Software Design: Advanced UML + Project Overview

Content in part from Chapter 5 of "Head First Software Development", Pilone et al.

Miami University Software Technology & Analysis Group (MUSTANG)
Computer Science & Software Engineering
Miami University, Oxford, Ohio, USA

Week's Agenda

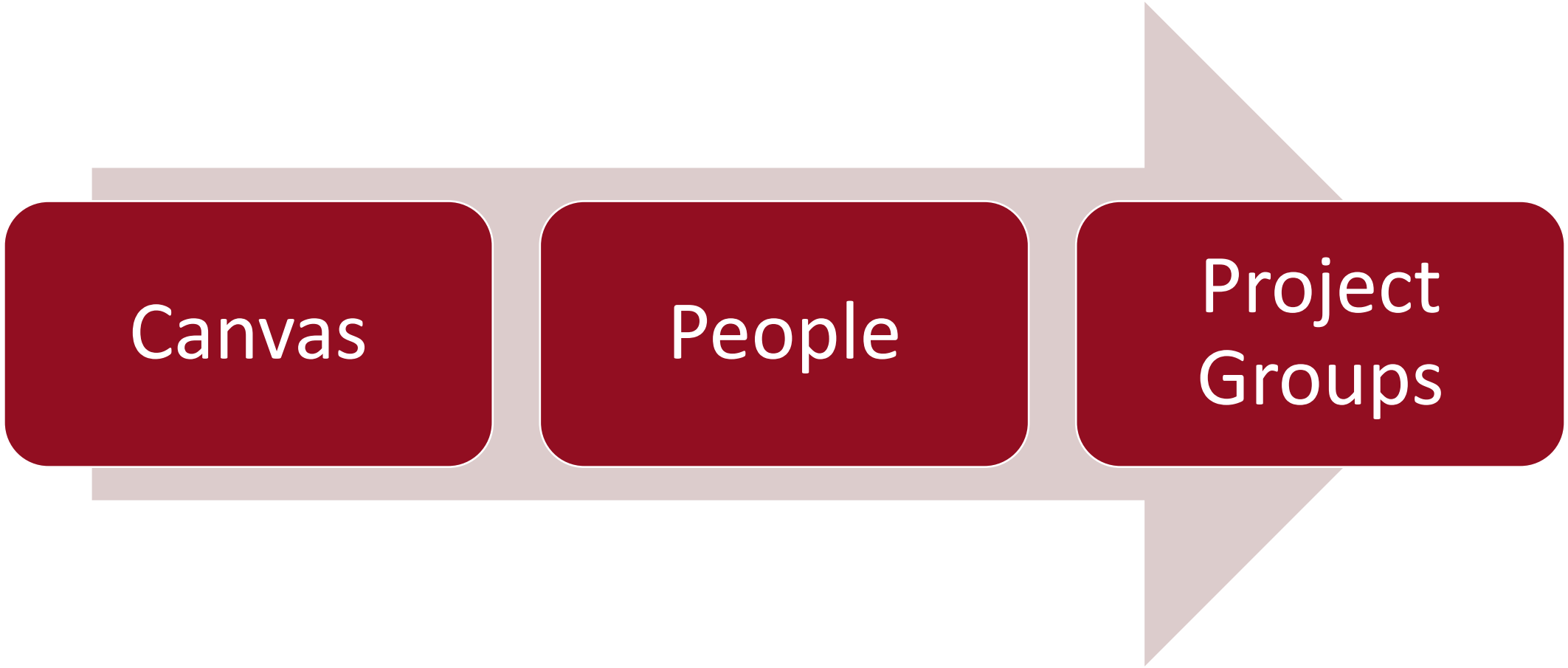
- Assignment #2 solution
- Project introduction
  - Fill out schedule survey this week.
  - Due by Friday!
  - 1st weekly customer meetings begin next week!
- Software design
  - Advanced UML
  - Sequence Diagrams & Design Patterns
- Midterm information
  - Thursday March 11th

PROJECT

PROJECT SCHEDULE

Week	Phase	Description
6	Preparation	Preparation and Planning. Learn technologies, and begin requirements elicitation, design, and plan iterations. Set up version control, and frameworks.
7		
8		
9	Iteration #1	**Apr 6th:** Preliminary demonstration. Presenting your requirements, design, and plans and all of the above and what you will finish by end of Iteration #1.
10		
11	Iteration #2	**April 20th:** Presentation of what you will finish by end of Iteration #2.
12		
13	Iteration #3	**May 4th:** Final features, testing, and documentation. Final presentation. What you will finish by end of Iteration #3.
14		

Project Teams



PROJECT

- This is software engineering!
  - This course is ABOUT the PROCESS
  - Graded on how well you do the steps **more so than the product**

Project Overview

- Creating a catalog application
- Default option is “Meta-App” App Catalog
- Whatever your choose, must meet following criteria:
  - Entries must have fields
    - Name
    - Description
    - 4 other fields
  - Same features as those described in Meta-App specification
    - Admins, moderators, users
    - Search bar, Request forms
    - And more!
  - Features agreed upon between you and customer!

META-APP

- Project specification is on Canvas

A41 | 3mensio Vascular

	A	B	C	D	E
1	Application Name	Major Versio	Minor Versio	Company Name	Company Home Page
2	.NET Memory Profiler	3	5	SciTech Software AB	http://www.scitech.se/
3	.NET Reporting Engine	8		Windward Studios Inc.	http://www.windwardreports.co
4	.print Application Server Engine	7	6	ThinPrint	http://www.thinprint.com/
5	.print Engine for Remote Web Workplace			ThinPrint	http://www.thinprint.com/
6	.print RDP Engine	7	6	ThinPrint	http://www.thinprint.com/
7	.print Server Engine	7	6	ThinPrint	http://www.thinprint.com/
8	@task	4	11	AtTask, Inc	http://www.attask.com/
9	100,000 ClipArt	3	12	Avanquest	http://www.avanquest.com/
10	100,000 Mahjongg Games	1	0	Viva Media	http://www.viva-media.com/
11	1001 Japanese Crosswords	1	0	Viva Media	http://www.viva-media.com/
12	1001 Minigolf Challenge	1	0	Viva Media	http://www.viva-media.com/
13	1001 Tangram Puzzles	1	0	Viva Media	http://www.viva-media.com/
14	123BCP	1		123BCP	http://www.123bcp.com/
15	1-2-3File All to PDF	4	1	1-2-3FileConvert	http://www.123fileconvert.com/
16	123PDF Converter	4	1	123 pdf converter	http://www.123pdfconverter.co
17	12Pay Payroll	1	4	12Pay Ltd	http://www.12pay.co.uk/
18	1ERP			Quidgest	http://www.quidgest.com/
19	1KEY Agile BI Suite	2	0	MAIA Intelligence Pvt. Ltd.	http://www.maia-intelligence.c
20	1KEY Agile Business Intelligence Suite	1	0	MAIA Intelligence Pvt. Ltd.	http://www.maia-intelligence.c
21	1st grade			Compedia Ltd.	http://www.compedia-usa.com/
22	20/20 vision	2009		20/20 vision Europe BV	http://www.2020vision.eu/
23	2002 Games	1	0	Viva Media	http://www.viva-media.com/
24	2002 Kakuro Puzzles	1	0	Viva Media	http://www.viva-media.com/
25	2002 Pentamino Puzzles	1	0	Viva Media	http://www.viva-media.com/
26	2002 Space Out Games	1	0	Viva Media	http://www.viva-media.com/
27	2002 Sudoku Games	1	0	Viva Media	http://www.viva-media.com/
28	20-20 Design	9	0	20-20 Technologies Inc	http://www.2020technologies.co
29	225,000 Clipart Images	3	2	Focus Multimedia Ltd.	http://www.focusmm.co.uk/
30	2sms Microsoft Outlook Solution	3		2sms	http://www.2sms.com/
31	2sms Microsoft SharePoint Solution	2		2sms	http://www.2sms.com/
32	2sms SMS Excel add-in	3		2sms	http://www.2sms.com/
33	30,000 Photos	3	1	Focus Multimedia Ltd.	http://www.focusmm.co.uk/
34	3003 Crystal Mazes	1	0	Viva Media	http://www.viva-media.com/

