IMPORTANT NOTICE TO STUDENTS

These slides are **NOT** to be used as a replacement for student notes.
These **slides** are sometimes **vague and incomplete on purpose** to spark class discussions

JEE – Design Patterns

Front Controller

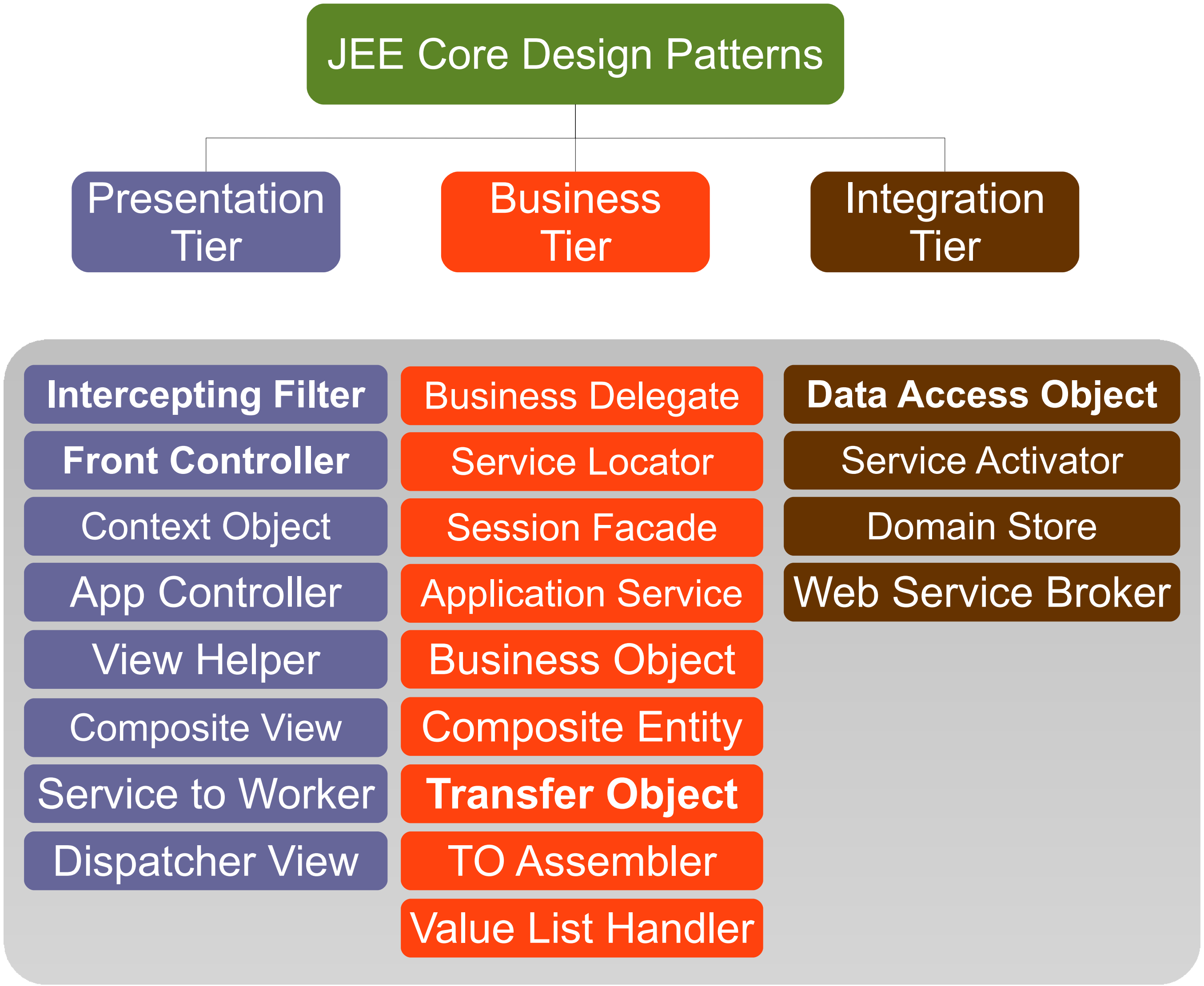
Intercepting Filter

Transfer Object

CS 446/646 ECE452
Jun 13th, 2011

WATERLOO
CHERITON SCHOOL OF COMPUTER SCIENCE

Overview



JEE Core Design Patterns

Presentation Tier

- **Intercepting Filter**
- **Front Controller**
- Context Object
- App Controller
- View Helper
- Composite View
- Service to Worker
- Dispatcher View

Business Tier

- Business Delegate
- Service Locator
- Session Facade
- Application Service
- Business Object
- Composite Entity
- **Transfer Object**
- TO Assembler
- Value List Handler

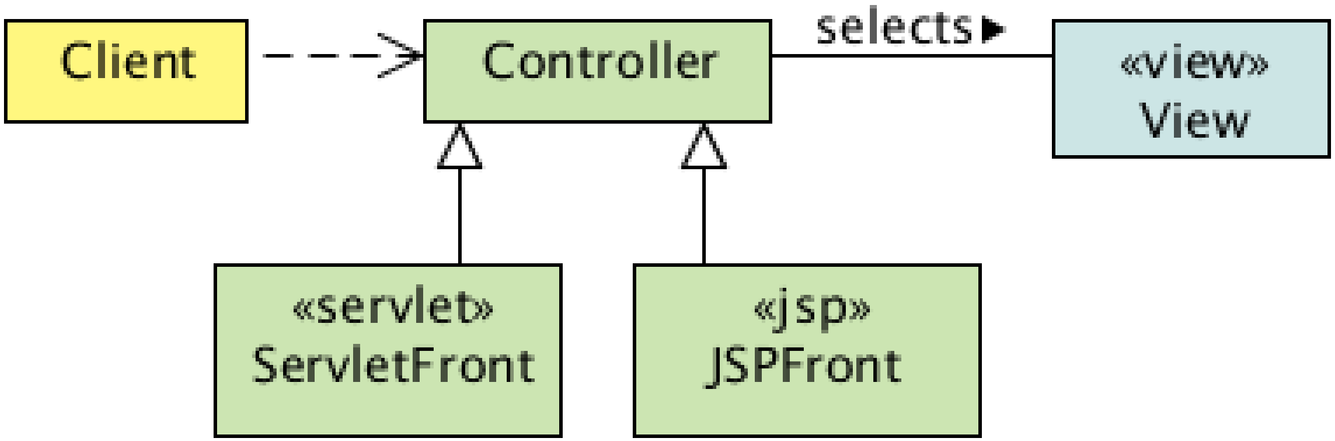
Integration Tier

- **Data Access Object**
- Service Activator
- Domain Store
- Web Service Broker

Front Controller

Intent

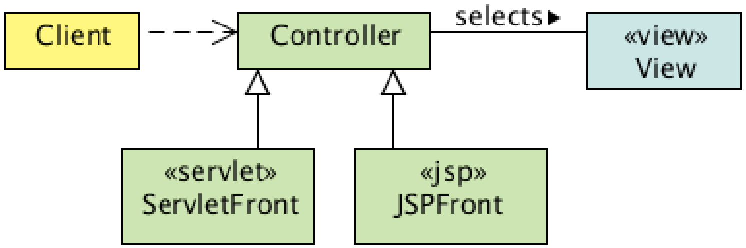
- provide a single point for processing user requests



