Design

Software Engineering

Theoretical Concepts

1. Software Design
    a. Definition.
    b. Principles.
    c. Technical Concepts.
    d. Artifacts and modeling.
    e. Software design aspects.

Software Design

- Software design process is a sequence of steps that enable the designer to describe all aspects of the software to be built.
- Software design has some principles that judge the design:
  - Design process should consider alternative approaches, judging each based on the requirements of the problem.

Software Design - Principles

- The design should be traceable to the analysis model.
- The design should not reinvent the wheel.
- The design should “minimize the intellectual distance” between the software and the problem as it exists in the real world.
- The design should exhibit uniformity and integration.
- The design should be structured to accommodate change.

Software Design – Principles (Cont.)

- The design should be structured to degrade gently, even when aberrant data, events, or operating conditions are encountered.
- The design should be assessed for quality as it is being created.
- The design should be reviewed to minimize conceptual (semantic) errors.

Software Design – Concepts

Technical Concepts:

- **Abstraction**: It is the process or result of generalization by reducing the information content of a concept .
- **Refinement**: It is the process of elaboration.
- **Modularity**: It is the software architecture that divides the system into components called modules.

Software Design – Concepts (Cont.)

- **Data Structure**: It is a representation of the logical relationship among individual elements of data.
- **Information Hiding**: Modules should be specified and designed so that information contained within a module is inaccessible to other modules that have no need for such information.

Software Design – Modeling

- Software modeling is the process of creating models that describe the software in terms of structure and procedures.
- Unified Modeling Language (UML)
  - Definition: It is a modeling language in the field of software engineering, which is designed to provide a standard way to visualize the design of a system.
  - It uses standard visual artifacts to describe these diagrams:
    - Use case diagram.
    - Component diagram.
    - Class diagram.
    - Activity diagram.
    - Sequence diagram.
    - And others...

Software Design – Modeling (Cont.)

- Using UML standard is important in software design process, as it simplifies:
  - The process of understanding the software even for non technical users.
  - Maintenance of the design as it uses a standard techniques known by most of software engineers.
- CASE tools are the set of tools and methods to a software system with the desired end result of high-quality, defect-free, and maintainable software products.
- Sample of software design CASE tools are UML studios (e.g. Visual Paradigm)

Software Design – Aspects

Software Design Aspects

- **Compatibility**: The software is able to operate with other products that are designed for interoperability with another product.
- **Extensibility**: New capabilities can be added to the software without major changes to the underlying architecture.
- **Fault-tolerance**: The software is resistant to and able to recover from component failure.
- **Maintainability**: A measure of how easily bug fixes or functional modifications can be accomplished. High maintainability can be the product of modularity and extensibility.

