

**Software Project Management 2018/2019**

**Bill Splitter**

**D2.2.1 - Software Architecture & Design**

**Autores**

* Paulo Dias ([dias.pauloalex@gmail.com](mailto:dias.pauloalex@gmail.com))
* Sérgio Soares ([smcsoares20@gmail.com](mailto:smcsoares20@gmail.com))
* Rui Mota ([ruifilipetmota@gmail.com](mailto:ruifilipetmota@gmail.com))
* Ruben Marques ([rubenantoniomarques@gmail.com](mailto:ruifilipetmota@gmail.com))

**Estado**

* Rascunho

**Versões Principais**

* v0.1, 1 de Dezembro, Ruben Marques, primeiro esboço
* v0.2, 11 de Dezembro, Ruben Marques, finalização do documento
* v0.3, 11 de Dezembro, Rui Mota, pequenas correções

**Versões Publicadas**

* v1, 18 de Dezembro, Paulo Dias, Versão publicada

**Índice**

1. [Introdução](#_gjdgxs)
2. [Diagramas de classe](#_30j0zll)
3. [Arquitetura do Sistema](#_3znysh7)

### **Introdução**

Este documento tem como objetivo definir os aspectos de programação com base nos requisitos e casos de uso definidos no [SRS](https://docs.google.com/document/d/166iWyW96q2MIrAJA39QLiA0KGNm1XJa05T1UOJeS0Mo/edit?usp=sharing). Serve principalmente para apoiar o programador na hora de escrever o código e também possibilitar a compreensão do algoritmo e arquitetura para os colaboradores não diretamente envolvidos na programação.

Para o âmbito e a dimensão desta aplicação foi decidido que seria suficiente ter um diagrama de classes que especifica os módulos e a relação entre as suas classes e um conjunto de diagramas de sequência que descreve os casos de uso definidos no [SRS](https://docs.google.com/document/d/166iWyW96q2MIrAJA39QLiA0KGNm1XJa05T1UOJeS0Mo/edit?usp=sharing).

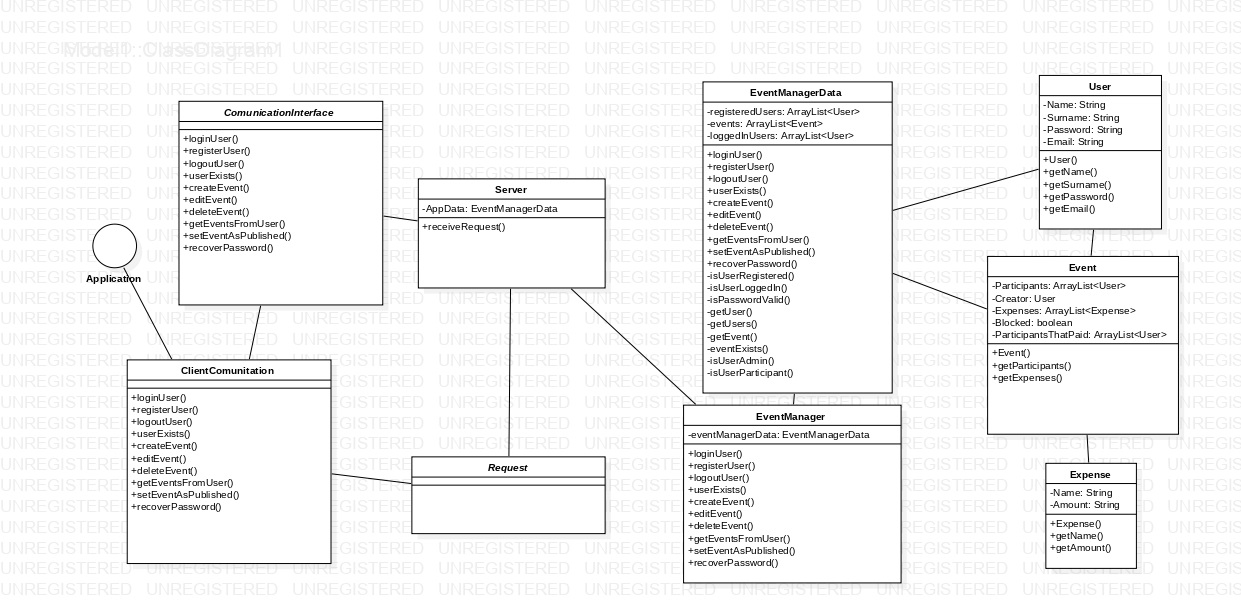
### **Diagrama de classe**

Aqui é apresentado o diagrama de classes relativo à gestão dos eventos. Por questões de redundância foram omitidas as classes Java relativas à aplicação Android, porque estas tornariam este diagrama mais ambíguo e menos objetivo.

