diversa da quella presente nello sviluppo software convenzionale: un programmatore che lavora su un sistema di intelligenza artificiale dipende dagli asset degli artisti per testare il comportamento dei personaggi, dagli script dei designer per capire quali comportamenti implementare, e dall'architettura dei programmatori di altre discipline per integrare il suo lavoro nel sistema complessivo. Brooks (1975), nel suo classico studio sulla gestione dei progetti software, aveva già identificato la dipendenza come una delle cause principali dei ritardi nei progetti complessi, formulando la celebre osservazione che aggiungere personale a un progetto in ritardo lo fa ritardare ulteriormente – la cosiddetta "legge di Brooks" – perché il tempo necessario per integrare i nuovi arrivati e il costo di coordinamento aggiuntivo superano il beneficio della capacità produttiva aggiunta. Nel game development, questa dinamica è esacerbata dalla natura multidisciplinare del lavoro: il coordinamento tra discipline diverse è intrinsecamente più costoso del coordinamento all'interno della stessa disciplina.

L'Agile, in questa fase, mostra risultati ambivalenti. Da un lato, la trasparenza del backlog e la cadenza degli sprint possono rendere visibile lo scope creep prima che diventi ingestibile, permettendo al team o al Product Owner di prendere decisioni consapevoli su cosa includere e cosa tagliare. Questa funzione di controllo dello scope è uno dei contributi più genuinamente utili dei framework Agile alla gestione dei progetti videoludici: il backlog prioritizzato, se correttamente mantenuto, fornisce in ogni momento una rappresentazione esplicita di cosa è stato deciso di fare e in quale ordine, rendendo ogni aggiunta al perimetro del progetto una scelta visibile che richiede una giustificazione e, potenzialmente, una compensazione attraverso la

rimozione di altro lavoro. De Prato et al. (2010), nella loro analisi dei modelli di produzione nell'industria videoludica europea, identificano la gestione del backlog come una delle pratiche Agile con il maggiore impatto positivo documentato sulla prevedibilità delle timeline, a condizione che il Product Owner abbia effettivo potere decisionale sullo scope e non sia vincolato da aspettative contrattuali o manageriali che rendono il taglio di feature politicamente impossibile.

Dall'altro, come documentano Hodgson & Briand (2013), la pressione a consegnare un incremento completato ad ogni sprint può incentivare i team a sottostimare sistematicamente la complessità dei task per far rientrare il lavoro nei tempi previsti – un fenomeno noto come *velocity inflation* che produce dati di avanzamento ottimistici mascherando il reale stato del progetto fino a quando il divario tra pianificato e reale diventa impossibile da ignorare. La *velocity inflation* è un esempio di come gli strumenti di trasparenza Agile possano essere – spesso inconsapevolmente – usati per produrre l'apparenza di controllo piuttosto che il controllo effettivo. Quando un team sa che la sua velocity viene monitorata dal management, ha un incentivo strutturale a mantenere la velocity alta, il che può tradursi in task artificialmente frammentati, stime sistematicamente ottimistiche, e una definizione di "completato" progressivamente più permissiva.

2.2.4 Alpha e Beta

Le fasi di alpha e beta segnano il passaggio dalla costruzione del gioco alla sua integrazione e verifica. In alpha, tutte le feature principali sono implementate ma non necessariamente bilanciate o prive di bug; in beta, il gioco è funzionalmente completo e il lavoro si concentra sulla correzione dei difetti, sull'ottimizzazione delle performance e sul polish finale. Questa distinzione, che nella documentazione di progetto appare netta, è nella pratica spesso sfumata: molti studi entrano formalmente in alpha con feature ancora in sviluppo, o dichiarano la beta per rispettare le aspettative contrattuali del publisher pur sapendo che il gioco non è funzionalmente completo. Le definizioni di alpha e beta nel game development sono notevolmente meno standardizzate di quanto la terminologia suggerisca, e la loro funzione principale è spesso contrattuale – segnalare il raggiungimento di una milestone per sbloccare un pagamento – più che tecnica. Questa ambiguità definitoria ha conseguenze dirette sul crunch: quando alpha e beta sono dichiarate prematuramente per ragioni contrattuali, il lavoro che dovrebbe essere già completato viene compresso nelle settimane successive, aumentando la pressione in modo

repentino e difficilmente gestibile.

Un aspetto strutturale di queste fasi che merita attenzione è il cambiamento nella natura del lavoro rispetto alla produzione. Durante la produzione, il lavoro è prevalentemente additivo: si costruiscono sistemi, si producono asset, si implementano meccaniche. In alpha e beta, il lavoro diventa prevalentemente diagnostico e correttivo: si identificano problemi, si determina la loro causa, si interviene in sistemi già costruiti cercando di non introdurre nuove regressioni. Questo tipo di lavoro ha caratteristiche cognitive molto diverse dal lavoro additivo: richiede comprensione profonda di sistemi complessi e spesso scarsamente documentati, capacità di ragionamento controfattuale – cosa succede se cambio questo parametro? – e una tolleranza all'ambiguità che la pressione temporale tende a erodere. Humphrey (1989), nel suo studio fondativo sulla gestione dei processi software, distingue tra defect prevention e defect detection and correction, mostrando come il costo unitario della correzione aumenti esponenzialmente con la maturità del progetto. Applicato al game development, questo principio implica che ogni bug non identificato durante la produzione ha un costo di correzione in alpha e beta che può essere un multiplo significativo del costo originale – una dinamica che contribuisce a spiegare perché le fasi finali assorbano una quota di lavoro sproporzionata rispetto alla loro posizione nella timeline.

Queste sono le fasi in cui il rischio di crunch diventa sistematicamente elevato, e le ragioni sono in parte strutturali e in parte psicologiche. Sul piano strutturale, è in alpha e beta che il debito tecnico accumulato durante la produzione presenta il conto: bug profondi che richiedono interventi architetturali, feature non integrate che devono essere riconciliate, performance insufficienti che richiedono ottimizzazioni non pianificate. Politowski et al. (2021), in un'analisi quantitativa dei repository di sviluppo di titoli open source e di documentazione pubblica di studi AAA, mostrano come la densità di bug riportati raggiunga tipicamente il picco nelle settimane immediatamente successive alla dichiarazione di alpha, e che la coda lunga di bug ad alta priorità che persiste fino alla gold master sia correlata positivamente con la presenza di debito tecnico documentato nelle fasi precedenti. Questa correlazione non è sorprendente dal punto di vista tecnico, ma ha implicazioni organizzative rilevanti: suggerisce che il crunch di alpha e beta non è un evento imprevedibile ma una conseguenza misurabile e in parte anticipabile delle scelte fatte nelle fasi precedenti, il che solleva interrogativi sull'efficacia dei processi di retrospettiva Agile nel capitalizzare queste informazioni per migliorare i cicli di sviluppo successivi.

In questa fase, l'Agile tende a perdere efficacia come strumento di contenimento del crunch. Le cerimonie Scrum – retrospettive, planning, review – continuano a svolgersi formalmente, ma il loro contenuto viene progressivamente svuotato dalla pressione della scadenza. Come osserva Schreier (2017) in diversi dei suoi casi studio, i team in alpha e beta tendono a trasformare le retrospettive in sessioni di gestione dell'emergenza piuttosto che in momenti di riflessione sul processo, e gli sprint si allungano o si interrompono informalmente quando le priorità operative prendono il sopravvento sulla struttura metodologica. Questo fenomeno – la persistenza delle forme rituali di un framework in assenza della sua sostanza – è stato descritto come cargo cult Agile: organizzazioni che adottano la terminologia e i rituali Agile senza interiorizzarne i principi, ottenendo i costi procedurali del framework senza i suoi benefici. Nel contesto delle fasi finali del game development, questa deriva è in parte comprensibile: quando la scadenza è fissa e il lavoro da completare è critico, la capacità di astrazione necessaria per riflettere sul processo è una risorsa cognitiva scarsa. Ma la conseguenza è che il framework smette di funzionare proprio nella fase in cui la pressione è più alta e la necessità di strumenti di gestione del carico di lavoro è più acuta.

Un meccanismo specifico di questa deriva merita di essere analizzato più nel dettaglio: la gestione del bug backlog. In alpha e beta, il backlog Scrum convenzionale viene tipicamente affiancato o sostituito da un bug tracker che gestisce la coda di difetti da correggere secondo criteri di priorità tecnica piuttosto che di valore per l'utente. Questa transizione rappresenta una rottura con la logica Agile standard, in cui il backlog è ordinato per valore di business e il Product Owner ha l'ultima parola sulle priorità. In un bug tracker di alpha, le priorità sono determinate principalmente da criteri tecnici – severità del difetto, frequenza di riproduzione, impatto sulle performance – e la gestione operativa tende a spostarsi dai Product Owner ai lead tecnici. Questa transizione, se non gestita consapevolmente, può frammentare la coerenza del processo decisionale e rendere difficile mantenere una visione integrata dello stato del progetto, contribuendo alla sensazione di perdita di controllo che caratterizza molte descrizioni del crunch finale.

2.2.5 Gold, lancio e post-lancio

La fase gold – il momento in cui il master del gioco è pronto per la distribuzione – rappresenta storicamente il picco del crunch nell'industria videoludica. È il momento in cui tutte le pres-

sioni accumulate nelle fasi precedenti convergono: le milestone contrattuali con il publisher, le finestre di lancio legate ai cicli delle console o alle stagioni commerciali, le aspettative del pubblico alimentate da anni di marketing. Il team lavora a ritmi che possono superare le ottanta ore settimanali per settimane o mesi consecutivi, in condizioni che la ricerca ha documentato essere sistematicamente dannose per la salute fisica e mentale dei lavoratori.

