# CS 417-505
## Design Patterns

## Observer and State patterns

Dr. Chad Williams
Central Connecticut State University

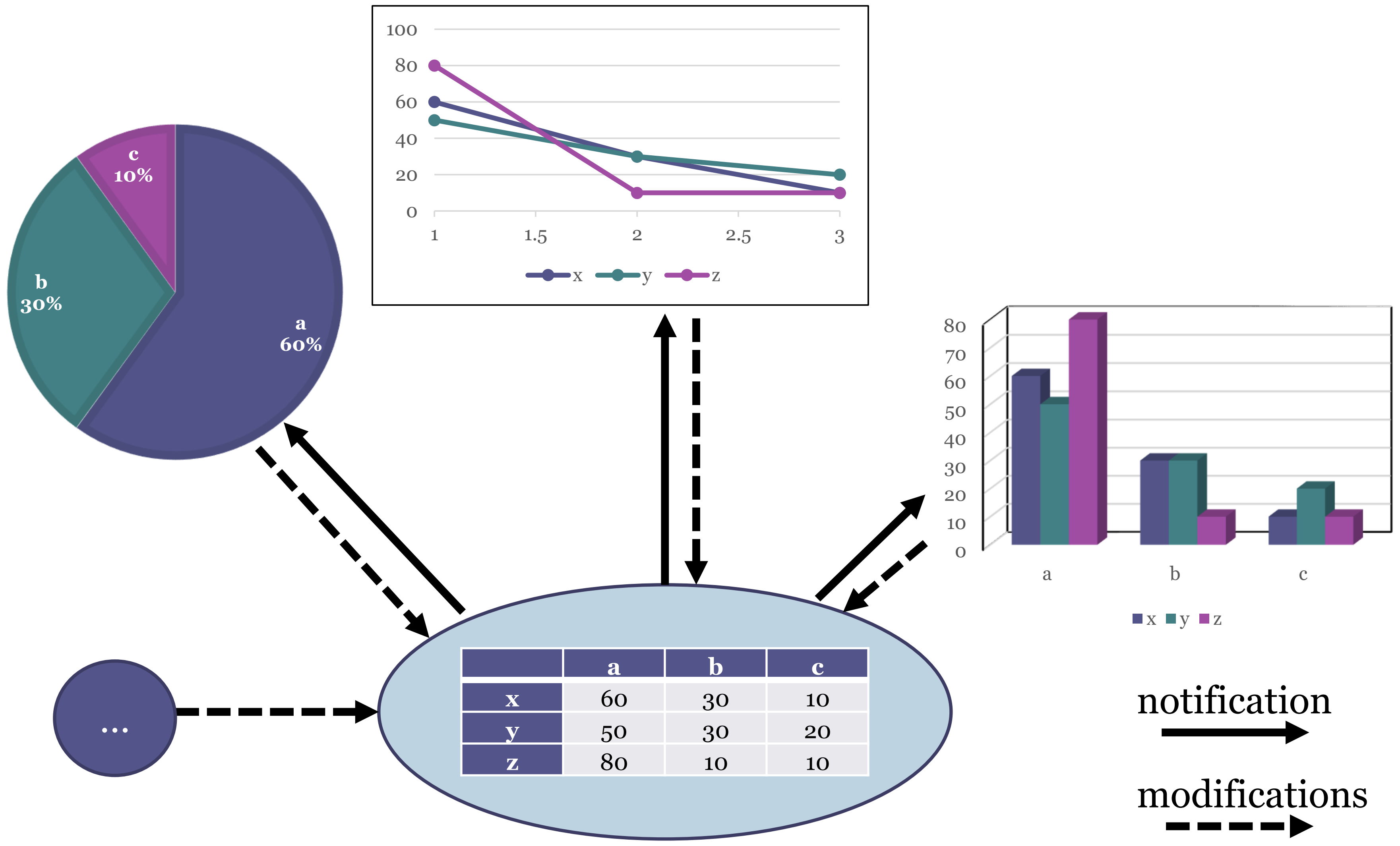
## Announcements

- **CS 505 teams**
  - Java interfaces, Javadoc for supported component due this Friday 10/27 to public team repo
  - UML for component will be due same day to our private team repo
- **First sprint will be due 11/3 (next Friday)**
  - Specific patterns and link to create private team repos will be posted this evening
  - UML class diagram
    - Add a note as to which classes are part of which pattern
  - Make sure you use packages where appropriate
  - All non-trivial objects are well behaved (equals, toString, hashCode)
  - Don’t forget your JUnit for any non-trivial functions

## Design pattern: Observer

- **Category**: Behavioral design pattern
- **Intent:**
  - Define one-to-many dependency so that when one object changes state all of its dependents are notified and updated automatically
- **Motivation**
  - Collection of cooperating objects needing to maintain consistency/updates without becoming tightly coupled

