



Norme di Progetto

v0.4.0

Registro delle Modifiche

Data	Versione	Descrizione	Redattore	Verificatore
2026/02/04	0.4.0	Rielaborazione delle Norme di Progetto relative al Piano di Progetto	Suar Alberto	
2026/01/15	0.3.0	Aggiunta prima versione sezione documenti	Basso Kevin	Suar Alberto
2026/01/09	0.2.1	Correzioni minori e miglioramenti alla documentazione	Basso Kevin	Martinello Riccardo
2026/01/02	0.2.0	Arricchimento con standard industriali, norme di codifica dettagliate e integrazioni progetto-specifiche	Martinello Riccardo	Suar Alberto, Basso Kevin
2025/12/28	0.1.0	Rilascio iniziale con norme, processi e strumenti aggiornati	Martinello Riccardo	Suar Alberto, Basso Kevin
2025/12/20	0.0.1	Prima bozza iniziale	Martinello Riccardo	Suar Alberto, Basso Kevin

Indice

1 Introduzione	6
1,1 Contesto del Progetto	6
1,2 Finalità del Documento	6
1,3 Glossario	6
1,4 Riferimenti	6
1,4,1 Riferimenti Normativi	6
1,4,2 Riferimenti Informativi	7
2 Processi Primari	7
2,1 Processo di Fornitura	7
2,1,1 Strumenti di supporto	8
2,1,2 Documentazione del Progetto	8
2,1,2,1 AdR - Analisi dei Requisiti	8
2,1,2,2 Glossario	8
2,1,2,3 NdP - Norme di Progetto	8
2,1,2,4 PdP - Piano di Progetto	8
2,1,2,5 PdQ - Piano di Qualifica	9
2,1,2,6 Verbali delle Riunioni	9
2,1,2,7 Diario di bordo	9
2,2 Processo di Sviluppo	10
2,2,1 Attività del Processo di Sviluppo	10
2,2,1,1 Definizione del processo:	10
2,2,1,2 Analisi dei Requisiti:	10
2,2,1,3 Progettazione dell'architettura di sistema:	10
2,2,1,4 Analisi dei Requisiti Software:	10
2,2,1,5 Progettazione dell'architettura software:	10
2,2,1,6 Progettazione di dettaglio:	10
2,2,1,7 Codifica e test delle unità:	10
2,2,1,8 Integrazione software:	10
2,2,1,9 Test di qualifica del software:	10
2,2,1,10 Integrazione e test di sistema:	10
2,2,1,11 Supporto all'accettazione:	10
2,2,2 Relazione con le Baseline di progetto	10
2,2,2,1 Analisi dei Requisiti	11
2,2,2,2 Codifica	11
2,2,2,2,1 Convenzioni	11
3 Processi di Supporto	12
3,1 Documentazione	12
3,1,1 Strumenti di Redazione	12
3,1,2 Gestione della Configurazione	12
3,1,3 Branching Strategy	12
3,1,4 Datazione e versionamento documenti	12
3,1,5 Configurazione dei Documenti	12
3,2 Qualifica	13
3,2,1 Verifica	13
3,2,1,1 Implementazione	13
3,2,1,1,1 Verifica	13

3,2,2 Validazione	15
3,2,2,1 Implementazione	15
4 Processi Organizzativi	15
4,1 Ruoli	15
4,1,1 Gestione dei task e allineamento	17
4,1,1,1 Riunioni di allineamento	17
4,1,1,1,1 Interne	17
4,1,1,1,2 esterne	17
4,1,1,1,3 Verbale	17
4,1,1,2 Comunicazione diretta	17
4,1,1,3 Strumenti di supporto all'organizzazione	17
4,1,1,4 Creazione e gestione degli strumenti organizzativi	18
4,1,1,4,1 Repository GitHub	18
4,2 Processo di miglioramento	18
4,3 Processo di formazione	19
5 Metriche	19
5,1 Strategie di Verifica	20
5,2 Software Quality Assurance <u>SQA</u>	20
5,3 Validazione	20

Indice tabelle

Table 1	Tabella riepilogativa dei documenti di progetto	9
Table 2	Ruoli di progetto e relative responsabilità	16
Table 3	Tecnologie e fonti di formazione	19
Table 4	Tabella delle metriche di progetto	20

1 Introduzione

1,1 Contesto del Progetto

Il presente documento descrive le Norme di Progetto relativo al progetto Code Guardian, commissionato dall'azienda Var Group e realizzato dal gruppo di studenti Skarab Group nell'ambito del corso di Ingegneria del Software presso l'Università degli Studi di Padova.

L'obiettivo del progetto è lo sviluppo di una piattaforma ad agenti per l'audit e la remediation automatizzata delle vulnerabilità presenti nei repository di codice sorgente, in conformità a quanto specificato nel capitolato C2. La piattaforma mira a supportare l'analisi statica del codice e l'individuazione di criticità di sicurezza, fornendo indicazioni di correzione mediante meccanismi automatizzati basati su modelli di linguaggio (LLM).

Ecco una rielaborazione con un registro formale e coerente con l'esempio fornito, mantenendo chiarezza, precisione terminologica e stile documentale:

—

1,2 Finalità del Documento

Le Norme di Progetto definiscono il quadro di riferimento metodologico, organizzativo e tecnologico adottato per lo sviluppo del sistema software.

