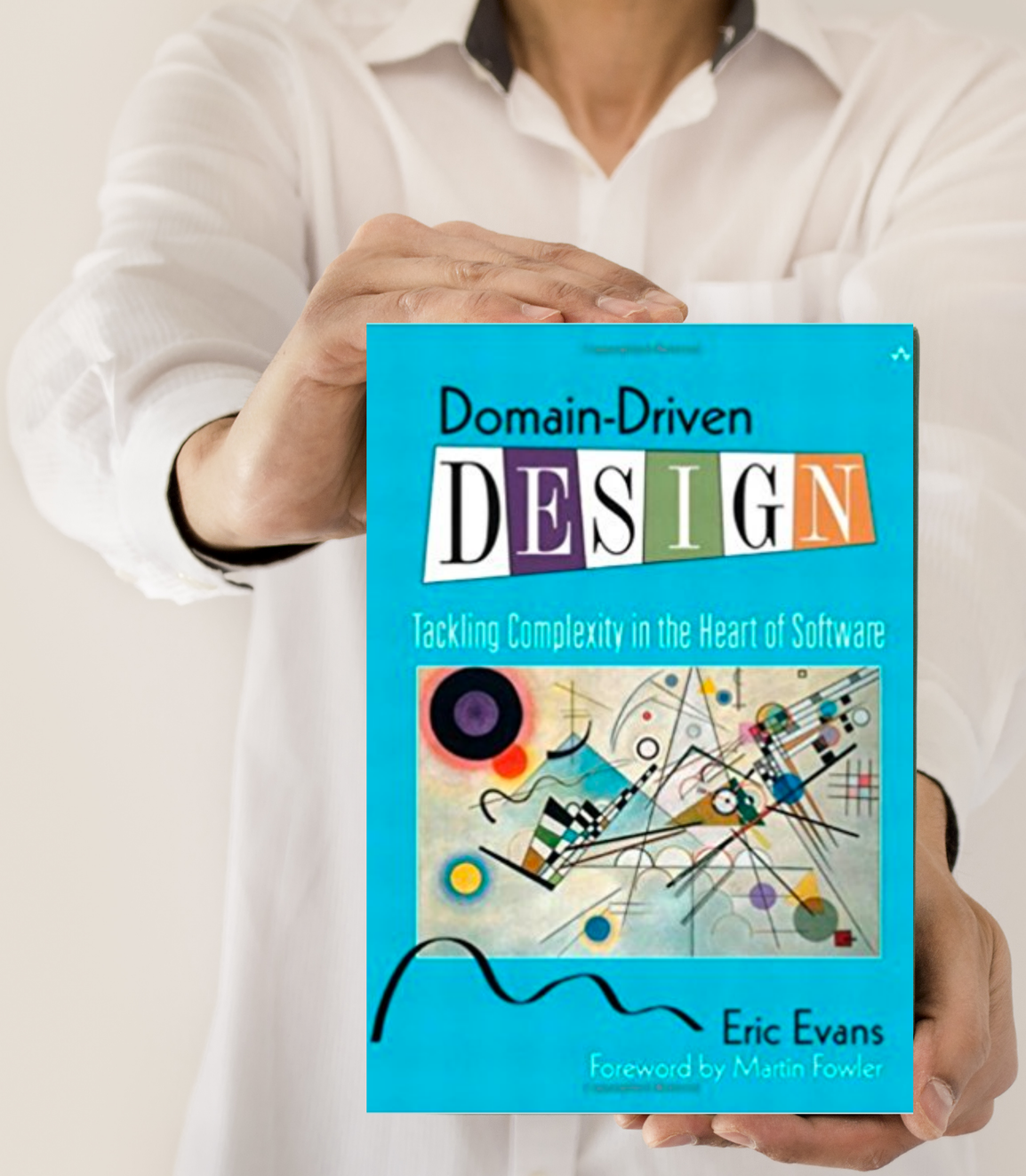
Nick Tune - @ntcoding

# DOMAIN-DRIVEN DESIGN

## HIDDEN LESSONS FROM THE BIG BLUE BOOK


ntcoding

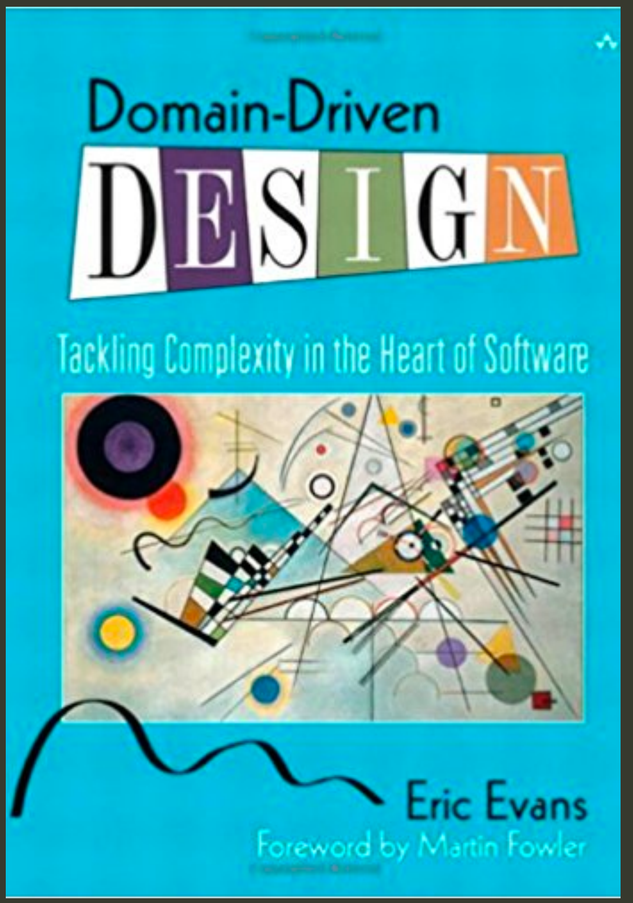

ntcoding


Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler


ntcoding

# ~10 years later

ntcoding


Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler

ntcoding

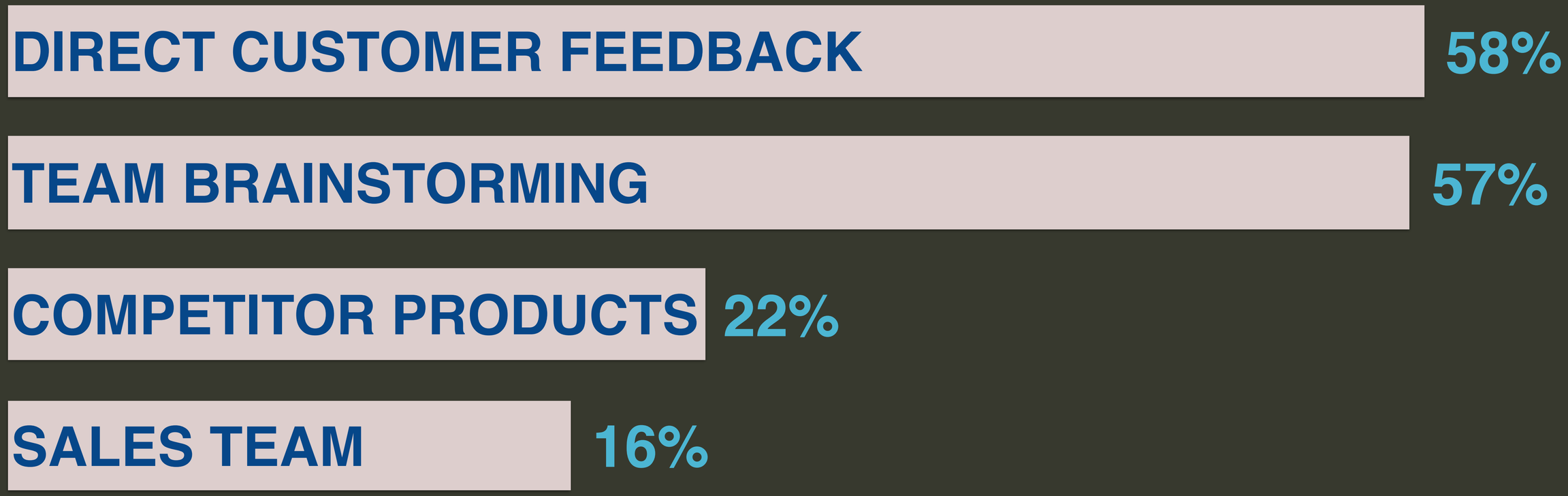
*How can the DDD book be more relevant now than when it was published 15 years ago?*

#1

# KEY TECHNOLOGY TRENDS

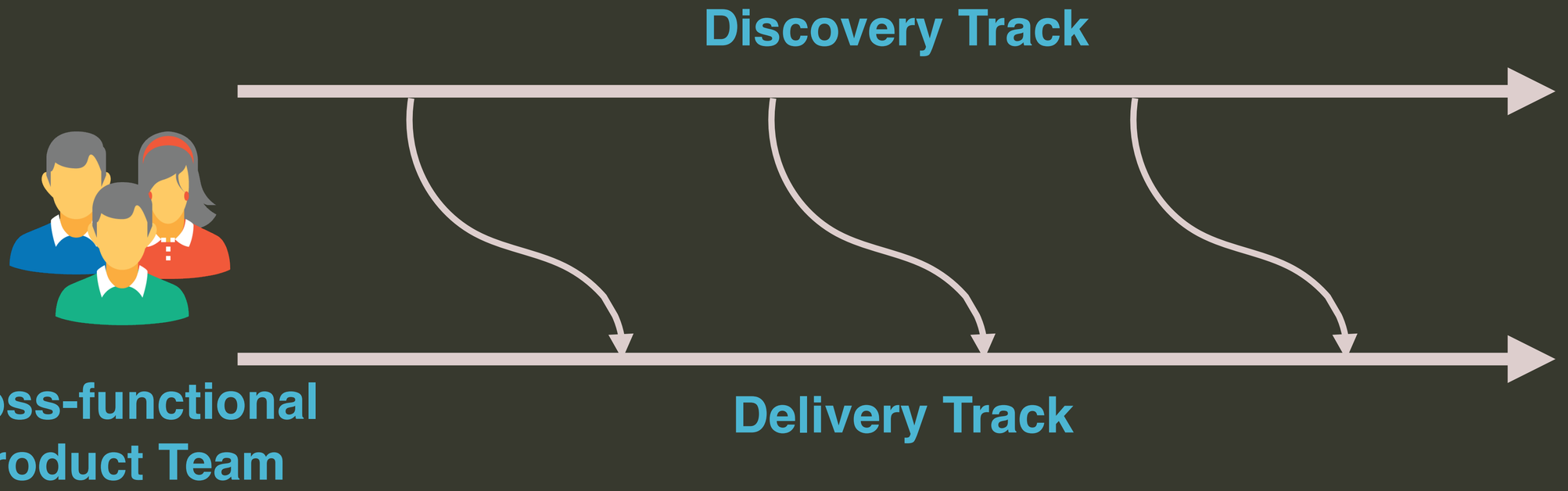
Serverless and cloud free-up developers to focus on higher-level activities.

