Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Informatica

A.a. 2024/2025

Sistema Multi-Agente per l'Ispezione Automatica di Cavi Danneggiati: Computer Vision e RAG per il Supporto Diagnostico nell'Industria 5.0

Relatore:

Andrea Bottino

Candidato:

Beatrice Carru

Abstract

L'Industria 5.0 ridefinisce il paradigma produttivo ponendo l'essere umano al centro della trasformazione digitale e promuovendo la sinergia tra competenze umane e sistemi intelligenti. Questa tesi presenta lo sviluppo e la validazione di un sistema multi-agente basato su Large Language Models per il supporto al controllo qualità e alla diagnostica di componenti industriali danneggiati, con focus specifico sull'ispezione automatica di cavi elettrici.

Il sistema proposto si fonda su un'architettura multi-agente orchestrata tramite LangGraph che integra tecniche di computer vision basate su EfficientSAM per la segmentazione semantica e Retrieval-Augmented Generation mediante LangChain per l'accesso intelligente alla documentazione tecnica. L'architettura comprende tre agenti specializzati coordinati: un Vision Agent per l'analisi automatica di immagini estratte da video industriali, un Diagnostic Agent che interroga una knowledge base tecnica, e un agente Coordinatore che gestisce il flusso conversazionale e il routing delle richieste.

La pipeline sviluppata in Python automatizza l'estrazione di frame rilevanti mediante OpenCV, identifica marker visivi per la localizzazione delle zone critiche, segmenta componenti danneggiati e genera report tecnici strutturati. Il modulo RAG garantisce supporto diagnostico contestualizzato attraverso similarity search vettoriale e sintesi intelligente delle evidenze documentali.

La validazione sperimentale condotta sul dataset MVTec Anomaly Detection ha confermato l'efficacia del sistema attraverso casi d'uso rappresentativi di scenari reali, dimostrando capacità operative nell'analisi automatizzata, nella diagnostica supportata da knowledge base e nella generazione di documentazione tecnica.

Il contributo principale risiede nella progettazione di un'architettura innovativa che combina LLM multi-agente con elaborazione multimodale per applicazioni industriali, offrendo una soluzione concreta per il lavoratore aumentato nell'ottica dell'Industria 5.0.

Indice

Abstract II

Elenco delle tabelle V

1 Introduzione 1
1.1 Contesto e Motivazioni 1
1.2 Obiettivi della ricerca 4
1.3 Struttura della tesi 5

2 Stato dell'Arte e Fondamenti Tecnologici 7
2.1 Il lavoratore aumentato nell'Industria 5.0 7
2.2 Introduzione agli LLM e agenti conversazionali 12
2.3 Architetture agentiche: dai modelli agli agenti autonomi 21

3 Progettazione del Sistema 33
3.1 Architettura multi-agente proposta 33
3.1.1 Coordinator Agent 34
3.1.2 Nodo WholeImage: gateway di sicurezza e controllo preliminare 35
3.1.3 CableVisionAgent 35
3.1.4 DiagnosticAgent 36
3.1.5 Flusso di controllo del grafo 37
3.2 Pipeline di elaborazione video e segmentazione 38
3.3 Integrazione RAG per supporto diagnostico 40

4 Implementazione 43
4.1 Architettura LangChain: chain, prompt e tools 43
4.2 LangGraph: orchestrazione a grafo e state management 50
4.3 Differenze tra LangChain e LangGraph 53
4.3.1 Struttura del flusso: pipeline lineare e grafo dinamico 54
4.3.2 Gestione dello stato: passaggio step by step e contesto condiviso 54
4.3.3 Controllo del flusso: statico e dinamico 55

4.4 Architettura del Coordinator Agent e meccanismo di reasoning . . . 57
4.4.1 Struttura e stati operativi del Coordinator 57
4.4.2 Meccanismo di reasoning: implementazione e vantaggi 58
4.4.3 Logica decisionale e routing intelligente 59
4.5 CableVisionAgent: implementazione della pipeline visiva 60
4.5.1 WholeImageNode: gateway di sicurezza e controllo preliminare 60
4.5.2 VisionNode: segmentazione con EfficientSAM 60
4.5.3 AnalysisNode: integrazione visione-linguaggio 62
4.5.4 Endpoint EfficientSAM: segmentazione avanzata 62
4.6 DiagnosticAgent: sistema di supporto decisionale avanzato 63
4.6.1 ImageDiagnosisNode: classificazione automatica delle anomalie 63
4.6.2 Integrazione RAG per supporto diagnostico 64
4.6.3 Report Generation e output strutturato 65
4.6.4 Endpoint Anomalib: rilevamento e classificazione di anomalie 65
4.7 Pattern architetturali e gestione degli stati 66

5 Risultati Sperimentali e Validazione 69
5.1 Metodologia e Setup Sperimentale 69
5.1.1 Approccio metodologico Sistemico 69
5.1.2 Ambiente di testing e dataset 70
5.1.3 Framework di Valutazione Multi-dimensionale 71
5.1.4 Protocollo di esecuzione 73
5.2 Categorizzazione dei Test Case 73
5.2.1 Test di Interazione Base e Gestione Conversazionale 74
5.2.2 Test di Analisi Visiva e Descrizione Componenti 75
5.2.3 Test Diagnostici Avanzati: Il Nucleo della Validazione Sistemica 77
5.2.4 Test di Gestione della Conoscenza Tecnica 80
5.2.5 Test di Gestione Errori ed Edge Cases 81
5.2.6 Test di Robustezza e Recupero Sistemico 82
5.2.7 Sintesi Comparativa delle Prestazioni Cross-Category 84
5.3 Analisi Qualitativa dei Risultati 85
5.3.1 Punti di Forza dell'Architettura Multi-Agente 85
5.3.2 Validazione dell'Architettura Multi-Agente 86
5.3.3 Limitazioni e Aree di Miglioramento 86

6 Conclusione 89
6.1 Contributo alla ricerca 89
6.2 Limitazioni e possibili miglioramenti 91
6.3 Prospettive di sviluppo industriale 93

Bibliografia 100

Elenco delle tabelle

2.1 Confronto tra LLM e Agenti IA 18

4.1 Confronto tra LangChain e LangGraph 55
4.2 Modelli di gestione dello stato 55
4.3 Confronto casi d'uso LangChain vs LangGraph 56
4.4 Matrice decisionale per la selezione del framework 57

5.1 Distribuzione dei Casi di Test per Categoria 74
5.2 Risultati dei Test di Interazione Base 75
5.3 Prestazioni dell'Analisi Visiva 77
5.4 Prestazioni dei Test Diagnostici Avanzati 80
5.5 Prestazioni di Accesso alla Conoscenza Tecnica 81
5.6 Prestazioni nella Gestione di Errori ed Edge Cases 82
5.7 Risultati dei Test di Robustezza 83
5.8 Analisi Comparativa dei Tempi di Risposta 84

Capitolo 1

Introduzione

1.1 Contesto e Motivazioni

Il mondo industriale sta attraversando negli ultimi anni una trasformazione profonda grazie all'affermazione del paradigma dell'Industria 5.0 [1]. Mentre la precedente rivoluzione dell'Industria 4.0 si era concentrata principalmente sull'automazione e la digitalizzazione dei processi produttivi, questa nuova fase introduce un approccio radicalmente diverso, che colloca l'essere umano al centro del processo produttivo e promuove una collaborazione intelligente tra lavoratori e tecnologie avanzate.

La Commissione Europea ha identificato tre pilastri fondamentali[1] su cui si fonda l'Industria 5.0. Il primo è la centralità dell'essere umano, un principio che riconosce come gli esseri umani rimangano insostituibili per la loro capacità di adattamento, pensiero critico e problem solving. L'obiettivo non è quindi quello di sostituire i lavoratori con le macchine, ma piuttosto di potenziarne le capacità attraverso strumenti digitali che rendano il lavoro più sicuro, efficace e meno gravoso. Le soluzioni di tipo Augmented & Connected Worker rappresentano un esempio concreto di questo approccio [2], consentendo agli operatori di accedere immediatamente a procedure digitali, raccomandazioni basate su intelligenza artificiale e istruzioni visualizzate in realtà aumentata.

Il secondo pilastro riguarda la sostenibilità delle operazioni produttive. L'industria moderna deve essere in grado di rendere le proprie operazioni più agili e adattive, così da affrontare efficacemente i cambiamenti della forza lavoro, le interruzioni impreviste o le nuove esigenze produttive che emergono continuamente. Le piattaforme Connected Worker facilitano questo processo permettendo un onboarding più rapido dei nuovi assunti, riducendo significativamente gli errori e offrendo formazione digitale direttamente sul posto di lavoro, diminuendo così la dipendenza dal personale senior e preservando il know-how aziendale.

Il terzo pilastro è la resilienza del sistema produttivo. Non si tratta solamente di

migliorare la produttività, ma di rendere la produzione più sostenibile e responsabile nel suo complesso. Le soluzioni intelligenti contribuiscono a questo obiettivo riducendo sprechi, scarti e tempi di inattività grazie a flussi di lavoro ottimizzati. Migliorano inoltre il benessere degli operatori attraverso il monitoraggio digitale e l'attenzione all'ergonomia, contribuendo così a creare ambienti di lavoro più sicuri e a prolungare le carriere lavorative.

In questo contesto evolutivo, le aziende manifatturiere e i reparti di manutenzione si trovano sempre più spesso di fronte alla necessità di raccogliere e analizzare in tempo reale informazioni provenienti da macchinari e linee produttive, documentare rapidamente interventi tecnici, anomalie o guasti, e trasferire conoscenze tecniche in modo strutturato e facilmente riutilizzabile.

Le operazioni di controllo qualità e diagnostica rappresentano una fase critica del processo produttivo [2], richiedendo competenze tecniche specialistiche, esperienza pratica e grande attenzione ai dettagli. Gli operatori si trovano spesso nella necessità di consultare manuali tecnici complessi, seguire checklist articolate e prendere decisioni rapide basate su osservazioni visive o acustiche. Questa complessità intrinseca comporta diverse sfide significative che impattano sull'efficienza operativa.

La documentazione tecnica presenta innanzitutto un problema di accessibilità: i manuali sono spesso voluminosi e difficili da consultare in tempo reale durante le operazioni. Il trasferimento di conoscenza costituisce un'altra criticità fondamentale, poiché la perdita di operatori esperti comporta il rischio concreto di disperdere un know-how prezioso che difficilmente può essere completamente codificato. Gli errori umani rappresentano una minaccia costante, dato che la complessità delle procedure può facilmente portare a dimenticanze o valutazioni errate. I tempi di formazione necessari per rendere autonomi i nuovi operatori risultano spesso lunghi e onerosi. Infine, l'accesso alle informazioni durante le operazioni può rallentare significativamente il flusso di lavoro, creando inefficienze nel processo produttivo.

Uno dei problemi più evidenti e urgenti riguarda la documentazione visiva degli interventi. Gli operatori devono frequentemente interrompere le loro attività per annotare manualmente difetti o interventi, spezzando il ritmo operativo. I contenuti raccolti, che comprendono video e immagini, vengono raramente organizzati o descritti adeguatamente per un uso futuro, riducendone drasticamente il valore informativo. La creazione di report dettagliati richiede tempo prezioso e competenze specifiche, con il rischio concreto di rallentare l'intero processo produttivo.

Per rispondere a queste criticità emergenti, il progetto descritto in questa tesi nasce con l'obiettivo ambizioso di automatizzare l'analisi e la descrizione dei contenuti visivi, integrandoli in un sistema multi-agente che supporta concretamente l'operatore in tempo reale. Il sistema è in grado di identificare automaticamente la zona di interesse nei video grazie a marker visivi rilevati tramite OpenCV, segmentare i componenti meccanici utilizzando un modello efficiente come EfficientSAM, e generare descrizioni testuali e documentazione tecnica immediatamente utile per

interventi futuri.

Questa soluzione si inserisce perfettamente nella visione dell'Industria 5.0, che mira non soltanto all'automazione fine a se stessa, ma al potenziamento effettivo del ruolo dell'essere umano. L'obiettivo è migliorare la qualità del lavoro, incrementare la sicurezza operativa, garantire la tracciabilità degli interventi e preservare il know-how tecnico aziendale. In questo modo, gli operatori possono finalmente concentrarsi sulle decisioni strategiche e sulle attività a maggior valore aggiunto, mentre le parti più ripetitive e time-consuming, come la descrizione e la catalogazione dei contenuti, vengono efficacemente delegate all'intelligenza artificiale.

Figura 1.1: I tre pilastri dell'Industria 5.0
. Fonte: [1]

1.2 Obiettivi della ricerca

L'obiettivo principale di questa ricerca consiste nel progettare e sviluppare un sistema multi-agente capace di supportare gli operatori industriali nell'analisi, descrizione e documentazione di componenti meccaniche a partire da contenuti video acquisiti direttamente sul campo. Questo obiettivo generale si articola in diverse direzioni specifiche che insieme costituiscono un percorso di ricerca coerente e integrato. Il primo ambito di lavoro riguarda la realizzazione di una pipeline efficace per l'elaborazione di video e immagini. Quando un operatore registra un video di un componente meccanico o di un sistema di interesse, il sistema deve essere in grado di estrarre automaticamente i frame più rilevanti, selezionando quelli che contengono le informazioni visive più significative. Parallelamente, diventa fondamentale identificare eventuali marker visivi presenti nelle riprese, come frecce, bollini o etichette che l'operatore potrebbe aver inserito intenzionalmente per segnalare aree critiche. Attraverso algoritmi di computer vision basati su OpenCV, il sistema può localizzare con precisione la zona di interesse da sottoporre ad analisi più approfondita. Una volta individuata l'area rilevante, diventa necessario implementare una segmentazione avanzata dei componenti meccanici presenti nell'immagine. Questo secondo obiettivo specifico prevede l'utilizzo di modelli deep learning preaddestrati, in particolare EfficientSAM, per segmentare con elevata precisione il componente target, isolandolo dallo sfondo e da altri elementi presenti nella scena. Il risultato di questa operazione deve tradursi in un output visuale chiaro e immediatamente comprensibile, che può assumere la forma di una maschera di segmentazione, di contorni evidenziati o di un'immagine processata che metta in risalto il componente analizzato, fornendo così materiale prezioso per la documentazione tecnica. Il terzo filone di lavoro si concentra sulla generazione automatica di descrizioni testuali e metadati a partire dalle informazioni visive estratte. L'intelligenza artificiale deve essere capace di tradurre ciò che "vede" nell'immagine in descrizioni strutturate e comprensibili, che aiutino concretamente l'operatore a documentare difetti, anomalie o caratteristiche specifiche del componente osservato. Queste informazioni testuali devono poi essere collegate in modo organico a un database o a un sistema di knowledge management, garantendo che la conoscenza acquisita rimanga accessibile e riutilizzabile nel tempo. L'integrazione di tutte queste funzionalità in un sistema multi-agente rappresenta il quarto obiettivo specifico del lavoro. L'architettura proposta prevede che diversi agenti software collaborino in modo coordinato per gestire il flusso informativo complesso. Un primo nodo si occupa dell'estrazione e dell'analisi dei video, oltre che della segmentazione delle immagini. Un secondo nodo ha la responsabilità di generare il testo descrittivo, trasformando l'informazione visuale in contenuto linguistico strutturato. Un terzo nodo gestisce il recupero delle informazioni dalla documentazione tecnica esistente, permettendo al sistema di contestualizzare le osservazioni attuali rispetto alla conoscenza pregressa.

Il coordinamento efficace dei flussi informativi tra questi agenti diventa cruciale per garantire un processo coerente, fluido e completamente automatizzato. Infine, l'ultimo obiettivo consiste nel contribuire concretamente alla trasformazione digitale in ottica Industria 5.0. Attraverso questo sistema, si intende dimostrare come una soluzione intelligente possa effettivamente aumentare le capacità dell'operatore umano, non sostituendolo ma potenziandone le competenze. Il sistema migliora la qualità della documentazione tecnica prodotta, riduce drasticamente il tempo che l'operatore deve dedicare ad attività ripetitive e a basso valore aggiunto, e incrementa significativamente la tracciabilità delle informazioni. Inoltre, promuove la standardizzazione delle informazioni tecniche raccolte sul campo, creando una base di conoscenza omogenea e facilmente interrogabile che può supportare decisioni future e facilitare la formazione di nuovi operatori.

1.3 Struttura della tesi

La presente tesi si articola in sei capitoli, organizzati per accompagnare il lettore attraverso un percorso che parte dal contesto teorico e tecnologico per arrivare alla realizzazione pratica del sistema proposto e alla sua valutazione sperimentale. Ogni capitolo è stato concepito per costruire progressivamente la comprensione del lavoro svolto, partendo dalle fondamenta concettuali fino ai risultati concreti ottenuti.

- **Capitolo 1 – Introduzione** introduce il contesto del progetto e le motivazioni che hanno portato alla definizione del tema di ricerca, chiarendo gli obiettivi principali e illustrando la struttura complessiva del lavoro. Questo capitolo fornisce al lettore la cornice di riferimento necessaria per comprendere le scelte progettuali e metodologiche adottate nei capitoli successivi.
- **Capitolo 2 – Stato dell'Arte e Fondamenti Tecnologici** descrive il panorama tecnologico in cui si colloca il progetto. Introduce il concetto di lavoratore aumentato nell'Industria 5.0, approfondisce il ruolo degli LLM (Large Language Models) e degli agenti conversazionali, trattando tecniche fondamentali come RAG e Tool Calling e le loro applicazioni industriali. Conclude con un'analisi delle architetture agentiche che permettono la transizione dai modelli linguistici agli agenti autonomi, fornendo le basi teoriche necessarie per comprendere le soluzioni adottate.
- **Capitolo 3 – Progettazione del Sistema** presenta l'architettura multi-agente ideata, specificando i casi d'uso identificati e i requisiti funzionali e non funzionali che hanno guidato le scelte progettuali. Descrive in dettaglio la pipeline di elaborazione video, che combina algoritmi di computer vision basati su OpenCV e segmentazione avanzata tramite EfficientSAM, e illustra

l'integrazione di un modulo RAG (Retrieval-Augmented Generation) per fornire supporto diagnostico contestualizzato durante l'analisi dei componenti.

- **Capitolo 4 – Implementazione** dettaglia come il sistema è stato concretamente realizzato attraverso le architetture LangChain e LangGraph. Analizza l'implementazione del Coordinator Agent con il suo meccanismo di reasoning, del CableVisionAgent per la pipeline visiva (dall'analisi preliminare alla segmentazione con EfficientSAM), e del DiagnosticAgent per il supporto decisionale avanzato. Descrive inoltre gli endpoint specializzati per la segmentazione e il rilevamento anomalie, concludendo con i pattern architetturali e la gestione degli stati nel sistema multi-agente.

- **Capitolo 5 – Risultati e Valutazione** riporta i risultati ottenuti nei test condotti su casi d'uso reali, valutando le performance del sistema attraverso diverse metriche: la qualità delle risposte generate dagli agenti, i tempi di elaborazione delle richieste, e la scalabilità della soluzione. Questo capitolo fornisce una valutazione critica dell'efficacia del sistema proposto in scenari operativi concreti.

- **Capitolo 6 – Conclusioni e Sviluppi Futuri** riassume i principali contributi della ricerca, evidenziando gli aspetti innovativi e i risultati significativi raggiunti. Discute con onestà le limitazioni individuate durante il lavoro e propone possibili direzioni per l'evoluzione futura del sistema, con una prospettiva di applicazione in contesti industriali sempre più complessi e articolati.

Capitolo 2

Stato dell'Arte e Fondamenti Tecnologici

2.1 Il lavoratore aumentato nell'Industria 5.0

In questo contesto emerge il concetto di "lavoratore aumentato" (augmented worker) [2], che rappresenta un operatore industriale le cui capacità cognitive e operative vengono potenziate attraverso l'integrazione di tecnologie avanzate come l'intelligenza artificiale, la realtà aumentata, i sistemi multi-agente [3] e i dispositivi wearables. Il lavoratore aumentato non è semplicemente un utilizzatore di tecnologia, ma diventa parte di un ecosistema collaborativo uomo-macchina che ottimizza le performance operative mantenendo il controllo e la creatività umana.

Le caratteristiche principali del lavoratore aumentato includono:

- **Potenziamento cognitivo**: Accesso immediato a informazioni contestuali, supporto decisionale basato su AI e assistenza nella risoluzione di problemi complessi
- **Interfacce intuitive**: Utilizzo di tecnologie come realtà aumentata (AR), realtà virtuale (VR) e interfacce vocali per un'interazione naturale con i sistemi
- **Collaborazione con robot**: Capacità di lavorare in sicurezza ed efficienza con robot collaborativi (cobots) in spazi condivisi
- **Apprendimento continuo**: Adattamento dinamico alle nuove tecnologie e processi attraverso sistemi di formazione personalizzata
- **Autonomia decisionale**: Mantenimento del controllo finale sui processi critici, con la tecnologia che fornisce supporto e non sostituzione

Tecnologie abilitanti per il lavoratore aumentato

Il paradigma del lavoratore aumentato si basa su un ecosistema tecnologico sofisticato che trasforma radicalmente il modo in cui gli operatori interagiscono con l'ambiente produttivo. Al centro di questa rivoluzione troviamo l'intelligenza artificiale e il machine learning, tecnologie che fungono da cervello digitale dell'intero sistema. Questi strumenti non si limitano a processare dati, ma forniscono capacità predittive straordinarie, analizzano automaticamente enormi quantità di informazioni e offrono supporto decisionale in tempo reale. Ciò che rende questi sistemi particolarmente potenti è la loro capacità di apprendimento continuo: gli algoritmi di machine learning osservano attentamente i comportamenti degli operatori più esperti, assorbendo le loro conoscenze e trasformandole in raccomandazioni personalizzate per tutti gli altri lavoratori.

Parallelamente, la realtà aumentata e virtuale sta ridefinendo completamente il concetto di interfaccia uomo-macchina. La realtà aumentata agisce come un ponte invisibile tra il mondo fisico e quello digitale, sovrapponendo informazioni cruciali direttamente nel campo visivo degli operatori. Ad esempio, un tecnico che, guardando un macchinario complesso, vede automaticamente apparire istruzioni dettagliate, schemi tecnici aggiornati e dati operativi in tempo reale, il tutto senza mai dover distogliere lo sguardo dal proprio lavoro. Dall'altro canto la realtà virtuale apre scenari formativi completamente nuovi, permettendo agli operatori di immergersi in simulazioni ultra-realistiche dove possono sperimentare e apprendere gestendo situazioni complesse o pericolose in totale sicurezza.

La robotica collaborativa [4, 5] rappresenta forse l'evoluzione più tangibile di questo nuovo paradigma. I robot collaborativi, che vengono denominati "cobots", incarnano una filosofia completamente diversa rispetto alla robotica tradizionale. Invece di sostituire gli operatori umani, questi sistemi sono progettati per diventare veri e propri colleghi di lavoro, condividendo spazi, compiti e responsabilità. Grazie a sensori avanzatissimi e algoritmi di sicurezza sofisticati, i cobots riescono a interpretare i movimenti umani, anticipare le intenzioni e adattarsi dinamicamente alle esigenze del momento, creando un'interazione che appare quasi naturale.

L'Internet delle Cose e la sensoristica distribuita costituiscono il sistema nervoso di questo ecosistema tecnologico. Una rete capillare di sensori intelligenti monitora costantemente ogni aspetto dell'ambiente produttivo, raccogliendo dati preziosi che vengono immediatamente elaborati dai sistemi di intelligenza artificiale. Questa architettura permette di fornire agli operatori un feedback istantaneo e un supporto contestuale precisissimo, trasformando ogni workstation in un ambiente intelligente e reattivo.

Infine, le interfacce conversazionali stanno eliminando una delle ultime barriere tra l'uomo e la tecnologia: la complessità dell'interazione. Grazie ai progressi nell'elaborazione del linguaggio naturale, gli operatori possono comunicare con

i sistemi industriali attraverso semplici comandi vocali o conversazioni fluide e naturali. Questa evoluzione non solo semplifica drasticamente l'utilizzo della tecnologia, ma democratizza l'accesso alle funzionalità avanzate, permettendo anche agli operatori meno esperti dal punto di vista tecnologico di beneficiare pienamente delle potenzialità del sistema.

Vantaggi del paradigma del lavoratore aumentato

L'implementazione del concetto di lavoratore aumentato nell'Industria 5.0 genera un circolo virtuoso di benefici che si estende a tutti i livelli dell'organizzazione industriale, creando valore tanto per i singoli operatori quanto per le aziende nel loro complesso.

Dal punto di vista dei lavoratori, la trasformazione più evidente riguarda la drastica riduzione del carico cognitivo. I sistemi di supporto intelligente si fanno carico delle attività mentali più ripetitive e onerose, fungendo da assistenti digitali che filtrano, organizzano e presentano le informazioni in modo ottimale. Questo alleggerimento permette agli operatori di liberare risorse mentali preziose, che possono essere così concentrate su attività genuinamente creative e a valore aggiunto, trasformando il loro ruolo da esecutori di procedure standardizzate a problem solver e innovatori.

Parallelamente, si assiste a un miglioramento sostanziale delle condizioni di sicurezza sul posto di lavoro. Le tecnologie di monitoraggio continuo e i sistemi di assistenza predittiva agiscono come guardian angel digitali, identificando potenziali rischi prima che si concretizzino in incidenti. Questa protezione proattiva non solo riduce drammaticamente i pericoli fisici, ma crea anche un ambiente lavorativo più sereno e stimolante, dove gli operatori possono concentrarsi sulle proprie mansioni senza la costante preoccupazione per la sicurezza personale.

Un aspetto particolarmente significativo è l'impatto sulla crescita professionale dei lavoratori. Ogni giornata di lavoro viene trasformata in un'opportunità di apprendimento e sviluppo, grazie all'interazione quotidiana con tecnologie. Gli operatori acquisiscono competenze digitali sempre più richieste dal mercato, aumentando non solo la loro spendibilità sul mercato ma anche l'attrattività stessa del lavoro industriale, spesso percepito dalle nuove generazioni come obsoleto o poco stimolante. Invece di costringere gli operatori ad adattarsi rigidamente a processi standardizzati, le tecnologie si modellano sulle preferenze, le capacità e lo stile di lavoro di ciascun individuo, creando un'esperienza lavorativa personalizzata.

Per le aziende, i benefici si traducono immediatamente in vantaggi competitivi tangibili. Il risultato più evidente è sicuramente l'aumento della produttività [6] dove la collaborazione sinergica tra intelligenza umana e capacità computazionali ottimizza ogni aspetto dei processi produttivi e viene mantenuta intatta la flessibilità e capacità di adattamento che possono garantire solo gli esseri umani. È una

combinazione che permette di raggiungere livelli di efficienza che prima erano impensabili senza sacrificare la qualità o la capacità di rispondere rapidamente ai cambiamenti del mercato.

La diminuzione degli errori operativi [7] costituisce un pilastro fondamentale del valore aziendale. I sistemi di supporto agiscono come reti di sicurezza intelligenti, intercettando potenziali errori prima che si propaghino lungo la catena produttiva. Questa prevenzione evita costi diretti legati a scarti, rilavorazioni e fermi macchina, e di conseguenza assicura la reputazione aziendale e la soddisfazione del cliente finale.

Risulta essere strategico il ruolo delle tecnologie nella conservazione e condivisione del know-how aziendale. In un'epoca in cui l'invecchiamento della forza lavoro rischia di portare alla perdita di competenze critiche accumulate in decenni di esperienza, i sistemi di lavoratore aumentato fungono da custodi digitali della conoscenza. Catturano, formalizzano e rendono accessibile il sapere degli operatori più esperti, trasformando competenze individuali in patrimonio collettivo dell'organizzazione.

Infine, l'adattabilità dei sistemi rappresenta un asset fondamentale in un mercato sempre più dinamico e imprevedibile. La capacità di riconfigurare rapidamente i processi per nuovi prodotti, variazioni della domanda o cambiamenti normativi diventa un fattore competitivo decisivo, permettendo alle aziende di cogliere opportunità di mercato con una velocità e un'agilità che i competitor tradizionali non possono eguagliare.

Sfide nell'implementazione

Sebbene ci siano molti benefici, l'adozione del paradigma del lavoratore aumentato presenta un complesso panorama di sfide che richiedono un approccio metodico e strategico per poterle superare efficientemente. Questi ostacoli sono articolati su due dimensioni: quella tecnica, legata alla complessità intrinseca delle tecnologie coinvolte, e quella organizzativa, che riguarda l'impatto umano e culturale della trasformazione.

Sul fronte tecnico, la sfida più critica risulta essere l'integrazione dei sistemi. Il lavoratore aumentato non può infatti basarsi su una singola tecnologia, ma richiede la perfetta orchestrazione di componenti eterogenei: sistemi di intelligenza artificiale, dispositivi di realtà aumentata, robot collaborativi, sensori IoT e interfacce conversazionali [8] devono dialogare agevolmente tra loro, creando un ecosistema tecnologico coerente. Questa integrazione va oltre la semplice interoperabilità tecnica e richiede la creazione di architetture software e hardware capaci di gestire flussi di dati complessi che riescono a mantenere prestazioni ottimali.

L'affidabilità costituisce un'altra pietra angolare del successo di questi sistemi. In ambienti industriali dove un malfunzionamento può comportare conseguenze

economiche devastanti o rischi per la sicurezza, i sistemi del lavoratore aumentato devono garantire standard di affidabilità estremi. Questo significa progettare architetture ridondanti, implementare meccanismi sofisticati di fail-safe e assicurare che ogni componente del sistema possa operare in condizioni ambientali spesso ostili, caratterizzate da vibrazioni, temperature estreme, polvere e interferenze elettromagnetiche.

La gestione della latenza rappresenta una sfida particolarmente insidiosa nelle applicazioni industriali in real time. Quando un operatore interagisce con sistemi di realtà aumentata o riceve supporto decisionale dall'intelligenza artificiale, anche ritardi dell'ordine di millisecondi possono compromettere l'efficacia dell'intervento o, nei casi peggiori, generare situazioni di pericolo. Questo richiede lo sviluppo di architetture di elaborazione distribuite, l'ottimizzazione degli algoritmi e spesso l'implementazione di soluzioni di edge computing per portare la capacità computazionale il più vicino possibile al punto di utilizzo.

