軟體工程
(Software Engineering)

安全和隱私
(Security and Privacy)

1091SE09
MBA, IM, NTPU (M5118) (Fall 2020)
Tue 2, 3, 4 (9:10-12:00) (B8F40)

Min-Yuh Day
戴敏育
Associate Professor
副教授
Institute of Information Management, National Taipei University
國立臺北大學 資訊管理研究所
https://web.ntpu.edu.tw/~myday

2020-12-01

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2020/09/15	軟體工程概論 (Introduction to Software Engineering)
2	2020/09/22	軟體產品與專案管理：軟體產品管理，原型設計 (Software Products and Project Management: Software product management and prototyping)
3	2020/09/29	敏捷軟體工程：敏捷方法、Scrum、極限程式設計 (Agile Software Engineering: Agile methods, Scrum, and Extreme Programming)
4	2020/10/06	功能、場景和故事 (Features, Scenarios, and Stories)
5	2020/10/13	軟體架構：架構設計、系統分解、分散式架構 (Software Architecture: Architectural design, System decomposition, and Distribution architecture)
6	2020/10/20	軟體工程個案研究 I (Case Study on Software Engineering I)

課程大綱 (Syllabus)

國立臺北大學 National Taipei University

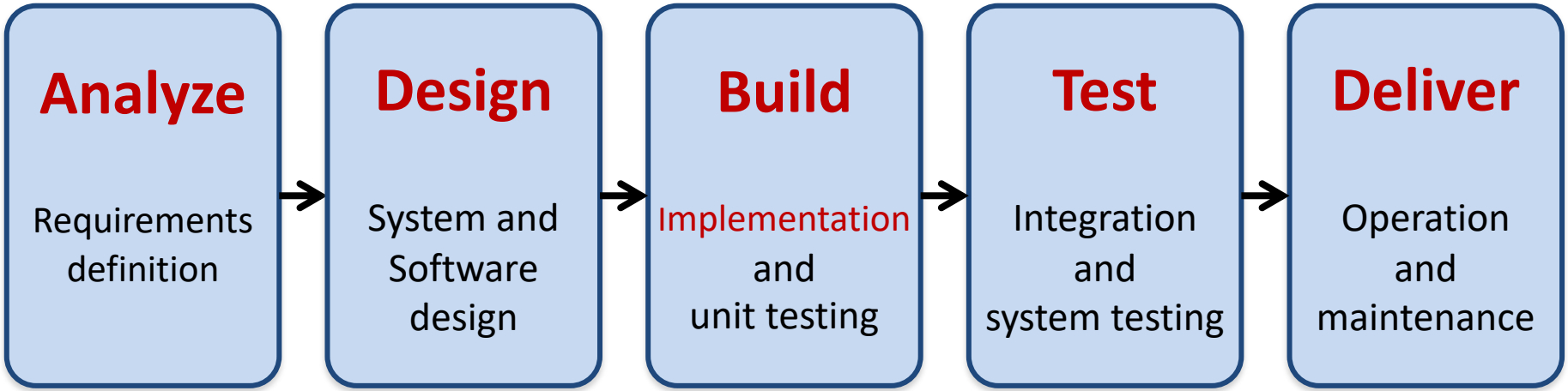
週次 (Week)	日期 (Date)	內容 (Subject/Topics)
7	2020/10/27	基於雲的軟體：虛擬化和容器、軟體即服務 (Cloud-Based Software: Virtualization and containers, Everything as a service, Software as a service)
8	2020/11/03	雲端運算與雲軟體架構 (Cloud Computing and Cloud Software Architecture)
9	2020/11/10	期中報告 (Midterm Project Report)
10	2020/11/17	微服務架構：RESTful服務、服務部署 (Microservices Architecture: RESTful services, Service deployment)
11	2020/11/24	軟體工程產業實務 (Industry Practices of Software Engineering)
12	2020/12/01	安全和隱私 (Security and Privacy)

課程大綱 (Syllabus)

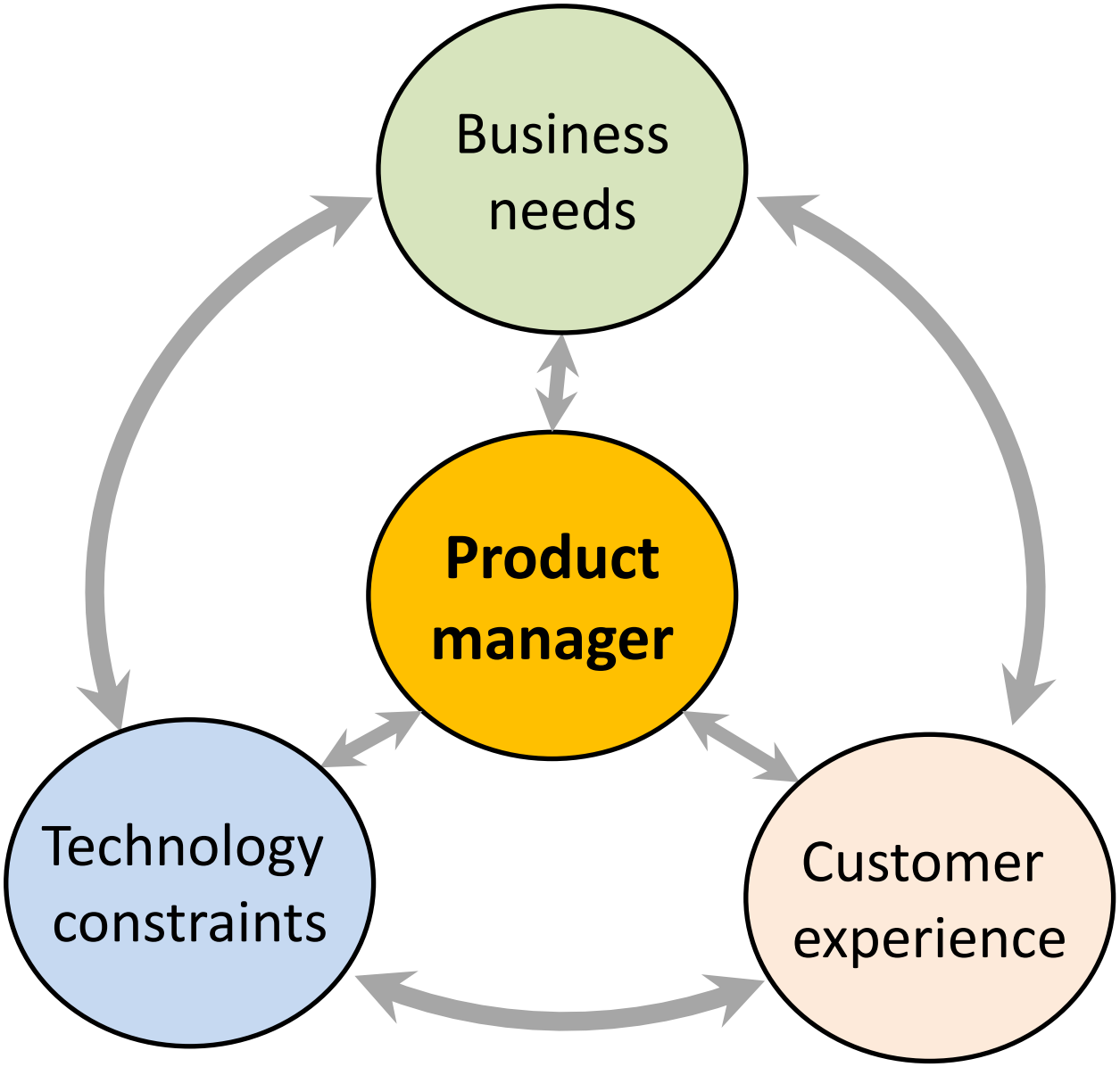
國立臺北大學 National Taipei University

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
13	2020/12/08	軟體工程個案研究 II (Case Study on Software Engineering II)
14	2020/12/15	可靠的程式設計 (Reliable Programming)
15	2020/12/22	測試：功能測試、測試自動化、測試驅動的開發、程式碼審查 (Testing: Functional testing, Test automation, Test-driven development, and Code reviews)
16	2020/12/29	DevOps和程式碼管理：程式碼管理和DevOps自動化 (DevOps and Code Management: Code management and DevOps automation)
17	2021/01/05	期末報告 I (Final Project Report I)
18	2021/01/12	期末報告 II (Final Project Report I)

Software Engineering and Project Management

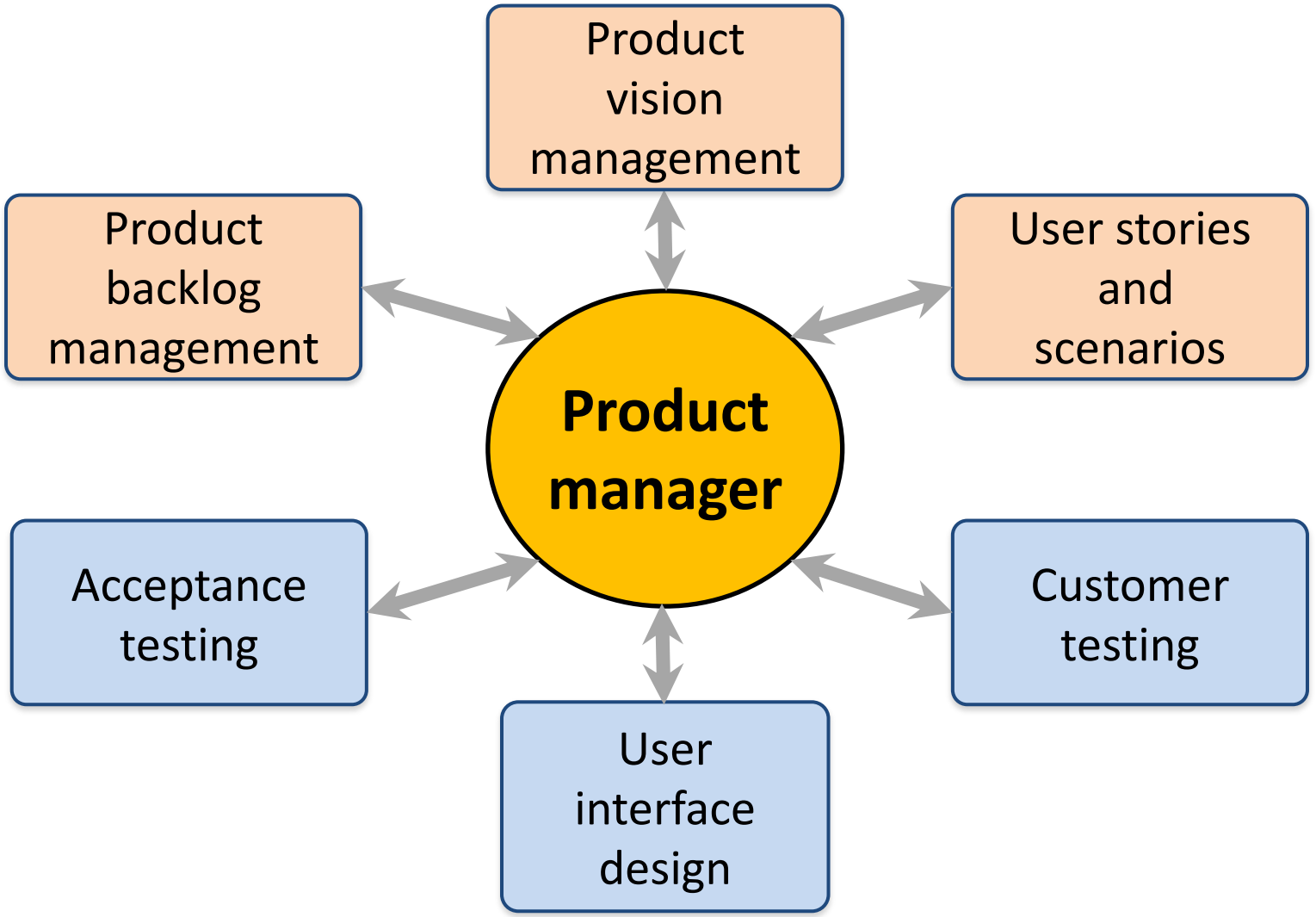


Product management concerns



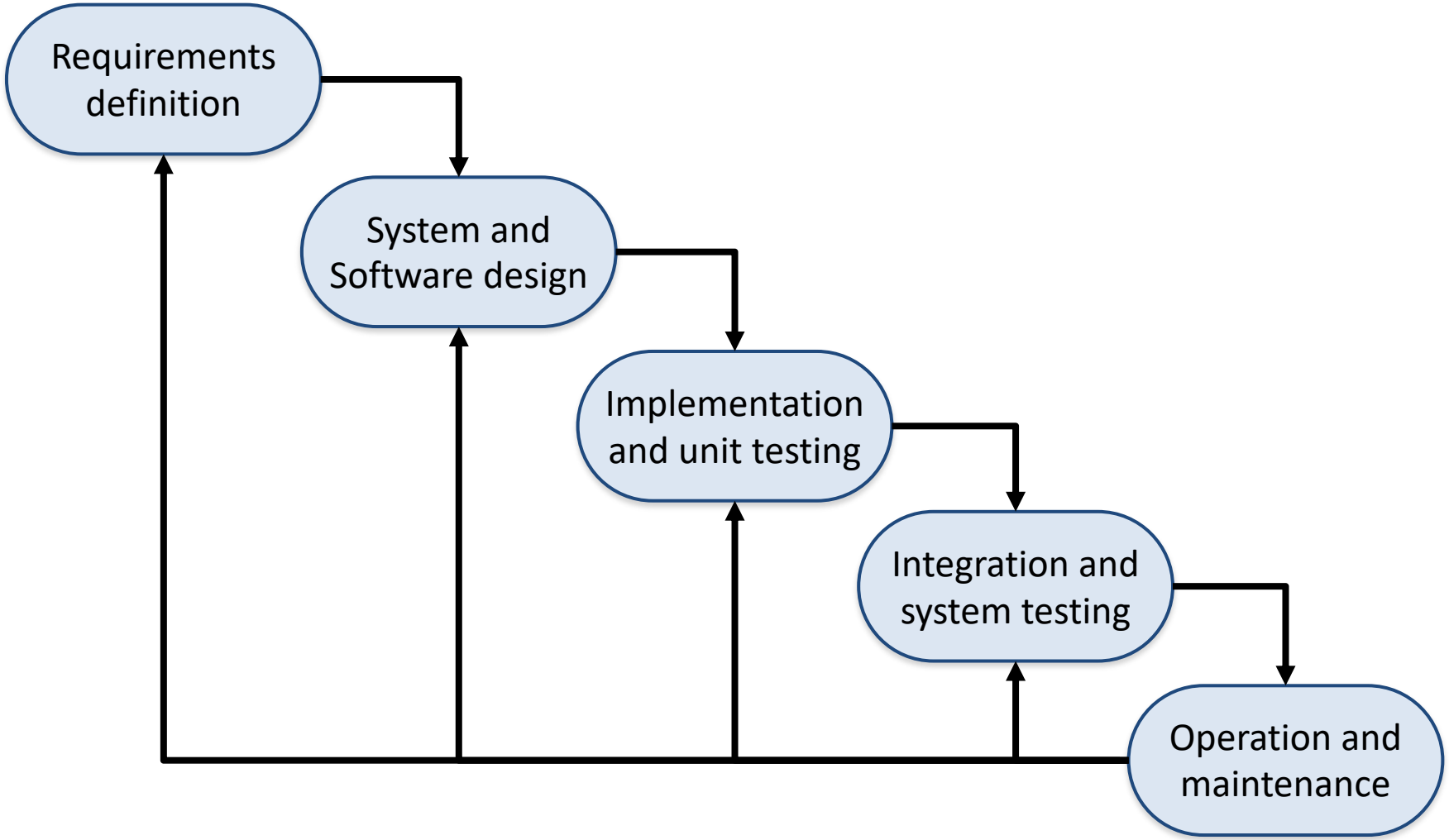
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Technical interactions of product managers