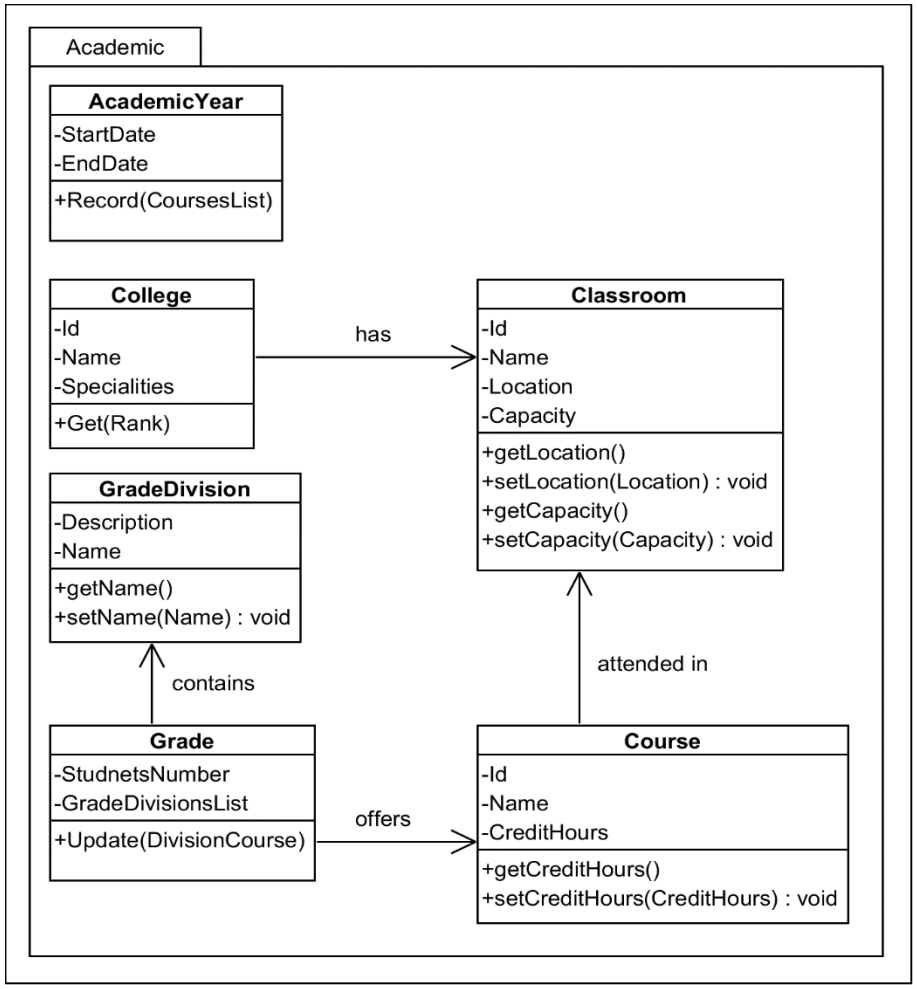
Software Design – Aspects (Cont.)

- **Modularity**: The resulting software comprises well defined, independent components. That leads to better maintainability.
- **Reliability**: The software is able to perform a required function under stated conditions for a specified period of time.
- **Reusability**: The software is able to add further features and modification with slight or no modification.
- **Robustness**: The software is able to operate under stress or tolerate unpredictable or invalid input.

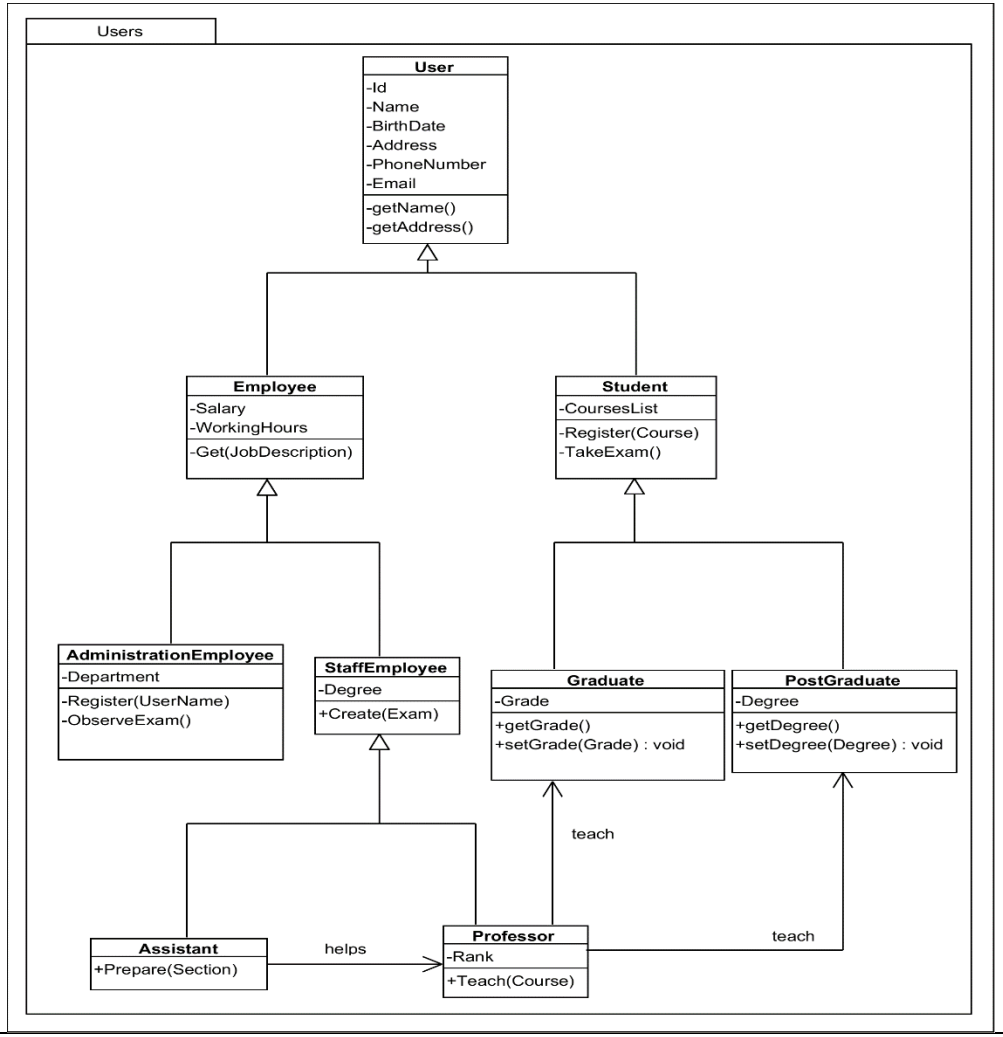
Software Design – Aspects (Cont.)

