

# Domain-Driven Design con le Minimal API

meno boilerplate, stesso dominio



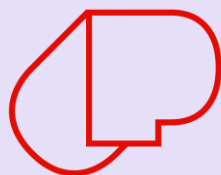
**Alberto Acerbis**  
*Software Architect – Trainer*  
[alberto.acerbis@intre.it](mailto:alberto.acerbis@intre.it)



# Sponsor



CONSORTIVM SERVICE S.r.l.



**UNIPORDENONE**  
moltiplica il valore della formazione



UNIPORDENONE  
moltiplica il valore della formazione



CONSORTIVM SERVICE S.r.l.

# Why we make software?

---

Your job is not to write working code,  
**Your job is to design a working system.**

(Mathias Verraes)







UNIPORDENONE  
moltiplica il valore della formazione



CONSORTIVM SERVICE S.r.l.



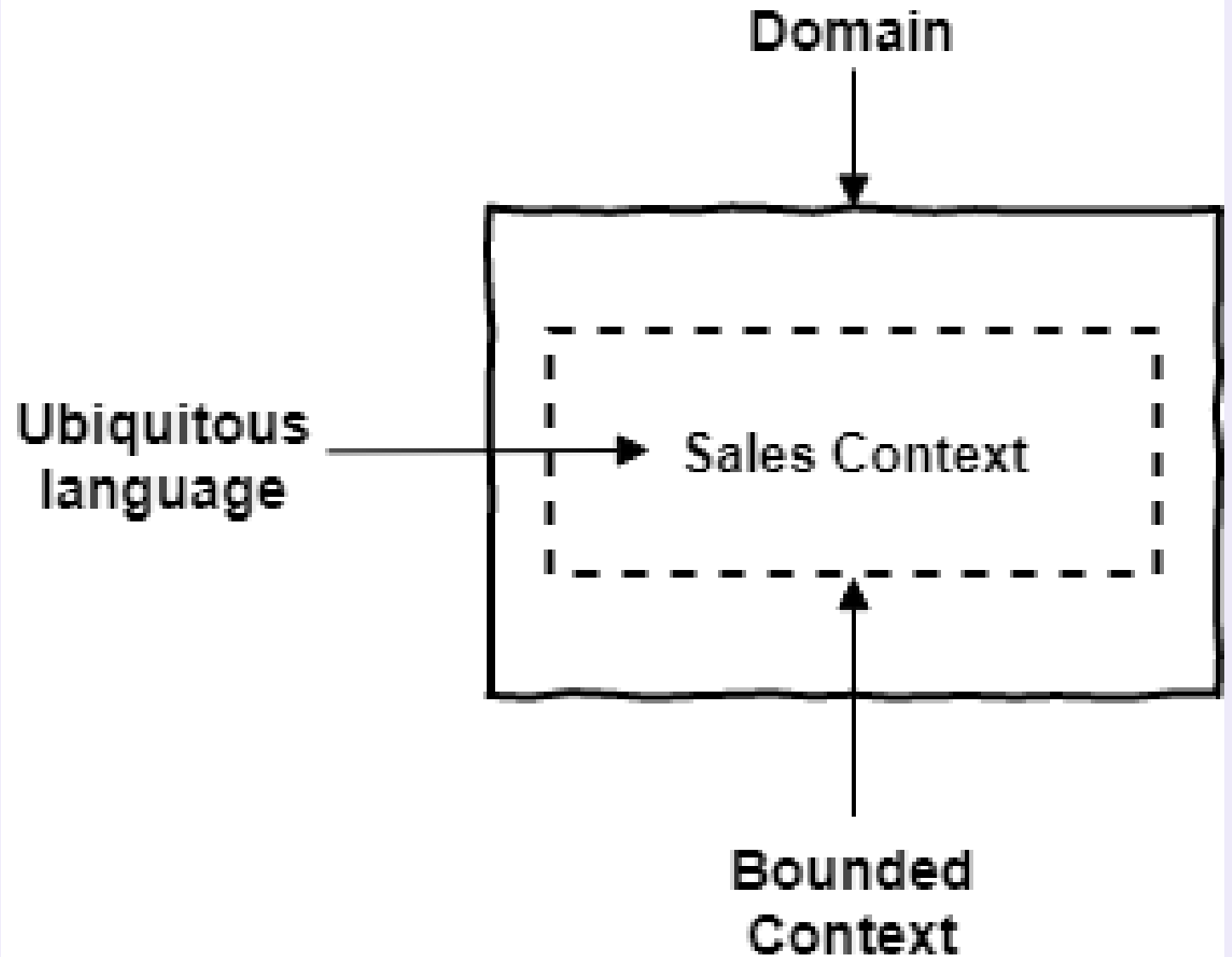
# Make the Implicit Explicit

- Nelle conversazioni umane, il contesto viene compreso perché le persone lo stabiliscono mentre parlano. “**di cosa/chi state parlando?**”.
- **Senza** conoscere il **contesto** quando ci si unisce a una **conversazione**, questa può facilmente **deragliare** se chi arriva tardi assume il contesto sbagliato.
- **Senza** conoscere il **contesto** è impossibile creare un **Modello** corretto.



