

# Practical DDD - Transforming theories into guidelines

Hila Fox



## Key takeaway n°1

When teams take responsibility for their components, architects must become supports rather than decision-makers

## Key takeaway n°2

To be effective, knowledge must make life easier for teams (through opinionated decisions), be actively shared and adapt itself

**AUGURY**

Physical probes for heavy industrial machines, for monitoring and preventive maintenance

IoT  
API

Big Data  
Mobile app

From 100 to 400 in a year and a half

How to scally effectively while keeping...



Independent teams  
Responsible for their assets



Clear perimeters  
Arch/tech consistency

## 1 Adapt the role of architects

An architect to help every Dev Group



Lead without authority

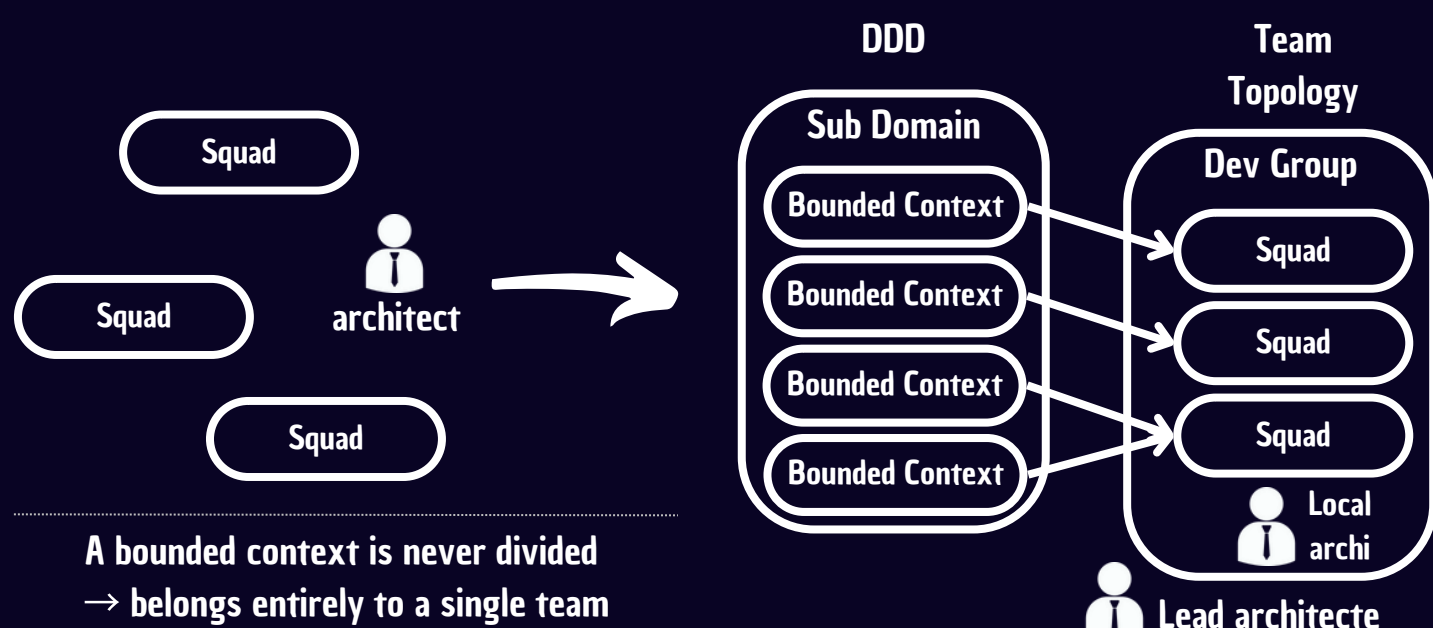
Helping teams make choices



Propagate knowledge

At the disposal of the teams

## 2 Giving teams a functional framework



## 3 Giving teams an architectural framework

Paradigms based on DDD

Conceptual choices already made, adapted to the company's context



6 months of research by the architects concentrated in a book, distributed to devs  
→ Developers don't have enough time to read the book  
→ Those who take the time cannot absorb the dense information