Ciò che rende questa fase particolarmente rilevante ai fini della presente ricerca è che si tratta anche della fase in cui il fallimento dell'Agile – ove si verifichi – è più visibile e meno negabile. Un team che ha adottato Scrum per tre anni e si trova comunque a fare crunch nelle settimane del gold è la dimostrazione empirica più diretta che l'Agile non ha risolto il problema strutturale. Questo non significa necessariamente che l'Agile abbia fallito in senso assoluto – potrebbe aver ridotto il crunch rispetto a quanto si sarebbe verificato in sua assenza – ma solleva domande legittime sull'entità e sulla natura del miglioramento. La difficoltà di rispondere a queste domande in modo rigoroso è essa stessa significativa: poiché non esistono controfattuali osservabili – non possiamo sapere come sarebbe andato lo stesso progetto senza Agile –, la valutazione dell'impatto dei framework iterativi sul crunch rimane intrinsecamente dipendente da testimonianze qualitative e comparazioni trasversali tra studi con caratteristiche molto diverse.
Un aspetto specifico della fase gold che i framework Agile gestiscono con particolare difficoltà è la gestione delle decisioni di taglio (cut decisions): la scelta di rimuovere dal prodotto finale feature, livelli o contenuti che non raggiungeranno il livello di qualità necessario entro la data di lancio. In teoria, questa è esattamente la tipologia di decisione per cui il backlog prioritizzato e la logica del valore incrementale Agile forniscono strumenti adeguati: se una feature non è pronta, non entra nel prodotto consegnato. In pratica, le decisioni di taglio in fase gold sono tra le più conflittuali e dolorose dell'intero processo di sviluppo, perché coinvolgono anni di lavoro investito, promesse fatte al pubblico attraverso il marketing e valutazioni estetiche su cosa costituisce un prodotto completo che non sono riducibili a metriche di backlog. Blow (2011), in un intervento alla Game Developers Conference, descrive il processo decisionale del taglio come fondamentalmente incompatibile con la logica del Product Owner Agile, perché richiede una comprensione olistica del gioco come esperienza che non può essere decomposta in user story indipendenti – tagliare una feature non significa solo rimuovere un elemento, ma potenzialmente alterare l'equilibrio e la coerenza dell'esperienza complessiva.
Il post-lancio introduce una dimensione ulteriore che la letteratura ha cominciato ad esplorare

solo di recente. Nei modelli games-as-a-service (GaaS), il lancio non è la fine del progetto ma l'inizio di un ciclo di aggiornamenti continui – stagioni, patch, DLC, eventi live – che mantiene il team in uno stato di produzione permanente. In questo contesto, il crunch non scompare ma si distribuisce nel tempo: non più un picco acuto e riconoscibile, ma una pressione cronica e a bassa intensità che può essere altrettanto logorante ma molto più difficile da identificare, misurare e contrastare. Lizardi (2020) descrive il modello GaaS come una forma di ≪produzione perpetua≫ che trasforma il rapporto tra sviluppatori e prodotto: il gioco non è mai finito, ogni patch introduce nuove possibilità di bug e nuove aspettative degli utenti, e il team vive in uno stato di reattività permanente che erode i confini tra il tempo di lavoro e il tempo di recupero.

Come si anticipava nella rassegna della letteratura, questa forma di crunch distribuito è quella per cui i framework Agile mostrano le maggiori promesse teoriche – la cadenza degli sprint dovrebbe garantire un ritmo sostenibile – ma anche i rischi più sottili, poiché la normalizzazione del ritmo produttivo può rendere invisibile il sovraccarico fino a quando le conseguenze sulla salute dei lavoratori diventano impossibili da ignorare. Derby & Larsen (2006), nel loro manuale sulle retrospettive Agile, identificano la cadenza regolare delle retrospettive come lo strumento principale per identificare precocemente i segnali di sovraccarico e intervenire prima che diventino irreversibili. Tuttavia, affinché questo meccanismo funzioni in un contesto GaaS, è necessario che le retrospettive abbiano reale potere di cambiare il ritmo produttivo – il che presuppone che il team abbia sufficiente autonomia rispetto alle pressioni esterne dei publisher, degli aggiornamenti stagionali e delle aspettative della community. In assenza di questa autonomia, la retrospettiva rischia di diventare, anche in questo contesto, uno spazio in cui il disagio viene nominato ma non affrontato: un rituale che produce la percezione di partecipazione senza modificare le condizioni strutturali che generano il problema.

Capitolo 3

Metodologia della ricerca

3.1 Disegno della ricerca

La domanda centrale di questa tesi – se l'Agile riduca il crunch nell'industria videoludica o ne mascheri le dinamiche profonde – non è rispondibile attraverso una misurazione puramente quantitativa. Il crunch non è un fenomeno che si esaurisce nel conteggio delle ore lavorate: è un'esperienza situata, mediata dalla cultura organizzativa dello studio, dalla posizione gerarchica del lavoratore, dal tipo di progetto e dal modello di business adottato. Comprenderne le dinamiche richiede di accedere alle interpretazioni, alle pratiche quotidiane e ai giudizi di chi lo vive dall'interno.

Per queste ragioni, il disegno di ricerca adottato è di natura qualitativa, con un orientamento esplorativo e interpretativo. Questa scelta si allinea con la tradizione di ricerca sul lavoro nell'industria videoludica inaugurata da O'Donnell (2014a)) e proseguita da Weststar & Legault (2017a), che hanno mostrato come i metodi qualitativi siano particolarmente adatti a cogliere le tensioni tra pratiche dichiarate e pratiche effettive, tra la retorica manageriale e l'esperienza vissuta dei lavoratori. Al tempo stesso, la ricerca non rinuncia a raccogliere elementi di natura descrittivo-quantitativa – come la diffusione di specifici framework o la frequenza dei periodi di crunch riferita dagli intervistati – che contribuiscono a contestualizzare e corroborare le interpretazioni qualitative.

Lo strumento principale di raccolta dei dati è l'intervista semi-strutturata, condotta con professionisti attivi nell'industria videoludica italiana. La scelta di focalizzarsi sul contesto italiano risponde a una precisa motivazione: l'industria videoludica italiana è un caso scarsamente studiato dalla ricerca accademica internazionale, nonostante la sua crescita significativa nell'ulti-

mo decennio e la presenza di realtà produttive di diversa scala – da studi indie a sussidiarie di publisher internazionali – che offrono una varietà di contesti analitici di rilievo.

3.2 Giustificazione della scelta qualitativa

La scelta di un approccio qualitativo è motivata da ragioni convergenti, ciascuna delle quali deriva direttamente dalle lacune identificate nel capitolo precedente. La prima riguarda la natura stessa del fenomeno indagato: il mascheramento del crunch – una delle ipotesi centrali di questa tesi – è per definizione qualcosa che sfugge alla misurazione diretta, poiché se il crunch fosse visibile e misurabile nelle sue nuove forme, non sarebbe mascherato. Individuarlo richiede di accedere alle percezioni soggettive dei lavoratori, alle pratiche informali che affiancano o contraddicono le procedure ufficiali, e alle narrazioni con cui i professionisti interpretano la propria esperienza lavorativa: elementi inaccessibili a un questionario standardizzato, ma che emergono naturalmente nel contesto di un'intervista approfondita. La seconda ragione riguarda invece la posizione degli intervistati: come si argomentava nella rassegna della letteratura, i profili intermedi della gerarchia organizzativa – producer, project manager, Scrum master – sono i più rilevanti per comprendere le dinamiche di implementazione dell'Agile, ma anche quelli che nella ricerca esistente hanno trovato meno spazio. Raggiungere queste voci richiede un metodo capace di esplorare esperienze complesse e posizioni talvolta contraddittorie, che un questionario chiuso tenderebbe invece a livellare. L'ultima ragione è infine di natura epistemica: in un campo in cui la letteratura è ancora frammentata e le domande di ricerca sono prevalentemente esplorative, l'approccio qualitativo permette di generare ipotesi e costrutti teorici che potranno essere testati in ricerche future con metodi più formalizzati. Questa tesi non aspira a produrre generalizzazioni statistiche, ma a sviluppare una comprensione più precisa e teoricamente fondata di un fenomeno che la ricerca ha nominato senza ancora analizzare in profondità

3.3 Il campione: criteri di selezione e composizione

3.3.1 Criteri di selezione

I partecipanti sono stati selezionati secondo un criterio di campionamento intenzionale (*purposive sampling*), che privilegia la rilevanza teorica rispetto alla rappresentatività statistica. In un

disegno qualitativo, l'obiettivo non è riprodurre la distribuzione della popolazione di riferimento, ma selezionare i casi più informativi rispetto alle domande di ricerca.
I criteri di inclusione applicati sono tre.

1. Il primo è l'esperienza diretta con processi produttivi videoludici: tutti i partecipanti devono aver lavorato attivamente allo sviluppo di almeno un titolo, in modo da poter parlare delle dinamiche di crunch e delle pratiche organizzative a partire da un'esperienza concreta e non meramente teorica

2. Il secondo è la posizione organizzativa intermedia o senior: la ricerca si concentra su producer, project manager, director e figure equivalenti, che si trovano all'intersezione tra le pressioni del management e le condizioni di lavoro dei team.

3. Il terzo è la varietà di contesti produttivi: il campione include deliberatamente studi di dimensioni, localizzazioni e modelli di business diversi, per permettere confronti tra contesti e individuare variazioni significative nelle dinamiche indagate.

3.3.2 Composizione del campione

Il campione è composto da 10 professionisti afferenti a 7 studi videoludici con sede in Italia, distribuiti tra Milano, Torino, Roma e Bologna. La composizione per studio è la seguente:

Azienda	**Luogo**	**Nome**	**Ruolo**
Jyamma Games	Milano	Andrea Beneduci	Executive Producer
		Edoardo Basile	Producer
Reply games	Milano	Samuele Perseo	Business Development Manager
Ubisoft milan	Milano	Alessandro Arndt Mucchi	Production Director
		Selvaggia Salvati	Producer
Studio Evil	Bologna	Domiziana Suprani	Producer
One O one games	Roma	Gianluca Nardis	Game Producer
Milestone	Milano	Silvia Brandi	Project Manager
		Lisa De Capitani	Project Manager
34bigthings	Torino	Matteo Adam Lanteri	Producer

Tabella 3.1: Lista degli intervistati.

La varietà degli studi presenti nel campione rappresenta un elemento di forza del disegno di ricerca. Il campione include realtà molto diverse tra loro: studi indie o di piccole dimensioni come Studio Evil; studi di medie dimensioni come Jyamma Games, 34BigThings e One O One Games; sussidiarie di grandi publisher internazionali come Ubisoft Milan; publisher e studi con funzioni produttive come Reply Games e Milestone. Questa eterogeneità permette di esplorare se e come le dinamiche di crunch e l'adozione dell'Agile varino in funzione della scala organizzativa, del modello di business e del grado di autonomia rispetto a publisher esterni. Sul piano dei profili coinvolti, il campione comprende prevalentemente figure operative di medio livello come producer, game producer e project manager – coerentemente con l'obiettivo di dare voce ai livelli intermedi della gerarchia produttiva – integrate da figure di livello più senior come executive producer e production director, che offrono una prospettiva sulle scelte strategiche e organizzative degli studi. È presente inoltre un profilo con funzioni di business development, che arricchisce il quadro con una prospettiva trasversale tra produzione e relazioni commerciali.

