Domain Driven Design – 10 study points

● what is DDD?

Domain-driven design (DDD) is the concept that the structure and language of software code (class names, class methods, class variables) should match the business domain. For example, if a software processes loan applications, it might have classes such as LoanApplication and Customer, and methods such as AcceptOffer and Withdraw.

Source: <https://en.wikipedia.org/wiki/Domain-driven_design>

● when did it originate?

2004

Eric Evans, introduced the concept in 2004, in his book Domain-Driven Design: Tackling Complexity in the Heart of Software.

Source: <https://medium.com/microtica/the-concept-of-domain-driven-design-explained-3184c0fd7c3f>

● which problems did it address?

Domain driven design is a key element of SOA(Service Oriented Architecture) architecture because it helps in encapsulating the business logic and rules in domain objects. The domain model also provides the language and context with which the service contract can be defined.

Source: <https://www.infoq.com/articles/ddd-in-practice/>

● which are the basic concepts and building blocks of a domain model

Domain-Driven Design articulates a number of high-level concepts and practices.

Of primary importance is domain, the subject area to which the user applies a program is the domain of the software. A software's domain governs its context, the setting in which a word or statement appears that determines its meaning. From this, developers build a domain model: a system of abstractions that describes selected aspects of a domain and can be used to solve problems related to that domain.

These aspects of domain-driven design aim to foster ubiquitous language, meaning that the domain model should form a common language shared by domain experts for describing system requirements, business users, sponsors and developers.

In domain-driven design, the domain layer is one of the common layers in an object-oriented multilayered architecture.

Source: <https://en.wikipedia.org/wiki/Domain-driven_design>

● how does DDD relate to microservices architecture?

Microservices have a symbiotic relationship with domain-driven design (DDD)—a design approach where the business domain is carefully modeled in software and evolved over time, independently of the plumbing that makes the system work

Source: <https://www.confluent.io/blog/microservices-apache-kafka-domain-driven-design/#:~:text=Microservices%20have%20a%20symbiotic%20relationship,that%20makes%20the%20system%20work>.