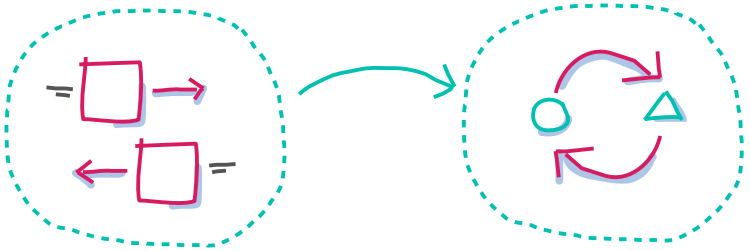
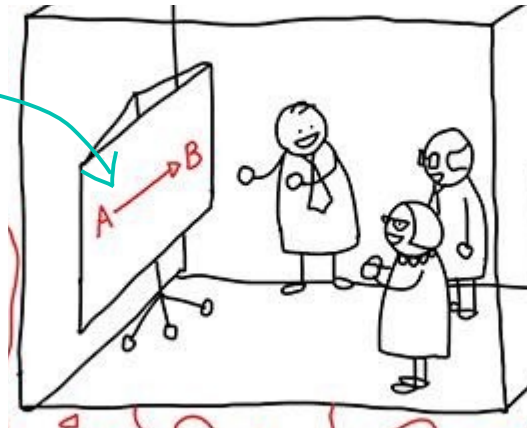


Building Adaptive Systems for a Fast Flow of Change

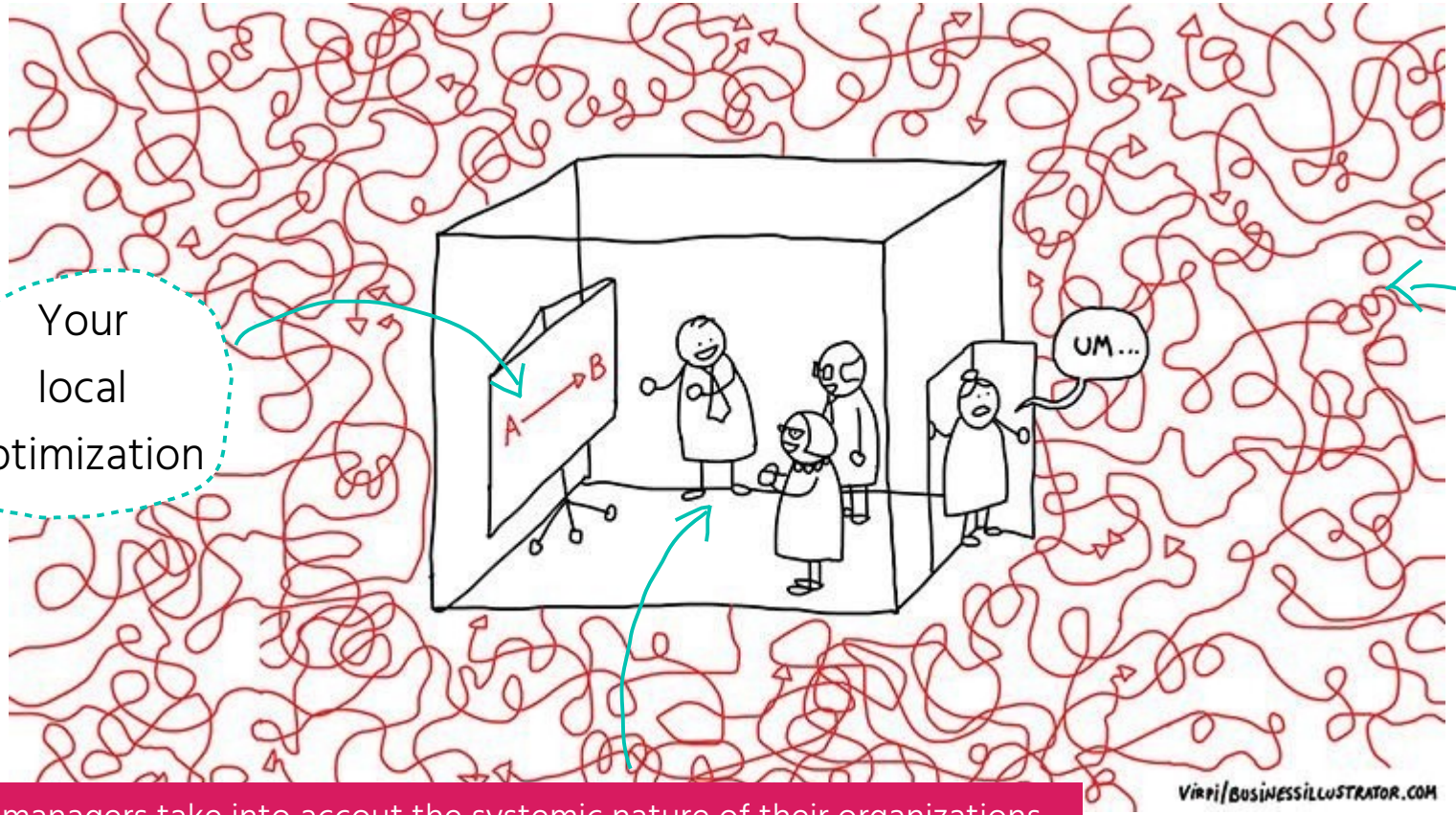
Susanne Kaiser
Independent Tech Consultant
@suksr

Problem with Local Optimization

Your local optimization



“A system is more than the sum of its parts, it’s a product of their interactions.” *



Your local optimization

Your system

“Until managers take into accout the systemic nature of their organizations, most of their efforts to improve their performance are doomed to failure.” *

*) Dr. Russell Ackoff

Challenges of Building Systems

Building the right thing

- How aligned is our solution to business / user needs?
- Have we understood the problem?
- Do we share the same common understanding?

Effectiveness

Building the thing right

- How efficient are our engineering practices?
- How fast can we deliver changes?
- How easy and fast can we change and adapt?

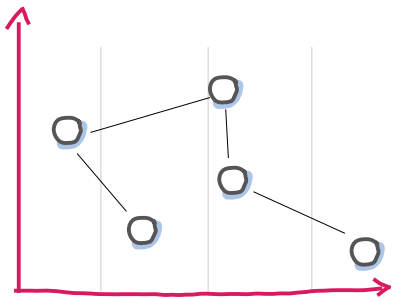
Efficiency

"Doing the wrong thing right is not nearly as good as doing the right thing wrong"

Dr. Russell Ackoff

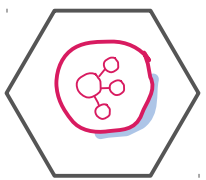
3 Perspectives to Build Adaptive Systems

1. Business-Strategy



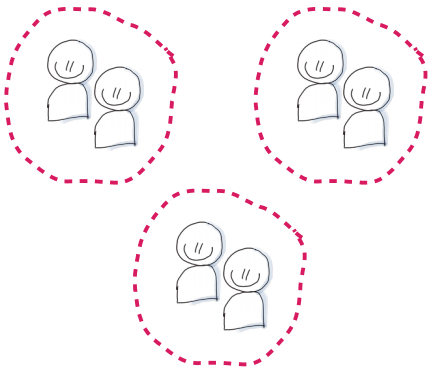
w/ Wardley Mapping

2. Software-Design/
-Architecture



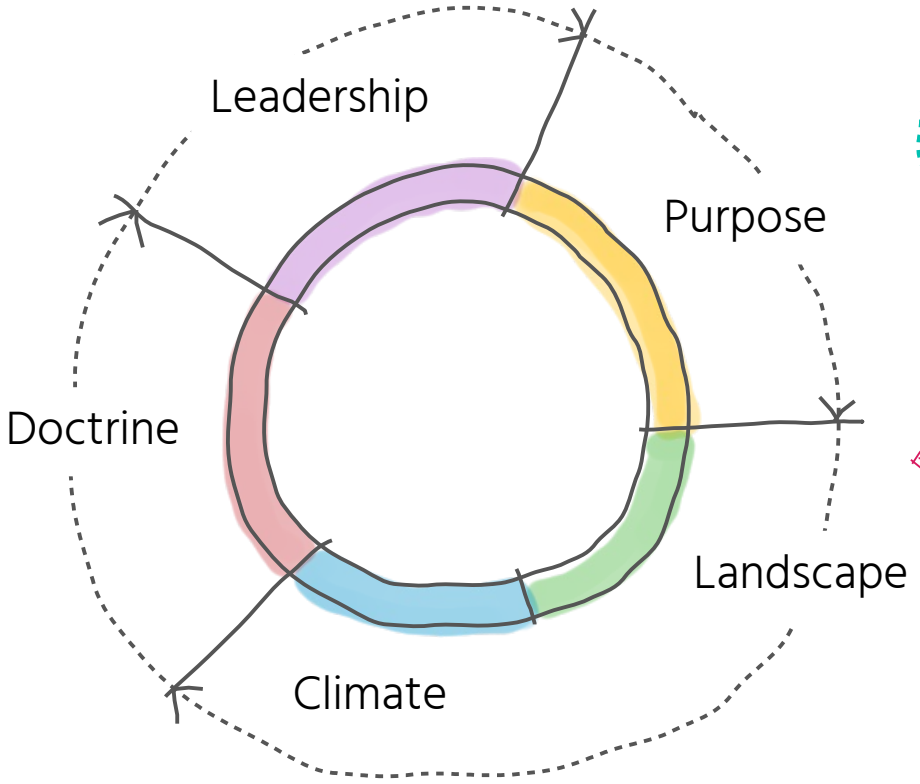
w/ Domain-Driven Design

3. Team-Organization



w/ Team Topologies

Business Strategy w/ Wardley Mapping



"The strategy cycle is a representation of change and how we need to react to it."

Simon Wardley

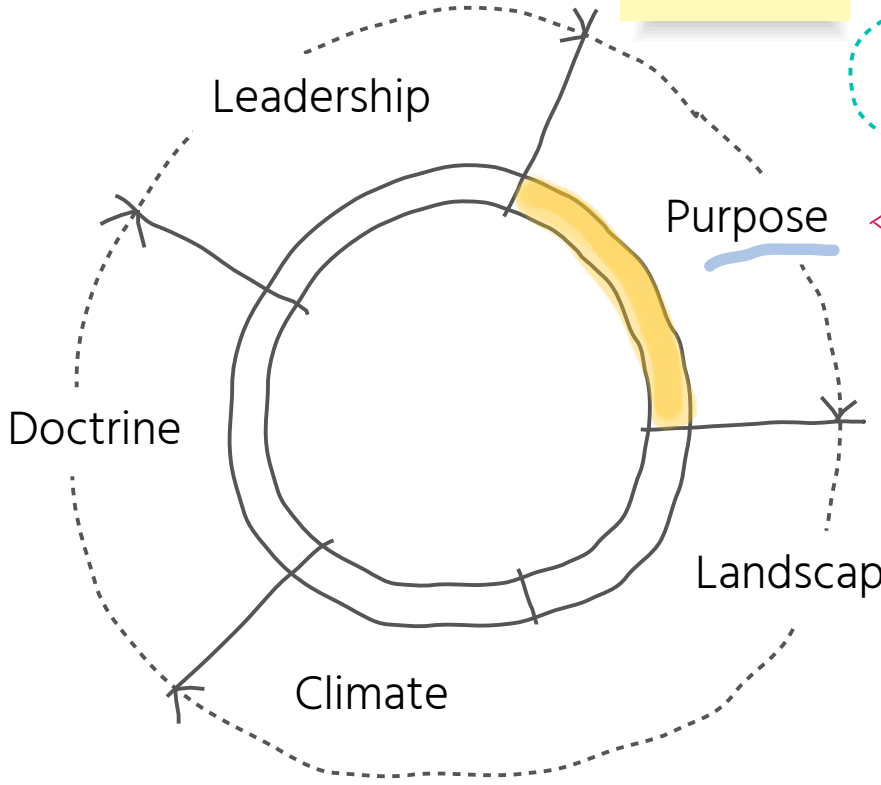
The Strategy Cycle

The Strategy Cycle of Wardley Mapping

the Why of the business

Our Why

Why should the customer choose us?

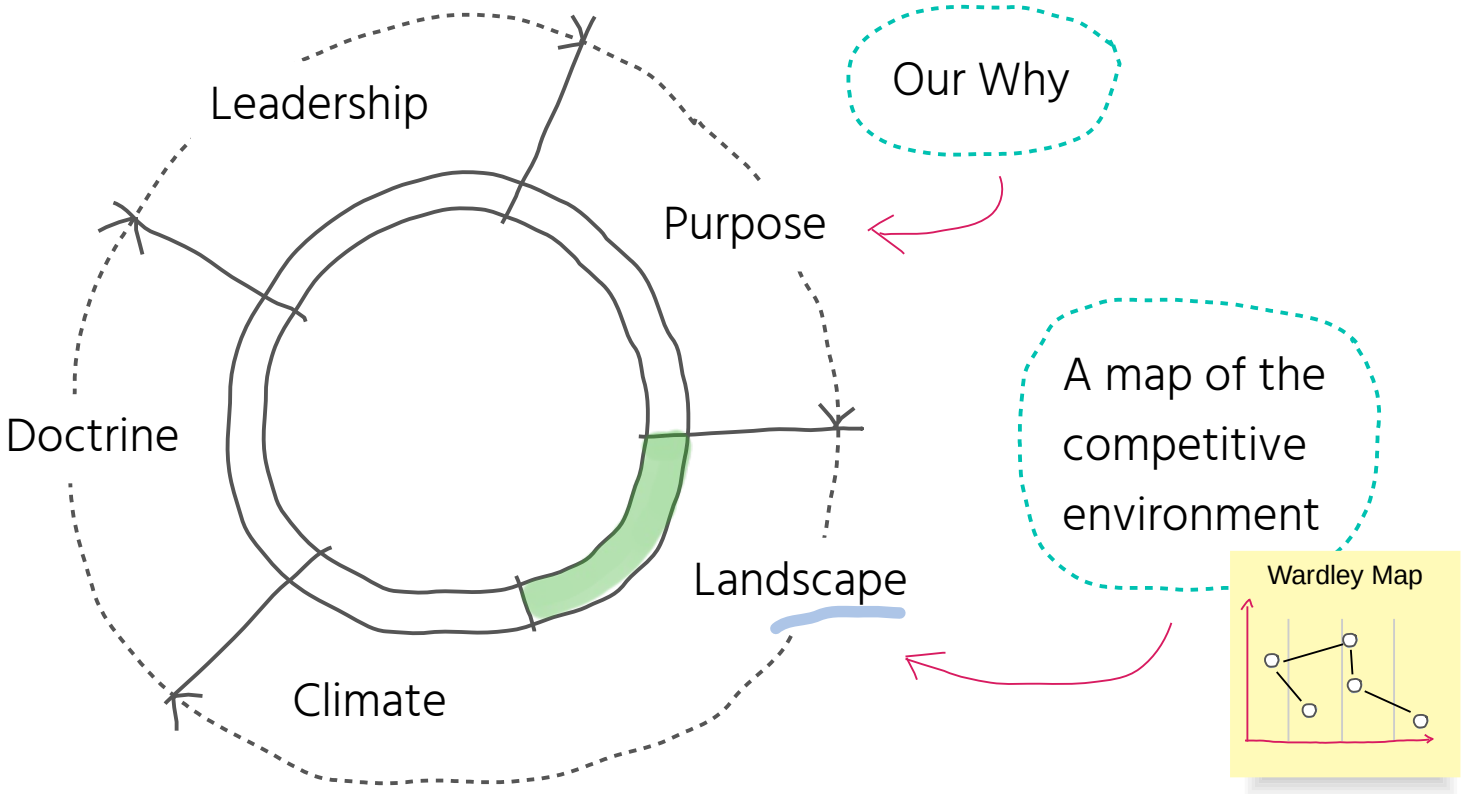


"We sell shirts" - The Shirt Company

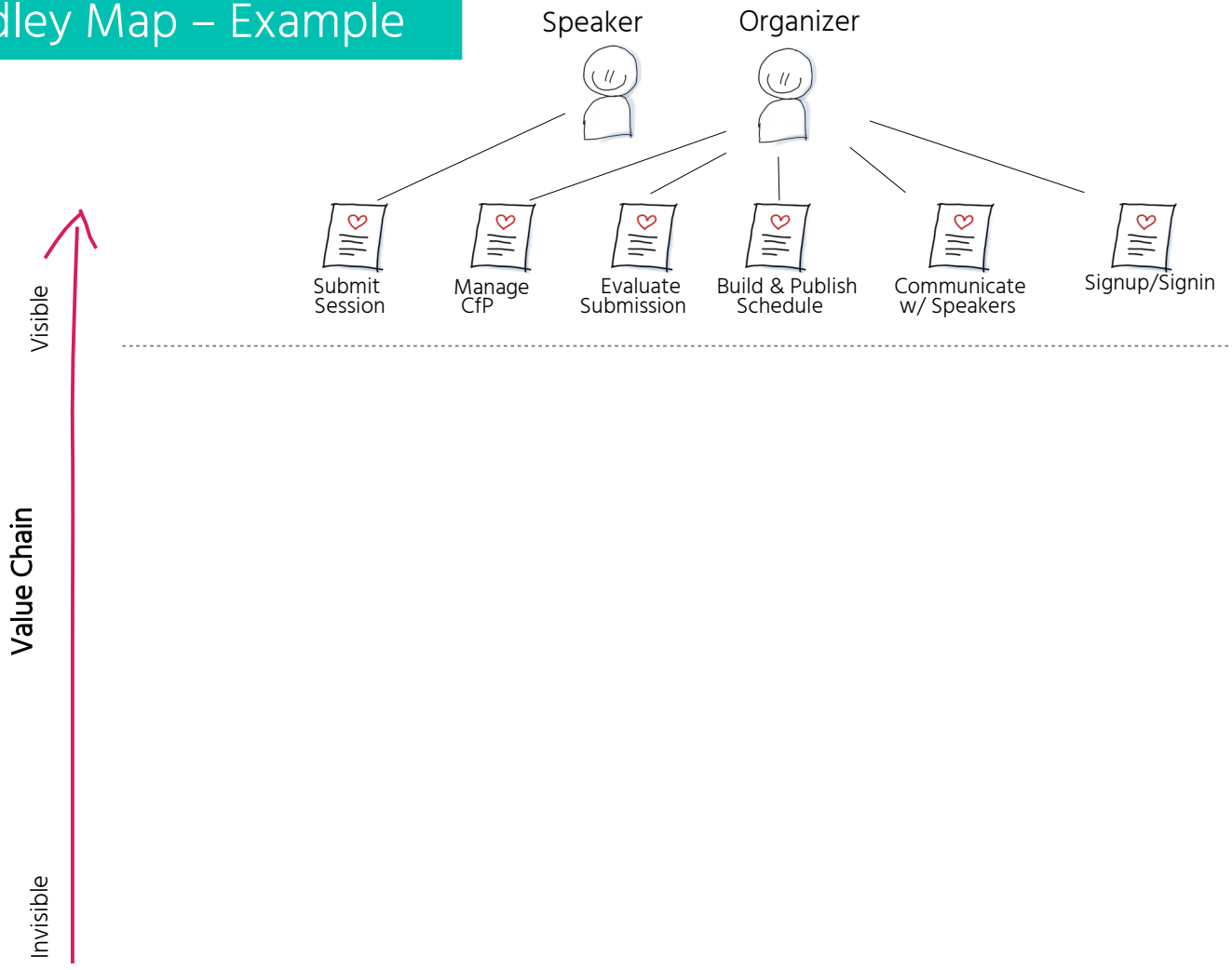
vs.

"Our shirts define fashion" - The Shirt Company

The Strategy Cycle of Wardley Mapping

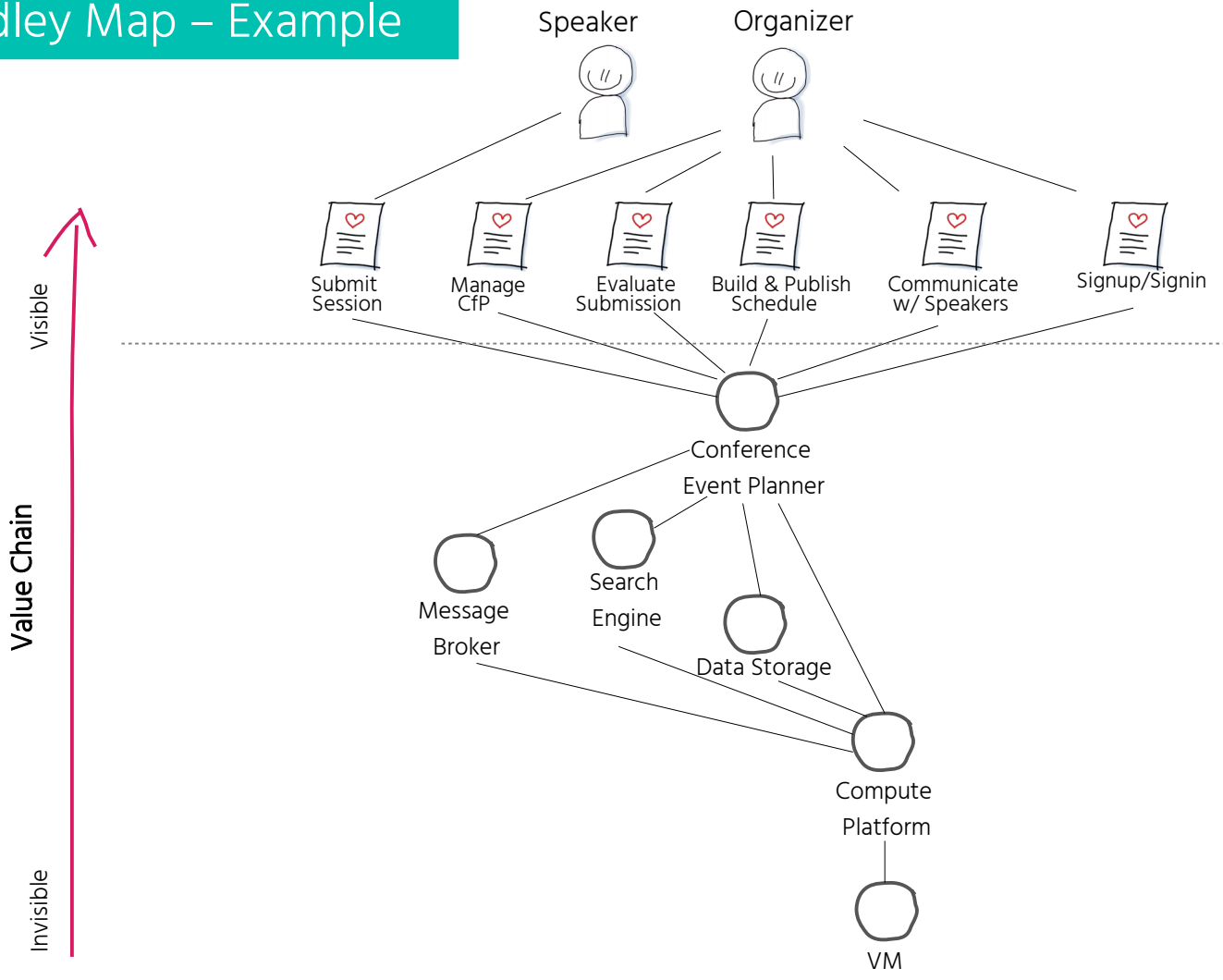


Wardley Map – Example



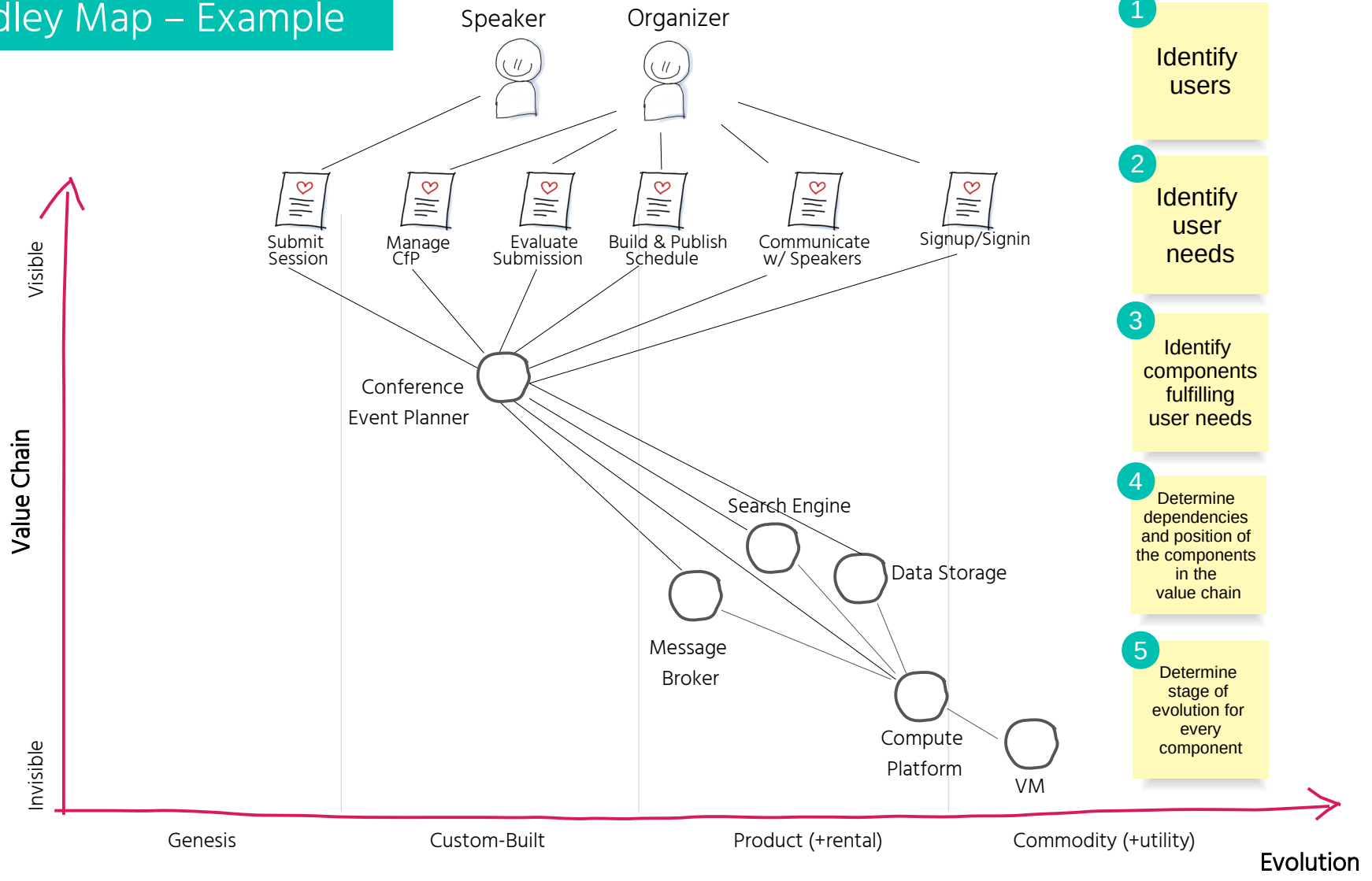
1. Identify users
2. Identify user needs