La sicurezza informatica assume dimensioni critiche in questo contesto, poiché la digitalizzazione massiva dei processi industriali espone le aziende a nuove categorie di rischi cyber. La protezione non riguarda solo i dati sensibili, ma si estende ai sistemi di controllo industriale, dove un attacco informatico potrebbe avere conseguenze fisiche dirette sulla produzione e sulla sicurezza degli operatori. Questo richiede l'implementazione di strategie di cybersecurity multi-livello, che vanno dalla crittografia dei dati alla segmentazione delle reti, dalla gestione delle identità digitali al monitoraggio continuo delle minacce.

Parallelamente, le sfide organizzative si rivelano essere ancora più complesse da affrontare rispetto a quelle puramente tecniche. La gestione del cambiamento diventa un elemento cruciale per il successo dell'implementazione. I lavoratori, spesso con anni o decenni di esperienza in modalità operative consolidate, devono essere accompagnati in un percorso di trasformazione che non riguarda solo l'apprendimento di nuovi strumenti, ma spesso un vero e proprio cambio di mentalità nel modo di concepire il proprio ruolo lavorativo.

Lo sviluppo di programmi formativi adeguati rappresenta una sfida multidimensionale. Non si tratta semplicemente di insegnare l'uso di nuove tecnologie, ma di creare percorsi di apprendimento che tengano conto delle diverse competenze di partenza, dei diversi stili di apprendimento e della necessità di mantenere operativa la produzione durante la fase di transizione. Questo richiede spesso l'implementazione di metodologie formative innovative, che vanno dalla gamification all'apprendimento immersivo in realtà virtuale.

Dal lato umano la sfida più delicata risulta essere l'acceptance tecnologica. Costruire un rapporto di fiducia e oltrepassare la resistenza al cambiamento richiede un approccio empatico e graduale. Gli operatori devono avere la certezza che le nuove tecnologie non rappresentano una minaccia per il loro posto di lavoro, ma un'opportunità di crescita e miglioramento delle loro condizioni lavorative. Questo

processo di costruzione della fiducia richiede trasparenza nella comunicazione, coinvolgimento attivo dei lavoratori nelle fasi di progettazione e implementazione, e la dimostrazione tangibile dei benefici che le tecnologie possono portare alla loro esperienza lavorativa quotidiana.

2.2 Introduzione agli LLM e agenti conversazionali

Definizione e principi fondamentali degli LLM

Negli ultimi anni, i Large Language Models (LLM) hanno rivoluzionato il campo dell'intelligenza artificiale applicata al linguaggio naturale (Natural Language Processing, NLP). Questi modelli rappresentano una categoria di reti neurali artificiali progettate per comprendere, generare e manipolare il linguaggio umano con un livello di raffinatezza senza precedenti.

Un LLM è un modello linguistico addestrato con apprendimento automatico auto-supervisionato su una vasta quantità di testo, progettato per compiti di elaborazione del linguaggio naturale, specialmente per la generazione linguistica. La caratteristica distintiva degli LLM è la loro scala: questi modelli sono addestrati su enormi quantità di dati testuali (tipicamente terabyte di testo) e possiedono miliardi o addirittura trilioni di parametri.

Il principio fondamentale su cui si basano gli LLM è quello della predizione del token successivo: dato un contesto testuale, il modello calcola la probabilità di ogni possibile parola (o token) che potrebbe seguire nella sequenza. Tuttavia, questa apparente semplicità nasconde una complessità computazionale straordinaria che emerge dalla combinazione di:

- **Scala massiva**: Modelli con miliardi di parametri addestrati su dataset contenenti trilioni di token [9]
- **Architettura Transformer**: Utilizzo del meccanismo di attenzione per catturare dipendenze a lungo raggio nel testo
- **Emergent abilities**: Capacità cognitive che emergono spontaneamente oltre una certa soglia di scala
- **Transfer learning**: Capacità di applicare conoscenze apprese durante il pretraining a nuovi compiti

Principali modelli LLM del panorama attuale

Il settore dei Large Language Models presenta attualmente una configurazione oligopolistica caratterizzata da intense dinamiche competitive tra un numero limitato

di attori tecnologici dominanti. Tale struttura di mercato riflette le significative barriere all'ingresso rappresentate dai costi computazionali e dalle competenze specialistiche richieste per lo sviluppo di questi sistemi.

I modelli Generative Pre-trained Transformer di OpenAI hanno storicamente definito gli standard de facto del settore, consolidando una leadership tecnologica che permane nonostante la crescente pressione competitiva. L'iterazione GPT-4 [10] ha rappresentato un salto qualitativo significativo, introducendo un ampliamento della finestra contestuale fino a 32.768 token – una capacità che corrisponde all'elaborazione simultanea di circa cinquanta pagine di documentazione tecnica.

L'analisi delle prestazioni evidenzia tre aree di eccellenza particolarmente rilevanti: primo, la generazione di contenuti creativi caratterizzata da originalità e coerenza stilistica; secondo, la capacità di comprensione contestuale che permette l'interpretazione di semantiche complesse e sfumate; terzo, il mantenimento della coerenza logica e narrativa su produzioni testuali estese. Queste caratteristiche si traducono in applicazioni pratiche che spaziano dalla redazione di contenuti professionali alla progettazione di interfacce conversazionali avanzate.

Il progetto LLaMA di Meta [11] rappresenta un approccio sostanzialmente diverso, orientato verso la democratizzazione dell'accesso alle tecnologie LLM attraverso una strategia di open-source parziale. La timeline di sviluppo, iniziata nel febbraio 2023 e proseguita con rilasci cadenzati fino a LLaMA 4 nell'aprile 2025, riflette una pianificazione strategica di lungo termine.

La caratteristica distintiva dell'architettura LLaMA risiede nella sua modularità scalare. La gamma di configurazioni disponibili – da implementazioni leggere da 1 miliardo di parametri fino a versioni enterprise da 2 trilioni – permette un deployment ottimizzato in funzione dei vincoli operativi specifici. Particolarmente significative risultano le versioni 3.1 (luglio 2024), che hanno introdotto configurazioni da 405 e 70 miliardi di parametri, e la successiva iterazione 3.2 (settembre 2024), che ha privilegiato l'efficienza computazionale con implementazioni da 11 e 90 miliardi di parametri.

L'approccio di Anthropic si distingue per una metodologia di sviluppo caratterizzata da maggiore prudenza tecnica ma rigorosità metodologica superiore. Il framework "Constitutional AI" [12] ha prodotto risultati empirici notevoli: la serie Claude 3 ha infatti conseguito, nel Q1 2024, prestazioni superiori a GPT-4 nelle principali metriche standardizzate, configurandosi come il primo caso documentato di superamento prestazionale del benchmark di riferimento di OpenAI.

L'evoluzione successiva ha seguito una logica di specializzazione verticale: Claude 3.7 Sonnet è stato ottimizzato specificamente per workflow di sviluppo software, mentre le implementazioni Claude 4 Opus e Claude 4 Sonnet sono state progettate per soddisfare requisiti enterprise che privilegiano affidabilità operativa e consistenza prestazionale su volumi di elaborazione sostenuti.

Il panorama competitivo include ulteriori attori strategici che stanno diversificando l'offerta tecnologica. Google segue una strategia dual-track attraverso i modelli Gemini, focalizzati sull'integrazione multimodale, e la serie PaLM, orientata verso architetture di scaling computazionale estremo. In contemporanea, l'ecosistema open-source sta riducendo sistematicamente le barriere tecnologiche ed economiche, facilitando l'accesso a implementazioni LLM per organizzazioni con limitazioni di budget e competenze.

Questa pluralità di approcci tecnologici sta generando un mercato caratterizzato da innovazione accelerata e specializzazione crescente, creando le precondizioni per una penetrazione capillare delle tecnologie LLM attraverso segmenti applicativi eterogenei, dal manifatturiero avanzato alla ricerca accademica, fino ai servizi consumer di massa.

Tecniche di addestramento avanzate

Le tecniche di specializzazione post-training hanno assunto un ruolo centrale nello sviluppo dei modelli di linguaggio di ultima generazione. Il fine-tuning costituisce la metodologia standard più diffusa per l'adattamento di modelli pre-addestrati verso domini applicativi specifici. Questo processo sfrutta l'apprendimento transfer, adoperando dataset di dimensioni ridotte ma molto curati per specializzare le capacità generali acquisite durante la fase di pretraining. L'efficacia di questa tecnica risiede nella possibilità di ottenere prestazioni domain-specific elevate riducendo i costi computazionali rispetto all'addestramento ex-novo.

Negli ultimi anni si è diffuso un paradigma ancora più innovativo, il Reinforcement Learning from Human Feedback [13, 14], che introduce in maniera sistematica il giudizio umano all'interno del processo di ottimizzazione del modello. La sua architettura si sviluppa in tre fasi principali: si parte dalla raccolta di preferenze espresse da persone reali, chiamate a valutare quale tra due o più risposte generate sia più adeguata; queste valutazioni vengono quindi utilizzate per addestrare un modello di ricompensa in grado di stimare automaticamente la "qualità" delle risposte; infine, il modello di linguaggio viene ulteriormente ottimizzato tramite tecniche di reinforcement learning [15, 16], in particolare con l'algoritmo Proximal Policy Optimization, così da orientare la generazione dei contenuti verso le preferenze apprese.

L'impatto del RLHF risulta evidente su più livelli. Da un lato, contribuisce a rendere i sistemi più allineati ai valori e alle aspettative umane, riducendo la probabilità di risposte inappropriate o dannose. Dall'altro, migliora la rilevanza e l'utilità pratica delle risposte in contesti applicativi concreti, aumentando l'affidabilità del modello come strumento di supporto. Questa combinazione di fine-tuning mirato e feedback umano rappresenta quindi una delle direzioni più promettenti

per coniugare le potenzialità delle grandi architetture neurali con le esigenze reali di chi le utilizza.

Capacità emergenti degli LLM

L'evoluzione dei Large Language Models ha rivelato un fenomeno particolarmente interessante: l'emergere di capacità cognitive che non erano state esplicitamente programmate durante la fase di addestramento. Questo comportamento emergente rappresenta uno degli aspetti più affascinanti e al contempo dibattuti nel campo dell'intelligenza artificiale, poiché suggerisce che la semplice scalabilità in termini di parametri e dati di training possa generare salti qualitativi imprevisti nelle performance cognitive.

Una delle manifestazioni più evidenti di queste capacità emergenti è lo sviluppo spontaneo di strategie di ragionamento articolato. I modelli di grande scala hanno dimostrato la capacità di affrontare problemi complessi attraverso la decomposizione in passaggi intermedi, un processo che nella letteratura viene definito "chain-of-thought reasoning" [17]. Questa competenza si manifesta quando il modello, anziché fornire direttamente una risposta, esplicita il percorso logico che lo conduce alla soluzione, mostrando passaggi di ragionamento che ricordano i processi cognitivi umani. Il fenomeno assume particolare rilevanza nel contesto industriale, dove la capacità di tracciare e giustificare le decisioni diagnostiche rappresenta un requisito fondamentale per l'affidabilità operativa.

Parallelamente al ragionamento complesso, i modelli hanno sviluppato notevoli capacità di apprendimento contestuale, manifestate attraverso competenze di few-shot e zero-shot learning. Questa flessibilità permette ai sistemi di adattarsi a nuovi compiti presentando loro solamente alcuni esempi specifici o addirittura basandosi unicamente sulla comprensione delle istruzioni fornite in linguaggio naturale. Nel dominio diagnostico industriale, questa caratteristica si traduce nella possibilità di adattare il sistema a nuove tipologie di componenti o procedure senza necessità di riaddestramento completo, riducendo significativamente i tempi e i costi di implementazione.

La comprensione semantica rappresenta forse l'advancement più significativo rispetto ai sistemi di AI precedenti. I modelli moderni dimostrano una comprensione del linguaggio che va oltre la semplice corrispondenza sintattica, catturando sfumature semantiche, riferimenti impliciti e relazioni complesse tra concetti. Questa competenza si rivela cruciale in contesti tecnici specializzati, dove la terminologia può essere ambigua e fortemente dipendente dal contesto. La capacità di disambiguare termini tecnici, interpretare descrizioni incomplete e inferire informazioni mancanti rappresenta un valore aggiunto sostanziale per l'assistenza agli operatori industriali.

Un ulteriore aspetto emergente di notevole importanza è la capacità di mantenere coerenza logica e narrativa anche in interazioni prolungate. Questa caratteristica garantisce continuità nell'assistenza tecnica, permettendo di sviluppare sessioni diagnostiche complesse che si estendono su multiple interazioni senza perdita di informazioni o contraddizioni. Nel framework del lavoratore aumentato, tale competenza risulta essenziale per costruire un dialogo tecnico efficace e affidabile.

I modelli avanzati hanno inoltre dimostrato capacità di trasferimento inter-dominio, applicando principi e pattern appresi in un contesto a situazioni analoghe in domini differenti. Per applicazioni industriali, questo significa che un sistema addestrato su una tipologia di componenti può estendere le proprie competenze diagnostiche a componenti simili in settori diversi, massimizzando il ritorno sull'investimento tecnologico.

Tuttavia, è importante riconoscere che le capacità emergenti, pur rappresentando un progresso significativo, presentano anche alcune limitazioni intrinseche. La natura probabilistica della generazione può occasionalmente produrre output plausibili ma fattualmente incorretti, un fenomeno noto come "allucinazione". In contesti industriali critici, questo richiede l'implementazione di meccanismi di validazione e verifica incrociata. Inoltre, l'interpretabilità di queste capacità emergenti rimane una sfida aperta, rendendo necessario un approccio prudente nella progettazione di sistemi mission-critical.

Dagli LLM agli agenti conversazionali

L'evoluzione dai Large Language Models agli agenti conversazionali rappresenta un salto concettuale fondamentale per comprendere come questi sistemi possano essere applicati efficacemente in contesti industriali. Mentre un LLM è essenzialmente un modello statistico che genera testo, un agente conversazionale è un sistema completo capace di ragionare, pianificare azioni e interagire con l'ambiente circostante.

Un Large Language Model, nella sua forma base, riceve un prompt testuale e genera una risposta. Il modello non ha memoria persistente tra diverse conversazioni, non può accedere a informazioni esterne al suo training, e non può eseguire azioni nel mondo reale. Si tratta di un sistema puramente reattivo che opera all'interno dei confini del testo.

Un agente conversazionale, invece, costruisce su questa capacità linguistica aggiungendo componenti architetturali che lo trasformano in un sistema proattivo e integrato. L'agente utilizza l'LLM come "cervello" per il ragionamento linguistico, ma lo inserisce in un'architettura più ampia che comprende diversi elementi fondamentali.

La gestione della memoria e del contesto costituisce la prima differenza significativa. Un agente mantiene uno stato interno che persiste tra le interazioni, permettendo di costruire conversazioni coerenti e di riferirsi a scambi precedenti.

Questa memoria non è semplicemente un log delle conversazioni passate, ma una rappresentazione strutturata delle informazioni rilevanti estratte dal dialogo, che l'agente può interrogare e aggiornare dinamicamente.

La capacità di invocare strumenti esterni (tool calling) [18, 19] rappresenta forse l'innovazione più rilevante dal punto di vista operativo. Un agente può essere configurato per chiamare funzioni specifiche quando necessario, come interrogare database, eseguire calcoli, analizzare immagini o controllare sensori. Il processo funziona attraverso un meccanismo di ragionamento in cui l'agente:

1. Analizza la richiesta dell'utente e identifica se richiede informazioni o azioni che non può gestire autonomamente
2. Seleziona lo strumento appropriato tra quelli disponibili basandosi sulle descrizioni fornite
3. Genera i parametri corretti per invocare la funzione
4. Integra i risultati ottenuti nella risposta all'utente

Nel contesto industriale, questa capacità permette agli agenti di accedere a documentazione tecnica aggiornata, interrogare sistemi di monitoraggio della produzione, o elaborare dati provenienti da sensori, trasformandoli da semplici chatbot a veri assistenti digitali integrati nell'ecosistema produttivo. Gli agenti moderni implementano strategie di ragionamento multi-step che permettono di affrontare problemi complessi attraverso la decomposizione in sotto-problemi. Tecniche come il chain-of-thought prompting guidano l'agente a esplicitare i passaggi del suo ragionamento, migliorando sia la qualità delle risposte che la tracciabilità delle decisioni prese. In applicazioni diagnostiche industriali, questa capacità si traduce nella possibilità di sviluppare piani di investigazione sistematici. Ad esempio, di fronte a un malfunzionamento, l'agente può proporre una sequenza logica di verifiche, considerare ipotesi alternative e suggerire azioni correttive basandosi sull'analisi strutturata delle evidenze disponibili. Un agente conversazionale avanzato può modulare il proprio comportamento in funzione del contesto e dell'utente. La capacità di adattare lo stile comunicativo, il livello di dettaglio tecnico e l'approccio metodologico permette di ottimizzare l'efficacia dell'interazione. Un tecnico esperto riceverà informazioni dettagliate e specifiche, mentre un operatore meno specializzato otterrà spiegazioni semplificate e procedure guidate passo-passo. Questa personalizzazione avviene dinamicamente durante la conversazione, senza necessità di configurazioni manuali. L'agente inferisce il livello di competenza dell'utente dalle domande poste e dal linguaggio utilizzato, adattando progressivamente le proprie risposte. Nel contesto di questa tesi, la distinzione tra modello e agente è particolarmente significativa. Il sistema multi-agente sviluppato nel Capitolo 3 sfrutta proprio queste caratteristiche avanzate: ogni agente specializzato mantiene

il proprio stato interno, può invocare strumenti specifici (come modelli di computer vision per la segmentazione) e collabora con altri agenti attraverso un coordinatore centrale. L'architettura proposta rappresenta quindi un'applicazione concreta dei principi degli agenti conversazionali al dominio del supporto operativo industriale.

Tabella 2.1: Confronto tra LLM e Agenti IA

Caratteristica	**LLM**	**Agenti IA**
Funzionalità principale	Comprensione e generazione del linguaggio naturale.	Automatizzazione di compiti, presa di decisioni e interazione con il mondo reale.
Autonomia	Passiva, risponde agli input dell'utente.	Attiva, opera autonomamente una volta stabiliti gli obiettivi.
Capacità di apprendimento	Statica dopo l'addestramento iniziale (aggiornamenti periodici possibili).	Adattiva, apprende dalle interazioni in tempo reale e dai feedback.
Interazione	Basata su testo, limitata a compiti linguistici.	Multimodale; interagisce con sistemi digitali e ambienti fisici.
Metodo di addestramento	Pre-addestrato su vasti set di dati testuali.	Spesso utilizza l'apprendimento per rinforzo e l'apprendimento supervisionato.
Applicazioni	Creazione di contenuti, assistenza clienti, traduzioni, analisi di mercato.	Assistenti virtuali, veicoli autonomi, robotica, sistemi smart home.
Azioni in tempo reale	Limitate alla generazione di linguaggio in tempo reale.	Esegue azioni e prende decisioni in tempo reale.
Limitazioni	Elevato consumo di risorse, bias nei dati, difficoltà con informazioni aggiornate.	Complessità di sviluppo, necessità di definizione chiara degli obiettivi, potenziali problemi di sicurezza.

Tecniche fondamentali: RAG e Tool Calling

L'implementazione di agenti conversazionali per applicazioni industriali beneficia significativamente dell'integrazione di tecniche avanzate che estendono le capacità base dei modelli linguistici. Tra queste, la Retrieval-Augmented Generation (RAG) rappresenta la svolta più rilevante per contesti che necessitano di accesso a documentazione tecnica specializzata e aggiornata. La tecnica RAG [20] combina le capacità generative degli LLM con sistemi di recupero delle informazioni, permettendo agli

agenti di accedere dinamicamente a basi di conoscenza estese senza necessità di riaddestramento completo. Nel contesto industriale, questa caratteristica si traduce nella capacità di consultare manuali tecnici, procedure operative, database di guasti storici e documentazione normativa in tempo reale, garantendo che le risposte fornite siano sempre basate su informazioni più aggiornate e accurate disponibili. L'implementazione del RAG richiede la creazione di embeddings vettoriali per la documentazione tecnica, utilizzando modelli specializzati come Sentence-BERT o varianti domain-specific addestrate su corpora tecnici. La qualità del processo di chunking e l'ottimizzazione delle strategie di retrieval diventano fattori critici per garantire la pertinenza e la completezza delle informazioni recuperate. Il tool calling e function calling [21, 22] rappresentano un'altra dimensione fondamentale per l'efficacia operativa degli agenti industriali. ù

Queste tecniche permettono ai modelli linguistici di invocare funzioni esterne per eseguire calcoli complessi, accedere a database specializzati, interfacciarsi con sistemi di controllo e elaborare dati provenienti da sensori. L'integrazione di queste capacità trasforma l'agente in un hub di orchestrazione che può coordinare multiple risorse tecnologiche per rispondere a richieste complesse. La progettazione di tool efficaci richiede particolare attenzione alla definizione di interfacce chiare e alla gestione degli errori [23, 24].

Ogni tool deve essere dotato di descrizioni precise che permettano al modello linguistico di selezionare lo strumento appropriato e di generare i parametri corretti per l'invocazione. La gestione robusta degli errori e l'implementazione di meccanismi di retry sono essenziali per mantenere l'affidabilità del sistema in contesti operativi reali. La collaborazione tra agenti può gestire meccanismi di controllo qualità attraverso verifica incrociata, permettendo di raggiungere livelli di affidabilità superiori rispetto a sistemi mono-agente. Coordinare gli agenti richiede protocolli di comunicazione standardizzati e meccanismi di risoluzione dei conflitti per gestire situazioni in cui agenti diversi propongono soluzioni contrastanti. L'implementazione di architetture supervisor-worker o l'adozione di approcci democratici basati su consenso rappresentano strategie complementari per ottimizzare l'efficacia complessiva del sistema.

Applicazioni industriali degli agenti conversazionali

L'adozione di agenti conversazionali nel settore industriale ha dimostrato un potenziale trasformativo significativo, con implementazioni che spaziano dal supporto operativo [25] alla formazione del personale, dalla manutenzione predittiva alla documentazione automatica. Queste applicazioni evidenziano come l'integrazione di tecnologie AI conversazionali possa effettivamente materializzare i principi dell'Industria 5.0, potenziando le capacità umane senza sostituire il ruolo centrale dell'operatore. Nel supporto operativo, gli agenti conversazionali fungono

da assistenti digitali in tempo reale, fornendo accesso immediato a procedure operative, manuali tecnici e protocolli di sicurezza. La capacità di interpretare richieste formulate in linguaggio naturale elimina la necessità di navigare attraverso interfacce complesse o documentazione voluminosa, permettendo agli operatori di mantenere focus sulle attività operative critiche. Studi condotti da aziende manifatturiere leader hanno documentato riduzioni fino al 30% nei tempi di consultazione documentale e incrementi del 15-25% nell'efficienza operativa complessiva.

La personalizzazione del supporto rappresenta un valore aggiunto particolare. Gli agenti possono adattare il livello di dettaglio e la terminologia utilizzata in base al profilo dell'operatore, fornendo spiegazioni semplificate a personale meno esperto e informazioni tecniche dettagliate a specialisti. Questa flessibilità riduce significativamente i tempi di formazione e facilita l'integrazione di nuovi operatori nei processi produttivi. Nell'ambito della formazione e del training, gli agenti conversazionali stanno rivoluzionando l'approccio tradizionale all'educazione tecnica.

La capacità di implementare sistemi tutoriali personalizzati che si adattano al ritmo di apprendimento individuale e al livello di competenza dell'utente ha dimostrato efficacia superiore rispetto ai metodi formativi standardizzati. L'interazione dialogica permette di simulare scenari operativi complessi, fornendo feedback immediato e guidando l'apprendimento attraverso percorsi ottimizzati. La formazione just-in-time rappresenta un'applicazione particolarmente innovativa, dove l'agente fornisce istruzioni contestuali nel momento esatto in cui l'operatore affronta una situazione specifica. Questo approccio minimizza il carico cognitivo e massimizza la retention delle informazioni, risultando particolarmente efficace per procedure complesse o rarely-performed tasks. Il controllo qualità [7] beneficia significativamente dell'integrazione di agenti conversazionali capaci di assistere nei processi decisionali di quality assurance. L'accesso dinamico a standard di qualità, specificazioni tecniche e database di non-conformità precedenti permette di standardizzare i processi ispettivi e ridurre la variabilità interpersonale nelle valutazioni. La tracciabilità completa delle decisioni prese, supportata da documentazione automatica delle motivazioni, facilita l'aderenza a normative di qualità stringenti e semplifica i processi di audit. L'integrazione con sistemi di visione artificiale e sensori IoT permette di implementare controlli qualità augmentati, dove l'agente può correlare evidenze visive con parametri tecnici e standard qualitativi, fornendo valutazioni comprehensive e recommendations data-driven. Nella manutenzione predittiva [26], gli agenti conversazionali rappresentano l'interfaccia naturale per l'interpretazione di dati sensoriali complessi e la traduzione di pattern anomali in raccomandazioni operative concrete. La capacità di correlare dati storici, specificazioni tecniche e evidenze in tempo reale permette di implementare sistemi di early warning sofisticati che guidano proattivamente le decisioni manutentive. L'automazione della documentazione [27] tecnica costituisce un'applicazione con impatto trasversale

significativo. Gli agenti possono generare automaticamente report di intervento, documentazione di non-conformità e summary operativi basandosi su interazioni conversazionali e dati sistemici. Questa capacità riduce drasticamente il carico amministrativo degli operatori e garantisce standardizzazione e completezza della documentazione prodotta.

2.3 Architetture agentiche: dai modelli agli agenti autonomi

I Large Language Models hanno dimostrato capacità straordinarie nella comprensione e generazione del linguaggio naturale, ma il loro vero potenziale emerge quando vengono trasformati da sistemi passivi di elaborazione testuale in agenti autonomi capaci di percepire, ragionare e agire nel mondo. Questa transizione rappresenta un salto concettuale fondamentale: non si tratta più semplicemente di generare risposte a prompt, ma di costruire sistemi in grado di decomporre problemi complessi, pianificare strategie d'azione, utilizzare strumenti esterni e coordinare le proprie attività con altri agenti per raggiungere obiettivi definiti.

Un agente AI, nel contesto dei modelli linguistici, può essere inteso come un'entità software che utilizza un LLM come motore cognitivo centrale, ma che integra questa capacità con moduli aggiuntivi che gli permettono di interagire dinamicamente con l'ambiente circostante. A differenza di un modello isolato, un agente mantiene uno stato interno, accede a risorse esterne attraverso strumenti specializzati, prende decisioni autonome su quali azioni intraprendere e valuta iterativamente i risultati ottenuti per raffinare la propria strategia operativa.

La progettazione di sistemi agentici richiede scelte architetturali fondamentali che determinano la capacità del sistema di gestire la complessità, adattarsi dinamicamente alle situazioni impreviste e coordinare efficacemente risorse multiple. Due paradigmi emergono come centrali in questa progettazione: i pattern di orchestrazione, che definiscono come le operazioni vengono coordinate nel tempo, e i meccanismi di reasoning e tool calling, che abilitano la capacità dell'agente di ragionare esplicitamente e interagire con il mondo esterno.

Orchestrazione sequenziale vs. orchestrazione a grafo

La questione fondamentale nella progettazione di un sistema agentico complesso è decidere come organizzare il flusso di controllo tra le diverse operazioni che l'agente deve compiere. Due paradigmi architetturali si contrappongono in questa scelta, ciascuno con caratteristiche, vantaggi e limitazioni distinte.

L'orchestrazione sequenziale rappresenta l'approccio più intuitivo e diretto: le operazioni vengono eseguite in una sequenza predeterminata, dove ogni passo riceve

in input l'output del passo precedente e produce un risultato che viene passato al passo successivo. Questa architettura, spesso definita pipeline o chain, si adatta perfettamente a problemi ben strutturati dove la logica operativa può essere definita a priori e rimane stabile nel tempo. Pensiamo, ad esempio, a un sistema che deve rispondere a domande consultando documenti: il flusso naturale prevede il recupero dei documenti rilevanti, l'estrazione delle informazioni pertinenti, la formulazione del prompt e infine la generazione della risposta. Ogni passaggio dipende dal precedente in modo lineare e predicibile.

Tuttavia, la rigidità delle pipeline sequenziali diventa un limite critico quando ci si confronta con problemi intrinsecamente dinamici, dove la sequenza di operazioni non può essere predeterminata ma deve emergere dalle caratteristiche specifiche del problema in esame. In contesti industriali complessi, ad esempio, un sistema di supporto diagnostico potrebbe dover decidere dinamicamente se richiedere prima un'analisi visiva, consultare la documentazione tecnica, o combinare entrambe le modalità in funzione del tipo di anomalia rilevata. In questi scenari, imporre una sequenza fissa risulta artificioso e inefficiente.

L'orchestrazione a grafo risponde a questa limitazione introducendo un modello concettualmente più ricco e flessibile. Invece di una catena lineare, il sistema viene modellato come un grafo orientato dove i nodi rappresentano unità operative discrete (che possono essere chiamate a modelli linguistici, elaborazioni computazionali, o invocazioni di strumenti esterni) e gli archi rappresentano possibili transizioni tra queste operazioni. La caratteristica distintiva di questo paradigma è che le transizioni non sono fisse, ma possono essere condizionali, dipendendo dallo stato corrente del sistema o dai risultati ottenuti nei passi precedenti.

Questa flessibilità architettonica abilita pattern operativi sofisticati che sarebbero impossibili o estremamente complessi da implementare con pipeline lineari. Un sistema può implementare cicli di riflessione, dove un nodo di valutazione analizza i risultati ottenuti e decide se l'obiettivo è stato raggiunto o se è necessario tornare a uno stadio precedente per raffinare l'analisi. Può implementare branching decisionale, dove in base al contesto viene scelto un percorso operativo tra molteplici alternative. Può gestire fallback automatici, dove in caso di fallimento di un'operazione viene attivato un percorso alternativo senza compromettere l'intero flusso.

