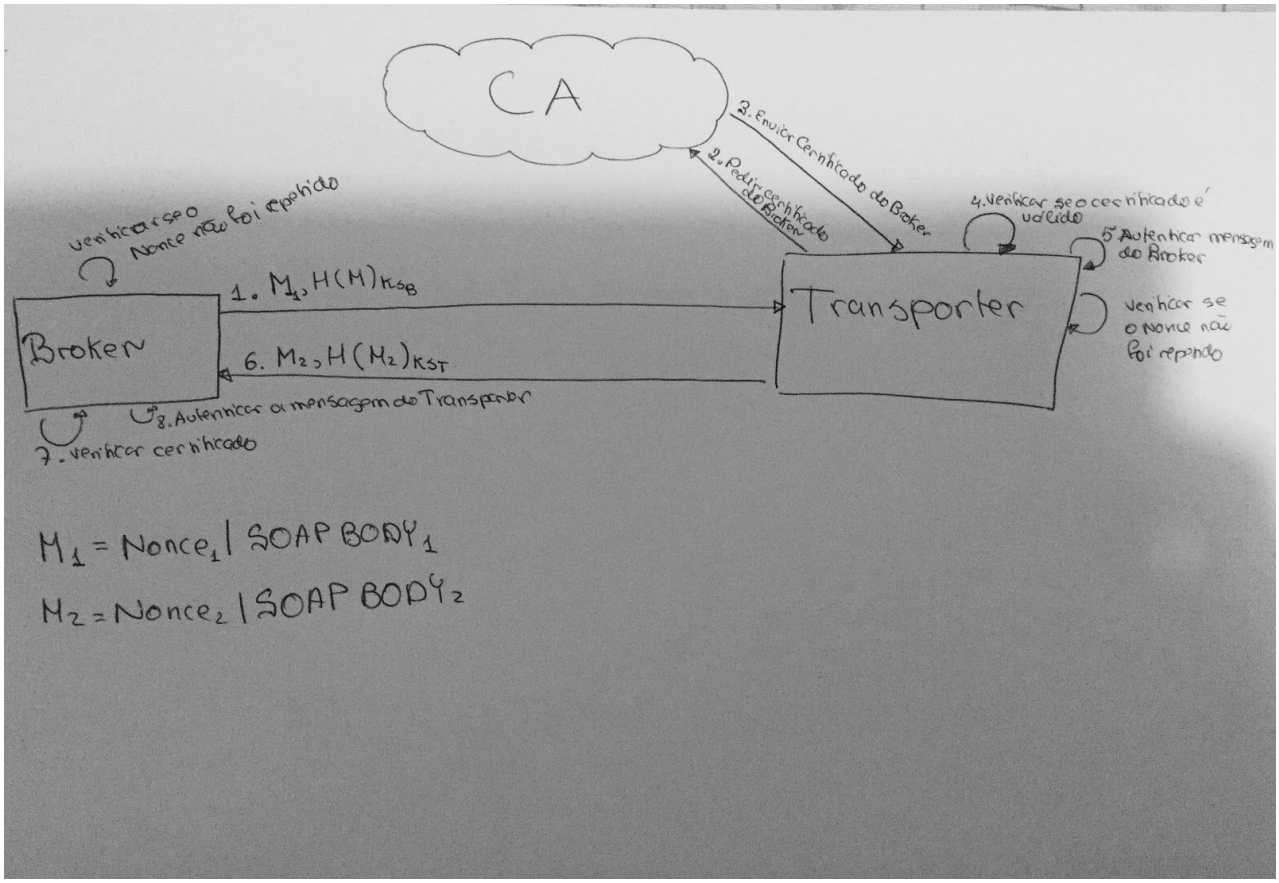
Sistemas Distríbuidos 2015-2016

Grupo A22

https://github.com/tecnico-distsys/A_22-project

- 
  - Número: ist1**79042**
  - Nome: Jorge Francisco Catarino Heleno
- 
  - Número: ist1**78454**
  - Nome: Nuno Miguel Ribeiro Silva
-
  - Número: ist1**78134**
  - Nome: Illya Gerasymchuk

Segurança



Descrição do problema

Não permitir repetição de mensagem, e ter a certeza que a mensagem foi enviada por quem o diz ter feito e que nao foi modificada.

