1º EXEMPLO

PONTE

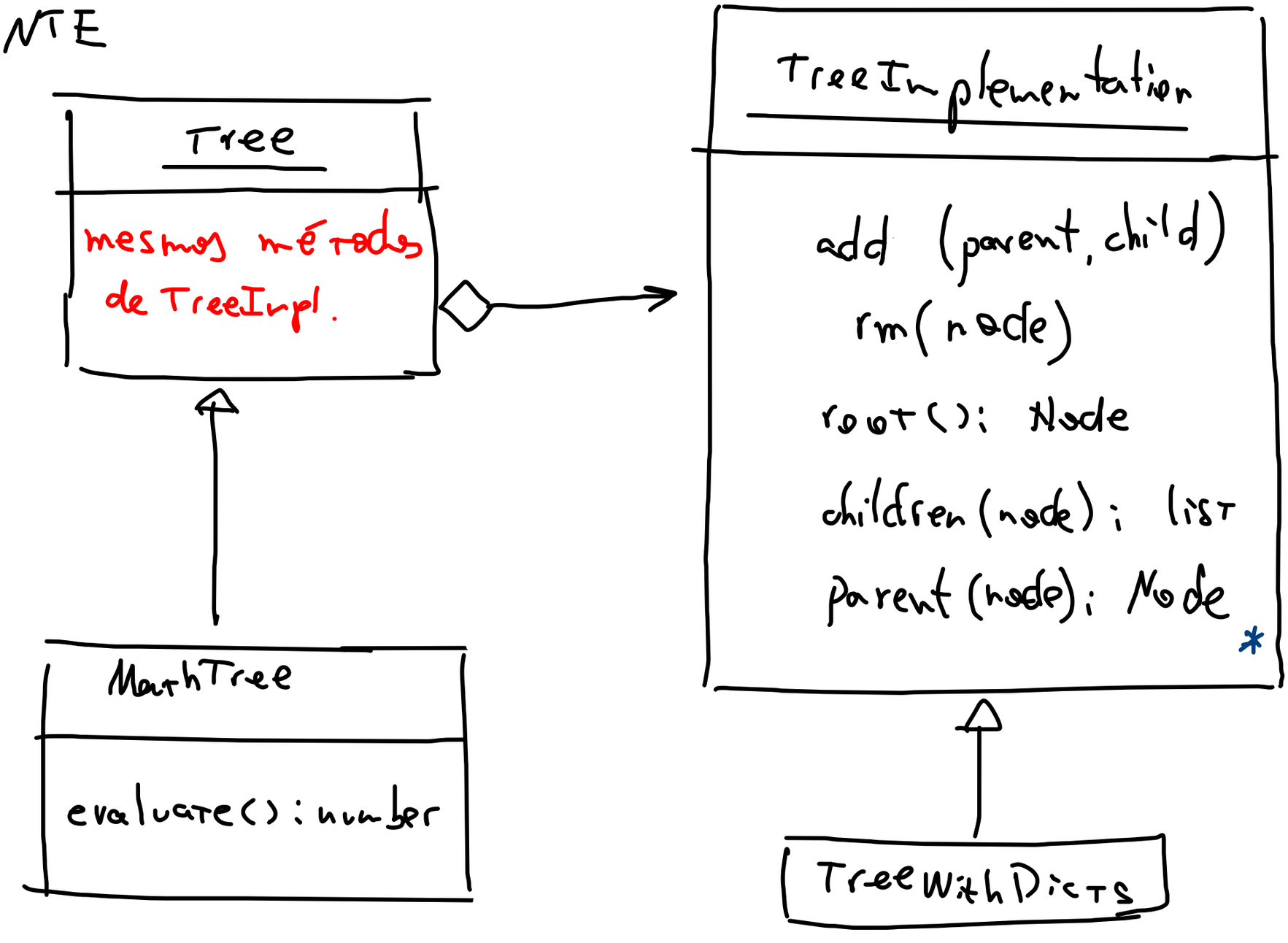
Como primeiro exemplo (Ponte), considere o tópico de árvores com duas dimensões de funcionalidade:

1) Podem ser implementadas de várias maneiras, apesar de disponibilizarem uma interface comum (dicionários, tuplas, composição, etc.)

2) Podem representar várias categorias de objeto: pode ser uma árvore de expressão matemática, uma árvore representando uma expressão regular (regexp), uma árvore de sintaxe abstrata (AST), etc.