Toda a interação vinda da aplicação Android está representada naquele círculo à esquerda e daí é desencadeado todo o processo relativo a um pedido do utilizador.

As classes do módulo da comunicação são as “ClientCommunication”, “CommunicationInterface”, “Request” e “Server”. A classe “Request” serve para representar um pedido num objeto que será transferido da aplicação para o servidor

As classes “EventManager”, “EventManagerData”, “User”, “Event” e “Expense” representam o módulo da lógica.



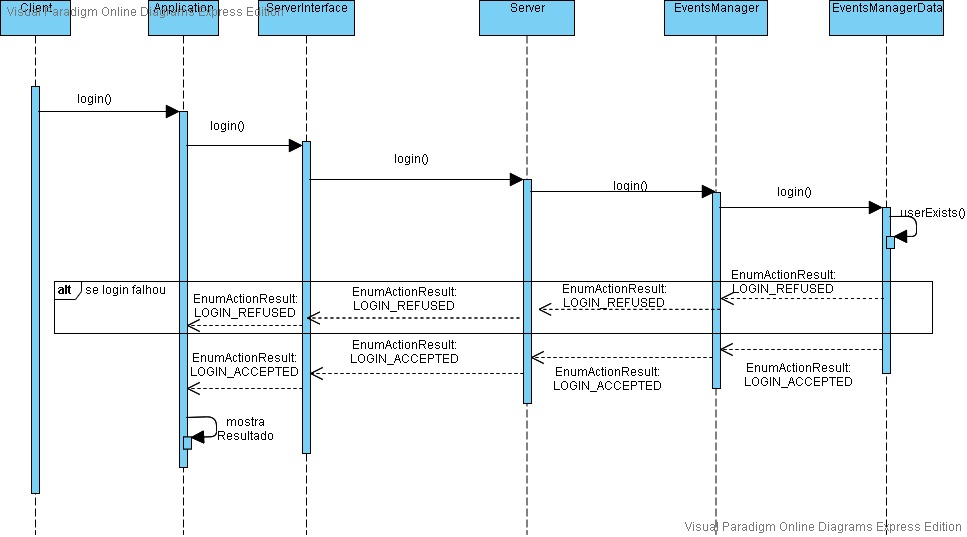
### **Diagramas de sequência**

Aqui nesta secção são apresentados os diagramas de sequência referentes aos casos de uso do [SRS](https://docs.google.com/document/d/166iWyW96q2MIrAJA39QLiA0KGNm1XJa05T1UOJeS0Mo/edit?usp=sharing). Todos eles seguem o mesmo caminho:

Application → ServerInterface → Server → EventsManager → EventsManagerData.

A classe EventsManager é a que realmente trata de processar o pedido e devolver uma resposta baseado na lógica programada. As restantes servem principalmente para encaminhar o pedido até à lógica.

