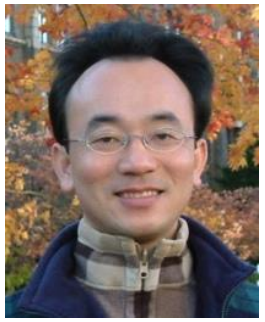
Software Engineering

Software Products and Project Management: Software product management and prototyping with Generative AI

1132SE02
MBA, IM, NTPU (M5010) (Spring 2025)
Wed 2, 3, 4 (9:10-12:00) (B3F17)

https://meet.google.com/ish-gzmy-pmo



Min-Yuh Day, Ph.D,
Professor

Institute of Information Management, National Taipei University

https://web.ntpu.edu.tw/~myday

2025-02-26

Syllabus

Week	Date	Subject/Topics
1	2025/02/19	Introduction to Software Engineering
2	2025/02/26	Software Products and Project Management: Software product management and prototyping with Generative AI
3	2025/03/05	Agile Software Engineering: Agile methods, Scrum, and Extreme Programming
4	2025/03/12	Case Study on Software Engineering I
5	2025/03/19	Features, Scenarios, and Stories
6	2025/03/26	Software Architecture: Architectural design, System decomposition, and Distribution architecture

Syllabus

Week	Date	Subject/Topics
7	2025/04/02	Make-up holiday for NTPU Sports Day (No Classes)
8	2025/04/09	Midterm Project Report
9	2025/04/16	Cloud-Based Software: Virtualization and containers, Everything as a service, Software as a service
10	2025/04/23	Cloud Computing and Cloud Software Architecture
11	2025/04/30	Case Study on Software Engineering II
12	2025/05/07	Microservices Architecture, RESTful services, Service deployment

Syllabus

Week	Date	Subject/Topics
13	2025/05/14	Industry Practices of Software Engineering
14	2025/05/21	Security and Privacy; Reliable Programming; Testing: Functional testing, Test automation, Test-driven development, and Code reviews; DevOps and Code Management: Code management and DevOps automation
15	2025/05/28	Final Project Report I
16	2025/06/04	Final Project Report II

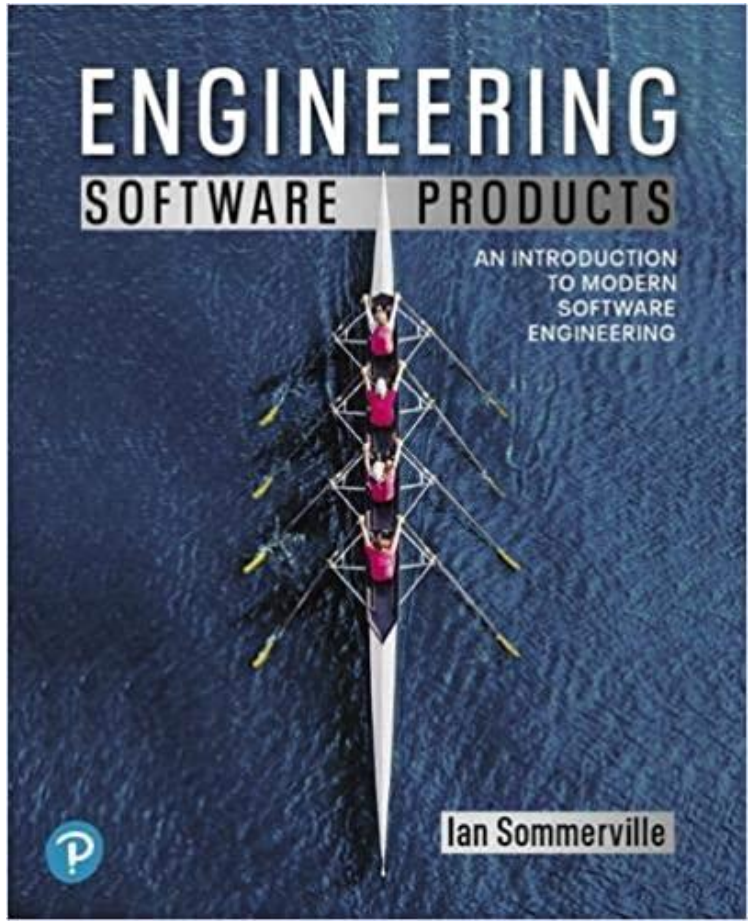
Software Products
and
Project Management:
Software product management
and
prototyping with Generative AI

Ian Sommerville (2019),

Engineering Software Products:

An Introduction to Modern Software Engineering,

Pearson.

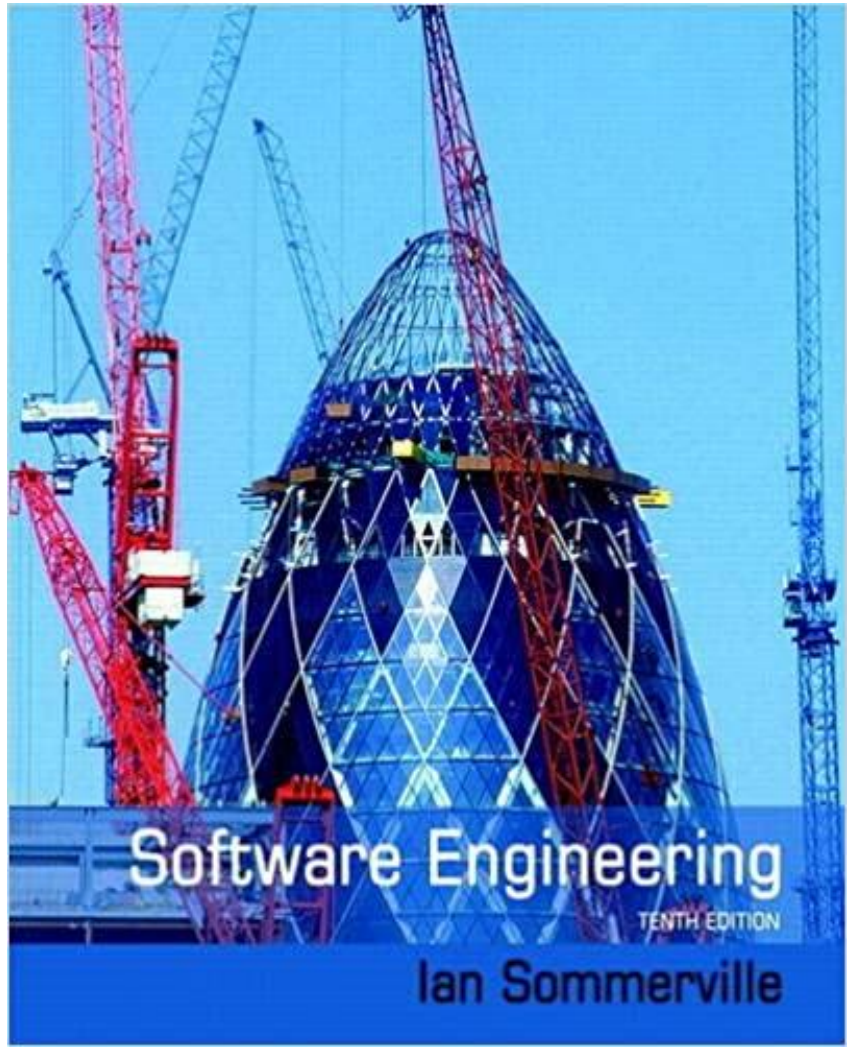


Source: https://www.amazon.com/Engineering-Software-Products-Ian-Sommerville/dp/013521064X

Ian Sommerville (2015),

Software Engineering,

10th Edition, Pearson.



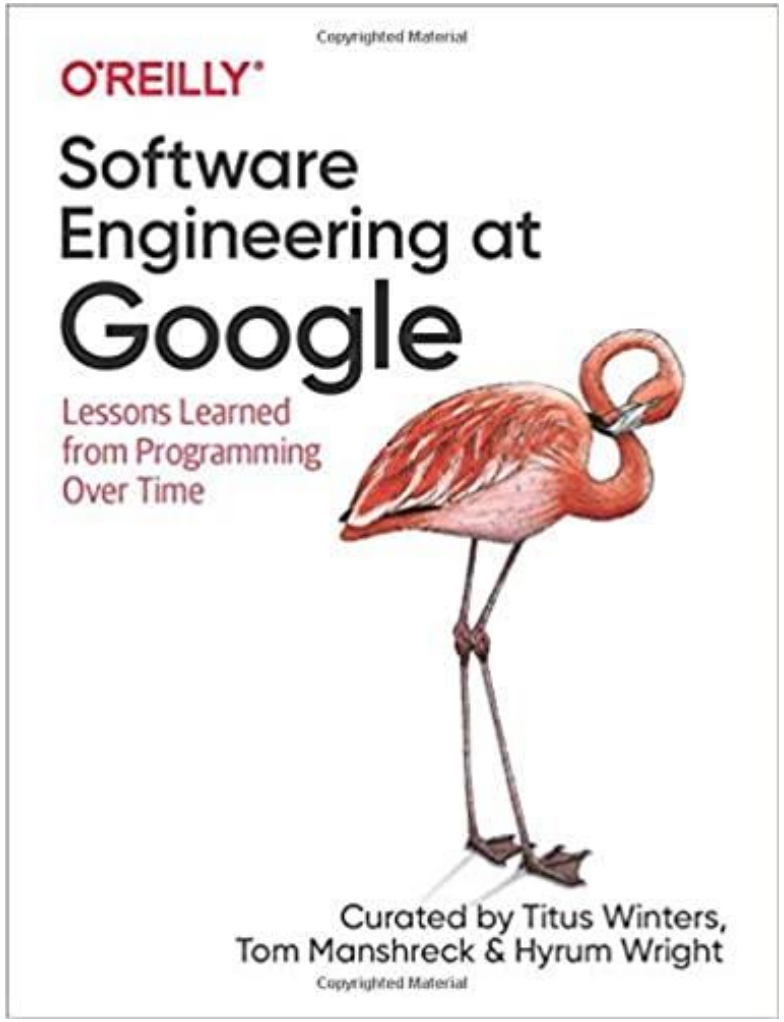
Source: https://www.amazon.com/Software-Engineering-10th-Ian-Sommerville/dp/0133943038

Titus Winters, Tom Manshreck, and Hyrum Wright (2020),

Software Engineering at Google:

Lessons Learned from Programming Over Time,

O'Reilly Media.

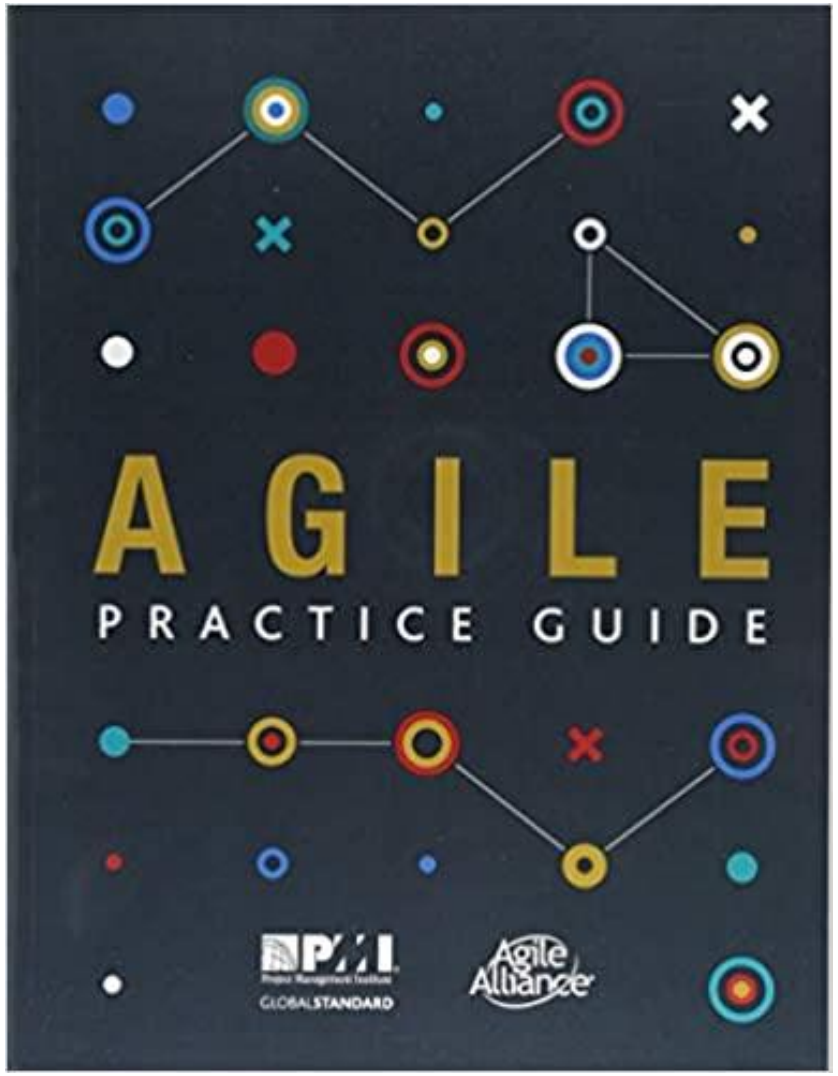


Source: https://www.amazon.com/Software-Engineering-Google-Lessons-Programming/dp/1492082791

Project Management Institute (2017),

Agile Practice Guide

PMI

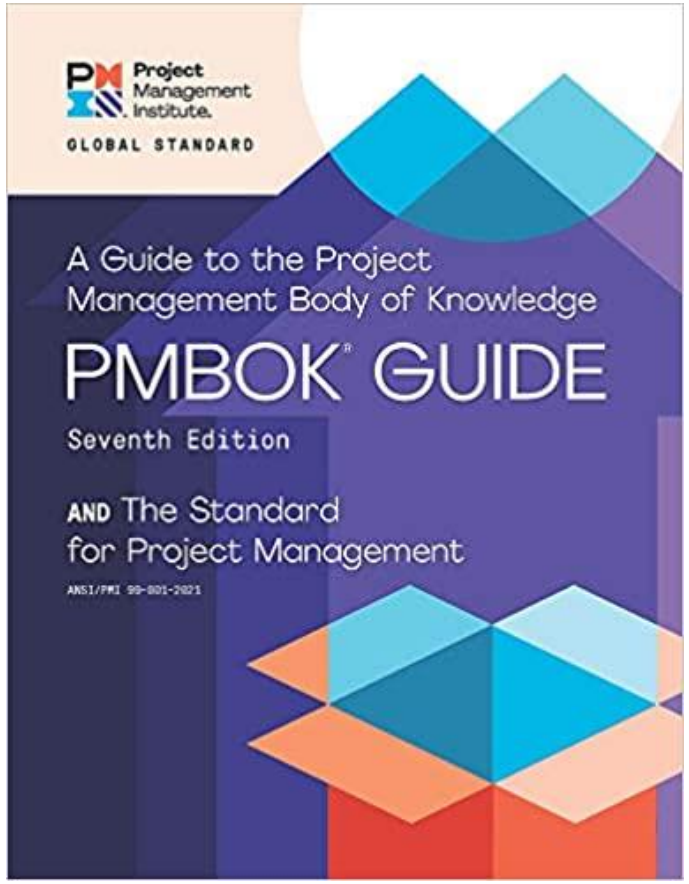


Source: https://www.amazon.com/Agile-Practice-Project-Management-Institute/dp/1628251999/

Project Management Institute (2021),

A Guide to the Project Management Body of Knowledge (PMBOK Guide) –

Seventh Edition and The Standard for Project Management



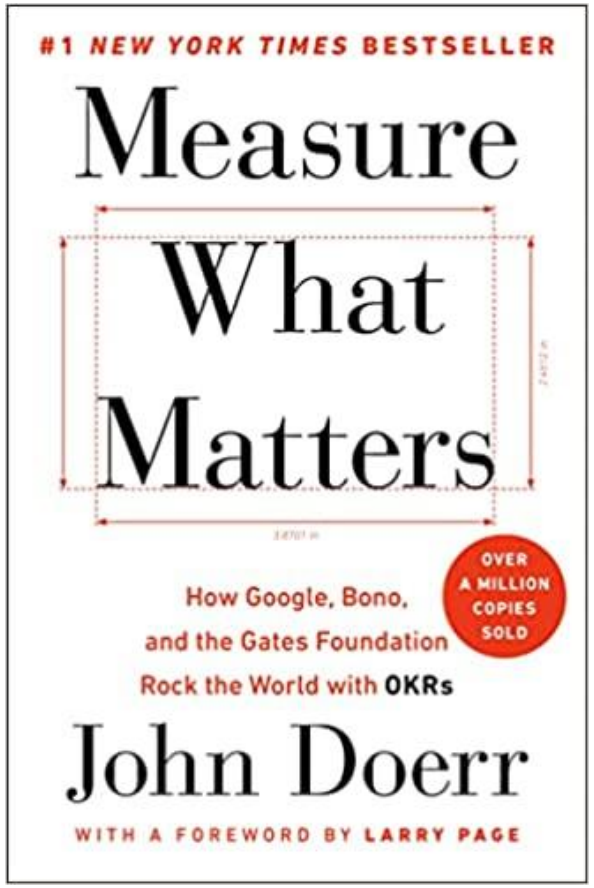
Source: https://www.amazon.com/Guide-Project-Management-Knowledge-PMBOK%C2%AE/dp/1628256648

John Doerr (2018),

Measure What Matters:

How Google, Bono, and the Gates Foundation Rock the World with OKRs,

Portfolio

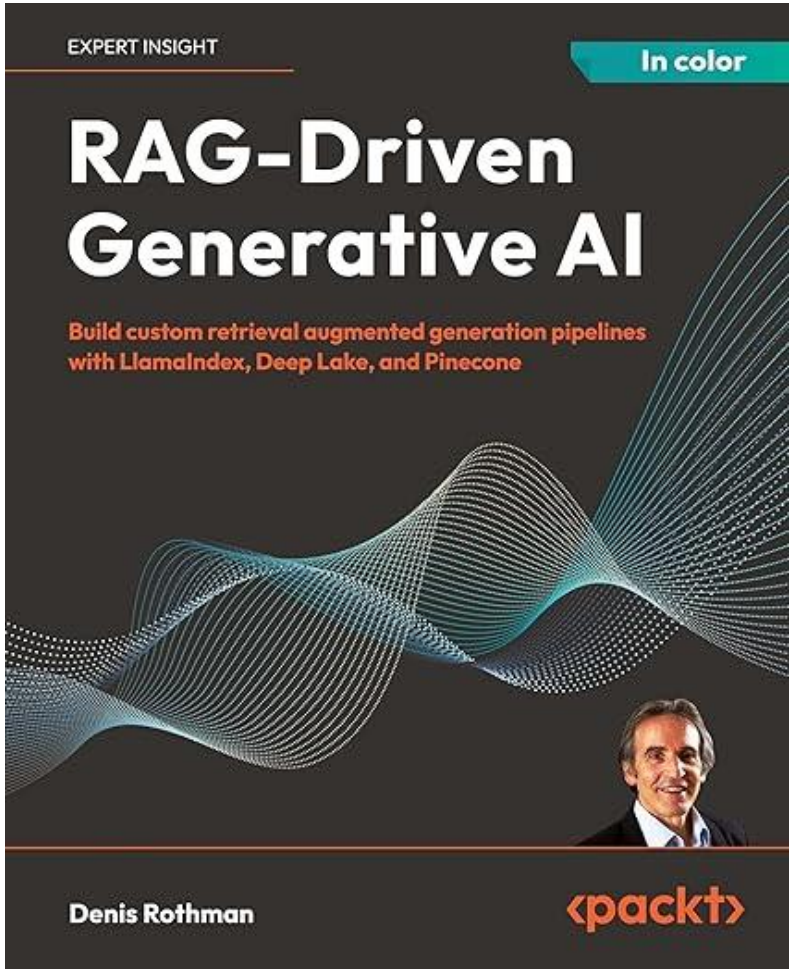


Source: https://www.amazon.com/Measure-What-Matters-Google-Foundation/dp/0525536221

Denis Rothman (2024),

RAG-Driven Generative AI:

Build custom retrieval augmented generation pipelines with LlamaIndex, Deep Lake, and Pinecone, Packt Publishing



Source: https://www.amazon.com/RAG-Driven-Generative-retrieval-generation-LlamaIndex/dp/1836200919/

NVIDIA Developer Program

https://developer.nvidia.com/join-nvidia-developer-program

NVIDIA Deep Learning Institute (DLI)

https://learn.nvidia.com/

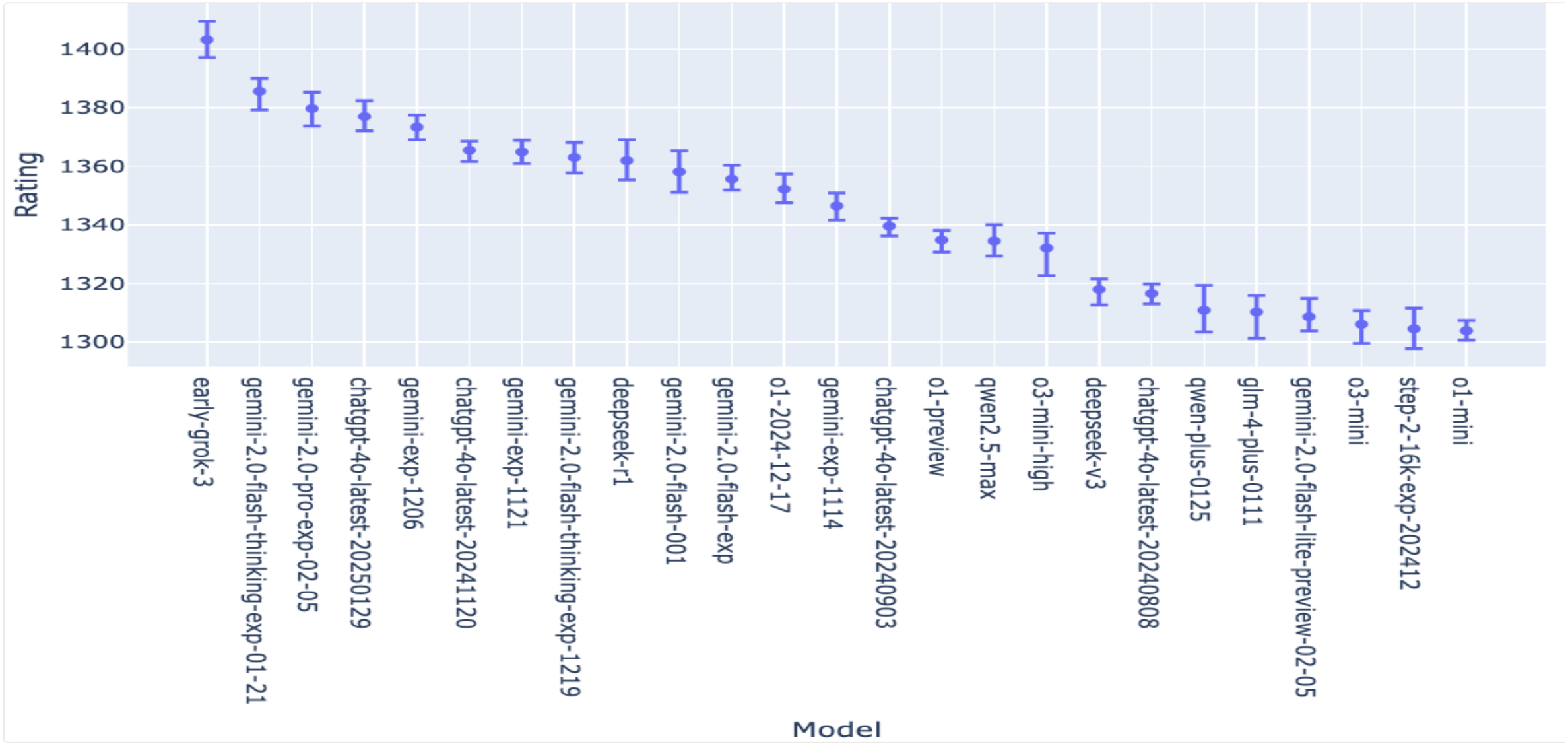
lmarena.ai Chatbot Arena Leaderboard

Rank* (UB)	Rank (StyleCtrl)	Model	Arena Score	95% CI	Votes	Organization	License
1	1	chocolate (Early Grok-3)	1403	+6/-6	9992	xAI	Proprietary
2	3	Gemini-2.0-Flash-Thinking-Exp-01-21	1385	+4/-6	15083	Google	Proprietary
2	3	Gemini-2.0-Pro-Exp-02-05	1380	+5/-6	13000	Google	Proprietary
2	1	ChatGPT-4o-latest (2025-01-29)	1377	+5/-5	13470	OpenAI	Proprietary
5	3	DeepSeek-R1	1362	+7/-7	6581	DeepSeek	MIT
5	8	Gemini-2.0-Flash-001	1358	+7/-7	10862	Google	Proprietary
5	3	o1-2024-12-17	1352	+5/-5	17248	OpenAI	Proprietary
8	7	o1-preview	1335	+3/-4	33169	OpenAI	Proprietary
8	8	Qwen2.5-Max	1334	+5/-5	9282	Alibaba	Proprietary
8	7	o3-mini-high	1332	+5/-9	5954	OpenAI	Proprietary
11	11	DeepSeek-V3	1318	+4/-5	19461	DeepSeek	DeepSeek
11	13	Qwen-Plus-0125	1311	+9/-7	5112	Alibaba	Proprietary
11	14	GLM-4-Plus-0111	1310	+6/-9	5134	Zhipu	Proprietary
11	13	Gemini-2.0-Flash-Lite-Preview-02-05	1309	+6/-5	10262	Google	Proprietary
12	12	o3-mini	1306	+5/-6	12179	OpenAI	Proprietary

https://huggingface.co/spaces/lmarena-ai/chatbot-arena-leaderboard

lmarena.ai Chatbot Arena Leaderboard

Confidence Intervals on Model Strength (via Bootstrapping)



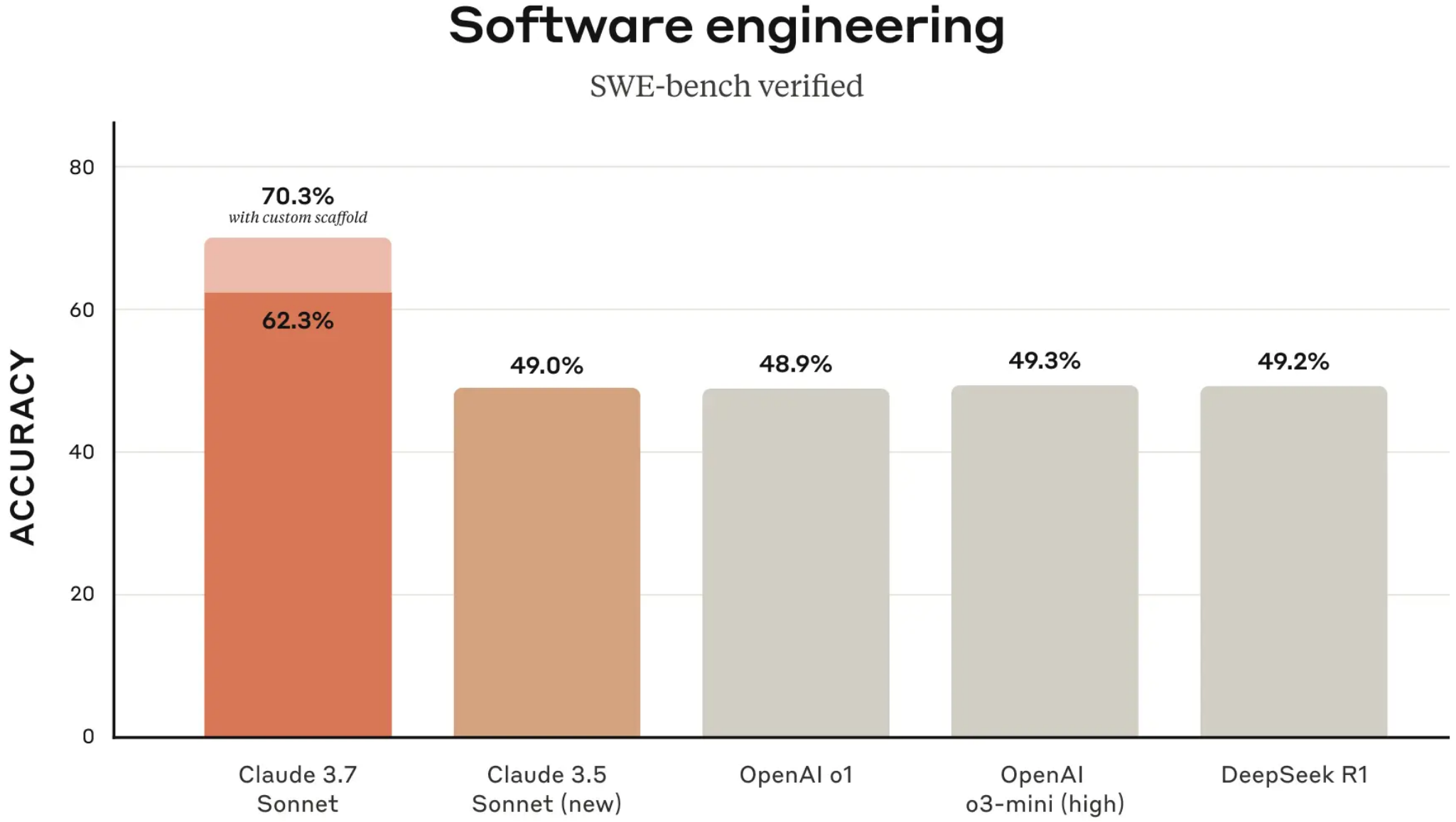
https://huggingface.co/spaces/lmarena-ai/chatbot-arena-leaderboard

Claude 3.7 Sonnet, Claude 3.5 Sonnet, OpenAI, DeepSeek, and Grok

	Claude 3.7 Sonnet *64K extended thinking*	Claude 3.7 Sonnet *No extended thinking*	Claude 3.5 Sonnet (new)	OpenAI o1[1]	OpenAI o3-mini[1] *High*	DeepSeek R1 *32K extended thinking*	Grok 3 Beta *Extended thinking*
Graduate-level reasoning *GPQA Diamond*[3]	78.2% / 84.8%	68.0%	65.0%	75.7% / 78.0%	79.7%	71.5%	80.2% / 84.6%
Agentic coding *SWE-bench Verified*[2]	—	62.3% / 70.3%	49.0%	48.9%	49.3%	49.2%	—
Agentic tool use *TAU-bench*	—	Retail 81.2%	Retail 71.5%	Retail 73.5%	—	—	—
	—	Airline 58.4%	Airline 48.8%	Airline 54.2%	—	—	—
Multilingual Q&A *MMMLU*	86.1%	83.2%	82.1%	87.7%	79.5%	—	—
Visual reasoning *MMMU (validation)*	75%	71.8%	70.4%	78.2 %	—	—	76.0% / 78.0%
Instruction-following *IFEval*	93.2%	90.8%	90.2%	—	—	83.3%	—
Math problem-solving *MATH 500*	96.2%	82.2%	78.0%	96.4%	97.9%	97.3%	—
High school math competition *AIME 2024*[3]	61.3% / 80.0%	23.3%	16.0%	79.2% / 83.3%	87.3%	79.8%	83.9% / 93.3%

Source: https://www.anthropic.com/news/claude-3-7-sonnet

Claude 3.7 Sonnet and Claude Code



Source: https://www.anthropic.com/news/claude-3-7-sonnet

Generative AI Meets Product Development

Use Case 1:

Enhancing Creativity and Design Workflows

Use Case 2:

GenAI for Customer Insights and Concept Validation

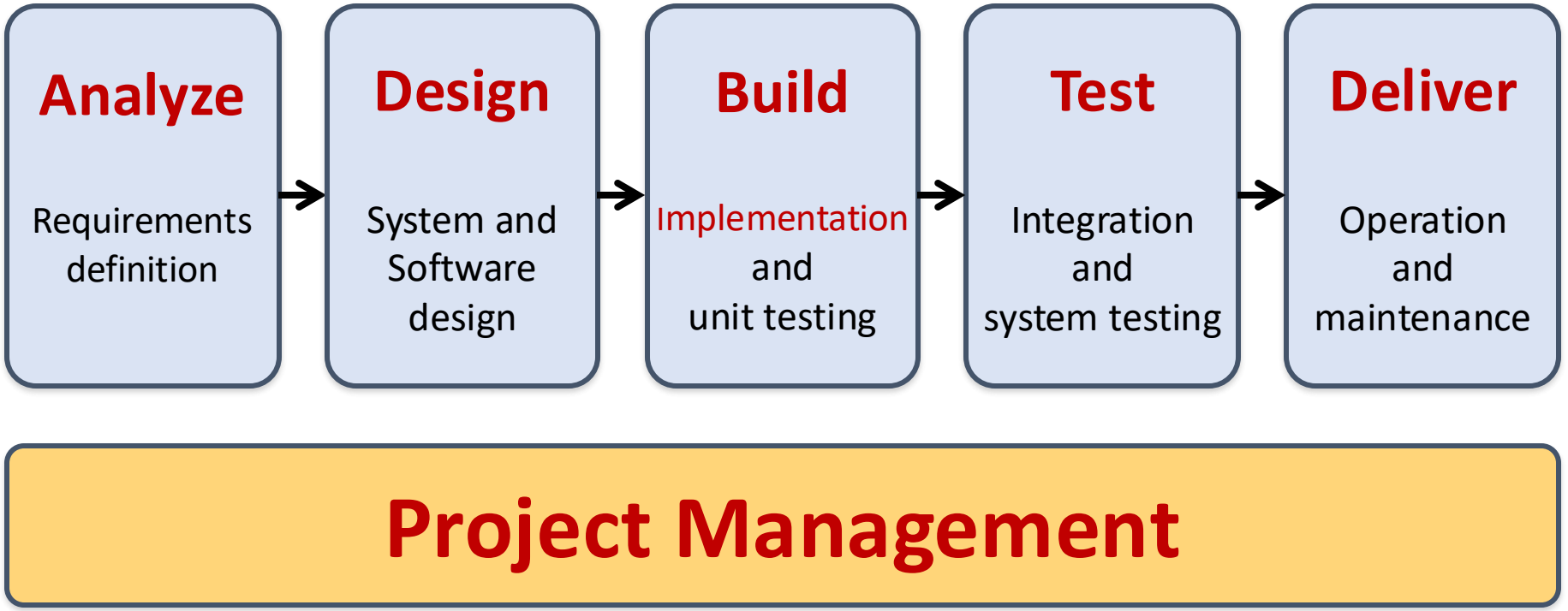
Use Case 3:

LLMs as Natural Language Interfaces to Complex Design Tools

Source: Tucker J. Marion, Mahdi Srour, and Frank Piller (2024), "When Generative AI meets product development." MIT Sloan Management Review 66, no. 1 : 14-15.