# TOP SOURCES OF PRODUCT IDEAS

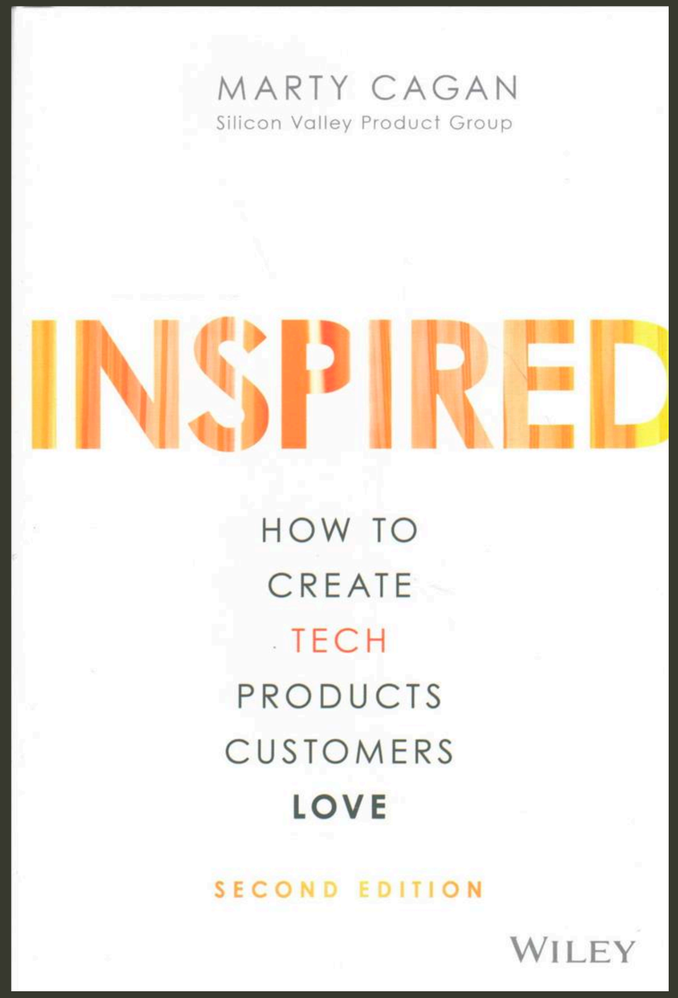


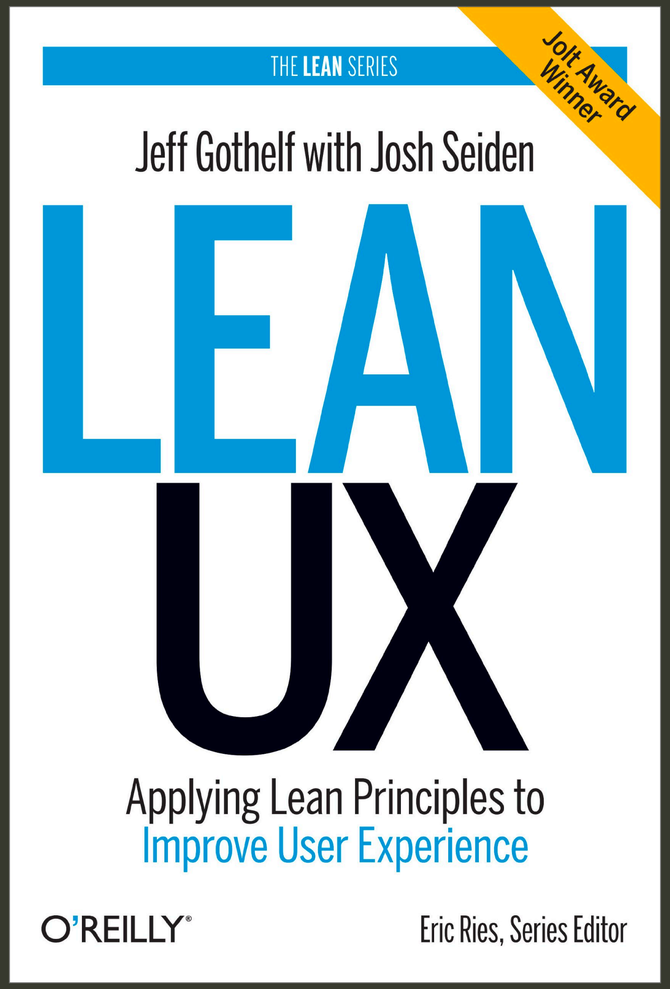

Source: Alpha UX Product Management Insights 2018

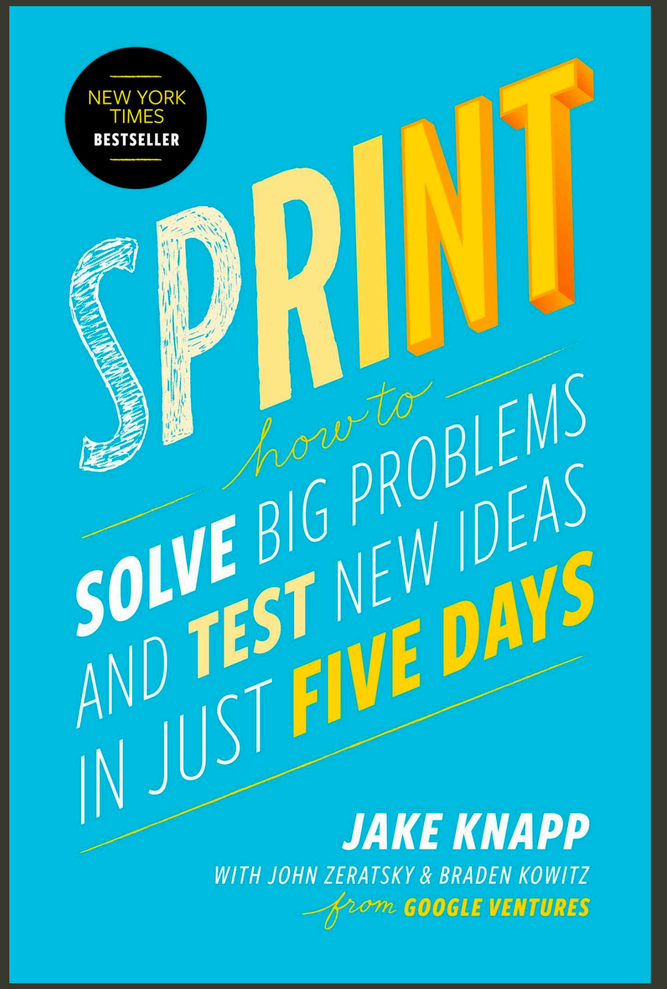
# CONTINUOUS DISCOVERY & DELIVERY



# HIGHLY RECOMMENDED READING



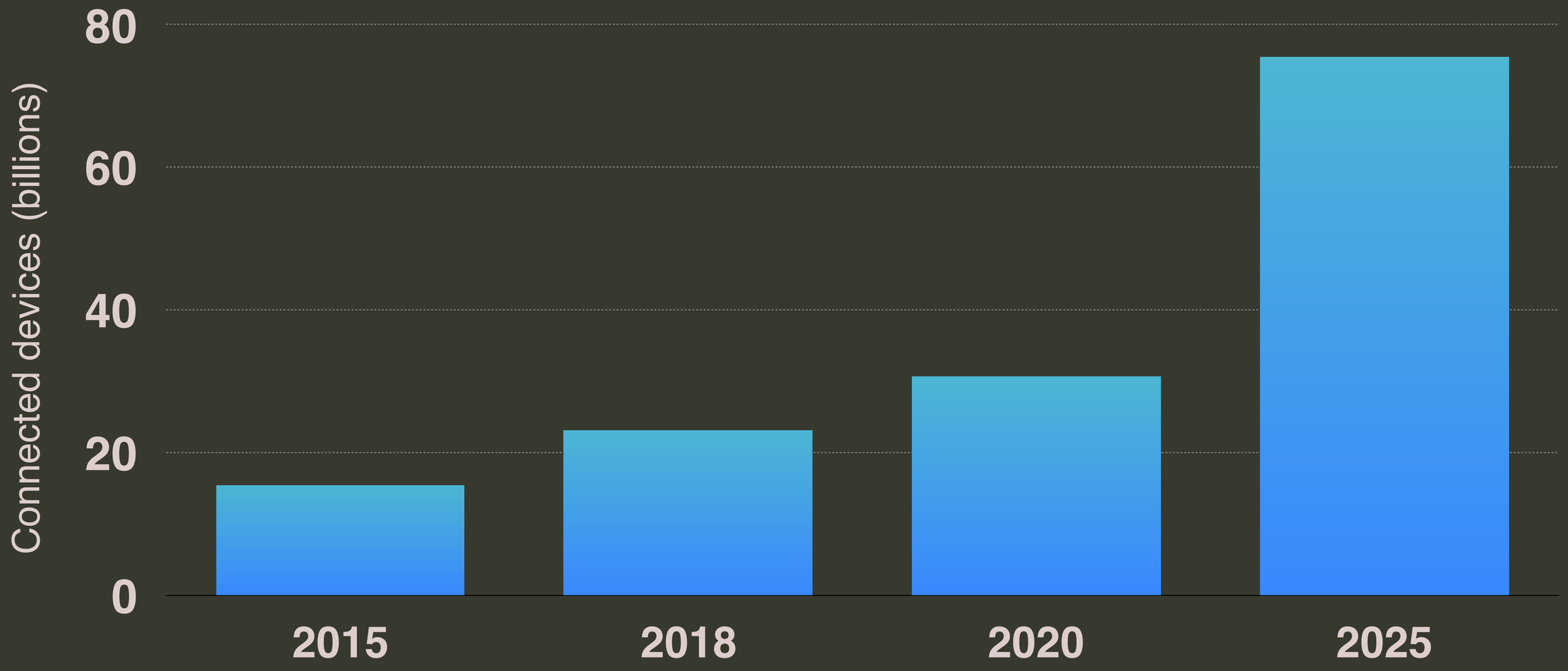




ntcoding

Systems are growing more fragmented and disparate: Microservices, IoT, etc…

# 30 BILLION ‘THINGS’ BY 2020




ntcoding

Source: Statista

There are hidden insights in the DDD book that teach us how to thrive in these conditions.

#2

# EXPLORATION

# SUSTAINABLE EVOLUTION

Build systems that can continuously be evolved to deliver what the market wants… at a competitive pace.

*Our codebase is so rotten we can't compete - we're killing the product (even though we have a substantially bigger engineering workforce)*

**— A True Enterprise Love Story**

# HOW TO CREATE EVOLVABLE SYSTEMS

- SOLID?
- High-quality, high-coverage test suite?
- Modular code?

# HOW TO CREATE EVOLVABLE SYSTEMS

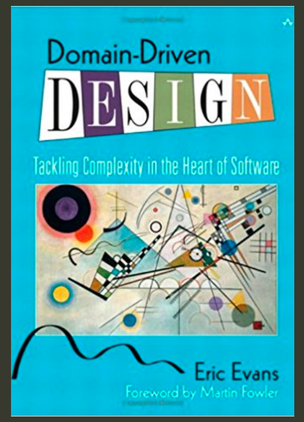
- SOLID?
- High-quality, high-coverage test suite?
- Modular code?

Not Enough

To minimise accidental complexity & create evolvable systems, we need to uncover essential domain complexity.

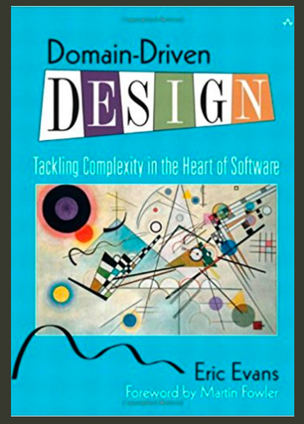
*initial models usually are naive and superficial, based on shallow knowledge.*

