Anti-Patterns and Common Mistakes

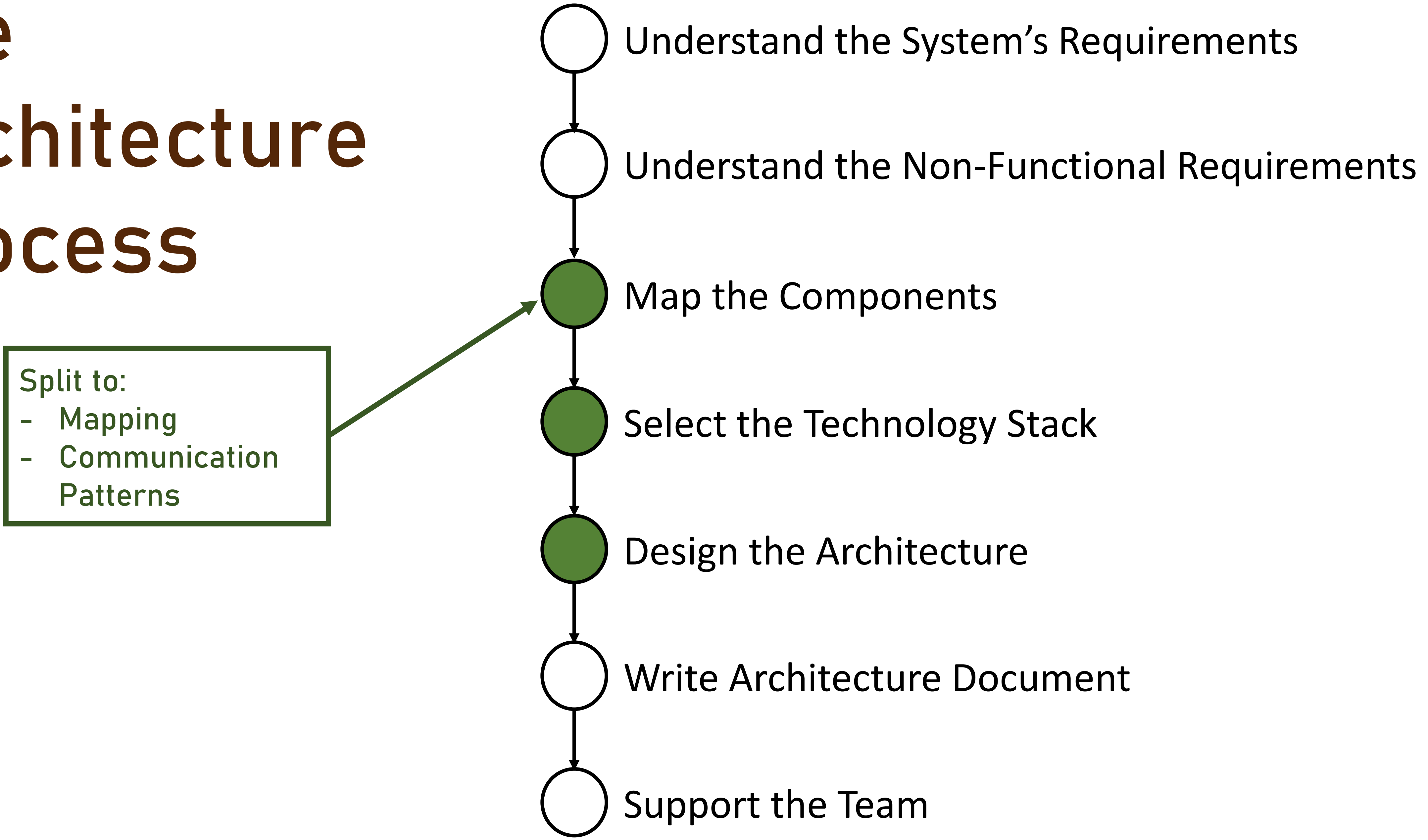
Memi Lavi
www.memilavi.com

Anti-Patterns and Common Mistakes

- Microservices require thorough design
- They are not “Fire and Forget”
- It’s easy to make mistakes that will cause the project to fail

