Sistemi Operativi II

Specifiche di progetto

Luca Geretti
luca.geretti@univr.it

Università di Verona
Dipartimento di Informatica

2025/2026

Panoramica

Introduzione

Scenario: Trattoria

Ovvero un piccolo ristorante familiare con pochi tavoli e basse pretese.

- Vi è un **personale** della trattoria.
- Vi sono multiple **famiglie** che rappresentano i clienti, in paziente coda per ottenere uno dei pochi tavoli e ricevere il servizio.

Obiettivo del progetto

- Lo scenario è una simulazione/gioco con:
  - **giocatori** (personale) in numero pari al numero di studenti del gruppo $+1$;
  - **tavoli** in numero pari al numero di membri del personale;
  - **avversari** (famiglie) in numero totale pari al numero di tavoli per 3.
- Gli studenti sviluppano un **client** C multi-threaded che controlla uno ad uno i membri del personale della trattoria.
- Il server pre-compilato, fornito agli studenti assieme all'header di interfaccia, modella il comportamento di tutto ciò che non è il personale.

Versioni uniche del progetto per ogni gruppo

- Ogni membro del personale ed ogni famiglia ha punteggi su un insieme di **parametri** rilevanti per lo scenario.
- Ogni gruppo di studenti ha una **versione unica ma deterministica** dello scenario.
- La versione dello scenario è:
  - generata in modo **pseudocasuale** usando i numeri di matricola degli studenti coinvolti;
  - si applica ai parametri e quindi
    1. a come il personale è in grado di svolgere il proprio compito, nonché
    2. a come le famiglie interagiscono col processo di ristorazione.

Processo di ristorazione lato famiglia

Abbiamo le seguenti fasi per una nuova famiglia:

1. Ingresso in trattoria e presa di un tavolo appena liberato;
2. Attesa di un aiutante che pulisca il tavolo da piatti e sporcizia precedenti;
3. Attesa di un cameriere per l'ordine;
4. Ordine del cibo;
5. Attesa della preparazione del cibo;
6. Attesa della consegna del cibo (perché dalla cucina viene messo in vista sul bancone);
7. Consumo del cibo;
8. Liberazione del tavolo ed attesa del turno in cassa;
9. Pagamento ed abbandono della trattoria.

Processo di ristorazione lato personale

Per ogni famiglia abbiamo le seguenti attività in ordine da dover compiere:

1. Pulire il tavolo;
2. Ricevere l'ordine;
3. Cucinare il cibo;
4. Consegnare il cibo;
5. Occuparsi del pagamento.

In più di tanto in tanto è necessario che qualcuno si occupi di lavare i piatti, disponibili in quantità limitata (sufficiente per il numero di tavoli presenti, nel caso di massima quantità di cibo ordinata).

Come giudicare la qualità del processo

Per poter giudicare la bontà del processo di ristorazione (e quindi il lavoro degli studenti) verranno introdotte due metriche:

- Profitto: inversamente proporzionale al tempo che la famiglia passa nella trattoria; maggiore il tempo, minore il tasso di sostituzione dei clienti al tavolo, minore il profitto;
- Reputazione: relativa alle recensioni che le famiglie svolgono a seguito della loro esperienza nella trattoria.

Dettagli di funzionamento

Principio di progettazione

Non tutti i dettagli sono conoscibili dagli studenti

Le informazioni accessibili al client sono perlopiù **parziali e qualitative**: questo obbliga nello sviluppo del client ad osservare manualmente il comportamento restituito dal server ed a pianificare di conseguenza la risposta.

Il personale - Abilità e tratti

Ogni membro del personale presenta 4 abilità e 4 tratti con valori casuali ma bilanciati (la somma delle abilità e la somma dei tratti sono costanti). Per ognuna è possibile sapere solo se è Bassa, Media o Alta.

Abilità

- Cameriere: prende gli ordini e consegna il cibo;
- Aiutante: pulisce/sparecchia i tavoli e lava i piatti;
- Cuoco: cucina;
- Cassiere: gestisce i pagamenti.

