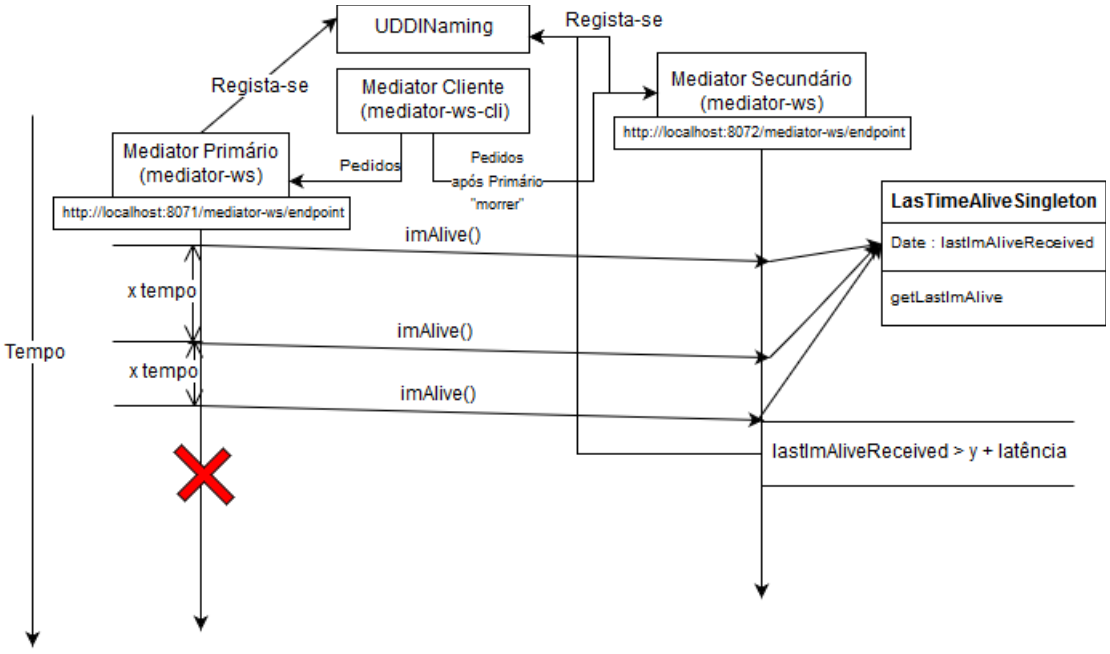
Sistemas Distribuídos 2016-2017

T06

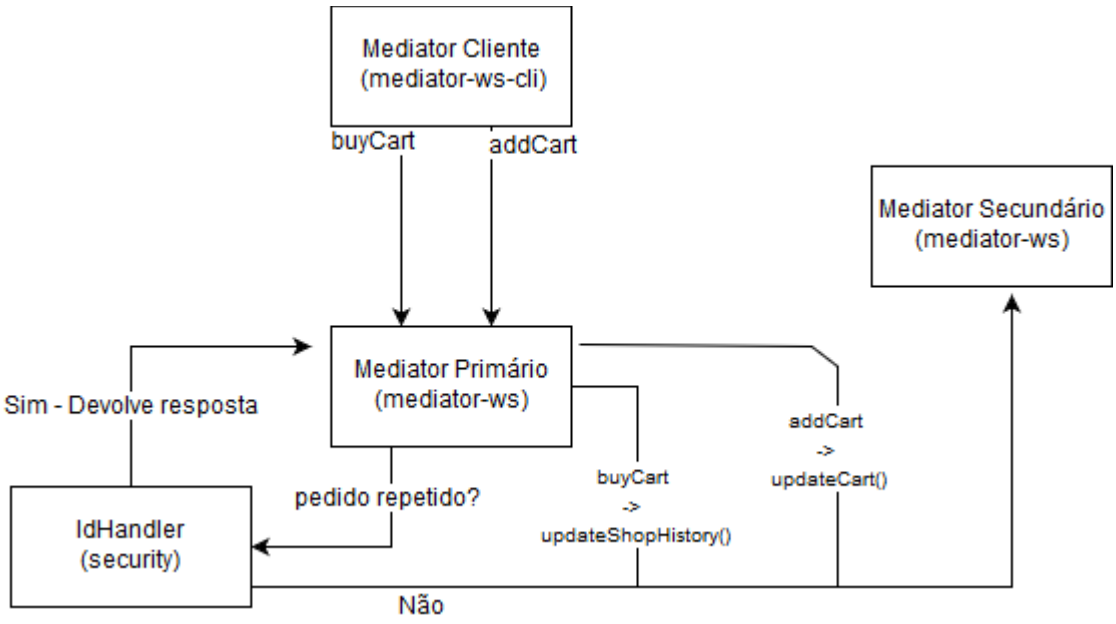
Leonor Clemente, 78054

Ana Silva, 79304

GITHUB: https://github.com/tecnico-distsys/T06-Komparator



Para garantir a **substituição** do servidor, neste caso do mediator primário, é criado um servidor secundário que irá apenas substituir o mediator primário caso este falhe. Começa-se por lançar o mediator secundário com o endereço "http://localhost:8072/mediator-ws/endpoint". Este servidor backup após ser lançado vai ficar à espera que o mediator primário se ligue. O mediator primário liga-se no endereço "http://localhost:8071/mediator-ws/endpoint" e começa a enviar de x em x tempo provas de vida (imAlive()) ao mediator secundário. Sempre que o mediator primário envia uma prova de vida é guardado numa variável global (lastImAliveReceived) a hora em que este método foi executado. O mediator secundário de y em y tempo vai verificar se a diferença entre variável global que guarda o tempo da última prova de vida e a hora actual é superior ao limite de tempo definido que resulta de y mais a latência. Caso seja, isto significa que o mediator primário falhou e o mediator secundário deve assumir o papel do mediator primário. Para tal regista-se no uddi com o mesmo nome que o mediator primário mas com o seu próprio endereço. A partir deste momento o mediator secundário passa a receber pedidos que anteriormente iriam para o mediator primário.