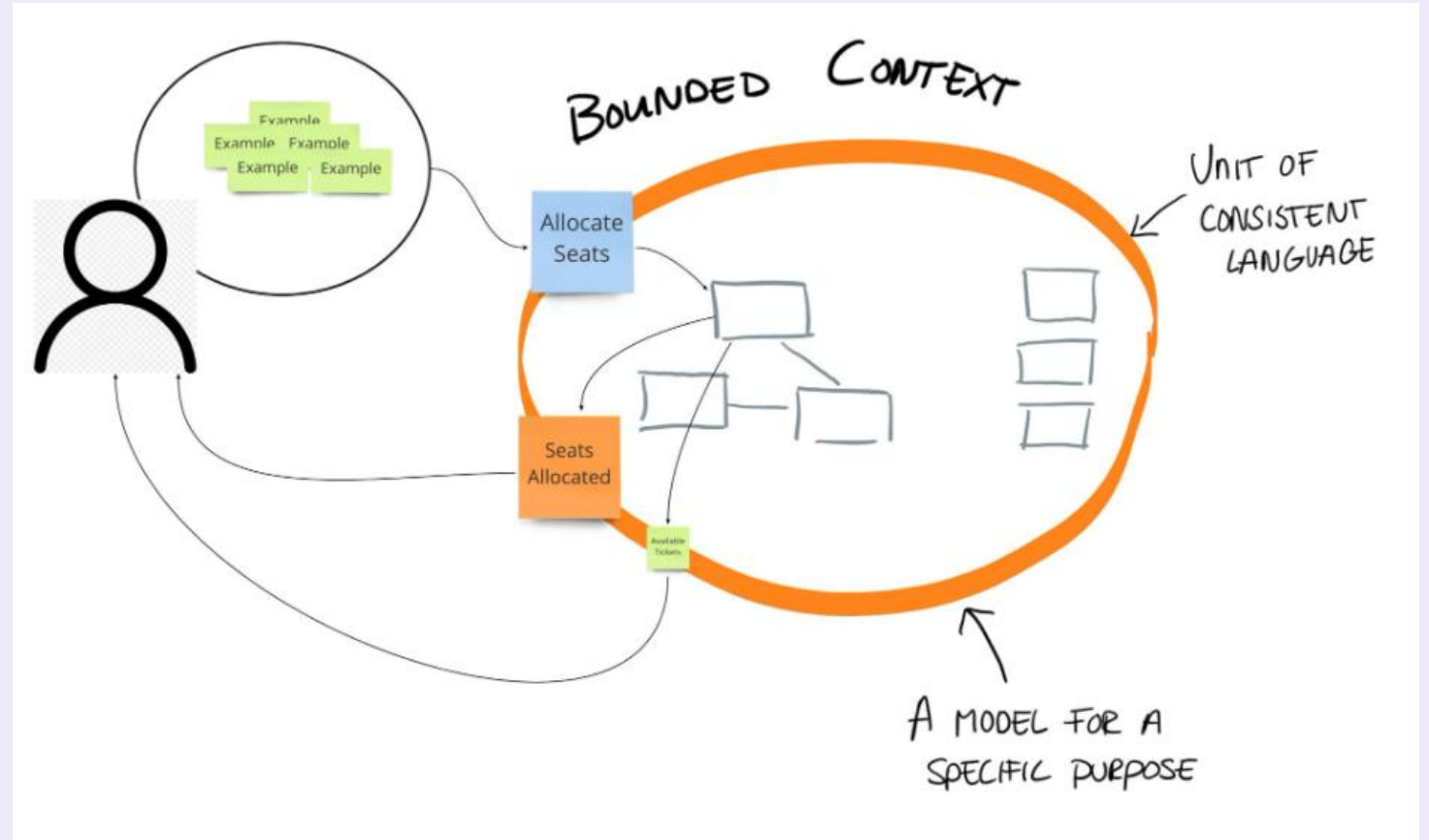
# Ubiquitous Language

- L'Ubiquitous Language è un linguaggio modellato all'interno di un **contesto limitato**, in cui i termini e i concetti del dominio aziendale sono indicati e in cui **non dovrebbero esserci ambiguità**.