Tratti

- Pazienza: si applica nella fase di ordine;
- Socievolezza: si applica nella fase di pagamento;
- Professionalità: si applica nelle fasi di ordine e di cucina;
- Resistenza: si applica a qualunque ruolo svolto (dettagli in seguito).

Il personale - Come usare abilità e tratti

Abilità

Maggiore l'abilità, più veloce è il completamento del compito relativo al ruolo a cui il membro del personale è assegnato.

Tratti

La pazienza, socievolezza e professionalità influenzano la qualità della relazione con una famiglia e di conseguenza la **valutazione** che quest'ultima vi dà alla fine.
La resistenza si riferisce a quanto un membro del personale riesce efficacemente a svolgere lo **stesso ruolo per lungo tempo**. Una bassa resistenza si traduce in un decadimento più rapido della abilità effettiva in quel ruolo e quindi un tempo maggiore nel completare un compito.

Le famiglie - Pretese e caratteristiche

Ogni famiglia presenta 4 pretese e 4 caratteristiche, i primi con valori casuali ma bilanciati, i secondi casuali ma non bilanciati. Salvo alcune eccezioni, non è possibile conoscere nemmeno il livello di tali aspetti.

Pretese

- Tempo di attesa: sulla velocità di servizio in generale;
- Qualità del cibo: sul cibo servito;
- Qualità del servizio: sull'ordine e sulla consegna;
- Cortesia: sull'ordine e sul pagamento.

Caratteristiche

- Velocità ad ordinare: quanto la famiglia perde tempo nel fare l'ordine;
- Cibo ordinato: quanto cibo ordina;
- Velocità a consumare: quanto veloce consuma il cibo[1];
- Pulizia personale: quanto sporca consumando il cibo[1].

[1] È comunque anche proporzionale al cibo ordinato.

Le famiglie - Come conoscere pretese e caratteristiche

Pretese

All'uscita dalla trattoria la famiglia effettua una recensione fornendo un giudizio (positivo, negativo o neutrale) su ognuno dei quattro aspetti (tempo, cibo, servizio, cortesia). Ogni giudizio è **oggettivo** ed emerge dal confronto con la qualità che avremmo con valori medi di abilità e tratti del personale allocato. La famiglia poi produce una valutazione qualitativa finale in cui pesa questi giudizi secondo le proprie pretese **soggettive**.

Caratteristiche

- Le velocità ad ordinare/consumare possono essere osservate approssimativamente dal tempo simulato richiesto per tali fasi;
- Il cibo ordinato e la pulizia personale possono essere osservati (qualitativamente) dopo l'ordine e dopo il consumo, rispettivamente.

Interfacciamento

Interfacciamento a grandi linee

Simulazione di una istanza

La comunicazione programmata del client con il server avviene via IPC System V tramite memorie condivise ed una coda di messaggi. Vanno usati semafori IPC System V per sincronizzare l'accesso dove necessario.

Ciclo di vita delle istanze

La comunicazione programmata del client con il server inizia tramite un messaggio di saluto dal client, a cui segue dal server un messaggio di benvenuto con le informazioni generali necessarie. Il server invia in ordine una o più istanze e notifica il completamento di ognuna nonché la terminazione del client.

Simulazione - Memoria condivisa

Come coordinare i membri del personale

Vi è una memoria condivisa per la lavagna dove ogni membro può andare a scrivere chi si occupa di fare il cuoco, il lavapiatti ed il cassiere, e chi si occupa di pulire, ordinare o servire ad ogni specifico tavolo.

Come osservare lo stato di sala e cucina

Vi è una memoria condivisa per entrambe, da intendersi in sola lettura dal client. La prima indica qualitativamente la quantità di sporcizia e la quantità di cibo relativi al tavolo. La seconda indica qualitativamente la quantità di piatti puliti e di piatti sporchi presenti, il numero di ordini pendenti e se il cibo di un determinato tavolo è pronto per la consegna.

Come osservare la stato della cassa

Vi è una memoria condivisa da intendersi in sola lettura dal client. Essa espone il numero di pagamenti pendenti.