La scelta tra orchestrazione sequenziale e orchestrazione a grafo non è puramente tecnica, ma riflette una filosofia progettuale diversa. Le pipeline privilegiano semplicità, predicibilità e facilità di debugging, rendendole ideali per prototipi rapidi e casi d'uso ben delimitati. I grafi privilegiano invece flessibilità, adattabilità e capacità di gestire complessità emergente, risultando essenziali per sistemi che devono operare in contesti industriali reali dove le situazioni impreviste sono la norma piuttosto che l'eccezione.

Pattern ReAct: Reasoning and Acting

Una delle innovazioni più significative nell'evoluzione degli agenti AI è rappresentata dal pattern ReAct (Reasoning and Acting), che formalizza un ciclo virtuoso tra ragionamento esplicito e azione concreta. L'intuizione fondamentale di questo approccio è che gli agenti più efficaci non agiscono impulsivamente, ma alternano deliberatamente fasi di riflessione (reasoning) e fasi di intervento (acting), in un processo iterativo che mima il modo in cui gli esseri umani affrontano problemi complessi.

Nel pattern ReAct, quando un agente riceve un compito, non passa immediatamente all'azione ma inizia con una fase di thought (pensiero), dove il modello linguistico verbalizza esplicitamente il proprio ragionamento: "Per rispondere a questa domanda, devo prima verificare se ho informazioni sufficienti nella documentazione tecnica. Se non le trovo, dovrò analizzare l'immagine fornita dall'utente." Questo reasoning esplicito non è un semplice esercizio stilistico, ma serve a guidare il modello verso decisioni più ponderate e a rendere il processo decisionale ispezionabile e debuggabile.

Dopo la fase di pensiero, l'agente procede con un'action (azione): può decidere di invocare uno strumento specifico, come un modulo di ricerca documentale o un sistema di analisi visiva. L'azione produce un'observation (osservazione), ovvero il risultato concreto dell'operazione eseguita. A questo punto, il ciclo si ripete: l'agente osserva il risultato ottenuto, riflette su cosa ha appreso e su cosa rimane ancora da fare, e decide se intraprendere un'ulteriore azione o se è pronto a fornire una risposta finale all'utente.

Questo pattern si rivela particolarmente potente in contesti diagnostici e di problem-solving, dove la soluzione emerge attraverso un processo incrementale di raccolta informazioni, verifica ipotesi e raffinamento progressivo. Un agente ReAct che supporta un operatore industriale nell'analisi di un componente meccanico potrebbe, ad esempio, iniziare consultando la documentazione per comprendere le specifiche tecniche del componente, poi richiedere un'analisi visiva per identificare anomalie, quindi confrontare quanto osservato con pattern noti di usura o difetti, e infine sintetizzare una diagnosi integrata che tiene conto di tutte le evidenze raccolte.

Tool calling: estendere le capacità degli LLM

I modelli linguistici, per quanto potenti, sono intrinsecamente limitati dalla loro natura statistica: non possono accedere a database in tempo reale, non possono eseguire calcoli precisi, non possono interagire con API esterne o manipolare file. Il tool calling rappresenta il meccanismo che supera queste limitazioni, trasformando i modelli da processori di testo isolati a orchestratori intelligenti di risorse computazionali esterne.

Il paradigma del tool calling si basa su un'idea elegante: esporre al modello linguistico un catalogo di strumenti (tools) disponibili, ciascuno descritto attraverso un nome semanticamente significativo, una descrizione funzionale che spiega cosa lo strumento è in grado di fare, e uno schema che definisce i parametri di input richiesti. Di fronte a una richiesta dell'utente, il modello analizza il compito, determina autonomamente se e quali strumenti sono necessari per portarlo a termine, genera i parametri appropriati per invocare questi strumenti, e infine integra i risultati ottenuti nella risposta finale.

Questo processo, apparentemente semplice, nasconde una sofisticazione cognitiva notevole. Il modello deve essere capace di mappare richieste espresse in linguaggio naturale a invocazioni strutturate di funzioni, deve comporre sequenze di chiamate a strumenti multipli quando un singolo tool non è sufficiente, deve gestire fallimenti e situazioni ambigue dove non è chiaro quale strumento utilizzare, e deve integrare i risultati in una narrazione coerente che risponde all'esigenza originale dell'utente.

I tool possono assumere forme estremamente diverse in base al dominio applicativo. In contesti industriali, strumenti tipici includono moduli di computer vision per l'analisi automatica di immagini e video, sistemi di retrieval che consultano documentazione tecnica strutturata, interfacce con database di produzione per verificare specifiche e tolleranze, e connessioni con sistemi di tracciabilità per ricostruire la storia di un componente specifico. La capacità di un agente di utilizzare efficacemente questi strumenti è ciò che lo rende realmente utile in scenari operativi concreti, dove l'informazione è distribuita tra fonti eterogenee e la soluzione richiede l'integrazione di evidenze multiple.

Architetture multi-agente

Quando la complessità di un problema supera le capacità di un singolo agente generalista, emerge naturalmente la necessità di organizzare sistemi multi-agente, dove agenti specializzati collaborano per raggiungere obiettivi comuni. Questo paradigma si ispira all'organizzazione del lavoro umano, dove team di esperti con competenze complementari coordinano le proprie attività per affrontare sfide che nessun individuo potrebbe gestire autonomamente.

In un'architettura multi-agente, ciascun agente mantiene una propria area di specializzazione: un agente potrebbe essere esperto nell'analisi visiva di componenti meccanici, un altro nella consultazione di documentazione tecnica normativa, un terzo nella generazione di report strutturati. La sfida centrale diventa allora quella del coordinamento: chi decide quale agente deve intervenire in una data situazione? Come viene gestita la comunicazione tra agenti? Come si garantisce che le competenze vengano utilizzate in modo sinergico piuttosto che ridondante o conflittuale?

Diversi pattern architetturali sono emersi per gestire questo coordinamento. Nel pattern gerarchico, un agente coordinatore (supervisor) riceve le richieste dell'utente, le analizza per comprenderne la natura, e delega selettivamente a uno o più agenti specializzati. Il coordinatore mantiene la visione d'insieme, gestisce lo stato globale del sistema, e sintetizza i contributi degli agenti specializzati in una risposta coerente. Questo pattern riflette strutture organizzative umane consolidate ed è particolarmente adatto quando esiste una chiara separazione di responsabilità tra componenti.

Nel pattern peer-to-peer, gli agenti comunicano direttamente tra loro senza un coordinatore centrale, negoziando autonomamente chi deve intervenire in una data situazione. Questo approccio è più flessibile ma richiede protocolli di comunicazione sofisticati e meccanismi di consenso per evitare conflitti o situazioni di stallo.

Nel pattern pipelined, gli agenti sono organizzati in sequenze dove l'output di un agente diventa l'input del successivo, creando catene di elaborazione specializzate.

La scelta del pattern dipende fortemente dal tipo di problema da affrontare. In contesti industriali dove è necessario supportare operatori in tempo reale, pattern gerarchici con un coordinatore intelligente si sono dimostrati particolarmente efficaci, perché garantiscono tempi di risposta predicibili e facilitano la gestione di situazioni anomale. Il coordinatore può implementare strategie di ottimizzazione, come invocare agenti in parallelo quando le loro competenze sono indipendenti, o serializzare le invocazioni quando esiste una dipendenza informativa tra i task.

Pattern architetturali avanzati

La progettazione di sistemi agentici per applicazioni reali richiede l'adozione di pattern architetturali consolidati che affrontano sfide ricorrenti come la gestione della memoria conversazionale, il mantenimento della coerenza attraverso interazioni multiple, la gestione di fallimenti e ambiguità, e l'ottimizzazione delle performance computazionali.

Il pattern Plan-Execute-Reflect formalizza un ciclo operativo in tre fasi dove l'agente prima elabora un piano strategico per affrontare il compito (plan), poi esegue le azioni previste dal piano utilizzando i propri strumenti (execute), e infine riflette criticamente sui risultati ottenuti per valutare se l'obiettivo è stato raggiunto o se è necessario rivedere la strategia (reflect). Questo pattern previene comportamenti impulsivi e permette all'agente di correggere errori o adattarsi a situazioni inattese.

Il pattern Human-in-the-Loop riconosce che in molti contesti critici, come quelli industriali o medici, le decisioni finali devono rimanere sotto controllo umano. In questo pattern, l'agente presenta raccomandazioni, evidenze e analisi all'operatore umano, che mantiene l'autorità decisionale finale. Il sistema non sostituisce l'esperto

ma lo aumenta, fornendogli strumenti cognitivi potenziati per prendere decisioni più informate in tempi più rapidi.

Il pattern Fallback e Degradazione Graduale affronta la realtà operativa dei sistemi reali, dove componenti possono fallire, dati possono essere incompleti, o situazioni possono essere così ambigue da non permettere una risposta definitiva. In questi casi, il sistema deve essere progettato per degradare gradualmente le proprie capacità piuttosto che collassare completamente: se un'analisi visiva non è possibile, il sistema può comunque fornire supporto consultando la documentazione tecnica; se la documentazione non contiene informazioni pertinenti, può almeno fornire linee guida generali o suggerire di consultare un esperto umano.

Il pattern Conversational Memory Management gestisce il problema della persistenza del contesto attraverso interazioni multiple. Un agente efficace deve ricordare cosa è stato discusso in precedenza, quali ipotesi sono state esplorate, quali strumenti sono già stati utilizzati, per evitare di ripetere operazioni inutili e per mantenere la coerenza della conversazione. Tuttavia, la memoria conversazionale ha un costo computazionale e può introdurre rumore informativo: pattern sofisticati implementano meccanismi di summarization progressiva dove interazioni vecchie vengono compresse preservando solo le informazioni essenziali, o memory pruning dove elementi irrilevanti vengono scartati selettivamente.

Considerazioni finali

La transizione dai modelli linguistici agli agenti autonomi rappresenta un percorso evolutivo inevitabile per portare l'intelligenza artificiale da contesti sperimentali a applicazioni industriali concrete. Tuttavia, questa transizione richiede un ripensamento architetturale profondo che va oltre la semplice integrazione di strumenti: richiede la progettazione di sistemi capaci di ragionare esplicitamente, coordinare risorse multiple, adattarsi a situazioni impreviste, e degradare gradualmente piuttosto che fallire catastroficamente.

I paradigmi di orchestrazione (sequenziale vs. grafo), i pattern di reasoning (ReAct, tool calling), e le architetture multi-agente rappresentano i mattoni concettuali con cui costruire sistemi robusti. La scelta tra queste opzioni non è puramente tecnica, ma riflette decisioni progettuali sul tipo di flessibilità, predicibilità e complessità gestibile che il sistema deve possedere. Nel contesto del lavoratore aumentato nell'Industria 5.0, dove l'obiettivo è potenziare le capacità umane senza sostituirle, architetture ibride che combinano intelligentemente questi pattern emergono come la soluzione più promettente: sistemi orchestrati a grafo che possono adattarsi dinamicamente, agenti specializzati che collaborano sotto supervisione di un coordinatore intelligente, e pattern human-in-the-loop che mantengono l'operatore al centro del processo decisionale.

I capitoli successivi descriveranno come questi principi architetturali astratti sono stati concretizzati nel design e nell'implementazione del sistema multi-agente sviluppato in questa ricerca, mostrando come la teoria delle architetture agentiche possa tradursi in soluzioni operative capaci di supportare efficacemente lavoratori industriali in scenari reali di diagnostica e controllo qualità. I Large Language Models hanno dimostrato capacità straordinarie nella comprensione e generazione del linguaggio naturale, ma il loro vero potenziale emerge quando vengono trasformati da sistemi passivi di elaborazione testuale in agenti autonomi capaci di percepire, ragionare e agire nel mondo. Questa transizione rappresenta un salto concettuale fondamentale: non si tratta più semplicemente di generare risposte a prompt, ma di costruire sistemi in grado di decomporre problemi complessi, pianificare strategie d'azione, utilizzare strumenti esterni e coordinare le proprie attività con altri agenti per raggiungere obiettivi definiti.

Un agente AI, nel contesto dei modelli linguistici, può essere inteso come un'entità software che utilizza un LLM come motore cognitivo centrale, ma che integra questa capacità con moduli aggiuntivi che gli permettono di interagire dinamicamente con l'ambiente circostante. A differenza di un modello isolato, un agente mantiene uno stato interno, accede a risorse esterne attraverso strumenti specializzati, prende decisioni autonome su quali azioni intraprendere e valuta iterativamente i risultati ottenuti per raffinare la propria strategia operativa.

La progettazione di sistemi agentici richiede scelte architetturali fondamentali che determinano la capacità del sistema di gestire la complessità, adattarsi dinamicamente alle situazioni impreviste e coordinare efficacemente risorse multiple. Due paradigmi emergono come centrali in questa progettazione: i pattern di orchestrazione, che definiscono come le operazioni vengono coordinate nel tempo, e i meccanismi di reasoning e tool calling, che abilitano la capacità dell'agente di ragionare esplicitamente e interagire con il mondo esterno.

La questione fondamentale nella progettazione di un sistema agentico complesso è decidere come organizzare il flusso di controllo tra le diverse operazioni che l'agente deve compiere. Due paradigmi architetturali si contrappongono in questa scelta, ciascuno con caratteristiche, vantaggi e limitazioni distinte.

L'orchestrazione sequenziale rappresenta l'approccio più intuitivo e diretto: le operazioni vengono eseguite in una sequenza predeterminata, dove ogni passo riceve in input l'output del passo precedente e produce un risultato che viene passato al passo successivo. Questa architettura, spesso definita pipeline o chain, si adatta perfettamente a problemi ben strutturati dove la logica operativa può essere definita a priori e rimane stabile nel tempo. Pensiamo, ad esempio, a un sistema che deve rispondere a domande consultando documenti: il flusso naturale prevede il recupero dei documenti rilevanti, l'estrazione delle informazioni pertinenti, la formulazione del prompt e infine la generazione della risposta. Ogni passaggio dipende dal precedente in modo lineare e predicibile.

Tuttavia, la rigidità delle pipeline sequenziali diventa un limite critico quando ci si confronta con problemi intrinsecamente dinamici, dove la sequenza di operazioni non può essere predeterminata ma deve emergere dalle caratteristiche specifiche del problema in esame. In contesti industriali complessi, ad esempio, un sistema di supporto diagnostico potrebbe dover decidere dinamicamente se richiedere prima un'analisi visiva, consultare la documentazione tecnica, o combinare entrambe le modalità in funzione del tipo di anomalia rilevata. In questi scenari, imporre una sequenza fissa risulta artificioso e inefficiente.

L'orchestrazione a grafo risponde a questa limitazione introducendo un modello concettualmente più ricco e flessibile. Invece di una catena lineare, il sistema viene modellato come un grafo orientato dove i nodi rappresentano unità operative discrete (che possono essere chiamate a modelli linguistici, elaborazioni computazionali, o invocazioni di strumenti esterni) e gli archi rappresentano possibili transizioni tra queste operazioni. La caratteristica distintiva di questo paradigma è che le transizioni non sono fisse, ma possono essere condizionali, dipendendo dallo stato corrente del sistema o dai risultati ottenuti nei passi precedenti.

Questa flessibilità architettonica abilita pattern operativi sofisticati che sarebbero impossibili o estremamente complessi da implementare con pipeline lineari. Un sistema può implementare cicli di riflessione, dove un nodo di valutazione analizza i risultati ottenuti e decide se l'obiettivo è stato raggiunto o se è necessario tornare a uno stadio precedente per raffinare l'analisi. Può implementare branching decisionale, dove in base al contesto viene scelto un percorso operativo tra molteplici alternative. Può gestire fallback automatici, dove in caso di fallimento di un'operazione viene attivato un percorso alternativo senza compromettere l'intero flusso.

La scelta tra orchestrazione sequenziale e orchestrazione a grafo non è puramente tecnica, ma riflette una filosofia progettuale diversa. Le pipeline privilegiano semplicità, predicibilità e facilità di debugging, rendendole ideali per prototipi rapidi e casi d'uso ben delimitati. I grafi privilegiano invece flessibilità, adattabilità e capacità di gestire complessità emergente, risultando essenziali per sistemi che devono operare in contesti industriali reali dove le situazioni impreviste sono la norma piuttosto che l'eccezione.

Una delle innovazioni più significative nell'evoluzione degli agenti AI è rappresentata dal pattern ReAct (Reasoning and Acting), che formalizza un ciclo virtuoso tra ragionamento esplicito e azione concreta. L'intuizione fondamentale di questo approccio è che gli agenti più efficaci non agiscono impulsivamente, ma alternano deliberatamente fasi di riflessione (reasoning) e fasi di intervento (acting), in un processo iterativo che mima il modo in cui gli esseri umani affrontano problemi complessi. Nel pattern ReAct, quando un agente riceve un compito, non passa immediatamente all'azione ma inizia con una fase di thought (pensiero), dove

il modello linguistico verbalizza esplicitamente il proprio ragionamento: "Per rispondere a questa domanda, devo prima verificare se ho informazioni sufficienti nella documentazione tecnica. Se non le trovo, dovrò analizzare l'immagine fornita dall'utente." Questo reasoning esplicito non è un semplice esercizio stilistico, ma serve a guidare il modello verso decisioni più ponderate e a rendere il processo decisionale ispezionabile e debuggabile.

Dopo la fase di pensiero, l'agente procede con un'action (azione): può decidere di invocare uno strumento specifico, come un modulo di ricerca documentale o un sistema di analisi visiva. L'azione produce un'observation (osservazione), ovvero il risultato concreto dell'operazione eseguita. A questo punto, il ciclo si ripete: l'agente osserva il risultato ottenuto, riflette su cosa ha appreso e su cosa rimane ancora da fare, e decide se intraprendere un'ulteriore azione o se è pronto a fornire una risposta finale all'utente.

Questo pattern si rivela particolarmente potente in contesti diagnostici e di problem-solving, dove la soluzione emerge attraverso un processo incrementale di raccolta informazioni, verifica ipotesi e raffinamento progressivo. Un agente ReAct che supporta un operatore industriale nell'analisi di un componente meccanico potrebbe, ad esempio, iniziare consultando la documentazione per comprendere le specifiche tecniche del componente, poi richiedere un'analisi visiva per identificare anomalie, quindi confrontare quanto osservato con pattern noti di usura o difetti, e infine sintetizzare una diagnosi integrata che tiene conto di tutte le evidenze raccolte.

I modelli linguistici, per quanto potenti, sono intrinsecamente limitati dalla loro natura statistica: non possono accedere a database in tempo reale, non possono eseguire calcoli precisi, non possono interagire con API esterne o manipolare file. Il tool calling rappresenta il meccanismo che supera queste limitazioni, trasformando i modelli da processori di testo isolati a orchestratori intelligenti di risorse computazionali esterne.

Il paradigma del tool calling si basa su un'idea elegante: esporre al modello linguistico un catalogo di strumenti (tools) disponibili, ciascuno descritto attraverso un nome semanticamente significativo, una descrizione funzionale che spiega cosa lo strumento è in grado di fare, e uno schema che definisce i parametri di input richiesti. Di fronte a una richiesta dell'utente, il modello analizza il compito, determina autonomamente se e quali strumenti sono necessari per portarlo a termine, genera i parametri appropriati per invocare questi strumenti, e infine integra i risultati ottenuti nella risposta finale.

Questo processo, apparentemente semplice, nasconde una sofisticazione cognitiva notevole. Il modello deve essere capace di mappare richieste espresse in linguaggio naturale a invocazioni strutturate di funzioni, deve comporre sequenze di chiamate a strumenti multipli quando un singolo tool non è sufficiente, deve gestire fallimenti e situazioni ambigue dove non è chiaro quale strumento utilizzare, e deve integrare

i risultati in una narrazione coerente che risponde all'esigenza originale dell'utente.

I tool possono assumere forme estremamente diverse in base al dominio applicativo. In contesti industriali, strumenti tipici includono moduli di computer vision per l'analisi automatica di immagini e video, sistemi di retrieval che consultano documentazione tecnica strutturata, interfacce con database di produzione per verificare specifiche e tolleranze, e connessioni con sistemi di tracciabilità per ricostruire la storia di un componente specifico. La capacità di un agente di utilizzare efficacemente questi strumenti è ciò che lo rende realmente utile in scenari operativi concreti, dove l'informazione è distribuita tra fonti eterogenee e la soluzione richiede l'integrazione di evidenze multiple. Quando la complessità di un problema supera le capacità di un singolo agente generalista, emerge naturalmente la necessità di organizzare sistemi multi-agente, dove agenti specializzati collaborano per raggiungere obiettivi comuni. Questo paradigma si ispira all'organizzazione del lavoro umano, dove team di esperti con competenze complementari coordinano le proprie attività per affrontare sfide che nessun individuo potrebbe gestire autonomamente.

In un'architettura multi-agente, ciascun agente mantiene una propria area di specializzazione: un agente potrebbe essere esperto nell'analisi visiva di componenti meccanici, un altro nella consultazione di documentazione tecnica normativa, un terzo nella generazione di report strutturati. La sfida centrale diventa allora quella del coordinamento: chi decide quale agente deve intervenire in una data situazione? Come viene gestita la comunicazione tra agenti? Come si garantisce che le competenze vengano utilizzate in modo sinergico piuttosto che ridondante o conflittuale? Diversi pattern architetturali sono emersi per gestire questo coordinamento. Nel pattern gerarchico, un agente coordinatore (supervisor) riceve le richieste dell'utente, le analizza per comprenderne la natura, e delega selettivamente a uno o più agenti specializzati.

Il coordinatore mantiene la visione d'insieme, gestisce lo stato globale del sistema, e sintetizza i contributi degli agenti specializzati in una risposta coerente. Questo pattern riflette strutture organizzative umane consolidate ed è particolarmente adatto quando esiste una chiara separazione di responsabilità tra componenti.

Nel pattern peer-to-peer, gli agenti comunicano direttamente tra loro senza un coordinatore centrale, negoziando autonomamente chi deve intervenire in una data situazione. Questo approccio è più flessibile ma richiede protocolli di comunicazione sofisticati e meccanismi di consenso per evitare conflitti o situazioni di stallo.

Nel pattern pipelined, gli agenti sono organizzati in sequenze dove l'output di un agente diventa l'input del successivo, creando catene di elaborazione specializzate.

La scelta del pattern dipende fortemente dal tipo di problema da affrontare. In contesti industriali dove è necessario supportare operatori in tempo reale, pattern gerarchici con un coordinatore intelligente si sono dimostrati particolarmente efficaci, perché garantiscono tempi di risposta predicibili e facilitano la gestione di situazioni anomale. Il coordinatore può implementare strategie di ottimizzazione,

come invocare agenti in parallelo quando le loro competenze sono indipendenti, o serializzare le invocazioni quando esiste una dipendenza informativa tra i task.

La progettazione di sistemi agentici per applicazioni reali richiede l'adozione di pattern architetturali consolidati che affrontano sfide ricorrenti come la gestione della memoria conversazionale, il mantenimento della coerenza attraverso interazioni multiple, la gestione di fallimenti e ambiguità, e l'ottimizzazione delle performance computazionali.

Il pattern Plan-Execute-Reflect formalizza un ciclo operativo in tre fasi dove l'agente prima elabora un piano strategico per affrontare il compito (plan), poi esegue le azioni previste dal piano utilizzando i propri strumenti (execute), e infine riflette criticamente sui risultati ottenuti per valutare se l'obiettivo è stato raggiunto o se è necessario rivedere la strategia (reflect). Questo pattern previene comportamenti impulsivi e permette all'agente di correggere errori o adattarsi a situazioni inattese. Il pattern Human-in-the-Loop riconosce che in molti contesti critici, come quelli industriali o medici, le decisioni finali devono rimanere sotto controllo umano. In questo pattern, l'agente presenta raccomandazioni, evidenze e analisi all'operatore umano, che mantiene l'autorità decisionale finale. Il sistema non sostituisce l'esperto ma lo aumenta, fornendogli strumenti cognitivi potenziati per prendere decisioni più informate in tempi più rapidi.

Il pattern Fallback e Degradazione Graduale affronta la realtà operativa dei sistemi reali, dove componenti possono fallire, dati possono essere incompleti, o situazioni possono essere così ambigue da non permettere una risposta definitiva. In questi casi, il sistema deve essere progettato per degradare gradualmente le proprie capacità piuttosto che collassare completamente: se un'analisi visiva non è possibile, il sistema può comunque fornire supporto consultando la documentazione tecnica; se la documentazione non contiene informazioni pertinenti, può almeno fornire linee guida generali o suggerire di consultare un esperto umano.

Il pattern Conversational Memory Management gestisce il problema della persistenza del contesto attraverso interazioni multiple. Un agente efficace deve ricordare cosa è stato discusso in precedenza, quali ipotesi sono state esplorate, quali strumenti sono già stati utilizzati, per evitare di ripetere operazioni inutili e per mantenere la coerenza della conversazione. Tuttavia, la memoria conversazionale ha un costo computazionale e può introdurre rumore informativo: pattern sofisticati implementano meccanismi di summarization progressiva dove interazioni vecchie vengono compresse preservando solo le informazioni essenziali, o memory pruning dove elementi irrilevanti vengono scartati selettivamente.

La transizione dai modelli linguistici agli agenti autonomi rappresenta un percorso evolutivo inevitabile per portare l'intelligenza artificiale da contesti sperimentali a applicazioni industriali concrete. Tuttavia, questa transizione richiede un ripensamento architetturale profondo che va oltre la semplice integrazione di strumenti:

richiede la progettazione di sistemi capaci di ragionare esplicitamente, coordinare risorse multiple, adattarsi a situazioni impreviste, e degradare gradualmente piuttosto che fallire catastroficamente.

I paradigmi di orchestrazione (sequenziale vs. grafo), i pattern di reasoning (ReAct, tool calling), e le architetture multi-agente rappresentano i mattoni concettuali con cui costruire sistemi robusti. La scelta tra queste opzioni non è puramente tecnica, ma riflette decisioni progettuali sul tipo di flessibilità, predicibilità e complessità gestibile che il sistema deve possedere. Nel contesto del lavoratore aumentato nell'Industria 5.0, dove l'obiettivo è potenziare le capacità umane senza sostituirle, architetture ibride che combinano intelligentemente questi pattern emergono come la soluzione più promettente: sistemi orchestrati a grafo che possono adattarsi dinamicamente, agenti specializzati che collaborano sotto supervisione di un coordinatore intelligente, e pattern human-in-the-loop che mantengono l'operatore al centro del processo decisionale.

I capitoli successivi descriveranno come questi principi architetturali astratti sono stati concretizzati nel design e nell'implementazione del sistema multi-agente sviluppato in questa ricerca, mostrando come la teoria delle architetture agentiche possa tradursi in soluzioni operative capaci di supportare efficacemente lavoratori industriali in scenari reali di diagnostica e controllo qualità.

Capitolo 3

Progettazione del Sistema

3.1 Architettura multi-agente proposta

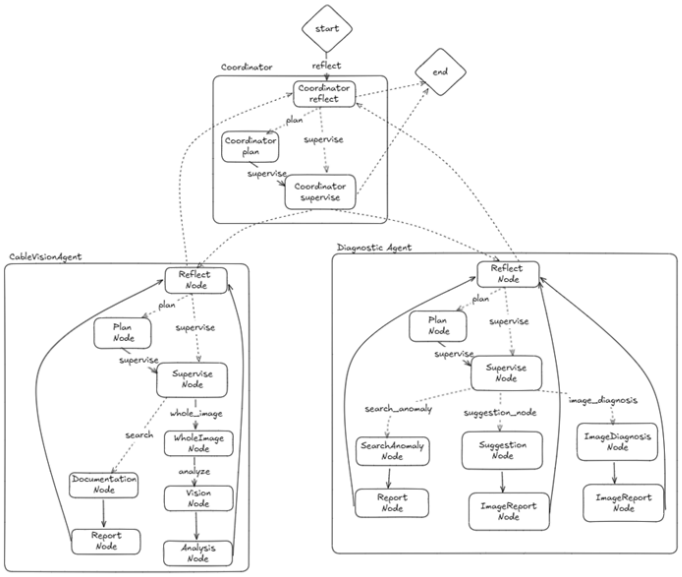

Figura 3.1: Architettura multi-agente proposta

Il sistema sviluppato si basa su un'architettura multi-agente progettata per supportare in modo modulare e collaborativo l'analisi, la diagnosi e la documentazione di anomalie in contesto industriale. Alla base di questa architettura c'é un Coordinator Agent, che ha il compito di ricevere gli input iniziali e decidere dinamicamente, in base al contesto, se coinvolgere uno o piú agenti specializzati. Il

sistema é progettato per gestire diverse tipologie di input che riflettono le modalitá operative tipiche in contesti industriali:

- **Input puramente testuale**: Per domande tecniche teoriche che non richiedono analisi visiva, come consultazioni di documentazione tecnica, quesiti su standard normativi o richieste di spiegazioni su principi di funzionamento.

- **Input video interattivoe**: L'operatore puó registrare un video della componente o del sistema di interesse, utilizzando la funzionalitá di pausa per selezionare il frame specifico che presenta l'anomalia o l'elemento da analizzare. Questa modalitá permette documentazione dinamica sul campo e selezione precisa del momento piú rappresentativo del problema.

- **Input immagine con markert**: L'operatore puó caricare direttamente un'immagine contenente marker visivi (frecce, cerchi, bollini colorati, indicatori grafici) che segnalano la zona specifica da analizzare. L'utente puó quindi formulare domande mirate per ottenere informazioni dettagliate riguardo a ció che é illustrato nel punto indicato dal marker.

