

**MBA
USP
ESALQ**

Event Storming em Projetos de Software

Prof. Helder Prado Santos

*A responsabilidade pela idoneidade, originalidade e licitude dos conteúdos didáticos apresentados é do professor.

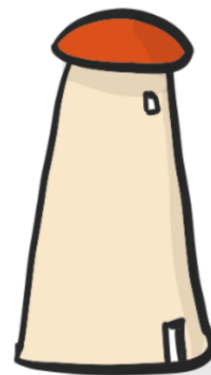
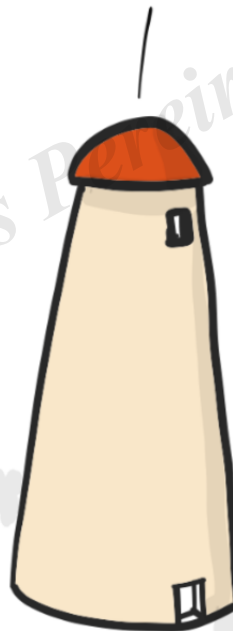
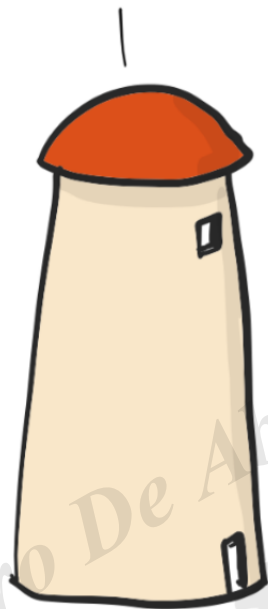
Proibida a reprodução, total ou parcial, sem autorização. Lei nº 9610/98

Os silos organizacionais

Ninguém conhece as nossas coisas
mais que a gente

Nós somos TOP!

Os outros silos
são uma bagunça

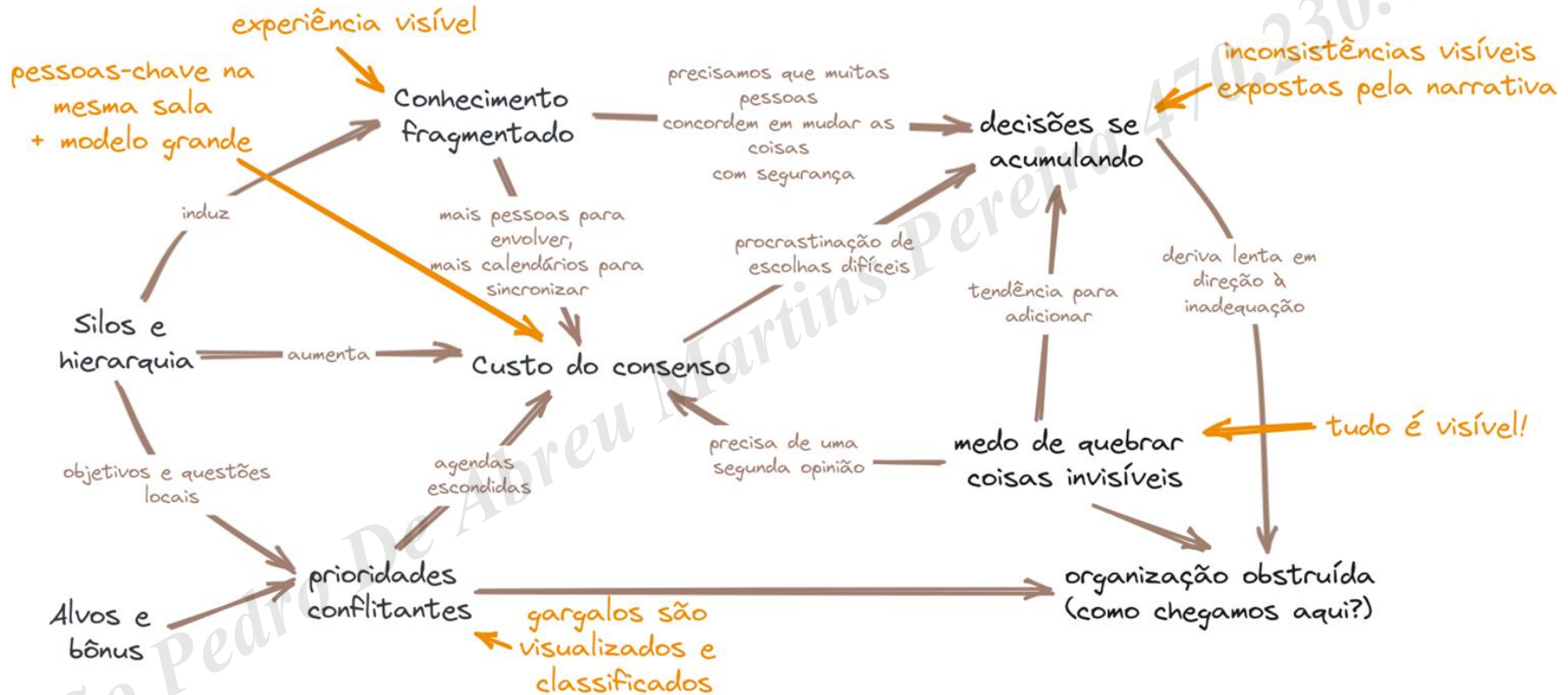


Não é nosso
problema, mesmo...