A **replicação** acontece quando existe algum acontecimento que altere os estado do servidor primário e essas mesmas alterações devem ser propagadas para o servidor secundário. Tanto o mediator primário como o mediator secundário devem estar ligados. Assim que o mediator primário receber um pedido de buyCart ou addCart ele começa por verificar se esse pedido é repetido, ou seja, se o mesmo já havia sido pedido antes. Esta verificação acontece através do handler idHandler disponível no módulo security, onde existe uma variável global que guarda o id único do pedido obtido através do cabeçalho da mensagem SOAP. Este id único é adicionado ao cabeçalho na altura que é efectuado o pedido. Nesta mesma classe existe uma hashmap que guarda o id e a resposta obtida para esse pedido. Assim sempre que o buyCart ou o addCart são invocados a primeira coisa que o mediator primário faz é verificar se nesta hashmap já existe o pedido. Se sim então obtém a resposta e o método devolve essa mesma resposta. Caso contrário, o mediator primário acede à instância do mediator cliente disponivel na classe LifeProof . Esta permite que sejam feitos pedidos ao mediator secundário. Através desta instância o mediator primário invoca os métodos responsavéis pelas actualização, ou seja, o updateShopHistory para o caso do buyCart e o updateCart para o caso de addCart. No final do método adiciona também a resposta da execução obtida à hashmap dos pedidos. No método updateShopHistory o mediator secundário adiciona a ShoppingResultView à sua lista de histórico de compras e no updateCart adiciona tanto o id como o carrinho à lista de carrinhos. Ambos os métodos update são invocado apenas pelo mediator primário pois não faz sentido quando o secundário assume o papel do primário fazer updates nele próprio.