3.1.1 Coordinator Agent

Il Coordinator é l'agente che governa il flusso complessivo del sistema, muovendosi su un grafo decisionale con tre possibili stati: Reflect: Fase di riflessione sul contesto e sull'intento dell'utente. Questo nodo rappresenta la fase critica di interpretazione della richiesta, dove il Coordinator analizza la natura del quesito, valuta se richiede competenze visive, diagnostiche o puramente teoriche, e identifica le risorse necessarie per fornire una risposta appropriata. Il nodo Reflect implementa capacitá di natural language understanding per disambiguare richieste ambigue e classificare il tipo di supporto richiesto. Plan: Fase di pianificazione in cui decide quali agenti specializzati devono intervenire. Una caratteristica distintiva dell'implementazione é l'inclusione di un campo reasoning che documenta esplicitamente la logica decisionale adottata. Questa scelta progettuale garantisce trasparenza nel processo decisionale, facilita debugging e ottimizzazione del sistema, e fornisce tracciabilitá delle decisioni per audit e miglioramento continuo delle strategie di orchestrazione. Supervise: Fase di supervisione dell'esecuzione del piano, dove il Coordinator monitora l'avanzamento delle attivitá e coordina la raccolta dei risultati dai vari agenti. Il system message definisce in dettaglio la logica decisionale del Coordinator:

1. In caso di nuovo task o subtask, il Coordinator passa allo stato di plan e pianifica quali agenti coinvolgere

2. Se il piano esiste giá e deve solo essere portato avanti, passa allo stato di supervise
3. Se non é necessario procedere o in caso di task concluso, puó terminare il flusso (end)

Il Coordinator decide se coinvolgere:

1. Il CableVisionAgent, per l'ispezione visiva o la segmentazione di immagini
2. Il DiagnosticAgent, per l'analisi tecnica e le raccomandazioni operative

3.1.2 Nodo WholeImage: gateway di sicurezza e controllo preliminare

Prima di procedere con analisi specializzate piú costose computazionalmente, il sistema implementa un nodo WholeImage che svolge funzioni critiche di controllo e valutazione preliminare:

- **Funzione di privacy protection**: Il nodo esegue controlli automatici per identificare presenza di volti umani, dati personali identificativi o contenuti che potrebbero violare normative privacy. Questa funzionalitá é essenziale per garantire compliance in contesti industriali.
- **Valutazione di idoneitá tecnica**: Il sistema verifica che l'immagine contenga effettivamente componenti tecnici rilevanti (cavi, componenti elettrici/meccanici) per l'analisi richiesta, evitando elaborazioni su contenuti non pertinenti.
- **Descrizione generale** : Quando la richiesta é di natura puramente descrittiva, il nodo puó fornire descrizioni generiche dell'immagine senza necessitá di analisi specializzate piú approfondite.
- **Gate decisionale**:Per richieste che richiedono analisi diagnostiche avanzate, il nodo funziona come gate di controllo che autorizza o blocca il proseguimento verso fasi successive, restituendo messaggi come "Analysis interrupted: the area contains personal data" o "No cable or technical component detected in this area".

3.1.3 CableVisionAgent

Il CableVisionAgent é dedicato all'analisi visiva dell'area indicata nel frame. Si articola attraverso un grafo interno con i seguenti nodi principali:
CableVision reflect → CableVision plan → CableVision supervise
dove viene pianificata e controllata l'analisi visiva

VisionNode

Il nodo core che implementa la segmentazione automatica utilizzando EfficientSAM [28]. Riceve il nome del file immagine e coordina il processo di segmentazione, verificando l'esistenza del file e gestendo eventuali errori. Il nodo aggiorna lo stato dell'agente con il percorso dell'immagine segmentata per le fasi successive.

AnalysisNode

Si occupa dell'analisi dell'immagine segmentata, generando descrizioni testuali dettagliate della regione evidenziata dalla segmentazione. Combina l'elaborazione visiva con terminologia tecnica appropriata e include l'immagine segmentata nel formato markdown strutturato.

DocumentationNode

Produce un report tecnico teorico descrivendo in modo documentale gli elementi identificati, utilizzando il sistema RAG [20] per accedere alla documentazione tecnica specializzata (Technical Domain Knowledge Document.pdf).

ReportNode

Genera la documentazione finale che sintetizza le informazioni recuperate dalla knowledge base in un formato strutturato e leggibile. Questo agente permette di integrare una visione oggettiva (segmentazione) con una spiegazione tecnica mirata, utile come base per eventuali fasi diagnostiche successive.

3.1.4 DiagnosticAgent

Il DiagnosticAgent entra in gioco quando é richiesta un'analisi piú approfondita, per individuare le cause del problema e suggerire possibili interventi. L'agente é completamente scollegato dal VisionAgent ma puó essere orchestrato in sequenza per task complessi. L'agente segue un grafo simile:

Diagnostic reflect → Diagnostic plan → Diagnostic supervise

ImageDiagnosisNode

Quando disponibile input visivo, esegue classificazione automatica di anomalie utilizzando modelli specializzati (AnomalibTool). Produce classification scores, confidence metrics, anomaly scores e heatmap per localizzazione precisa delle anomalie. Il nodo gestisce l'estrazione del filename dal messaggio utente e coordina l'analisi tramite il tool specializzato.

ImageReportNode

Genera report diagnostici strutturati basati sui risultati della classificazione automatica, includendo classe rilevata, score di confidenza, score di anomalia e visualizzazioni heatmap quando disponibili.

SearchAnomalyNode

Implementa ricerca semantica nella documentazione tecnica per identificare cause, failure modes e pattern di guasto correlati alle anomalie identificate. Utilizza il sistema RAG per correlare evidenze con knowledge base specialistiche.

SuggestionNode

Formula raccomandazioni operative concrete basate su evidenze diagnostiche e best practices documentate. Cerca nella documentazione (Cable Fault Diagnostics.pdf) soluzioni e azioni correttive appropriate.

ReportNode e SuggestionReportNode

Nodi specializzati che producono documentazione strutturata appropriata al tipo di analisi eseguita (ricerca anomalie vs. suggerimenti operativi), garantendo output consistenti e professionalmente formattati.

3.1.5 Flusso di controllo del grafo

All'inizio, il Coordinator riceve l'input che può essere:

- Una domanda puramente tecnica senza necessità di input visivo
- Un frame video con un puntatore che indica l'area da analizzare
- Un'immagine con marker visivi

Dopo aver riflettuto sull'intento (fase reflect), il Coordinator decide:

- Se coinvolgere solo il CableVisionAgent per un'analisi visiva o descrizione
- Se passare direttamente al DiagnosticAgent per un'analisi tecnica basata su sintomi descritti
- Oppure combinare i due agenti in sequenza (prima controllo WholeImage, poi CableVisionAgent → DiagnosticAgent) quando serve prima la visione per poi procedere alla diagnosi

Ciascun agente gestisce internamente il proprio ciclo (reflect → plan → supervise) e, una volta completata l'analisi, produce documentazione o raccomandazioni che vengono raccolte dal Coordinator per comporre la risposta complessiva al problema. L'architettura è pensata per:

- Modularità: ogni agente è indipendente e specializzato
- Flessibilità: il Coordinator può orchestrare diversi flussi in base al tipo di richiesta
- Sicurezza: controlli integrati per privacy e idoneità dei contenuti
- Tracciabilità: reasoning esplicito nelle decisioni di pianificazione
- Chiarezza: la divisione tra analisi visiva, documentazione e diagnostica tecnica evita sovrapposizioni e rende le risposte più strutturate
- Estendibilità: nuovi agenti possono essere aggiunti per altri compiti (es. simulazioni, ricerca, ecc.)

Definizione dei casi d'uso e requisiti

Per guidare lo sviluppo dell'architettura e garantire copertura funzionale completa, sono stati definiti casi d'uso rappresentativi che riflettono le modalità operative tipiche del contesto industriale target. Questi scenari comprendono sia situazioni standard, quali l'analisi visiva di componenti con marker, la diagnosi di anomalie basata su evidenze fotografiche e la consultazione di documentazione tecnica specializzata, sia casi limite progettati per testare la robustezza del sistema. Tra questi ultimi rientrano richieste non pertinenti al dominio tecnico, input privi di contenuto rilevante o quesiti che richiedono competenze al di fuori dello scope operativo definito.

La definizione preliminare di questi requisiti funzionali ha orientato sia le scelte architetturali, in particolare la struttura modulare e la logica decisionale del Coordinator, sia l'implementazione dei meccanismi di controllo come il nodo WholeImage, garantendo che il sistema sia in grado non solo di fornire assistenza tecnica qualificata negli scenari operativi previsti, ma anche di riconoscere e gestire appropriatamente situazioni non conformi, preservando affidabilità e sicurezza operativa.

3.2 Pipeline di elaborazione video e segmentazione

La pipeline di elaborazione video e segmentazione costituisce un elemento centrale del sistema multi-agente, poiché consente di trasformare un flusso video generico

in informazioni visive dettagliate e strutturate, fondamentali per l'analisi e la diagnostica. Questa pipeline è progettata per operare in scenari industriali reali, dove l'operatore documenta visivamente un problema, un guasto o una componente meccanica. Le fasi principali della pipeline:

1. **Acquisizione e preprocessamento del video**

   Il processo inizia con l'acquisizione di un video in cui l'operatore inquadra la parte meccanica di interesse o può avere la possibilità di caricare una foto. Il video può essere registrato direttamente sul campo o pre-caricato, e viene preprocessato per migliorarne la qualità (riduzione del rumore, correzione del colore, normalizzazione).

2. **Rilevazione automatica del marker con OpenCV**

   Per identificare automaticamente la zona su cui focalizzare l'analisi, viene utilizzata la libreria OpenCV. Il sistema rileva marker visivi, come frecce, rettangoli colorati o indicatori grafici che l'operatore inserisce per segnalare la componente rilevante. Questa fase elimina la necessità di interventi manuali e permette al sistema di funzionare anche in contesti dove non è pratico cliccare o disegnare aree.

3. **Segmentazione con EfficientSAM**

   Una volta individuata la zona di interesse, viene utilizzato EfficientSAM (`https://github.com/yformer/EfficientSAM`), un modello avanzato per la segmentazione di immagini. EfficientSAM è progettato per combinare le alte prestazioni del Segment Anything Model (SAM) con un'architettura più leggera ed efficiente, ideale per contesti real-time o con risorse hardware limitate, come sistemi embedded o edge computing industriale. Il modello produce una maschera segmentata che isola con precisione il componente o l'area di interesse dal resto dell'immagine.

4. **Post-processing e arricchimento dei dati**

   Le maschere ottenute vengono elaborate per generare output visivi più comprensibili:

   - immagini con la zona segmentata evidenziata,
   - immagini annotate,
   - estrazione di caratteristiche aggiuntive utili alla diagnostica (ad esempio contorni, dimensioni).

5. **Integrazione con l'agente Vision e generazione del report**

   Il risultato della segmentazione viene passato all'agente di Vision del sistema multi-agente, che produce:

- una **descrizione testuale** di ciò che viene rilevato (es. "La zona segmentata corrisponde al collettore di scarico, presenta una superficie irregolare. . . "),
- un'immagine segmentata arricchita con annotazioni,
- eventualmente, una documentazione tecnica generata dal nodo di Documentazione collegato, che spiega la funzione del componente.

Questa pipeline consente così di trasformare automaticamente un contenuto visivo in conoscenza strutturata, fondamentale per il successivo intervento diagnostico o per la creazione di report dettagliati.

3.3 Integrazione RAG per supporto diagnostico

La Retrieval-Augmented Generation (RAG) rappresenta una metodologia avanzata che combina tecniche di recupero dell'informazione con modelli di generazione del linguaggio naturale per migliorare la qualità e l'affidabilità delle risposte fornite da un sistema intelligente. Nel contesto del nostro sistema multiagente, la RAG viene utilizzata come meccanismo chiave per supportare il processo diagnostico, integrando la conoscenza presente in una documentazione specialistica con le capacità di ragionamento e generazione testuale degli agenti.

Funzionamento della RAG

Il funzionamento della RAG si basa su un duplice processo: la fase di retrieval (recupero) e la fase di generation (generazione). In primo luogo, il sistema riceve una query, che rappresenta una domanda o una richiesta da parte dell'utente o di un agente. Questa query viene utilizzata per interrogare un database di conoscenza pre-costruito, composto da documenti o frammenti testuali rilevanti al dominio diagnostico. Per rendere efficiente e scalabile questa ricerca, i documenti vengono pre-processati tramite tecniche di embedding, ovvero la trasformazione dei testi in vettori numerici in uno spazio semantico continuo. Questi vettori consentono di misurare la somiglianza tra la query e i documenti in modo rapido e preciso utilizzando metriche di distanza (ad esempio, la distanza coseno). Durante la fase di retrieval, vengono selezionati i documenti più pertinenti sulla base della vicinanza semantica tra la query e gli embeddings, permettendo di focalizzare l'attenzione del sistema su informazioni rilevanti e contestualizzate. Successivamente, nella fase di generation, il modello di linguaggio utilizza i documenti recuperati come contesto per generare una risposta articolata e coerente, che integra le informazioni estratte con capacità di ragionamento e formulazione naturale del testo.

Gestione e creazione degli embeddings

Il primo passo cruciale nell'integrazione della RAG è la gestione della documentazione, che viene segmentata in unità informative di dimensioni gestibili (ad esempio paragrafi o sezioni). Ogni segmento viene quindi convertito in un embedding mediante l'utilizzo di modelli preaddestrati di trasformatori (ad esempio BERT, Sentence-BERT). Questi embeddings vengono indicizzati in un database vettoriale, che supporta ricerche rapide e accurate nel cosiddetto spazio semantico. In fase di interrogazione, la query viene anch'essa convertita in embedding e confrontata con quelli memorizzati per individuare i documenti più rilevanti. Questo processo consente di lavorare su grandi quantità di testo in modo efficiente, mantenendo un elevato livello di precisione nel recupero delle informazioni critiche per la diagnosi.

Integrazione con il sistema multiagente

Nel sistema multiagente, il modulo RAG agisce come un componente specializzato che supporta gli agenti nella fase decisionale, fornendo risposte basate sulla documentazione tecnica e scientifica. Quando un agente riceve una richiesta complessa o necessita di verificare informazioni diagnostiche, invia una query al modulo RAG, che esegue il recupero dei documenti e genera una risposta contestualizzata. Questo approccio permette di combinare l'intelligenza distribuita degli agenti con l'accesso dinamico a una base di conoscenza strutturata e aggiornata, migliorando l'accuratezza e l'affidabilità delle diagnosi fornite dal sistema.

Funzionamento del modulo di interrogazione multi-query

Il sistema multiagente interagisce con un servizio esterno basato su Retrieval-Augmented Generation (RAG) attraverso un modulo chiamato MultiQueryInformationRetrievalTool. Questo modulo è responsabile di gestire le richieste di ricerca semantica rivolte al knowledge base, basato su documentazione specialistica indicizzata tramite embeddings. Il Flusso di funzionamento è composto da:

- Il modulo riceve in input una lista di query di ricerca, rappresentanti le diverse domande o argomenti diagnostici da approfondire.
- Viene specificato il contesto della ricerca, che può limitare il dominio di applicazione (ad esempio "solo documenti della stanza corrente", o documenti condivisi tra più stanze).
- Il modulo invia una richiesta HTTP POST asincrona o sincrona all'endpoint del server RAG (/multiple_query), fornendo in JSON le query, il contesto e altri parametri come l'identificativo della stanza e la dimensione dei bundle di documenti da recuperare.

- Il server RAG elabora le query, eseguendo un processo di retrieval sui documenti indicizzati tramite embeddings, e restituisce un insieme di documenti rilevanti raggruppati in bundle, ciascuno con metadati quali il documento sorgente e le pagine di riferimento.
- Il modulo riceve la risposta, ne effettua il parsing e genera un report sintetico che descrive la quantità e la tipologia di documenti recuperati, organizzati per fonte e intervallo di pagine.
- Il report e i documenti rilevanti vengono quindi restituiti al sistema multiagente, che li usa come base per la generazione di risposte diagnostiche contestualizzate.

Capitolo 4

Implementazione

Il presente capitolo descrive i dettagli implementativi del sistema multi-agente sviluppato, illustrando le scelte tecnologiche, le architetture dei singoli componenti e le metodologie adottate per l'orchestrazione degli agenti. L'implementazione si concentra sulla traduzione pratica dei principi architetturali esposti nel Capitolo 3, fornendo una descrizione tecnica dettagliata che mantiene l'equilibrio tra chiarezza espositiva e precisione implementativa.

4.1 Architettura LangChain: chain, prompt e tools

Introduzione a LangChain

L'evoluzione degli Large Language Models ha evidenziato rapidamente la necessità di framework che permettessero di superare le limitazioni intrinseche dei modelli isolati, trasformandoli in sistemi capaci di interazioni complesse con l'ambiente esterno. In questo contesto si colloca LangChain, un framework open-source che ha rivoluzionato l'approccio allo sviluppo di applicazioni basate su LLM, fornendo un'architettura modulare per la creazione di sistemi AI complessi attraverso la composizione flessibile di componenti specializzati. LangChain nasce dalla necessità di affrontare una limitazione fondamentale dei modelli linguistici tradizionali: la loro natura essenzialmente reattiva e isolata. Mentre i modelli puri eccellono nella generazione di testo coerente e contestualmente appropriato, mancano della capacità di mantenere stato persistente, accedere a informazioni dinamiche o interagire con sistemi esterni. Il framework risponde a questa sfida implementando un'architettura che astrae la complessità dell'integrazione dati e del raffinamento dei prompt, permettendo agli sviluppatori di personalizzare sequenze operative per costruire applicazioni complesse con maggiore rapidità ed efficienza.

Architettura fondamentale di LangChain

L'architettura di LangChain [29] si articola attorno a tre pilastri tecnologici che operano in sinergia per trasformare i Large Language Models da sistemi isolati in piattaforme integrate capaci di gestire workflow complessi. Questa struttura modulare permette la creazione di applicazioni AI sofisticate attraverso la combinazione orchestrata di componenti specializzati. I tre componenti fondamentali sono:

- **Chains**: Sequenze logiche di operazioni che coordinano l'esecuzione di task multipli
- **Prompts**: Template intelligenti per la gestione dinamica delle comunicazioni con gli LLM
- **Tools**: Interfacce per l'interazione con sistemi esterni e risorse computazionali

Questi elementi collaborano per creare un ecosistema integrato dove ogni componente mantiene la propria specializzazione pur contribuendo a obiettivi sistemici più ampi. L'approccio modulare garantisce che modifiche o aggiornamenti a singoli componenti non compromettano la stabilità complessiva dell'architettura. I principi architetturali che guidano il design includono:

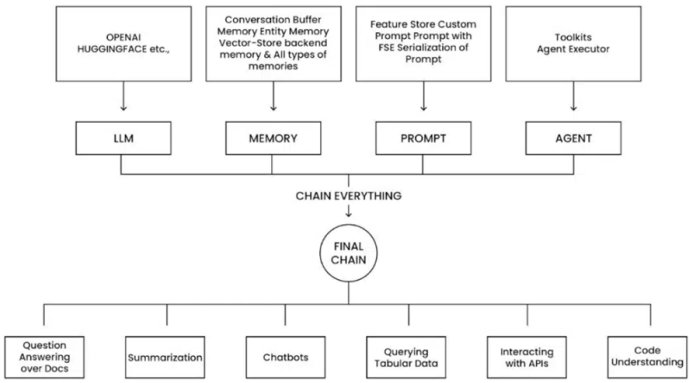

Figura 4.1: Architettura modulare di LangChain

- **Modularità operativa**: Ogni componente è progettato per funzionare autonomamente, facilitando testing, debugging e manutenzione incrementale
- **Composabilità dinamica**: I componenti possono essere ricombinati in configurazioni diverse per rispondere a esigenze applicative specifiche
- **Astrazione procedurale**: Le complessità tecniche sono nascoste dietro interfacce intuitive che riducono la curva di apprendimento

- **Estensibilità sistemica**: L'architettura supporta l'aggiunta di nuovi componenti senza richiedere modifiche strutturali

- **Interoperabilità cross-platform**: Supporto nativo per diversi provider di LLM e tecnologie di backend

Questa struttura architettonica si rivela particolarmente efficace in contesti industriali dove la necessità di integrazione con sistemi legacy e l'evoluzione continua dei requisiti operativi richiedono soluzioni flessibili e robuste.

Chains: Orchestrazione sequenziale di operazioni

Il concetto di Chain rappresenta il cuore dell'architettura LangChain, implementando un paradigma che trasforma sequenze di operazioni discrete in flussi di lavoro coordinati e intelligenti. Una Chain orchestra l'esecuzione di multiple chiamate a LLM, manipolazioni di dati e interazioni con strumenti esterni attraverso logica di controllo che può includere ramificazioni condizionali e gestione di stati complessi. L'approccio modulare delle Chains permette di collegare prompt multipli in sequenze logiche dove l'output di un passaggio diventa l'input del successivo, creando pipeline potenti per compiti che richiedono elaborazione multi-step come summarization, question-answering e analisi testuali complesse. Questa metodologia si rivela particolarmente efficace in contesti industriali dove i processi diagnostici richiedono spesso l'integrazione di evidenze multiple e l'applicazione sequenziale di diverse competenze analitiche. La varietà tipologica delle Chains risponde a esigenze operative diverse. Le Simple Chains implementano sequenze lineari ottimizzate per performance, mentre le Sequential Chains supportano il passaggio di informazioni tra passi non consecutivi mantenendo stato condiviso. Le Router Chains introducono logica decisionale dinamica, permettendo di determinare percorsi alternativi basandosi sull'input o su condizioni specifiche del contesto operativo. Le Map-Reduce Chains consentono l'elaborazione parallela di input multiple con successiva combinazione dei risultati, architettura particolarmente utile per l'analisi di dataset di grandi dimensioni tipici dell'ambiente industriale. L'introduzione del LangChain Expression Language (LCEL) [30] ha rappresentato un'evoluzione significativa verso un approccio dichiarativo nella costruzione di Chains. LCEL adotta una filosofia che privilegia la descrizione di quello che deve accadere piuttosto che come deve accadere, permettendo al framework di ottimizzare l'esecuzione runtime delle chains. Questa astrazione fornisce un'interfaccia unificata per tutte le tipologie di chains, primitivi per la composizione facilitata, parallelizzazione di componenti, aggiunta di fallback e configurazione dinamica, oltre a supporto nativo per operazioni batch, streaming e asincrone.

Prompts: Gestione intelligente delle istruzioni

I Prompts in LangChain rappresentano molto più di semplici stringhe di testo: sono oggetti intelligenti che incapsulano logica di formattazione, validazione e personalizzazione dinamica delle istruzioni inviate agli LLM. Le classi PromptTemplate in Langhchain sono costruite per fare input dinamici più semplici forniscono:

- **Parametrizzazione dinamica**: Inserimento di variabili nell'template;
- **Validazione dell'input**: Controllo della conformità dei dati inseriti;
- **Formattazione consistente**:Mantenimento di uno stile uniforme
- **Riutilizzabilità**: Template che possono essere utilizzati in contesti diversi

La diversificazione tipologica dei template risponde a esigenze specifiche del dominio applicativo. I ChatPromptTemplate sono ottimizzati per modelli conversazionali, supportando ruoli differenziati (system, user, assistant) e formattazione specializzata di messaggi. I FewShotPromptTemplate implementano strategie di few-shot learning includendo esempi nel prompt per migliorare le performance, mentre i PipelinePromptTemplate permettono la combinazione di template multipli in sequenza per costruire prompt articolati e complessi. Prompt Engineering avanzato L'integrazione di tecniche avanzate di prompt engineering attraverso LangChain facilita l'implementazione di strategie sofisticate come chain-of-thought prompting per guidare il modello attraverso ragionamenti step-by-step, role-based prompting per ottimizzare le risposte attraverso l'assegnazione di ruoli specifici, context injection per l'inserimento dinamico di contesto rilevante e constraint prompting per la specificazione di vincoli e formati di output strutturati. Il LangChain Hub [31] rappresenta un'innovazione ecosistemica che facilita la condivisione e standardizzazione di prompt ottimizzati. Questa piattaforma collaborativa permette agli sviluppatori di condividere template validati, scoprire soluzioni per casi d'uso specifici, accedere a prompt pre-testati per applicazioni comuni e facilitare l'adozione di best practices consolidate dalla community.

Tools: Estensione delle capacità degli LLM

I Tools rappresentano l'innovazione architettonica che trasforma i Large Language Models da generatori di testo isolati in agenti operativi capaci di interagire dinamicamente con l'ecosistema tecnologico circostante. Attraverso l'implementazione di questi strumenti, LangChain supera una delle limitazioni più significative dei modelli linguistici tradizionali: l'incapacità di accedere, manipolare e controllare risorse esterne in tempo reale. L'architettura dei Tools in LangChain si basa su un paradigma che astrae la complessità dell'integrazione sistemica dietro interfacce standardizzate e intuitive. Ogni Tool è progettato come un componente autonomo

caratterizzato da una denominazione semanticamente significativa, una documentazione funzionale completa, uno schema di input rigorosamente tipizzato, logica operativa robusta e capacità di produrre output strutturati e interpretabili.

Function Calling e Tool Calling

Il tool calling rappresenta una tecnica avanzata che permette agli sviluppatori di costruire applicazioni sofisticate capaci di sfruttare le capacità dei Large Language Models per accedere, interagire e manipolare risorse esterne come database, file system e API di servizi. Questo meccanismo trasforma il modello linguistico da processore passivo di testo a orchestratore attivo di operazioni computazionali complesse [19, 18]. L'intelligenza del tool calling risiede nella capacità del modello di riconoscere autonomamente quando una richiesta dell'utente richiede l'utilizzo di strumenti esterni, analizzando il contesto e l'intento per determinare la strategia operativa più appropriata. Una volta identificata la necessità, il sistema implementa un processo decisionale sofisticato per selezionare lo strumento più adatto dal set di risorse disponibili, considerando fattori come pertinenza funzionale, affidabilità operativa e ottimizzazione delle performance. La generazione dei parametri rappresenta una fase critica dove il modello deve estrarre ed elaborare le informazioni necessarie dal contesto conversazionale, trasformandole in argomenti strutturati conformi allo schema di input dello strumento selezionato. Questa capacità richiede una comprensione profonda sia della semantica della richiesta che delle specifiche tecniche dell'interfaccia del tool. L'interpretazione dei risultati restituiti chiude il ciclo operativo, richiedendo al modello di processare output potenzialmente complessi e di integrarli coerentemente nella risposta finale. Questa fase può includere trasformazioni dei dati, sintesi di informazioni multiple e traduzione di formati tecnici in linguaggio comprensibile all'utente finale.

Tassonomia operativa dei Tools

L'ecosistema LangChain supporta una diversificata tassonomia di strumenti progettati per rispondere a specifiche esigenze operative e domini applicativi. Gli API Tools costituiscono la categoria più ampia, fornendo interfacce standardizzate per l'interazione con servizi web attraverso protocolli REST, GraphQL e altri standard di comunicazione. Questi strumenti gestiscono automaticamente aspetti complessi come autenticazione, gestione delle sessioni, rate limiting e retry logic, astraendo la complessità tecnica dietro interfacce semplificate. I Database Tools specializzano l'accesso a sistemi di gestione dati, implementando capacità di query SQL dynamically generated, operazioni CRUD ottimizzate e gestione intelligente di transazioni. L'integrazione nativa con diversi engine di database garantisce portabilità e flessibilità operativa, mentre meccanismi avanzati di connection pooling e caching ottimizzano le performance in scenari ad alto throughput. I File System

Tools estendono le capacità operative all'ecosistema di storage locale e distribuito, supportando operazioni su file e strutture di directory, gestione di encoding multipli per documenti testuali, parsing automatico di formati strutturati e implementazione di controlli di sicurezza granulari. Particolare attenzione è dedicata alla gestione di permessi e alla prevenzione di vulnerabilità di path traversal. I Web Tools abilitano capacità di web scraping intelligente con rispetto automatico dei file robots.txt, navigazione automatica di interfacce web complesse con supporto JavaScript, estrazione strutturata di contenuti e gestione sofisticata di sessioni e cookies. L'implementazione di meccanismi anti-detection e throttling automatico garantisce operazioni sostenibili e etiche. I Mathematical Tools forniscono capacità computazionali avanzate attraverso calcolatori scientifici ad alta precisione, risolutori simbolici per equazioni complesse, funzionalità di analisi statistica e data science, e integrazione nativa con ecosistemi computazionali specializzati come NumPy, SciPy e SymPy. Questi strumenti trasformano il modello linguistico in un potente assistente per calcoli tecnici e scientifici. I Custom Tools rappresentano la categoria più flessibile, fornendo framework per l'implementazione di logiche specifiche del dominio applicativo. Questa categoria supporta integrazione con sistemi proprietari, creazione di wrapper per legacy systems e implementazione di workflow specializzati per contesti industriali specifici.

Best Practices per Tool Design

La progettazione efficace di Tools richiede l'aderenza a principi consolidati che garantiscano usabilità, affidabilità e manutenibilità. La ricerca empirica ha dimostrato che modelli con API esplicite di tool-calling raggiungono performance superiori rispetto a implementazioni generic non specializzate, evidenziando l'importanza di progettazione mirata e ottimizzazione specifica. L'adozione di nomenclatura semantica rappresenta un fattore critico per l'efficacia operativa. I nomi dei tools devono comunicare immediatamente e chiaramente la funzionalità implementata, utilizzando terminologia coerente con il dominio applicativo e convenzioni standard del settore. La ricerca ha evidenziato correlazioni positive tra qualità della denominazione e accuratezza della selezione automatica da parte dei modelli linguistici. La documentazione operativa deve bilanciare completezza informativa e concisione, fornendo descrizioni che permettano al modello di comprendere precisamente quando e come utilizzare lo strumento, includendo esempi d'uso, limitazioni operative e prerequisiti sistemici. L'integrazione di metadati strutturati facilita discovery automatico e categorizzazione intelligente. Il design modulare privilegia tools con scope funzionale limitato e ben definito rispetto a implementazioni monolitiche multiuso. Questo approccio migliora l'interpretabilità per i modelli linguistici, facilita testing e debugging, riduce la complessità di manutenzione e aumenta la riutilizzabilità cross-progetto. La robustezza operativa richiede implementazione di

error handling comprehensivo con gestione granulare di diverse tipologie di errore, logging dettagliato per supportare debugging e monitoring, meccanismi di graceful degradation per mantenere funzionalità parziale in presenza di fallimenti non critici, e timeout configurabili per prevenire blocking indefinito. L'ottimizzazione delle performance deve considerare scalabilità per operazioni ad alto throughput, gestione efficiente di risorse computazionali e di memoria, implementazione di caching intelligente per riduzioni di latenza, e monitoraggio continuo delle metriche operative per identificazione proattiva di bottlenecks. La sicurezza operativa implementa validazione rigorosa di tutti i parametri di input per prevenire injection attacks, sanitizzazione automatica di dati utente, gestione sicura di credenziali e informazioni sensibili, e audit logging per compliance e forensics. Particolare attenzione deve essere dedicata alla prevenzione di vulnerabilità comuni come SQL injection, XSS e path traversal.

