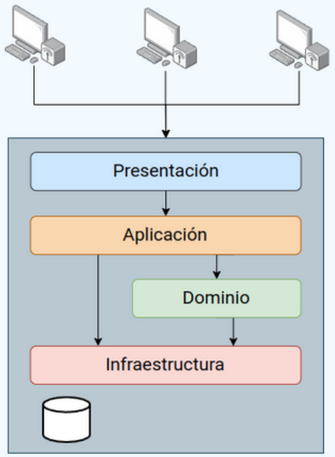
DOMAIN DRIVEN DESIGN

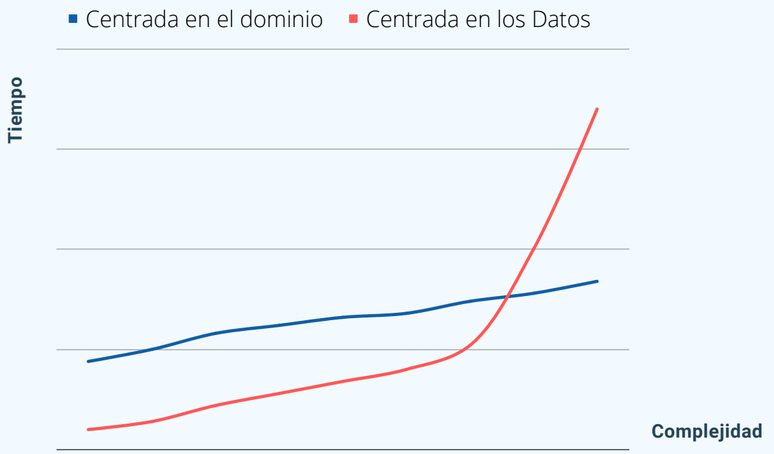
PROS Y CONTRAS

Daniel Blanco Calviño

ASPECTOS POSITIVOS

- **Lenguaje común compartido** por todos los integrantes del proyecto. Perfiles técnicos y no técnicos.
- Al tener lógica de aplicación y dominio separadas, el **coste de realizar modificaciones suele ser menor,** ya que el dominio cambia con poca frecuencia.
- El código del dominio es autoexplicativo.
- **Mayor velocidad de desarrollo a medio y largo plazo.** Mucho más mantenible.





ASPECTOS NEGATIVOS 

- Mucho más **lento al principio** de un proyecto.
- Requiere tener **expertos en el dominio del problema.**
- Cambio de mentalidad de los desarrolladores para enfocarse en la funcionalidad más que en los datos.
- Los frameworks actuales nos "empujan" a un modelo anémico y a pensar centrándonos en los datos.

CUÁNDO USAR

- Proyectos complejos con un **largo tiempo de vida esperado.**
- **Incertidumbre en los casos de uso.** Previsión de cambios en el futuro.
- Problemas con una **lógica de dominio presente.**
  - Si lo único que necesitas es un CRUD, no tiene sentido.
- Disponibilidad de un equipo comprometido en analizar detalladamente el dominio.

DOMAIN DRIVEN DESIGN

PROS Y CONTRAS

Daniel Blanco Calviño