Application Architecture

Optimizing For

— Team-Scale Autonomy
— Long Term Evolution
— Disposability

Fundamental Tools

1. Name
2. Abstract (generalize)
3. Instatiate (specialize)
4. Connect
5. Separate

Name

Name things to emphasize similarities or call out differences.

Example: P2P Lending

1. Commitment to fund part of a loan.
2. Piece of text that needs translation.

"Reservation"

Example: P2P Lending

Common behavior: Limited time claim on a thing with a quantity.

Common data? URL, expiration, quantity available to reserve.

Abstract

Emphasize one common aspect, hide others.

E.g. "parity". Emphasizes evenness/oddness. Applies to many different groups.

Instantiate

Counterexample: Many classes, one instance of each class.

Specialize via interactions

1. Instantiate "Reservations" as "Loan Funding Commitment"
2. No new behavior required, but useful for homogeneity of data.
3. Helps with analytics

Connect

	StudioServer	StudioClient	DvdLoader	StudioCommon	RenderEngine	ProductionToolbox	PcsInterface	RenderInterface	Common
StudioServer	-			Y					Y
StudioClient		-		Y					Y
DvdLoader			-				Y		Y
StudioCommon				-					Y
RenderEngine					-		Y	Y	Y
ProductionToolbox						-		Y	Y
PcsInterface							-		Y
RenderInterface								-	Y
Common									-

Adjacency Matrix

Connect

1. Initiation
2. Transport
3. Framing
4. Encoding
5. Semantics

Separate

— Split functions to liberate from their original context
— Create different teams where you want a boundary

"Reverse Conway Maneuver"

Separating Lifecycle vs. Instance Data

```
public interface Item {
    String getName(int version);
    String getDescription(int version);
    void publishItem(int version);
    int getLatestVersion();
    int[] getAllVersions();
    ...
}
```

Every consumer had to deal with the versioning & lifecycle.

Segregated Interface

```
public interface ItemVersions {
    ItemDetails getLatestVersion();
    ItemDetails[] getAllVersions();
    void publishItem(ItemDetails details);
}

public interface ItemDetails {
    String getName();
    String getDescription();
    ...
}
```

Separating Lifecycle vs. Instance Data

- JSON API
  - Including status flags, effective date, etc.
- Most apps aren't involved in versioning and editing

Dimensions to Work With

1. Library
2. Executable
3. Process
4. Host
5. Service
6. Geography