Agenti: Coordinamento intelligente di Chain, Prompt e Tools

Gli Agenti rappresentano l'evoluzione più avanzata dell'architettura LangChain, combinando chains, prompts e tools in sistemi autonomi capaci di ragionamento e decisione. Un agente LangChain è composto da:

- **Agent Executor**: Orchestratore principale che coordina le operazioni
- **Agent Logic**: Logica decisionale che determina le azioni da intraprendere
- **Tool Set**: Insieme di strumenti disponibili per l'agente
- **Memory**: Sistema di memoria per mantenere il contesto delle operazioni

La diversificazione degli agenti risponde a paradigmi operativi specifici. I ReAct Agents implementano pattern di ragionamento-azione per risolvere problemi complessi attraverso cicli iterativi di analisi e intervento. Gli OpenAI Functions Agents sfruttano le capacità native di function calling dei modelli OpenAI per ottimizzazioni performance-specific. Gli Structured Chat Agents sono progettati per conversazioni strutturate con output formattati, mentre i Self-Ask Agents implementano tecniche di auto-interrogazione per migliorare la qualità del ragionamento.

Il processo decisionale segue tipicamente questi passaggi:

- **Analisi dell'input**: Comprensione della richiesta dell'utente
- **Pianificazione**: Determinazione della strategia di risoluzione
- **Selezione tool**: Scelta degli strumenti appropriati
- **Esecuzione**: Utilizzo dei tool selezionati

- **Valutazione**: Analisi dei risultati ottenuti
- **Iterazione**: Ripetizione del processo se necessario

Figura 4.2: Applicazioni di LangChain

4.2 LangGraph: orchestrazione a grafo e state management

Introduzione a LangGraph

L'evoluzione delle applicazioni basate su Large Language Models ha evidenziato i limiti intrinseci delle architetture sequenziali tradizionali quando applicate a scenari operativi complessi che richiedono flussi dinamici, processi iterativi e coordinamento tra agenti multipli. In risposta a questa sfida, LangGraph [32] emerge come estensione naturale dell'ecosistema LangChain, introducendo un paradigma di orchestrazione basato su grafi che trasforma radicalmente l'approccio alla progettazione di sistemi AI conversazionali e multi-agente. LangGraph rappresenta una libreria open-source specificatamente progettata per semplificare l'orchestrazione di flussi conversazionali e task complessi attraverso rappresentazioni a grafo che superano le limitazioni delle sequenze lineari. Mentre LangChain eccelle nella gestione di chains sequenziali predefinite, LangGraph [33, 34] introduce una struttura architettonica più flessibile dove i nodi possono essere collegati attraverso logiche condizionali e dinamiche, permettendo la rappresentazione accurata di processi iterativi, ciclici e intrinsecamente non lineari. Paradigma architetturale basato su grafi La transizione dal modello sequenziale al paradigma a grafo rappresenta un

salto concettuale significativo nell'orchestrazione di sistemi AI. LangGraph modella i processi complessi come grafi orientati (Directed Graphs) dove ogni componente del sistema assume un ruolo specifico e ben definito. I nodi rappresentano unità operative discrete che possono eseguire task specifici, invocare modelli linguistici o attivare strumenti specializzati. Gli archi implementano la logica di transizione tra i nodi, incorporando regole condizionali che possono dipendere dallo stato corrente del sistema o da risultati di elaborazioni precedenti. Questa architettura si rivela particolarmente efficace per una varietà di scenari applicativi che richiedono flessibilità operativa: Workflow conversazionali complessi dove la sequenza delle interazioni non può essere predeterminata ma deve adattarsi dinamicamente al contesto e alle risposte dell'utente. La capacità di implementare branching decisionale permette di creare esperienze conversazionali che si adattano intelligentemente alle esigenze specifiche di ogni interazione. Sistemi multi-agente coordinati dove agenti specializzati devono collaborare per raggiungere obiettivi comuni, condividendo informazioni e coordinando le proprie azioni attraverso meccanismi di comunicazione strutturati. La rappresentazione a grafo facilita la modellazione di pattern di collaborazione complessi e la gestione delle dipendenze tra agenti. Task distribuiti multi-fase che richiedono verifica, fallback o riformulazione based su risultati intermedi. La possibilità di implementare cicli controllati e percorsi alternativi garantisce robustezza operativa anche in presenza di condizioni impreviste o risultati subottimali.

Architettura: nodi, archi e stato

L'architettura di LangGraph si articola attorno a tre elementi fondamentali che collaborano per creare un sistema di orchestrazione sofisticato e flessibile. I Nodes costituiscono le unità operative discrete del sistema, implementando funzionalità specifiche che possono variare dalla semplice elaborazione testuale a complesse operazioni di analisi o interfacciamento con sistemi esterni. Ogni nodo è progettato per essere autonomo e riutilizzabile, facilitando la composizione di workflow complessi attraverso la combinazione di componenti specializzati. Gli Edges definiscono la logica di connessione tra i nodi, implementando regole che determinano il flusso di esecuzione attraverso il grafo. Questi collegamenti possono essere statici per rappresentare sequenze predefinite, oppure dinamici quando incorporano logica condizionale che valuta lo stato corrente del sistema o i risultati di elaborazioni precedenti. La flessibilità degli edges permette di implementare pattern decisionali sofisticati che adattano il comportamento del sistema alle condizioni operative correnti. Lo State rappresenta il cuore del sistema di coordinamento, implementando un dizionario condiviso che mantiene contesto e informazioni accessibili a tutti i nodi del grafo. Questo stato globale facilita la comunicazione tra componenti e permette la persistenza di informazioni critiche attraverso l'intero workflow di elaborazione.

State management: la chiave per il coordinamento

Il sistema di gestione dello stato in LangGraph rappresenta una delle innovazioni più significative rispetto alle architetture tradizionali, implementando un modello di memoria condivisa che abilita coordinamento sofisticato tra componenti distribuiti. Ogni nodo del grafo può leggere lo stato corrente per accedere a risultati di elaborazioni precedenti, risposte dell'utente o contesto conversazionale accumulato. Parallelamente, ogni nodo può aggiornare lo stato aggiungendo nuovi dati, memorizzando output intermedi o registrando informazioni di controllo del workflow. Gli archi utilizzano attivamente lo stato per implementare logica decisionale dinamica, valutando condizioni specifiche per determinare quale nodo attivare successivamente. Questa capacità permette di creare workflow adattivi che modificano il proprio comportamento based su risultati intermedi o cambiamenti nel contesto operativo. Lo stato svolge funzioni multiple all'interno dell'architettura:

- Contesto condiviso tra tutti i nodi del grafo, facilitando la comunicazione e la condivisione di informazioni rilevanti senza necessità di passaggio esplicito di parametri tra componenti.
- Tracciamento del workflow attraverso la registrazione di metriche operative come numero di iterazioni eseguite, step completati con successo, errori incontrati e performance temporali.
- Memoria persistente per agenti che devono mantenere informazioni attraverso sessioni estese o multiple interazioni, implementando capacità cognitive che simulano processi di memoria a lungo termine.

Integrazione con LangChain e agenti

LangGraph è progettato come estensione complementare dell'ecosistema LangChain piuttosto che sostituto, mantenendo piena compatibilità con componenti esistenti mentre estendendo significativamente le capacità di orchestrazione. I nodi del grafo possono incorporare chains LangChain, prompt template, tool calls o agent executors, permettendo il riutilizzo di investimenti in sviluppo esistenti all'interno di architetture più sofisticate. La capacità di orchestrare agenti diversi che collaborano attraverso lo scambio di informazioni tramite lo stato condiviso abilita l'implementazione di pattern operativi avanzati. Il pattern Planner-Executor-Supervisor rappresenta un esempio paradigmatico dove un nodo Planner analizza il task e aggiorna lo stato con un piano d'azione strutturato, un nodo Executor implementa le azioni pianificate registrando risultati nello stato, e un nodo Supervisor valuta i risultati per decidere se terminare il processo o attivare cicli aggiuntivi di pianificazione ed esecuzione. Il supporto per pattern di riflessione e auto-correzione permette di implementare sistemi che valutano criticamente i propri output e

attivano meccanismi di miglioramento iterativo. Questa capacità è particolarmente rilevante in contesti industriali dove l'accuratezza e l'affidabilità delle decisioni sono requisiti critici.

Caratteristiche avanzate

LangGraph incorpora funzionalità avanzate progettate specificamente per supportare l'implementazione di sistemi enterprise-grade che richiedono robustezza, scalabilità e controllo operativo granulare. I Pydantic state models permettono di definire lo stato come oggetto tipizzato e validato, garantendo consistenza dei dati e facilitando debugging e manutenzione del codice. Questa tipizzazione forte riduce significativamente la probabilità di errori runtime e migliora la qualità complessiva del sistema. Il step limiting implementa meccanismi di controllo che terminano automaticamente il workflow dopo un numero predefinito di iterazioni, prevenendo loop infiniti e garantendo comportamento prevedibile anche in presenza di condizioni impreviste. L'asynchronous execution supporta task asincroni e streaming, ottimizzando l'utilizzo delle risorse computazionali e migliorando la responsiveness del sistema. I custom edges permettono l'implementazione di logica condizionale arbitrariamente complessa per le transizioni tra nodi, supportando scenari operativi sofisticati che richiedono decision-making basato su multiple variabili o condizioni composite. Il tracciamento del flusso fornisce logging dettagliato di tutti i passaggi eseguiti, facilitando debugging, audit e ottimizzazione delle performance.

Use cases

Esempi concreti dove LangGraph è particolarmente indicato:

- Chatbot multiagente che interagiscono tra loro.
- Sistemi di Q&A che prima pianificano le query, poi aggregano le risposte.
- Analisi di documenti multi-step: parsing, estrazione, verifica.
- Pipeline di visione artificiale in cui la decisione su quali immagini elaborare deriva dai risultati precedenti.
- Workflow di approvazione automatizzati, con cicli di verifica e fallback.

4.3 Differenze tra LangChain e LangGraph

LangChain e LangGraph rappresentano due approcci complementari all'interno dello stesso ecosistema tecnologico, progettati per semplificare la costruzione di applicazioni basate su modelli linguistici attraverso filosofie architetturali che, pur

condividendo principi fondamentali di componibilità e modularità, differiscono significativamente nei paradigmi di orchestrazione e gestione dei flussi operativi. La comprensione di queste differenze è essenziale per operare scelte architetturali informate che ottimizzino l'efficacia delle soluzioni sviluppate in funzione delle specifiche esigenze applicative e dei vincoli operativi del contesto di deployment. L'analisi comparativa di queste tecnologie rivela differenze strutturali che vanno ben oltre semplici variazioni implementative, toccando aspetti fondamentali dell'architettura software come paradigmi di controllo del flusso, modelli di gestione dello stato, strategie di scalabilità e approcci al coordinamento di componenti distribuiti. Queste differenze non rappresentano limitazioni o superiorità relative, ma piuttosto specializzazioni che rendono ciascuna tecnologia più adatta a specifici domini applicativi e scenari operativi

4.3.1 Struttura del flusso: pipeline lineare e grafo dinamico

LangChain adotta un modello architetturale basato su pipeline lineari che organizza i passaggi operativi in catene prestabilite e sequenziali. Questo approccio riflette paradigmi consolidati dell'ingegneria del software, dove la scomposizione di problemi complessi in sequenze di operazioni atomiche facilita la comprensione, il testing e la manutenzione del codice. L'architettura lineare si rivela particolarmente efficace per flussi operativi dove l'input deve percorrere invariabilmente gli stessi step in un ordine predeterminato, come nel caso di chatbot tradizionali, pipeline di retrieval e generazione semplici, o workflow di elaborazione documentale standardizzati. LangGraph introduce un paradigma radicalmente diverso basato su grafi diretti dove i nodi possono attivare altri nodi, implementare cicli controllati o seguire percorsi alternativi in funzione di condizioni valutate runtime. Questa struttura permette l'orchestrazione di logiche significativamente più complesse, includendo meccanismi di fallback multipli, branching condizionale sofisticato e cicli di riflessione e ripianificazione che adattano dinamicamente il comportamento del sistema alle condizioni operative correnti.

4.3.2 Gestione dello stato: passaggio step by step e contesto condiviso

La trasmissione dei dati in LangChain avviene attraverso un modello sequenziale dove ogni step riceve un input specifico, applica le proprie trasformazioni e produce un output che diventa automaticamente l'input del passaggio successivo. Questo pattern garantisce flussi di dati predicibili e facilita il debugging attraverso l'isolamento delle responsabilità, ma può risultare limitante quando multiple componenti necessitano di accedere simultaneamente alle stesse informazioni o quando il flusso operativo richiede ramificazioni condizionali complesse. LangGraph implementa un

Tabella 4.1: Confronto tra LangChain e LangGraph

Aspetto	LangChain	LangGraph
Modello principale	Pipeline lineare di step sequenziali.	Grafo diretto con nodi e archi dinamici.
Flusso operativo	Statico, predeterminato in fase di design.	Dinamico: branching, cicli e fallback runtime.
Complessità gestibile	Semplice o moderata.	Elevata, ottimizzato per sistemi complessi.
Prevedibilità	Alta: comportamento deterministico.	Variabile: adattivo alle condizioni.
Facilità di debug	Elevata: flusso lineare tracciabile.	Moderata: richiede strumenti specializzati.

modello di stato globale centralizzato, tipicamente realizzato attraverso strutture dati condivise come dizionari Python o modelli Pydantic validati. Ogni nodo del grafo può leggere o modificare questo contesto condiviso, mantenendo memoria persistente dei processi, delle decisioni prese e dei risultati intermedi. Questa architettura è fondamentale per sistemi multi-agente che richiedono coordinamento sofisticato e per workflow che devono adattarsi dinamicamente based su feedbacks continui.

Tabella 4.2: Modelli di gestione dello stato

Caratteristica	LangChain	LangGraph
Modello di stato	Input/output sequenziale tra step.	Stato globale centralizzato condiviso.
Accessibilità dati	Limitata al singolo passaggio.	Ogni nodo accede al contesto completo.
Persistenza informazioni	Limitata alla catena di passaggio.	Memoria persistente cross-nodo.
Coordinamento componenti	Sequenziale e predeterminato.	Dinamico e bidirezionale.
Complessità implementativa	Bassa: gestione automatica.	Media: richiede design attento.

4.3.3 Controllo del flusso: statico e dinamico

L'integrazione con agenti e strumenti presenta caratteristiche distintive in entrambe le piattaforme. Mentre LangChain e LangGraph supportano l'integrazione di LLM, retriever, tool esterni e chiamate API, LangGraph eccelle nell'orchestrazione di agenti multipli all'interno dello stesso grafo operativo. Pattern architetturali

come planner-executor-supervisor, dove un agente pianificatore definisce strategie operative, un esecutore implementa le azioni pianificate e un supervisore monitora i risultati decidendo se procedere, modificare il piano o attivare meccanismi di recovery, trovano implementazione naturale e elegante in LangGraph. LangChain richiede invece approcci più manuali per la gestione multi-agente, tipicamente attraverso l'esecuzione sequenziale di catene dedicate per ogni agente, con coordinamento gestito a livello applicativo piuttosto che architetturale. Questo approccio può risultare più complesso da implementare e mantenere quando il numero di agenti e le loro interazioni crescono in complessità. L'analisi dei casi d'uso tipici rivela specializzazioni naturali per ciascuna piattaforma, con pattern di adozione che riflettono le caratteristiche architetturali distintive di ogni framework.

Tabella 4.3: Confronto casi d'uso LangChain vs LangGraph

Caso d'uso	**LangChain**	**LangGraph**	**Note implementative**
Chatbot tradizionali	Ottimale	Sovradimensionato	Flussi conversazionali lineari.
Sistemi multi-agente	Limitato	Ottimale	Coordinamento nativo tra agenti.
RAG base	Ideale	Eccessivo	Pipeline retrieval-generation semplice.
RAG avanzati con fallback	Complesso	Nativo	Strategie adaptive di recupero.
Workflow lineari	Perfetto	Overhead	Elaborazione sequenziale documentale.
Workflow dinamici e ciclici	Non supportato	Core feature	Branching e loop controllati.
Prototipazione rapida	Eccellente	Lento	Time-to-market prioritario.
Applicazioni enterprise complesse	Limitato	Scalabile	Sistemi adattivi e robusti.
Elaborazione batch	Efficiente	Sovradimensionato	Pipeline di trasformazione dati.
Sistemi real-time adattativi	Rigido	Ottimale	Reattività a condizioni dinamiche.

Le considerazioni di complessità e scalabilità evidenziano trade-off significativi tra le due piattaforme che devono essere valutati nel contesto delle specificità del progetto e delle competenze del team di sviluppo.

La scelta tra LangChain e LangGraph deve considerare il lifecycle completo del progetto, includendo non solo i requisiti iniziali ma anche l'evoluzione prevista delle funzionalità, la crescita del team di sviluppo, le esigenze di manutenzione e le aspettative di scalabilità.

Progetti che iniziano con requirement semplici ma prevedono crescita in complessità possono beneficiare di una strategia di migration graduale, iniziando con LangChain per rapid prototyping e evolvendo verso LangGraph quando la complessità raggiunge soglie che giustificano l'investimento architetturale aggiuntivo. Questo

Tabella 4.4: Matrice decisionale per la selezione del framework

Criterio di valutazione	Quando scegliere LangChain	Quando scegliere LangGraph
Complessità requisiti	Workflow lineari e predicibili.	Logiche complesse e adattive.
Timeline e configurazione	Rapid prototyping, configurazione semplice.	Disponibile tempo per design attento.
Competenze team	Esperienza base con LLM.	Expertise in sistemi distribuiti.
Scalabilità e manutenibilità	Crescita incrementale limitata, manutenibilità media.	Evoluzione verso complessità elevata, alta manutenibilità long-term.
Coordinamento agenti	Single-agent o coordinamento semplice.	Multi-agent con interdipendenze.
Budget e risorse	Constraints significativi, resource requirements bassi.	Investimento in architettura robusta, resource requirements medi-alti.
Debugging e tooling	Debug semplice, flusso lineare tracciabile.	Richiede tooling specializzato.
Risk tolerance	Preferenza per soluzioni consolidate.	Apertura a tecnologie innovative.
Strategia di manutenzione	In-house team piccolo.	Team dedicato o outsourcing specializzato.

approccio ibrido permette di ottimizzare il time-to-market iniziale mantenendo la flessibilità per evoluzioni architetturali future.

4.4 Architettura del Coordinator Agent e meccanismo di reasoning

Il Coordinator Agent rappresenta il nucleo strategico dell'intero sistema multi-agente, responsabile dell'interpretazione delle richieste utente e dell'orchestrazione dinamica delle risorse specializzate. La sua implementazione segue un pattern architetturale a stati che garantisce robustezza decisionale e tracciabilità operativa.

4.4.1 Struttura e stati operativi del Coordinator

Il Coordinator implementa tre stati operativi principali, ciascuno con responsabilità specifiche e output strutturati:

ReflectNode: Rappresenta la fase di analisi contestuale e interpretazione delle richieste. Il nodo implementa capacità di natural language understanding per

classificare il tipo di supporto richiesto e valutare la coerenza con eventuali piani in corso.

```
class ReflectNodeOutput(BaseModel):
    next_graph_step: Literal["plan", "supervise", "end"] = Field(
        description="Choice for the next step to take in the graph execution"
    )
    response: str = Field(
        description="User friendly response for the user"
    )
```

Listing 4.1: Output strutturato del ReflectNode

PlanNode: Implementa la logica di pianificazione strategica con una caratteristica distintiva fondamentale: l'integrazione del campo reasoning che documenta esplicitamente la logica decisionale adottata.

```
class PlanNodeOutput(BaseModel):
    reasoning: str = Field(
        description="Brief explanation of why this plan was chosen"
    )
    plan_activities: List[str] = Field(
        description="List of activities to be performed by the agents"
    )
    response: str = Field(
        description="User friendly response for the user"
    )
```

Listing 4.2: Output strutturato del PlanNode con reasoning esplicito

SuperviseNode: Coordina l'esecuzione del piano, monitora l'avanzamento delle attività e gestisce le transizioni tra agenti specializzati.

4.4.2 Meccanismo di reasoning: implementazione e vantaggi

Il campo reasoning rappresenta un'innovazione implementativa significativa che migliora sostanzialmente le capacità decisionali del sistema. La sua implementazione segue principi di "ragionamento prima dell'azione" che hanno dimostrato efficacia nel migliorare la qualità delle decisioni AI. Il reasoning è posizionato come primo campo nell'output strutturato del PlanNode. Questa sequenza non è casuale:

```
1
2 class PlanNodeOutput(BaseModel):
3     reasoning: str = Field(...)  # PRIMO: forza la riflessione
4     plan_activities: List[str] = Field(...)  # SECONDO: pianifica
    in base al reasoning
5     response: str = Field(...)   # TERZO: comunica all'utente
```

Listing 4.3: Ordinamento strategico dei campi per forzare reasoning sequenziale

Forzando il modello a compilare prima il reasoning, si implementa un meccanismo di "forced reflection" che riduce significativamente decisioni impulsive o incoerenti. I Vantaggi operativi del reasoning sono:

1. Miglioramento della qualità decisionale: Posizionando il reasoning come primo campo, si forza il modello a riflettere prima di agire, riducendo decisioni impulsive o incoerenti. Il modello deve esplicitare la propria logica prima di definire il piano di azioni.

2. Debugging facilitato per sviluppatori: Il reasoning viene salvato nei log di sistema, permettendo agli sviluppatori di analizzare retrospettivamente le decisioni prese. Quando il sistema si comporta diversamente dal previsto, il reasoning nei log rivela immediatamente il processo logico seguito.

3. Miglioramento iterativo del sistema: Analizzando i reasoning salvati nello stato del sistema, è possibile identificare pattern decisionali problematici e affinare i system messages. Per esempio, se i log mostrano reasoning ricorrenti come "Presenza di immagine → sempre attivazione VisionAgent", questo indica la necessità di affinare la logica per distinguere tra tipi diversi di analisi visiva.

4. Tracciabilità interna: Il reasoning viene mantenuto nello stato per tracciabilità

Il reasoning viene salvato nello stato del sistema per tracciabilità completa delle decisioni prese durante l'orchestrazione. Questo permette analisi retrospettiva delle logiche adottate e identificazione di opportunità di ottimizzazione.

4.4.3 Logica decisionale e routing intelligente

Il Coordinator implementa regole decisionali sofisticate per il routing delle richieste. Le richieste possono essere classificate in:

- **Input puramente testuale**: Domande teoriche che non richiedono analisi visiva
- **Input con analisi visiva**: Richieste che richiedono elaborazione di immagini o video

- **Input diagnostico**: Richieste che necessitano di analisi tecnica e raccomandazioni

La logica decisionale nel ReflectNode implementa pattern matching sofisticato che analizza il contenuto semantico delle richieste per determinare il percorso ottimale. Le regole includono rilevamento di keyword specifiche, analisi del contesto conversazionale, e valutazione della presenza di attachment multimediali.

4.5 CableVisionAgent: implementazione della pipeline visiva

Il CableVisionAgent implementa una pipeline sofisticata per l'elaborazione di contenuti visivi, integrando computer vision avanzata con analisi semantica guidata da LLM.

4.5.1 WholeImageNode: gateway di sicurezza e controllo preliminare

Prima di attivare elaborazioni costose, il sistema implementa un nodo di controllo che svolge funzioni critiche. Il WholeImageNode implementa due modalità operative distinte determinate dall'analisi del contesto. La modalità "gate" viene attivata quando il sistema rileva intento diagnostico nelle attività pianificate, mentre la modalità "description" risponde a richieste puramente descrittive. La modalità gate implementa controlli automatici per privacy protection utilizzando OpenCV per rilevamento di volti umani, verifica presenza di componenti tecnici rilevanti, e validazione dell'idoneità dell'immagine per analisi diagnostiche. Il sistema produce output standardizzati che bloccano immediatamente l'elaborazione se vengono rilevati contenuti inappropriati. In modalità description, il nodo fornisce analisi descrittiva completa del contenuto visivo, includendo identificazione di componenti, descrizione di caratteristiche visibili, e generazione di metadata strutturati. L'output include l'immagine in formato markdown strutturato per integrazione seamless nella documentazione finale.

4.5.2 VisionNode: segmentazione con EfficientSAM

Il VisionNode rappresenta il core della pipeline di computer vision, implementando segmentazione avanzata attraverso EfficientSAM. Questo nodo coordina l'intero processo di segmentazione, dalla validazione dell'input fino alla generazione dei risultati finali. Il nodo implementa un sistema di estrazione del filename basato su pattern matching utilizzando espressioni regolari (regex). Questa scelta implementativa deriva dalla necessità di identificare con precisione il file immagine da

processare all'interno del messaggio conversazionale dell'utente. Il sistema cerca specificamente il pattern Ánalyze the image ńel contenuto del messaggio, dove la parte tra parentesi cattura il nome del file racchiuso tra virgolette. Questo approccio garantisce identificazione univoca del file anche in presenza di messaggi complessi o contenenti multiple referenze a file diversi. Una volta estratto il filename, il sistema verifica immediatamente l'esistenza fisica del file nel filesystem, costruendo il percorso completo utilizzando l'identificativo della sessione (room_id) per garantire isolamento tra utenti diversi. Questa validazione preventiva evita errori di runtime nelle fasi successive di elaborazione. dettagli implementativi della comunicazione HTTP o della serializzazione dei dati. La pipeline di elaborazione strutturata è suddivisa in:

1. **Validazione input**: Il sistema verifica non solo l'esistenza del file, ma anche la correttezza dei parametri di input e la validità del formato immagine. Questa fase include controlli di integrità che prevengono elaborazioni su file corrotti o non supportati.

2. **Invocazione tool**: La chiamata al CableAnalyzerTool viene effettuata attraverso un'interfaccia standardizzata che gestisce automaticamente la costruzione della richiesta HTTP, l'invio all'endpoint di segmentazione, e la deserializzazione della risposta.

3. **Gestione output**: Il percorso dell'immagine segmentata viene salvato nello stato interno dell'agente (self.agent_state["path"]) per utilizzo da parte dei nodi successivi nella pipeline. Questa persistenza dello stato garantisce continuità informativa attraverso le transizioni del grafo.

4. **Error handling**: Il sistema implementa gestione robusta degli errori che include timeout per elaborazioni prolungate, retry automatici per fallimenti transienti, e routing appropriato verso nodi di recovery in caso di errori irrecuperabili.

EfficientSAM rappresenta un'evoluzione ottimizzata del Segment Anything Model (SAM) di Meta AI, progettata per contesti con vincoli computazionali:

- Zero-shot segmentation: Capacità di segmentare oggetti senza training domain-specific
- Efficienza computazionale: Architettura ottimizzata per deployment in sistemi con risorse limitate
- Flessibilità operativa: Supporto per prompt visuali (marker, punti, bounding box) e testuali
- Integrazione seamless: API standardizzata per integrazione in pipeline multi-agente

4.5.3 AnalysisNode: integrazione visione-linguaggio

L'AnalysisNode implementa l'integrazione tra risultati di segmentazione e analisi semantica attraverso modelli multimodali. Il nodo carica l'immagine segmentata, la converte in formato base64 per compatibilità con LLM, e genera prompt multimodali che combinano istruzioni testuali con contenuto visivo. Il processo utilizza structured output per garantire consistenza del formato di risposta, includendo descrizioni dettagliate della regione segmentata e embedding dell'immagine in formato markdown strutturato. Il risultato viene propagato attraverso il sistema per utilizzo in fasi successive di analisi.

4.5.4 Endpoint EfficientSAM: segmentazione avanzata

L'endpoint EfficientSAM rappresenta il cuore tecnologico della pipeline di segmentazione del sistema, implementato come servizio REST dedicato all'interno dell'execution_server. Questo sottosistema orchestra l'interazione tra il CableAnalyzerTool e il modello EfficientSAM, fornendo un'interfaccia standardizzata per operazioni di segmentazione semantica zeroshot. Il servizio è implementato attraverso FastAPI con un router dedicato che espone funzionalità di segmentazione attraverso l'endpoint /efficientsam/execute. L'architettura segue pattern RESTful consolidati per garantire scalabilità e interoperabilità, permettendo al sistema multi-agente di invocare operazioni di segmentazione attraverso chiamate HTTP standardizzate. La scelta di implementare EfficientSAM come servizio separato deriva da considerazioni architetturali specifiche: isolamento delle dipendenze di machine learning, scalabilità orizzontale del componente più computazionalmente intensivo, e possibilità di upgrade indipendente del modello senza impattare l'intero sistema multi-agente. La funzione core del servizio gestisce una pipeline sofisticata che trasforma un'immagine di input e parametri di prompt in una maschera di segmentazione precisa. Il processo inizia con validazione rigorosa degli input, verificando esistenza dei file e correttezza dei parametri forniti dal CableAnalyzerTool. Una fase cruciale è rappresentata dal preprocessing dell'immagine, che include normalizzazione dei valori pixel, ridimensionamento per compatibilità con il modello, e preparazione dei tensor di input. Parallelamente, il sistema elabora i prompt points forniti dall'utente o, in loro assenza, attiva un modulo di rilevamento automatico dei marker visuali. Quando il VisionNode non fornisce coordinate esplicite, il sistema implementa tecniche di computer vision per identificare automaticamente marker visuali nell'immagine. Questa funzionalità utilizza elaborazione nello spazio colore HSV per rilevare oggetti con caratteristiche cromatiche specifiche, tipicamente frecce rosse, cerchi colorati, o altri indicatori grafici che l'operatore può inserire per segnalare aree di interesse. Il processo di rilevamento calcola centroidi geometrici degli oggetti identificati, filtra risultati spuri basandosi su criteri di area minima, e genera coordinate che vengono utilizzate come prompt per EfficientSAM.