O problema do espaço organizacional

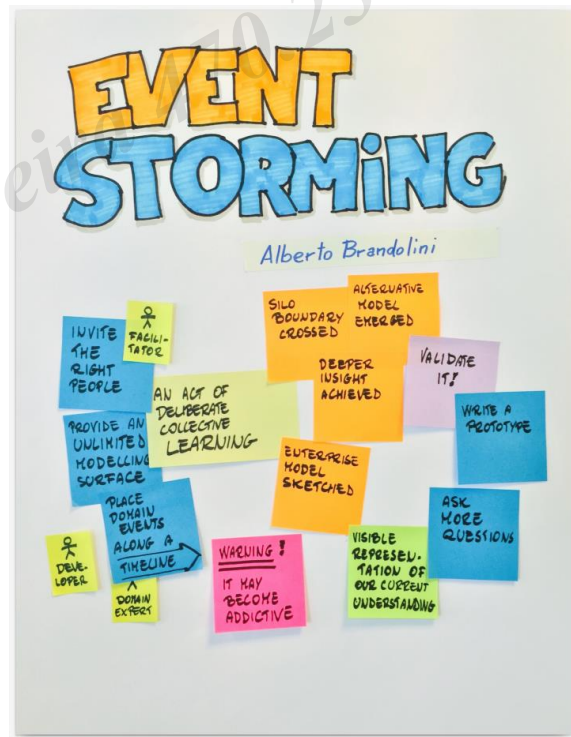


Como o Event Storming pode ajudar

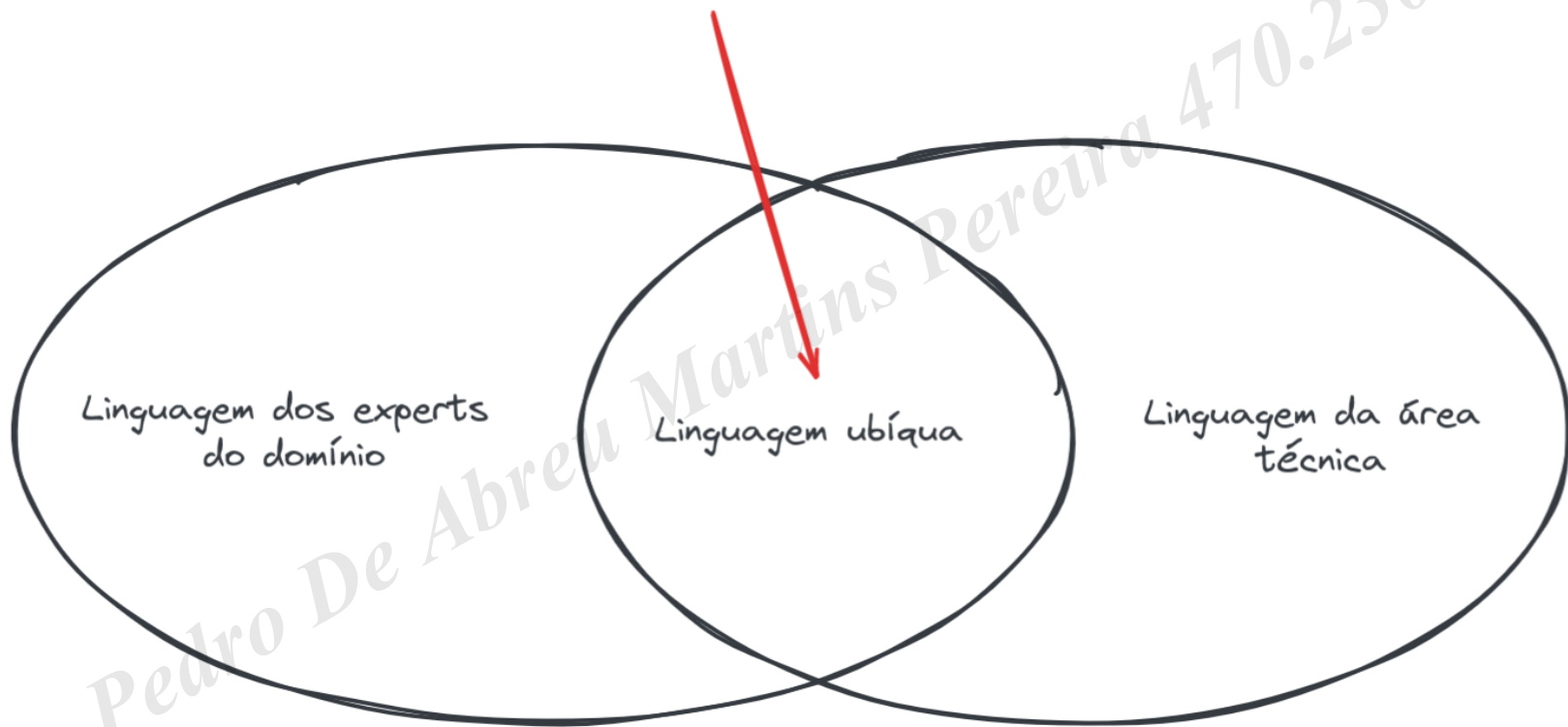


Conhecendo o Event Storming

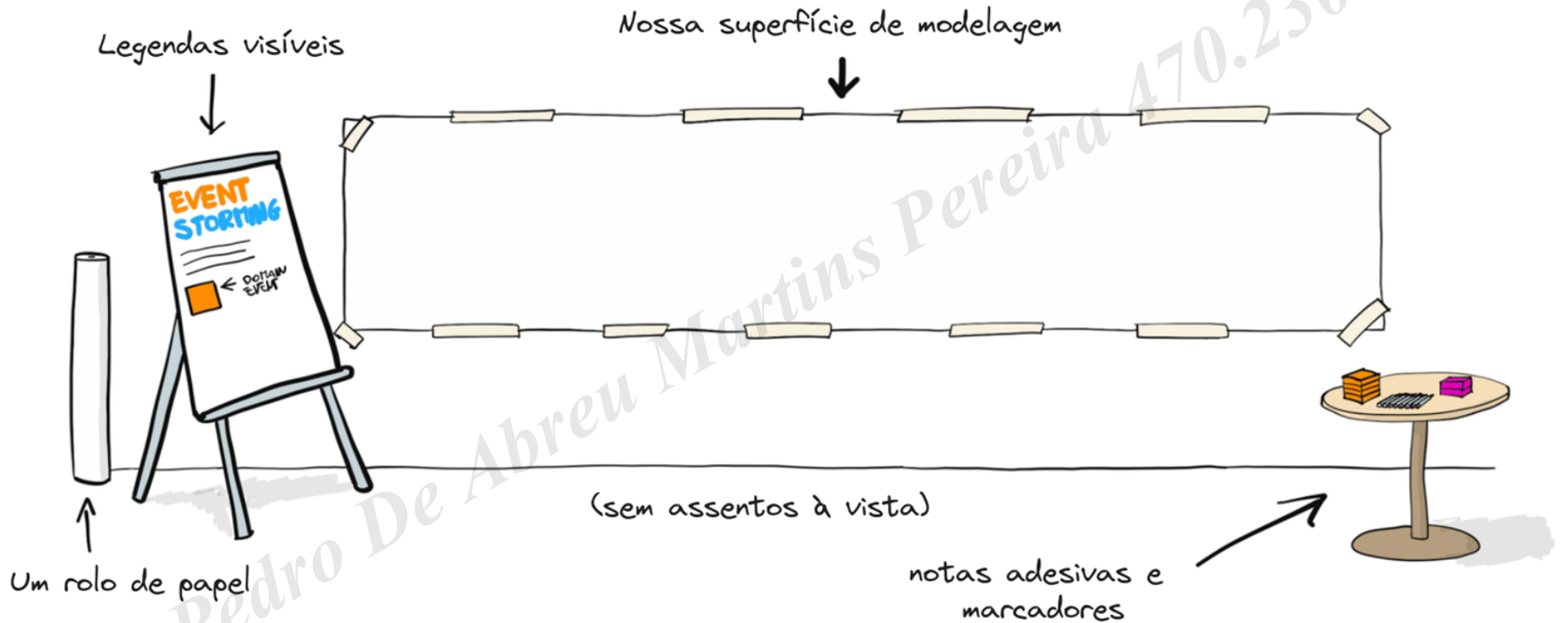
- ❑ Criado por **Alberto Brandolini**
- ❑ Técnica para **entendimento** do **domínio** das aplicações através dos seus **eventos**
- ❑ Realizado em workshops envolvendo **experts do domínio** e a **área técnica**



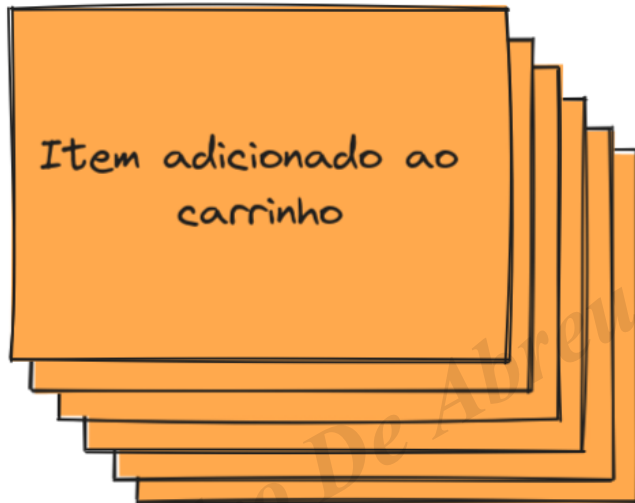
Desenvolvendo a linguagem ubíqua



O workshop



Mapeamento dos eventos do sistema

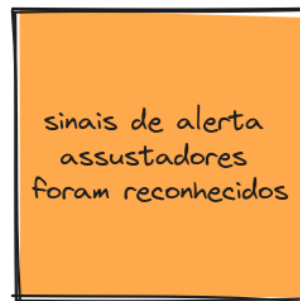
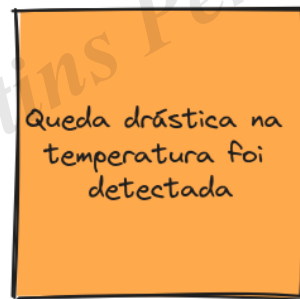
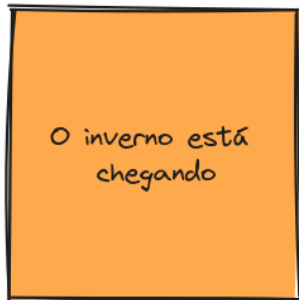


Evento de Domínio:

- ☐ Papel de cor **LARANJA**
- ☐ Verbo no passado
- ☐ Relevante para experts do domínio

Entendendo o que são eventos

Isso aqui não é um evento!

