Scuola universitaria professionale della Svizzera italiana
Dipartimento tecnologie innovative

SUPSI

Braccio meccanico

Studente/i	Relatore	Correlatore
Denis Beqiraj, Lorenzo Ronzani, Elia Salmina		

Corso di laurea	Modulo / Codice Progetto	Anno
Ingegneria Informatica		2021/2022

Committente	Data
Giancarlo Corti, Achille Peternier	17/01/2022

STUDENTSUPSI

Motivazione/Contesto

- Applicazione concetti teorici
- Curiosità
- Contesto accademico
- Applicazione di design-pattern
- Applicazione metodologia SCRUM
- Richiesta committente

Problema

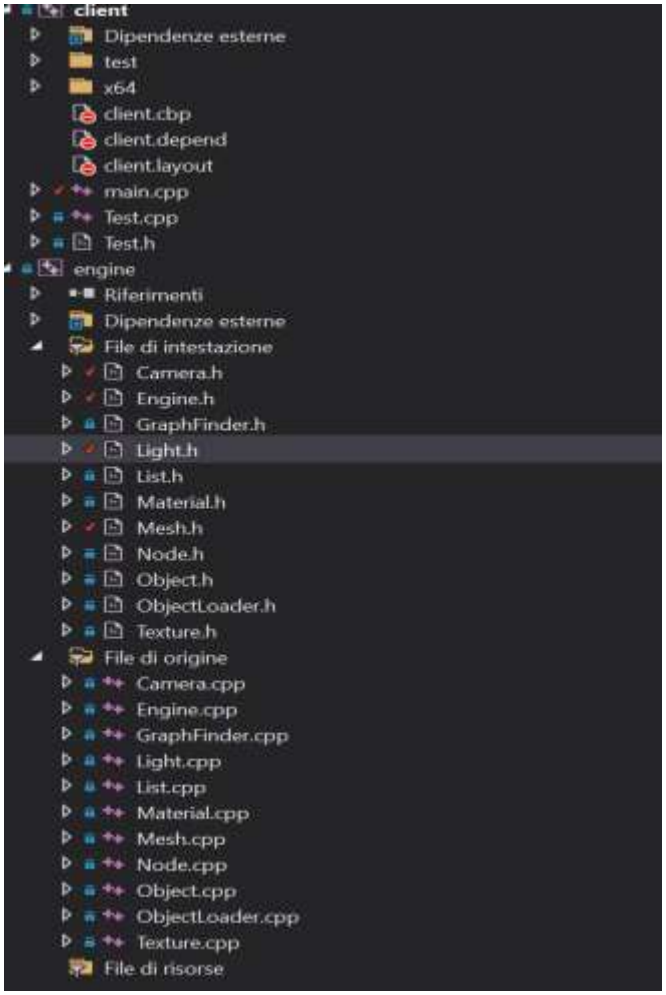
- Creazione engine generico
- Braccio meccanico a 3 segmenti
- Fonte di luce:
  - Fissa
  - Mobile
- Camera:
  - Fissa
  - Mobile

Problema

- Texture
- Ombre planari
- Multipiattaforma
- Palla prendibile e deformabile

Architettura

- Backend:
  - Engine
  - Creazione dll
- Frontend:
  - Client
  - Utilizza l’engine
  - Creazione gioco



