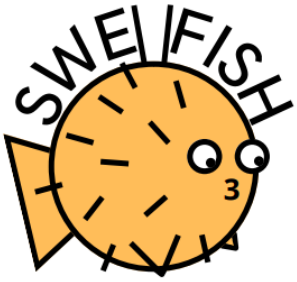

Norme di progetto

swellfish14@gmail.com

Informazioni	
Redattori	[Andrea Veronese][Claudio Giaretta]
Revisori	[Davide Porporati]
Responsabili	[Elena Marchioro]
Uso	[Interno]

Descrizione
File contenente tutte le best practices applicate al progetto

Contents

Versione	Data	Redattore	Verificatore	Descrizione
1.0.0	19/07/2023	Claudio Giaretta	Davide Porporati, Andrea Veronese	Revisione generale del documento e avanzamento di versione
0.1.1	12/07/2023	Claudio Giaretta	Andrea Veronese	Aggiunti strumenti di codifica utilizzati e corrette alcune sezioni del documento
0.1.0	04/05/2023	Claudio Giaretta	Jude Vensil Braceros	Verifica generale del documento
0.0.2	23/04/2023	Davide Porporati, Elena Marchioro, Francesco Naletto	Jude Vensil Braceros	Aggiunte sezioni verifica e scrum
0.0.1	19/04/2023	Andrea Veronese	Davide Porporati	Aggiunta convenzioni adottate per lo sviluppo
0.0.0	22/03/2023	Andrea Veronese	Davide Porporati	Versione preliminare con convenzioni di base

1 Introduzione

Lo scopo del documento è quello di fornire un insieme di regole, strumenti e standard di qualità necessari per creare un Way of Working condiviso ed efficacemente impiegabile da tutti i membri del gruppo. Questo documento conterrà tutte le convenzioni da utilizzare per lo sviluppo del progetto e verrà aggiornato in maniera incrementale durante tutte le fasi dello sviluppo, pertanto è da considerarsi come work-in-progress.

2 Processi Primari

2.1 Fornitura

2.1.1 Documenti

I documenti da redarre per la fornitura sono:

- Piano di progetto
- Piano di qualifica

2.1.1.1 Piano di progetto Il piano di progetto è il documento, redatto dagli amministratori e dal responsabile del progetto, che definisce l'approccio e le attività che saranno utilizzate per gestire e completare un progetto con successo. Il documento è diviso nelle seguenti parti:

- Introduzione
- Analisi dei rischi
- Pianificazione generale
- Metodo di lavoro
- Sprint Scrum

2.1.1.2 Piano di qualifica Il piano di qualifica è il documento, redatto dal verificatore, che definisce le attività e le risorse necessarie per garantire che un prodotto, un servizio o un processo soddisfi i requisiti di qualità specificati. Il piano di qualifica descrive in dettaglio le attività che devono essere svolte, le responsabilità dei membri del team e le risorse necessarie per raggiungere gli obiettivi di qualità. Il documento è formato dalle seguenti parti:

- Qualità dei processi
- Qualità del prodotto
- Specifica dei test
- Resoconto attività di verifica

2.1.2 Repository GitHub

Data la necessità di effettuare un lavoro sostanzioso e condiviso, il team ha deciso di utilizzare una repository pubblica su GitHub per rendere agevole la collaborazione e la visione delle modifiche apportate alle varie parti del progetto. E' stata quindi creata un'organizzazione chiamata SWEllFish e al suo interno sono state create delle specifiche repository:

- SWELLFish: questa repository conterrà tutto il codice necessario per il progetto
- Documentazione: questa repository contiene tutta la documentazione di interesse per il committente e il proponente.
- docs: repository che contiene i file sorgenti neccessari per la documentazione

2.1.3 Contatti con il committente/proponente

Per semplificare la gestione dei contatti e delle eventuali richieste con l'azienda committente e con i Professori Vardenega e Cardin, è stata creata una mail, swellfish14@gmail.com, utilizzabile da tutti i componenti del gruppo. L'uso di questa casella è stato regolamentato con un insieme di norme condivise dagli sviluppatori, pertanto un messaggio nasce dalla necessità di uno o più

componenti del gruppo di avere delucidazioni da parte del committente o per sottoporre una modifica in attesa di accettazione. Il canale telegram verrà usato per chiedere informazioni in modo informale con il proponente oppure per richiedere un confronto veloce. I colloqui vengono richiesti quando le domande da porre necessitano di una discussione diretta con il proponente. Questi ultimi vengono richiesti via mail oppure via telegram, tramite l'apposito gruppo creato con il responsabile incaricato di Imola Informatica.