Wardley Map – Example



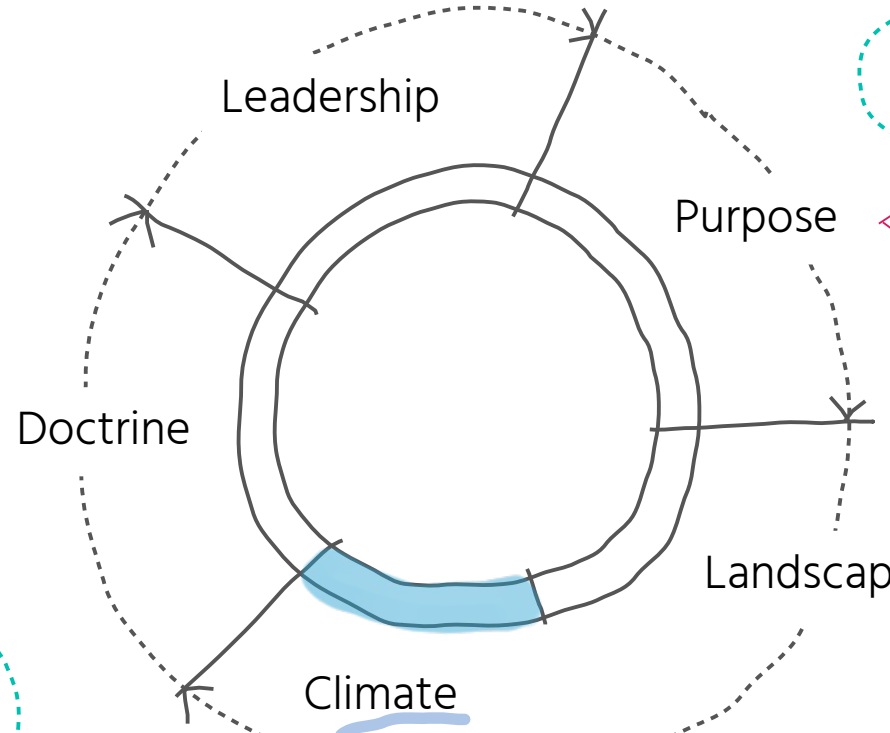
1. Identify users
2. Identify user needs
3. Identify components fulfilling user needs
4. Determine dependencies and position of the components in the value chain

Wardley Map – Example



1. Identify users
2. Identify user needs
3. Identify components fulfilling user needs
4. Determine dependencies and position of the components in the value chain
5. Determine stage of evolution for every component

The Strategy Cycle of Wardley Mapping

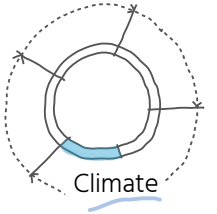


Our Why

A map of the competitive environment

External forces impacting the landscape

The Climatic Patterns (extract)



Demand Competition

Supply Competition

Everything evolves

Visible

Value Chain

Invisible

Past

Current

Future

Movement

Genesis

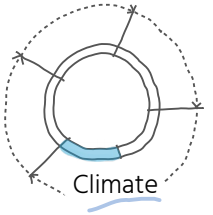
Custom-Built

Product (+rental)

Commodity (+utility)

Evolution

The Climatic Patterns (extract)



Characteristics change

Uncharted

Industrialised

Visible

Invisible

Value Chain

Genesis	Custom-Built	Product (+rental)	Commodity (+utility)
Undefined market	Forming market	Growing market	Mature market
Uncertain	Learning on use	Learning on operation	Known / accepted
Unpredictable	Increasing understanding	Increasing education	Stable
Rare	Slowly increasing consumption	Rapidly increasing consumption	Widespread and stabilising
Poorly understood	Rapid increases in learning	Rapid increase in use	Commonly understood
Chaotic	Domain of experts	Increasing expectation of use	Ordered

Past

Current

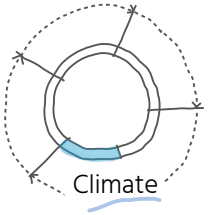
Future

Movement

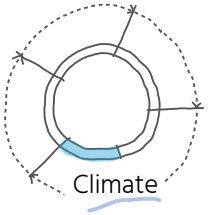
Evolution

@suksr

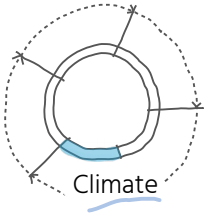
The Climatic Patterns (extract)



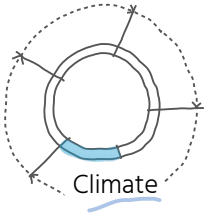
The Climatic Patterns (extract)



The Climatic Patterns (extract)



The Climatic Patterns (extract)



@suksr

Example: Nokia

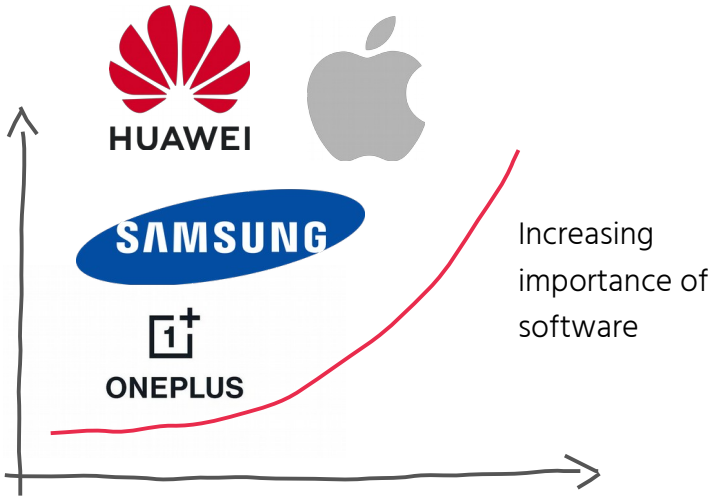
Nokia N-Series

Apple iPhone

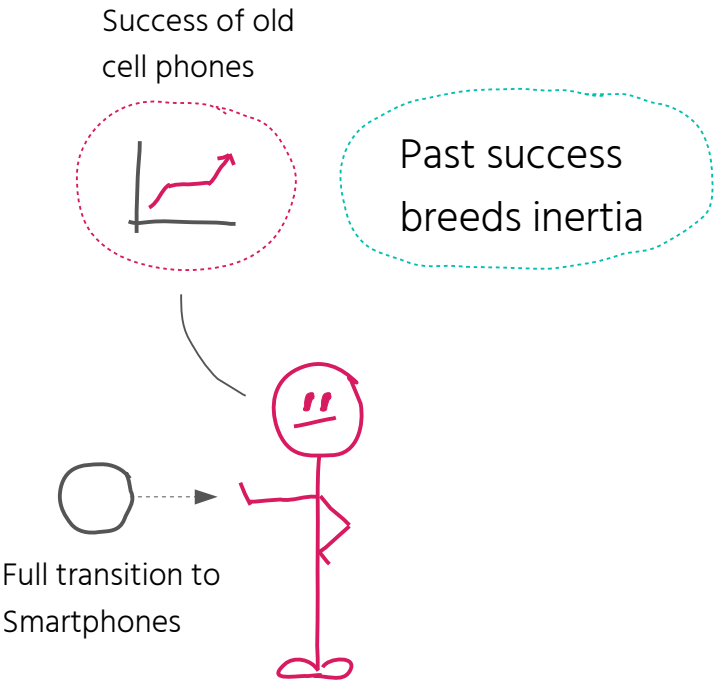
2005

2007

Competitors' Actions and Inertia to Change

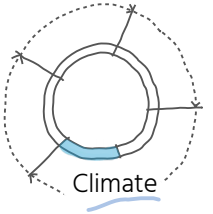


Competitors' actions will change the game



Inertia can kill an organisation

Understanding Climatic Patterns



Speaker

Organizer

Submit Session

Manage CfP

Evaluate Submission

Build & Publish Schedule

Communicate w/ Speakers

Signup/Signin

"Understanding climatic patterns is important when anticipating change."
- Simon Wardley

Climatic patterns give you an idea what can change and where to invest

Conference Event Planner

Search Engine

Data Storage

Message Broker

Everything evolves through supply and demand competition

Points of potential change / where to invest?

Cloud-hosted Services

Serverless

Compute Platform

VM

Visible

Invisible

Value Chain

Genesis

Custom-Built

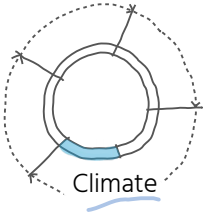
Product (+rental)

Commodity (+utility)

Evolution

@suksr

Understanding Climatic Patterns



Speaker
Organizer
Attendee

Rate Session
Submit Session
Manage CfP
Evaluate Submission
Build & Publish Schedule
Communicate w/ Speakers
Signup/Signin

"Understanding climatic patterns is important when anticipating change."
- Simon Wardley

Climatic patterns give you an idea what can change and where to invest

Visible
Invisible
Value Chain

Higher order systems create new source of value
+ Worth

Conference
Event Planner

Search Engine
Data Storage
Message Broker
Everything evolves through supply and demand competition
Compute Platform
VM

+ Efficiency
Cloud-hosted Services
+ Efficiency
Serverless

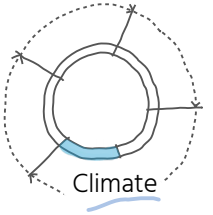
Efficiency enables innovation

Points of potential change / where to invest?

Genesis
Custom-Built
Product (+rental)
Commodity (+utility)
Evolution

@suksr

Understanding Climatic Patterns



Speaker
Organizer
Attendee

Rate Session
Submit Session
Manage CfP
Evaluate Submission
Build & Publish Schedule
Communicate w/ Speakers
Signup/Signin

"Understanding climatic patterns is important when anticipating change."
- Simon Wardley

Climatic patterns give you an idea what can change and where to invest

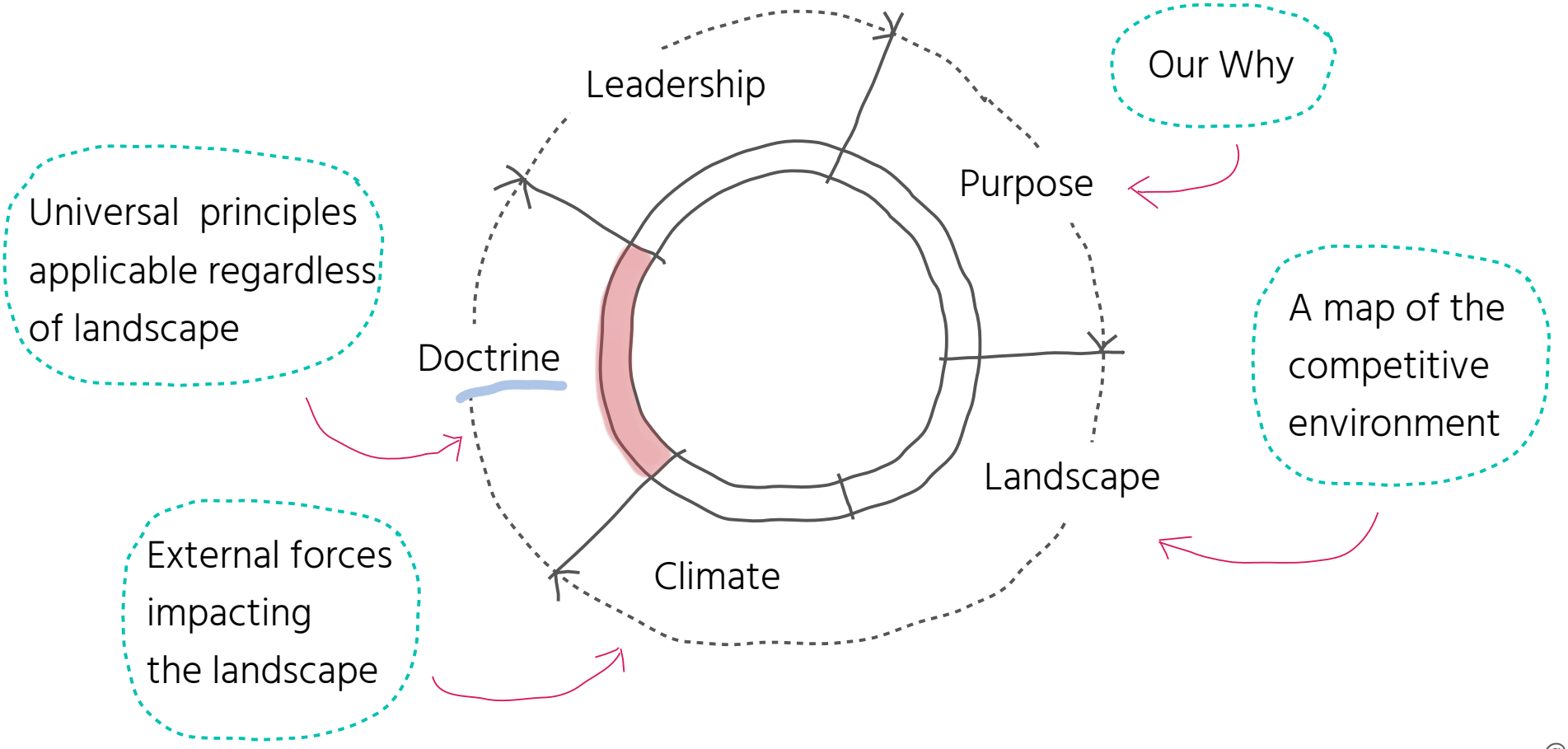
Visible
Invisible
Value Chain

Higher order systems create new source of value
+ Worth
Conference
Event Planner
Success breeds inertia
+ Inertia
Search Engine
+ Efficiency
Data Storage
Cloud-hosted Services
Efficiency enables innovation
Points of potential change / where to invest?
Message Broker
Everything evolves through supply and demand competition
+ Efficiency
+ Inertia
Serverless
Compute Platform
VM

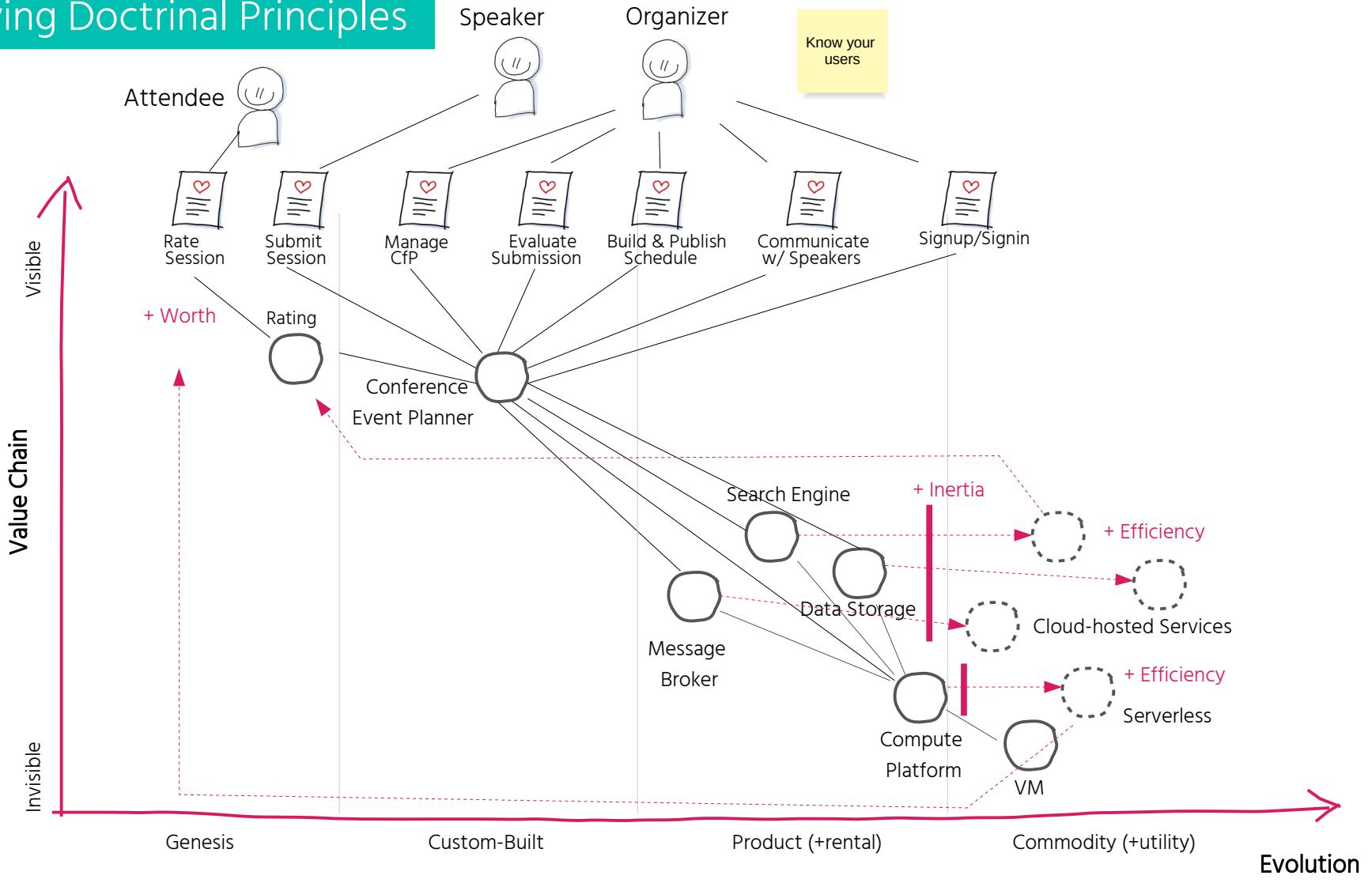
Genesis
Custom-Built
Product (+rental)
Commodity (+utility)
Evolution