http://java.sun.com/blueprints/corej2eepatterns/Patterns/FrontController.html

Front Controller

Motivation



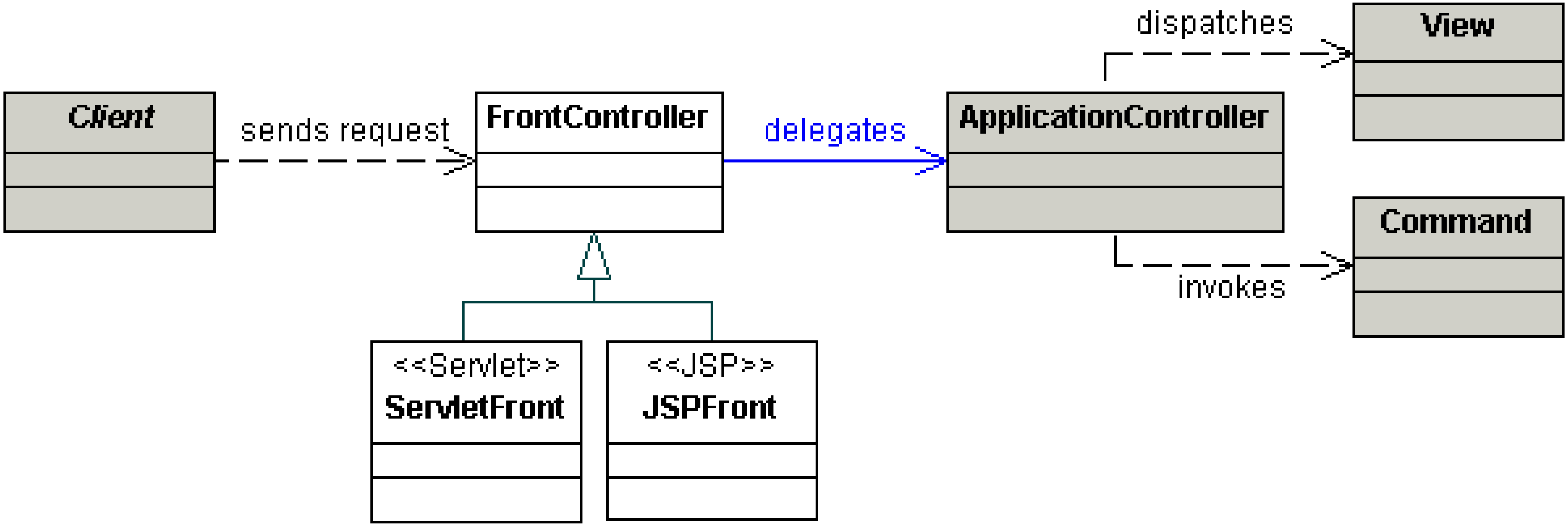
- single processing point for all client requests
  - across views & session
  - can be used to inject cross-cutting concerns
    - logging
    - security
  - **can we have multiple controllers?**
  - **we can map multiple requests to the same controller?**

http://java.sun.com/blueprints/corej2eepatterns/Patterns/FrontController.html

Front Controller

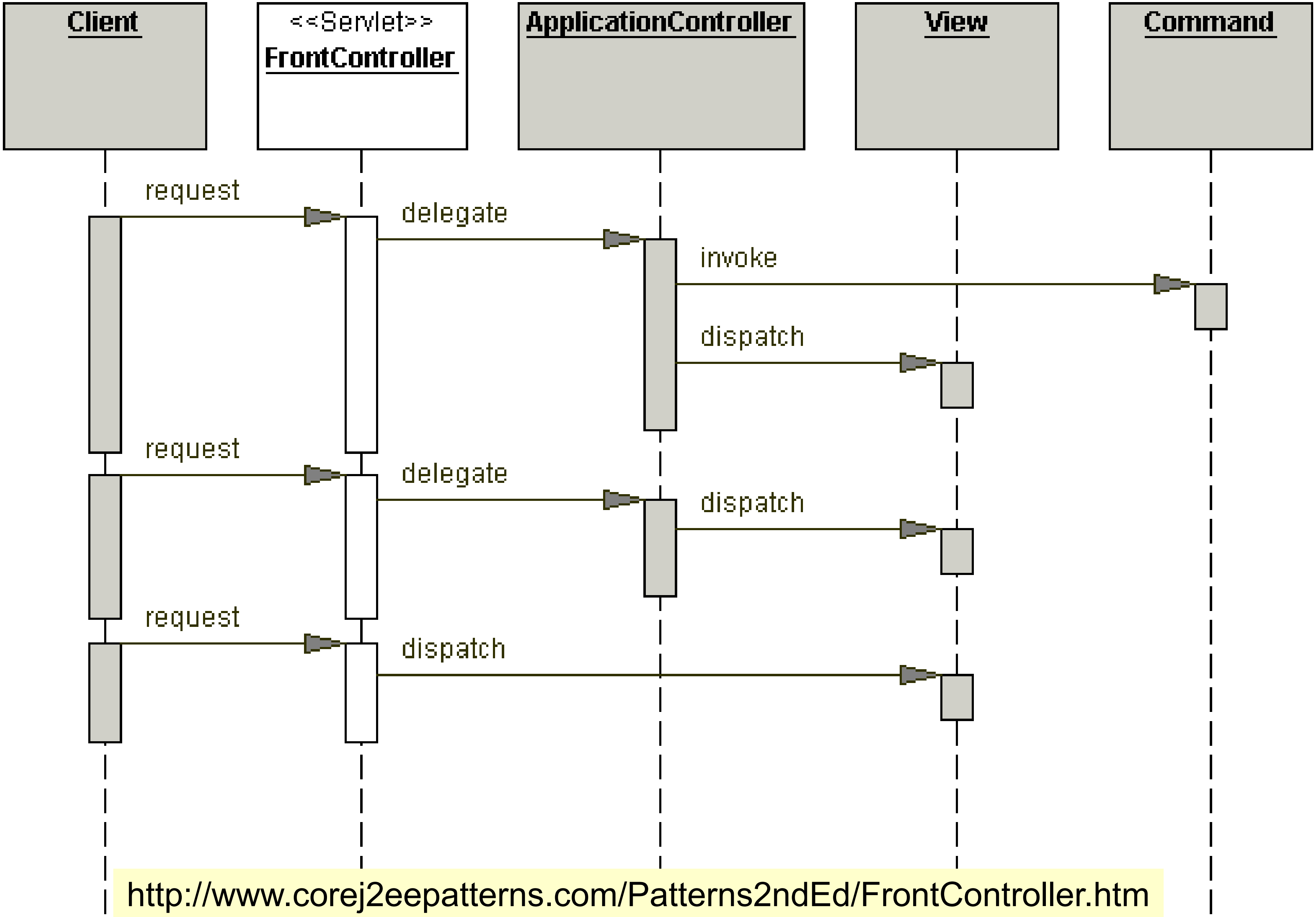
Motivation

- separation of concerns
  - business code from presentation code



http://www.corej2eepatterns.com/Patterns2ndEd/FrontController.htm

Front Controller



http://www.corej2eepatterns.com/Patterns2ndEd/FrontController.htm

Front Controller

Motivation

- provides resource mapping
  - physical
    - http://server/**resource.jsp**
  - virtual/logical
    - http://server/servlet/**resourceController**
    - http://server/servlet/**page1.help**
    - **logical partitioning of application views?**

Front Controller

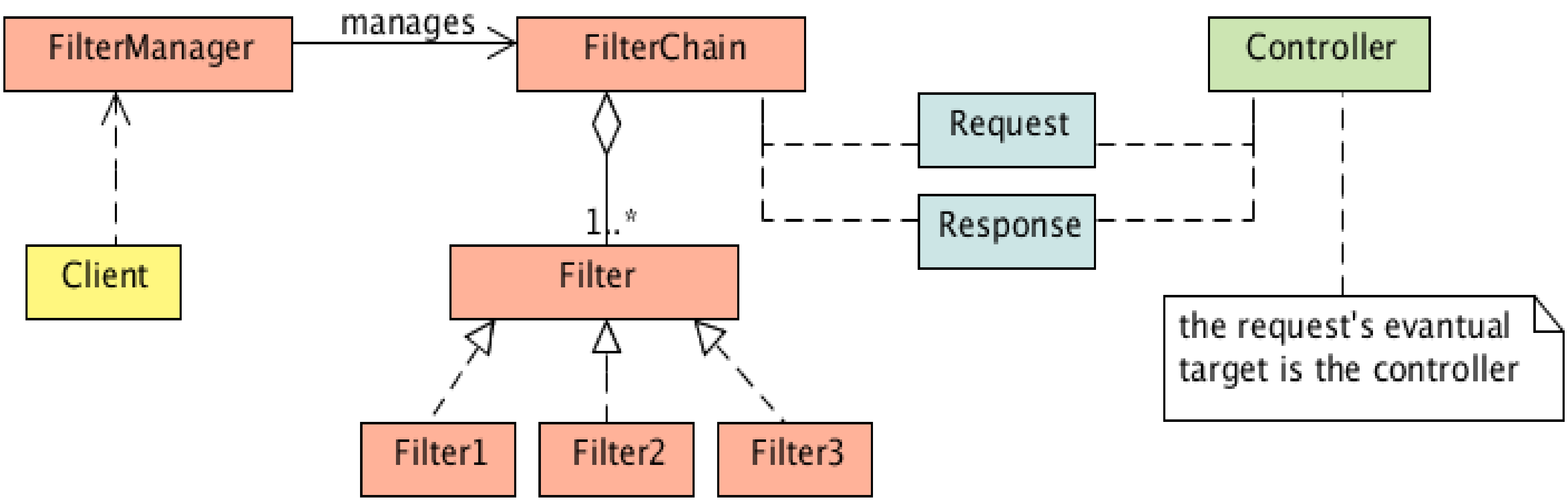
Motivation

- organic growth
  - controllers can be specialized (sub-classing)
    - GWTServlet
- reusability
  - **declarative** vs. non-decalrative mapping
  - ***what is declarative mapping?***
  - ***why do we care?***