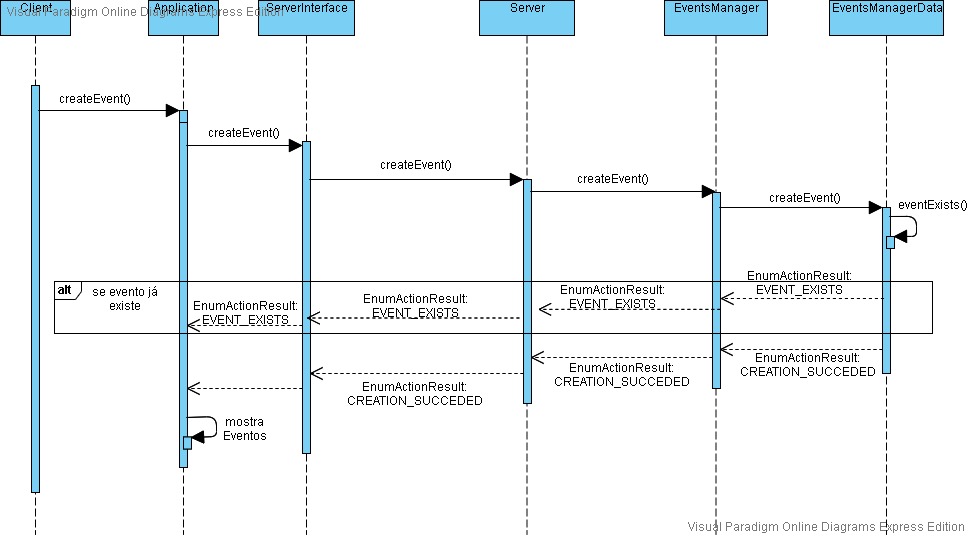
**Login**



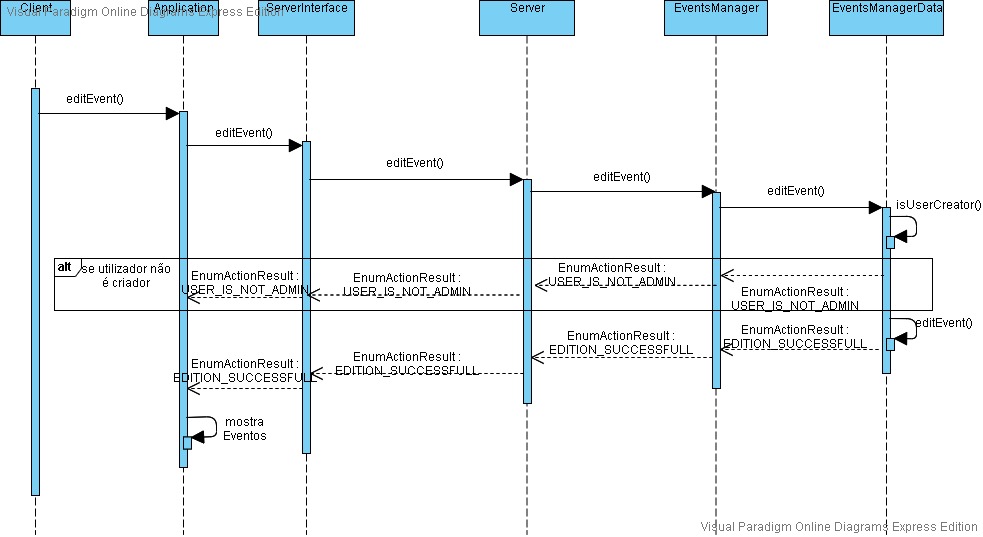
**Register**

#### 

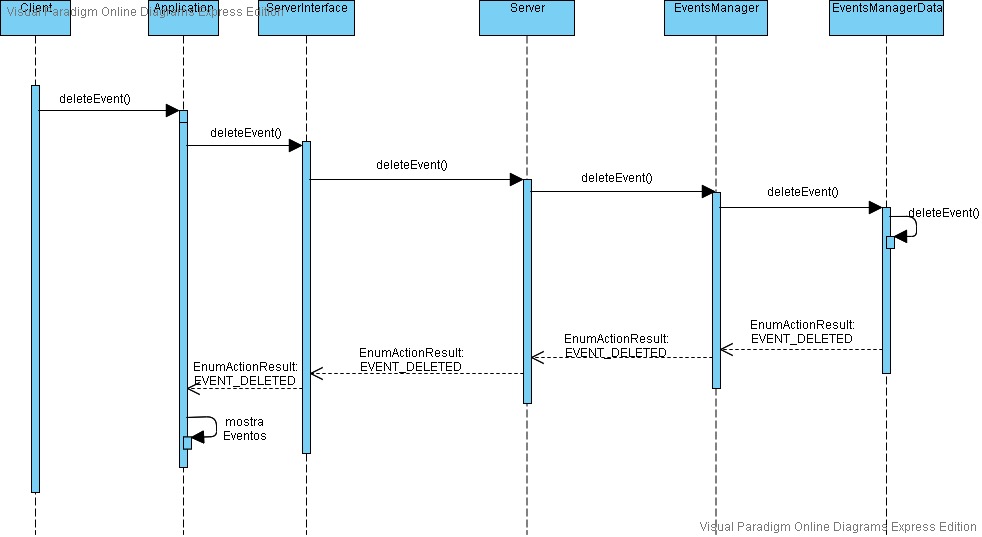
