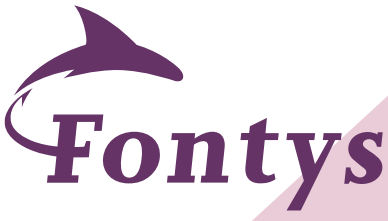
C4 model

How to Create a C4 Model for Software Architecture

Tomasz Olejarczuk



14-12-2023

Contents

C4 model **2**

How to Create a C4 Model for Software Architecture **2**

1. Context Diagram (Level 1: System Context): 2
2. Container Diagram (Level 2: Container): 2
3. Component Diagram (Level 3: Component): 2
4. Code (Level 4: Code): 2
5. Annotations and Documentation: 3
6. Tooling: 3
7. Review and Update: 3
8. Sources 3

C4 model

The C4 model is a framework for visualizing the architecture of a software system. It provides a set of diagrams at different levels of abstraction, allowing you to communicate effectively with various stakeholders. C4 stands for “Context, Containers, Components, and Code,” representing four levels of abstraction.

How to Create a C4 Model for Software Architecture

1. Context Diagram (Level 1: System Context):

- Identify the system’s external actors (users, systems, or external services).
- Create a simple diagram showing the system boundary and external actors.
- Use boxes to represent the system and external actors, and lines to show the interactions between them.

2. Container Diagram (Level 2: Container):

- Identify the major containers within the system (e.g., web servers, databases, mobile apps).
- For each container, define its responsibilities and dependencies.
- Create a diagram with boxes representing containers and lines indicating the communication between them.
- Optionally, include key technologies and data stores associated with each container.

3. Component Diagram (Level 3: Component):

- For each container, break it down into its major components (e.g., classes, modules, services).
- Describe the responsibilities and interfaces of each component.
- Create a diagram with boxes representing components and lines indicating dependencies or communication between them.

4. Code (Level 4: Code):

- Optionally, you can dive deeper into the code level if necessary.
- This level involves detailed class diagrams, sequence diagrams, or other relevant diagrams to represent specific aspects of the code.
- Focus on key components or critical code areas.

5. Annotations and Documentation:

- Use annotations to provide additional information on each diagram.
- Include relevant details such as technology choices, deployment environments, and constraints.
- Provide documentation to explain the decisions made in the architecture.

6. Tooling:

- Use tools that support the creation of C4 diagrams, or you can use general-purpose diagramming tools like draw.io, Lucidchart, or Visio.
- There are dedicated tools such as Structurizr that are specifically designed for creating and maintaining C4 models.

7. Review and Update:

- Regularly review and update the C4 diagrams as the system evolves.
- Use the diagrams as a living documentation that reflects the current state of the architecture.

8. Sources

- C4 Model
- https://structurizr.com/
- https://github.com/cecilphillip-stripe/stripe-checkout-events-sample/tree/dapr