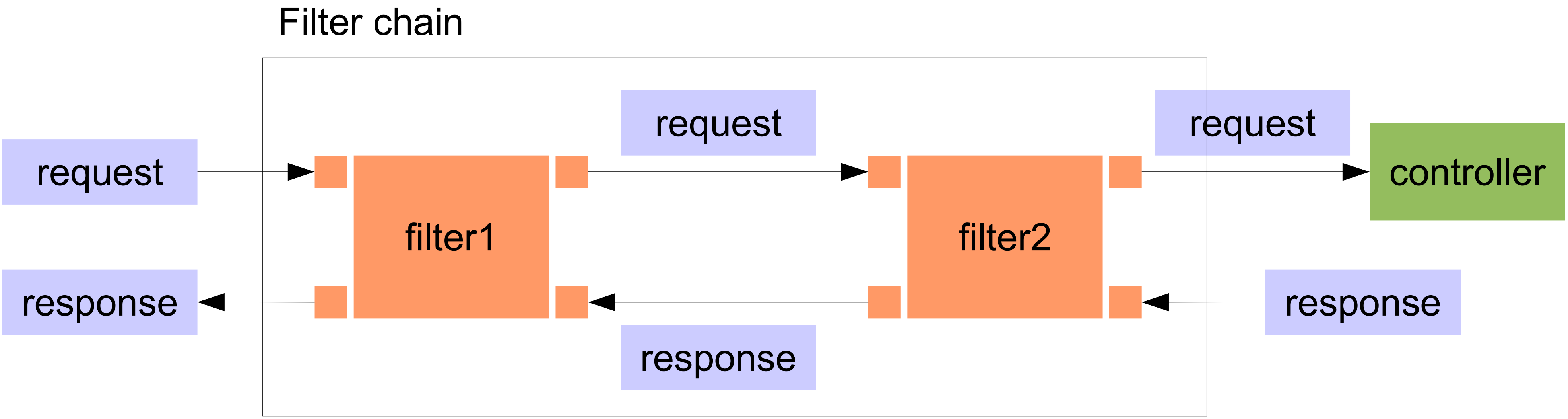
Intercepting Filter

Intent

- common mechanism for pre & post processing of user requests



Intercepting Filter



how is this different from traditional pipes & filters?

how is this different from batch pipes & filters?

what type of data flows through the pipes?

Intercepting Filter

Context

- "presentation-tier request handling mechanism receives many ***different types of requests***, which require ***varied types of processing***"*

Common Examples*

- has the client been **authenticated**?
- does the client have a **valid session**?
- is the client's IP address from a **trust**ed network?
- does the request path violate any constraints?
- what encoding does the client use to send the data?
- do we support the **browser type** of the client?

* http://java.sun.com/blueprints/corej2eepatterns/Patterns/InterceptingFilter.html

Intercepting Filter

Easy Solution

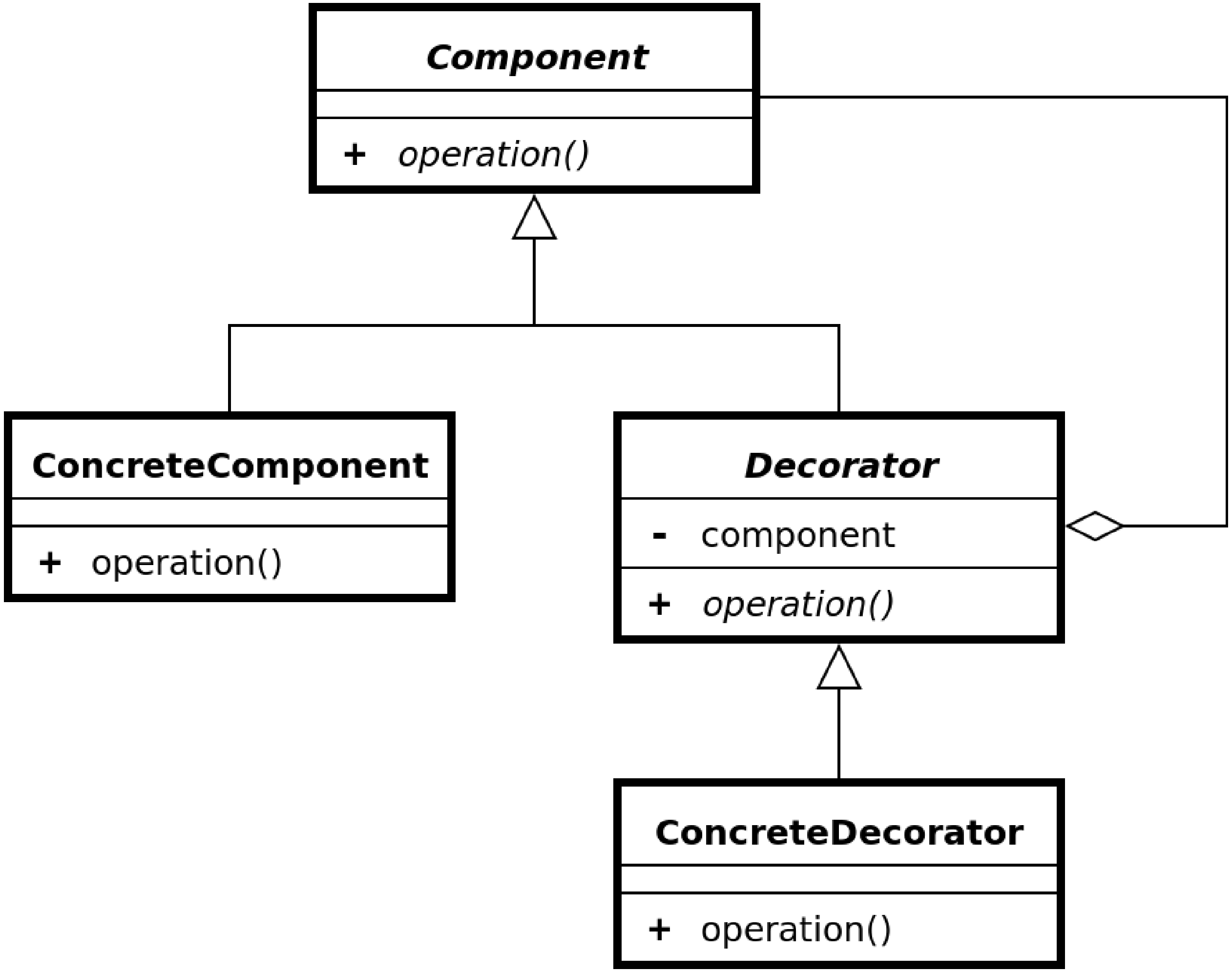
- if-then-else in the controller (***why not?***)

An Elegant Solution

- common reusable processing
- easy to add/modify/delete filtering functionality
  - even better if declarative