**CreateEvent**

****

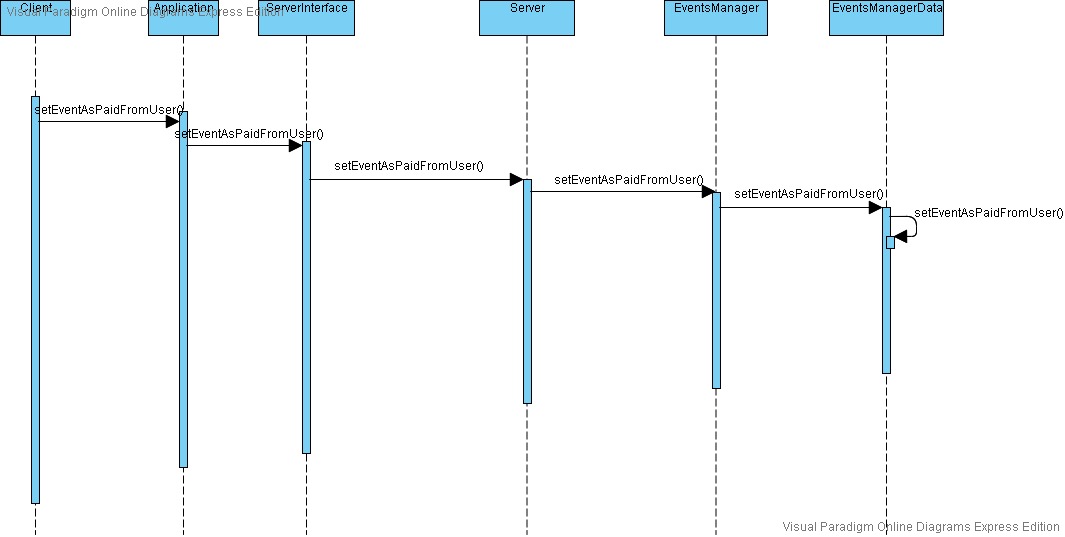
**EditEvent**

****

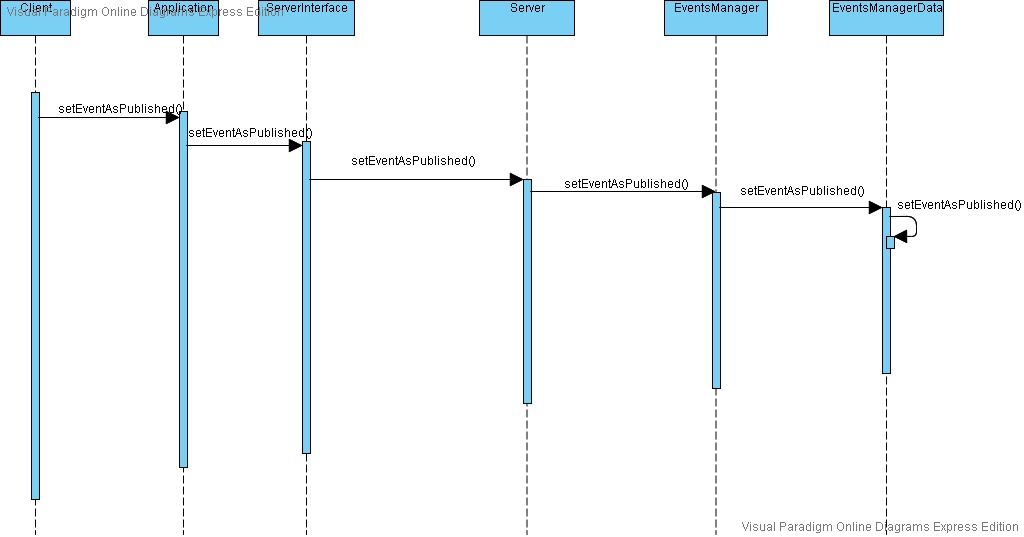
