

Distributed Systems

ALI KAMANDI, PH.D.

SCHOOL OF ENGINEERING SCIENCE

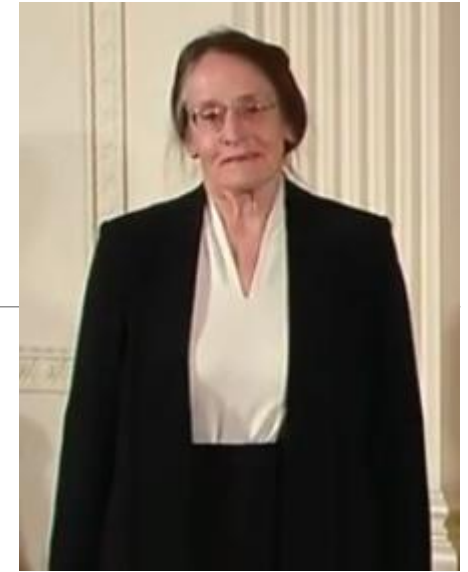
COLLEGE OF ENGINEERING

UNIVERSITY OF TEHRAN

KAMANDI@UT.AC.IR

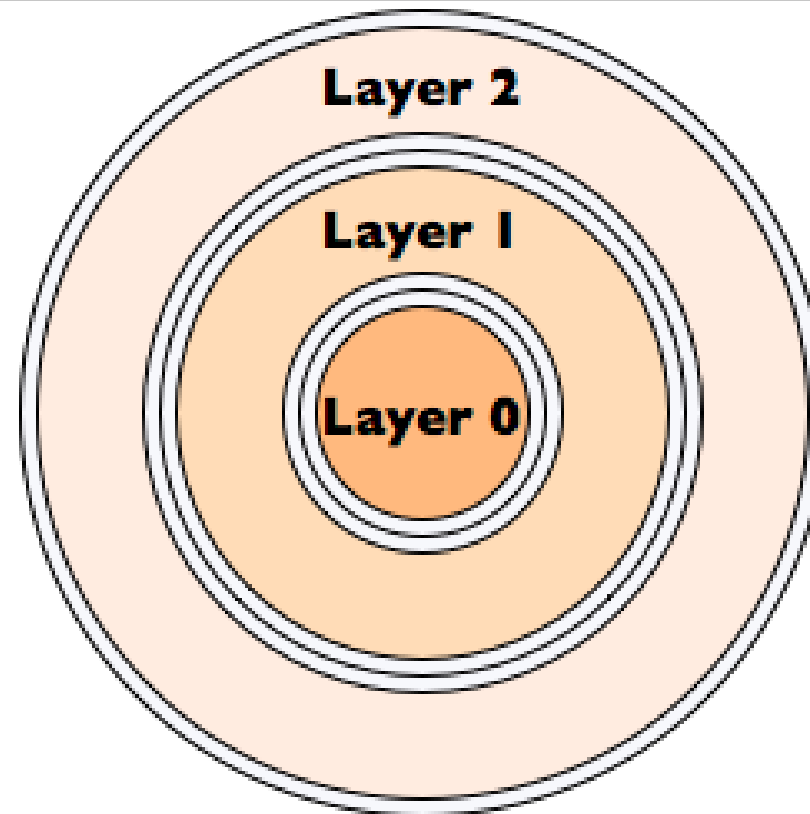
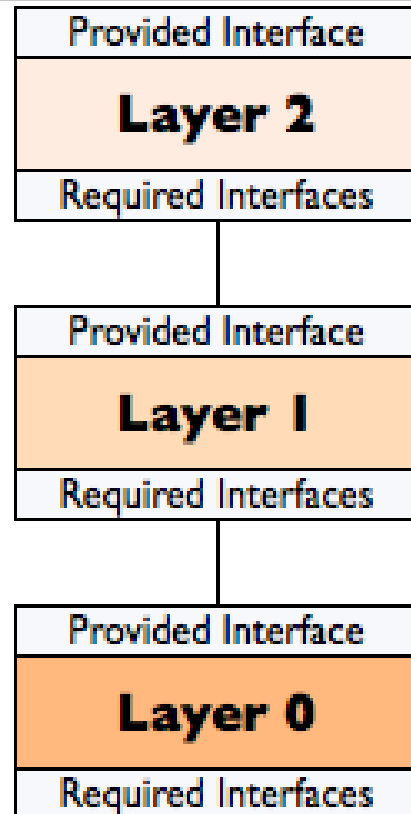


