Resumo da Mudança UML

✔ 1. Separação de responsabilidades

O diagrama separa:

A) Núcleo do Mercado (Mercado, MatchingEngine, OrderBook, Ordens, Transações)

Isso é o *core* da microestrutura.

- ✔ `MatchingEngine` isolado
- ✔ `OrderBook` como entidade de estado
- ✔ Diferenciação entre `OrdemLimite` e `OrdemMercado`
- ✔ `Transacao` como evento

Isso é exatamente o padrão de engines reais (FIX/LOB).

B) Núcleo de Agentes

A classe:

- `Investidor` (abstrato)
- PF e PJ como **heranças corretas**
- Estratégia plugável via composição (`estrategia: EstrategiaDecisao`)

Esse é o *design pattern Strategy* aplicado corretamente.

C) Infraestrutura

Muito bem isolada:

- `Simulacao`
- `Config`
- `Tempo`
- `EventBus`

- `SeedService`
- `Storage`

Isso é a arquitetura de um framework — não mistura lógica do domínio com runtime.

✔️ 2. O grande acerto: **Investidor ≠ Estratégia**

Antes o diagrama confundia:

- “ser ruído”
- com
- “ser pessoa física/jurídica”

Agora está:

Identidade (quem ele é)

- PF
- PJ

Comportamento (como ele decide)

- Ruído
- Especulador
- Fundamentalista
- (e futuros plugins)

✔️ 3. Parte financeira — **Carteira, Posicao, Dinheiro**

- `Dinheiro` como value object
- `Posicao` como agregado mínimo
- `Carteira` como entidade com regras

Isso permite garantir:

- consistência de saldo
- atualização de posição
- preço médio
- impedimento de vendas a descoberto (se quiser)

E deixa pronto para multi-moeda.

✔️ 4. Simulação e Reprodutibilidade

O bloco:

- `SeedService`
- `Tempo`
- `EventBus`
- `Storage`

é literalmente:

- **determinismo**
- **relógio lógico**
- **event streaming interno**
- **trilha de auditoria**

Isso é um padrão muito forte de frameworks de ABM/mercado.

✔️ 5. O diagrama está coerente com a narrativa do Capítulo 4

Eu revisei a estrutura do capítulo que já usa:

- Agentes heterogêneos
- Microestrutura explícita
- Mecanismo de casamento
- Parametrização externa
- Execução determinística
- Modulação de estratégias

Tudo está **refletido no UML**.

> “Claro, isso é um *framework*, não um modelo fechado.”

O que era a meta.

✔️ 6. Multiplicidades e relações

- Mercado contém vários OrderBooks
- Carteira contém várias posições
- Investidor tem exatamente uma carteira
- Estratégia é plugável (usa, não “é um tipo de”)
- Transação notifica investidor

Dá para implementar direto essa arquitetura.

✔️ 7. Está pronto para documentação do framework (API pública)

Você pode transformar cada bloco em uma seção:

- **mercadolab.market**
- **mercadolab.orderbook**
- **mercadolab.engine**
- **mercadolab.agents**
- **mercadolab.strategies**
- **mercadolab.runtime**

O UML já é a espinha dorsal da documentação.

🟢 Agora ao meu ver

O diagrama:

- reflete o que é um framework modular;
- separa domínio, agentes, microestrutura e runtime;
- removeu equívocos da UML anterior;
- permite extensões reais (plugins);
- pode ser transcrito 1:1 para Python de forma limpa;
- está totalmente alinhado com a proposta do capítulo 4, que ainda está em desenvolvimento.