— **Eric Evans** (@ericevans0)



*The refactorings that have the greatest impact on the viability of the system are those motivated by new insights into the domain*

— **Eric Evans** (@ericevans0)



Many developers skip the discovery element of DDD because there are no simple domain discovery recipes.


He tried **UML** with domain experts!
ntcoding


ntcoding

# Alberto Brandolini

# THE SAVIOUR OF DDD


ntcoding

# EVENT STORMING

# MODEL DOMAIN AS TIMELINE

CFP Opened

Talk proposal submitted

CFP Closed

Talk Accepted

# ADD COMMANDS, USERS, POLICIES


Submit Proposal

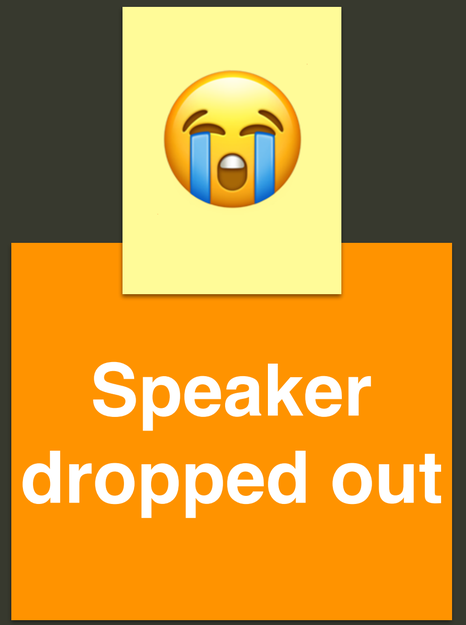
Talk proposal submitted

Proposal Limit Policy

Talk Proposal Rejected

# EXTERNAL SYSTEMS, MONEY, UX


Tickets downloaded




Event booking service

After 10 years of DDD, Brandolini hit gold and made collaborative domain modelling accessible to all of us. No more excuses.

Event Storming gives us the power to eliminate huge swathes of complexity and build evolvable systems.

# PERSONAL ES TRIUMPHS

- Uniting my teams and ‘the business’
- Modelling entire domains
- Engaging many domain experts for hours
- Properly understanding domains like I never have before

# HIDDEN LESSONS

- Domain discovery is fundamental to DDD?
- Discovery enables us to build evolvable software systems?
- We should all be doing event storming?