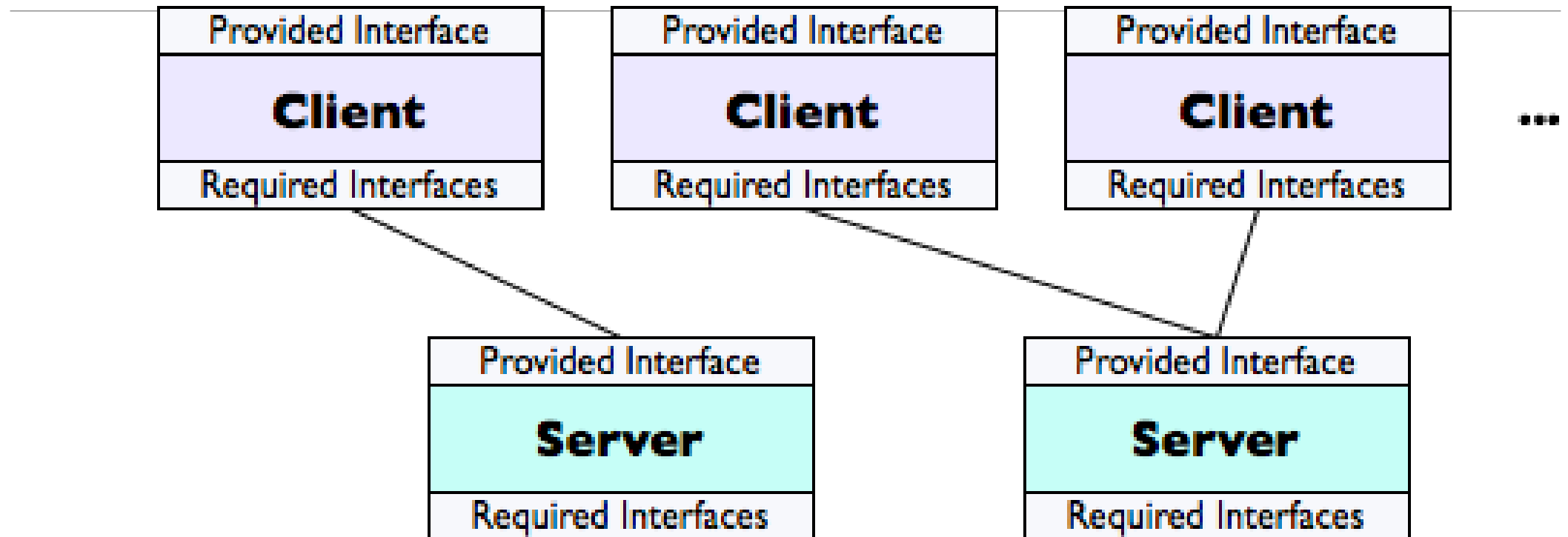


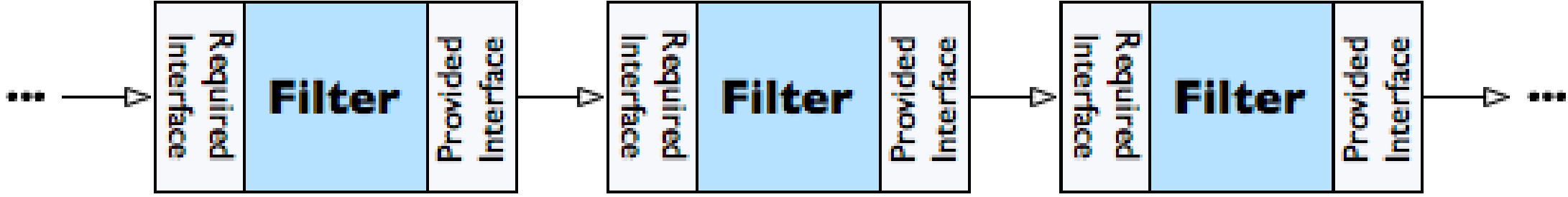


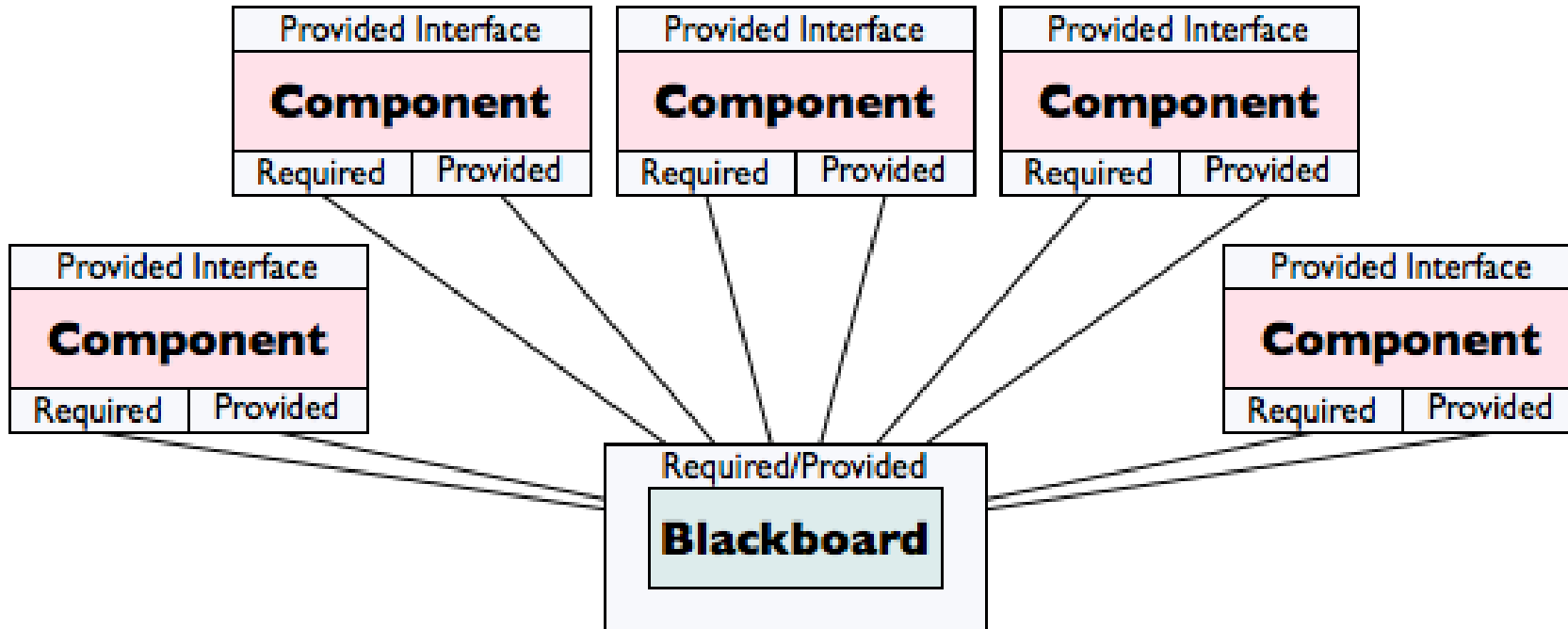
Mary Shaw and David Garlan. *Software Architecture: Perspectives on an Emerging Discipline*, Prentice Hall, 1996.

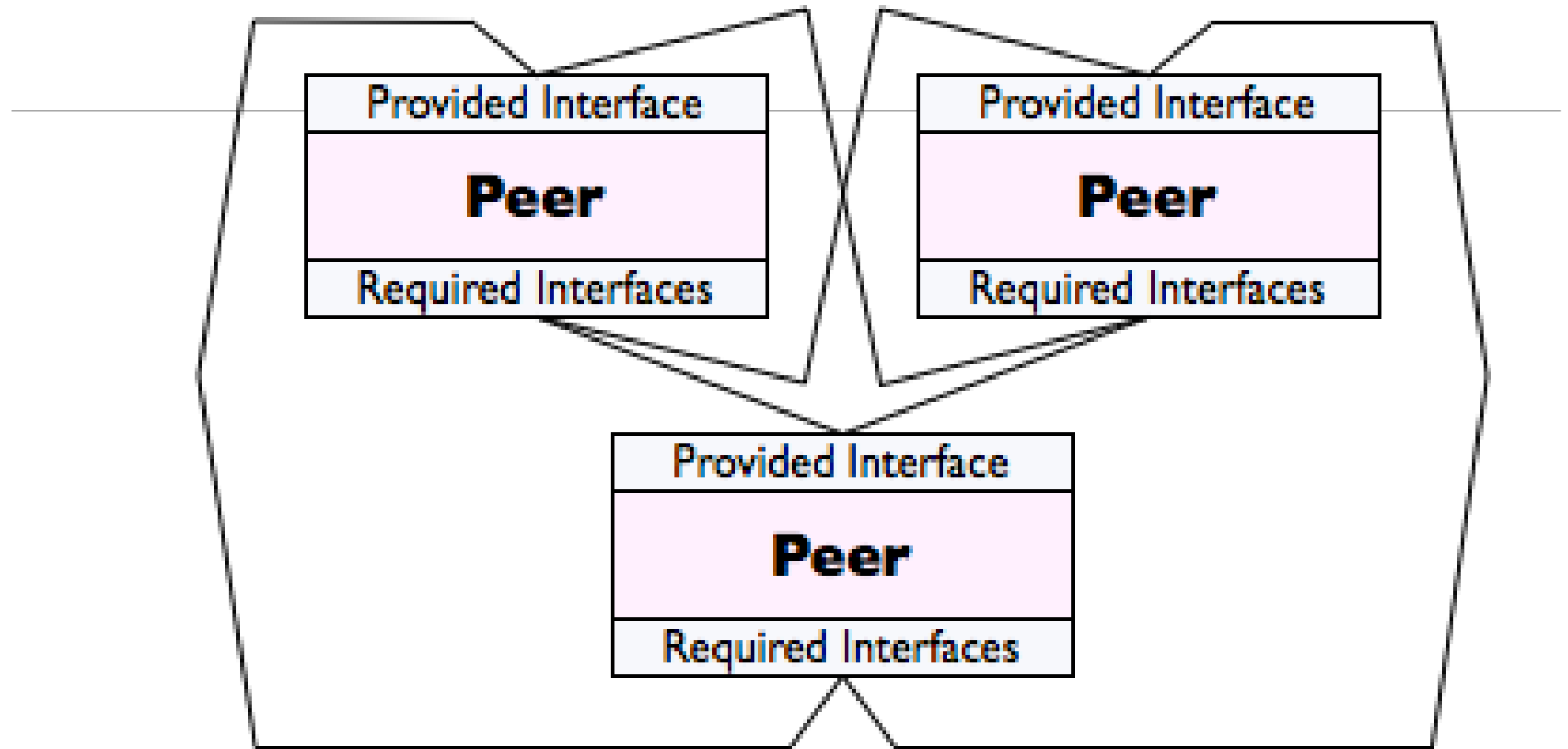
Mary Shaw. "Comparing Architectural Design Styles". in: *IEEE Software*, 12(6):27–41, 1995.











