Stijn Volders

@ONE75

http://blog.one75.be

Context Mapping

Why a Context Map is essential for the success of your project

domaindriven.be – DDD Basics

What

- A simple diagram with Bounded Context's who are involved in your project an your relation with them

Why

- Get insight in inter-team communication an their bandwith
- Measure to which extent a vision could be shared
- Help to decide if DDD is appropriate

It’s all about relations

Relations fall into Patterns

- Partnership
- Customer/Supplier
- Big Ball of Mud
- Conformist
- Shared Kernel
- Separate Ways

All relations have their price

Corbis

Definition: Partnership

When teams in 2 contexts succeed or fail together

- Co-ordintad planning of development and joint management of integration
- Interdependent features should be scheduled so that they are completed for the same release

Definition: Customer - Supplier

- 2 teams in an upstream/downstream relationship
- Upstream can succeed interdependently of the downstream team
- Downstream priorities factor into upstream planning

Definition: Big Ball of Mud

When you have to deal with a big, monolithic (legacy) system

- Often with large intermixed models

Definition: Conformist

- 2 teams in an upstream/downstream relationship
- Upstream has no motivation to provide for the downstream team’s needs
- Downstream team doesn’t put effort in translation

Definition: Shared Kernel

Sharing a part of the model

- Very intimite interdependency
- Can leverage or undermine design work
- Must be kept small!

Definition: Separate Ways

Used to cut loose systems

- Integration is always expensive
- When there is little to zero interest/benefit in integration

Definition: Anticorruption Layer

- Isolating layer to limit impact of changes in an upstream system
- Requires little or no modification to the upstream system
- Translates in both directions between the 2 models

Definition: Open Host Service

When you want to open your system for integration

- Protocol that gives access to your system as a set of services
- Open for everyone that needs to integrate with your system
- New services can be added on request but need to serve a "general" purpose

Definition: Published Language

When you want a common language for translation between 2 bounded contexts

- Often combined with Open Host Service

Upstream/Downstream

Upstream/Downstream

- Not about the direction of the data!
- Upstream will influence downstream
- Might be technical (code)
- But also: schedule, responsiveness to external requests…

Attention!

- Map the current situation, not the (desired) future
- If it’s a mess, show it. It’ll bring focus to the problems & risks involved in your project.

Let’s try it!

Domain: Amazon

Step 1

- Draw each Context with the name (as known in the UL)

Step 2

- Define which models communicate
    - Bandwith

Step 3

- Indicate the direction of the relation, if applicable (Upstream/Downstream)

Step 4

- Put the name of the Pattern on the relationship
  - Partnership
  - Customer/Supplier
  - Conformist

  - Separate Ways
  - Big Ball of Mud
  - Shared Kernel

Step 5

- Indicate which BC’s have explicit translation or sharing
    - Anti Corruption Layer
    - Open Host/Published Language

Questions?

Stijn Volders

@ONE75

http://blog.one75.be