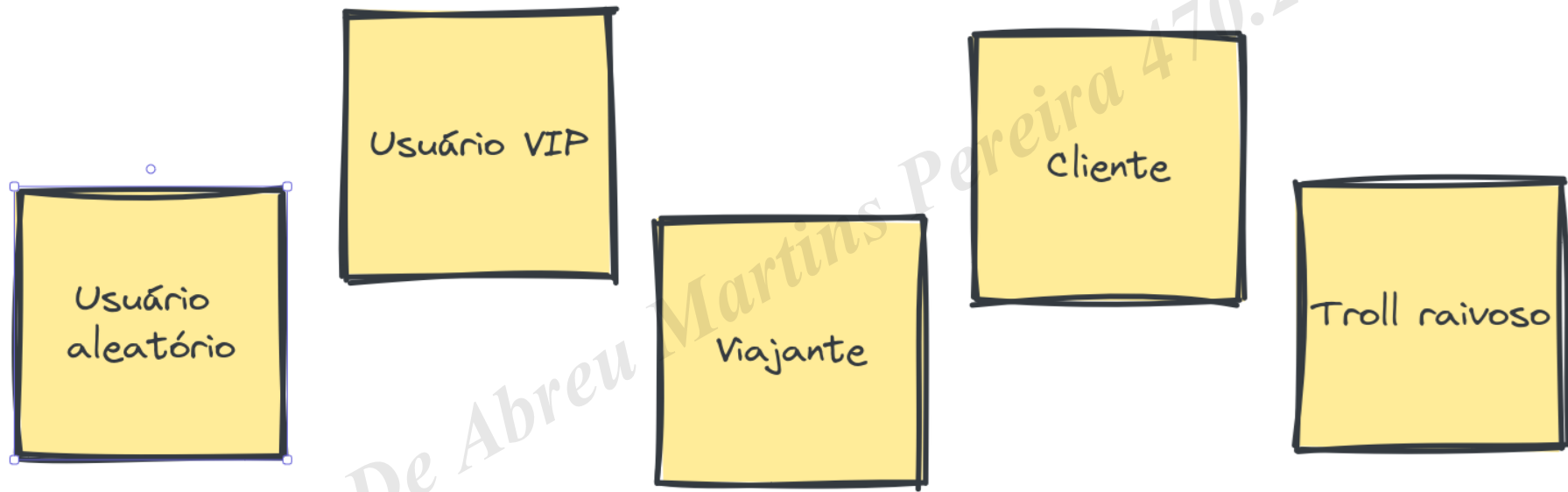


Esses aqui podem ser eventos

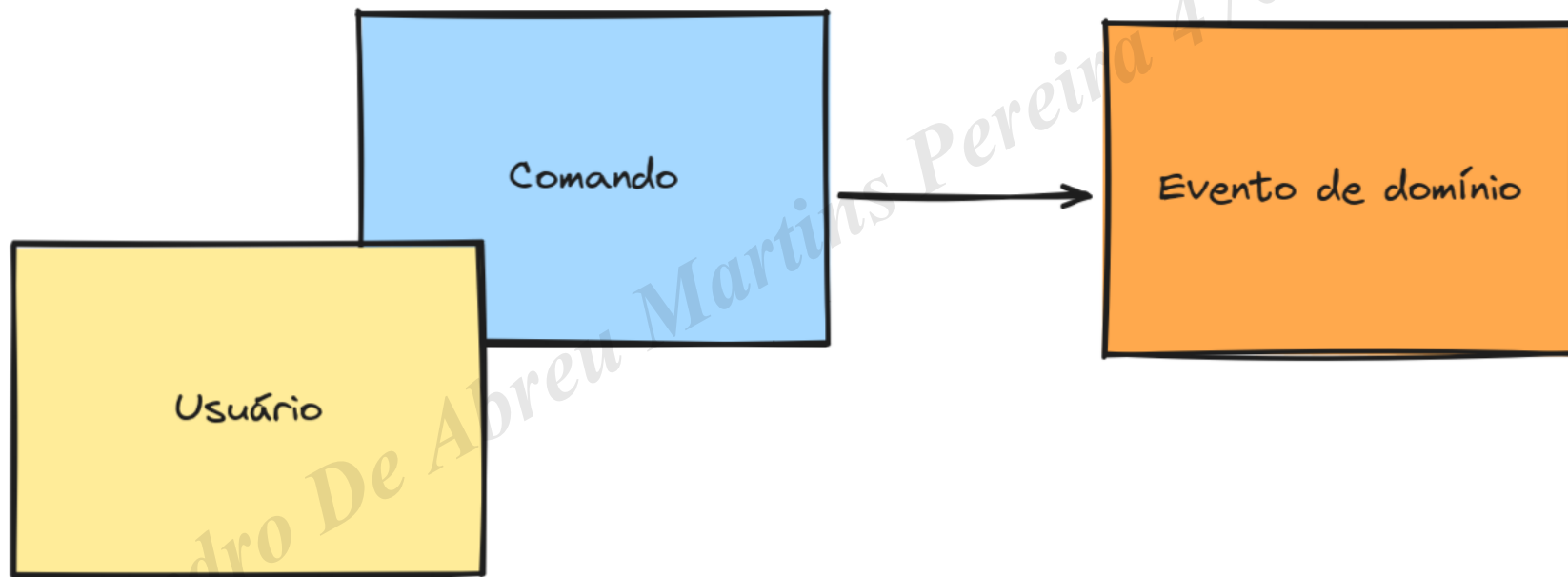
Alguns exemplos de eventos de domínio



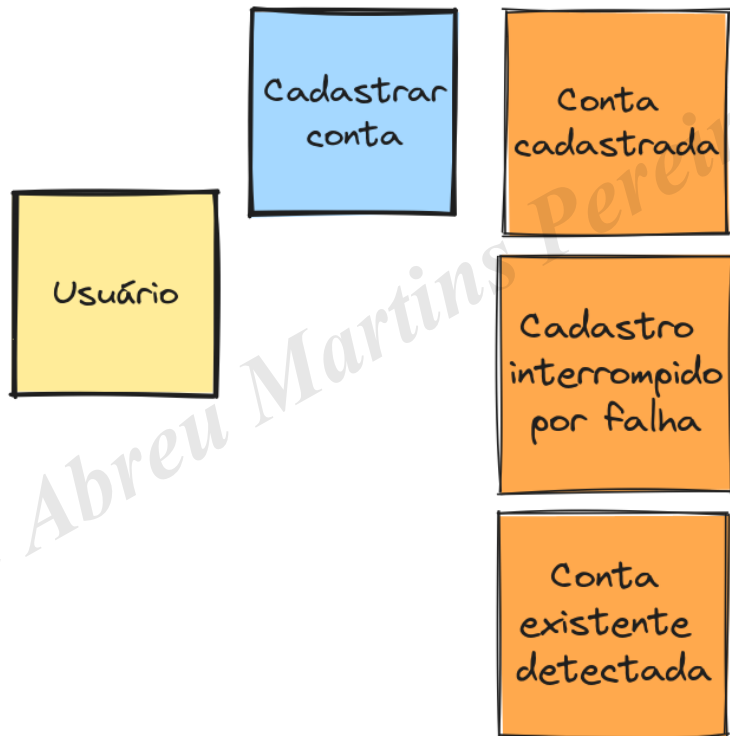
Bloco: Pessoa



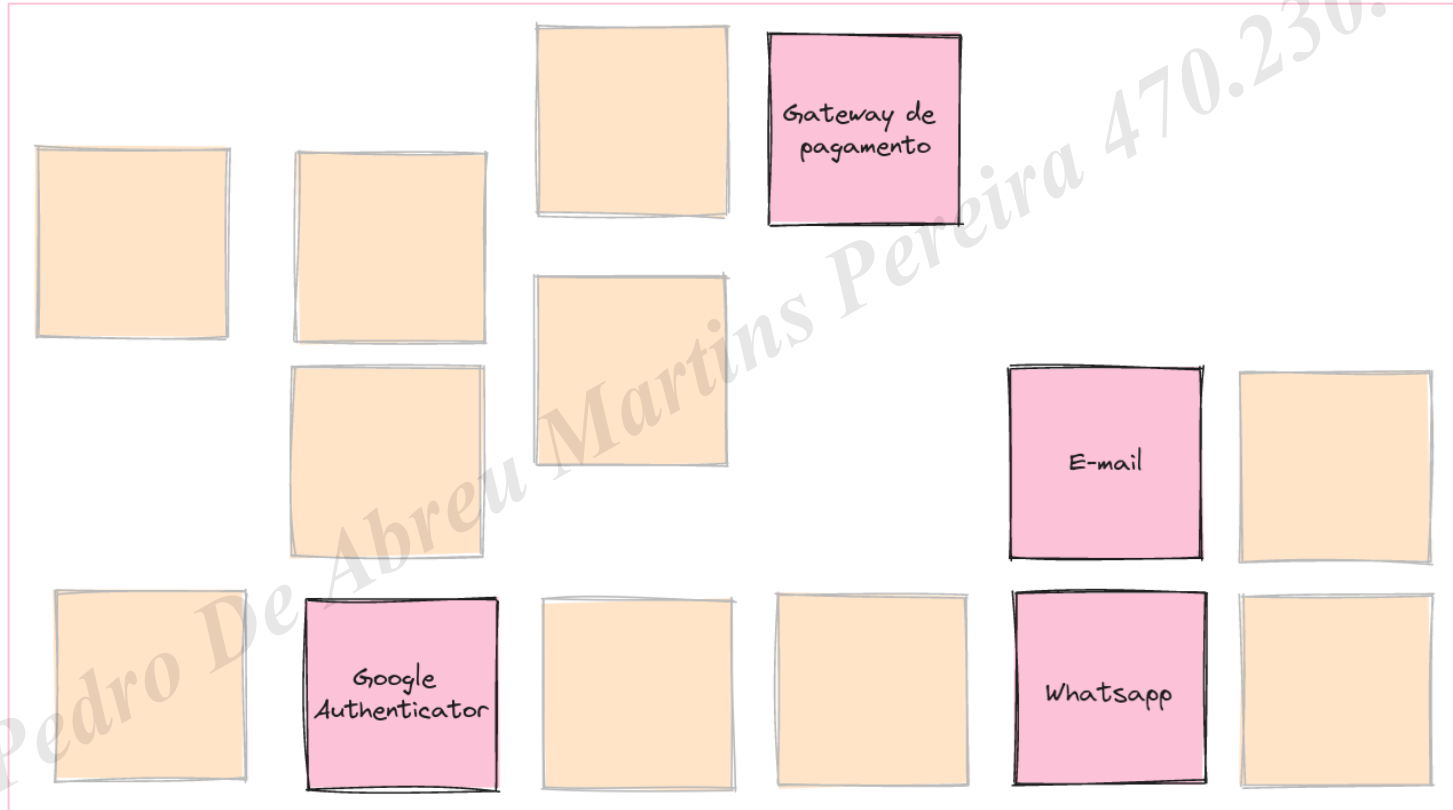
Bloco: Comando



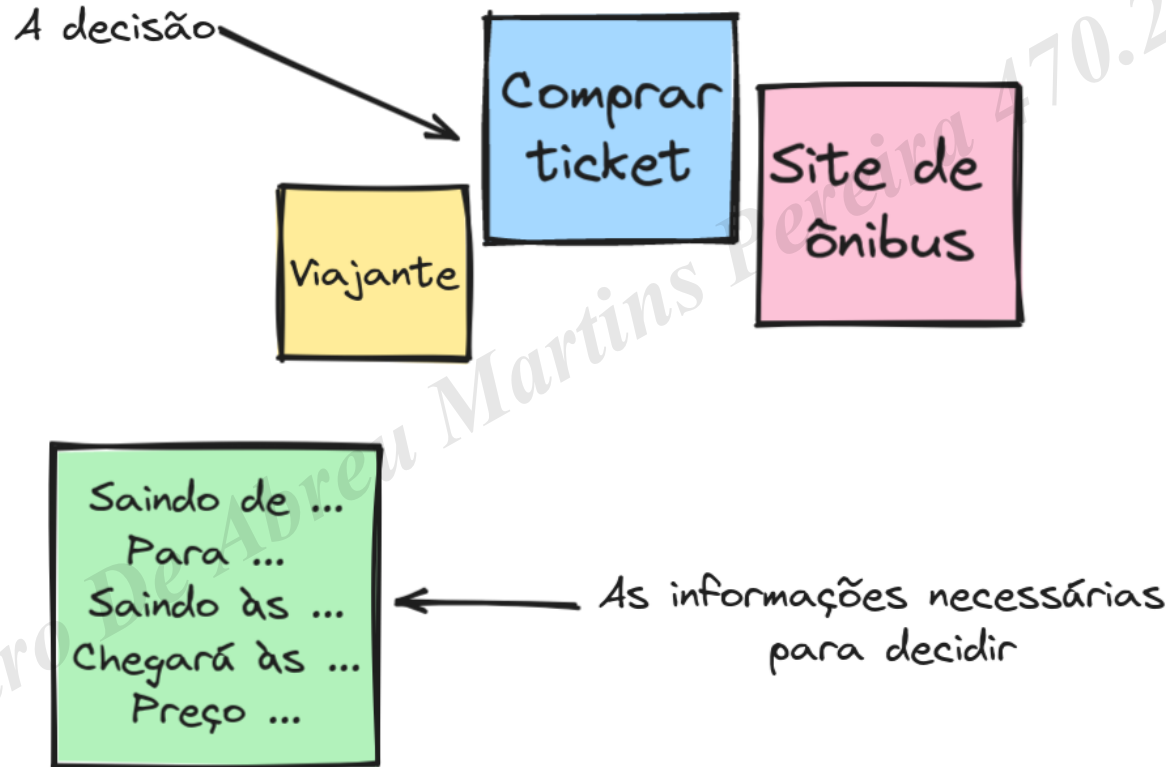
Múltiplos eventos para a mesma interação



Bloco: Sistemas externos



Bloco: Query Model / Informação



Bloco: Policy (Política)

Sempre que [evento(s)] então [comando(s)]

Cadastro finalizado

Política de boas-vindas

Enviar pacote de boas-vindas

Exemplo de policy para DEVs

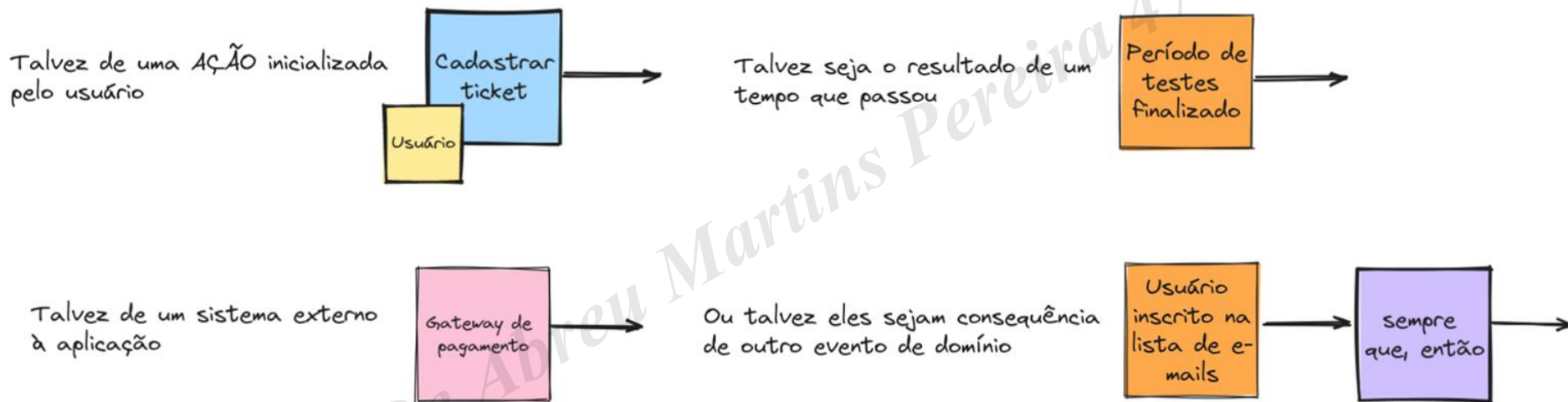
```
# Evento: "Assinatura ativada"
class EventoAssinaturaAtivada:
    def __init__(self, assinatura_id, usuario_email):