Software Engineering

Software Engineering
and
Project Management



Information Management

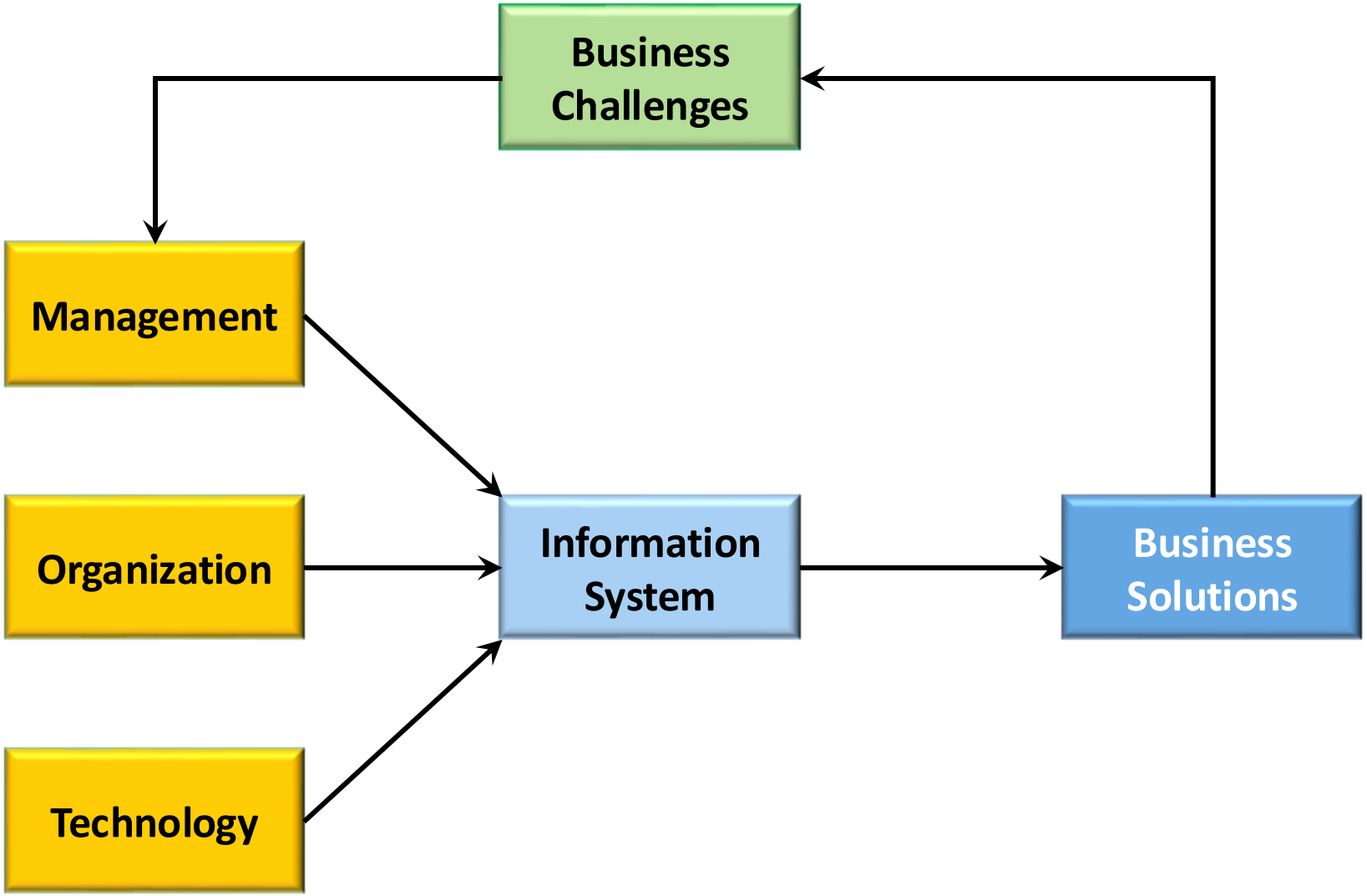
Management Information Systems (MIS)

Information Systems

Information Management (MIS)
Information Systems

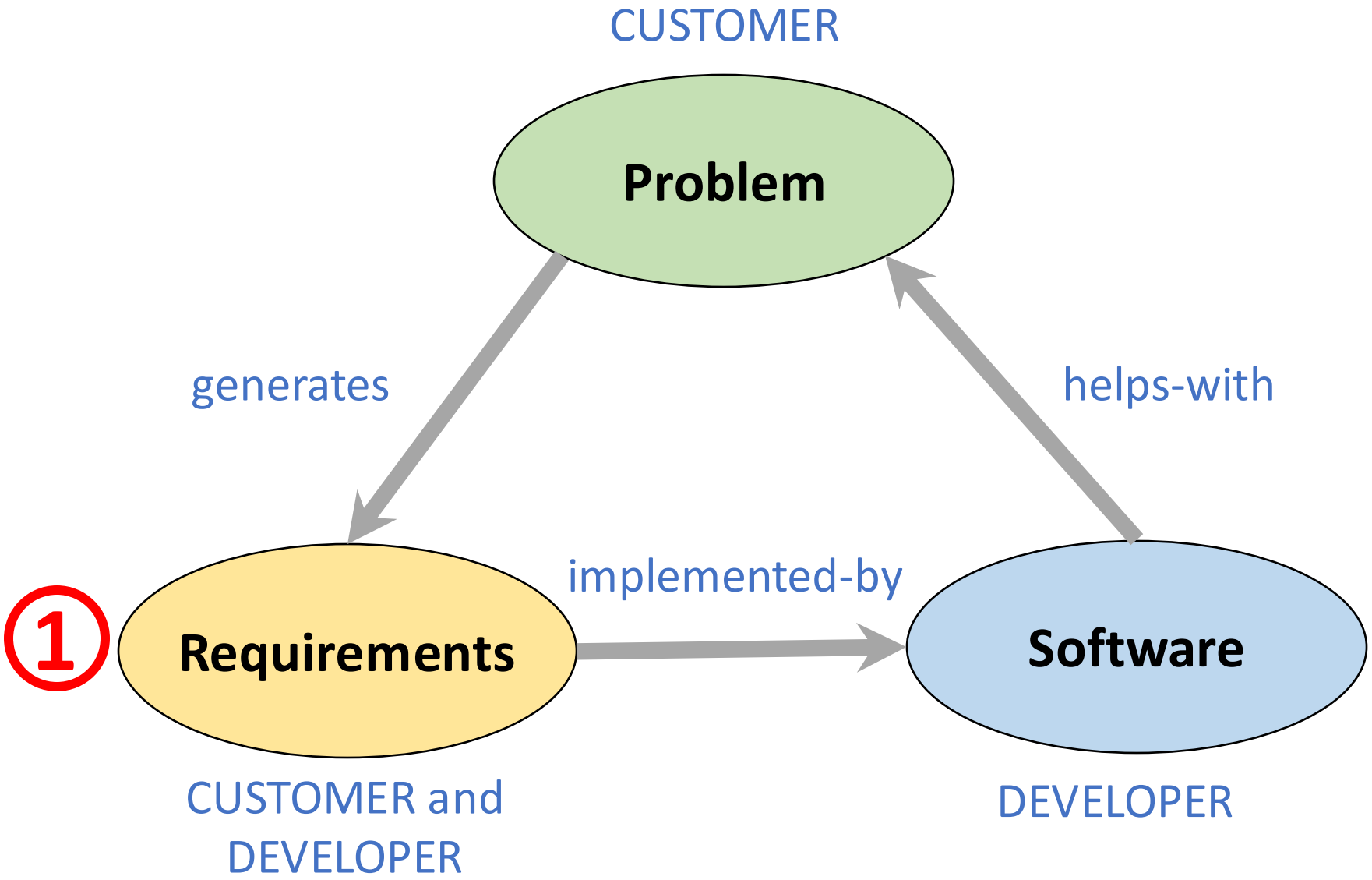
Source: Kenneth C. Laudon & Jane P. Laudon (2014), Management Information Systems: Managing the Digital Firm, Thirteenth Edition, Pearson.

Fundamental MIS Concepts



Source: Kenneth C. Laudon & Jane P. Laudon (2014), Management Information Systems: Managing the Digital Firm, Thirteenth Edition, Pearson.

Project-based software engineering



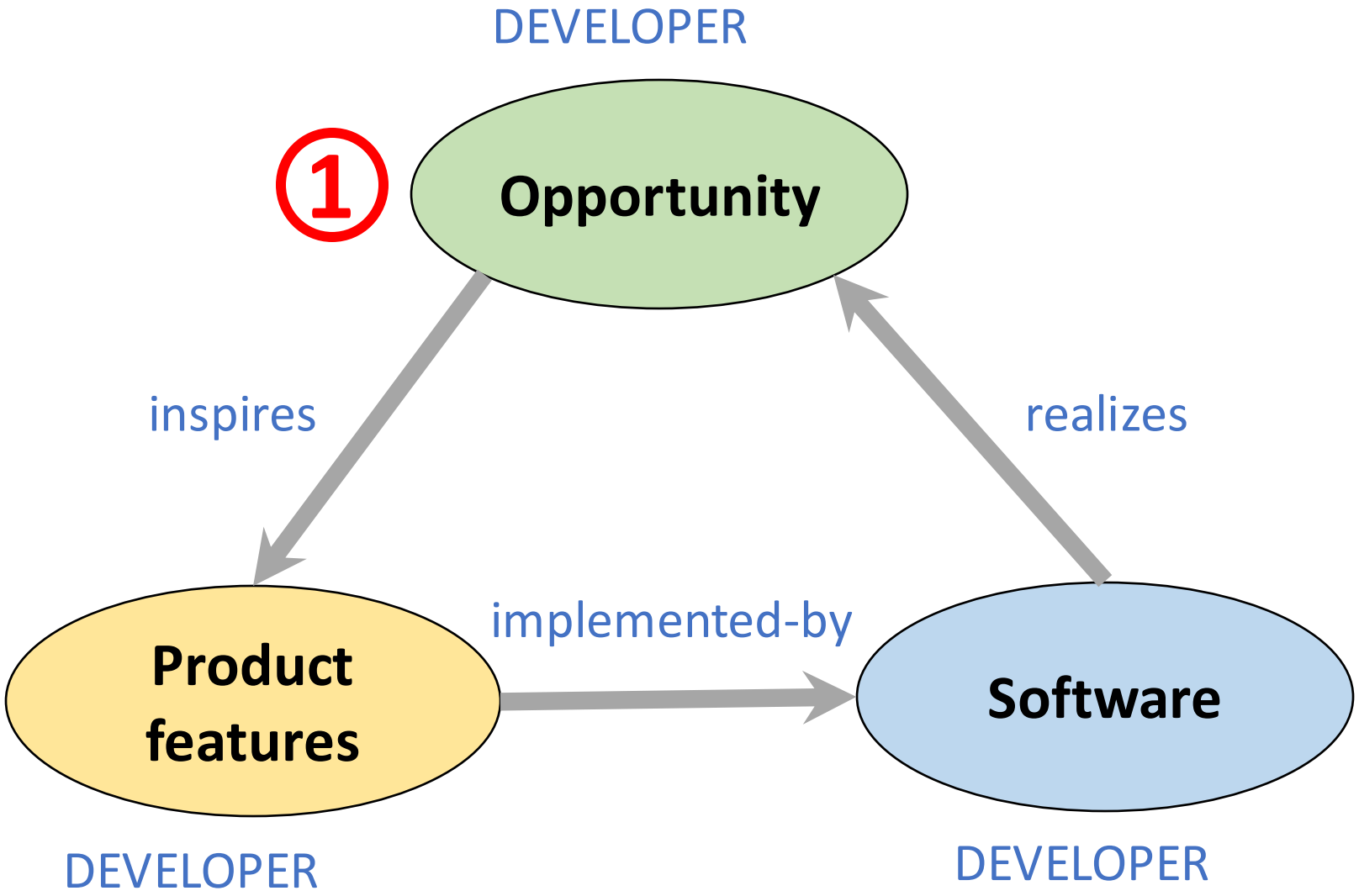
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Project-based software engineering

- The starting point for the software development is a set of 'software requirements' that are owned by an external client and which set out what they want a software system to do to support their business processes.
- The software is developed by a software company (the contractor) who design and implement a system that delivers functionality to meet the requirements.
- The customer may change the requirements at any time in response to business changes (they usually do). The contractor must change the software to reflect these requirements changes.
- Custom software usually has a long-lifetime (10 years or more) and it must be supported over that lifetime.

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

Product software engineering



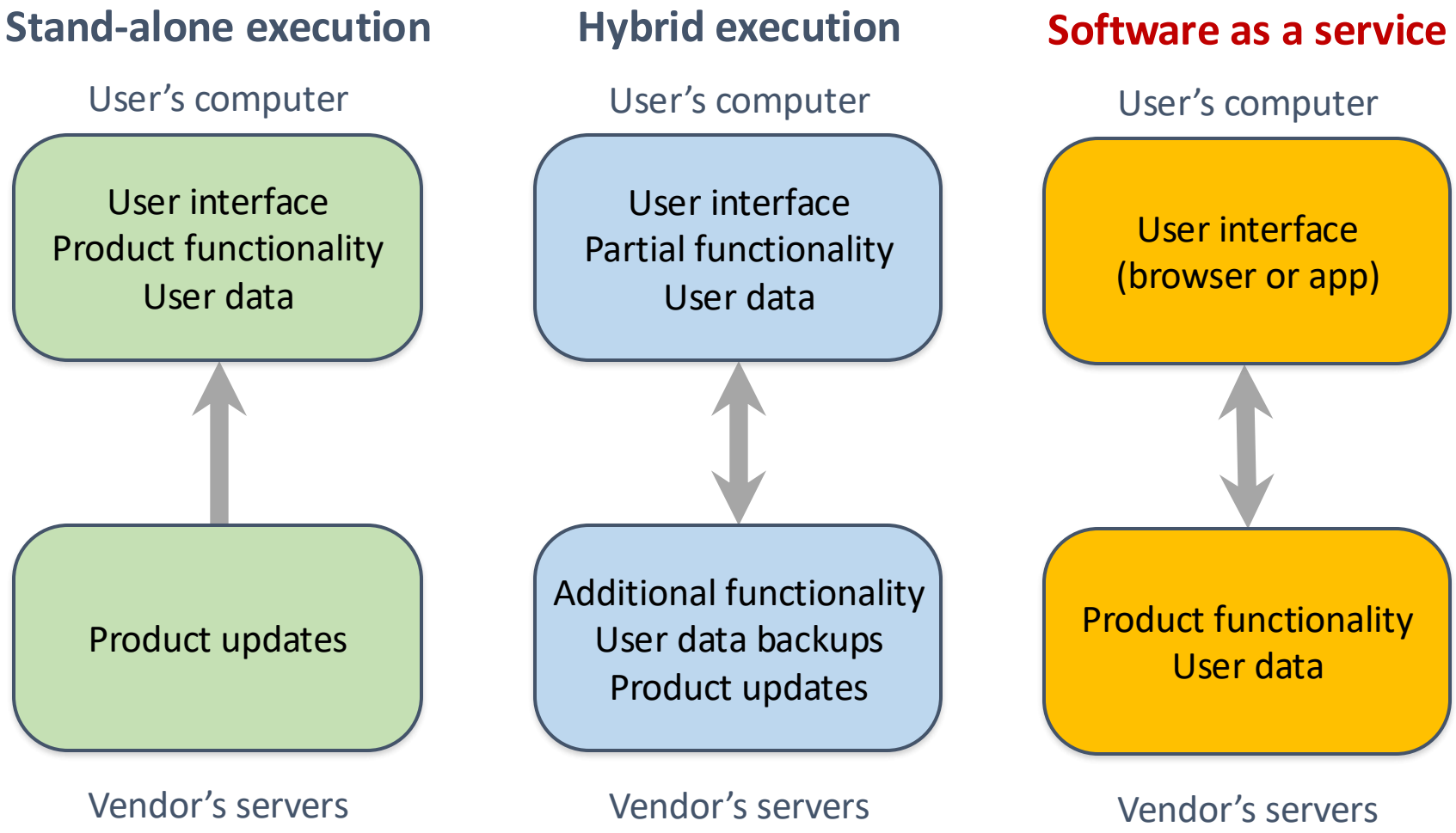
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Product software engineering

- The starting point for product development is a business opportunity that is identified by individuals or a company.
  They develop a software product to take advantage of this opportunity and sell this to customers.
- The company who identified the opportunity design and implement a set of software features that realize the opportunity and that will be useful to customers.
- The software development company are responsible for deciding on the development timescale, what features to include and when the product should change.
- Rapid delivery of software products is essential to capture the market for that type of product.

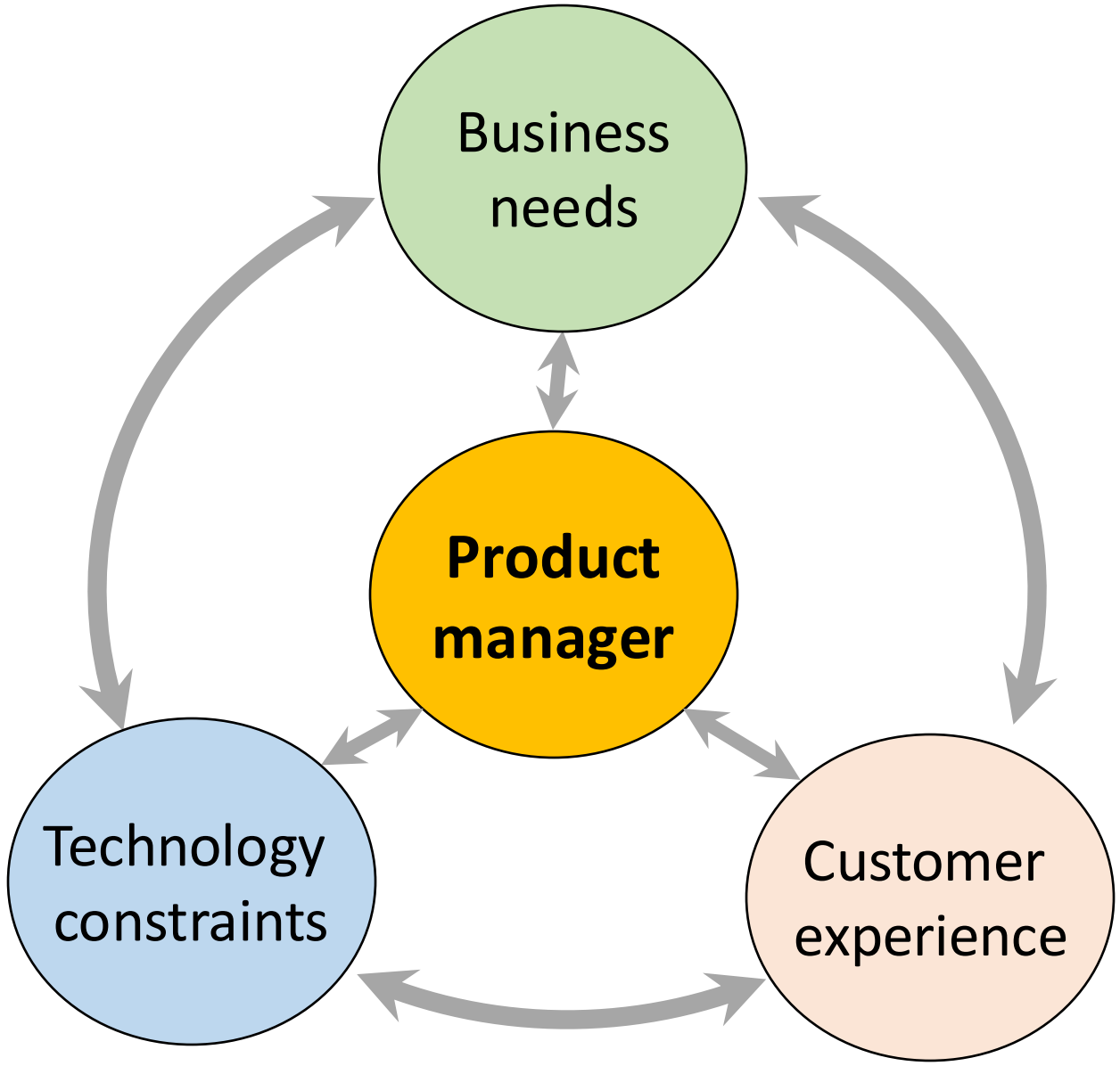
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Software execution models



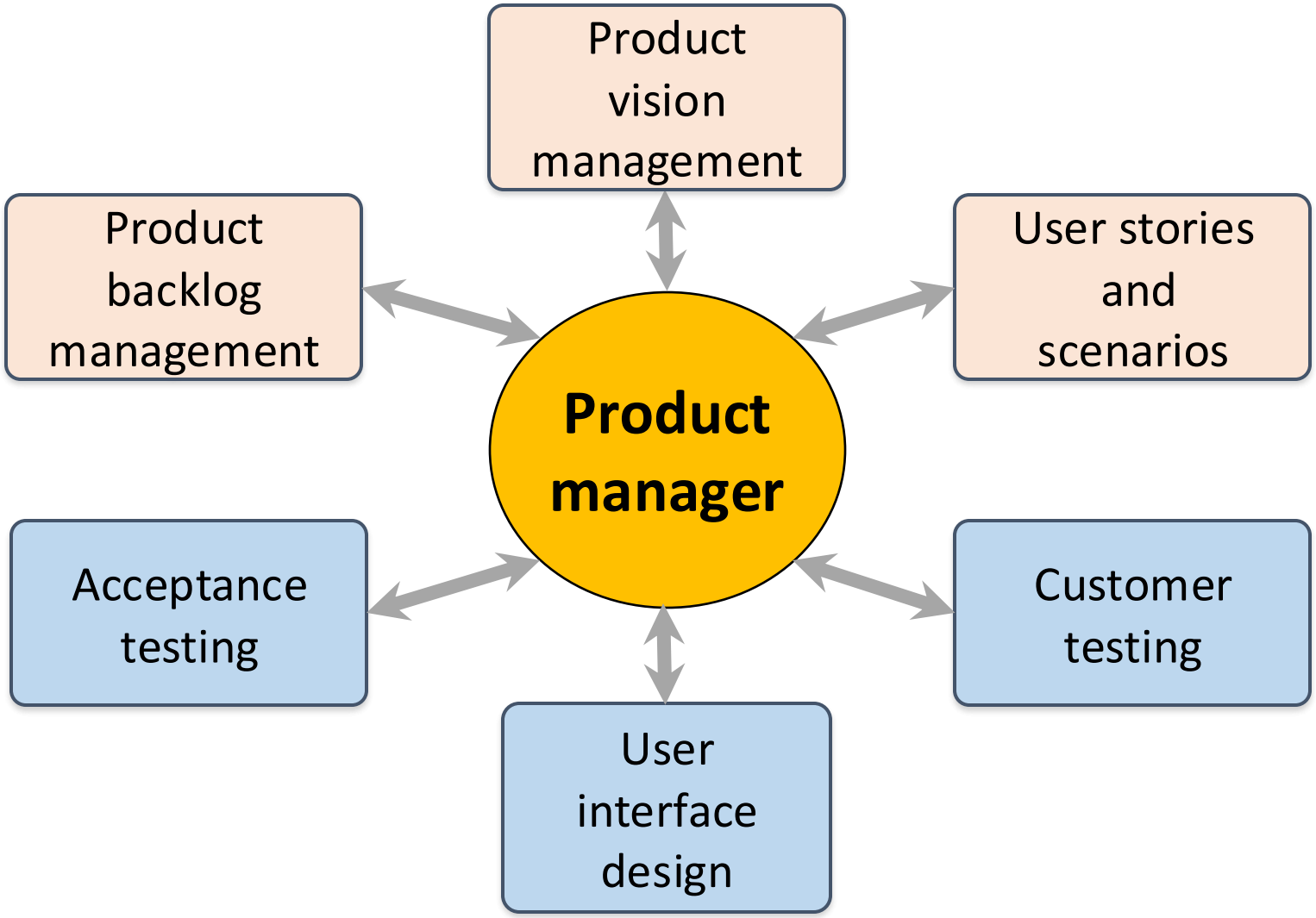
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Product management concerns



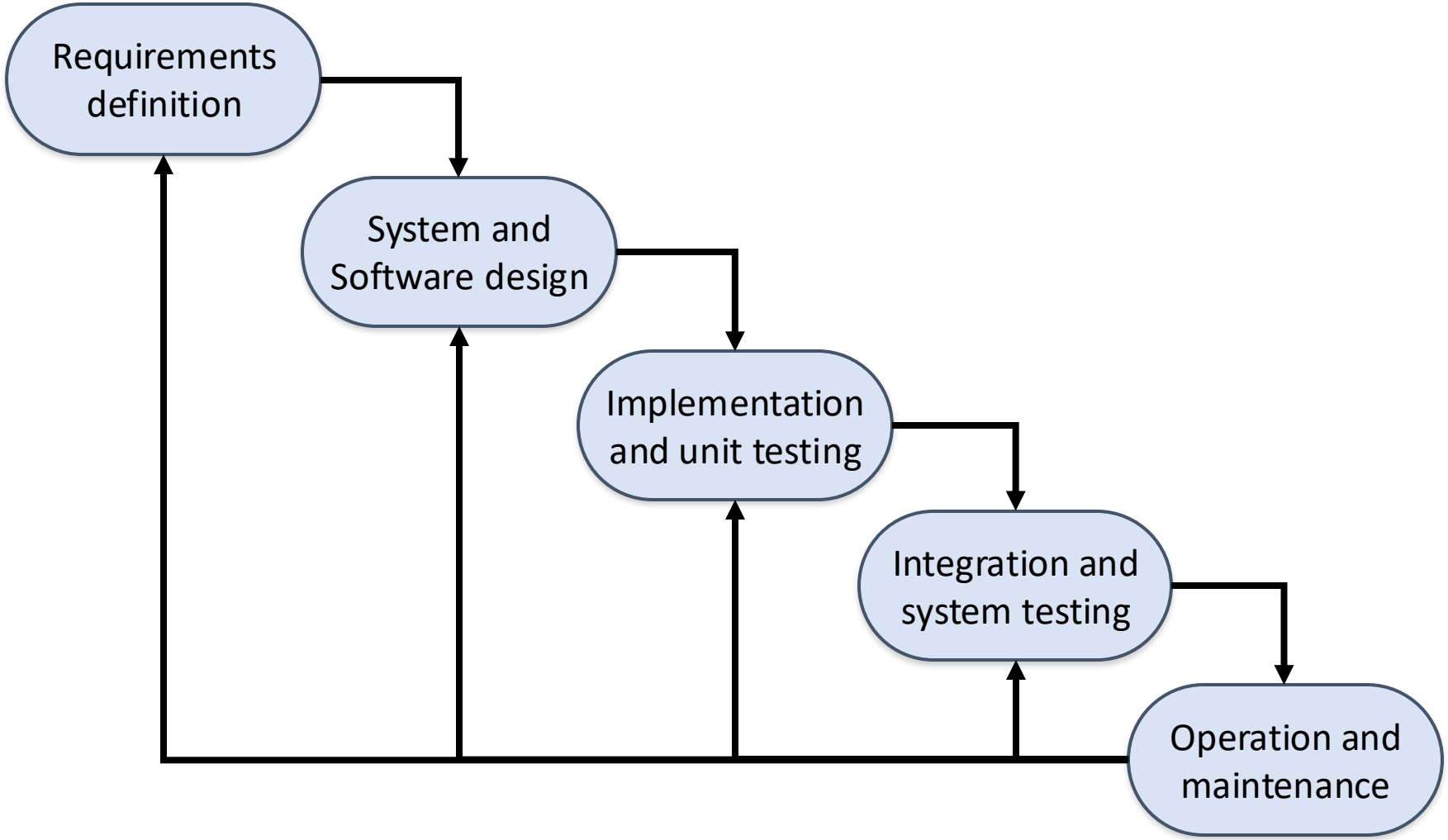
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Technical interactions of product managers



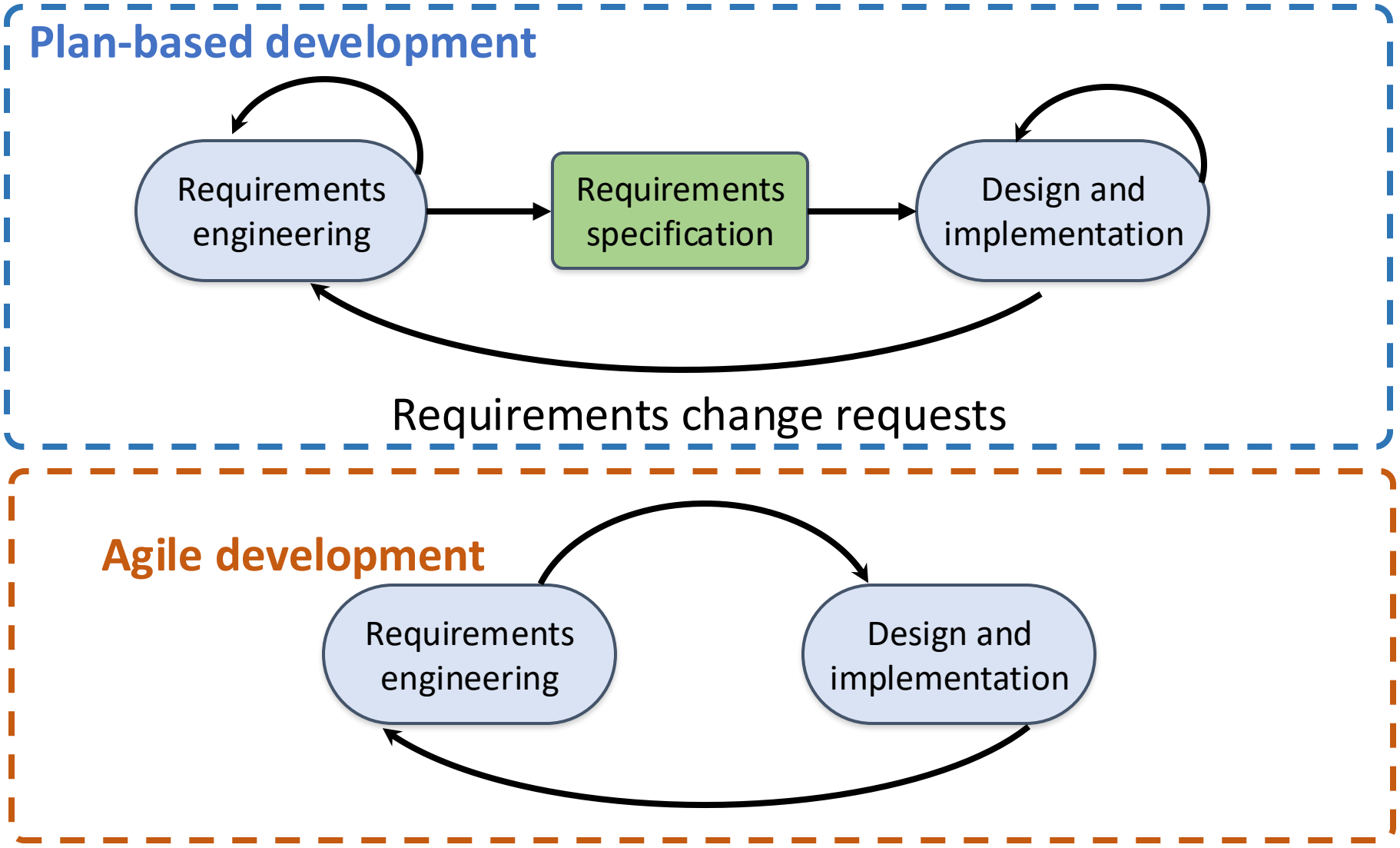
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Software Development Life Cycle (SDLC)
The waterfall model



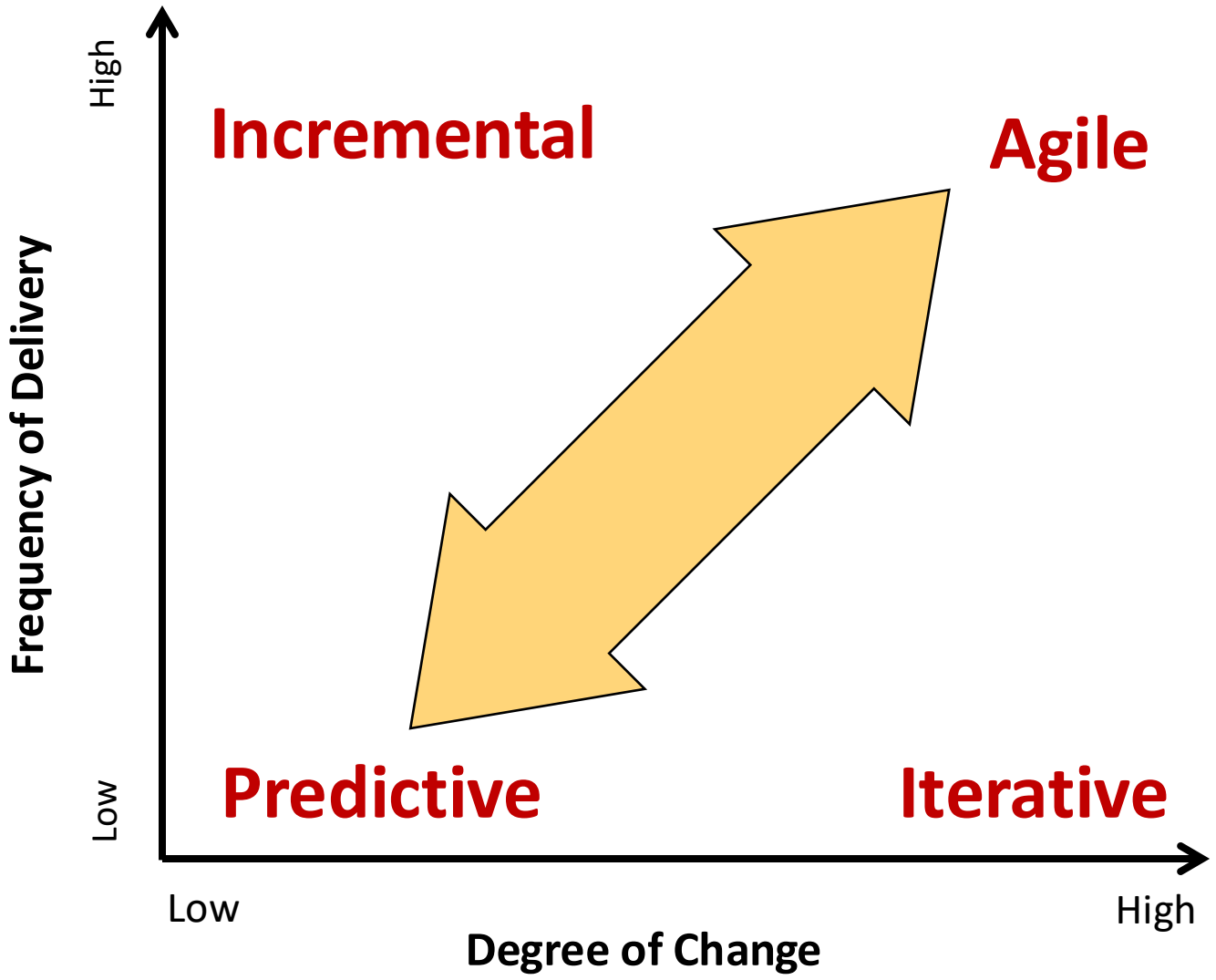
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Plan-based and Agile development



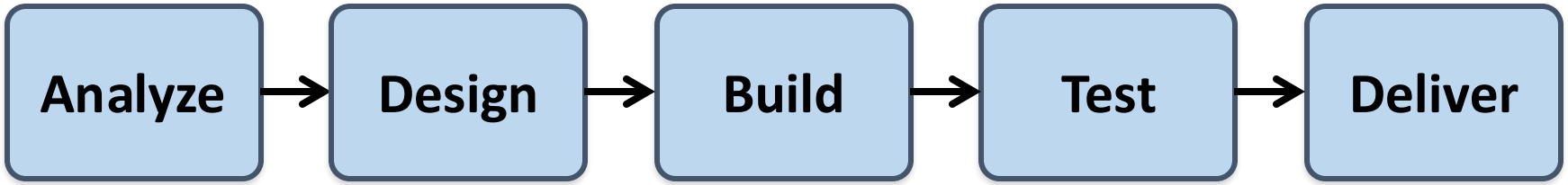
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

The Continuum of Life Cycles



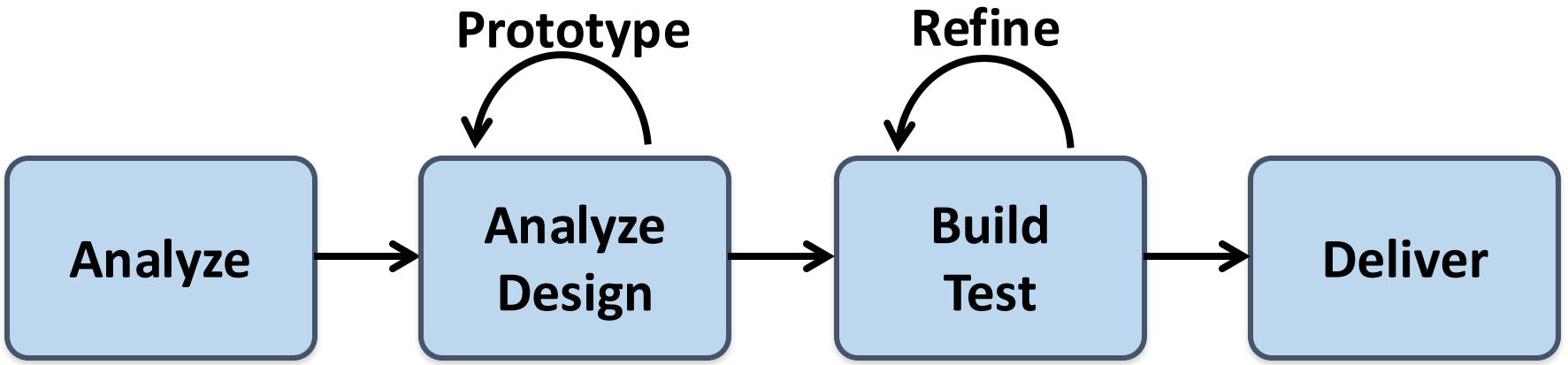
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Predictive Life Cycle



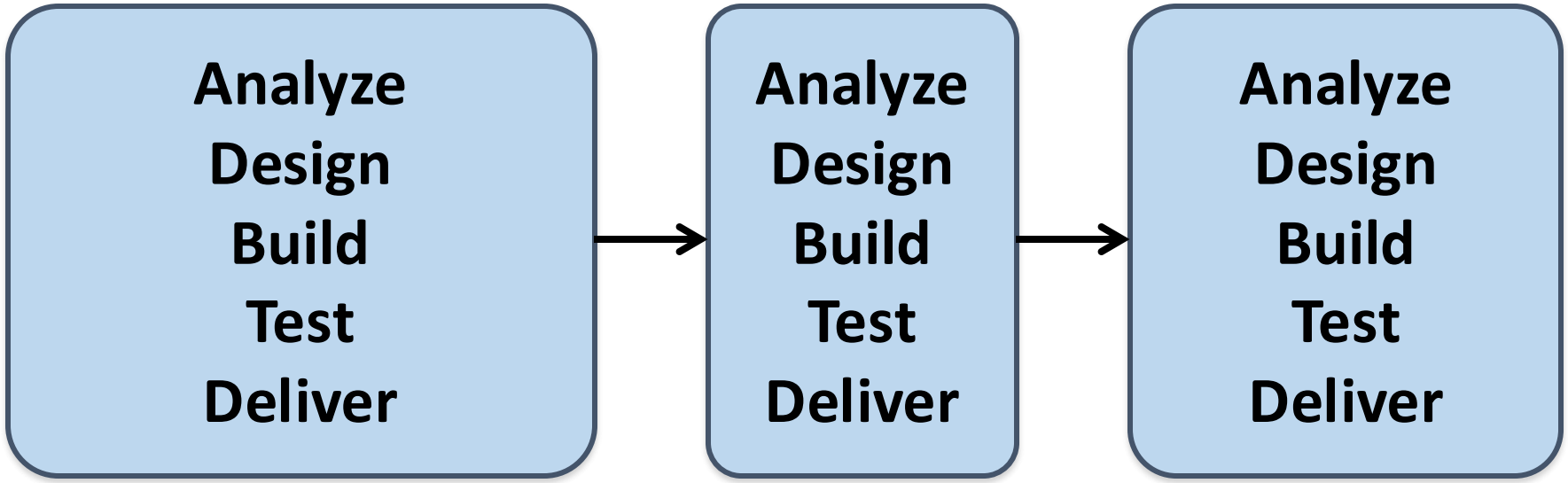
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

