Devfest · October 17, 2020

Modern Approaches to Software Development

With examples from a large fintech



Agenda

Modern development layers

- Brute force: fit and failure
- Design: requirements, domain, Clean Architecture
- Storage, NoSQL, microservices
- Delivery, infra, QA, security

Part 1

01

Brute-Force Development

When it works and when it breaks

Idea → Dev → Deploy

Cycle Brute force

Linear cycle and pain

- Business wants the neighbor’s, only better
- Requirements outrun code
- Deploy stays an event
- Few tests, high prod risk

When brute force fits

Applicability When OK

You don’t always need Clean Architecture

- Prototypes for testing an idea
- Research and development experiments — R&D

Part 2

02

Development with Design

When investments in design pay off

What design adds

Layers of meaning

- Requirements gathering and formalization
- Use cases
- Domain model and language
- Patterns and ADRs

Clean Architecture layers

Uncle Bob

Clean Architecture

Dependencies point inward

- Entities — business objects and invariants
- Use Cases — application business rules
- Interface Adapters — controllers, presenters
- Frameworks & Drivers + Dependency Rule

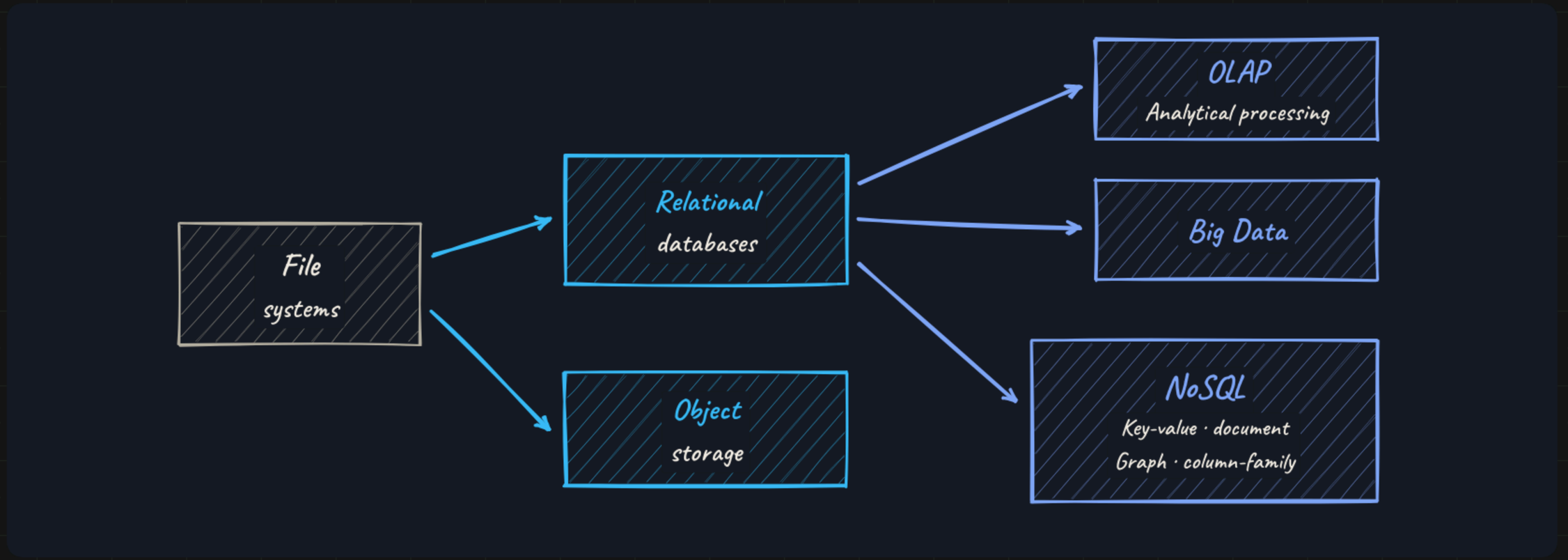
Part 3

03

Data Evolution

From file systems to different storage models

Different tasks need different storage



NoSQL families

Engine per access pattern

- Key-Value — fast lookup
- Column-Family — wide rows
- Document — flexible schema
- Graph — relationships and traversals

Part 4

04

System Architecture

Monolith → SOA → Microservices

Monolith → microservices

Evolution Architecture

Solutions create complexity

- Monolith — simple deploy
- SOA — heavy middleware
- Microservices — harder ops
- DDD sets service boundaries