Standard Vs. Bonus Project Overview

- No Bonus Option – Capable of doing with CSE 271 + 274
  - Java – OO Program
  - Swing graphics or other Java library
  - Text file storage or simple database
  - Recommend development environment is Eclipse
- Optional Bonus Web-based (desktop & mobile application)
  - Option PHP+LAMP stack, Ruby on Rails, Other Web Technologies
    - Handles a lot of the database and object aspects for you
  - Other Option using Java + Web technologies
    - Bootstrap
    - JQuery
    - Other options
  - SQL backend

Code Reuse

- No Bonus Option – Capable of doing with CSE 271 + 274
  - Must be done from scratch other than using Java libraries and other provided Java frameworks (including JDBC, Swing, and others) only.
- Optional Bonus Web-based (desktop & mobile application)
  - If you are attempting the bonus modifier/aspects of the project, you are allowed to use existing web frameworks and technologies (Bootstrap, JQuery, or other GitHub solutions) as a starting point.
  - **However, you must be extremely diligent in citing your work, including an explanation of how you used and modified the existing web frameworks and technologies.**

Customer Meetings

- Each week, you'll be required to meet with the customer
  - TA (Joe) = Customer Representative/Person who hired you
  - Me = Customers' Supervisor
    - Email both of us your deliverables BEFORE each week's meeting
  - Initial meeting scheduled based on your surveys.
    - After that, feel free to reschedule with them!
- Aside from him checking off and viewing deliverables, he'll be your customer
  - Elicit additional requirements from him
  - Have him prioritize requirements
  - Ask him questions for clarifications of what he wants, like we taught in class. Requirements elicitation!

Customer Meetings (Requirements)

- You should come prepared to every customer meeting with a full agenda. Realistically, you should have your agenda prepared for 2 meetings at a time: *The immediate one and the one that follows*.
  - You are in charge of these meetings
  - You should have an agenda of things you're going to talk about, questions, etc.
  - Make sure you email your customer an agenda and all deliverables at least an hour before their meeting
- You will be graded on these things and your iteration deliverables/customer satisfaction
- Rubric next slide

Weekly Customer Meeting Grading Scheme

	Novice (Beginner/ Needs Improvement) 5-6/10	Apprentice (Making progress, but not yet at working level) 7-8/10	Proficient (Meets Expectations for course) 8-9/10	Excelling (Exceeds expectation) 10/10
Function in a team				
Distribution of work	1 Person is doing all the work	Work is not evenly distributed	Work is evenly distributed	Self-organizing
Delivery of product (Deliverables) (10 x 2 points)	Not working	Working software but missing major aspects	Working software	Managed backlog effectively, including measurement of scope/feature creep, velocity
Meeting management				
Plan	Nothing is planned beforehand. No email is sent.		Agenda is always created without prompting	Agendas are created and distributed early and 2 weeks at a time.
Organize			Logistics of meeting always handled without prompting	Excellent, professional, and smooth meeting flow as a result of organization.
Attend	Very few team members present with no explanation (beforehand) provided	Not all team members are present with no explanation (beforehand) provided	Team members are always present	Team members are present and all are on time
Conduct	Non-productive nor constructive.		Meeting is conducted smoothly, on time and to task	Student-driven meeting; driving and engaging

ITEMS FOR THIS WEEK-DUE FRIDAY

- Submit to Canvas by 11:59 proof/statements of the following
  - Define a project manager (primary communication with client, iteration planning, task assignment/division)
  - Define a technical manager (manage task board technology, architectural design/technology solutions)
  - Name your Application
  - Name your Software Development Company
  - Fill out scheduling google form
  - Gather at least 10 requirements by yourself (customer validation will come next week) based on document
  - Choose a task board (virtual? portable?) and populate it with requirements. Let us know which one/send screenshot

ITEMS FOR NEXT WEEK-DUE 1ST CUSTOMER MEETING

- Show your customer all the things mentioned in first week deliverables on Canvas
- Describe your roles to the customer
- Describe your team's meeting schedule. When and what frequency? How many times per week will you be meeting?
- Validate your 10 requirements with the customer. Let him change them.
- Show initial UML Class diagrams for your requirements
- BlueSky and elicit more/all requirements from your customer
- Begin and show evidence that you've put User Stories and Tasks for Iteration 1 on your board and assigned them to your specific team members.
- Show evidence that you have started setting up a code repository

End of Iteration Demos to Class

- To ensure you have a working product at the end of iteration, you will be doing a demo at the end of each iteration.
  - 10 minutes, plus or minus a minute (9-11 minutes)
  - Doesn’t need to be a “live” demo of product but better/more believable if it is.
  - Bad example of a live demo: http://www.kaltura.com/tiny/yb6re
- With the shift to "online", and in order to make your already stressful adjustment/lives easier, we have decided to allow you to asynchronously upload a video of your presentation by the due date.
  - The video must be shot in one take and group members must be visible while talking.
- Feel free to submit any (playable in windows and IOS) Video, or an external link to a video, for example a shared Google Drive link.

RECOMMENDATIONS

PEER FEEDBACK AT END

DIVIDE UP THE WORK

Assign and Define Roles 4/5 team members

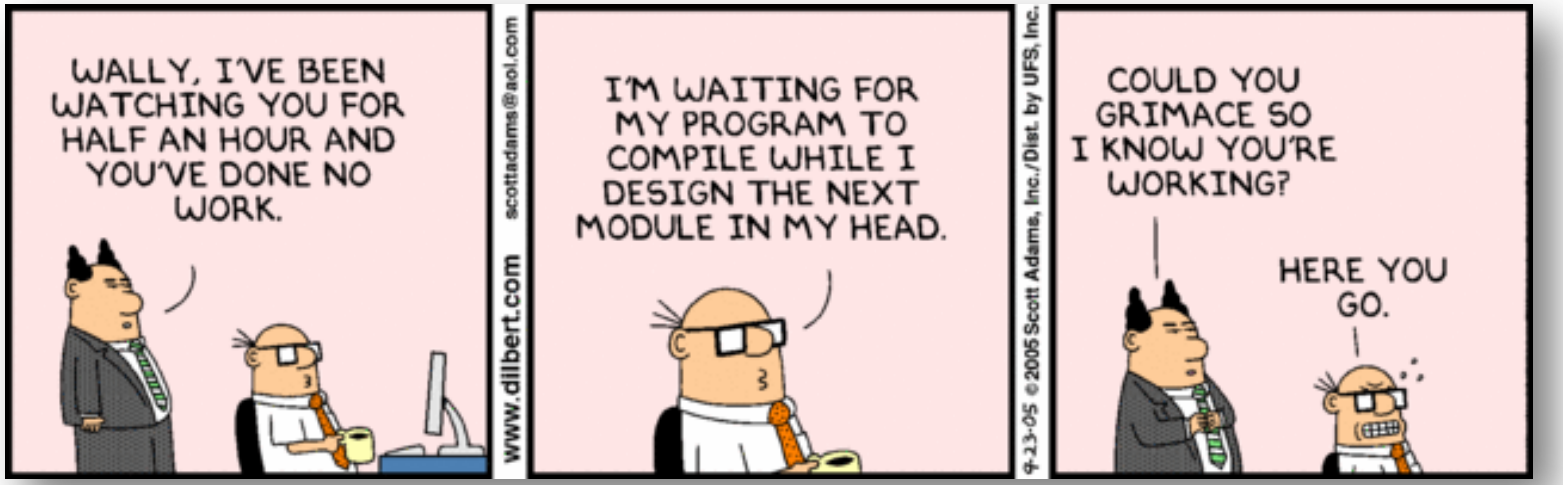
- A mix of
  - 1 Project Manager (defined in earlier slide)
  - 1 Technical Manager + tester (defined in earlier slide)
  - 1 Data Layer Person (UML models, SQL, schema) (less coding)
  - 2 developers
    - 1 should be 50% code, 50% documentation
    - 1 can be one of above roles

Possible Virtual Task Boards

- https://www.symphonical.com
- http://www.pivotaltracker.com
- https://trello.com
- https://zenhub.com

QUESTIONS FROM LAST CLASS?