A Ponte separa cada dimensão de funcionalidade numa hierarquia de herança individual:

PONTE



*: A interface Node é arbitrária. Trata-se de um "parâmetro" de TreeImplementation no estilo dos "generics" de Java.

2º exemplo

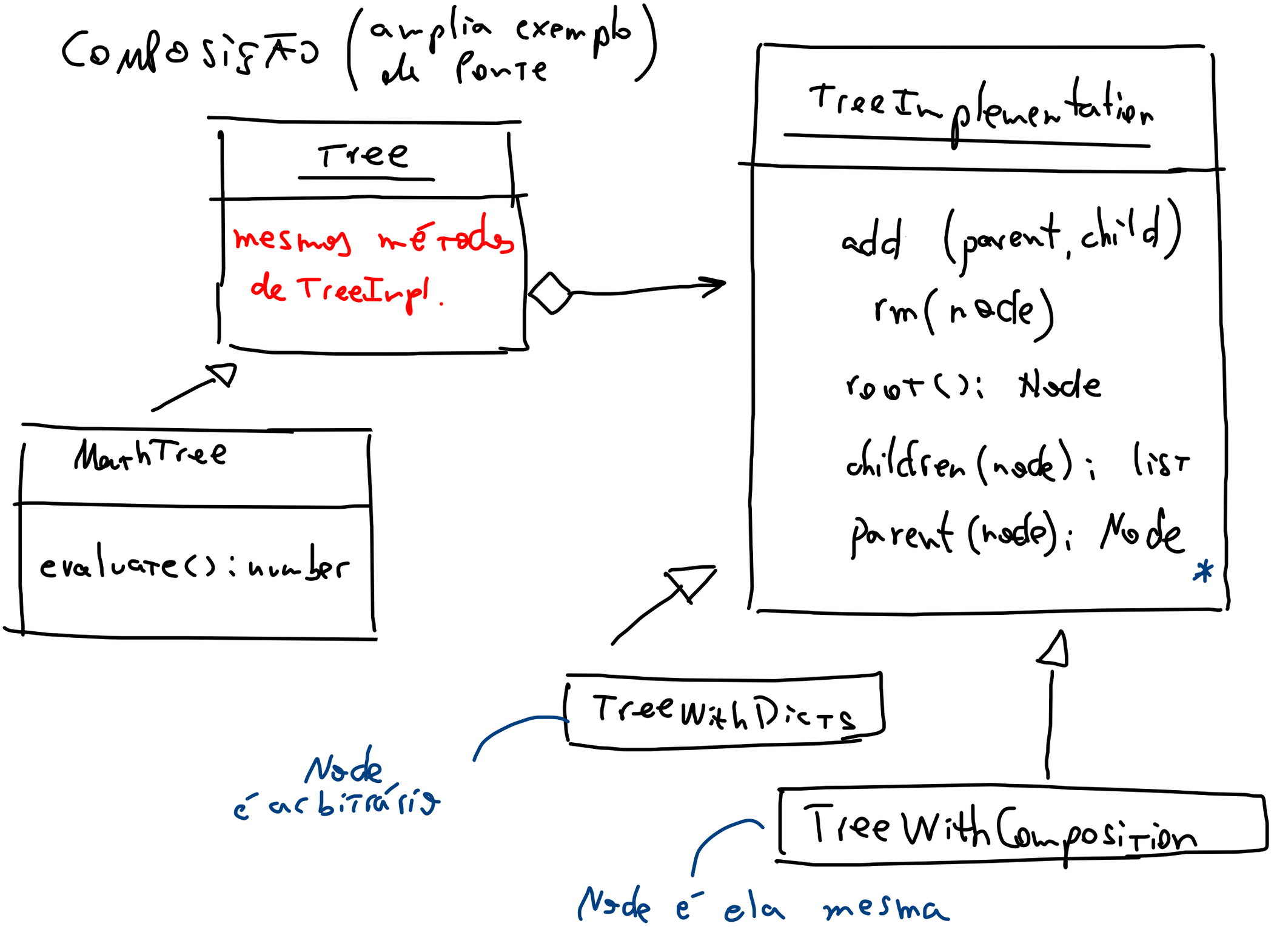
Composição

Eu disse que árvores podem ser implementadas por composição, então ampliemos o exemplo anterior com uma nova subclasse de TreeImplementation:

TreeWithComposition.