- **Security**: The software is able to withstand hostile acts and influences.
- **Usability**: The software user interface must be usable for its target user/audience.
- **Performance**: The software performs its tasks within a user-acceptable time. The software does not consume too much memory.
- **Portability**: The usability of the same software in different environments.
- **Scalability**: The software adapts well to increasing data or number of users.

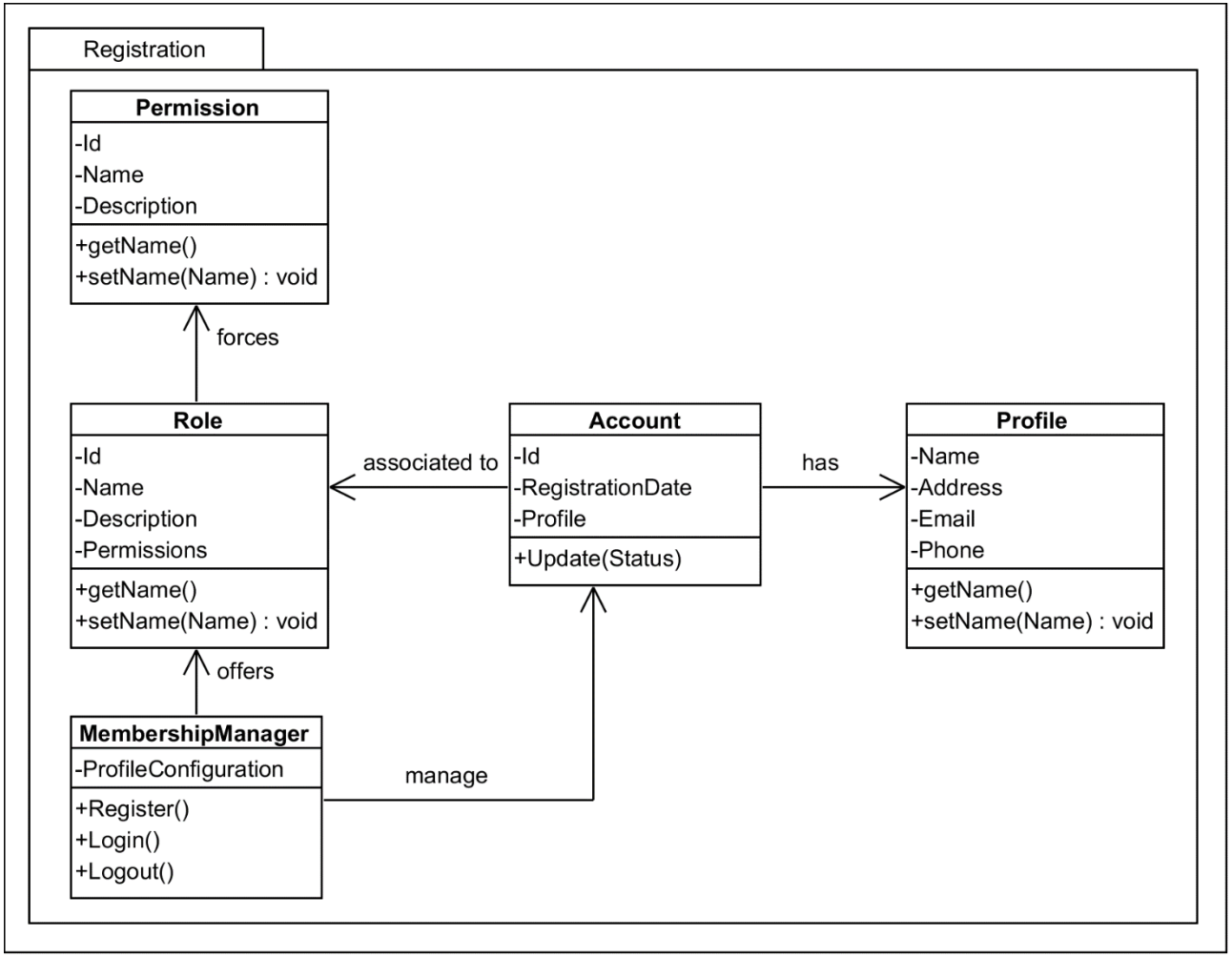
Examination Module Class Diagrams (Academic)



