IST-TG

Sistemas Distribuídos – Entrega 3

Relatório – Tolerância a Faltas

Identificador do grupo: T49

Miguel Silveira

Nº 81984

João Pedro Meira

Nº 82014

Maria Inês Guerra

Nº 82248

Repositório GitHub: https://github.com/tecnico-distsys/T49-Komparator

Tolerância a Faltas:

A política de tolerância a faltas baseia-se na existência de um **mecanismo redundante** que possibilite que a função da componente comprometida seja obtida de outra forma, sem comprometer o bom funcionamento do sistema.

No contexto do nosso projeto, a deteção da falha do servidor principal é feita a partir do envio de um sistema síncrono de 'lifeproofs' a cada 5 segundos para o servidor secundário. Se o último 'lifeproof' recebido demorar mais que 5 segundos, o secundário assume que houve uma falha no primário, fazrebind no UDDI, e torna-se assim o servidor principal. Exemplo de falhas: signKill(CTRL-C),desligar a interface da rede, término remoto, entre outros.
Efetivamente, não são feitos cálculos sobre a falha ocorrida, no entanto, para evitar a perda de informação do servidor, este manda 'updates' ao servidor secundário sempre que é feita alguma modificação nas compras feitas ou nos carts criados anteriormente, replicando a informação dada. Assim quando existir uma falha, o cliente pode passar a comunicar com o servidor de "backup" sem conflitos.

Na solução encontrada para a implementação da tolerância a faltas, foram necessários os métodos de provas de vida do servidor, e os de atualização de carrinhos e de histórico de compras.

Início do Programa(ver fig.1) :

1. Quando o programa se inicia é criado um servidor primário e um alternativo para backup.
2. O primário, sendo o único que comunica com o cliente, publica-se no UDDI.
3. Em intervalos de 5 segundos, envia provas de vida pelo método imAlive() com uma timestamp, para que o servidor de backup saiba que tudo corre com normalidade.
4. Se o cliente adicionar itens a um carrinho, essa informação é enviada para o servidor alternativo pelo métodoupdateCart() com a semântica no-max-1-vez.
5. Se o cliente comprar os itens de um carrinho, essa informação é enviada para o servidor alternativo pelo método updateShopHistory()com a semântica no-max-1-vez.

Todos os requisitos anteriormente citados foram implementados mas a semântica é a de pelo-menos-uma-vez.

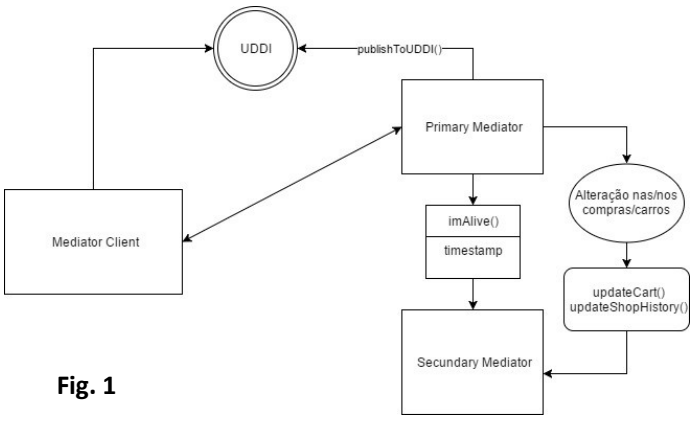

Fig. 1

Falha no Mediador Primário (ver fig. 2):

1. Quando o servidor primário morre, o secundário não recebe mais imAlive().
2. No servidor de backup, é feita a verificação do timestamp enviado no imAlive(), vai ser ultrapassado o tempo de tolerância(5 segundos) e então assumida a falha no servidor principal.
3. O servidor de backup faz rebind, tornando-se assim o servidor principal e pode agora ser feita a comunicação entre ele e os clientes.
4. A substituição de servidores para a comunicação, não deve ser notada pelo cliente.

Todos os requisitos anteriormente citados foram implementados.

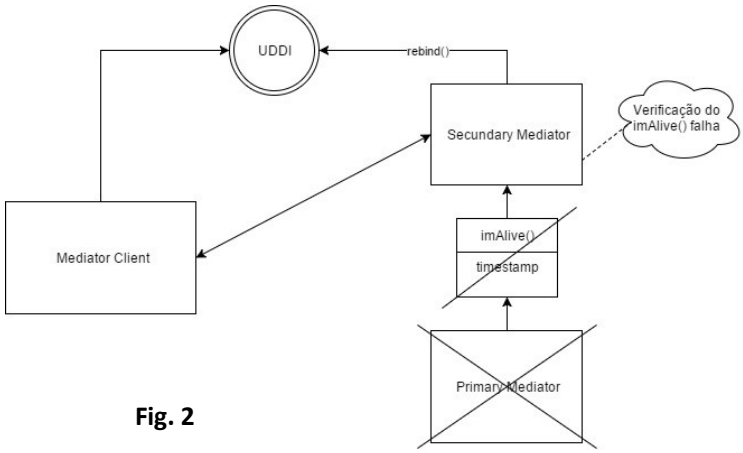

Fig. 2