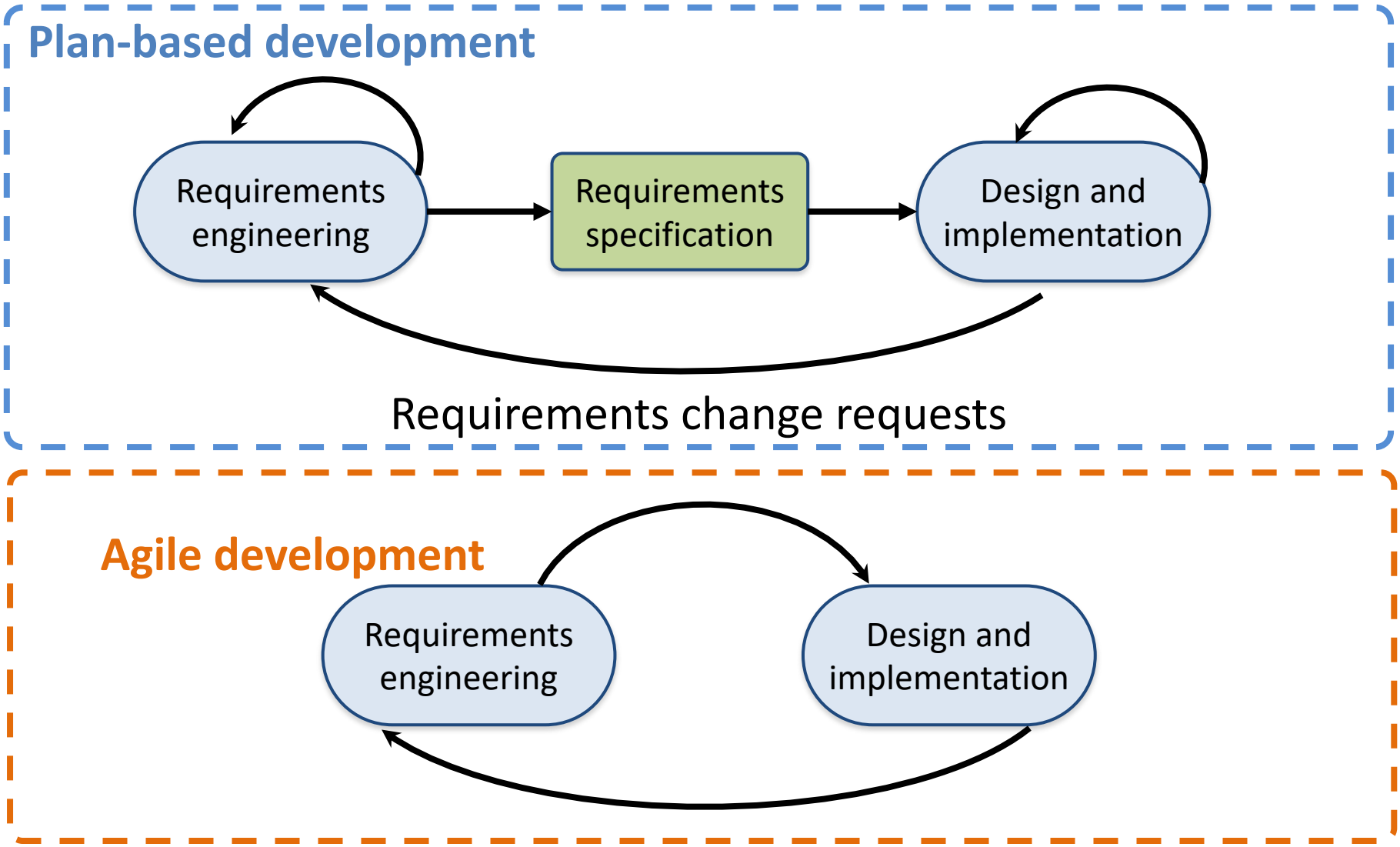
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Software Development Life Cycle (SDLC)
The waterfall model



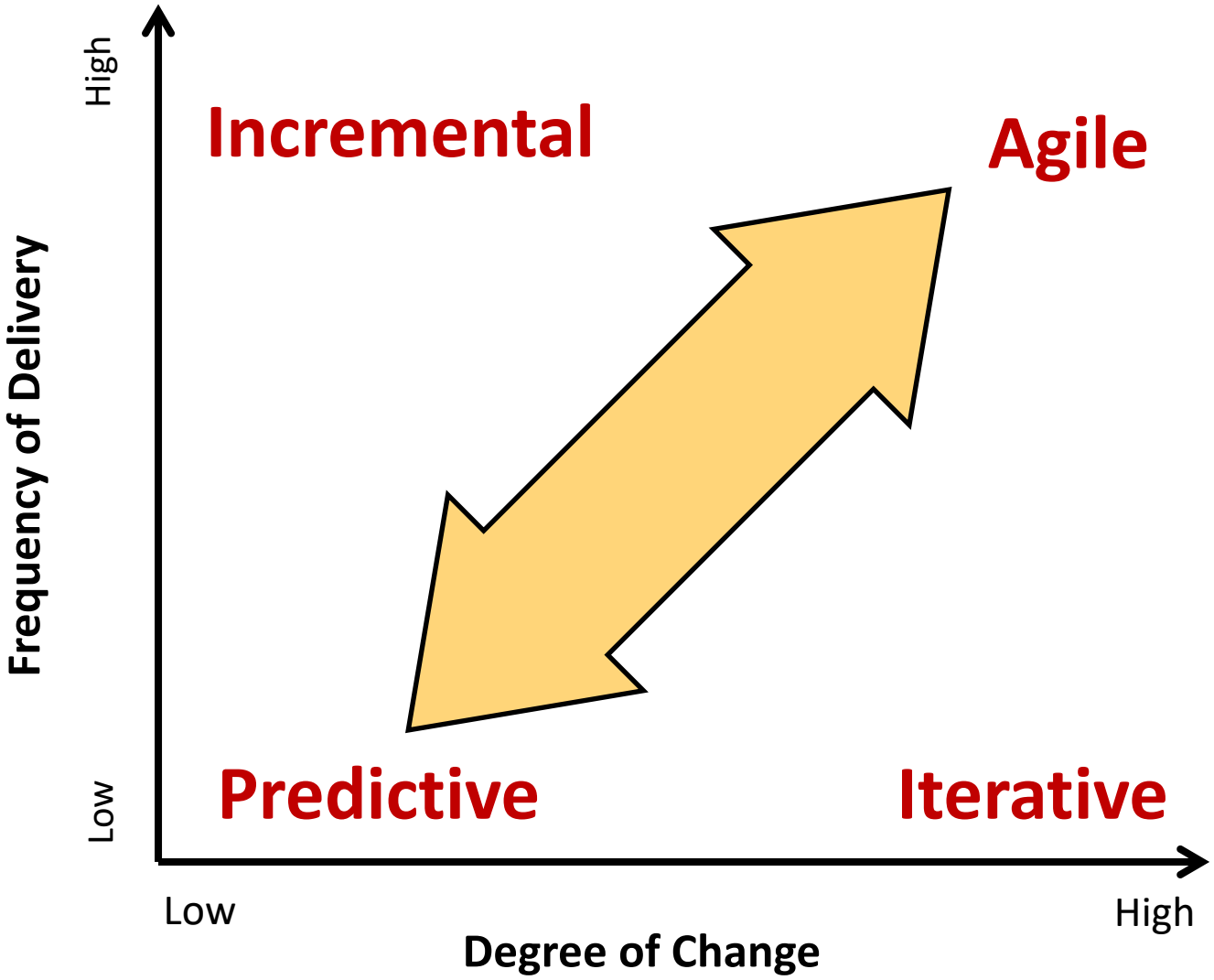
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Plan-based and Agile development



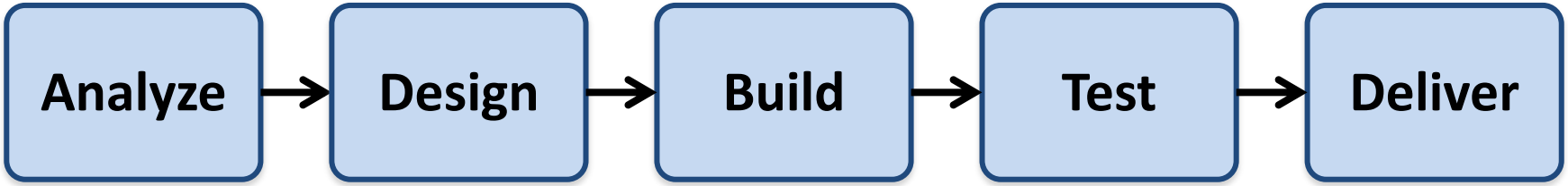
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

The Continuum of Life Cycles



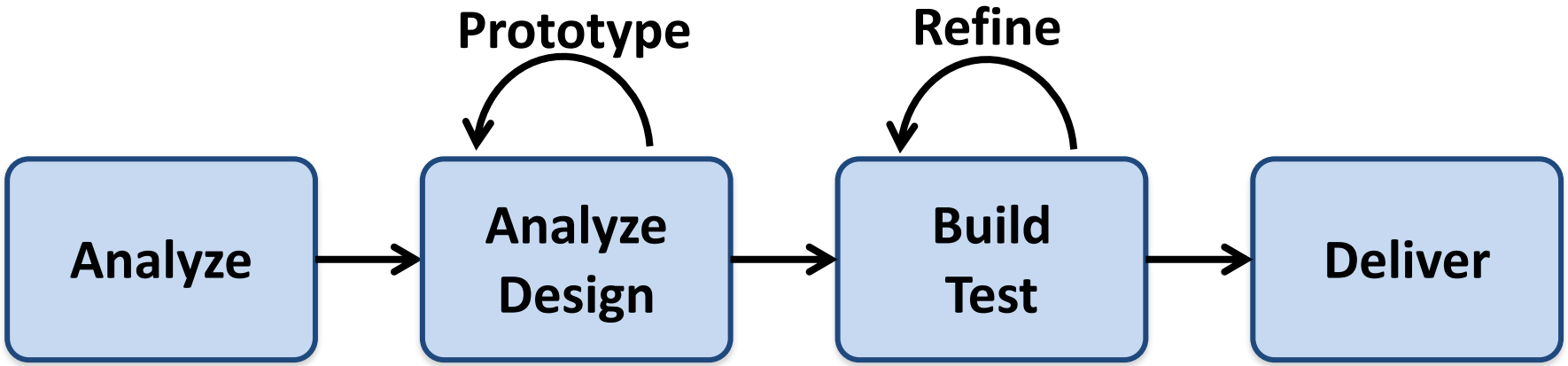
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Predictive Life Cycle