Il documento ha valore vincolante per l'intero gruppo di lavoro e persegue i seguenti obiettivi: **stabilire un insieme di regole comuni e convenzioni condivise atte a garantire uniformità nello sviluppo del prodotto software**; individuare e descrivere gli strumenti, le tecnologie e gli ambienti di lavoro adottati dal gruppo; **formalizzare i processi operativi, organizzativi e di controllo applicati nel corso del progetto**; assicurare la coerenza delle attività progettuali con gli standard industriali e le buone pratiche di riferimento.

Il rispetto delle norme definite nel presente documento è obbligatorio per tutti i membri del gruppo di lavoro e costituisce un requisito fondamentale per garantire qualità, tracciabilità e controllo dell'intero ciclo di sviluppo.

1,3 Glossario

Al fine di prevenire ambiguità interpretative, è stato redatto un glossario che definisce in modo univoco la terminologia tecnica, gli acronimi e i concetti di dominio utilizzati all'interno della documentazione.

Nel testo, **ogni termine evidenziato tramite sottolineatura**, qualora **non sia esplicitamente indicato come collegamento a un documento o a una sezione specifica**, rimanda alla voce corrispondente del Glossario pubblicato sul sito ufficiale del gruppo, consentendo al lettore di accedere direttamente alla definizione associata.

La versione più recente del Glossario è disponibile al seguente link: [Link al Glossario](#).

1,4 Riferimenti

1,4,1 Riferimenti Normativi

Il progetto adotta standard industriali riconosciuti al fine di garantire qualità, conformità e aderenza alle best practices nel ciclo di vita del software. In particolare:

- **Capitolato C2:** Piattaforma ad agenti per l'audit e la remediation dei repository software.
<https://www.math.unipd.it/~tullio/IS-1/2025/Progetto/C2.pdf>
(ultimo accesso: 31/01/2026)

- **IEEE 830 – Standard for Software Requirements Specifications:** guida alla redazione e classificazione dei requisiti funzionali e non funzionali. (ultimo accesso: **02/01/2026**)
- **IEEE 1016 – Recommended Practice for Software Design Descriptions:** linee guida per la descrizione dell'architettura software e dei principali design pattern. (ultimo accesso: **02/01/2026**)
- **IEEE 829 – Standard for Software and System Test Documentation:** definizione della struttura dei piani di test, casi di test, procedure e report. (ultimo accesso: **02/01/2026**)
- **ISO/IEC 12207 – Software Life Cycle Processes:** framework internazionale per la gestione dei processi di sviluppo, manutenzione e dismissione del software. (ultimo accesso: **02/01/2026**)

Tali standard sono integrati nei processi di analisi, progettazione, verifica e documentazione descritti nelle presenti Norme di Progetto, al fine di assicurare tracciabilità, qualità e controllo sistematico delle attività.

1,4,2 Riferimenti Informativi

- **Dispense del corso di Ingegneria del Software relative ai Processi di Ciclo di Vita del Software.**

<https://www.math.unipd.it/~tullio/IS-1/2025/Dispense/T02.pdf>

(ultimo accesso: **31/01/2026**)

- **Dispense del corso di Ingegneria del Software relative alla Gestione di Progetto.**

<https://www.math.unipd.it/~tullio/IS-1/2025/Dispense/T04.pdf>

(ultimo accesso: **31/01/2026**)

2 Processi Primari

2,1 Processo di Fornitura

La fornitura è il processo adottato dal fornitore del prodotto finale, il quale scopo è quello di acquisire i requisiti del cliente e fornire il prodotto conforme a tali requisiti. Tale processo comprende una fase iniziale di studio dei requisiti che il progetto dovrà soddisfare. Da questa analisi deriva il materiale necessario per avviare la contrattazione dei requisiti con il proponente e per comunicare una pianificazione preliminare delle attività, includendo una stima della data di consegna. Il processo di fornitura si compone di molte attività, tra cui:

- Inizializzazione della fornitura
 - Momento in cui il fornitore analizza le richieste del proponente e valuta la sua capacità di soddisfarle.
- Preparazione della proposta
 - Redazione della proposta tecnica ed economica in risposta al capitolato.
- Negoziazione e contrattazione
 - Discussione e definizione dei termini contrattuali con il proponente.
- Pianificazione della fornitura
 - Viene stabilita l'organizzazione del lavoro, le risorse necessarie, il modello di ciclo di vita da adottare oltre che la valutazione dei rischi.
- Esecuzione della fornitura
 - Realizzazione del prodotto secondo i requisiti concordati.
- Valutazione della fornitura
 - Verifica della conformità del prodotto ai requisiti e alla qualità attesa, questo viene fatto mantenendosi in costante contatto con la proponente.
- Consegna del prodotto
 - Rilascio formale del prodotto al proponente, accompagnato dalla documentazione necessaria.

2,1,1 Strumenti di supporto

- **GitHub**: Per il versionamento del codice e la gestione del repository.
- **Telegram, Discord**: Per la comunicazione interna del team.
- **Draw.io**: Per la creazione di diagrammi UML e altri schemi visivi.
- **Jira**: Per la gestione delle attività e il monitoraggio del progresso.
- **VS Code**: Per la scrittura del codice e la redazione dei documenti in Typst.
- **gmail**: Per la comunicazione con il proponente.
- **Google calendar**: Per la pianificazione dei requisiti funzionali e non funzionali, classificati in obblighi la pianificazione delle riunioni e delle scadenze.
- **Microsoft Teams**: Per le riunioni virtuali e la collaborazione in tempo reale.