        self.assinatura_id = assinatura_id
        self.usuario_email = usuario_email
}

# Comando: "Enviar e-mail de boas-vindas"
def enviar_email_boas_vindas(email_cliente):
    # Lógica para enviar o email
    print(f"Email de boas-vindas enviado para {email_cliente}")

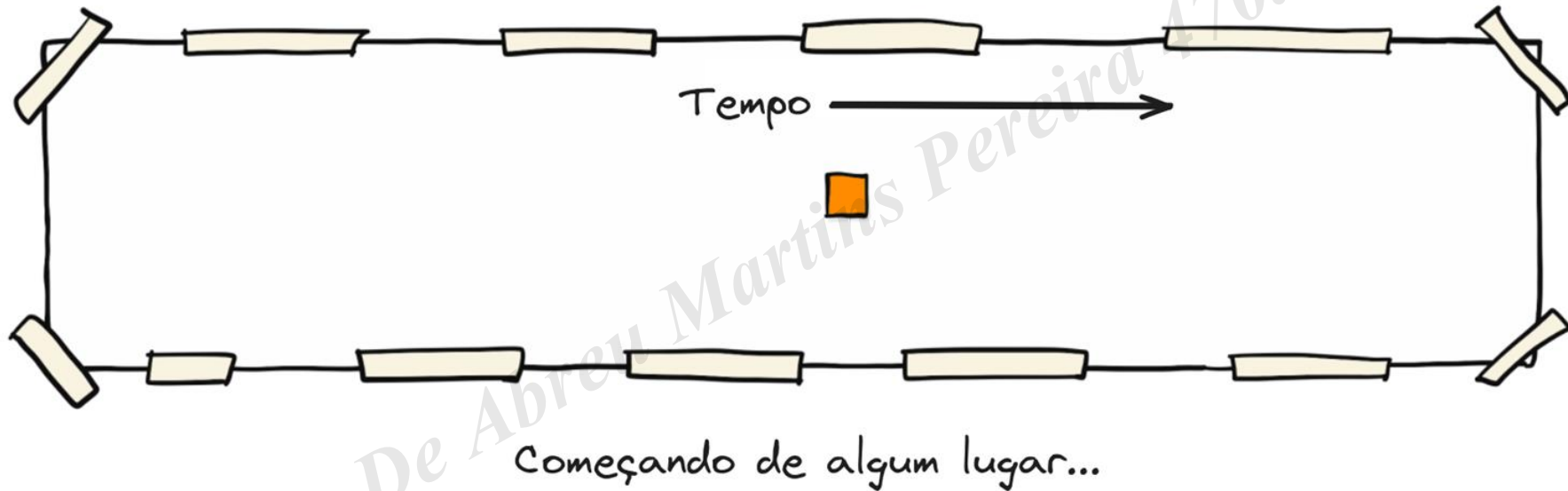
# Policy: "Política de boas-vindas"
def on_evento_assinatura_ativada(evento):
    enviar_email_boas_vindas(evento.usuario_email)

# Simulando o evento de ativação de assinatura
evento_ativacao = EventoAssinaturaAtivada(assinatura_id=101,
usuario_email="cliente@example.com")
# Policy reagindo ao evento de ativação de assinatura
on_evento_assinatura_ativada(evento_ativacao)
```

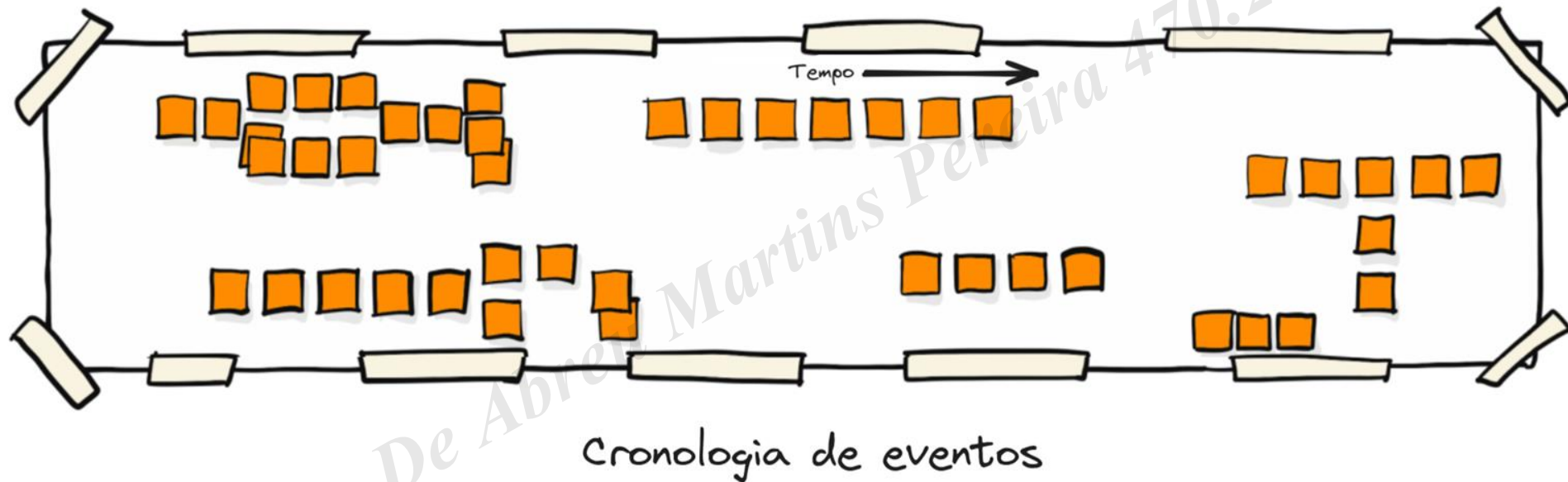

De onde vêm os eventos de domínio?