Simulazione - Coda di messaggi

Come osservare la stanchezza del personale

Ogni volta che un membro del personale incrementa il livello qualitativo di stanchezza, un messaggio viene mandato in una coda comune. L'mtype del messaggio è pari all'identificativo numerico del membro del personale + 1 (perché mtype deve essere >0). Questo permette ad ogni filo di estrarre solo i messaggi pertinenti. Inoltre alla fine di ogni istanza per ogni membro del personale viene stampata la stanchezza nei ruoli in cui è presente.

Simulazione - Lato server

Sala

Ogni tavolo è gestito da un processo separato del server: appena il tavolo è libero dalla famiglia precedente, vi associa la prossima famiglia ed avanza nelle fasi di (1) pulizia, (2) ordine cibo, (3) consegna cibo e (4) consumo cibo. Le prime tre fasi non avanzano se non c'è un membro del personale che se ne occupi. Quando una fase inizia, essa procede per la durata relativa fino a completamento (nessuna prelazione).

Cucina

Vi sono un processo server per preparazione cibo ed uno per lavaggio piatti. Il primo prepara una ad una le ordinazioni in ordine di arrivo, il secondo lava i piatti sporchi a blocchi del 25% dei piatti totali disponibili.

Cassa

Singolo processo server che gestisce le famiglie in ordine di arrivo.

Cosa deve fare il client

- Sviluppare **due strategie** specifiche, selezionabili
  - Strategia **profitto**: minimizzare la media del tempo totale speso da una famiglia nella trattoria;
  - Strategia **reputazione**: massimizzare la media delle valutazioni complessive delle famiglie.
- Implementare un **filo distinto per ogni membro del personale**
  - Ogni filo prende decisioni operative (cambio di ruolo) per la strategia interagendo con il server tramite una API fornita.

Per il resto l'implementazione è **completamente libera**: è sufficiente che l'interazione con il server porti (per entrambe le strategie) al **completamento** della gestione di tutte le famiglie.

Interfacciamento con il server - Modalità

Il server può operare in due modalità: normale e verifica.