2,1,2 Documentazione del Progetto

In questa sezione si elencano tutti i documenti che Skarab Group produrrà durante il ciclo di vita del progetto Code Guardian. I documenti sono tutti scritti utilizzando il linguaggio typst per garantire una formattazione uniforme e professionale.

2,1,2,1 AdR - Analisi dei Requisiti

Documento che raccoglie e specifica i requisiti funzionali e non funzionali del sistema software da sviluppare, basato sul capitolato fornito dal Proponente. Questo documento contiene:

- Introduzione al progetto e obiettivi.
- Lista degli attori coinvolti.
- Lista dei casi d'uso.
 - Essi sono descritti in UCx.y.z. ...
 - UC è un acronimo che sta per usecase, x indica il numero del caso d'uso, y,z e i numeri successivi sono i sotto casi d'uso, ovvero inclusioni, composizioni o estensioni del caso d'uso principale.
- Lista dei requisiti funzionali e non funzionali, classificati in obbligatori, desiderabili e opzionali.

TODO: come sono acronimizzati i requisiti?

2,1,2,2 Glossario

Documento che definisce i termini tecnici e specifici utilizzati nel progetto, per garantire una comprensione comune tra tutti i membri del team e gli stakeholder esterni. Questo documento serve a fornire definizioni chiare e univoche per evitare ambiguità. Oltre che il documento in sé per sé, il glossario è consultabile tramite una pagina nel sito web di Skarab Group. Questo permette interattività e link esterni all'interno di ogni documento che compone la documentazione di progetto. Per convenzione, un termine sarà un link al glossario sul sito solo la prima volta che compare all'interno di un documento, assumendo una lettura ordinata del documento stesso da parte del lettore.

2,1,2,3 NdP - Norme di Progetto

Documento che definisce il WoW, le norme, le convenzioni, gli strumenti e i processi che i membri del gruppo devono seguire durante il ciclo di vita del progetto.

2,1,2,4 PdP - Piano di Progetto

Documento che descrive l'organizzazione del lavoro, le risorse, il modello di ciclo di vita adottato, la pianificazione delle attività divise per sprint e la valutazione dei rischi associati al progetto. Inoltre, il PdP riporta attivamente il monitoraggio del lavoro di ogni componente del gruppo, con l'obiettivo di garantire il rispetto delle tempistiche e le ore di lavoro previste per i vari ruoli.

2,1,2,5 PdQ - Piano di Qualifica

Documento che definisce le strategie, le metodologie e le attività di verifica e validazione per garantire che i prodotti del progetto soddisfino i requisiti specificati. Include la definizione delle metriche di qualità, i piani di test e le procedure di controllo qualità.

2,1,2,6 Verbali delle Riunioni

Sono documenti che riportano in modo sintetico e chiaro i punti discussi durante le riunioni, con le decisioni prese e i compiti assegnati. Ogni verbale include:

- Data, ora e luogo della riunione.
- Elenco dei partecipanti.
- Ordine del giorno.
- Sintesi delle discussioni.
- Decisioni prese e azioni da intraprendere.

I verbali possono essere di 2 tipi:

- **Verbali Interni:** Riunioni tra i membri del gruppo per coordinare le attività di progetto.
- **Verbali Esterni:** Riunioni con il Proponente esterni per discutere requisiti, progressi e feedback.

2,1,2,7 Diario di bordo

Documento che permette un allineamento settimanale con il professor Vardanega Tullio, in cui il redattore riporta:

- Attività svolte durante la settimana.
- Problemi incontrati e soluzioni adottate.
- Pianificazione delle attività per la settimana successiva.

Tutti i documenti devono avere un formato uniforme e professionale strutturato in questo modo:

- Prima pagina con logo e nome del team e versione attuale del documento.
- Intestazioni e piè di pagina con titolo del documento e numero di pagina.
- Indice automatico.
- Indice delle tabelle (se presenti).
- Contenuto.

Documento	Redattori	Destinatari	Uso
Lettera di Presentazione	Responsabile	Proponente, Cardin Riccardo, Vardanega Tullio, Skarab Group	Esterno
AdR – Analisi dei Requisiti	Analisti	Proponente, Cardin Riccardo, Vardanega Tullio, Skarab Group	Esterno
Glossario	Tutti i membri del gruppo	Stakeholder interni ed esterni	Esterno
NdP – Norme di Progetto	Responsabili di processo	Skarab Group, Cardin Riccardo, Vardanega Tullio	Interno
PdP – Piano di Progetto	Project Manager	Proponente, Cardin Riccardo, Vardanega Tullio, Skarab Group	Esterno
PdQ – Piano di Qualifica	Responsabili Qualità	Skarab Group, Cardin Riccardo, Vardanega Tullio, Proponente	Esterno
Verbali delle Riunioni	Segretario della riunione	Proponente (se esterni), Cardin Riccardo, Vardanega Tullio, Skarab Group	Interno / Esterno

Table 1: Tabella riepilogativa dei documenti di progetto

2,2 Processo di Sviluppo

Il Processo di Sviluppo adottato da Skarab Group definisce l'insieme delle attività necessarie alla realizzazione del prodotto software, dalla definizione dei requisiti fino alla consegna e accettazione del sistema finale. Tale processo è conforme allo standard ISO/IEC 12207:1995 e garantisce un approccio strutturato, tracciabile e orientato alla qualità.