Definition of Distributed System

A distributed system is a collection of independent entities that cooperate to solve a problem that cannot be individually solved. Distributed systems have been in existence since the start of the universe. [Singhal, 2008]



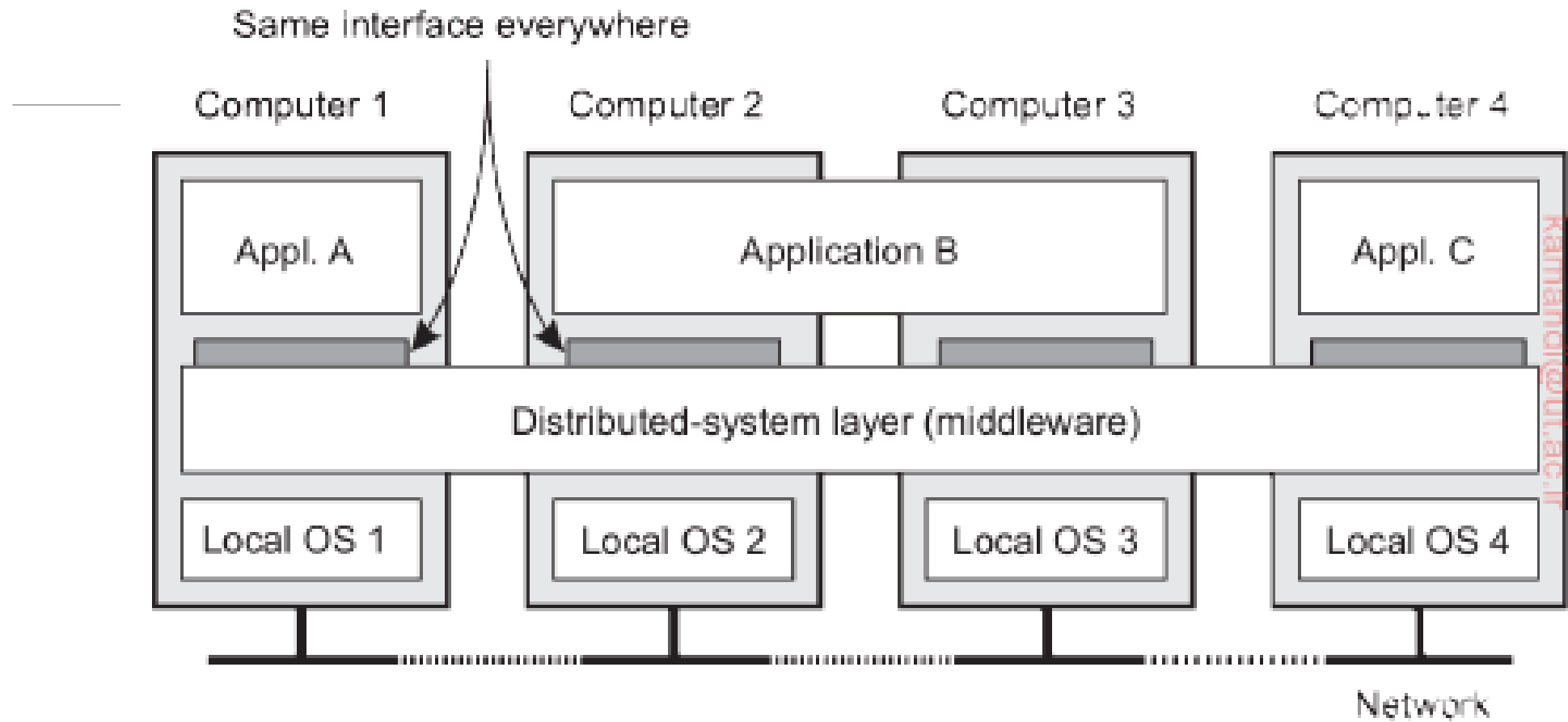
Distributed System: Definition (Tanenbaum)

Definition

A distributed system is a collection of **autonomous computing elements** that appears to its users as a **single coherent system**.

Characteristic features

- Autonomous computing elements, also referred to as **nodes**, be they hardware devices or software processes.
- Single coherent system: users or applications perceive a single system \Rightarrow nodes need to **collaborate**.



[Tanenbaum, 2017]

Definition of Distributed System

Distributed systems are characterized by their structure: a typical distributed system will consist of some large number of interacting devices that each run their own programs but that are affected by receiving messages, or observing shared-memory updates or the states of other devices. Examples of distributed computing systems range from simple systems in which a single client talks to a single server to huge amorphous networks like the Internet as a whole.

[Aspnes, 2017]

Definition of Distributed System

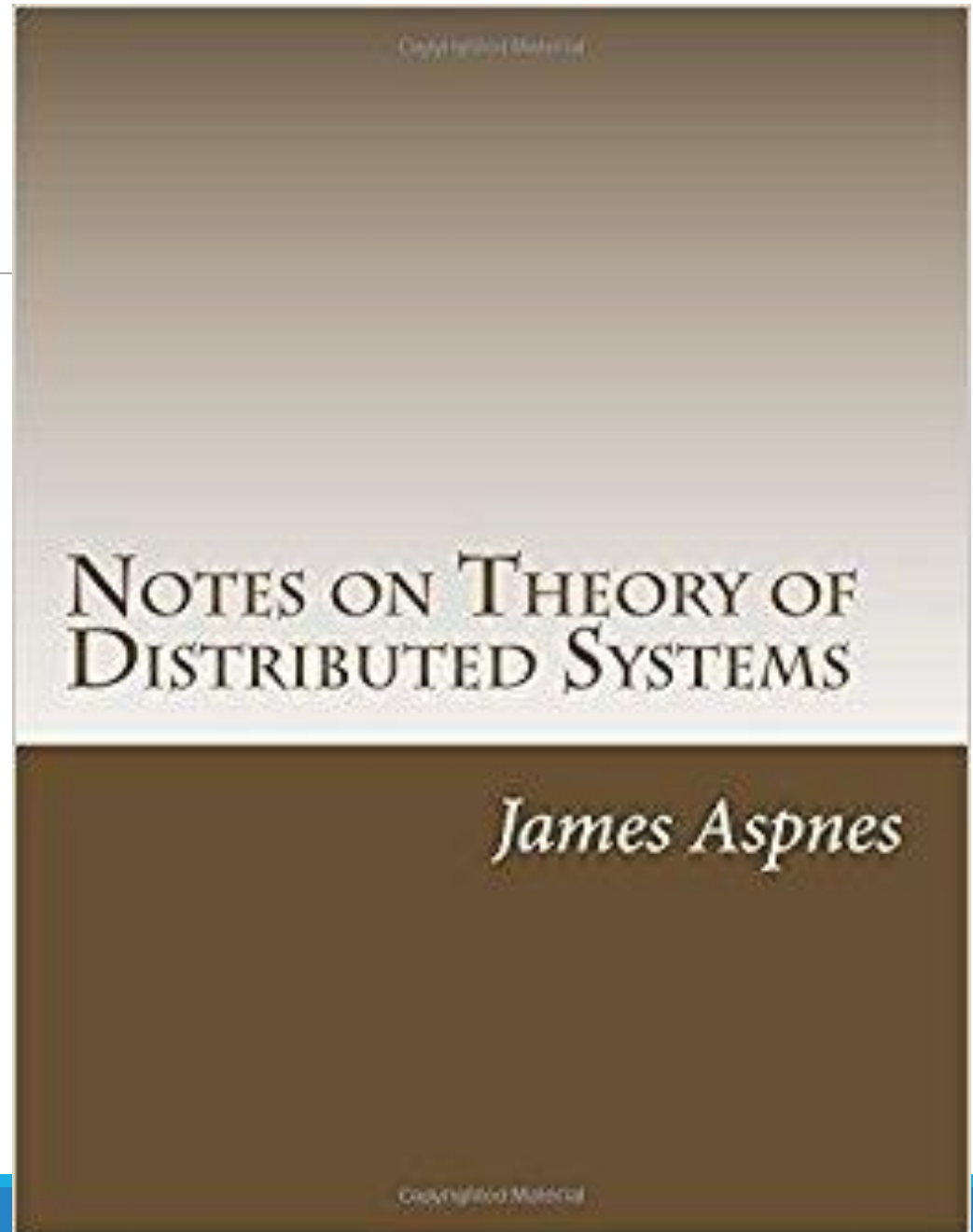
Leslie **Lamport** once said: *A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.*

[Ghosh, 2007]