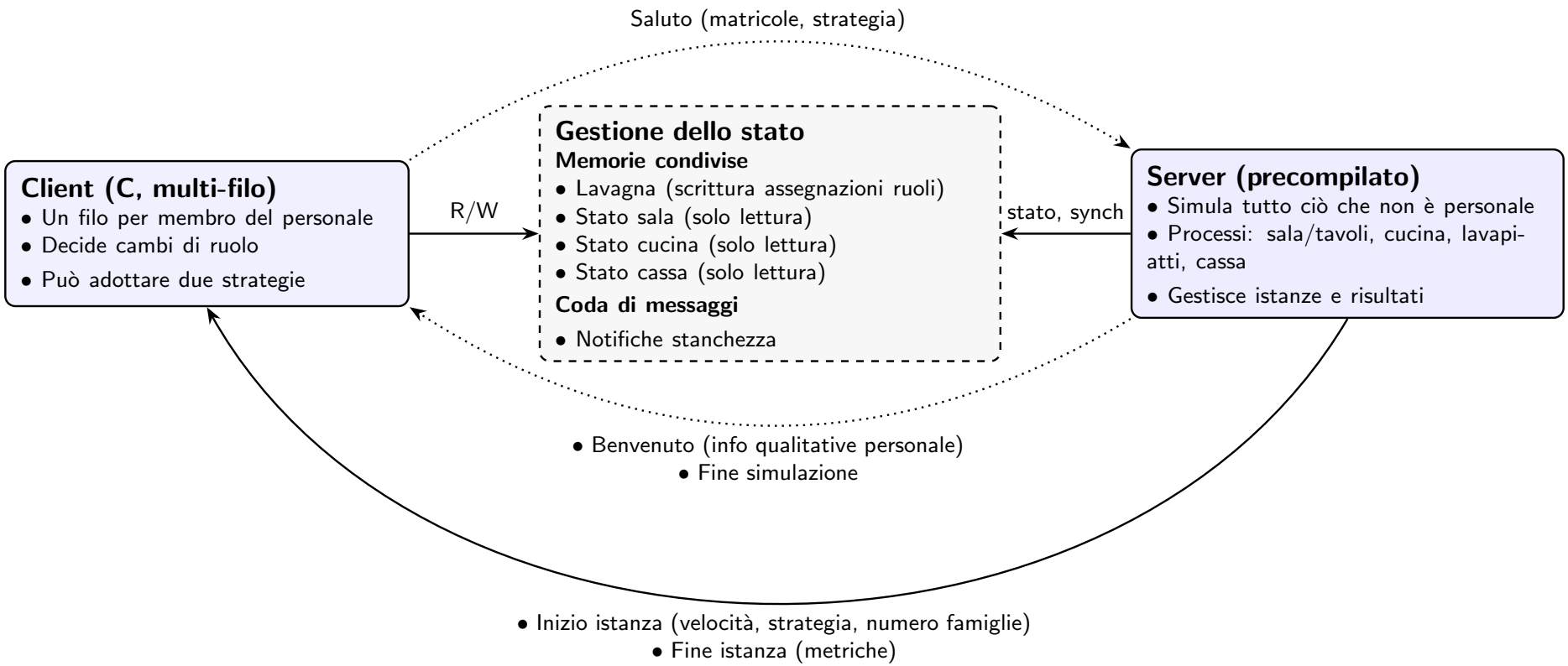
- **Normale**: viene eseguita una **unica istanza** di simulazione, con possibilità di terminazione automatica del client alla fine
  - Il client nel messaggio di saluto deve specificare la strategia usata;
  - Il server stampa gli eventi che descrivono l'andamento della simulazione della istanza.
  - Flag opzionale `--speed <N>` ed un valore intero (predefinito: 1) per velocizzare di un fattore N la simulazione (N.B.: può influenzare la sincronizzazione in base al comportamento del client)
- **Verifica** (flag `--verify`): vengono eseguite **multiple istanze** di simulazione per poter recuperare tutti i dati per verificare in modo programmatico il comportamento del client.
  - La strategia specificata dal client viene ignorata;
  - Flag opzionale aggiuntivo `--verbose` per stampare come in modalità normale; normalmente nascosto a causa della grande quantità di istanze eseguite.

Interfacciamento con il server - Protocollo

Le fasi sono le seguenti:

1. Il client invia un messaggio di **saluto** con l'insieme dei numeri di matricola degli studenti e l'eventuale strategia da imporre (se non siamo in modalità di verifica)
2. Il server risponde con un messaggio di **benvenuto** con l'insieme delle informazioni (qualitative) sui membri del personale
3. Il server manda, una alla volta, le **istanze da eseguire**
   - Ogni istanza specifica velocità, strategia usata e numero di famiglie
4. Quando una istanza viene completata, il server manda un messaggio di **completamento** con i risultati di media di tempo trascorso dalle famiglie e media di valutazioni delle famiglie
5. Quando tutte le istanze previste sono state completate, il server manda un messaggio di **fine** per la terminazione del client

Visione schematica interfacciamento



Ciclo di vita delle istanze

1. Il server invia una istanza (parametri: velocità, strategia usata, numero famiglie);
2. Il client coordina i fili via IPC (lavagna + osservazione stato);
3. Il server invia completamento istanza (medie: tempo e valutazioni);
4. Dopo tutte le istanze, il server invia messaggio di fine (terminazione client).

Verifica e valutazione

Valutazione del modulo in generale

Il progetto arriva fino ad un potenziale di 33/30 punti. Tuttavia il voto sulle domande dello scritto va a pesare i punti sopra il 18, quindi è questo extra (da 0 a 15) che conta veramente nel voto finale del corso. La formula è:

$$V = ceiling((S_T + 18 + S_L/30(P - 18))/2)$$

ossia la media fra il voto totale dello scritto S_T ed il voto di progetto, ottenuto pesando i punti del progetto oltre la sufficienza P (da 0 a 15 appunto) moltiplicati per il voto della parte dello scritto di laboratorio S_L diviso 30. Il tutto è arrotondato per eccesso.

Valutazione del progetto

In ogni caso il progetto **deve raggiungere la sufficienza**.

