

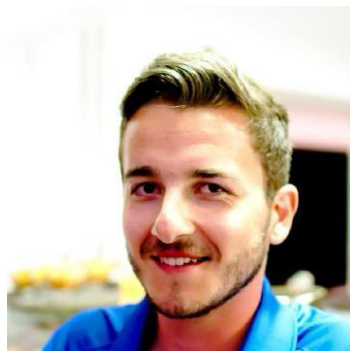
# Sistemas Distribuídos 2015-2016

Grupo A65

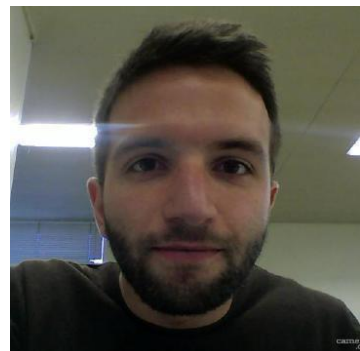
[https://github.com/tecnico-distsys/A\\_65-project](https://github.com/tecnico-distsys/A_65-project)



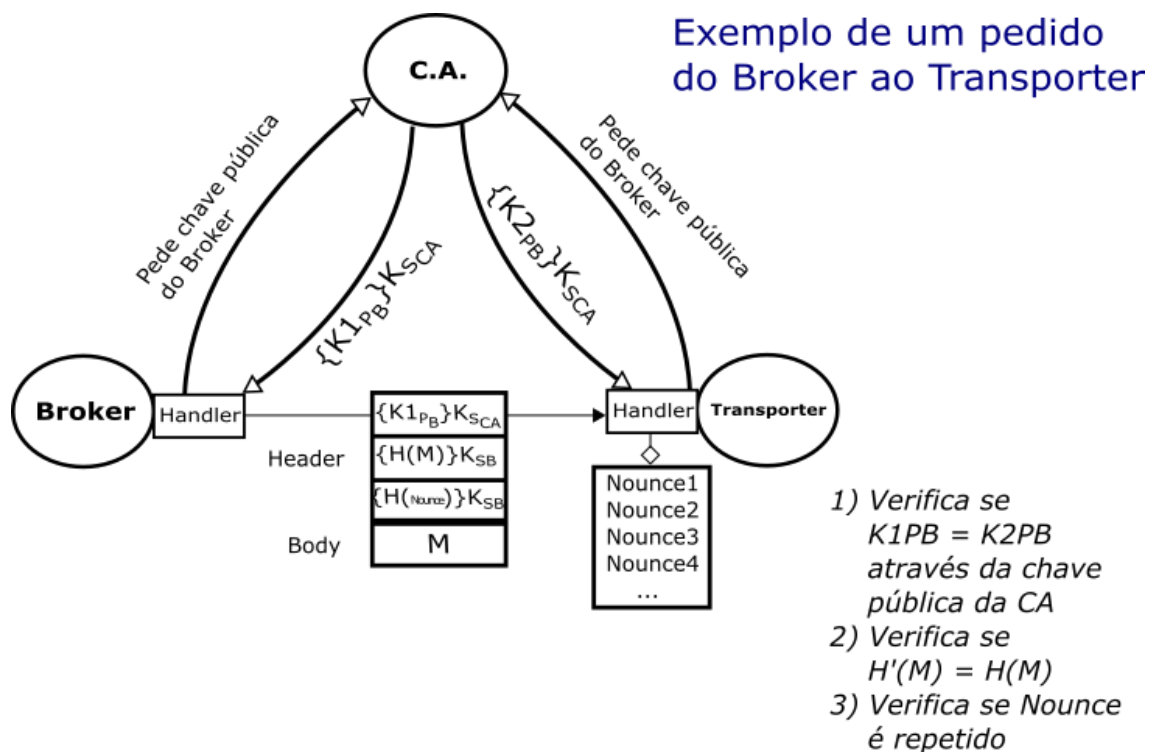
**André Silva**  
68707



**Duarte Coelho**  
73339



**João Martins**  
73438



## Segurança – Descrição da Figura e Explicação da Solução

Durante a implementação da parte da Segurança, infelizmente deparamo-nos com problemas na verificação das mensagens SOAP (em contexto de *inbound*). Como resultado, tomámos a difícil decisão de não implementar na versão final do projeto o envio e intercepção das mensagens (que permitiriam assegurar a autenticidade, integridade, e não-repúdio).

Ainda assim, o nosso SOAP Handler (“SecurityHandler” na estrutura do projeto) constrói o cabeçalho das mensagens com assinatura digital. Outras partes essenciais da solução estão também implementadas, como, por exemplo, a *Certification Authority* (CA), implementada em *contract-first* com um WSDL criado de raiz por nós.