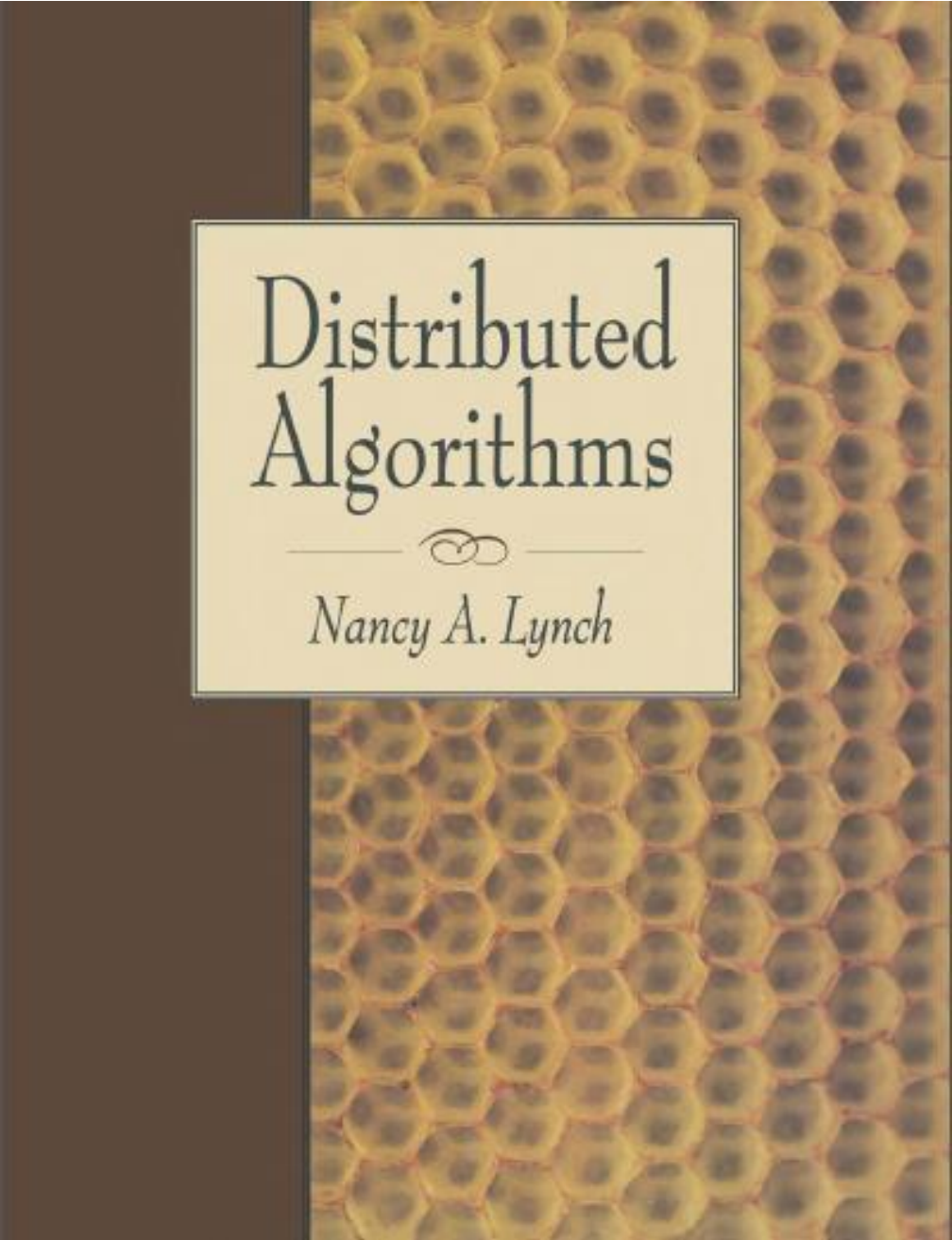
Distributed systems are a computing paradigm whereby two or more nodes work with each other in a coordinated fashion in order to achieve a common outcome and it's modeled in such a way that end users see it as a single logical platform.

[Bashir]

James Aspnes, Notes on Theory of Distributed Systems, 2017.



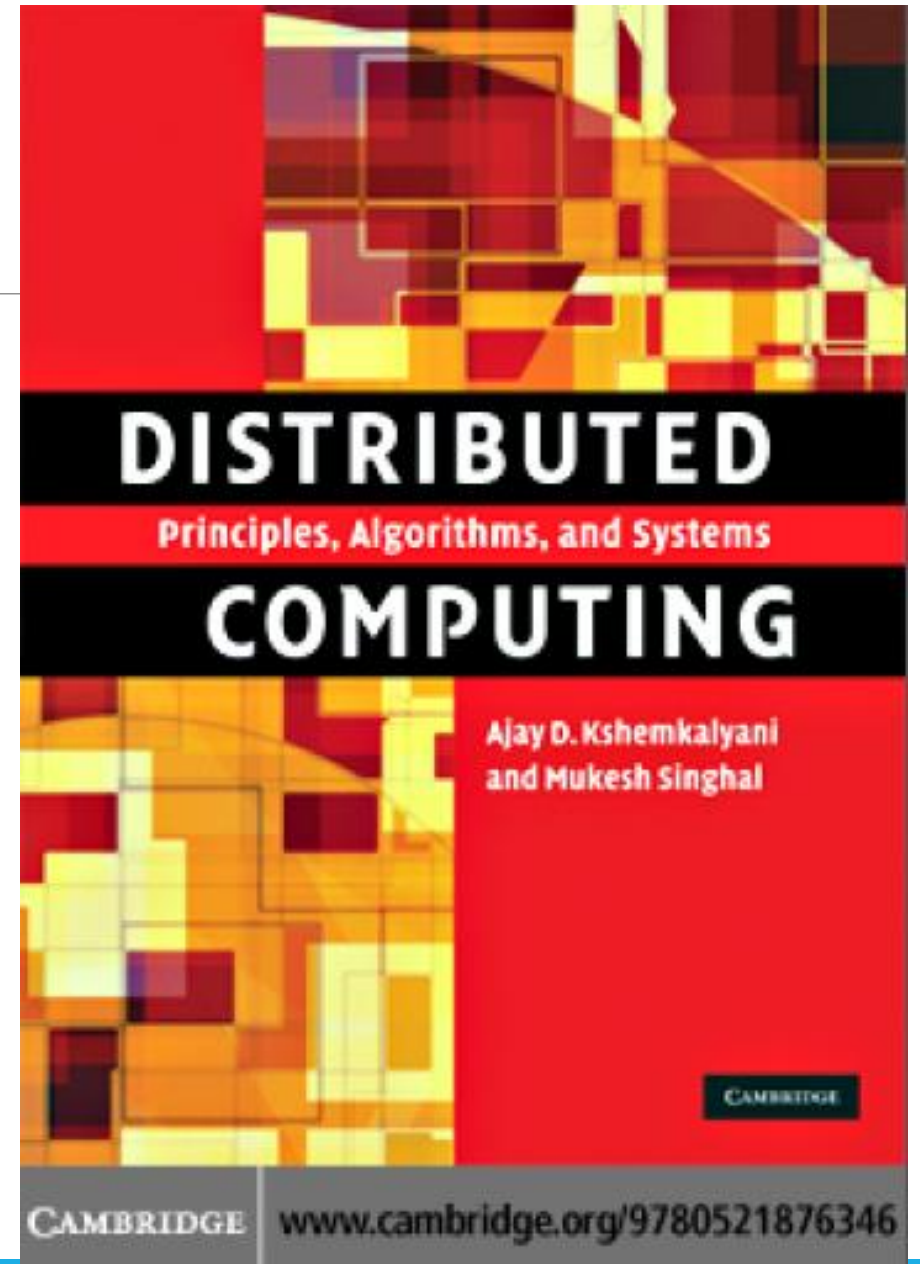
Nancy A. Lynch, Distributed Systems, 1996.

The image shows the front cover of the book 'Distributed Algorithms' by Nancy A. Lynch. The cover has a dark brown spine and a light beige front panel. The front panel features a repeating pattern of small, dark, hexagonal shapes, resembling a honeycomb or a cellular structure. The title 'Distributed Algorithms' is printed in a large, black, serif font. Below the title is a small, decorative flourish. The author's name, 'Nancy A. Lynch', is printed in a smaller, black, serif font.