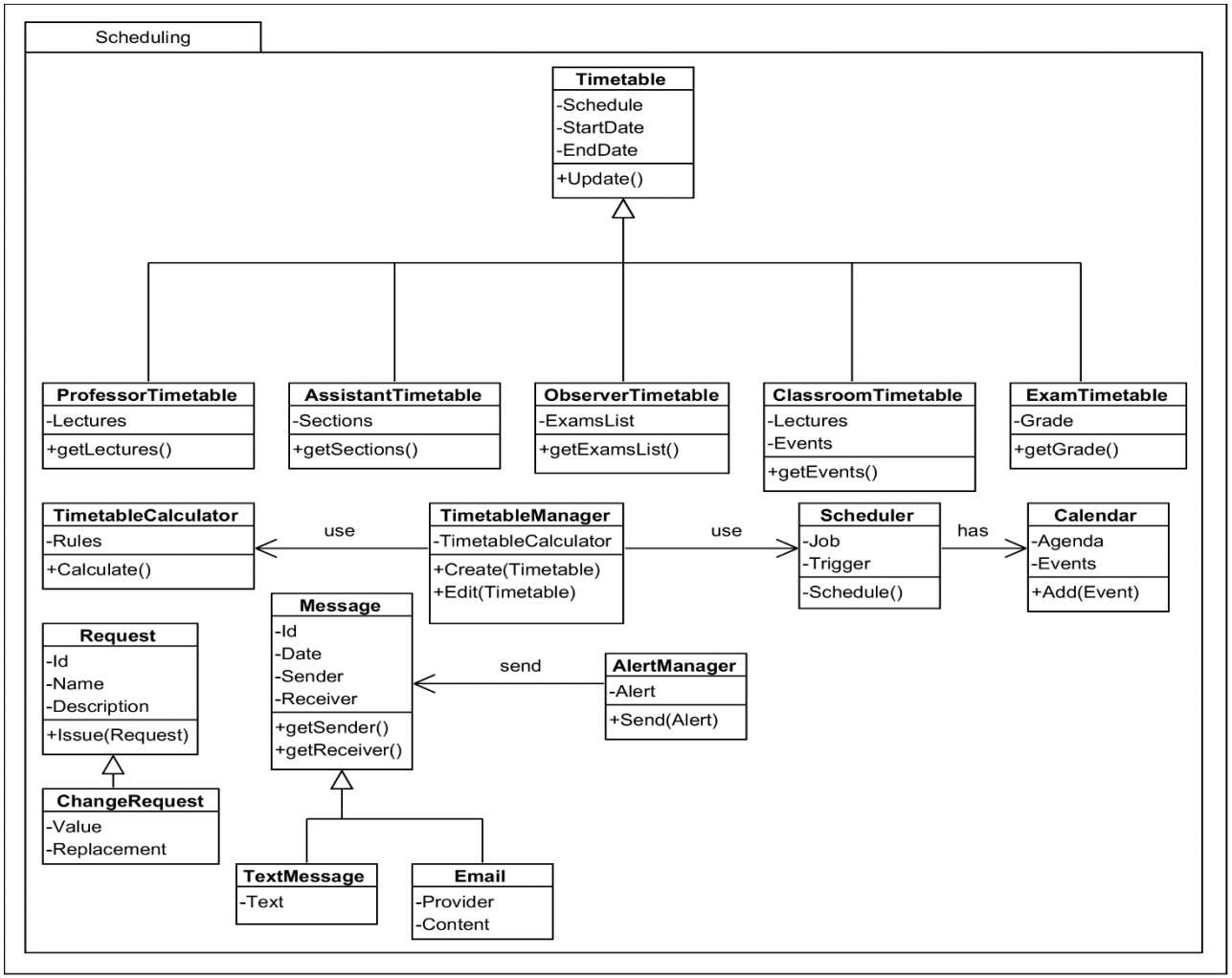
Examination Module Class Diagrams (Users)