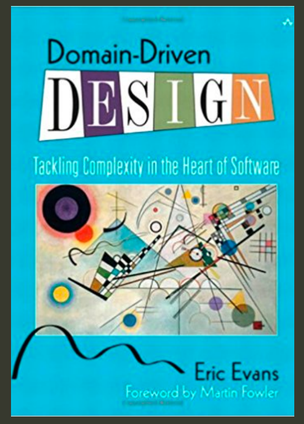
# HIDDEN LESSONS

- Domain discovery is fundamental to DDD?
- Discovery enables us to build evolvable software systems?
- We should all be doing event storming?

Not Hidden Lessons

*model-based communication is not limited to diagrams in UML. To make most effective use of a model, it needs to pervade every medium of communication*

— **Eric Evans** (@ericevans0)

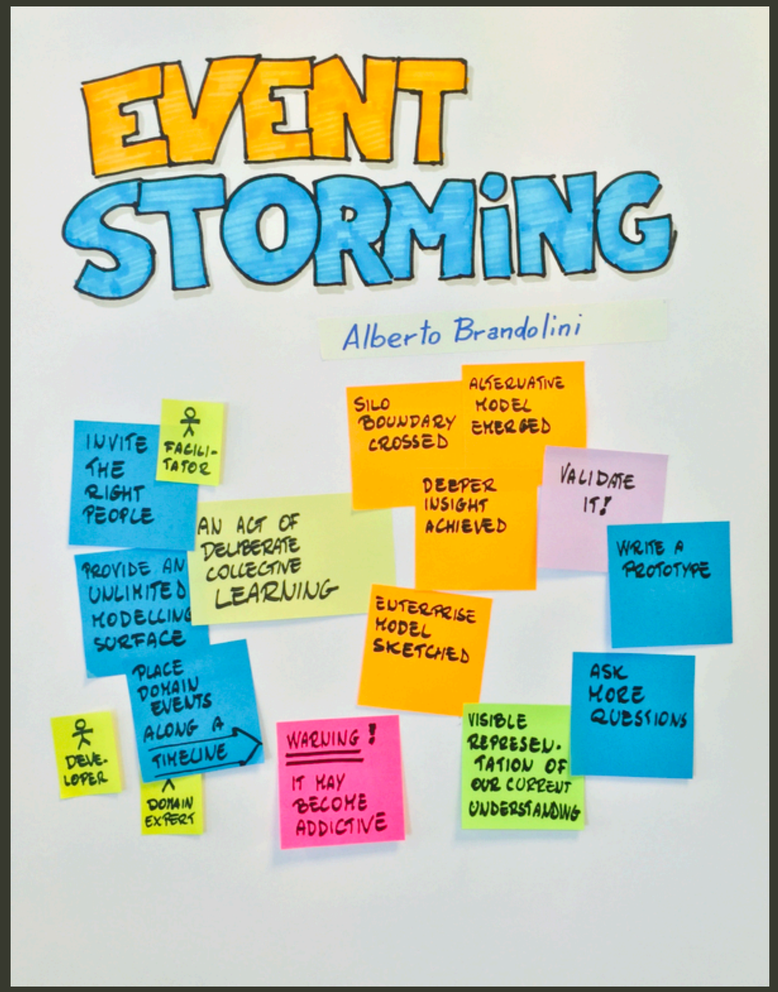


# HIDDEN LESSON

We all need to be passionate about discovery and finding better domain discovery techniques.

*EventStorming is my pizza.*
*Add your own topping.*

— **Alberto Brandolini** (@ziobrando)

