Allen Vartanian

Dr. Johnson

CMSI 402-01

12 February 2018

Homework 2

5.1) What's the difference between a component-based architecture and a service-oriented architecture?

- In component-based software engineering, you regard the system as a collection of loosely coupled components that provide services for each other. A component-based architecture decouples the pieces of code and contains them within the same executable program, so they communicate directly instead of across a network.
- A component-based architecture is similar to a component-based architecture except the pieces are implemented as services. A service is a self-contained program that runs on its own and provides some kind of service for its clients.

5.2) Suppose you're building a phone application that lets you play tic-tac-toe against a simple computer opponent. It will display high scores stored on the phone, not in an external database. Which architectures would be most appropriate and why?

- A simple monolithic architecture would be appropriate in this case because the data is kept locally and does not need any complex storage or trigger systems to display the data.

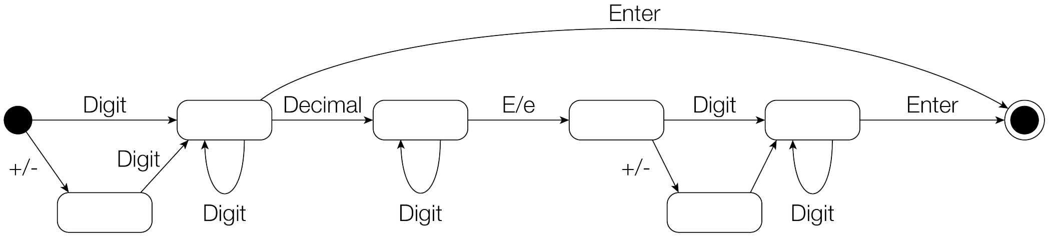
5.4) Repeat question 3 [after thinking about it; it repeats question 2 for a chess game] assuming the chess program lets two users play against each other over an Internet connection.

- Some form of a combination of a distributed, event-driven architecture would be appropriate because the players' turns switch on the event of a move and the fact theat they are on separate computers over a network requires a distributed system.

5.6) What kind of database structure and maintenance should the `ClassyDraw` application use?

- The application can have a simple monolithic architecture with an event driven user interface.

5.8) Draw a state machine diagram to let a program read floating point numbers in scientific notation as in +37 or -12.3e+17 (which means -12.3 x 10^{17}). Allow both E and e for the exponent symbol.

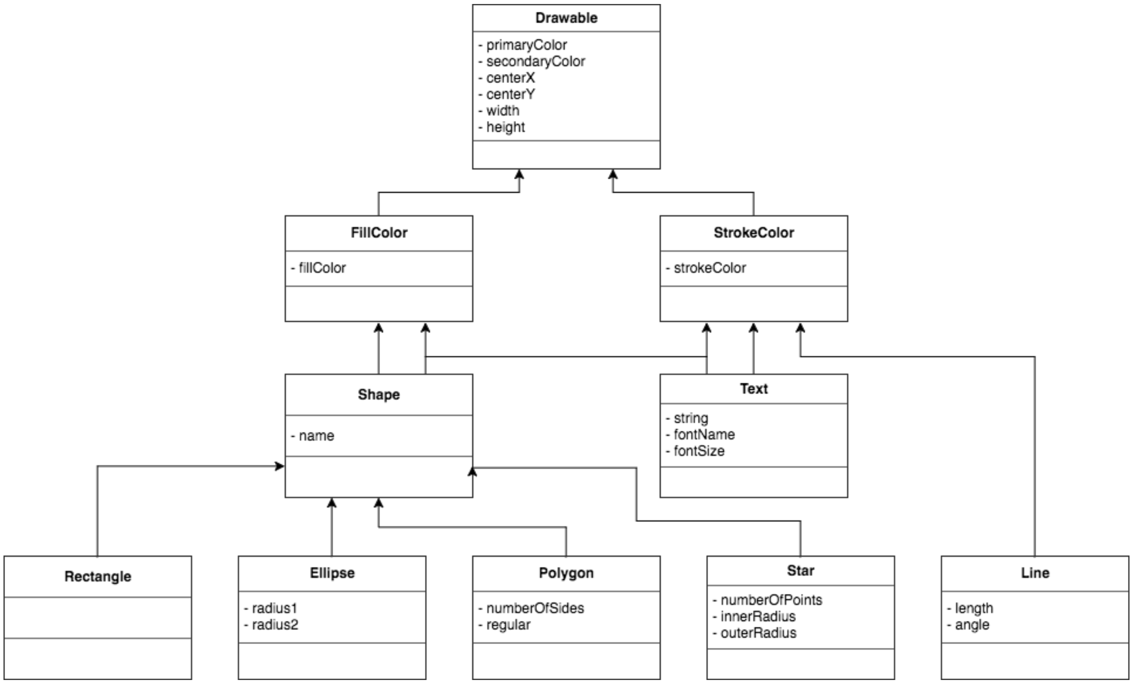


6.1) Consider the `ClassyDraw` classes `Line`, `Rectangle`, `Ellipse`, `Star`, and `Text`. What properties do these classes all share? What properties do they not share? Are there any properties shared by some classes and not others? Where should the shared and nonshared properties be implemented?

- They all let you click their object to select it, move the object to the top or bottom of the drawing order, move the object, and so forth.
- Some of the classes may have some measurement properties (e.g. length and width for `Rectangle` and `Line`) while others don't (e.g. Text).

- The shared properties should be implemented in a parent class (e.g. `Shapes` or `Objects`), and the non-shared properties should be implemented in the classes specific to those shapes and objects.