- In the “real world” we’re taught that design is *pretty.*
- Here we learn that design is about *productivity*.



ARE CLASS DIAGRAMS IMPORTANT? JON WHITTLE'S SURVEY

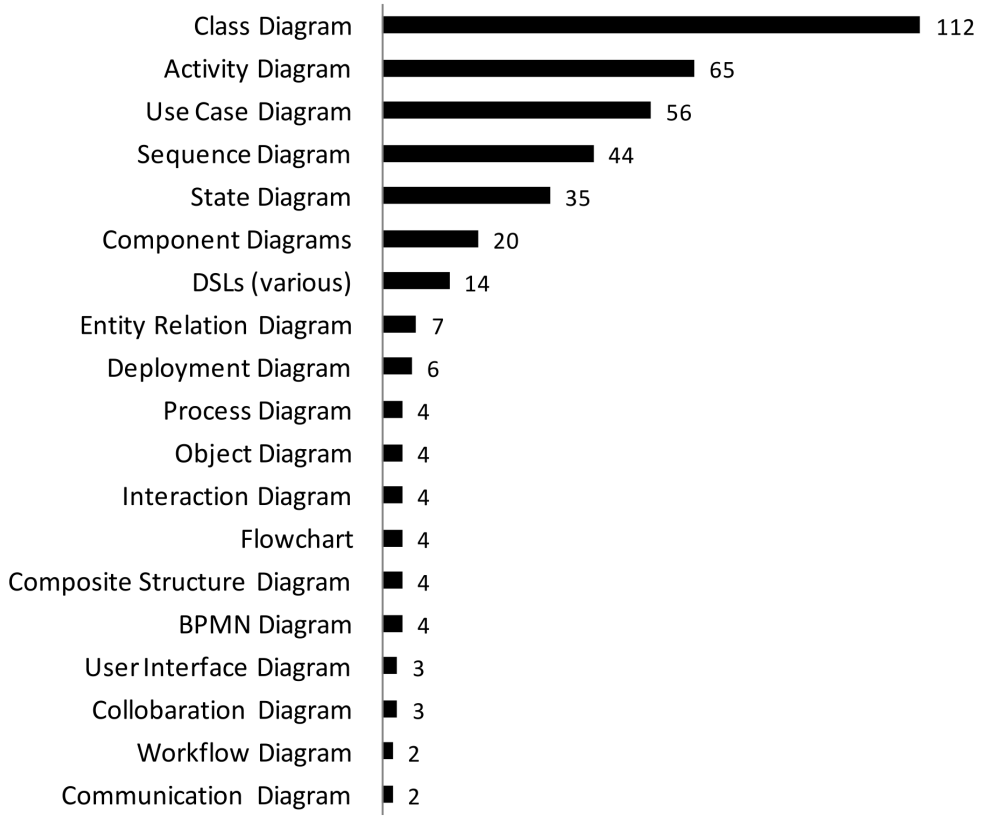


Fig. 3. Diagrams used regularly for modeling (# respondents).

WHICH MODELING LANGUAGE? JON WHITTLE'S SURVEY

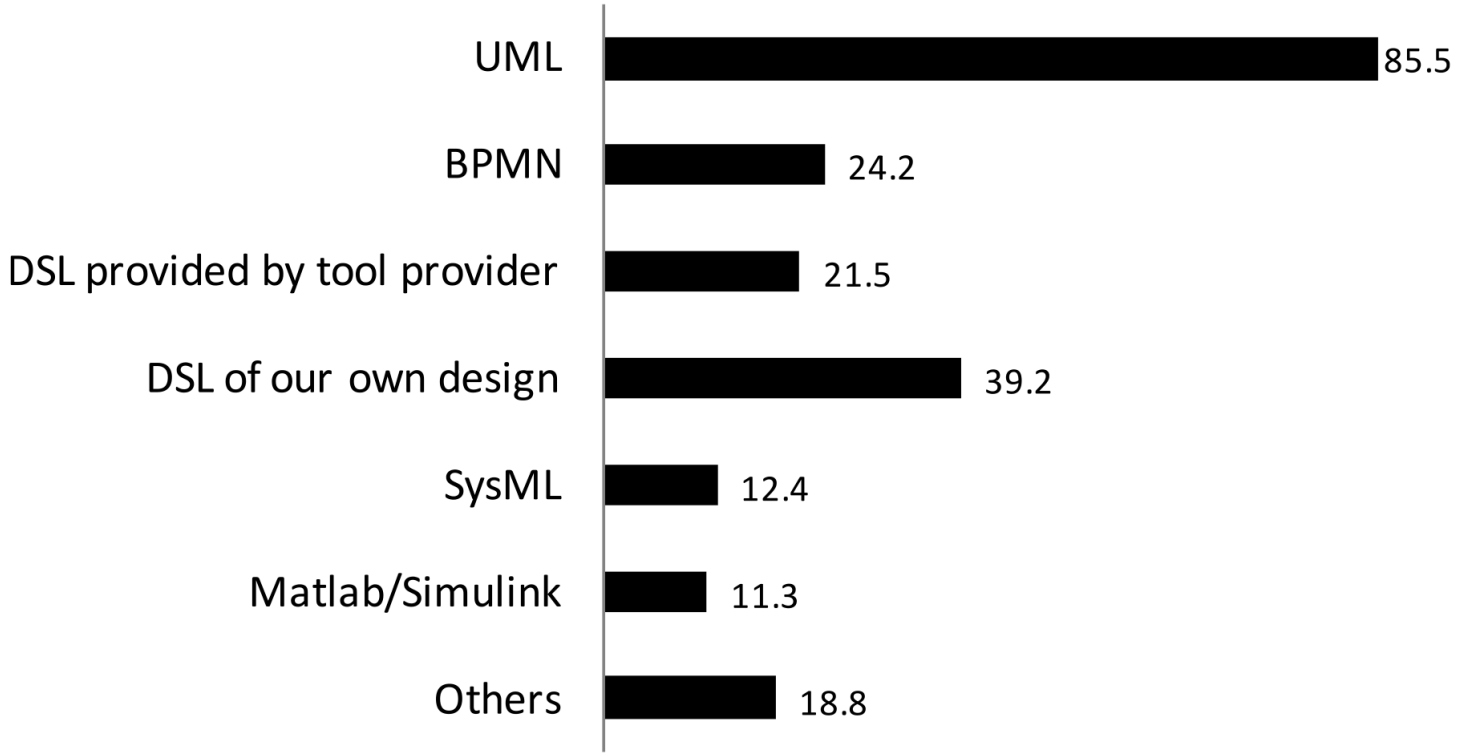
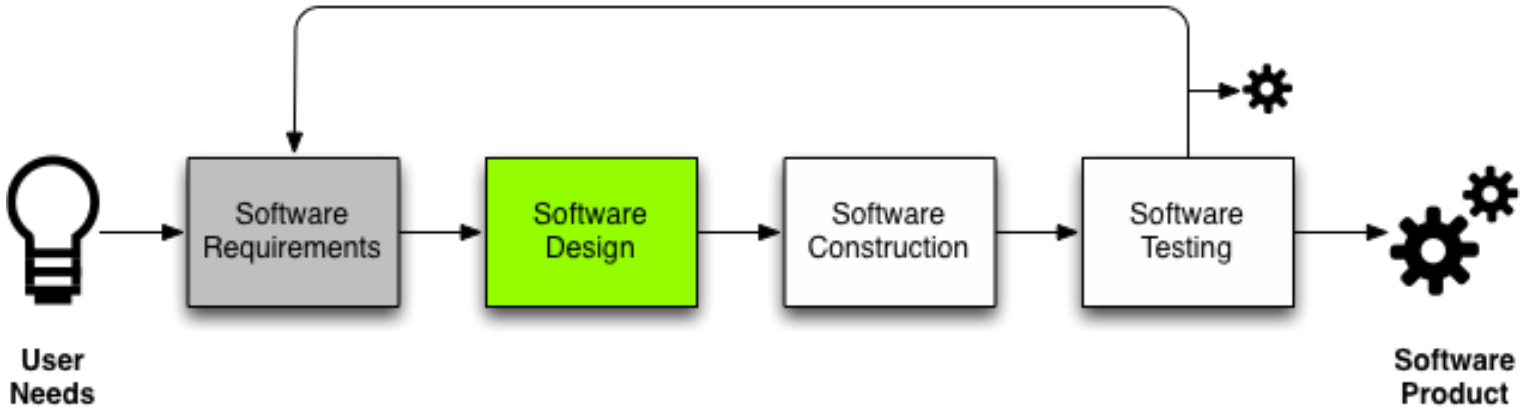


Fig. 2. Percentage of respondents using each modeling language.