3.4 Lo strumento: l'intervista semi-strutturata

3.4.1 Struttura delle domande

Le interviste sono condotte in modalità semi-strutturata: ogni conversazione segue una traccia di domande predefinite, ma lascia spazio all'intervistatore di approfondire temi emergenti, chiedere chiarimenti e seguire direzioni impreviste che si rivelino analiticamente rilevanti. Questa flessibilità è essenziale in un disegno esplorativo: alcune delle dinamiche più significative emerse nella letteratura sul crunch – il controllo normativo, la velocity inflation, la normalizzazione del sovraccarico – sono fenomeni che i partecipanti difficilmente nominano spontaneamente, ma che emergono attraverso domande indirette e approfondimenti situazionali.

La traccia di intervista si articola in cinque domande principali, progettate per coprire progressivamente i temi centrali della ricerca:

1. La prima domanda – relativa alla metodologia di sviluppo adottata dallo studio – ha funzione orientativa: permette di contestualizzare le risposte successive e di identificare il grado di formalizzazione delle pratiche Agile nello studio dell'intervistato. Non si

assume che tutti gli studi adottino un framework codificato: la domanda è aperta proprio per cogliere la varietà tra approcci formalizzati, ibridi e informali.

2. La seconda domanda – sulla pianificazione e gestione delle attività – approfondisce le pratiche operative concrete: come vengono usati sprint, backlog e milestone nella vita quotidiana del team. Questa domanda permette di distinguere tra un'adozione Agile genuina e una adozione di facciata, in cui il vocabolario metodologico è presente ma le pratiche sottostanti restano sostanzialmente tradizionali.

3. La terza domanda – sulle difficoltà nella gestione delle scadenze – introduce il tema delle pressioni temporali senza nominare esplicitamente il crunch, permettendo agli intervistati di descrivere le proprie esperienze con il linguaggio che preferiscono. Questa scelta è deliberata: il termine "crunch" ha connotazioni negative che possono indurre risposte difensive, specialmente in rappresentanti di studi che si presentano pubblicamente come attenti al benessere dei propri dipendenti.

4. La quarta domanda – la più diretta, esplicitamente focalizzata sul rapporto tra Agile e crunch – è posizionata dopo le prime tre proprio perché a questo punto dell'intervista il rapporto di fiducia con l'intervistatore è già stabilito e il partecipante ha già articolato la propria visione del processo produttivo. La domanda chiede esplicitamente se, secondo la loro esperienza, l'Agile aiuti a gestire il carico di lavoro: la specificazione "secondo la vostra esperienza" segnala che si ricerca una valutazione soggettiva e situated, non una risposta teorica.

5. La quinta domanda – sulle strategie adottate durante i picchi di pressione produttiva – è progettata per raccogliere dati sulle pratiche concrete di gestione del crunch, indipendentemente da come i partecipanti le nominino. È in questa domanda che emergono con maggiore frequenza le tensioni tra pratiche dichiarate e pratiche effettive, e che i meccanismi di mascheramento – ove presenti – tendono a rendersi visibili.

3.4.2 Modalità di conduzione

Le interviste sono condotte individualmente, in modo da evitare che la presenza di colleghi dello stesso studio inibisca risposte candide su dinamiche organizzative potenzialmente sensibili. La durata prevista è di venticinque-trenta minuti per ciascuna intervista. Le interviste

sono registrate – previo consenso esplicito del partecipante – e successivamente trascritte per l'analisi.

3.5 Procedura di analisi dei dati

I dati raccolti attraverso le interviste sono analizzati seguendo i principi dell'analisi tematica, un metodo consolidato nella ricerca qualitativa in ambito organizzativo e di studi sul lavoro. Il processo si articola in sei fasi: familiarizzazione con i dati attraverso lettura ripetuta delle trascrizioni; generazione dei codici iniziali; ricerca di temi ricorrenti tra i codici; revisione e raffinamento dei temi; definizione e denominazione dei temi finali; produzione del report analitico.

La codifica è di tipo ibrido: parte dei codici sono definiti *a priori* sulla base della letteratura – ad esempio, scope creep, controllo normativo, velocity inflation, pressione da publisher – e parte emergono induttivamente dai dati, permettendo di catturare dinamiche non anticipate dalla cornice teorica. Questo approccio ibrido è particolarmente adatto a una ricerca esplorativa che intende sia verificare ipotesi derivate dalla letteratura sia generare nuove categorie analitiche.

L'analisi mira a identificare sia i pattern trasversali – temi e dinamiche che ricorrono in più studi e profili – sia le variazioni significative tra contesti diversi, che permettono di formulare ipotesi sulle condizioni che favoriscono o ostacolano l'efficacia dell'Agile nel contenimento del crunch.

3.6 Validità, affidabilità e limiti

Validità

La validità di una ricerca qualitativa non si misura con gli stessi criteri di una ricerca quantitativa, ma si valuta in termini di credibilità, trasferibilità e coerenza interna. La credibilità è garantita dalla varietà del campione – che include studi di dimensioni, localizzazioni e modelli di business diversi – e dalla struttura semi-strutturata delle interviste, che permette di approfondire e verificare le interpretazioni durante la conversazione stessa. La trasferibilità – la possibilità di applicare le conclusioni a contesti diversi da quelli studiati – è limitata dall'ancoraggio al contesto italiano, ma è supportata dalla descrizione densa dei casi, che permette al lettore di valutare autonomamente la pertinenza delle conclusioni rispetto al proprio contesto.

Affidabilità

L'affidabilità del processo analitico è garantita attraverso la trasparenza del processo di codifica – reso esplicito nella presentazione dei risultati – e dalla coerenza nell'applicazione dei codici all'intero corpus delle trascrizioni. Nella misura in cui le risorse lo permettono, si adotta il principio della triangolazione delle fonti: le affermazioni degli intervistati sono confrontate con dati pubblicamente disponibili sugli studi (titoli prodotti, modelli di business, comunicazioni ufficiali) per identificare eventuali discrepanze tra narrativa pubblica e esperienza riferita privatamente.

Limiti

Tre limiti strutturali del disegno di ricerca meritano di essere riconosciuti esplicitamente. Il primo è il bias di selezione: i professionisti disponibili a partecipare a un'intervista sul crunch sono probabilmente quelli con esperienze meno traumatiche o con una posizione lavorativa sufficientemente stabile da non temere conseguenze. Le voci di chi ha lasciato il settore per ragioni legate al crunch – potenzialmente le più rilevanti – sono sistematicamente assenti da questo disegno di ricerca. Il secondo è il bias di desiderabilità sociale: gli intervistati, specialmente quelli che rappresentano studi con una visibilità pubblica significativa, possono tendere a presentare le pratiche del proprio studio in una luce favorevole. Le domande indirette e la struttura progressiva dell'intervista mitigano parzialmente questo rischio, ma non lo eliminano. Il terzo limite è la focalizzazione geografica: il campione italiano, pur offrendo un contributo originale alla ricerca, non permette generalizzazioni all'industria internazionale, che presenta dinamiche produttive e culturali significativamente diverse, specialmente nei contesti AAA nordamericani e nelle grandi realtà asiatiche.

3.7 Considerazioni etiche

La ricerca è condotta nel rispetto dei principi fondamentali dell'etica della ricerca con soggetti umani. Tutti i partecipanti sono stati informati in modo chiaro e trasparente degli obiettivi della ricerca, della natura volontaria della partecipazione e del loro diritto di ritirare il consenso in qualsiasi momento senza conseguenze. Le interviste sono registrate solo previo consenso esplicito; in assenza di consenso, il ricercatore prende appunti scritti durante la conversazione.

I dati raccolti sono trattati nel rispetto della normativa vigente in materia di protezione dei dati personali. Nelle citazioni utilizzate nell'analisi, i partecipanti sono identificati attraverso un sistema di pseudonimizzazione che preserva le informazioni analiticamente rilevanti – ruolo, dimensione dello studio, modello di business – senza permettere l'identificazione diretta delle persone. Agli intervistati è offerta la possibilità di richiedere la revisione delle citazioni che li riguardano prima della pubblicazione finale.

Un'attenzione particolare è dedicata alla asimmetria di potere implicita nella ricerca: alcuni intervistati ricoprono posizioni di responsabilità in studi che potrebbero essere riconoscibili anche attraverso la pseudonimizzazione. In questi casi, le citazioni sono ulteriormente generalizzate per proteggere sia l'intervistato sia lo studio, senza compromettere la sostanza analitica del dato.

Capitolo 4

Risultati

I risultati presentati in questo capitolo derivano dall'analisi tematica del corpus di dieci interviste semi-strutturate condotte con professionisti dell'industria videoludica italiana. La presentazione segue la struttura delle otto domande di ricerca - principale e secondarie - definite nel Capitolo 3, organizzando i dati in modo da rispondere progressivamente alle domande poste senza anticiparne l'interpretazione. Quest'ultima è riservata al Capitolo 5.

I risultati sono organizzati in quattro sezioni. La prima descrive le pratiche metodologiche effettivamente adottate dagli studi del campione. La seconda presenta i risultati relativi al rapporto tra metodologie Agile e crunch, così come emerge dalle percezioni degli intervistati. La terza documenta i meccanismi attraverso cui la pressione lavorativa si genera e si distribuisce. La quarta riporta le evidenze sulle pressioni strutturali esterne agli studi. Per ciascuna sezione, le evidenze qualitative sono integrate da tabelle di sintesi che restituiscono la distribuzione dei pattern nel campione.

4.1 Le pratiche metodologiche adottate

4.1.1 Distribuzione dei framework

La prima domanda esplorativa riguardava quale metodologia di sviluppo viene effettivamente adottata dagli studi del campione. I risultati mostrano che nessuno studio adotta un singolo framework nella sua forma codificata. La pratica universale – presente in tutti e dieci gli intervistati – è quella dei modelli ibridi, ovvero combinazioni personalizzate di elementi prove-

nienti da Scrum, Kanban e Waterfall, modulate in funzione della fase produttiva e del contesto organizzativo.