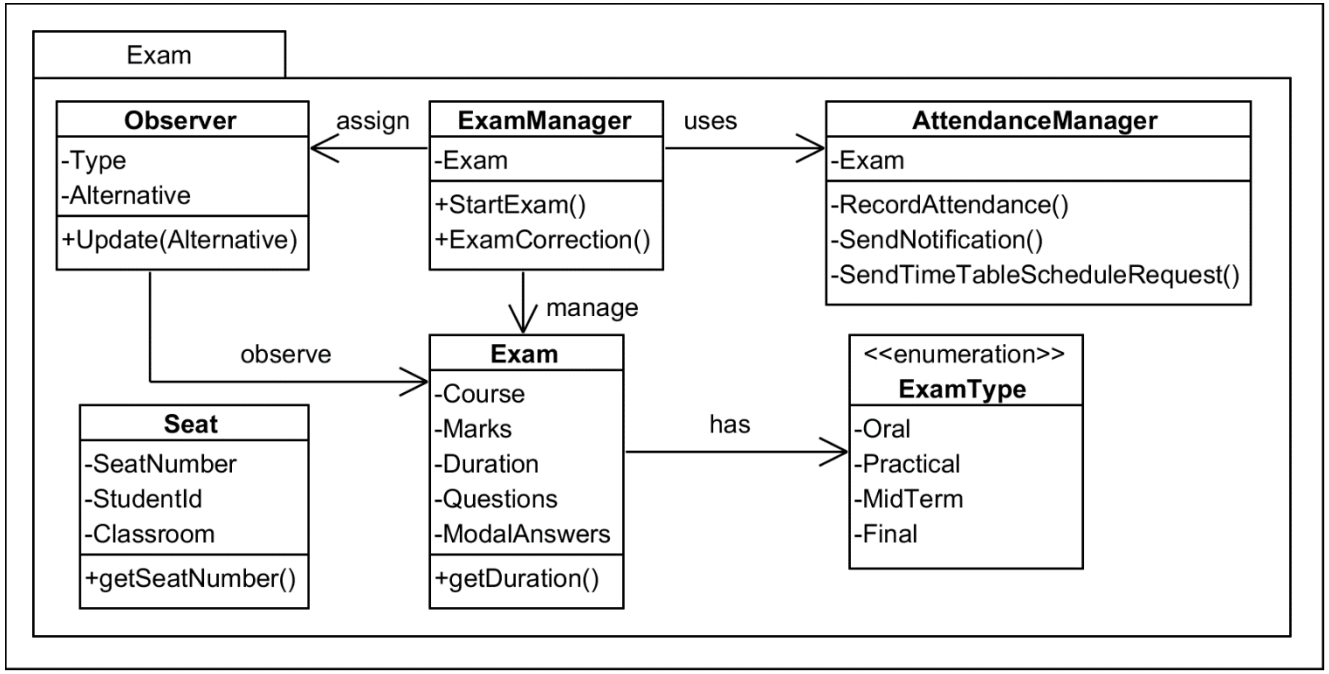
Examination Module Class Diagrams (Registration)