Intercepting Filter

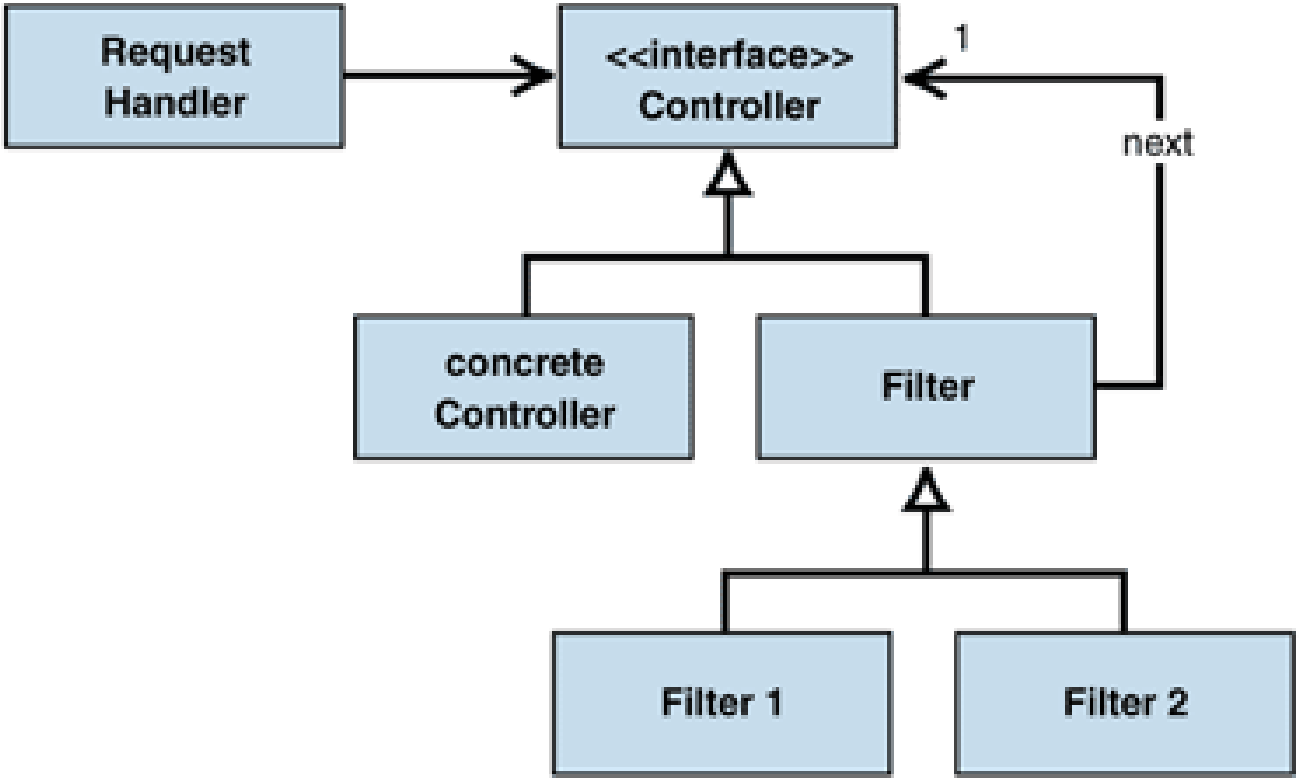
Sounds like Decorator (**how?**)



http://en.wikipedia.org/wiki/Decorator_pattern

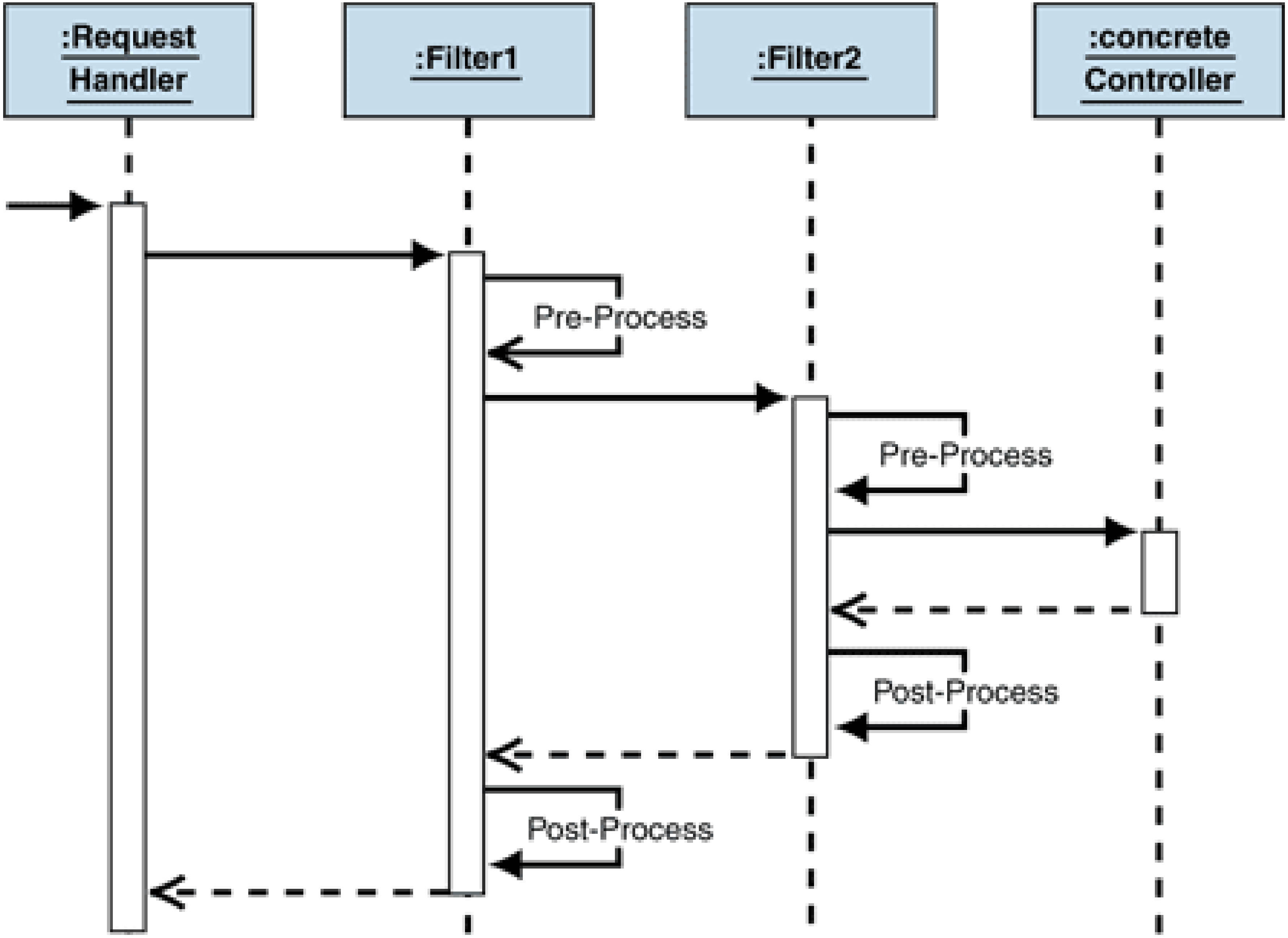
Intercepting Filter

Sounds like Decorator



http://msdn.microsoft.com/en-us/library/ff647251.aspx

Intercepting Filter



http://msdn.microsoft.com/en-us/library/ff647251.aspx

Intercepting Filter

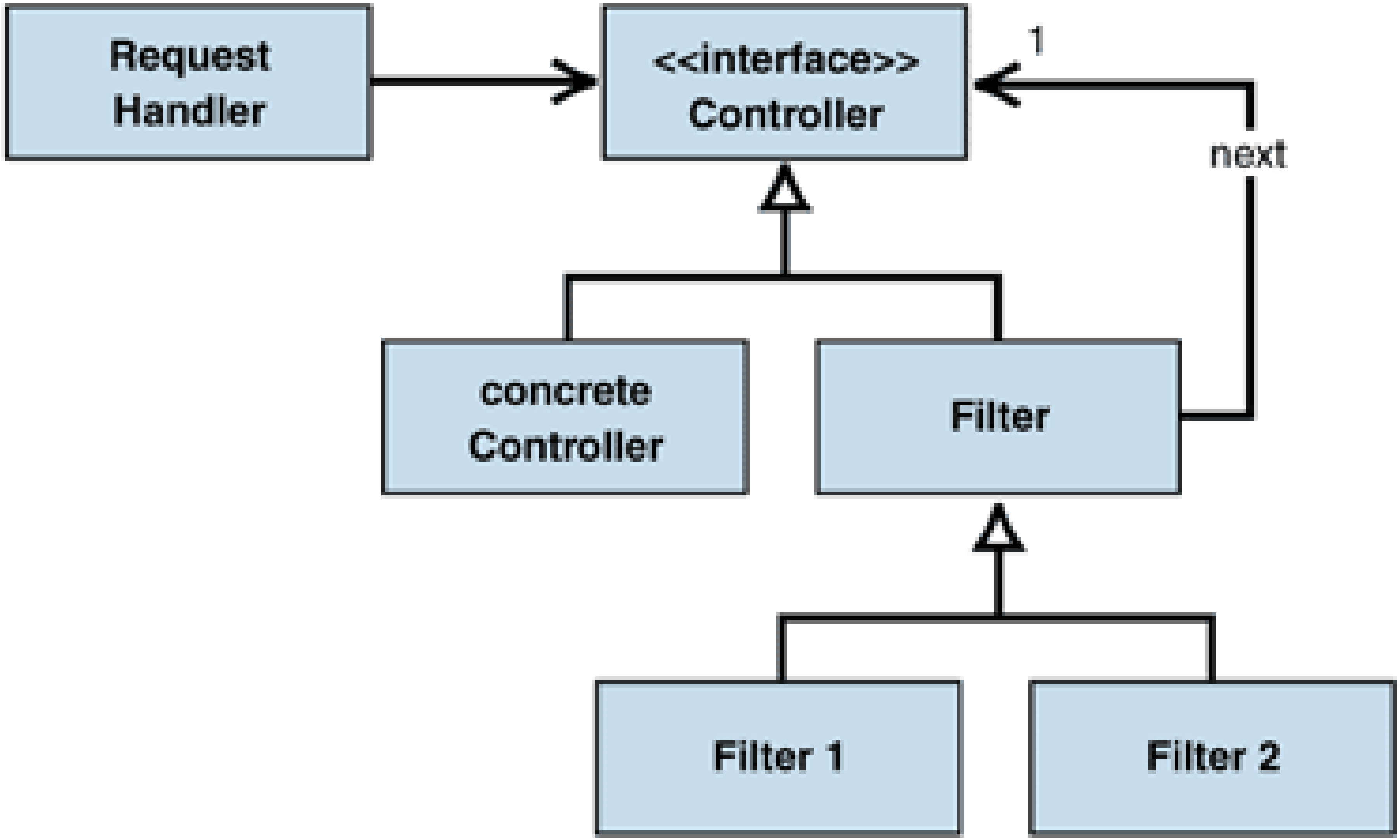
What is the different between the two?