4.1.2 Modulazione per fase produttiva

I dati mostrano un pattern consistente nella modulazione del framework in funzione della fase produttiva. Kanban o approcci a bassa struttura vengono preferiti nelle fasi di ricerca e sviluppo e di prototipazione, dove l'incertezza è alta e la pianificazione a sprint risulterebbe artificiosamente rigida. Scrum viene applicato prevalentemente nella fase di produzione piena, dove il lavoro è sufficientemente definibile da poter essere scomposto in sprint. Waterfall riemerge nelle fasi finali – pipeline artistiche procedurali, fasi di certificazione, lavoro su asset con dipendenze lineari – e come struttura macro per la definizione delle milestone lungo l'intero progetto.

4.1.3 Durata degli sprint

La durata degli sprint varia significativamente sia tra studi diversi che all'interno dello stesso studio in funzione della fase produttiva. Gli sprint settimanali vengono adottati prevalentemente da studi di piccole e medie dimensioni nelle fasi di produzione attiva, dove la cadenza ravvicinata permette aggiustamenti frequenti. Gli sprint bisettimanali rappresentano lo standard più diffuso negli studi strutturati, in particolare in quelli con team numerosi o con vincoli di milestone contrattuali. Sprint più lunghi – tre o quattro settimane – vengono impiegati nelle fasi esplorative o in contesti in cui il lavoro non è sufficientemente granulare da giustificare una review settimanale.

4.2 Il rapporto tra Agile e crunch

4.2.1 Posizioni sulla domanda centrale

La domanda centrale dell'intervista – se l'Agile aiuti a gestire il carico di lavoro e a prevenire il crunch – produce un ventaglio di risposte che non si distribuisce su un asse binario sì/no, ma su un continuum che va dalla negazione esplicita all'affermazione condizionata. Nessun intervistato risponde affermativamente in modo incondizionato.

Figura 4.1: Rapporto Agile-Crunch

Le posizioni di negazione esplicita provengono prevalentemente da producer di studi indie o di studi che gestiscono più progetti contemporaneamente. In questi contesti, l'Agile viene descritto come uno strumento che, per la sua stessa natura adattiva, apre spazio a derive di scope e perfezionismo che generano pressione lavorativa piuttosto che ridurla. La posizione ricorrente è che il problema non risieda nel metodo in sé, ma nella natura umana dei team creativi, che l'Agile – con la sua enfasi sulla flessibilità – tende ad amplificare anziché contenere.

Le posizioni di ambivalenza condizionata sono le più articolate nel corpus e provengono da profili diversi per ruolo e contesto. Il denominatore comune è la distinzione tra teoria e pratica: l'Agile, in linea teorica, dovrebbe ridurre il crunch; in linea pratica, l'esito dipende da variabili che il framework stesso non controlla – cultura organizzativa, pressioni esterne, dimensione del team, presenza o assenza di publisher. Un game producer di uno studio medio sintetizza questa posizione osservando che l'Agile funzionerebbe se il progetto e l'azienda lo consentissero – ma che nella pratica queste condizioni raramente sono tutte soddisfatte simultaneamente.

Le posizioni di affermazione condizionata provengono prevalentemente da studi di grandi dimensioni o strutturati, con team dedicati e processi formalizzati. In questi contesti, il beneficio principale attribuito all'Agile non è la riduzione del crunch in senso assoluto, ma la sua trasformazione da fenomeno sistemico a evento circoscritto: l'Agile non elimina il crunch, ma ne limita la durata e la frequenza, concentrandolo in momenti eccezionali anziché distribuirlo lungo l'intero sviluppo.

4.2.2 Benefici concreti attribuiti all'Agile

Indipendentemente dalla posizione complessiva sul rapporto Agile–crunch, gli intervistati identificano una serie di benefici concreti che il framework porta alla gestione del lavoro. I più citati sono riportati nella tabella seguente.

Beneficio	Descrizione operativa	n
Visibilità sull'avanzamento	Backlog e sprint rendono lo stato del progetto leggibile	8/10
Identificazione precoce problemi	Review iterative permettono di correggere prima che sia tardi	7/10
Gestione delle priorità	Backlog ordinato riduce lavoro su feature non critiche	7/10
Autonomia e motivazione del team	Auto-organizzazione aumenta ownership e senso di progresso	6/10
Adattabilità ai cambiamenti	Iterazioni brevi facilitano pivot senza costi eccessivi	6/10
Standardizzazione dei flussi	Linguaggio e processi comuni tra team e dipartimenti	5/10

* n = numero di intervistati che hanno menzionato esplicitamente il beneficio.

Tabella 4.1: Benefici Agile

4.2.3 Limiti e criticità attribuiti all'Agile

Accanto ai benefici, gli intervistati identificano una serie di limiti che riducono o annullano l'efficacia dell'Agile nel contenere il crunch. I più ricorrenti sono sintetizzati nella tabella seguente.

Limite / criticità	Meccanismo descritto	n
Adozione come strumento, non mentalità	Cerimonie formali senza cambiamento culturale reale	7/10
Pressioni esterne non controllabili	Publisher, engine, licenze annullano i benefici interni	8/10
Flessibilità come amplificatore di scope	L'adattabilità giustifica aggiunte continue non pianificate	6/10
Multi-progetto e focus frammentato	Agile presuppone un team dedicato: non vale per PMI multi-progetto	4/10
Incompatibilità con la creatività non lineare	Sprint non si adattano a lavoro non granularizzabile	6/10

* n = numero di intervistati che hanno menzionato esplicitamente il limite.

Tabella 4.2: Limiti Agile

4.3 Meccanismi di generazione e distribuzione della pressione lavorativa

4.3.1 La specificità italiana

Una delle evidenze più consistenti del corpus riguarda la distanza tra le dinamiche di crunch documentate nella letteratura internazionale e quelle riportate dagli intervistati nel contesto italiano. Otto su dieci intervistati fanno riferimento esplicito alle tutele del diritto del lavoro italiano come fattore che limita strutturalmente la possibilità di imporre straordinari prolungati. La quantità di overtime effettivamente riportata è significativamente inferiore ai casi documentati nella letteratura angloamericana: le descrizioni si concentrano su qualche sabato nel corso di un progetto pluriennale, periodi limitati di qualche ora settimanale aggiuntiva, o fasi di rush finale della durata di poche settimane.

Nessun intervistato descrive episodi paragonabili ai casi estremi documentati da Schreier (2017) o ai dati IGDA su settimane da 60-80 ore prolungate per mesi. Questo non implica l'assenza di pressione lavorativa, ma segnala che la sua forma nel contesto italiano è strutturalmente diversa da quella prevalente nella letteratura di riferimento.

4.3.2 Il crunch volontario: pattern e distribuzione

Un pattern emerso induttivamente dal corpus riguarda una forma di overtime che non viene riconosciuta come crunch dagli stessi intervistati, perché nasce dall'interno dei team piuttosto che da coercizione esplicita. Questo fenomeno – definito in questa analisi crunch volontario – è presente in cinque interviste e assume caratteristiche ricorrenti.

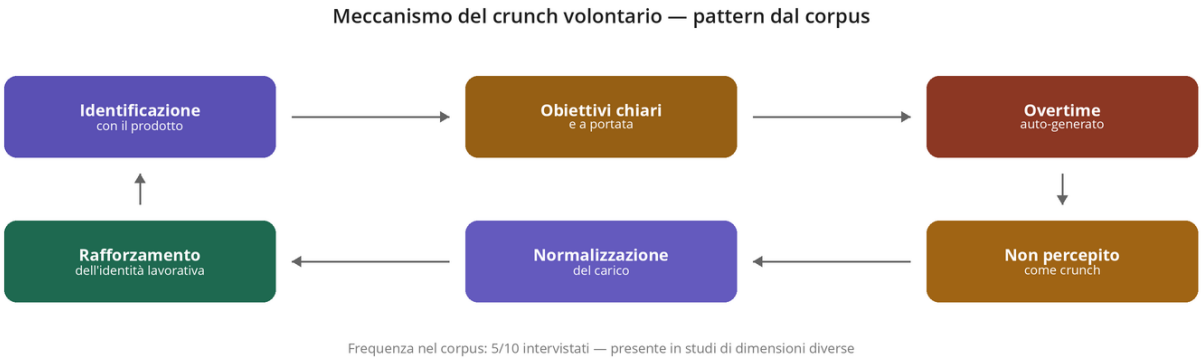

Figura 4.2: Schema Crunch

Il meccanismo descritto dagli intervistati segue un pattern ricorrente: l'identificazione dei lavoratori con il prodotto genera una disposizione a lavorare oltre l'orario previsto in modo spontaneo, senza richiesta esplicita da parte del management. Nei casi documentati, questo overtime viene descritto in termini positivi – come prova di coinvolgimento, di qualità del team, di cultura organizzativa sana. La sua natura problematica non viene riconosciuta dagli stessi intervistati nel momento della descrizione.

Un elemento ulteriore emerge in due interviste: la passione dei lavoratori viene esplicitamente riconosciuta dal management come risorsa disponibile nei momenti di pressione, anche quando non viene direttamente sollecitata. Questo dato – emerso in modo incidentale, non come risposta alla domanda sul crunch ma come descrizione delle strategie di gestione della pressione – segnala la presenza di un meccanismo di utilizzo implicito della motivazione intrinseca dei lavoratori come leva gestionale.

4.3.3 Le principali fonti di crunch per fase

Sulla base delle descrizioni degli intervistati, è possibile mappare le principali fonti di pressione lavorativa per fase produttiva. I dati mostrano che le fonti variano significativamente in funzione della fase, confermando l'eterogeneità del fenomeno già ipotizzata nel Capitolo 2.

Fonti di pressione lavorativa per fase produttiva

Fase	Fonti di pressione prevalenti (con frequenza nel corpus)
Pre-produzione / R&D	Pitch/greenlight con scadenza (5/10) · Pivoting loop (4/10) Crunch da commitment (3/10)
Prototipazione	Debito tecnico da iterazione rapida (7/10) · Perfezionismo creativo (5/10) Crunch volontario (4/10)
Produzione	Scope creep (8/10) · Dipendenze inter-team (6/10) Rework di feature (7/10) · Velocity inflation (4/10)
Alpha / Beta	Saldo debito tecnico (8/10) · Bug/regressioni (7/10) Pressione milestone contrattuale (6/10)
Gold / Post-lancio	Certificazioni (5/10) · Licenze (3/10) · Rush finale da scadenza fissa (9/10)

Tabella 4.3: Principali fonti del crunch per fase

4.4 Le pressioni strutturali esterne

4.4.1 Presenza di publisher e autonomia finanziaria

Una variabile che attraversa il corpus in modo consistente è la presenza o assenza di un publisher esterno, che si configura come fattore strutturante delle dinamiche di crunch indipendentemente dalla metodologia adottata. Il corpus permette di distinguere tre configurazioni principali.