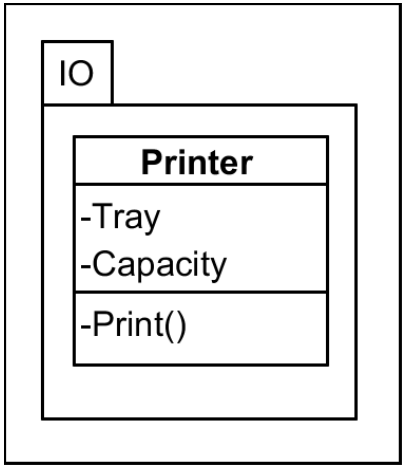
Examination Module Class Diagrams (Scheduling)

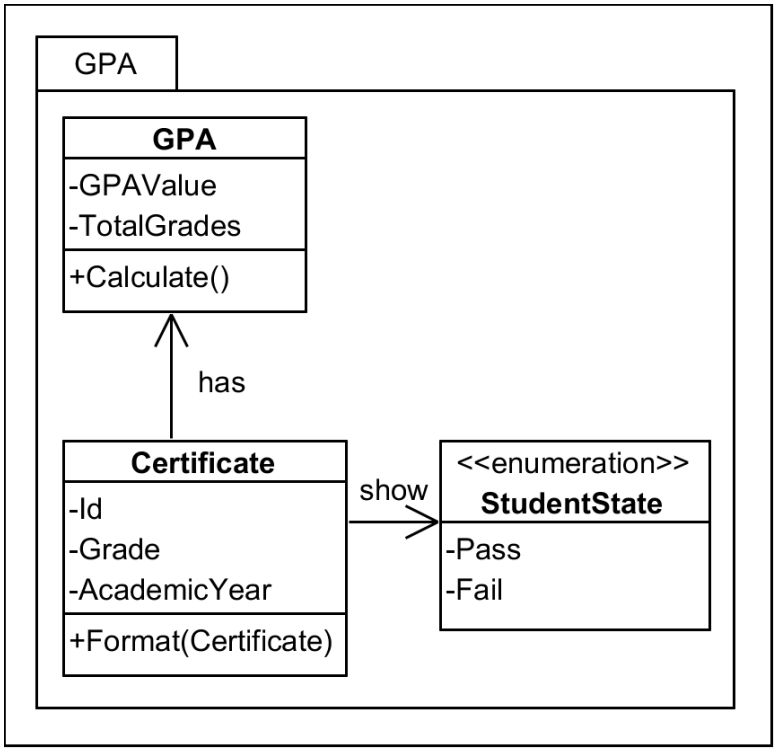


Examination Module Class Diagrams (Exam)

