Domain driven design

WEIMAR QUINTERO CALLE
cc 1017146093

Desde que iniciamos en el mundo de los sistemas de información, la gran mayoría de nosotros, comenzamos a seguir un “molde” para hacer casi todo lo que nos exige nuestro campo. Tratamos de ver objetos y relaciones en todos los diseños que nos plantean crear, la esencia de la POO.
Pero los sistemas han avanzado hasta el punto en el cual es difícil distinguir o más bien obtener un buen modelo basándonos solo en la POO.
La computación en la nube y por ende los sistemas distribuidos hacen que nuestro enfoque de trabajo cambie un poco y se centre también en los eventos en un sistema. Aquí es donde está la esencia para entender un dominio porque nos dan una visión más global del comportamiento de todo el sistema, además se interpreta de una mejor manera las metodologías ágiles generando una mejor visión de la arquitectura.

DDD es un enfoque de desarrollo de software que busca una profunda conexión entre los conceptos y áreas del negocio y las arquitecturas aplicativas.

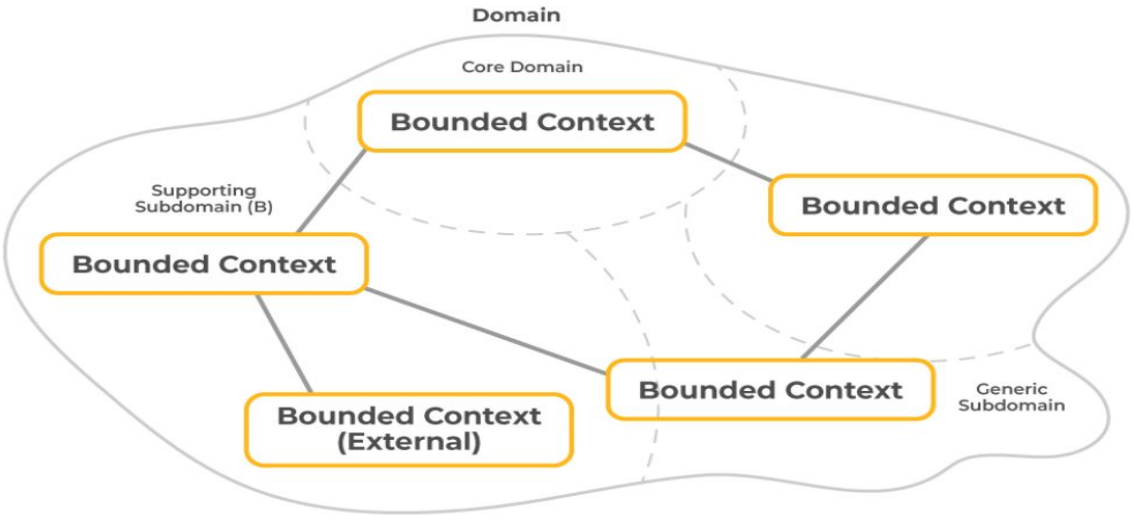
Uno de los mayores problemas que se presentan en el desarrollo de un sistema es la comunicación confusa entre los desarrolladores y los expertos del dominio. Los expertos del dominio tienen claro su conocimiento de este, pero son escasos sus conocimientos técnicos del software. Por el contrario para los desarrolladores el problema radica en que es muy poco lo que entienden del dominio y muchas veces malinterpretan las expectativas que tienen los expertos del dominio acerca de lo que quieren que haga el sistema. Los procesos de traducción de ambas partes muchas veces generan más problemas que soluciones provocando que muchos de los interesados entiendan conceptos de manera diferente. Un lenguaje común será la solución y para dominarlo se necesita que los desarrolladores obtengan el conocimiento necesario del dominio y los expertos del dominio formen parte activa en el desarrollo del software.

Modelado estratégico

En esta parte del DDD se busca un modelo que nos ayude a entender y resolver las necesidades o problemas que se tienen en el dominio. Para la implementación de la etapa estratégica se deben tener en cuenta los siguientes conceptos:

- **Dominio:** Es el problema que se busca modelar y solucionar.

- **Lenguaje ubicuo:** Lenguaje común entre el equipo de tecnología y los usuarios.
- **Contexto Delimitado:** Fronteras lógicas dentro del modelo, pueden incluir uno o más subdominios.
- **Subdominios Core:** Son los procesos (contextos delimitados) más críticos del negocio, la razón del modelado.
- **Subdominio:** Es la separación del dominio en diferentes problemas o requerimientos.
- **Mapa de contexto:** Es el resultado final de las relaciones entre los contextos delimitados.



Modelado Táctico

La parte táctica corresponde a la implementación de la solución. Aquí se deben tener en cuenta los siguientes aspectos:

- **Entidades:** Cualquier objeto del dominio que mantiene un estado y comportamiento más allá de la ejecución de la aplicación y que necesita ser distinguido de otro que tenga las mismas propiedades y comportamientos.
- **Objetos de Valor:** Al contrario que las entidades estos representan conceptos que no tiene identidad y nos interesan más sus atributos.
- **Servicios:** La mayoría de los verbos del lenguaje ubicuo se convertirán en métodos de los objetos de la capa de negocios. Pero hay verbos o comportamientos que no es fácil concretar a cual de los objetos corresponden. Esos comportamientos suelen ser en realidad Servicios.
- **Módulos:** permiten seguir con facilidad el concepto de 'acoplamientos débiles y alta cohesión'. Cuando una aplicación comienza a ser compleja, es preferible dividirla en módulos.

- **Agregados:** Mientras los módulos suelen estar formados de clases relacionadas con los servicios o la funcionalidad de la aplicación, los agregados son grupos de entidades relacionadas entre sí a nivel de negocio.
- **Factorías:** Son clases encargadas de crear entidades o agregados, construyendo todas las entidades contenidas en ellos. Son útiles especialmente cuando las reglas de creación de estas entidades o agregados son complejas.