Configurazioni produttive e dinamiche di pressione associate

Configurazione	Fonte pressione prevalente	Efficacia Agile sul crunch
Studio senza publisher (autofinanziato)	Scope creep interno Perfezionismo creativo	Parziale — flessibilità ma scope incontrollato
Studio con publisher (milestone contrattuali)	Scadenze non negoziabili Pressione esterna diretta	Limitata — crunch strutturale residuo
Studio AAA / sussidiaria (struttura interna)	Licenze, certificazioni Richieste corporate	Moderata — crunch circoscritto e gestito

Tabella 4.4: Configurazioni produttive

4.4.2 Dipendenze tecnologiche e imprevisti non pianificabili

Otto intervistati su dieci fanno riferimento a fonti di pressione legate a fattori tecnologici non controllabili dallo studio: aggiornamenti obbligatori dell'engine imposti da scadenze di certificazione dei platform holder, bug sistemici emersi in fase di integrazione, limitazioni non previste degli strumenti di terze parti. Questi eventi vengono descritti come generatori di crunch indipendenti dalla qualità della pianificazione e dalla metodologia adottata – situazioni in cui nessun framework avrebbe potuto prevenire la pressione.

La distribuzione di questi eventi segue un pattern temporale specifico: tendono a manifestarsi nelle fasi di alpha e beta, quando il lavoro di integrazione tra sistemi precedentemente sviluppati in parallelo rivela incompatibilità non anticipabili nelle fasi precedenti. Questo concentra la pressione nelle settimane critiche prima delle milestone contrattuali, producendo un picco di carico che si sovrappone alla pressione già elevata di quella fase.

4.5 Il ruolo del producer come mediatore della pressione

4.5.1 Funzioni di mediazione documentate

Un pattern emerso induttivamente in sei interviste riguarda il ruolo del producer e del project manager come figura che assorbe, filtra e redistribuisce la pressione organizzativa tra il livello strategico e il team operativo. Le funzioni di mediazione descritte nel corpus sono riportate nella tabella seguente.

Funzione	Modalità operativa descritta	n
Assorbimento della pressione	Filtrare le richieste dall'alto prima di trasmetterle	6/10
Intervento psicologico	Gestione dello stato emotivo del team con dati	5/10
Riordino delle priorità	Taglio delle feature non critiche in fase di crisi	7/10
Gestione info verso il team	Comunicare solo info filtrate e al momento giusto	5/10
Negoziazione scope	Far capire ai superiori i limiti reali del team	6/10

* n = numero di intervistati che hanno descritto la funzione, anche senza nominarla esplicitamente.

Tabella 4.5: Rapporto tra funzione e modalità operativa

4.5.2 Il costo nascosto del ruolo di mediazione

In tre interviste emerge, in modo incidentale e non sollecitato dalla traccia, un riferimento al costo personale associato al ruolo di mediazione. I producer che descrivono se stessi come figura di assorbimento della pressione non nominano questo aspetto come problematico, ma le descrizioni contengono segnali di un carico lavorativo specifico del ruolo – emotivo e relazionale oltre che tecnico – che non compare nelle statistiche sugli straordinari e che non viene riconosciuto come forma di crunch né dagli stessi intervistati né, apparentemente, dalle organizzazioni in cui operano.

4.6 Sintesi dei risultati

I risultati dell'analisi tematica possono essere sintetizzati in sei evidenze principali, organizzate in ordine di rilevanza rispetto alle domande di ricerca.

Prima evidenza: Nessuno studio del campione adotta un framework Agile nella sua forma codificata. La pratica universale è quella dei modelli ibridi, modulati per fase produttiva e contesto organizzativo. Questo dato è rilevante perché rende impropria qualsiasi valutazione dell'Agile come entità unitaria nel contesto videoludico italiano.
Seconda evidenza: La posizione dominante nel corpus è di accettazione pragmatica dell'Agile: il framework viene utilizzato perché non esistono alternative adeguate alla natura creativa del videogioco, non perché venga considerato efficace nel ridurre il crunch. Nessun intervistato esprime un'affermazione incondizionata sull'efficacia dell'Agile rispetto al crunch.
Terza evidenza: Il crunch nella forma estrema documentata dalla letteratura internazionale è strutturalmente raro nel contesto italiano, attribuito dagli intervistati principalmente alle tutele del diritto del lavoro. Le forme di pressione lavorativa presenti nel campione sono quantitativamente inferiori ma strutturalmente diverse – più diffuse, meno visibili e meno riconosciute come problematiche.
Quarta evidenza: Il corpus documenta la presenza di crunch volontario in cinque studi su dieci: overtime auto-generato dalla passione dei lavoratori per il prodotto, non riconosciuto come crunch né dai lavoratori né dal management, e in alcuni casi esplicitamente riconosciuto come risorsa gestionale disponibile nei momenti di pressione.
Quinta evidenza: Le pressioni strutturali esterne – publisher, engine, licenze, certificazioni –

vengono identificate da otto intervistati su dieci come la principale causa di crunch residuo, indipendentemente dalla metodologia adottata. Questo segnala un limite strutturale dell'Agile che nessuna implementazione interna può superare.

Sesta evidenza: Il producer emerge come figura centrale nel determinare se e come la pressione lavorativa diventa visibile nel team. Le sue funzioni di mediazione – assorbimento, filtraggio, riordino delle priorità, gestione delle informazioni – producono un effetto di contenimento della pressione sul team che ha tuttavia un costo personale per chi svolge il ruolo, sistematicamente invisibile nelle metriche organizzative standard.

Capitolo 5

Discussioni

Questo capitolo interpreta i risultati presentati nel Capitolo 4 alla luce della domanda di ricerca principale e dei tre quesiti secondari formulati nel Capitolo 1. L'obiettivo non è riassumere i dati, già sintetizzati nelle sei evidenze principali, bensì discuterne il significato, le implicazioni teoriche e le ricadute pratiche, posizionando i risultati all'interno del dibattito accademico e professionale sul crunch e sull'Agile nell'industria videoludica.

5.1 Risultati principali e secondari

La domanda centrale di questa ricerca chiedeva se l'adozione di metodologie Agile negli studi di sviluppo videoludico costituisca una risposta strutturalmente efficace al problema del crunch, oppure ne trasformi le manifestazioni senza eliminarne le cause. I risultati del corpus consentono di rispondere con una tesi articolata: l'Agile non elimina il crunch, ma può modificarne la forma, la visibilità e la distribuzione lungo il ciclo di sviluppo, con esiti che variano sistematicamente in funzione del contesto organizzativo, della presenza di pressioni esterne e della cultura dello studio.

Questa conclusione non è banale. Essa implica, innanzitutto, che la domanda stessa è mal posta se formulata in termini binari. Il corpus mostra che le due opzioni non sono mutualmente esclusive: l'Agile può ridurre alcune forme di crunch (quelle legate alla disorganizzazione progettuale, alla scarsa visibilità sullo stato di avanzamento, alla propagazione tardiva dei problemi tecnici) e simultaneamente crearne o mascherarne altre (il crunch volontario alimentato dalla logica dello sprint, il crunch da commitment nelle fasi esplorative, il crunch da pressioni

esterne che nessuna metodologia interna può assorbire). Questa simultaneità è la risposta più onesta che i dati consentono di dare.

Il primo quesito secondario chiedeva in quali fasi del ciclo di sviluppo l'Agile dimostri maggiore efficacia nel prevenire il crunch, e in quali tenda a fallire. I risultati indicano una risposta relativamente chiara: l'Agile funziona meglio nella fase di produzione piena, dove il lavoro è sufficientemente granulare da poter essere scomposto in sprint, il backlog prioritizzato rende visibile lo scope creep, e la cadenza delle review permette di identificare precocemente i problemi. I benefici più citati nel corpus – visibilità sull'avanzamento (8/10), identificazione precoce dei problemi (7/10), gestione delle priorità (7/10) – sono tutti benefici che si manifestano con maggiore intensità in questa fase. Al contrario, nelle fasi di pre-produzione e prototipazione l'Agile mostra i suoi limiti strutturali più acuti: la natura esplorativa del lavoro creativo non è scomponibile in sprint con definition of done verificabile, e i tentativi di imporre una struttura iterativa rigida generano o un false sense of control (sprint formalmente completati su task mal definiti) o un adattamento informale che svuota il framework della sua sostanza. Nella fase di alpha e beta, infine, il framework tende a perdere efficacia proprio quando la pressione è più alta: le cerimonie Scrum si svuotano o si trasformano in sessioni di gestione dell'emergenza, confermando quanto già osservato da Schreier (2017) in relazione al fenomeno del cargo cult Agile. Queste evidenze si collegano direttamente alle tensioni strutturali identificate nel Capitolo 2. La tensione tra pianificabilità e creatività si manifesta empiricamente nella difficoltà degli studi di applicare sprint a lavoro non granularizzabile; la tensione tra autonomia del team e pressioni esterne si rispecchia nella quinta evidenza, che identifica nelle pressioni strutturali esterne la principale causa residua di crunch indipendentemente dalla metodologia; la tensione tra trasparenza e controllo trova riscontro nel crunch volontario, dove gli strumenti di visibilità possono essere mobilitati per amplificare la pressione piuttosto che proteggervi.

