Architectural Views

Software Architecture

Richard Thomas

February 24, 2025

Interesting Software is Complex

Many aspects to the design of its architecture.

Architectural Design

Managing technical complexity.

Question

How do you describe a complex architecture, without making it too difficult to understand?

Question

How do you describe a complex architecture, without making it too difficult to understand?

Answer

Architectural Views

– Only consider one aspect at a time.

Architectural Views

- C4 Model *[Brown, 2023]*
  - context, structure, behaviour, infrastructure

Architectural Views

- C4 Model *[Brown, 2023]*
  - context, structure, behaviour, infrastructure
- 4+1 Views *[Kruchten, 1995]*
  - logical, process, development, physical, scenario

Architectural Views

- C4 Model *[Brown, 2023]*
  - context, structure, behaviour, infrastructure
- 4+1 Views *[Kruchten, 1995]*
  - logical, process, development, physical, scenario
- Software Architecture in Practice *[Bass et al., 2021]*
  - module, component-and-connector, allocation

Architectural Views

- C4 Model [Brown, 2023]
  - context, structure, behaviour, infrastructure
- 4+1 Views [Kruchten, 1995]
  - logical, process, development, physical, scenario
- Software Architecture in Practice [Bass et al., 2021]
  - module, component-and-connector, allocation
- NATO Architecture Framework [Team, 2020]
  - concepts, service, logical, physical resource, architecture foundation

Architectural Views

- C4 Model [Brown, 2023]
  - context, structure, behaviour, infrastructure
- 4+1 Views [Kruchten, 1995]
  - logical, process, development, physical, scenario
- Software Architecture in Practice [Bass et al., 2021]
  - module, component-and-connector, allocation
- NATO Architecture Framework [Team, 2020]
  - concepts, service, logical, physical resource, architecture foundation
- The Open Group Architecture Framework (TOGAF) [Forum, 2022]
- ISO/IEC/IEEE 42010:2011 [iso, 2022]

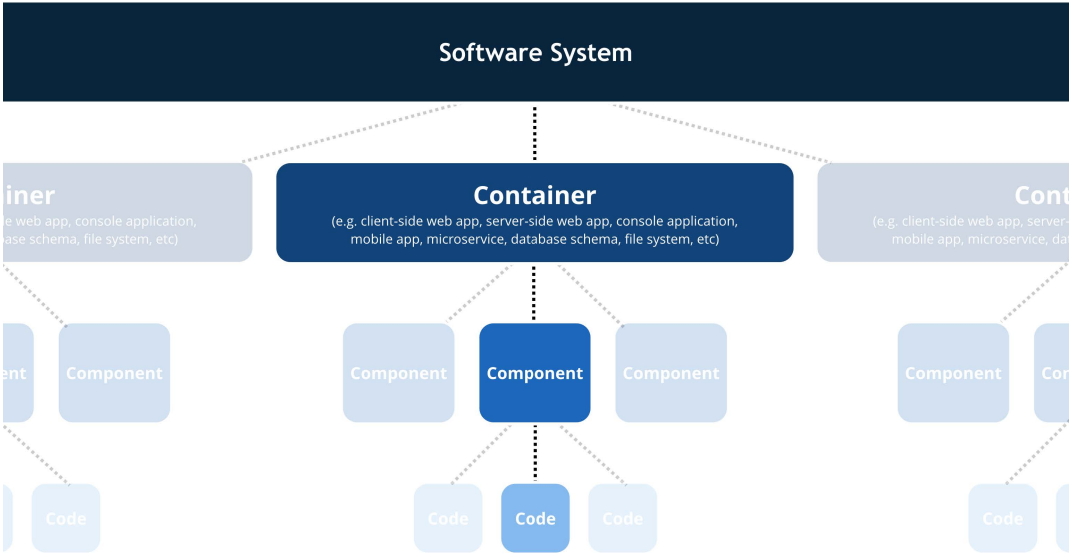
Diagrams & Notation

- A *good* diagram is worth a thousand words
  - A thousand diagrams is just confusing