Architettura

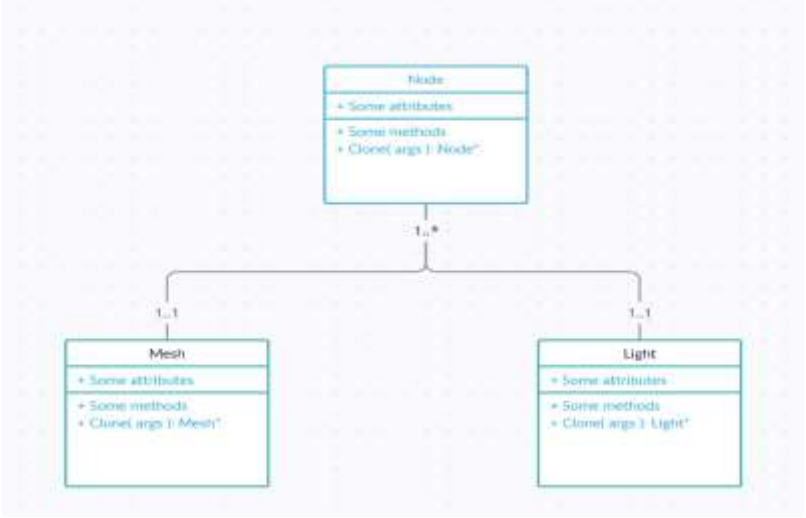
- Singleton

```
class LIB_API Engine {
public:
    struct Handler {
        std::function<void(int,int)> reshape;
        std::function<void(int,int,int)> special;
        std::function<void(unsigned char,int,int)> keyboard;
        std::function<void(int,int)> mouse;
        int width;
        int height;
    };
    static void init(Handler p_handler);
    static void clear();
    static void render(const List& list, std::shared_ptr<Camera> camera);
    static void swap();
    static std::shared_ptr<Node> load(std::string file);
    static void free();
    static void update();
    static void drawText(const std::string& text,float x,float y);
    Engine(Engine& other) = delete;
    void operator=(const Engine&) = delete;
    Engine() = delete;
    ~Engine() = delete;
};
```

Pattern

- Virtual copy pattern (virtual constructor)

```
virtual Node* clone();
Mesh* clone() override;
Light* clone() override;
```



Testing

- Unit testing list
- Unit testing load file ovo
- Unit testing recursive search

```
List list;
list.add(std::make_shared<Light>());
list.add(std::make_shared<Mesh>());
list.add(std::make_shared<Light>());
list.add(std::make_shared<Mesh>());
assert(dynamic_cast<Light*>(list[0].get()), true);
assert(dynamic_cast<Light*>(list[1].get()), true);
assert(dynamic_cast<Mesh*>(list[2].get()), true);
assert(dynamic_cast<Mesh*>(list[3].get()), true);
auto node = Engine::load("test/cube.OVO");
auto plane = node->getChildByName("Plane001");
assert(node, node != nullptr);
assert(plane, node != nullptr);
```

Risultati

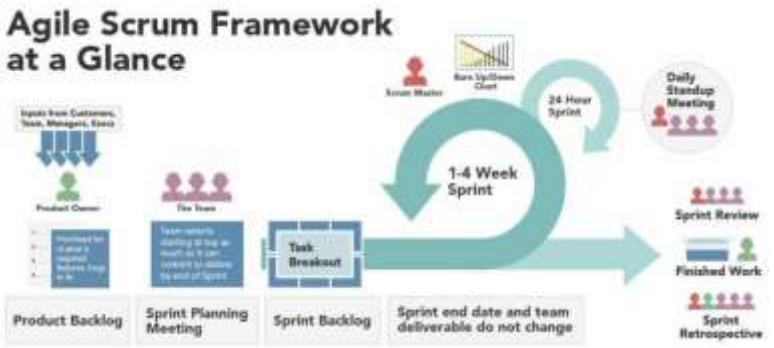
- Engine generico funzionante
- Simulazione del braccio meccanico
- 4 fonti di luce:
    - Omnidirezionale, in tutta la scena
    - Spot, illumina braccio
    - Spot, automatica che percorre la scena
    - Spot, manuale sul braccio

Risultati

- 3 camere:
  - Mobile, spostabile con il mouse
  - 2 Fisse, poste sulle pareti
- Scena texturizzata
- Ombre
- Multipiattaforma
- Palla prendibile e deformabile

Conclusioni

- Prodotto funzionante
- Collegamento tra teoria e pratica
- Prima esperienza di lavoro agile
- Applicazione di design pattern
- Version control system
- Multipiattaforma
- Organizzazione di più progetti



Demo