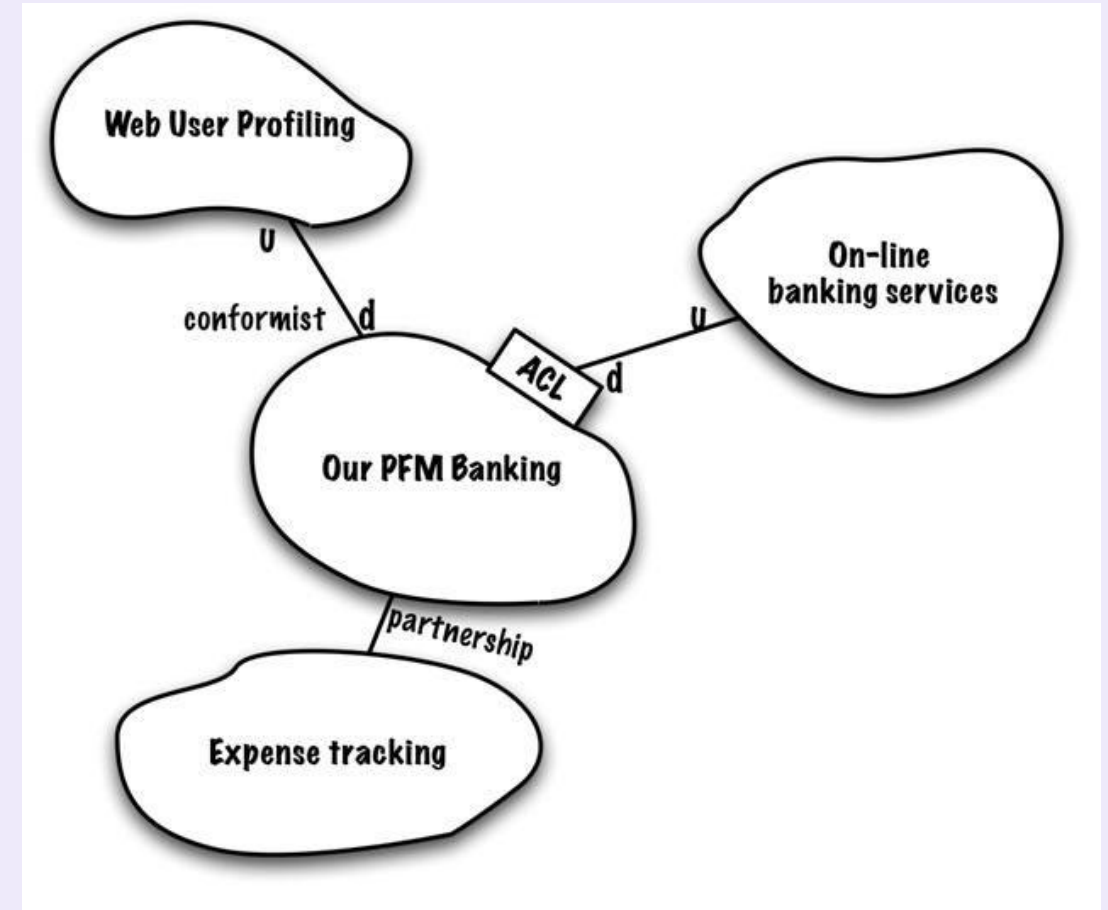
# Bounded Context

- **Context:** Insieme delle circostanze in cui appare una parola o un'affermazione che **ne determina il significato**.
- **Bounded Context** è un confine semantico.