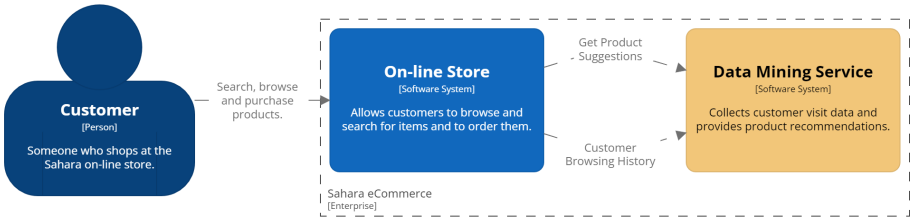
Diagrams & Notation

- A *good* diagram is worth a thousand words
  - A thousand diagrams is just confusing
- C4 – informal, simple structure *[Brown, 2023]*
- UML – formal, well-defined language *[uml, 2017]*
- You probably don't want to know about alternatives

C4 Model: Levels

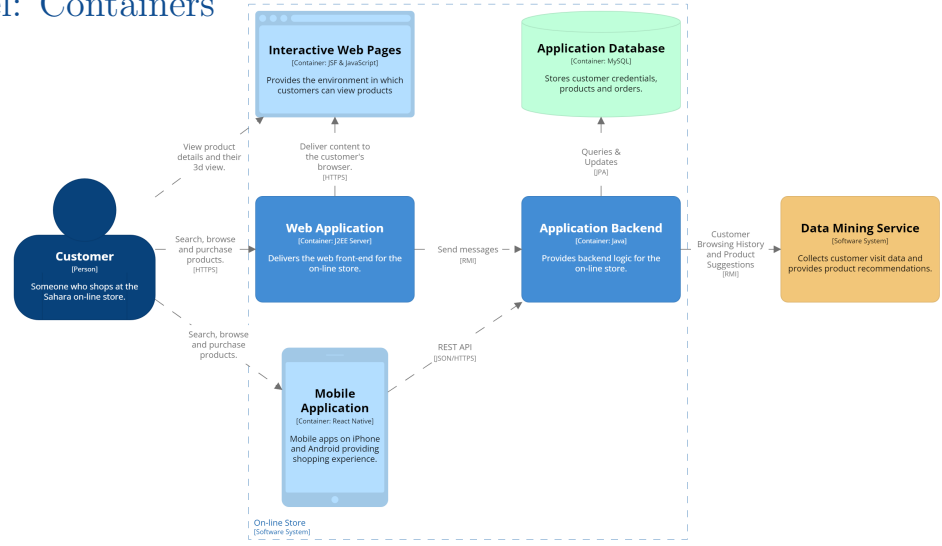


C4 Model: Context



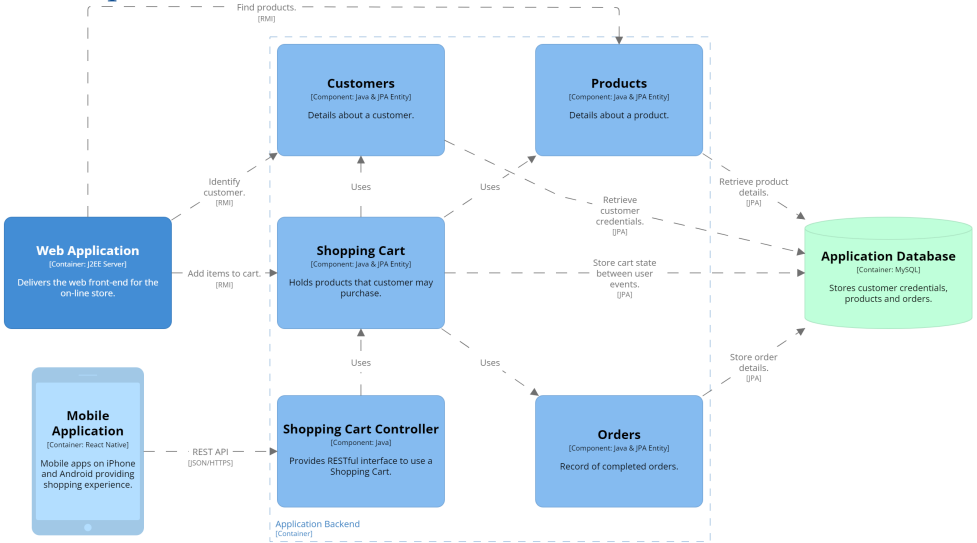
How software system fits into broader *environment*