Il secondo quesito secondario interrogava i meccanismi attraverso cui l'Agile può trasformare il crunch da fenomeno imposto esternamente a fenomeno auto-generato dai team, e se questo cambiamento rappresenti un miglioramento reale delle condizioni lavorative. La quarta evidenza fornisce una risposta empiricamente fondata: il crunch volontario, presente in cinque studi su dieci, segue un pattern ricorrente e riconoscibile – identificazione con il prodotto, percezione degli obiettivi come raggiungibili, overtime auto-generato, normalizzazione del carico, rinforzo

dell'identità lavorativa – che corrisponde precisamente al meccanismo di controllo normativo teorizzato da Peticca-Harris et al. (2015). La rilevanza di questo risultato è duplice. Da un lato conferma empiricamente, per la prima volta in un campione italiano, una dinamica che la letteratura aveva finora prevalentemente teorizzato. Dall'altro, e forse più significativamente, mostra che questa forma di crunch non viene percepita come tale né dai lavoratori né dal management: viene descritta in termini positivi, come indicatore di motivazione e cultura aziendale sana. Questo invisibilità è esattamente ciò che la letteratura critica sull'Agile intendeva con il concetto di mascheramento: il problema non scompare, ma perde i suoi marcatori di riconoscibilità. Quanto al quesito se questo cambiamento rappresenti un miglioramento reale, la risposta che i dati suggeriscono è ambigua. In senso stretto, sostituire la coercizione esplicita con la pressione internalizzata non costituisce di per sé un miglioramento delle condizioni lavorative, anche se riduce il conflitto visibile tra lavoratori e management. In senso più ampio, tuttavia, il fatto che il crunch italiano sia quantitativamente inferiore ai casi estremi documentati nella letteratura angloamericana è un dato reale, che i risultati attribuiscono in larga misura alle tutele del diritto del lavoro italiano piuttosto che alle metodologie adottate. Questo disaccoppiamento tra efficacia metodologica e condizioni lavorative effettive è uno dei risultati più rilevanti dell'intera ricerca.

Il terzo quesito secondario chiedeva in che misura le pressioni esterne agli studi limitino o annullino la capacità dell'Agile di incidere strutturalmente sul crunch. La quinta evidenza risponde in modo inequivocabile: otto intervistati su dieci identificano le pressioni strutturali esterne – publisher, engine, licenze, certificazioni – come la principale causa di crunch residuo, indipendentemente dalla metodologia. Questo risultato non sorprende alla luce della letteratura esistente: Hutchinson (2021), Weststar & Legault (2022) e Tschang (2007) avevano già mostrato come i contratti con i publisher e le logiche delle finestre di lancio commerciale creino pressioni che nessuna metodologia interna può neutralizzare. Il contributo specifico di questa ricerca è mostrare come questo limite venga esplicitamente riconosciuto dagli stessi practitioner, che adottano l'Agile con consapevolezza pragmatica delle sue limitazioni strutturali, non con la convinzione che esso risolva il problema.

5.2 Implicazioni teoriche

I risultati di questa ricerca producono tre contributi teorici che si posizionano in relazione alla letteratura esistente in modi distinti: il primo la conferma empiricamente, il secondo la estende, il terzo propone una revisione.

Conferma e consolidamento empirico della teoria del controllo normativo.
Il corpus fornisce la prima documentazione empirica sistematica del meccanismo di controllo normativo di Peticca-Harris et al. (2015) in un contesto videoludico italiano. La teoria prevedeva che i lavoratori, avendo negoziato autonomamente gli sprint e le stime, si sentissero moralmente obbligati a rispettare impegni autoimposti, riproducendo il crunch per pressione interna piuttosto che per coercizione esplicita. I dati mostrano che questo meccanismo è attivo, diffuso e – crucialmente – non riconosciuto come tale dai soggetti coinvolti. Questa invisibilità non era specificatamente teorizzata nella letteratura, e costituisce un'estensione empirica del modello: il controllo normativo è più efficace, e più problematico, quando non è percepito come forma di controllo.

Estensione della teoria al ruolo del producer come buffer organizzativo.
La sesta evidenza introduce un contributo teorico che non trova precedenti diretti nella letteratura. Il producer emerge non come semplice figura operativa, ma come meccanismo organizzativo di assorbimento, filtraggio e redistribuzione della pressione tra il livello strategico e il team. Questo ruolo di buffer non è formalizzato nei framework Agile – né Scrum né Kanban prevedono esplicitamente questa funzione – e non è visibile nelle metriche standard di monitoraggio del progetto. Eppure è proprio questo ruolo a determinare, in larga misura, se e come la pressione strutturale esterna diventi crunch vissuto dal team. Questa evidenza suggerisce di integrare la letteratura sul crunch con concetti tratti dalla teoria delle organizzazioni, in particolare dall'analisi delle boundary roles e dalla letteratura sul lavoro emotivo nelle organizzazioni: il producer svolge un lavoro emotivo e relazionale che ha un costo personale reale, sistematicamente invisibile nelle metriche organizzative e, nei dati raccolti, non riconosciuto come forma di crunch.

Revisione critica della dicotomia Agile autentico vs. Agile di facciata.
Una parte significativa della letteratura recente distingue tra un'adozione genuina dei principi

Agile e una adozione superficiale che mantiene il vocabolario senza modificare le strutture di potere. I risultati di questa ricerca suggeriscono che questa dicotomia sia analiticamente insufficiente. Nessuno degli studi del campione adotta un framework codificato in forma pura: tutti operano con modelli ibridi, modulati per fase produttiva e contesto. Questa ibridazione non è necessariamente un segno di adozione superficiale: è una risposta razionale alla natura eterogenea del lavoro videoludico, che richiede approcci diversi in fasi diverse. La domanda non è se si adotti Agile in modo autentico o di facciata, ma quali specifiche pratiche, in quali specifiche fasi, producano benefici reali e quali invece generino nuove forme di pressione. Questa formulazione è più precisa e operativamente utile della dicotomia autenticità/facciata. Sul piano delle teorie organizzative più ampie, i risultati di questa ricerca si inseriscono nel dibattito sul lavoro creativo nell'economia digitale aperto da Dyer-Witheford & De Peuter (2009). L'industria videoludica italiana, pur presentando caratteristiche specifiche legate al contesto normativo e alla scala organizzativa, conferma la dinamica fondamentale descritta da questi autori: il discorso sull'autonomia creativa e sulla cultura organizzativa positiva convive con, e in parte maschera, strutture di pressione che rimangono sostanzialmente invariate rispetto a quelle documentate nella letteratura angloamericana. La differenza quantitativa nel livello di crunch è reale, ma è attribuibile in larga misura al quadro giuridico piuttosto che alle scelte metodologiche

5.3 Implicazioni pratiche

Le implicazioni pratiche di questa ricerca si rivolgono a tre categorie di destinatari: i practitioner (producer, project manager, Scrum master e management degli studi), i ricercatori e, in prospettiva più ampia, chi si occupa di regolazione del lavoro nel settore creativo digitale.

Per i practitioner.
Il risultato più immediatamente operativo è la mappatura delle condizioni in cui l'Agile produce benefici reali rispetto a quelle in cui rischia di amplificare le pressioni. Nella fase di produzione piena, con team dedicati e scope gestibile internamente, i framework iterativi mostrano un valore concreto nella gestione delle priorità e nell'identificazione precoce dei problemi: queste pratiche andrebbero adottate e presidiate con cura, investendo nella formazione dei Product Owner affinché abbiano effettivo potere decisionale sullo scope e non siano vincolati da aspettative contrattuali che rendono il taglio di feature politicamente impossibile. Nelle fasi di pre-

produzione e prototipazione, al contrario, l'imposizione di strutture iterative rigide rischia di produrre più danni che benefici. I dati suggeriscono che in queste fasi siano più utili approcci a bassa struttura – come Kanban o approcci informali di tipo spike – che legittimano il lavoro esplorativo senza richiedere la granularizzazione artificiale di task per natura non scomponibili. La raccomandazione pratica è di adottare una metodologia che moduli esplicitamente la struttura in funzione della fase: non un unico framework per tutto il progetto, ma un'architettura metodologica consapevole che preveda transizioni definite tra approcci diversi. Il crunch volontario merita un'attenzione specifica da parte del management. I risultati mostrano che questa forma di pressione non viene riconosciuta come problematica, e che in alcuni casi la passione dei lavoratori viene esplicitamente identificata come risorsa disponibile nei momenti di pressione. Questo meccanismo, per quanto comprensibile dal punto di vista gestionale, ha costi a lungo termine documentati dalla letteratura: erosione del benessere, abbandono del settore, perdita di talenti. Una pratica concreta per rendere visibile questo fenomeno è includere nelle retrospettive Agile domande esplicite sul carico lavorativo percepito e sulle ore effettivamente lavorate, creando uno spazio in cui il disagio possa essere nominato senza implicazioni negative. Il prerequisito è che le retrospettive abbiano reale potere di cambiare il ritmo produttivo, il che richiede un allineamento esplicito con il management e, dove presente, con il publisher. Il ruolo del producer come buffer organizzativo è uno dei risultati più operativamente rilevanti. Gli studi dovrebbero riconoscere esplicitamente il costo emotivo e relazionale di questo ruolo, includerlo nelle valutazioni di sostenibilità del lavoro, e evitare di costruire modelli organizzativi che scaricano sistematicamente sul producer la gestione delle contraddizioni tra pressioni esterne e capacità del team. Una pratica concreta è definire esplicitamente i limiti del ruolo di mediazione – cosa il producer può filtrare e cosa deve invece trasmettere – e prevedere meccanismi di supporto (supervisione, peer support tra producer) che rendano visibile il costo di questo lavoro.

Per i ricercatori.

Questa tesi suggerisce alcune direzioni metodologiche per la ricerca futura. In primo luogo, la distinzione tra crunch quantitativo (ore lavorate) e crunch qualitativo (pressione percepita, normalizzazione del sovraccarico, invisibilità delle forme di crunch volontario) dovrebbe diventare una distinzione analitica standard nella ricerca sul benessere lavorativo nell'industria videoludica. Affidarsi esclusivamente alle ore lavorate come proxy del crunch produce dati che

sistematicamente sottostimano il fenomeno, specialmente in contesti come quello italiano dove il quadro giuridico comprime le forme più visibili. In secondo luogo, la ricerca futura dovrebbe sviluppare strumenti di rilevazione del crunch volontario – ad esempio attraverso diari di lavoro longitudinali o questionari che includano item sulla pressione autoimposta – che permettano di rendere misurabile un fenomeno che per definizione sfugge alla misurazione diretta. In terzo luogo, il ruolo del producer come buffer organizzativo merita un programma di ricerca dedicato, che includa sia studi qualitativi approfonditi sia strumenti quantitativi per misurare il carico emotivo e relazionale di queste figure.

Per chi si occupa di regolazione del lavoro.

