1. As engineers, we sometimes recommend technologies, methodologies, and approaches for solving problems because deep down we want to have these on our resume, not because they are the best solution for the problem. Such decisions very rarely result in happy outcomes.
2. In large-scale software, removing accidental complexity while retaining the solution to the essential complexity is challenging. How do you do this? Prefer frameworks derived from working code rather than ones cast down from ivory towers. Look at the percentage of code you have in a solution that directly addresses the business problem versus code that merely services the boundary between the application and the users. Cast a wary eye on vendor-driven solutions. They may not be inherently bad, but vendors often push accidental complexity.
3. Right now someone is failing to build a system as promised. Why? Was it because they chose Ruby over Java, or Python over Smalltalk? Or because they decided to use Postgres rather than Oracle? Most projects are built by people, and those people are the foundation for success and failure. So, it pays to think about what it takes to help make those people successful.