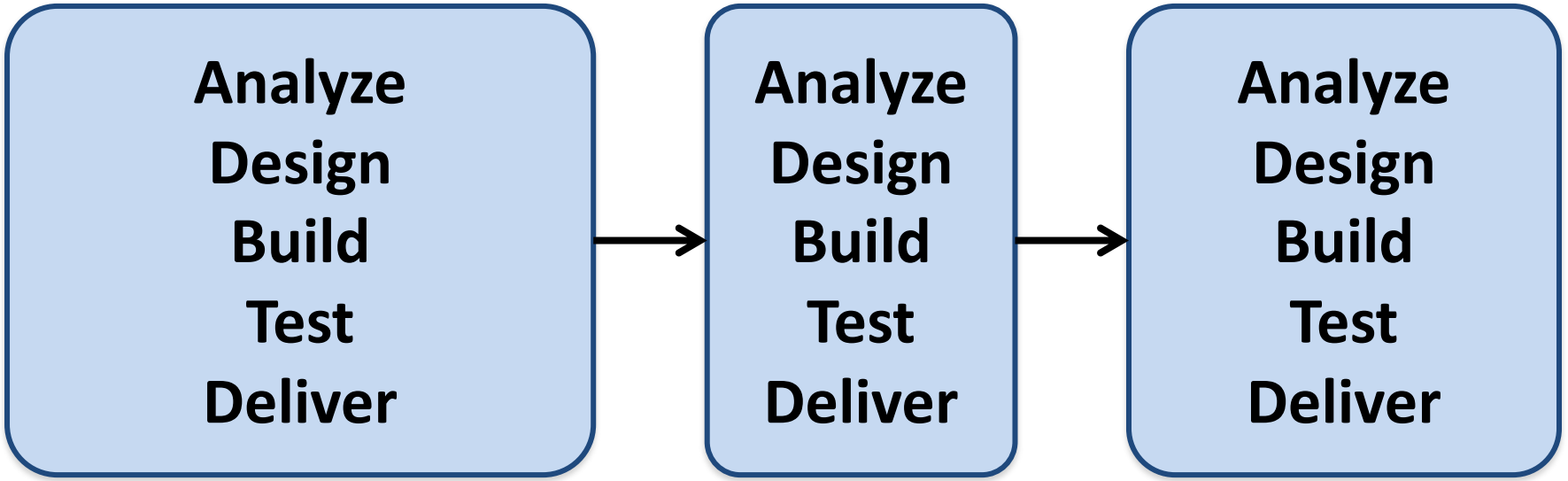
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Iterative Life Cycle



Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

A Life Cycle of Varying-Sized Increments



Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Iteration-Based and Flow-Based Agile Life Cycles

Iteration-Based Agile

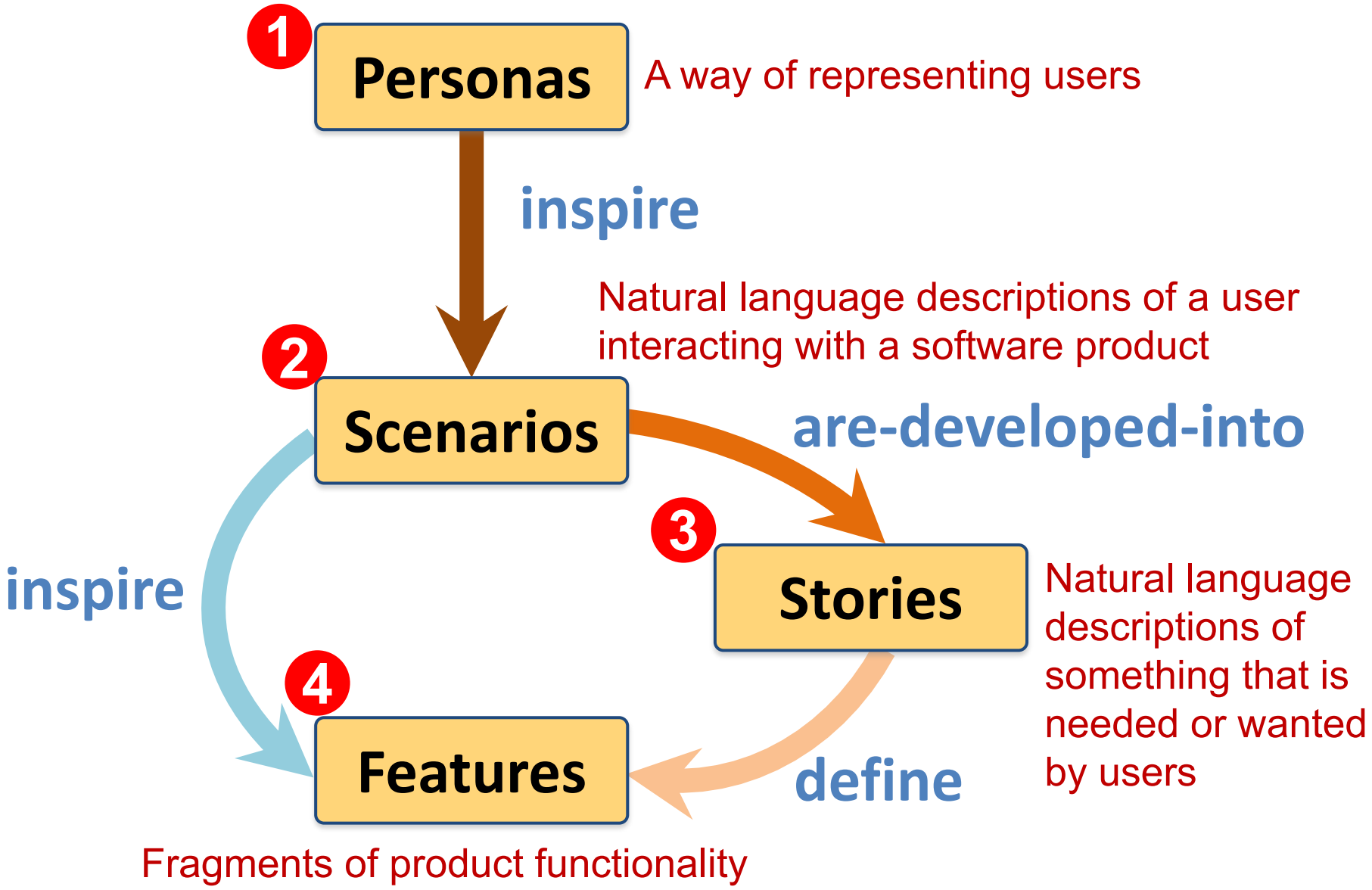
Requirements **Analysis** **Design** **Build** **Test**	**Requirements** **Analysis** **Design** **Build** **Test**	**Requirements** **Analysis** **Design** **Build** **Test**	**Requirements** **Analysis** **Design** **Build** **Test**	**Repeat** **as needed** **…**	**Requirements** **Analysis** **Design** **Build** **Test**	**Requirements** **Analysis** **Design** **Build** **Test**

Flow-Based Agile

Requirements **Analysis** **Design** **Build** **Test** the number of features in the WIP limit	**Requirements** **Analysis** **Design** **Build** **Test** the number of features in the WIP limit	**Requirements** **Analysis** **Design** **Build** **Test** the number of features in the WIP limit	**Repeat** **as needed** **…**	**Requirements** **Analysis** **Design** **Build** **Test** the number of features in the WIP limit	**Requirements** **Analysis** **Design** **Build** **Test** the number of features in the WIP limit

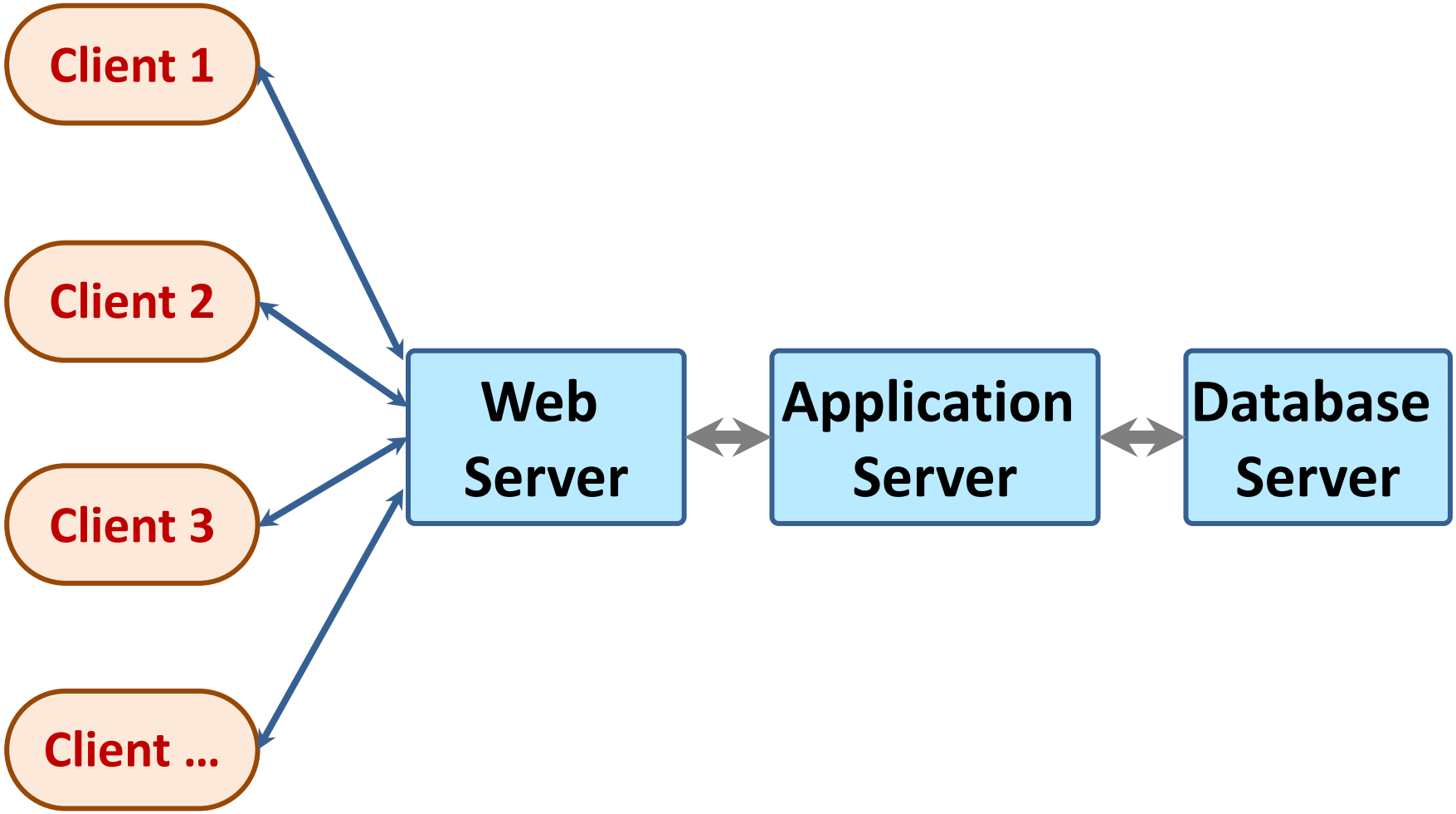
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

From personas to features



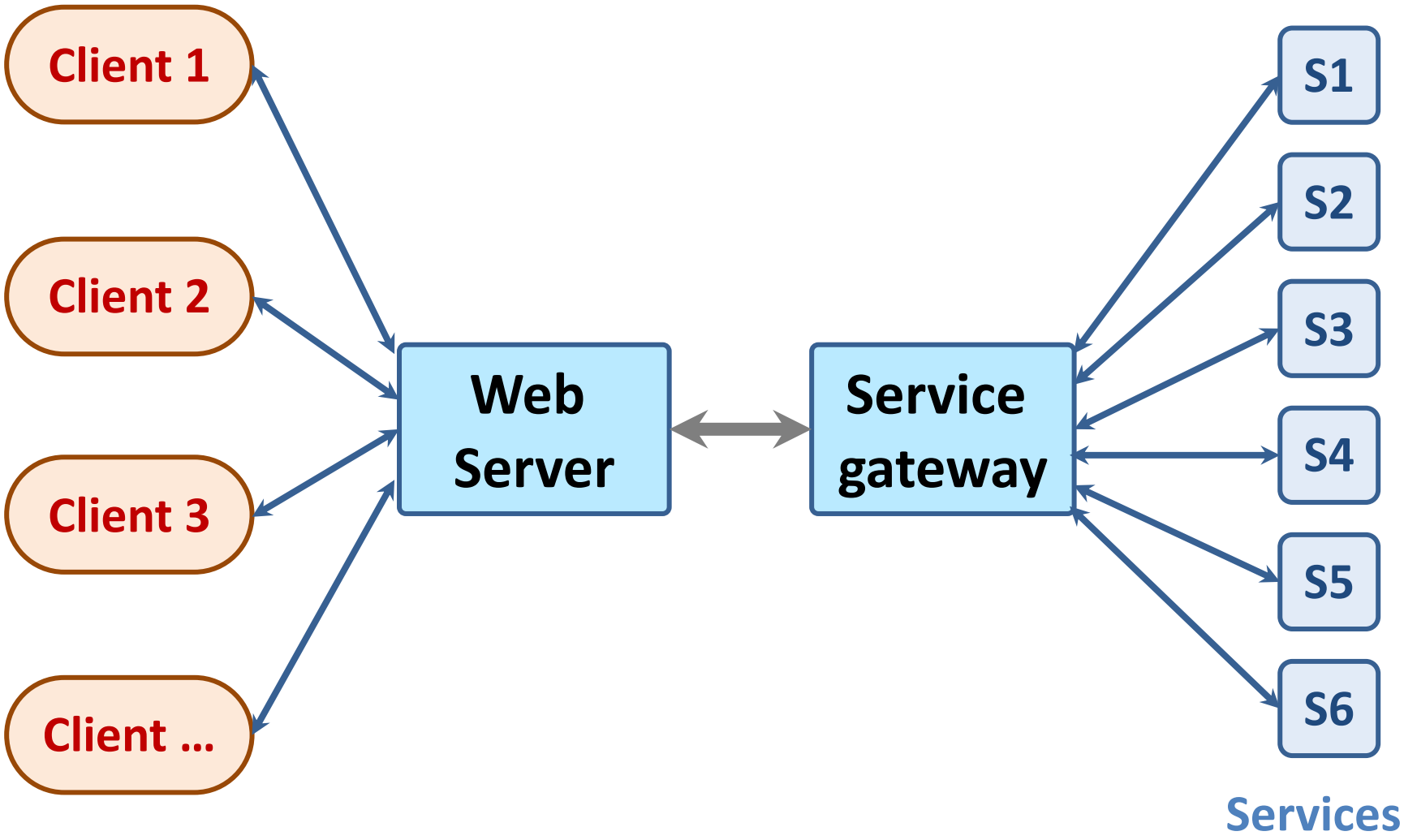
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Multi-tier client-server architecture