I risultati suggeriscono che le tutele del diritto del lavoro italiano hanno un effetto reale nel limitare le forme più estreme di crunch, ma che questo effetto è parziale: le forme di crunch invisibile – volontario, da commitment, da pressione del producer – rimangono al di fuori dell'ambito di applicazione delle normative attuali, che regolano le ore lavorate ma non la pressione percepita né il costo emotivo del lavoro di mediazione. Una politica pubblica efficace nel settore creativo digitale dovrebbe includere strumenti di monitoraggio delle condizioni lavorative che vadano oltre il conteggio degli straordinari, includendo indicatori di benessere psicologico e di sostenibilità dei ritmi produttivi.

Capitolo 6

Conclusioni

Questa tesi si è proposta di rispondere a una domanda centrale: se l'adozione di metodologie Agile negli studi di sviluppo videoludico costituisca una risposta strutturalmente efficace al problema del crunch, oppure ne trasformi le manifestazioni senza eliminarne le cause. La domanda era articolata in tre sotto-domande, ciascuna delle quali ha ricevuto risposta attraverso l'analisi tematica del corpus di interviste e il confronto con la letteratura esistente.

La prima sotto-domanda – in quali fasi del ciclo di sviluppo l'Agile dimostra maggiore efficacia nel prevenire il crunch – ha ottenuto una risposta differenziata. I risultati mostrano che l'Agile produce benefici concreti e riconoscibili nelle fasi di produzione piena, dove la trasparenza del backlog e la cadenza degli sprint riducono il rischio di scope creep non governato e permettono l'identificazione precoce dei problemi. L'efficacia si riduce progressivamente nelle fasi di alpha e beta, quando la pressione delle scadenze contrattuali tende a svuotare le cerimonie Agile della loro funzione sostanziale, e diventa marginale nella fase gold, dove le cause strutturali del crunch – scadenze fisse, certificazioni, pressioni dei publisher – operano indipendentemente da qualsiasi framework metodologico interno.

La seconda sotto-domanda – attraverso quali meccanismi l'Agile può trasformare il crunch da fenomeno imposto a fenomeno auto-generato – ha trovato nel corpus una conferma empirica parziale ma significativa. Il meccanismo del crunch volontario, documentato in cinque studi su dieci, mostra come l'identificazione dei lavoratori con il prodotto – amplificata dalla cultura iterativa e dalla chiarezza degli obiettivi che l'Agile promuove – possa generare overtime spontaneo non riconosciuto come crunch né dai lavoratori né dal management. Questo meccanismo non è esclusivamente una conseguenza dell'Agile, ma la struttura iterativa e orientata agli obiettivi del framework crea condizioni favorevoli alla sua manifestazione.

La terza sotto-domanda – in che misura le pressioni esterne agli studi limitino la capacità dell'Agile di incidere strutturalmente sul crunch – ha ricevuto la risposta più netta del corpus. Otto intervistati su dieci identificano publisher, engine di terze parti, licenze e certificazioni come fonti di pressione che operano indipendentemente dalla metodologia adottata. Il risultato convergente è che l'Agile può migliorare le condizioni interne di lavoro, ma non può risolvere il crunch quando le sue cause risiedono in fattori che gli studi non controllano. Sintetizzando: i dati suggeriscono che l'Agile riduce alcune forme di crunch – quelle legate alla disorganizzazione, alla mancanza di trasparenza e alla scoperta tardiva dei problemi – e non incide su altre – quelle radicate nelle pressioni strutturali esterne. Esiste inoltre una terza categoria, la più rilevante per il contributo teorico originale di questa tesi: forme di pressione lavorativa che l'Agile non genera direttamente ma le cui condizioni favorisce, e che sfuggono al riconoscimento proprio perché nascono dall'interno del team anziché dall'esterno. La risposta alla domanda principale è quindi né affermativa né negativa: l'Agile non risolve il crunch, non lo maschera sistematicamente, ma in determinate condizioni ne trasforma le manifestazioni in forme meno visibili e meno riconoscibili. La distinzione tra riduzione e mascheramento dipende in modo critico da tre variabili: la presenza o assenza di pressioni esterne non negoziabili, il grado di adozione culturale – non solo strumentale – del framework, e la consapevolezza con cui il management riconosce e gestisce i meccanismi di pressione auto-generata.

6.1 Delimitazioni

Le delimitazioni di questa ricerca sono le scelte intenzionali che ne definiscono il perimetro e che sono state adottate in funzione degli obiettivi dichiarati, non come conseguenza di vincoli contingenti.

La prima delimitazione riguarda la focalizzazione geografica sul contesto italiano. La scelta di concentrarsi esclusivamente su studi con sede in Italia risponde a due motivazioni convergenti: da un lato, il contesto italiano è sistematicamente assente dalla letteratura accademica sul crunch, che si concentra quasi esclusivamente su casi nordamericani, britannici e giapponesi; dall'altro, il quadro normativo italiano – con le sue tutele sul lavoro straordinario significativamente più robuste rispetto ai mercati angloamericani – offre un caso comparativo di rilievo teorico, permettendo di isolare la variabile metodologica da quella normativa in modo più netto

che in altri contesti.

La seconda delimitazione riguarda la selezione dei profili intervistati. La ricerca si è deliberatamente concentrata su producer, project manager e figure di livello intermedio-senior, escludendo i developer junior e i lavoratori di livello esecutivo. Questa scelta risponde alla lacuna identificata nella letteratura, che sovrarappresenta i developer junior come vittime del crunch e sottorappresenta le figure di mediazione come agenti organizzativi. L'obiettivo non era documentare le esperienze di chi subisce il crunch, ma comprendere i meccanismi attraverso cui viene gestito, trasformato o invisibilizzato.

La terza delimitazione riguarda la scelta del metodo qualitativo in luogo di un approccio quantitativo o misto. Come argomentato nel Capitolo 3, la domanda di ricerca – focalizzata sui meccanismi di trasformazione del crunch piuttosto che sulla sua misurazione – non è rispondibile attraverso indicatori numerici. La ricerca qualitativa permette di accedere alle interpretazioni e alle pratiche informali che un questionario standardizzato livella o invisibilizza. Questa scelta comporta una rinuncia esplicita alla generalizzabilità statistica, compensata dalla profondità analitica dei dati.

La quarta delimitazione riguarda il perimetro temporale del corpus. Tutte le interviste documentano esperienze maturate prevalentemente negli anni 2018-2024, un periodo caratterizzato dall'accelerazione dell'adozione Agile nell'industria e dall'emergere dei modelli games-as-a-service. Esperienze precedenti – in particolare quelle relative ai modelli di sviluppo degli anni 2000-2015 – non sono state sistematicamente esplorate, il che circoscrive la capacità di questa ricerca di tracciare evoluzioni longitudinali nelle pratiche di gestione del crunch.

6.2 Limitazioni

Le limitazioni di questa ricerca sono i vincoli che possono aver influenzato i risultati indipendentemente dalle scelte metodologiche intenzionali, e che ne circoscrivono la validità e la trasferibilità.

La prima e più rilevante limitazione è il bias di selezione del campione. I professionisti di-

sponibili a partecipare a un'intervista sul crunch sono, con alta probabilità, coloro che hanno esperienze meno traumatiche o che si trovano in una posizione lavorativa sufficientemente stabile da non temere conseguenze. Le voci di chi ha abbandonato l'industria videoludica per ragioni legate al sovraccarico lavorativo – potenzialmente le più rilevanti per la domanda di ricerca – sono strutturalmente assenti da questo disegno. Questo produce un campione orientato verso esperienze positive o neutrali, che potrebbe sottorappresentare le forme più acute di crunch presenti anche nel contesto italiano.

La seconda limitazione riguarda il bias di desiderabilità sociale. Diversi intervistati rappresentano studi con una visibilità pubblica significativa nel mercato italiano, e la narrazione del proprio studio come ambiente di lavoro sano e ben organizzato ha un valore reputazionale che può aver influenzato le risposte. Le domande indirette e la struttura progressiva dell'intervista hanno mitigato parzialmente questo rischio, ma non lo eliminano. In particolare, le risposte alla domanda centrale sul crunch – quella più direttamente esposta a risposte socialmente desiderabili – potrebbero essere parzialmente influenzate da questa dinamica.

La terza limitazione riguarda la dimensione del campione. Dieci interviste, pur offrendo un corpus ricco per un'analisi qualitativa esplorativa, non permettono di raggiungere la saturazione tematica necessaria a escludere pattern rilevanti non ancora emersi. In particolare, la distribuzione non uniforme dei profili – con una sovrarappresentazione di studi milanesi e di figure di producer rispetto ad altri ruoli e territori – potrebbe aver limitato la capacità di identificare variazioni significative legate alla localizzazione geografica o alla dimensione del team.

La quarta limitazione è l'assenza di triangolazione con dati osservativi. L'analisi si basa esclusivamente sulle narrazioni degli intervistati, senza possibilità di confrontarle con osservazioni dirette delle pratiche di lavoro negli studi. La distanza tra pratiche dichiarate e pratiche effettive – una delle ipotesi centrali di questa tesi – non può essere verificata attraverso interviste, ma richiederebbe metodi etnografici o osservazione partecipante che esulano dal perimetro di questa ricerca.

La quinta limitazione riguarda la specificità del contesto normativo italiano. Come documentato nel Capitolo 4, il quadro di tutele del diritto del lavoro italiano limita strutturalmente le

forme più estreme di crunch. Questo rende i risultati poco trasferibili ai contesti nordamericani e asiatici dove si concentra la letteratura internazionale di riferimento, e circoscrive la portata delle conclusioni a un contesto normativo specifico.

6.3 Direzioni per la ricerca futura

I risultati di questa tesi aprono una serie di direzioni di ricerca che rispondono sia alle lacune non colmate da questo studio sia alle nuove domande generate dai dati.

1. **Studi longitudinali sull'efficacia dell'Agile nel tempo.** La limitazione più evidente della ricerca qualitativa basata su interviste è l'impossibilità di misurare l'evoluzione delle condizioni lavorative nel corso dell'adozione Agile. Una direzione di ricerca particolarmente rilevante riguarda la conduzione di studi longitudinali che misurino le ore lavorate, la percezione del carico di lavoro e gli indicatori di benessere psicologico prima, durante e dopo l'introduzione di framework Agile in studi che ne erano privi. Questo tipo di ricerca permetterebbe di rispondere alla domanda che questa tesi ha posto senza poter risolvere: l'Agile riduce oggettivamente il crunch, o solo la percezione del crunch?

