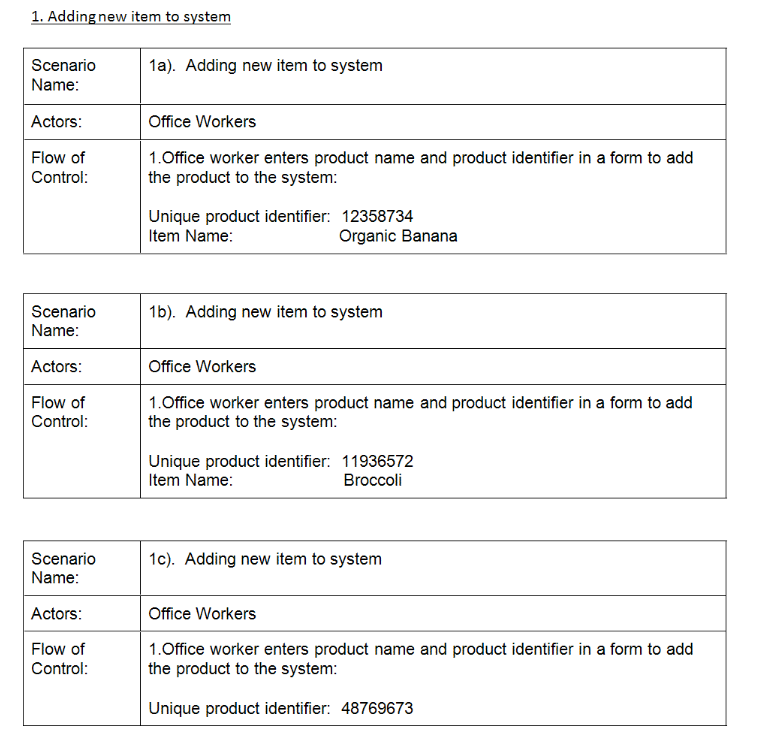
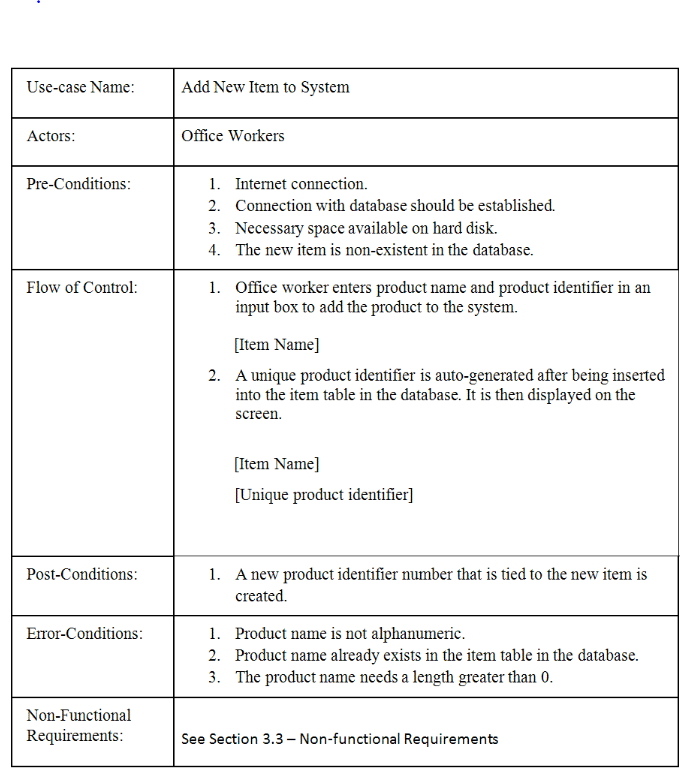
**CECS 491A Exam Study Guide**

\*\*\* Multiple Choice, Short Answers, Draw Sequence Diagrams, Draw UML Diagrams \*\*\*