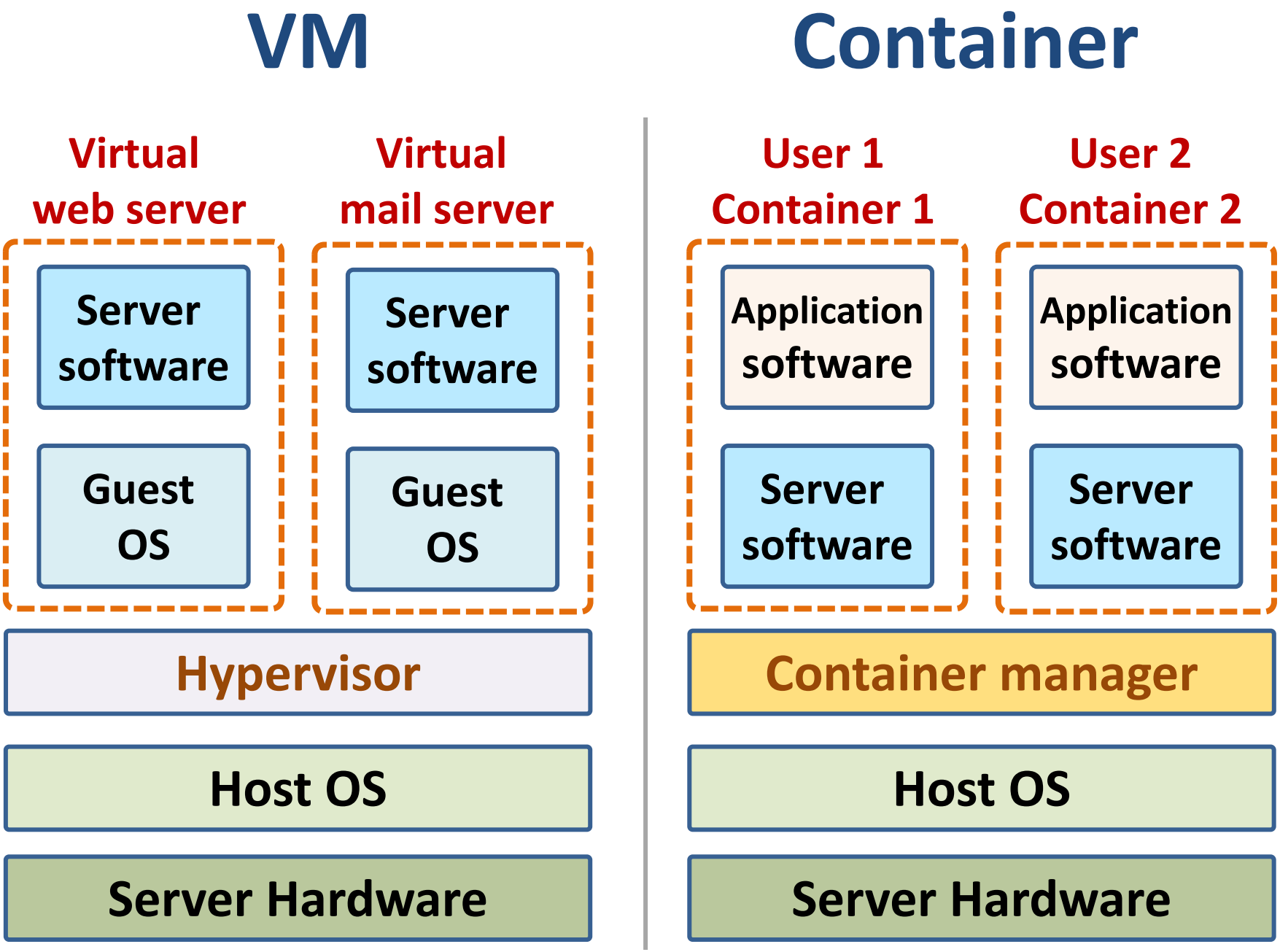


Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Service-oriented Architecture

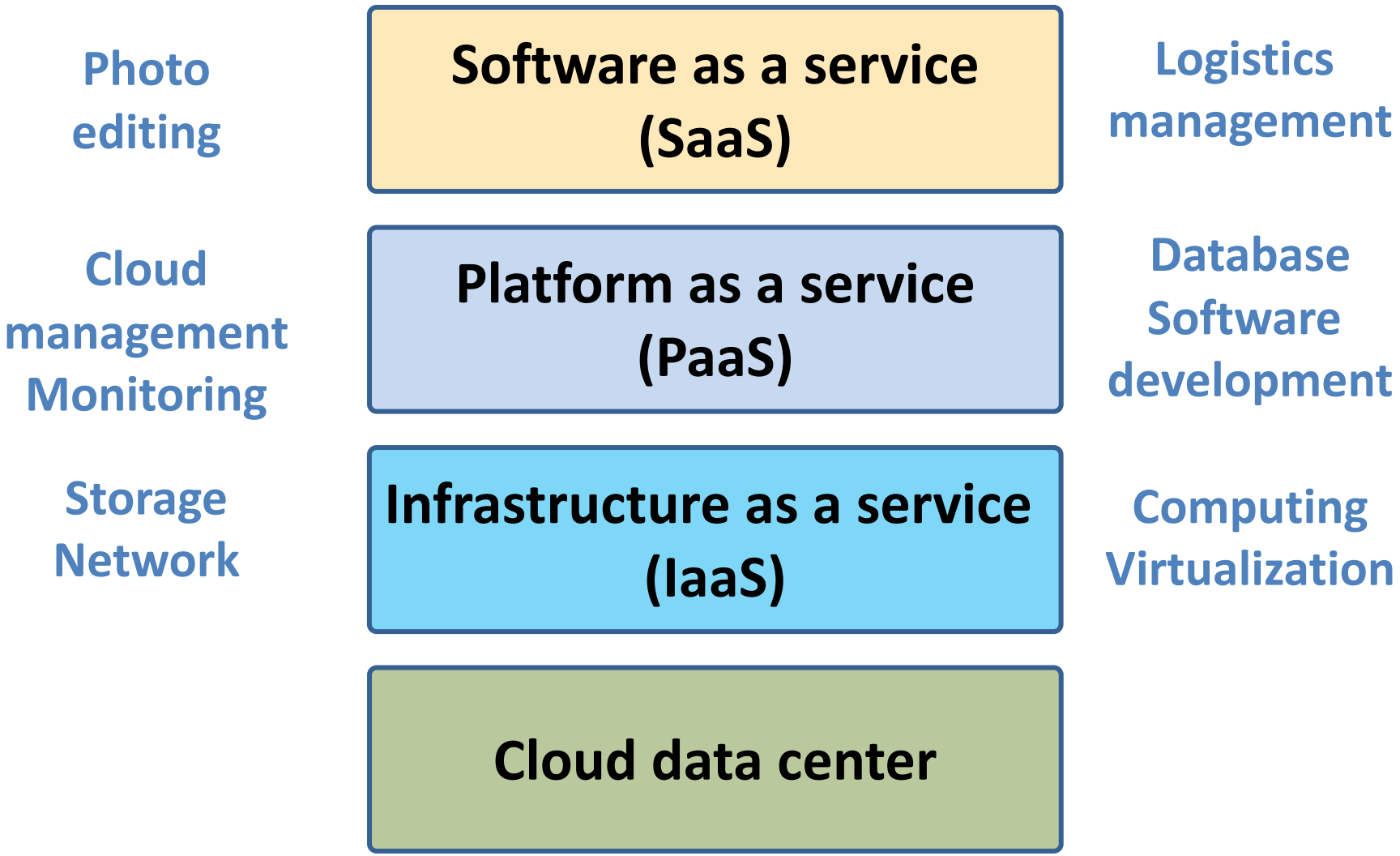


Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.



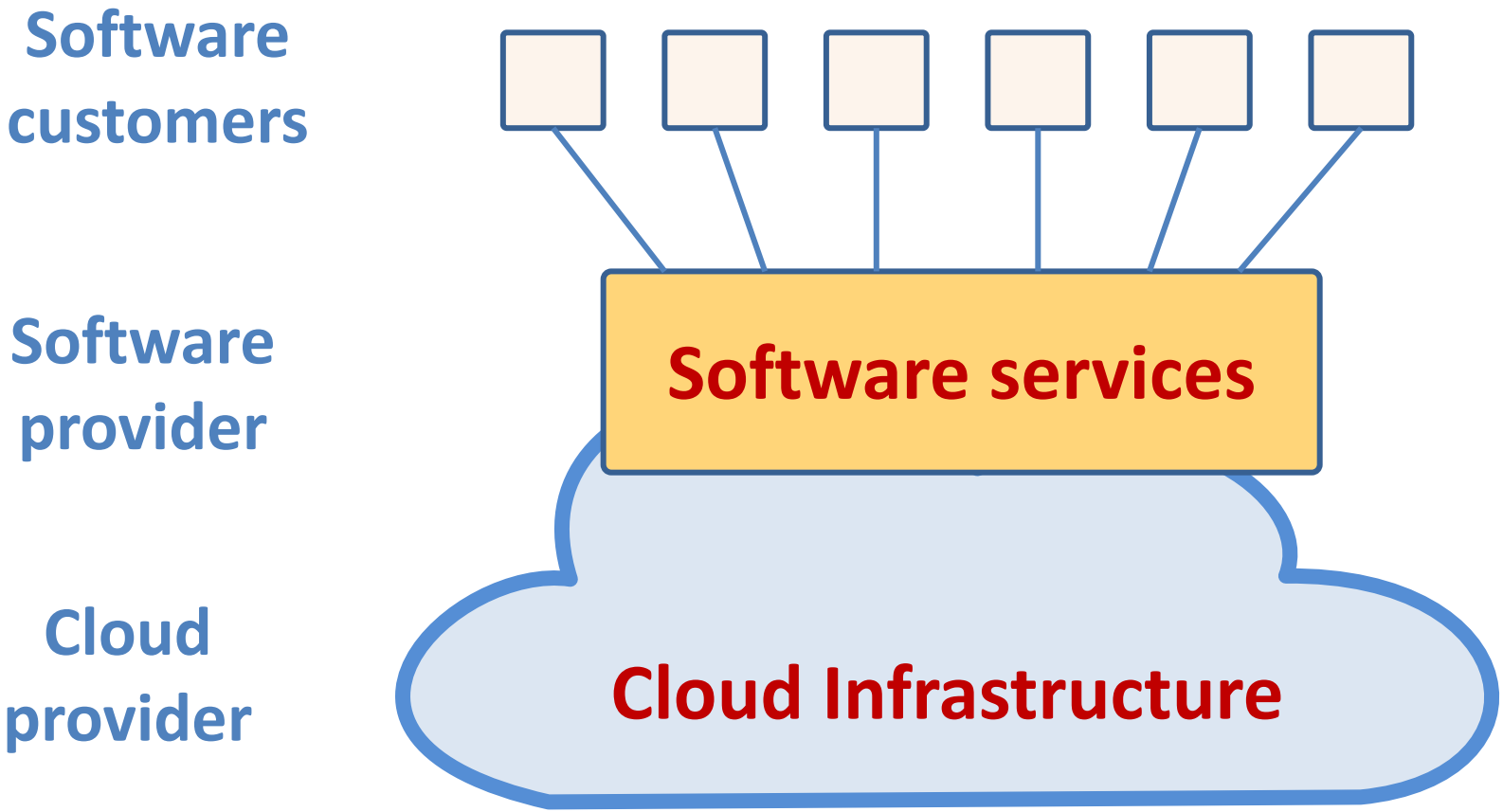
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Everything as a service



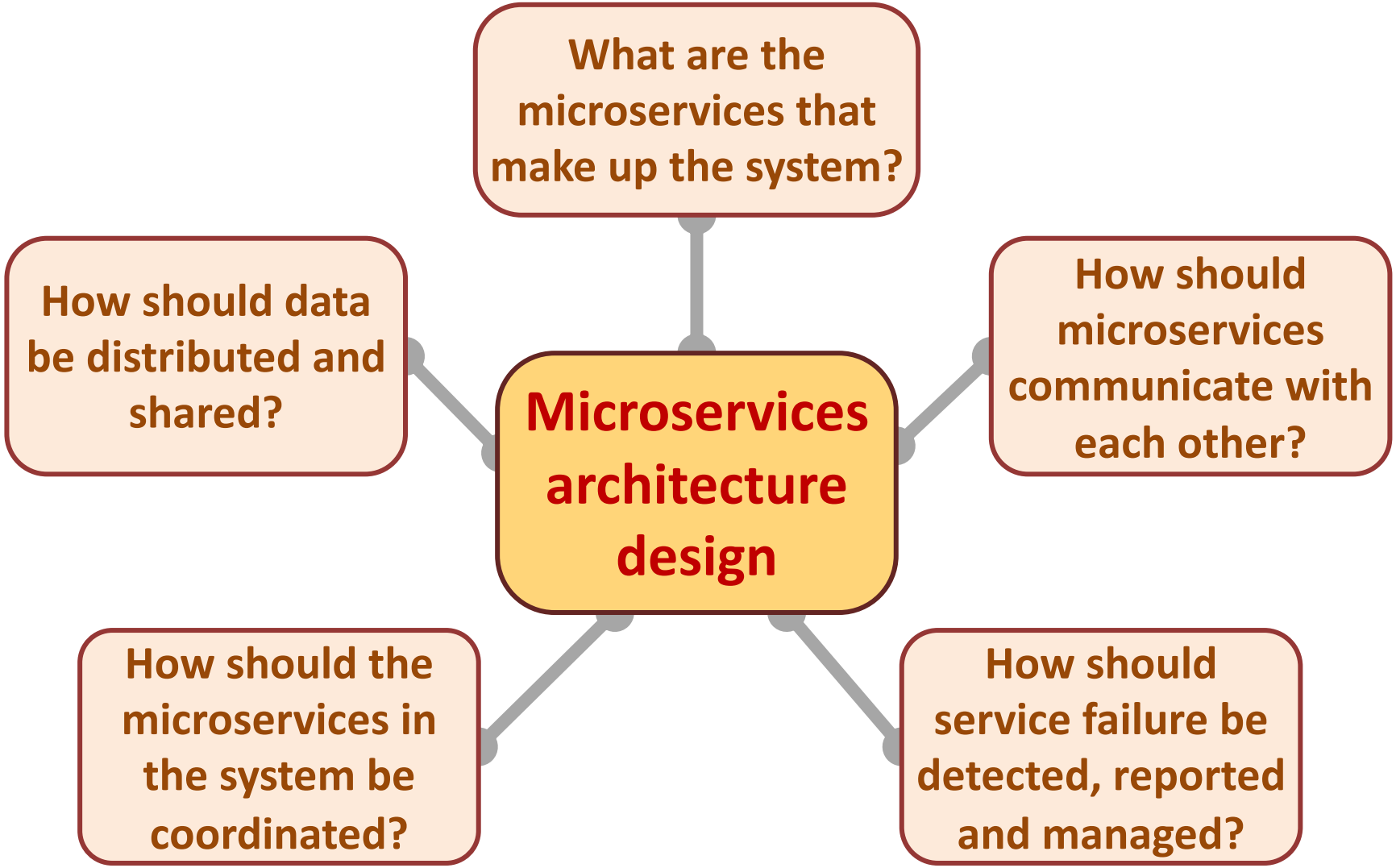
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Software as a service