2. **Ricerca comparativa tra contesti normativi diversi.** Il corpus documenta una specificità italiana che merita approfondimento comparativo. Una ricerca che mettesse a confronto studi con pratiche Agile simili in contesti normativi diversi – Italia, Regno Unito, Canada, Stati Uniti, Giappone – permetterebbe di isolare il contributo della variabile normativa rispetto a quella metodologica nella determinazione del crunch. In altri termini: quanto del minor crunch osservato in Italia è attribuibile all'Agile e quanto alle tutele lavorative? La risposta a questa domanda ha implicazioni sia teoriche che pratiche di primo piano.

3. **Analisi del crunch nei modelli games-as-a-service.** Come anticipato nella rassegna della letteratura e confermato dall'assenza quasi totale di questo tema nel corpus, il crunch post-lancio nei modelli GaaS rimane uno degli ambiti meno studiati dell'industria videoludica. La produzione continua di contenuti stagionali – aggiornamenti, patch, eventi live – genera una forma di pressione lavorativa strutturalmente diversa dal crunch da lancio, distribuita nel tempo e potenzialmente più difficile da riconoscere. Una ricerca dedicata

a questo fenomeno, con un campione che includa specificamente team operanti su titoli live service, colmerebbe una lacuna sia teorica che pratica di notevole rilevanza.

4. **Il costo del ruolo di mediazione per i producer.** Il tema emergente del producer come figura di assorbimento della pressione – documentato in questa tesi in modo preliminare – merita un'indagine dedicata. Una ricerca focalizzata specificamente sui producer e sui project manager, con strumenti che includano misure di burnout e di carico emotivo oltre alle interviste narrative, permetterebbe di quantificare il costo personale di questo ruolo e di valutare se e in che misura l'Agile contribuisca ad alleggerirlo o ad appesantirlo. Questo filone di ricerca avrebbe implicazioni dirette per la progettazione dei ruoli organizzativi negli studi e per le politiche di welfare dei lavoratori del settore.

5. **Test quantitativo delle ipotesi di mascheramento.** Questa tesi ha sviluppato una cornice teorica per distinguere la riduzione autentica del crunch dal suo mascheramento, ma le evidenze raccolte sono di natura qualitativa ed esplorativa. Una direzione di ricerca futura di grande valore sarebbe la traduzione di questa cornice in un set di indicatori misurabili – ore di overtime non dichiarate, percezione soggettiva del carico versus misurazione oggettiva, discrepanza tra velocity pianificata e realizzata – e la loro applicazione a un campione ampio attraverso metodi survey. Questo permetterebbe di testare statisticamente le ipotesi formulate in questa tesi e di valutarne la generalizzabilità oltre il contesto italiano.

6. **Il ruolo della sindacalizzazione e della rappresentanza collettiva.** Un tema completamente assente dal corpus – e dalla letteratura sul crunch in generale – riguarda il potenziale ruolo della rappresentanza sindacale e delle forme di organizzazione collettiva dei lavoratori nel contenimento del crunch. In Italia, dove le tutele del diritto del lavoro sono più robuste che in altri contesti, questa variabile potrebbe risultare particolarmente rilevante. Una ricerca che esplorasse la relazione tra presenza di rappresentanza sindacale, adozione Agile e dinamiche di crunch offrirebbe un contributo originale sia alla letteratura accademica che al dibattito politico sulla regolazione del lavoro nel settore creativo digitale.

Bibliografia

Ågerfalk, P. J. & Fitzgerald, B. (2006). "Flexible and Distributed Software Processes: Old Petunias in New Bowls?" *Communications of the ACM*, 49:10, 26–34.

Anderson, D. J. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business*. Sequim, WA: Blue Hole Press.

Beck, K. (1999). *Extreme Programming Explained: Embrace Change*. Addison-Wesley.

Beck, K. et al. (2001). *Manifesto for Agile Software Development*. `https://agilemanifesto.org/iso/it/manifesto.html`.

Bethke, E. (2003). *Game Development and Production*. Plano, TX: Wordware Publishing.

Blow, J. (2011). *Story in Games*. Conferenza presentata al Game Developers Conference. San Francisco, CA.

Boehm, B. W. (1981). *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice Hall.

Brechner, E. (2015). *Agile Project Management with Kanban*. Redmond, WA: Microsoft Press.

Brooks, F. P. (1975). *The Mythical Man-Month: Essays on Software Engineering*. Reading, MA: Addison-Wesley.

Capobianco, M. T. (2024). *PwC Global Entertainment & Media Outlook 2024-2028*. `https://blog.pwc.it/pwc-global-entertainment-amp-media-outlook-2024-2028/`.

Cockburn, A. (2001). *Agile Software Development*. Boston, MA: Addison-Wesley.

Cunningham, W. (1992). *The WyCash Portfolio Management System*. `http://c2.com/doc/oopsla92.html`.

De Prato, G. et al. (2010). *Born Digital / Grown Digital: Assessing the Future Competitiveness of the EU Video Games Software Industry*. JRC Scientific and Technical Report. Luxembourg: European Commission – Joint Research Centre.

DeMarco, T. & Lister, T. (1987). *Peopleware: Productive Projects and Teams*. Dorset House.

Derby, E. & Larsen, D. (2006). *Agile Retrospectives: Making Good Teams Great*. Raleigh, NC: Pragmatic Bookshelf.

Dyer-Witheford, N. & De Peuter, G. (2009). *Games of Empire: Global Capitalism and Video Games*. Minneapolis, MN: University of Minnesota Press.

EA_spouse (2004). *EA: The Human Story*. `https://ea-spouse.livejournal.com/274.html`.

Fullerton, T. (2014). *Game Design Workshop: A Playcentric Approach to Creating Innovative Games*. Boca Raton, FL: CRC Press.

Gillespie, N. et al. (2010). "Recovering and Rebuilding Trust in the Wake of Failures and Crises". *Research on Managing Groups and Teams*, 13, 273–305.

Gray, K. L. & Leonard, D. J. (2018). *Woke Gaming: Digital Culture and Critical Interventions*. Seattle, WA: University of Washington Press.

Hardt, M. & Negri, A. (2000). *Empire*. Cambridge, MA: Harvard University Press.

Highsmith, J. (2002). *Agile Software Development Ecosystems*. Boston, MA: Addison-Wesley.

Hodgson, D. & Briand, L. (2013). "Controlling the Uncontrollable: 'Agile' Teams and Illusions of Autonomy in Creative Work". *Work, Employment and Society*, 27:2, 308–325.

Humphrey, W. (1989). *Managing the Software Process*. Reading, MA:Pearson Education.

Hutchinson, C. (2021). "Precarity and the Productivity of Play: Labor in the Video Game Industry". *Media, Culture & Society*, 43:6, 1054–1071.

Keith, C. (2010). *Agile Game Development with Scrum*. Upper Saddle River, NJ:Addison-Wesley.

Kerr, A. (2006). *The Business and Culture of Digital Games: Gamework/Gameplay*. London: SAGE Publications.

Lazzarato, M. (1996). "Immaterial Labor". *Radical Thought in Italy: A Potential Politics*. Minneapolis, MN: University of Minnesota Press, 133–147.

Lizardi, R. (2020). "Games as a Service: The Perpetual Console Cycle". *Journal of Consumer Culture*, 20:3, 318–334.

McConnell, S. (1996). *Rapid Development: Taming Wild Software Schedules*. Redmond, WA: Microsoft Press.

O'Donnell, C. (2014a). *Developer's Dilemma: The Secret World of Videogame Creators*. Cambridge, MA: The MIT Press.

O'Donnell, C. (2014b). "Getting Played: Gamification, Bullshit, and the Rise of Algorithmic Surveillance". *Surveillance and Society*, 12:3, 349–359.

Perlow, L. A. (1999). "The Time Famine: Toward a Sociology of Work Time". *Administrative Science Quarterly*, 44:1, 57–81.

Peticca-Harris, A. et al. (2015). "The Perils of Project-Based Work: Attempting Resistance to Extreme Work Practices in Video Game Development". *Organization*, 22:4, 570–587.

Politowski, C. et al. (2021). "Are Video Games Development Projects Agile? An Empirical Study Using GitHub". *Journal of Systems and Software*, 176, 1–15.

Royce, W. W. (1970). "Managing the Development of Large Software Systems". *Proceedings of IEEE WESCON*, 1–9.

Schell, J. (2008). *The Art of Game Design: A Book of Lenses*. Burlington, MA: Morgan Kaufmann.

Schön, D. A. (1983). *The Reflective Practitioner: How Professionals Think in Action*. New York, NY: Basic Books.

Schreier, J. (2017). *Blood, Sweat, and Pixels: The Triumphant, Turbulent Stories Behind How Video Games Are Made*. New York: HarperCollins.

Schreier, J. (2021). *Press Reset: Ruin and Recovery in the Video Game Industry*. New York: Grand Central Publishing.

Schwaber, K. & Sutherland, J. (1995). "SCRUM Development Process". *Proceedings of the Workshop on Business Object Design and Implementation, OOPSLA 1995*. Austin, TX: Springer.

— (2020). *The Scrum Guide: The Definitive Guide to Scrum:The Rules of the Game*. `https://scrumguides.org/scrum-guide.html`.

Tokumitsu, M. (2015). *Do What You Love: And Other Lies About Success and Happiness*. New York: Regan Arts.

Tschang, F. T. (2007). "Balancing the Tensions Between Rationalization and Creativity in the Video Games Industry". *Organization Science*, 18:6, 989–1005.

Weststar, J. & Legault, M.-J. (2017a). "Videogame Developers among "Extreme" Workers: Are Death Marches Over?" *E-Journal of International and Comparative Labour Studies*, 6:3.

— (2017b). "Why Might a Videogame Developer Join a Union?" *Labor Studies Journal*, 42:4, 295–321.

Weststar, J. & Legault, M.-J. (2022). “The Video Game Industry: A Case Study of Structural Pressures and Labor Agency”. *Handbook of Labor in the Media and Creative Industries*. London: Routledge, 185–204.

Weststar, J., Legault, M.-J. & O’Meara, V. (2017). *Developer Satisfaction Survey 2017: Summary Report*. https://igda.org/resources-archive/developer\-satisfaction-survey-summary-report-2017/.