- Il voto di partenza è determinato da una valutazione (automatica) del software sviluppato, che può essere controllata dagli studenti in qualunque momento usando la modalità di verifica del server;
- A questo si sottrae la valutazione della relazione del lavoro svolto, che può confermare il voto del software o decrementarlo a discrezione del docente. Maggiori dettagli più avanti.

Nel seguito verranno indicati i requisiti software per la sufficienza, per un voto potenziale di 30 e per arrivare a 33.

Esecuzioni del client effettuate per la valutazione

Il server in modalità verifica effettua 100 esecuzioni per entrambe le strategie, a velocità 1000, per i seguenti casi:

1. Numero di famiglie nominale (3 volte il numero di tavoli);
2. Numero di famiglie doppio (6 volte il numero di tavoli).

La variante con il doppio di famiglie serve a verificare una eventuale implementazione non-dedicata ma è considerata opzionale e dunque è pensata principalmente per superare una votazione di 30.

Verifiche per la sufficienza

1. Completamento per **profitto** e per **reputazione**;
2. Strategia profitto: **tempo di completamento** minore della strategia reputazione in oltre il 50% dei casi;
3. Strategia reputazione: **punteggio** migliore della strategia profitto in oltre il 50% dei casi.

Perché una valutazione statistica?

Lo schedulatore può influenzare l'ordine di assegnazione dei ruoli e quindi impattare la strategia. Anche la velocizzazione estrema può avere un impatto in base a come è realizzata la strategia. Una buona strategia è perlopiù invariante alla velocità di simulazione (il cui valore è accessibile dal client) ed alla schedulazione.

Verifiche per punti extra fino a 12

1. (2.5 punti) Strategia profitto: **tempo di completamento minore** della strategia reputazione in oltre il 75% dei casi;
2. (2.5 punti) Strategia reputazione: **punteggio maggiore** della strategia profitto in oltre il 75% dei casi.
3. (2 punti) Strategia profitto: **tempo di completamento minore** della strategia reputazione in oltre il 90% dei casi;
4. (2 punti) Strategia reputazione: **punteggio maggiore** della strategia profitto in oltre il 90% dei casi.
5. (1.5 punti) Strategia profitto: **nessuna stanchezza alta** in nessun ruolo per nessun membro del personale in oltre il 90% dei casi;
6. (1.5 punti) Strategia reputazione: **nessuna stanchezza alta** in nessun ruolo per nessun membro del personale in oltre il 90% dei casi.

Verifiche per punti extra fino a 15

Svolte con un numero di famiglie doppio: poiché gli studenti non possono simulare questa situazione, le strategie devono essere sufficientemente generali e scalabili.

Le verifiche sono uguali a quelle per i punti extra fino a 12, ma con punti:

1. (0.75 punti) Profitto: **tempo minore** di reputazione $> 75\%$;
2. (0.75 punti) Reputazione: **punteggio maggiore** di profitto $> 75\%$;
3. (0.5 punti) Profitto: **tempo minore** di reputazione $> 90\%$;
4. (0.5 punti) Reputazione: **punteggio maggiore** di profitto $> 90\%$;
5. (0.25 punti) Profitto: **nessuna stanchezza alta** $> 90\%$;
6. (0.25 punti) Reputazione: **nessuna stanchezza alta** $> 90\%$.

Indicazioni pratiche per un buon esito

- Sfruttare le informazioni esposte dalla interfaccia, fare tentativi ed osservare il comportamento (trial-and-error) per eventualmente dedurre informazioni nascoste;
- Progettare le due strategie in modo chiaramente distinguibile:
  - Profitto: dà priorità al tempo di completamento dei compiti;
  - Reputazione: dà priorità alla qualità del servizio (accettando tempi peggiori).
- Rispettare la stanchezza del personale.
- Puntare alla robustezza: nonostante le 100 istanze, i risultati possono essere molto variabili se non si privilegia la comunicazione fra i membri del personale; si consideri che la verifica automatica del client verrà svolta **una volta sola** dal docente.

La relazione

La relazione non aggiunge punti alla valutazione del software, ma può sottrarne.

