1 Chapter

1. Software architecture is often compared to the architecture of buildings as a conceptual analogy. What are the strong points of that analogy? What is the correspondence in buildings to software architecture structures and views? To patterns? What are the weaknesses of the analogy? When does it break down?
2. Do the architectures you've been exposed to document different structures and relations like those described in this chapter? If so, which ones? If not, why not?
3. Is there a different definition of software architecture that you are familiar with? If so, compare and contrast it with the definition given in this chapter. Many definitions include considerations like "rationale" (stating the reasons why the architecture is what it is) or how the architecture will evolve over time. Do you agree or disagree that these considerations should be part of the definition of software architecture?
4. Discuss how an architecture serves as a basis for analysis. What about decision-making? What kinds of decision-making does an architecture empower?
5. What is architecture 's role in project risk reduction?
6. Find a commonly accepted definition of system architecture and discuss what it has in common with software architecture. Do the same for enterprise architecture.
7. Find a published example of an architecture. What structure or structures are shown? Given its purpose, what structure or structures should have been shown? What analysis does the architecture support? Critique it: What questions do you have that the representation does not answer?
8. Sailing ships have architectures, which means they have "structures" that lend themselves to reasoning about the ship ' s performance and other quality attributes. Look up the technical definitions for barque, brig, cutter, frigate, ketch, schooner, and sloop. Propose a useful set of "structures" for distinguishing and reasoning about ship architectures.

2 Chapter

1. Suppose you want to introduce architecture-centric practices to your organization. Your management is open to the idea, but wants to know the ROI for doing so. How would you respond?
2. Prioritize the list of thirteen points in this chapter according to some criteria meaningful to you. Justify your answer. Or, if you could choose only two or three of the reasons to promote the use of architecture in a project, which would you choose and why?

3 Chapter

1. What kinds of business goals have driven the construction of the following:
    a. The World Wide Web
    b. Amazon's EC2 cloud infrastructure
    c. Google's Android platform
2. What mechanisms are available to improve your skills and knowledge? What skills are you lacking?

3. Describe a system you are familiar with and place it into the AIC. Specifically, identify the forward and reverse influences on contextual factors.

4 Chapter

1 What is the relationship between a use case and a quality attribute scenario? If you wanted to add quality attribute information to a use case, how would you do it?
2. Do you suppose that the set of tactics for a quality attribute is finite or infinite? Why?
3. Discuss the choice of programming language (an example of choice of technology) and its relation to architecture in general, and the design decisions in the other six categories? For instance, how can certain programming languages enable or inhibit the choice of particular coordination models?

5 Chapter

1. Write a set of concrete scenarios for availability using each of the possible responses in the
general scenario.
2. Write a concrete availability scenario for the software for a (hypothetical) pilotless passenger
aircraft.
3. Write a concrete availability scenario for a program like Microsoft Word.
4. Redundancy is often cited as a key strategy for achieving high availability. Look at the tactics presented in this chapter and decide how many of them exploit some form of redundancy and how many do not.
5. How does availability trade off against modifiability? How would you make a change to a system that is required to have "24/7" availability (no scheduled or unscheduled downtime, ever)?
6. Create a fault tree for an automatic teller machine. Include faults dealing with hardware component failure, communications failure, software failure, running out of supplies, user errors,
and security attacks. How would you modify your automatic teller machine design to accommodate these faults?
7. Consider the fault detection tactics (ping/echo, heartbeat, system monitor, voting, and exception detection). What are the performance implications of using these tactics?

6 Chapter

1. Find a web service mashup. Write several concrete interoperability scenarios for this system.
2. What is the relationship between interoperability and the other quality attributes highlighted in this book? For example, if two systems fail to exchange information properly, could a security
flaw result? What other quality attributes seem strongly related (at least potentially) to interoperability?

3. Is a service-oriented system a system of systems? If so, describe a service-oriented system that is directed, one that is acknowledged, one that is collaborative, and one that is virtual.
4. Universal Description, Discovery, and Integration (UDDI) was touted as a discovery service, but
commercial support for UDDI is being withdrawn. Why do you suppose this is? Does it have anything to do with the quality attributes delivered or not delivered by UDDI solutions?
5. Why has the importance of orchestration grown in recent years?
6. If you are a technology producer, what are the advantages and disadvantages of adhering to
interoperability standards? Why would a producer not adhere to a standard?
7. With what other systems will an automatic teller machine need to interoperate? How would you change your automatic teller system design to accommodate these other systems?

7 Chapter

1. In a certain metropolitan subway system, the ticket machines accept cash but do not give change. There is a separate machine that dispenses change but does not sell tickets. In an average station there are six or eight ticket machines for every change machine. What modifiability tactics do you see at work in this arrangement? What can you say about availability?
2. For the subway system in the previous question, describe the specific form of modifiability (using a modifiability scenario) that seems to be the aim of arranging the ticket and change machines as described.
3. A wrapper is a common aid to modifiability. A wrapper for a component is the only element
allowed to use that component; every other piece of software uses the component's services by
going through the wrapper. The wrapper transforms the data or control information for the component it wraps. For example, a component may expect input using English measures but find itself in a system in which all of the other components produce metric measures. A wrapper
could be employed to translate. What modifiability tactics does a wrapper embody?
4. Once an intermediary has been introduced into an architecture, some modules may attempt to circumvent it, either inadvertently (because they are not aware of the intermediary) or intentionally (for performance, for convenience, or out of habit). Discuss some architectural means to prevent inadvertent circumvention of an intermediary.