No Well-Defined Services

The Architecture Process


Understand the System's Requirements
Understand the Non-Functional Requirements
Map the Components
Select the Technology Stack
Design the Architecture
Write Architecture Document
Support the Team

Split to:
- Mapping
- Communication Patterns

Mapping the Components

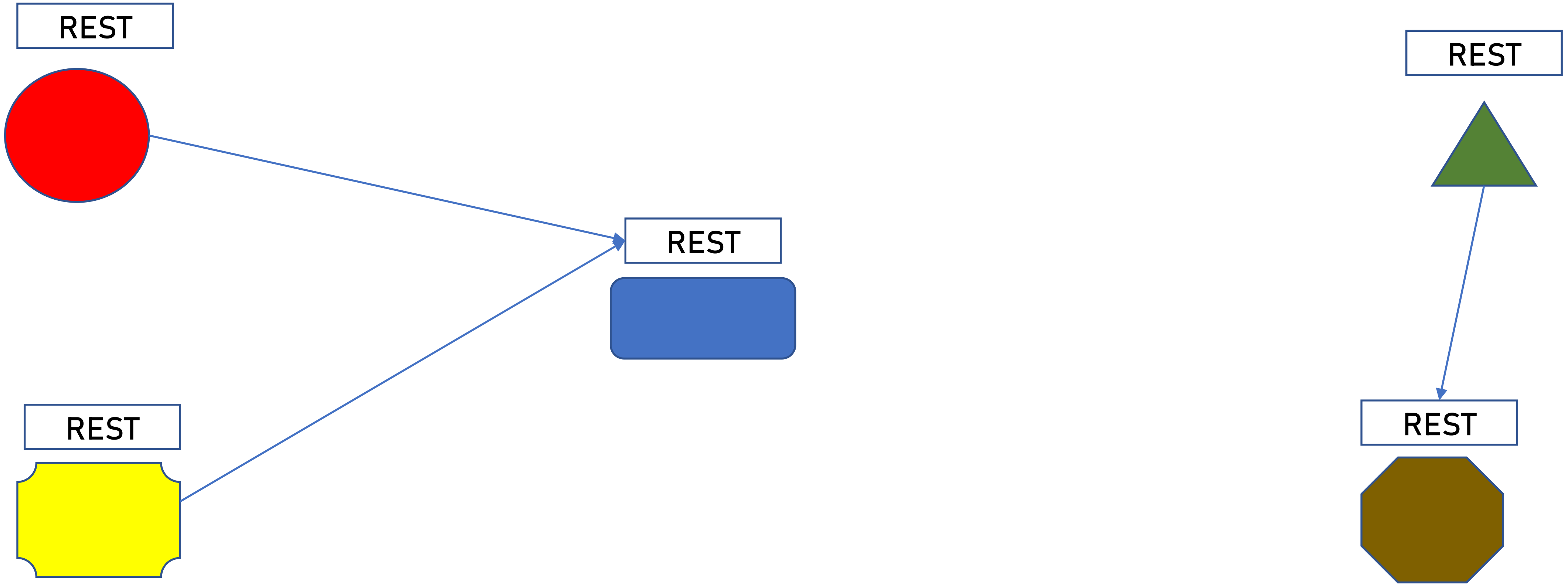
- The single most important step in the whole process
- Determines how the system will look like in the long run
- Once set – not easy to change

No Well-Defined Services

- Negligent mapping results in bloated services
- Dependent functionality gets added continuously…
- …and creates a mini-monolith

No Well-Defined API

- API is the door to the service



No Well-Defined API

- Should be well thought of and easy to learn
- MUST be consistent
- MUST be versioned
- MUST be platform agnostic
- MUST be part of the design

API Design Example

Functionality	Path	Return Codes

Get next list to be processed	`GET /api/v1/lists/next?location=…`	`200 OK`
		`400 Bad Request`