2,2,1 Attività del Processo di Sviluppo

Il processo comprende le seguenti macro-attività:

2,2,1,1 Definizione del processo:

Selezione del ciclo di vita del software più idoneo in base alla complessità, agli obiettivi e ai vincoli del progetto.

2,2,1,2 Analisi dei Requisiti:

Individuazione e formalizzazione delle esigenze dell'utente finale e dei vincoli imposti dal proponente, includendo funzionalità, requisiti di qualità e limitazioni progettuali.

2,2,1,3 Progettazione dell'architettura di sistema:

identificazione delle componenti hardware e software necessarie a soddisfare i requisiti, garantendo la tracciabilità degli stessi.

2,2,1,4 Analisi dei Requisiti Software:

studio di come il software risponde ai requisiti utente, includendo aspetti funzionali, prestazionali, di sicurezza e di interfaccia.

2,2,1,5 Progettazione dell'architettura software:

definizione della struttura complessiva del sistema software e delle relazioni tra le sue componenti, senza entrare nel dettaglio implementativo.

2,2,1,6 Progettazione di dettaglio:

specificazione delle singole unità software che compongono il sistema.

2,2,1,7 Codifica e test delle unità:

implementazione del codice sorgente e verifica del corretto funzionamento delle singole unità.

2,2,1,8 Integrazione software:

combinazione progressiva delle componenti software, supportata da test di integrazione.

2,2,1,9 Test di qualifica del software:

verifica della conformità del prodotto agli obiettivi di qualità definiti.

2,2,1,10 Integrazione e test di sistema:

assemblaggio del sistema completo e validazione del suo comportamento complessivo. Installazione: rilascio del software nell'ambiente concordato con il cliente.

2,2,1,11 Supporto all'accettazione:

assistenza al proponente durante la fase di verifica finale del prodotto.

2,2,2 Relazione con le Baseline di progetto

In relazione alle baseline previste dal progetto, (RTB e PB), le attività di sviluppo si più significative sono:

2,2,2,1 Analisi dei Requisiti

L'Analisi dei Requisiti rappresenta una fase centrale del progetto e ha lo scopo di identificare in modo completo e non ambiguo tutti i requisiti che il sistema deve soddisfare. Essa è documentata nell'apposito documento di Analisi dei Requisiti e costituisce il riferimento principale per le successive attività di progettazione, sviluppo e verifica.

I Casi d'Uso sono identificati tramite una nomenclatura univoca che ne garantisce la tracciabilità, la nomenclatura è la seguente: UCx.y.z, dove:

- UC indica che si tratta di un Caso d'Uso.
- x è un numero intero che identifica il Caso d'Uso principale.
- y, z e i numeri successivi sono numeri interi che identificano sotto-casi d'Uso, ovvero inclusioni, composizioni o estensioni del Caso d'Uso principale.

I requisiti, invece < **TODO: come sono acronimizzati i requisiti?** >

2,2,2,2 Codifica

La codifica ha come obiettivo l'implementazione fedele delle soluzioni progettate, nel rispetto degli standard di qualità definiti dal gruppo.

I linguaggi di programmazione utilizzati per il progetto sono TypeScript per il frontend e gli script di automazione, Python e Node.js per il backend e gli agenti intelligenti e MongoDB per la gestione del database.

Mentre per la documentazione viene utilizzato il linguaggio Typst.

2,2,2,2,1 Convenzioni

Per garantire uniformità e manutenibilità del codice, sono adottate delle convenzioni di codifica comuni a tutti i membri del gruppo. Queste convenzioni riguardano:

- Nomenclatura di variabili, funzioni, classi e componenti, in particolare, si utilizza il camelCase per variabili e funzioni, e il PascalCase per classi e componenti.
- Struttura e organizzazione del codice sorgente, con l'utilizzo di moduli e cartelle per separare le diverse funzionalità.
- Commenti e documentazione del codice, ogni funzione deve possedere un commento che la descriva, i parametri in ingresso e il valore di ritorno.
- Evitare funzioni lambda complesse e mantenere le funzioni con una singola responsabilità.

3 Processi di Supporto

3,1 Documentazione

Il ciclo di vita dei documenti prevede:

1. **Creazione/Modifica:** Redazione o aggiornamento del contenuto in formato source (Typst).
2. **Verifica:** Controllo di conformità (ortografia, stile, contenuti) da parte di un membro diverso dal redattore.
3. **Approvazione:** Validazione finale per il rilascio.

3,1,1 Strumenti di Redazione

- **Visual Studio Code:** Editor principale per la scrittura dei documenti.
- **Typst:** Linguaggio di markup utilizzato per la formattazione dei documenti. Questo linguaggio permette di creare documenti con una struttura chiara e professionale, facilitando la gestione di elementi come intestazioni, tabelle, figure e riferimenti incrociati. Inoltre, si possono creare funzioni personalizzate per automatizzare parti della formattazione.
- **Tinymist:** Estensione per VS Code che fornisce supporto e preview per Typst

3,1,2 Gestione della Configurazione

Utilizzo di Git come sistema di controllo versione.