# Motivation cont.



| | a | b | c |
|---|---|---|---|
| x | 60 | 30 | 10 |
| y | 50 | 30 | 20 |
| z | 80 | 10 | 10 |

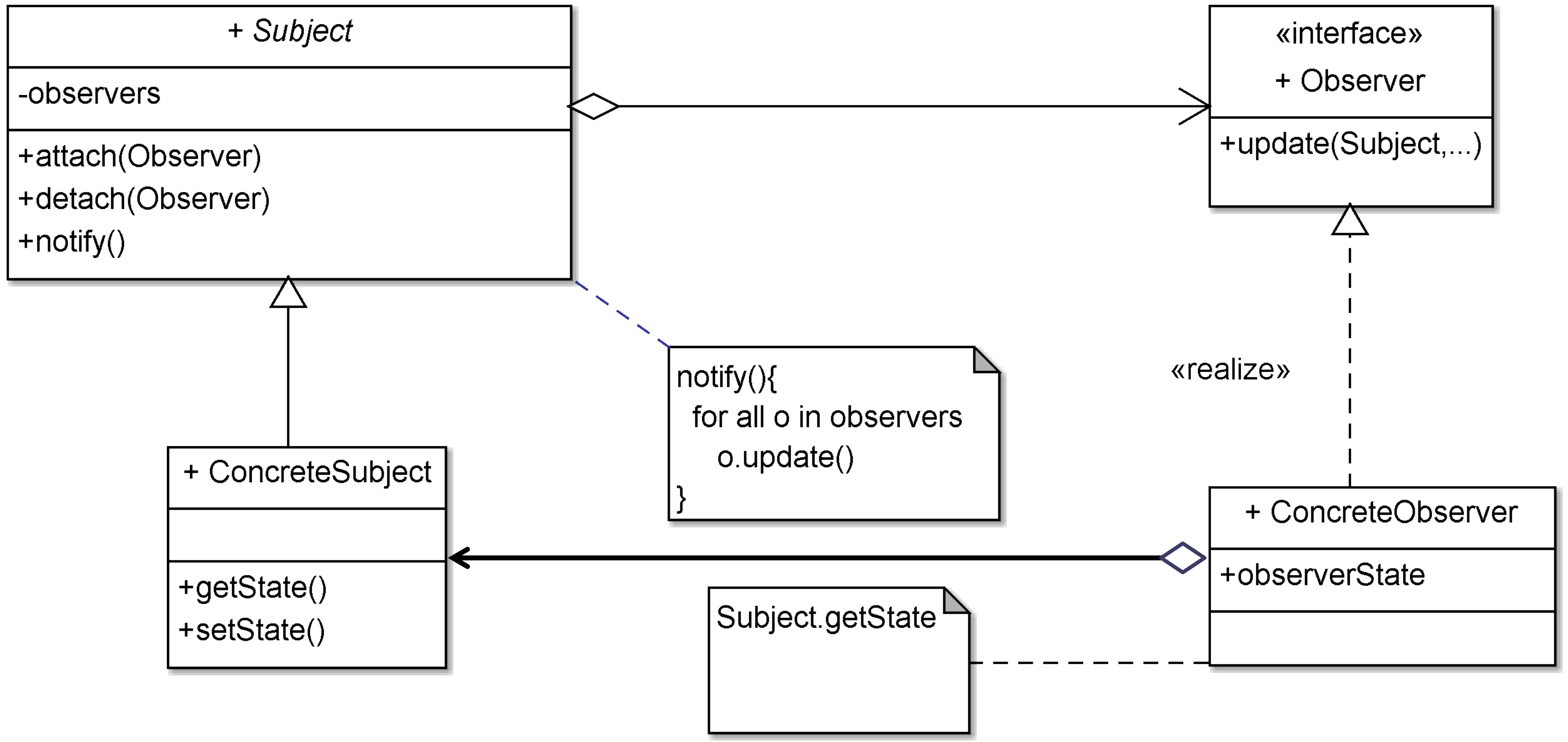
## Applicability

- Abstraction has two aspects, one dependent on the other. Encapsulating in separate objects lets you vary and reuse them independently
- Changing one requires changing others and you don’t know how many others will need to change
- Object should be able to notify others without making assumptions about that object – you don’t want the objects tightly coupled

