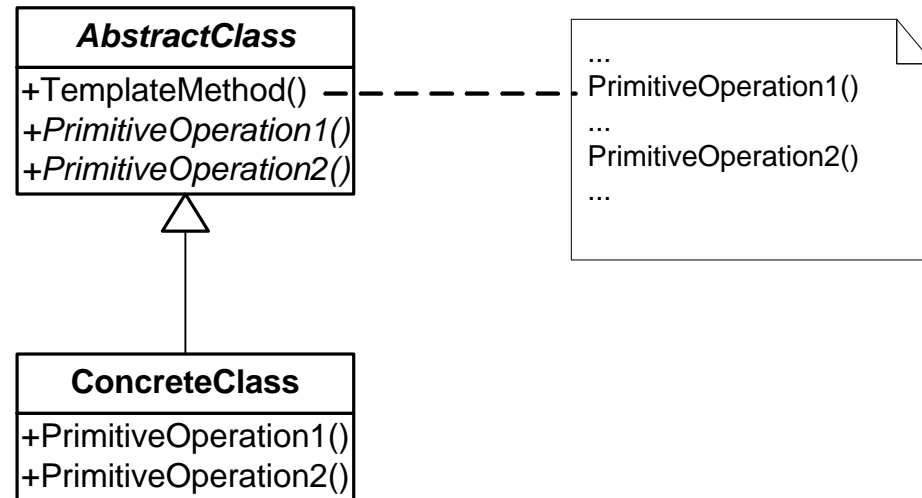
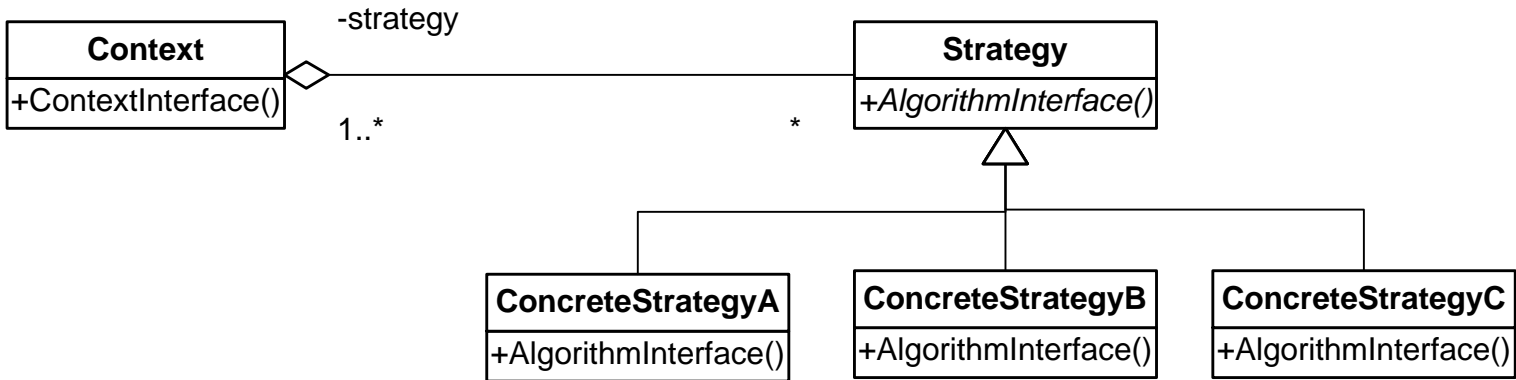


Review: Template Method Pattern



- Uses inheritance to vary part of an algorithm
- The superclass operation (i.e. Template Method) defines a sequence of steps required to perform some function
- The subclass fills in only the specific behaviors through the implementing methods (i.e. Primitive Operations)
- Implicitly using Factory Method (the Primitive Operations)
- Example: the standard interface for *HttpServlet* has
 - `service()`, `doGet()`, `doPost()`