2.2 Sviluppo

2.2.1 Attività

La parte di sviluppo è suddivisa nelle seguenti attività:

- Analisi dei requisiti
- Progettazione
- Codifica

2.2.1.1 Analisi dei requisiti Durante l'analisi dei requisiti ci si pone l'obiettivo d'individuare i requisiti impliciti ed espliciti, facendo un attento studio del capitolato, e di definire i casi d'uso del prodotto stesso.

Casi d'uso I casi d'uso descrivono le interazioni tra un sistema e gli attori che interagiscono con esso.Servono a identificare, documentare e comprendere i requisiti funzionali di un sistema, fornendo una rappresentazione chiara e concisa delle azioni che devono essere eseguite per raggiungere uno specifico obiettivo. Per ogni caso d'uso si specificano:

- Attori primari
- Attori secondari
- Precondizioni
- Postcondizioni
- Estensioni

Ogni caso d'uso verrà rappresentato da un diagramma UML

UML I diagrammi UML (Unified Modeling Language) verranno realizzati usando la versione 2 del linguaggio.

Requisiti I requisiti vengono tracciati attraverso una tabella suddivisa in questo modo:

Requisito	Descrizione	Classificazione	Fonti

- **Requisito** :Ad ogni requisito viene associato un codice identificativo.Ogni requisito ha alla fine del codice un numero puntato;
- **Descrizione** :breve decrizione del requisito;
- **Classificazione** : priorità attribuita al requisito, può essere obbligatorio, opzionale o desiderabile;
- **Fonte** : L'origine del requisito.

2.2.1.2 Progettazione Durante fase di progettazione vengono definite le funzionalità, la struttura e i comportamenti del prodotto, appoggiandosi all'analisi dei requisiti fatta preventivamente.

Inizialmente tutto ciò si tradurrà in un proof of Concept, ossia una prima demo per dimostrare la fattibilità del prodotto, e in seguito approfondita e descritta nel documento tecnico allegato alla Product Baseline.

La progettazione è divisa in due fasi:

Requirements & Technology Baseline I documenti neccessari in questa fase sono:

- Piano di progetto
- Piano di qualifica
- Norme di progetto
- Verbali interni ed esterni

Product Baseline I documenti necessari in questa fase sono:

- Manuale utente
- Allegato tecnico
- Verbali di periodo

2.2.1.3 Codifica La codifica ha lo scopo di concretizzare il prodotto tramite la programmazione

Stile della codifica

- **Variabili**: Le variabili devono seguire la notazione camelCase, iniziando quindi con la lettera minuscola e se composte da più parole, ogni parola successiva alla prima dovrà iniziare con la maiuscola
- **Metodi**: I metodi, devono seguire la stessa nomenclatura (camelCase) delle variabili
- **Indentazione**: Tutto il codice deve essere indentato seguendo le regole di indentazione del linguaggio di programmazione scelto
- **Commenti**: Nel caso il codice venga ritenuto di difficile comprensione il programmatore può inserire un commento per esplicitarlo. Tutti i commenti vanno inseriti la riga sopra del codice da commentare, in modo da rendere subito visibile il commento e renderlo di facile lettura
- **Blocchi condizionali**: tutti i blocchi condizionali devono essere racchiusi all'interno delle parentesi grafe, anche quando non sono necessarie, questo per rendere esplicito l'inizio e la fine di un blocco
- **Classi**: tutte le classi devono avere la lettera iniziale maiuscola, se composte da più parole ogni parola successiva dovrà anch'essa avere la prima lettera maiuscola
- **File**: Dovrà avere la lettera maiuscola ed un nome significativo rispetto al suo contenuto

Strumenti per la codifica

- **React**:Un framework per la realizzazione applicazioni Web

`https://react.dev/`

- **Node.js**: Un ambiente di runtime JavaScript open-source che consente l'esecuzione di codice JavaScript al di fuori di un browser.

`https://nodejs.org/en`

- **Express**: Un framework web per Node.js che semplifica la creazione di applicazioni web

`https://expressjs.com/`

- **Bulma**: Un framework CSS open-source che semplifica la creazione di layout responsive.

