SE exam notes

from profs paper – how are **frameworks** like **components**?

how are they different?

frameworks are sold as products by vendors, and an application migt use several frameworks

frameworks are more customizable and have more complex interfaces

components = code reuse, frameworks = design reuse – provide a standard way of how components interface with each other, error handling

**Frameworks** and **Patterns**

patterns are just ideas, frameworks = ideas + code

A single framework usually contains many patterns, so these patterns are smaller than frameworks. Moreover, the design patterns cannot be expressed as C++ or Smalltalk classes and then just reused by inheritance or composition. Therefore, those patterns are more abstract than frameworks.

e.g. MVC framework and patterns relationship

Model/View/Controller can be decomposed into three major design patterns, and several less important ones [6].

It uses the Observer pattern to ensure the view’s picture of the model is up-to-date, the Composite pattern to nest views, and the Strategy pattern to cause views to delegate responsibility for handling user events to their controller.

**Patterns**

* INTERPRETER PATTERN says have a different expression evaluator function for each expression, all these subclasses should extend a main expression class
* VISITOR PATTERN separates algo from object. let every object accept visitor, and call visit function from each object.
* FAÇADE PATTERN – invoke one function and let it invoke all others
* TEMPLATE PATTERN – keep common parts of an algo in superclass, and differences in subclasses
* STRATEGY PATTERN – decide at runtime which implementation of an interface will be called.

Context determines which concrete implementation of strategy interface will be called.

* OBSERVER PATTERN
* COMMAND PATTERN – client decided whts the command, and receiver of command.

invoker invokes command

* ADAPTER PATTERN –

Q- diff between adapter and façade?

in both cases external class calls a function of adaptor/ façade class, which further calls adaptee / other class functions.