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

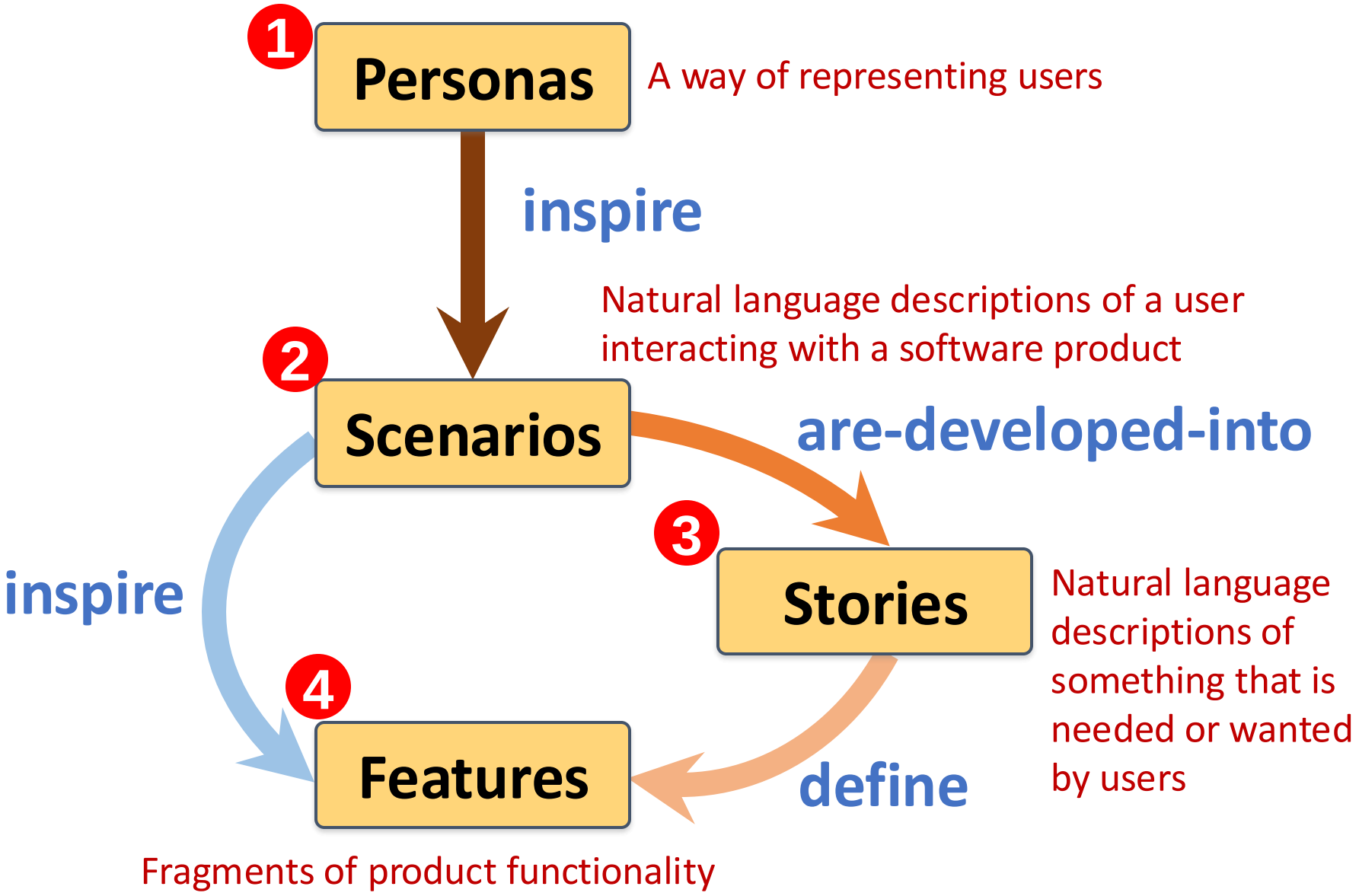
Requirements Analysis Design Build Test	Requirements Analysis Design Build Test	Requirements Analysis Design Build Test	Requirements Analysis Design Build Test	Repeat as needed …	Requirements Analysis Design Build Test	Requirements Analysis Design Build Test

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

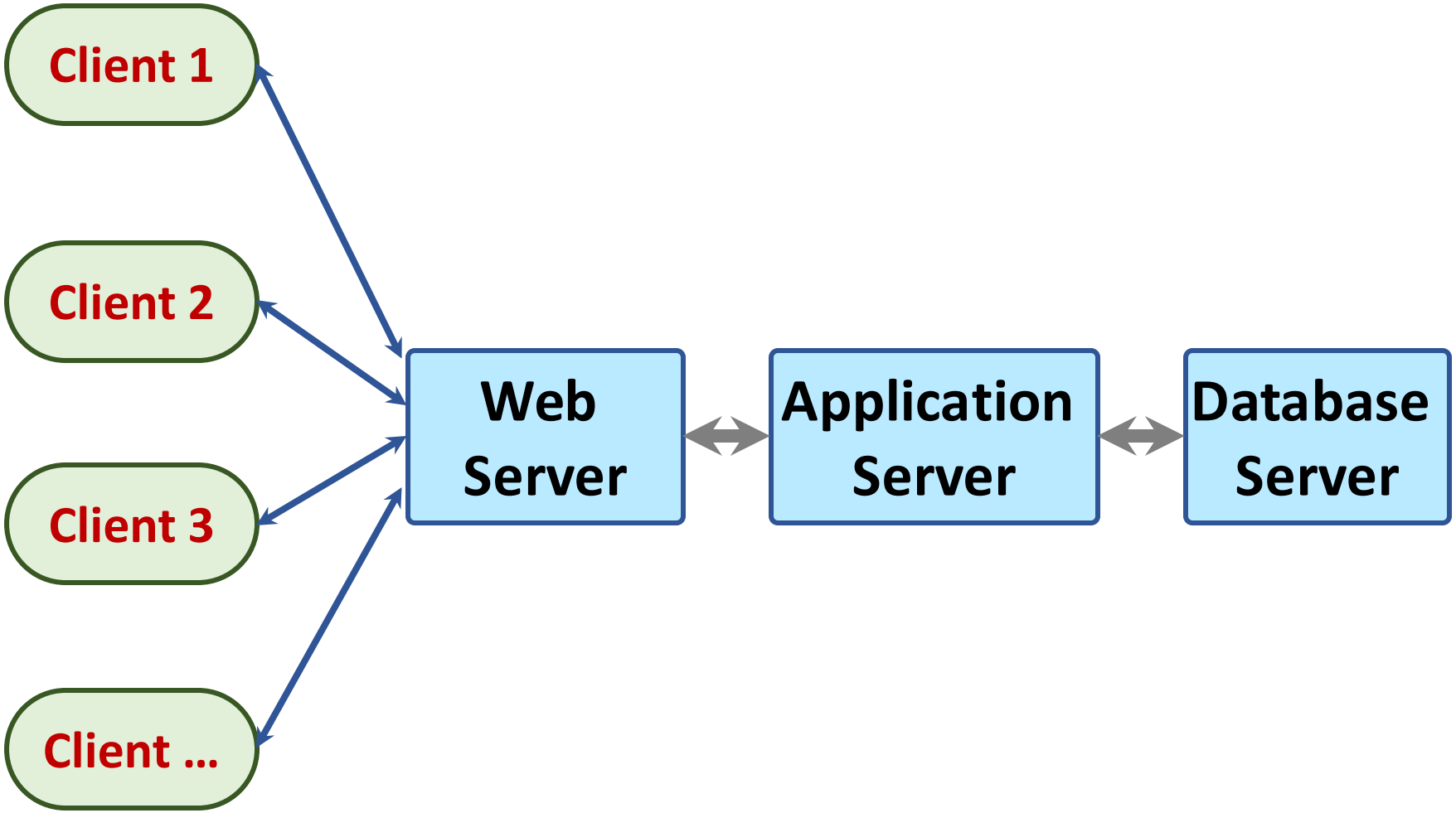
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

From personas to features



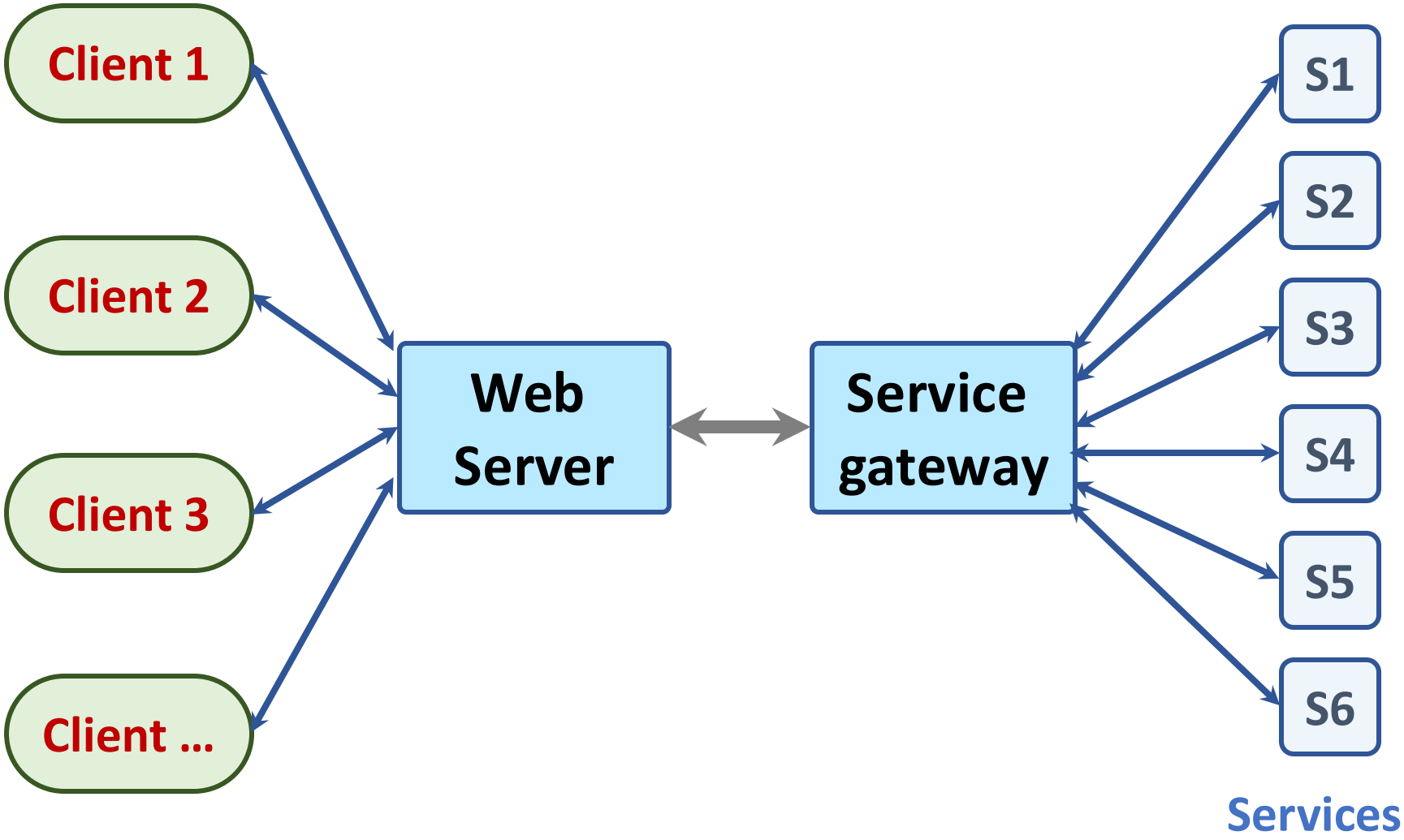
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Multi-tier client-server architecture

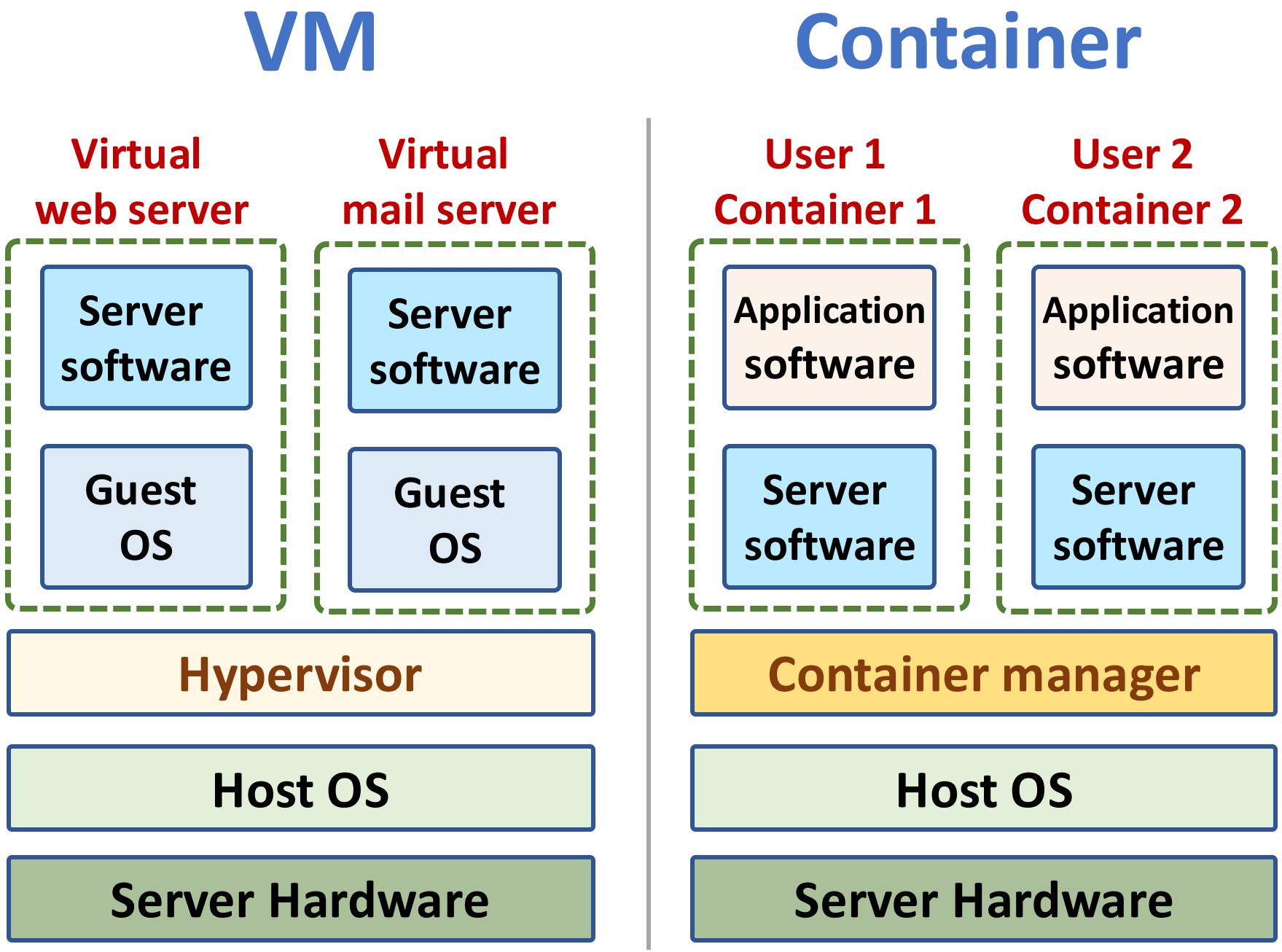


Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Service-oriented Architecture

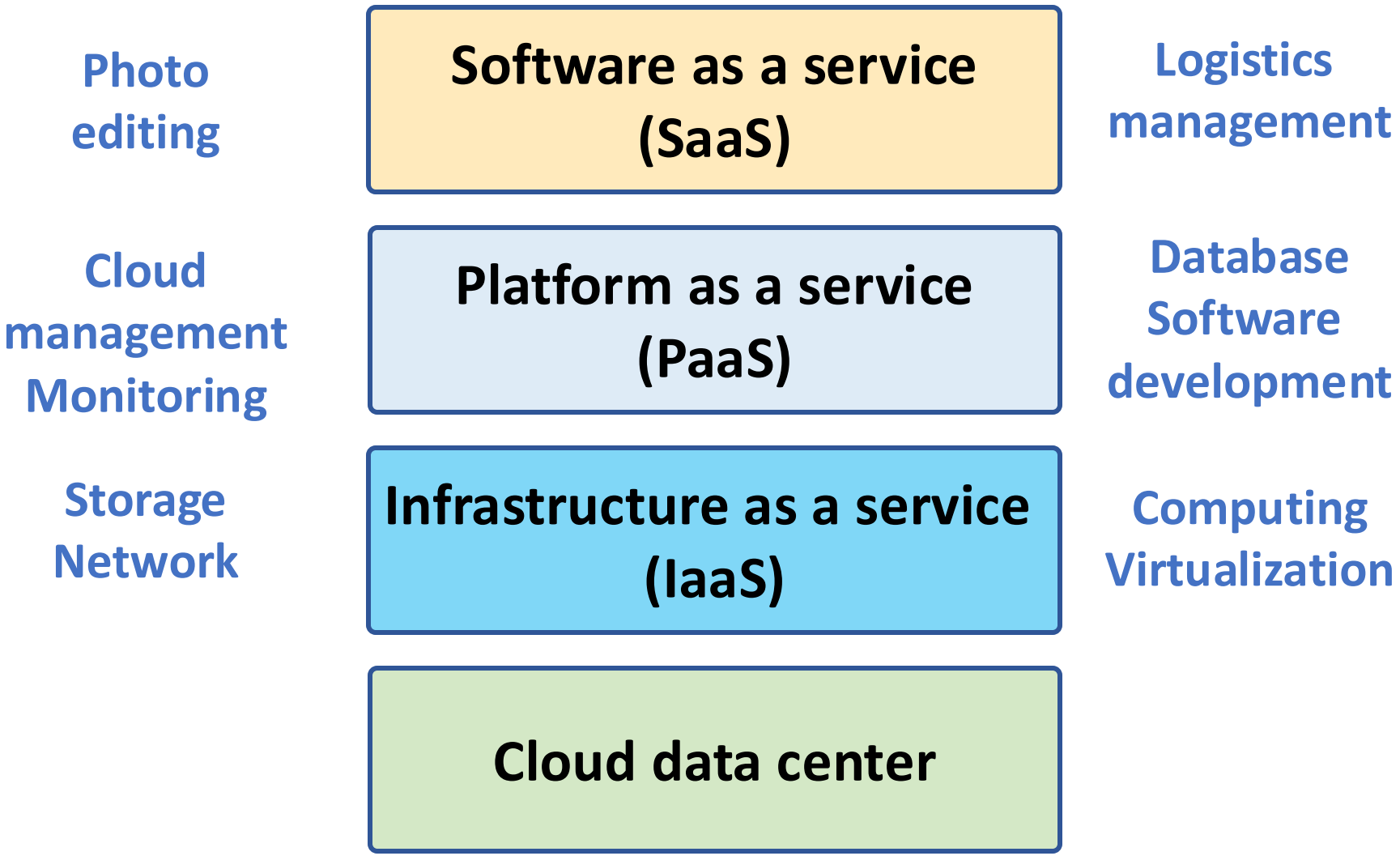


Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.



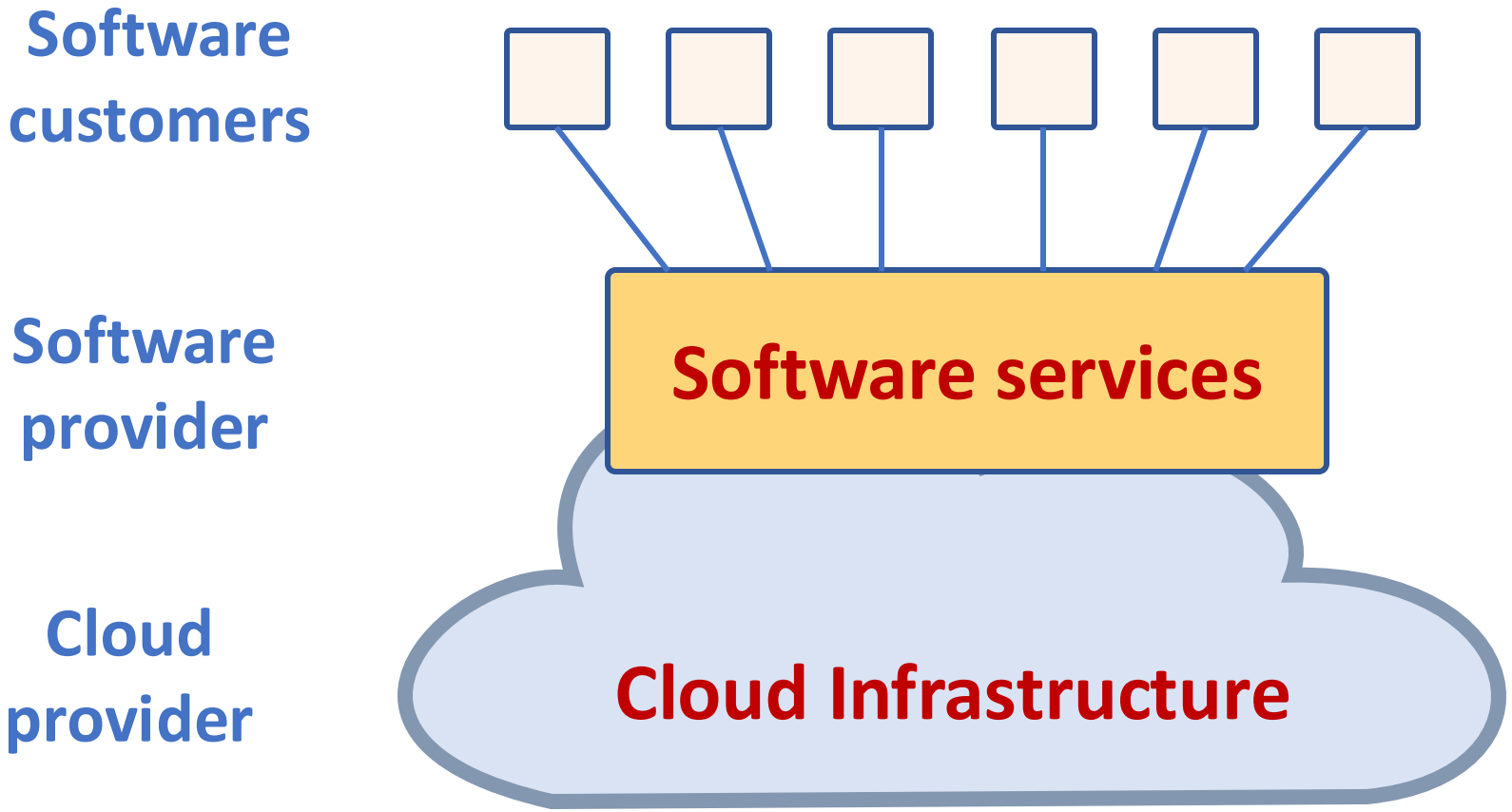
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Everything as a service



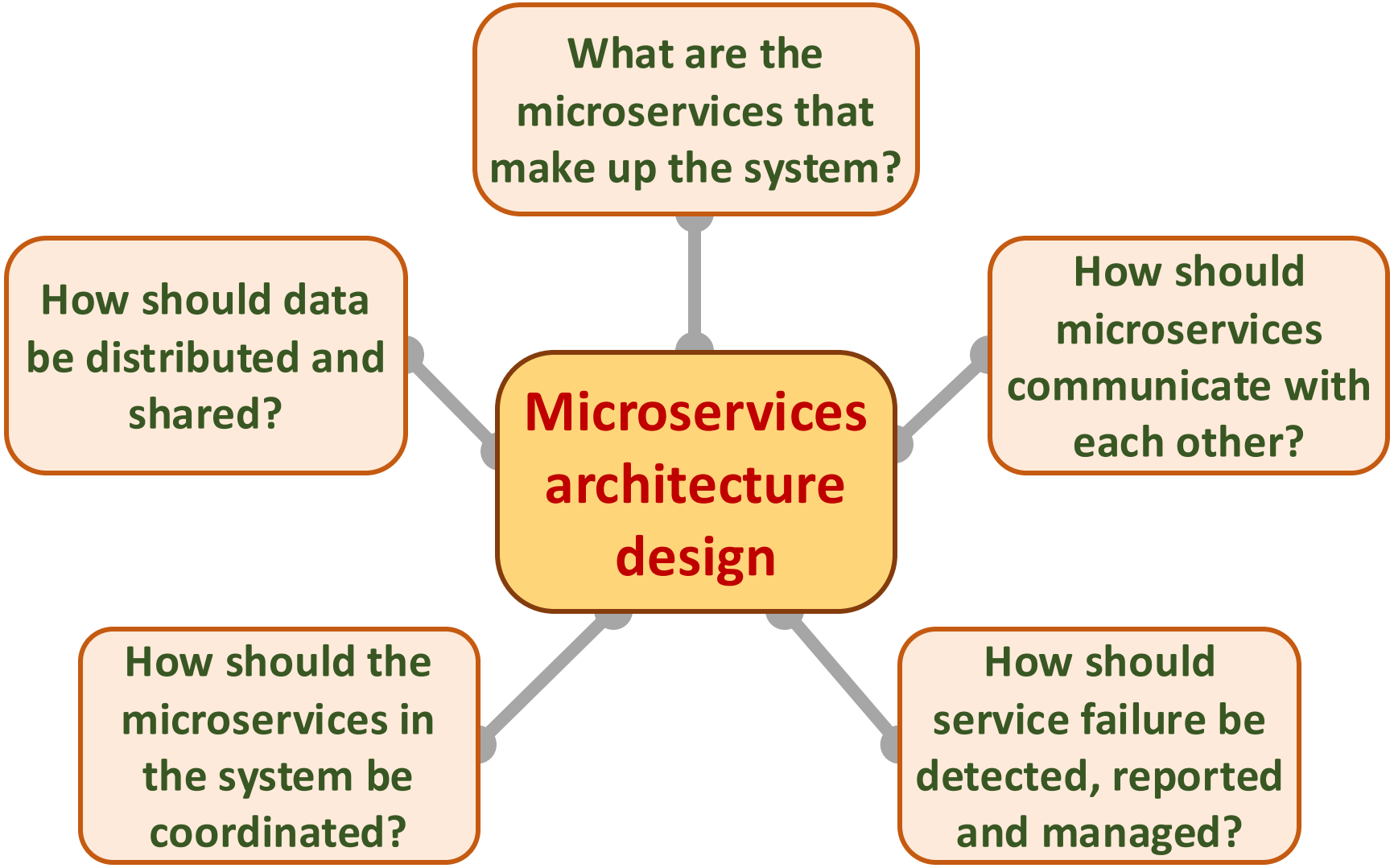
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Software as a service



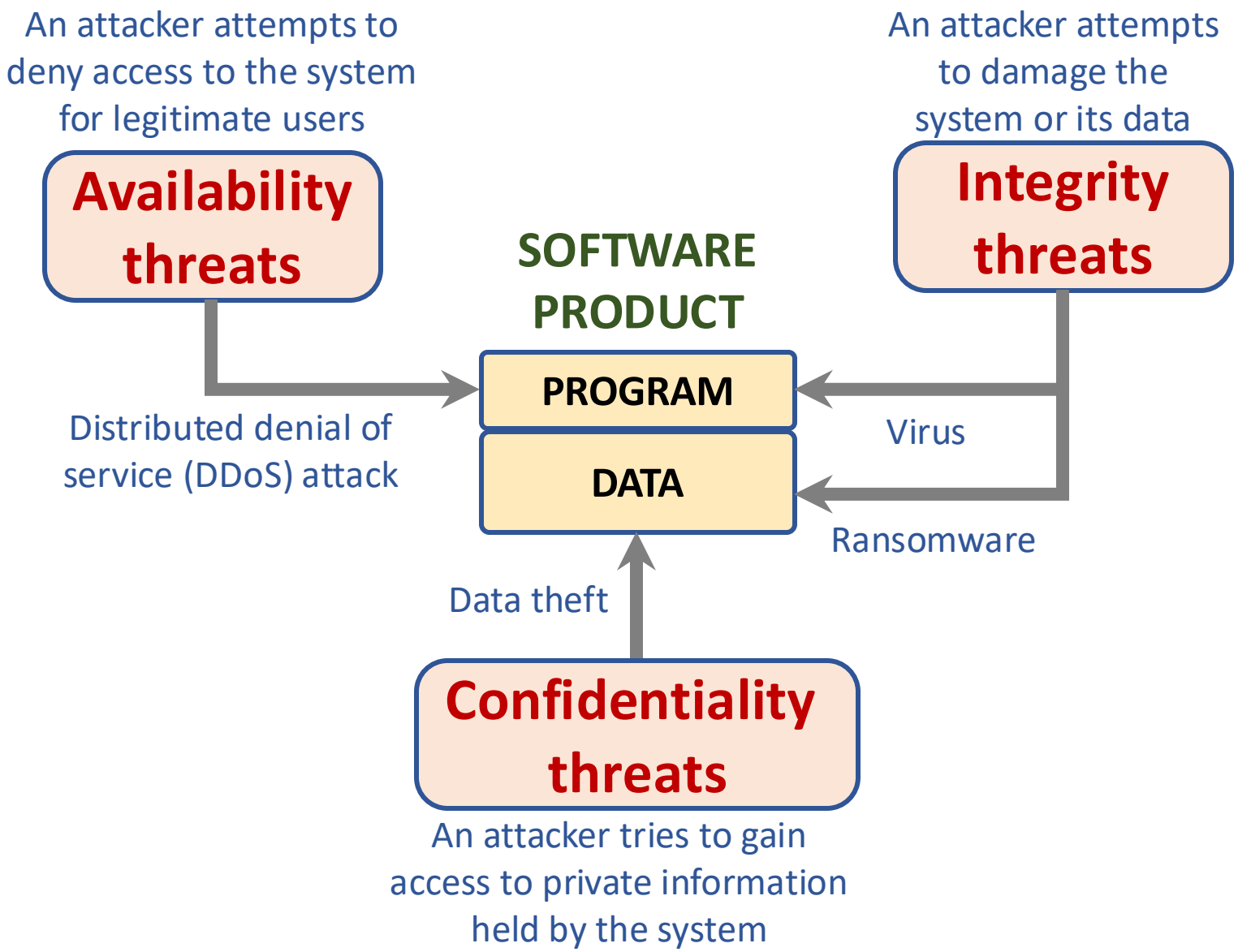
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Microservices architecture – key design questions



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

Types of security threat

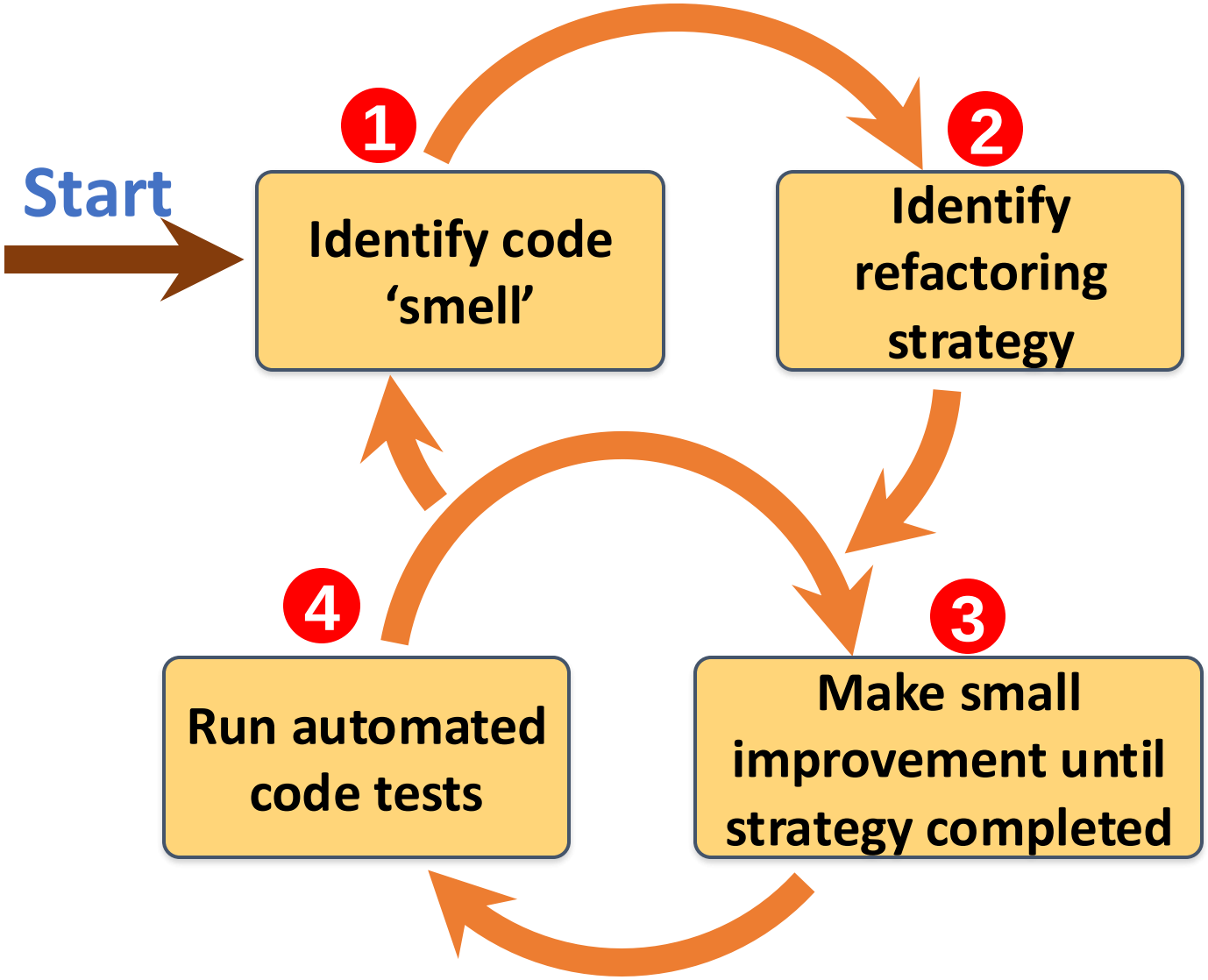


Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Software product quality attributes

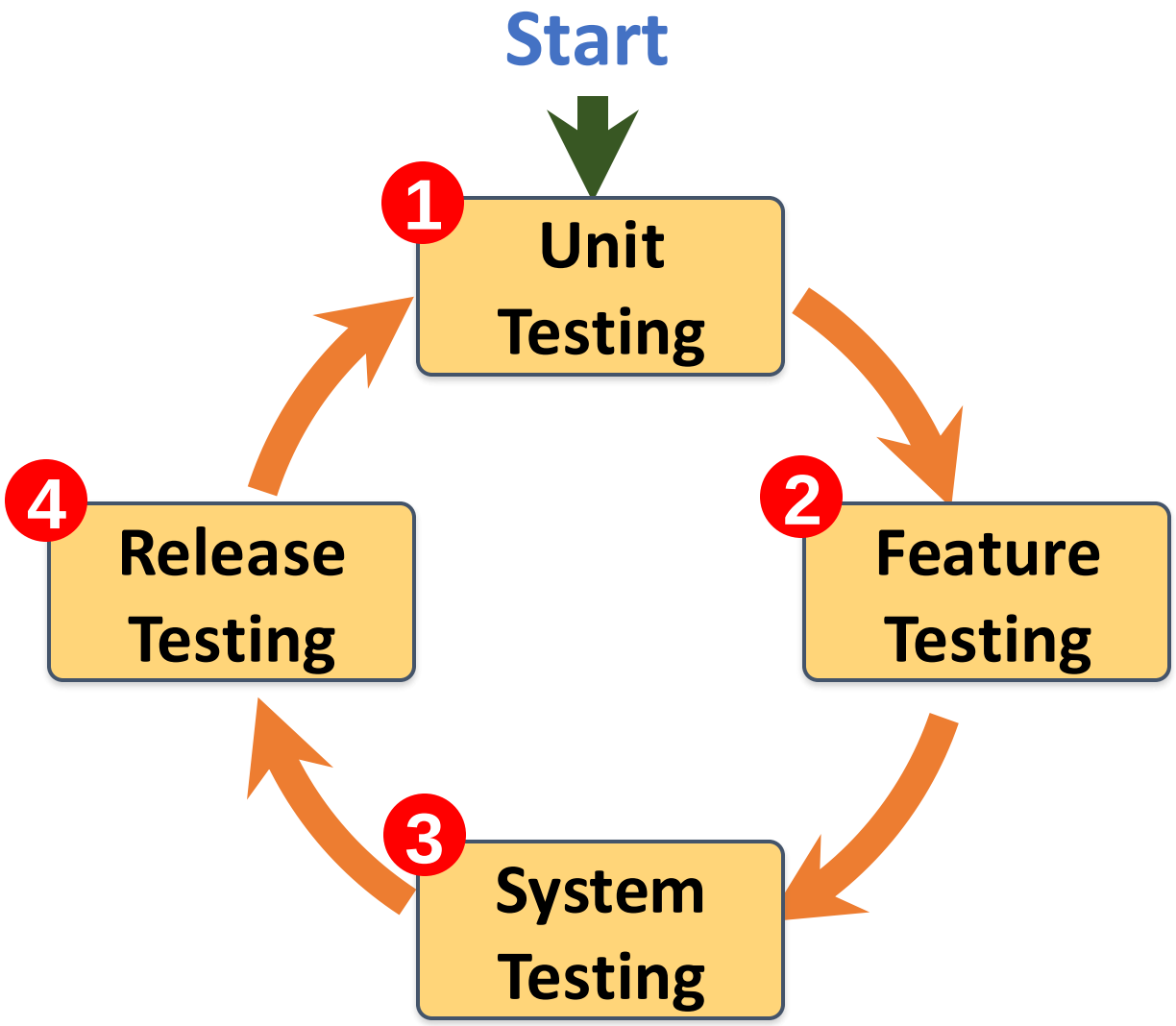
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