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Microservices architecture – key design questions



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Security and Privacy

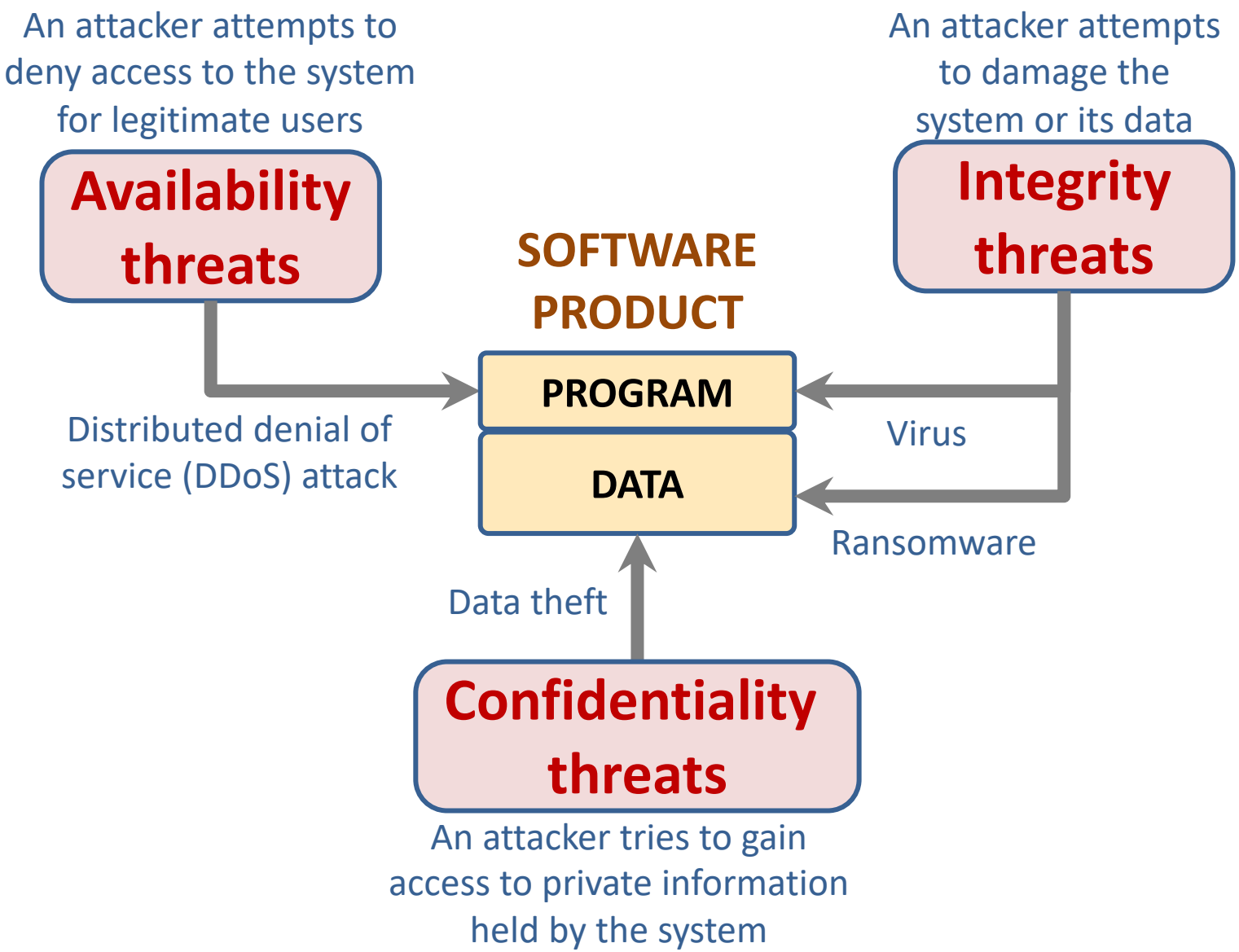
Outline

- **Security**
- **Privacy**

Software security

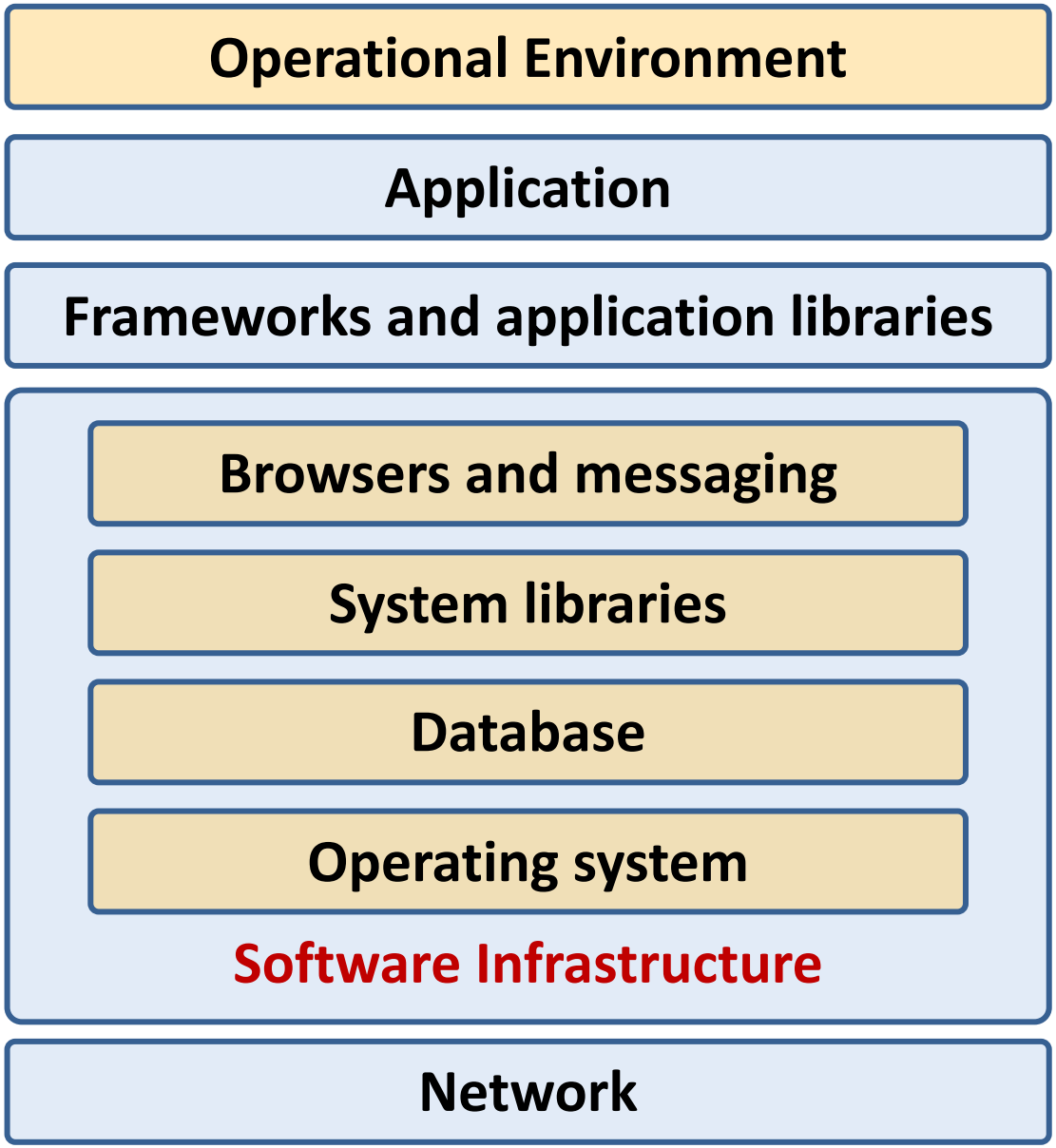
- Software security should always be a high priority for product developers and their users.
- If you don't prioritize security, you and your customers will inevitably suffer losses from malicious attacks.
- In the worst case, these attacks could can put product providers out of business.
  - If their product is unavailable or if customer data is compromised, customers are liable to cancel their subscriptions.
- Even if they can recover from the attacks, this will take time and effort that would have been better spent working on their software.

Types of security threat



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

System infrastructure stack



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Security management

- **Authentication and authorization**

  You should have authentication and authorization standards and procedures that ensure that all users have strong authentication and that they have properly access permissions properly.

- **System infrastructure management**

  Infrastructure software should be properly configured and security updates that patch vulnerabilities should be applied as soon as they become available.

- **Attack monitoring**

  The system should be regularly checked for possible unauthorized access. If attacks are detected, it may be possible to put resistance strategies in place that minimize the effects of the attack.

- **Backup**

  Backup policies should be implemented to ensure that you keep undamaged copies of program and data files. These can then be restored after an attack.

Operational security

- **Operational security** focuses on helping users to maintain security. User attacks try to trick users into disclosing their credentials or accessing a website that includes malware such as a key-logging system.
- **Operational security procedures and practices**
  - **Auto-logout**, which addresses the common problem of users forgetting to logout from a computer used in a shared space.
  - **User command logging**, which makes it possible to discover actions taken by users that have deliberately or accidentally damaged some system resources.
  - **Multi-factor authentication**, which reduces the chances of an intruder gaining access to the system using stolen credentials.

Injection attacks

- Injection attacks are a type of attack where a malicious user uses a valid input field to input malicious code or database commands.

- These malicious instructions are then executed, causing some damage to the system. Code can be injected that leaks system data to the attackers.

- Common types of injection attack include buffer overflow attacks and SQL poisoning attacks.

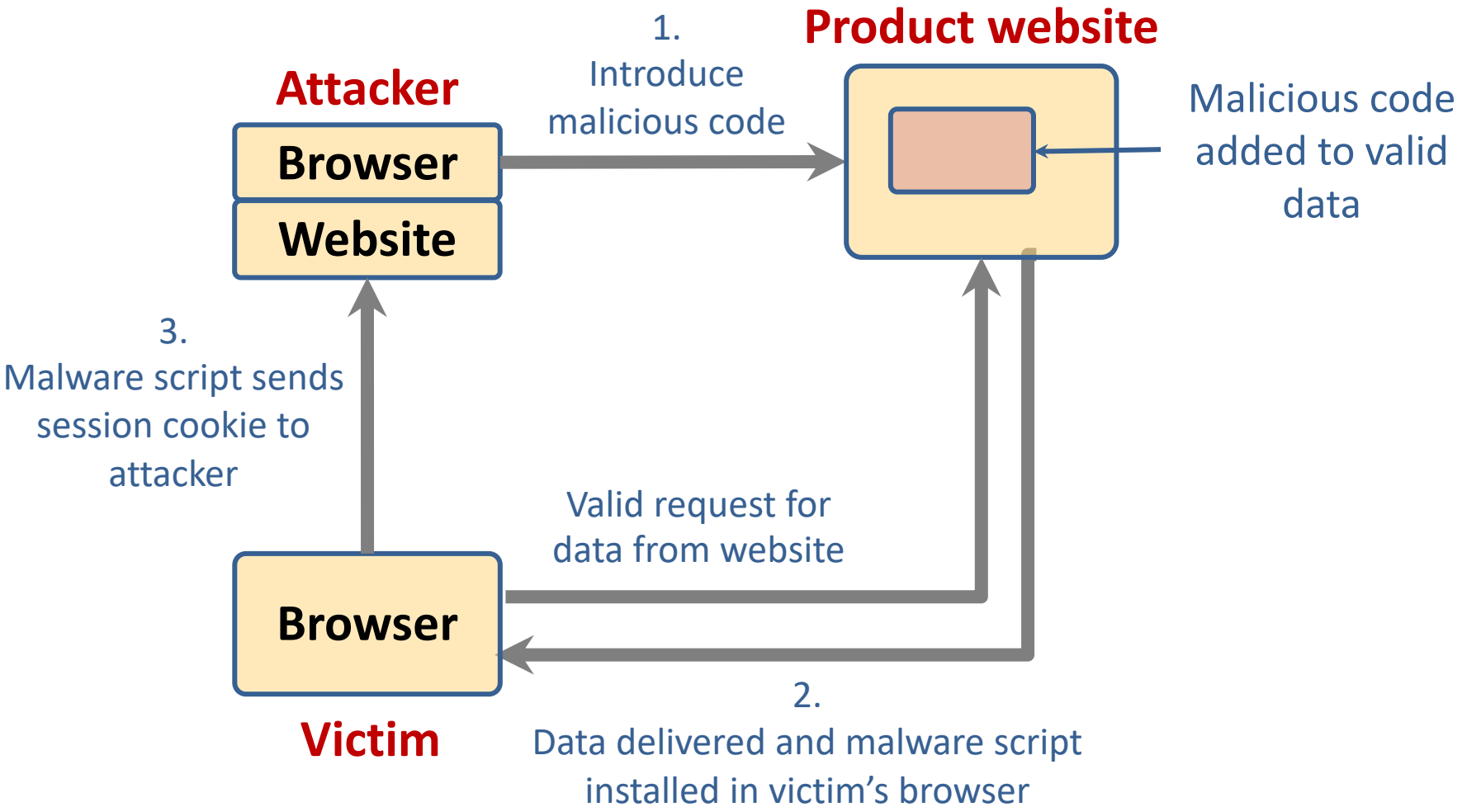
SQL poisoning attacks

- **SQL poisoning attacks** are attacks on software products that use an SQL database.
- They take advantage of a situation where a user input is used as part of an SQL command.
- A malicious user uses a form input field to input a fragment of SQL that allows access to the database.
- The form field is added to the SQL query, which is executed and returns the information to the attacker.