Software Engineering Process

- Input
  - Requirements
- Output
  - Models and artifacts that record major design decisions



OBJECTS AND CLASSES

- ***Class*** – a *generalization* of a set of entities with common structure, behavior, and relationships to other classes. An abstract data type.
  - A person, an employee

- ***Object*** – an *instance* of a class. It has a state, value, and scope of existence
  - Joe Smith, Jane Doe

ASSOCIATION

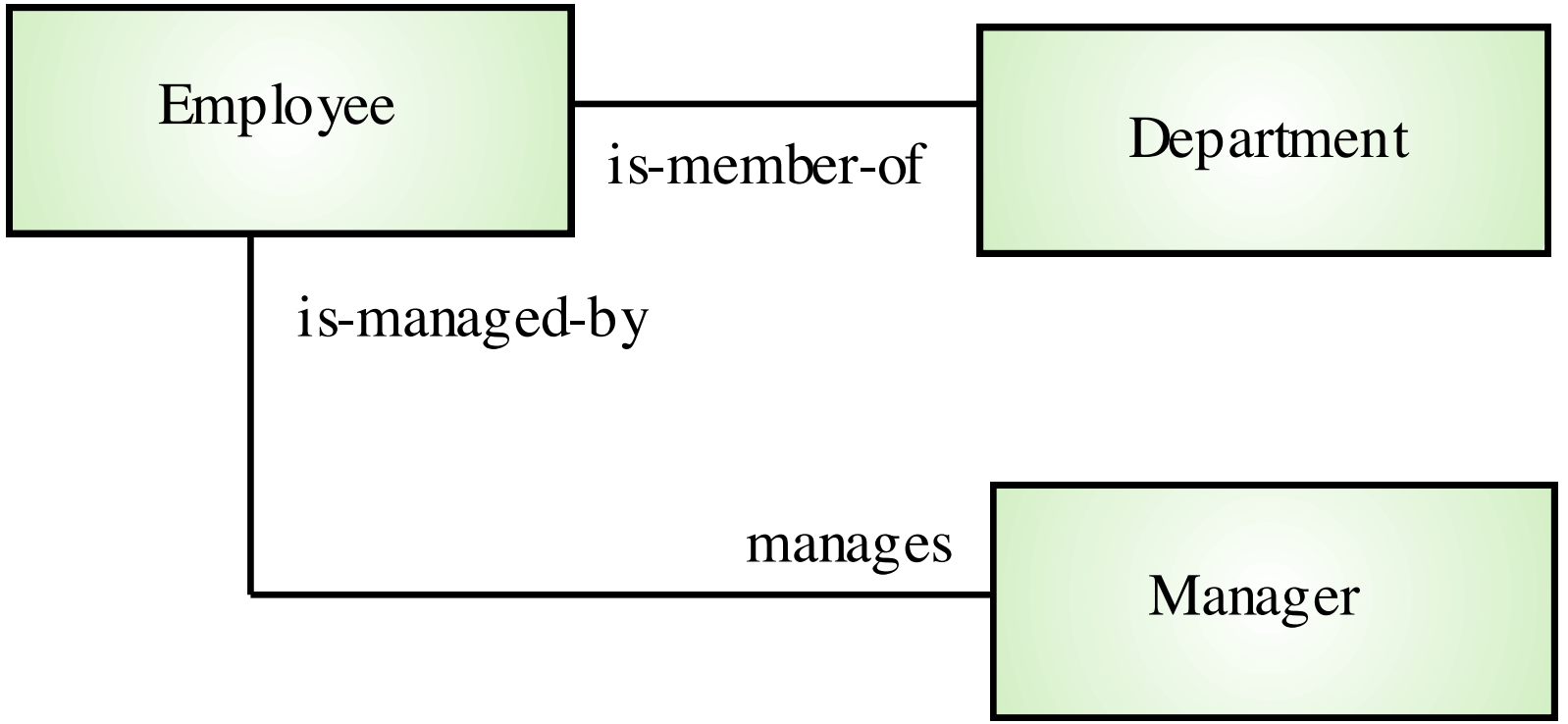
- Structural *relationship* between peer classes (or objects).
- Association can have
  - Name
  - Direction, or be bi-directional
  - *Role names* for each end of the association
  - *Multiplicity* of the relationship

ASSOCIATION CODE EXAMPLE

```
class Person {
public:
private:
  Company *employer;
};
```

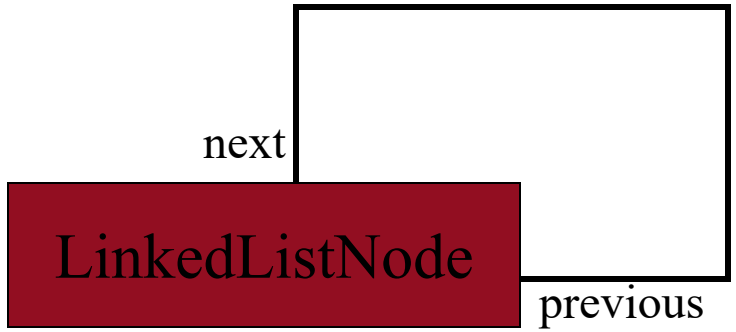
- Each instance of Person has a pointer to its employer