Esta nova classe vai exigir que o parâmetro Node (explicado em azul) seja a própria TreeWithComposition (porque composição é recursiva. O novo diagrama é:

Composição (amplia exemplo da Ponte)



Exemplo 3:

MEDIADOR

com

FÁBRICA ABSTRATA

com

ESTADO

com

DECORADOR

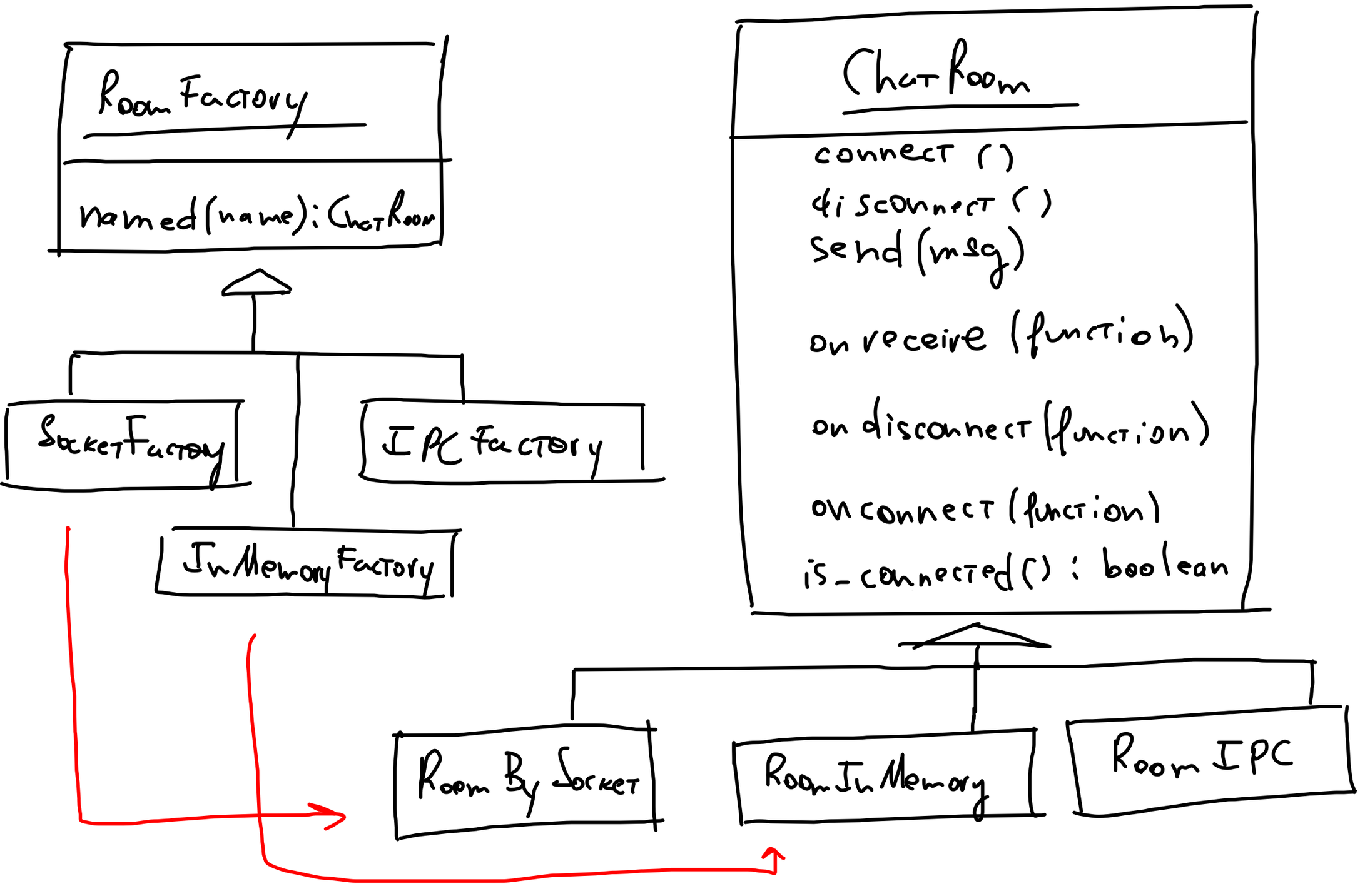
O exemplo é: clientes podem passar e receber mensagens de outros clientes em "canais" de chat.