Book split into



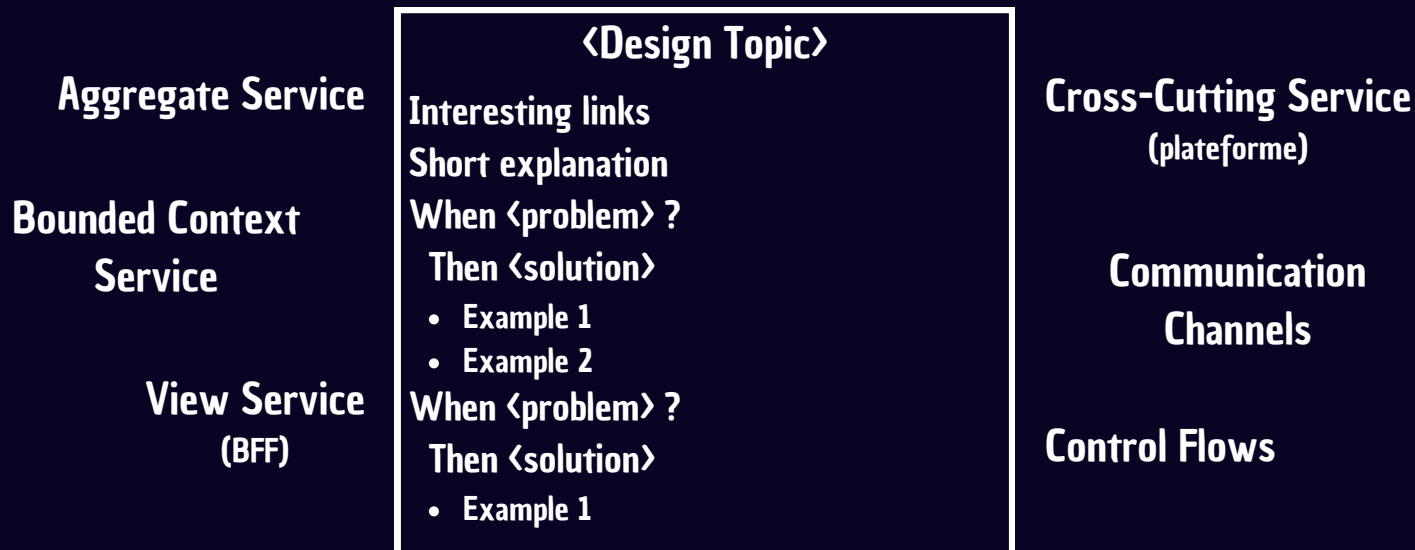
A simplified dictionary  
Tailored to the context



A set of actionable sheets  
The Designs Topics

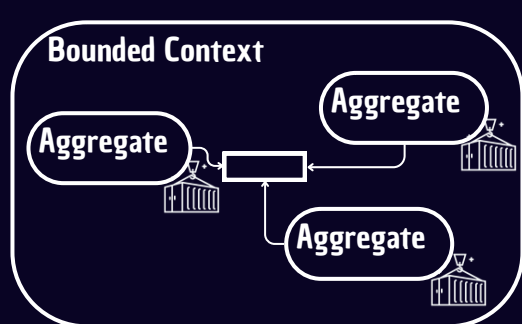


Knowledge that is easier to grasp and spread by architects



### Example: Design Topic Aggregate (micro) service

Technical choice: MicroService at aggregate level



Easier to

- Change the scope of a team
- Split a bounded context



But When the domain is new / with unclear boundaries, Then a microservice at the bounded context level is possible, in trajectory

### Exemple: Design Topic Communication Channels

When to generate a new domain event ?

When should we generate service requests ?



When should we use commands ?

When an aggregate should consume a domain event ?

When to enrich an event with the data ?

What's next? Keep creating and spreading new Design Topics



Distributed entities



Micro frontends



Context Mapping



Generic Domains