- structure
  - FilterChain is no longer required

- runtime behaviour

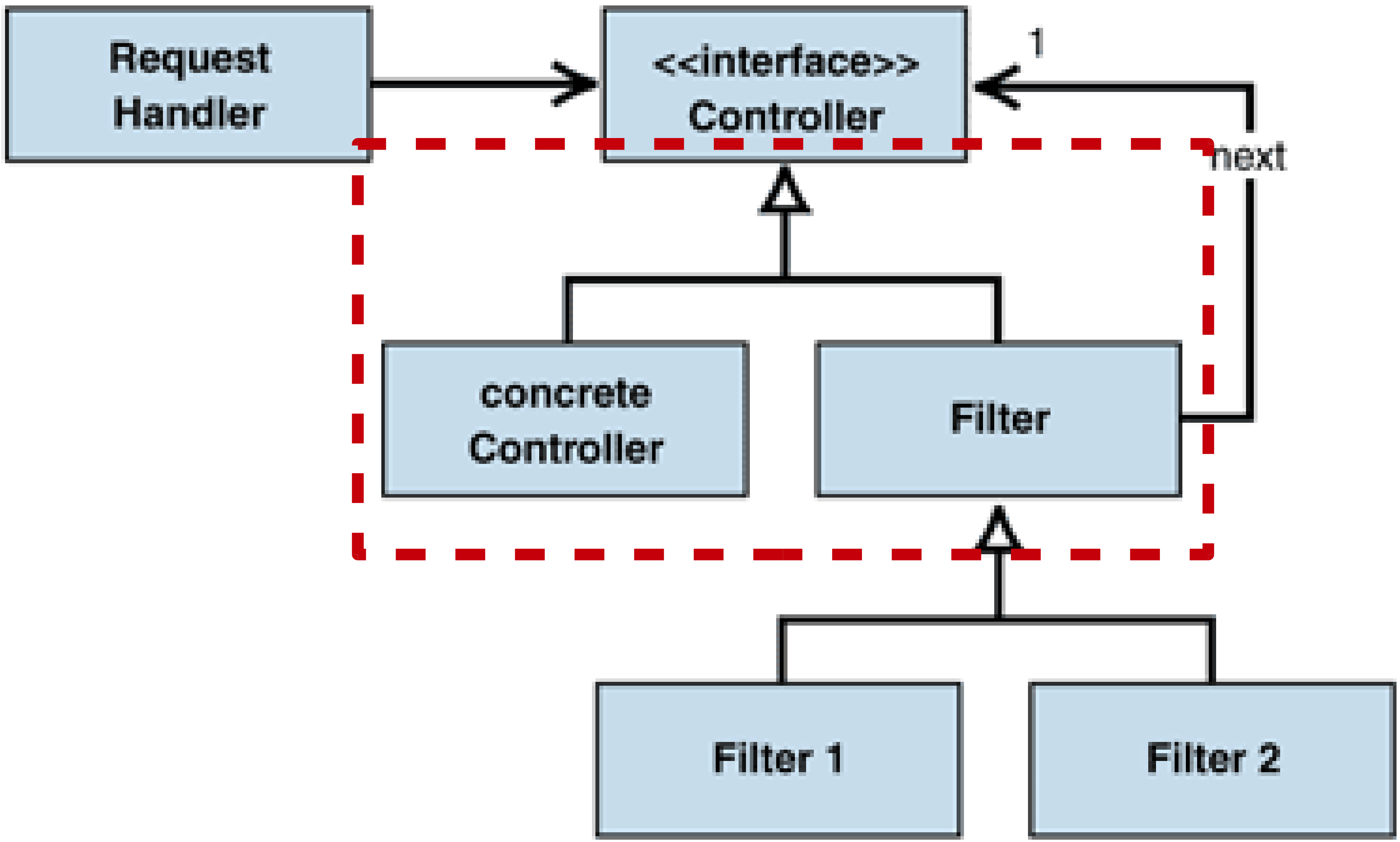
- take another look at the diagram

Intercepting Filter



http://msdn.microsoft.com/en-us/library/ff647251.aspx

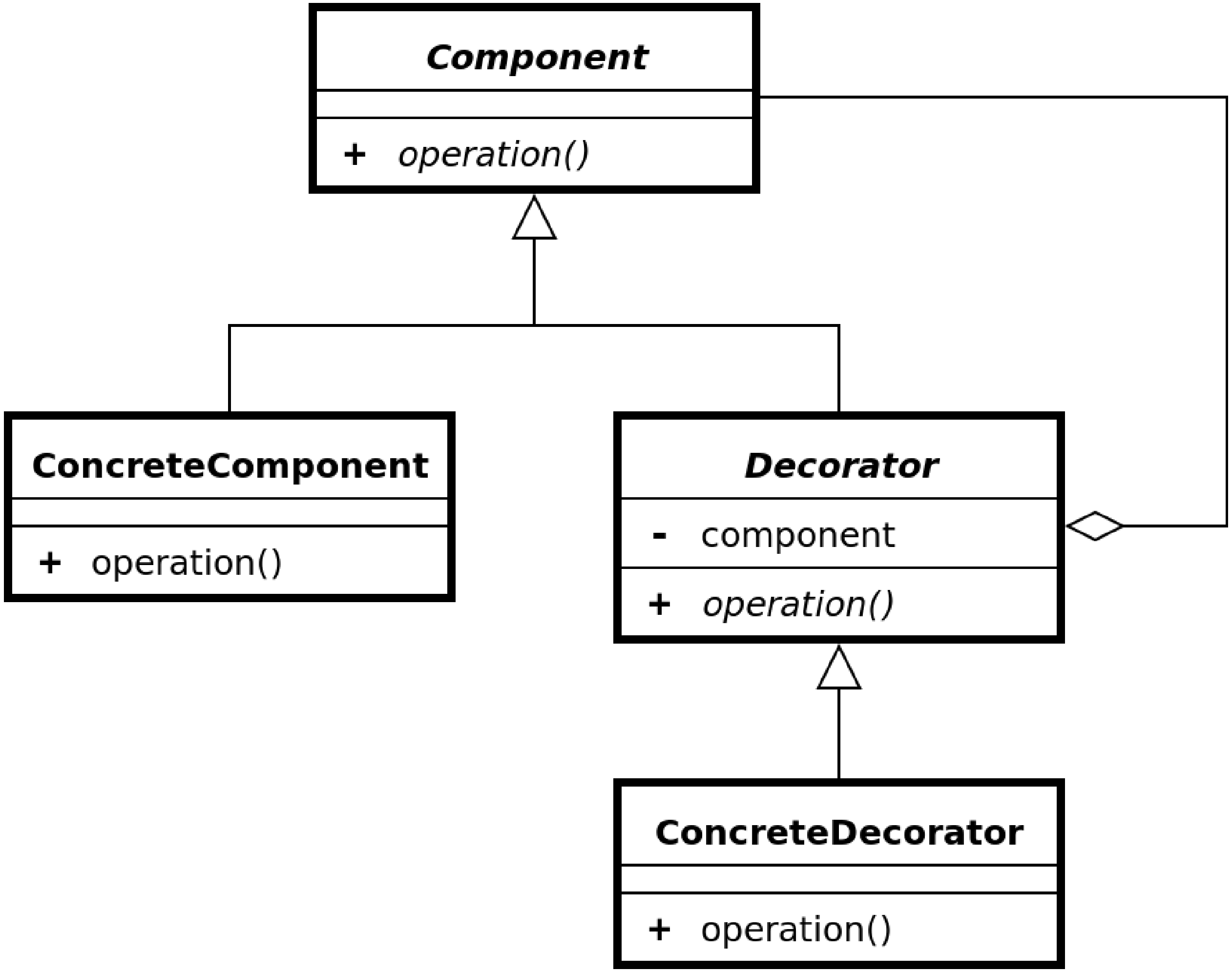