Mark item as collected / unavailable	`PUT /api/v1/list/`*`{listId}`*`/item/`*`{itemId}`*	`200 OK`
		`404 Not Found`

Export list's payment data	`POST /api/v1/list/`*`{listId}`*`/export`	`200 Ok`
		`404 Not Found`

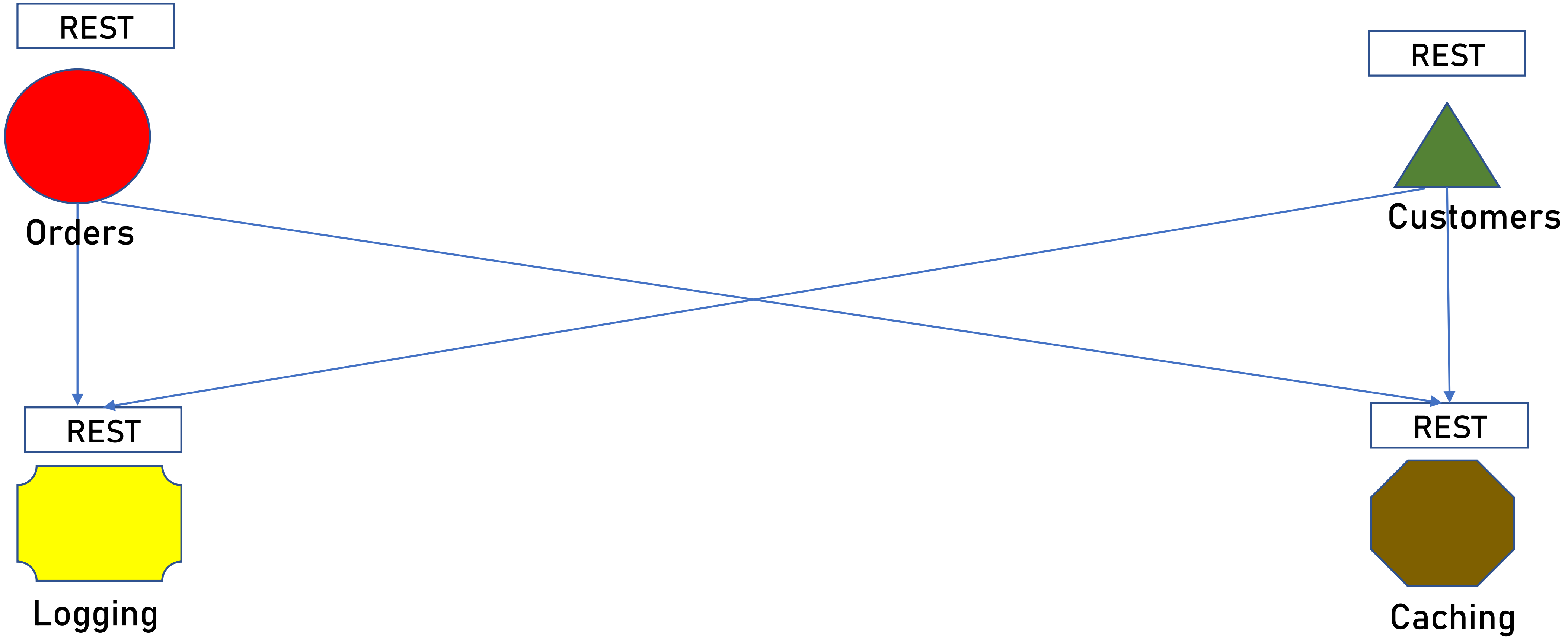
Implementing Cross-Cutting Last

- Every system has cross-cutting (system-wide) services
  - Logging
  - Caching
  - Users Management
  - Authz & Authn
  - And more…

Implementing Cross-Cutting Last

- Should be implemented first
- Other services are going to use them
- No one likes to go back and modify existing code

Cross Cutting Concerns



Expanding Service Boundaries

- Every service has well-defined boundaries
- Expanding these boundaries makes the service inefficient and bloated
- It's tempting – don't to that!
- Many times new service should be used instead of expanding existing service's boundaries