AN ASSOCIATION MODEL

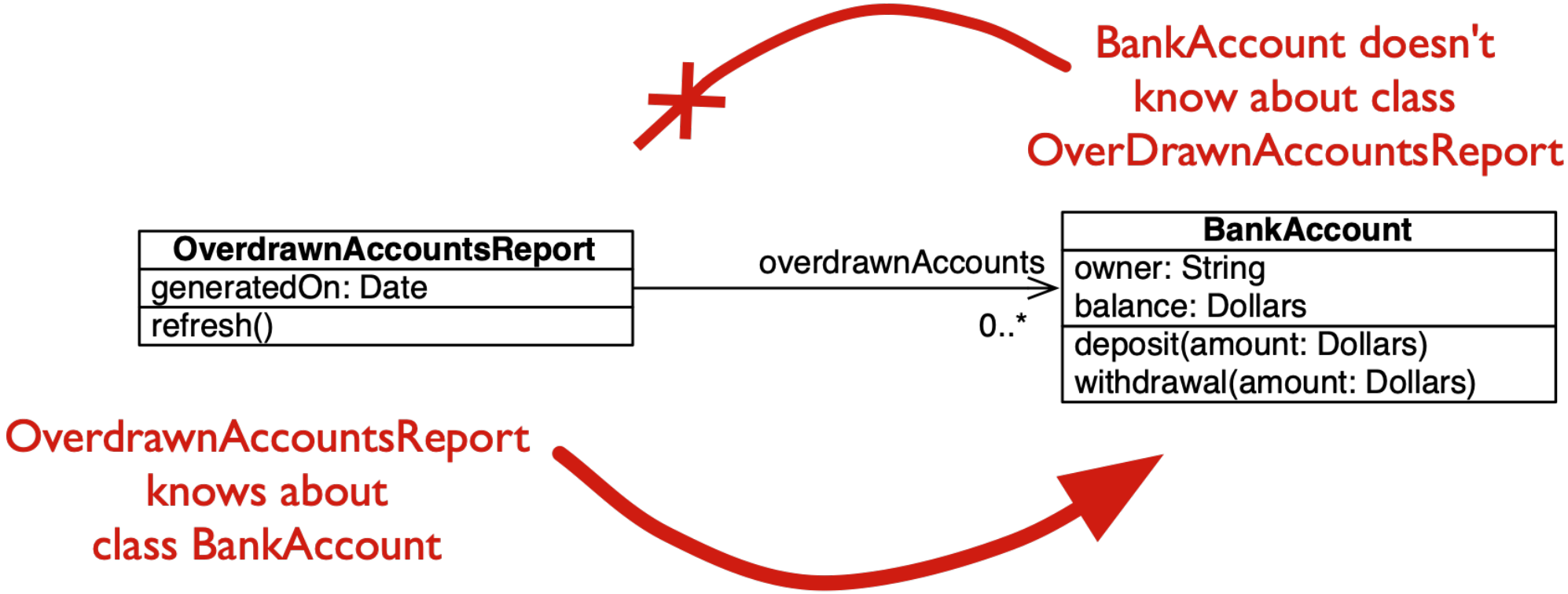


Association Relationships

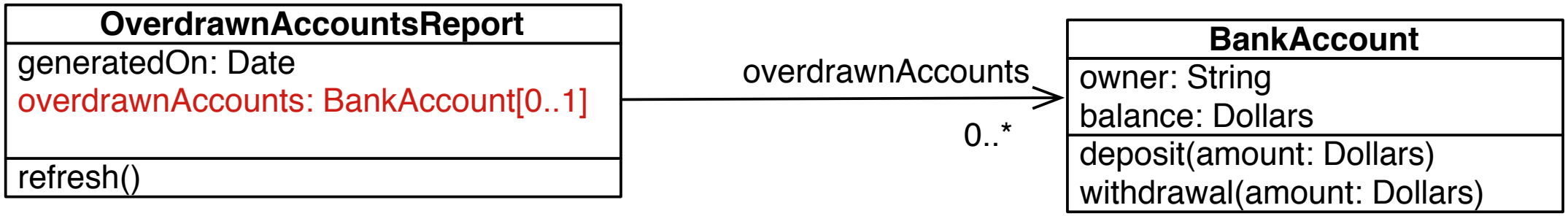
- A class can have a *self association*.