3,1,3 Branching Strategy

- **main:** Ramo stabile, contenente le versioni rilasciate o pronte al rilascio.
- **develop:** Ramo di integrazione principale.
- **feature/nome-feature:** Rami per lo sviluppo di nuove funzionalità.
- **fix/nome-fix:** Rami per la correzione di bug.
- **'acronimo-documento':** Rami dedicati alla redazione dei documenti (es. NdP, PdP, PdQ, AdR).

Il ramo develop é protetto, questo vuol dire che ogni modifica deve essere effettuata tramite **pull request** e sottoposta a revisione da parte di almeno un membro del gruppo diverso dal redattore prima di essere unita. Questo assicura che ogni documento sia verificato prima di entrare nel branch.

3,1,4 Datazione e versionamento documenti

Ogni documento deve includere una sezione di cronologia delle versioni che riporta:

- Data della modifica.
- Numero di versione
 - Il numero della versione deve seguire lo schema $x.y.z$, dove:
 - x : Major version, incrementata per cambiamenti significativi o incompatibili.
 - y : Minor version, incrementata per l'aggiunta di funzionalità mantenendo la compatibilità.
 - z : Patch version, incrementata per correzioni di bug o modifiche minori.
- Descrizione delle modifiche apportate.
- Nomi dei redattori coinvolti.
- Nomi dei verificatori coinvolti.

3,1,5 Configurazione dei Documenti

Tutti i documenti devono essere archiviati nel repository GitHub del progetto, organizzati in cartelle in base a Quale sezione del progetto appartengono, se appartengono al sito, ad una baseline specifica o ad altro. Ogni documento deve essere salvato in formato source (Typst) per permettere modifiche future e versionamento.

Dal momento che SkarabGroup utilizza Jira per la gestione delle attività di progetto, ogni documento deve essere associato all'attività corrispondente in Jira . Questo facilita il tracciamento delle modifiche e la gestione delle responsabilità durante i vari sprint.

Alla fine di ogni sprint, i documenti completati e verificati devono essere uniti nel ramo develop tramite pull request, seguendo la procedura di revisione stabilita. questo permette di iniziare lo sprint successivo con una base documentale aggiornata e approvata in quanto, la p[rima attività di ogni sprint é quella di fare un pull del develop sul branch del documento a cui si sta lavorando.

3,2 Qualifica

3,2,1 Verifica

La verifica ha l'obiettivo di accertare che i prodotti soddisfino i requisiti, ovvero di rispondere alla domanda: "Did I build the sistem right?".

Tutti i prodotti del progetto sono soggetti a verifiche periodiche per garantire la conformità agli standard di qualità definiti nel Piano di Qualifica (PdQ).

3,2,1,1 Implementazione

Dal momento in cui per poter portare un documento, o parte di esso nel branch principale develop, esso deve essere stato verificato e approvato, la verifica viene implementata tramite il sistema di **pull request** di GitHub. Questo assicura che tutto il codice e la documentazione presenti nel ramo principale abbiano superato un processo di verifica e che, quindi, siano conformi agli standard di qualità stabiliti.

3,2,1,1,1 Verifica

La Verifica rappresenta un processo fondamentale per Skarab Group e accompagna l'intero ciclo di vita del progetto (a.a. 2025/2026), con l'obiettivo di garantire la correttezza, la qualità e la conformità dei prodotti realizzati rispetto ai requisiti definiti. Tutte le informazioni relative agli esiti delle attività di verifica, incluse misurazioni e risultati dei test, sono documentate nel Piano di Qualifica.

3,2,1,1,1,1 Verifica della Documentazione

Nelle fasi iniziali del progetto, particolare attenzione è stata dedicata alla verifica della documentazione. Ogni documento, una volta redatto, viene sottoposto a verifica prima dell'integrazione nel ramo principale del repository. L'attività comprende:

- controllo grammaticale e sintattico;
- verifica della correttezza e coerenza dei contenuti.
- Le modalità operative di redazione e verifica dei documenti sono descritte nelle regole di gestione dei branch e delle issue.

3,2,1,1,1,2 Verifica del Codice

Le attività di verifica del codice saranno approfondite in modo più dettagliato una volta raggiunta la Requirements and Technology Baseline (RTB). In generale, la verifica del software si articola in due approcci complementari: Analisi Statica e Analisi Dinamica.

3,2,1,1,1,3 Analisi Statica

L'Analisi Statica comprende tutte le attività di verifica che non richiedono l'esecuzione dell'oggetto analizzato. Essa si concentra sull'individuazione di errori sintattici, strutturali o concettuali, permettendo di prevenire problemi prima della fase di esecuzione.

Può essere svolta tramite:

- metodi formali, basati su dimostrazioni matematiche;
- metodi di lettura, più flessibili e applicabili alla documentazione e al codice.

Tra i metodi di lettura:

il Walkthrough, che prevede un'analisi completa e approfondita dell'oggetto, ma risulta costoso e poco automatizzabile; l'Ispezione, basata su checklist di controllo, meno esaustiva ma più efficiente e facilmente automatizzabile. Considerata l'elevata quantità di artefatti da verificare, Skarab Group privilegia l'ispezione.