@suksr

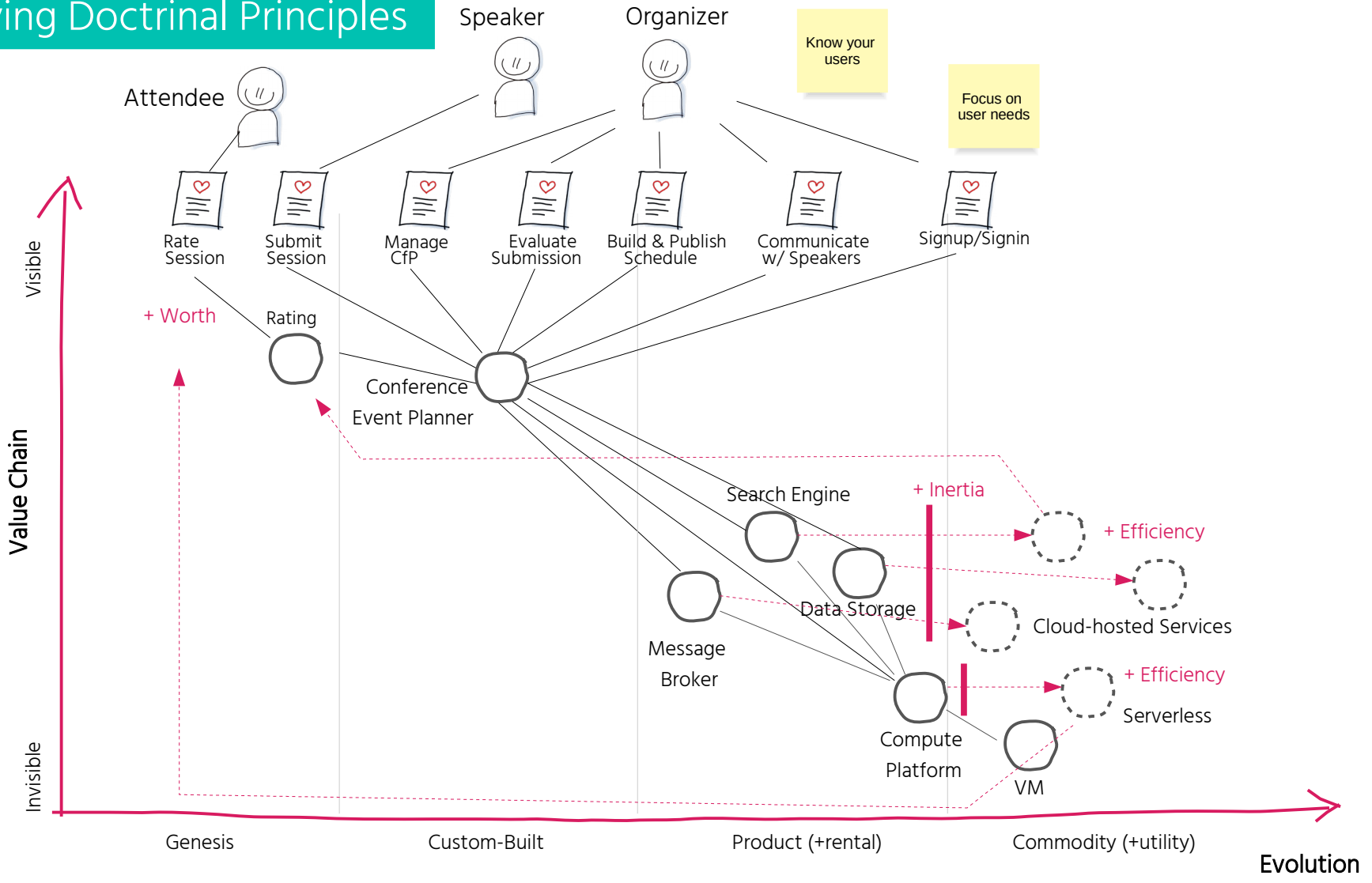
The Strategy Cycle of Wardley Mapping



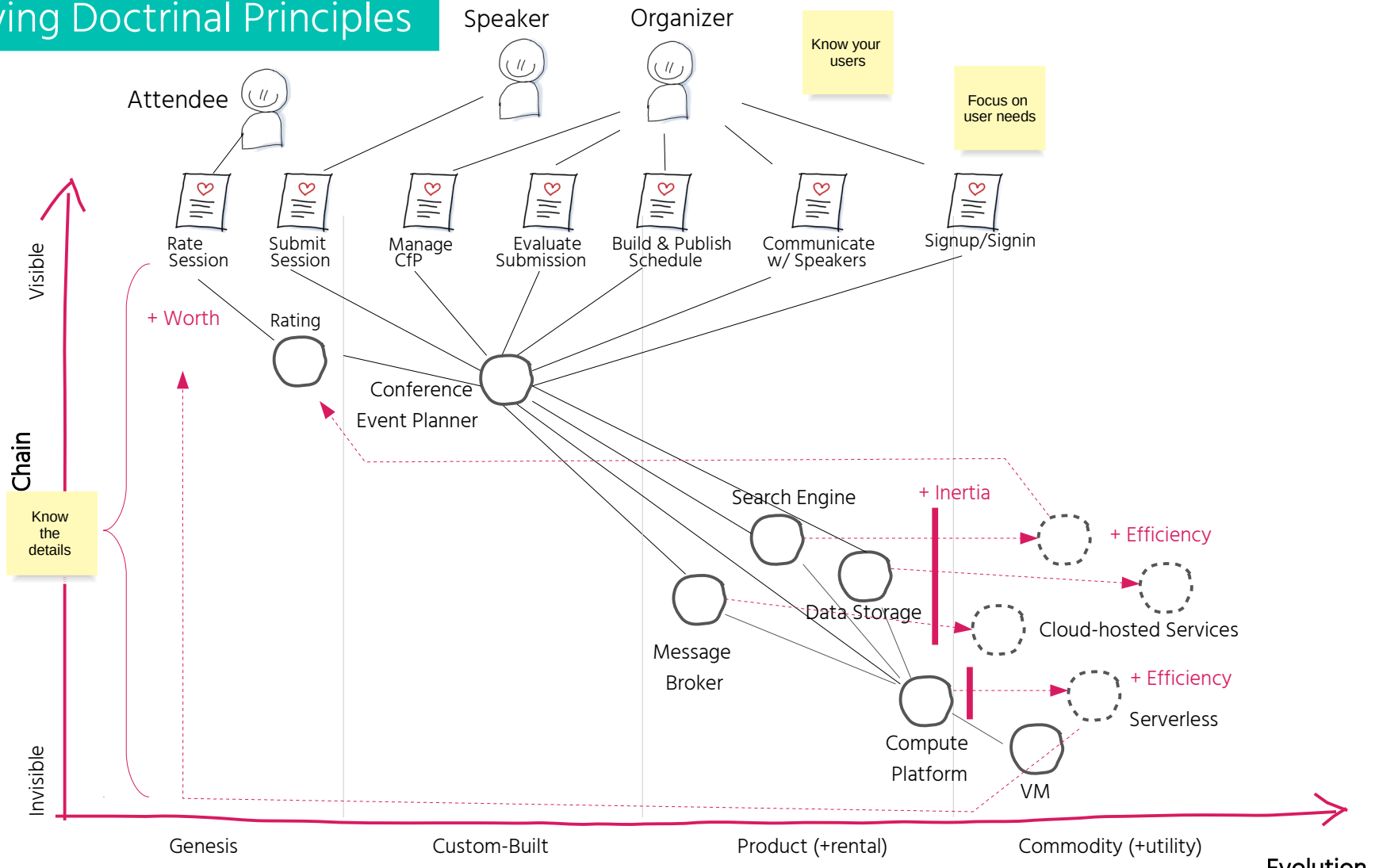
Applying Doctrinal Principles



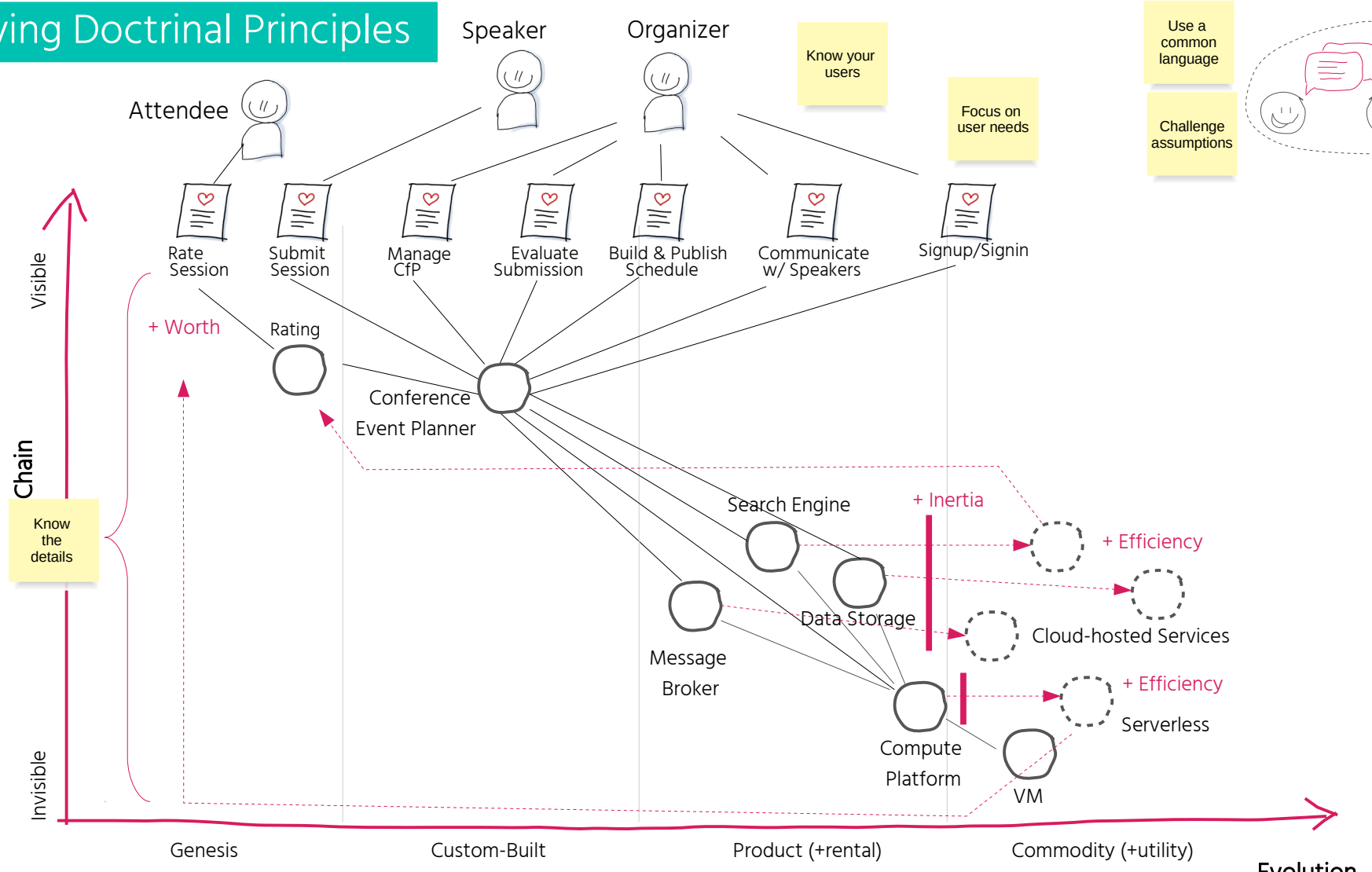
Applying Doctrinal Principles



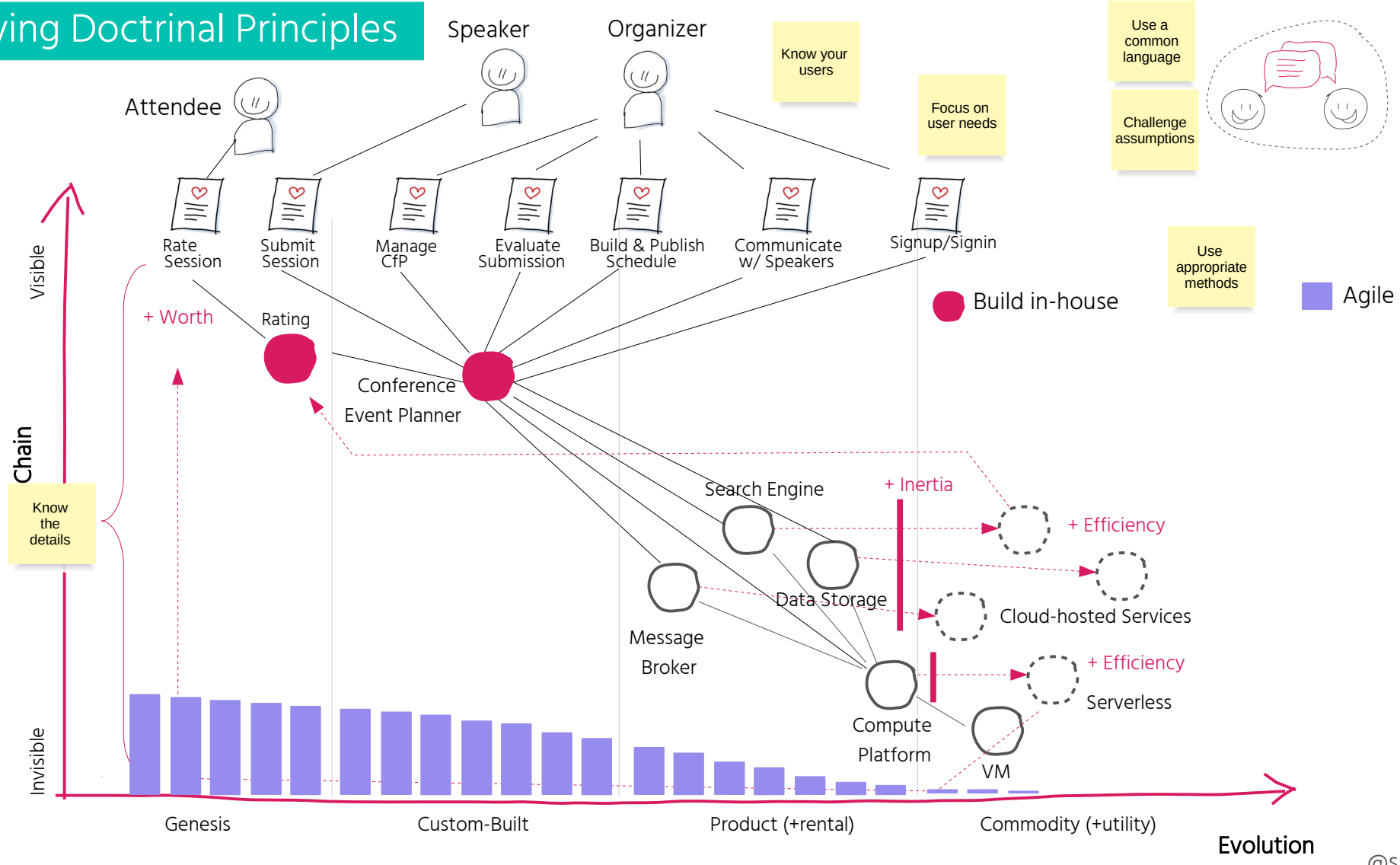
Applying Doctrinal Principles



Applying Doctrinal Principles

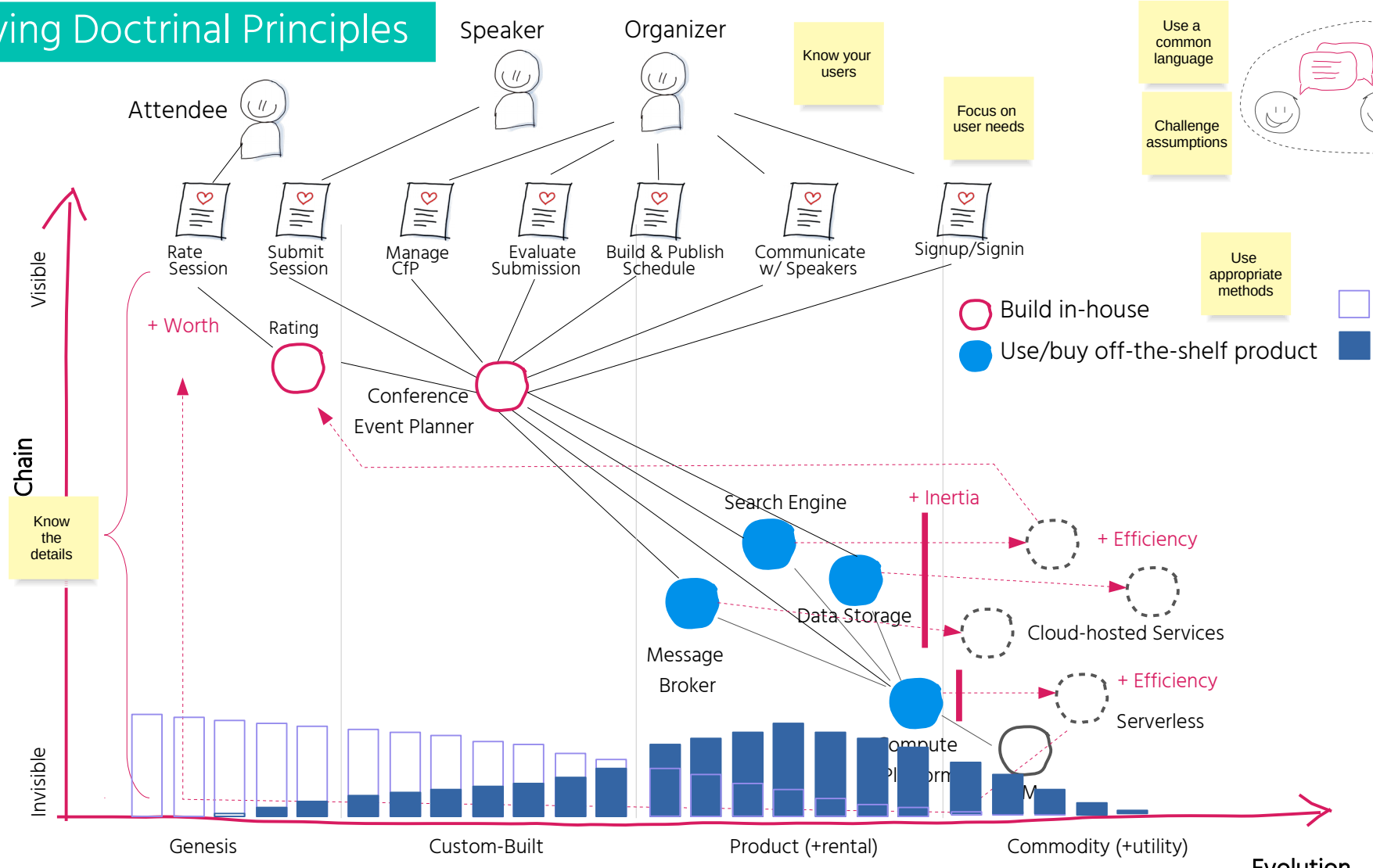


Applying Doctrinal Principles

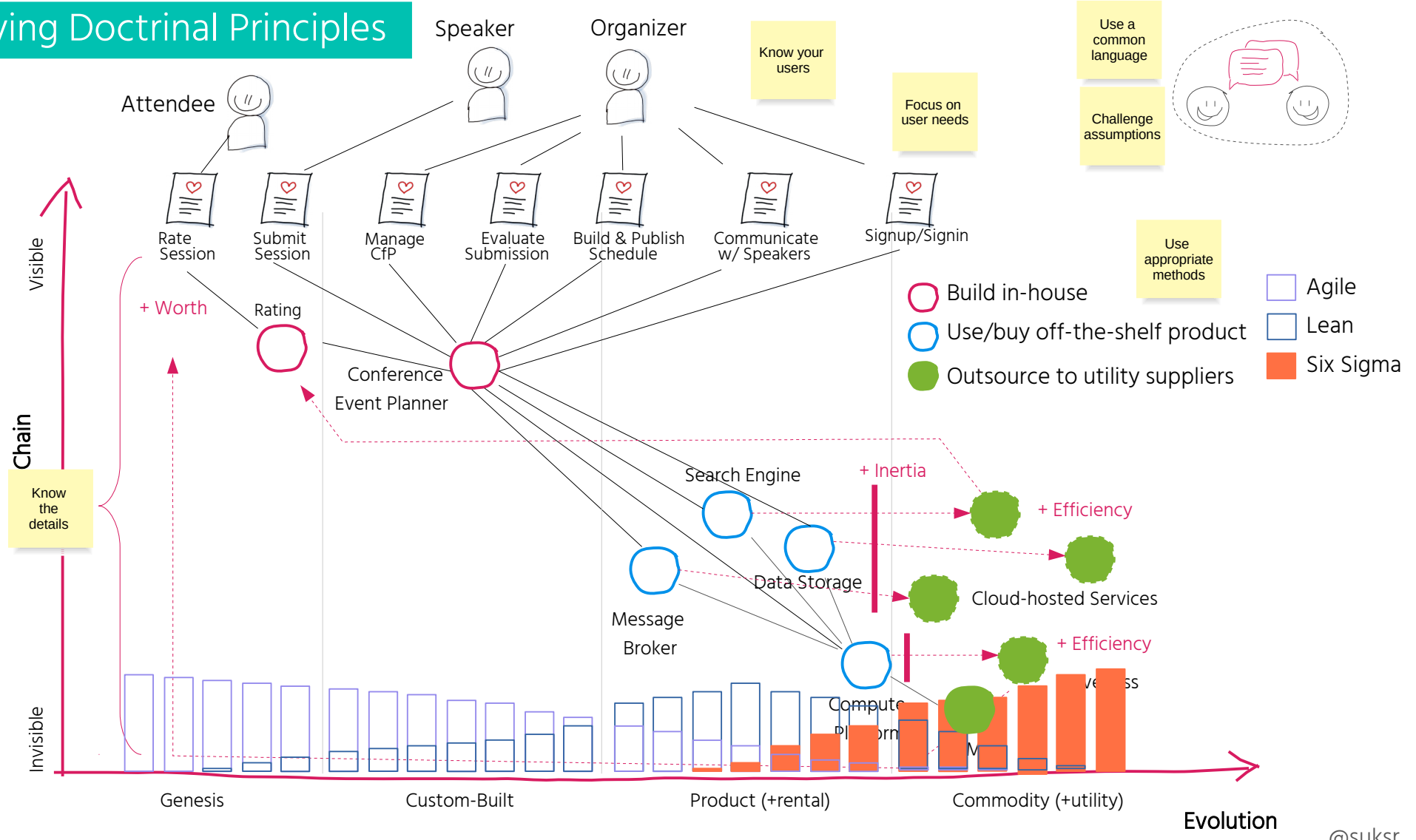


@suksr

Applying Doctrinal Principles

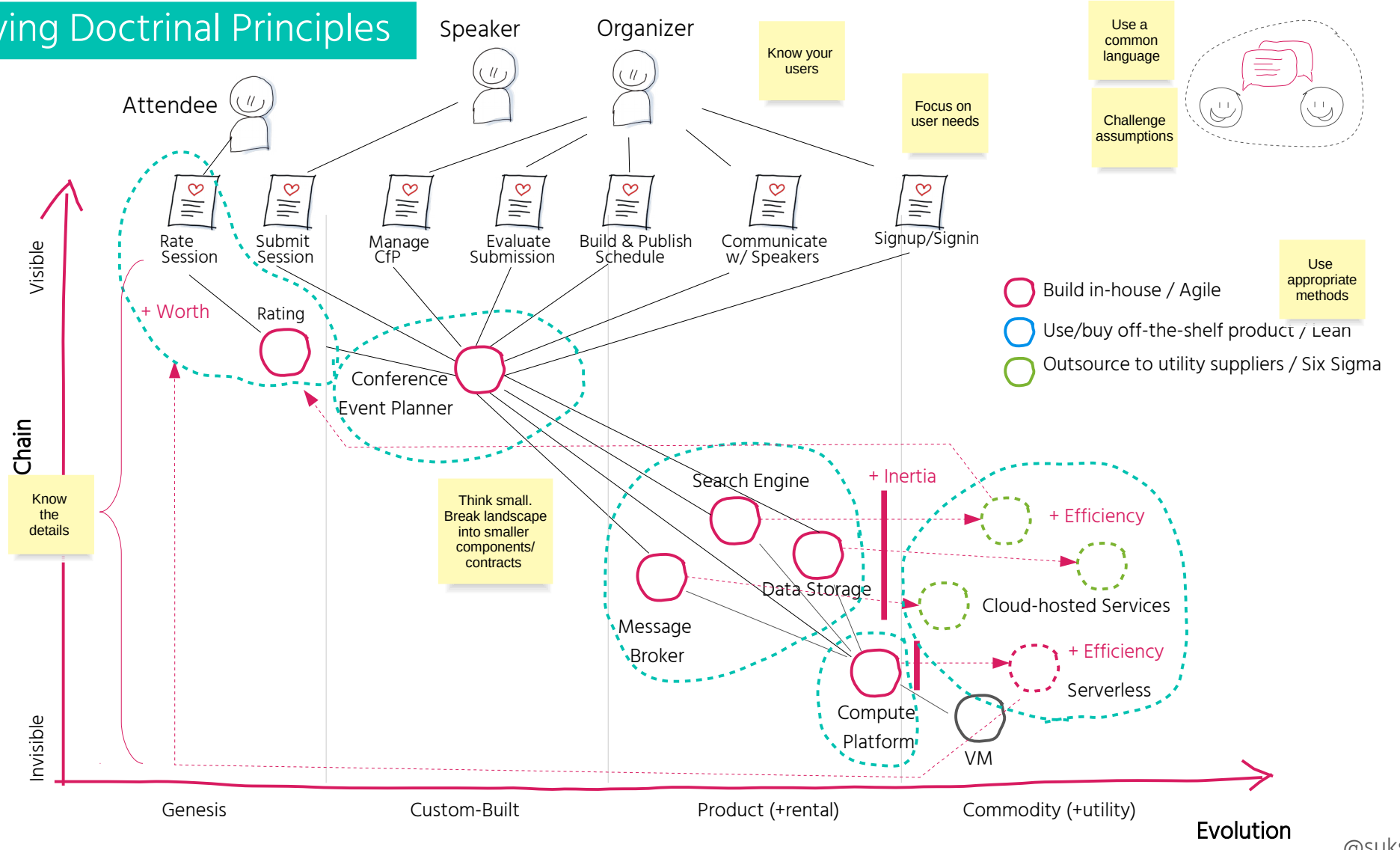


Applying Doctrinal Principles



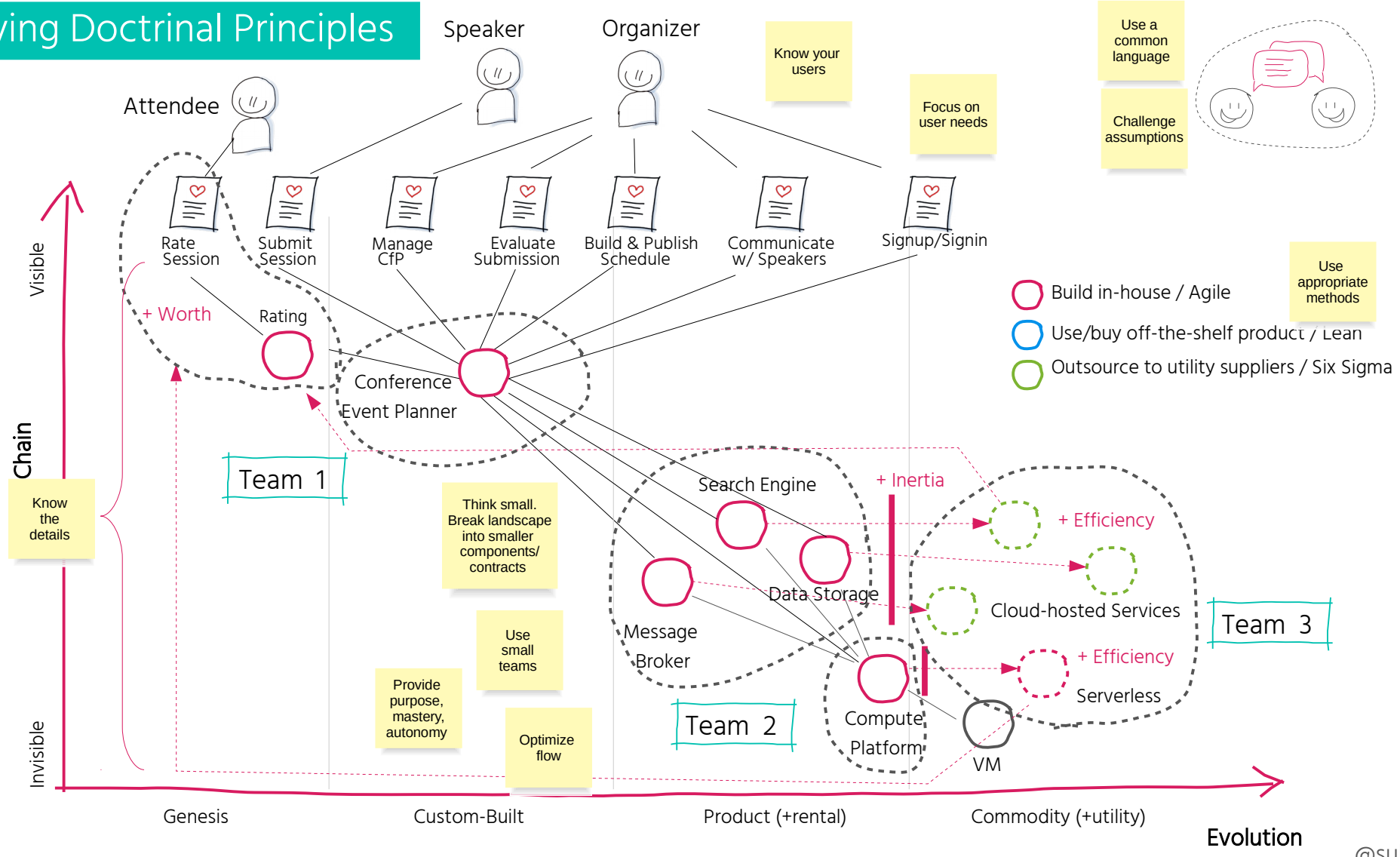
@suksr

Applying Doctrinal Principles



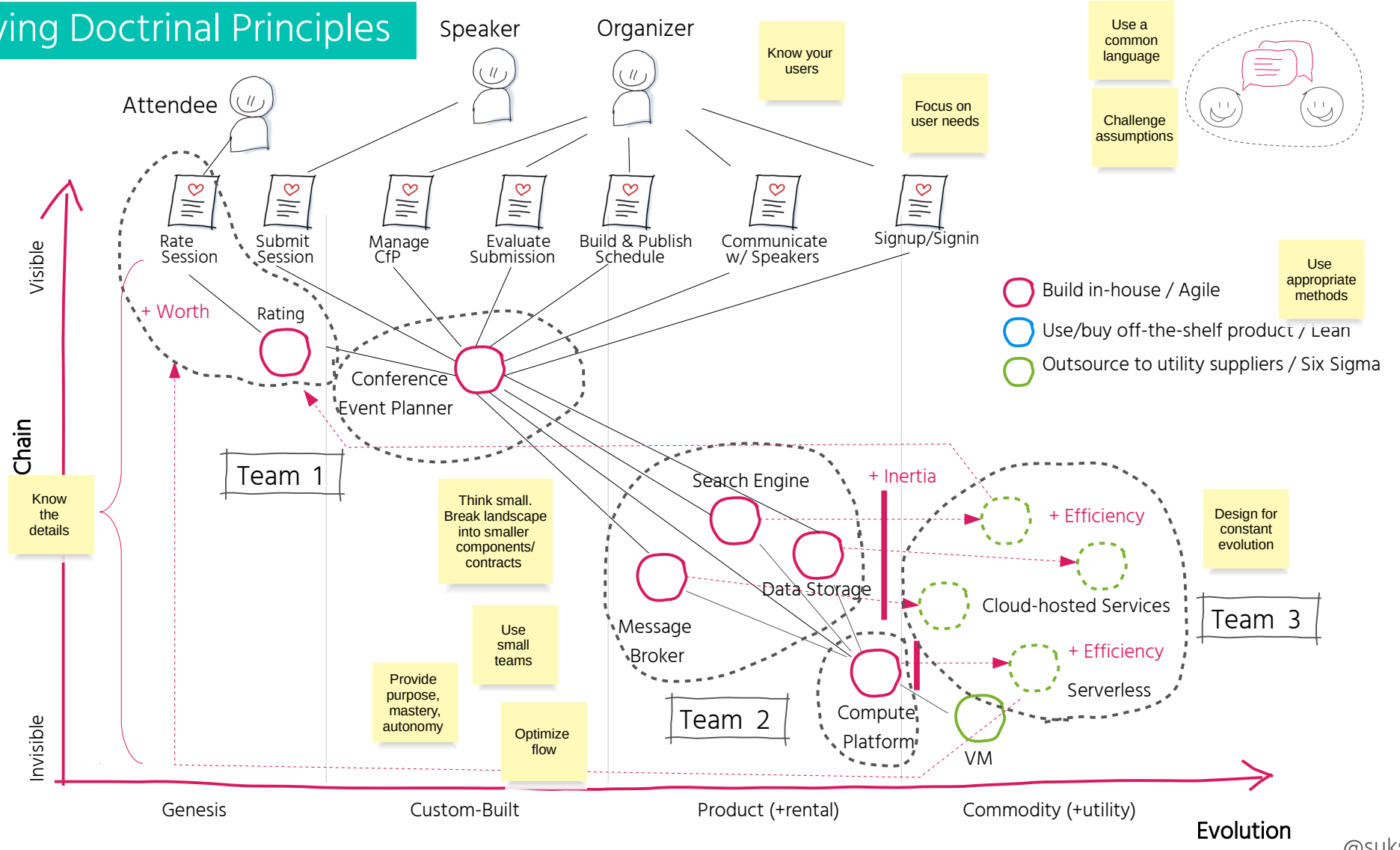
@suksr

Applying Doctrinal Principles



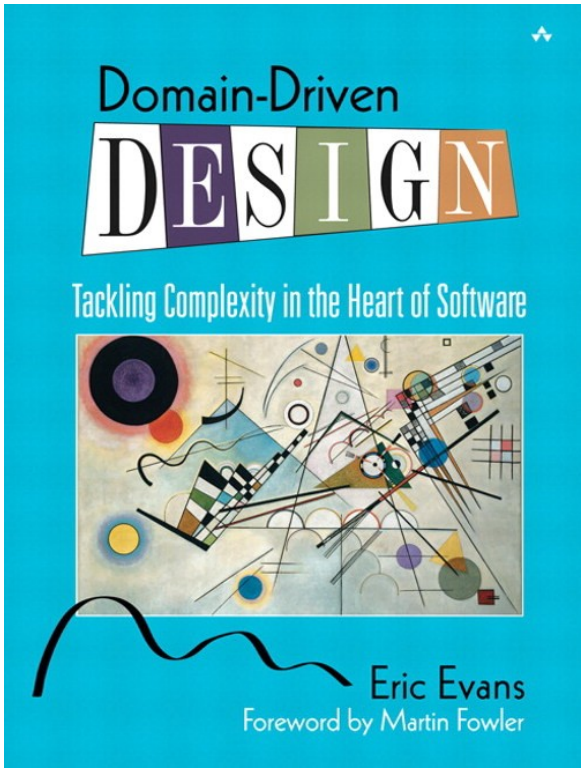
@suksr

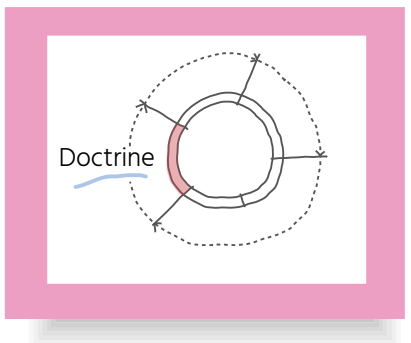
Applying Doctrinal Principles

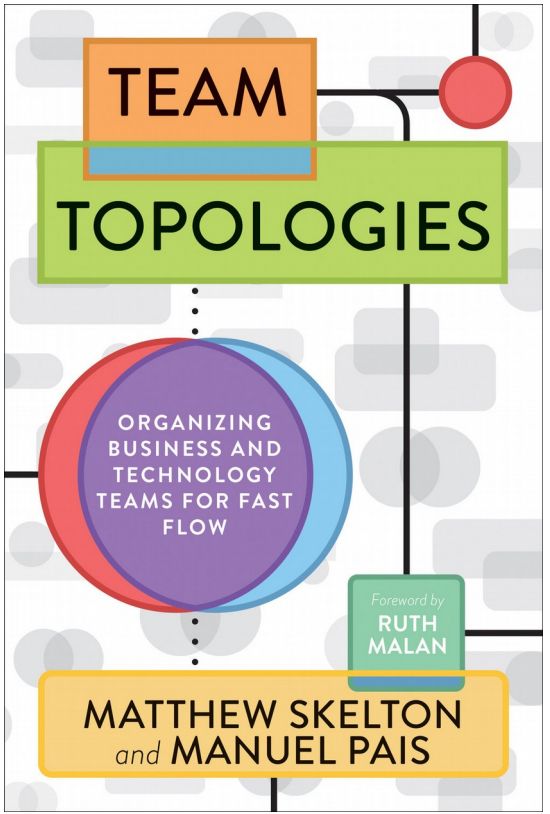


@suksr

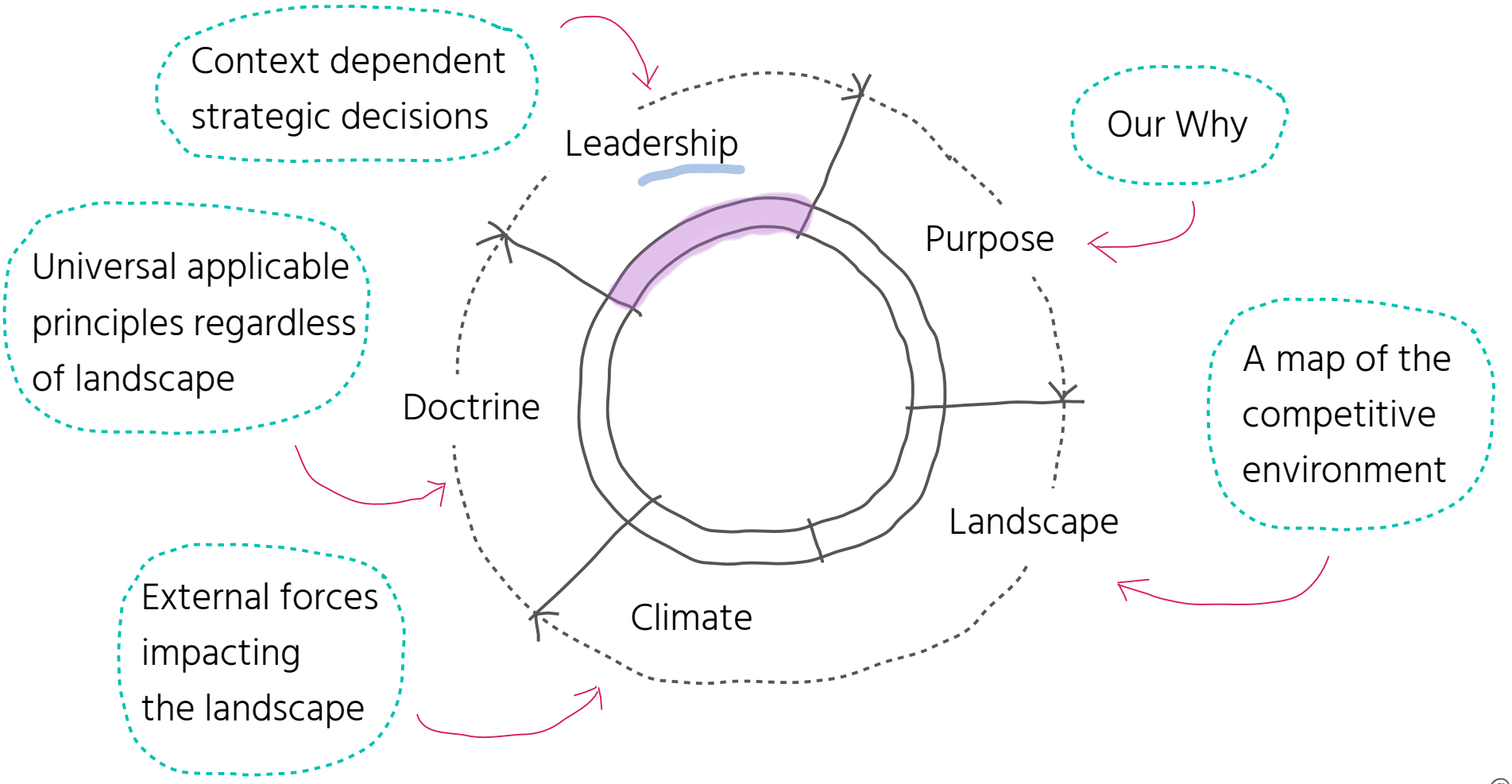
Applying Doctrinal Principles

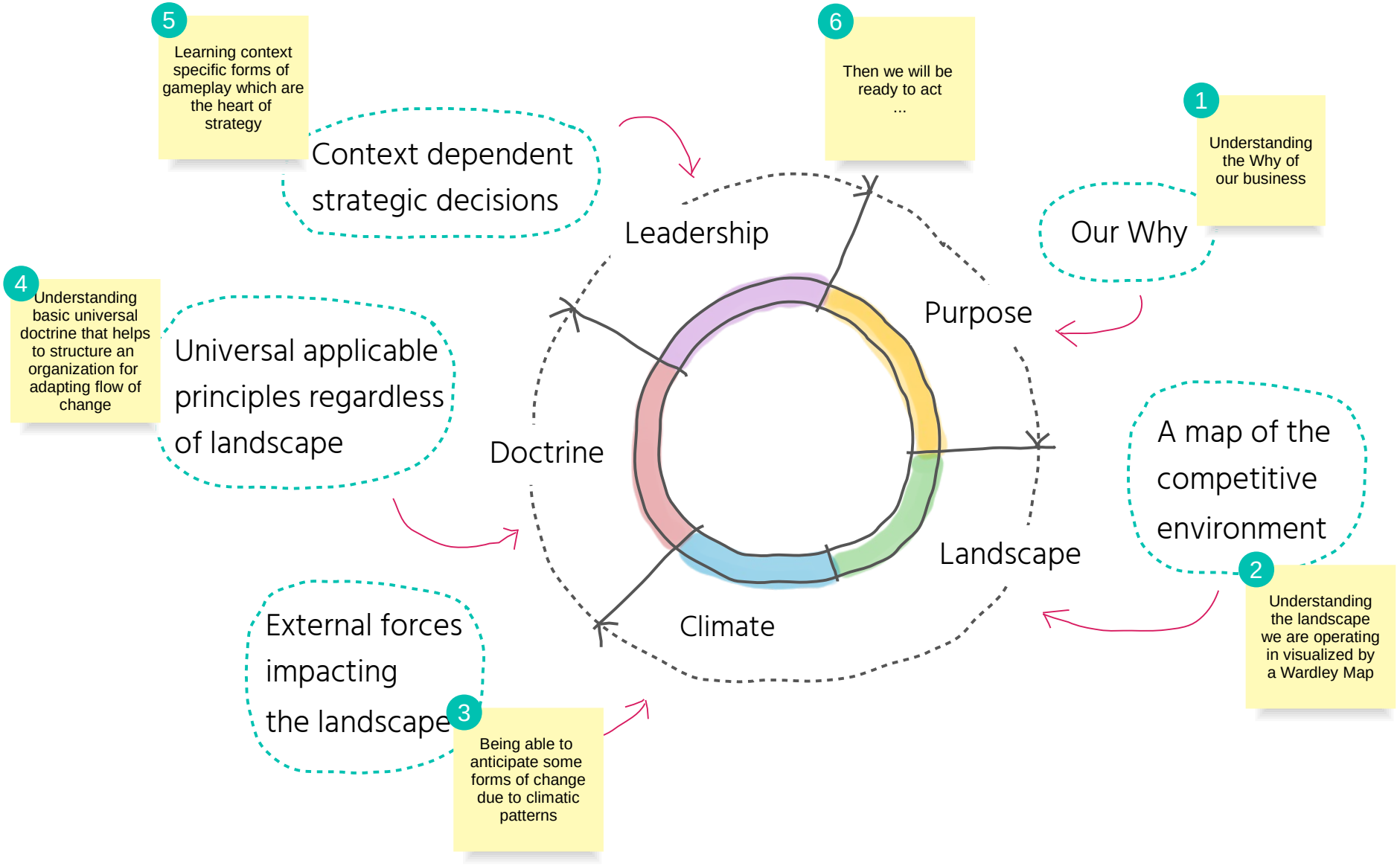






The Strategy Cycle of Wardley Mapping



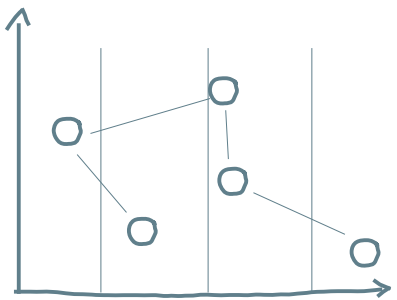

5
Learning context specific forms of gameplay which are the heart of strategy
Context dependent strategic decisions
6
Then we will be ready to act ...
1
