About Part II

Part II of this textbook continues a main theme of Part I—enforcing modularity—by introducing still stronger forms of modularity. Part I introduces the client/service model and virtualization, both of which help prevent accidental errors in one module from propagating to another. Part II introduces stronger forms of modularity that can help protect against component and system failures, as well as malicious attacks. Part II explores communication networks, constructing reliable systems from unreliable components, creating all-or-nothing and before-or-after transactions, and implementing security. In doing so, Part II also introduces additional design principles to guide a designer who needs to build computer systems that have stronger modularity. Following is a brief summary of the topics of those [on-line] chapters and supplementary materials. In addition, the Table of Contents for Part I lists the sections of the Part II chapters.

Chapter 7 [on-line]: ***Networks***. By running clients and services on different computers that are connected by a network, one can build computer systems that exploit geographic separation to tolerate failures and construct systems that can enable information sharing across geographic distances. This chapter approaches the network as a case study of a system and digs deeply into how networks are organized internally and how they work. After a discussion that offers insight into why networks are built the way they are, it introduces a three-layer model, followed by a major section on each layer. A discussion of congestion control helps bring together the complete picture of interaction among the layers. The chapter ends with a short collection of war stories about network design flaws.

Chapter 8 [on-line]: ***Fault tolerance***. This chapter introduces the basic techniques to build computer systems that, despite component failures, continue to provide service. It offers a systematic development of design principles and techniques for creating reliable systems from unreliable components, based on modularity and on generalization of some of the techniques used in the design of networks. The chapter ends with a case study of fault tolerance in memory systems and a set of war stories about fault tolerant systems that failed to be fault tolerant. This chapter is an unusual feature for an introductory text—this material, if it appears at all in a curriculum, is usually left to graduate elective courses—yet some degree of fault tolerance is a requirement for almost all computer systems.

Chapter 9 [on-line]: ***Atomicity***. This chapter deals with the problem of making flawless updates to data in the presence of concurrent threads and despite system failures. It expands on concepts introduced in Chapter 5, taking a cross-cutting approach to atomicity—making actions atomic with respect to failures and also with respect to concurrent actions—that recognizes that atomicity is a form of modularity that plays a fundamental role in operating systems, database management, and processor design. The chapter begins by laying the groundwork for intuition about how a designer achieves atomicity, and then it introduces an easy-to-understand atomicity scheme.

This basis sets the stage for straightforward explanations of instruction renaming, transactional memory, logs, and two-phase locking. Once an intuition is established about how to systematically achieve atomicity, the chapter goes on to show how database systems use logs to create all-or-nothing actions and automatic lock management to ensure before-or-after atomicity of concurrent actions. Finally, the chapter explores methods of obtaining agreement among geographically separated workers about whether or not to commit an atomic action. The chapter ends with case studies of atomicity in processor design and management of disk storage.

Chapter 10 [on-line]: ***Consistency***. This chapter discusses a variety of requirements that show up when data is replicated for performance, availability, or durability: cache coherence, replica management for extended durability, and reconciliation of usually disconnected databases (e.g., "hotsync" of a personal digital assistant or cell phone with a desktop computer). The chapter introduces the reader to the requirements and the basic mechanisms used to meet those requirements. Sometimes these topics are identified with the label "distributed systems".

Chapter 11 [on-line]: ***Security***. Earlier chapters gradually introduced more powerful and far-reaching methods of enforcing modularity. This chapter cranks up the enforcement level to maximum strength by introducing the techniques of ensuring that modularity is enforced even in the face of adversaries who behave malevolently. It starts with design principles and a security model, and it then applies that model both to enforcement of internal modular boundaries (traditionally called "protection") and to network security. An advanced topics section explains cryptographic techniques, which are the basis for most network security. A case study of the Secure Socket Layer (SSL) protocol and a set of war stories of protection system failures illustrate the range and subtlety of considerations involved in achieving security.

Suggestions for further reading. The suggested reading list in Part II is, apart from updates, the same as the one in this book.

Problem sets. The Part II collection of problem sets includes both the Part I problem sets and many additional problem sets for the Part II chapters.

Glossary. The on-line Glossary is identical to the one in this book. In addition to its primary purpose of supporting this textbook, the on-line Glossary also can serve as a reference source that workers in other specialties may find useful in coordinating their terminology with that of the field of systems.

Comprehensive Index. The on-line Index of Concepts provides page numbers for both Part I and Part II in a single alphabetic list.