- **All'interno** del perimetro ogni componente ha un **significato preciso**.
- E risolve **problemi specifici**.



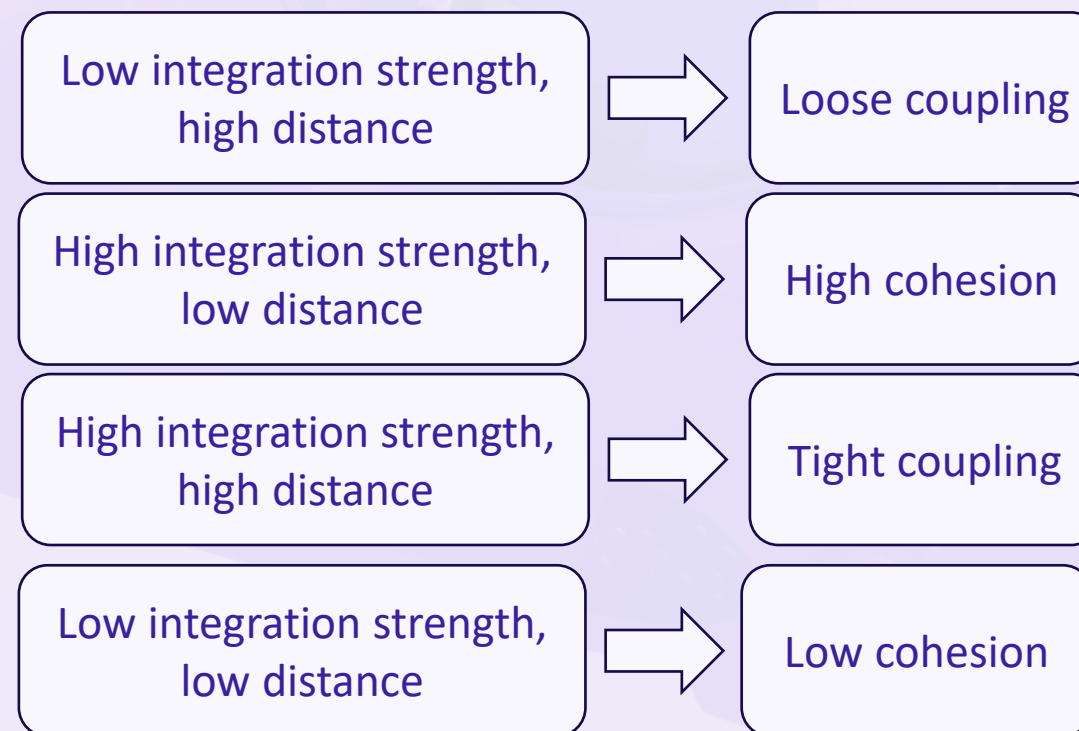
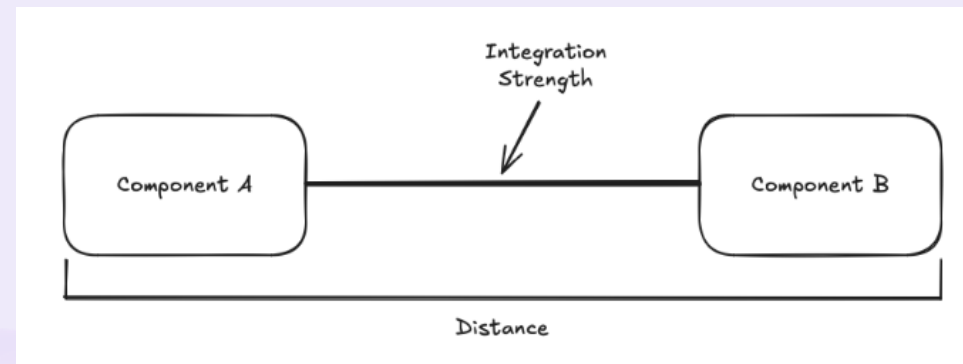
# Context Mapping