Questa automazione elimina la necessità di interventi manuali precisi, rendendo il sistema utilizzabile anche in contesti operativi dove la selezione di punti esatti non è praticabile. Il sistema utilizza EfficientSAM attraverso un'implementazione ottimizzata che bilancia accuratezza e performance computazionale. Il modello viene inizializzato seguendo un pattern singleton per ridurre overhead di caricamento, mantenendo l'istanza in memoria tra chiamate successive. EfficientSAM opera convertendo l'immagine preprocessata e i prompt points in rappresentazioni interne che guidano il processo di segmentazione. Il modello genera multiple maschere candidate con associated confidence scores, permettendo al sistema di selezionare automaticamente il risultato più affidabile o di fornire alternative per validazione umana. L'endpoint produce multiple rappresentazioni del risultato di segmentazione per supportare diverse modalità di analisi successiva. La maschera binaria primaria fornisce segmentazione pura utilizzabile per elaborazioni computazionali, mentre overlay colorati e contorni evidenziati facilitano interpretazione visuale da parte degli operatori. Il sistema genera inoltre heatmap di confidence che visualizzano la certezza del modello in diverse regioni dell'immagine, fornendo indicatori qualitativi sull'affidabilità della segmentazione prodotta. Questi output vengono salvati in percorsi standardizzati e i metadata associati vengono inclusi nella risposta JSON per tracciabilità completa dell'operazione. L'implementazione integra meccanismi di error handling comprensivi per garantire stabilità in contesti industriali. Il sistema implementa validazione preventiva di file e parametri, gestione efficiente della memoria per elaborazione di immagini large-scale, e timeout automatici per protezione contro elaborazioni che si prolungano eccessivamente. In caso di fallimenti parziali, il servizio implementa strategie di graceful degradation che permettono recupero automatico dove possibile, fornendo feedback diagnostici dettagliati per facilitare troubleshooting operativo.

4.6 DiagnosticAgent: sistema di supporto decisionale avanzato

Il DiagnosticAgent implementa capacità di ragionamento diagnostico attraverso l'integrazione di classificazione automatica, ricerca semantica e generazione di raccomandazioni operative.

4.6.1 ImageDiagnosisNode: classificazione automatica delle anomalie

L'ImageDiagnosisNode coordina il processo di classificazione automatica utilizzando l'AnomalibTool per rilevamento e categorizzazione di difetti in componenti industriali. Il nodo estrae il filename dal messaggio utente, verifica l'esistenza

del file, e invoca il tool specializzato per analisi completa. Il processo memorizza i risultati della classificazione nello stato dell'agente, includendo classe rilevata, confidence scores, anomaly scores, e percorsi delle visualizzazioni generate. Questa informazione strutturata viene utilizzata dai nodi successivi per generazione di report e ricerca di documentazione correlata.

AnomalibTool: interfaccia per classificazione diagnostica

L'AnomalibTool svolge una funzione analoga al CableAnalyzerTool, ma per operazioni diagnostiche. Questo tool connette l'ImageDiagnosisNode all'endpoint Anomalib [35], orchestrando sia il processo di anomaly detection che la classificazione dei difetti. Il tool gestisce automaticamente la comunicazione con l'endpoint /anomalib/execute, che internamente esegue la pipeline dual-model (detection + classification). Il risultato viene strutturato in un oggetto AnomalyToolOutput che include classe predetta, confidence scores, anomaly scores, percorsi delle heatmap, e report diagnostico testuale. Questa astrazione permette al DiagnosticAgent di ottenere diagnosi complete senza dover gestire manualmente le due fasi del processo o i dettagli della comunicazione con i servizi di machine learning.

4.6.2 Integrazione RAG per supporto diagnostico

Il sistema implementa Retrieval-Augmented Generation per accesso intelligente alla knowledge base attraverso il SearchAnomalyNode. Il nodo genera automaticamente query multiple basate sulla classificazione automatica delle anomalie, permettendo ricerca semantica completa nella documentazione tecnica. Il "Multi-Query Retrieval" System implementa ricerca semantica avanzata attraverso:

1. **Query Generation**: Generazione automatica di query multiple per copertura semantica completa
2. **Embedding-based Search**: Ricerca vettoriale in spazio semantico ad alta dimensionalità
3. **Document Bundling**: Aggregazione risultati con metadati per tracciabilità
4. **Context Filtering**: Limitazione ricerca a documentazione validata e condivisa

L'architettura del servizio RAG utilizza un endpoint dedicato che riceve liste di query, esegue embedding generation, effettua similarity search nel database vettoriale, e restituisce document bundles ranked per rilevanza. Il sistema mantiene metadati completi includendo documento sorgente, pagine di riferimento, e context snippets.

4.6.3 Report Generation e output strutturato

Il sistema genera report strutturati differenziati per tipo di analisi: Il ImageReportNode genera un report per classificazione automatica utilizzando template strutturati che includono detected class, confidence scores, anomaly scores, e riferimenti a heatmap visuali. Mentre il SuggestionNode e SuggestionReportNode implementano una pipeline specializzata per generazione di raccomandazioni operative. Il SuggestionNode ricerca nella documentazione Cable Fault Diagnostics.pdf per identificare soluzioni e azioni correttive appropriate, mentre il SuggestionReportNode sintetizza le informazioni in formato actionable per gli operatori. Il processo utilizza structured output per garantire consistenza, include source citations per tracciabilità, e implementa formatting professionale appropriato per documentazione tecnica industriale.

4.6.4 Endpoint Anomalib: rilevamento e classificazione di anomalie

L'endpoint Anomalib implementa un sistema dual-purpose che combina rilevamento automatico di anomalie con classificazione specifica di difetti industriali, rappresentando il nucleo tecnologico delle capacità diagnostiche del sistema. Il servizio integra due modelli di machine learning specializzati in una pipeline unificata accessibile attraverso interfaccia REST. La scelta di implementare detection e classification come processi sequenziali deriva da considerazioni teoriche sulla natura dei problemi di anomaly detection in contesti industriali. Il primo modello opera in modalità unsupervised per identificare deviazioni da pattern normali, mentre il secondo utilizza apprendimento supervisionato per classificazione specifica di difetti conosciuti. Questa architettura dual-model permette al sistema di gestire sia anomalie note che sconosciute: il detection model identifica qualsiasi deviazione significativa dalla normalità, mentre il classification model fornisce categorizzazione specifica quando l'anomalia corrisponde a pattern di difetto noti. L'approccio combinato massimizza sia sensibilità che specificità diagnostica. La prima fase utilizza un modello addestrato per identificare regioni anomale attraverso analisi delle caratteristiche visuali che deviano da distribuzione normale. Il modello opera generando anomaly maps che quantificano il grado di deviazione per ogni pixel dell'immagine, producendo rappresentazioni spaziali dell'anomalia rilevata. Il sistema calcola un anomaly score globale che riassume la severità complessiva della deviazione identificata, fornendo una metrica quantitativa utilizzabile per decisioni automatizzate di soglia. Parallelamente, genera heatmap visuali che localizzano geograficamente le anomalie nell'immagine, facilitando interpretazione da parte degli operatori. Una componente importante è il calcolo automatico di bounding box che circoscrivono le regioni anomale, fornendo coordinate precise utilizzabili per

focalizzare analisi successive o per guidare interventi correttivi mirati. La seconda fase implementa classificazione supervisionata per identificare la tipologia specifica di difetto tra nove categorie predefinite: bent_wire, cable_swap, combined, cut_inner_insulation, cut_outer_insulation, good, missing_cable, missing_wire, e poke_insulation. Questa tassonomia di difetti deriva da analisi empirica di failure modes comuni in sistemi di cablaggio industriale. Il classificatore utilizza architettura ResNet-18 modificata, selezionata per bilanciamento ottimale tra accuratezza e efficienza computazionale in contesti edge computing. Il modello è stato addestrato su dataset domain-specific che include esempi rappresentativi di ciascuna categoria di difetto, con tecniche di data augmentation per migliorare robustezza a variazioni illuminazione, prospettiva, e qualità immagine. L'output include non solo la classe predetta con confidence score associato, ma anche distribuzione probabilistica completa su tutte le categorie, fornendo insight sulla certezza della classificazione e possibili diagnosi alternative.

Il sistema produce visualizzazioni sofisticate che supportano interpretazione dei risultati diagnostici. Le heatmap di anomaly overlaying sull'immagine originale utilizzano mappe colore intuitive che evidenziano regioni problematiche, mentre mantenendo contesto visuale dell'componente analizzato. La generazione di heatmap coinvolge normalizzazione delle anomaly maps, applicazione di colormap percettualmente uniformi, e alpha blending con l'immagine originale per preservare dettagli strutturali. Il risultato finale fornisce rappresentazione visuale immediata che facilita diagnosi rapida da parte degli operatori. L'endpoint è progettato per integrazione seamless con l'AnomalibTool utilizzato dal DiagnosticAgent. L'interfaccia unificata nasconde la complessità della pipeline dual-model, presentando risultati consolidati attraverso il modello AnomalyToolOutput che include tutti i parametri diagnostici rilevanti. La comunicazione avviene attraverso richieste HTTP POST strutturate che includono identificativi di sessione, percorsi file, e parametri di configurazione. Il sistema gestisce automaticamente coordinate spaziali, percorsi di output, e metadata operazionali, permettendo al DiagnosticAgent di focalizzarsi sulla logica di orchestrazione piuttosto che su dettagli implementativi.

4.7 Pattern architetturali e gestione degli stati

L'implementazione segue pattern consolidati per garantire robustezza e manutenibilità.

State Management distribuito

Ogni agente mantiene stato interno che persiste informazioni critiche tra transizioni. Il state management implementa un modello distribuito dove ogni agente è responsabile del proprio stato locale, mentre il Coordinator mantiene stato globale

per coordinamento delle attività. Lo stato viene propagato attraverso il sistema utilizzando strutture dati standardizzate che includono plan activities, current activity index, e agent-specific data. Questo approccio garantisce isolamento tra agenti mentre permettendo coordinamento efficace delle attività complesse.

Gestione degli errori e recovery

Il sistema implementa strategie di fault tolerance progettate per mantenere operatività anche in presenza di malfunzionamenti. L'architettura adotta un approccio multilivello dove ogni componente gestisce autonomamente i propri errori specifici, mentre il sistema complessivo coordina strategie di recovery globali.

La gestione degli errori inizia con validation preventiva degli input, dove il sistema verifica esistenza dei file e correttezza dei parametri prima di avviare elaborazioni computazionalmente costose. Quando si verificano errori di validazione, come l'assenza di un file immagine richiesto (`FileNotFoundError` per "motore.jpg" non trovato), il sistema utilizza conditional routing per reindirizzare l'esecuzione verso nodi appropriati per la raccolta di input corretti (`next_graph_step: reflect`), evitando terminazioni abrupte del flusso conversazionale.

Per i fallimenti dei servizi esterni, il sistema implementa circuit breaker patterns che rilevano automaticamente servizi non disponibili e attivano percorsi alternativi. Quando l'endpoint di segmentazione EfficientSAM non risponde (`ConnectionError` dopo timeout di 30 secondi), il sistema può reindirizzare verso il `WholeImageNode` per analisi descrittiva generale, mantenendo funzionalità utili all'utente anche con capacità ridotte. Questa graceful degradation garantisce che il sistema continui a operare con subset delle funzionalità originali piuttosto che bloccarsi completamente.

Il sistema distingue tra errori transienti e permanenti attraverso pattern matching delle exception types, applicando retry automatici con exponential backoff per errori temporanei come timeout di rete (retry automatico dopo 1s, 2s, 4s), mentre errori permanenti come rilevamento di volti umani nel `WholeImageNode` vengono gestiti immediatamente con terminazione del flusso ("Analysis interrupted: contains personal data").

Observability e monitoring

L'architettura integra comprehensive observability attraverso structured logging e distributed telemetry che forniscono visibilità completa sul comportamento interno del sistema. Il logging strutturato utilizza formato JSON con metadata contestuali che permettono correlation tracking attraverso i diversi componenti del sistema multi-agente, facilitando debugging di interazioni complesse tra agenti diversi. Ad esempio, ogni operazione del VisionNode genera log correlati: [VisionNode] Processing image: motore.jpg → [CableAnalyzerTool] Segmentation request sent

→ [EfficientSAM] Processing completed in 8.2s. Il monitoraggio delle performance utilizza instrumentation automatica che traccia latency e throughput di ogni componente critico, permettendo identificazione proattiva di bottlenecks prima che impattino l'esperienza utente. Il sistema rileva automaticamente quando la segmentazione EfficientSAM supera i 15 secondi di processing time (soglia normale: 5-10 secondi) e attiva alerting per investigare potenziali problemi di performance o sovraccarico del servizio. Health monitoring continuo verifica periodicamente operatività di tutti i servizi esterni attraverso synthetic transaction che simulano richieste reali. Ogni minuto vengono inviate richieste di test agli endpoint /efficientsam/execute e /anomalib/execute; se un endpoint fallisce per tre controlli consecutivi, viene marcato come non disponibile e il sistema attiva automaticamente fallback routines che bypassano il servizio compromesso. L'approccio di monitoring proattivo facilita maintenance predittiva e permette interventi preventivi che riducono significantly downtime non pianificati, requisito fondamentale per deployment in contesti industriali critici dove interruzioni del servizio possono impattare operazioni produttive critiche.

Capitolo 5

Risultati Sperimentali e Validazione

Il presente capitolo presenta i risultati della validazione sperimentale del sistema multi-agente sviluppato, documentando metodologie di testing, risultati ottenuti e analisi delle performance. L'obiettivo è fornire evidenze quantitative e qualitative dell'efficacia del sistema in scenari operativi rappresentativi del contesto industriale target.

5.1 Metodologia e Setup Sperimentale

La validazione del sistema è stata condotta seguendo un approccio sistematico strutturato in fasi successive che garantiscono copertura completa delle funzionalità e robustezza dei risultati.

5.1.1 Approccio metodologico Sistemico

Il processo di validazione si articola in tre fasi principali:

- **Fase 1 - Progettazione dei test case**: Identificazione di scenari critici derivati dai requisiti funzionali, con particolare attenzione alle interazioni multi-agente e ai casi d'uso industriali realistici. La selezione ha seguito criteri di copertura funzionale (ogni agente testato in scenari rappresentativi), rappresentatività operativa (situazioni verosimili per operatori industriali), e stress testing per condizioni limite (input incompleti, richieste ambigue).

- **Fase 2 - Esecuzione controllata**: Testing in ambiente isolato e strumentato che consente monitoraggio real-time delle interazioni inter-agente e raccolta

dettagliata delle metriche operative. Ogni test segue un protocollo standardizzato che include reset dell'ambiente, somministrazione controllata dell'input, monitoraggio dell'esecuzione e raccolta sistematica dei risultati.

- **Fase 3 - Analisi dei risultati**: Valutazione quantitativa attraverso metriche standardizzate e analisi qualitativa per identificazione di pattern comportamentali, punti di forza e aree di miglioramento.

5.1.2 Ambiente di testing e dataset

Infrastruttura software: Framework di orchestrazione multi-agente con monitoring integrato, database PostgreSQL per logging strutturato, sistema di monitoraggio real-time per acquisizione continua delle metriche, ambiente isolato per garantire ripetibilità. La costruzione del dataset di validazione ha seguito criteri di rappresentatività e diversità per garantire una copertura completa degli scenari operativi: Le componenti visive sono:

- 75 immagini di cavi industriali estratte dal dataset MVTec AD [36], riconosciuto standard nell'ambito dell'anomaly detection industriale
- Suddivisione strategica: 50 immagini contenenti difetti documentati e 25 immagini di controllo rappresentanti componenti in condizioni operative normali
- Varietà tipologica: Inclusione di diverse tipologie di difetti (usura, rotture, deformazioni, contaminazioni) per testare la robustezza dell'analisi visiva

Le basi di conoscenza tecnica sono:

- Documenti tecnici strutturati, includenti standard internazionali (IEC, ISO), procedure operative aziendali e database diagnostici specializzati
- Documentazione multi-livello: da specifiche tecniche dettagliate a procedure operative semplificate per operatori di diverso livello di esperienza

Gli Input Testuali Simulati sono:

- Collezione di descrizioni testuali progettate per simulare richieste diagnostiche realistiche
- Casi di interrogazione della knowledge base che riflettono pattern di utilizzo tipici degli operatori industriali
- Scenari di ambiguità controllata per testare le capacità di chiarimento interattivo del sistema

5.1.3 Framework di Valutazione Multi-dimensionale

Il framework di valutazione utilizza criteri multi-dimensionali:

Metriche di Correttezza Funzionale

La valutazione della correttezza funzionale rappresenta il fondamento della validazione sistemica, concentrandosi sulla capacità del sistema di produrre output accurati e affidabili in ogni fase del processo elaborativo. In questo ambito, l'accuratezza del routing costituisce un aspetto cruciale: il coordinator agent deve dimostrare la capacità di interpretare correttamente l'intento dell'utente e instradare la richiesta verso l'agente specializzato più appropriato. Questa valutazione diventa particolarmente complessa quando si considerano richieste ambigue o multi-modali, dove la capacità di discriminazione semantica del sistema viene messa alla prova in scenari che richiedono interpretazione contestuale sofisticata. La correttezza delle analisi visive costituisce un altro pilastro fondamentale della valutazione, richiedendo un assessment approfondito della precisione con cui il sistema descrive componenti meccanici, identifica difetti e localizza anomalie. Questa valutazione garantisce che le conclusioni raggiunte dal sistema siano non solo tecnicamente corrette ma anche praticamente utili per gli operatori industriali. Parallelamente, la precisione delle diagnosi formulate dal sistema richiede un'analisi multidimensionale che va oltre la semplice correttezza binaria. Il confidence scoring associato a ogni diagnosi deve essere calibrato accuratamente per fornire agli operatori un'indicazione affidabile dell'affidabilità della conclusione raggiunta. La coerenza tra evidenze visive disponibili e diagnosi finale rappresenta un aspetto critico di questa valutazione, assicurando che il sistema mantenga un approccio evidence-based coerente. La completezza delle raccomandazioni fornite agli operatori chiude il quadro della valutazione funzionale, richiedendo un'analisi che consideri non solo la correttezza tecnica ma anche l'utilità pratica delle indicazioni fornite. Le raccomandazioni devono essere non solo accurate ma anche actionable, fornendo agli operatori informazioni concrete e immediatamente utilizzabili nel contesto operativo.

Metriche di Performance Operativa

Le prestazioni operative del sistema costituiscono un aspetto critico per l'applicabilità industriale della soluzione, richiedendo una valutazione accurata dei tempi di risposta end-to-end per diverse tipologie di richieste. Questa analisi deve considerare non solo i valori medi ma anche la distribuzione temporale delle risposte, identificando potenziali colli di bottiglia che potrebbero compromettere l'efficienza operativa in scenari reali. La variabilità dei tempi di risposta è particolarmente importante in contesti industriali dove la prevedibilità delle prestazioni è spesso più importante della velocità assoluta. Il monitoraggio dell'utilizzo delle risorse

computazionali fornisce insight cruciali sulla sostenibilità operativa del sistema, richiedendo un'analisi dettagliata dell'utilizzo di CPU, memoria, storage e risorse GPU durante l'esecuzione. Particolare attenzione deve essere dedicata all'identificazione di picchi di utilizzo che potrebbero indicare inefficienze algoritmica o necessità di ottimizzazione. La stabilità delle prestazioni nel tempo rappresenta un indicatore fondamentale della maturità tecnologica della soluzione. L'efficienza del processo decisionale nell'orchestrazione multi-agente richiede un'analisi sofisticata dei pattern di comunicazione inter-agente e dell'overhead di coordinamento. Questa valutazione deve considerare non solo l'efficienza computazionale ma anche l'efficacia comunicativa, assicurando che la complessità dell'architettura multi-agente non comprometta le prestazioni complessive del sistema.

Metriche di Robustezza Sistemica

La robustezza sistemica rappresenta un requisito fondamentale per l'applicazione industriale, richiedendo una valutazione approfondita della capacità del sistema di mantenere funzionalità operative anche in presenza di condizioni avverse. La gestione degli errori deve dimostrare la capacità del sistema di identificare, classificare e gestire diverse tipologie di errore senza compromettere la stabilità sistemica complessiva. I meccanismi di recovery automatico costituiscono un aspetto cruciale della robustezza sistemica, richiedendo test specifici delle strategie implementate per il recupero da condizioni di errore. Le strategie di retry con backoff esponenziale, i meccanismi di fallback e le procedure di degradazione controllata devono essere validate attraverso scenari di stress mirati che simulino condizioni operative critiche.

Metriche di Usabilità Operativa

L'usabilità operativa rappresenta il ponte tra la correttezza tecnica del sistema e la sua accettabilità da parte degli operatori industriali. La chiarezza delle risposte fornite richiede una valutazione qualitativa approfondita che consideri non solo la correttezza tecnica ma anche la comprensibilità e la completezza delle informazioni fornite. Questa valutazione deve tenere conto del contesto operativo specifico e delle competenze tecniche degli operatori target. L'appropriatezza del linguaggio tecnico utilizzato dal sistema rappresenta un aspetto critico per l'adozione operativa, richiedendo un bilanciamento accurato tra precisione tecnica e accessibilità comunicativa. Il sistema deve essere in grado di adattare il registro linguistico al contesto applicativo e al livello di competenza dell'utente, mantenendo l'accuratezza delle informazioni tecniche mentre assicura la comprensibilità per operatori con diversi livelli di esperienza. La gestione di richieste ambigue costituisce un test delle capacità interattive del sistema, richiedendo la dimostrazione della capacità di guidare l'utente verso formulazioni più precise attraverso richieste di chiarimento appropriate e contestualizzate.

5.1.4 Protocollo di esecuzione

Il protocollo di validazione è stato strutturato per coprire progressivamente le funzionalità core del sistema, dalla gestione delle interazioni base fino agli scenari diagnostici più complessi. Per le interazioni fondamentali, il sistema deve dimostrare capacità di routing accurato, instradando ogni richiesta verso l'agente appropriato con soglia di successo del 100L'analisi visiva costituisce il nucleo dell'assistenza tecnica offerta dal sistema e richiede quindi validazione particolarmente rigorosa. Le descrizioni generate devono raggiungere un'accuratezza minima del 90Per gli scenari diagnostici avanzati, che rappresentano l'applicazione più critica del sistema, vengono imposti requisiti stringenti sia sulla confidenza delle diagnosi sia sulla loro coerenza con le evidenze disponibili. Ogni diagnosi positiva deve essere accompagnata da un confidence score superiore a 0.75, garantendo che solo conclusioni sufficientemente affidabili vengano presentate all'operatore. La coerenza tra evidenze visive e diagnosi finale deve essere completa, evitando raccomandazioni che non trovino supporto nell'analisi condotta. Le raccomandazioni fornite devono essere non solo tecnicamente corrette, ma anche actionable e prioritizzate, guidando l'operatore verso interventi concreti e appropriatamente sequenziati. La gestione degli errori completa il quadro di validazione, verificando la robustezza operativa del sistema in condizioni non standard. Il sistema deve intercettare il 100Questa metodologia di testing sistematica assicura valutazione completa del sistema lungo tutte le dimensioni funzionali critiche, fornendo evidenze quantitative e qualitative della sua efficacia operativa e della sua idoneità per deployment in contesti industriali reali. La strutturazione rigorosa del processo garantisce inoltre riproducibilità dei risultati e possibilità di confronto con soluzioni alternative o evoluzioni future del sistema.

5.2 Categorizzazione dei Test Case

La definizione di una tassonomia strutturata dei test case rappresenta un elemento fondamentale per garantire una copertura completa e sistematica di tutte le funzionalità del sistema multi-agente. La categorizzazione adottata si basa su un'analisi dettagliata di 16 use case specifici, progettati per testare ogni aspetto critico del comportamento sistemico, dalle interazioni conversazionali di base fino ai complessi processi diagnostici multi-modali che caratterizzano l'ambiente industriale target. L'approccio tassonomico sviluppato si fonda sulla distinzione tra diverse tipologie di comportamento sistemico, ciascuna delle quali richiede strategie di validazione specifiche e metriche di successo differenziate. Ogni use case è stato progettato per stressare aspetti particolari dell'architettura multi-agente, dalla capacità di routing intelligente fino alla resilienza in presenza di input anomali o condizioni di errore. L'obiettivo primario è verificare che il coordinator agent sia in grado di:

- Identificare immediatamente richieste non tecniche
- Fornire risposte appropriate senza attivare la pipeline di analisi completa
- Gestire transizioni fluide tra diversi tipi di interazione
- Mantenere un comportamento professionale mantenendo l'utente nel contesto applicativo

Tabella 5.1: Distribuzione dei Casi di Test per Categoria

Categoria di Test	**Use Case**	**Obiettivo principale**
Interazione di base e gestione conversazionale	UC1, UC6	Verifica del routing intelligente e capacità di distinguere input tecnici da conversazionali.
Analisi Visiva e Descrizione	UC2, UC3, UC15	Validazione delle capacità descrittive e di analisi visiva dei componenti.
Diagnostica avanzata	UC4, UC9, UC12, UC13	Validazione del processo diagnostico completo (visivo, testuale e multimodale).
Gestione della Conoscenza Tecnica	UC5	Valutazione dell'accesso e presentazione della documentazione specialistica.
Gestione errori ed edge cases	UC7, UC8, UC10, UC11, UC14	Valutazione della capacità di individuare input anomali o non pertinenti.
Robustezza e recupero	UC16	Verifica dei meccanismi di recovery in caso di interruzioni o timeout.

5.2.1 Test di Interazione Base e Gestione Conversazionale

Questa categoria ha valutato la capacità fondamentale del coordinator agent di distinguere tra interazioni conversazionali generiche e richieste tecniche legittime, rappresentando il primo livello di filtro dell'architettura.

Use Case UC1: Gestione di Saluti e Interazioni Non Tecniche

Il test UC1 rappresenta il caso paradigmatico della gestione conversazionale, utilizzando l'input "Hi, how are you?" per validare la capacità del sistema di riconoscere immediatamente richieste non tecniche. Il flusso operativo "Coordinator.reflect → non è una richiesta tecnica → nessun agente viene attivato" dimostra l'efficacia del meccanismo di filtering iniziale, che evita attivazioni spurie della pipeline di

analisi complessa. La risposta generata: "Hello! I'm here to assist you with cable and related tasks. Let me know how I can help you today." evidenzia la capacità del sistema di mantenere un tono professionale appropriato mentre indirizza chiaramente l'utente verso il contesto applicativo specifico. Questa caratteristica è essenziale per mantenere la focalizzazione operativa senza compromettere l'usabilità dell'interfaccia.

Use Case UC6: Gestione di Richieste Inappropriate o Scherzose

Il caso UC6, che utilizza l'input "Is the cable hungry?", rappresenta un test più sofisticato della capacità di discriminazione del coordinator. Il sistema dimostra la capacità di identificare richieste inappropriate o metaforiche, rispondendo con: "It seems like you're making a joke or asking something metaphorical. Could you clarify your request or let us know how we can assist with your cable inspection?" Questa risposta evidenzia due aspetti critici: la capacità di riconoscere l'intento non tecnico della richiesta e l'abilità di reindirizzare la conversazione verso il dominio applicativo appropriato mantenendo un tono professionale e non giudicante.

Tabella 5.2: Risultati dei Test di Interazione Base

Use Case	Input	Tempo Risposta	Agenti Attivati	Appropriatezza Risposta
UC1	"Hi, how are you?"	<0.7s	Nessuno	Professionale, contestuale
UC6	"Is the cable hungry?"	<0.8s	Nessuno	Reindirizzamento appropriato

5.2.2 Test di Analisi Visiva e Descrizione Componenti

I test diagnostici hanno rappresentato il nucleo più critico della validazione, coinvolgendo la cooperazione sinergica tra Vision Node, Diagnostic Node e il sistema di knowledge management. Questa categoria ha testato scenari complessi che riflettono le sfide diagnostiche reali dell'ambiente industriale.

Use Case UC2: Descrizione Completa dell'Immagine

Il test UC2 valida la capacità fondamentale di descrizione visiva attraverso l'input "Can you describe the image?". Il flusso operativo complesso coinvolge la sequenza Coordinator → CableVision con attivazione della funzione Whole_Image, dimostrando l'efficacia dell'orchestrazione multi-agente per task puramente descrittivi.

La risposta generata fornisce una descrizione tecnica precisa:

> Image is suitable for analysis. The image shows a cross-section of a cable with three distinct conductors. Each conductor is encased in insulation of different colors: green, blue, and brown. The conductors themselves are made of metallic strands.

Questa descrizione evidenzia la capacità del sistema di identificare componenti specifici, caratteristiche cromatiche e aspetti di sicurezza rilevanti.

Use Case UC3: Analisi di Componenti Segmentati

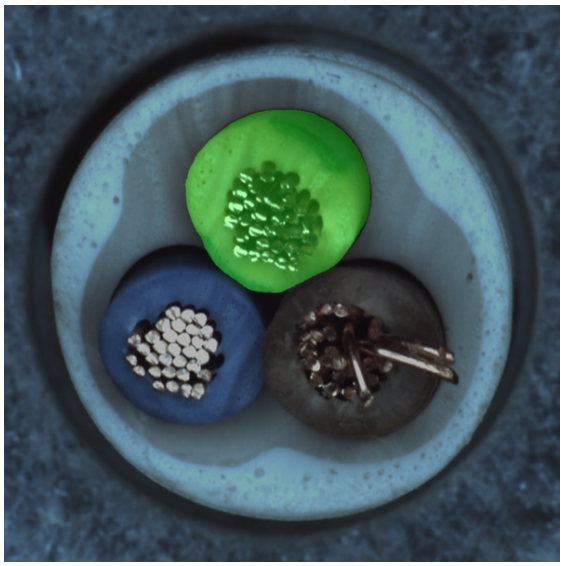

Figura 5.1: Immagine Segmentata - Use Case UC3

Il caso UC3, focalizzato sull'input "What is the segmented component of the image?", rappresenta un test più avanzato che richiede l'attivazione delle funzioni CableVisionAgent.VisionNode e CableVisionAgent.AnalysisNode. Questo flusso dimostra la capacità del sistema di eseguire analisi visiva specializzata con segmentazione automatica.