Intercepting Filter



http://msdn.microsoft.com/en-us/library/ff647251.aspx

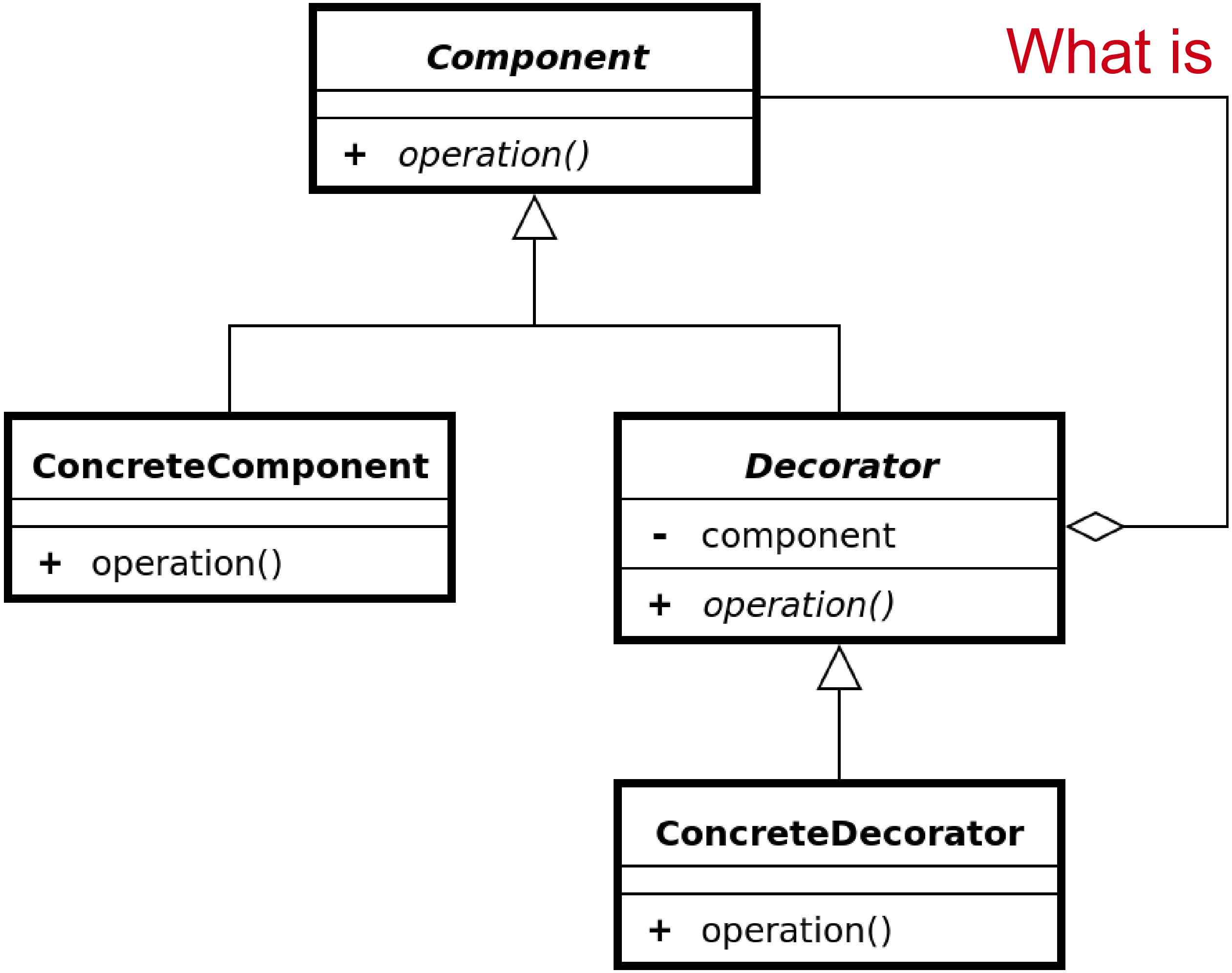
Intercepting Filter

Evolution



Intercepting Filter

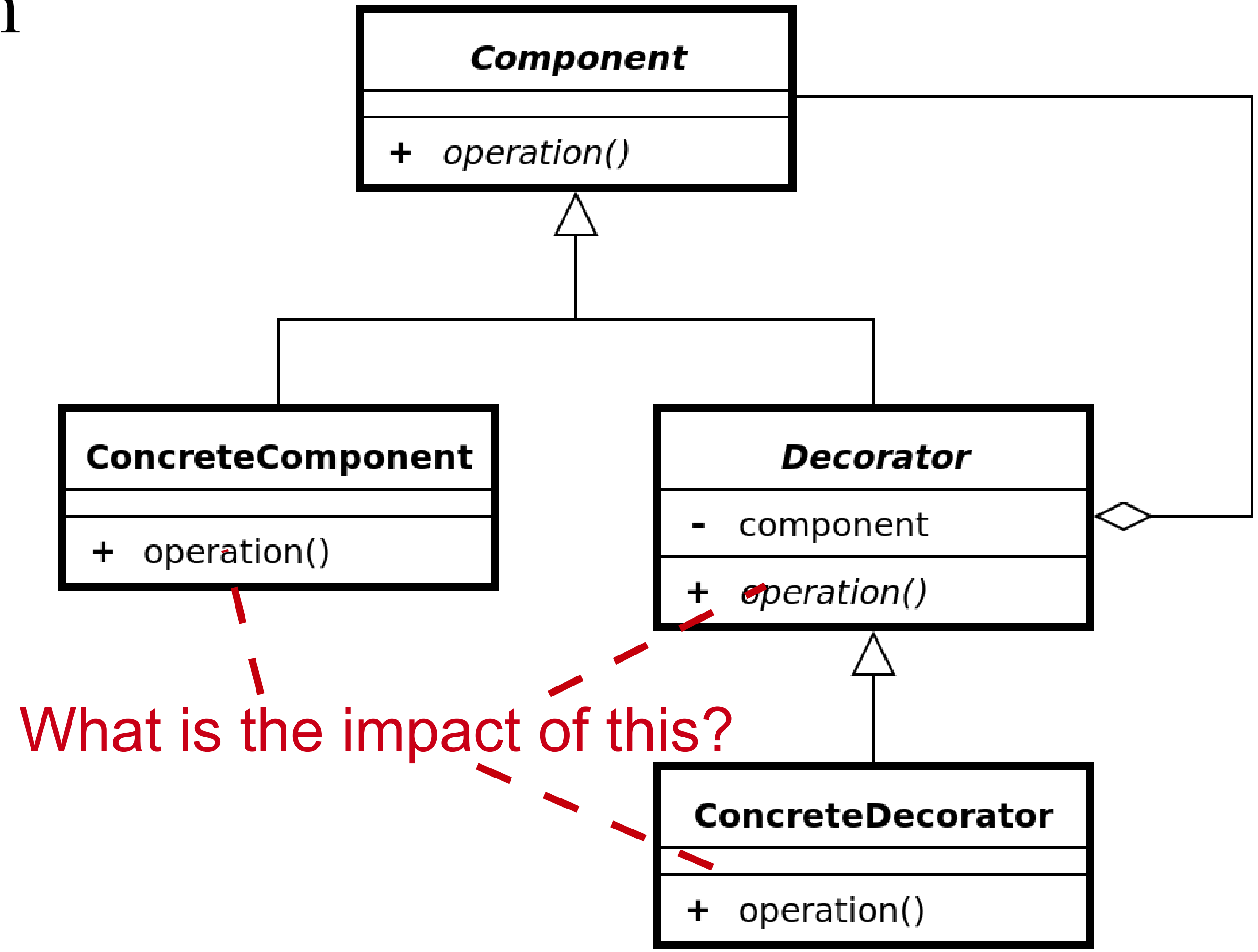
Evolution



What is the impact of this?

