Problem 5.1, Stephens page 116

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

A component-based architecture tries to make each piece of the system as separate as possible so that different teams of developers can work on them separately. A service-oriented architecture is similar, except the pieces are implemented as services, self-contained programs that run on their own to provide some kind of service to their clients.

Problem 5.2, Stephens page 116

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 monolithic architecture would be great for a simple small application like this, where all components are client-side and simple to design.

Problem 5.4, Stephens page 116

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.

The chess program would be a bit more complex and since you’re using some parts of a server and a client working between each other, you would likely opt for a client/server architecture.

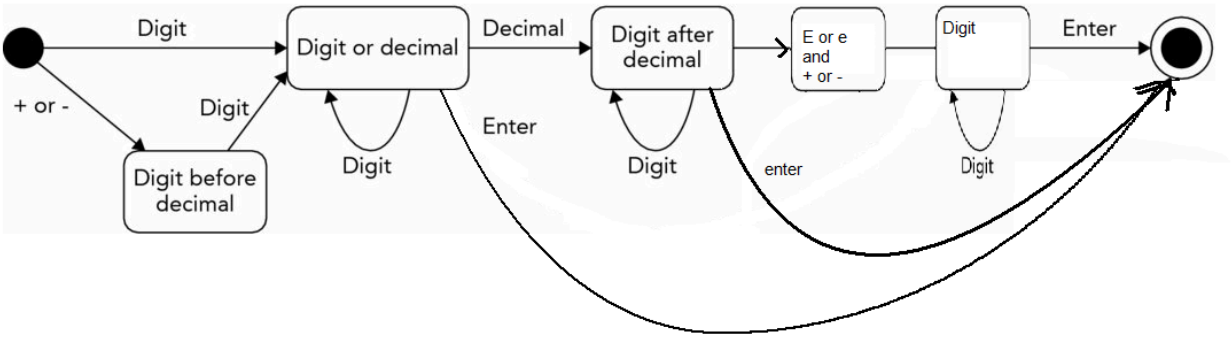
Problem 5.6, Stephens page 116

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

ClassyDraw has the user define the drawings, so there's no data around which to organize the program. Instead, the program would be a simple monolithic architecture with an event-driven UI.

Problem 5.8, Stephens page 116

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 1017). Allow both E and e for the exponent symbol. [Jeez, is this like Dr. Dorin's DFAs, or what???]



Problem 6.1, Stephens page 138

Consider the ClassyDraw classes Line, Rectangle, Ellipse, Star, and Text.

1. What properties do these classes all share?

   They all draw drawable objects to the screen, in a certain color, position, shape and size.

2. What properties do they NOT share?

   Only some of the classes draw using lines, text with fonts. They don't all share the same basic mechanism to draw.

3. Are there any properties shared by some classes and not others?

Yes. Lines, rectangles, and stars can all be composed of simple straight lines, so they all share being drawn using lines. Stars, ellipses, and rectangles all can have a fill color, but not lines or text.

4. Where should the shared and nonshared properties be implemented?

Shared properties like lines should be implemented as both an object and an exportable method with which to implement stars and rectangles, the line's thickness and is it's dashed, as well as the fill color of the rectangles, ellipses and stars; ellipses should handle curves by themselves, and text should have its own separate suite of methods for rendering and drawing text to the screen.

Problem 6.2, Stephens page 138

Draw an inheritance diagram showing the properties you identified for Exercise 6.1. [Create parent classes as needed, and don't forget the Drawable class at the top.]