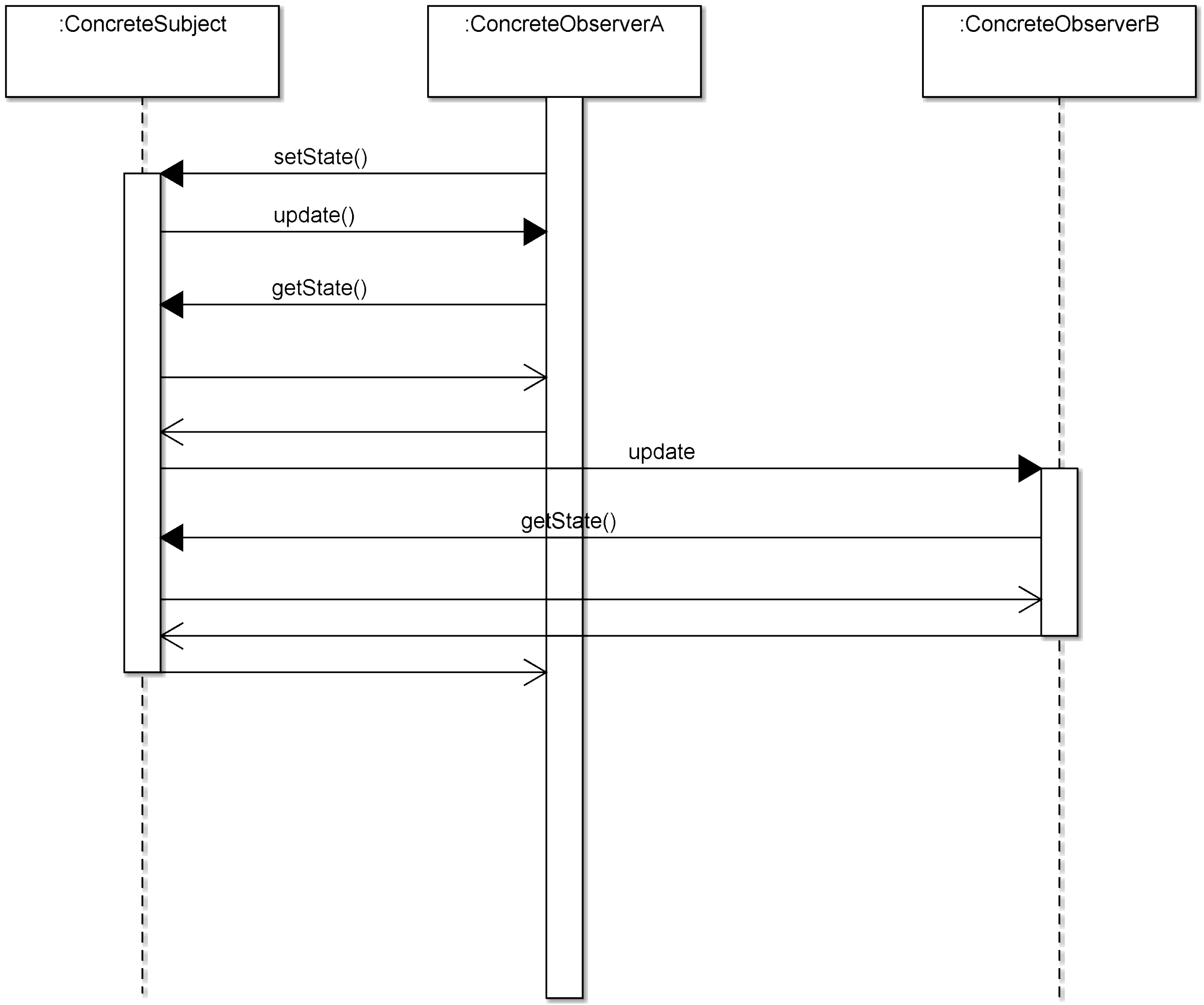
# Participants

- Subject
  - Knows its observers.  Any number of Observer objects may observe
  - Provides interface for attaching/detaching Observer object
- Observer
  - Defines an updating interface for objects that should be notified of changes in a subject
- ConcreteSubject
  - Stores state of interest to ConcreteObservers
  - Sends notification to its observers when its state changes
- ConcreteObserver
  - Maintains reference to ConcreteSubject object
  - Stores state that should stay consistent with the subject's
  - Implements the Observer updating interface

## Observer UML



# Sequence


:ConcreteSubject
:ConcreteObserverA
:ConcreteObserverB
setState()
update()
getState()
update
getState()

## In class examples

- Excel document
- Class registration waitlist

## Design pattern: State

- **Category**: Behavioral design pattern
- **Intent:**
  - Allow an object to alter its behavior when internal state changes.  The object appears to change its class
- **Motivation**
  - Significant changes in behavior of same object depending on state
  - Reduce complexity of long conditional logic

## Applicability

Use in either of these cases:

- Object’s behavior depends on its state, and it must change its behavior at runtime depending on state
- Operations have large multipart conditional logic with several containing same conditional structure

## Participants

- Context
  - Class defines the interface of interest to client
  - Maintains an instance of ConcreteState subclass that defines current state
- State
  - Defines interface for encapsulating the behavior associated with particular state of the Context
- ConcreteState subclasses
  - Each subclass implements a behavior associated with a state of the Context

## In class examples

- TCP connection
  - Open
  - PassiveOpen
  - Closed
- Phone
  - Off
  - Locked
  - On
  - Camera