- Definisce **contratti di integrazione** e direzioni di dipendenza.
- Evidenzia traduzioni di modello e **Anti-Corruption-Layer**.
- Riduce **coupling accidentale**.
- Guida scelte architetturali (API, eventi, ownership dei modelli).

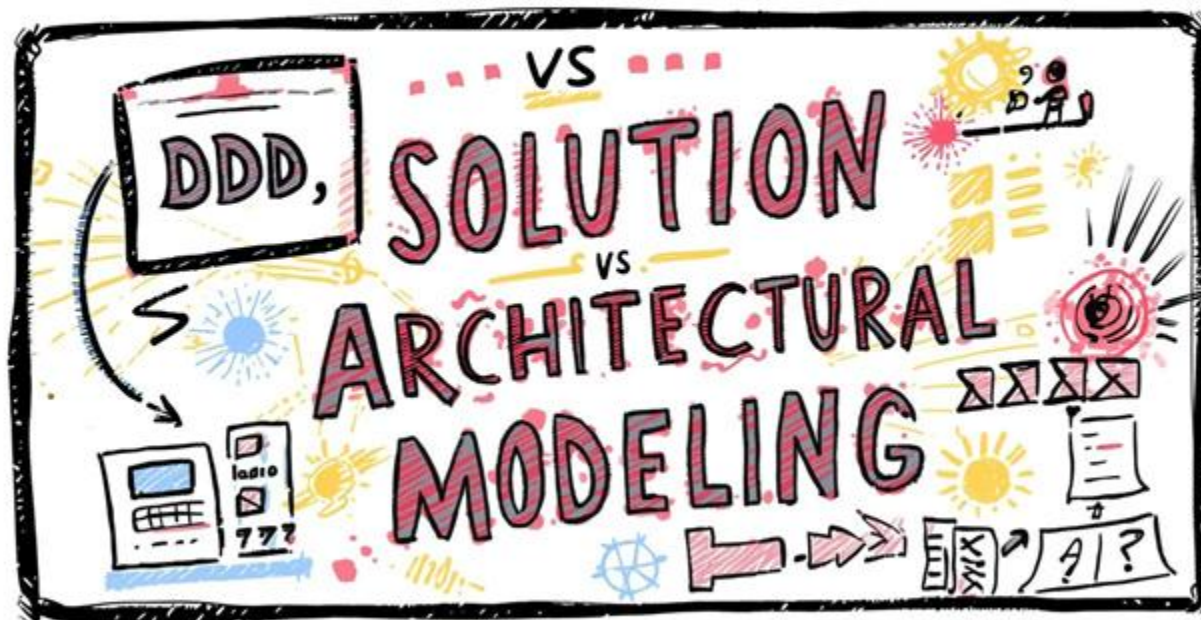




Cohesion	High	Low
Distance	High	Low
High	Tight Coupling	Loose Coupling
Low	High Cohesion	Low Cohesion



## DDD vs. NON-DDD.



Ah Ha, a clear view



UNIPORDENONE  
moltiplica il valore della formazione



CONSORTIVM SERVICE S.r.l.

# Tips and Tricks



- DDD non è una vitamina: non fa bene a prescindere!
- Le Minimal API semplificano, non facilitano!
- Il vero anti-pattern è usare DDD quando non serve.

# Grazie!

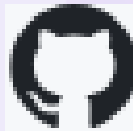


**Alberto Acerbis**

*Software Architect - Trainer*



[alberto.acerbis@intre.it](mailto:alberto.acerbis@intre.it)



<https://github.com/Ace68/ArchitettureEvolutive>

