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 can work on each piece separately. In component-based software engineering, you regard the system as a collection of loosely coupled components that provide services for each other.

Meanwhile, a service-oriented architecture is similar to a component-based architecture except the pieces are implemented as service.

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?

Rule Based, Event Driven and Component based architectures would be appropriate for this application. This is because these architectures do not require an external database to store the data (data-centric). I did not include Service oriented or distributed architecture because that would seem like overkill for such an application such as tic tac toe.

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.

For a chess game over an internet connection, I would say that service oriented architecture would make the most sense. SOA allows you to break the chess functionalities into individual services that communicate over a network. For example, you could have separate services for matchmaking, leaderboard management, and game state tracking.

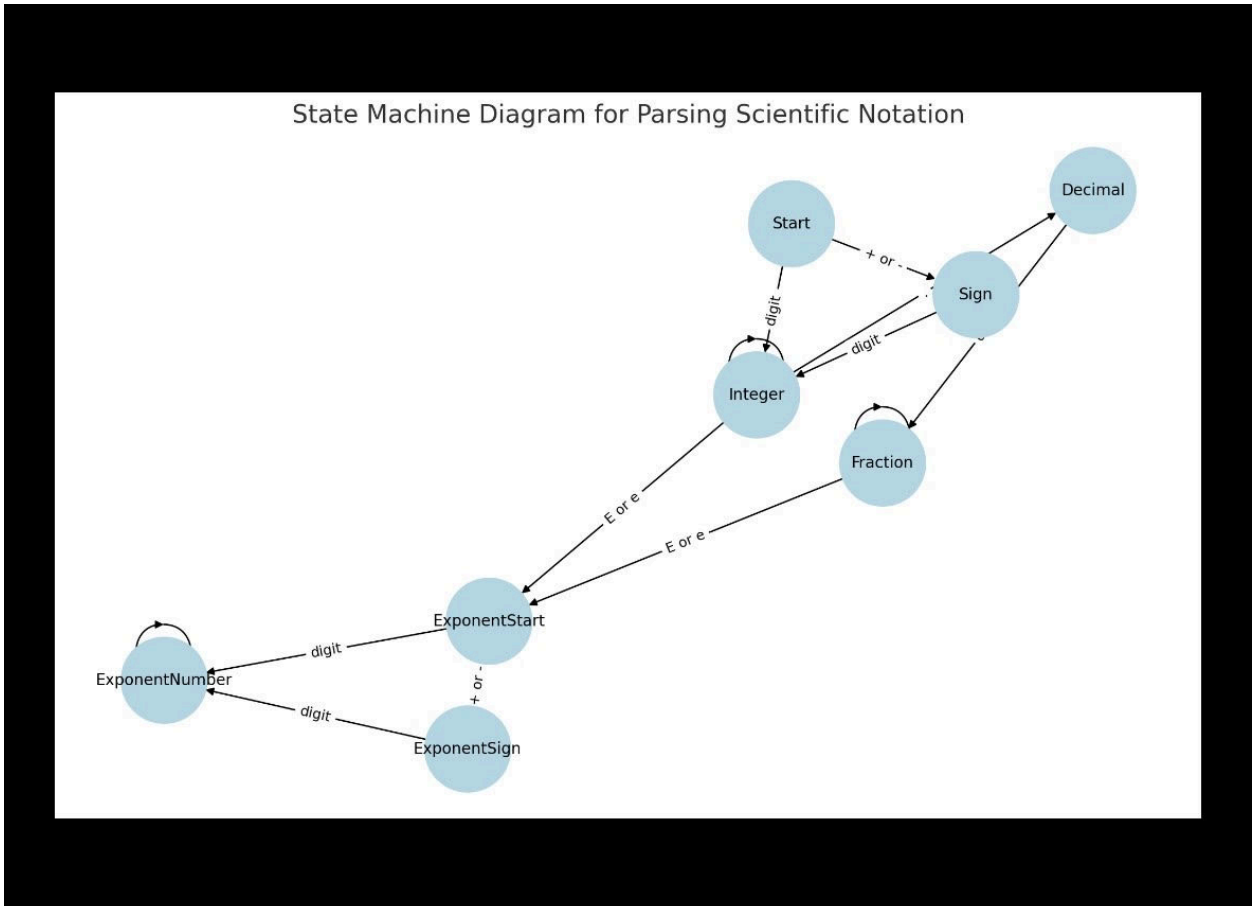
Problem 5.6, Stephens page 116

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

The component based architecture makes the most sense. This architecture is centered around dividing the app into independent components. For ClassyDraw, this involves breaking down components like the canvas, toolbar, brush settings, and layer manager. Since it seems that ClassyDraw requires internet connection for the ‘online help’ and ‘online tutorial’, having a service oriented architecture could also work. Though, If the focus is on modular design within the app, component based architecture leads to maintainable and scalable code.

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 10^{17}). 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?

   The common properties are : x, y (coordinates of the shape's position), a fill color, width, height.

2. What properties do they `NOT` share?

   The Text class will have a property for its text content. A Line may have a starting and ending point. Some shapes, like Rectangle, might need to support rotation.

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

The Rectangle and Ellipse might support the rotate property. They also might have rounded corners or some form of curvature (corner radius?).

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

   I imagine you can create a parent class called “Shape” which has all the common properties while creating child classes for each shape with its specific properties. The ‘shared by some’ can be repeated in another parent class but might as well just copy-paste that part of the code for each child for clarity’s sake (also because I’m a lazy programmer).

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.]