

Computação Orientada a Serviços

Trabalho Prático 2

MEIC, Semestre de Inverno 2013/2014

**Docente: Discente:**

Cátia Vaz Ricardo Nunes, nº 31656

Diogo Cardoso, nº 32466

Índice

[1. Introdução 1](#_Toc377848856)

[2. Coreografia 2](#_Toc377848857)

[2.1 Participantes e Relações 2](#_Toc377848858)

[2.2 Mensagens 2](#_Toc377848859)

[2.3 Fluxo 4](#_Toc377848860)

[3. Cenários 7](#_Toc377848861)

[Operação completa com sucesso 7](#_Toc377848862)

[Cancelamento de estudo com sucesso 8](#_Toc377848863)

[Falha na conclusão de um estudo inexistente 9](#_Toc377848864)

[Falha no cancelamento de um estudo inexistente 10](#_Toc377848865)

# Introdução

Nesta fase do trabalho é-nos pedido para desenhar e validar a coreografia associada ao serviço desenhado na primeira fase. O relatório está dividido em duas partes: na primeira iremos abordar os vários componentes da coreografia, (participantes, mensagens e fluxo) enquanto que na segunda iremo-nos debruçar sobre 4 cenários que serão corridos contra coreografia desenvolvida.

# Coreografia

## Participantes e Relações

A coreografia tem 3 participantes:

1. Cliente (*Client*)
2. Funcionário (*Employee*)
3. Serviço (*BrandAnalyticsService*)

Abaixo é ilustrada a relação entre os vários participantes envolvidos na coreografia, onde podemos ver que se mantém o princípio definido na primeira parte deste trabalho: tanto o cliente como o empregado apenas conseguem comunicar com o serviço.

## 

Ilustração Participantes na coreografia

## Mensagens

Para cada mensagem trocada entre dois participantes existe um tipo que descreve a mensagem trocada. Neste caso definimos duas mensagens por cada operação, uma mensagem para o pedido, a outra para a resposta. A nomenclatura usada é usarmos o nome da operação para a mensagem de pedido e o nome da operação com o sufixo *Response* para a resposta (e.g. *Cancel* para pedido e *CancelResponse* para resposta). As operações definidas são:

* *Client* → *Service:*
  + *RequestStudy*
  + *GetCompletedStudies*
  + *CancelStudy*
* *Employee* → *Service*:
  + *GetPendingStudies*
  + *StartStudy*
  + *SetStudyAsCompleted*

Abaixo temos uma ilustração dos tipos de mensagens (*Information Types*) definidos no projecto (pedido e resposta) e o *schema* usado para a sua validação (por motivos de simplificação todas as mensagens são representadas como *string*).

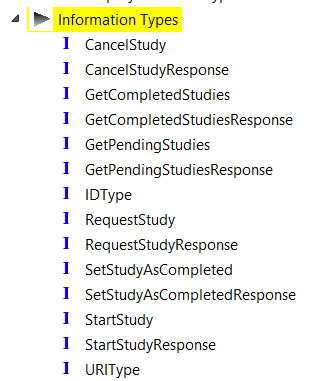


Ilustração – Tipos de mensagens trocados nesta coregorafia

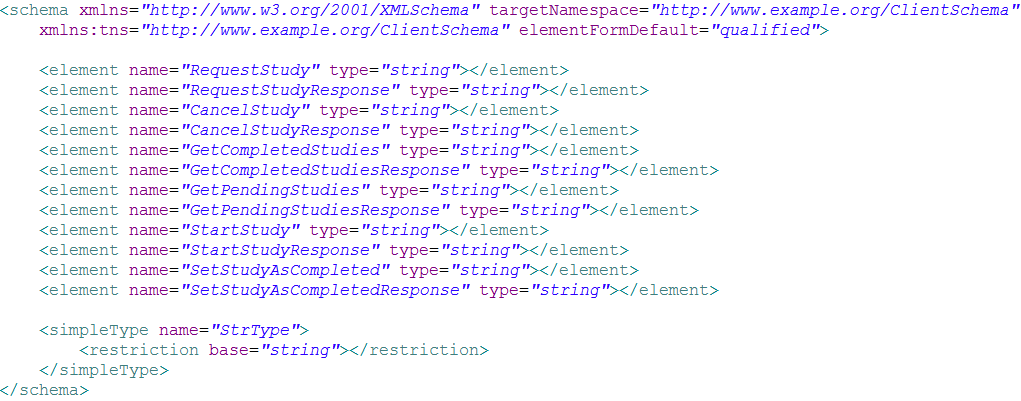


Ilustração *Schema* das mensagens trocadas entre os vários participantes

## Fluxo

Abaixo está o fluxo definido para esta coreografia. Uma explicação breve de todos os passos:

1. *RequestStudy*: Cliente pede a serviço para começar o estudo. A resposta contém um identificador criado para o estudo pedido.
2. *GetPendingStudies*: Funcionário pede lista de estudos pendentes (não estão completos). A resposta contém uma lista dos mesmos.
3. *StartStudy*: Funcionário pede ao serviço para começar um novo estudo com base nas keywords associadas.

Nesta fase podem acontecer duas coisas: ou o estudo é cancelado pelo cliente (ver Ilustração 5*,* à esquerda) ou o funcionário dá o estudo como completo (ver Ilustração 5*,* à direita), com posterior consulta dos resultados por parte do cliente.

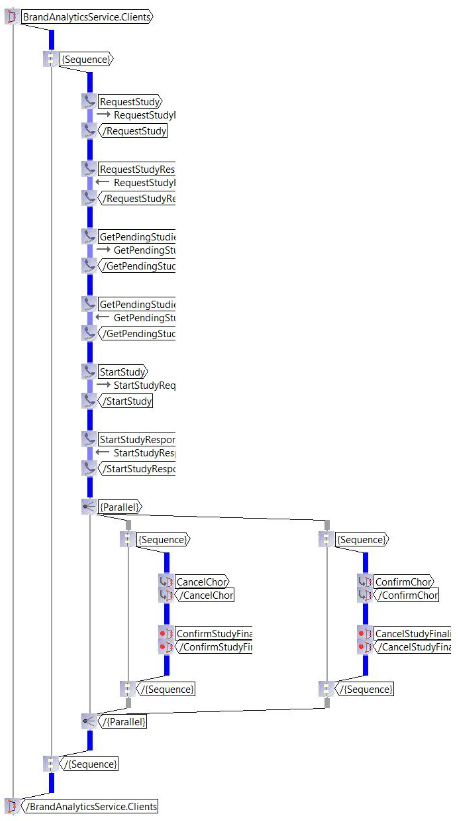


Ilustração – Fluxo definido para a coreografia

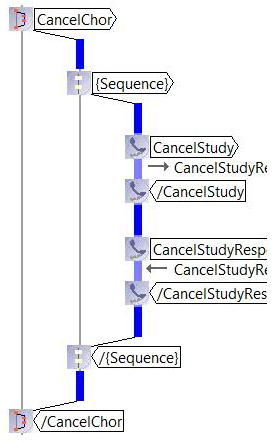
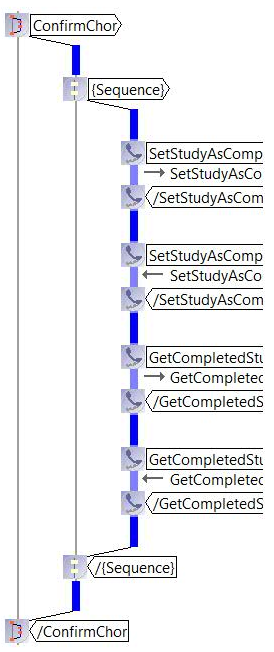
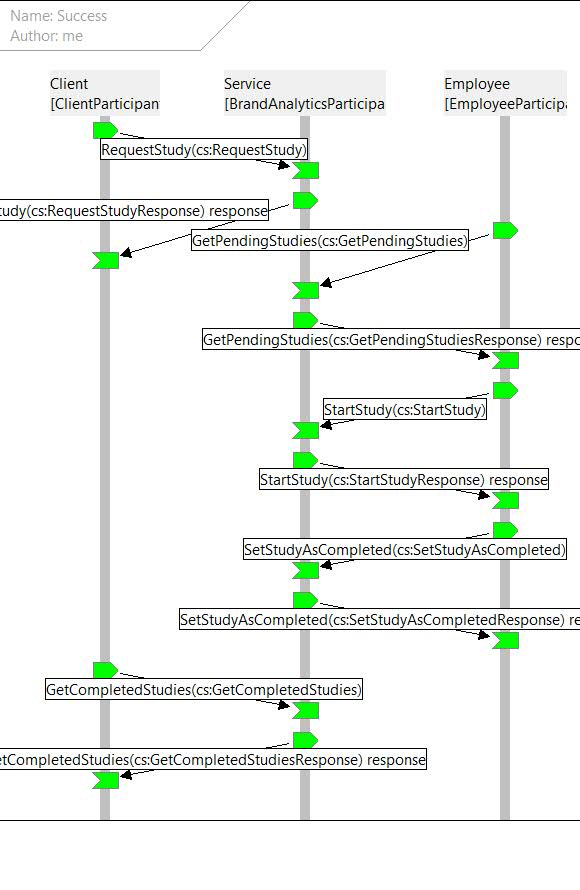