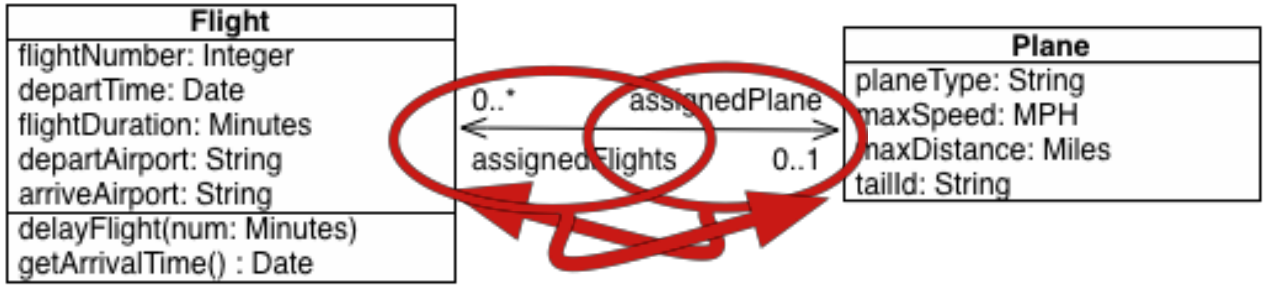
UNI-DIRECTIONAL ASSOCIATIONS



UNI-DIRECTIONAL ASSOCIATIONS



BIDIRECTIONAL ASSOCIATIONS



0 or more flights per plane

0 or 1 planes assigned to each flight

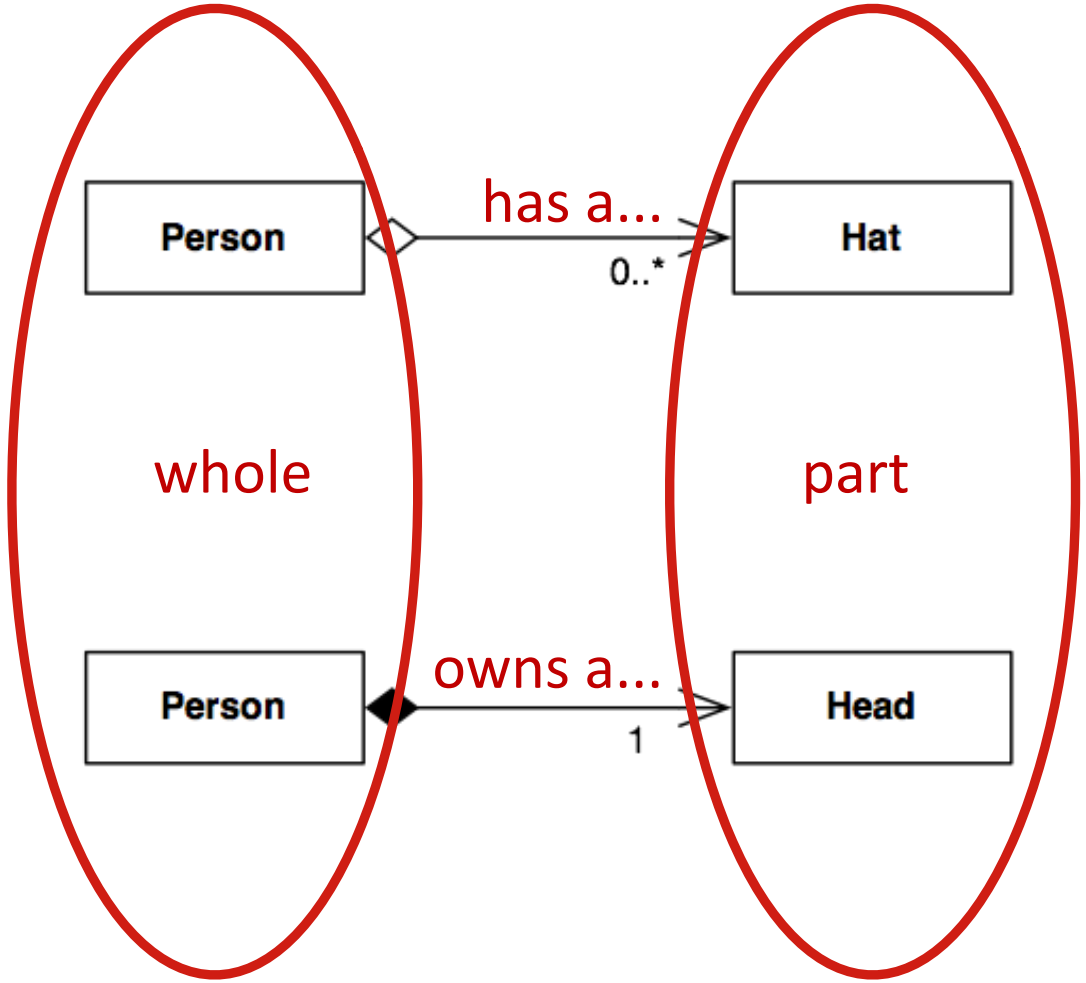
BIDIRECTIONAL ASSOCIATIONS

Bidirectional Associations use either line type

open arrow head

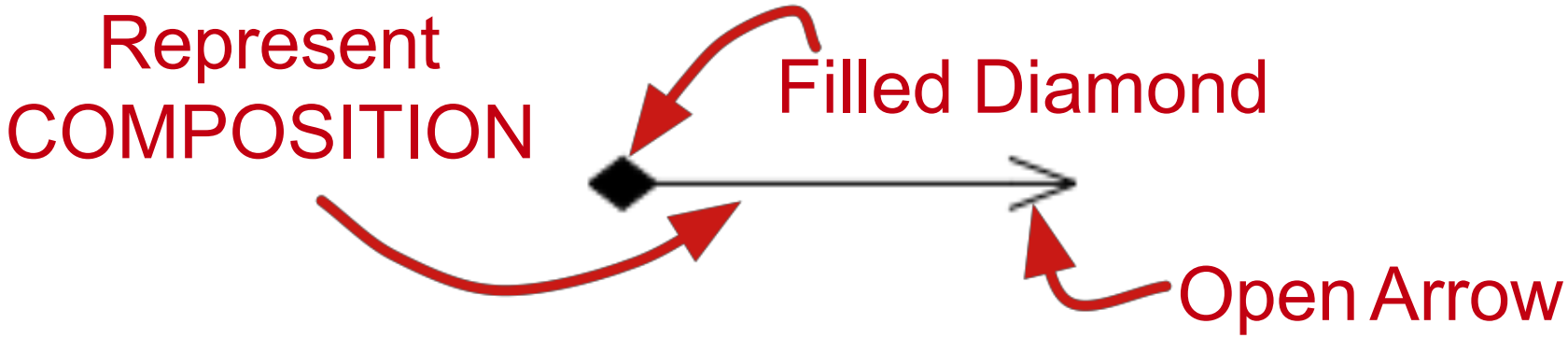
Aggregation & Composition Associations

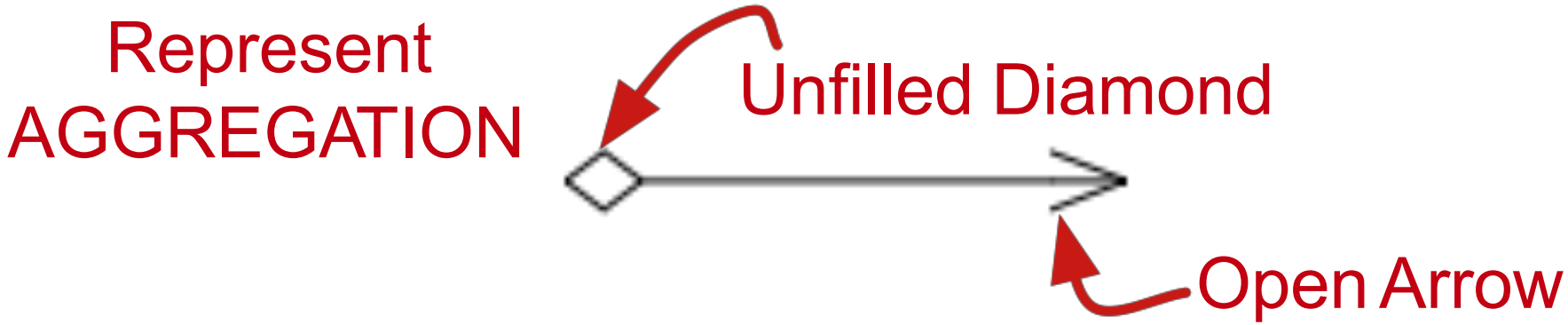
Aggregation



Composition

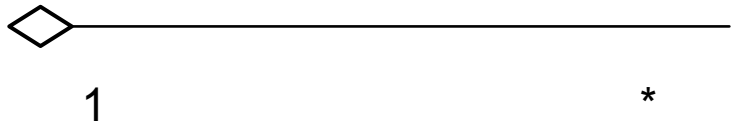
COMPOSITION VS. AGGREGATION



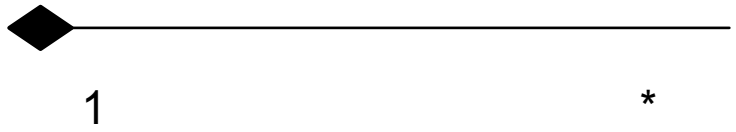


AGGREGATION VS. COMPOSITION

- Aggregation is a shared containment.
  No strong life time dependency.



- Composition is constrained aggregation that implies ownership. *The life time of the aggregates are determined by the object*.



ASSOCIATION VS. AGGREGATION – ARGUMENT AGAINST?

- The difference is one of implication. Aggregation denotes whole/part relationships whereas associations do not. However, there is not likely to be much difference in the way that the two relationships are implemented. That is, it would be very difficult to look at the code and determine whether a particular relationship ought to be aggregation or association. For this reason, it is pretty safe to ignore the aggregation relationship altogether.
- [Robert C. Martin | UML]

EXAMPLE

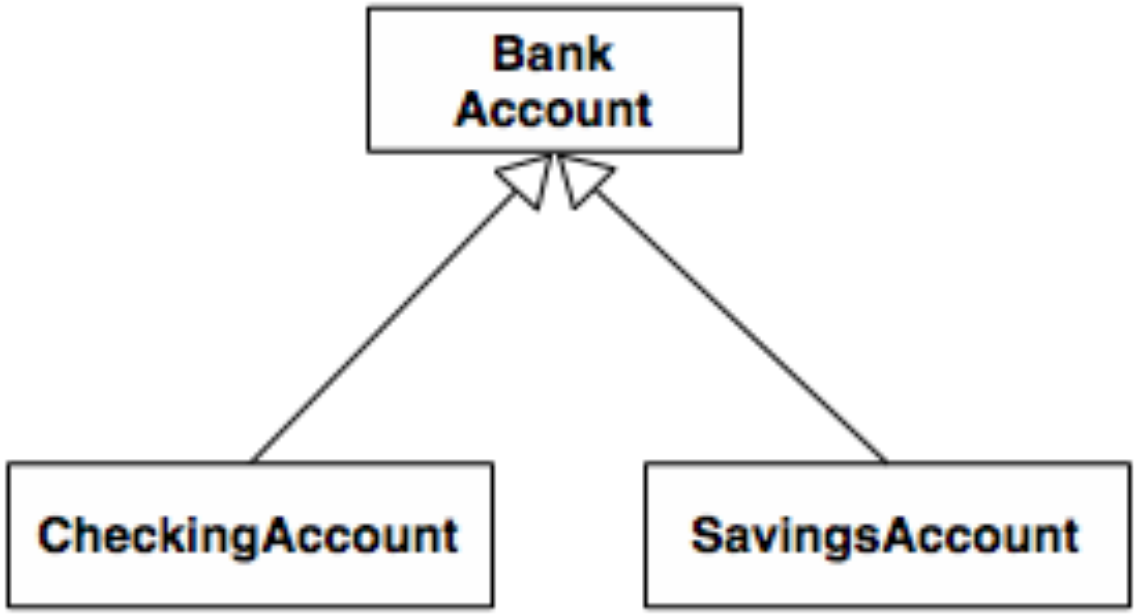
- The following is an example of:
  1. Interface
  2. Inheritance
  3. Extension
  4. Superclass

```
public class ExampleActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // The activity is being created.
    }
    @Override
    protected void onStart() {
        super.onStart();
        // The activity is about to become visible.
    }
    @Override
    protected void onResume() {
```

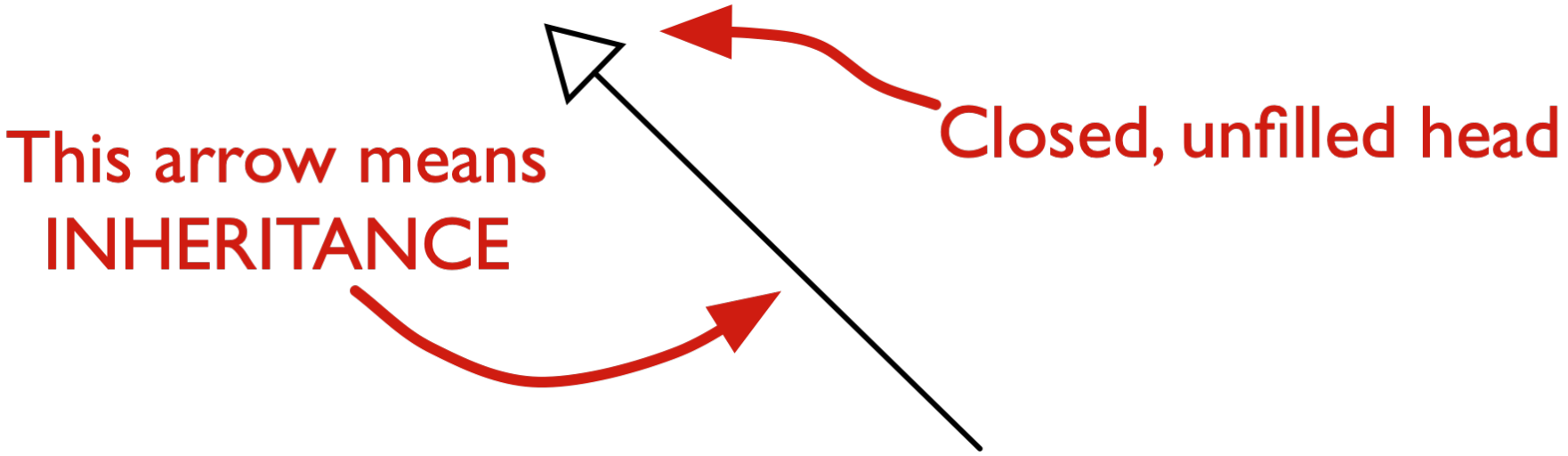
GENERALISATION AND INHERITANCE

- Objects are members of classes which define attribute types and operations
- Classes may be arranged in a class hierarchy where one class (a super-class) is a generalisation of one or more other classes (sub-classes)
- A sub-class inherits the attributes and operations from its super class and may add new methods or attributes of its own
- Generalisation in the UML is implemented as inheritance in OO programming languages