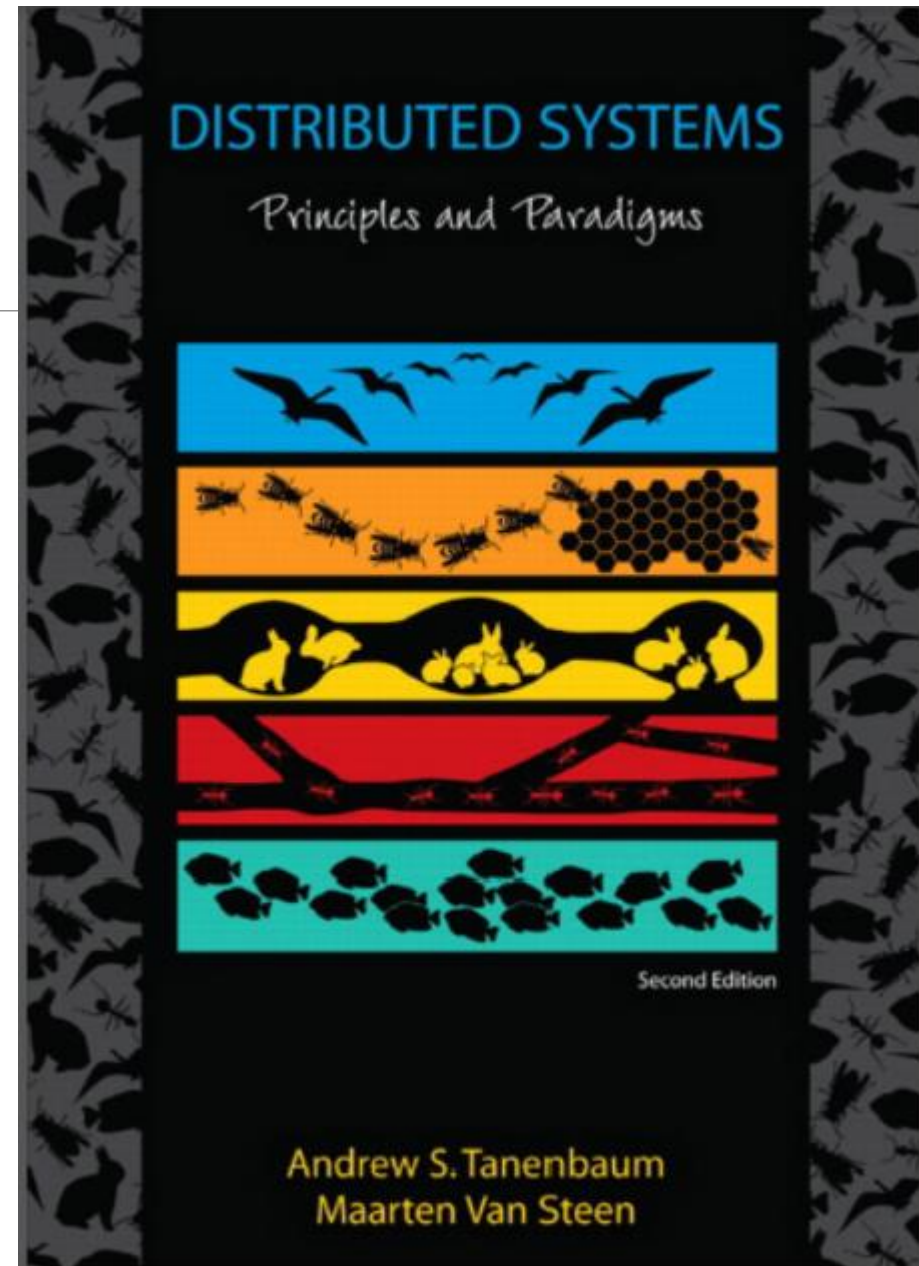
Distributed Algorithms

Nancy A. Lynch

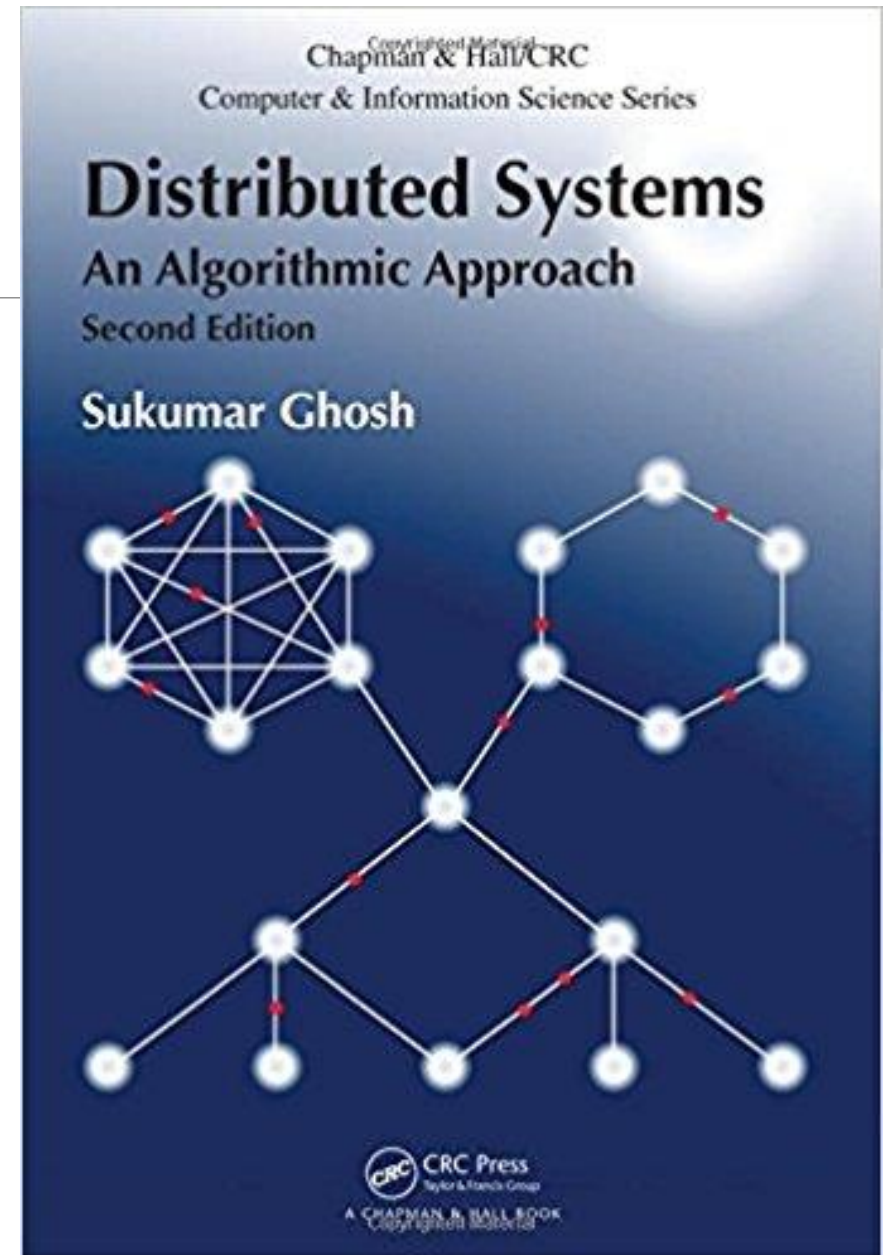
Ajay D. Kshemkalyani and Mukesh Singhal, Distributed Computing, 2008.