A refactoring process



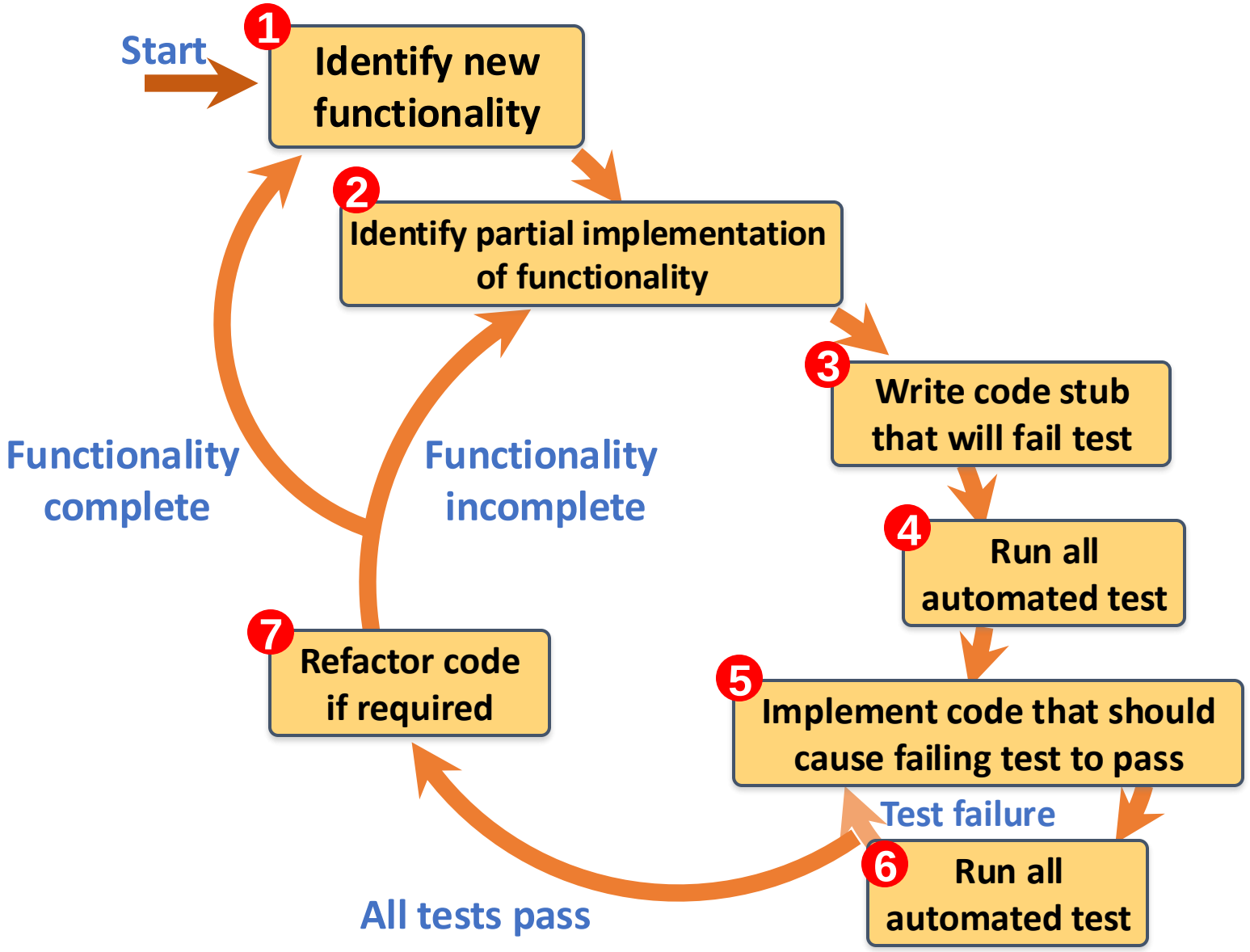
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Functional testing



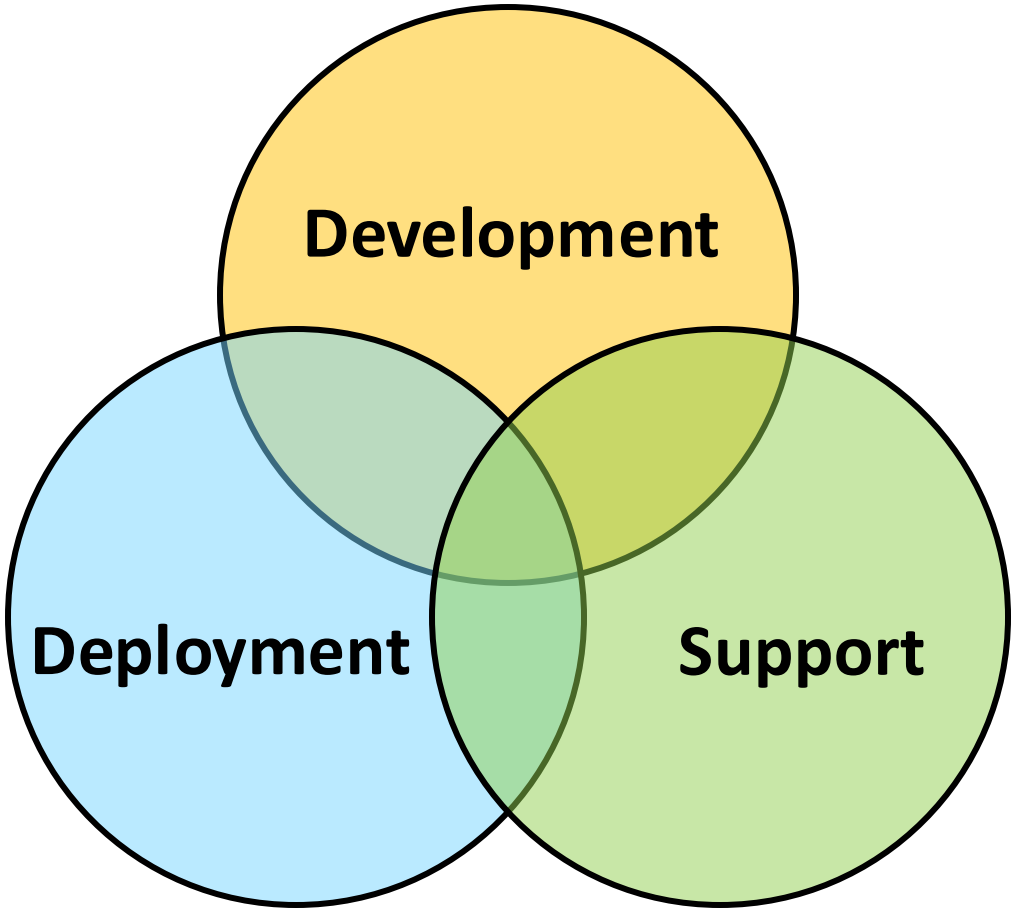
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Test-driven development (TDD)



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

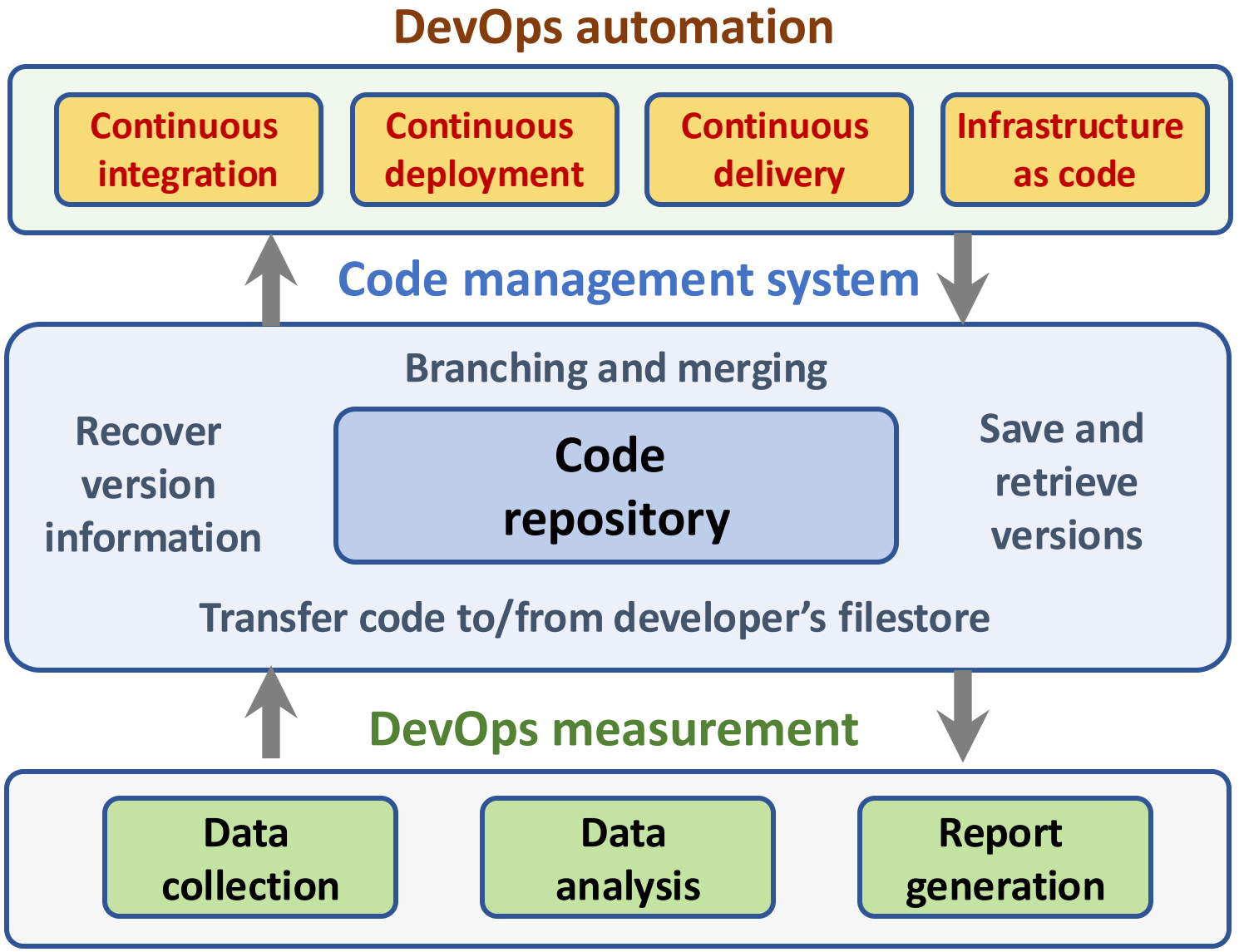
DevOps



Multi-skilled DevOps team

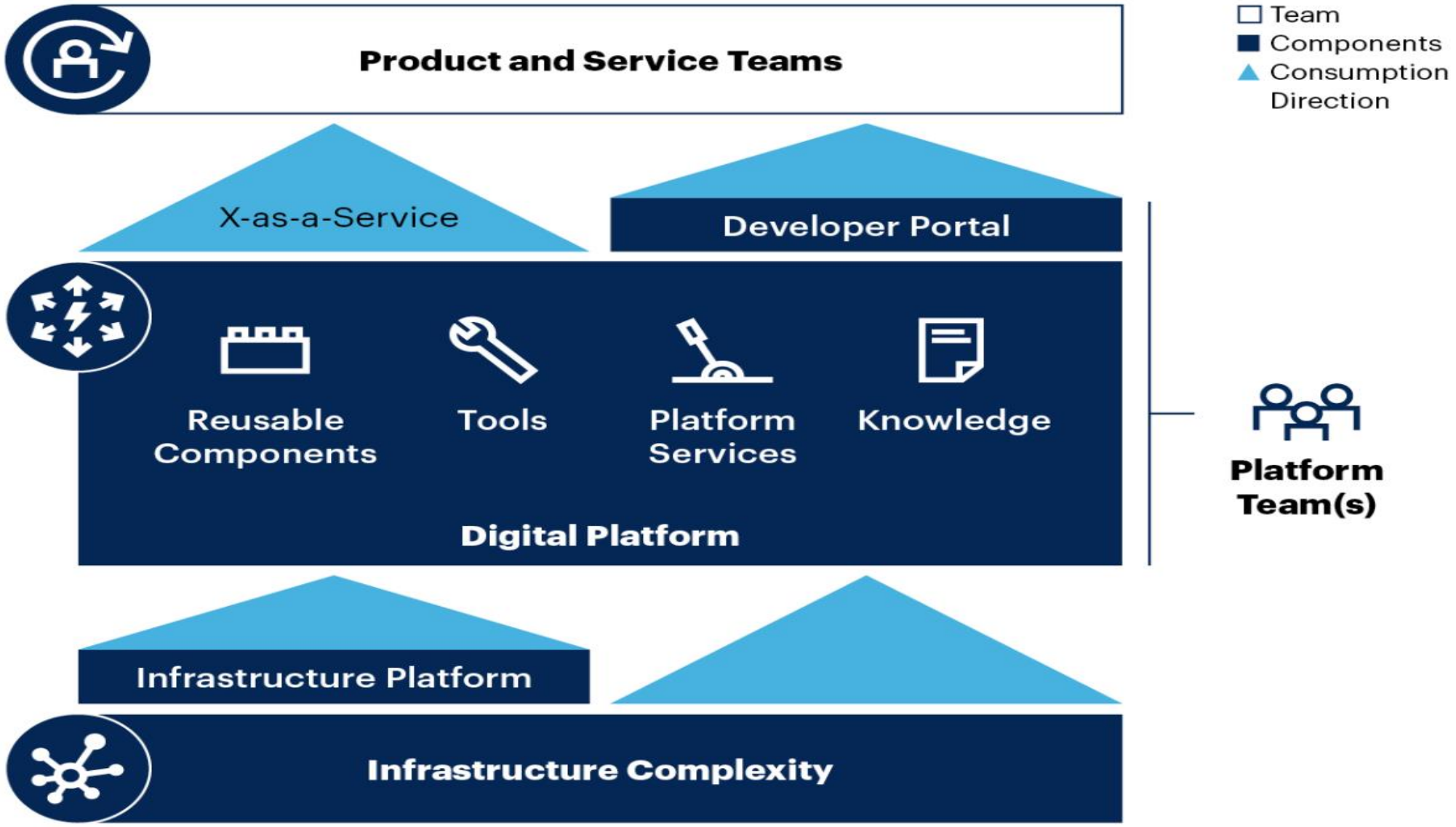
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Code management and DevOps



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

Platform Engineering



Source: https://www.gartner.com/en/articles/what-is-platform-engineering

Marketing

Marketing

“Meeting

needs

profitably”

Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

Marketing

"Marketing is an organizational function
and a set of processes for
creating, communicating, and delivering
value to customers and
for managing customer relationships
in ways that benefit the organization and its stakeholders."

Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

Marketing Management

Marketing Management

"Marketing management is the art and science of choosing target markets and getting, keeping, and growing customers through creating, delivering, and communicating superior customer value."

Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

Marketing Management

1. Understanding Marketing Management
2. Capturing Marketing Insights
3. Connecting with Customers
4. Building Strong Brands
5. Creating Value
6. Delivering Value
7. Communicating Value
8. Conducting Marketing Responsibly for Long-term Success

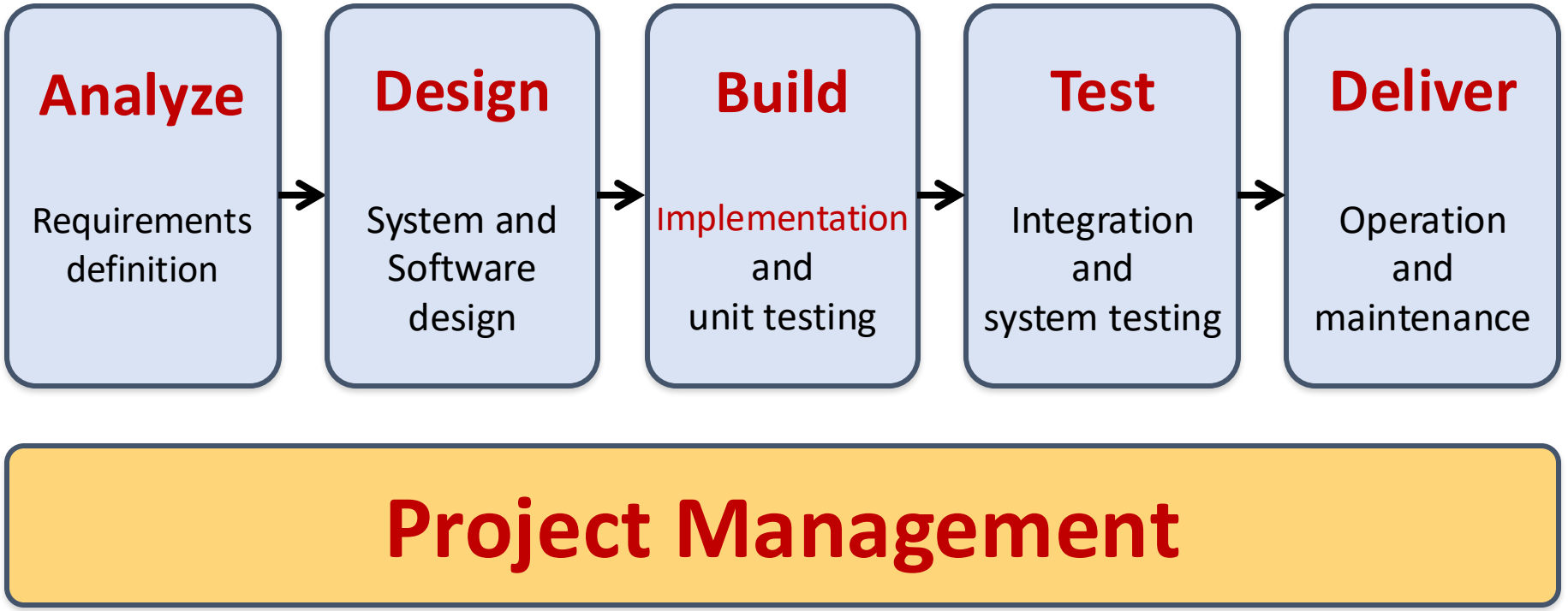
Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

OKRs, CFRs, KPIs
Agile Performance Management

- **OKRs**
  - **Objectives and Key Results**
- **CFRs**
  - **Conversations, Feedback, Recognition**
- **KPIs**
  - **Key Performance Indicators**

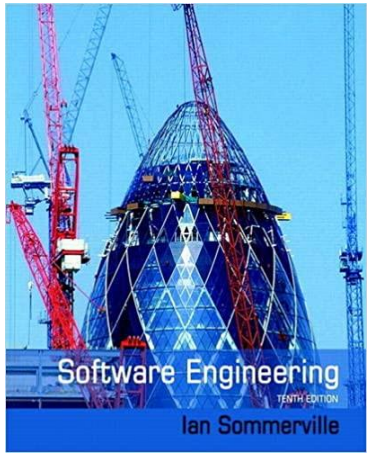
Software Engineering

Software Engineering
and
Project Management



Software Engineering

- **Software engineering** is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use.



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

What is software?

- **Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.**

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

What are the attributes of good software?

- **Good software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.**

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

What is software engineering?

- **Software engineering is an engineering discipline that is concerned with all aspects of software production from initial conception to operation and maintenance.**

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

What are the fundamental software engineering activities?

- **Software specification, software development, software validation and software evolution.**

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

What is the difference between software engineering and computer science?

- **Computer science focuses on theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.**

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

What are the best software engineering techniques and methods?

- **While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system.**
- **For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed.**
- **There are no methods and techniques that are good for everything.**

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

What are the costs of software engineering?

- **Roughly 60% of software costs are development costs, 40% are testing costs.**
- **For custom software, evolution costs often exceed development costs.**

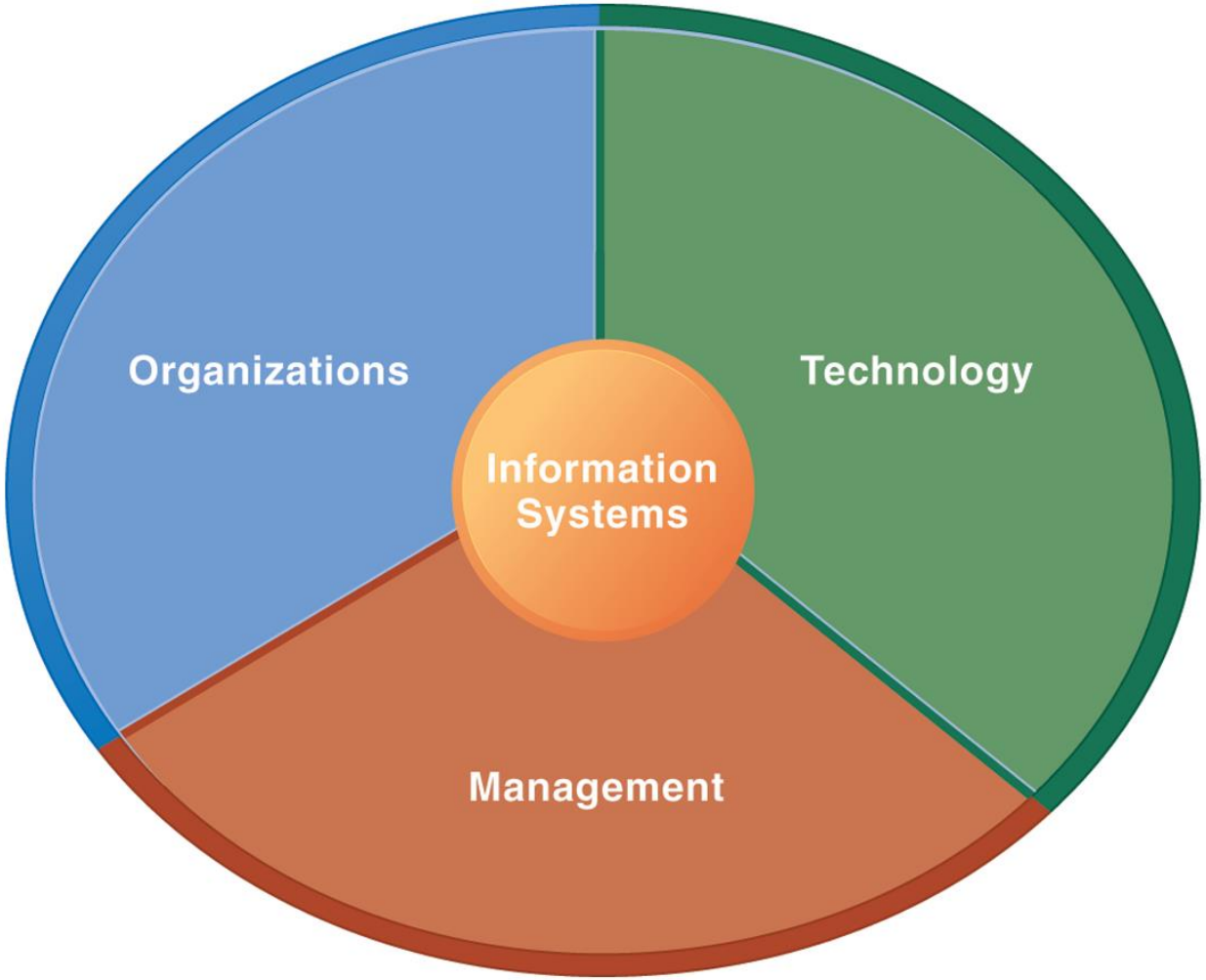
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Information Management

Management Information Systems (MIS)

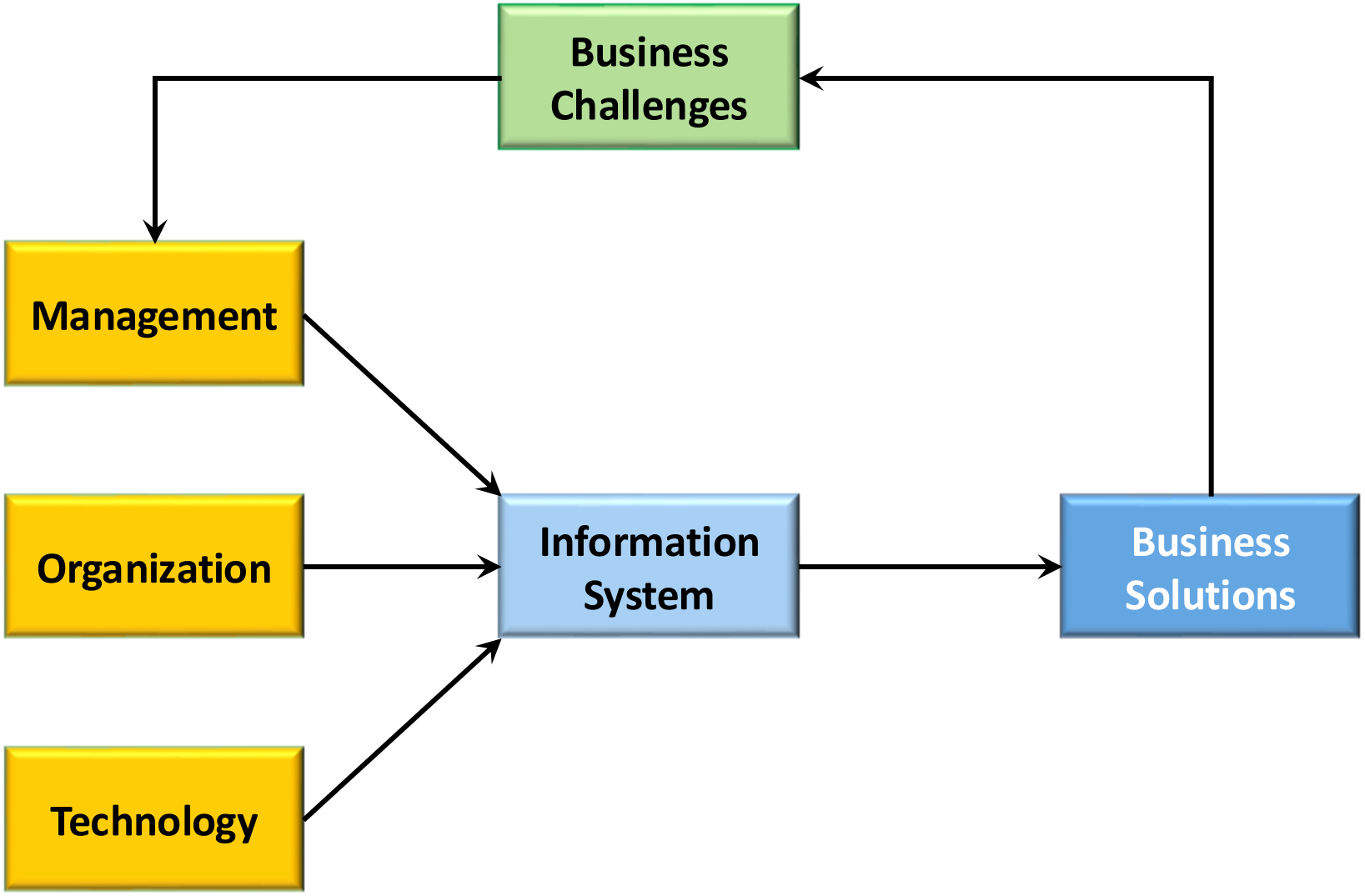
Information Systems

Information Management (MIS)
Information Systems



Source: Kenneth C. Laudon & Jane P. Laudon (2014), Management Information Systems: Managing the Digital Firm, Thirteenth Edition, Pearson.

Fundamental MIS Concepts



Source: Kenneth C. Laudon & Jane P. Laudon (2014), Management Information Systems: Managing the Digital Firm, Thirteenth Edition, Pearson.

Software products

- **Software products are generic software systems that provide functionality that is useful to a range of customers.**
- **Software products:**
  - **Large-scale business systems (e.g. MS Excel)**
  - **Personal products (e.g. Evernote)**
  - **Simple mobile phone apps and games (e.g. Suduko).**

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

Software product engineering

- **Software product engineering methods and techniques have evolved from software engineering techniques that support the development of one-off, custom software systems.**

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

Software projects

- **Custom software systems are still important for large businesses, government and public bodies.**
- **They are developed in dedicated software projects.**

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

Project

- **A project is a temporary endeavor undertaken to create a unique product, service, or result.**

Source: Project Management Institute (2017), A Guide to the Project Management Body of Knowledge (PMBOK Guide), Sixth Edition, PMI

Project Management Body of Knowledge (PMBOK Guide) PMBOK v6 vs. PMBOK v7

PMBOK Guide v6

Project Management Body of Knowledge:

- Introduction
- Project Environment
- Role of the Project Manager
- 10 Knowledge Areas
  - Integration
  - Scope
  - Schedule
  - Cost
  - Quality
  - Resources
  - Communications
  - Risk
  - Procurement
  - Stakeholders

The Standard for Project Management (5 Process Groups):

- Initiating
- Planning
- Executing
- Monitoring and Controlling
- Closing

PMBOK Guide v7

The Standard for Project Management :

- Introduction
- System for Value Delivery
- 12 Project Management Principles:
  - 1. Stewardship, 2. Team
  - 3. Stakeholders, 4. Value
  - 5. Systems Thinking, 5. Leadership
  - 7. Tailoring, 8. Quality
  - 9. Complexity, 10, Risk
  - 11. Adaptability and Resiliency
  - 12. Change

Project Management Body of Knowledge:

- 8 Project Performance Domains:
  - 1. Stakeholders, 2. Team,
  - 3. Development approach and Life Cycle
  - 4. Planning, 5. Project Work, 6. Delivery,
  - 7. Measurement, 8. Uncertainty
- Tailoring
- Models, Methods, and Artifacts

Source: Project Management Institute (2021), A Guide to the Project Management Body of Knowledge (PMBOK Guide) – Seventh Edition and The Standard for Project Management, PMI

Project Management Knowledge Areas

(PMBOK v6)

1. Project Integration Management
2. Project Scope Management
3. Project Schedule Management
4. Project Cost Management
5. Project Quality Management
6. Project Resource Management
7. Project Communications Management
8. Project Risk Management
9. Project Procurement Management
10. Project Stakeholder Management

Source: Project Management Institute (2017), A Guide to the Project Management Body of Knowledge (PMBOK Guide), Sixth Edition, PMI

Project Management Process Groups

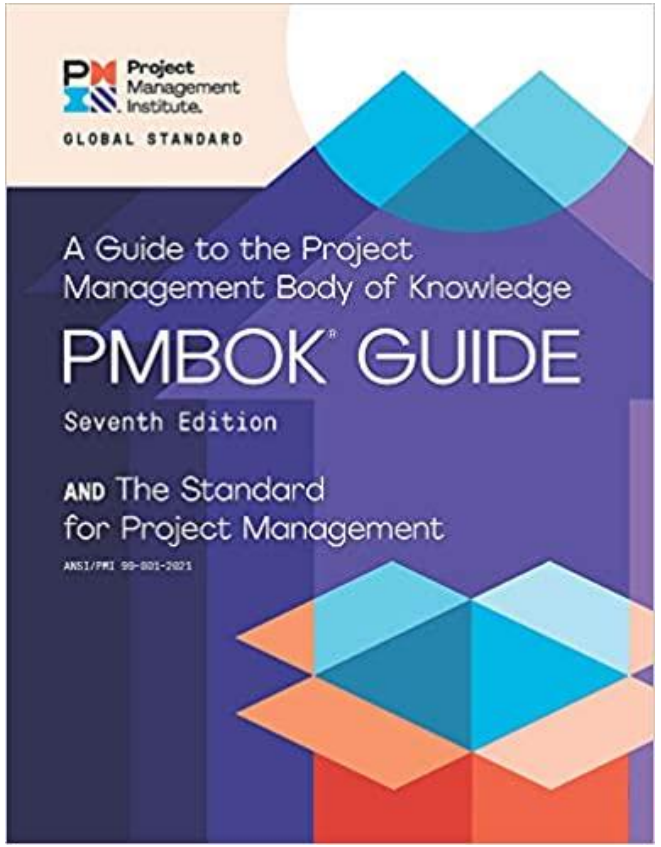
(PMBOK v6)

1. Initiating Process Group
2. Planning Process Group
3. Executing Process Group
4. Monitoring and Controlling Process Group
5. Closing Process Group

Source: Project Management Institute (2017), A Guide to the Project Management Body of Knowledge (PMBOK Guide), Sixth Edition, PMI

Project Management 12 Principles
(PMBOK v7)

1. Stewardship
2. Team
3. Stakeholders
4. Value
5. Systems Thinking
6. Leadership
7. Tailoring
8. Quality
9. Complexity
10. Risk
11. Adaptability and Resiliency
12. Change



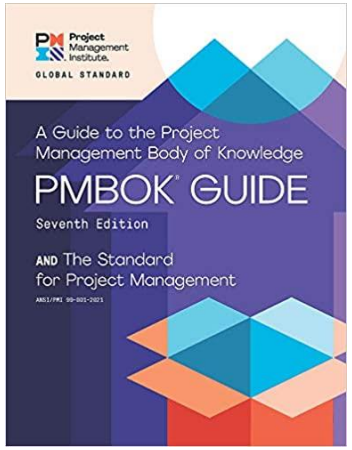
Source: Project Management Institute (2021), A Guide to the Project Management Body of Knowledge (PMBOK Guide) – Seventh Edition and The Standard for Project Management, PMI

Project Management

8 Project Performance Domains

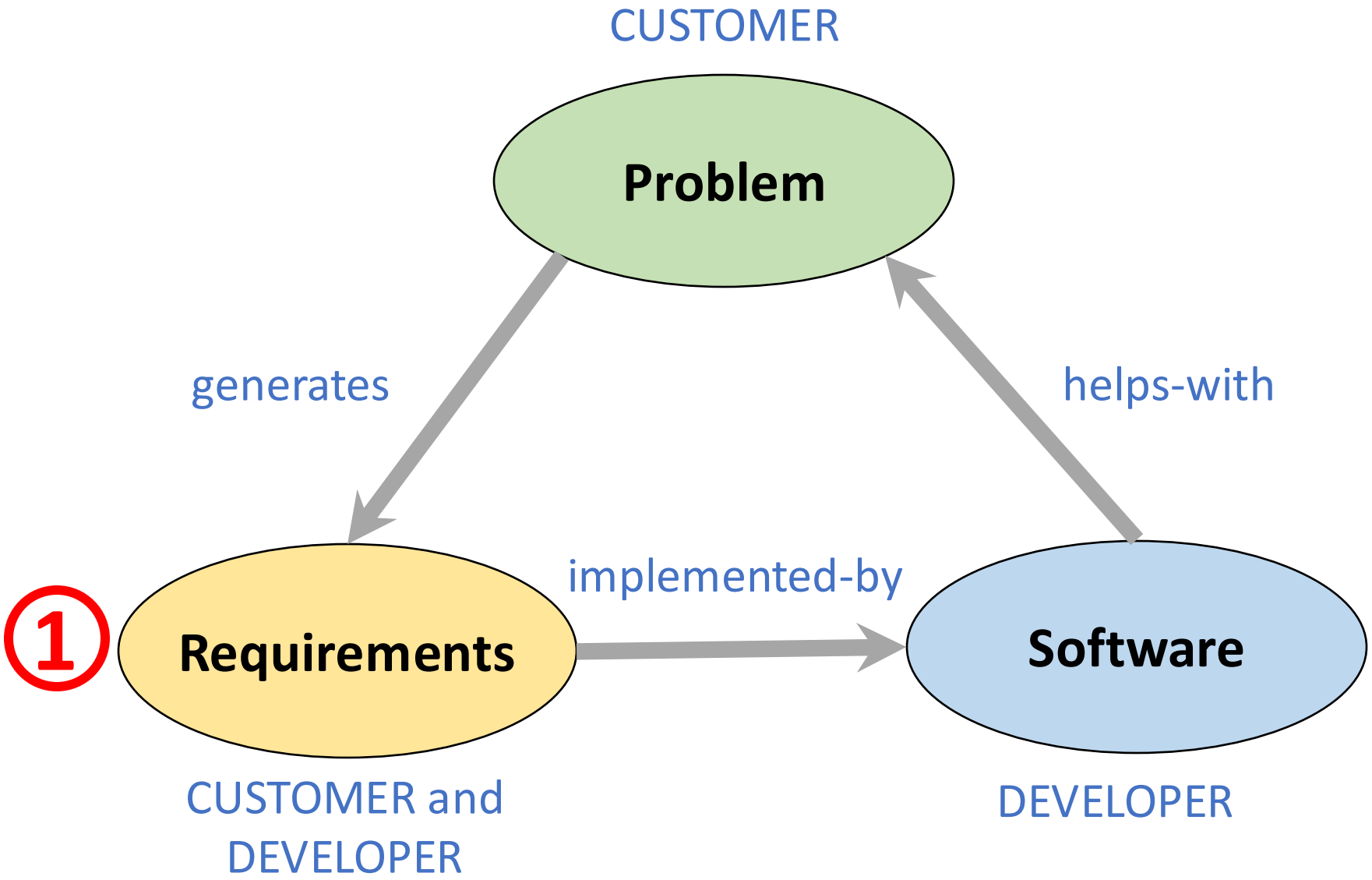
(PMBOK v7)