3,2,1,1,1,4 Analisi Dinamica

L'Analisi Dinamica prevede l'esecuzione dell'oggetto da verificare e si basa sull'identificazione delle failure, ossia comportamenti non conformi alle aspettative, al fine di individuare e correggere i relativi fault. Questa analisi è realizzata tramite Test, che devono essere:

- ripetibili;
- indipendenti dall'ambiente di esecuzione;
- automatizzabili.

L'automazione dei test è supportata dall'uso di driver, stub e logger per simulare e monitorare l'esecuzione del software.

3,2,1,1,1,5 Tipologie di Test

I test adottati nel progetto si suddividono nelle seguenti categorie:

Test di Unità

Test di Integrazione

Test di Sistema

Test di Regressione

Ogni test è identificato da una nomenclatura standard e possiede uno stato che ne indica il livello di implementazione o superamento.

3,2,1,1,1,5,1 Test di Unità

I Test di Unità verificano il corretto funzionamento delle singole unità software. Si distinguono in:

- test funzionali (black-box), che analizzano input e output;
- test strutturali, che verificano i percorsi logici e le istruzioni eseguite.

3,2,1,1,1,5,2 Test di Integrazione

I Test di Integrazione verificano la corretta interazione tra le unità. L'integrazione può avvenire secondo due strategie:

Top-Down, partendo dalle componenti di livello più alto;

Bottom-Up, partendo dalle componenti di base.

3,2,1,1,1,5,3 Test di Sistema

I Test di Sistema valutano il comportamento del sistema nel suo complesso, verificandone la conformità ai requisiti e l'efficacia globale.

3,2,1,1,1,5,4 Test di Regressione

I Test di Regressione garantiscono che le modifiche apportate al sistema non reintroducano errori precedentemente risolti. In caso di correzione di un fault, l'intera suite di test viene rieseguita per assicurare la stabilità del sistema.

3,2,2 Validazione

La Validazione ha l'obiettivo di accertare che il prodotto soddisfi le esigenze della proponente e le aspettative degli utenti finali, rispondendo alla domanda: "Did I build the right system?".

3,2,2,1 Implementazione

La validazione viene implementata attraverso test di accettazione e raccolta di feedback dagli utenti finali. Queste attività assicurano che il prodotto finale risponda alle esigenze e ai requisiti definiti all'inizio del progetto.

4 Processi Organizzativi

I processi organizzativi sono tutti quei processi di supporto che permettono al gruppo di lavorare in modo efficiente e coordinato. La gestione dei processi organizzativi include la definizione di ruoli, responsabilità, comunicazione e gestione delle riunioni.

4,1 Ruoli

La seguente tabella riporterà tutti i ruoli presenti all'interno del progetto con le relative responsabilità.

Ruolo	Responsabilità
Responsabile	Coordina e supervisiona l'intero progetto, garantendo il rispetto delle scadenze e del budget. Rappresenta il team nei rapporti con il committente e gli stakeholder esterni. Pianifica le attività, assegna le risorse, monitora l'avanzamento attraverso metriche e indicatori, gestisce i rischi e le criticità. Approva i documenti ufficiali e autorizza le spese. Convoca e presiede le riunioni, facilita la comunicazione interna ed esterna, risolve conflitti e prende decisioni strategiche per il successo del progetto.
Amministratore	Gestisce l'infrastruttura tecnica e organizzativa del progetto. Configura e mantiene gli strumenti di versionamento, continuous integration, issue tracking e documentazione. Amministra i repository, definisce workflow e convenzioni, monitora l'ambiente di sviluppo. Gestisce la documentazione di processo, redige le norme di progetto, mantiene aggiornato il piano di qualifica. Supporta il team nella risoluzione di problemi tecnici relativi all'ambiente di lavoro, garantisce backup e sicurezza dei dati, ottimizza i processi di automazione.
Verificatore	Assicura la qualità dei prodotti attraverso attività di verifica sistematica. Controlla la conformità dei documenti alle norme redazionali e tipografiche, verifica la correttezza tecnica e la coerenza dei contenuti. Esegue review del codice secondo checklist predefinite, controlla il rispetto degli standard di programmazione. Verifica la tracciabilità tra requisiti, design e implementazione. Redige verbali di verifica, segnala anomalie e non conformità, propone azioni correttive. Monitora le metriche di qualità e valida i test eseguiti dai programmatori.
Analista	Raccoglie, analizza e formalizza i requisiti del sistema attraverso l'interazione con committente e stakeholder. Studia il dominio applicativo, comprende le esigenze degli utenti, identifica vincoli e opportunità. Redige l'Analisi dei Requisiti classificando requisiti funzionali, di qualità, di vincolo e prestazionali. Definisce casi d'uso, scenari operativi e modelli del dominio. Mantiene la tracciabilità tra requisiti e fonti, gestisce l'evoluzione dei requisiti durante il progetto. Collabora con i progettisti per garantire la fattibilità tecnica delle soluzioni proposte.
Progettista	Definisce l'architettura del sistema e il design dettagliato dei componenti software. Individua pattern architetturali appropriati, decompone il sistema in moduli, definisce interfacce e dipendenze. Specifica la struttura delle classi, i diagrammi di sequenza e collaborazione, le strutture dati e gli algoritmi principali. Redige la Specifica Tecnica e il Piano di Qualifica nella parte relativa ai test di integrazione e sistema. Effettua scelte tecnologiche motivate, considera aspetti di manutenibilità, estensibilità e performance. Fornisce ai programmatori le specifiche di dettaglio necessarie all'implementazione.
Programmatore	Implementa il codice seguendo le specifiche fornite dai progettisti e rispettando gli standard di codifica definiti. Scrive codice pulito, manutenibile e ben documentato. Sviluppa unit test per verificare la correttezza delle singole unità software, garantendo un'adeguata code coverage. Gestisce le dipendenze, utilizza strumenti di build automation, integra librerie esterne. Documenta il codice con commenti significativi e mantiene aggiornata la documentazione tecnica. Esegue debugging e risolve difetti segnalati, ottimizza le performance quando necessario. Collabora con verificatori e progettisti per migliorare continuamente la qualità del software.