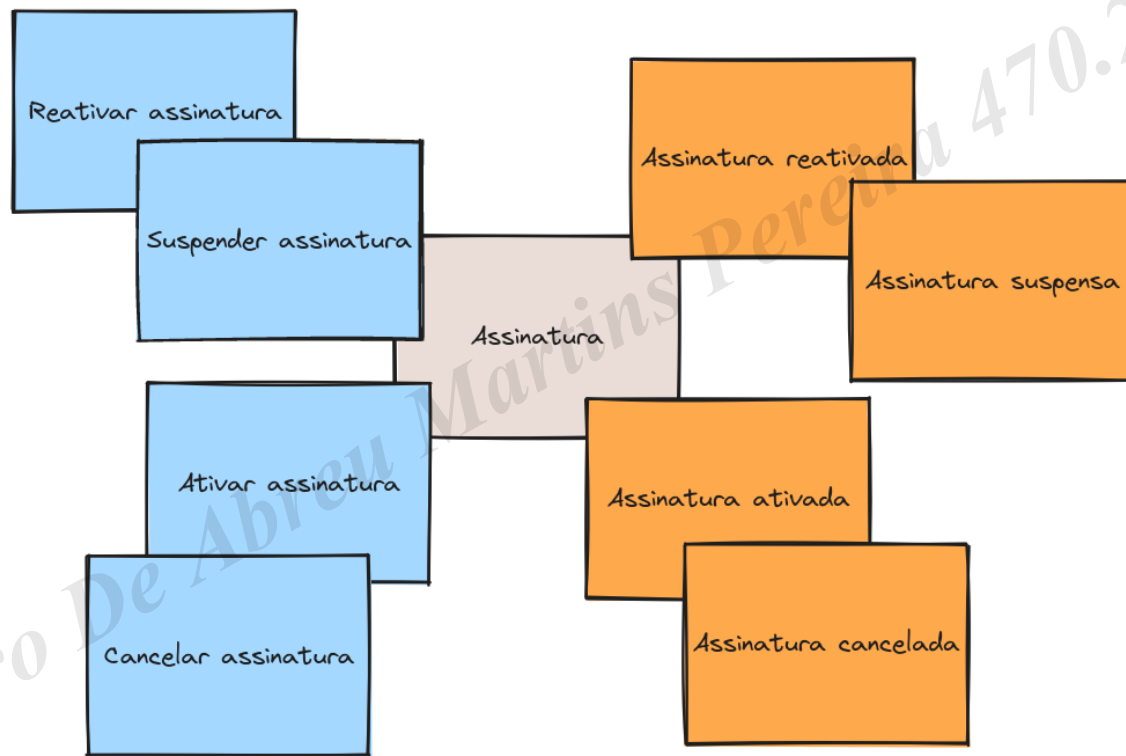
Cronologia dos eventos



Cronologia de eventos



Bloco: Agregados



Exemplo de agregados para DEVs

```
from datetime import datetime

class Usuario:
    def __init__(self, id, nome, email):
        self.id = id
        self.nome = nome
        self.email = email
        self.ativo = True
        self.data_criacao = datetime.now()
        self.data_atualizacao = datetime.now()

    def atualizar_nome(self, novo_nome):
        self.nome = novo_nome
        self.data_atualizacao = datetime.now()

    def atualizar_email(self, novo_email):
        self.email = novo_email
        self.data_atualizacao = datetime.now()

    def desativar(self):
        self.ativo = False
        self.data_atualizacao = datetime.now()
```


Casos de usos e possíveis eventos

```
usuario = Usuario(id=1, nome="João Silva", email="joao.silva@example.com")
print(f"Usuário criado: {usuario.nome}, {usuario.email}, Ativo: {usuario.ativo}")

# Atualizar nome do usuário
usuario.atualizar_nome("João Pereira")
print(f"Nome atualizado: {usuario.nome}")

# Atualizar email do usuário
usuario.atualizar_email("joao.pereira@example.com")
print(f"Email atualizado: {usuario.email}")

# Desativar usuário
usuario.desativar()
print(f"Usuário desativado: Ativo: {usuario.ativo}")
```

Ex.: Agregado

Assinatura

```
from datetime import datetime, timedelta

class Assinatura:
    def __init__(self, id, usuario_id, data_inicio, duracao_meses):
        self.id = id
        self.usuario_id = usuario_id
        self.data_inicio = data_inicio
        self.duracao_meses = duracao_meses
        self.data_fim = self.calcular_data_fim()
        self.ativa = True
        self.data_cancelamento = None

    def calcular_data_fim(self):
        return self.data_inicio + timedelta(days=30 * self.duracao_meses)

    def renovar(self, meses_adicionais):
        if self.ativa:
            self.duracao_meses += meses_adicionais
            self.data_fim = self.calcular_data_fim()
            print(f"Assinatura renovada até {self.data_fim}")
        else:
            print("Não é possível renovar uma assinatura inativa.")

    def cancelar(self):
        self.ativa = False
        self.data_cancelamento = datetime.now()
        print(f"Assinatura cancelada em {self.data_cancelamento}")

    def ativar(self):
        if not self.ativa:
            self.ativa = True
            self.data_fim = self.calcular_data_fim()
            self.data_cancelamento = None
            print(f"Assinatura reativada até {self.data_fim}")
        else:
            print("A assinatura já está ativa.")
```

Ex.: Agregado Assinatura

```
# Criar uma nova assinatura
data_inicio = datetime.now()
assinatura = Assinatura(id=1, usuario_id=123, data_inicio=data_inicio, duracao_meses=6)
print(f"Assinatura criada: {assinatura.data_inicio} até {assinatura.data_fim},
      Ativa: {assinatura.ativa}")

# Renovar assinatura por mais 3 meses
assinatura.renovar(3)

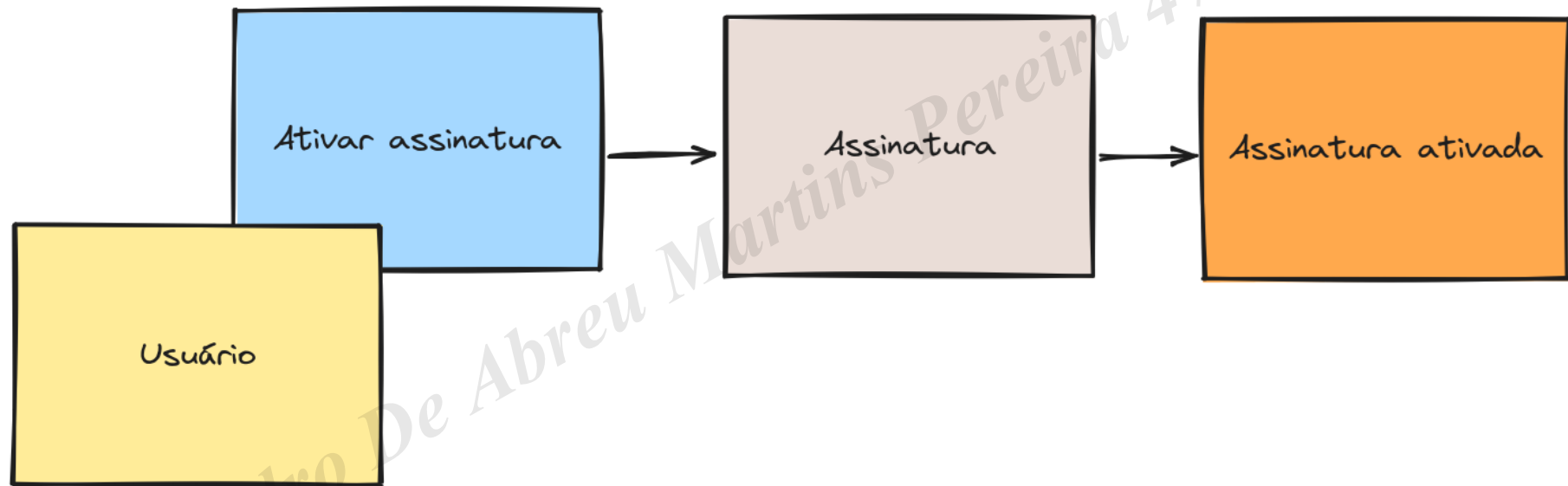
# Cancelar assinatura
assinatura.cancelar()

# Tentar renovar assinatura cancelada
assinatura.renovar(2)

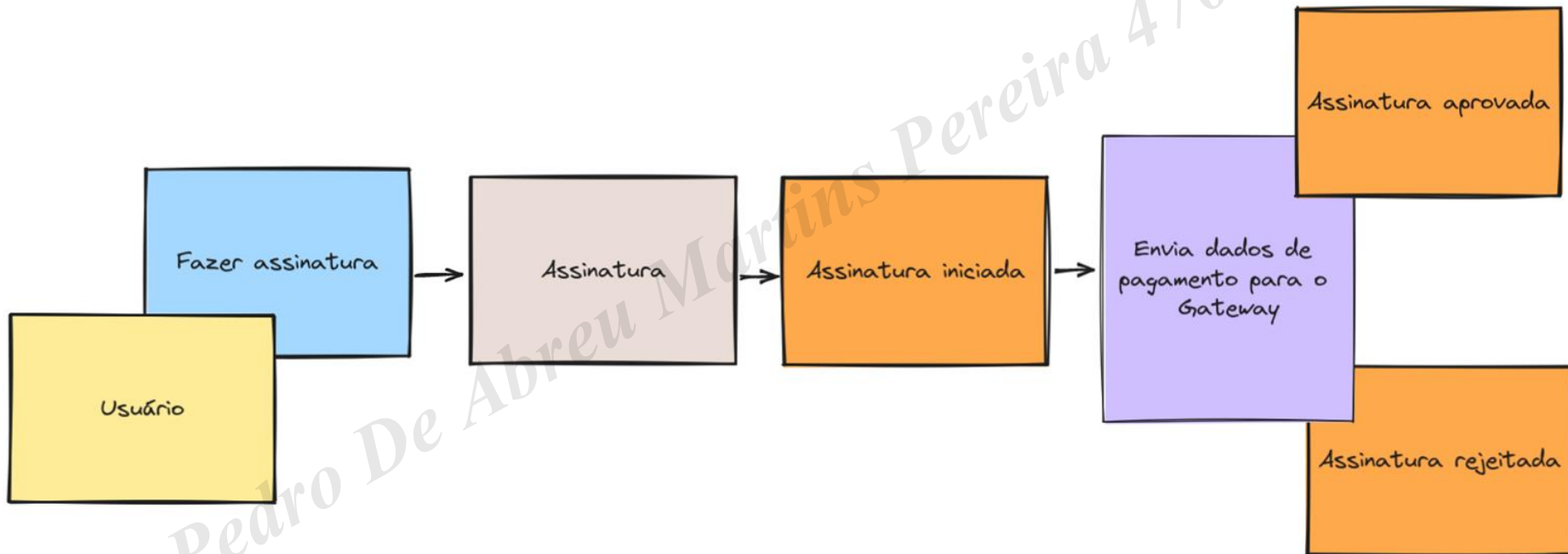
# Ativar assinatura novamente
assinatura.ativar()

# Renovar assinatura reativada
assinatura.renovar(2)
```