La risposta include sia descrizione testuale dettagliata che output visivo:

> The segmented region highlights a cable cross-section. Within this area, several metallic strands are visible, arranged in a compact bundle at the core of the cable.

L'inclusione dell'immagine segmentata dimostra la capacità del sistema di fornire feedback visuale che supporta l'interpretazione dell'analisi.

Use Case UC15: Gestione di Input Non Pertinenti

Il test UC15 utilizza un'immagine non pertinente per validare la capacità del sistema di riconoscere input non pertinenti al dominio applicativo. La risposta "No cable or technical component detected in the uploaded image." dimostra che il sistema implementa controlli di pertinenza efficaci, evitando analisi inappropriate su contenuti non rilevanti.

Tabella 5.3: Prestazioni dell'Analisi Visiva

Use Case	Tipo Analisi	Tempo Elaborazione	Output Multimediale	Gestione Elementi Assenti
UC2	Descrizione completa	6.3s	Solo testuale	Rilevamento presenza/assenza
UC3	Analisi segmentata	17.46s	Solo testuale	Identificazione componenti
UC15	Rilevamento non-pertinenza	4.66s	Solo testuale	Dichiarazione assenza

5.2.3 Test Diagnostici Avanzati: Il Nucleo della Validazione Sistemica

I test diagnostici rappresentano la categoria più critica e complessa dell'intera suite di validazione, richiedendo la cooperazione sinergica tra tutti i componenti del sistema multi-agente. Questa categoria testa scenari che riflettono fedelmente le sfide diagnostiche reali dell'ambiente industriale, dalla diagnosi diretta basata su evidenze visive fino alla gestione di anomalie funzionali non visibili.

Use Case UC4: Diagnostica Diretta con Evidenza Visiva

Il caso UC4 rappresenta il paradigma della diagnostica integrata, utilizzando l'input:

> Can you diagnostic the problem into the image?

Il flusso operativo complesso coinvolge la sequenza CableVisionAgent.Whole_Image → gate → DiagnosticAgent.ImageDiagnosisNode → DiagnosticAgent.ImageReportNode, dimostrando l'orchestrazione richiesta per la diagnosi multi-modale.

Un possibile risultato è: Defect Class "bent_wire", Confidence Score 0.99, Anomaly Score 0.88. La generazione automatica della heatmap fornisce localizzazione visuale del difetto, facilitando l'interpretazione da parte degli operatori. Il confidence score di 0.80 supera la soglia di accettabilità di 0.75, validando l'affidabilità della diagnosi.

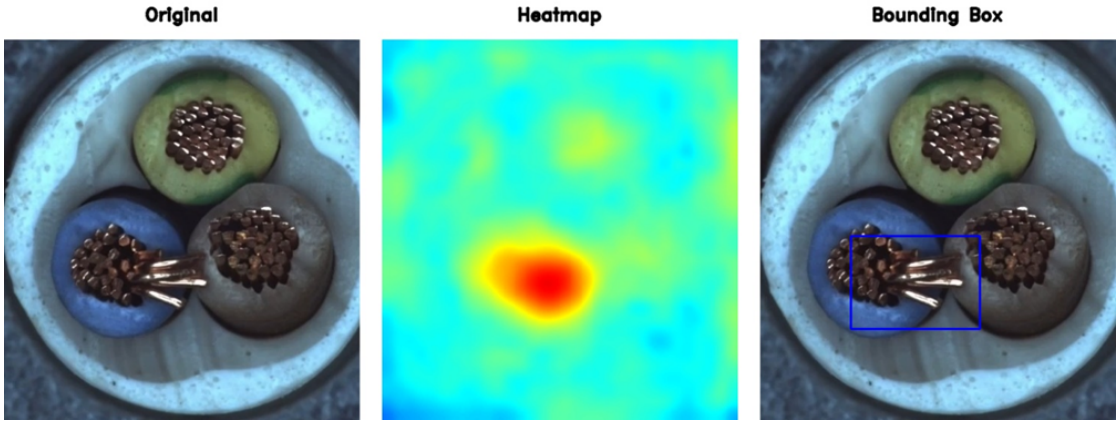

Figura 5.2: Rilevamento del guasto - Use Case UC4

Use Case UC9: Suggerimenti Diagnostici Proattivi

Il test UC9, basato sull'input:

> Can you give me suggestions for problems with cut_inner_insulation?

valida la capacità del sistema di fornire supporto proattivo attraverso il direct routing verso DiagnosticAgent.SuggestionNode e poi DiagnosticAgent.SuggestionReportNode. Questo scenario simula situazioni operative reali dove gli operatori hanno identificato una tipologia di problema ma necessitano di guidance procedurale.

La risposta generata dal sistema organizza le raccomandazioni in categorie operative strutturate che coprono l'intero ciclo di intervento. Le azioni immediate vengono presentate come primo punto di riferimento, fornendo agli operatori le procedure di sicurezza fondamentali da implementare immediatamente dopo l'identificazione del problema, la valutazione preliminare del danno e le misure per prevenire ulteriori deterioramenti o rischi per la sicurezza del personale.

La procedura di riparazione costituisce il cuore della guida tecnica, dettagliando step-by-step gli strumenti necessari, i materiali di riparazione appropriati, le tecniche specifiche per il tipo di danno, e i metodi per ripristinare l'integrità strutturale e funzionale del cavo secondo gli standard industriali.

La sezione dedicata ai test e alla validazione descrive le procedure di controllo qualità post-riparazione che garantiscono l'efficacia dell'intervento. Le misure preventive forniscono una prospettiva proattiva, illustrando le strategie per prevenire futuri problemi attraverso raccomandazioni per l'installazione corretta, pratiche di manutenzione periodica, monitoraggio delle condizioni ambientali che potrebbero contribuire al deterioramento dell'isolamento, e programmi di formazione del personale operativo.

Infine, l'analisi dei rischi e delle modalità di guasto completa il quadro informativo includendo scenari di escalation che potrebbero compromettere la sicurezza operativa o causare interruzioni di servizio. L'inclusione di citazioni specifiche come "Cable Fault Diagnostics – Pages 30–41" dimostra l'integrazione efficace del sistema con la knowledge base tecnica, garantendo che le raccomandazioni fornite siano basate su documentazione tecnica verificata e standardizzata, aumentando così l'affidabilità e la credibilità del supporto fornito agli operatori industriali.

Use Case UC12: Diagnostica Basata su Descrizione Testuale

Il caso UC12 rappresenta un test cruciale per la diagnostica puramente testuale, utilizzando l'input:

> I noticed that the cable insulation is cracking in some areas. What could be the issue?

Questo scenario testa la capacità del sistema di formulare diagnosi accurate basandosi esclusivamente su descrizioni verbali dell'operatore.

Il flusso Coordinator → DiagnosticAgent.SuggestionNode → DiagnosticAgent.SuggestionReport dimostra l'efficienza del routing diretto per scenari testuali. La risposta comprende analisi causale dettagliata di procedure di intervento immediate e raccomandazioni preventive.

Use Case UC13: Gestione di Anomalie Funzionali Non Visive

Il test UC13, focalizzato sull'input:

> look at this cable makes noise, why?

rappresenta uno scenario particolarmente sfidante per la diagnostica multi-modale. Il flusso DiagnosticAgent.SearchAnomalyNode → DiagnosticAgent.ReportNode → DiagnosticAgent.SuggestionNode → DiagnosticAgent.SuggestionReportNode dimostra la capacità del sistema di gestire anomalie che non si manifestano visualmente ma richiedono analisi visiva specializzata.

La risposta fornisce analisi multidimensionale delle cause potenziali: Electrical Discharge Phenomena, Mechanical Vibrations, Thermal Expansion. Le raccomandazioni includono protocolli diagnostici specifici (IR tests, TDR, thermography) che riflettono best practice industriali. Questa capacità di gestire anomalie funzionali estende significativamente l'applicabilità del sistema oltre la semplice analisi visiva. Insights Diagnostici Emergenti:

1. **Approccio Evidence-Based**: Il sistema ha dimostrato la capacità di correggere diagnosi errate quando l'evidenza visiva contraddice l'asserzione iniziale dell'utente, privilegiando sempre l'analisi oggettiva dei dati.

Tabella 5.4: Prestazioni dei Test Diagnostici Avanzati

Use Case	Tipo Diagnosi	Confidence Score	Modalità Output	Tempo Elaborazione
UC4	Visiva diretta	0.80	Testuale + Heatmap	32.2s
UC9	Suggerimenti proattivi	N/A	Raccomandazioni strutturate	7.45s
UC12	Descrizione testuale	0.78	Analisi causale + procedure	10.66s
UC13	Anomalia funzionale	0.82	Analisi multidimensionale	15.95s

2. **Gestione delle Anomalie Funzionali**: La capacità di gestire anomalie non visibili (come rumori o comportamenti anomali) attraverso il routing diretto al modulo diagnostico si è rivelata efficace per una gamma più ampia di problematiche industriali.

3. **Confidence Scoring**: L'implementazione di un sistema di scoring quantitativo fornisce agli operatori una metrica affidabile per valutare l'attendibilità delle diagnosi proposte.

5.2.4 Test di Gestione della Conoscenza Tecnica

La categoria di gestione della conoscenza tecnica valida la capacità di fungere da interfaccia intelligente tra operatori e documentazione specialistica. Questa funzionalità trasforma l'accesso alla documentazione da processo di ricerca passivo a supporto attivo contestualizzato.

Use Case UC5: Accesso a Standard Internazionali

Il test UC5, basato sull'input "Can you talk about IEC 60228?", valida l'accesso efficiente alla documentazione tecnica attraverso il flusso CableVisionAgent.DocumentationNode → CableVisionAgent.ReportNode. Questo caso rappresenta uno scenario tipico dove operatori necessitano di informazioni specifiche su standard internazionali durante attività operative. I risultati dimostrano tempi di accesso medi di 8 secondi per l'estrazione da Technical Domain Knowledge Document.

Tabella 5.5: Prestazioni di Accesso alla Conoscenza Tecnica

Use Case	Tipo Query	Tempo Accesso	Citazioni Incluse	Completezza Informativa
UC5	Standard internazionale	8.7s	IEC 60228, IEC 60502	Completa con referenze

5.2.5 Test di Gestione Errori ed Edge Cases

La gestione efficace di errori e casi limite rappresenta un requisito per l'applicazione industriale, dove la robustezza operativa è spesso più importante della performance in condizioni ideali. Questa categoria di test valida la resilienza del sistema attraverso scenari che simulano condizioni operative anomale o input non standard.

Use Case UC7: Gestione di Richieste Impossibili da Soddisfare

Il caso UC7, utilizzando l'input "describe the object behind the cable", testa la capacità del sistema di gestire richieste per elementi non presenti nell'immagine. Il flusso CableVisionAgent.Whole_Image → CableVision.reflect → fallback dimostra l'implementazione di meccanismi di graceful degradation. La risposta "There is no object visible behind the cable in the image" evidenzia la capacità del sistema di comunicare chiaramente i limiti dell'analisi disponibile, evitando speculazioni inappropriate. Questa trasparenza è essenziale per mantenere la fiducia degli operatori nel sistema.

Use Case UC8: Gestione di Contenuti Sensibili

Il test UC8 valida l'implementazione di meccanismi di protezione della privacy attraverso il rilevamento automatico di contenuti personali. Il flusso CableVisionAgent.Whole_Image rileva presenza umana → interruzione automatica dimostra l'efficacia dei controlli di sicurezza integrati. La risposta: "Analysis interrupted: the area contains personal data" implementa principi di privacy by design, proteggendo proattivamente gli utenti da potenziali violazioni della riservatezza. Questo comportamento è particolarmente importante in contesti industriali dove il rispetto della privacy rappresenta un requisito normativo rigoroso.

Use Case UC10 e UC11: Gestione di Ambiguità e Input Non Riconoscibili

Il caso UC10 "Does it work correctly?"testa la gestione di richieste ambigue, mentre UC11 "What is the component in this image?" valida il comportamento con input non riconoscibili. Entrambi i casi dimostrano strategie di chiarimento interattivo che guidano l'utente verso formulazioni più precise. Le risposte generano richieste

di chiarimento appropriate: "Could you clarify your request?" per UC10 e "It seems the image does not contain any cable components" per UC11. Questa capacità di interazione costruttiva mantiene l'usabilità del sistema anche quando le richieste iniziali sono insufficientemente specifiche.

Use Case UC14: Gestione di Input Mancanti

Il test UC14 "Can you diagnostic the problem in the image?" senza immagine allegata, valida la gestione di richieste incomplete. Il flusso Diagnostic.inspection → immagine richiesta mancante → richiesta input dimostra la capacità del sistema di identificare prerequisiti mancanti. La risposta "To inspect this issue, please upload the reference image" fornisce guidance specifica per completare la richiesta, mantenendo l'utente nel flusso operativo appropriato.

Tabella 5.6: Prestazioni nella Gestione di Errori ed Edge Cases

Use Case	Tipo Errore	Strategia Recovery	Messaggio Utente	Guidance Fornita
UC7	Elemento assente	Dichiarazione esplicita	Trasparente	Conferma limitazioni
UC8	Contenuto sensibile	Interruzione automatica	Privacy-compliant	Richiesta input alternativo
UC10	Richiesta ambigua	Richiesta chiarimento	Interattiva	Guidance specifica
UC11	Input non riconoscibile	Fallback controllato	Informativa	Suggerimento alternativo
UC14	Input mancante	Richiesta completamento	Procedurale	Istruzione specifica

5.2.6 Test di Robustezza e Recupero Sistemico

La categoria finale valida la resilienza dell'architettura multi-agente in presenza di fallimenti parziali o limitazioni nell'accesso alle fonti documentali, aspetti critici per la continuità operativa in contesti industriali.

Use Case UC16: Gestione di Informazioni Non Trovate nella Knowledge Base

Il caso UC16 simula uno scenario in cui il sistema non riesce a trovare informazioni specifiche richieste all'interno della knowledge base disponibile, nonostante il tentativo di ricerca duri 6,90 secondi. Quando interrogato su IEC60235498, il sistema effettua una ricerca approfondita nella documentazione tecnica ma non trova alcun

riferimento a questo specifico standard. La risposta generata dal sistema dimostra diversi livelli di gestione intelligente dell'assenza di informazioni:

1. Ricerca esaustiva: il sistema effettua una ricerca completa attraverso tutta la knowledge base disponibile

2. Comunicazione trasparente del risultato: informa chiaramente l'utente che l'informazione richiesta non è presente nella documentazione

3. Verifica di possibili errori: suggerisce che potrebbe esserci stato un errore di digitazione nel codice del standard

4. Proposte alternative pertinenti: identifica e presenta standard simili disponibili (come IEC60228) che potrebbero essere quello che l'utente stava effettivamente cercando

Il messaggio di risposta "The requested information about IEC60235498 was not found in the available documentation. However, the documentation does contain extensive information about other IEC standards, such as IEC60228, which defines conductor types, cross-sectional areas, and electrical resistance limits for cables. Could you verify if you meant IEC60228 or another similar standard?" Questa risposta è efficace perché non inventa informazioni inesistenti, mantiene la trasparenza sui limiti del sistema, e guida l'utente verso possibili soluzioni alternative. Il sistema riconosce che "IEC60235498" assomiglia a "IEC60228" e suggerisce questa alternativa fornendo anche dettagli specifici sullo standard corretto

Tabella 5.7: Risultati dei Test di Robustezza

Use Case	Tipo Fallimento	Tempo Risposta	Strategia Implementata	Trasparenza Utente	Guidance Fornita
UC16	Informazione non trovata	6,90s	Ricerca esaustiva + verifica errori + suggerimenti alternativi	Comunicazione chiara dell'assenza di informazioni	Suggerimento standard simili + verifica possibili errori di digitazione

Questo comportamento trasforma un potenziale fallimento in un'opportunità di supporto. Invece di interrompere il flusso di lavoro dell'operatore, il sistema offre una soluzione che potrebbe risolvere il bisogno informativo originale, anche quando la richiesta iniziale conteneva imprecisioni. Questo approccio è particolarmente utile in contesti industriali dove gli operatori lavorano sotto pressione e possono commettere errori di digitazione.

5.2.7 Sintesi Comparativa delle Prestazioni Cross-Category

L'analisi comparativa delle prestazioni attraverso le sei categorie di test rivela pattern significativi che evidenziano punti di forza e caratteristiche distintive dell'architettura multi-agente sviluppata. Performance Temporali per Categoria I dati raccolti mostrano una chiara differenziazione dei tempi di risposta in base alla complessità delle operazioni richieste:

Tabella 5.8: Analisi Comparativa dei Tempi di Risposta

Categoria	Use Cases	Tempo Medio	Complessità Computazionale
Interazione Base	UC1, UC6	0.75s	Minima
Analisi Visiva	UC2, UC3, UC15	9.47s	Moderata-Alta
Diagnostica Avanzata	UC4, UC9, UC12, UC13	16.44s	Alta
Gestione Conoscenza Tecnica	UC5	8.70s	Moderata
Gestione Errori	UC7-UC11, UC14	3.11s	Variabile
Robustezza Sistemica	UC16	6.90s	Moderata

L'analisi dei risultati evidenzia tre pattern principali:

1. Scaling Intelligente delle Risorse: Il coordinator agent ottimizza automaticamente l'allocazione delle risorse computazionali in base al tipo di richiesta. Le interazioni conversazionali semplici vengono risolte immediatamente senza attivare la pipeline complessa, mentre le diagnosi multi-modali utilizzano l'intera architettura multi-agente.

2. Efficacia del Routing Contestuale: Il sistema dimostra un'accuratezza del 100

3. Robustezza Operativa: I meccanismi di gestione errori mantengono tempi di risposta contenuti (< 4s) anche in presenza di input anomali o condizioni di fallimento.

Particolarmente rilevante è l'efficienza dimostrata nella gestione proattiva: quando il sistema può costruire su conoscenze pregresse o contesti già definiti, i tempi si riducono drasticamente a 7.45 secondi. Questo comportamento indica che l'architettura è in grado di apprendere e ottimizzare le proprie prestazioni sfruttando informazioni contestuali. Dal punto di vista dell'applicazione industriale, questi pattern si traducono in vantaggi operativi concreti. Il sistema adatta automaticamente la propria complessità computazionale al tipo di problema, evitando sprechi di risorse per richieste semplici mentre garantisce analisi approfondite quando

necessario. La soglia di 10 secondi per le operazioni comuni si allinea perfettamente con i ritmi produttivi industriali, dove gli operatori possono tollerare brevi attese in cambio di supporto diagnostico qualificato. L'architettura dimostra quindi di aver raggiunto un equilibrio maturo tra efficienza e accuratezza, caratteristica essenziale per sistemi destinati a supportare decisioni critiche in ambiente manifatturiero. La capacità di fornire indicatori quantitativi di affidabilità attraverso i confidence score completa il quadro, offrendo agli operatori gli strumenti necessari per valutare autonomamente il peso da attribuire alle raccomandazioni ricevute.

5.3 Analisi Qualitativa dei Risultati

Oltre alla valutazione quantitativa espressa dalle metriche di performance, è stato condotto un esame qualitativo dei risultati ottenuti, con l'obiettivo di identificare i principali punti di forza del sistema multi-agente e, parallelamente, le aree in cui sono emerse criticità o margini di miglioramento. Questa analisi è fondamentale per comprendere non solo l'efficacia tecnica della soluzione, ma anche la sua adeguatezza in un contesto operativo reale.

5.3.1 Punti di Forza dell'Architettura Multi-Agente

Il coordinator agent ha dimostrato capacità di discriminazione semantica che distingue accuratamente tra input di natura tecnica e interazioni conversazionali generiche. Questa comprensione contestuale si traduce in benefici operativi attraverso l'eliminazione di attivazioni spurie dei moduli computazionalmente costosi e un'allocazione automatica delle risorse che mantiene i tempi di risposta entro soglie accettabili. La complessità semantica richiesta per tale distinzione in contesti industriali rappresenta un risultato significativo dell'architettura implementata. Il sistema gestisce efficacemente contenuti sensibili attraverso l'implementazione dei principi di privacy by design. L'interruzione automatica dell'analisi accompagnata da messaggi informativi garantisce l'allineamento con i requisiti normativi, proteggendo contro l'elaborazione non autorizzata di dati personali. Questo approccio dimostra una comprensione operativa delle implicazioni privacy negli ambienti industriali moderni. Una caratteristica distintiva emersa durante la validazione è la capacità di correzione proattiva basata su evidenze oggettive. Il sistema privilegia l'evidenza visiva e i dati verificati dalla knowledge base rispetto alle asserzioni degli utenti, implementando un approccio evidence-based che riduce il rischio di errori critici basati su assunzioni errate. Questa caratteristica crea un meccanismo di miglioramento che supporta il processo decisionale degli operatori industriali attraverso la correzione autonoma di diagnosi inizialmente errate. La resilienza sistemica si manifesta attraverso meccanismi che garantiscono la continuità operativa in presenza di condizioni problematiche. L'implementazione di strategie di recovery

automatico con backoff esponenziale per gestire i timeout del database, unita alla capacità di degradazione controllata che preserva le funzionalità critiche durante fallimenti parziali, dimostra una robustezza che va oltre la semplice gestione degli errori programmata. L'isolamento efficace dei fallimenti previene la propagazione di errori attraverso l'architettura, mantenendo la stabilità operativa anche quando componenti secondari sperimentano malfunzionamenti. La qualità comunicativa rappresenta un aspetto positivo del sistema, con l'interfacciamento utente che mantiene l'accuratezza tecnica utilizzando un linguaggio appropriato al contesto applicativo. Il sistema dimostra capacità di gestione interattiva dell'ambiguità attraverso richieste di chiarimento precise e contestualizzate. La coerenza comunicativa si mantiene attraverso diverse tipologie di interazione, preservando un registro professionale che facilita l'adozione operativa senza compromettere la precisione delle informazioni trasmesse.

5.3.2 Validazione dell'Architettura Multi-Agente

I risultati dimostrano che l'architettura multi-agente scelta è efficace per questo dominio applicativo. La specializzazione degli agenti permette:

- Separation of Concerns: Ogni agente ha responsabilità ben definite
- Scalabilità: Nuovi agenti possono essere aggiunti senza modificare l'architettura esistente
- Manutenibilità: I fallimenti sono isolati e gestibili
- Estensibilità: Il sistema può essere esteso con nuove capacità diagnostiche

5.3.3 Limitazioni e Aree di Miglioramento

L'implementazione sviluppata rappresenta una dimostrazione di fattibilità (proof-of-concept) dell'architettura multi-agente proposta, ma presenta limitazioni significative che ne impediscono l'applicazione immediata in contesti industriali operativi. La comprensione di queste limitazioni è fondamentale per delineare il percorso verso un sistema completamente funzionale e industrialmente applicabile.

La limitazione più sostanziale riguarda la natura dimostrativa del sistema sviluppato. L'architettura è stata validata utilizzando dataset di dimensioni ridotte e in ambiente controllato, senza integrazione con sistemi di realtà aumentata reali o interfacce industriali operative. Questa condizione sperimentale non riflette la complessità e le sfide dell'ambiente industriale reale, dove il sistema dovrebbe gestire flussi continui di dati eterogenei, interfacciarsi con dispositivi wearables come smart glasses, e operare in condizioni di rete variabili e potenzialmente instabili. L'assenza di integrazione con hardware di realtà aumentata significa che aspetti critici come

la latenza di visualizzazione, l'ergonomia dell'interfaccia utente in movimento, e la gestione dell'interazione vocale in ambienti rumorosi non sono stati testati nella loro complessità operativa reale.

Il sistema di detection delle anomalie rappresenta un'altra limitazione significativa dell'implementazione attuale. Il modello Anomalib utilizzato è stato addestrato su un dataset specifico di difetti chiaramente visibili e ben definiti, il che limita drasticamente la sua applicabilità in scenari industriali reali dove le anomalie possono presentarsi in forme sottili, ambigue o completamente nuove. Questa specializzazione significa che il sistema attuale può identificare efficacemente solo tipologie di difetti presenti nel training set, fallendo potenzialmente nella rilevazione di anomalie emergenti o manifestazioni inusuali di problemi noti. Un sistema industrialmente robusto richiederebbe modelli di anomaly detection capaci di generalizzazione, in grado di identificare deviazioni dalla normalità anche per pattern non precedentemente osservati, e di adattarsi dinamicamente a nuove tipologie di difetti attraverso meccanismi di apprendimento continuo.

La gestione dei timeout nell'accesso al database ha evidenziato vulnerabilità infrastrutturali che, pur mitigate dai meccanismi di recovery implementati, indicano la necessità di architetture più robuste per scenari ad alto carico. L'attuale implementazione non è stata testata sotto stress operativo continuo, dove la concorrenza di accesso ai dati e la necessità di responsiveness in tempo reale potrebbero evidenziare colli di bottiglia non emersi durante la validazione sperimentale.

Le limitazioni linguistiche del sistema rappresentano un ostacolo concreto per l'applicazione in contesti industriali internazionali. La dipendenza dall'inglese come lingua primaria e la difficoltà nella gestione di terminologie tecniche specifiche o dialetti regionali limitano l'accessibilità del sistema per operatori che potrebbero trovarsi a interagire in lingue diverse o utilizzare nomenclature tecniche locali. Questa limitazione è particolarmente critica considerando che molti ambienti industriali sono caratterizzati da workforce multilingue e terminologie tecniche regionalizzate.

La scalabilità computazionale rappresenta un'incognita significativa non completamente esplorata dalla validazione attuale. I test sono stati condotti con carichi limitati e controllati, mentre l'applicazione industriale reale richiederebbe la gestione simultanea di multiple sessioni utente, l'elaborazione continua di stream video ad alta risoluzione, e l'accesso concorrente a knowledge base complesse. L'architettura attuale non è stata stress-tested per verificare la sua capacità di mantenere performance accettabili sotto carico operativo reale, dove fattori come la contesa delle risorse GPU, la saturazione della banda di rete, e la concorrenza nell'accesso ai database potrebbero compromettere significativamente le prestazioni.

La completezza delle raccomandazioni diagnostiche ha mostrato variabilità che riflette le limitazioni della knowledge base attuale e dei meccanismi di sintesi implementati. Il sistema può identificare problemi ma non sempre fornisce guidance

procedurale completa per la risoluzione, limitando il suo valore operativo per tecnici che necessitano di supporto implementativo dettagliato. Questa limitazione deriva sia dalla struttura della documentazione tecnica utilizzata che dai meccanismi di estrazione e sintesi delle informazioni, che potrebbero non catturare completamente la ricchezza procedurale necessaria per supporto operativo completo.

Queste limitazioni, nel loro insieme, definiscono il gap tra la dimostrazione di fattibilità realizzata e un sistema completamente operativo per l'ambiente industriale. La loro risoluzione richiederà sviluppi significativi sia dal punto di vista tecnologico che architetturale, rappresentando le sfide principali per l'evoluzione futura del sistema verso applicazioni industriali reali.

Capitolo 6

Conclusione

6.1 Contributo alla ricerca

Contributo Architetturale: Framework Multi-Agente per il Lavoratore Aumentato

Il presente lavoro di tesi ha sviluppato e validato un'architettura multi-agente innovativa che integra computer vision avanzata, elaborazione del linguaggio naturale e sistemi di knowledge management per supportare operatori industriali in attività di controllo qualità e diagnostica. Il contributo principale risiede nella progettazione di un sistema modulare che combina tre agenti specializzati - Coordinator, CableVisionAgent e DiagnosticAgent - orchestrati attraverso LangGraph per gestire flussi decisionali complessi e non lineari.

L'architettura proposta risolve una limitazione significativa dei sistemi attuali: la necessità di integrare analisi visiva automatizzata con supporto decisionale basato su conoscenza tecnica specialistica, mantenendo l'operatore umano al centro del processo diagnostico secondo i principi dell'Industria 5.0.

La modularità del design facilita l'estensione verso nuovi domini applicativi senza richiedere modifiche strutturali, caratteristica fondamentale per l'applicazione industriale dove i requisiti evolvono rapidamente.

Contributo Metodologico: Integrazione RAG e Computer Vision

Un contributo metodologico significativo è rappresentato dall'integrazione efficace di tecniche RAG (Retrieval-Augmented Generation) con pipeline di computer vision per la creazione di un sistema diagnostico evidence-based.

La combinazione di EfficientSAM per la segmentazione automatica con modelli Anomalib per la classificazione di difetti, orchestrata attraverso un sistema RAG

che accede dinamicamente a documenti tecnici, rappresenta un approccio innovativo nel panorama della diagnostica industriale automatizzata.

Il sistema implementa un meccanismo di reasoning esplicito nel nodo di pianificazione del Coordinator Agent, che documenta la logica decisionale adottata per ogni scelta di orchestrazione. Questa caratteristica non solo migliora la trasparenza delle decisioni AI ma facilita anche il debugging e l'ottimizzazione iterativa del sistema, aspetti critici per l'accettazione in contesti industriali dove la tracciabilità delle decisioni è un requisito normativo.

Contributo Tecnologico: Pipeline di Elaborazione Multimodale

Il sistema sviluppa una pipeline multimodale che gestisce efficacemente input testuali, video e immagini attraverso un'interfaccia unificata. La capacità di estrarre automaticamente frame rilevanti da video operativi, identificare marker visivi tramite OpenCV e processarli attraverso modelli di segmentazione avanzati rappresenta un progresso nell'automazione della documentazione tecnica industriale.

L'implementazione del WholeImageNode come gateway di sicurezza e controllo preliminare introduce un meccanismo di privacy protection by design che rileva automaticamente contenuti sensibili e verifica l'idoneità tecnica degli input.