C4 Model: Containers



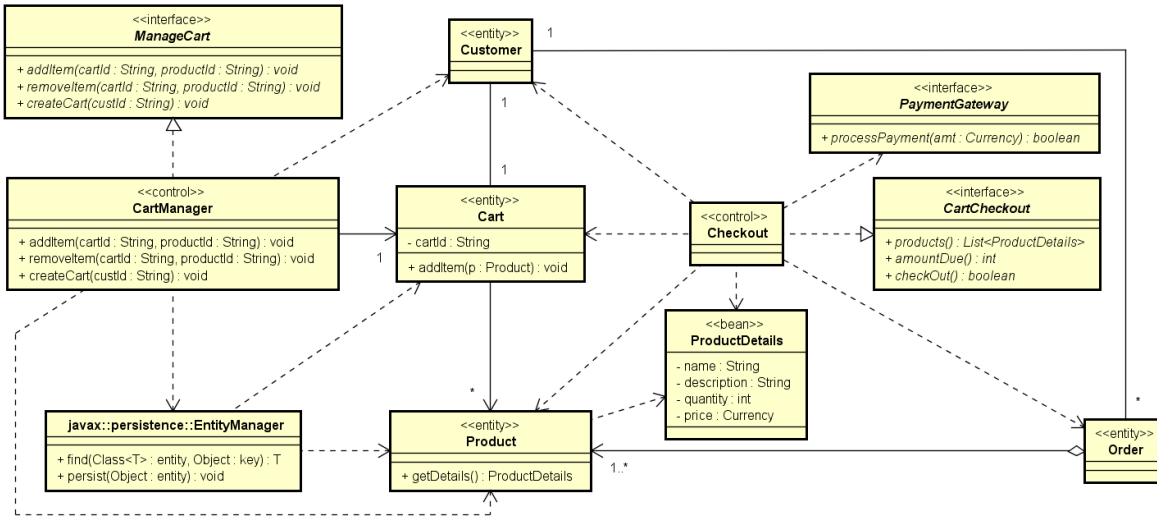
Structure of the software system

C4 Model: Components



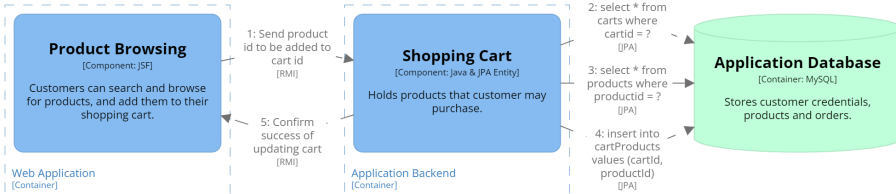
Elements that implement a container

C4 Model: Code



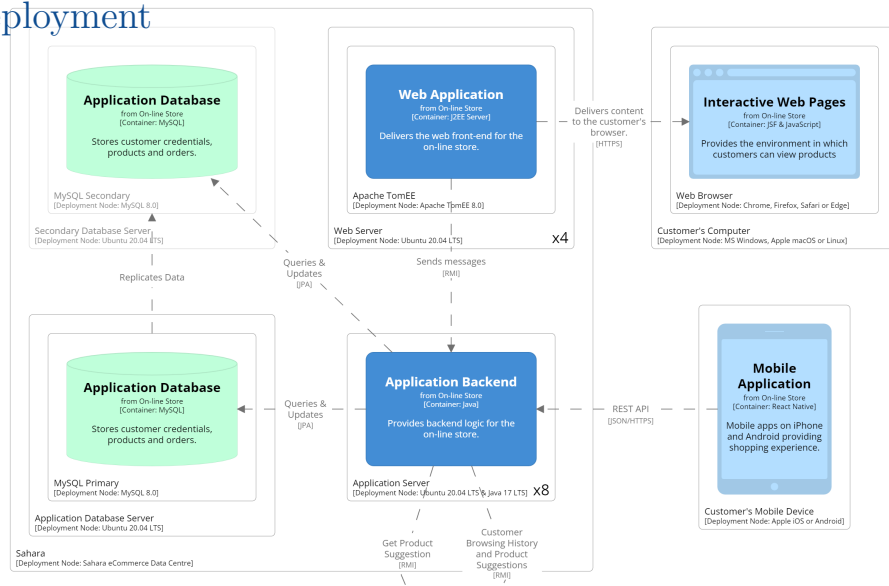
Structure of code implementing a component

C4 Model: Dynamic



How parts of the model *collaborate* to deliver *behaviour*

C4 Model: Deployment



Infrastructure on which system will be deployed

4+1 Views

Logical – *Structure* of how the software is implemented

- components/classes, relationships, interactions

4+1 Views

Logical – *Structure* of how the software is implemented

- components/classes, relationships, interactions

Process – *Dynamic* behaviour

- concurrency & distribution, fault tolerance, process control, ...

4+1 Views

Logical – *Structure* of how the software is implemented

- components/classes, relationships, interactions

Process – *Dynamic* behaviour

- concurrency & distribution, fault tolerance, process control, ...

Development – *Organisation* of the software in the development environment

4+1 Views

Logical – *Structure* of how the software is implemented

- components/classes, relationships, interactions

Process – *Dynamic* behaviour

- concurrency & distribution, fault tolerance, process control, ...

Development – *Organisation* of the software in the development environment

Physical – *Map* executable software containers to hardware

- address non-functional requirements
  - availability, reliability, scalability, throughput, ...

4+1 Views

Logical – *Structure* of how the software is implemented

- components/classes, relationships, interactions

Process – *Dynamic* behaviour

- concurrency & distribution, fault tolerance, process control, ...