Part 5

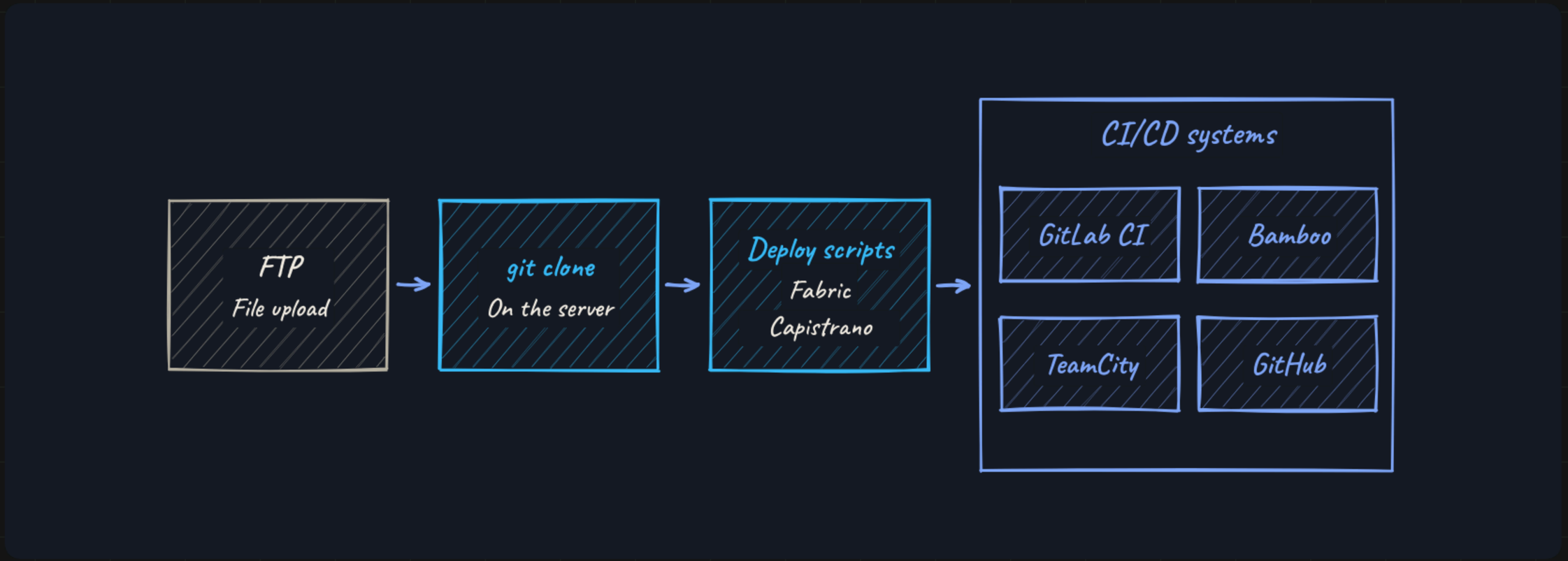
05

Delivery and Infrastructure

From FTP to DevOps and IaaC

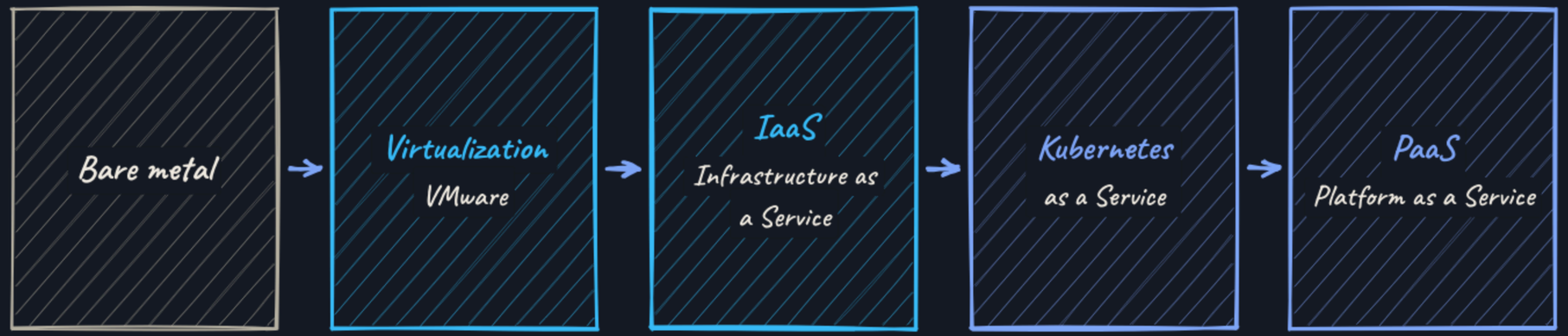
From manual deploys to CI/CD

History Delivery



Infrastructure becomes a product

Infrastructure IaaC



Part 6

06

Quality, Security, Data

What you can’t skip

Shift-left + test pyramid

Quality QA

Build quality in

- Unit tests — fast pyramid base
- Integration / contract — service boundaries
- E2E — few but critical scenarios
- Production testing: canary, shadow, chaos

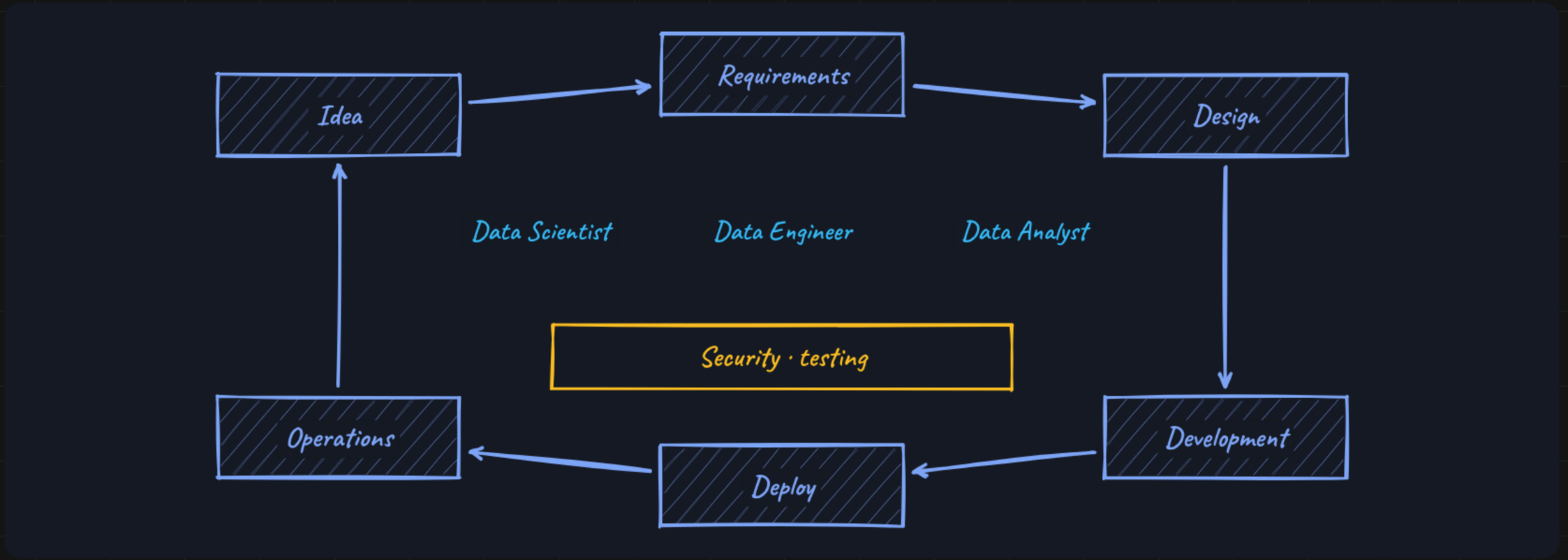
Security in CI

Security AppSec

DevSecOps by design

- SAST / DAST in CI
- SCA — dependencies and CVEs
- Secrets — Vault, KMS
- Threat modeling + least privilege

DataOps brings roles into one cycle



The development loop only grows

- Brute force belongs in prototypes
- Design makes decisions explicit
- Storage is chosen for the task
- Delivery and infrastructure became code
- Quality and security live inside the loop

- every practice adds a step to the same loop

References

Architecture, data, DevOps

- Clean Architecture + Microservices
- Domain-Driven Design
- Martin Kleppmann — Designing Data-Intensive Applications
- The DevOps Handbook + Release It

polomodov.tech

Thank you!

All slides and links are in the Telegram channel

Alexander Polomodov

Technical Director & Fellow,
T-Technologies



@book_cube