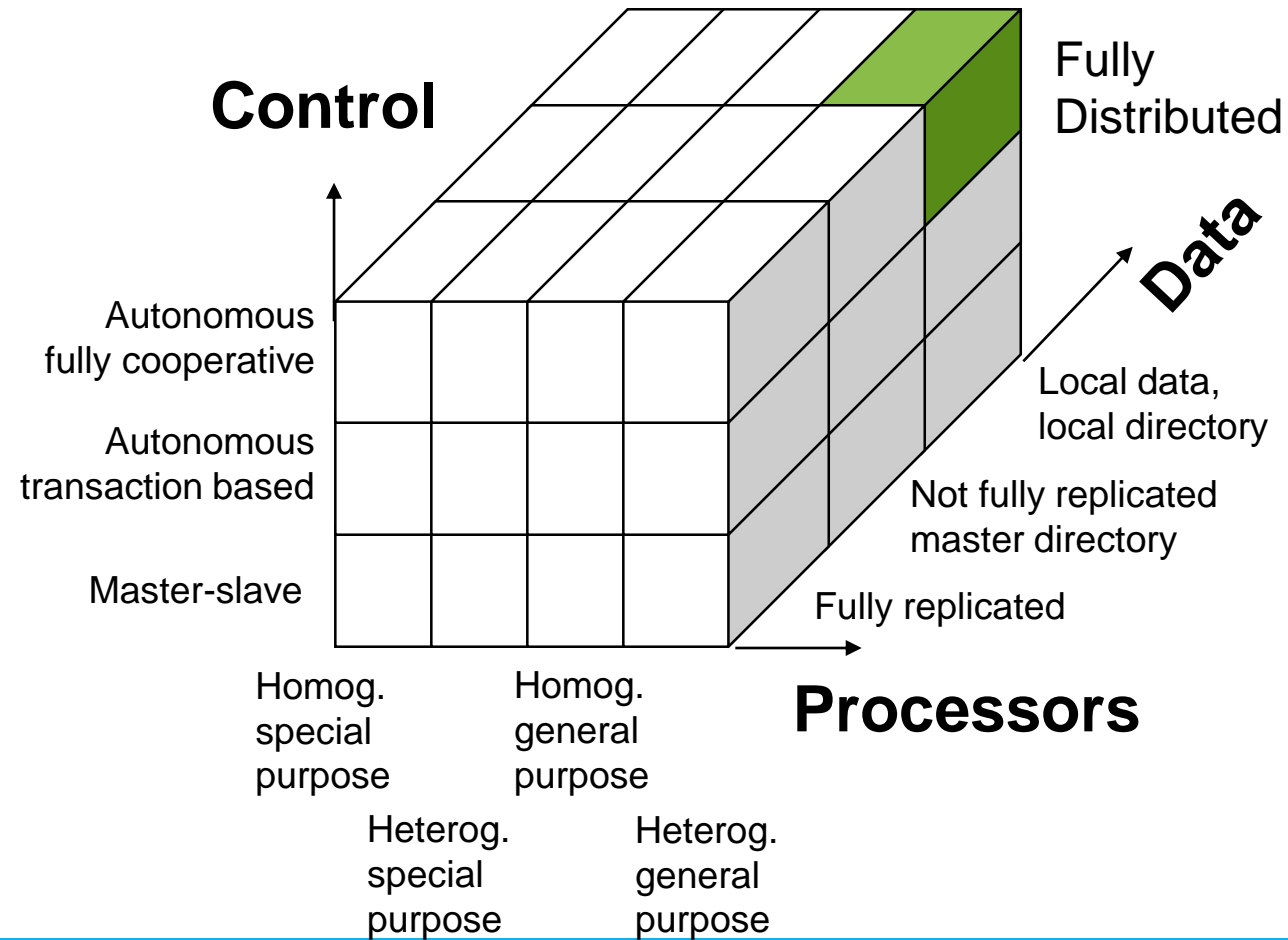
Andrew S. Tanenbaum, Maarten Van Steen,
Distributed systems, principles and paradigms,
3rd edition, 2017.



Sukumar Ghosh, Distributed Systems, an algorithmic approach, second edition, 2015.



Distributed System Types



CAP Theorem

This is also known as Brewer's theorem, introduced originally by Eric Brewer as a conjecture in 1998; in 2002 it was proved as a theorem by Seth Gilbert and Nancy Lynch.

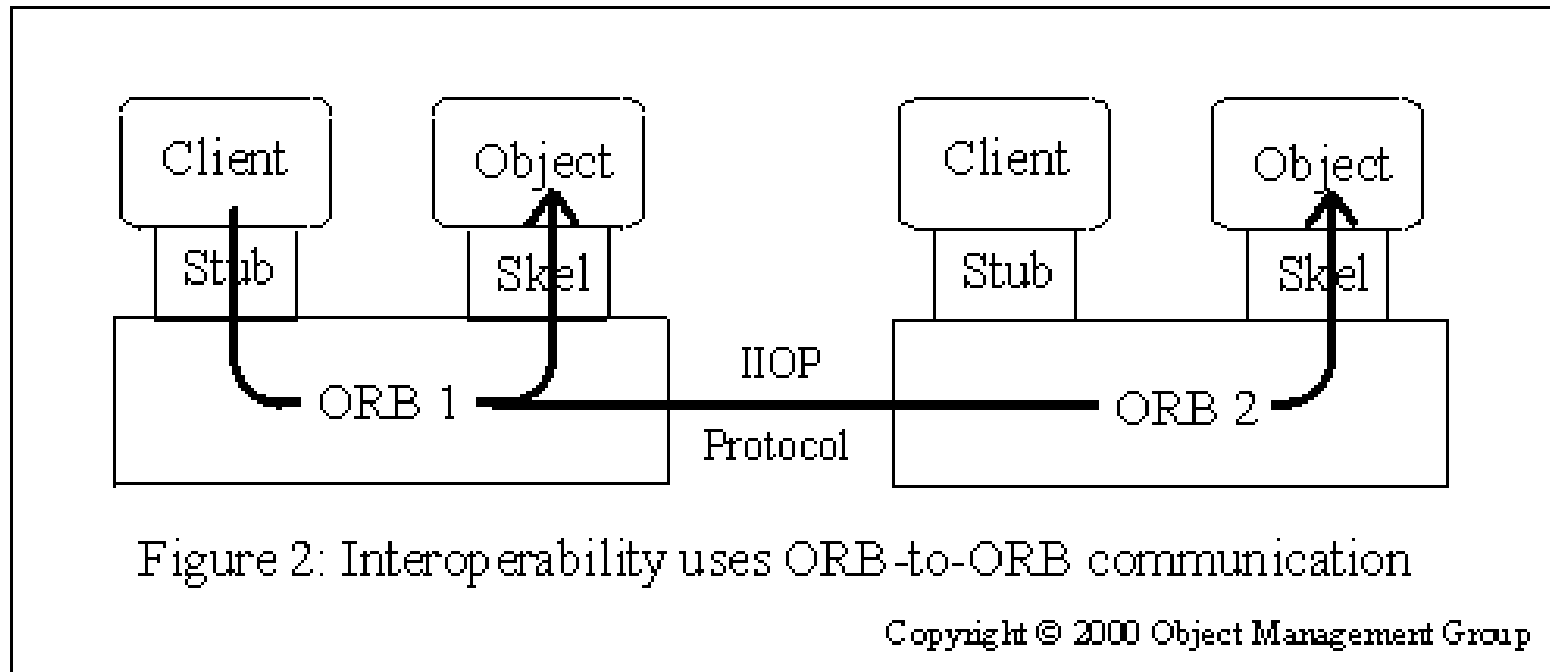
Any distributed system cannot have Consistency, Availability, and Partition tolerance simultaneously:

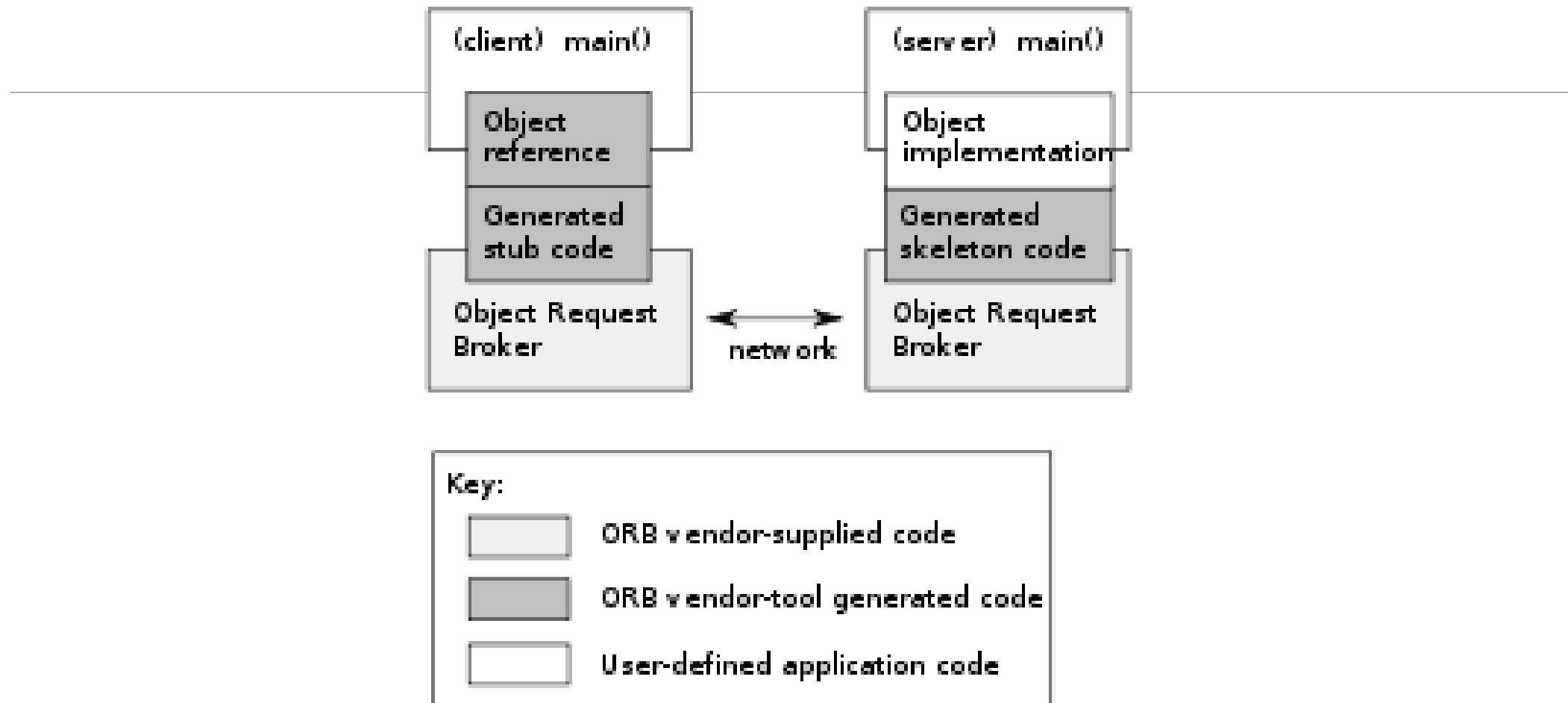
Consistency is a property that ensures that all nodes in a distributed system have a single latest copy of data