I requisiti per una relazione con penalità nulle sul voto sono:

1. Descrizione delle caratteristiche del personale ottenuto;
2. Osservazioni sulle caratteristiche delle famiglie ottenute;
3. Spiegazione delle scelte fatte nella progettazione della strategia, con specifico riferimento alle caratteristiche del personale;
4. Spiegazione dei criteri usati per soddisfare i vincoli di tempo di completamento minore, di reputazione maggiore ed eventualmente di assenza di stanchezza alta;
5. Descrizione delle problematiche incontrate e delle soluzioni adottate;
6. Spiegazione degli apporti specifici di ogni membro del gruppo di studenti nello sviluppo del codice e della relazione.

L'assenza completa di uno o più di questi aspetti può impattare molto negativamente sulla valutazione finale.

Materiale disponibile e consegna

Materiale messo a disposizione degli studenti

1. Progetto CMake vuoto con gli header di interfaccia
2. Eseguibile del server
3. Eseguibile di un client di esempio

Progetto CMake

L'interfaccia è fornita tramite due header:

- scenario.h: fornisce le enumerazioni utilizzate nello scenario;
- ipc.h: (include scenario.h) fornisce le strutture dati per la comunicazione con il server.

Per il resto il client è volutamente vuoto: finché viene rispettata l'interfaccia, qualunque realizzazione è valida.

Eseguibile del server

Correntemente fornito per le seguenti piattaforme:

- macOS arm64 (indicato con 'macOS' nel nome del file)[1];
- Linux x86-64 (indicato con 'Linux' nel nome del file).

Per quanto al momento non garantito, si prevede di verificare il supporto a WSL (Windows Subsystem for Linux) x86-64.

Il server stampa testo in inglese per maggiore accessibilità agli studenti stranieri.

All'esecuzione, il server mostra la versione dell'eseguibile: si prevedono aggiornamenti nel caso emergano migliorie o correzioni da implementare. Verrà mantenuta su Moodle la versione più recente per ogni piattaforma. L'interfaccia non verrà mai modificata.

[1] Per macOS può essere necessario dare il permesso di esecuzione degli eseguibili: dopo averli lanciati una volta, andare nelle Impostazioni di Sistema sotto Privacy e sicurezza, sul fondo della pagina abilitare gli eseguibili nella sezione Sicurezza.

Eseguibile di un client di esempio

- Fornito per le stesse piattaforme del server;
  - Si tratta di circa 1000 righe di codice tutto incluso;
- Dimostra il comportamento per un gruppo fisso di 3 matricole;
  - In verità le strategie sviluppate non sono specializzate e funzionano per 1-3 matricole con qualunque numero di famiglie;
- Con il flag `--strategy <profit|reputation>` si può scegliere la strategia;
- Stampa solo i messaggi di interscambio con il server;
- Restituisce successo tipicamente per tutti i controlli nella verifica.

Invio del progetto

Prima di inviare il progetto:

- Verificare la **compilazione** da zero del codice;
- Verificare l'**esecuzione** prevista in modalità verifica del server;
- Prima di creare l'archivio compresso, **rimuovere file estranei** (test interni, cartella di build, ecc.): nessun file fuori dal codice per l'eseguibile del client verrà considerato nella valutazione.

Il progetto completato va inviato via email a luca.geretti@univr.it da un solo membro del gruppo, mettendo in copia i restanti membri. Il progetto deve essere un archivio contenente la relazione e la base di codice del progetto CMake. Per quanto un invio di un allegato non sia proibito, si considera preferenziale un collegamento OneDrive.

- Il limite di validità del progetto è il **31 gennaio 2027**;
- La consegna può avvenire in qualunque momento.

Verifica e consegna in presenza del docente

Poiché l'esecuzione del progetto in modalità verifica viene svolta **una singola volta** da parte del docente, il gruppo di studenti se lo desidera può concordare un incontro con il docente in cui la verifica viene eseguita su un laptop scelto dal gruppo in presenza del docente. Il risultato, qualora sufficiente, viene annotato, dopodiché in presenza l'archivio del progetto viene inviato via email al docente.