`https://bulma.io/`

2.2.2 Workflow GitHub

Tutti i documenti presenti nella repository docs, attraversano il seguente workflow:

- Creazione del branch remoto per il rispettivo documento
- Lavoro in locale con successivo push sul branch remoto corrispondente
- Viene verificato che il documento prodotto si attenga alle linee guida presenti in questo documento
- Viene verificata la qualità del contenuto
- Se i precedenti punti sono soddisfatti viene fatto il merge del branch in dev.

3 Processi di supporto

3.1 Documentazione

3.1.1 Ciclo di vita del documento

Ogni documento che verrà redatto attraverserà diversi processi, ognuno dei quali vedrà impegnati uno o più componenti del gruppo. Di seguito diamo una visione d'insieme delle fasi che ogni documento attraverserà:

- Pianificazione: il file viene ideato e discusso con i membri del team sulla base delle necessità che deve soddisfare
- Stesura: il testo e il contenuto del documento viene realizzato dal redattore
- Revisione: il revisore incaricato si occupa di rileggerlo e correggere eventuali errori e/o imprecisioni, controllando che siano state rispettate le convenzioni stabilite in questo documento.
- Approvazione: il responsabile designato controlla che il documento sia corretto e coerente.

3.1.2 Strumenti utilizzati

Di seguito gli strumenti utilizzati dal gruppo per la stesura della documentazione:

- **Latex**: linguaggio di Markup per la creazioni di testi. E' stato scelto in quanto permette la stesura dei documenti con una struttura formale e standardizzata, oltre alla moltitudine di strumenti che mette a disposizione il linguaggio che facilitano la stesura della documentazione

  `https://www.latex-project.org/`

- **Visual Studio Code e Latex workshop**: Editor popolare per la scrittura del codice in vari linguaggi di programmazione, utilizzato insieme al framework Latex WorkShop che permette di operare anche su codice latex. E' stato preferito a Overleaf in quanto permette di lavorare in locale e facilita l'interazione con sistemi di versionamento remoto come GitHub

  `https://code.visualstudio.com/docs`

3.1.3 Template

Il team ha deciso di utilizzare dei template scritti in Latex per facilitare il lavoro ed evitare codice ridondante tra i vari file della documentazione. Di seguito sono indicati i template utilizzati dal gruppo:

- **first_page**: template presente in tutti i documenti per la creazione della prima pagina
- **styles**: template presente in tutti i documenti che contiene dettagli di stile comuni a tutti i file della documentazione
- **tabella_versioni**: template per facilitare la creazione della tabella delle versioni nei documenti in cui è previsto versionamento.

Tutti i template si possono trovare all'interno della repository nella cartella `https://github.com/SWEllfish14/docs/tree/dev/templates`

3.1.4 Struttura del documento

Ogni documento che non sia un verbale ha una struttura ben specifica:

- Nome del gruppo
- Logo
- Tabella con le modifiche fatte e relativo numero di versionamento (se previsto)
- Tabella con ruoli dei componenti del gruppo e finalità d'uso del documento
- Indice
- Breve descrizione dello scopo del documento

3.1.4.1 Struttura e stile delle pagine

- Intestazione:
  - Lato sinistro: logo del gruppo
  - Lato destro: nome del documento
- Corpo del documento: strutturato in sezioni e sottosezioni, ognuna con il proprio titolo e sottotitolo.Il corpo del documento include anche qualsiasi supporto visivo, come tabelle, figure, grafici, diagrammi o illustrazioni, che aiutano a presentare le informazioni in modo più accessibile o coinvolgente.
- Piè di pagina: al centro è presente il numero di pagina.

3.1.4.2 Glossario Il documento Glossario elenca e spiega tutti i termini e le definizioni utilizzate nel contesto del progetto

3.1.4.3 Registro delle versioni Ogni file, fatta esclusione per i verbali, contiene una tabella che riporta le modifiche apportate al file nel corso del tempo. La tabella è così strutturata:

- **Versione**: versione attuale del documento (la nomenclatura segue le Regole per il versionamento)
- **Data**: data in cui viene rilasciata la corrispondente versione del documento
- **Redattore**: Indica chi ha apportato i cambiamenti al documento
- **Verificatore**: Indica chi si è occupato della verifica del documento
- **Descrizione**: Breve descrizione della modifica apportata nella corrispondente versione