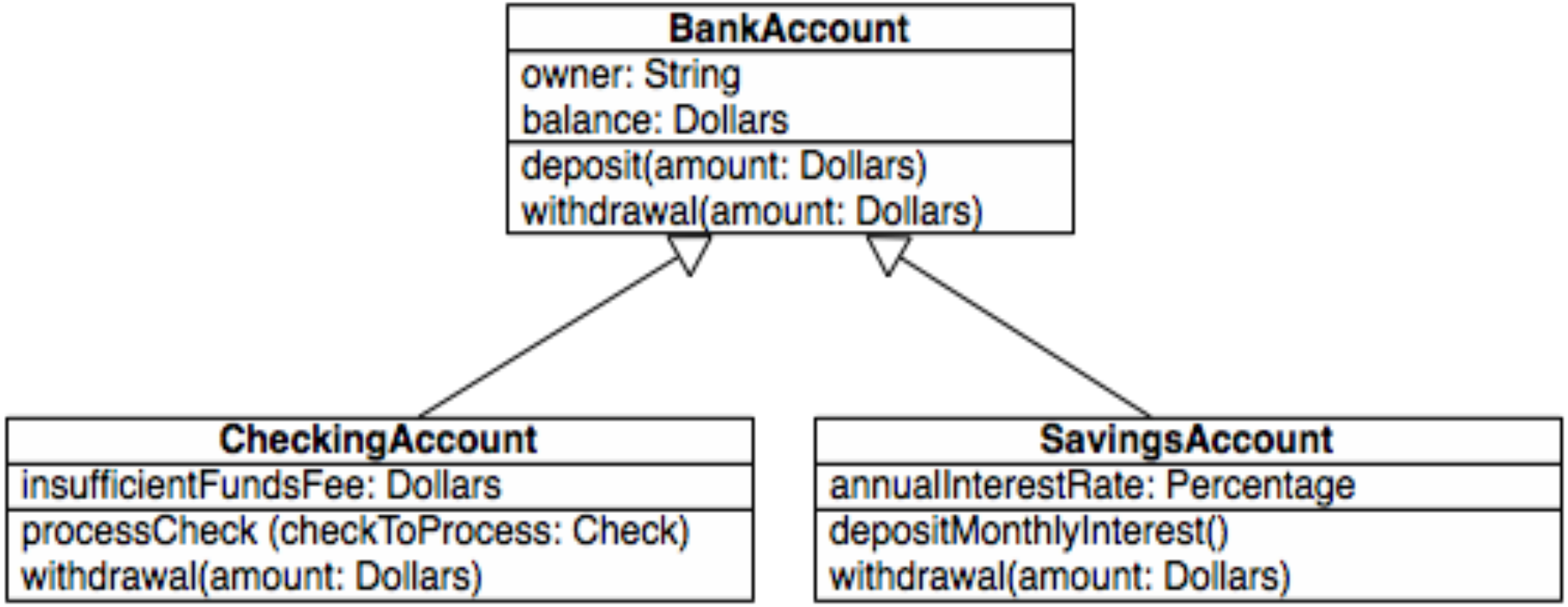
INHERITANCE



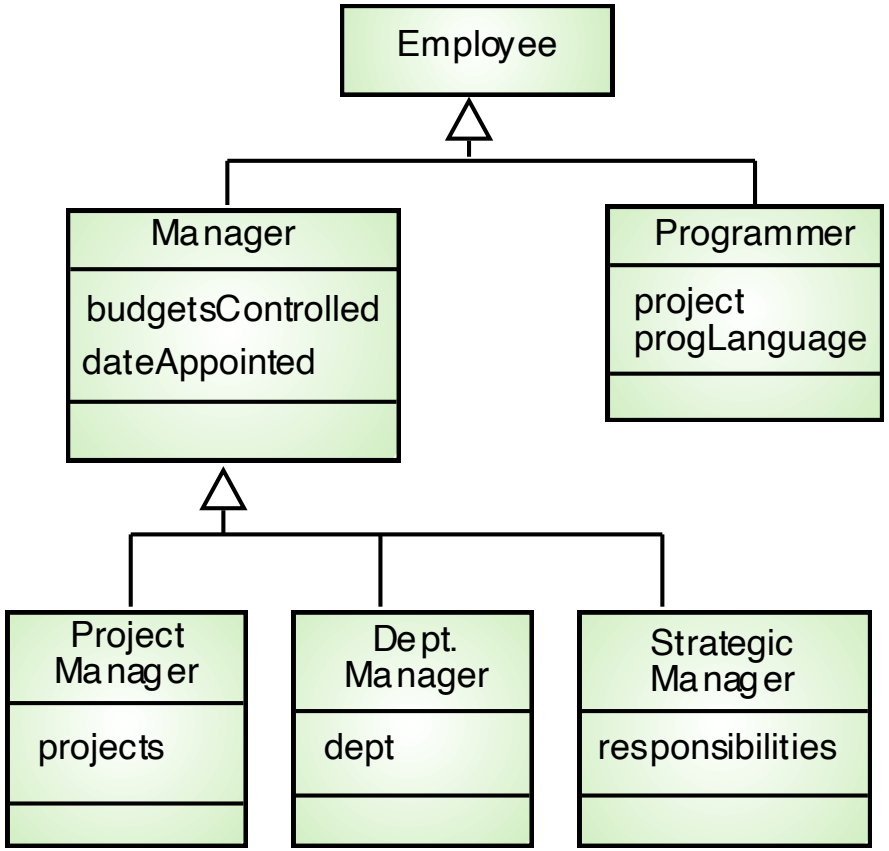
Inheritance Arrow



INHERITANCE



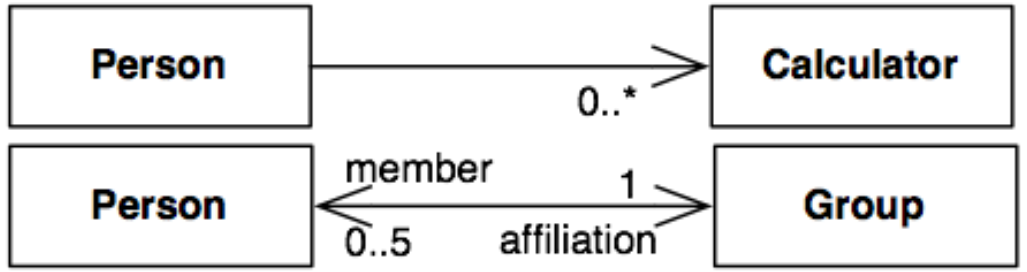
A GENERALISATION HIERARCHY



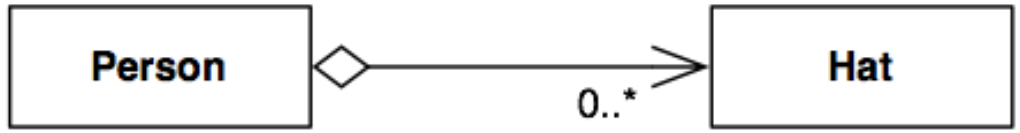
Basic Associations

Association

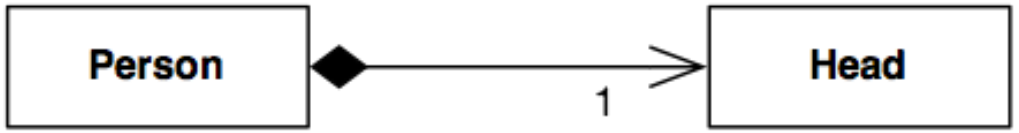
(Bi-Directional)