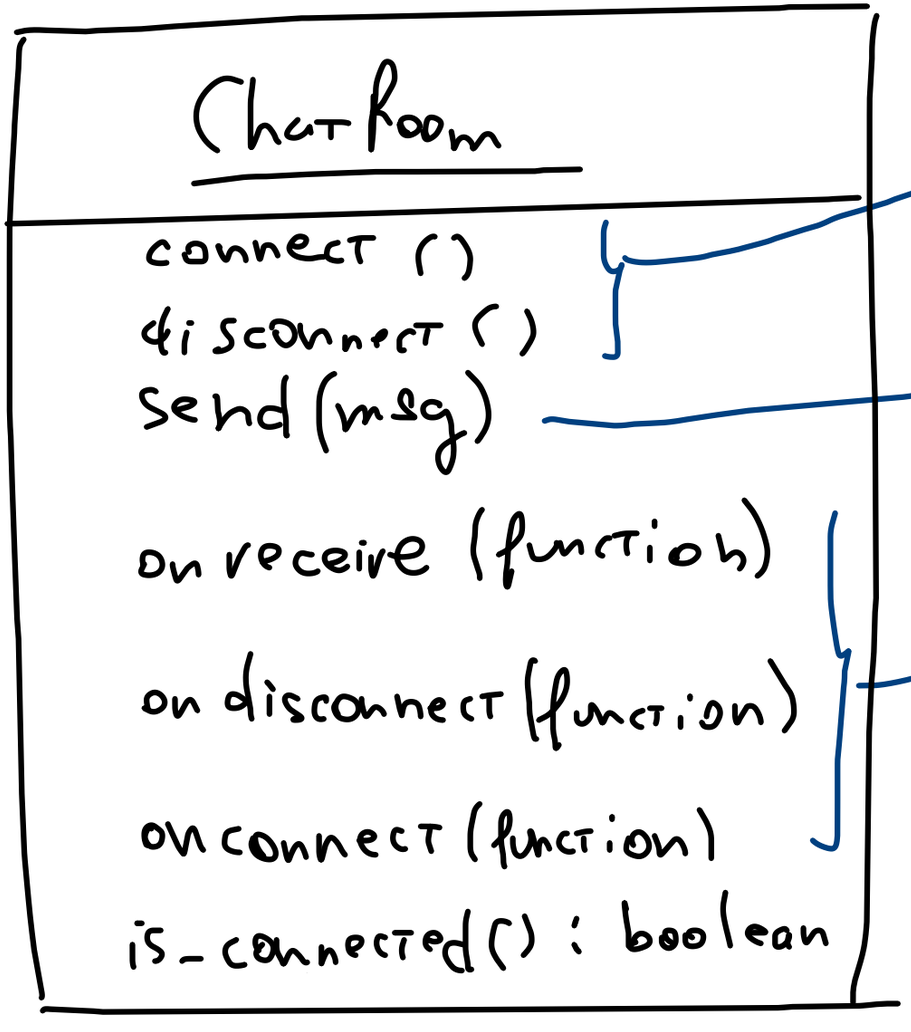
Esses canais podem ser via SOCKET, ou podem ser de um processo a outro (IPC), ou ainda podem ser de um objeto a outro dentro do mesmo programa ("in memory").

A FÁBRICA ABSTRATA consiste em alternar entre essas 3 plataformas sem que o cliente perceba a diferença.

O MEDIADOR está no fato de que, no caso dos canais "in memory", cada cliente poderia falar com os outros diretamente, o que vai ser evitado pela mediação da classe ChatRoom.

O diagrama parcial fica:


RoomFactory
named(name): ChatRoom
SocketFactory
IPCFactory
InMemoryFactory
ChatRoom
connect()
disconnect()
send(msg)
onreceive(function)
ondisconnect(function)
onconnect(function)
is_connected(): boolean
RoomBySocket
RoomInMemory
RoomIPC

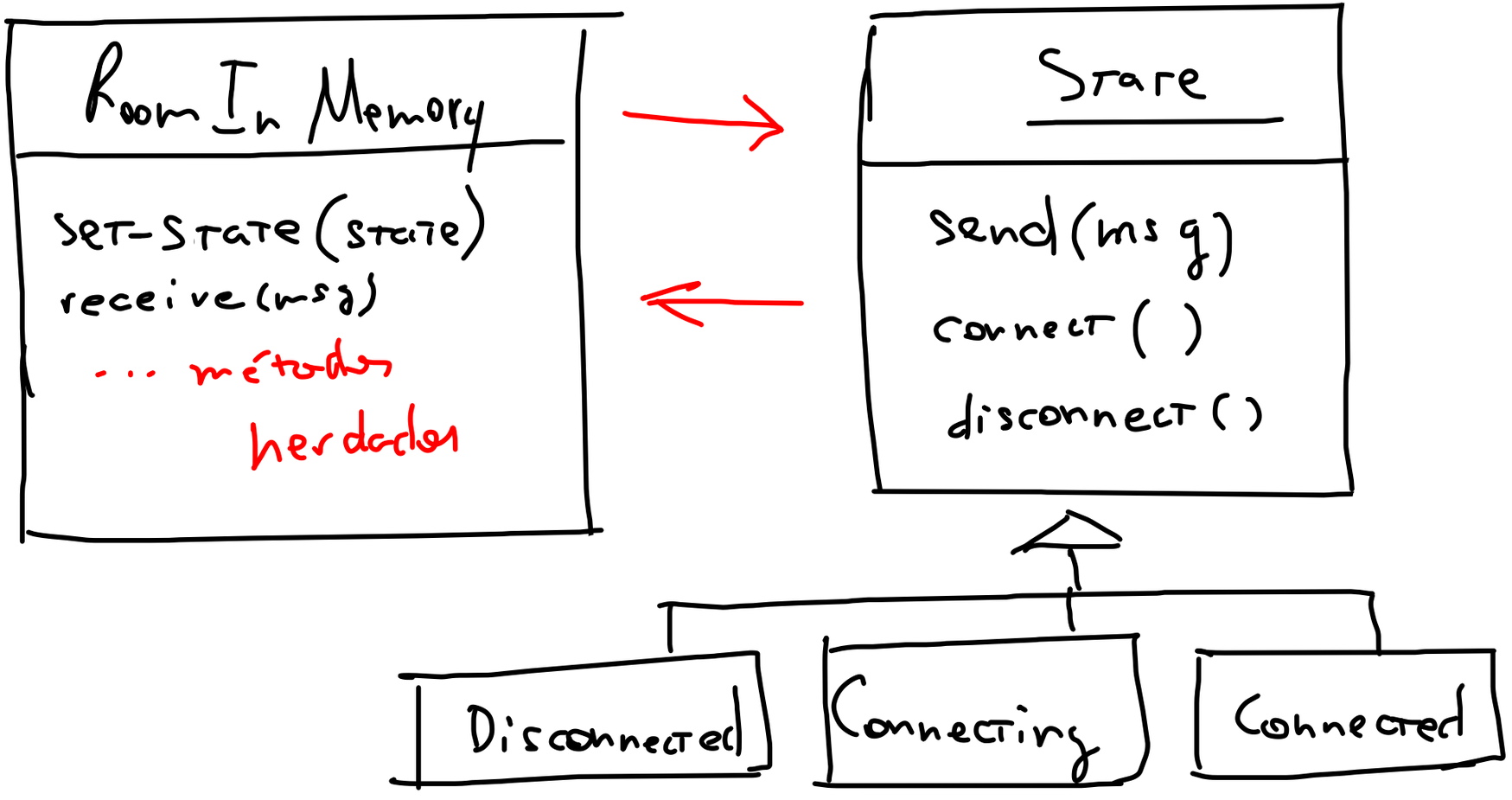


cliente pede conexão ou desconexão

cliente manda mensagem

"function" é uma função que o ChatRoom executará quando chegar mensagem / desconectar / conectar

A DP ESTADO aparece como detalhe de implementação de Room In Memory:



Assim, em vez de ter vários "if else" na classe Room In Memory, posso fazer cada estado se comportar

de maneira única:

- Disconnected impede que "send()" funcione
- Connecting impede que "connect()" tenha qualquer efeito
- Connected é a única para a qual "send()" funciona

Por último, introduzimos a classe KeepAlive, subclasse de ChatRoom, para funcionar como decorador: se x for instância de alguma subclasse de ChatRoom, então KeepAlive(x) instancia um objeto que, diferentemente do anterior, não "joga fora"

mensagens que por acaso tentem ser mandadas enquanto o canal não está conectado.

A nova instância vai manter estas mensagens guardadas e vai enviá-las assim que o canal conectar novamente. Além disso, KeepAline tentará reconectar o canal toda vez que ele cair.