7. Software Architecture

How to architect?

The definition of software architecture is difficult. But to try and sum it up, one can say:

- "Expert developers' shared understanding of the system design“
- "Architecture is the decisions that you *wish* you could get right early in a project.”

Architectural styles

Pros / Cons

	Layers	Pipes and filters	Message bus	N-tier
Decoupling	+	+	+	+
Structure	+		-	-
Scalability		-	+	+
Complexity		-	-	-

	Layers	Pipes and filters	Message bus	N-tier
Decoupling	Seperated layers. Layers can be replaced.	SRP OCP Parallelable	Replacement and OCP Parallelable	Seperated tiers update on tier Parallelable

	Layers	Pipes and filters	Message bus	N-tier
Structure	Hierarchial Higher levels depend on lower		Rigid: Hard-defined message structure	Communication: Hard to develop and maintain
Scalability		Performance effected by many filters and complex data	Many modules	Multiple devices
Complexity		Depend on filters = complex	Many different message routes = comples	Complex to build

Layers

> Partitions the concerns of the application into stacked groups (layers).
> Layers is a logical separation.

Pipes and filters

> Data flows and gets transformed in a pipeline.

Message bus

> Application interact via a communication channel.

N-tier

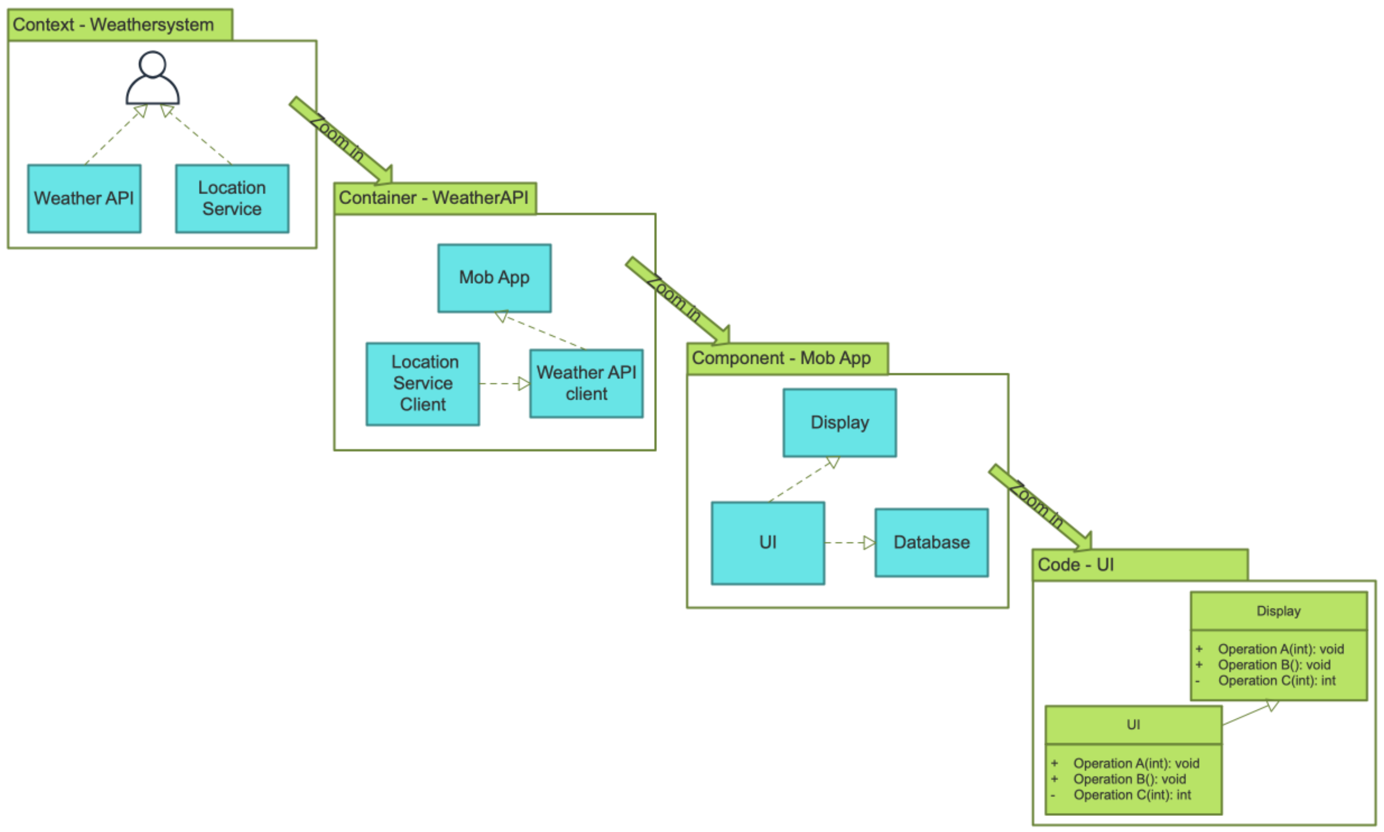
> Similar to layers, but each layer is in a seperate computer
> Tier represent a physical separation.
> Client-server (2-tier)

Documentation

There are a lot of models, I will focus on C4.

C4

- Context - External systems, actors
- Container - Major software systems, making up overall application
- Component - represent different functional units.
- Code (class diagram often)



SOLID

Also apply on architecture level, but relates to modules instead of classes.