Cross-site scripting attacks

- Cross-site scripting attacks are another form of injection attack.
- An attacker adds malicious Javascript code to the web page that is returned from a server to a client and this script is executed when the page is displayed in the user's browser.
- The malicious script may steal customer information or direct them to another website.
- This may try to capture personal data or display advertisements.
- Cookies may be stolen, which makes a session hijacking attack possible.
- As with other types of injection attack, cross-site scripting attacks may be avoided by input validation.

Cross-site scripting attack



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Session hijacking attacks

- When a user authenticates themselves with a web application, a session is created.
  - A session is a time period during which the user's authentication is valid. They don't have to re-authenticate for each interaction with the system.
  - The authentication process involves placing a session cookie on the user's device
- Session hijacking is a type of attack where an attacker gets hold of a session cookie and uses this to impersonate a legitimate user.

Session hijacking attacks

- There are several ways that an attacker can find out the session cookie value including cross-site scripting attacks and traffic monitoring.
  - In a cross-site scripting attack, the installed malware sends session cookies to the attackers.
  - Traffic monitoring involves attackers capturing the traffic between the client and server. The session cookie can then be identified by analyzing the data exchanged.

Actions to reduce the likelihood of hacking

- **Traffic encryption**

  Always encrypt the network traffic between clients and your server. This means setting up sessions using https rather than http. If traffic is encrypted it is harder to monitor to find session cookies.

- **Multi-factor authentication**

  Always use multi-factor authentication and require confirmation of new actions that may be damaging. For example, before a new payee request is accepted, you could ask the user to confirm their identity by inputting a code sent to their phone.

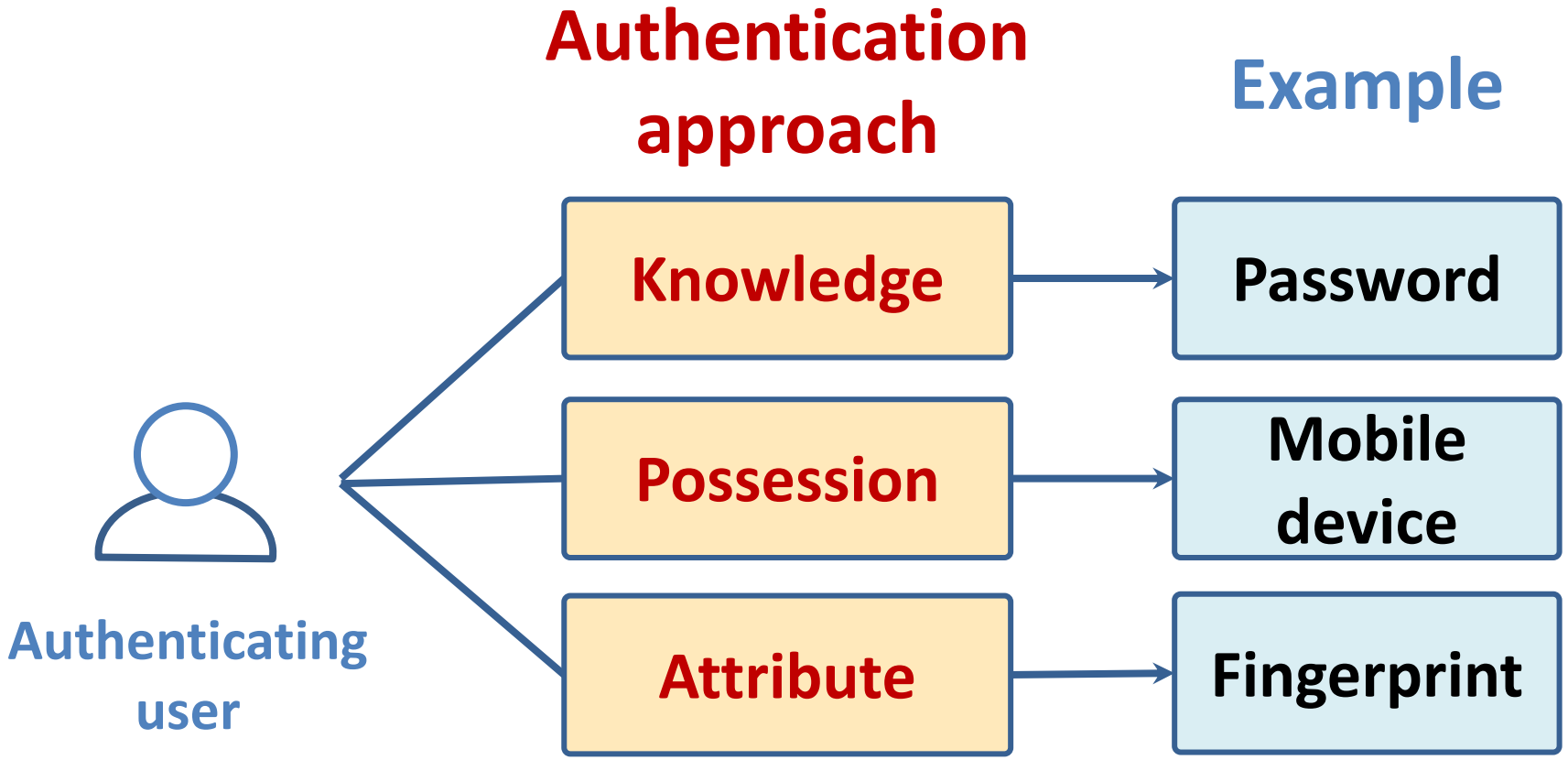
- **Short timeouts**

  Use relatively short timeouts on sessions. If there has been no activity in a session for a few minutes, the session should be ended and future requests directed to an authentication page.

Authentication

- Authentication is the process of ensuring that a user of your system is who they claim to be.
- You need authentication in all software products that maintain user information, so that only the providers of that information can access and change it.
- You also use authentication to learn about your users so that you can personalize their experience of using your product.

Authentication approaches



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Weaknesses of password-based authentication

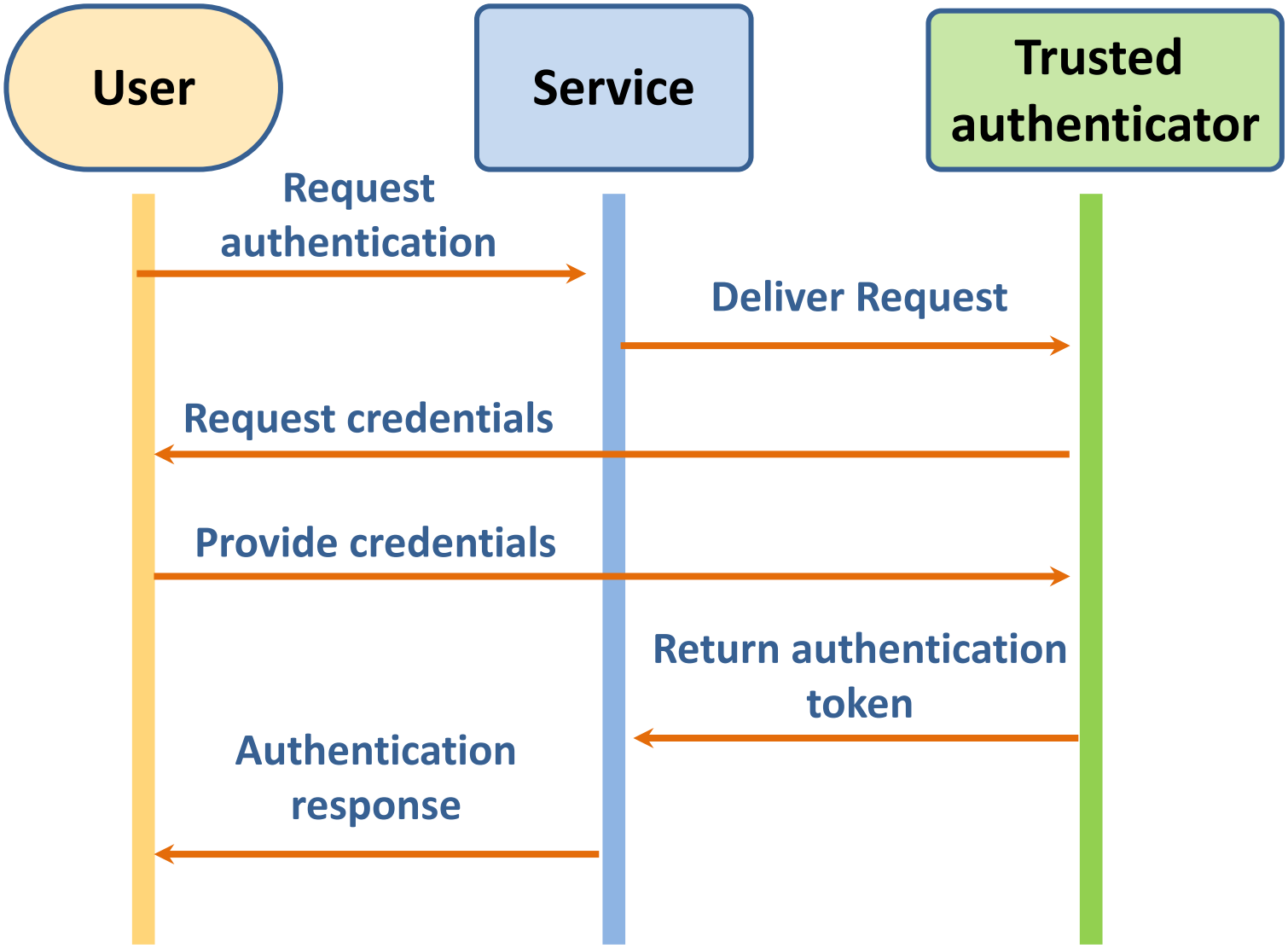
- **Insecure passwords**
  Users choose passwords that are easy to remember.
- **Phishing attacks**
  Users click on an email link that points to a fake site that tries to collect their login and password details.
- **Password reuse**
  Users use the same password for several sites.
- **Forgotten passwords**
  Users regularly forget their passwords so that you need to set up a password recovery mechanism to allow these to be reset.

Federated identity

- **Federated identity** is an approach to authentication where you use an external authentication service.
- **'Login with Google'** and **'Login with Facebook'** are widely used examples of authentication using federated identity.
- The advantage of federated identity for a user is that they have a single set of credentials that are stored by a trusted identity service.
- Instead of logging into a service directly, a user provides their credentials to a known service who confirms their identity to the authenticating service.
- They don't have to keep track of different user ids and passwords.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Federated identity



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Authorization

- Authentication involves a user proving their identity to a software system.
- Authorization is a complementary process in which that identity is used to control access to software system resources.
  - For example, if you use a shared folder on Dropbox, the folder's owner may authorize you to read the contents of that folder, but not to add new files or overwrite files in the folder.
- When a business wants to define the type of access that users get to resources, this is based on an access control policy.
  - This policy is a set of rules that define what information (data and programs) is controlled, who has access to that information and the type of access that is allowed

Access control policies

- **Explicit access control policies** are important for both legal and technical reasons.
- Data protection rules limit the access the personal data and this must be reflected in the defined access control policy.
  - If this policy is incomplete or does not conform to the data protection rules, then there may be subsequent legal action in the event of a data breach.
- Technically, an access control policy can be a starting point for setting up the access control scheme for a system.
- For example, if the access control policy defines the access rights of students, then when new students are registered, they all get these rights by default.