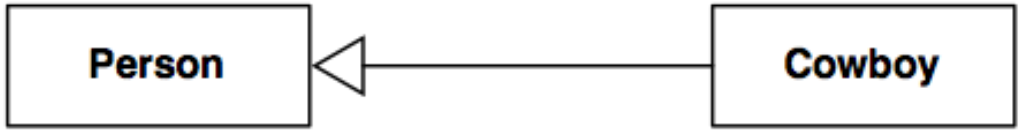
Aggregation



Composition



Inheritance

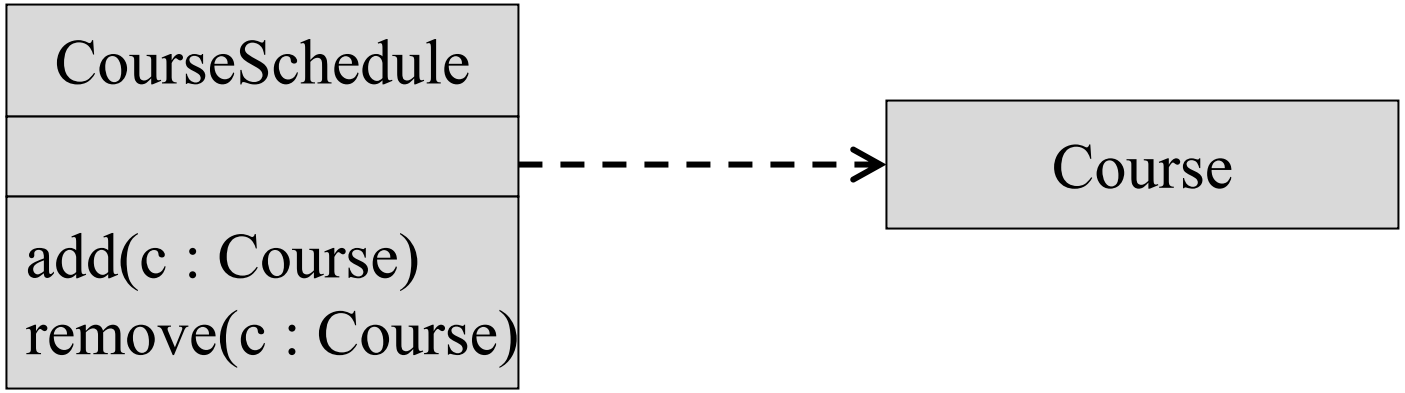


DEPENDENCY

- Represents *a using* relationship
- If a change in specification in one class effects another class, then there is a dependency
- A dependency indicates a semantic relationship between two or more elements.

EXAMPLE

- The dependency from CourseSchedule to Course exists because Course is used in both the add and remove operations of CourseSchedule.



Which Relation is Right?

- Aggregation – aka is-part-of, is-made-of, contains
- Use association when specific (persistent) objects have multiple relationships
  - e.g., there is only one Bill Gates at MS
- Use dependency when working with static objects, or if there is only one instance
- Do not confuse part-of with is-a

How To Start

- Look for nouns in your use case
  - Nouns are candidates for classes
  - Look for similar nouns, are they similar, inheritance?

- Look for verbs
  - Verbs may be good operations

DESIGN

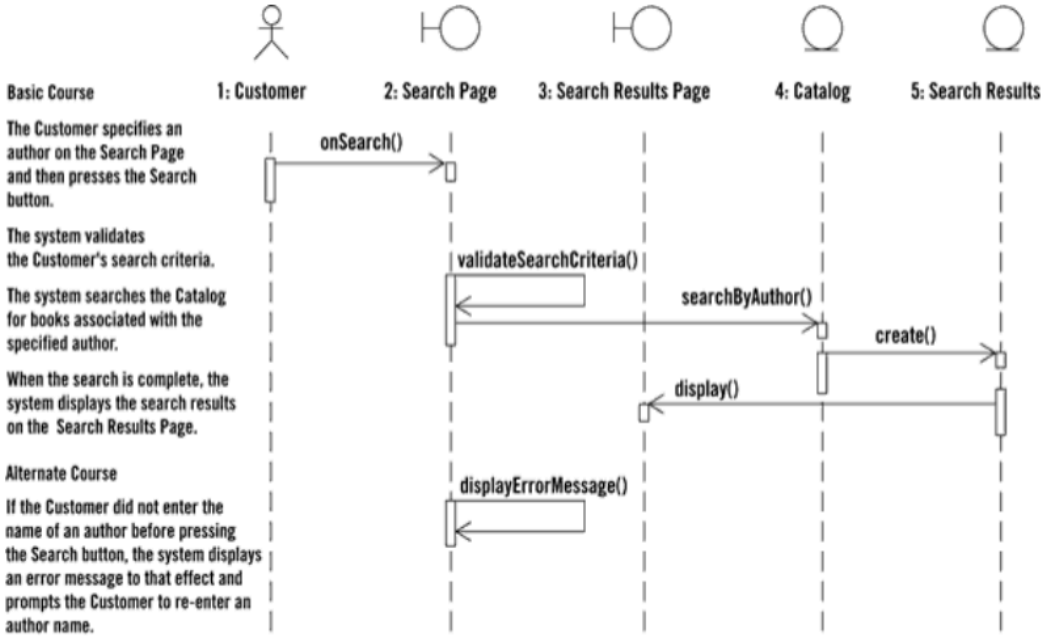
- Design as an art
  - Design is a creative activity
  - You get better at it as you go along
- Design as a science
  - Established practices and patterns
  - Commonly occurring problems have established solutions

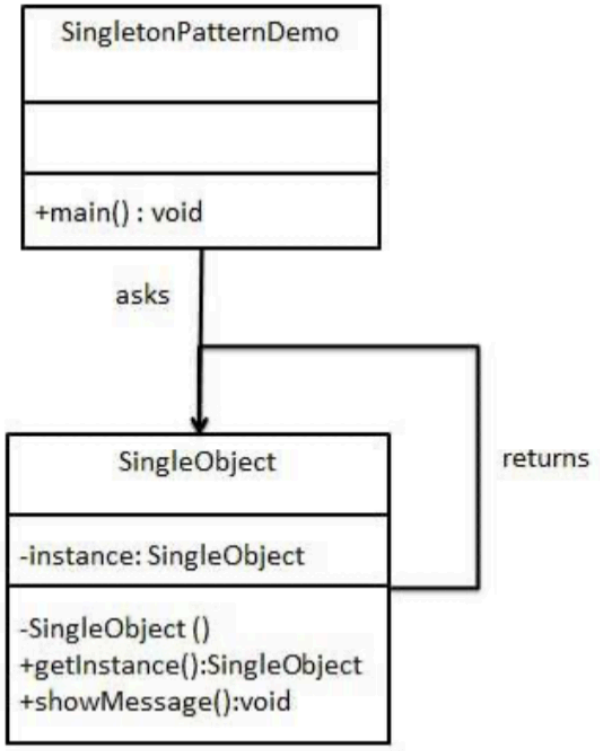
EXAMPLE

- This system provides the basic services to manage bank accounts at a bank called OxfordBank.
- OxfordBank has many branches, each of which has an address and branch number. A client opens accounts at a branch. Each account is uniquely identified by an account number; it has a balance and a credit or overdraft limit. There are many types of accounts, including: A mortgage account (which has a property as collateral), a checking account, and a credit card account (which has an expiry date and can have secondary cards attached to it). It is possible to have a joint account (e.g. for a husband and wife). Each type of account has a particular interest rate, a monthly fee and a specific set of privileges (e.g. ability to write checks, insurance for purchases etc. OxfordBank is divided into divisions and subdivisions (such as Planning, Investments and Consumer), the branches are considered subdivisions of the Consumer Division. Each division has a manager and a set of other employees. Each customer is assigned a particular employee as his or her 'personal banker'.

NEXT CLASS

- Sequence Diagrams and Design Patterns





RETROSPECTIVE QUESTIONS

- What are the 5 association types? Which have you used before?
- How would you realize the association types in Java (code)?
- Given requirements, how do you start modeling/what do you look for?
- We said design is both art and science. How do you feel about code?