Visão do exemplo no Event Storming

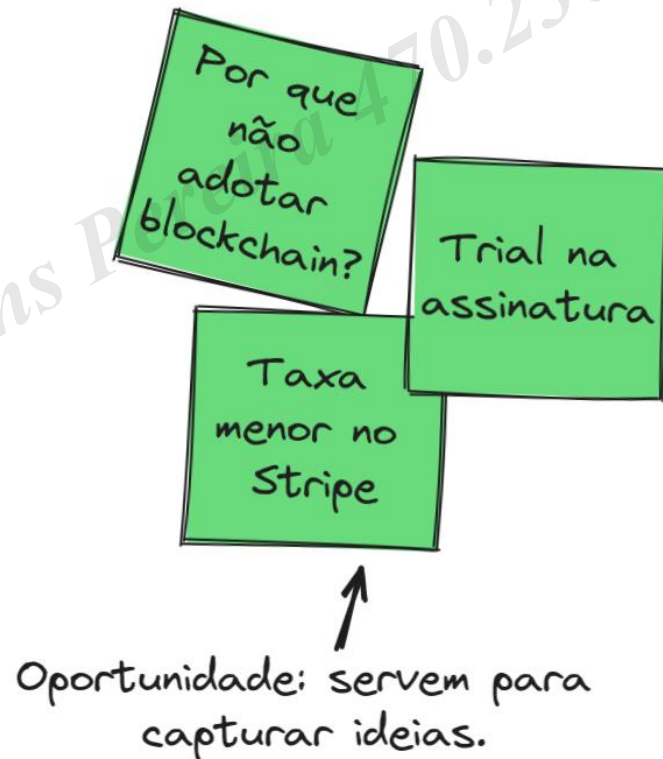


... algo mais próximo da realidade



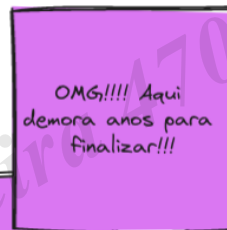
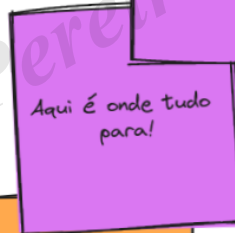
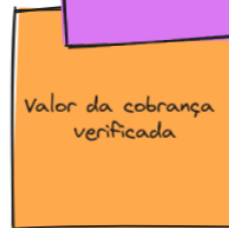
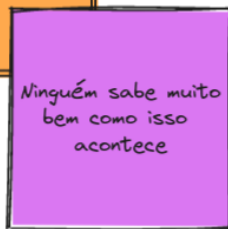
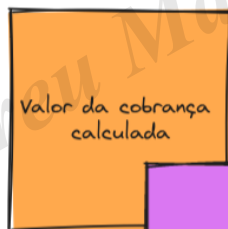
Blocos: Hotspots e Oportunidades

Hotspots: servem para mapear problemas, riscos, etc.