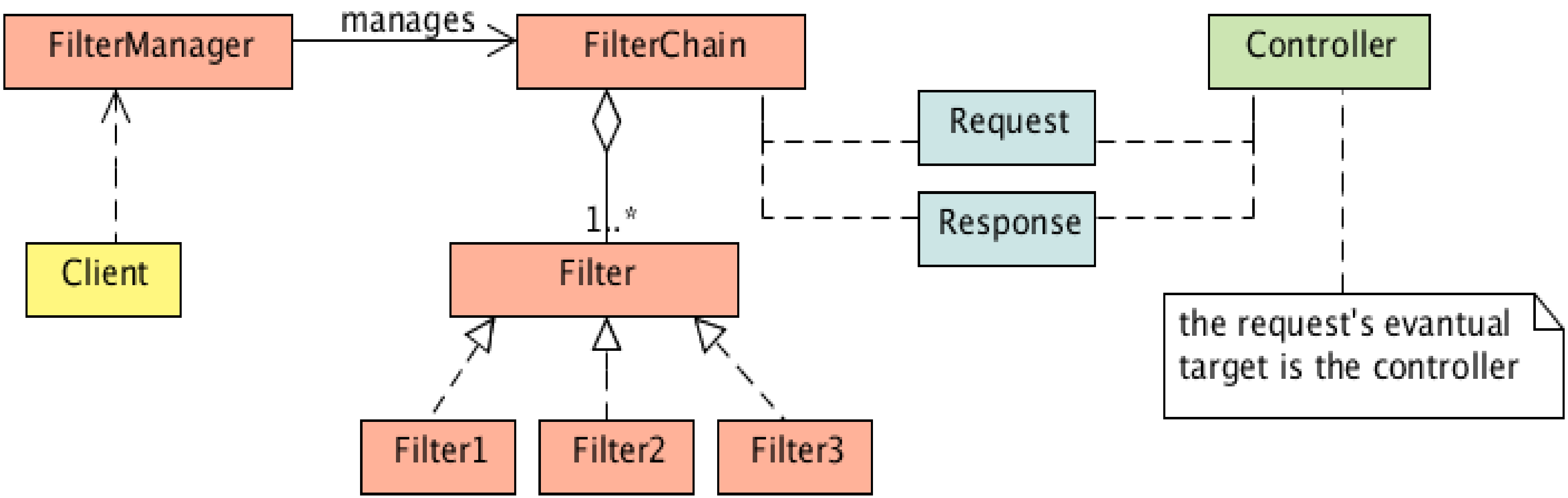
Intercepting Filter

Evolution



What is the impact of this?

Intercepting Filter



List some drawback of this design?

Intercepting Filter

Motivation

- functionality injection
- improved reusability
  - filter chains can be defined in a number of ways
- declarative configuration

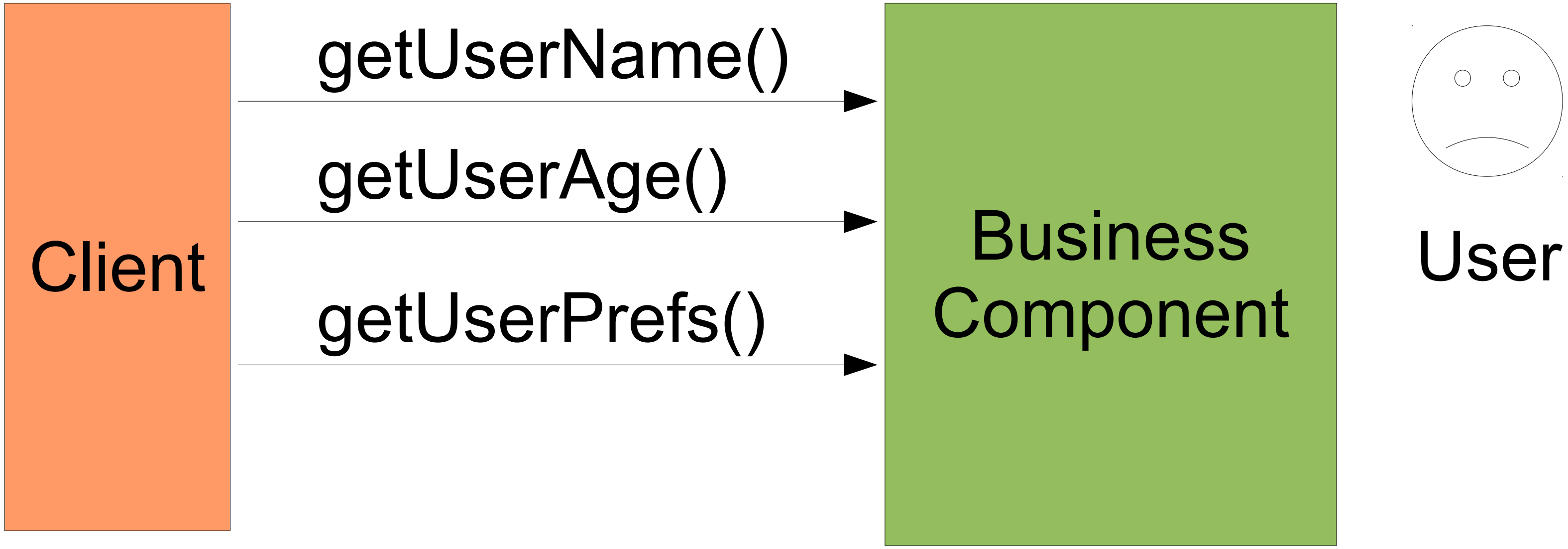
Difficulties

- information sharing
- fault-tolerance

Transfer Object

Intent

- to transfer multiple data elements over a tier



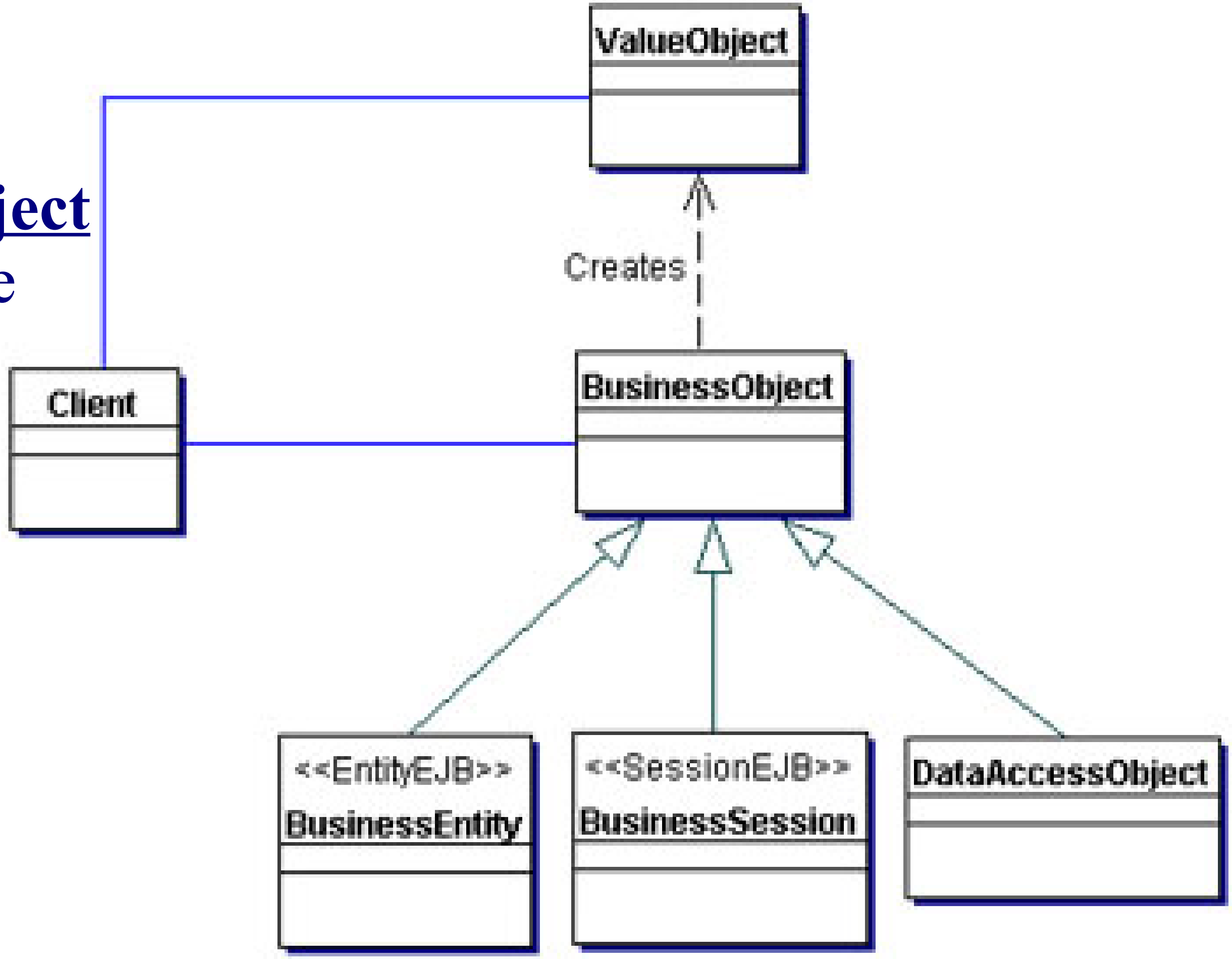
Looks harmless??