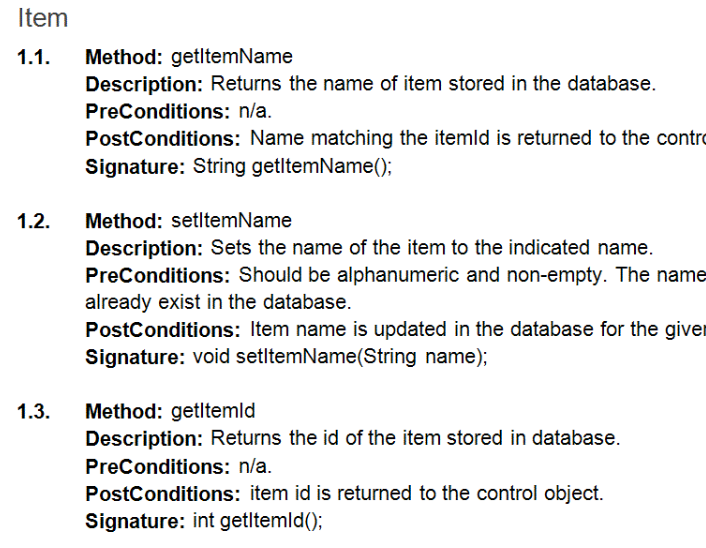
* Request for Proposal (RFP)
  + **A document that an organization posts to elicit bids from potential vendors for a desired IT solution.**
  + Business Requirements
    - Examples: users, inventory info, inventory management, reporting services
  + Software Requirements
    - Examples: digitalization of existing records, UI, validation, security, data in sync
  + Hardware Requirements
    - Examples: data storage, devices
* Scenarios
  + **Natural language formulation of a concrete usage of the system.**
  + No pronouns or general terms. Must consist of instantiated variables.
  + Example:



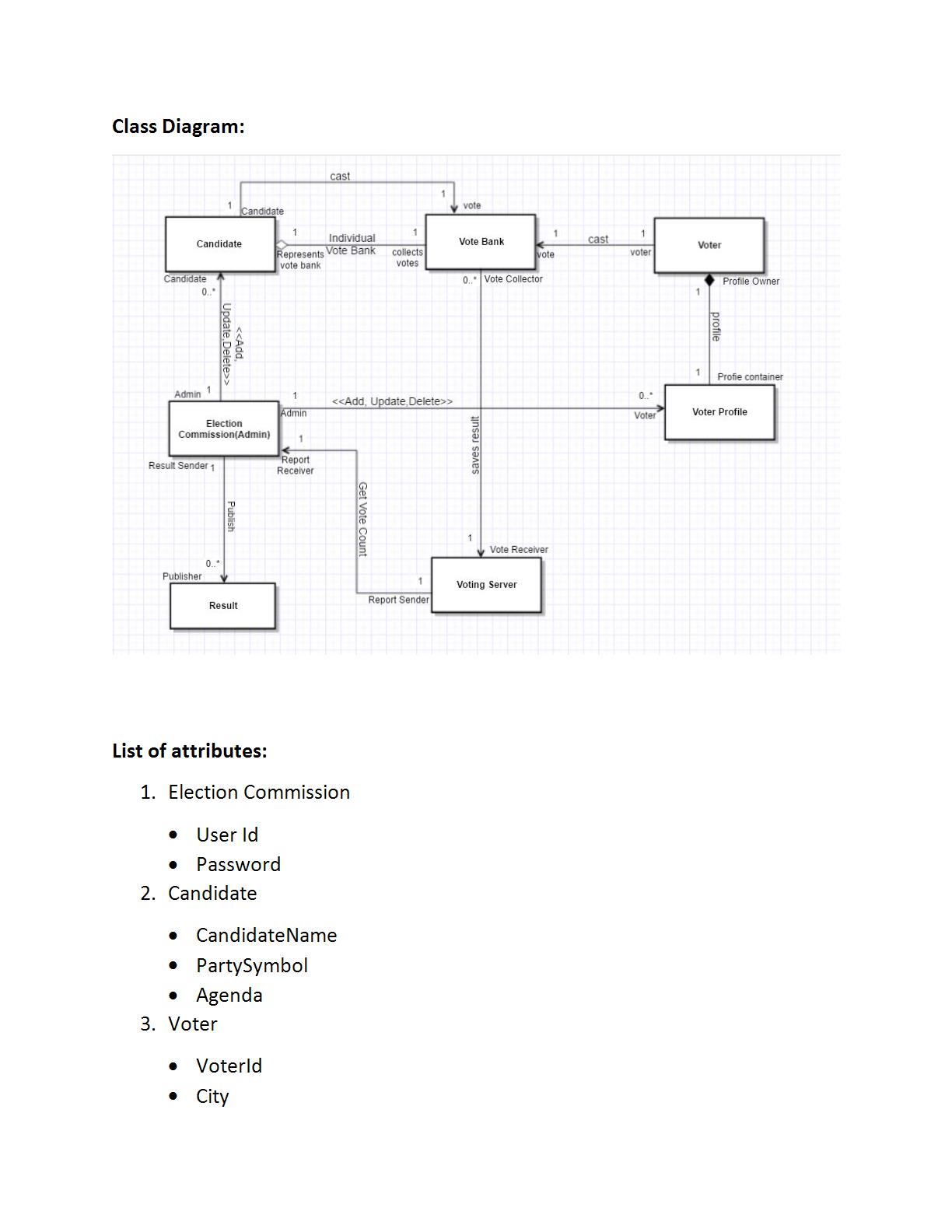
* + How do we find scenarios?
    - Record real life transactions and talk to the client.
    - You help the client formulate the requirements.
    - The client helps you to understand the requirements.
    - The requirements evolve while the scenarios are being developed.
  + What are they used for?
    - **To find abstractions.**
    - **To find patterns of requirements.**
    - **To create use-cases.**
* Use Cases
  + **Natural language formulation of the functions of the system.**
  + Like scenarios but replace the instantiated variables with general terms.
  + Example:



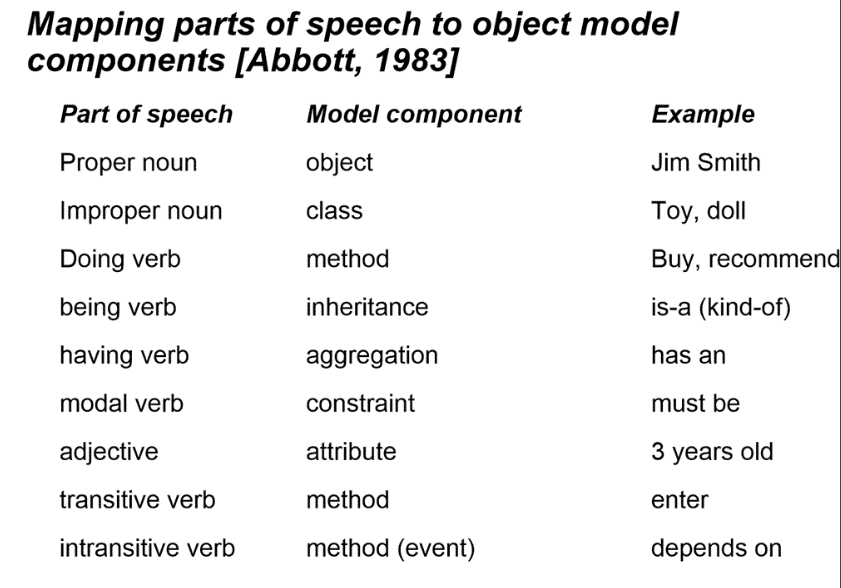
* + How do we find use cases?
    - Select a scenario.
    - Discuss the scope with the user.
    - Find out what the user does.
* Method Definitions
  + Name
  + Description
  + Pre-Conditions
  + Post-Conditions
  + Side Effects
  + Signature
    - Return Value
    - Name
    - Argument list
  + Example:

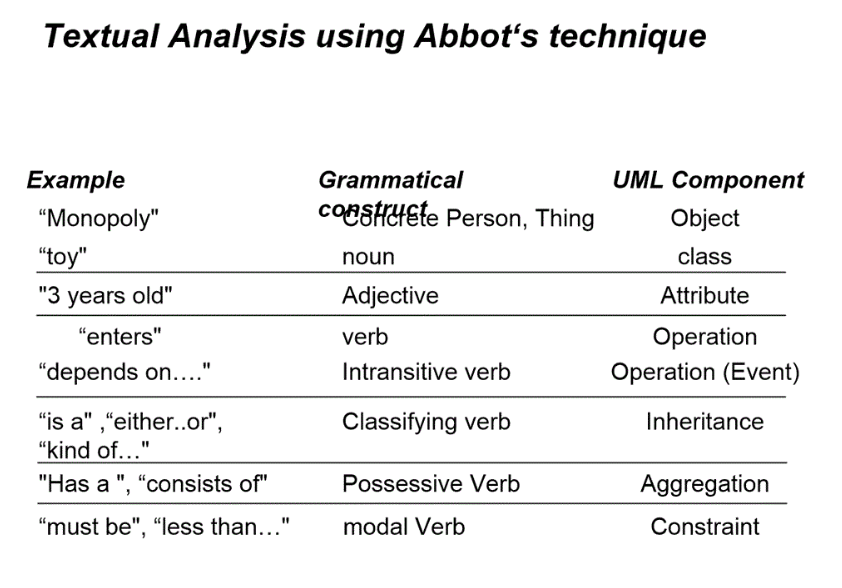


* Class Diagrams w/ UML
  + **Inheritance** symbol (IS-A)
    - Association between two independent objects.
    - Both objects have their own life cycle.
    - No ownership.
    - **Line w/ half diamond (not filled-in)**
  + **Aggregation** symbol (HAS-A)
    - Specialized form of association between two objects.
    - Both objects have their own life cycle.
    - Ownership exists. Child objects cannot belong to another parent object.
    - **Line w/ diamond (not filled-in)**
  + **Composition** symbol (HAS-A)
    - Specialized form of aggregation between two objects.
    - Child does not have its own life cycle. If parent is deleted, so is the child.
    - Ownership exists. Child objects cannot belong to another parent object.
    - **Line w/ diamond (filled in)**
  + Cardinality/Relationship/Roles
    - Cardinality
    - Relationship
    - Roles
  + Example:



* Nouns/Verbs/Adjectives/Adverbs





* Classes/Methods/Attributes/Non-Functional Requirements
  + Classes
    - How do we find classes?
      * Take the flow of events and find participating objects in use cases.
      * Apply design knowledge.
      * Do a syntactic analysis.
  + Methods
    - Functions of the class
  + Attributes
    - Variables of the class
  + Non-Functional Requirements
    - All remaining requirements which are not covered by the functional requirements.
* Functional vs. Non-Functional Requirements
  + “Is this requirement specific to this application?”
  + If *yes*, it is a functional requirement.
  + If *no*, we need to know the definition of a non-functional requirement relative to the application to be certain.
  + Functional Requirements – **specify something the system should do**.
    - Example: business rules, audit tracking, historical data, external interfaces
  + Non-Functional Requirements – **describe what the system should do**.
    - Example: reliability, security, performance, reliability, usability, environmental
* Sequence Diagrams
  + What is it?
    - **They are diagrams that describe dynamic behavior of a set of objects arranged in time sequence.**
    - Stair Diagram
  + **Entity Objects** –model “real word” objects and are almost always persistent.
    - Examples: Item, ItemLabel, Scanner, NetProfitReport, ExpirationReport
  + **Boundary Objects** – objects used by the system to interact w/ outside agents.
    - Examples: AddItemBoundary, CreateLabelBoundary, SellItemBoundary
  + **Control Objects** – contain the logic to direct the flow of control of a use case.
    - Typically have a 1-1 relationship w/ use case.
    - Examples: ScannerControl, InventoryControl, NetProfitReportControl
  + Examples:

