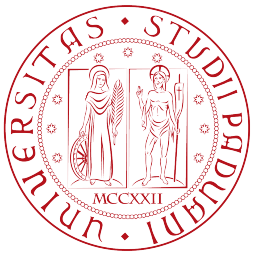
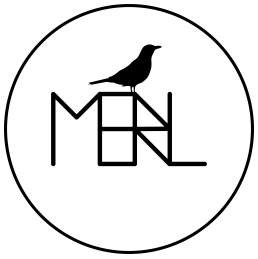


Università degli Studi di Padova
Laurea: Informatica
Corso: Ingegneria del Software
Anno Accademico: 2021/2022



Gruppo: MERL
Email: merlunipd@gmail.com

Verbale Riunione

20 Gennaio 2022

1 Informazioni generali

- **Tipo riunione**: interna
- **Luogo**: meeting Zoom
- **Data**: 20/01/2022
- **Ora inizio**: 12:15
- **Ora fine**: 13:00
- **Responsabile**: Pase Emanuele
- **Moderatore**: Zanellato Mattia
- **Scriba**: Zanellato Mattia
- **Verificatore**: Mamprin Marco
- **Partecipanti**:
  - Contin Riccardo
  - Mamprin Marco
  - Mazzucato Marco
  - Onelia Lorenzo
  - Pase Emanuele
  - Vukovic Marko
  - Zanellato Mattia
  - Cardin Riccardo (Docente)

2 Diario della riunione

- Discussione sull'analisi dei requisiti, in particolare sono stati trattati i seguenti argomenti:
  - Diagrammi UML, utilizzo e convenienza;
  - Utilizzo di *include* ed *extend*;
  - Gestione degli errori come casi d'uso separati;
  - Differenza tra *condition* ed *extension point*;
  - Scelta di utilizzare o meno un framework per la scrittura del codice;
  - Requisiti funzionali e di qualità.
- Discussione sulla gestione della milestone in corso visti i frequenti impegni legati agli esami;
- Risposta ad alcune nostre domande:

Domande	**Risposte**
Ha senso avere un UML unico o conviene dividerlo in più UML separati?	La scelta va fatta in base al vantaggio che porta, se avere un UML unico porta dei vantaggi può essere usato. Attenzione al fatto che un UML unico può essere più complicato da manutenere.
È possibile utilizzare un UML per spiegare in dettaglio uno UC avendo quindi come confine dell'UML il caso che si vuole analizzare?	Assolutamente si, l'importante è mantenere la giusta coerenza.
È accettabile avere un numero ridotto di UC?	Non esiste un numero giusto o sbagliato, l'importante è riuscire ad analizzare bene le funzionalità del prodotto richiesto.

Come va gestita la visualizzazione dell'errore?	Un errore unico e generale non dà alcun dettaglio sull'errore e di conseguenza non risulta utile perché non fornisce un anticipo sull'analisi. Il modo migliore è quello di avere uno UC che gestisce l'errore per ogni caso dato che pre e post condizioni dello UC risultano diversi ogni volta.
Può risultare conveniente utilizzare un framework per la stesura del codice, in particolare per mantenere il codice manutenibile, testabile e con un certo livello di qualità?	Non avendo molta esperienza nel campo un framework può risultare assolutamente conveniente per vari motivi: impone un modo di programmare che solitamente è quello corretto, evita di farci commettere errori che a lungo andare diventano irrisolvibili e permette di avere una struttura del codice più manutenibile.
Manuale utente e sviluppatore devono essere inseriti tra i requisiti funzionali o tra i requisiti di qualità?	La scelta corretta è quella di inserirli in entrambe le parti, ma in forme diverse: tra i requisiti funzionali ci sarà la possibilità di accedere ai manuali, tra i requisiti di qualità ci sarà la presenza dei manuali.

3 Todo

Durante la riunione sono emersi i seguenti task da svolgere.

Tutti	Continuazione delle issue presenti su *GitHub*
Tutti	Revisione dell'AdR e delle issue di *GitHub*