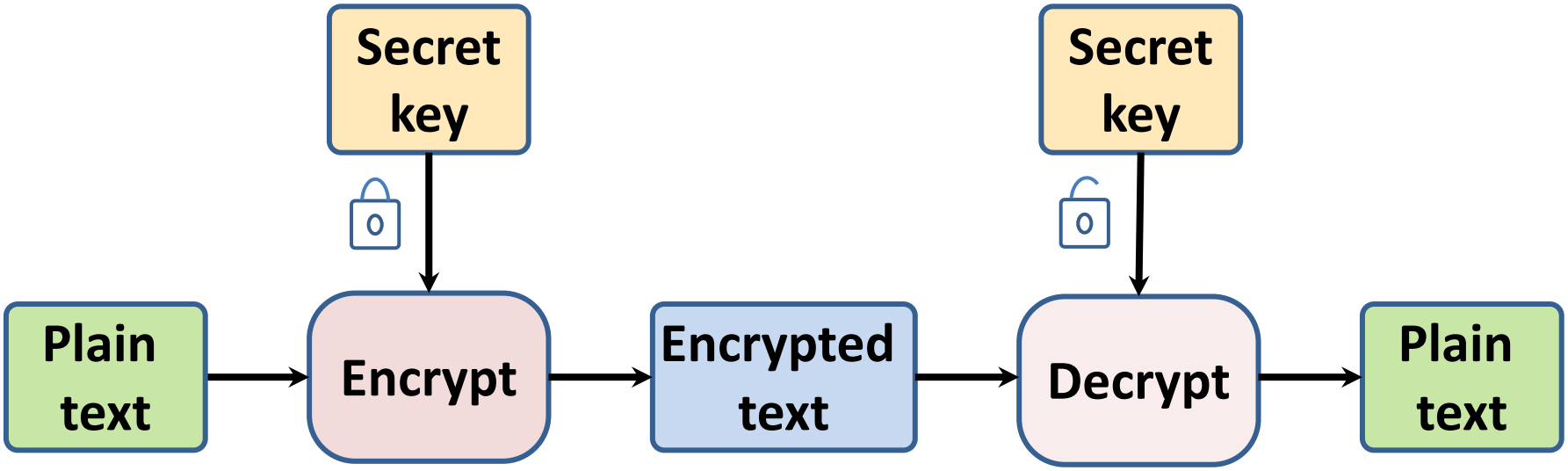
Access Control Lists (ACL)

- Access control lists (ACLs) are used in most file and database systems to implement access control policies.
- Access control lists are tables that link users with resources and specify what those users are permitted to do.
- If access control lists are based on individual permissions, then these can become very large. However, you can dramatically cut their size by allocating users to groups and then assigning permissions to the group

Encryption

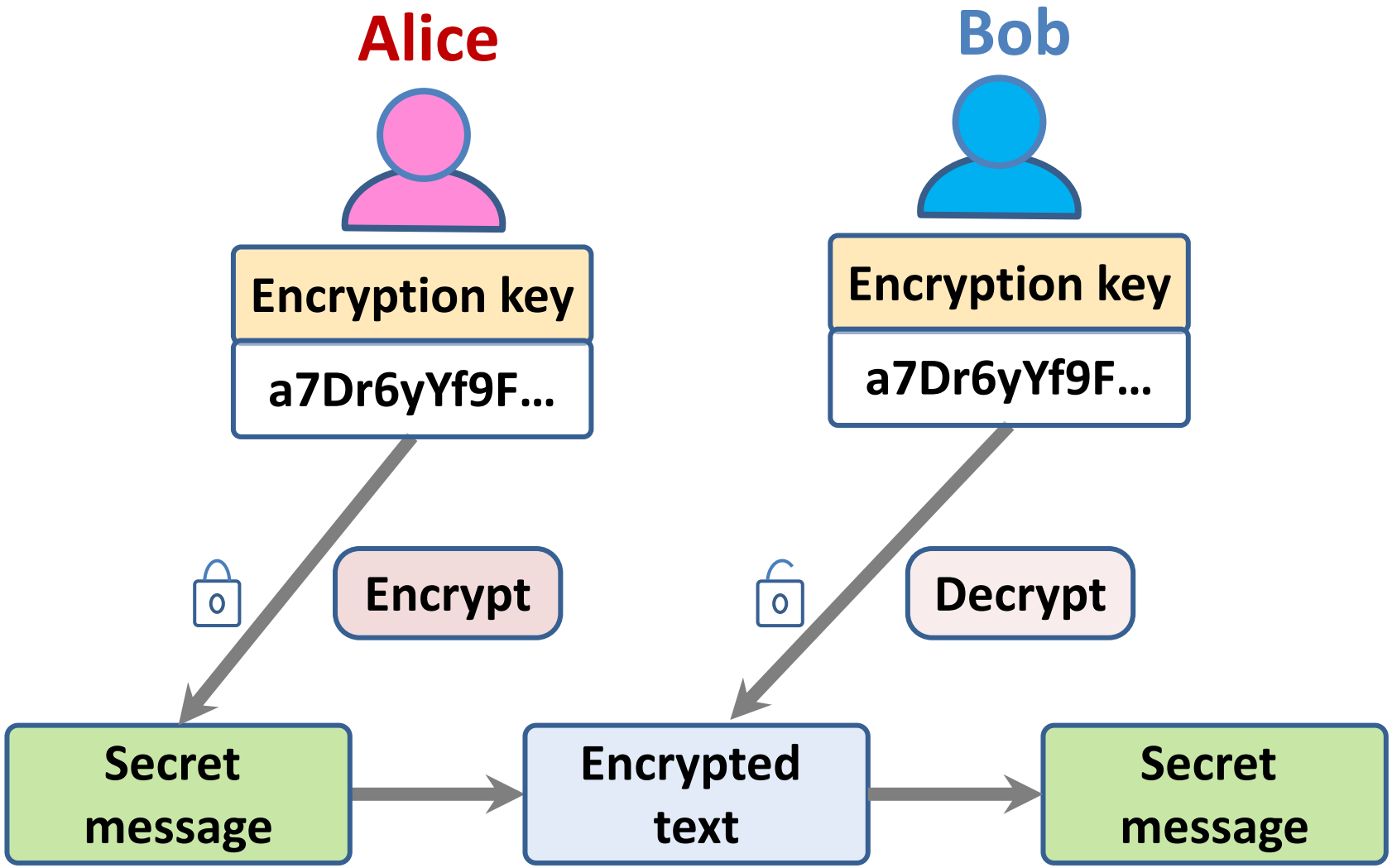
- **Encryption** is the process of making a document unreadable by applying an algorithmic transformation to it.
- A secret key is used by the encryption algorithm as the basis of this transformation. You can decode the encrypted text by applying the reverse transformation.
- Modern encryption techniques are such that you can encrypt data so that it is practically uncrackable using currently available technology.
- History has demonstrated that apparently strong encryption may be crackable when new technology becomes available.
- If commercial quantum systems become available, we will have to use a completely different approach to encryption on the Internet.

Encryption and decryption



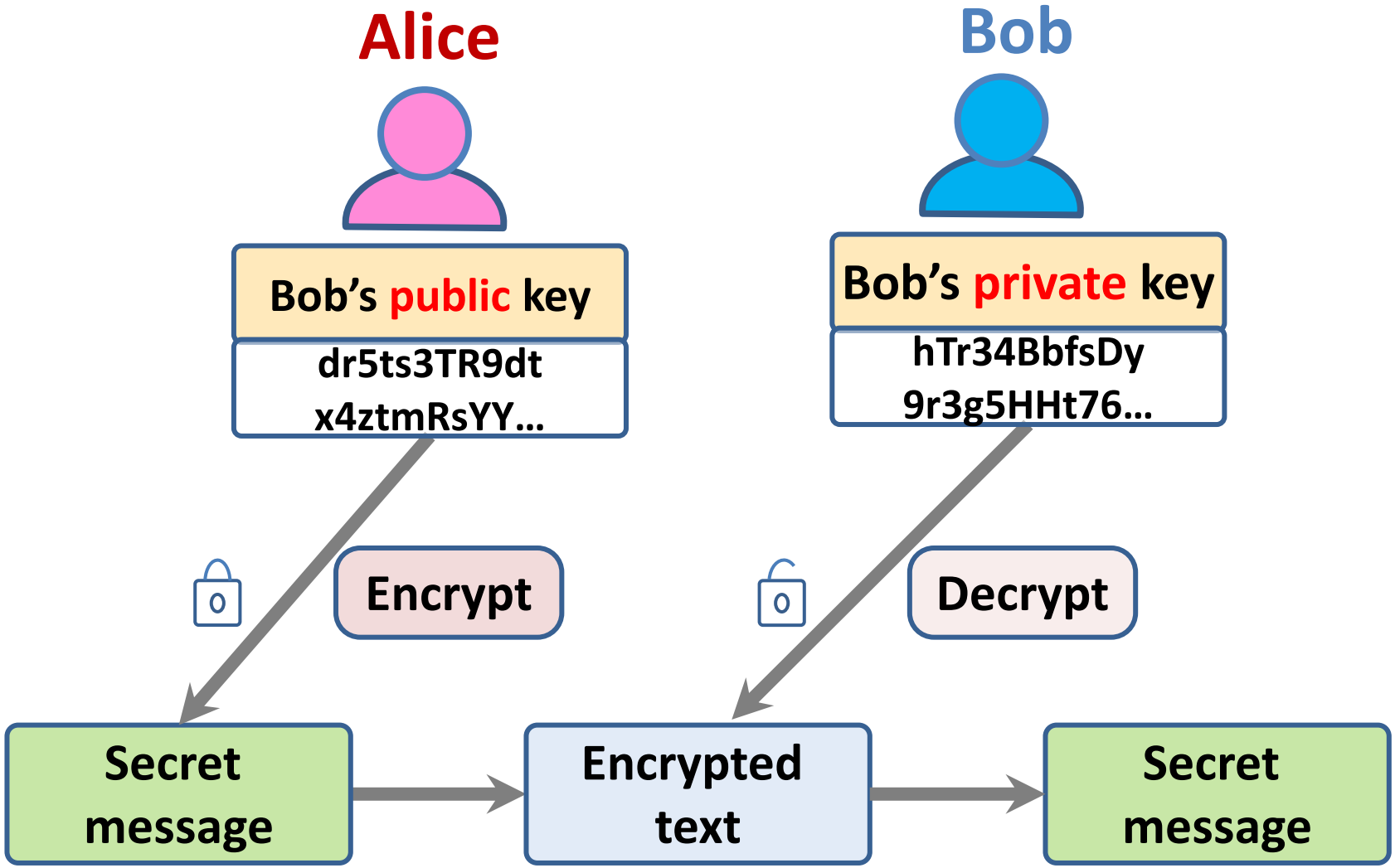
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Symmetric encryption



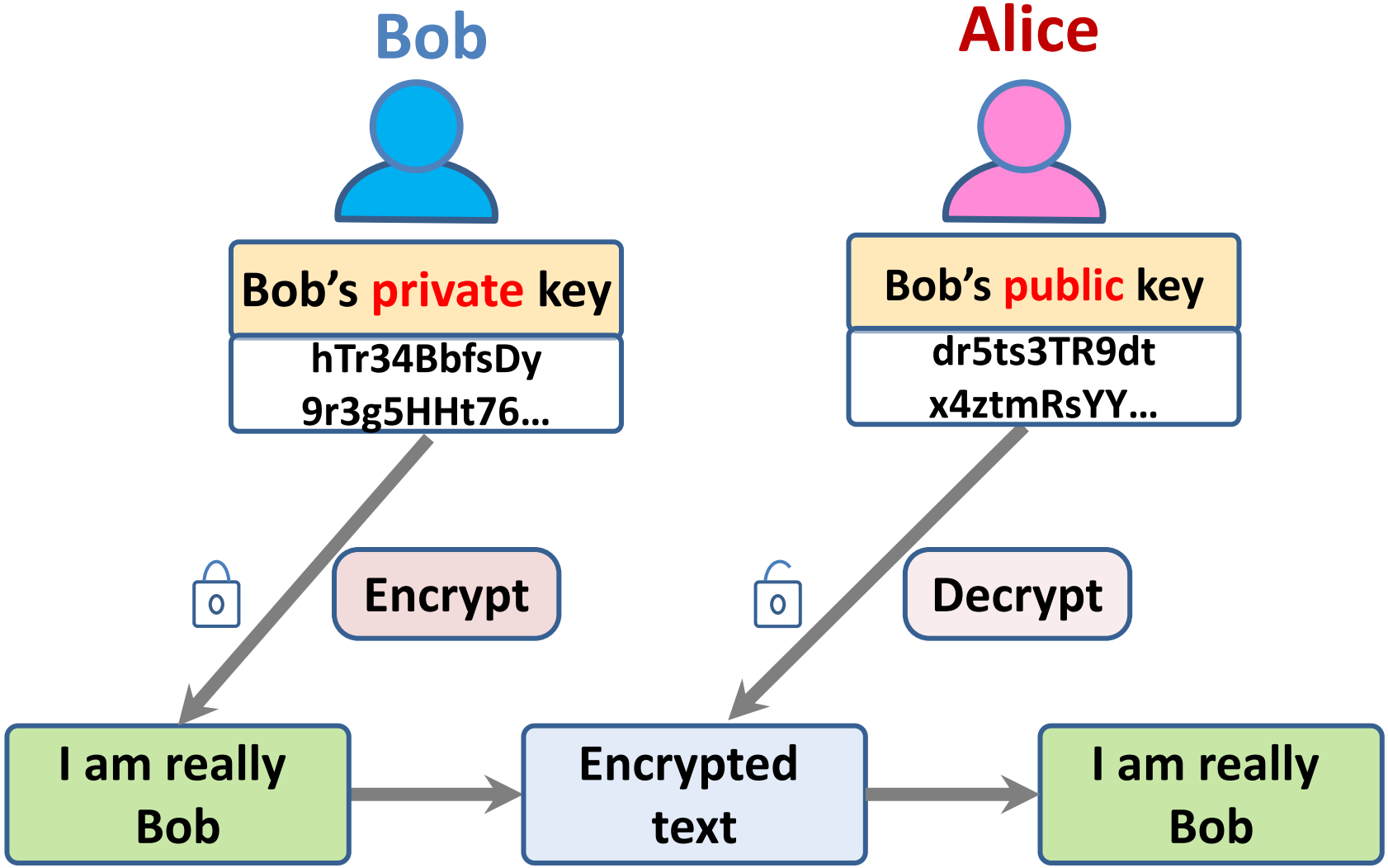
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Asymmetric encryption



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Encryption for authentication



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

TLS and digital certificates

- The https protocol is a standard protocol for securely exchanging texts on the web.
- It is the standard http protocol plus an encryption layer called TLS (Transport Layer Security).
- This encryption layer is used for 2 things:
  - to verify the identity of the web server;
  - to encrypt communications so that they cannot be read by an attacker who intercepts the messages between the client and the server

TLS and digital certificates

- TLS encryption depends on a digital certificate that is sent from the web server to the client.
  - Digital certificates are issued by a **certificate authority (CA)**, which is a trusted identity verification service.
  - The CA encrypts the information in the certificate using their private key to create a unique signature. This signature is included in the certificate along with the public key of the CA. To check that the certificate is valid, you can decrypt the signature using the CA's public key.

Key management

- Key management is the process of ensuring that encryption keys are securely generated, stored and accessed by authorized users.
- Businesses may have to manage tens of thousands of encryption keys so it is impractical to do key management manually and you need to use some kind of automated key management system (KMS).