1. Stakeholders
2. Team
3. Development Approach and Life Cycle
4. Planning
5. Project Work
6. Delivery
7. Measurement
8. Uncertainty



Source: Project Management Institute (2021), A Guide to the Project Management Body of Knowledge (PMBOK Guide) – Seventh Edition and The Standard for Project Management, PMI

Project-based software engineering



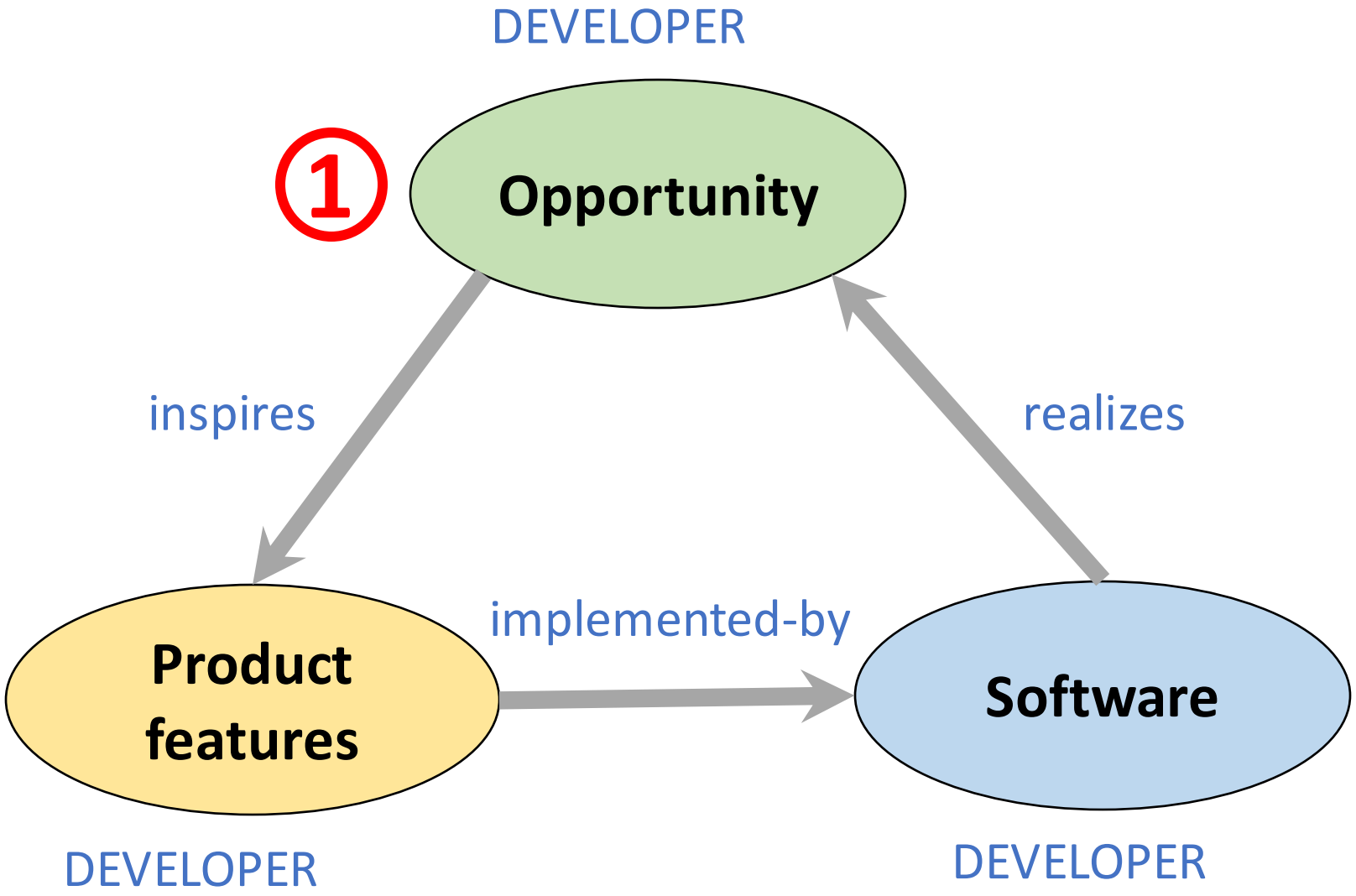
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Project-based software engineering

- The starting point for the software development is a set of 'software requirements' that are owned by an external client and which set out what they want a software system to do to support their business processes.
- The software is developed by a software company (the contractor) who design and implement a system that delivers functionality to meet the requirements.
- The customer may change the requirements at any time in response to business changes (they usually do). The contractor must change the software to reflect these requirements changes.
- Custom software usually has a long-lifetime (10 years or more) and it must be supported over that lifetime.

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

Product software engineering



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

Product software engineering

- The starting point for product development is a business opportunity that is identified by individuals or a company.
  They develop a software product to take advantage of this opportunity and sell this to customers.
- The company who identified the opportunity design and implement a set of software features that realize the opportunity and that will be useful to customers.
- The software development company are responsible for deciding on the development timescale, what features to include and when the product should change.
- Rapid delivery of software products is essential to capture the market for that type of product.

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

Software product line

- **A set of software products that share a common core.**
- **Each member of the product line includes customer-specific adaptations and additions.**
- **Software product lines may be used to implement a custom system for a customer with specific needs that can’t be met by a generic product.**

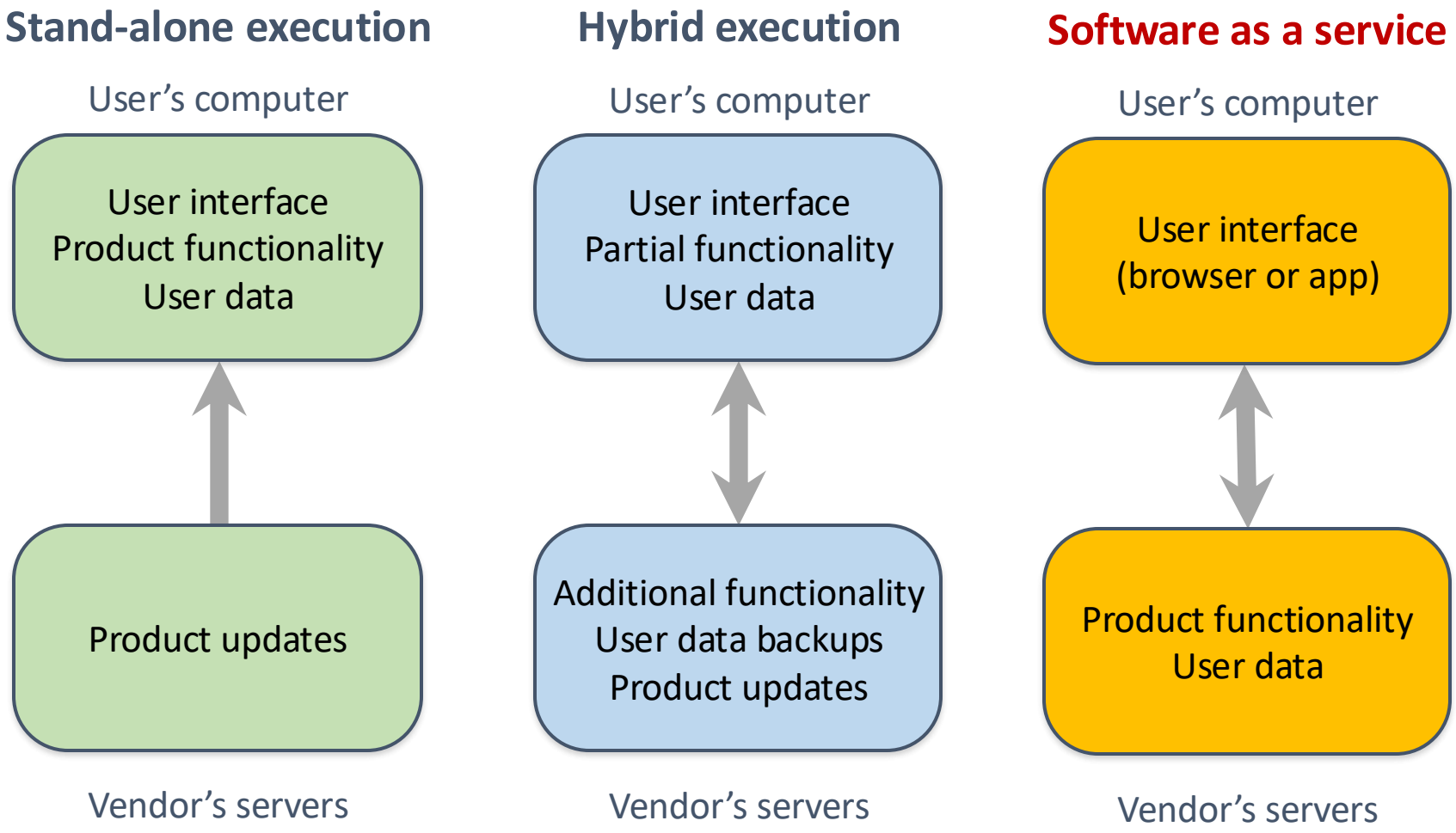
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Platform

- A software (or software + hardware) product that includes functionality so that new applications can be built on it.
- An example of a platform that you probably use is Facebook.
- It provides an extensive set of product functionality but also provides support for creating ‘Facebook apps’.
- These add new features that may be used by a business or a Facebook interest group.

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

Software execution models



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

Software execution models

- **Stand-alone**
  - The software executes entirely on the customer's computers.
- **Hybrid**
  - Part of the software's functionality is implemented on the customer's computer but some features are implemented on the product developer's servers.
- **Software service**
  - All of the product's features are implemented on the developer's servers and the customer accesses these through a browser or a mobile app.

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

Comparable software development

- The key feature of product development is that there is no external customer that generates requirements and pays for the software.
- **Student projects**
  - Individuals or student groups develop software as part of their course. Given an assignment, they decide what features to include in the software.
- **Research software**
  - Researchers develop software to help them answer questions that are relevant to their research.
- **Internal tool development**

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

The product vision

- The starting point for software product development is a ‘product vision’.
- Product visions are simple statements that define the essence of the product to be developed.
- The product vision should answer three fundamental questions:
  - What is the product to be developed?
  - Who are the target customers and users?
  - Why should customers buy this product?

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

Moore's vision template

- **FOR (target customer)**
- **WHO (statement of the need or opportunity)**
- **The (PRODUCT NAME) is a (product category)**
- **THAT (key benefit, compelling reason to buy)**
- **UNLIKE (primary competitive alternative)**
- **OUR PRODUCT (statement of primary differentiation)**

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

Vision template example

- "**FOR** a mid-sized company's marketing and sales departments
  WHO need basic CRM functionality,
  THE CRM-Innovator **is** a Web-based service
  THAT provides sales tracking, lead generation, and sales representative support features that improve customer relationships at critical touch points.
  UNLIKE other services or package software products,
  OUR product provides very capable services at a moderate cost."

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

Information sources for developing a product vision

- **Domain experience**
- **Product experience**
- **Customer experience**
- **Prototyping and playing around**

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

Information sources for developing a product vision

- **Domain experience**
  - The product developers may work in a particular area (say marketing and sales) and understand the software support that they need.
  - They may be frustrated by the deficiencies in the software they use and see opportunities for an improved system.

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

Information sources for developing a product vision

- **Product experience**
  - Users of existing software (such as word processing software) may see simpler and better ways of providing comparable functionality and propose a new system that implements this.
  - New products can take advantage of recent technological developments such as speech interfaces.

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

Information sources for developing a product vision

- **Customer experience**
  - The software developers may have extensive discussions with prospective customers of the product to understand the problems that they face, constraints, such as interoperability, that limit their flexibility to buy new software, and the critical attributes of the software that they need.

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

Information sources for developing a product vision

- **Prototyping and playing around**
  - Developers may have an idea for software but need to develop a better understanding of that idea and what might be involved in developing it into a product.
  - They may develop a prototype system as an experiment and ‘play around’ with ideas and variations using that prototype system as a platform.

A vision statement for the iLearn system

- FOR teachers and educators WHO need a way to help students use web-based learning resources and applications, THE iLearn system is an open learning environment THAT allows the set of resources used by classes and students to be easily configured for these students and classes by teachers themselves. UNLIKE Virtual Learning Environments, such as Moodle, the focus of iLearn is the learning process rather than the administration and management of materials, assessments and coursework. OUR product enables teachers to create subject and age-specific environments for their students using any web-based resources, such as videos, simulations and written materials that are appropriate.
- Schools and universities are the target customers for the iLearn system as it will significantly improve the learning experience of students at relatively low cost. It will collect and process learner analytics that will reduce the costs of progress tracking and reporting.

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

The Essence of
Strategic Marketing (STP)

Segmentation

Targeting

Positioning

Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

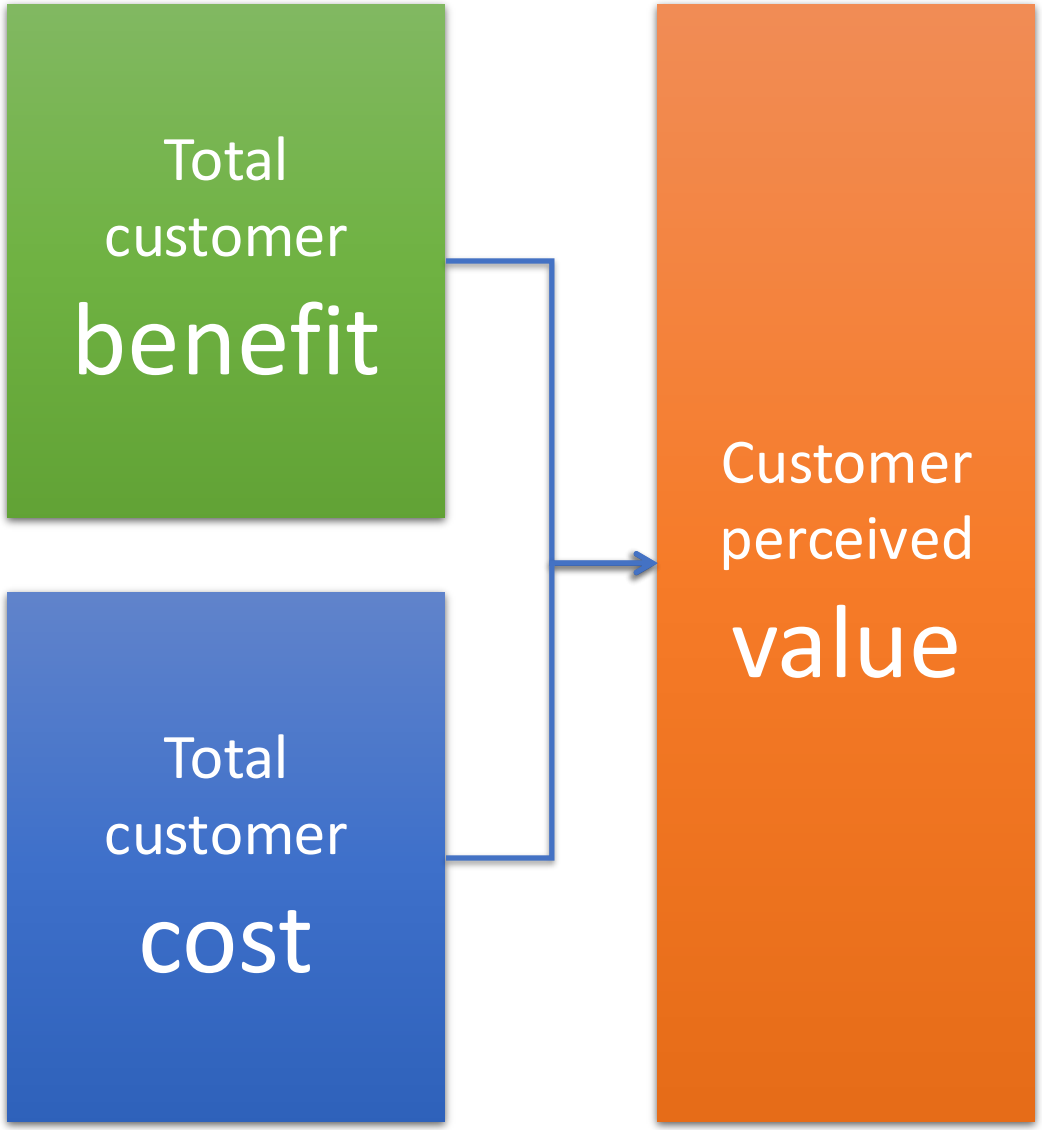
Customer Value

Value

the sum of the tangible and intangible benefits and costs

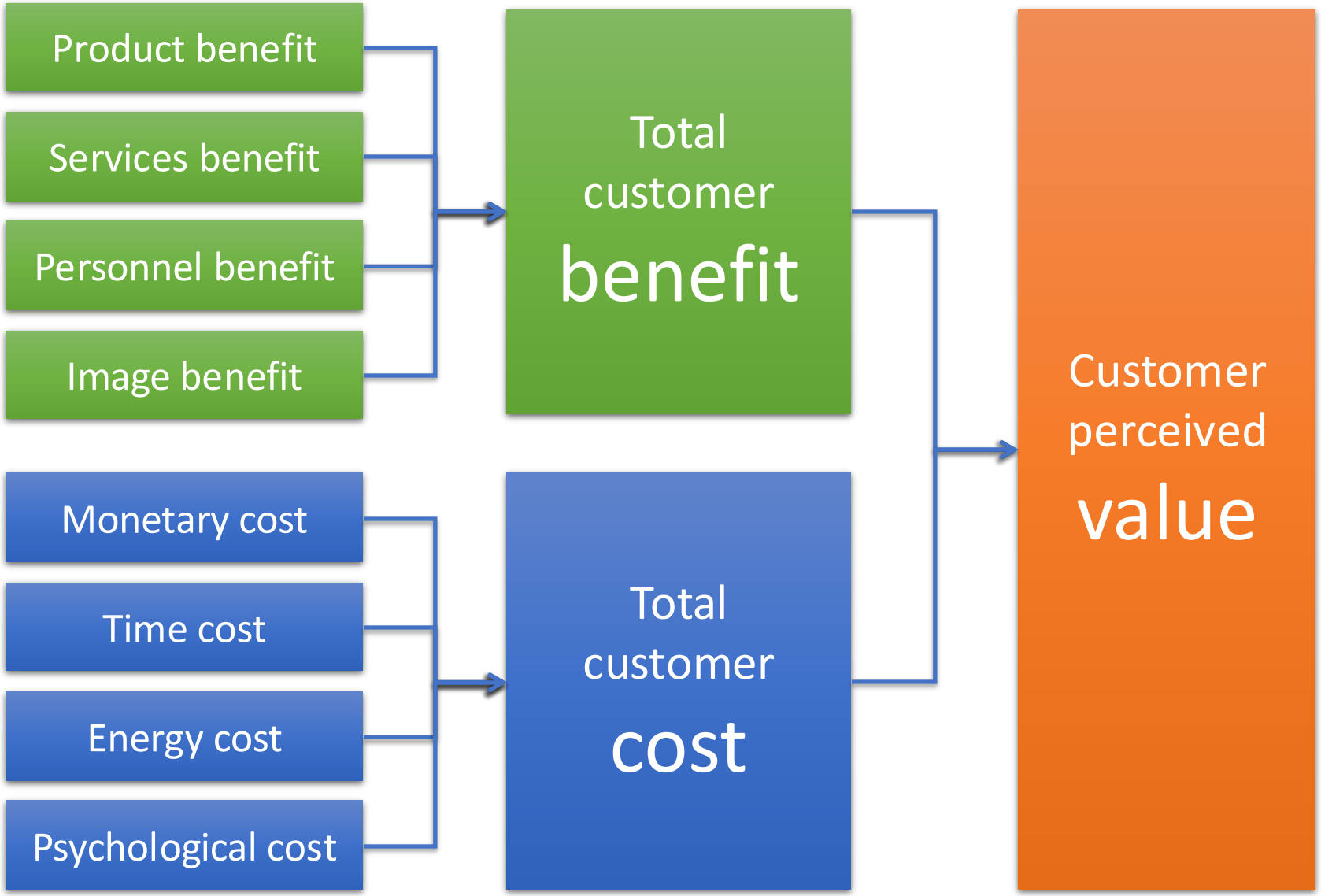
Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

Value



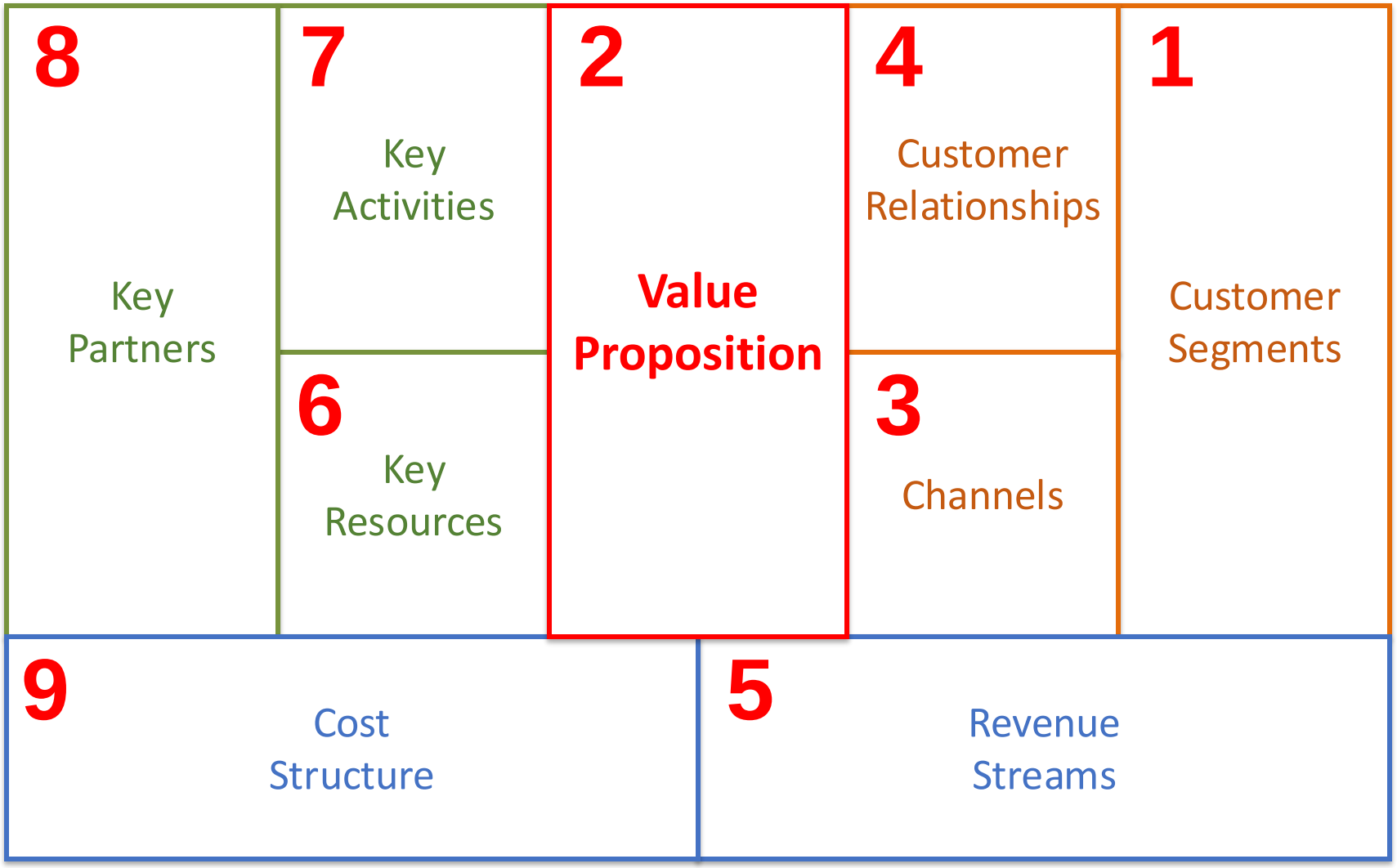
Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

Customer Perceived Value



Source: Philip Kotler and Kevin Lane Keller (2016), Marketing Management, 15th edition, Pearson.

Business Model



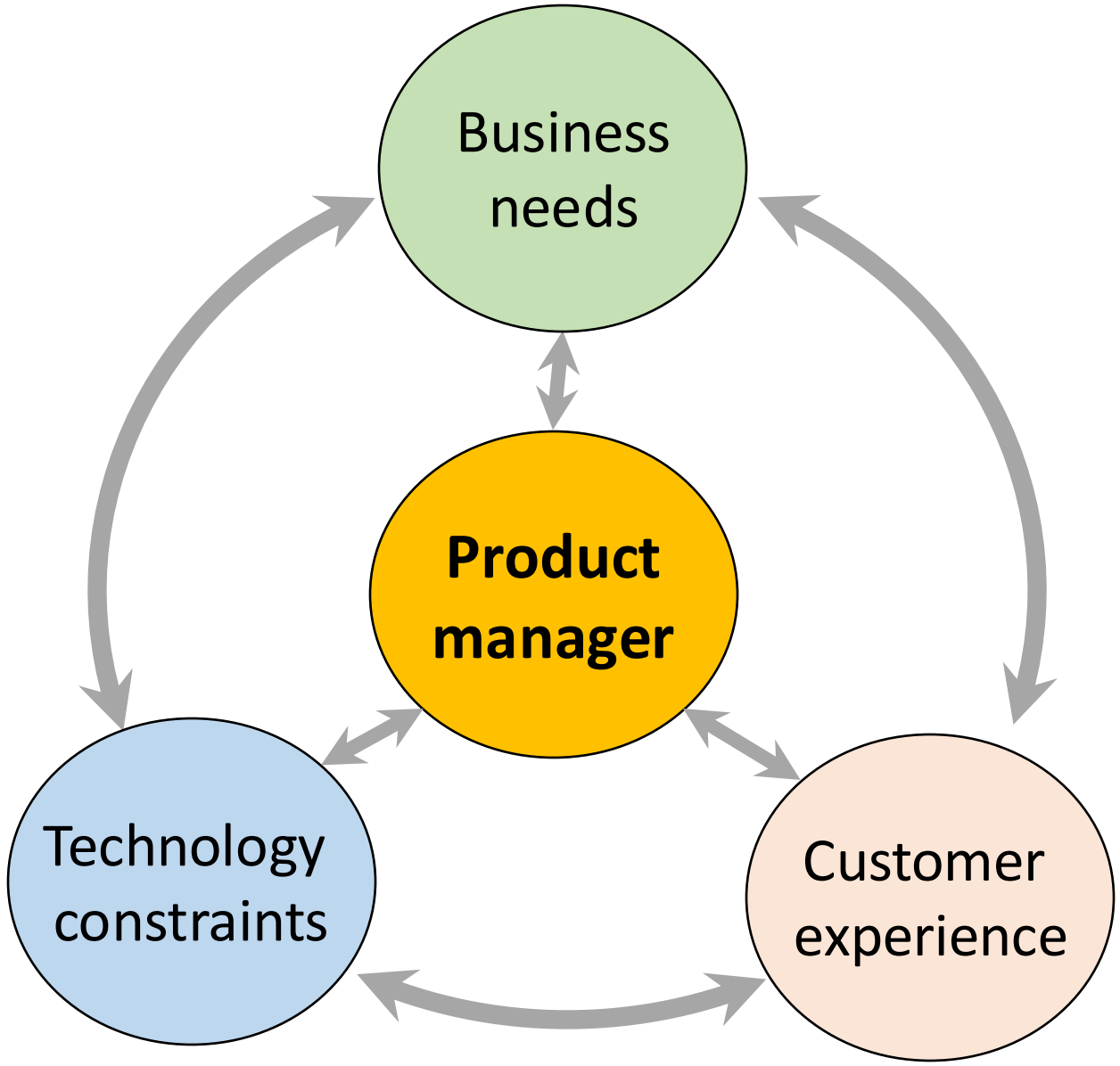
Source: Alexander Osterwalder & Yves Pigneur, Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers, Wiley, 2010.

Software product management

- **Software product management is a business activity that focuses on the software products developed and sold by the business.**
- **Product managers (PMs) take overall responsibility for the product and are involved in planning, development and product marketing.**
- **Product managers are the interface between the organization, its customers and the software development team. They are involved at all stages of a product's lifetime from initial conception through to withdrawal of the product from the market.**
- **Product managers must look outward to customers and potential customers rather than focus on the software being developed**

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

Product management concerns



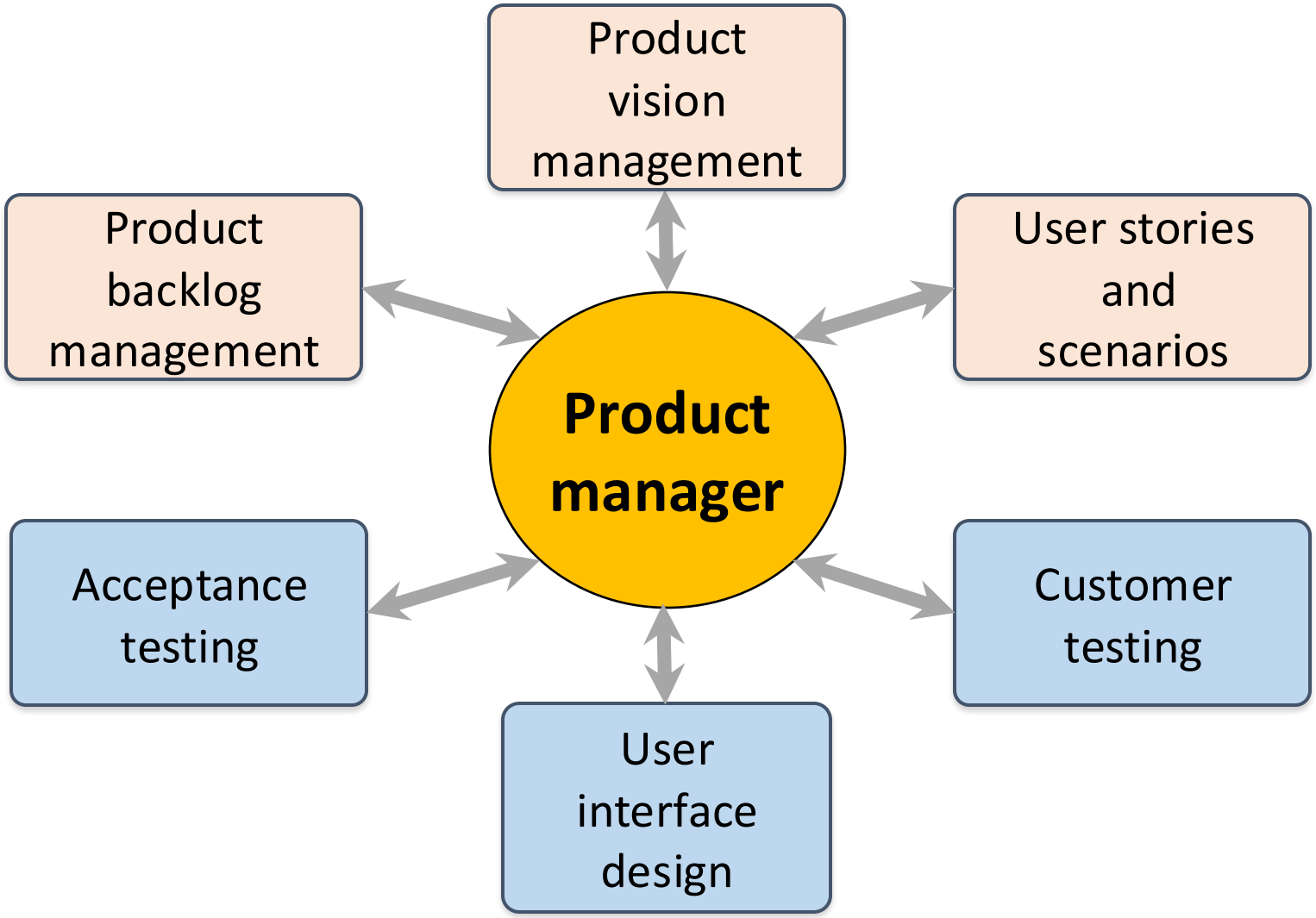
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Product management concerns

- **Business needs**
  - **PMs have to ensure that the software being developed meets the business goals of the software development company.**
- **Technology constraints**
  - **PMs must make developers aware of technology issues that are important to customers.**
- **Customer experience**
  - **PMs should be in regular contact with customers and potential customers to understand what they are looking for in a product, the types of users and their backgrounds and the ways that the product may be used.**

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.