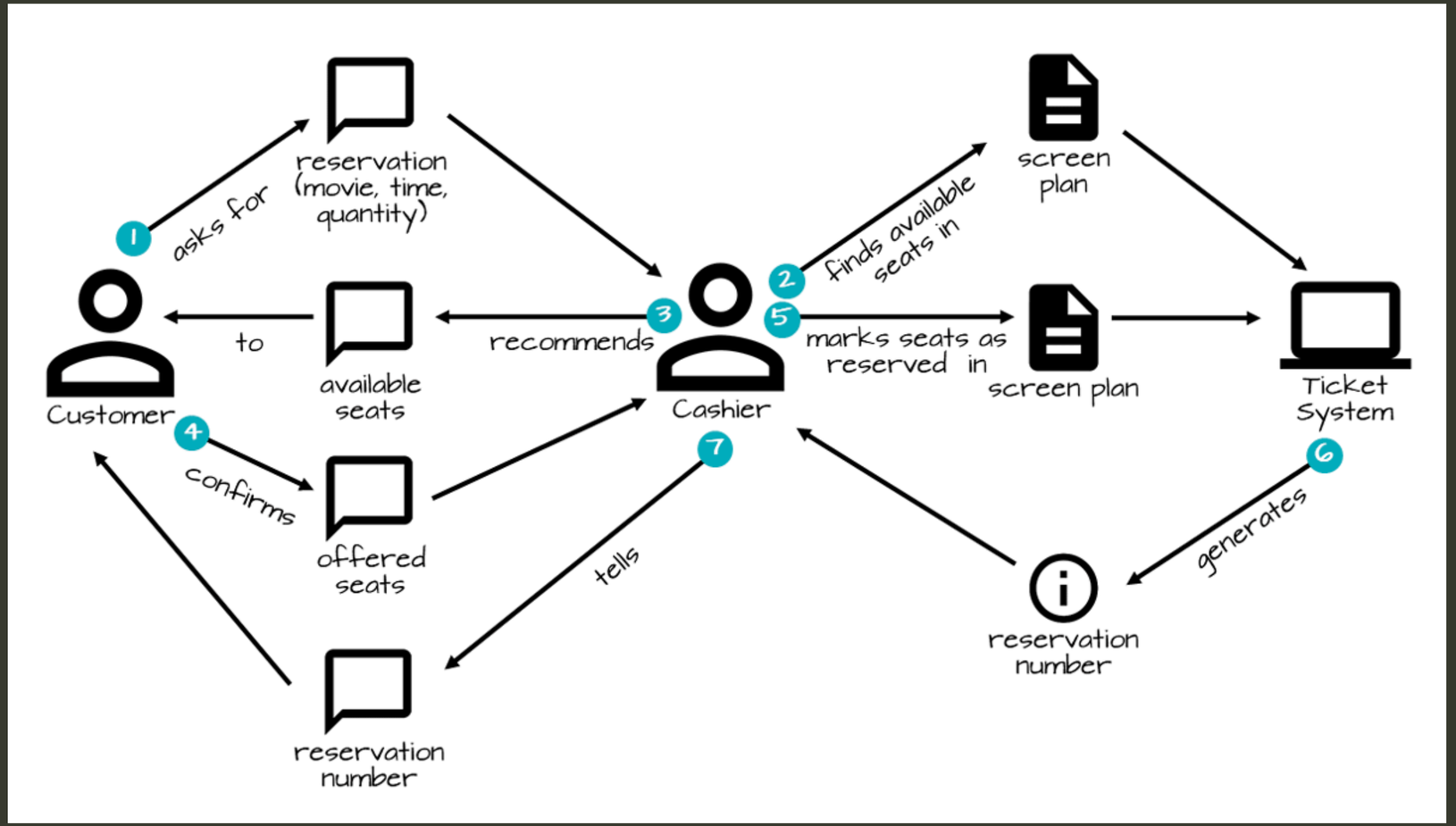



leanpub.com/introducing_eventstorming

eventstorming.com

github.com/mariuszgil/awesome-eventstorming

# DOMAINSTORYTELLING.ORG



#3

# DESIGN

# ANTI-CORRUPTION LAYER

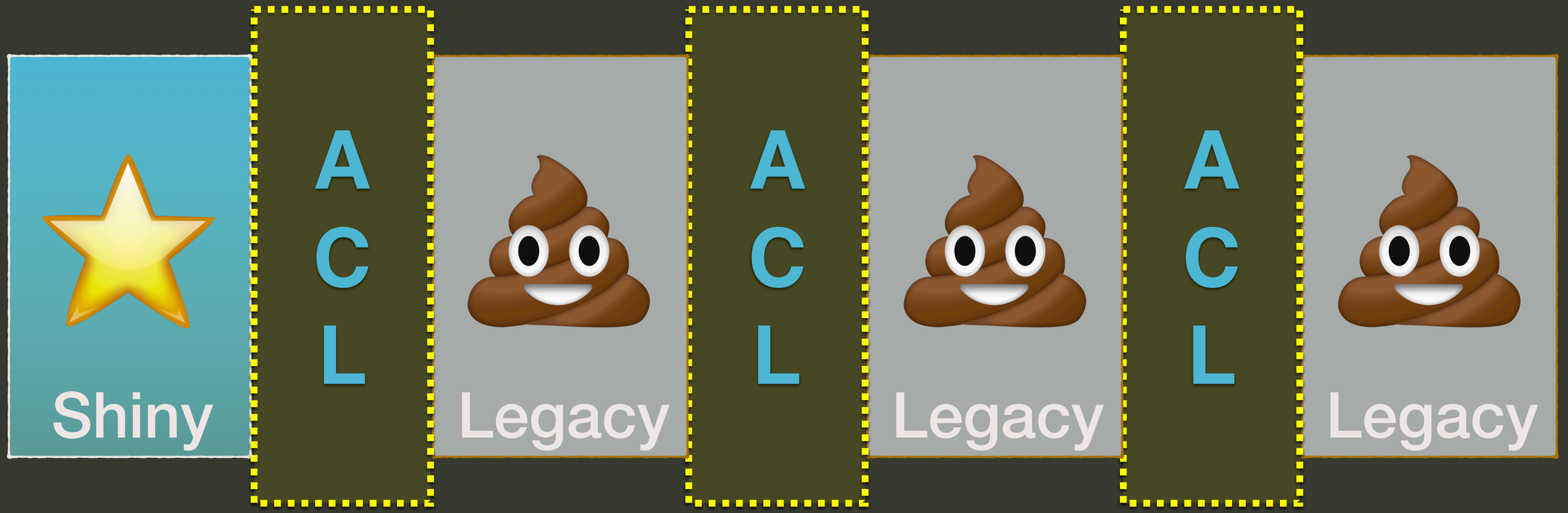
# ANTI-CORRUPTION LAYER


ntcoding


Not time
for an ACL
ntcoding

Sometimes it’s better to clean up the mess rather than throwing another layer of junk on top of it.

# LEGACY LASAGNE




ntcoding

# LAVA LAYER ANTI-PATTERN

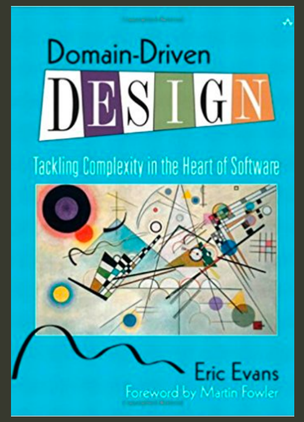
http://mikehadlow.blogspot.co.uk/2014/12/the-lava-layer-anti-pattern.html

ntcoding

*How do expert software designers make good decisions and not become attached to certain principles?*

*Effective domain modelers take a torrent of information searching for the simple view that makes sense of the mass. Many models are tried and rejected or transformed.*

— **Eric Evans** (@ericevans0)



# HIDDEN LESSON

Expert software designers cultivate a toolbox of design heuristics.

There is never a heuristic that is always correct.

*Look for heuristics that challenge your assumption or disprove your theory... that's when you learn something*

— **Mathias Verraes** (@mathiasverraes)

# HEURISTICS FROM THE BOOK

- Use language to identify boundaries
- Look for same thing with different names (duplicate concepts)
- Look for different things hidden under the same name (false cognates)
- Make the implicit explicit (find missing concepts)

# MORE HEURISTICS

- **Money heuristic** (isolate high ROI)
- **Marry-me heuristic** (if two modules know a lot about each others internals they should get married)
- **T-1000 heuristic** (find distributed cohesion)
- **Erlang heuristic** (isolate from failure)

# GAME OF THRONES HEURISTIC

ntcoding

# HEURISTICS MASTERS

**Rebecca Wirfs-Brock** (@rebeccawb)

Cultivating Your Design Heuristics -
https://www.youtube.com/watch?v=fWCt5KWfTuo

**Mathias Verraes** (@mathiasverraes)

Design Heuristics -
https://bit.ly/2kpqMmh

ntcoding

*I'm not a genius consultant....*
*I just have better heuristics*

— **Mathias Verraes** (@mathiasverraes)

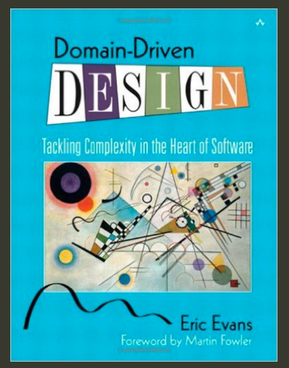
#4

# ORGANISATION


Sooooo many meeeetings!