Review: Strategy Pattern



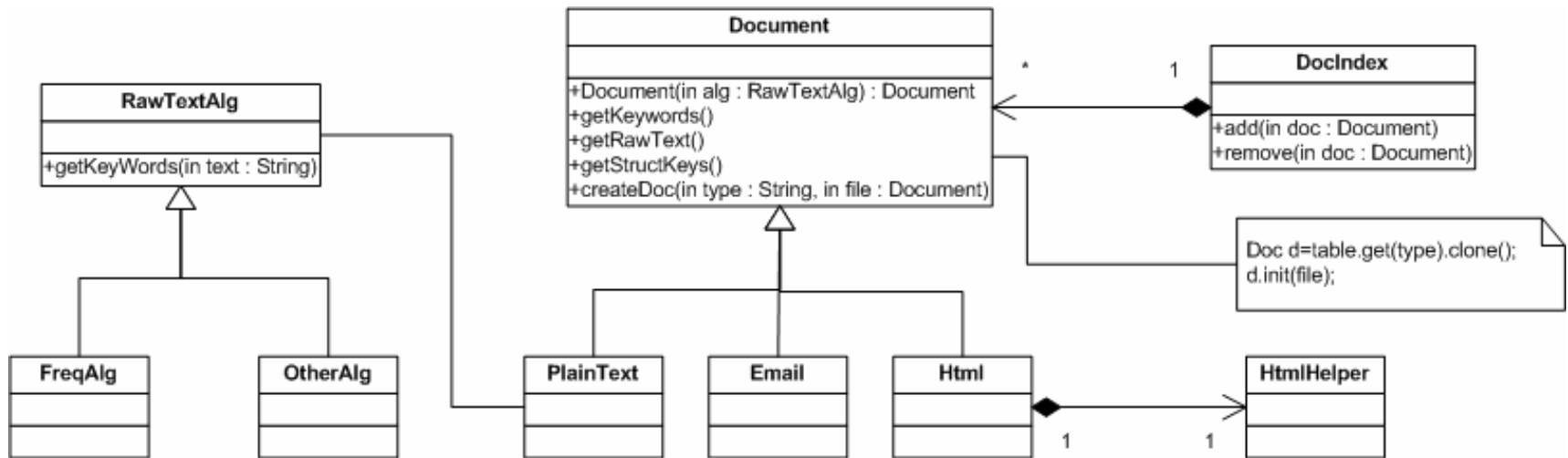
- Uses delegation to vary the entire algorithm
- Different algorithms may be appropriate at different times
- Client (i.e. Context or calling class) has a choice on which algorithm to use
- Notice that the entire algorithm is changed when a new concrete strategy is used in the Strategy pattern
- But in the Template Method, only individual primitive operations of the algorithm change when a new concrete class is used, the overall sequence of steps (defined in the template method) do not change

Recall Example: Document Retrieval System

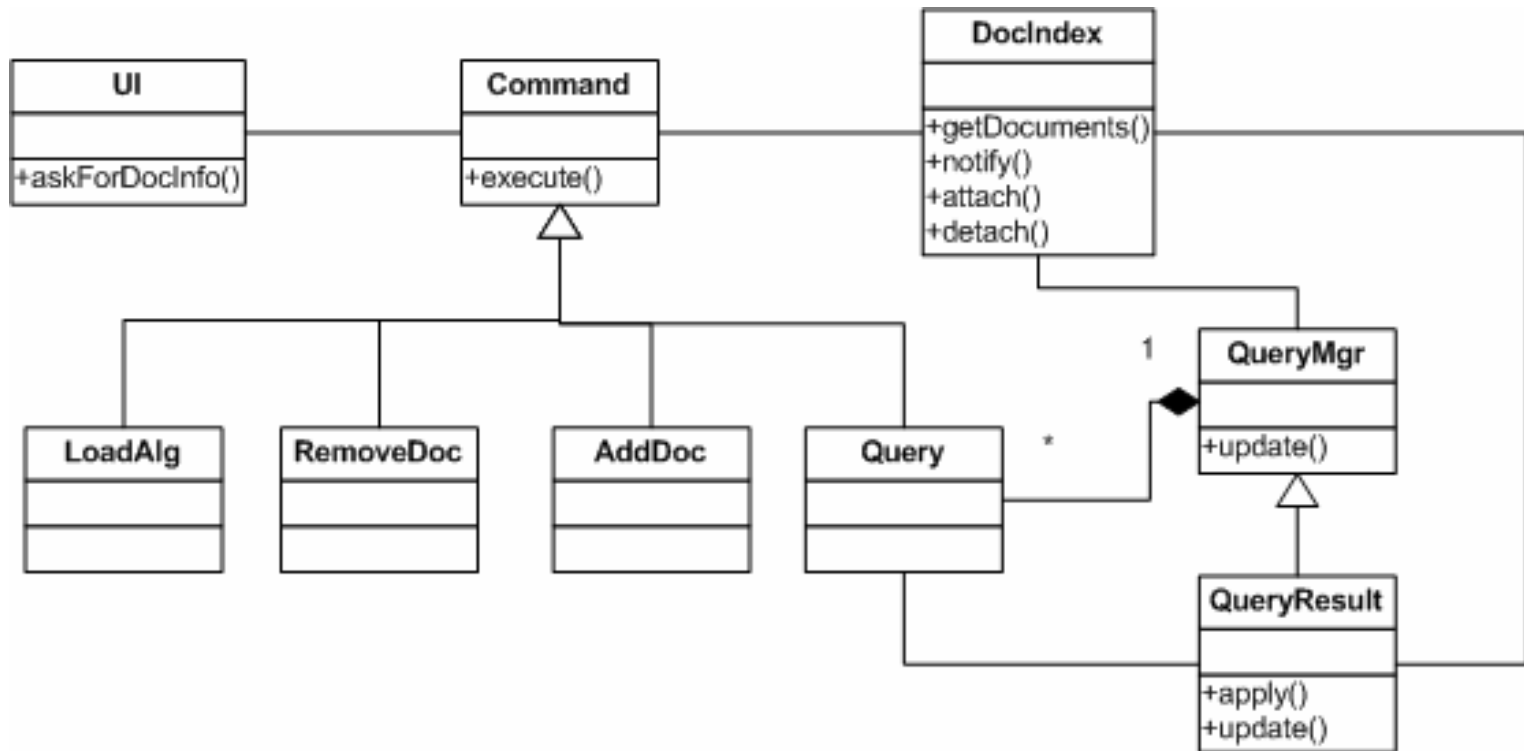
- Overview:
 - A system that enables the users to add and remove documents to and from the system, and retrieve documents based on simple keywords queries. This example is taken from 2002 fall term assignment 2.
- Key requirements:
 1. Document types include plain text, email, and HTML
 2. Queries and their results are stored
 3. Query results must be consistent with the current document collections in the system
 4. New queries can be added, and existing queries can be deleted or modified
 5. Allows dynamic use of different raw text keyword extraction algorithms
 6. Allows the use of off-the-shelf HTML keyword extraction package