Technical interactions of product managers

- **Product vision management**
  - **The product manager may be responsible for helping with the development of the product vision.**
  - **The should always be responsible for managing the vision, which involves assessing and evaluating proposed changes against the product vision.**
  - **They should ensure that there is no ‘vision drift’**

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

Technical interactions of product managers

- **Product roadmap development**
  - **A product roadmap is a plan for the development, release and marketing of the software.**
  - **The PM should lead roadmap development and should be the ultimate authority in deciding if changes to the roadmap should be made.**

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

Technical interactions of product managers

- **User story and scenario development**
  - **User stories and scenarios are used to refine a product vision and identify product features.**
  - **Based on his or her knowledge of customers, the PM should lead the development of stories and scenarios.**

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

Technical interactions of product managers

- **Product backlog creation and management**
  - **The product backlog is a prioritized ‘to-do’ list of what has to be developed.**
  - **PMs should be involved in creating and refining the backlog and deciding on the priority of product features to be developed.**

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

Technical interactions of product managers

- **Acceptance testing**
  - **Acceptance testing is the process of verifying that a software release meets the goals set out in the product roadmap and that the product is efficient and reliable.**
  - **The PM should be involved in developing tests of the product features that reflect how customers use the product.**

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

Technical interactions of product managers

- **Customer testing**
  - **Customer testing involves taking a release of a product to customers and getting feedback on the product's features, usability and business.**
  - **PMs are involved in selecting customers to be involved in the customer testing process and working with them during that process.**

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

Technical interactions of product managers

- **User interface design**
  - **Product managers should understand user limitations and act as surrogate users in their interactions with the development team.**
  - **They should evaluate user interface features as they are developed to check that these features are not unnecessarily complex or force users to work in an unnatural way.**

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

Product prototyping

- **Product prototyping is the process of developing an early version of a product to test your ideas and to convince yourself and company funders that your product has real market potential.**

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

Product prototyping

- **You may be able to write an inspiring product vision, but your potential users can only really relate to your product when they see a working version of your software.**
- **They can point out what they like and don't like about it and make suggestions for new features.**
- **A prototype may be also used to help identify fundamental software components or services and to test technology.**

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

Product prototyping

- **Building a prototype should be the first thing that you do when developing a software product.**
- **Your aim should be to have a working version of your software that can be used to demonstrate its key features.**
- **You should always plan to throw-away the prototype after development and to re-implement the software, taking account of issues such as security and reliability.**

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

Two-stage prototyping

1. Feasibility demonstration

- **You create an executable system that demonstrates the new ideas in your product.**
- **The aims at this stage are to see if your ideas actually work and to show funders and/or company management the original product features that are better than those in competing products.**

2. Customer demonstration

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

Two-stage prototyping

1. **Feasibility demonstration**
2. **Customer demonstration**
   - **You take an existing prototype created to demonstrate feasibility and extend this with your ideas for specific customer features and how these can be realized.**
   - **Before you develop this type of prototype, you need to do some user studies and have a clearer idea of your potential users and scenarios of use.**

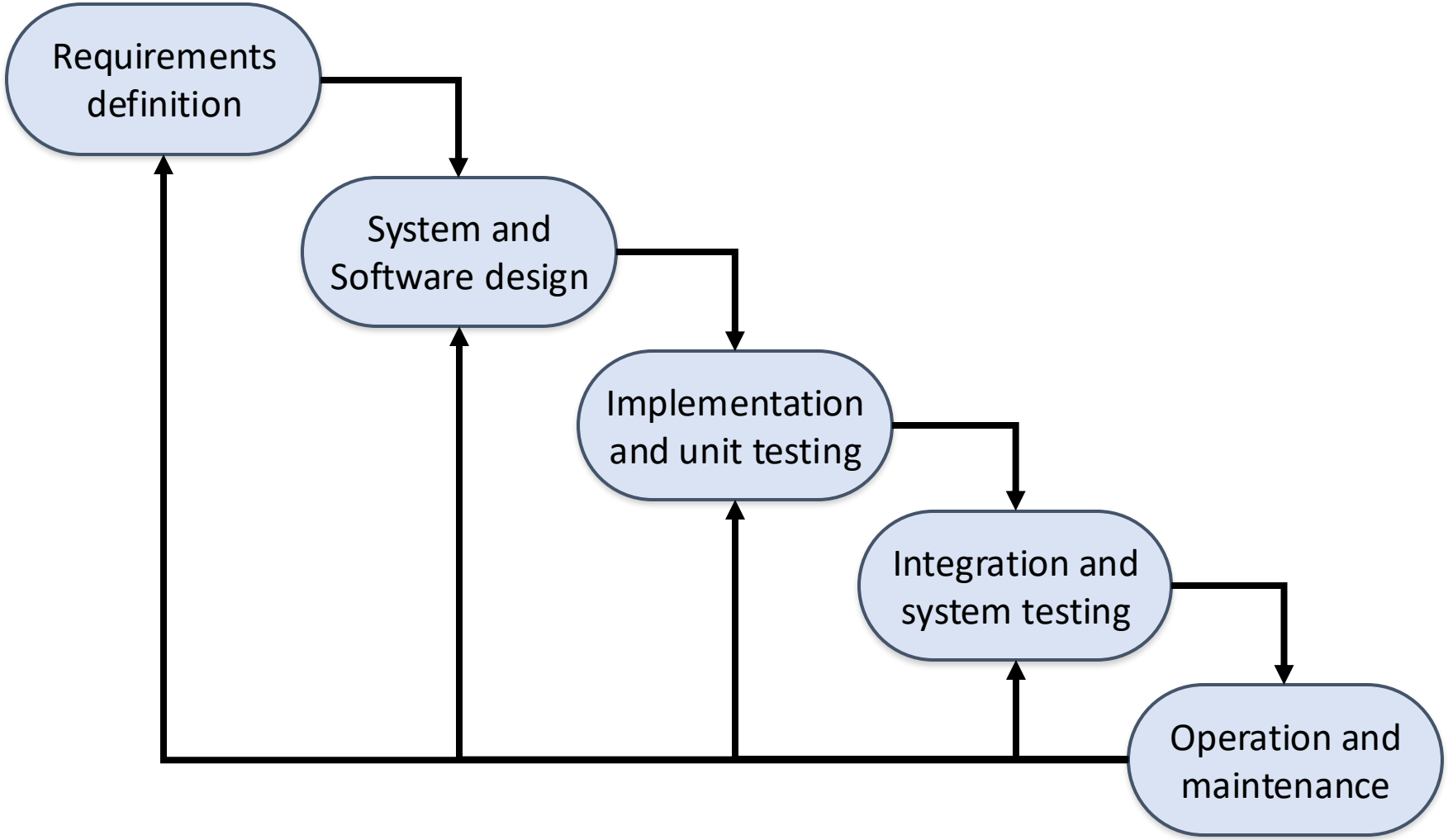
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Software process models

- **The waterfall model**
  - **This takes the fundamental process activities of specification, development, validation, and evolution and represents them as separate process phases such as requirements specification, software design, implementation, and testing.**
- **Incremental development**
  - **This approach interleaves the activities of specification, development, and validation. The system is developed as a series of versions (increments), with each version adding functionality to the previous version.**
- **Integration and configuration**
  - **This approach relies on the availability of reusable components or systems. The system development process focuses on configuring these components for use in a new setting and integrating them into a system.**

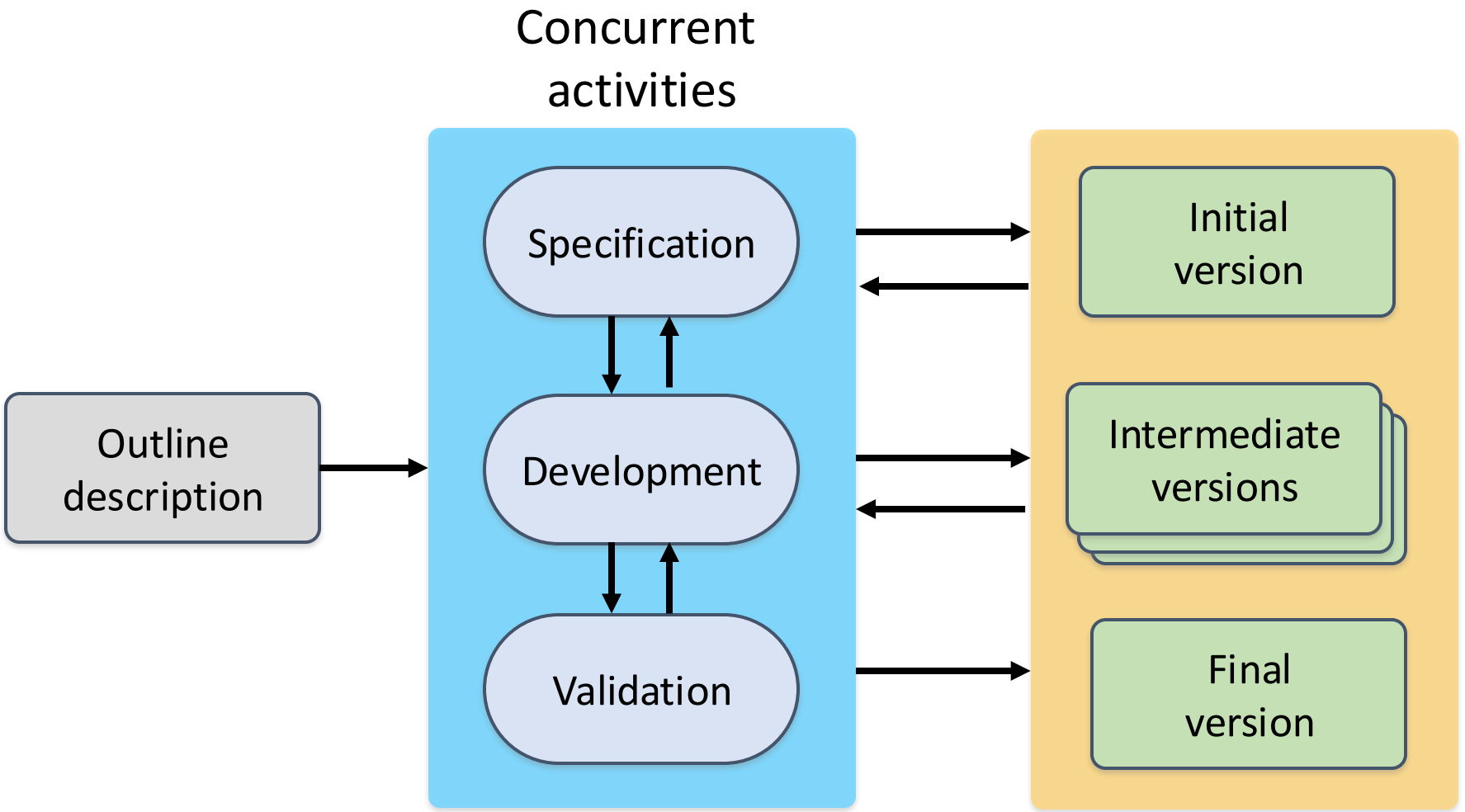
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Software Development Life Cycle (SDLC)
The waterfall model



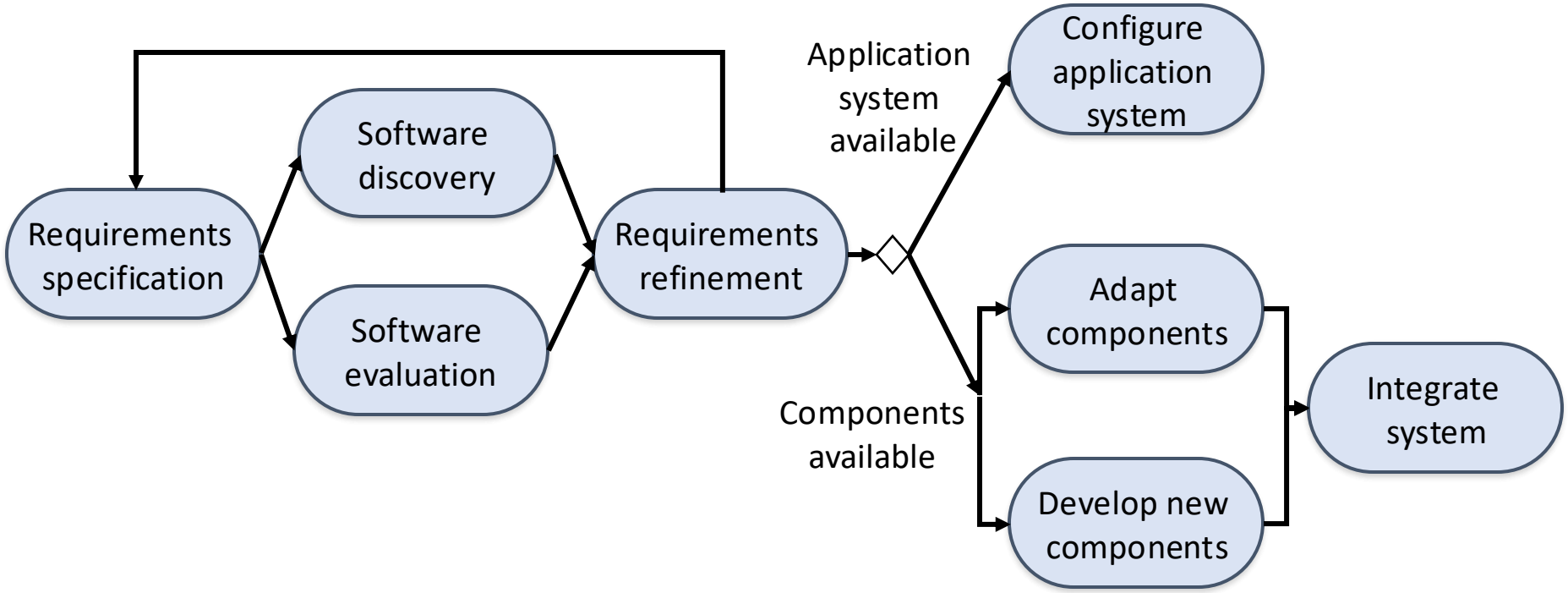
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Incremental development



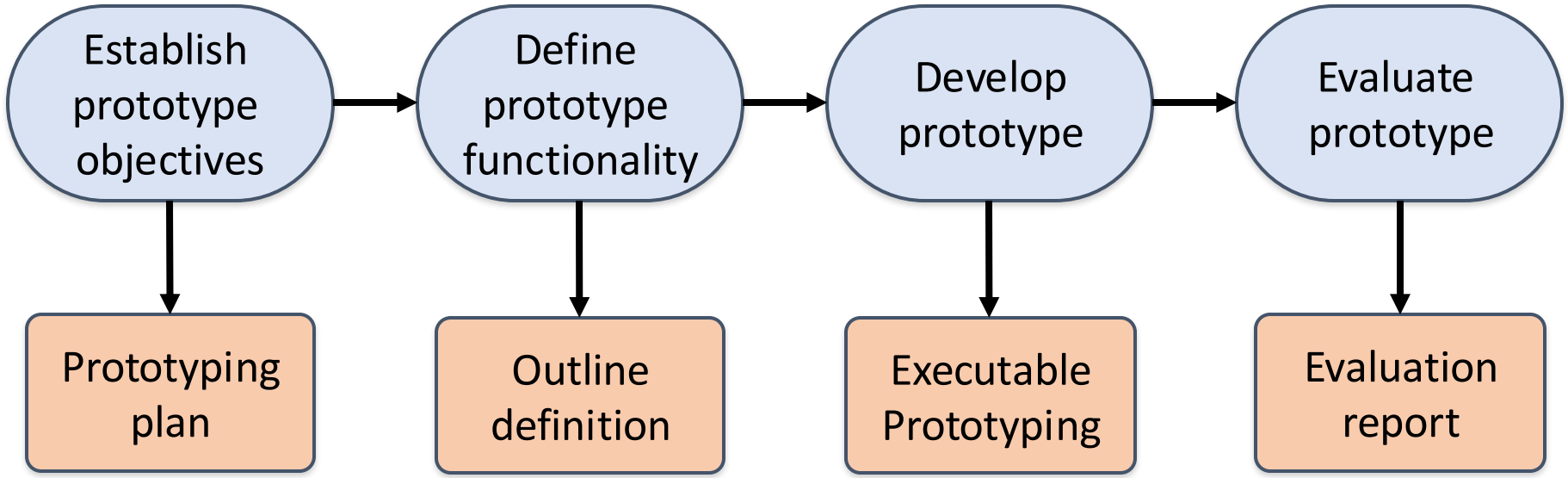
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Reuse-oriented software engineering



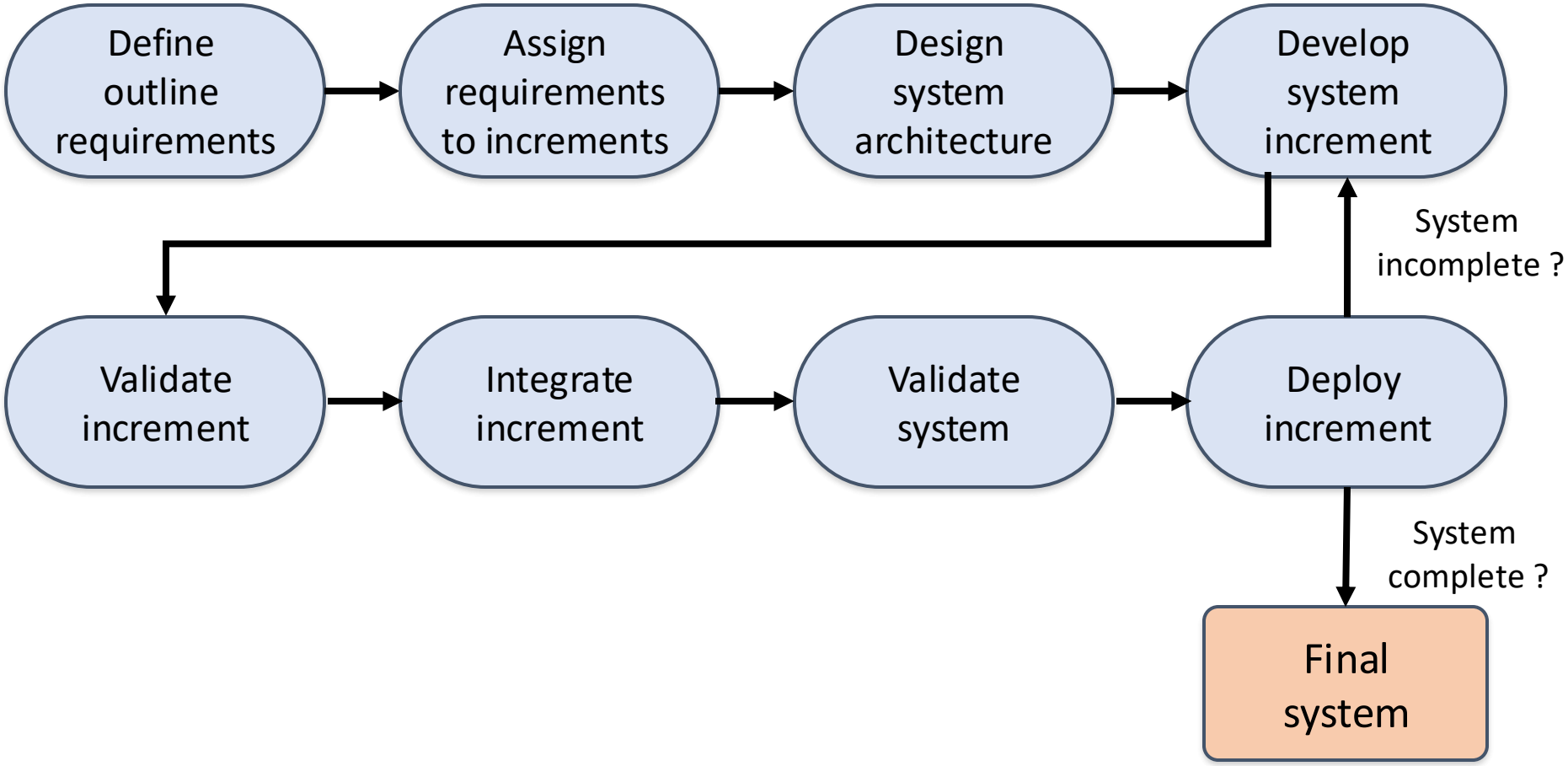
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Prototype development



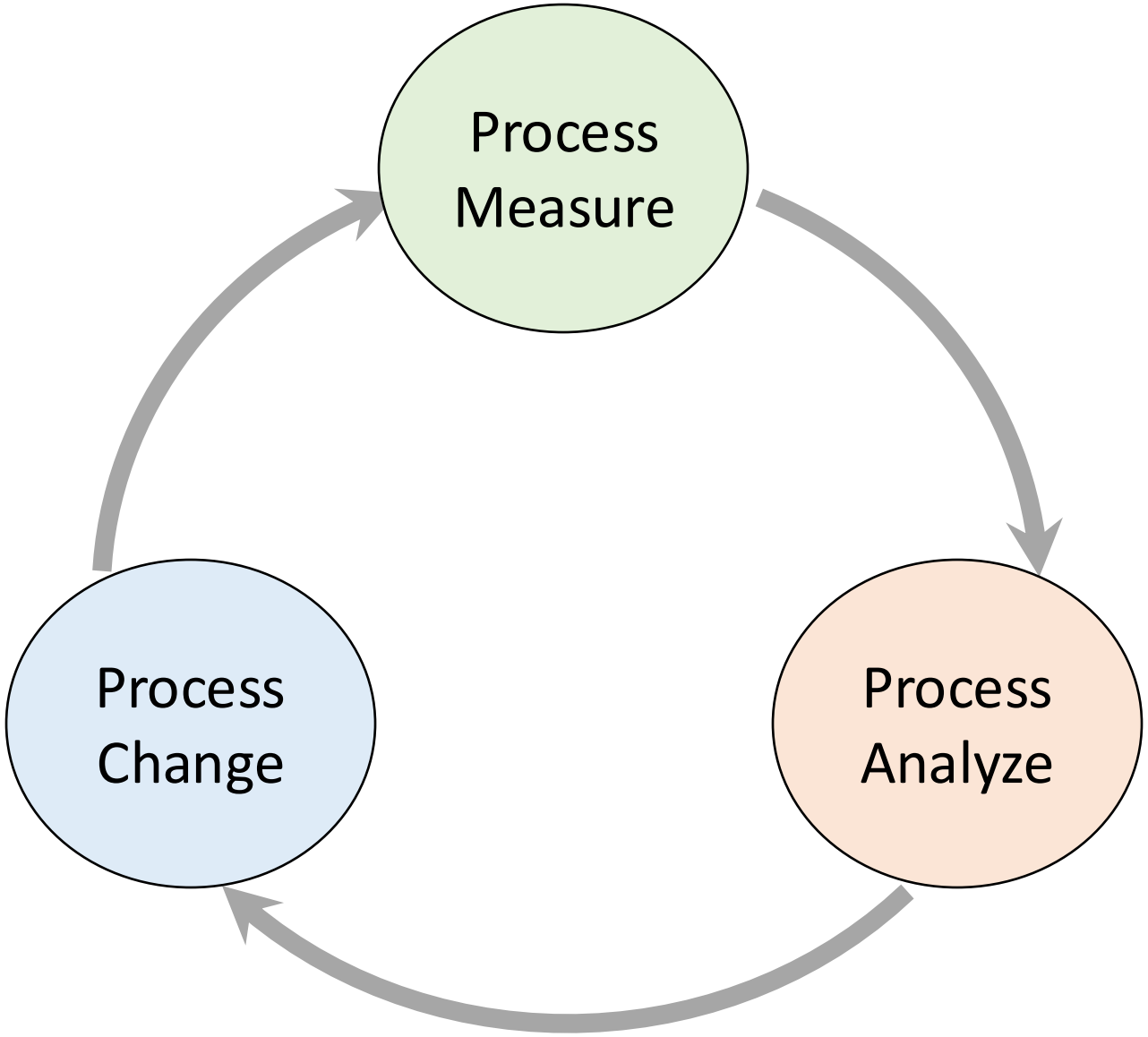
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Incremental delivery



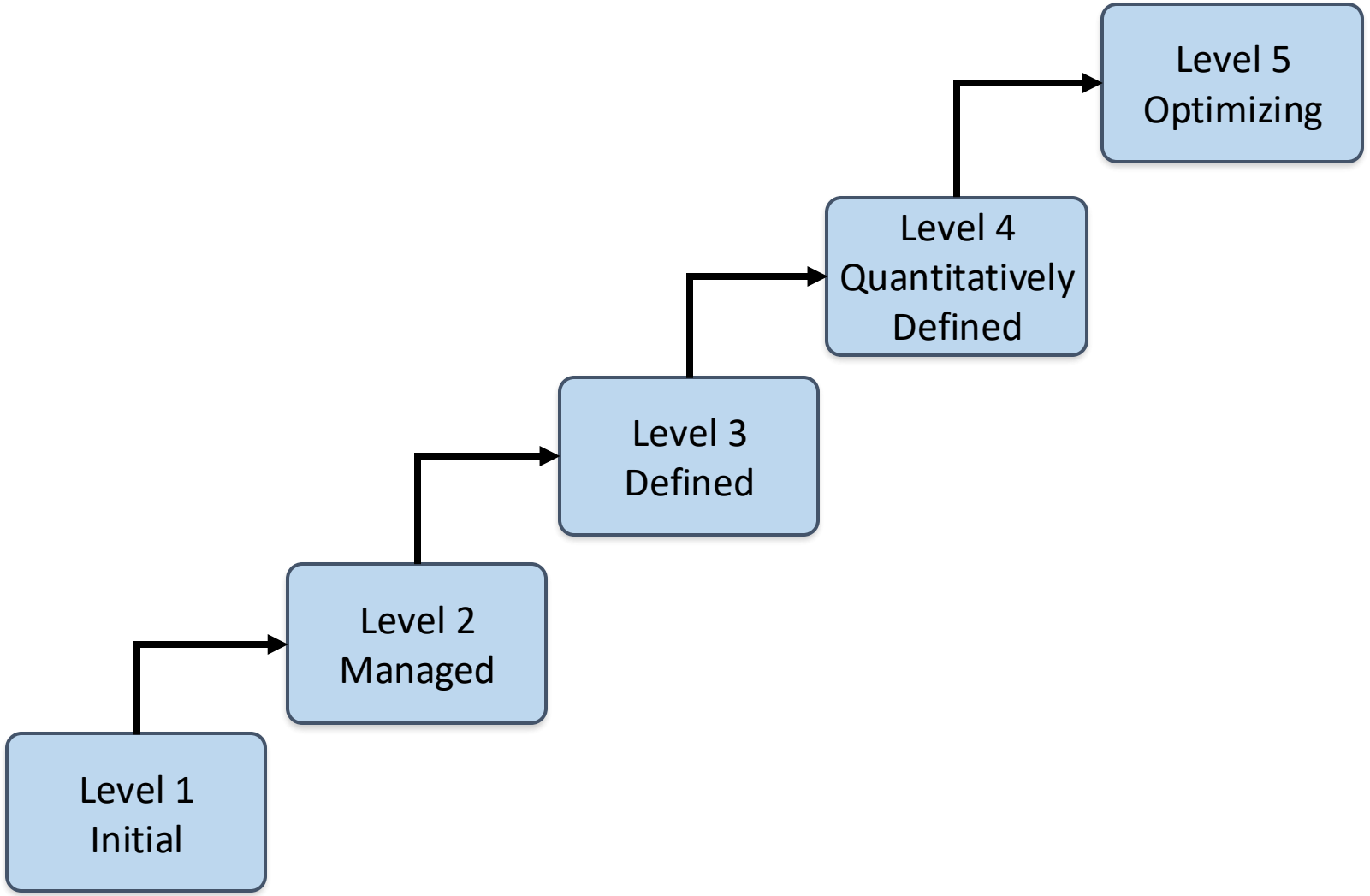
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

The process improvement model



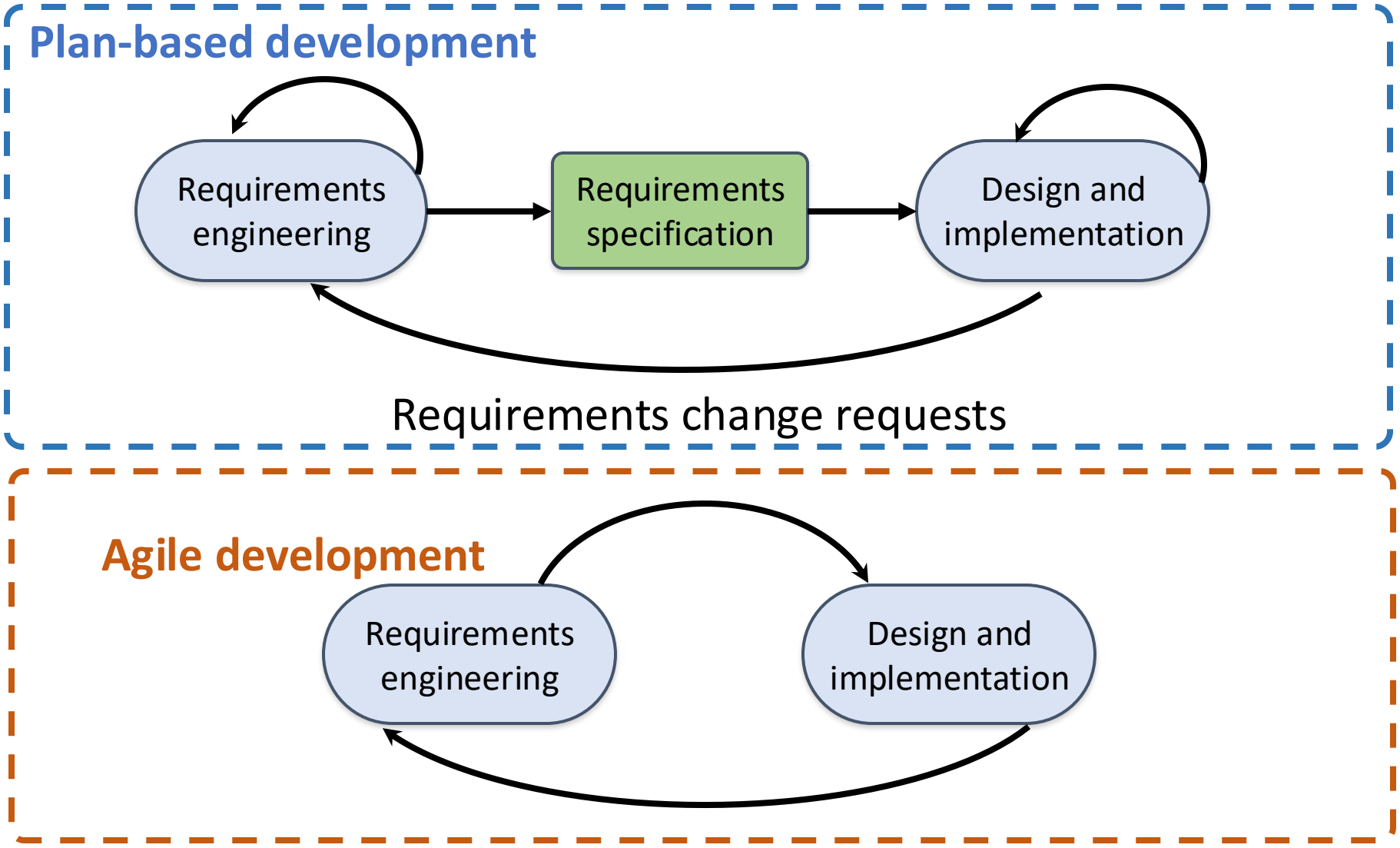
Source: Ian Sommerville (2015), Software Engineering, 10th Edition, Pearson.

Capability maturity levels



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

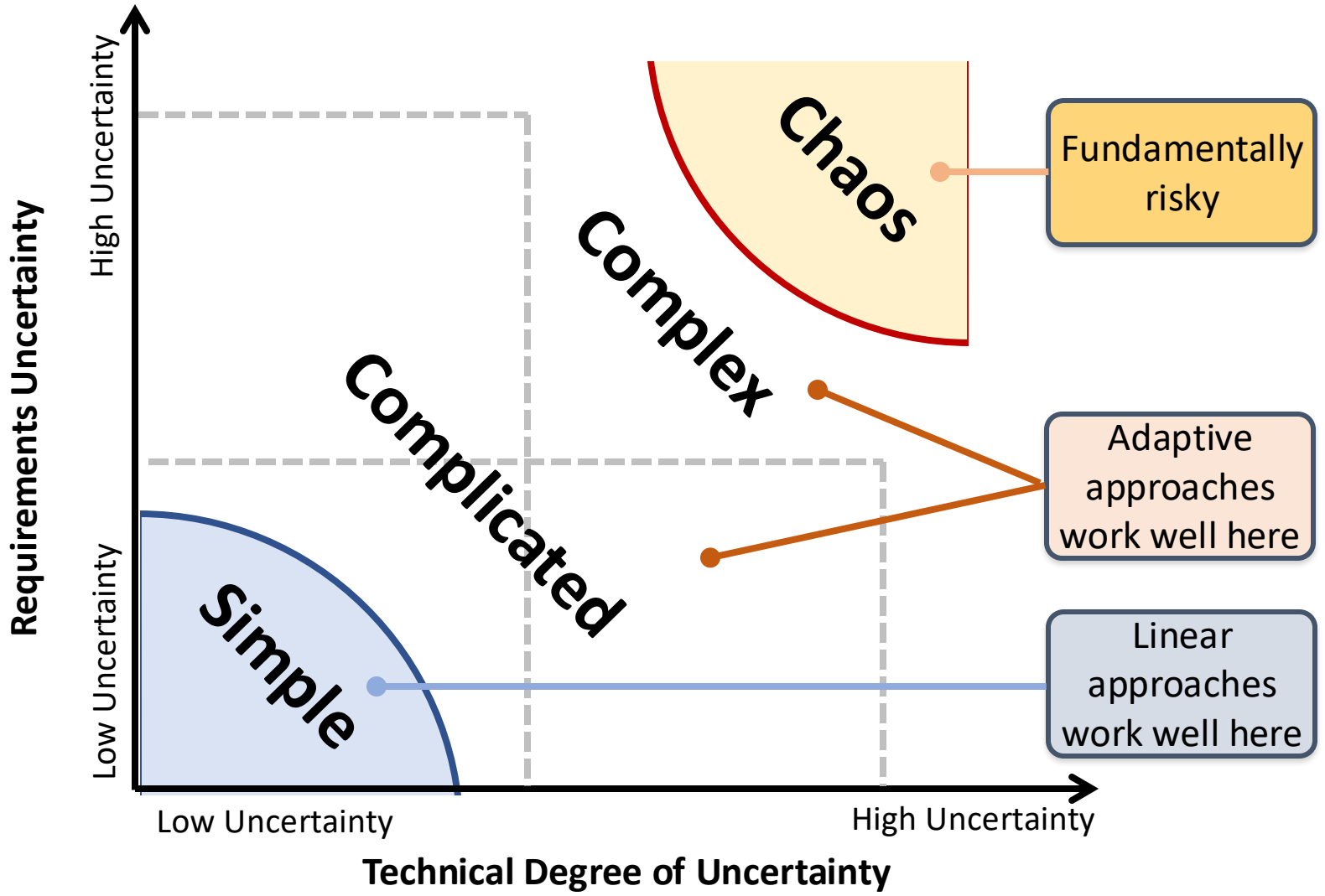
Plan-based and Agile development



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

Uncertainty and Complexity Model

Inspired by the Stacey Complexity Model



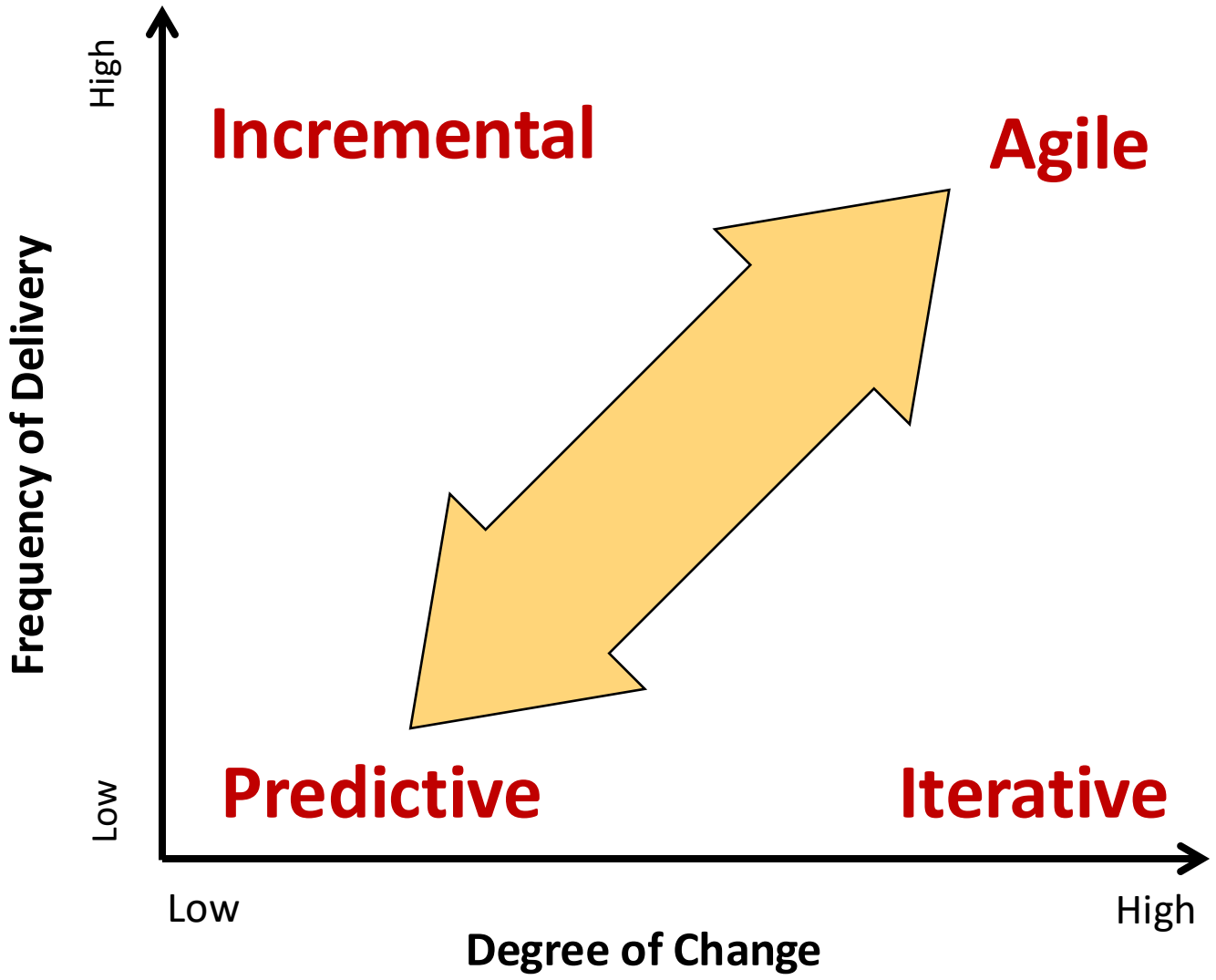
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Characteristics of Four Categories of Life Cycles

Approach	Requirements	Activities	Delivery	Goal
Predictive	Fixed	Performed once for the entire project	Single delivery	Manage cost
Iterative	Dynamic	Repeated until correct	Single delivery	Correctness of solution
Incremental	Dynamic	Performed once for a given increment	Frequent smaller deliveries	Speed
Agile	Dynamic	Repeated until correct	Frequent smaller deliveries	Customer value via frequent deliveries and feedback

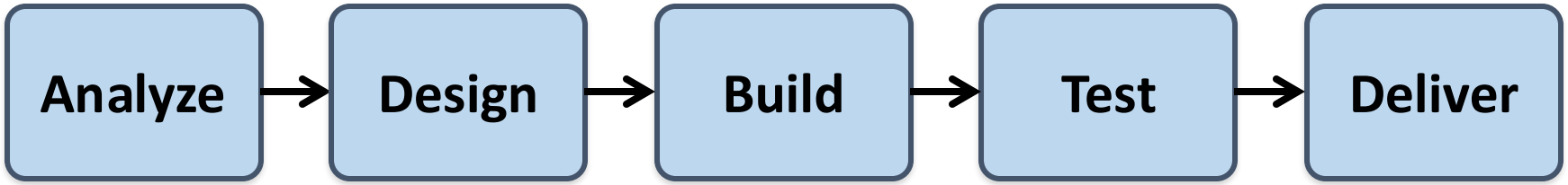
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