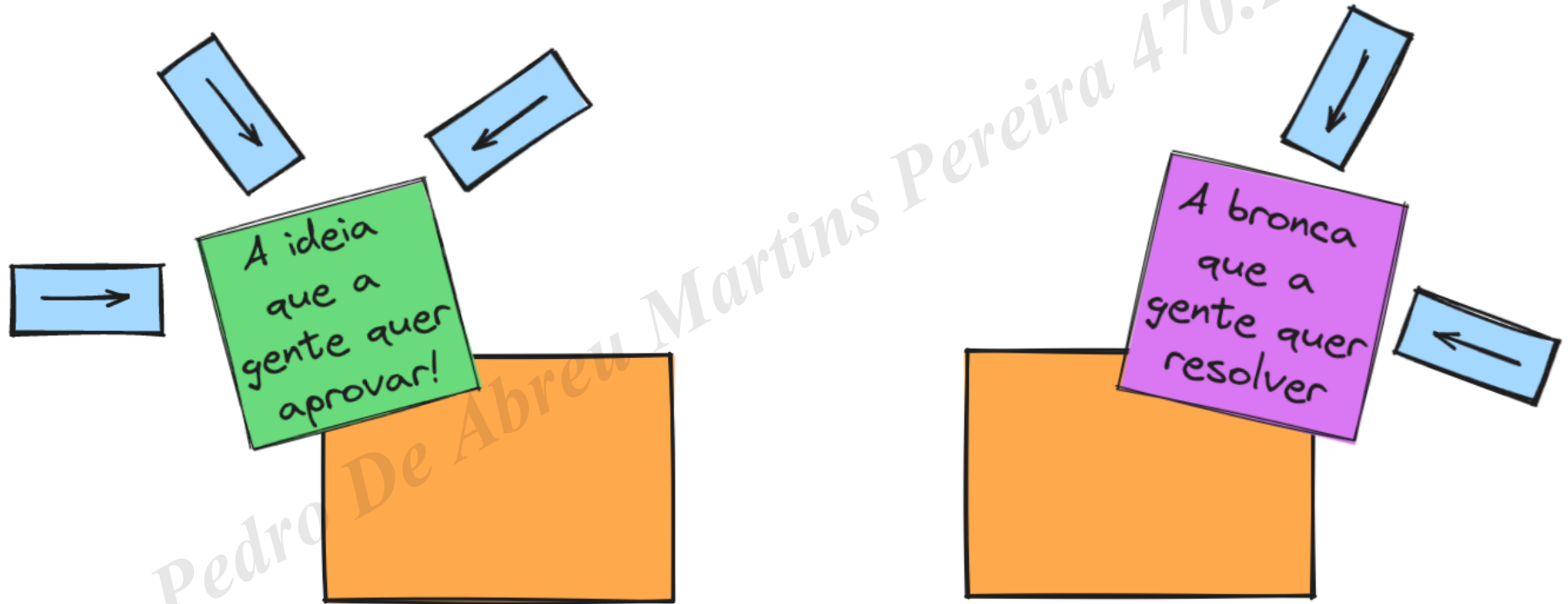


Utilizando Hotspots

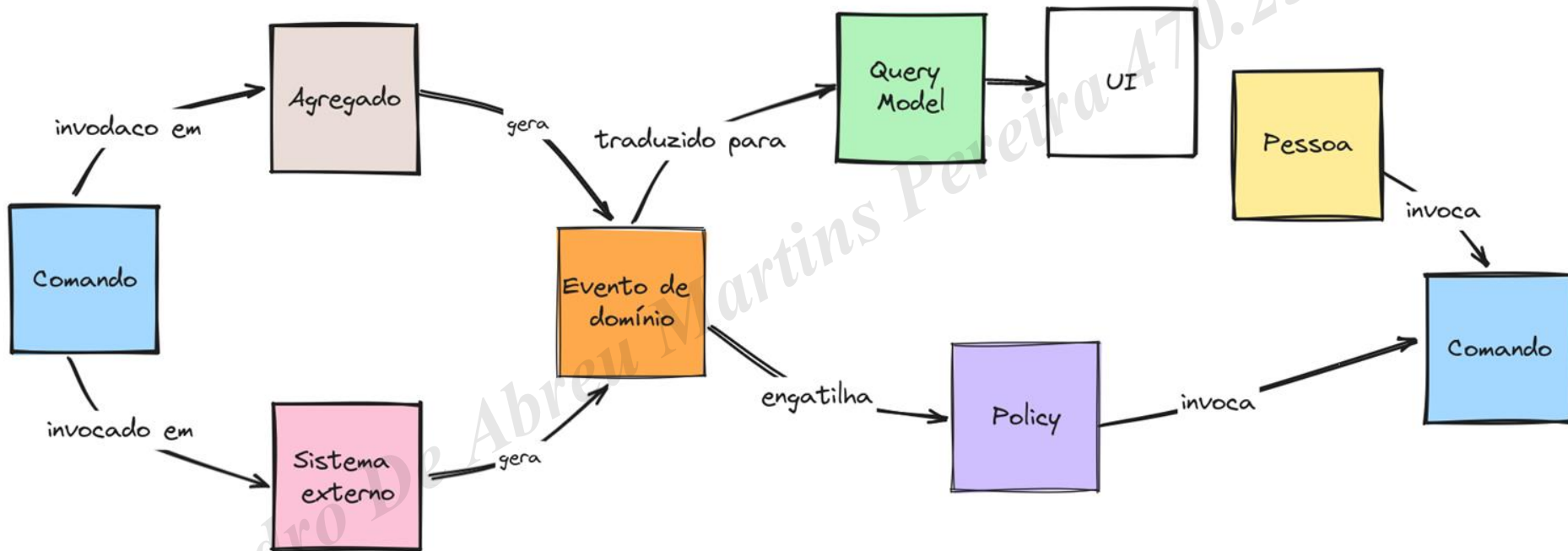
Eventos acionados
por tempo



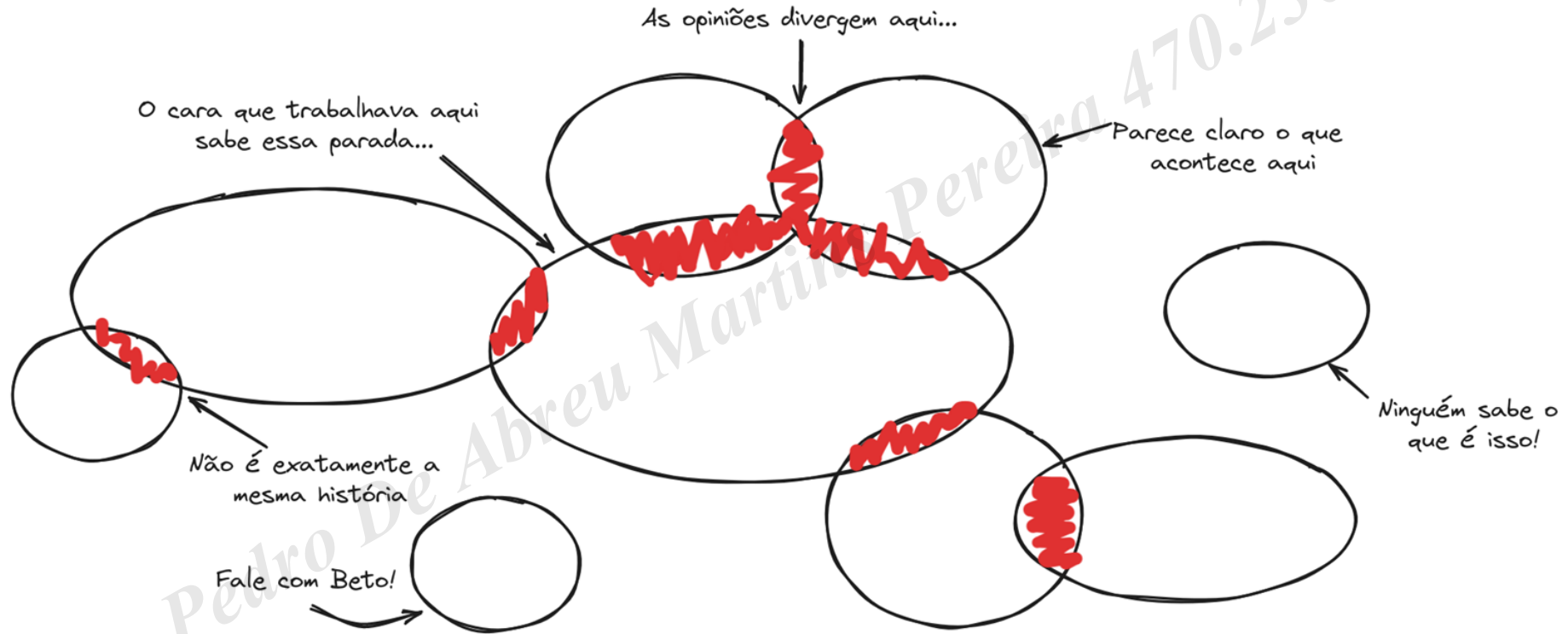
Bloco: Arrow Voting



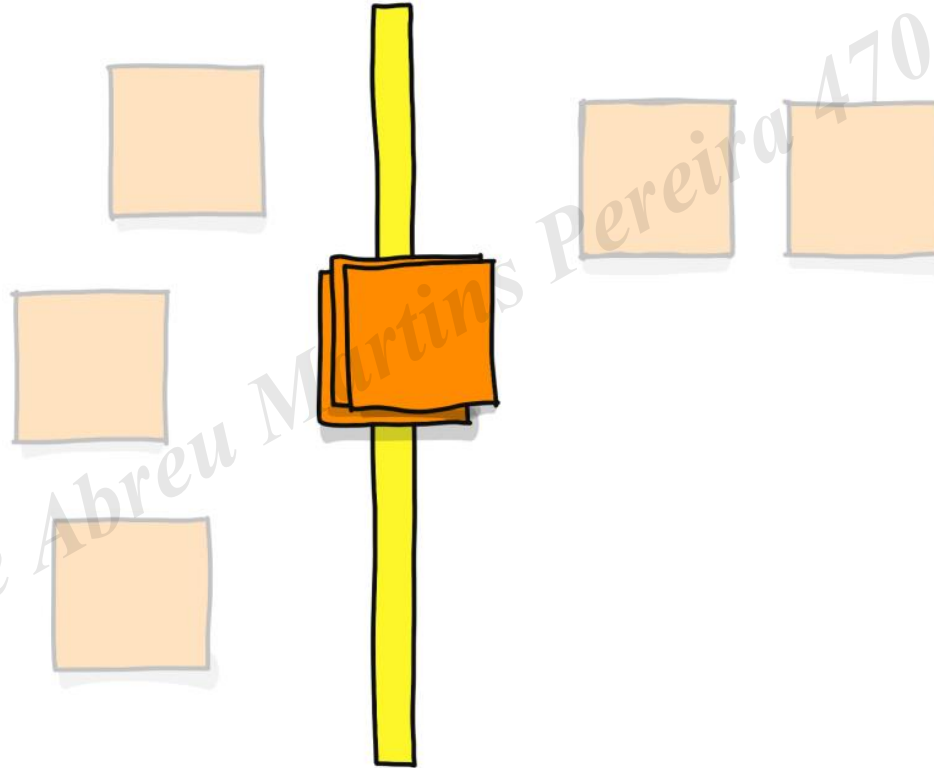
“A figura que explica todas as coisas”