Understanding the Why of our business
Our Why
Leadership
Purpose
4
Understanding basic universal doctrine that helps to structure an organization for adapting flow of change
Universal applicable principles regardless of landscape
Doctrine
A map of the competitive environment
Landscape
2
Understanding the landscape we are operating in visualized by a Wardley Map
External forces impacting the landscape
3
Being able to anticipate some forms of change due to climatic patterns
Climate

@suksr

3 Perspectives to Build Adaptive Systems

1.

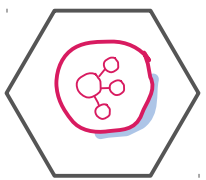
Business-Strategy



w/ Wardley Maps

2.

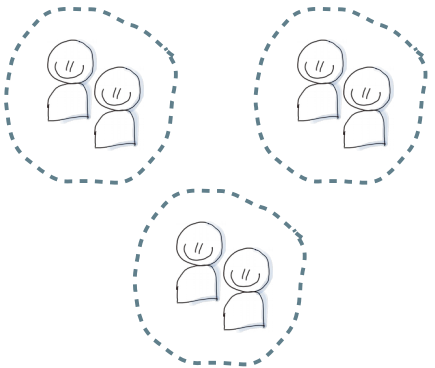
Software-Design/
-Architecture



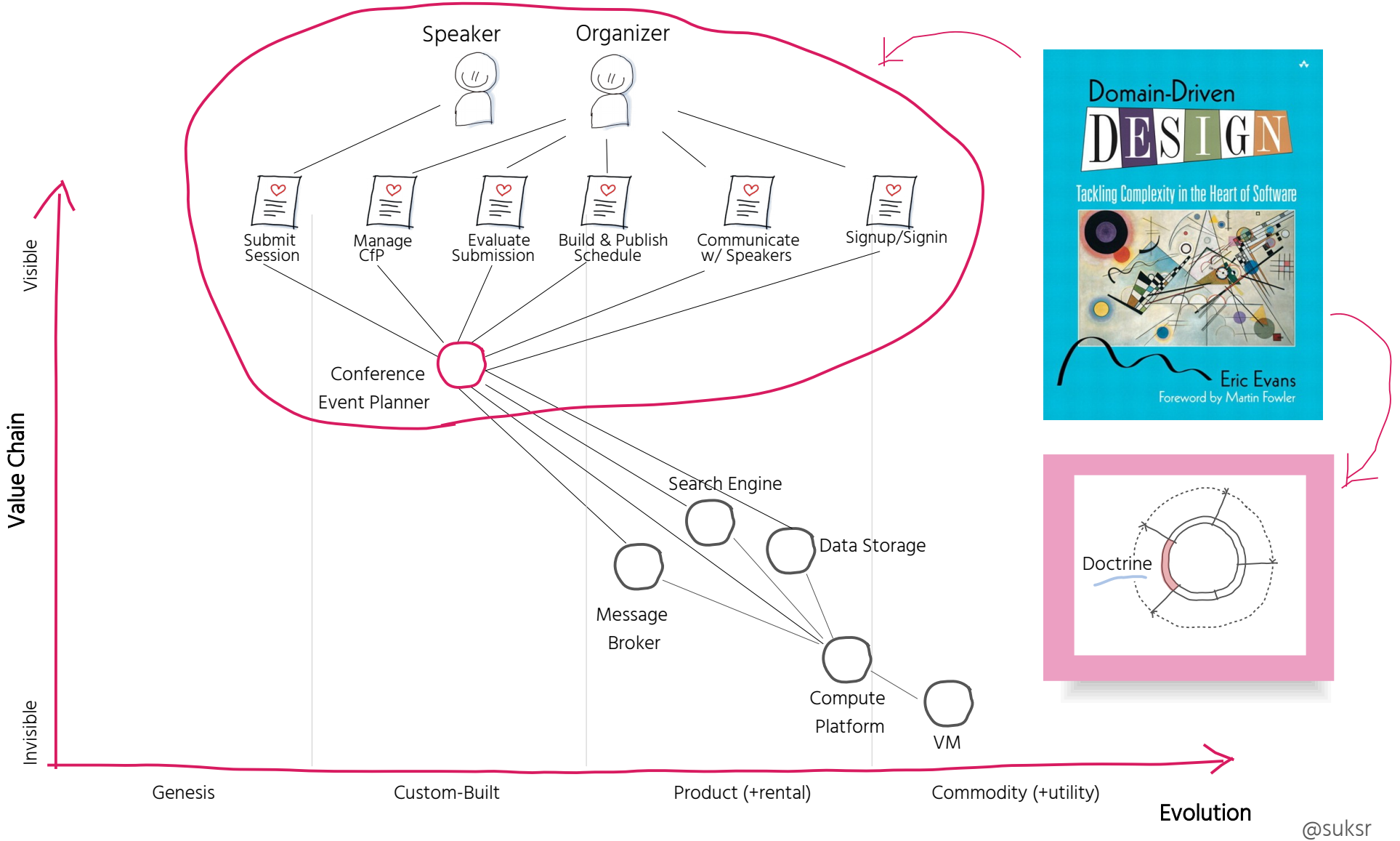
w/ Domain-Driven Design

3.

Team-Organization



w/ Team Topologies

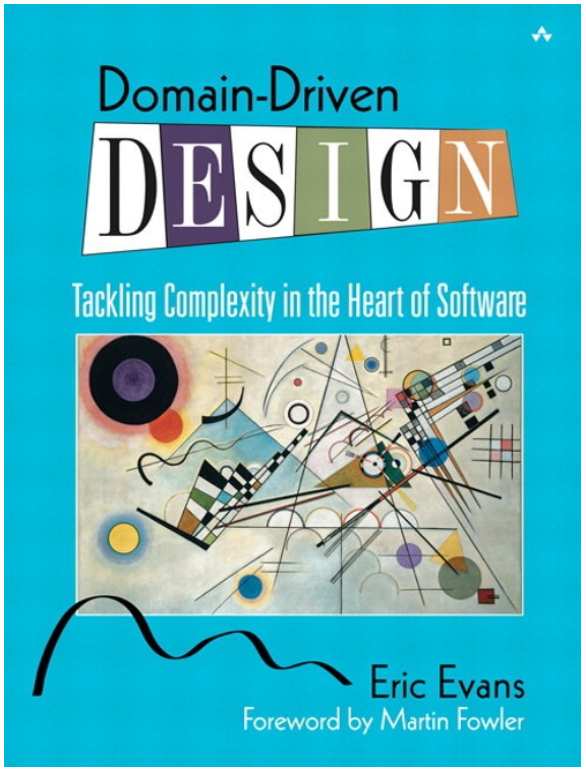

Speaker
Organizer
Submit Session
Manage CfP
Evaluate Submission
Build & Publish Schedule
Communicate w/ Speakers
Signup/Signin
Conference Event Planner
Search Engine
Data Storage
Message Broker
Compute Platform
VM
Visible
Invisible
Value Chain
Genesis
Custom-Built
Product (+rental)
Commodity (+utility)
Evolution

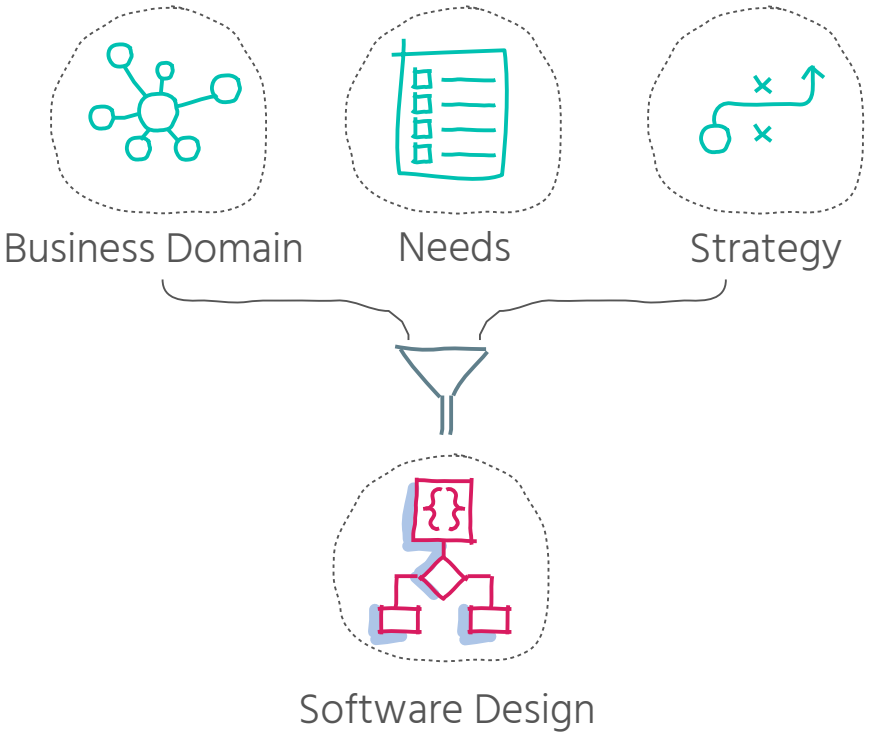
Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler

Doctrine

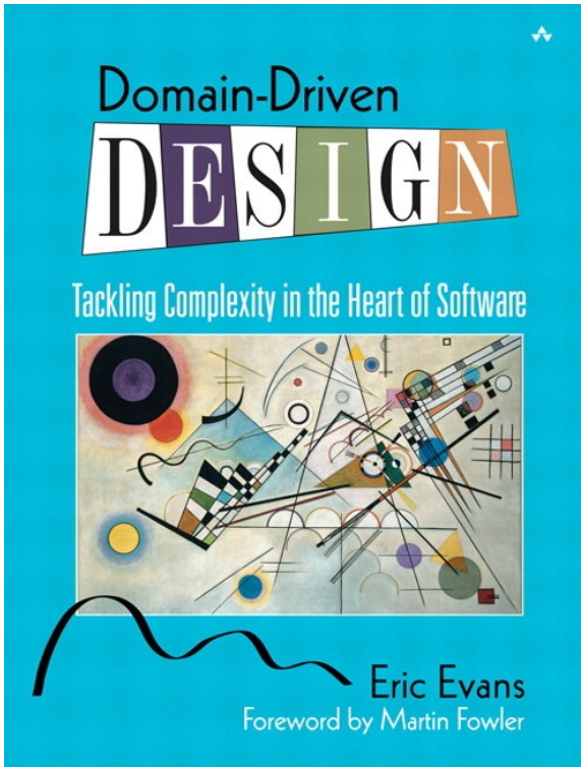
@suksr

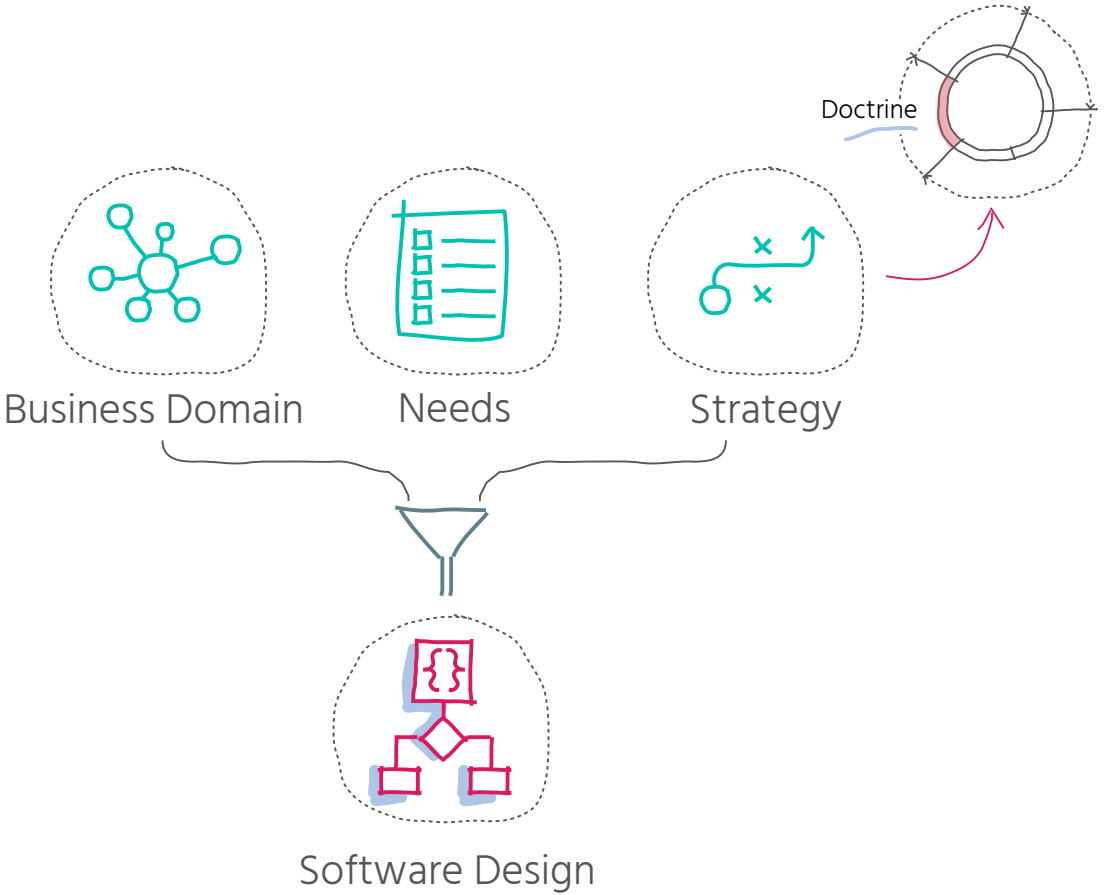
Domain-Driven Design (DDD)



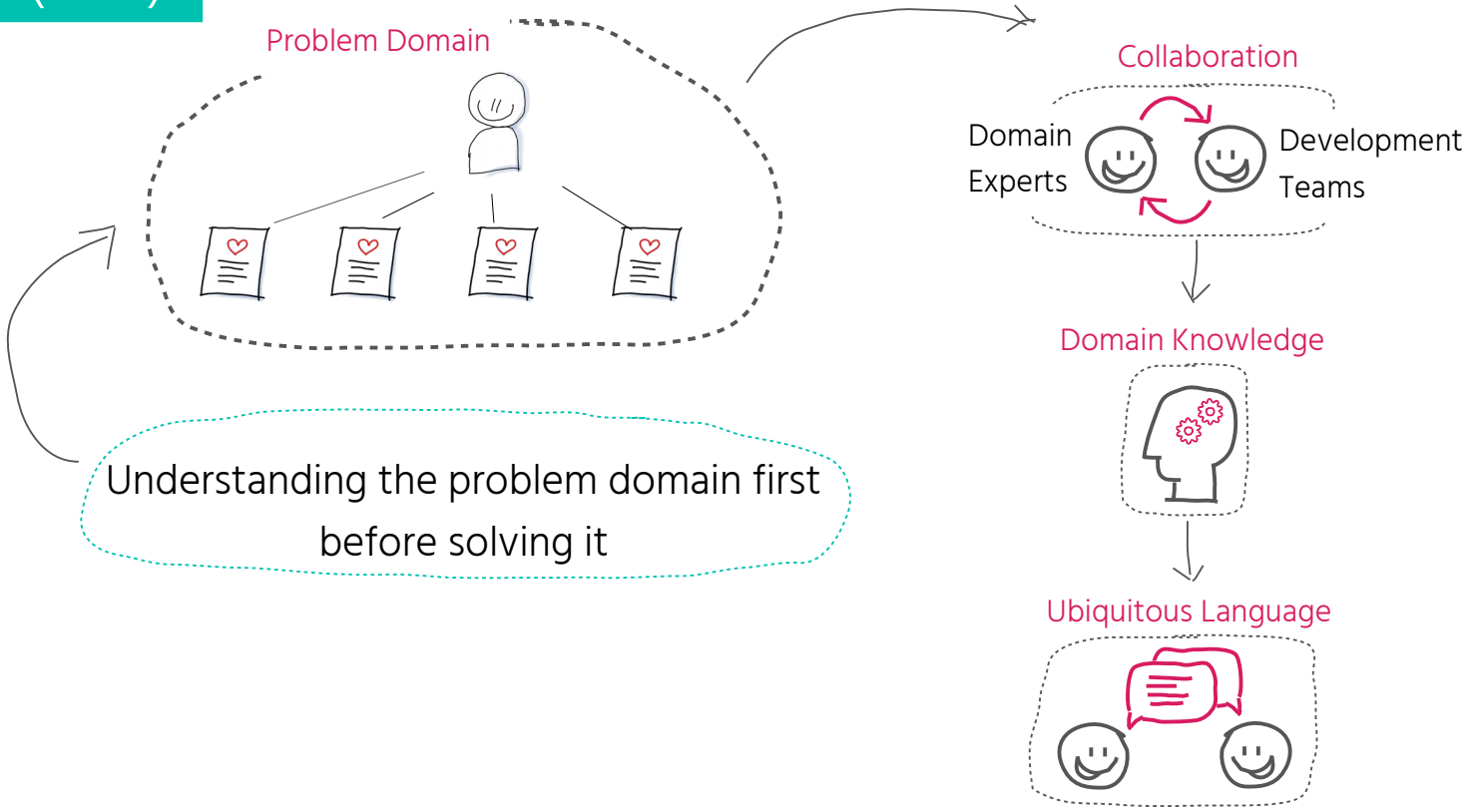


Domain-Driven Design (DDD)



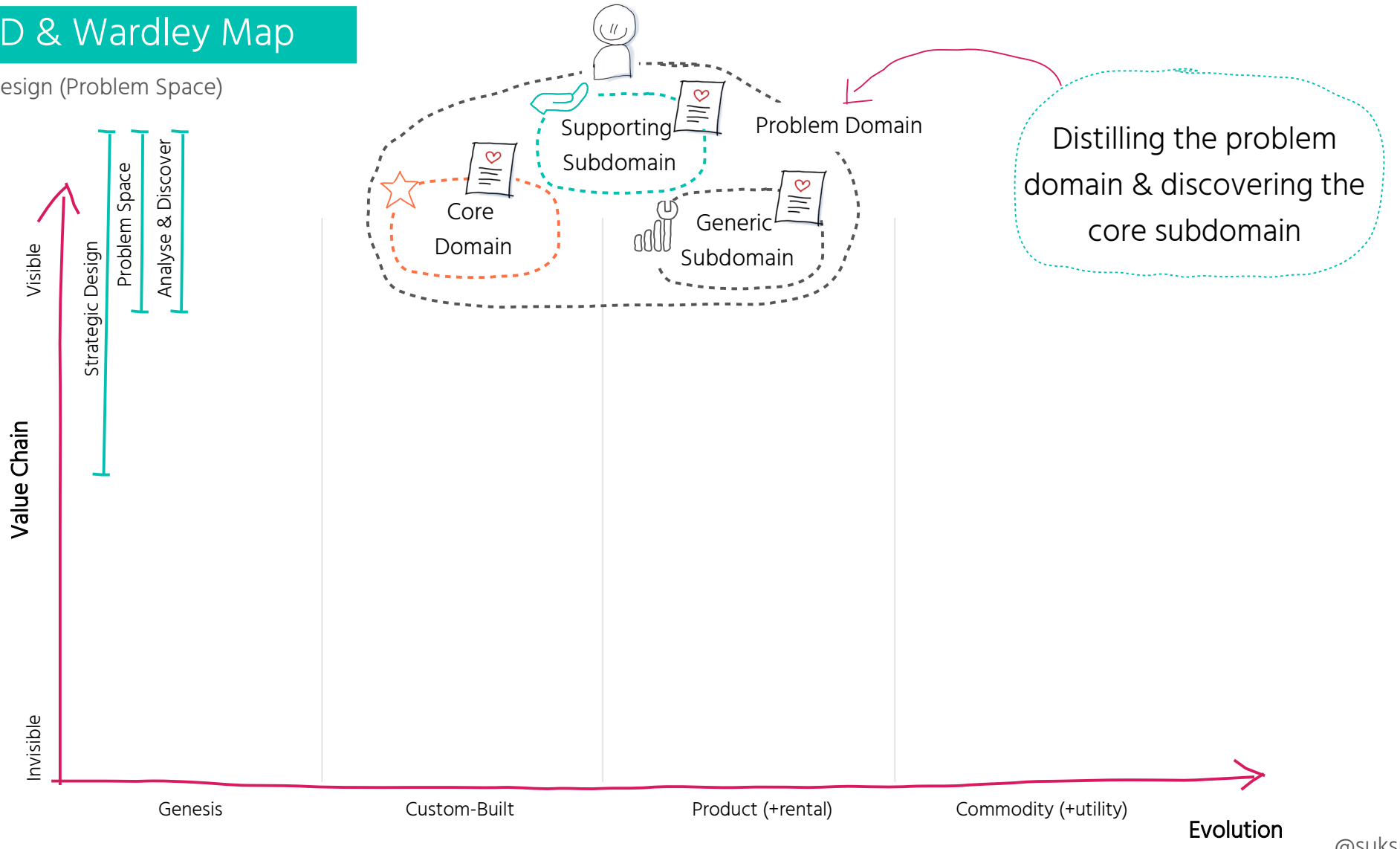


Domain-Driven Design (DDD)



DDD & Wardley Map

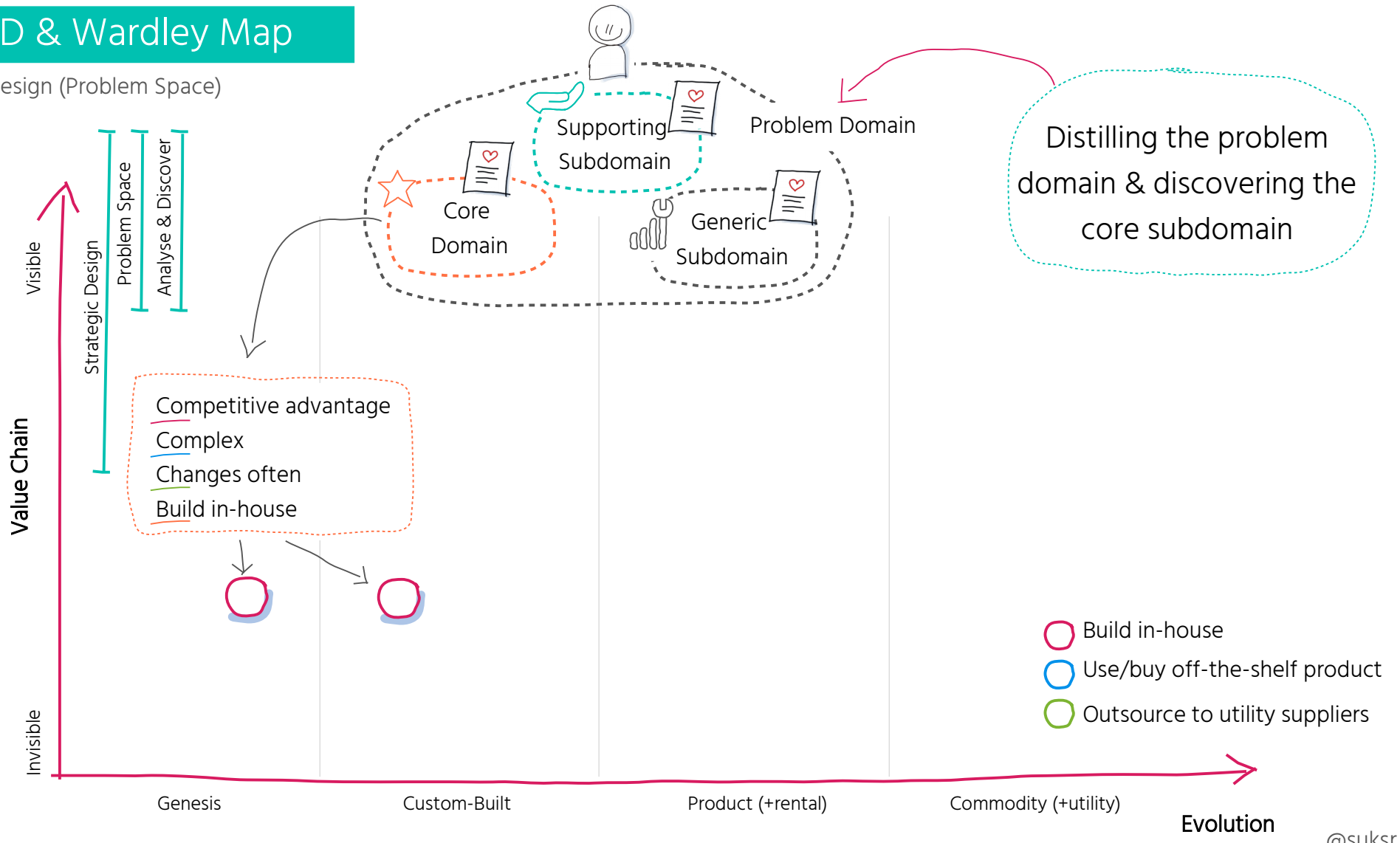
Strategic Design (Problem Space)



@suksr

DDD & Wardley Map

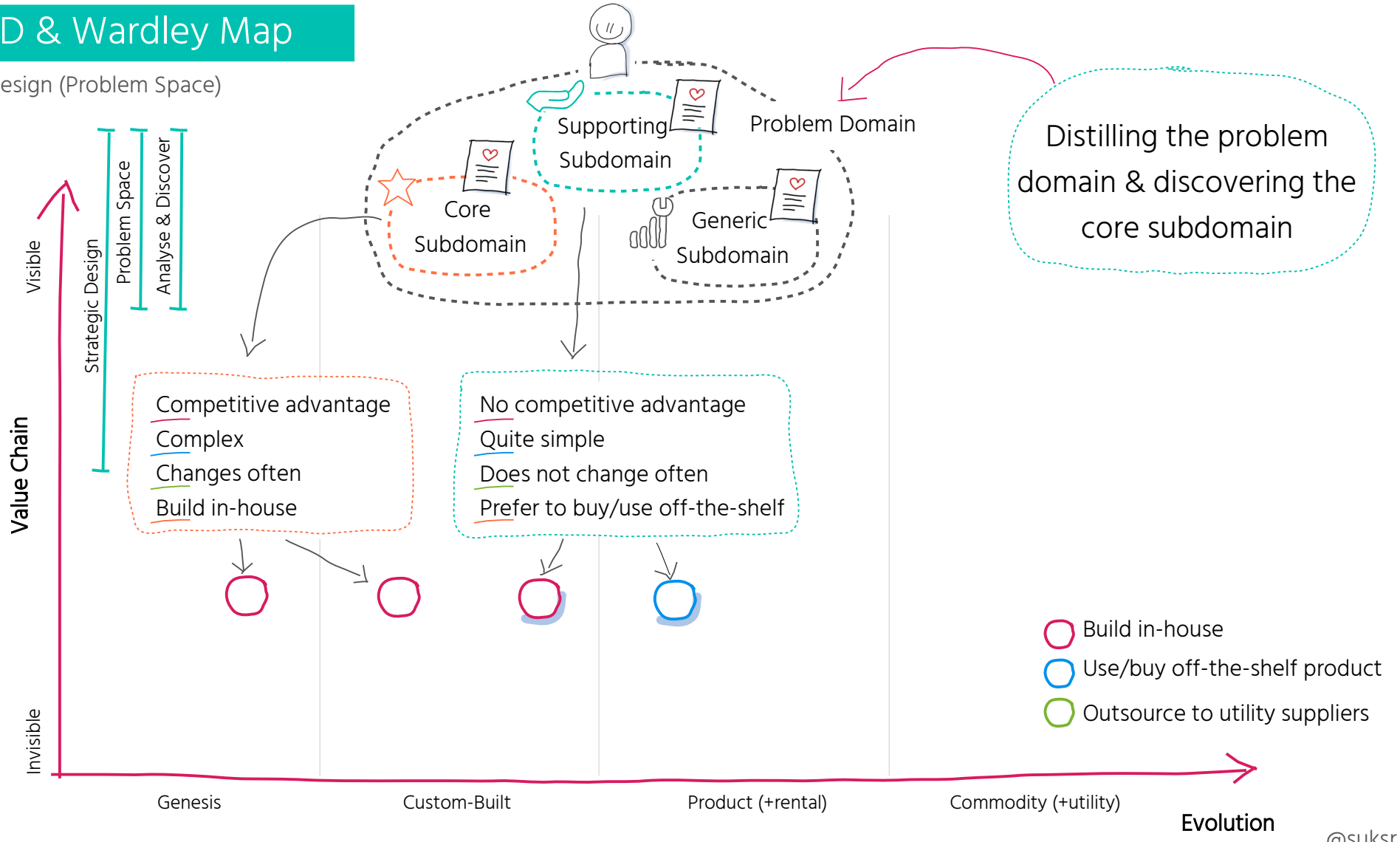
Strategic Design (Problem Space)



@suksr

DDD & Wardley Map

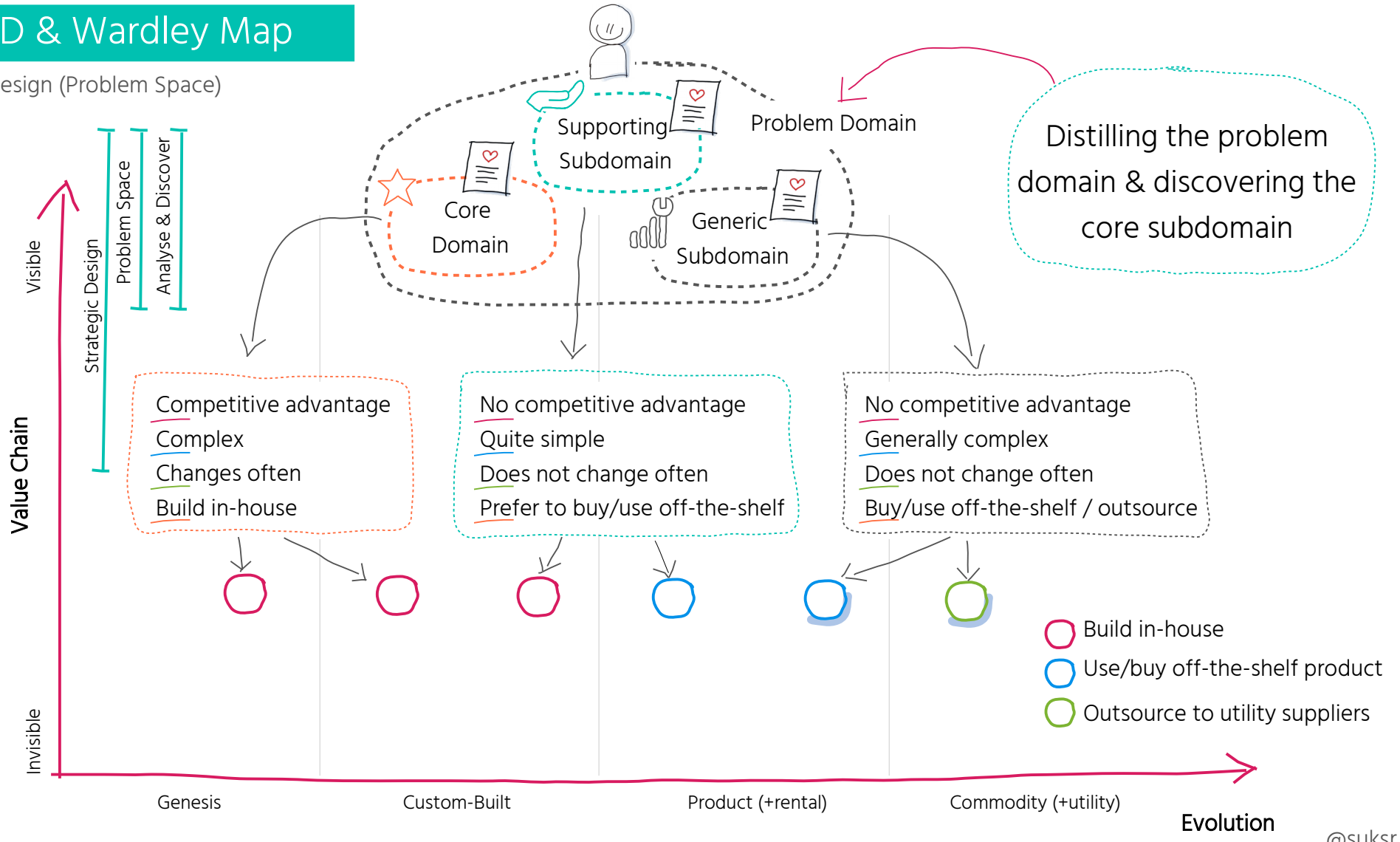
Strategic Design (Problem Space)



@suksr

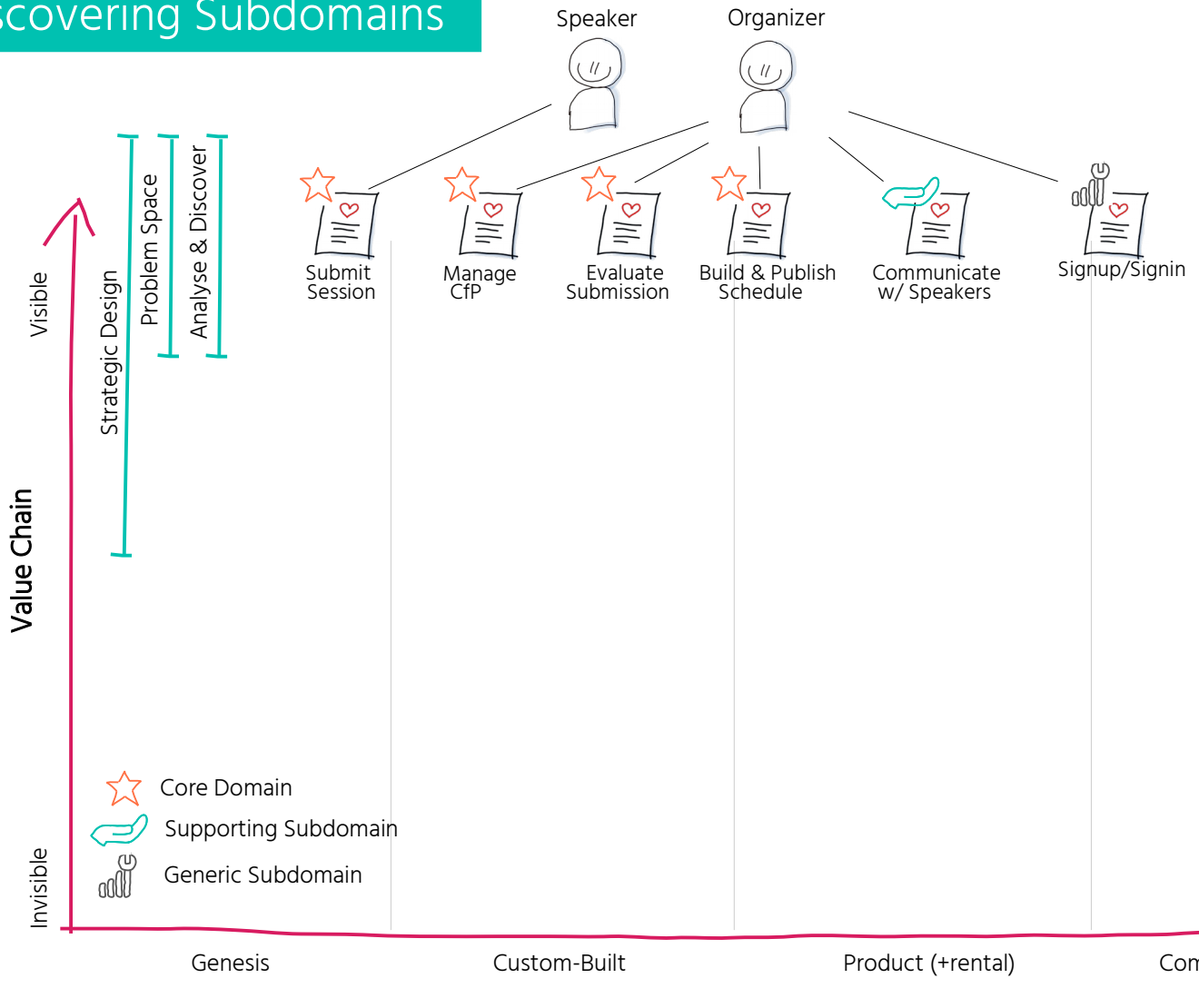
DDD & Wardley Map

Strategic Design (Problem Space)



@suksr

Discovering Subdomains



Is this subdomain providing competitive advantage?