The Continuum of Life Cycles



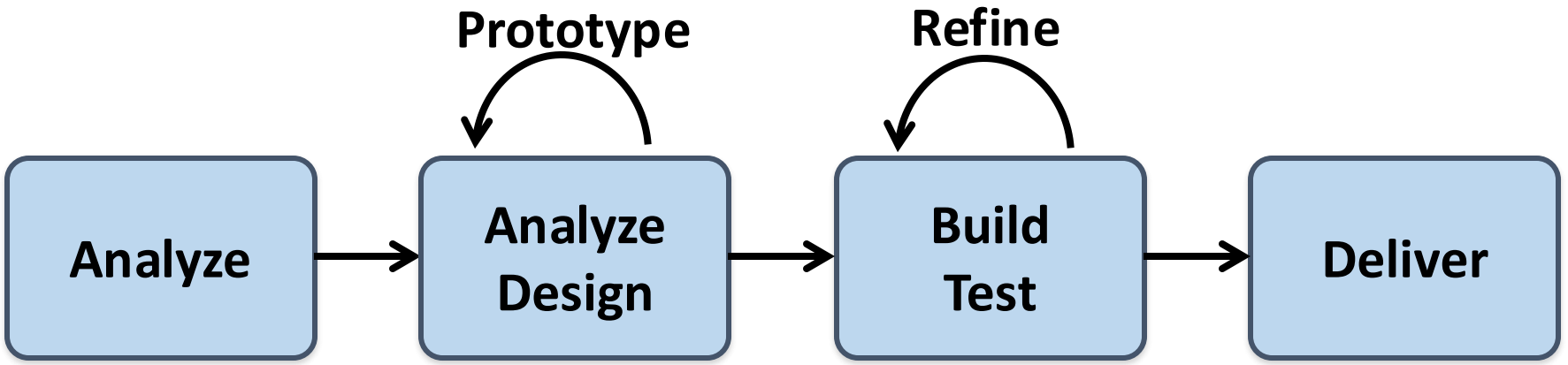
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Predictive Life Cycle



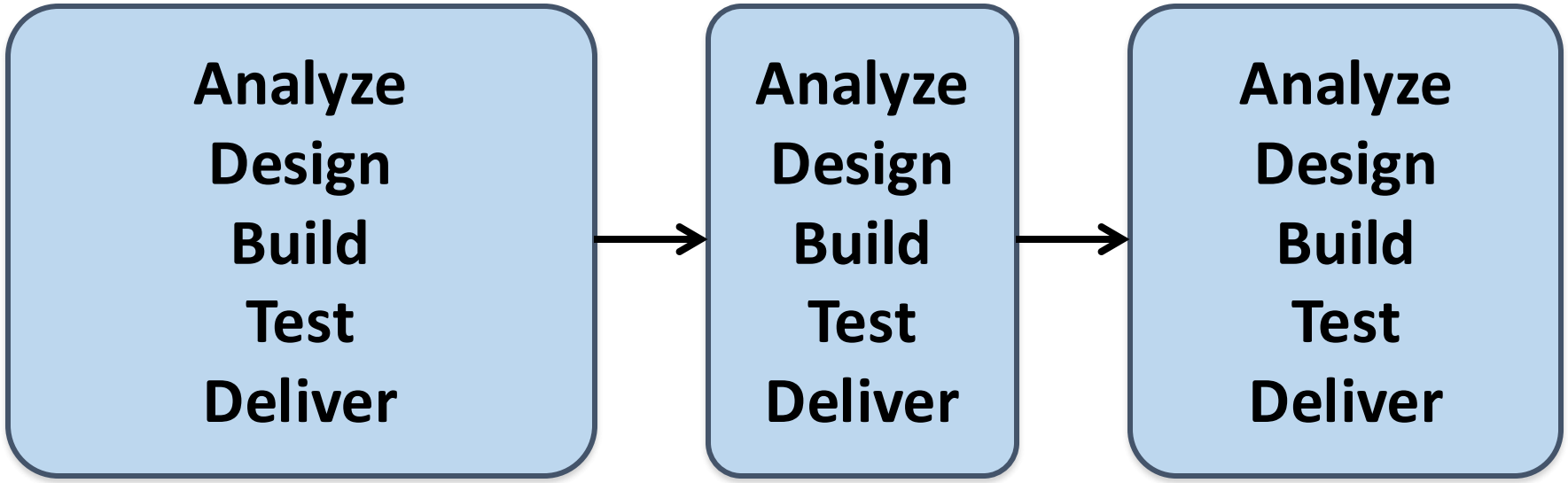
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Iterative Life Cycle



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

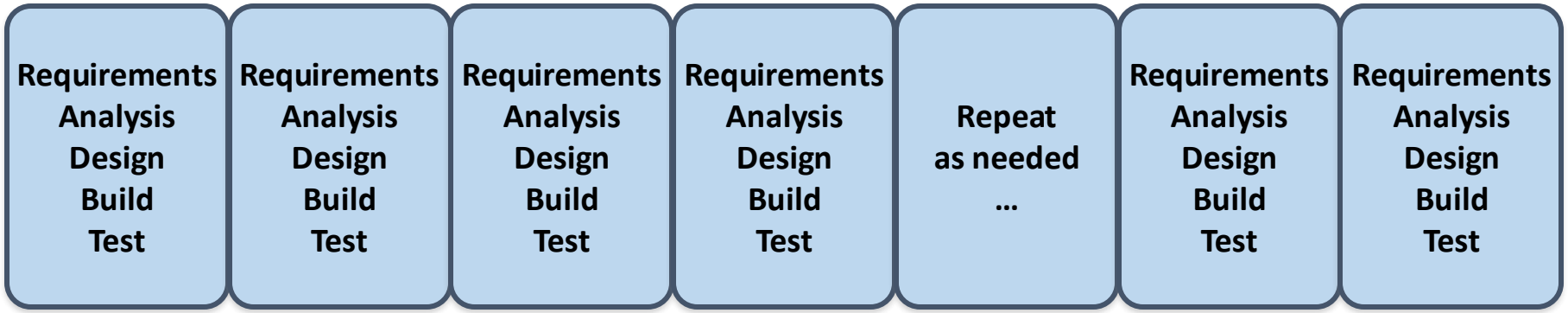
A Life Cycle of Varying-Sized Increments



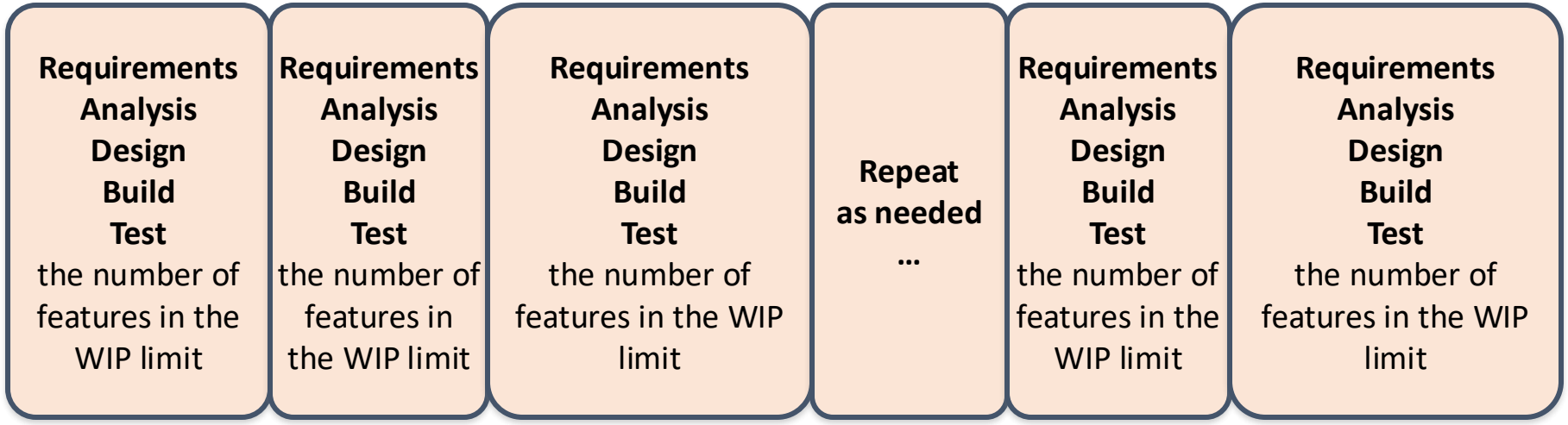
Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Iteration-Based and Flow-Based Agile Life Cycles

Iteration-Based Agile



Flow-Based Agile



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

Agentic AI

for

Agile AI Software Engineering

NVIDIA Developer Program

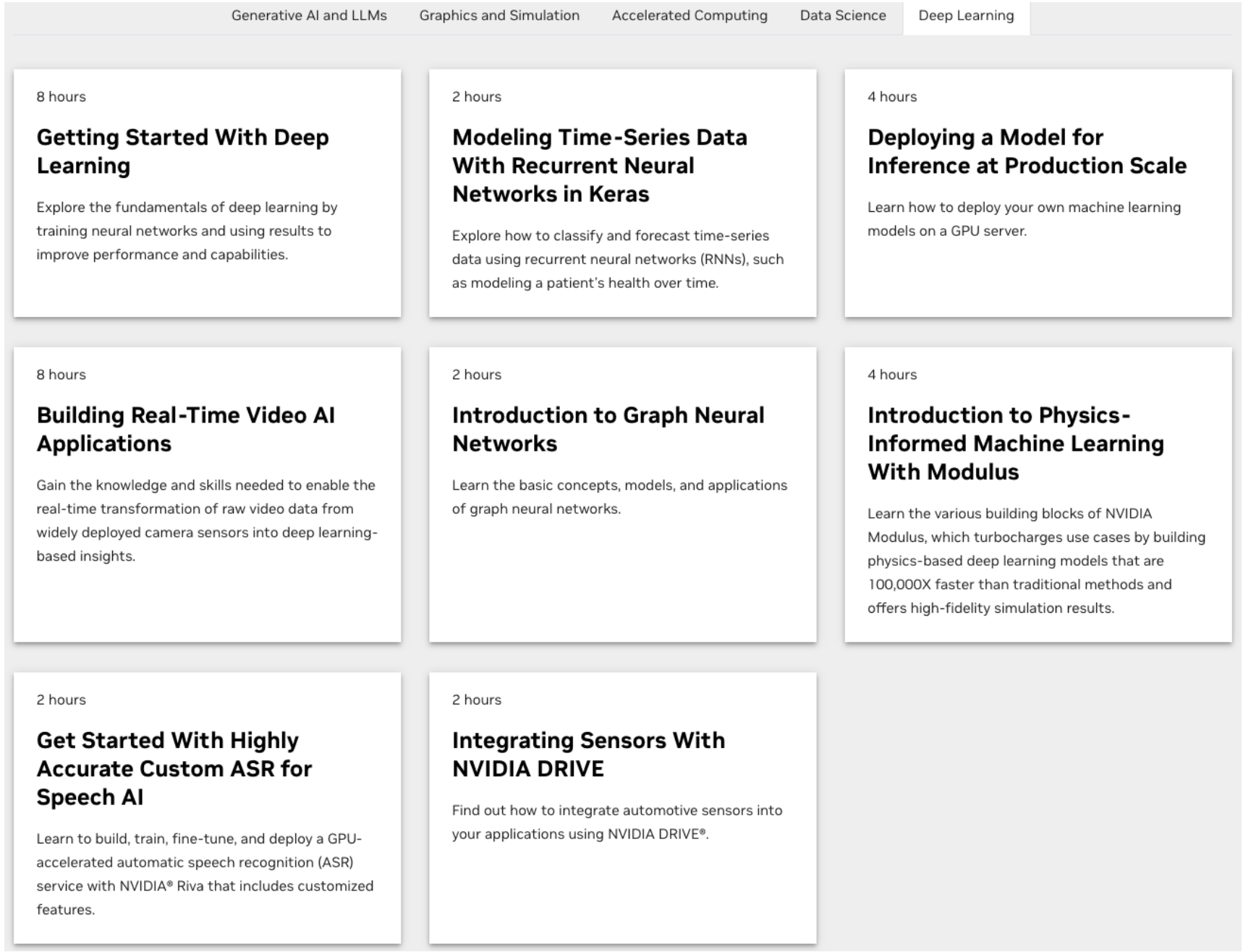
https://developer.nvidia.com/join-nvidia-developer-program

NVIDIA Deep Learning Institute (DLI)

https://learn.nvidia.com/

Join the NVIDIA Developer Program

take one of the complimentary technical self-paced courses (worth up to $90)



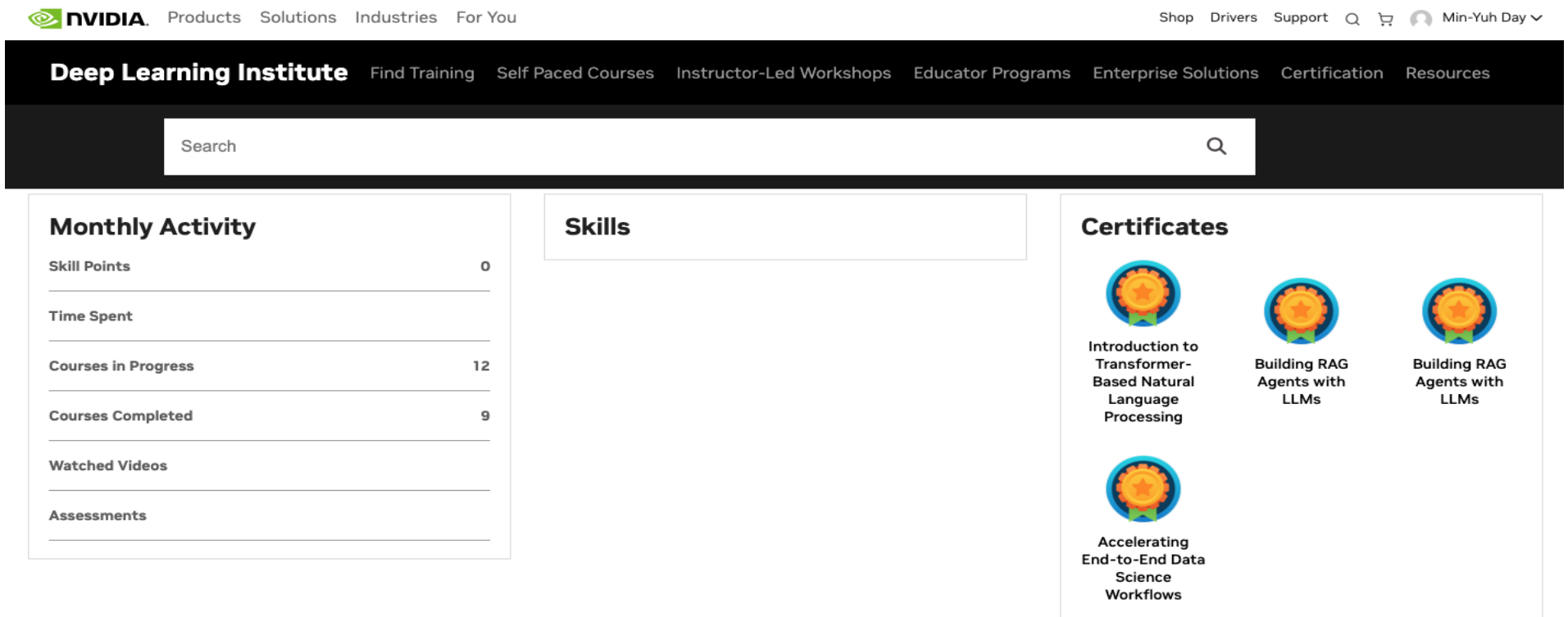
https://developer.nvidia.com/join-nvidia-developer-program

NVIDIA Deep Learning Institute (DLI)

Self-Paced Course **Generative AI Explained** Free 2 hours	Self-Paced Course **Getting Started With Deep Learning** Certificate available $90 8 hours	Instructor-Led Workshop **Fundamentals of Deep Learning** Certificate available $500 8 hours	Self-Paced Course **Introduction to Transformer-Based Natural Language Processing** Certificate available $30 6 hours
Self-Paced Course **Building RAG Agents With LLMs** Certificate available Free 8 hours	Instructor-Led Workshop **Building RAG Agents With LLMs** Certificate available $500 8 hours	Self-Paced Course **Generative AI with Diffusion Models** Certificate available $90 8 hours	Instructor-Led Workshop **Generative AI with Diffusion Models** Certificate available $500 8 hours

https://www.nvidia.com/en-us/learn/learning-path/generative-ai-llm/

Deep Learning Institute (DLI)



https://learn.nvidia.com/my-learning

NVIDIA

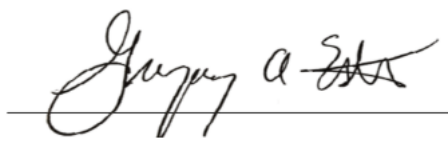
Certificate of Completion

This certificate is awarded to

Min-Yuh Day

for successfully completing

Building RAG Agents with LLMs



Greg Estes

Vice President, NVIDIA

Issue Date: : December 8, 2024

Certification ID: ed-qOCIMQaatzU8SNUNxgw |

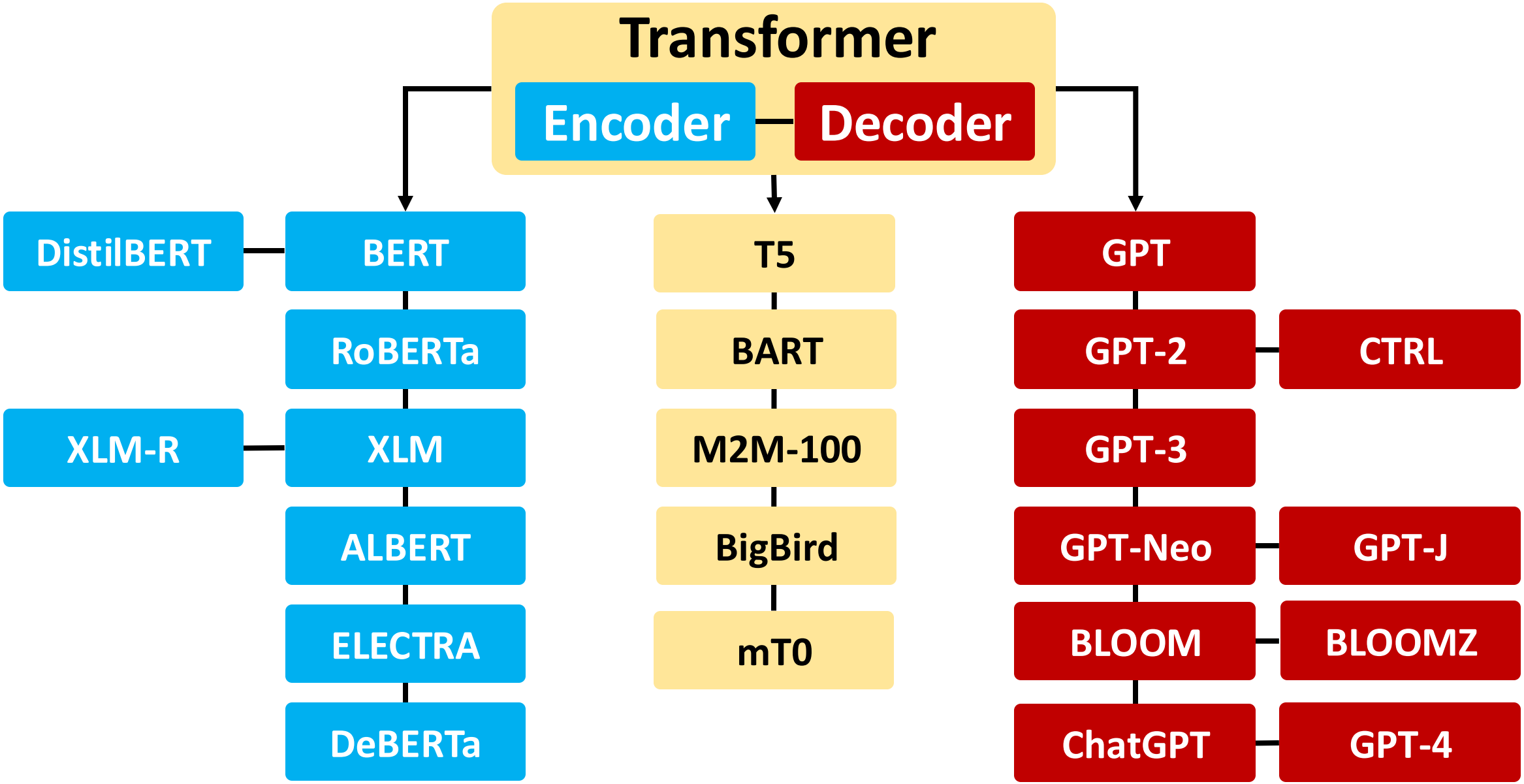
https://learn.nvidia.com/certificates?id=ed-qOCIMQaatzU8SNUNxgw/courses/course?course_id=course-v1:DLI+S-FX-15+V1

https://learn.nvidia.com/certificates?id=ed-qOCIMQaatzU8SNUNxgw

Generative AI

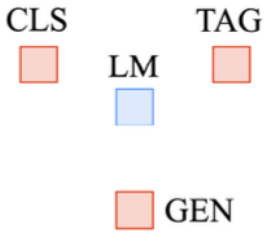
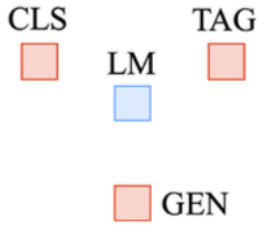
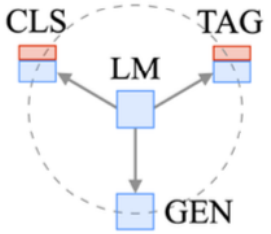
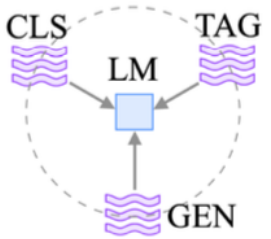
Text, Image, Video, Audio Applications

Transformer Models



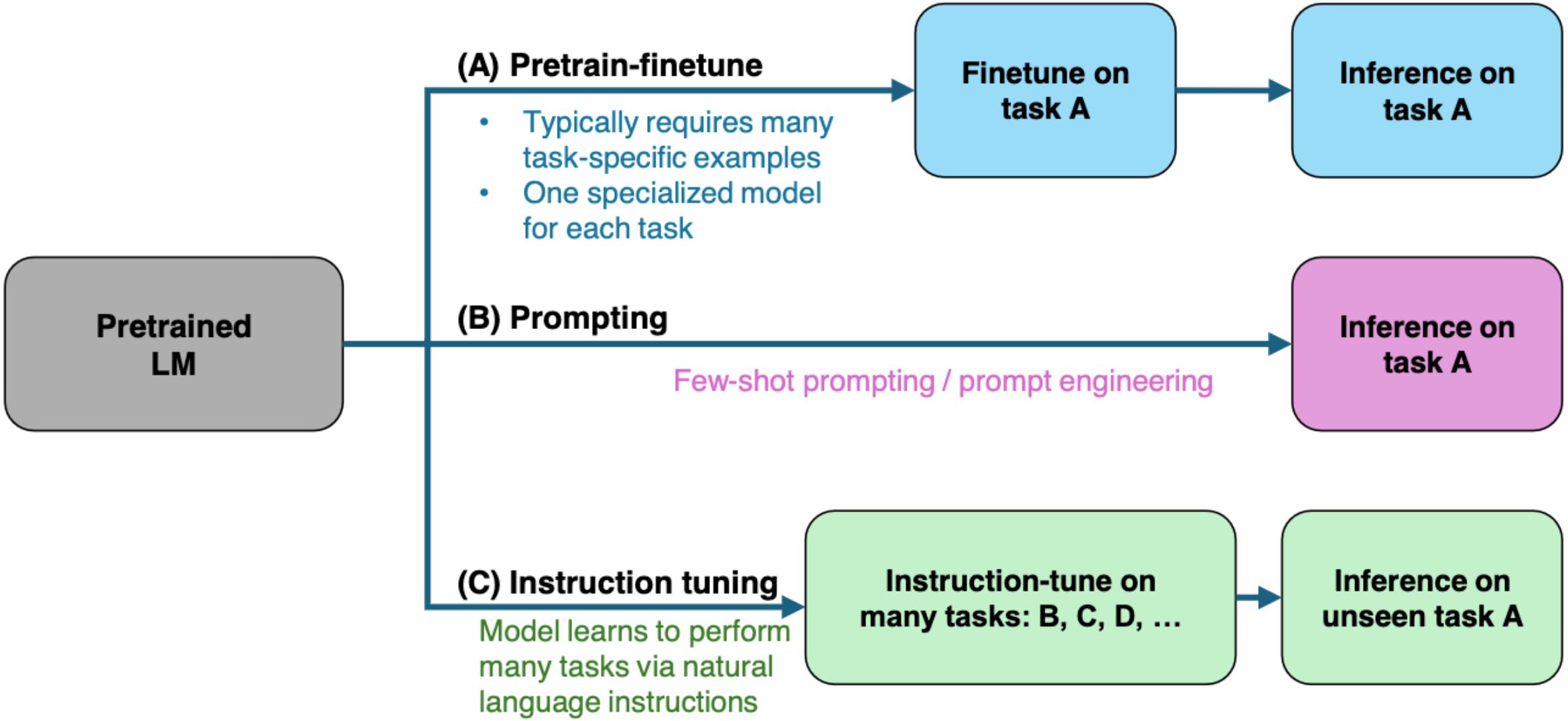
Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.

Four Paradigms in NLP (LM)

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Feature (e.g. word identity, part-of-speech, sentence length)	
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	
Transfer Learning: Pre-training, Fine-Tuning (FT) c. Pre-train, Fine-tune	Objective (e.g. masked language modeling, next sentence prediction)	
GAI: Pre-train, Prompt, and Predict (Prompting) d. Pre-train, Prompt, Predict	Prompt (e.g. cloze, prefix)	

Source: Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. (2023) "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing." ACM Computing Surveys 55, no. 9 (2023): 1-35.

Large Language Models (LLM)
Three typical learning paradigms

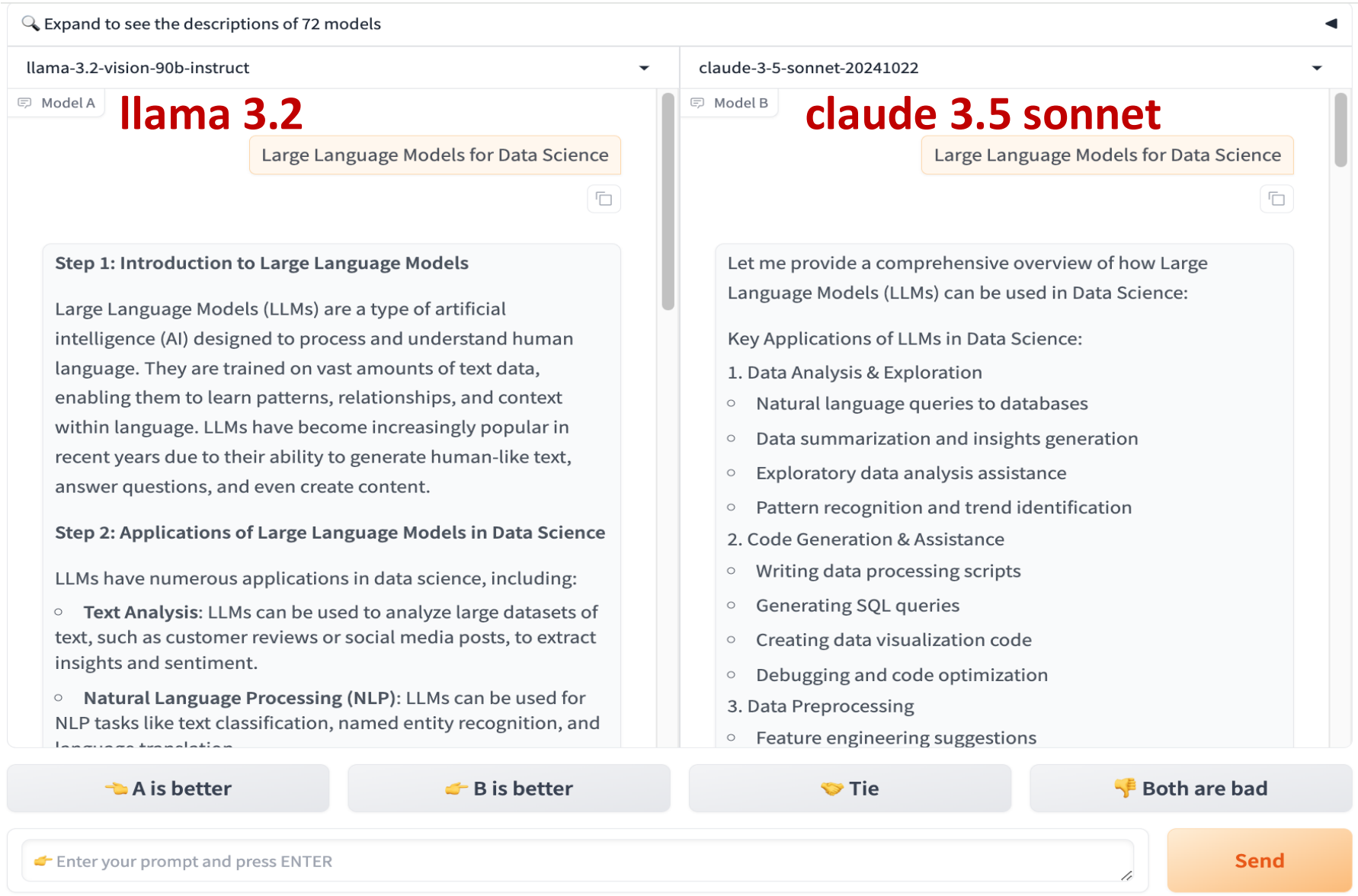


Source: Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. (2024) "A survey on multimodal large language models." National Science Review (2024): nwae403.

Popular Generative AI

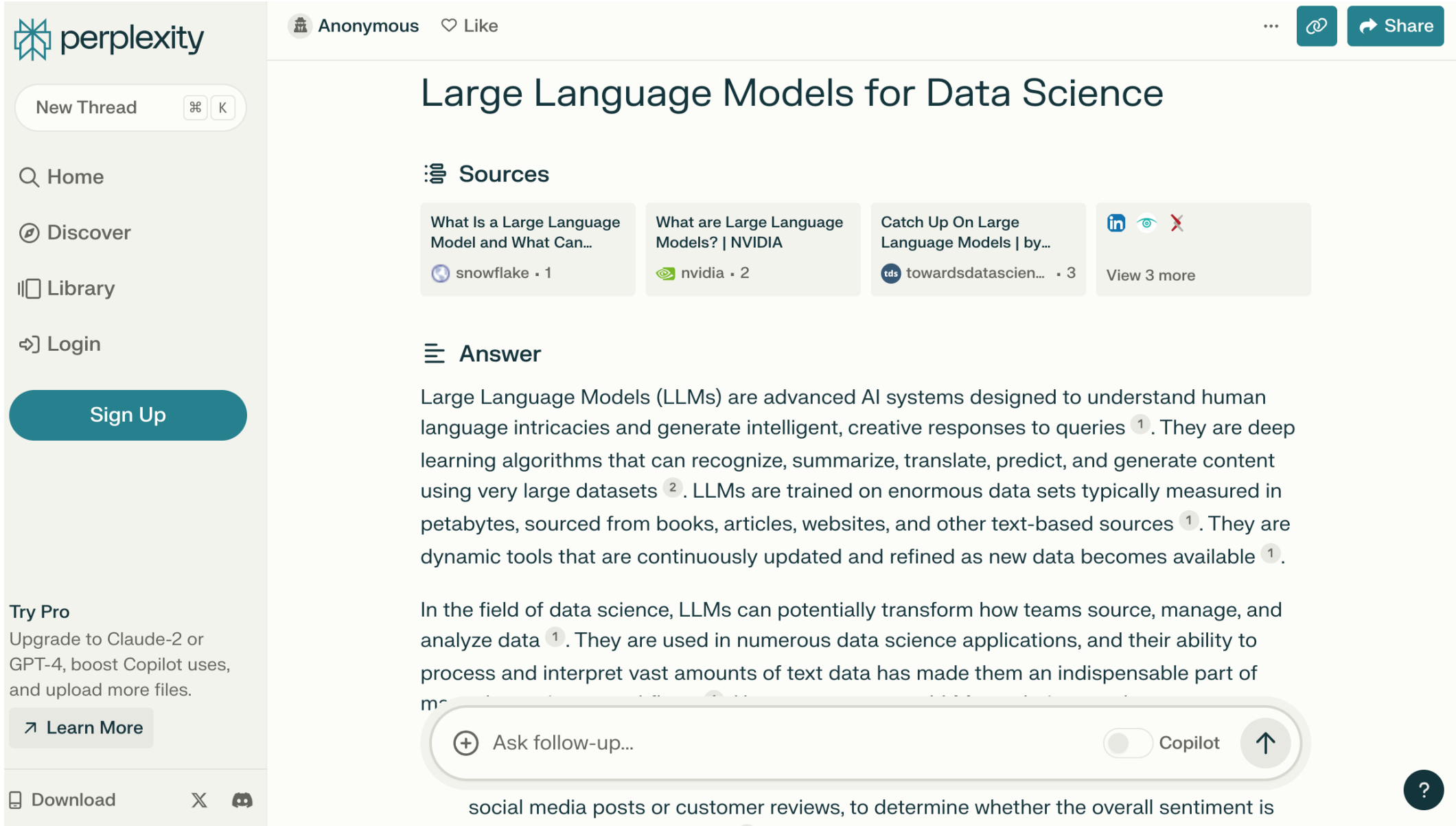
- OpenAI ChatGPT (GPT-o1, GPT-4o, GPT-4)
- Claude.ai (Claude 3.5)
- Google Gemini
- Meta Llama 3.3, Llama 3.2 Vision
- Mixtral Pixtral (mistral.ai)
- DeepSeek
- Chat.LMSys.org (lmarena.ai)
- Perplexity.ai (Perplexity Deep Research)
- Stable Diffusion
- Video: D-ID, Synthesia
- Audio: Speechify