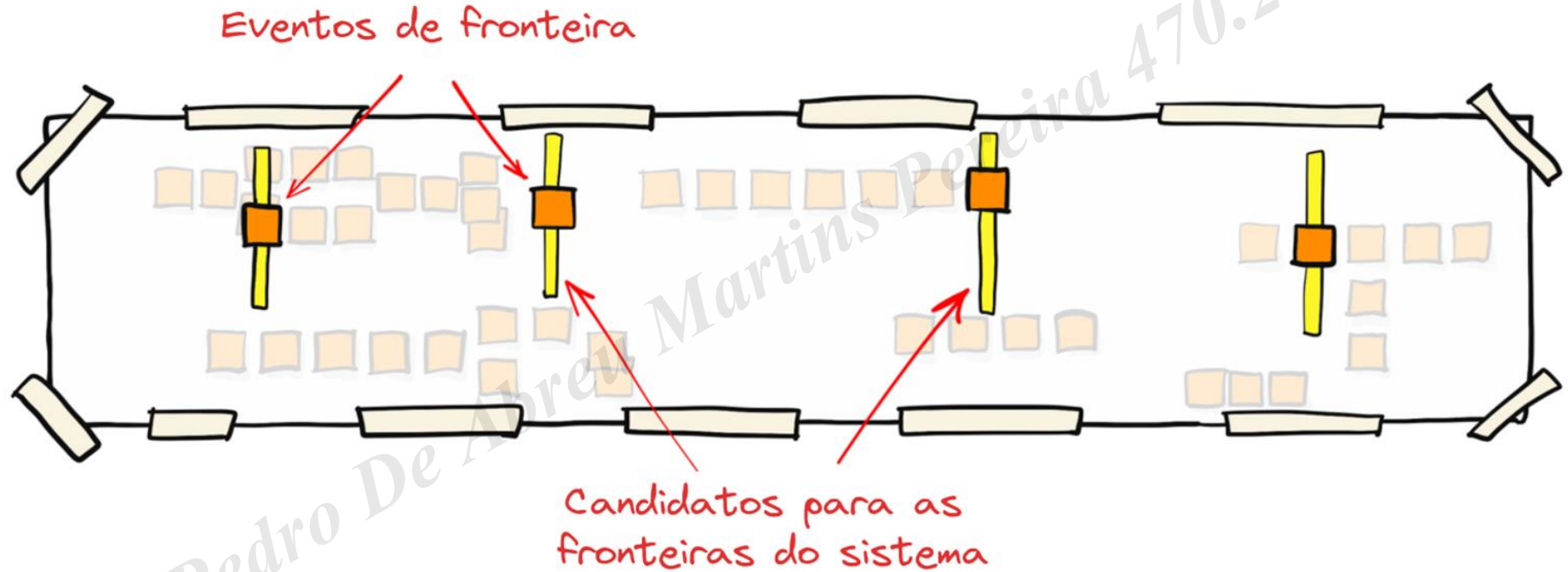
Bounded Contexts



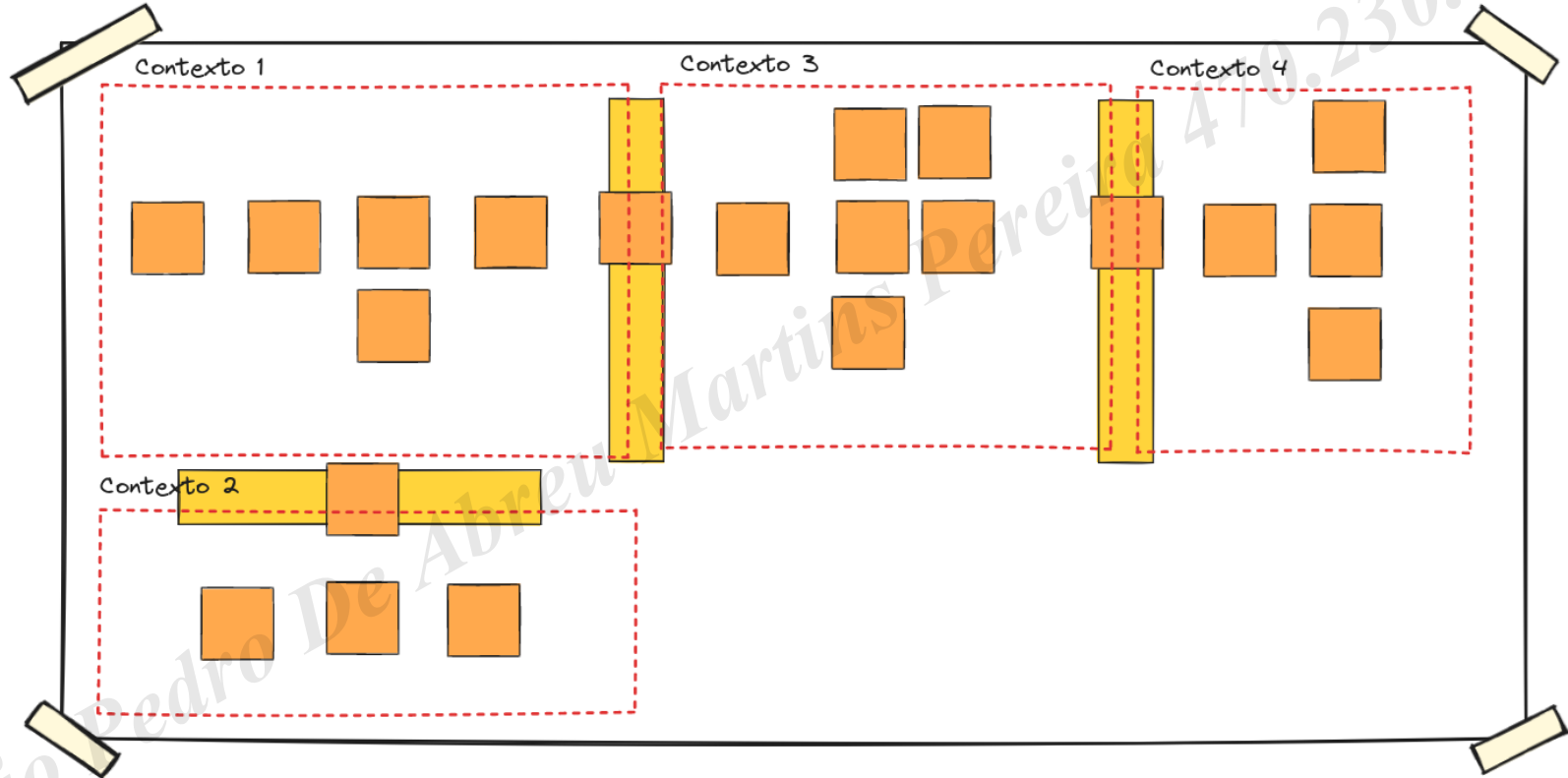
Eventos de fronteira



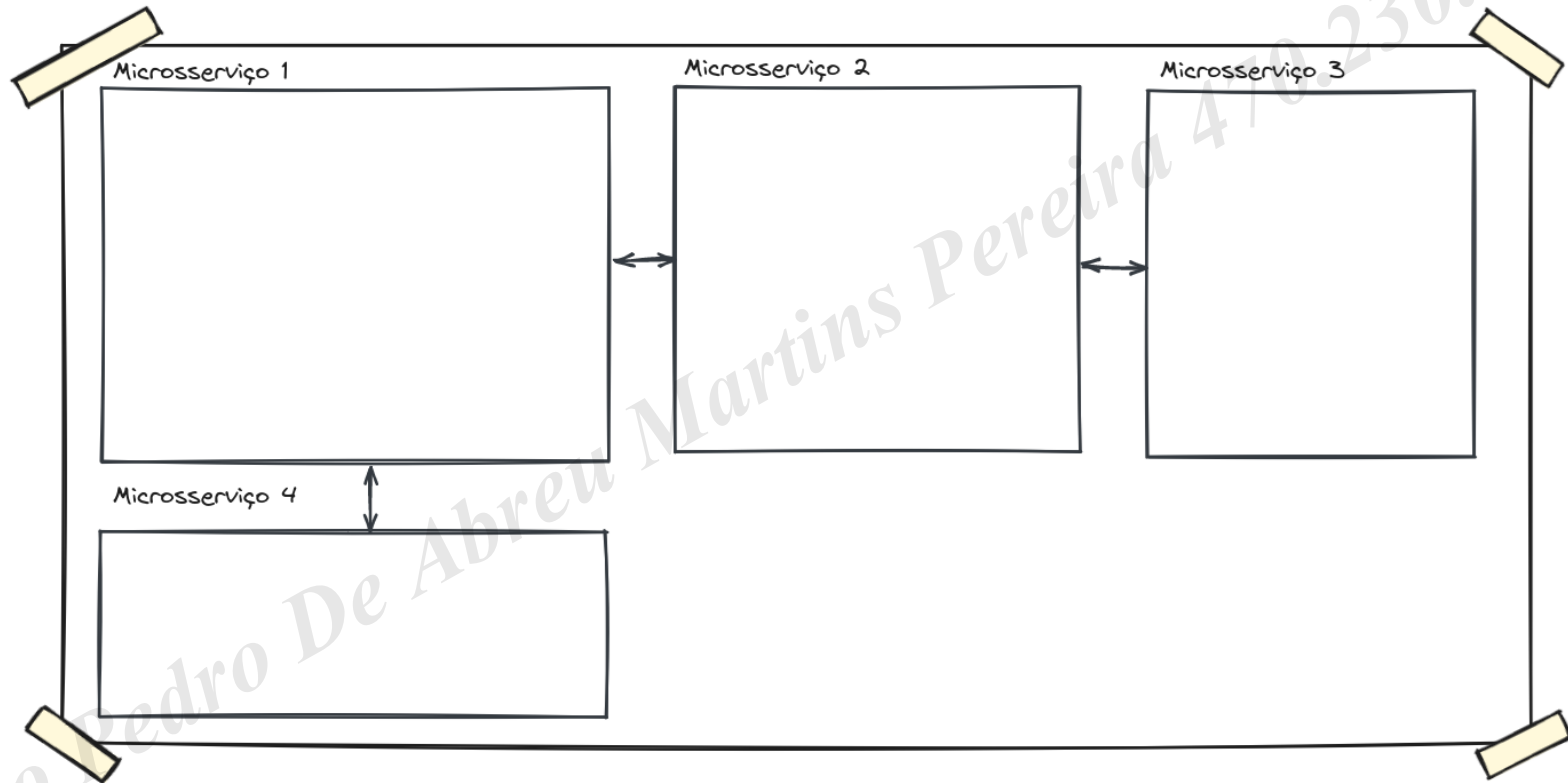
Descobrimo Bounded Contexts



Visualizando os Contextos



Visualizando possíveis microserviços



Revisando os blocos do Event Storming





Mãos à obra!!!



Exportar, preferências, idiomas...



Alt

Para mover a tela, segure a roda do mouse ou a barra de espaço enquanto arrasta ou use a ferramenta de mão

Compartilhar

Biblioteca

Escolha uma ferramenta e comece a desenhar!



Todos os dados são salvos localmente no seu navegador.

Abrir

Ctrl+O

Ajudar

?

Colaboração ao vivo...

Sign up

Atalhos e ajuda



<https://excalidraw.com/>

Referências

1. BRANDOLINI, A. Introducing EventStorming. [s.l.] Leanpub, 2015.
2. VLAD KHONONOV. Learning Domain-Driven Design. [s.l.] “O’Reilly Media, Inc.”, 2021.

OBRIGADO!

[linkedin.com/in/helderprado](https://www.linkedin.com/in/helderprado)