Availability means that the system is up, accessible for use, and is accepting incoming requests and responding with data without any failures as and when required

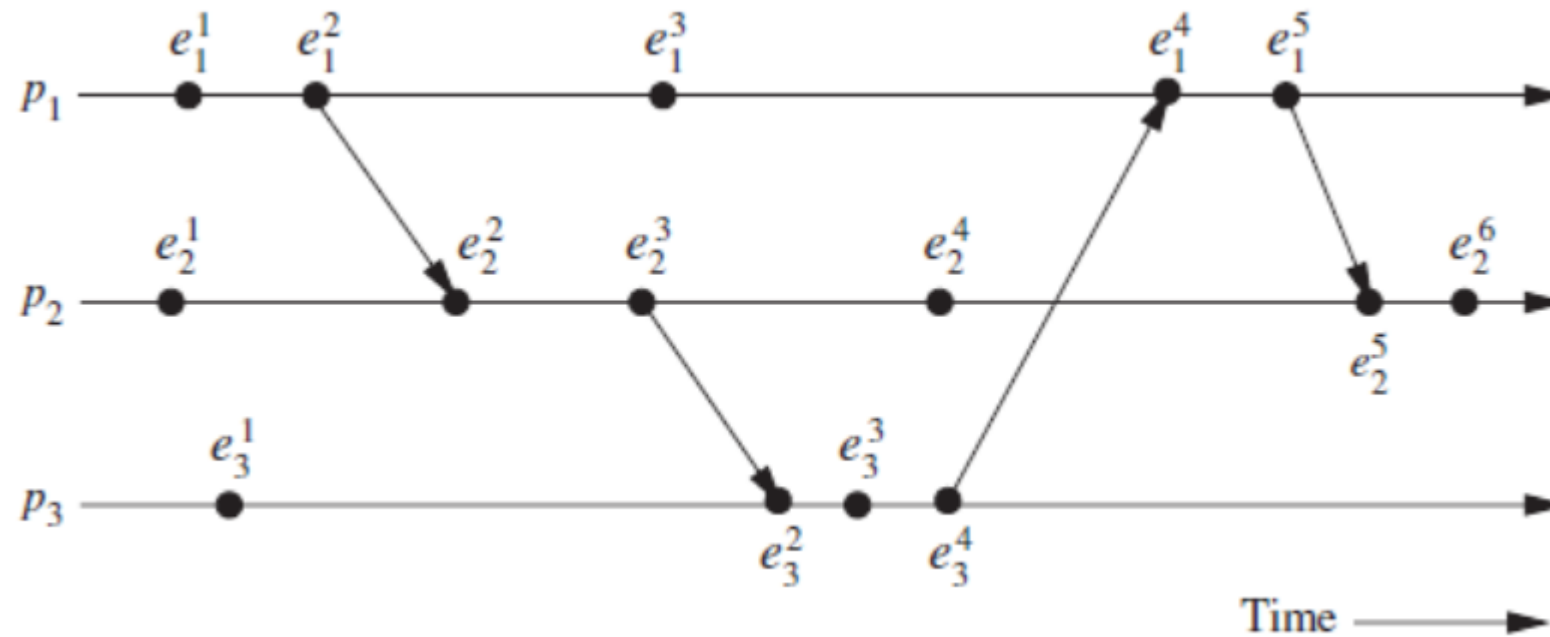
Partition tolerance ensures that if a group of nodes fails the distributed system still continues to operate correctly