Chat with Open Large Language Models: Chatbot Arena



https://lmarena.ai/

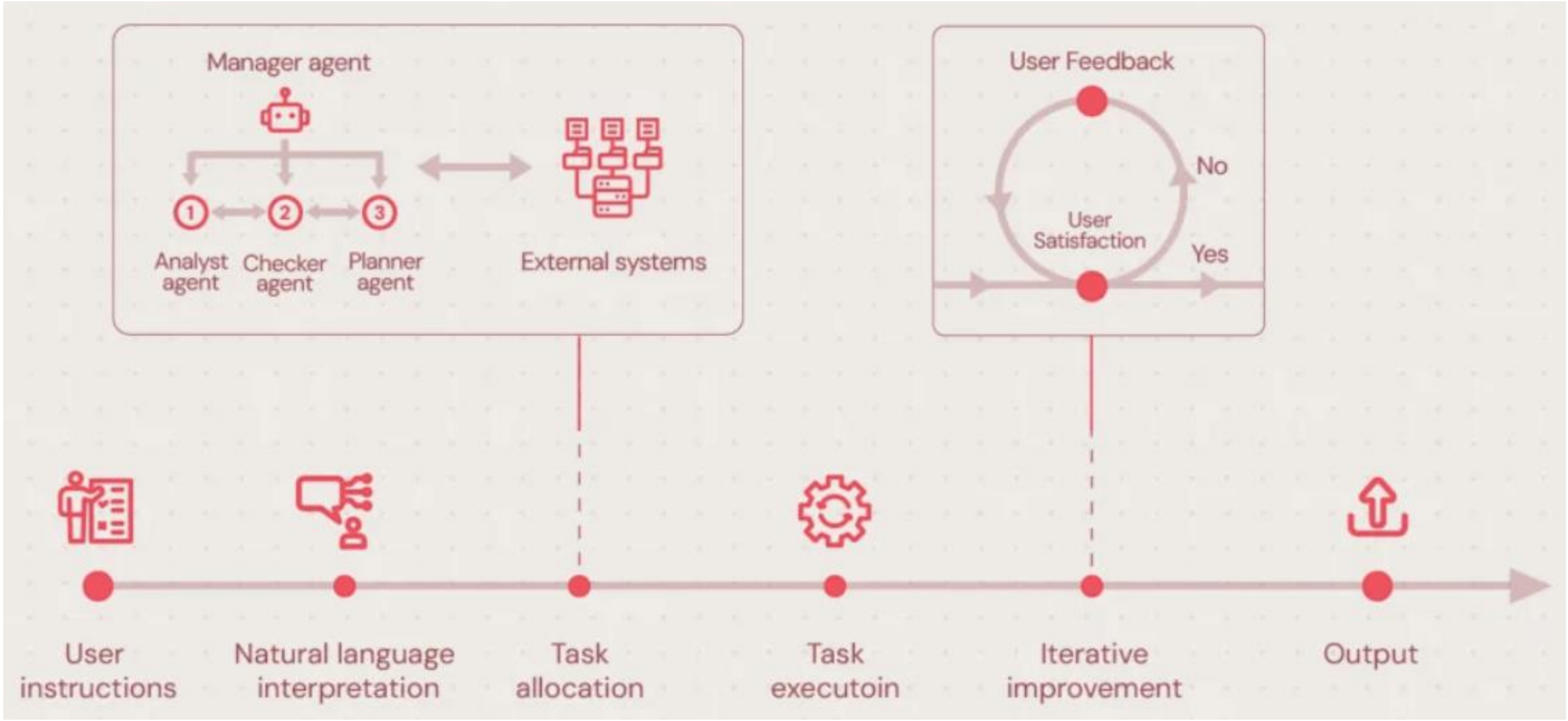
Perplexity.ai



https://www.perplexity.ai/

Agentic AI Systems
RAG Agents with LLM
Dialogue Systems

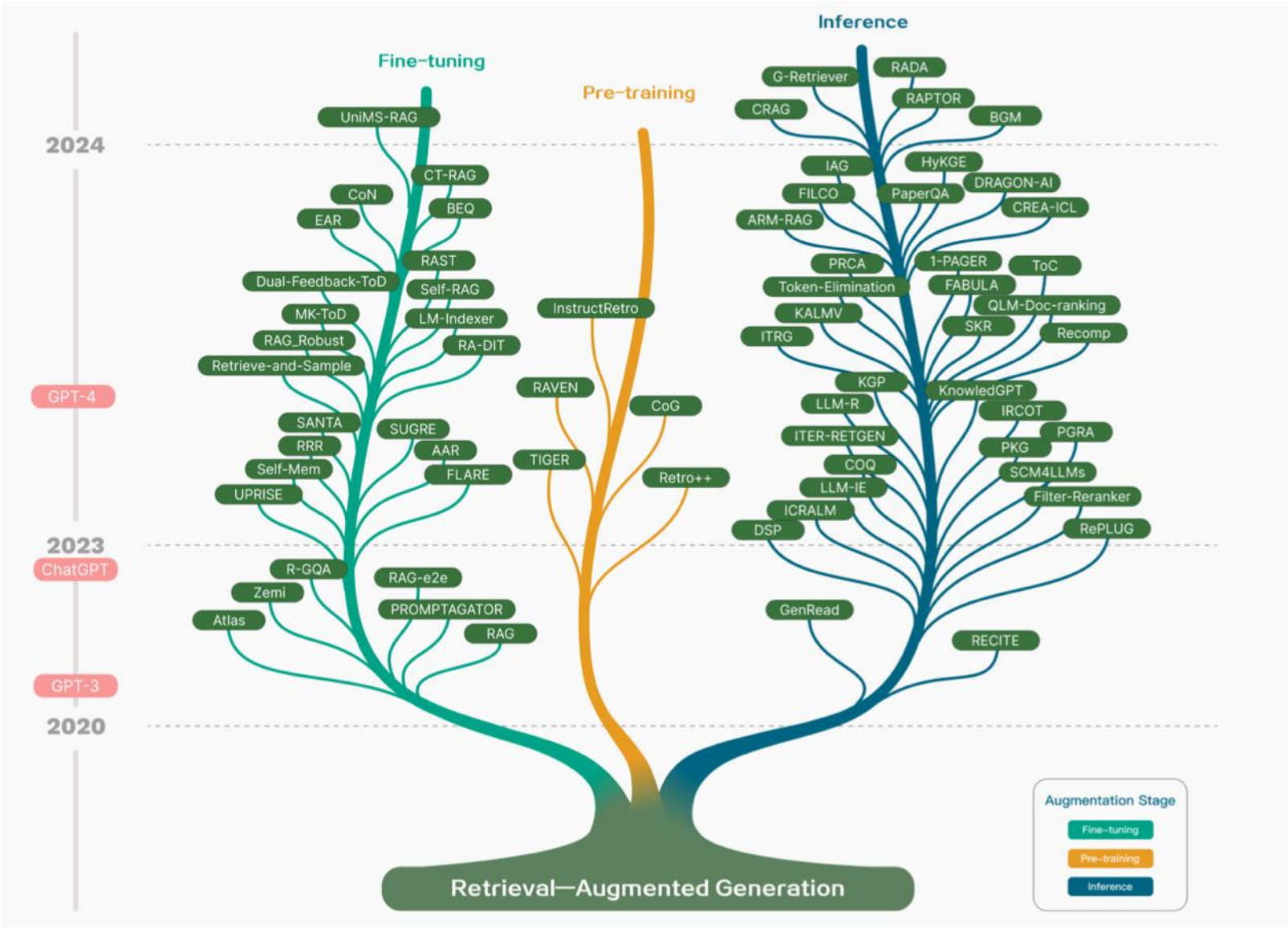
Agentic AI Systems



Source: https://neontri.com/blog/agentic-ai/

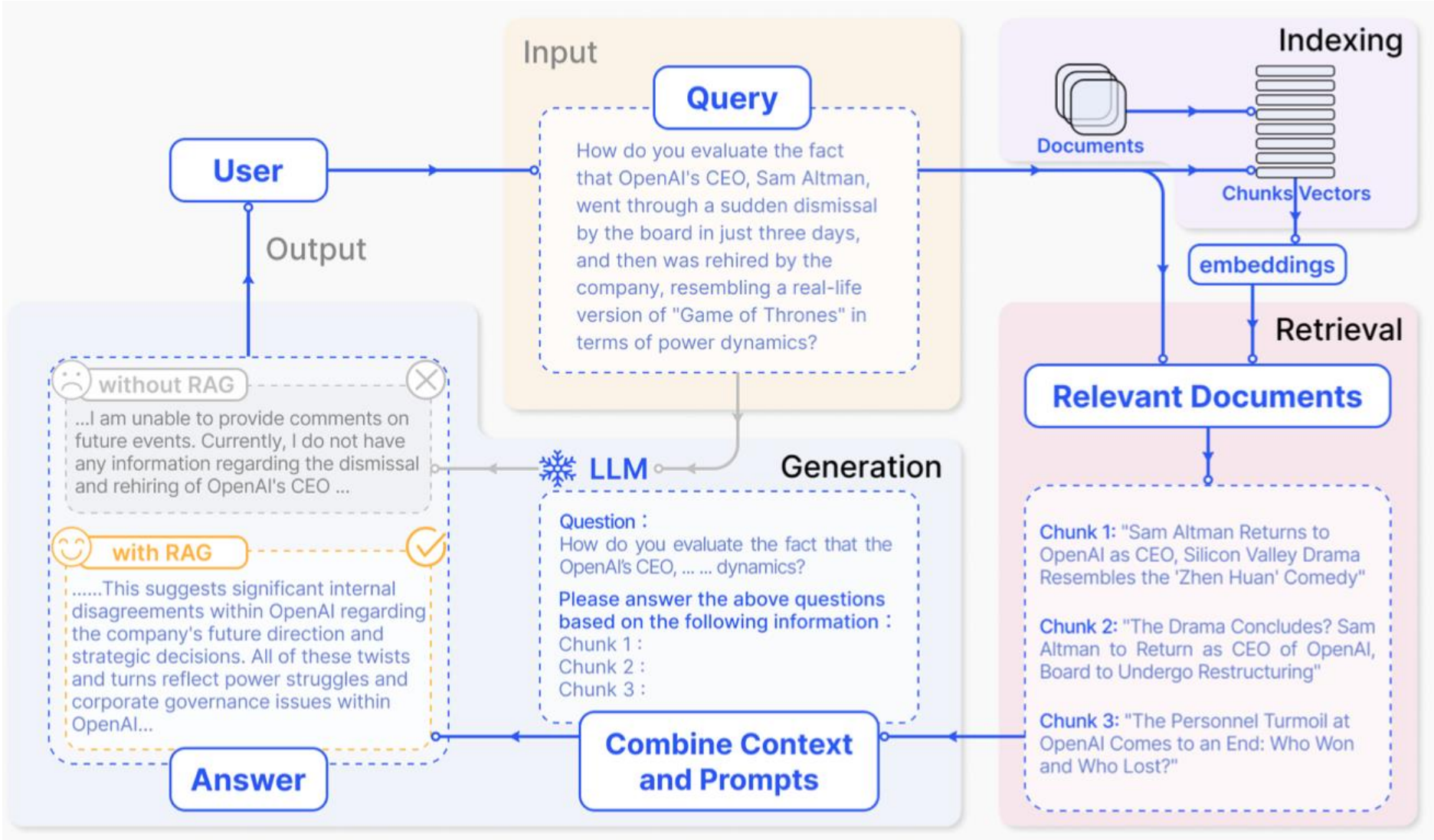
Technology Tree of RAG Research

Retrieval-Augmented Generation (RAG) for Large Language Models (LLMs)



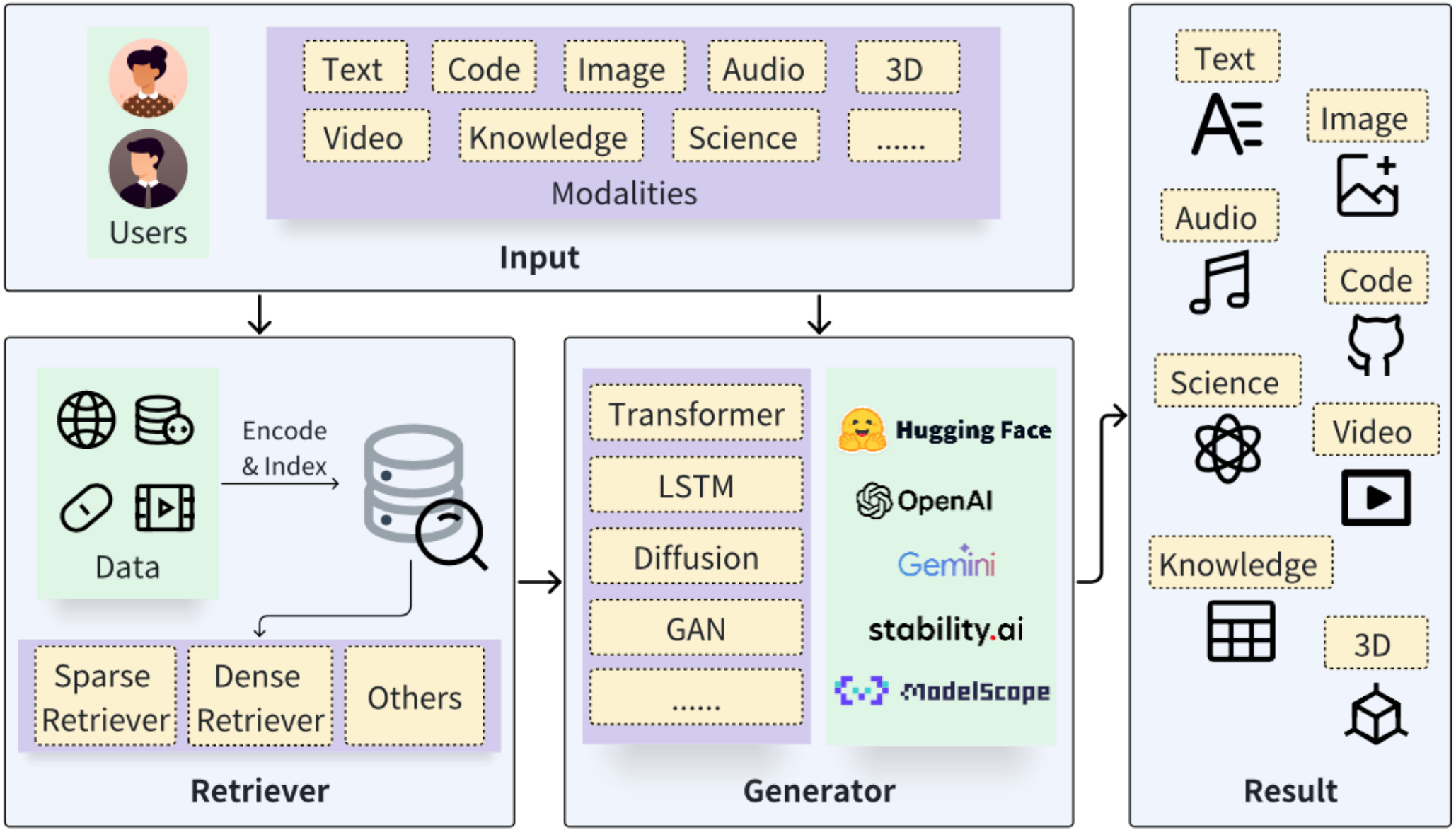
Source: Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997.

Retrieval-Augmented Generation (RAG) for Large Language Models (LLMs)



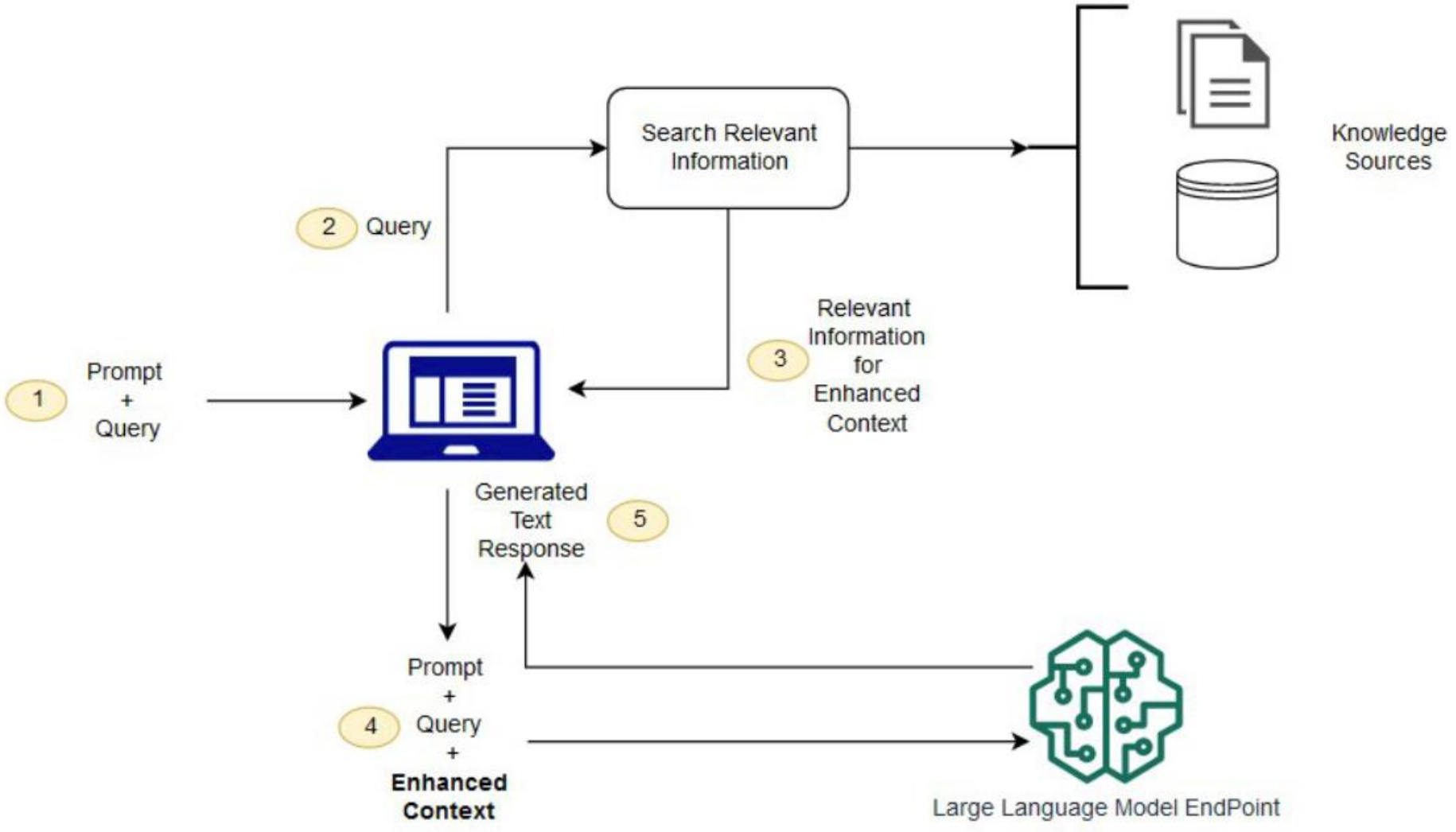
Source: Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997.

Retrieval-Augmented Generation (RAG) Architecture



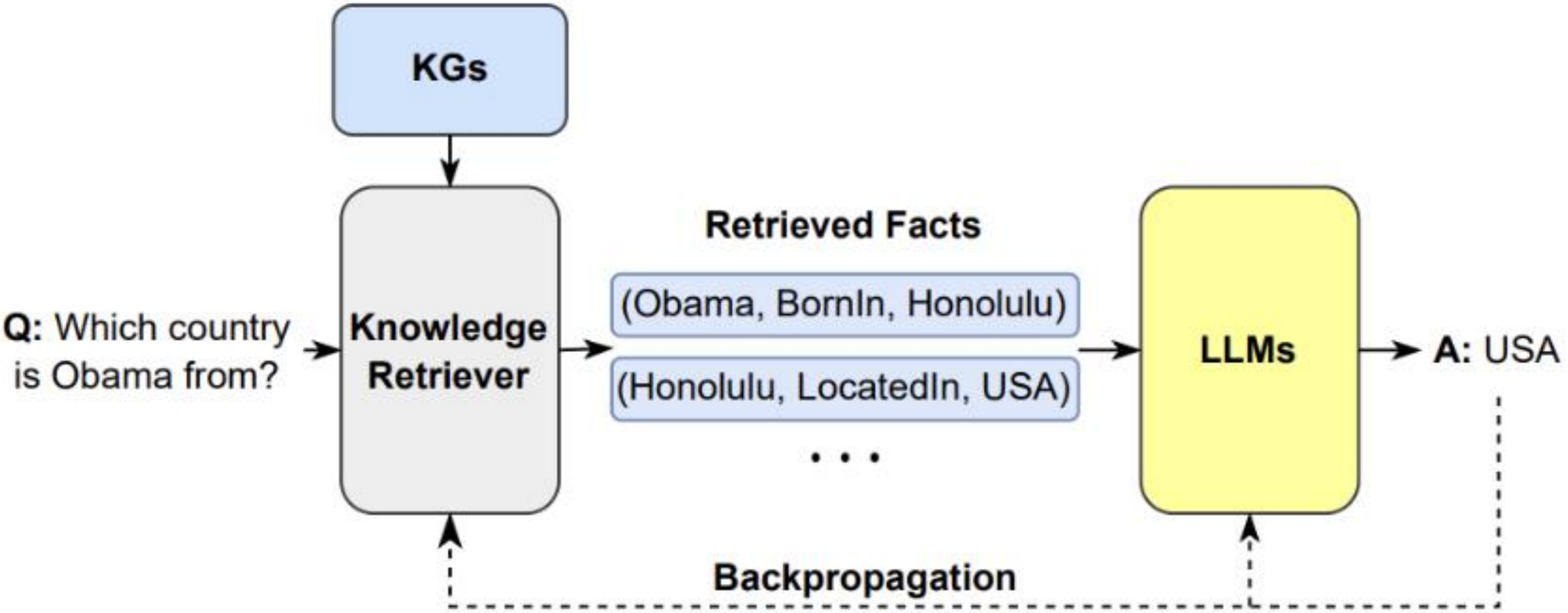
Source: Zhao, P., Zhang, H., Yu, Q., Wang, Z., Geng, Y., Fu, F., ... & Cui, B. (2024). Retrieval-augmented generation for ai-generated content: A survey. arXiv preprint arXiv:2402.19473.

Synthesizing RAG with LLMs for Question Answering Application



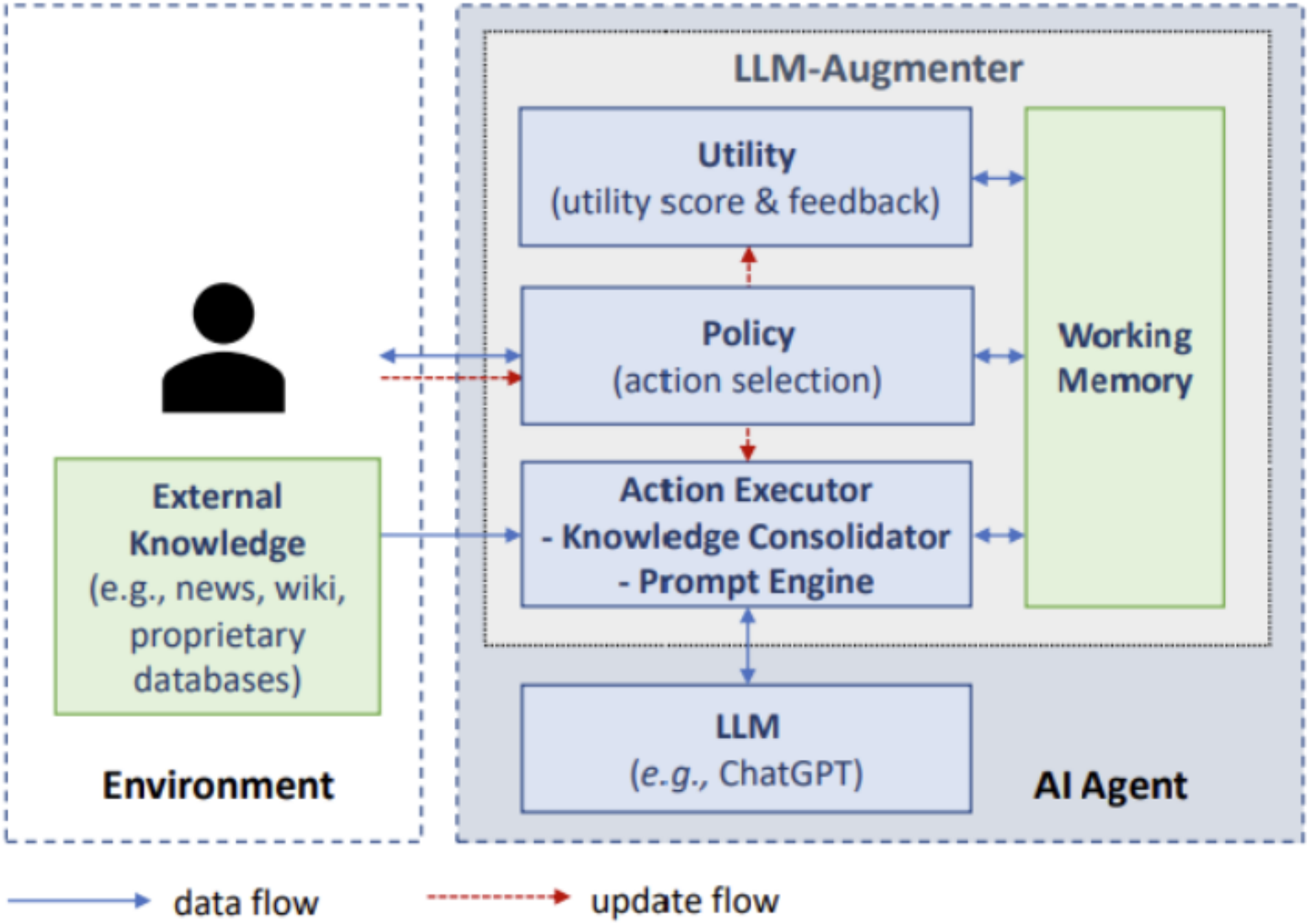
Source: Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., & Gao, J. (2024). Large language models: A survey. arXiv preprint arXiv:2402.06196.

Synthesizing the KG as a Retriever with LLMs



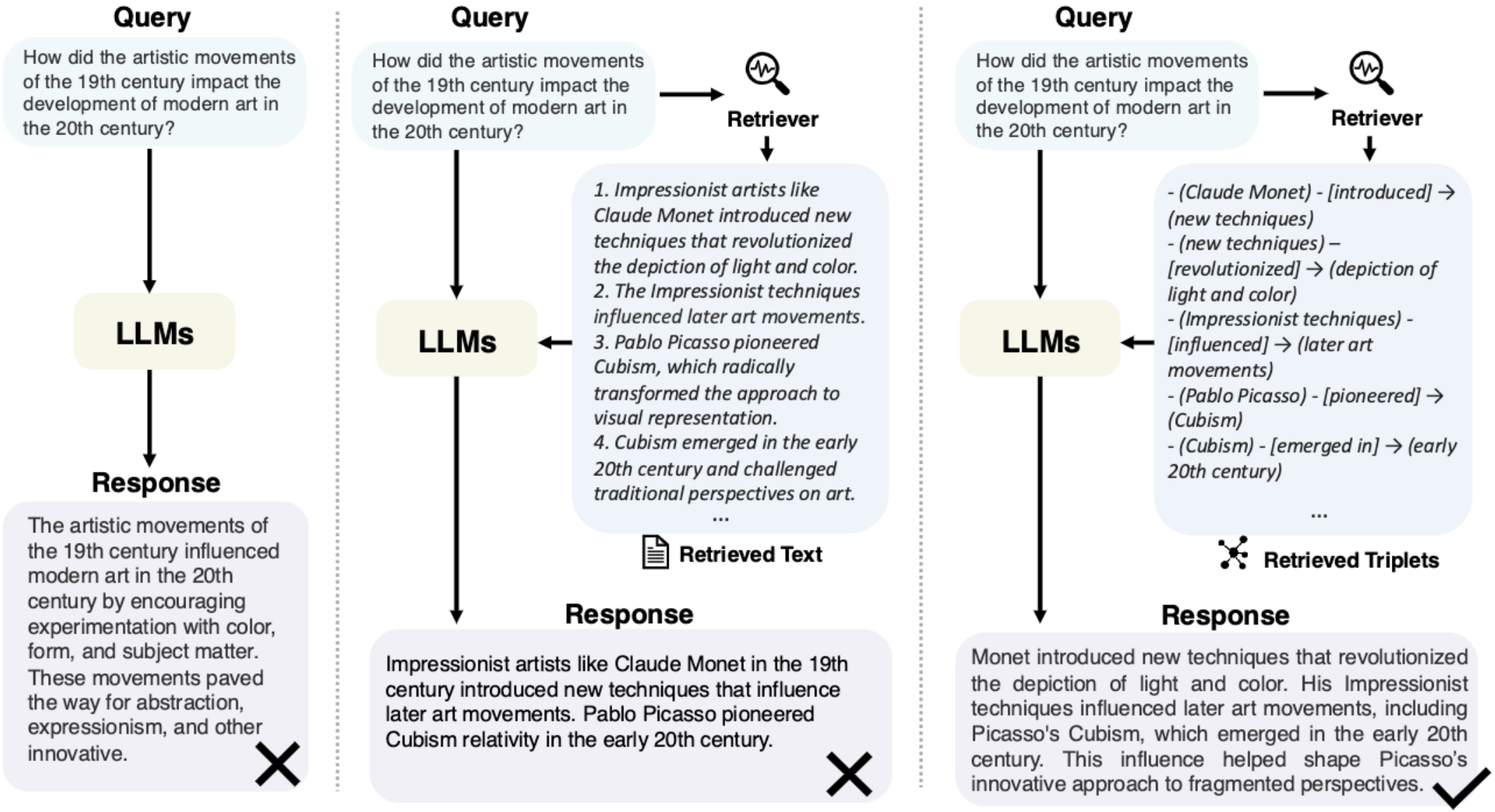
Source: Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., & Gao, J. (2024). Large language models: A survey. arXiv preprint arXiv:2402.06196.

A LLM-based Agent for Conversational Information Seeking



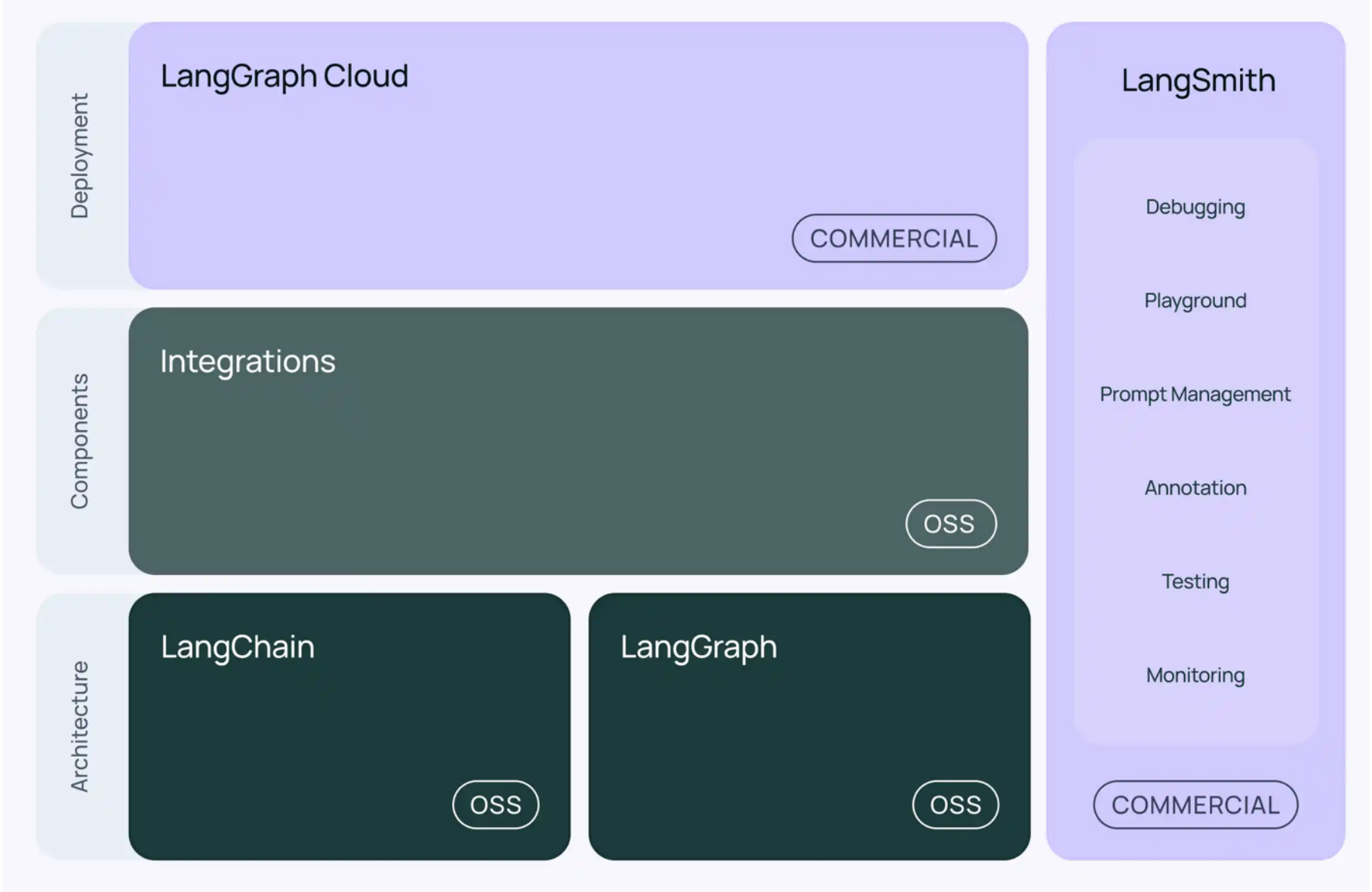
Source: Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X., & Gao, J. (2024). Large language models: A survey. arXiv preprint arXiv:2402.06196.

Direct LLM, RAG, and GraphRAG



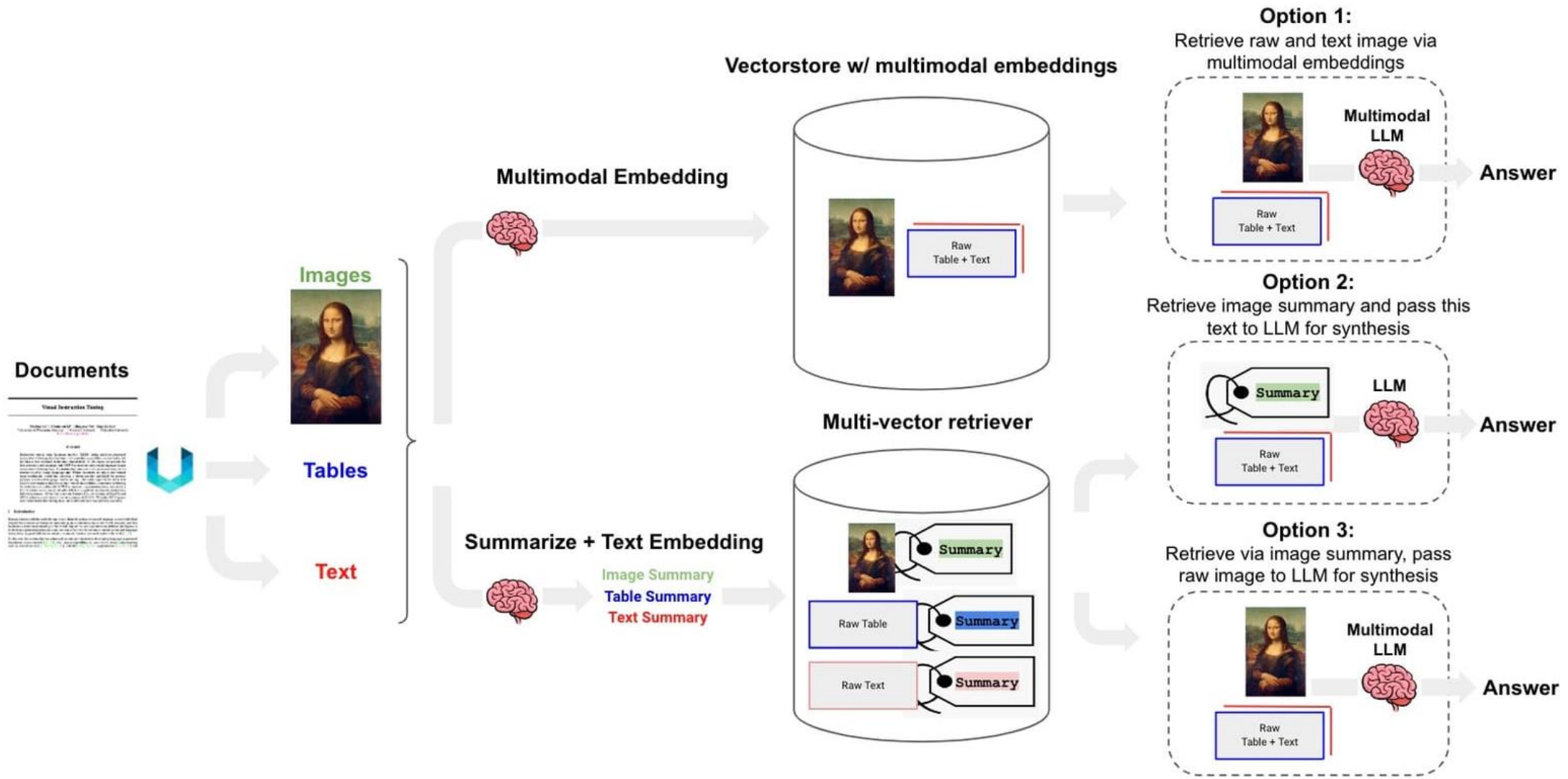
Source: Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H., Hong, C., ... & Tang, S. (2024). Graph retrieval-augmented generation: A survey. arXiv preprint arXiv:2408.08921.

LangChain Architecture



Source: https://www.langchain.com/

Multimodal LLM RAG
Multi-Vector Retriever for RAG



Source: https://blog.langchain.dev/deconstructing-rag/

Evaluating RAG with Ragas Metrics

ragas score

generation

faithfulness

how factually acurate is
the generated answer

answer relevancy

how relevant is the generated
answer to the question

retrieval

context precision

the signal to noise ratio of retrieved
context

context recall

can it retrieve all the relevant information
required to answer the question

Source: https://blog.langchain.dev/evaluating-rag-pipelines-with-ragas-langsmith/

Summary

- **Software products are software systems that include general functionality that is likely to be useful to a wide range of customers.**

- **In product software engineering, the same company is responsible for deciding on the features that should be part of the product and the implementation of these features.**

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

Summary

- **Software products may be delivered as stand-alone systems running on the customer's computers, hybrid systems or service-based systems.**
- **In hybrid systems, some features are implemented locally and others are accessed over the Internet.**
- **All product features are remotely accessed in service-based products.**

Summary

- **A product vision should succinctly describe what is to be developed, who are the target customers for the product and why they should buy the product that you are developing.**
- **Domain experience, product experience, customer experience and an experimental software prototype may all contribute to the development of the product vision.**

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

Summary

- Key responsibilities of product managers are product vision ownership, product roadmap development, creating user stories and the product backlog, customer and acceptance testing and user interface design.
- Product managers work at the interface between the business, the software development team and the product customers.
- They facilitate communications between these groups.

Summary

- **You should always develop a product prototype to refine your own ideas and to demonstrate the planned product features to potential customers**

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 (2021), A Guide to the Project Management Body of Knowledge (PMBOK Guide) – Seventh Edition and The Standard for Project Management, PMI.
- 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.
- Denis Rothman (2024), RAG-Driven Generative AI: Build custom retrieval augmented generation pipelines with LlamaIndex, Deep Lake, and Pinecone, Packt Publishing
- NVIDIA DLI (2025), Building RAG Agents with LLMs, https://learn.nvidia.com/courses/course-detail?course_id=course-v1:DLI+S-FX-15+V1
- NVIDIA DLI (2025), Generative AI with Diffusion Models, https://learn.nvidia.com/courses/course-detail?course_id=course-v1:DLI+S-FX-14+V1
- Tucker J. Marion, Mahdi Srour, and Frank Piller (2024), "When Generative AI meets product development." MIT Sloan Management Review 66, no. 1 : 14-15.