Per facilitare l'inserimento della tabella all'interno dei vari file è stato creato un template apposito come spiegato nella sezione Template

3.1.5 Struttura di un verbale

La struttura di un verbale utilizza la struttura di base del template impiegato per la stesura di un documento, con le seguenti aggiunte

- Data
- Ora
- Durata
- Partecipanti
- Ordine del giorno
- Riassunto contenuti

3.1.5.1 Nomenclatura verbali Per la nomenclatura dei verbali invece segue la seguente struttura:

- verbale
- nome azienda/docente/attività svolta
- data

3.2 Gestione della configurazione

3.2.1 Versionamento

3.2.1.1 Sistemi software utilizzati Il VCS scelto dal gruppo per facilitare la gestione delle modifiche è Git, mediante l'utilizzo di GitHub. A questo scopo è stata creata un'organizzazione denominata "SwellFish", al cui interno sono state create le opportune repository in base alle necessità del progetto. Le varie repository presenti nell'organizzazione sono raggiungibili al seguente link : `https://github.com/orgs/SWEllfish14/repositories`.

3.2.1.2 Codice di versione Come prassi per la nomenclatura dei file il gruppo ha concordato la seguente struttura condivisa:

- nome file: tutto in minuscolo, con le parole separate dall'underscore
- numero versione

Per la verifica e la validazione dei file il gruppo ha deciso di utilizzare la seguente prassi:

- Creazione di un nuovo branch per un apposito documento
- Stesura del documento
- Caricamento sul branch appena creato
- Verifica della struttura, contenuti e rispetto delle convenzioni stabilite
- Merge sul branch dev
- Rilascio sul branch master

3.2.1.3 Regole per il versionamento E' stato concordato l'uso del Versionamento Semantico, affidandosi alle caratteristiche principali. La base del versionamento ha struttura "X.Y.Z" e le seguenti regole verranno usate per indicare una modifica minore o sostanziale del documento.

- Major Zero: E' la versione 0.X.Y, denota una versione preliminare del documento
- Patch Z: un aumento del valore della "Z" indica che sono state introdotte delle modifiche al testo da parte del redattore
- Minor Y: l'aumento della 'Y' indica una modifica testuale che è stata approvata dal verificatore
- Major X: l'incremento della 'X' si ha se è stata pubblicata dal responsabile una modifica non compatibile con le versioni precedenti del documento

3.3 Gestione della Qualità

La gestione di qualità è un processo che viene descritto nel piano di qualifica, in esso sono definite le metriche ritenute necessarie per valutare la qualità dei processi e dei prodotti. In questa sezione delle norme di progetto vengono descritte le singole metriche, mentre nel file "piano_di_qualifica" viene mostrato come esse vengono impiegate.

3.3.1 Indici utilizzati

- **BAC - Budget at Completion**: Indica il budget prefissato dal gruppo per la realizzazione del progetto. Il suo valore è fisso ed è di 11.900 €

3.3.2 Descrizione delle metriche

Qui vengono descritte le metriche utilizzate nel file "piano_di_qualifica".

3.3.2.1 Qualità di processo

- **MPC01: Actual Cost (AV)**: Rappresenta i costi affrontati dall'inizio del progetto fino ad un determinata momento

- **MPC02: Planned Value (PV)**: Rappresenta il costo previsto del prodotto in un determinato momento, se si è in pari con la progettazione
- **MPC03: Earned Value (EV)**: Rappresenta il valore effettivo del progetto fino ad un determinato momento
- **MPC04: Cost Variance (CV)**: Rappresenta la differenza tra EV e AC. Utile per capire quando siamo sopra o sotto il budget atteso
- **MPC05: Schedule Variance (SV)**: Rappresenta la differenza tra EV e PV. Utile per capire se siamo avanti o indietro rispetto a quanto pianificato in termini di denaro
- **MPC06: Cost Performance Index (CPI)**: Misura l'efficienza del lavoro svolto fino ad un determinato momento
- **MPC07: Estimate At Completion (EAC)**: Rappresenta la previsione del costo finale del progetto.
- **MPC08: Estimate To Complete (ETC)**: Rappresenta una previsione dei costi necessari per il completamento del progetto

Processi di Supporto

- **MPC09: Indice di Gulpease**: L'Indice Gulpease è un indice di leggibilità di un testo tarato sulla lingua italiana
- **MPC10: Errori ortografici**: Indica il numero di errori ortografici presente nella documentazione
- **MPC11: Code Coverage**: Indica la percentuale di codice testato

3.3.2.2 Qualità di prodotto

- **MPD01: Percentuale di difetti del prodotto**: Indica la capacità del prodotto di svolgere le sue funzioni, anche in presenza di errori, provando ad eliminarne la loro incidenza.
- **MPD02: Tempo medio di risposta**: Indica la capacità del prodotto di svolgere le sue funzioni utilizzando il minor numero di risorse possibile.