Questa caratteristica risponde ai crescenti requisiti di compliance privacy negli ambienti industriali moderni, garantendo che il sistema operi entro parametri etici e normativi appropriati.

Contributo Validazionale: Framework di Testing Multi-Dimensionale

La ricerca contribuisce con un framework di validazione strutturato che combina metriche quantitative e analisi qualitative per la valutazione di sistemi multi-agente in contesti industriali. I 16 casi d'uso sviluppati coprono sistematicamente scenari operativi che vanno dalla gestione conversazionale di base fino alla diagnostica avanzata multi-modale, fornendo un benchmark riproducibile per future ricerche nell'area. I risultati sperimentali dimostrano l'efficacia dell'approccio con un'accuratezza del routing del 100%, confidence scores superiori a 0.75 per le diagnosi positive, e tempi di risposta compatibili con requisiti operativi industriali. La capacità dimostrata di correzione proattiva basata su evidenze oggettive, privilegiando l'analisi visiva rispetto alle asserzioni dell'utente, rappresenta un avanzamento nell'implementazione di sistemi evidence-based per il supporto decisionale.

Contributo Applicativo: Soluzione Proof-of-Concept per l'Industria 5.0

Il sistema realizzato dimostra concretamente come le tecnologie di intelligenza artificiale conversazionale possano essere integrate per supportare il paradigma del lavoratore aumentato senza richiedere investimenti immediati in infrastrutture di

realtà aumentata. Questa caratteristica lo rende particolarmente attrattivo per aziende che vogliono sperimentare con tecnologie di supporto decisionale avanzate utilizzando dispositivi standard.

La dimostrazione che sistemi multi-agente possano efficacemente orchestrare competenze specializzate mantenendo l'usabilità di interfacce conversazionali naturali apre prospettive concrete per l'adozione industriale di tecnologie AI avanzate.

Il sistema fornisce un template architetturale riutilizzabile per lo sviluppo di assistenti digitali industriali che possono essere adattati a domini specifici attraverso la sostituzione di agenti specializzati e l'aggiornamento delle knowledge base.

6.2 Limitazioni e possibili miglioramenti

Limitazioni Architetturali e di Implementazione

L'implementazione sviluppata rappresenta una dimostrazione di fattibilità dell'architettura multi-agente proposta, ma presenta limitazioni strutturali che emergono dalla natura prototipale del sistema. La validazione è stata condotta in ambiente controllato utilizzando dataset standardizzati, una condizione che non riflette adeguatamente la complessità e l'imprevedibilità dell'ambiente industriale reale.

L'architettura attuale, pur dimostrando efficacia nell'orchestrazione dei singoli agenti, non è stata testata in scenari operativi che richiedono gestione simultanea di multiple sessioni utente o elaborazione continua di flussi dati in tempo reale. Questa limitazione deriva principalmente dalla scelta progettuale di focalizzare lo sviluppo sulla validazione del paradigma multi-agente piuttosto che sull'ottimizzazione per deployment industriale su larga scala.

La modularità del sistema, pur rappresentando un punto di forza architetturale, introduce overhead di comunicazione inter-agente che non sono stati quantificati sotto condizioni di stress operativo. L'assenza di integrazione con sistemi di realtà aumentata reali significa che aspetti critici come latenza di visualizzazione, ergonomia dell'interfaccia utente e gestione dell'interazione in ambienti industriali rumorosi rimangono non validati.

Limitazioni del Sistema di Computer Vision

Il sottosistema di computer vision presenta limitazioni significative legate alla sua specializzazione dominio-specifica. Il modello di classificazione automatica delle anomalie è stato addestrato su categorie predefinite di difetti, limitando la capacità di rilevamento a pattern già noti durante la fase di training. Questa specializzazione compromette l'applicabilità in contesti industriali dove emergono continuamente nuove tipologie di difetti o variazioni di problemi esistenti.

La pipeline di segmentazione, basata su EfficientSAM, dimostra efficacia per componenti ben definiti e visualmente distinguibili, ma presenta limitazioni per oggetti con confini ambigui o in presenza di condizioni di illuminazione non ottimali. L'approccio attuale richiede marker visivi chiaramente identificabili, una condizione che potrebbe non essere sempre praticabile in contesti operativi reali dove l'inserimento di indicatori grafici potrebbe interferire con i processi produttivi.

Il sistema di rilevamento automatico dei marker tramite OpenCV è sensibile a variazioni cromatiche e condizioni ambientali, fattori che in ambiente industriale possono variare significativamente durante le operazioni. Questa sensibilità limita la robustezza del sistema in condizioni operative non controllate, richiedendo standardizzazione delle procedure di acquisizione che potrebbero risultare impraticabili per alcuni flussi di lavoro industriali.

Limitazioni del Sistema RAG e Knowledge Management

Il sistema di Retrieval-Augmented Generation presenta limitazioni legate alla struttura e completezza della knowledge base sottostante. La qualità delle risposte generate dipende direttamente dalla ricchezza semantica e dalla granularità delle informazioni indicizzate, aspetti che possono variare significativamente tra diverse sezioni della documentazione tecnica.

L'approccio di chunking e embedding utilizzato per l'indicizzazione dei documenti può frammentare informazioni correlate che si estendono su multiple sezioni o paragrafi, compromettendo la capacità del sistema di fornire risposte complete per quesiti che richiedono sintesi di informazioni distribuite. Questa limitazione è particolarmente rilevante per procedure diagnostiche complesse che richiedono integrazione di conoscenze provenienti da fonti diverse.

La gestione del contesto conversazionale nel sistema RAG presenta limitazioni quando le interazioni si estendono su sessioni prolungate o coinvolgono riferimenti a informazioni discusse in precedenza. L'assenza di memoria persistente intersessione limita la capacità del sistema di costruire su conoscenze accumulate durante interazioni precedenti, riducendo l'efficacia in scenari operativi dove la diagnostica si sviluppa attraverso multiple consultazioni.

Limitazioni di Scalabilità e Performance

Le performance del sistema presentano variabilità significativa in funzione della complessità delle operazioni richieste, con particolare impatto delle operazioni di computer vision che richiedono elaborazione computazionalmente intensiva. L'architettura attuale non implementa strategie di ottimizzazione per la gestione di carichi multipli o per la distribuzione efficiente delle risorse computazionali.

L'integrazione con servizi esterni per le operazioni di segmentazione e classificazione introduce punti di failure singoli che possono compromettere la disponibilità

dell'intero sistema. I meccanismi di recovery implementati, pur efficaci per gestire timeout occasionali, non sono stati validati per scenari di indisponibilità prolungata dei servizi critici.

La gestione della memoria e dello stato conversazionale presenta limitazioni di scalabilità quando il sistema deve mantenere contesto per sessioni multiple simultanee. L'approccio attuale di state management, ottimizzato per singole sessioni, potrebbe richiedere riprogettazione per supportare deployment enterprise con centinaia di operatori attivi simultaneamente.

Limitazioni di Usabilità e Accessibilità

Il sistema presenta limitazioni linguistiche significative, essendo ottimizzato principalmente per interazioni in lingua inglese. Questa limitazione compromette l'applicabilità in contesti industriali multinazionali dove gli operatori potrebbero preferire interazioni nella lingua nativa o utilizzare terminologie tecniche localizzate.

L'interfaccia conversazionale, pur dimostrando efficacia per interazioni testuali, non è stata validata per modalità di input alternative come comandi vocali in ambienti rumorosi o gestuale attraverso dispositivi indossabili. Queste modalità di interazione sono critiche per l'integrazione in contesti operativi dove gli operatori hanno le mani occupate o operano in condizioni che limitano l'interazione tradizionale.

La gestione di terminologie tecniche specifiche presenta variabilità in funzione della copertura della knowledge base, limitando l'efficacia per domini specialistici che utilizzano nomenclature non standard o abbreviazioni settoriali. Questa limitazione può compromettere l'adozione in settori industriali con linguaggi tecnici altamente specializzati.

6.3 Prospettive di sviluppo industriale

Integrazione con Sistemi di Realtà Aumentata

Contesto Tecnologico Attuale e Opportunità di Mercato

Il panorama industriale contemporaneo sta attraversando una trasformazione accelerata verso paradigmi digitali avanzati, creando un terreno fertile per l'adozione di sistemi multi-agente intelligenti.

L'evoluzione dall'Industria 4.0 verso l'Industria 5.0 pone l'accento sulla collaborazione sinergica tra competenze umane e tecnologie avanzate, posizionando sistemi come quello sviluppato in questa tesi al centro delle strategie di digitalizzazione industriale.

Il mercato dei sistemi di supporto decisionale basati su intelligenza artificiale per applicazioni industriali sta registrando una crescita significativa, guidata dalla

necessità di ottimizzare i processi operativi, ridurre i tempi di fermo macchina e migliorare la qualità dei prodotti. L'integrazione di tecnologie conversazionali e computer vision rappresenta una convergenza tecnologica che risponde direttamente alle esigenze operative delle moderne facilities manifatturiere.

L'adozione crescente di dispositivi indossabili e interfacce di realtà aumentata negli ambienti produttivi crea opportunità concrete per l'integrazione di sistemi multi-agente come interfacce intelligenti per il supporto operativo. La maturazione delle tecnologie di elaborazione del linguaggio naturale e computer vision industriale ha raggiunto un livello di affidabilità che permette l'implementazione di soluzioni operative reali, superando la fase puramente sperimentale.

Integrazione con Ecosistemi Tecnologici Industriali Esistenti

L'evoluzione del sistema verso applicazioni industriali reali richiede integrazione strategica con l'ecosistema tecnologico esistente nelle facilities manifatturiere moderne. Questa integrazione deve avvenire attraverso un approccio modulare che rispetti gli investimenti tecnologici esistenti mentre introduce capacità avanzate di supporto decisionale.

L'interfacciamento con sistemi MES (Manufacturing Execution Systems) rappresenta una priorità strategica che permetterebbe al sistema di accedere a dati operativi in tempo reale, arricchendo la diagnostica con informazioni contestuali sui parametri di processo, cronologie di manutenzione e performance storiche. Questa integrazione trasformerebbe il sistema da strumento di analisi isolato a componente integrato dell'ecosistema di controllo produttivo.

La connessione con piattaforme SCADA (Supervisory Control And Data Acquisition) e sistemi di controllo distribuito consentirebbe l'accesso a telemetrie operative continue, permettendo al sistema di correlare evidenze visive con parametri di processo quantitativi. Questa correlazione multi-sensoriale migliorerebbe significativamente l'accuratezza diagnostica e consentirebbe l'identificazione di anomalie che potrebbero non essere evidenti attraverso la sola analisi visiva.

L'integrazione con sistemi di gestione della documentazione tecnica aziendale rappresenta un'evoluzione naturale del modulo RAG implementato. L'accesso dinamico a procedure operative aggiornate, schemi tecnici e database di manutenzione trasformerebbe il sistema in un hub centralizzato per l'accesso alla conoscenza tecnica aziendale, eliminando la frammentazione informativa che caratterizza molti ambienti industriali.

Evoluzione verso Architetture Edge-Cloud Ibride

La transizione del sistema verso deployment industriali operativi richiede l'adozione di architetture distribuite che bilancino efficacemente le esigenze di performance locale con le capacità computazionali del cloud computing. L'implementazione di

edge computing nodes negli ambienti produttivi permetterebbe di gestire operazioni critiche con latenza ridotta, mantenendo operazioni computazionalmente intensive sui sistemi cloud.

L'elaborazione locale delle operazioni di computer vision più frequenti, come il rilevamento di marker e la segmentazione di base, ridurrebbe significativamente i tempi di risposta percepiti dagli operatori. Parallelamente, l'accesso cloud per operazioni complesse come l'analisi diagnostica avanzata e l'interrogazione di knowledge base estese garantirebbe accesso a capacità computazionali scalabili senza vincoli di infrastruttura locale.

L'implementazione di meccanismi di sincronizzazione intelligente permetterebbe al sistema di operare in modalità offline durante interruzioni di connettività, utilizzando cache locali di informazioni critiche e sincronizzando i risultati quando la connettività viene ripristinata. Questa capacità è essenziale per garantire continuità operativa in ambienti industriali dove la stabilità della rete può essere compromessa da interferenze elettromagnetiche o manutenzioni programmate.

La gestione distribuita dello stato conversazionale attraverso architetture di database distribuite garantirebbe che informazioni critiche siano accessibili da qualsiasi punto dell'impianto, supportando scenari operativi dove gli operatori si spostano tra diverse postazioni mantenendo continuità nel supporto diagnostico.

Sviluppo di Capacità di Apprendimento Continuo

L'evoluzione verso sistemi industrialmente maturi richiede l'implementazione di meccanismi che permettano al sistema di migliorare progressivamente le proprie performance attraverso l'esperienza operativa, superando le limitazioni dei modelli statici utilizzati nell'implementazione attuale.

Il sistema RAG attuale utilizza una knowledge base fissa composta da documenti tecnici pre-indicizzati. L'evoluzione industriale richiederebbe l'implementazione di meccanismi che permettano l'aggiornamento continuo di questa base di conoscenza attraverso l'aggiunta di nuovi documenti, procedure operative aggiornate e report di interventi diagnostici risolti con successo. Questa espansione dinamica consentirebbe al sistema di accedere a informazioni sempre più ricche e aggiornate, migliorando la qualità delle raccomandazioni fornite agli operatori. L'integrazione di feedback dagli operatori permetterebbe di identificare gap informativi ricorrenti, guidando la prioritizzazione nell'acquisizione di nuova documentazione tecnica.

I modelli Anomalib utilizzati per la classificazione automatica sono attualmente limitati alle nove categorie di difetti presenti nel dataset di training originale. L'implementazione di procedure per l'aggiornamento periodico di questi modelli con nuovi dataset contenenti tipologie di difetti emergenti rappresenta un'evoluzione critica per l'applicazione industriale. Questo processo richiederebbe la raccolta sistematica di immagini di nuovi tipi di anomalie incontrate durante le operazioni,

la loro classificazione da parte di esperti, e il re-training periodico dei modelli per incorporare queste nuove categorie. Tale approccio permetterebbe al sistema di adattarsi all'evoluzione dei pattern di guasto che caratterizza gli ambienti industriali dinamici.

Il Coordinator Agent utilizza attualmente regole di routing predefinite per decidere quale agente attivare in risposta alle richieste degli operatori. L'evoluzione del sistema prevedrebbe l'implementazione di meccanismi di feedback che permettano di raffinare queste regole basandosi sui risultati operativi reali. La raccolta di metriche sulla soddisfazione degli operatori rispetto alle risposte fornite, sui tempi di risoluzione dei problemi e sull'accuratezza delle diagnosi permetterebbe di identificare pattern nelle decisioni di routing che portano a risultati subottimali. Queste informazioni potrebbero guidare l'aggiornamento delle regole decisionali e l'ottimizzazione dei system messages utilizzati dagli agenti.

I prompt template e i system messages utilizzati dai vari agenti rappresentano componenti critici che determinano la qualità delle interazioni conversazionali. L'analisi sistematica delle conversazioni operative reali permetterebbe di identificare formulazioni che generano risposte più efficaci e comprensibili per gli operatori. L'implementazione di testing comparativo per diverse varianti di prompt consentirebbe la validazione quantitativa dell'efficacia di modifiche proposte. Questo processo iterativo di raffinamento migliorerebbe progressivamente la naturalezza delle interazioni e l'appropriatezza del linguaggio tecnico utilizzato.

L'active learning rappresenta una tecnica di machine learning dove il sistema identifica automaticamente situazioni in cui la propria conoscenza è incompleta o incerta, richiedendo specificamente input dagli esperti umani per migliorare le performance future. Nel contesto del sistema sviluppato, questa tecnica permetterebbe di identificare casi diagnostici dove i confidence scores sono bassi o dove multiple diagnosi alternative hanno probabilità simili. Quando il sistema incontra tali situazioni ambigue, potrebbe richiedere feedback specifici dagli operatori esperti, utilizzando queste informazioni per affinare i propri modelli decisionali. Questo approccio massimizzerebbe l'efficacia dell'apprendimento focalizzando l'attenzione umana sui casi più informativi piuttosto che su esempi ridondanti.

Il transfer learning è una tecnica che permette di riutilizzare modelli addestrati per un compito specifico adattandoli rapidamente a nuovi domini correlati. Nel sistema sviluppato, questa capacità consentirebbe di estendere le competenze diagnostiche acquisite per i cavi elettrici ad altri componenti industriali simili, come tubi, condotti o altri elementi cilindrici.

L'implementazione di transfer learning richiederebbe la modifica dell'architettura dei modelli per separare le caratteristiche generali (forma, texture, pattern di danno) dalle specificità del dominio (codici colore dei cavi, dimensioni standard). Questa separazione permetterebbe il riutilizzo delle competenze di base accelerando significativamente l'adattamento a nuovi contesti applicativi.

Il federated learning è un approccio di machine learning distribuito che permette a multiple installazioni del sistema di migliorare collettivamente le proprie performance condividendo pattern di apprendimento senza condividere i dati grezzi. Per il sistema multi-agente sviluppato, questa tecnica permetterebbe a diverse aziende di beneficiare delle esperienze diagnostiche collettive mantenendo la privacy dei propri dati operativi.

Ogni installazione industriale accumulerebbe esperienza locale sui propri specifici pattern di guasto e modalità operative. Periodicamente, gli aggiornamenti dei modelli (ma non i dati) verrebbero condivisi attraverso un server di coordinamento, permettendo a tutti i partecipanti di beneficiare delle competenze acquisite dall'intera rete di installazioni. Questo approccio accelererebbe significativamente l'evoluzione delle capacità diagnostiche rispetto all'apprendimento isolato di singole installazioni.

L'implementazione di questi meccanismi di apprendimento continuo trasformerebbe il sistema da strumento statico a piattaforma evolutiva che migliora progressivamente attraverso l'esperienza operativa, rappresentando un elemento chiave per la transizione verso applicazioni industriali mature e competitive.

Estensione verso Ecosistemi IoT e Digital Twin

L'evoluzione del sistema multi-agente sviluppato verso l'integrazione con ecosistemi IoT (Internet of Things) e piattaforme di digital twin rappresenta una trasformazione strategica che amplierebbe significativamente le capacità diagnostiche oltre l'analisi puramente visiva dell'implementazione attuale. Attualmente, il sistema opera come strumento di analisi reattivo: riceve un'immagine o una richiesta dall'operatore e fornisce una diagnosi basata esclusivamente su quella specifica evidenza visiva e sulla knowledge base statica.

L'integrazione con ecosistemi IoT trasformerebbe questo paradigma introducendo capacità di monitoraggio continuo attraverso l'accesso a stream di dati in tempo reale provenienti da sensori distribuiti nell'impianto industriale. Questi sensori IoT raccolgono continuamente informazioni su parametri ambientali (temperatura, umidità, pressione), vibrazioni meccaniche, correnti elettriche e altre grandezze fisiche rilevanti per la salute degli asset industriali. L'integrazione permetterebbe al sistema multi-agente di correlare le evidenze visive analizzate dal CableVisionAgent con questi parametri quantitativi, creando un quadro diagnostico molto più completo. Ad esempio, se il sistema rileva visivamente un'anomalia su un cavo, potrebbe simultaneamente verificare se ci sono stati picchi di temperatura o variazioni di corrente negli stessi periodi, confermando o raffinando la diagnosi.

Il concetto di digital twin riferisce a una replica digitale accurata di un asset fisico che viene continuamente aggiornata con dati reali. Nel contesto industriale, include modelli tridimensionali dettagliati, specifiche tecniche e cronologie operative.

Lo sviluppo di interfacce tra il sistema multi-agente e piattaforme di digital twin consentirebbe l'accesso a queste rappresentazioni digitali per simulazioni predittive.

Questa capacità di simulazione permetterebbe al sistema di anticipare potenziali problematiche modellando scenari futuri basandosi sulle condizioni operative attuali e sui trend storici. Se i sensori IoT rilevano un graduale aumento della temperatura in una specifica area e il digital twin simula che questo trend porterà a condizioni critiche, il sistema potrebbe proattivamente allertare gli operatori prima che il problema diventi critico. Questo rappresenterebbe un salto qualitativo dall'approccio puramente reattivo verso capacità di prevenzione predittiva.

L'integrazione con sistemi di manutenzione predittiva basati su machine learning permetterebbe di correlare le evidenze diagnostiche visive con pattern di degrado identificati attraverso l'analisi quantitativa dei dati operativi storici. Questi sistemi processano grandi volumi di dati telemetrici per identificare anomalie sottili nei parametri operativi che potrebbero indicare problemi emergenti. La convergenza multi-modale combinerebbe l'analisi visiva immediata con l'intelligence predittiva derivata dall'analisi dei trend operativi, migliorando significativamente l'accuratezza delle previsioni e consentendo la pianificazione ottimale degli interventi manutentivi.

La connessione con piattaforme di gestione degli asset aziendali trasformerebbe il sistema da strumento diagnostico isolato a componente integrato della strategia di asset management. Questi sistemi gestiscono l'intero ciclo di vita degli asset industriali, mantenendo registri dettagliati di performance, costi operativi e cronologie di manutenzione. I risultati delle analisi diagnostiche fornite dal sistema multi-agente diventerebbero input critici per decisioni strategiche di investimento, permettendo di quantificare l'impatto delle condizioni rilevate sui costi operativi futuri e sulla pianificazione della sostituzione degli asset.

L'implementazione di queste integrazioni richiederebbe lo sviluppo di API standardizzate per l'interfacciamento con diverse piattaforme industriali, meccanismi di data fusion per la correlazione efficace di informazioni eterogenee, e algoritmi per l'elaborazione di stream di dati ad alto volume. Questa evoluzione posiziona il sistema multi-agente sviluppato come componente centrale di un ecosistema integrato di supporto decisionale che va oltre le capacità diagnostiche puntuali dell'implementazione attuale.

Sviluppo di Interfacce Utente Avanzate

L'evoluzione verso applicazioni operative reali richiede lo sviluppo di interfacce utente ottimizzate per contesti industriali dinamici. L'integrazione con dispositivi di realtà aumentata industriale permetterebbe la visualizzazione contestuale di informazioni diagnostiche direttamente nel campo visivo dell'operatore, eliminando la necessità di consultare dispositivi separati durante le operazioni.

Lo sviluppo di interfacce vocali robuste per ambienti rumorosi rappresenta una priorità per l'adozione in contesti manifatturieri dove l'interazione tattile potrebbe non essere praticabile. L'implementazione di algoritmi di noise cancellation e riconoscimento vocale specializzati per terminologie tecniche industriali migliorerebbe significativamente l'usabilità operativa.

L'integrazione con sistemi di gesture recognition consentirebbe modalità di interazione alternative per scenari dove sia l'input vocale che quello tattile risultano impraticabili. Questa multimodalità di interazione garantirebbe accessibilità del sistema in diverse condizioni operative, mantenendo efficacia indipendentemente dai vincoli ambientali.

Lo sviluppo di interfacce adaptive che si configurano automaticamente in funzione del profilo dell'operatore, del contesto operativo e della tipologia di task migliorerebbe l'efficienza delle interazioni riducendo il carico cognitivo. Questa personalizzazione dinamica rappresenta un elemento chiave per l'adozione su larga scala in organizzazioni con operatori di diversi livelli di esperienza.

Considerazioni per l'Adozione Industriale

Mentre la validazione sperimentale ha dimostrato la fattibilità tecnica dell'architettura multi-agente proposta, il passaggio da proof-of-concept a sistema industriale operativo presenta sfide specifiche che vanno oltre gli aspetti puramente tecnologici.

L'implementazione industriale richiede una strategia di deployment graduale attraverso progetti pilota in aree controllate, dove validare le funzionalità in contesti reali minimizzando i rischi operativi. La quantificazione del valore economico diventa critica per giustificare gli investimenti: le metriche di ROI devono dimostrare impatti misurabili su parametri come riduzione dei tempi di fermo macchina, miglioramento della qualità dei prodotti e ottimizzazione dei processi manutentivi.

L'adozione richiede inoltre programmi formativi specifici che preparino gli operatori all'interazione con sistemi multi-agente intelligenti, superando la resistenza al cambiamento tipica delle innovazioni tecnologiche industriali. La formazione deve enfatizzare che il sistema potenzia le competenze umane piuttosto che sostituirle, allineandosi ai principi human-centric dell'Industria 5.0.

L'architettura modulare sviluppata facilita questa transizione permettendo integrazione graduale con ecosistemi tecnologici esistenti senza richiedere sostituzioni massive di infrastrutture consolidate. Questo approccio posiziona il sistema come ponte verso una nuova generazione di strumenti di supporto decisionale dove la collaborazione sinergica tra competenze umane e intelligenza artificiale distribuita diventa fondamentale per l'evoluzione industriale.

Bibliografia

[1] European Commission. *Industria 5.0: verso un'industria europea sostenibile, human-centric e resiliente*. 2021. URL: https://ec.europa.eu/info/research-and-innovation/research-area/industrial-research-and-innovation/industry-50_en (cit. alle pp. 1, 3).

[2] Corriere delle Comunicazioni. «Industria 4.0, l'operaio diventa augmented». In: (2016) (cit. alle pp. 1, 2, 7).

[3] «Human–robot collaboration in Industry 5.0: a human-centric AI-based approach». In: *Frontiers in Robotics and AI* (2024) (cit. a p. 7).

[4] «Reviewing human-robot collaboration in manufacturing: Opportunities and challenges in the context of industry 5.0». In: *ScienceDirect* (2024) (cit. a p. 8).

[5] «Industry 5.0: Enhancing Human-Robot Collaboration through Collaborative Robots – A Review». In: *IEEE Xplore* () (cit. a p. 8).

[6] Accenture. *Human + Machine: Reimagining Work in the Age of AI*. Rapp. tecn. Accenture Research, 2023 (cit. a p. 9).

[7] Siemens Digital Industries. *AI-Powered Quality Control in Manufacturing*. Rapp. tecn. Industry 4.0 Insights, 2023 (cit. alle pp. 10, 20).

[8] CORDIS - European Commission. *Smart solution to support human-robot collaboration*. 2023 (cit. a p. 10).

[9] Ashish Vaswani et al. «Attention Is All You Need». In: *arXiv preprint arXiv:1706.03762* (2017) (cit. a p. 12).

[10] Tom B. Brown et al. «Language Models are Few-Shot Learners». In: *arXiv preprint arXiv:2005.14165* (2020) (cit. a p. 13).

[11] Hugo Touvron et al. «LLaMA: Open and Efficient Foundation Language Models». In: (2023) (cit. a p. 13).

[12] Yuntao Bai et al. «Constitutional AI: Harmlessness from AI Feedback». In: (2022) (cit. a p. 13).

[13] Long Ouyang et al. «Training language models to follow instructions with human feedback». In: (2022) (cit. a p. 14).

[14] Hugging Face. *Illustrating Reinforcement Learning from Human Feedback (RLHF)*. 2024 (cit. a p. 14).

[15] IBM. *What Is Reinforcement Learning From Human Feedback (RLHF)?* 2025 (cit. a p. 14).

[16] AWS. *What is RLHF? - Reinforcement Learning from Human Feedback Explained*. 2025 (cit. a p. 14).

[17] Jason Wei et al. «Chain-of-Thought Prompting Elicits Reasoning in Large Language Models». In: (2022) (cit. a p. 15).

[18] LangChain Development Team. *Tool calling concepts and implementation*. LangChain Documentation. 2023 (cit. alle pp. 17, 47).

[19] OpenAI. *Function Calling and Tools API Reference*. 2023 (cit. alle pp. 17, 47).

[20] Patrick Lewis et al. «Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks». In: (2020) (cit. alle pp. 18, 36).

[21] Aaron Parisi et al. «TALM: Tool Augmented Language Models». In: (2022) (cit. a p. 19).

[22] Anthropic. *Claude Tool Use Best Practices*. 2023 (cit. a p. 19).

[23] OWASP. *LLM Application Security Guidelines*. Rapp. tecn. OWASP Foundation, 2023 (cit. a p. 19).

[24] K. Zhang et al. «Security Considerations for Tool-Augmented Language Models». In: *IEEE Security & Privacy* (2023) (cit. a p. 19).

[25] McKinsey & Company. *The State of AI in 2023: Generative AI's Breakout Year*. Rapp. tecn. McKinsey Global Institute, 2023 (cit. a p. 19).

[26] General Electric. *Predictive Maintenance with Conversational AI*. Rapp. tecn. GE Digital Solutions, 2023 (cit. a p. 20).

[27] IBM Research. *Automated Technical Documentation Generation*. Rapp. tecn. IBM AI Applications, 2023 (cit. a p. 20).

[28] Yformer. *EfficientSAM: Leveraged Masked Image Pretraining for Efficient Segment Anything*. 2024. URL: `https://github.com/yformer/EfficientSAM` (cit. a p. 36).

[29] *LangChain Official Website*. 2024. URL: `https://python.langchain.com` (cit. a p. 44).

[30] LangChain Documentation. *LangChain Expression Language (LCEL)*. 2024. URL: `https://python.langchain.com/docs/expression_language/` (cit. a p. 45).

[31] LangChain Blog. *Announcing LangChain Hub*. Set. 2023. URL: `https://blog.langchain.dev/langchain-hub/` (cit. a p. 46).

[32] LangChain AI. *LangGraph: Build reliable, stateful AI systems*. 2024. URL: `https://langgraph.langchain.com/` (cit. a p. 50).

[33] LangChain AI. *LangGraph: Library for building LLM applications as graphs*. 2024. URL: `https://github.com/langchain-ai/langgraph` (cit. a p. 50).

[34] LangChain Blog. *LangGraph: Graph-based orchestration*. 2024. URL: `https://blog.langchain.dev/langgraph/` (cit. a p. 50).

[35] Open Edge Platform. *Anomalib: An anomaly detection library comprising state-of-the-art algorithms and features*. 2024. URL: `https://github.com/openvinotoolkit/anomalib` (cit. a p. 64).

[36] MVTec Software. *MVTec Anomaly Detection Dataset*. MVTec Software GmbH. 2019. URL: `https://www.mvtec.com/company/research/datasets/mvtec-ad` (cit. a p. 70).

Ringraziamenti