Table 2: Ruoli di progetto e relative responsabilità

4,1,1 Gestione dei task e allineamento

La gestione dei task avviene tramite **Jira**, dove ogni membro del gruppo ha accesso alle proprie attività assegnate. Ogni task include una descrizione dettagliata, una stima del tempo necessario per il completamento e una scadenza. I membri del gruppo sono responsabili di aggiornare lo stato dei propri task in **Jira**, segnalando eventuali impedimenti o ritardi. Per garantire un allineamento costante, si tengono riunioni settimanali di team in cui si discutono i progressi, si risolvono problemi e si pianificano le attività future.

4,1,1,1 Riunioni di allineamento

4,1,1,1,1 Interne

Le riunioni di allineamento si tengono settimanalmente e hanno lo scopo di:

- Condividere lo stato di avanzamento delle attività.
- Identificare e risolvere eventuali ostacoli o problemi.
- Pianificare le attività per la settimana successiva.

Le riunioni vengono svolte su Discord e permettono al Responsabile di Progetto di monitorare l'andamento del lavoro e di garantire che tutti i membri del team siano allineati sugli obiettivi comuni.

4,1,1,1,2 esterne

Le riunioni esterne si tengono con il Proponente e altri stakeholder per:

- Presentare lo stato di avanzamento del progetto.
- Discutere requisiti, feedback e modifiche.
- Allineare le aspettative e pianificare le prossime fasi.

é compito del Responsabile di Progetto organizzare e condurre queste riunioni, assicurando una comunicazione efficace tra il team di sviluppo e gli stakeholder esterni.

4,1,1,1,3 Verbale

Per ogni riunione, interna o esterna, viene redatto un verbale che riporta:

- Data, ora, luogo e partecipanti.
- Ordine del giorno.
- Riassunto delle discussioni.
- Decisioni prese e task assegnati.

4,1,1,2 Comunicazione diretta

La comunicazione interna del team avviene tramite varie piattaforme:

- Whatsapp: Per comunicazioni rapide e urgenti, tramite un gruppo dedicato, utilizzato per aggiornamenti veloci e coordinamento immediato giornalmente.
- Telegram: Per discussioni più strutturate e condivisione di file, utilizzato per comunicazioni di gruppo e condivisione di documenti importanti.
- Discord: Per riunioni virtuali e collaborazione in tempo reale, utilizzato per meeting settimanali e sessioni di lavoro condivise.
- Email: Per comunicazioni formali e invio di documenti ufficiali, utilizzato per comunicazioni con il proponente e con i professori ed é gestito principalmente dal Responsabile di Progetto.
- Jira: Per la gestione delle attività e il monitoraggio del progresso, utilizzato per assegnare task, tracciare lo stato di avanzamento e gestire le scadenze.

4,1,1,3 Strumenti di supporto all'organizzazione

Come già riportato in precedenza, il gruppo utilizza diversi strumenti per supportare l'organizzazione del lavoro anche in modo totalmente asincrono:

- Google calendar: Per la pianificazione delle riunioni esterne.
- Scripts di automazione: Per automatizzare attività ripetitive e migliorare l'efficienza del lavoro.
 - queste comprendono, al momento del PoC: generazione glossario, sia in formato documento che in pagina web
- Repository GitHub: Per il versionamento del codice e la gestione della documentazione di progetto.

4,1,1,4 Creazione e gestione degli strumenti organizzativi

La creazione e gestione degli strumenti organizzativi è responsabilità dell'Amministratore di Progetto, che si occupa di:

- Configurare e mantenere gli strumenti di comunicazione e gestione delle attività.
- Assicurare che tutti i membri del team abbiano accesso agli strumenti necessari.
- Monitorare l'utilizzo degli strumenti e proporre miglioramenti o cambiamenti se necessario.

4,1,1,4,1 Repository GitHub

L'Amministratore di Progetto è responsabile della gestione del repository GitHub, che include:

- Creazione e organizzazione delle cartelle per i documenti di progetto.
- Definizione delle regole di branching e delle convenzioni di commit.
- Monitoraggio delle pull request e delle revisioni del codice.
- Assicurare che la documentazione sia aggiornata e accessibile a tutti i membri del team.