- **MPD03: Percentuale di copertura dei requisiti**: Indica la capacità del prodotto di svolgere tutte le funzioni previste in modo completo e corretto.
- **MPD04: Percentuale di comprensibilità del codice**: Indica la capacità del prodotto di essere modificato e mantenuto in modo efficiente.
- **MPD05: Percentuale di compatibilità del prodotto**: Indica la capacità del prodotto di essere utilizzato in diverse piattaforme e ambienti.
- **MPD06: Numero di errori compiuti dagli utenti durante l'utilizzo del prodotto**: Indica la capacità del prodotto di essere utilizzato in modo efficace, efficiente e soddisfacente dagli utenti finali.

3.4 Verifica

Durante questo processo, il compito dei verificatori è quello di effettuare l'analisi dei prodotti del team. Tale analisi si differenzia in due diverse tipologie:

- **Analisi statica**:processo di valutazione di un sistema o di un suo componente basato sulla sua forma, struttura, contenuto, documentazione. Questo tipo di analisi viene generalmente svolto tramite ispezioni e revisioni e può essere svolta sui documenti così come sul software o su parti di esso;
- **Analisi dinamica**: processo di valutazione di un sistema software o di un suo componente basato sull'osservazione del suo comportamento in esecuzione. Questo tipo di analisi viene generalmente chiamato testing e per motivi logici non può essere svolta sui documenti.

3.4.1 Verifica della documentazione

Nel caso la verifica venga fatta manualmente vengono usate le seguenti tecniche:

- **Walkthrough**: Un controllo esaustivo del documento (è un processo oneroso che può richiedere molto tempo e risorse)

- **Inspection**: Un controllo parziale legato solo ai punti critici del documento

Il team ha deciso di utilizzare una tecnica **Walkthrough** nella fase iniziale del documento e prima del rilascio di versioni "major". Dopo la formazione del documento ed un primo controllo esaustivo, si procede passando ad una tecnica di tipo **Inspection**

3.4.1.1 Verifica ortografica Per la verifica ortografica dei documenti si utilizzerà lo strumento di correzione automatica italiancorrector:
`https://www.italiancorrector.com/`

3.4.2 Verifica del codice

Il processo di verifica del codice avviene tramite due fasi:

- **Analisi statica**:tecnica per esaminare il codice sorgente senza eseguirlo, al fine di identificare errori e problemi di progettazione prima dell'esecuzione del software. Si utilizzano strumenti o revisioni manuali per individuare errori di sintassi, accessi alla memoria non validi, problemi di sicurezza e violazioni delle best practice di programmazione. L'obiettivo è migliorare la qualità del codice e ridurre i costi di risoluzione degli errori.

- **Analisi dinamica**:tecnica che consiste nell'eseguire il software e testarlo per identificare errori, problemi di prestazioni e di sicurezza. Si generano dati di esecuzione per individuare anomalie e miglioramenti. L'obiettivo è verificare il corretto funzionamento del software e migliorarne la qualità complessiva.

3.4.2.1 Test attività che consentono di verificare se un software funziona correttamente, identificando errori e problemi. Si utilizzano input specifici per osservare il comportamento e i risultati, al fine di garantire la qualità del prodotto e apportare eventuali miglioramenti.

3.4.2.2 Classificazione Test

3.5 Validazione

Processo per assicurarsi che il prodotto sviluppato rispetti i requisiti concordati con il proponente. Per fare ciò il prodotto verrà esaminato dai verificatori, verificando che esso sia conforme a quanto scritto nell'analisi dei requisiti. Sarà poi il responsabile successivamente che deciderà se accettare il prodotto o rifiutarlo chiedendo un ulteriore verifica.