Racional

Usar **Nonce** para proteger-se dos ataques de repetição. Fazer **digest** de NONCE | SOAP BODY e cifrar isso com a chave privada para garantir que a mensagem nao foi modificada e que foi enviada por quem o diz ter feito. Fazemos **cache** de certificados (10 segundos, configurável), se o cert for mais antigo, vamos a **CA.** Os certificados são **verificados**, para garantir que são mesmo da entitdade. Em cada request verificamos se já houve um request com o **Nounce** que aparece no header, se sim, recusamos (proteção contra replay attacks).

Replicação

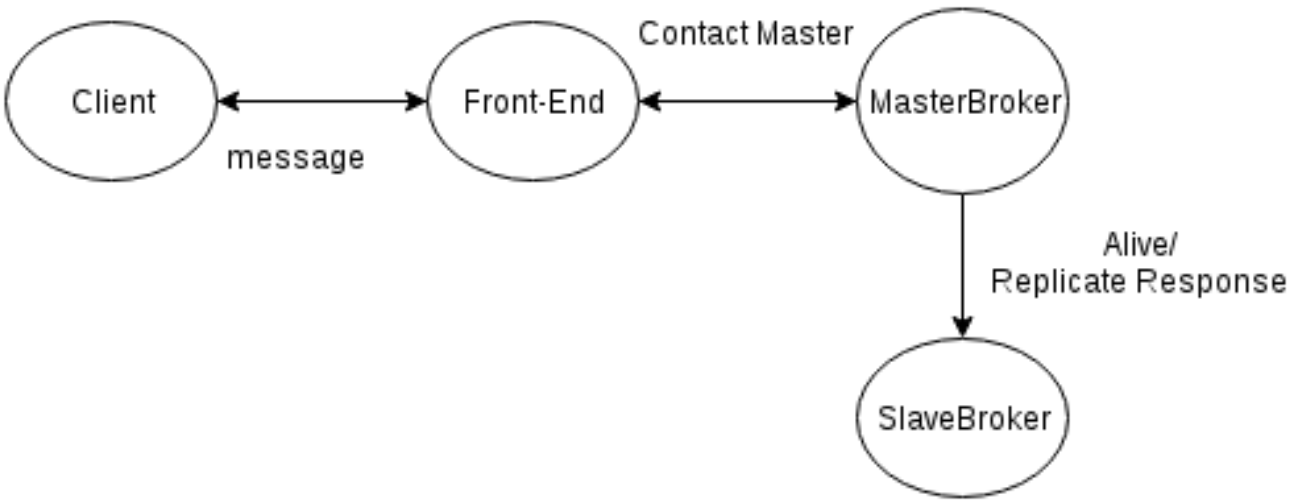


Fig 1: Esquema da solução para o problema da replicação

Descrição do problema

O problema abordado consiste em garantir a tolerância a uma falta (e apenas uma) do Broker, ou seja, garantir que caso este pare por alguma razão, a aplicação continua a correr como se nada tivesse acontecido.

Racional

Para resolver o problema criámos uma classe com o nome FrontEnd cuja função é fazer de intermediário entre o cliente e o BrokerPort, repetindo pedidos falhados de forma transparente para o cliente. Para além disso, criámos ainda uma classe SlaveBrokerPort (adiante designado por Slave) cuja função é detetar se o MasterBrokerPort (broker principal naquele momento e adiante designado por Master), está ativo. Em caso negativo, o Slave toma o seu lugar, registando-se no UDDI como se fosse o Master. Assim, sempre que um pedido do FrontEnd falha, este apenas tem de ir ao UDDI buscar o endereço do (novo) Master (que foi substituído pelo Slave).

Ainda no lado do Broker, para não haver perda de estado entre as réplicas (Master e Slave), é utilizado um esquema de *passive replication*, onde cada pedido que o FrontEnd envia para o Master tem associado um identificador único, gerado aleatoriamente no FrontEnd. Quando o Broker processa um pedido, é guardado em cache uma associação entre esse identificador e a resposta obtida depois da execução, sendo essa associação depois replicada para o Slave ainda antes de ser enviada de volta para o FrontEnd. Desta forma, é trivial identificar pedidos duplicados e respondê-los usando a sua resposta original.