La repository DocumentazioneProgetto è organizzata in questo modo:

- .github/: Cartella contenente le configurazioni specifiche di GitHub, come azioni automatizzate e modelli di issue.
- .github/workflows/: Contiene i file di configurazione per le GitHub Actions utilizzate nel progetto.
- assets/: Cartella per le risorse multimediali utilizzate nei documenti (immagini, grafici, ecc.).
- diarioDiBordo/: Contiene il diario di bordo del progetto.
- Glossario/: Contiene l'infrastruttura necessaria per l'html del glossario dei termini tecnici utilizzati nel progetto.
- lib/: Cartella che contiene tutti i file che copntengono le funzioni utilizzate nbei vari documenti typst.
- RTB/: Cartella dedicata alla Requirements and Technology Baseline, contenente tutti i documenti relativi a questa fase del progetto.
- verbali: Cartella che contiene tutti i verbali delle riunioni interne ed esterne.
- website/: Cartella contenente i file sorgente del sito web di Skarab Group.

4,1,1,4,1,1 Librerie .typ

La cartella lib/ contiene librerie di funzioni Typst che sono utilizzate in vari documenti del progetto per garantire coerenza e riusabilità del codice. Questa cartella è creata e gestita dall'amministratore del progetto, questo permette di avere un punto centrale dove vengono definite tutte le funzioni comuni, facilitando la manutenzione e l'aggiornamento delle stesse.

4,2 Processo di miglioramento

Il processo di miglioramento continuo è fondamentale per garantire che i prodotti del progetto soddisfino gli standard di qualità definiti e le aspettative degli stakeholder. QWuesto processo si articola in 3 fasi principali:

- Inizializzazione: Fase in cui vengono definiti i processi organizzativi che guideranno il lavoro del team.
- Valutazione: Fase in cui vengono monitorati i processi e i prodotti attraverso metriche specifiche.

- Miglioramento: Fase in cui vengono implementate azioni correttive basate sui risultati della valutazione.

4,3 Processo di formazione

Il processo di formazione è progettato per garantire che tutti i membri del team abbiano le competenze necessarie per svolgere efficacemente i propri ruoli all'interno del progetto. Questo processo include:

- Identificazione delle competenze chiave richieste per ogni ruolo.
- Pianificazione di sessioni di formazione e workshop.
- Monitoraggio del progresso formativo e valutazione delle competenze acquisite.

Le ore svolte per formazione non sono rendicontate in quanto non fanno parte del lavoro di progetto vero e proprio ma della parte di 'palestra' che il corso universitario vuole fornire agli studenti. La seguente tabella riporta le fonti utilizzate per la formazione dei membri del gruppo in base alla tecnologia studiata.

Tecnologia	Fonte di formazione
Python	Documentazione ufficiale Python, youtube tutorial
Agenti AI	Documentazione framework specifici, paper accademici, documentazione AWS
AWS	documentazione ufficiale AWS
Git	Documentazione ufficiale Git, guide interattive, corso di Metodi e tecnologie per lo sviluppo software
Node.js	Documentazione ufficiale Node.js, tutorial youtube
MongoDB	documentazione ufficiale,

Table 3: Tecnologie e fonti di formazione

Inoltre l'azienda proponente Var group ha tenuto delle lezioni specifiche per ognuna delle tecnologie necessarie allo sviluppo del progetto e ha dato disponibilità per la risoluzione di dubbi e domande.

5 Metriche

Il gruppo adotta un sistema di metriche per monitorare processi e prodotti.

ID	Descrizione	Soglia / Obiettivo
<u>MPC1</u> (SV)	Schedule Variance: differenza tra lavoro pianificato ed eseguito	≥ 0
<u>MPC2</u> (BV)	Budget Variance: differenza tra costo pianificato ed effettivo	≥ 0
<u>MPD1</u> (Gulpease)	Indice di <u>Gulpease</u> per documenti in italiano	> 40 (accettabile) > 60 (buono)
<u>MPD2</u> (Ortografia)	Numero di errori ortografici rilevati	0
<u>MPS1</u> (Coverage)	<u>Code Coverage</u> (Unit Test)	$\geq 80\%$
<u>MPS2</u> (Req. Obb.)	Percentuale soddisfacimento requisiti obbligatori	100% al rilascio
<u>MPS3</u> (Comprens.)	<u>Complessità Ciclomatica</u> media delle funzioni	≤ 15

Table 4: Tabella delle metriche di progetto

5,1 Strategie di Verifica

- **Analisi Statica:** Review manuale del codice e uso di linter (ESLint per JS/TS, Pylint/Black per Python).
- **Analisi Dinamica:** Esecuzione della suite di test.
- **Test:**
 - **Unit Test:** Verifica di singole unità di codice.
 - **Integration Test:** Verifica delle interazioni tra moduli.
 - **System Test:** Verifica del sistema completo rispetto ai requisiti.

5,2 Software Quality Assurance SQA

La SQA monitora tutti i processi per garantire conformità agli standard (es. ISO 9001, ISO 25010).

Include:

- **Politiche:** Definizione di procedure per ogni fase (requisiti, design, testing).
- **Audit:** Revisioni periodiche per identificare non conformità.
- **Attività:** Review di documenti, controllo qualità codice, Gestione Rischi.

5,3 Validazione

La validazione conferma che il prodotto soddisfi le esigenze degli utenti.

- **Test di Accettazione:** Verifica finale con stakeholder per requisiti non funzionali (es. usabilità, prestazioni).
- **Feedback Utente:** Raccolta di input durante demo o beta testing.
- **Allineamento Obiettivi:** Verifica rispetto agli obiettivi di progetto (es. automazione analisi qualità repository).