Would buying/using off-the-shelf products or outsource to utility suppliers jeopardize the business success?

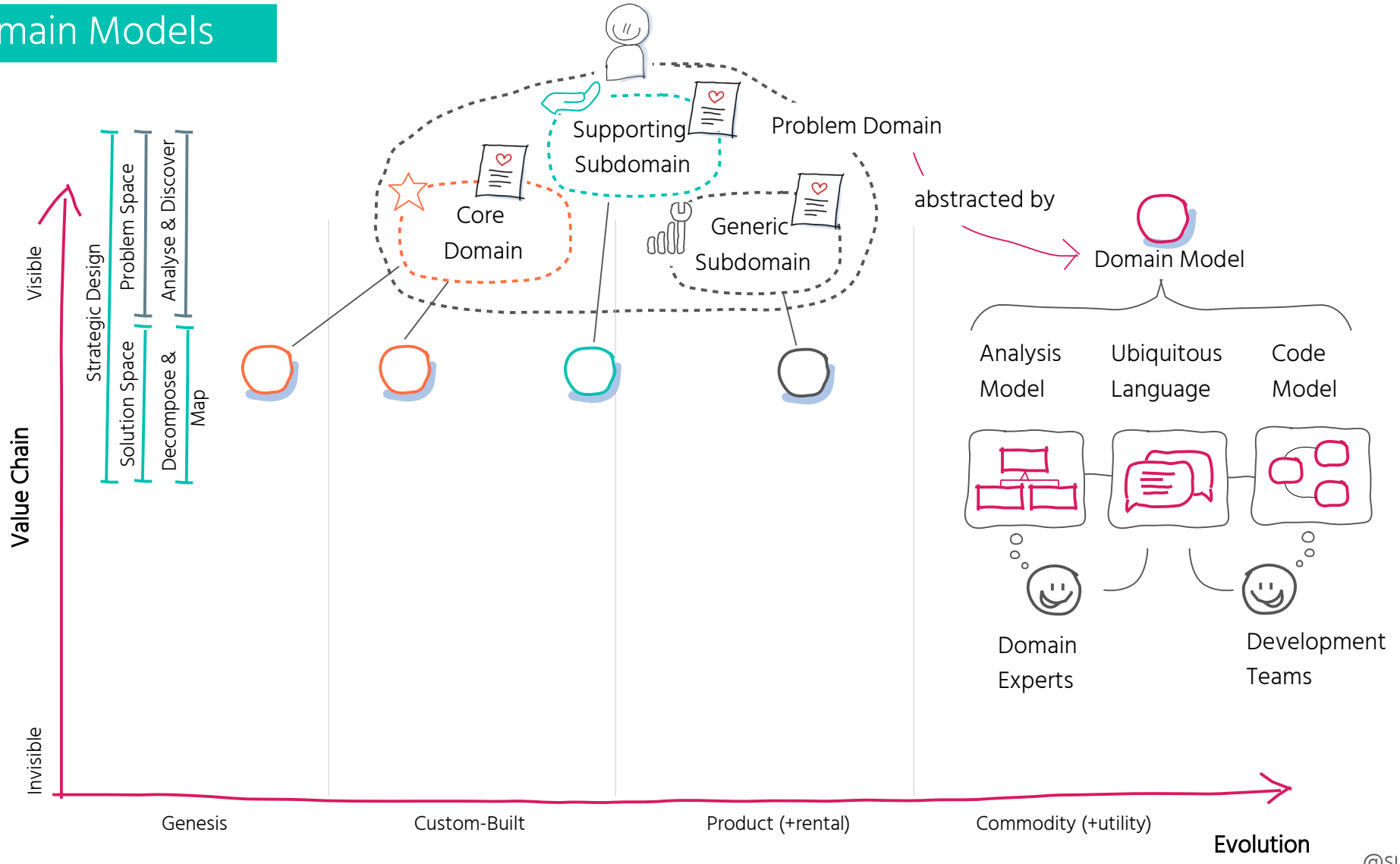
Do many business systems need this subdomain?

Is this subdomain worth custom-building to be able to deliver a specialized solution creating competive difference?

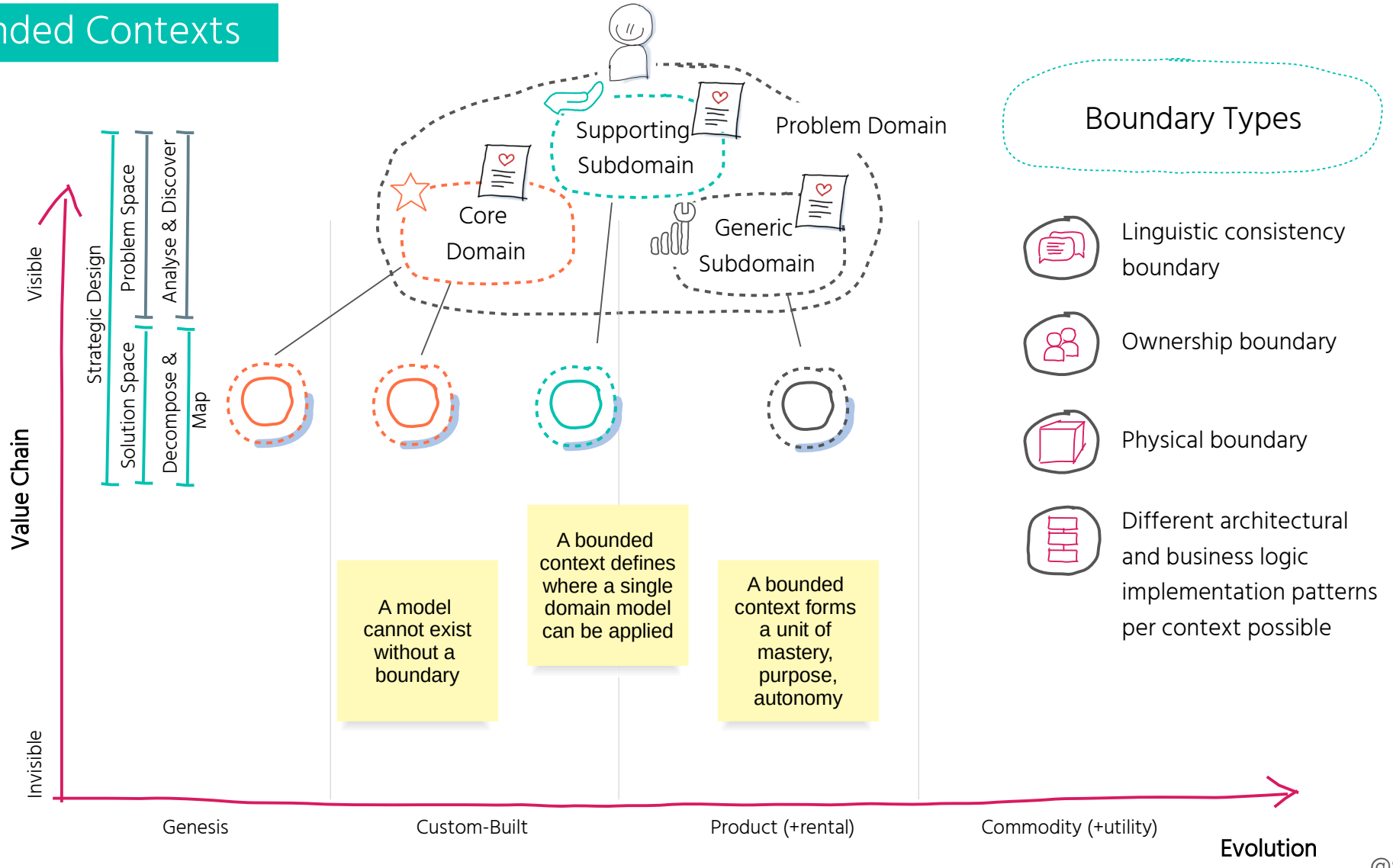
Do their business logic go beyond simple CRUD solutions and simple input validation?

@suksr

Domain Models



Bounded Contexts

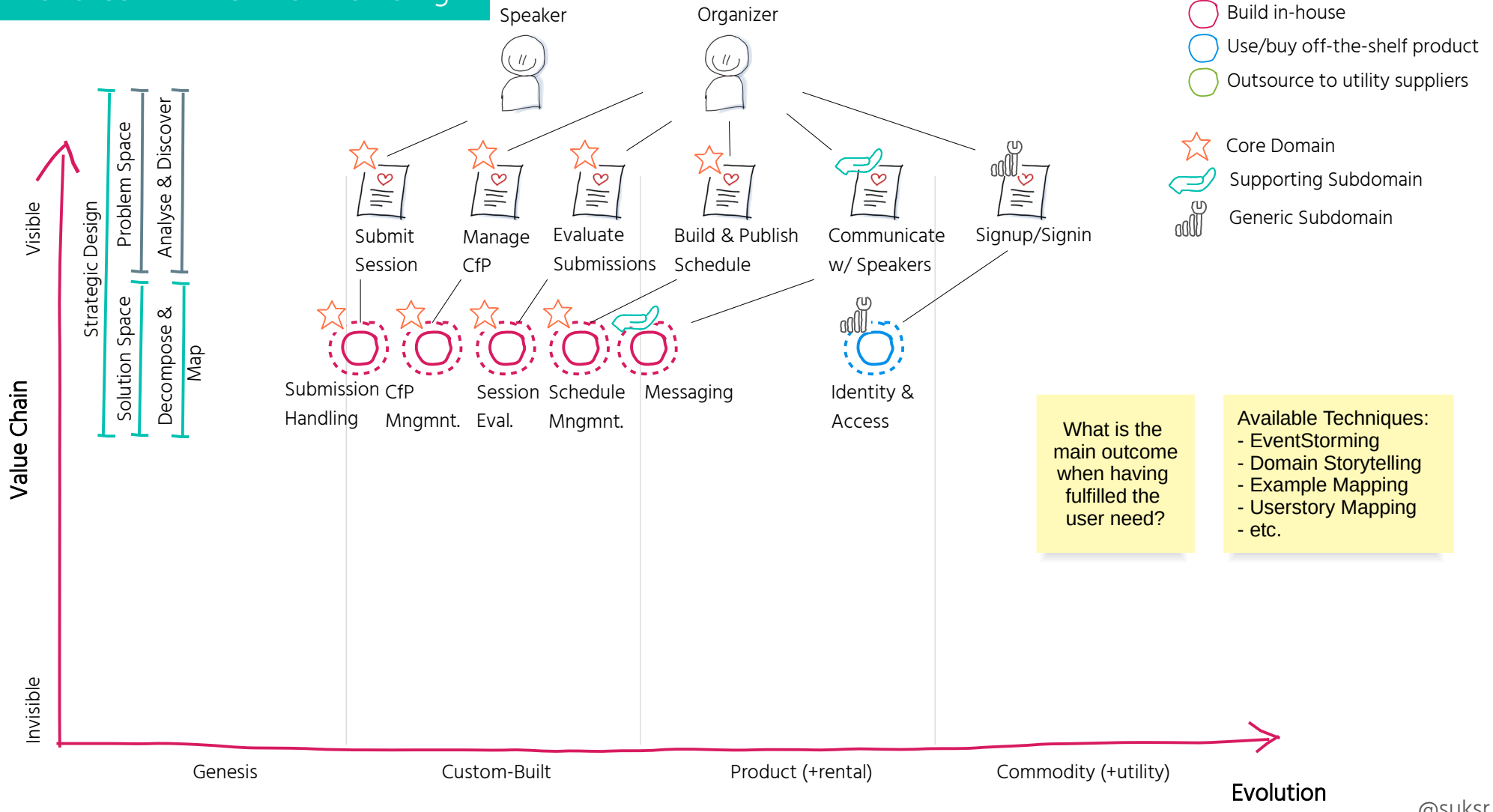


Boundary Types

- Linguistic consistency boundary
- Ownership boundary
- Physical boundary
- Different architectural and business logic implementation patterns per context possible

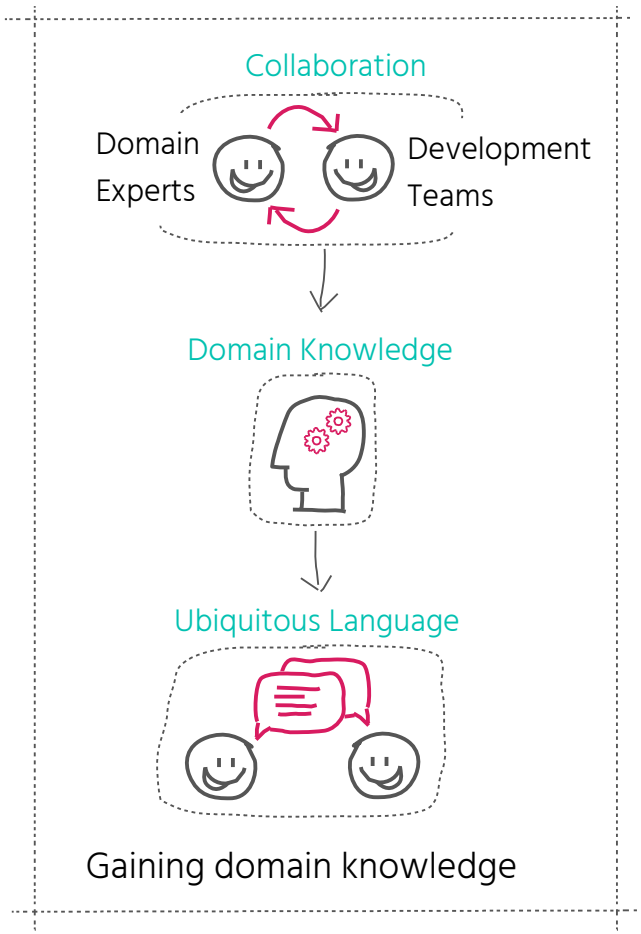
@suksr

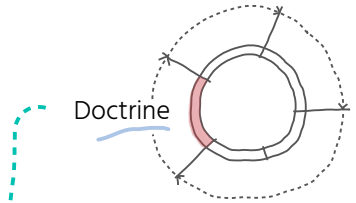
Bounded Contexts & Evolution Stages



@suksr

Strategic DDD & Doctrine





Challenge assumptions

Know the details

Use a common language

Strategic DDD & Doctrine

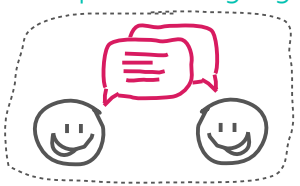
Collaboration

Domain Experts

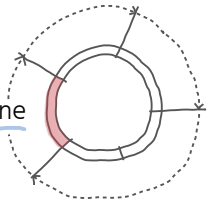
Development Teams

Domain Knowledge

Ubiquitous Language



Gaining domain knowledge



Challenge assumptions

Know the details

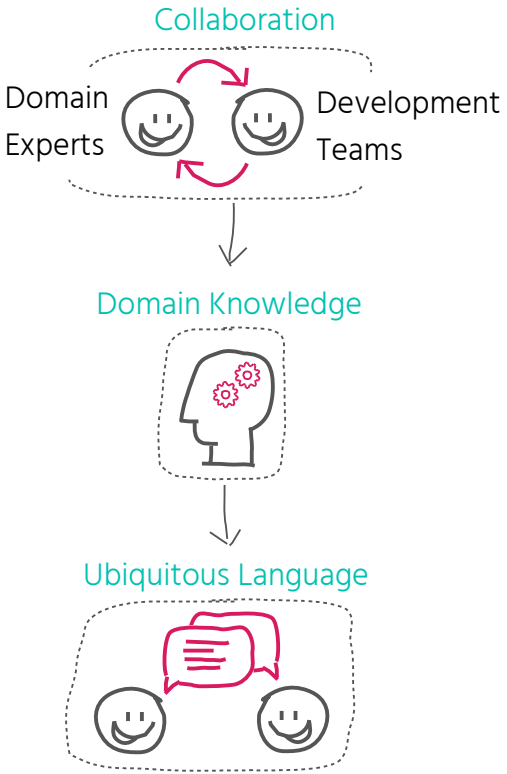
Use a common language

Focus on high situational awareness

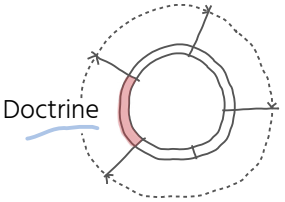
Core Domain

Discovering the core domain

Strategic DDD & Doctrine



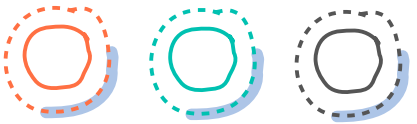
Gaining domain knowledge



- Challenge assumptions
- Know the details
- Use a common language
- Focus on high situational awareness
- Think small (as in contracts)
- Provide purpose, mastery, and autonomy

Discovering the core domain

Decomposing into modular components (Bounded Contexts)



Strategic DDD & Doctrine

Collaboration

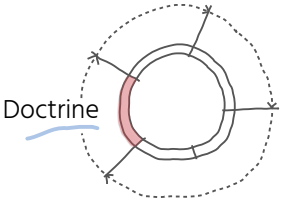
Domain Experts

Development Teams

Domain Knowledge

Ubiquitous Language

Gaining domain knowledge



Challenge assumptions

Know the details

Use a common language

Focus on high situational awareness

Think small (as in contracts)

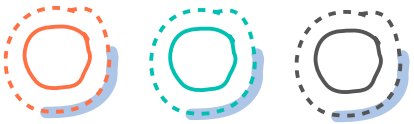
Provide purpose, mastery, and autonomy

Use appropriate methods

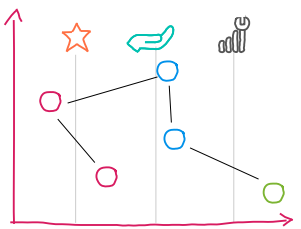
Core Domain

Discovering the core domain

Decomposing into modular components (Bounded Contexts)



Subdomain categories can be mapped to evolution stages



Conway's Law

"Any organization that designs a system [...] will produce a design whose structure is a copy of the organization's communication structure."

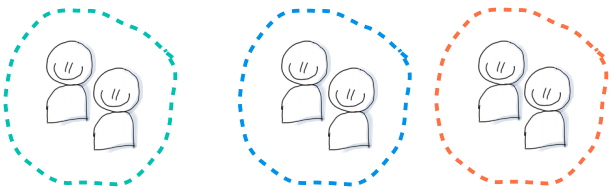
Melvin Conway

To optimize for flow of change requires ...

cross-functional, autonomous teams

no handover between teams

restricting communication between teams



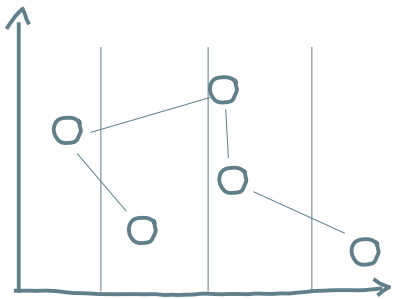
small, long-lived teams

minimizing cognitive load

team ownership

3 Perspectives to Build Adaptive Systems

1. Business-Strategy

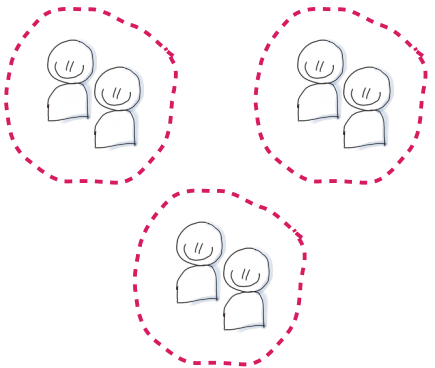


w/ Wardley Maps

2. Software-Design/
-Architecture

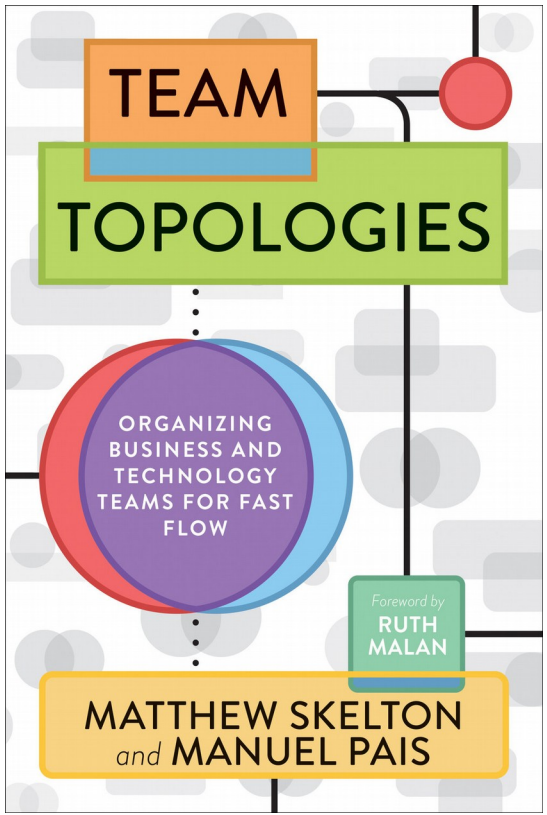
w/ Domain-Driven Design

3. Team-Organization



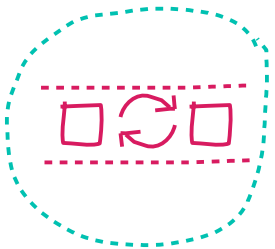
w/ Team Topologies

Team Topologies



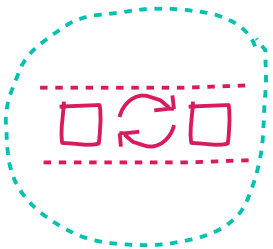
"Overall, the Team Topologies approach advocates for organization design that optimizes for flow of change and feedback from running systems."