Digital certificates

- **Subject information**
  Information about the company or individual whose web site is being visited. Applicants apply for a digital certificate from a certificate authority who checks that the applicant is a valid organization.
- **Certificate authority information**
  Information about the certificate authority (CA) who has issued the certificate.
- **Certificate information**
  Information about the certificate itself, including a unique serial number and a validity period, defined by start and end dates.

Digital certificates

- **Digital signature**
  The combination of all of the above data uniquely identifies the digital certificate. The signature data is encrypted with the CA's private key to confirm that the data is correct. The algorithm used to generate the digital signature is also specified.
- **Public key information**
  The public key of the CA is included along with the key size and the encryption algorithm used. The public key may be used to decrypt the digital signature.

Data encryption

- As a product provider you inevitably store information about your users and, for cloud-based products, user data.
- **Encryption** can be used to reduce the damage that may occur from data theft. If information is encrypted, it is impossible, or very expensive, for thieves to access and use the unencrypted data.
  - **Data in transit**
  - **Data at rest**
  - **Data in use**

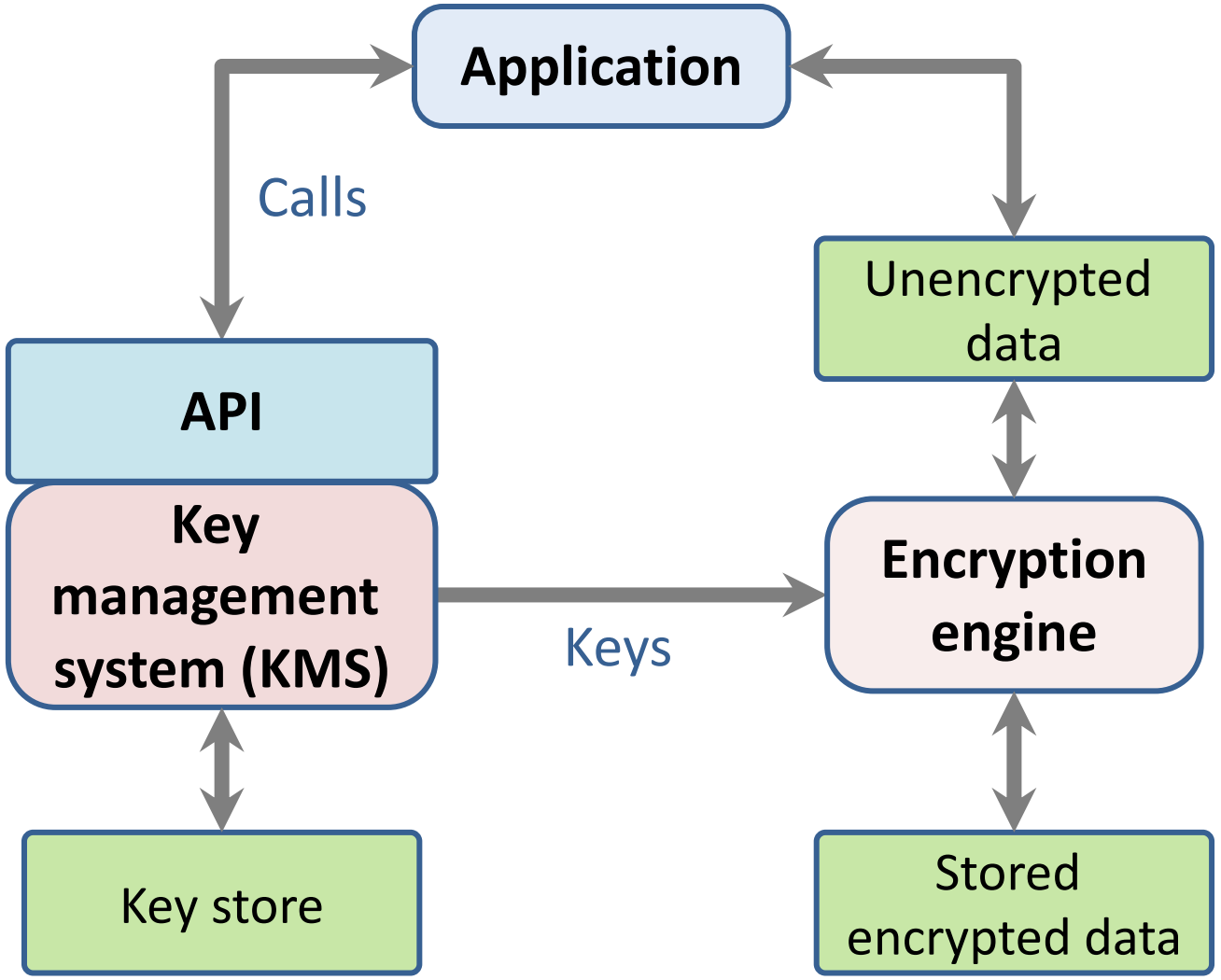
Encryption levels

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Key management

- Key management is important because, if you get it wrong, unauthorized users may be able to access your keys and so decrypt supposedly private data. Even worse, if you lose encryption keys, then your encrypted data may be permanently inaccessible.
- A key management system (KMS) is a specialized database that is designed to securely store and manage encryption keys, digital certificates and other confidential information.

Using a KMS for encryption management



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Long-term key storage

- Business may be required by accounting and other regulations to keep copies of all of their data for several years.
  - For example, in the UK, tax and company data has to be maintained for at least six years, with a longer retention period for some types of data. Data protection regulations may require that this data be stored securely, so the data should be encrypted.
- To reduce the risks of a security breach, encryption keys should be changed regularly. This means that archival data may be encrypted with a different key from the current data in your system.
- Therefore, key management systems must maintain multiple, timestamped versions of keys so that system backups and archives can be decrypted if required.

Privacy

- **Privacy** is a social concept that relates to the collection, dissemination and appropriate use of personal information held by a third-party such as a company or a hospital.
- The importance of privacy has changed over time and individuals have their own views on what degree of privacy is important.
- Culture and age also affect peoples' views on what privacy means.
  - Younger people were early adopters of the first social networks and many of them seem to be less inhibited about sharing personal information on these platforms than older people.
  - In some countries, the level of income earned by an individual is seen as a private matter; in others, all tax returns are openly published.

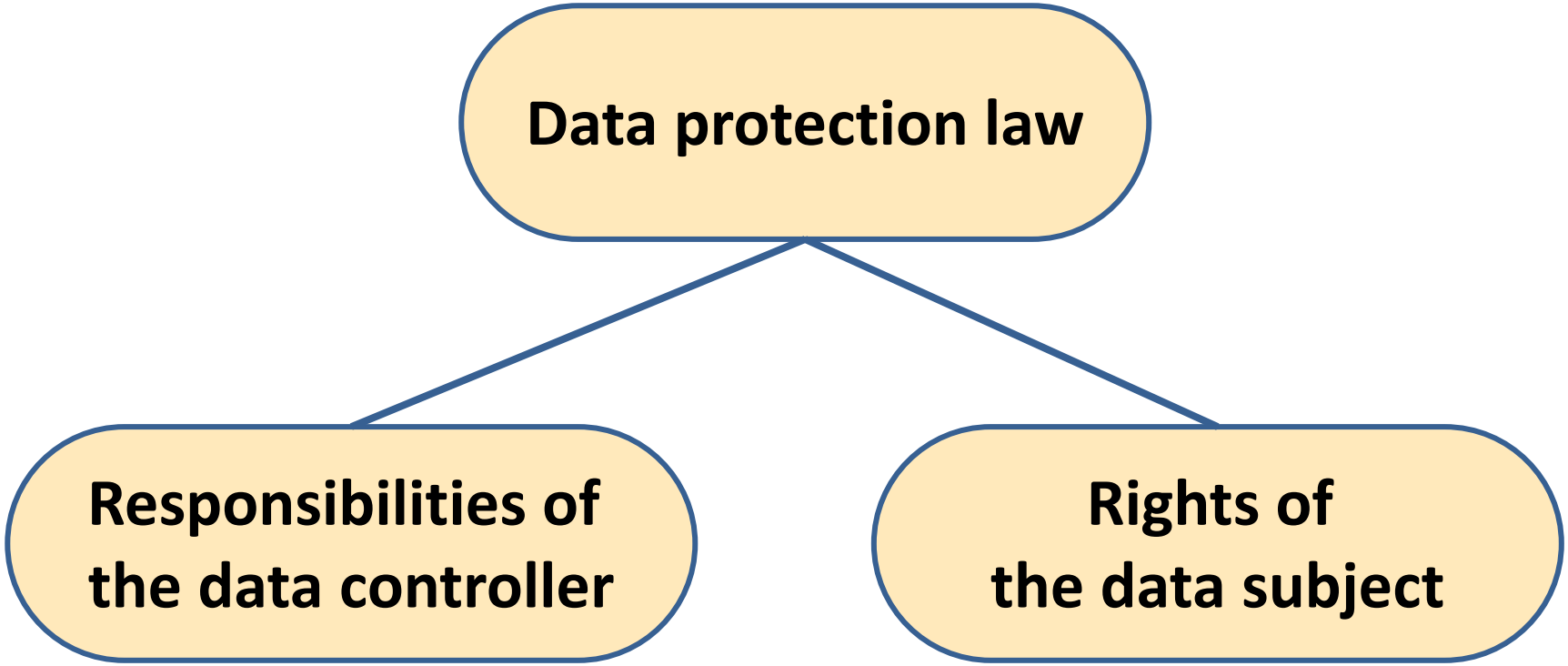
Business reasons for privacy

- If you are offering a product directly to consumers and you fail to conform to privacy regulations, then you may be subject to legal action by product buyers or by a data regulator. If your conformance is weaker than the protection offered by data protection regulations in some countries, you won't be able to sell your product in these countries.
- If your product is a business product, business customers require privacy safeguards so that they are not put at risk of privacy violations and legal action by users.
- If personal information is leaked or misused, even if this is not seen as a violation of privacy regulations, this can lead to serious reputational damage. Customers may stop using your product because of this.

Data protection laws

- In many countries, the right to individual privacy is protected by **data protection laws**.
- These laws limit the collection, dissemination and use of personal data to the purposes for which it was collected.
  - For example, a travel insurance company may collect health information so that they can assess their level of risk. This is legal and permissible.
  - However, it would not be legal for those companies to use this information to target online advertising of health products, unless their users had given specific permission for this.

Data protection laws



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Data protection principles

- **Awareness and control**
  Users of your product must be made aware of what data is collected when they are using your product, and must have control over the personal information that you collect from them.
- **Purpose**
  You must tell users why data is being collected and you must not use that data for other purposes.
- **Consent**
  You must always have the consent of a user before you disclose their data to other people.
- **Data lifetime**
  You must not keep data for longer than you need to. If a user deletes their account, you must delete the personal data associated with that account.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Data protection principles

- **Secure storage**
  You must maintain data securely so that it cannot be tampered with or disclosed to unauthorized people.
- **Discovery and error correction**
  You must allow users to find out what personal data that you store. You must provide a way for users to correct errors in their personal data.
- **Location**
  You must not store data in countries where weaker data protection laws apply unless there is an explicit agreement that the stronger data protection rules will be upheld.

Privacy policy

- You should to establish a **privacy policy** that defines how personal and sensitive information about users is **collected, stored and managed**.
- Software products use data in different ways, so your privacy policy has to define the personal data that you will collect and how you will use that data.
- Product users should be able to review your privacy policy and change their preferences regarding the information that you store.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Privacy policy

- Your **privacy policy** is a **legal document** and it should be auditable to check that it is consistent with the **data protection laws** in countries where your software is sold.
- Privacy policies should not be expressed to users in a long 'terms and conditions' document that, in practice, nobody reads.
- The **General Data Protection Regulation (GDPR)** now require software companies to include a summary of their privacy policy, written in plain language rather than legal jargon, on their website.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Summary

- **Security** is a technical concept that relates to a software system's ability to protect itself from malicious attacks that may threaten its **availability**, the **integrity** of the system and/or its data, and the theft of **confidential** information.
- **Common types of attack on software products** include
  - **injection attacks**,
  - **cross-site scripting attacks**,
  - **session hijacking attacks**,
  - **denial of service attacks and**
  - **brute force attacks.**

Summary

- **Authentication** may be based on something a user knows, something a user has, or some physical attribute of the user.
- **Federated authentication** involves devolving responsibility for authentication to a third-party such as Facebook or Google, or to a business's authentication service.
- **Authorization** involves controlling access to system resources based on the user's authenticated identity. Access control lists are the most commonly-used mechanism to implement authorization.

Summary

- **Symmetric encryption** involves encrypting and decrypting information with the same secret key.
- **Asymmetric encryption** uses a key pair – a private key and a public key. Information encrypted using the public key can only be decrypted using the private key.
- A major issue in symmetric encryption is key exchange.
- The **Transport Layer Security (TLS) protocol**, which is used to secure web traffic, gets around this problem by using asymmetric encryption for transferring information used to generate a shared key.

Summary

- If your product stores sensitive user data, you should encrypt that data when it is not in use.
- A **key management system (KMS)** stores encryption keys. Using a KMS is essential because a business may have to manage thousands or even millions of keys and may have to decrypt historic data that was encrypted using an obsolete encryption key.

Summary

- **Privacy** is a social concept that relates to how people feel about the release of their personal information to others. Different countries and cultures have different ideas on what information should and should not be private.
- **Data protection laws** have been made in many countries to protect individual privacy. They require companies who manage user data to store it securely, to ensure that it is not used or sold without the permission of users, and to allow users to view and correct personal data held by the system.

References

- Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.
- Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.
- Titus Winters, Tom Manshreck, and Hyrum Wright (2020), Software Engineering at Google: Lessons Learned from Programming Over Time, O'Reilly Media.
- Project Management Institute (2017), A Guide to the Project Management Body of Knowledge (PMBOK Guide), Sixth Edition, Project Management Institute
- Project Management Institute (2017), Agile Practice Guide, Project Management Institute