http://java.sun.com/blueprints/corej2eepatterns/Patterns/TransferObject.html

Transfer Object

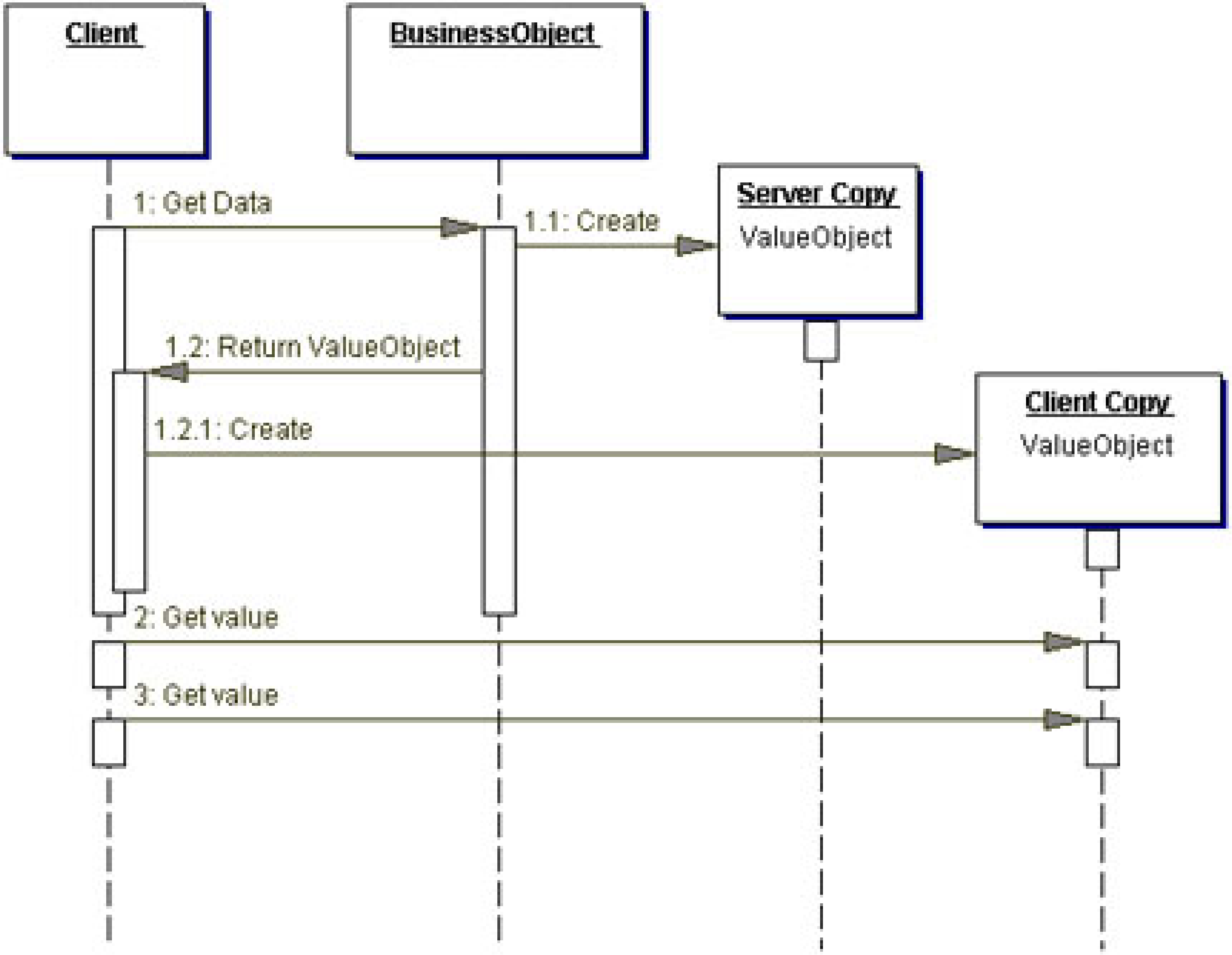
Solution

- use a **Transfer Object** to encapsulate the data



http://java.sun.com/blueprints/corej2eepatterns/Patterns/TransferObject.html

Transfer Object

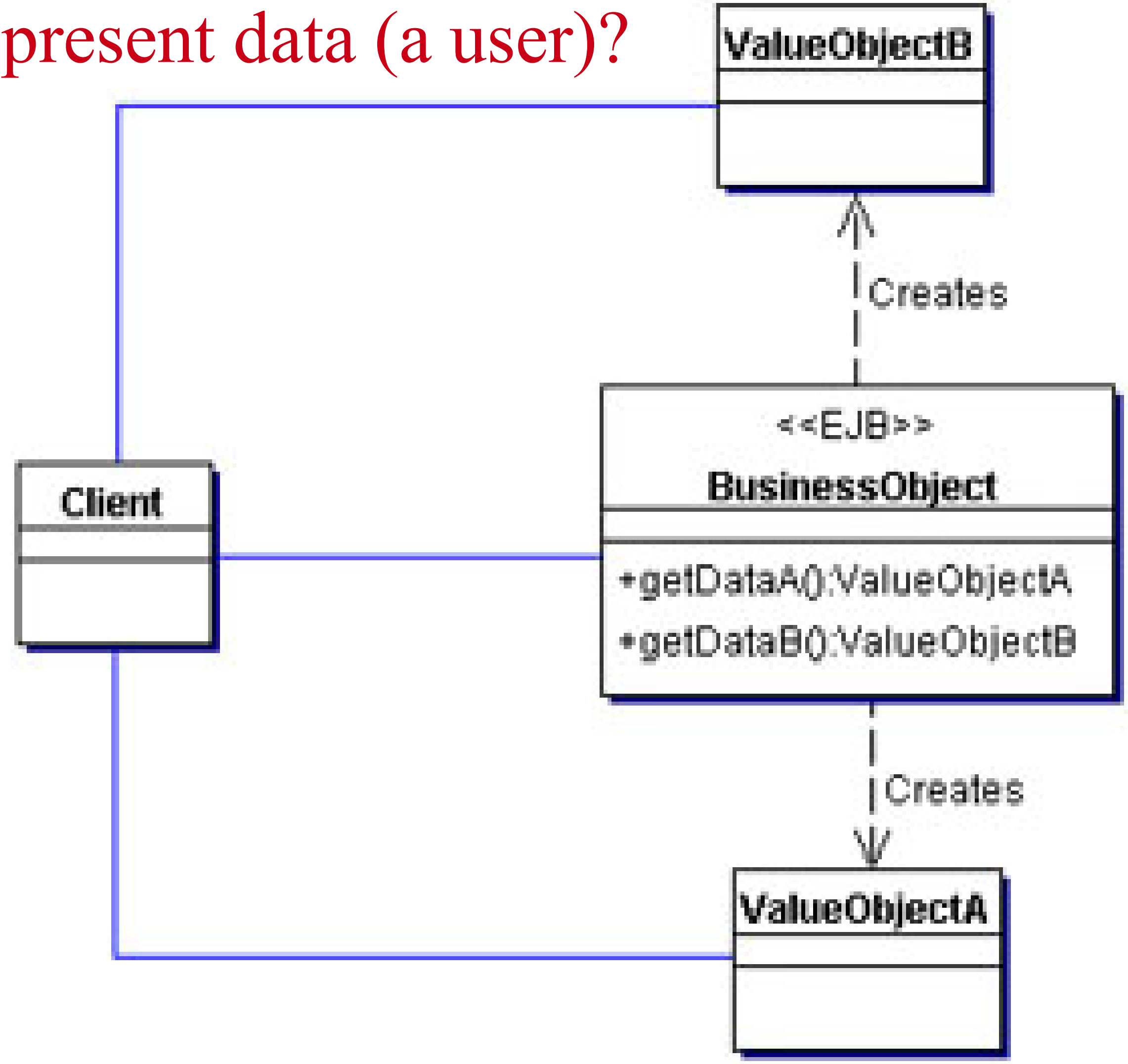


http://java.sun.com/blueprints/corej2eepatterns/Patterns/TransferObject.html

Transfer Objects

Scope & Complexity?

- can we have multiple types of transfer objects to represent data (a user)?
- create the transfer objects required
- how complex?



Transfer Object

Consequences

- simplified remote interface
- fewer remote calls
- **stale objects**: very common problem for web apps
  - what are stale objects?
  - how would you deal with it?
- serialization over the wire?