Four Team Types



Stream-aligned team

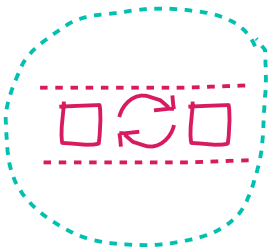
Four Team Types



Stream-aligned team

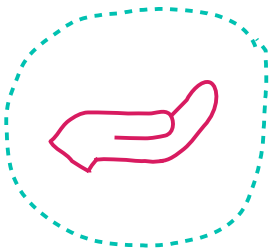
Platform team

Four Team Types



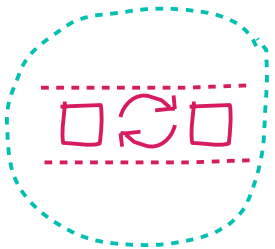
Stream-aligned team

Platform team



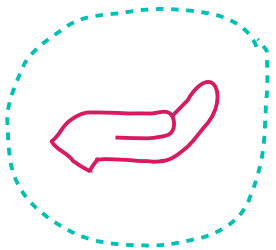
Enabling team

Four Team Types

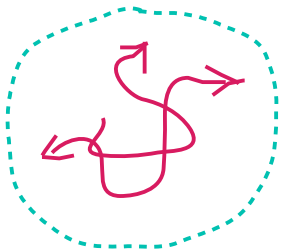


Stream-aligned team

Platform team



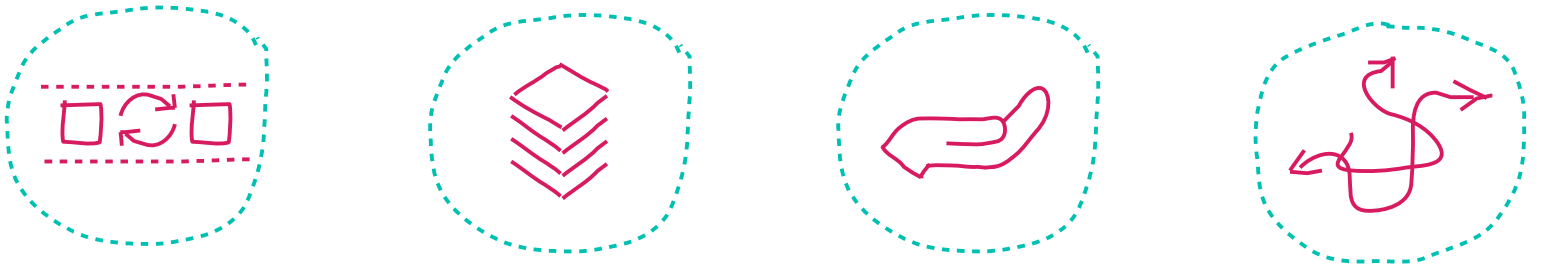
Enabling team



Complicated subsystem team

Four Team Types

Fast flow of change



Stream-aligned team

Platform team

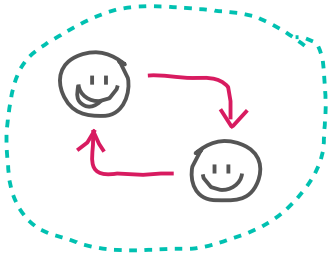
Enabling team

Complicated subsystem team

Increasing autonomy

Reducing cognitive load

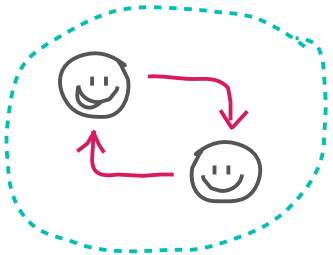
Three Interaction Modes



Collaboration

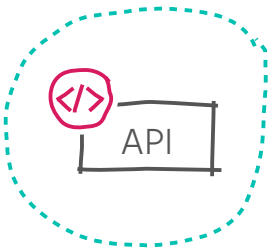
Rapid discovery

Three Interaction Modes



Collaboration

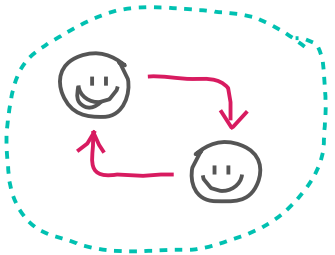
Rapid discovery



X-as-a-Service

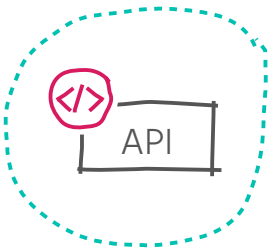
Predictable delivery

Three Interaction Modes



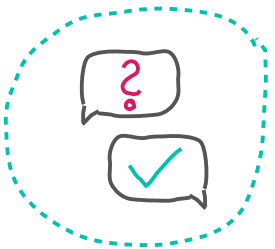
Collaboration

Rapid discovery



X-as-a-Service

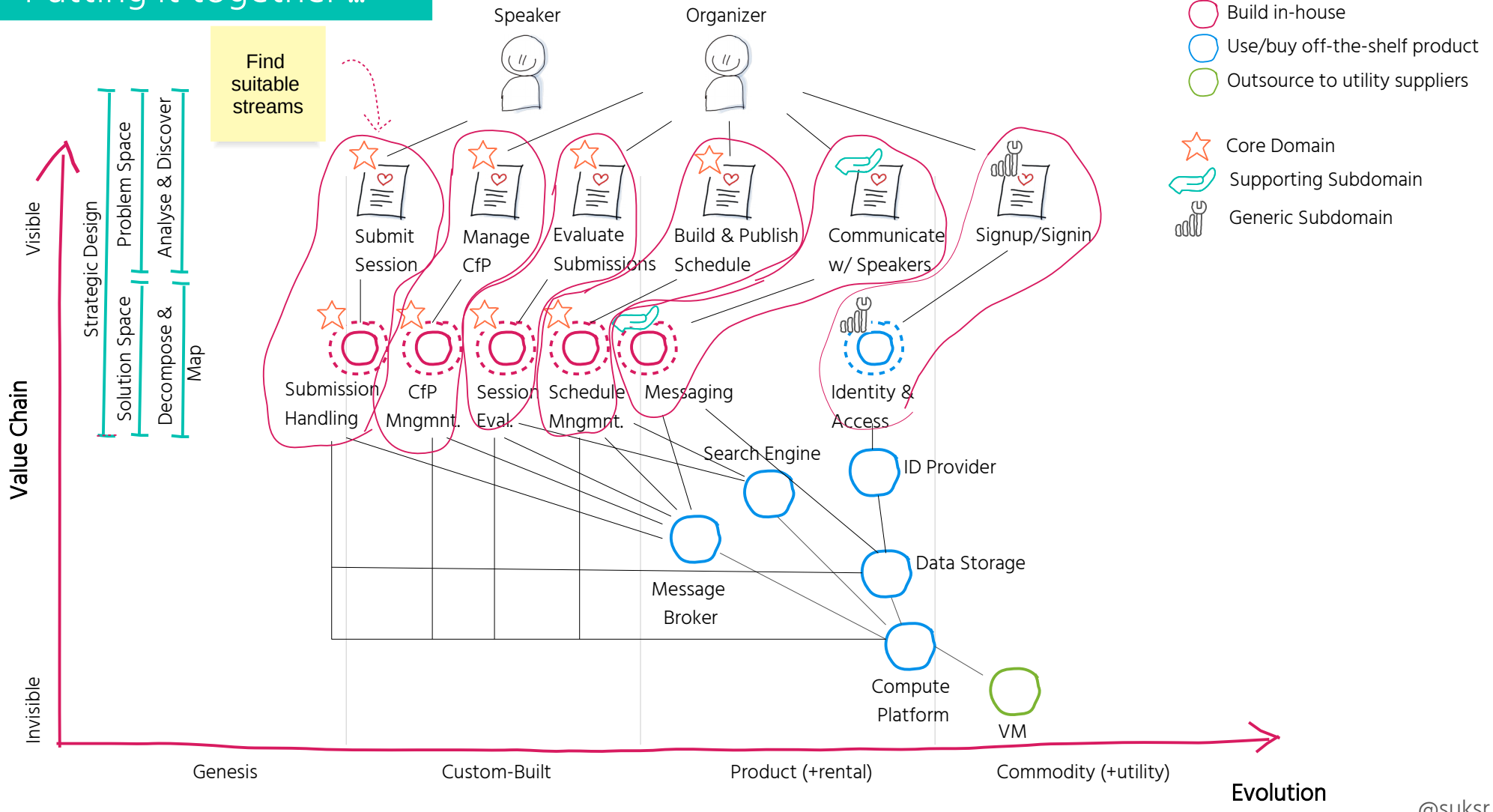
Predictable delivery



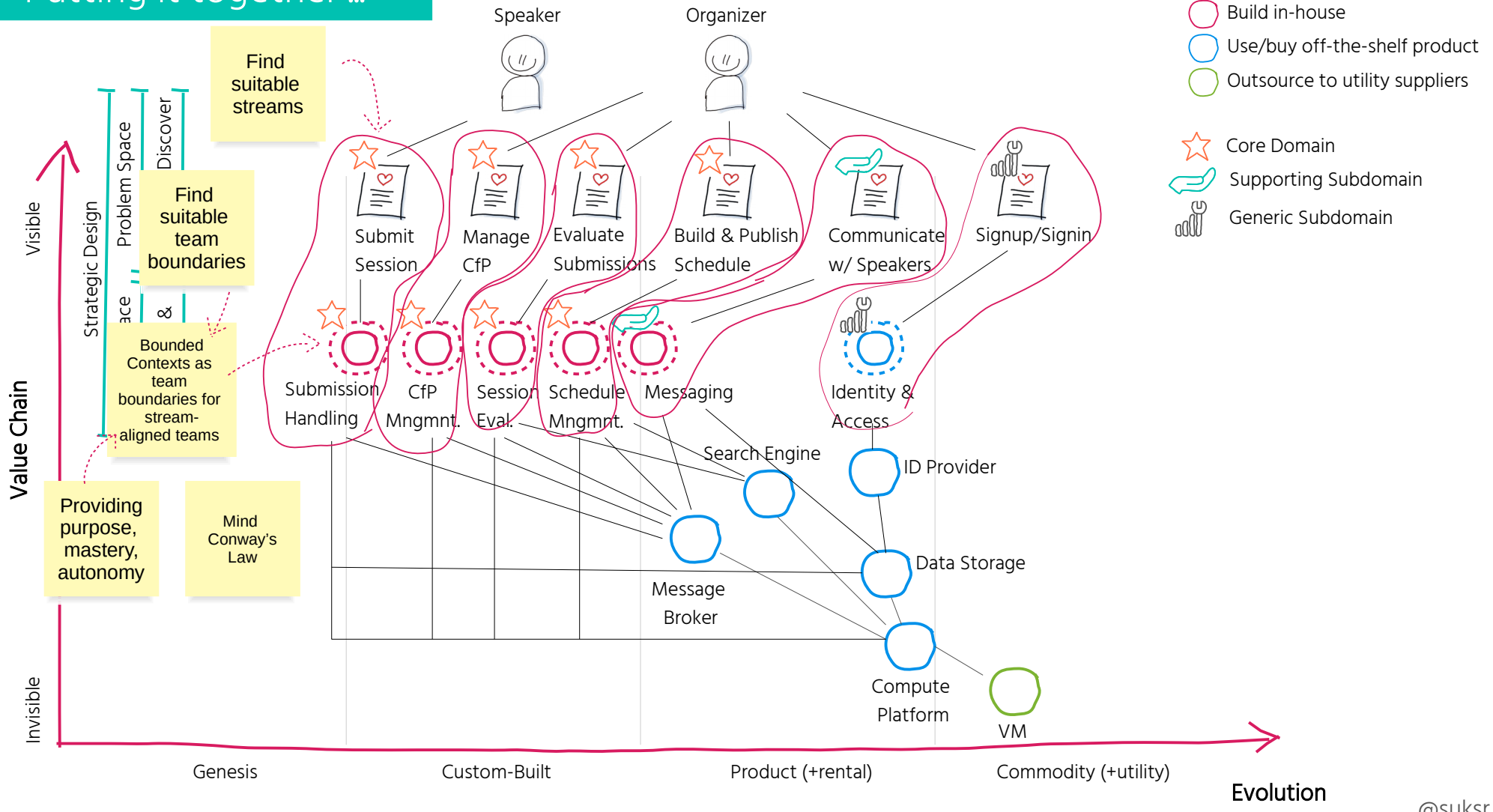
Facilitating

Active help

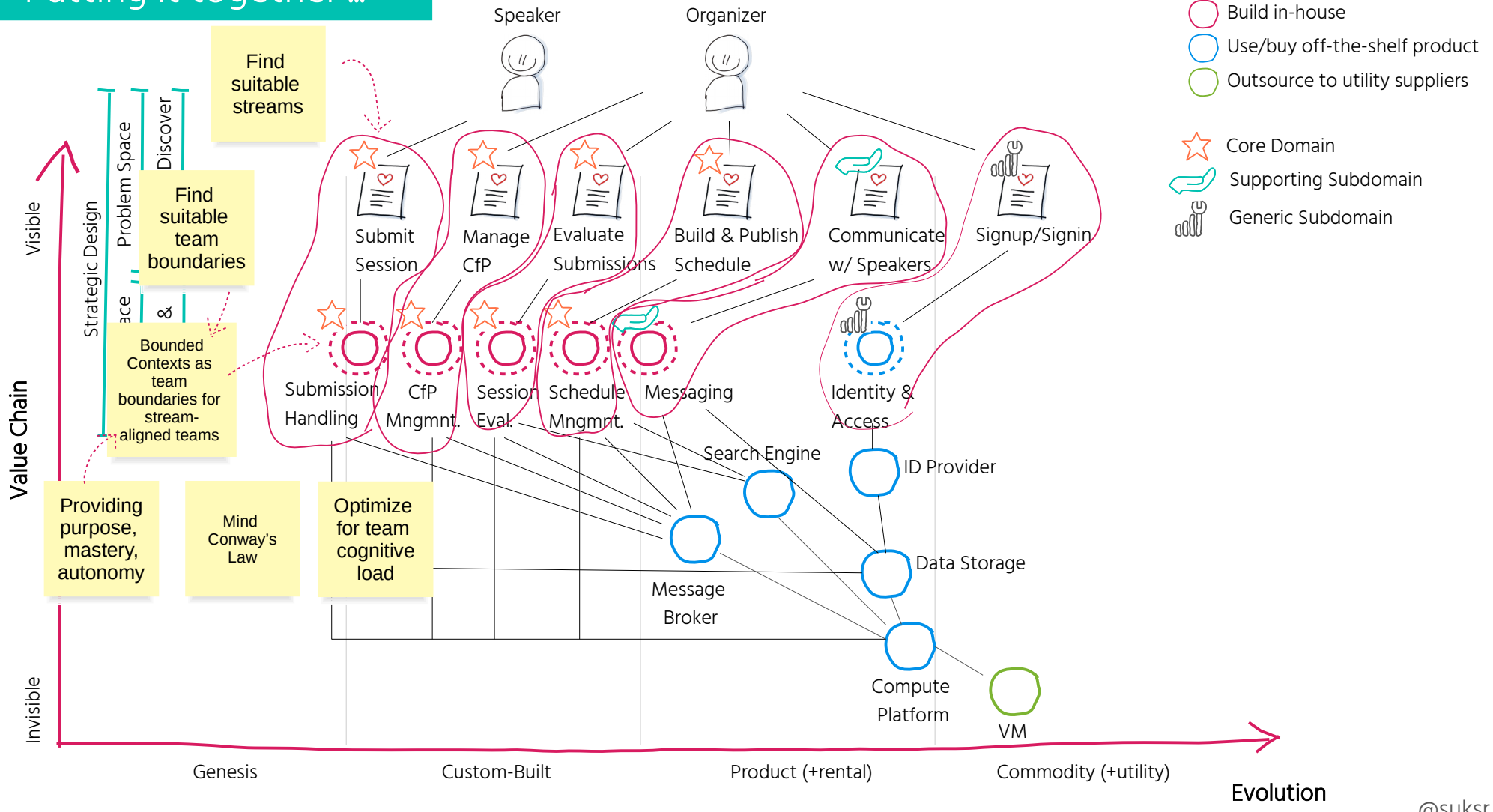
Putting it together ...



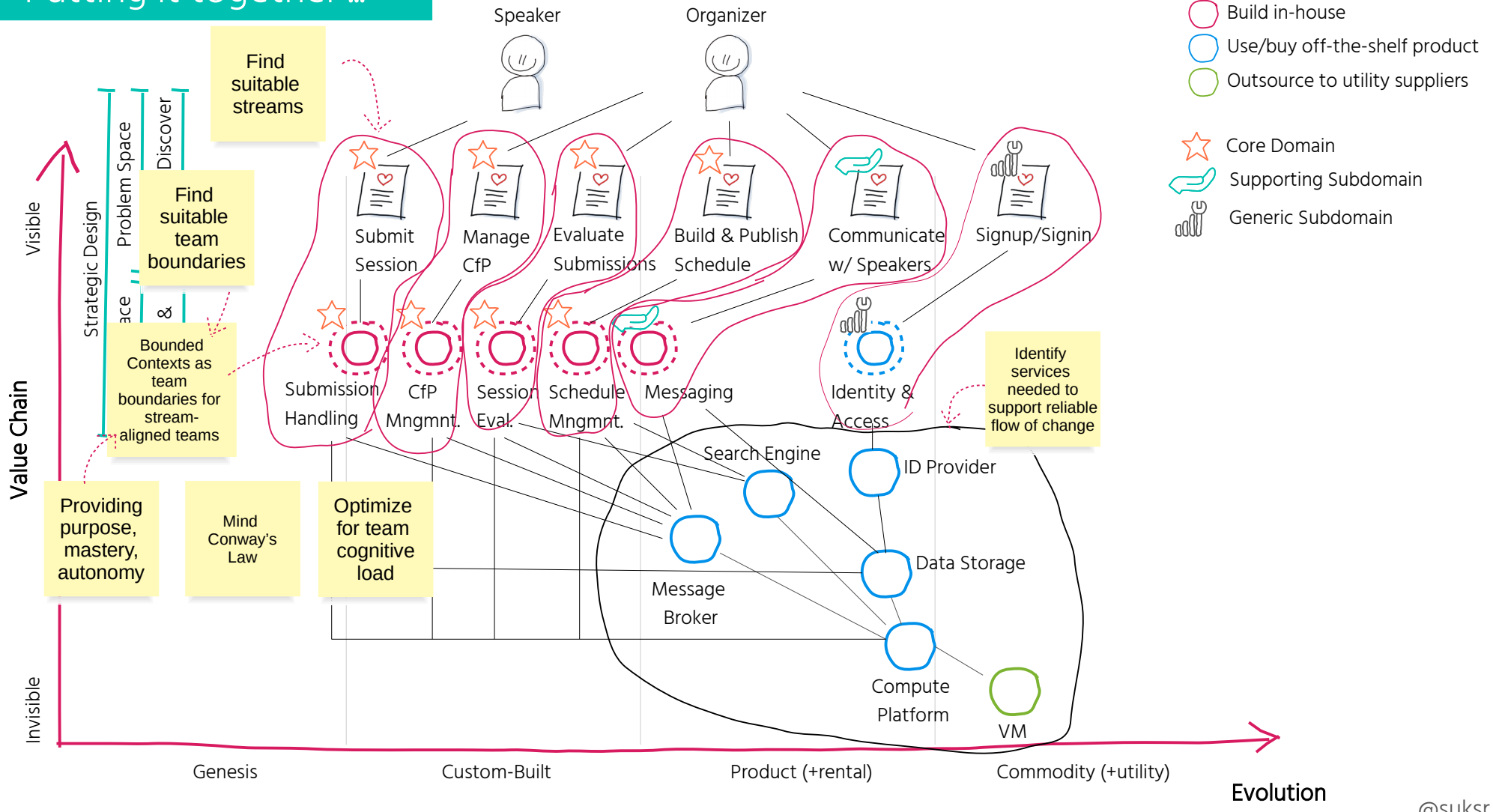
Putting it together ...



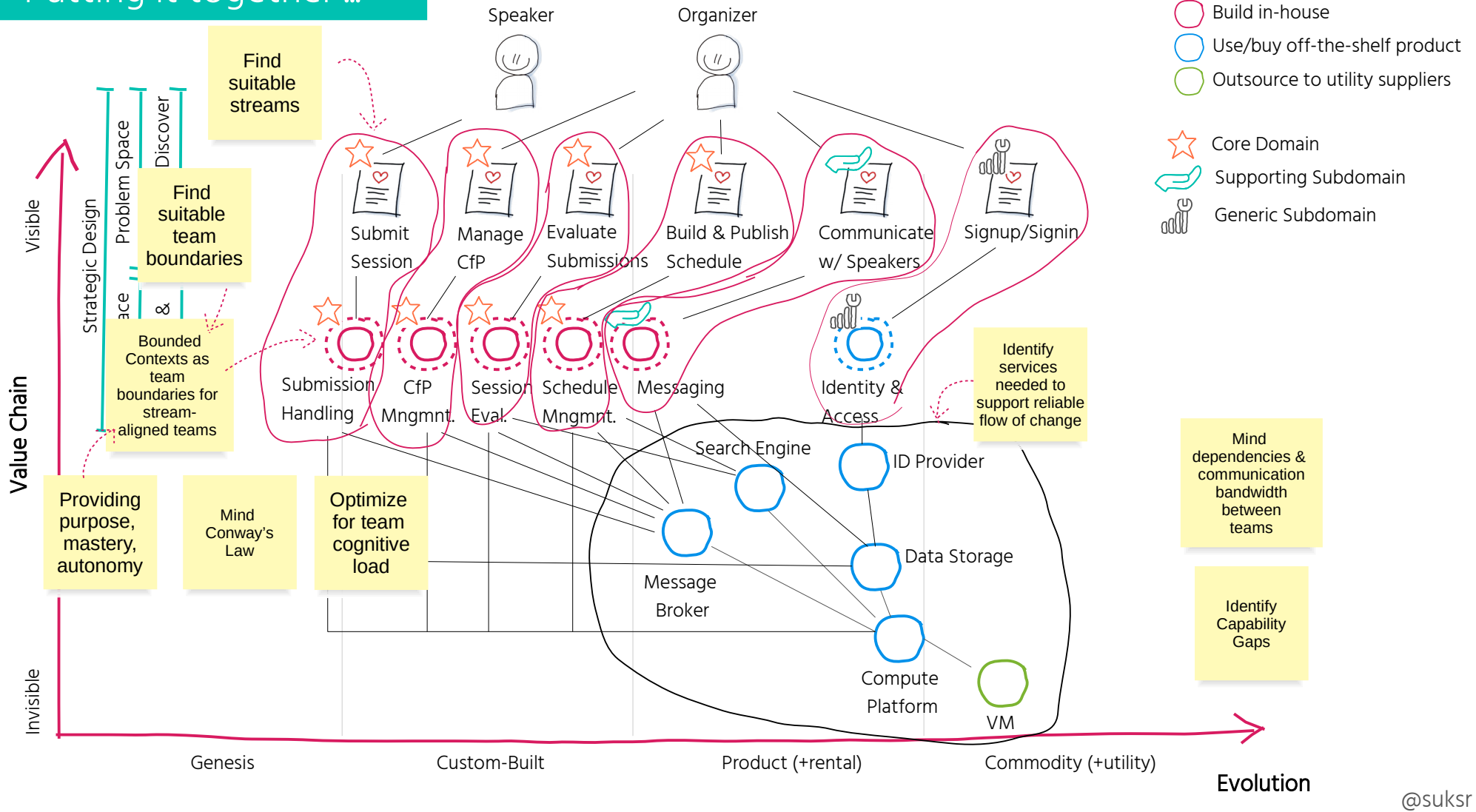
Putting it together ...



Putting it together ...

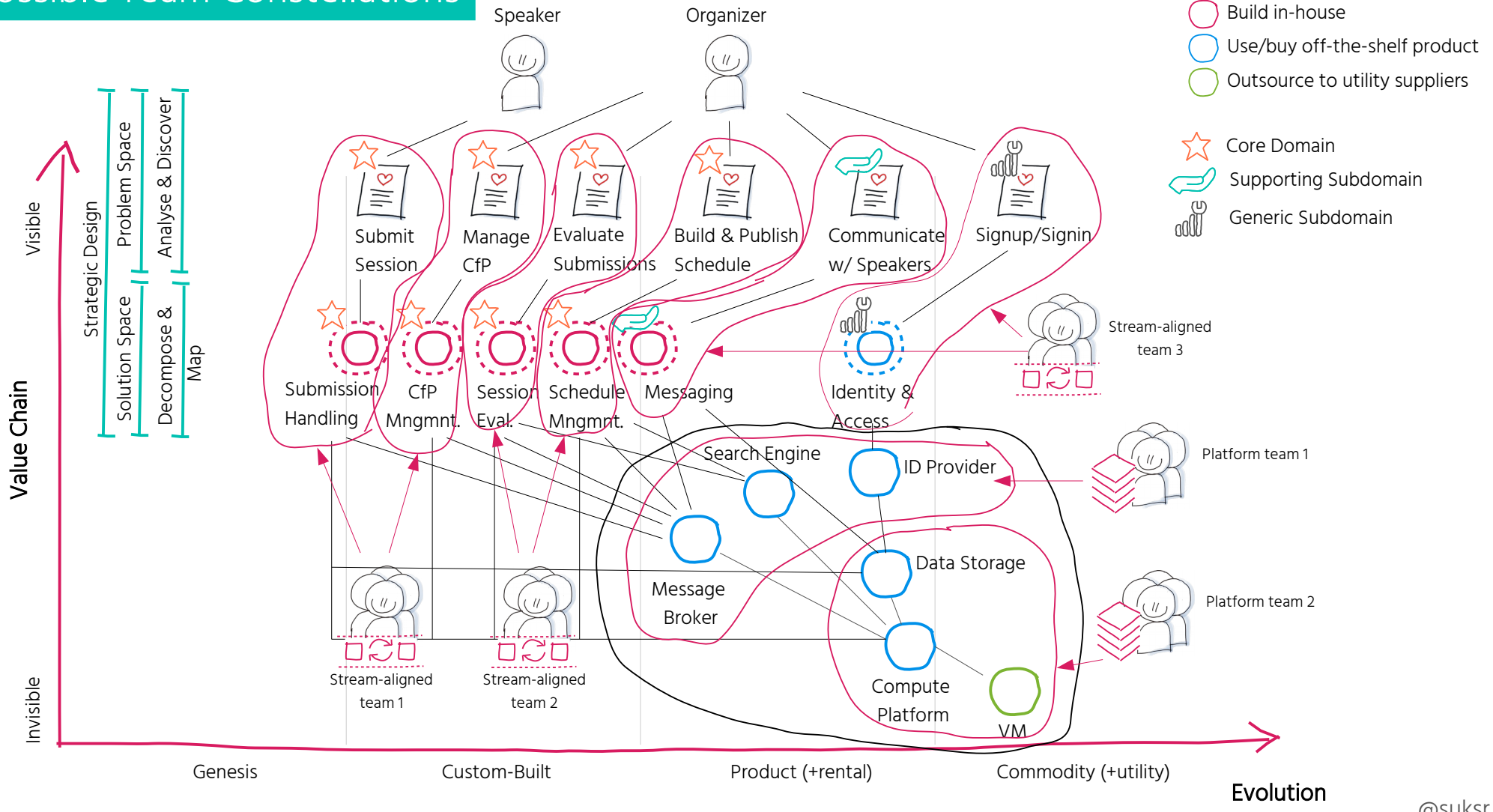


Putting it together ...