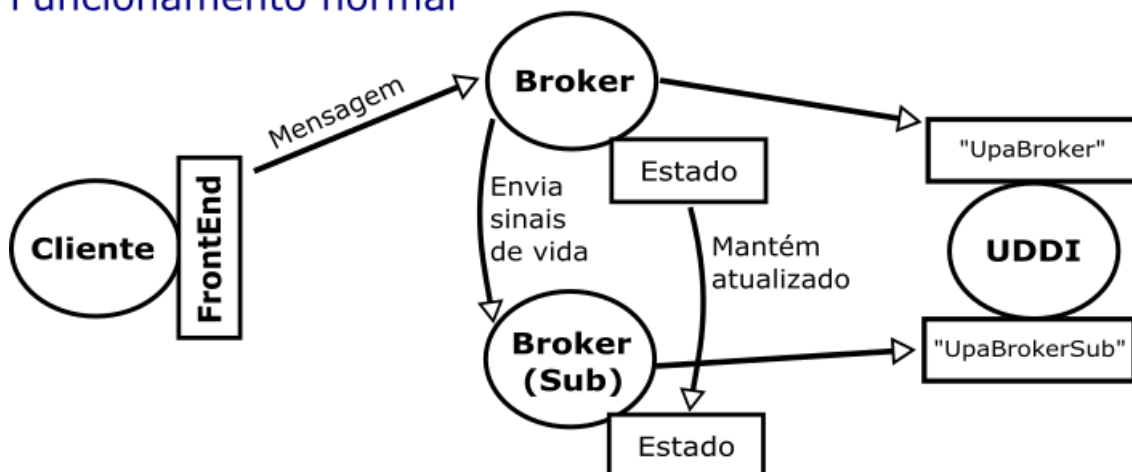
Desta forma, entregamos o código e estruturas a avaliação, não sendo possível testar o seu funcionamento.

Funcionalidades concretizadas:

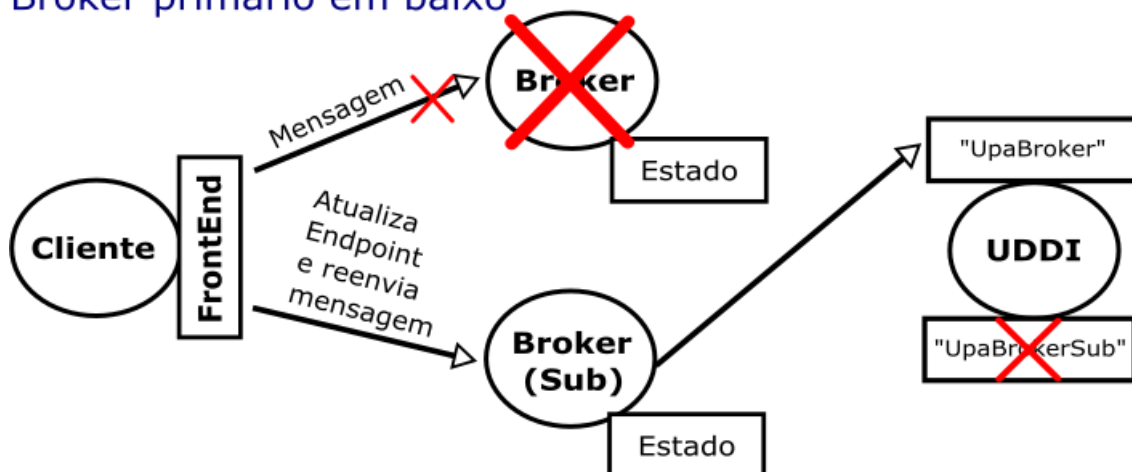
- CA (“ca-ws” e “ca-ws-cli” na estrutura do projeto)
- Assinatura Digital
- Conteúdo da mensagem “*digested*” e cifrado com chave privada
- Nounce “*digested*” e cifrado com chave privada

A figura representa o que seria o objetivo final da nossa implementação: uma entidade quando envia um pedido a outra, é adicionado um *header* à mensagem na entidade que envia para se poder proceder à verificação da sua autenticidade, integridade, e não-repúdio na entidade recetora.

## Funcionamento normal



## Broker primário em baixo



## Replicação – Descrição da Figura e Explicação da Solução

Todas as funcionalidades necessárias à Replicação foram implementadas com sucesso.

Na primeira figura, demonstra-se o funcionamento normal da replicação. O Broker Primário processa as mensagens enviadas por Clientes. Intermitentemente, envia sinais de vida ao Secundário. Sempre que o seu estado é atualizado, notifica o Secundário para imitar a atualização. Os dois Brokers conhecem-se através da pesquisa dos seus nomes no UDDI.

Na segunda figura, representa-se o caso em que o Broker Primário deixou de funcionar. O Broker Secundário, ao deixar de receber sinais de vida do Primário, assume o nome do Primário, e regista o seu *endpoint* no UDDI, através de um processo de *rebind*. Por seu lado, o FrontEnd do Cliente, ao detetar que um dos seus pedidos ao Broker fez disparar um *timeout*, volta a procurar no UDDI pelo "UpaBroker" (o único nome que conhece), obtendo desta vez o *endpoint* do Broker que substituiu o antigo Primário. Após este processo, reenvia o pedido que deu origem ao *timeout* ao novo Broker. A partir daí, o Cliente funciona normalmente.

Não representados nas figuras estão todos os Transporters registados no UDDI, que são sempre conhecidos pelo atual Broker Primário.