Development – *Organisation* of the software in the development environment

Physical – *Map* executable software containers to hardware

- address non-functional requirements
  - availability, reliability, scalability, throughput, ...

Scenario – *Demonstrate* functionality delivered by architecture

- use case details
  - *drive* functional design of architecture
  - *validate* design of architecture
  - *illustrate* purpose of architecture

Reading...

"Architectural Views" Notes[1] *[Thomas and Webb, 2023]*

[1] Remember, I said you had to read the notes.

References

[uml, 2017] (2017).
Unified Modeling Language.
OMG, 2.5.1 edition.
https://www.uml.org/.

[iso, 2022] (2022).
Software, Systems and Enterprise – Architecture Description (ISO/IEC/IEEE 42010:2022).
International Organization for Standardization.

[Bass et al., 2021] Bass, L., Clements, P., and Kazman, R. (2021).
Software Architecture in Practice.
Addison-Wesley, 4th edition.

[Brown, 2023] Brown, S. (2023).
The C4 Model for Visualising Software Architecture.
Leanpub.
https://leanpub.com/visualising-software-architecture.

[Forum, 2022] Forum, T. O. G. A. (2022).
The Open Group Architecture Framework Standard – Architecture Development Method.
The Open Group, 10 edition.
https://pubs.opengroup.org/togaf-standard/.

[Kruchten, 1995] Kruchten, P. (1995).
Architectural blueprints — the '4+1' view model of software architecture.
IEEE Software, 12(6):42–50.
https://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf.

[Team, 2020] Team, A. C. (2020).
NATO Architecture Framework.
NATO, 4th edition.

[Thomas and Webb, 2023] Thomas, R. and Webb, B. (2023).
Architectural views.
`https://csse6400.uqcloud.net/handouts/views.pdf`.