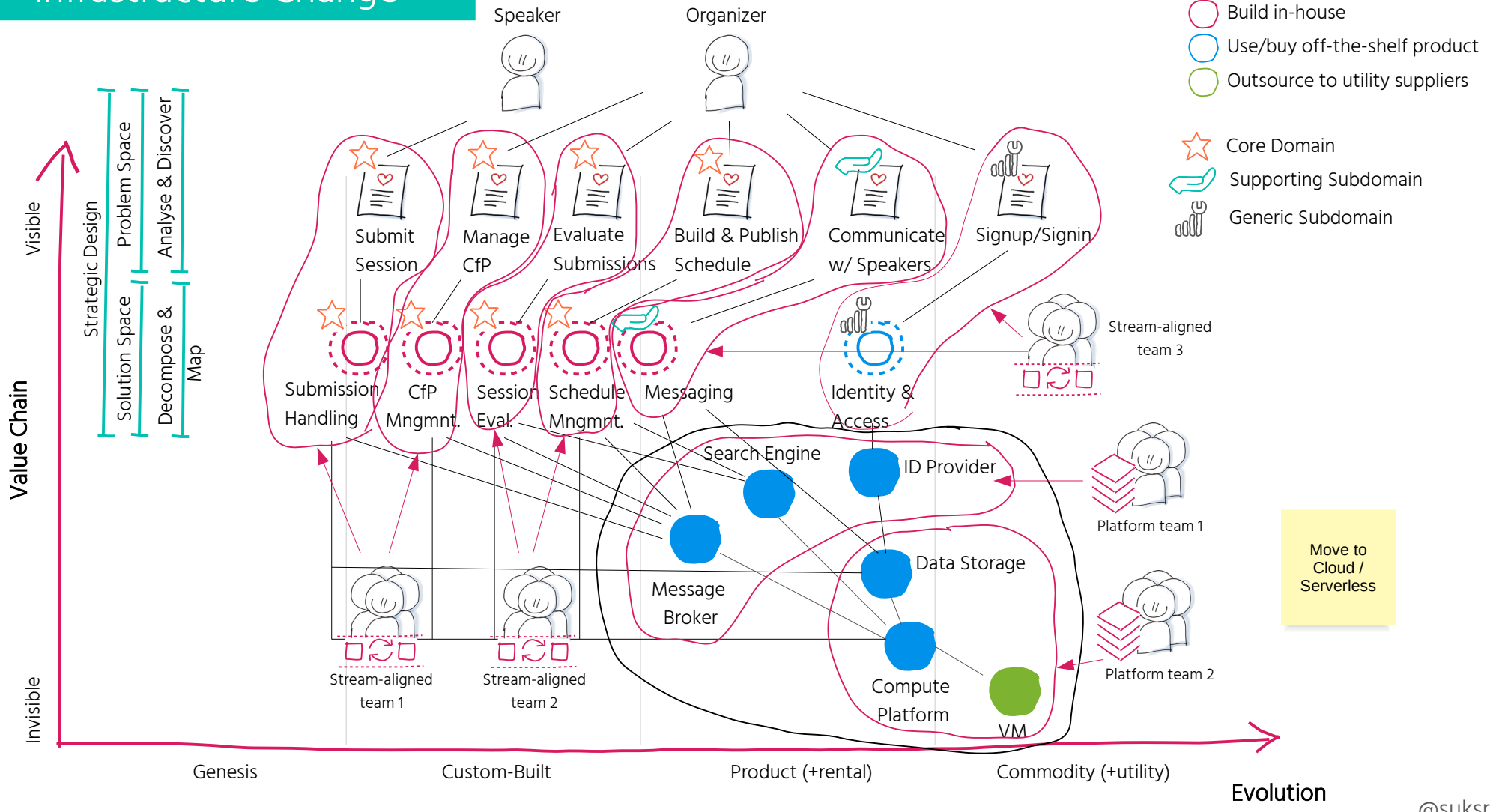
@suksr

Possible Team Constellations



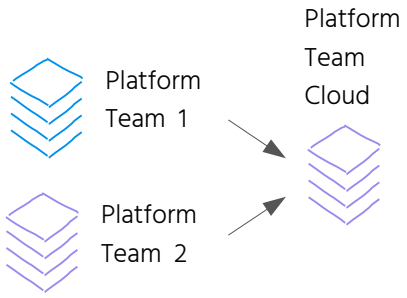
@suksr

Infrastructure Change

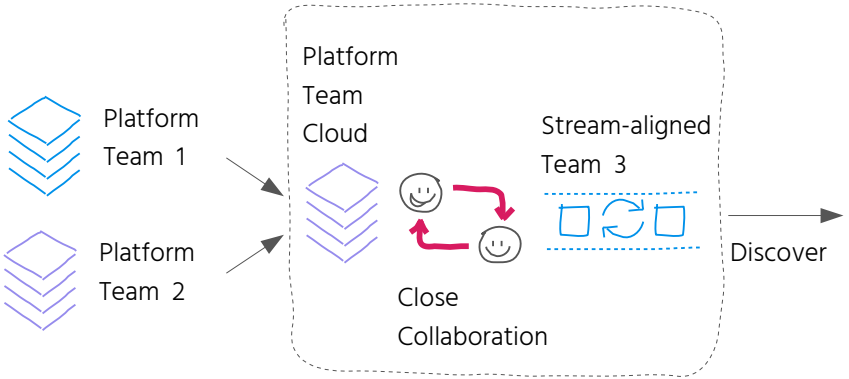


@suksr

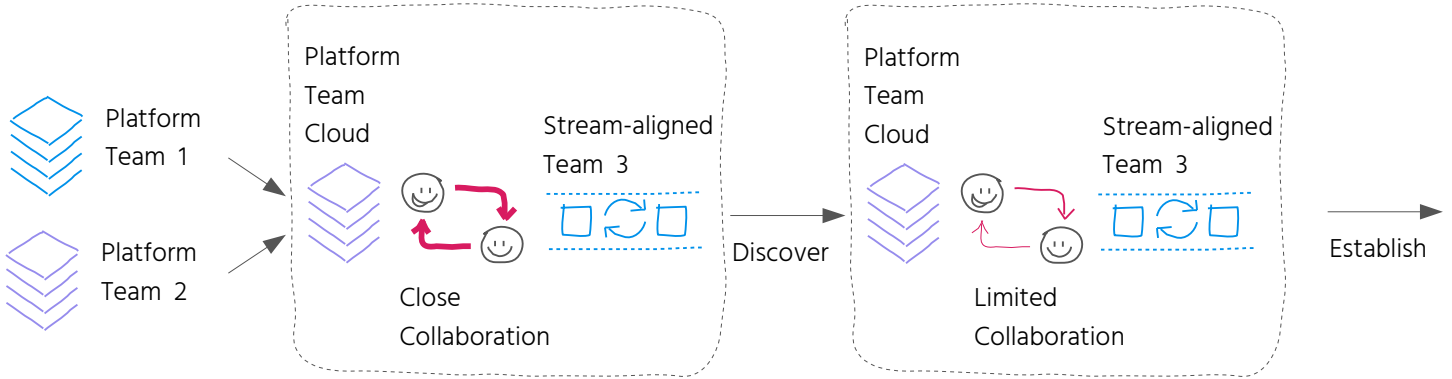
Possible Evolution of Team Topologies



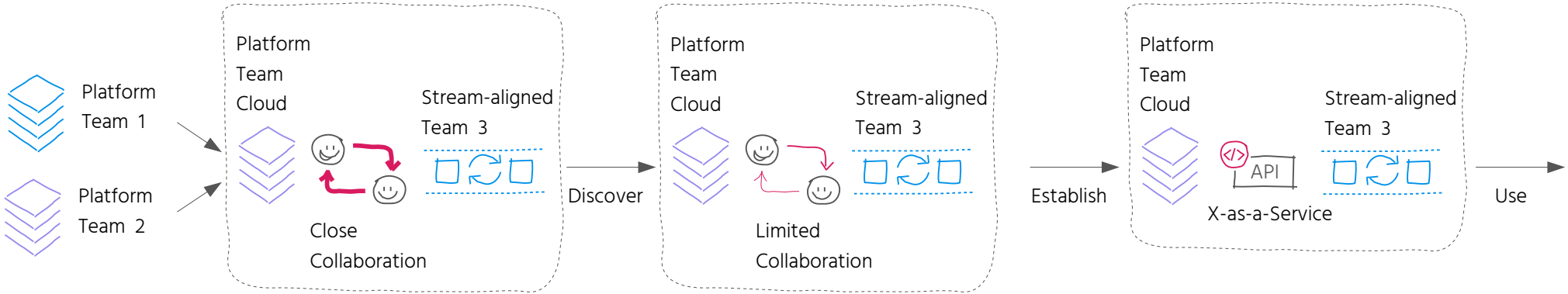
Possible Evolution of Team Topologies



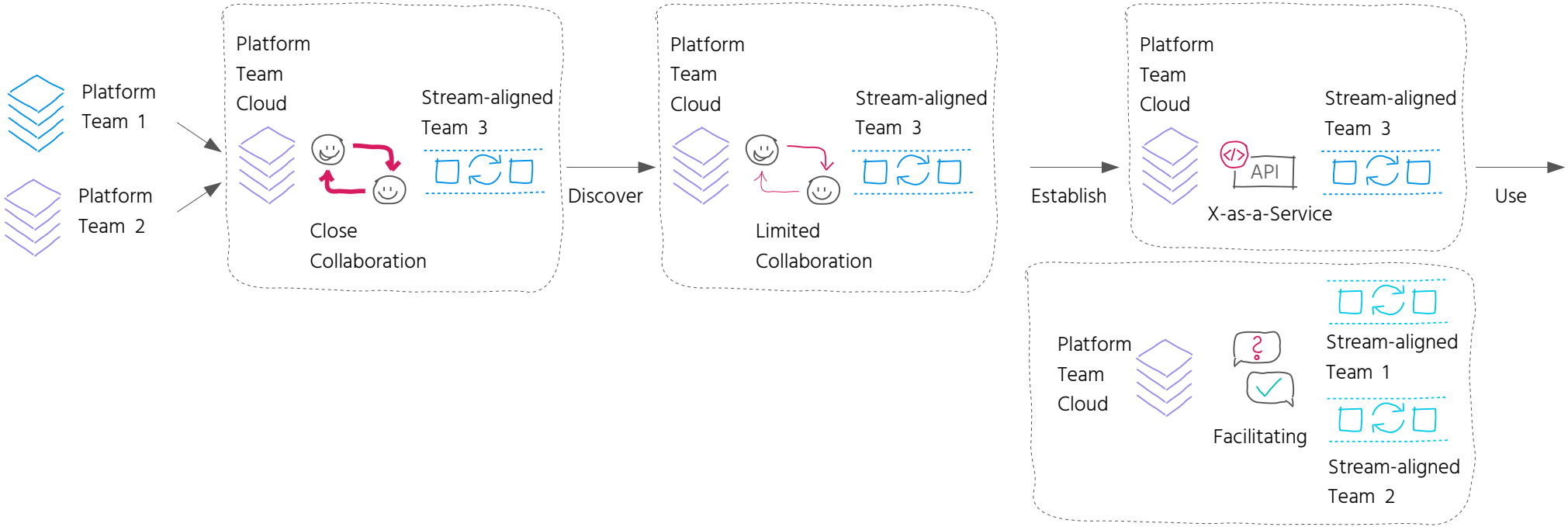
Possible Evolution of Team Topologies



Possible Evolution of Team Topologies

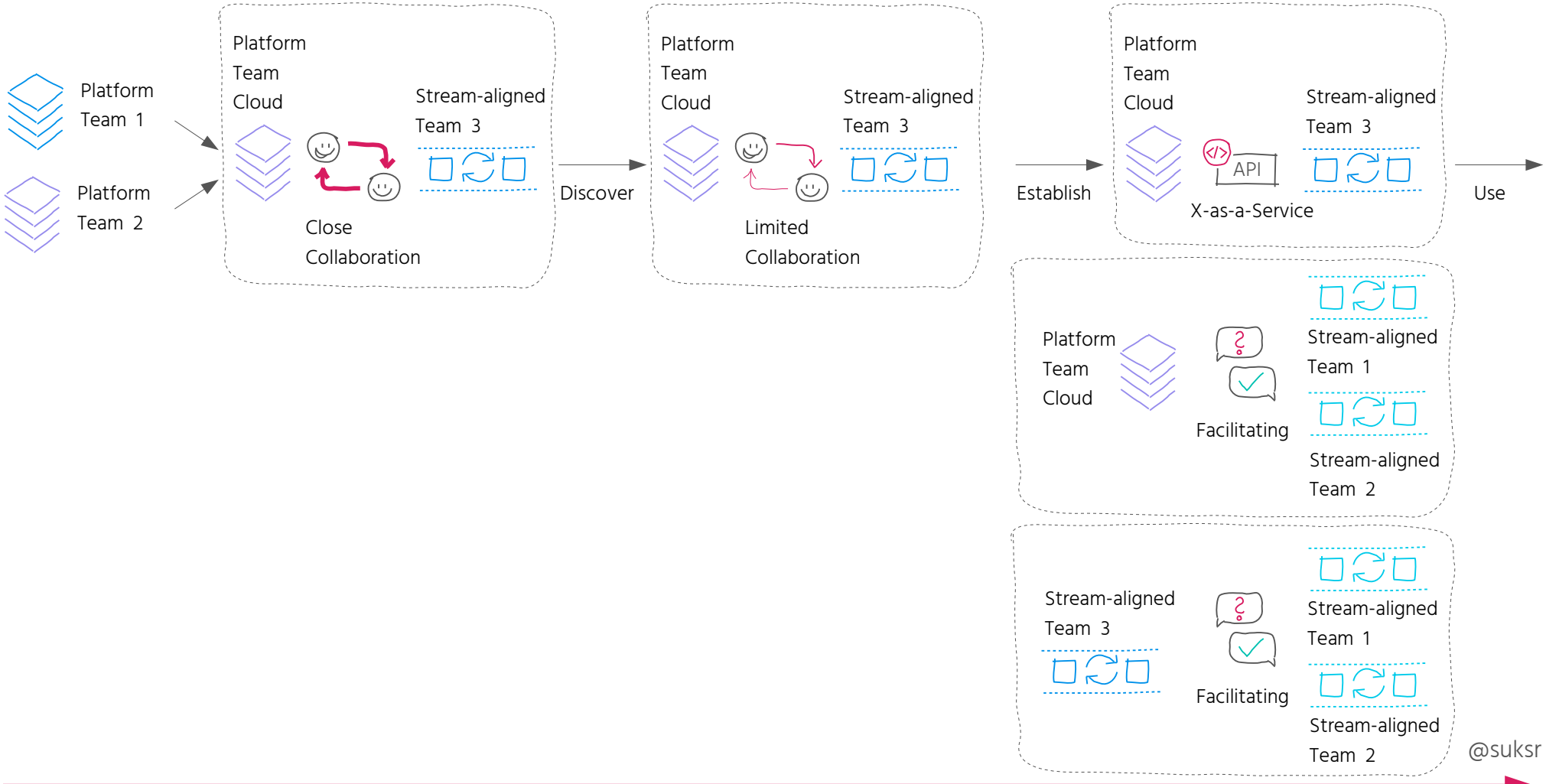


Possible Evolution of Team Topologies

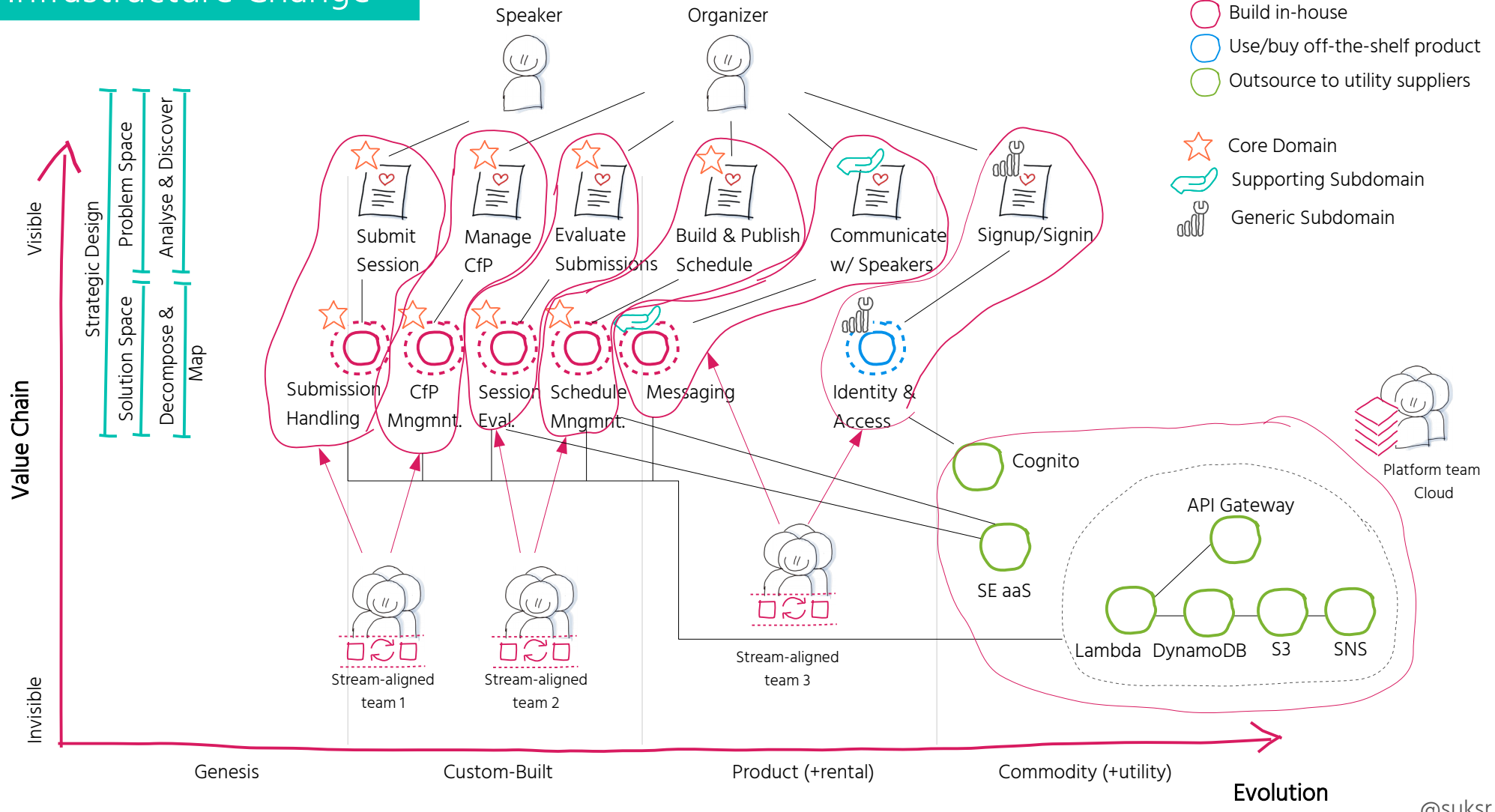


@suksr

Possible Evolution of Team Topologies



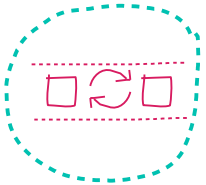
Infrastructure Change



@suksr

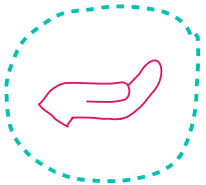
Team Topologies & Doctrine

Stream-aligned team

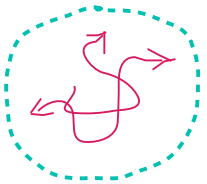


Platform team

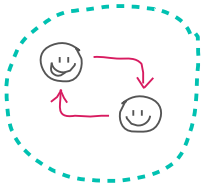
Enabling team



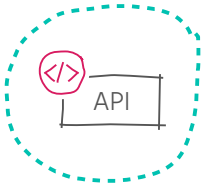
Complicated subsystem team



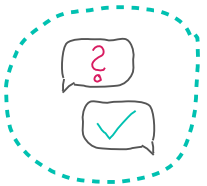
Collaboration

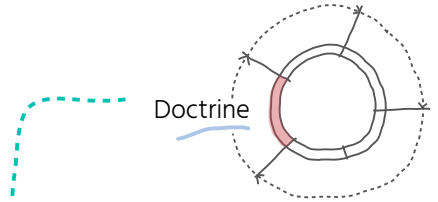


X-as-a-Service



Facilitating





Think small teams

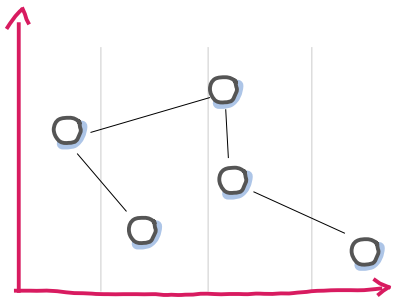
Optimize flow

Provide purpose,
mastery & autonomy

Design for constant evolution

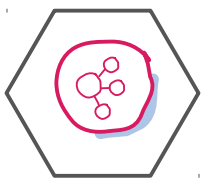
Building Adaptive Systems for a Fast Flow of Change

1. Business-Strategy



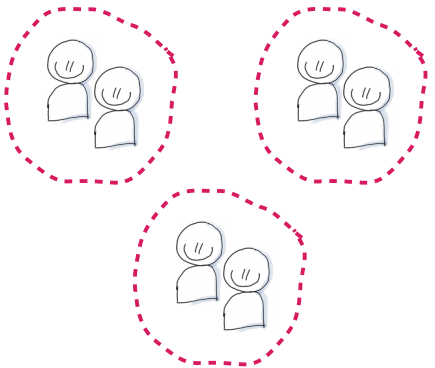
w/ Wardley Mapping

2. Software-Design/
-Architecture



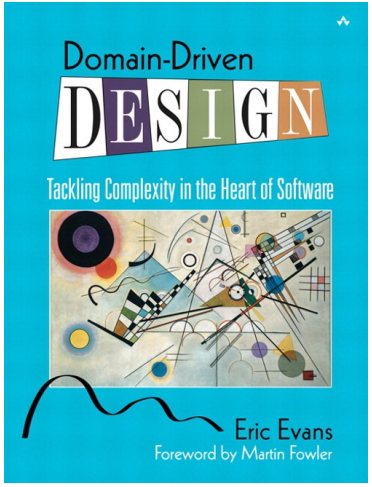
w/ Domain-Driven Design

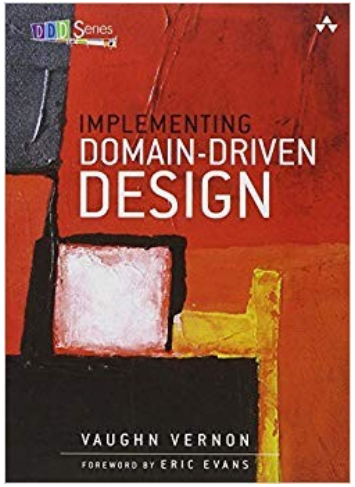
3. Team-Organization

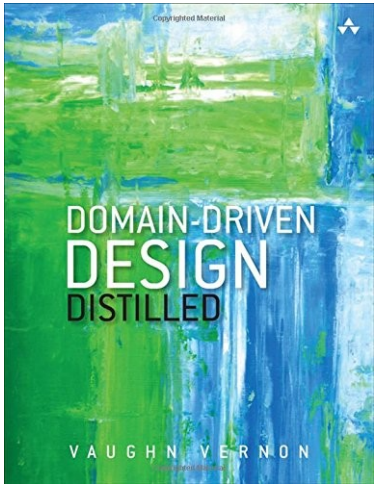


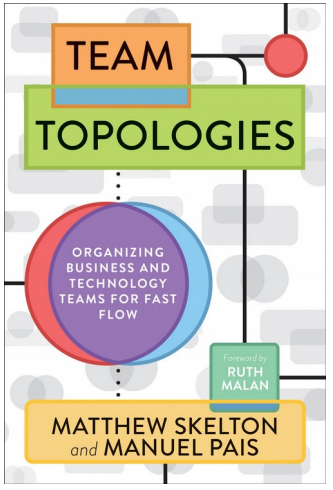
w/ Team Topologies

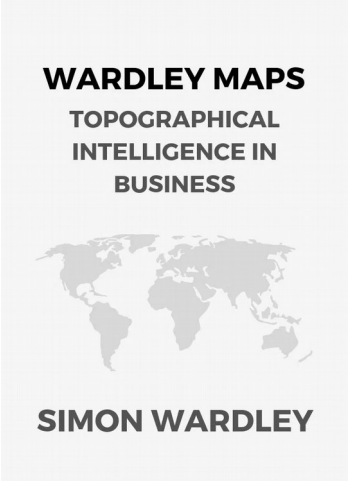
Some References











https://medium.com/wardleymaps
https://learnwardleymapping.com/
https://github.com/wardley-maps-community/awesome-wardley-maps
https://github.com/ddd-crew
https://www.dddheuristics.com

THANK YOU

Susanne Kaiser
Independent Tech Consultant
@suksr
susanne@susannekaiser.net