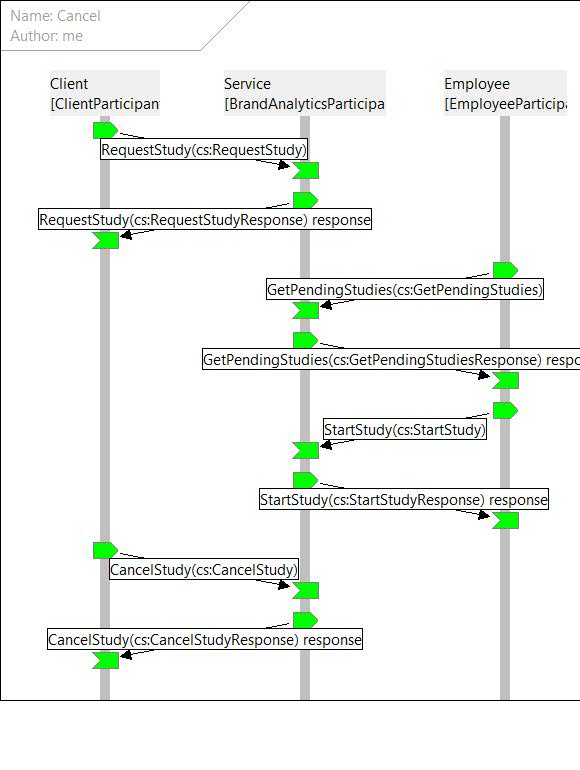
Ilustração Fluxo de finalização (à esquerda) e cancelamento (à direita) de estudo.

# Cenários

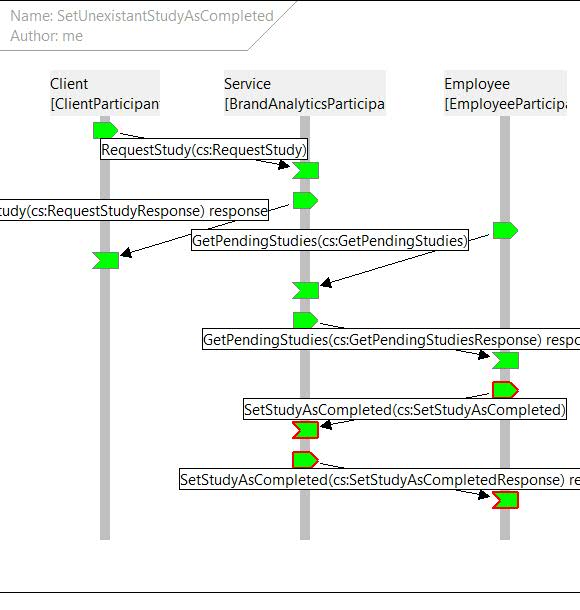
### Operação completa com sucesso



### Cancelamento de estudo com sucesso



### Falha na conclusão de um estudo inexistente



### Falha no cancelamento de um estudo inexistente

### D:\Rnunes\ISEL\COS\cos\images\cancelunexistantstudy.jpeg

# Conclusão

Este trabalho permitiu-nos perceber as capacidades de uma coreografia no desenvolvimento de serviços e qual o seu papel neste sector. Infelizmente as ferramentas usadas não estão completas, o que nos atrasou bastante o desenvolvimento.