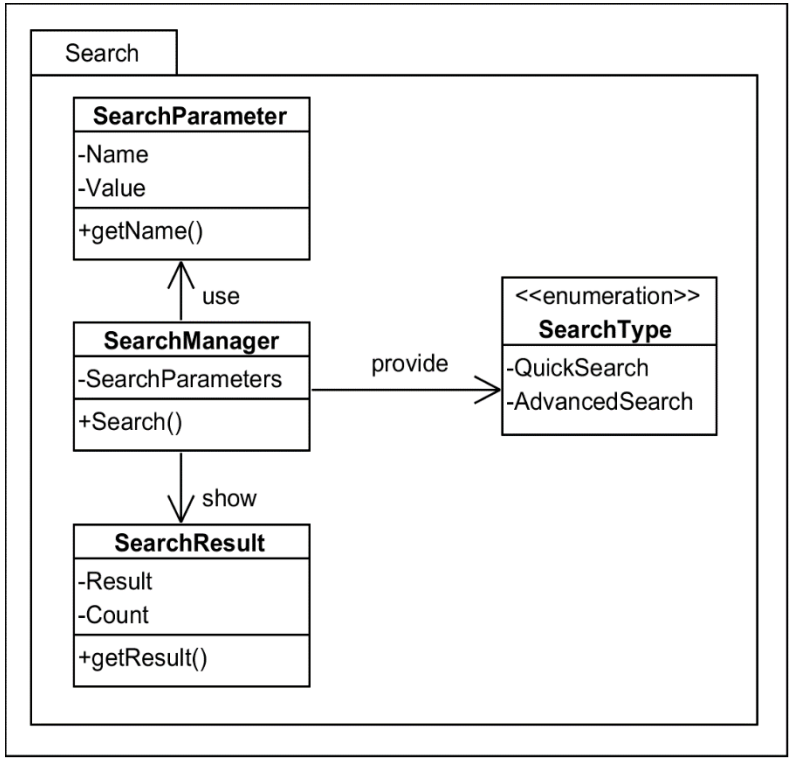


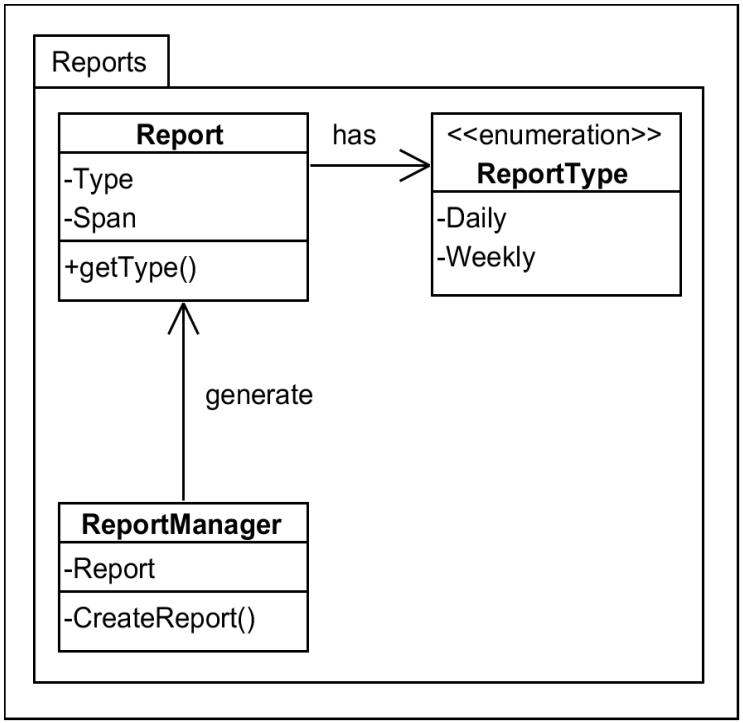
Examination Module Class Diagrams (IO & GPA)





Examination Module Class Diagrams (Search & Reports)





Thanks!