*Generally speaking, there is a correspondence of one team per BOUNDED CONTEXT...*

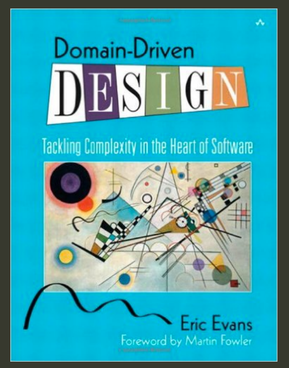
*[BCs] must guide design decisions to reduce the interdependence of parts...*




— **Eric Evans** (@ericevans0)

*...these two groups had different models, but they did not realize it*

— **Eric Evans** (@ericevans0)



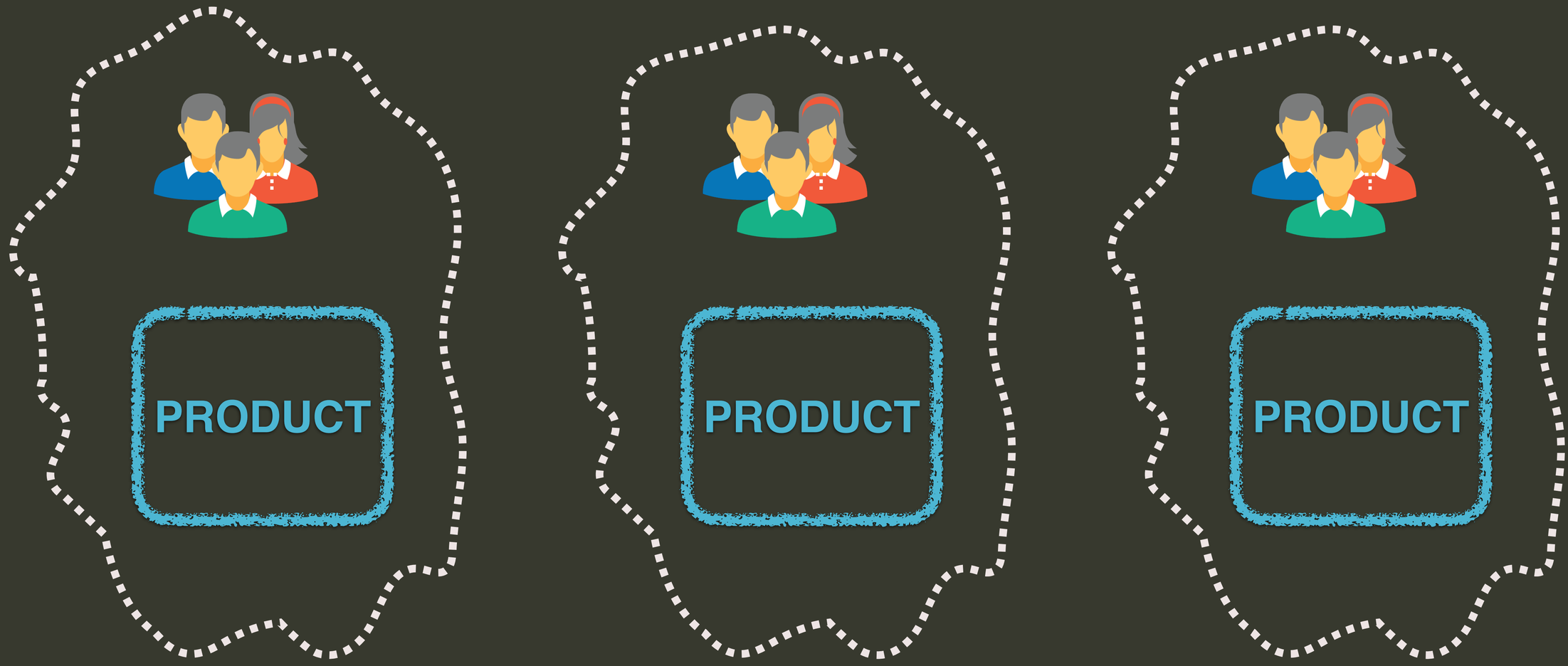
# MULTIPLE HIDDEN CONTEXTS


ntcoding

# MULTIPLE HIDDEN CONTEXTS


Not bounded contexts

# MULTIPLE BOUNDED CONTEXTS




ntcoding


How dare you question my beautiful data model!
ntcoding


ntcoding

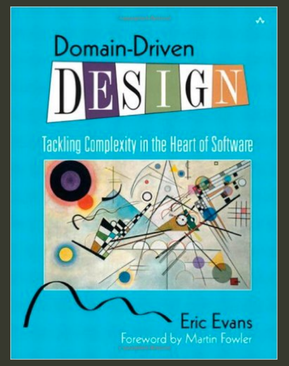
# HIDDEN LESSONS

Bounded contexts decouple **PARTS**. Parts are code and teams.

Bounded contexts are about enabling team autonomy.

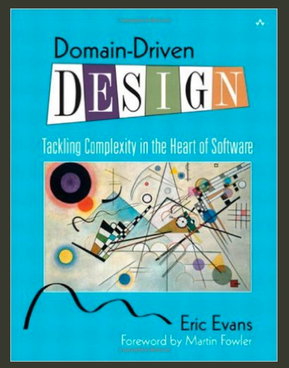
*teams have to make decisions about where to define BOUNDED CONTEXTS and what sort of relationships to have between them. Teams have to make these decisions*

— **Eric Evans** (@ericevans0)



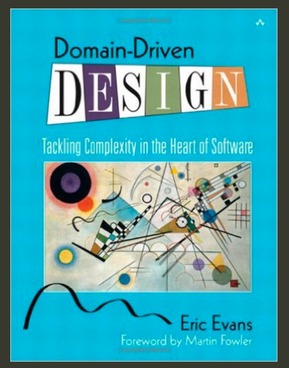
*This definition has to be reconciled with the team organization.*

— **Eric Evans** (@ericevans0)