3.6 Comandi utili di Git

I comandi più frequentemente utilizzati sono riassunti in seguito:

- Sincronizzazione con repository remota: *git pull*;
- Creazione di un nuovo branch: *git branch nome_branch*;
- Passaggio a un branch specificato: *git checkout nome_branch*;
- Aggiunta delle modifiche alla stage area: *git add file_da_aggiungere*;
- Creazione del commit con le modifiche: *git commit*;
- Push sul remote di un nuovo branch: *git push –set-upstream origin nome branch*;
- Push su un branch già esistente: *git push*;

4 Processi organizzativi

4.1 Gestione dei processi

4.1.1 Organizzazione dei File

Tutta la documentazione inerente al progetto sarà inclusa nella repository "Documentazione" all'interno dell'organizzazione chiamata "SwellFish". I documenti di interesse per il proponente/committente si trovano in Documentazione/esterni, mentre quelli necessari al team di sviluppo si trovano in Documentazione/interni. I verbali interni ed esterni sono rispettivamente contenuti in Documentazione/interni/verbali e Documentazione/esterni/verbali.

4.1.2 Scrum

Il team utilizzerà questo framework per gestire le varie fasi di sviluppo del progetto.

4.1.2.1 Organizzazione degli Sprint Ogni sprint avrà le seguenti caratteristiche:

- Durata: una settimana. Inizia il giovedì e termina il giovedì successivo con la riunione concordata
- Riunione: viene effettuata il giovedì al termine dello sprint e vengono discussi i seguenti punti:
  - Resoconto delle attività completate
  - Problemi riscontrati e soluzioni adottate
  - Analisi e resoconto delle ore e dei costi sostenuti
  - Pianificazione data inizio prossimo sprint e obiettivi da fissare

Gli obiettivi pianificati per lo sprint successivo verranno tradotti in "item" da aggiungere alla/e board presenti su Github e ad ogni item verranno assegnati dei membri del gruppo, coerentemente con la suddivisione dei ruoli per lo sprint in questione.

4.1.3 Rotazione dei ruoli

Vista l'adozione del framework scrum, il team ha deciso di far ruotare i ruoli dei componenti in corrispondenza della fine dello sprint, seguendo l'ordine alfabetico del nome dei componenti. Pertanto ogni membro del team avrà impegni e responsabilità diverse ogni settimana. Questo approccio permette di ottenere un maggiore coinvolgimento.

4.2 Gestione delle infrastrutture

4.2.1 Comunicazioni interne

Le comunicazioni interne avvengono principalmente tramite:

- **Discord**: Per quanto riguarda le riunioni settimanali e non. E' stato scelto in quanto conosciuto da tutti membri del gruppo, e la piattaforma web lo rende di facile accesso.
- **Telegram**: Per quanto riguarda comunicazioni minori o per comunicazioni di carattere logistico. Scelto in quanto conosciuto e usato da tutti i membri del gruppo.

4.2.2 Comunicazioni esterne

Le comunicazioni esterne avvengono principalmente tramite:

- **E-mail** utilizzando l'email del gruppo "swellfish14@gmail.com"
- **Google meet** tramite link inviato dal proponente
- **Telegram** tramite apposito gruppo messo a disposizione dal proponente

4.2.3 Gestione dei task

Per organizzare meglio il lavoro e avere una panoramica chiara dello svolgimento delle attività e dello stato di avanzamento, è stato concordato l'uso della **Project Board** fornita da GitHub. Le task riguardanti la documentazione e lo svolgimento del progetto vengo assegnate ai componenti dal responsabile del progetto che ha anche il compito chiuderle o modificarle in base alle esigenze. Per agevolare l'attività di documentazione è stata creata una board denominata "Documentazione", dove in concomitanza dello sprint settimanale vengono aggiunte nuove issue riguardanti la creazione o la modifica dei documenti da affrontare nel prossimo sprint. Per gestire le attività di sviluppo del prodotto verrà utilizzata un ulteriore board chiamata "SwellFish". Queste board sono raggiungibili tramite l'apposita sezione "Projects" all'interno dell'organizzazione SwellFish.

4.3 Miglioramento del processo

Durante le riunioni settimanali, si guarda in retrospettiva il lavoro fatto nel periodo corrispondente e si valutano eventuali correzioni laddove si sono riscontrati problemi o la qualità complessiva del lavoro svolto non è soddisfatta.