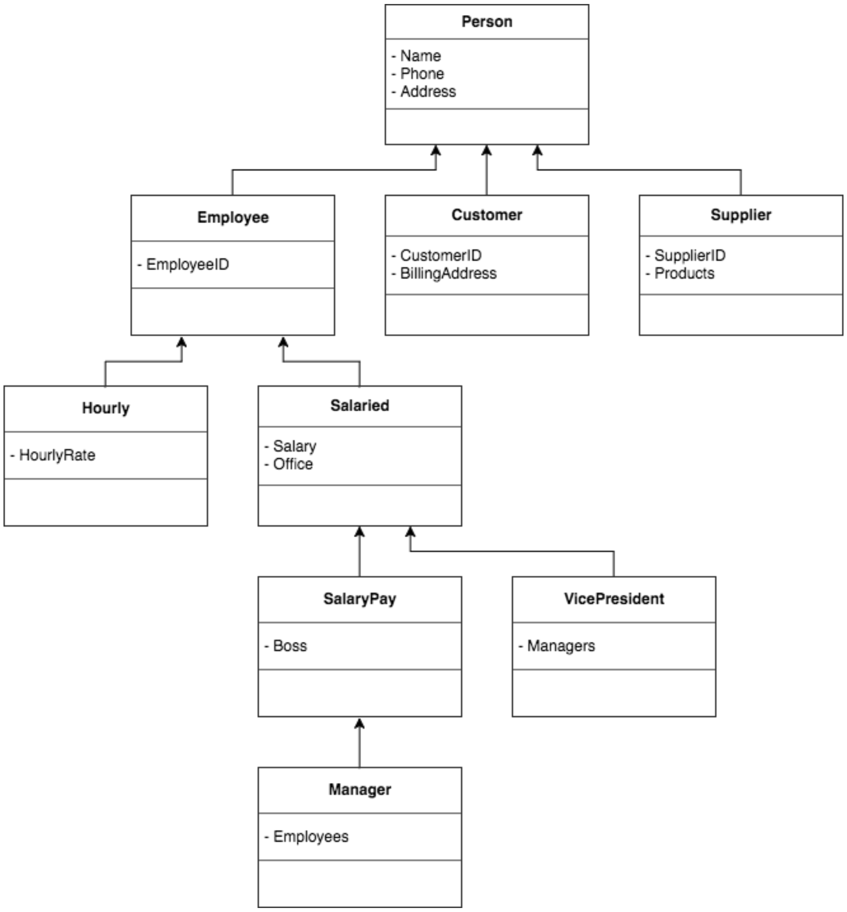
6.2) Draw an inheritance diagram showing the properties you identified for Exercise 1. (Create parent classes as needed, and don't forget the `Drawable` class at the top.)



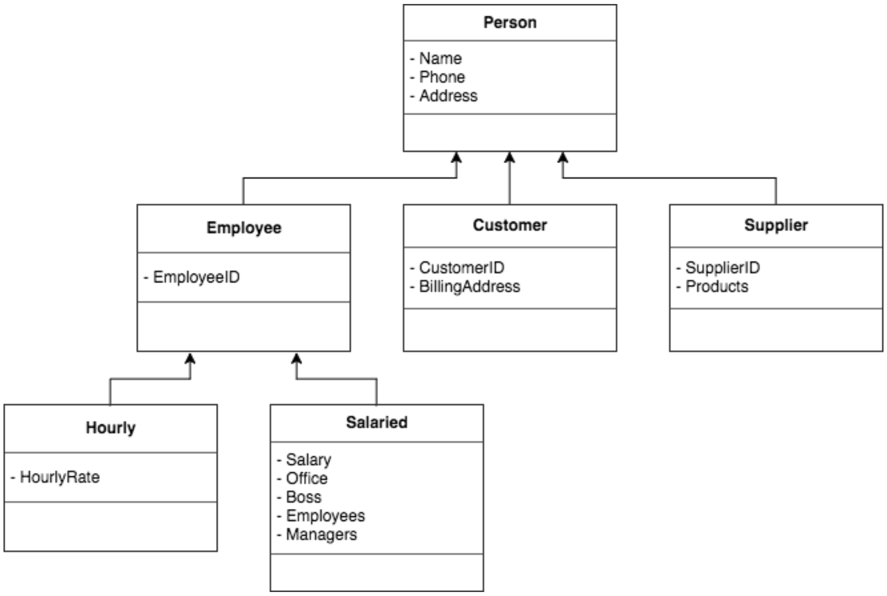
6.3) The following list gives the properties of several business-oriented classes.

- Customer — Name, Phone, Address, BillingAddress, CustomerID
- Hourly — Name, Phone, Address, EmployeeID, HourlyRate
- Manager — Name, Phone, Address, EmployeeID, Office, Salary, Boss, Employees
- Salaried — Name, Phone, Address, EmployeeID, Office, Salary, Boss
- Supplier — Name, Phone, Address, Products, SupplierID
- VicePresident — Name, Phone, Address, EmployeeID, Office, Salary, Managers

Assuming a Supplier is someone who supplies products for your business, draw an inheritance diagram showing the relationships among these classes. (Hint: Add extra classes if necessary.)



6.6) Suppose your company has many managerial types such as department manager, project manager, and division manager. You also have multiple levels of vice president, some of whom report to other manager types. How could you combine the `Salaried`, `Manager`, and `VicePresident` types you used in Exercise 3? Draw the new inheritance hierarchy.


Person
- Name
- Phone
- Address
Employee
- EmployeeID
Customer
- CustomerID
- BillingAddress
Supplier
- SupplierID
- Products
Hourly
- HourlyRate
Salaried
- Salary
- Office
- Boss
- Employees
- Managers