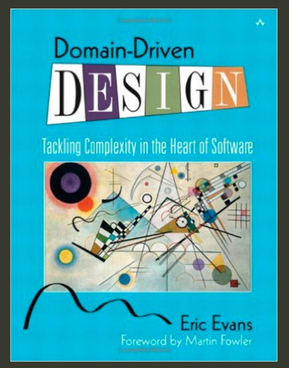
*the problem starts in the way teams are organized and the way people interact.*

— **Eric Evans** (@ericevans0)



*And the emergence of different models can be a result of team organization and development process.*

— **Eric Evans** (@ericevans0)



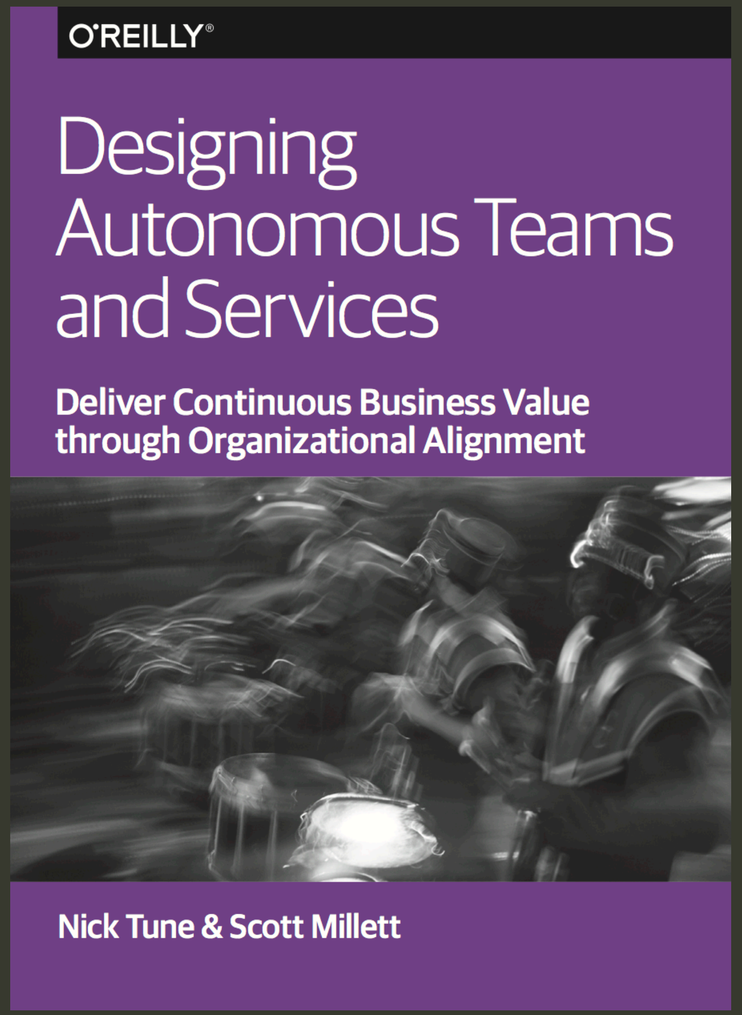
# HIDDEN LESSONS

Software architecture is really sociotechnical architecture.

We can’t ignore teams when shaping the software.

# HOW TO FIND BOUNDARIES

1. EventStorm
2. Context map
3. Apply design heuristics
4. Try lots of models
5. Repeat regularly




ntcoding


Nick! What have you done!
ntcoding
Credit: jaxenter.de

#5

# THE HIDDEN SECRETS

# CONTINUOUS DISCOVERY & DELIVERY

EventStorming brings development teams closer to the business, enabling teams to use customer insights to contribute to the product roadmap.

# DESIGNING COMPLEX SYSTEMS

Cultivating a toolbox of design heuristics equips you to manage complexity in unique, changing and novel environments.

# EVOLVING SYSTEMS

Aligning teams and software systems for high autonomy and motivation is fundamental to sustainably evolving systems.

**Developers** of the future will be product-focused sociotechnical thinkers…

What will **you** be?

# SPEAK TO YOU SOON?

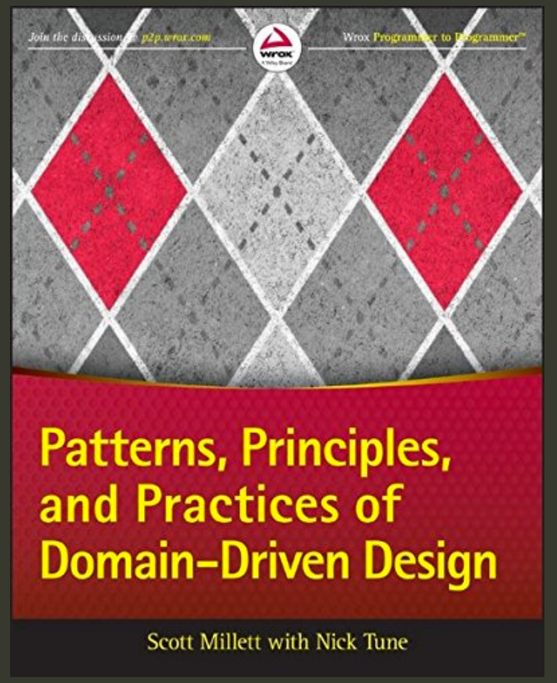
ntcoding.co.uk/workshops

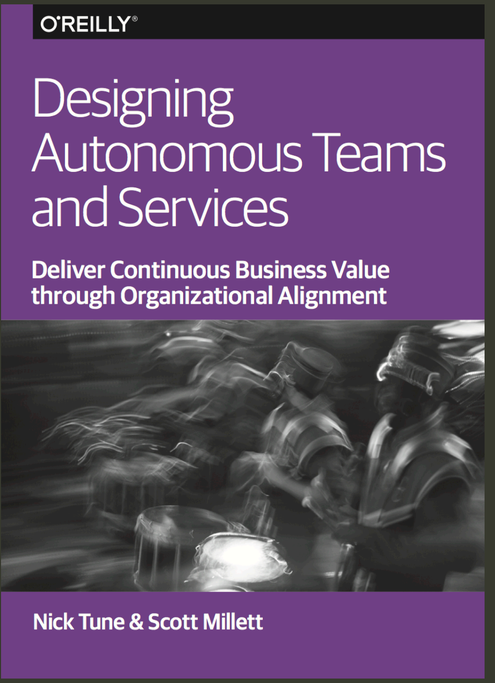
ntcoding.co.uk/blog

ntcoding.co.uk/speaking

@ntcoding

/in/ntcoding




ntcoding