More details of this system are available at:
<http://www.cdf.utoronto.ca/~neilg/CSC407F-2002/A2>

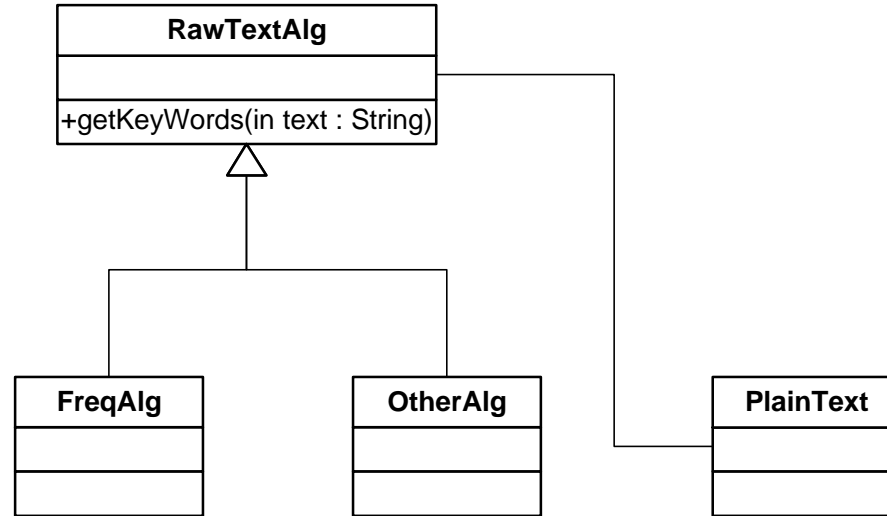
Document Retrieval System: Partial Solution Part I



Partial Solution Part II



Strategy Pattern Used in the Example



- Participants:
 - Context : PlainText
 - Strategy : RawTextAlg
 - ConcreteStrategy: FreqAlg, OtherAlg
- Applicability and purpose:
 - Allows the user to dynamically use different keyword extraction algorithms
 - Thus satisfies Key Req #5

Review and Exercises

We have seen many of the following patterns in the running example

- Behavioral
 - Observer
 - Visitor
 - Mediator (with modifications)
 - Strategy
 - Template method (Document and its subclasses)
- Structural
 - Adapter
- Creational
 - Singleton
 - Prototype
 - Factory method

How would you apply the following?

- Command
- Iterator