CORBA





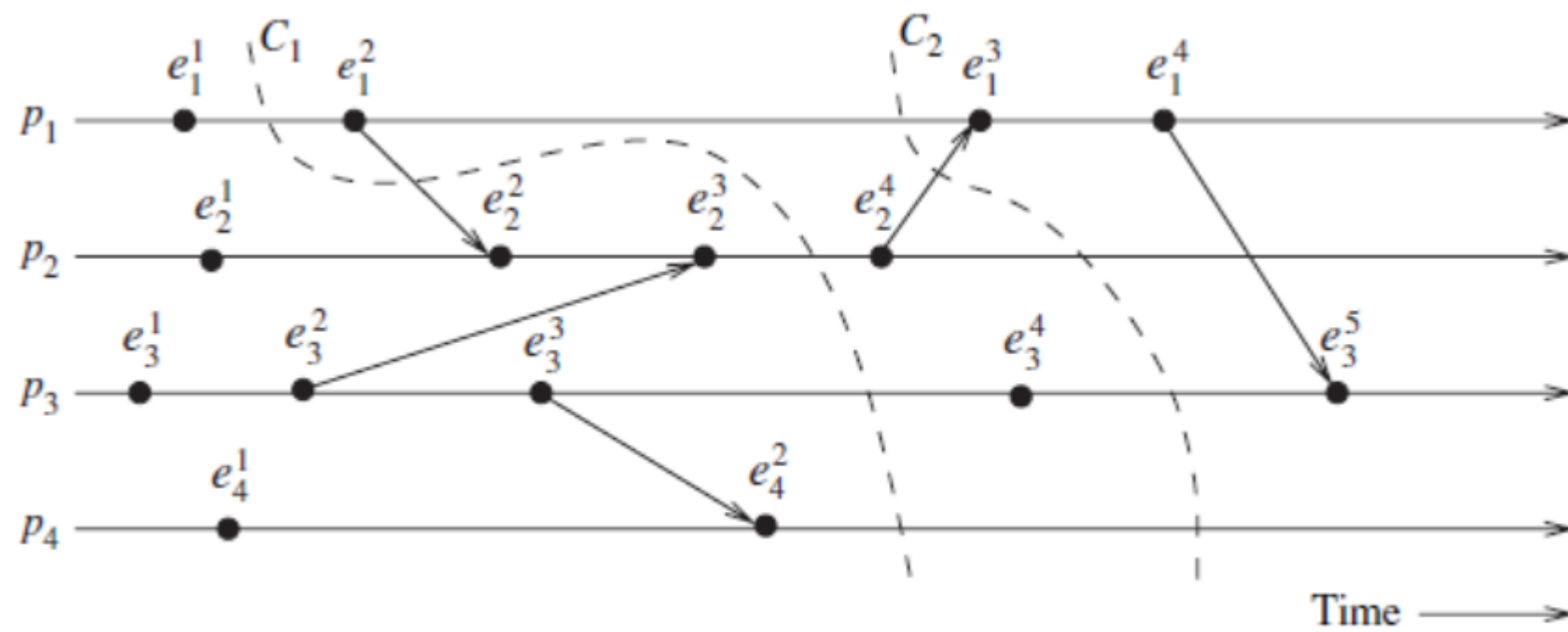
Model of Distributed Execution



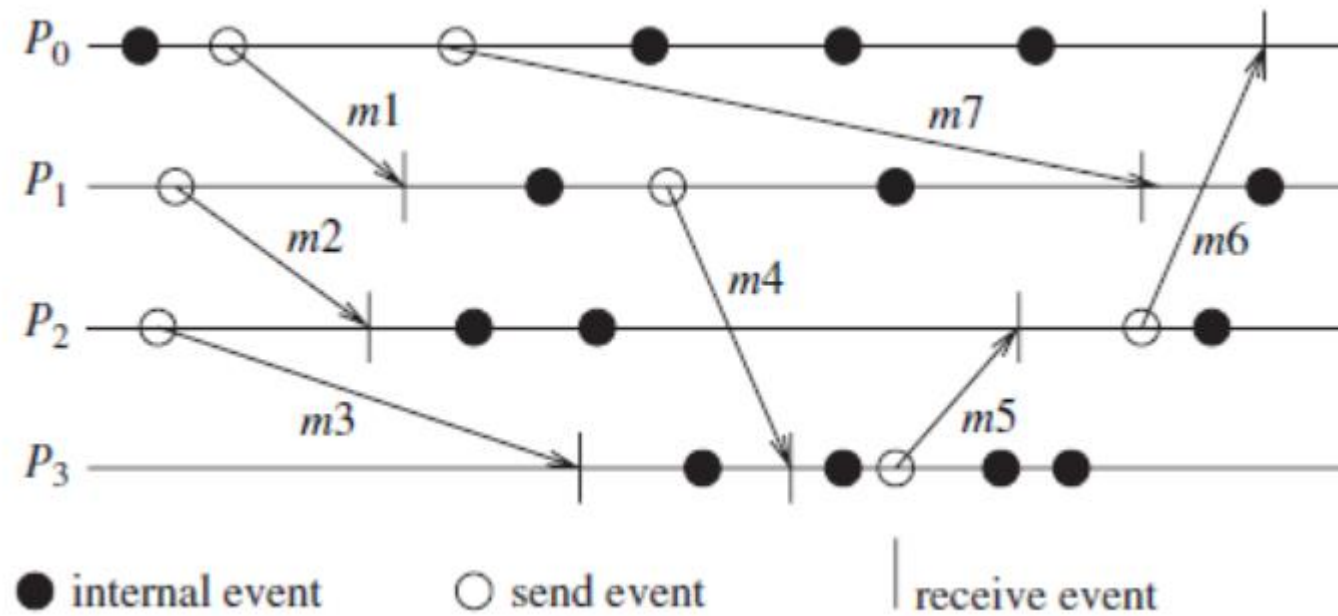
Causal precedence relation

$$\forall e_i^x, \forall e_j^y \in H, \quad e_i^x \rightarrow e_j^y \Leftrightarrow \left\{ \begin{array}{l} e_i^x \rightarrow_i e_j^y \text{ i.e., } (i = j) \wedge (x < y) \\ \text{or} \\ e_i^x \rightarrow_{msg} e_j^y \\ \text{or} \\ \exists e_k^z \in H : e_i^x \rightarrow e_k^z \wedge e_k^z \rightarrow e_j^y \end{array} \right.$$

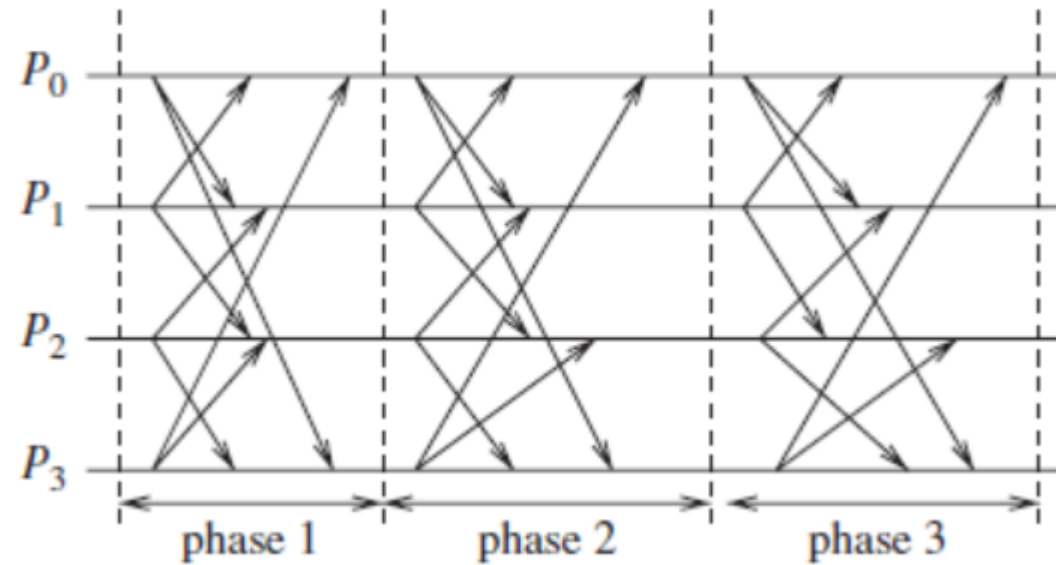
CO: For any two messages m_{ij} and m_{kj} , if $send(m_{ij}) \longrightarrow send(m_{kj})$,
then $rec(m_{ij}) \longrightarrow rec(m_{kj})$.

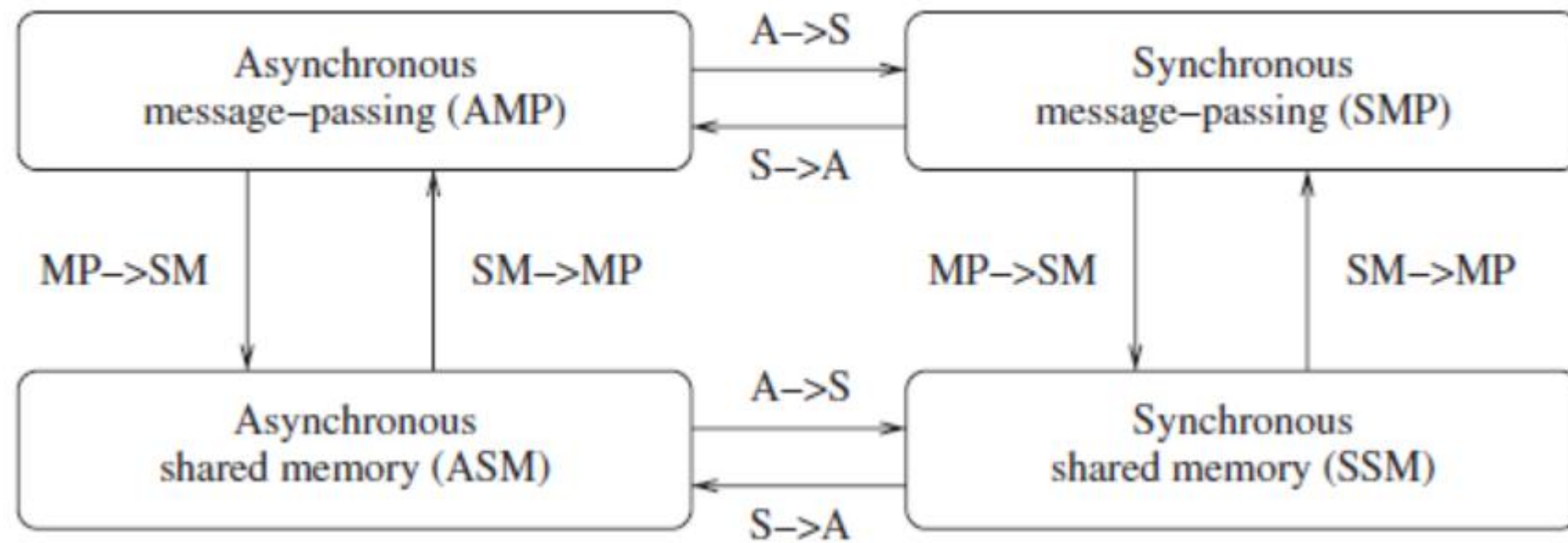


Asynchronous Execution



Synchronous Execution





تمرین

در مورد قضیه CAP تحقیق کنید و گزارشی در حد یک صفحه تهیه کنید.

در مورد رابطه سیستم های توزیع شده مبتنی بر ارسال پیام و مبتنی بر حافظه مشترک تحقیق کنید. آیا هر سیستم توزیع شده ای که با یکی از این دو مدل طراحی شود را می توان با مدل دیگر نیز طراحی کرد. در مورد پاسخ خود استدلال کنید و گزارشی در حد ۱ الی ۲ صفحه تهیه کنید.

مقاله زیر را مطالعه کنید و خلاصه آن را در ۲ صفحه تهیه کنید.

FARHAD ARBAB, Reo: a channel-based coordination model for component composition, Math. Struct. in Comp. Science (2004), vol. 14, pp. 329–366.