**DeleteEvent**

****

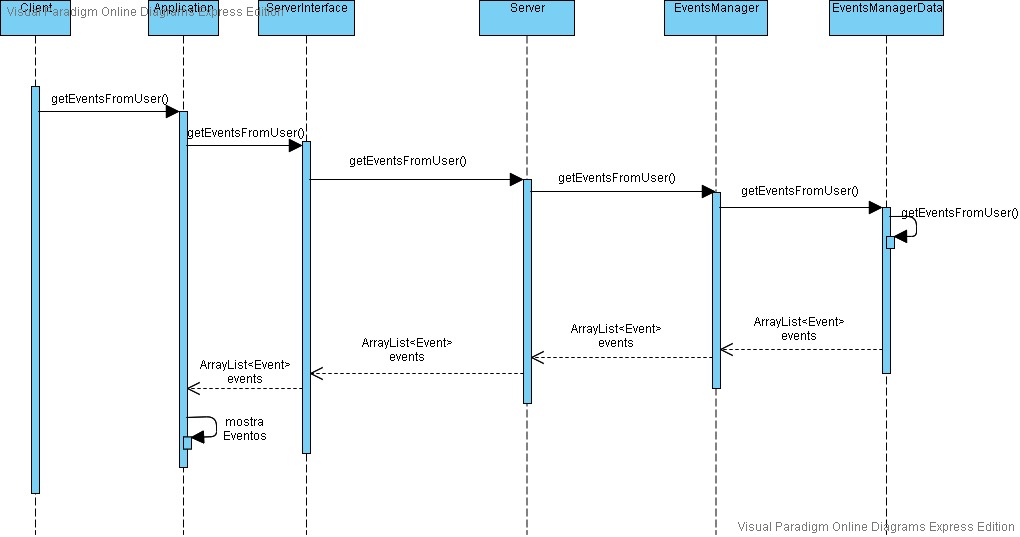
**SetEventAsPaidFromUser**

****

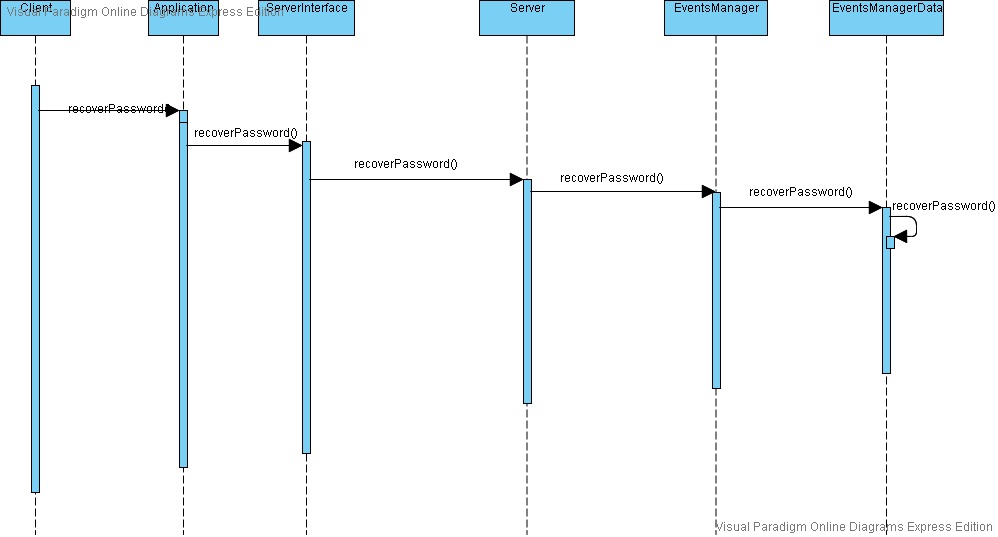
**SetEventAsPublished**

****

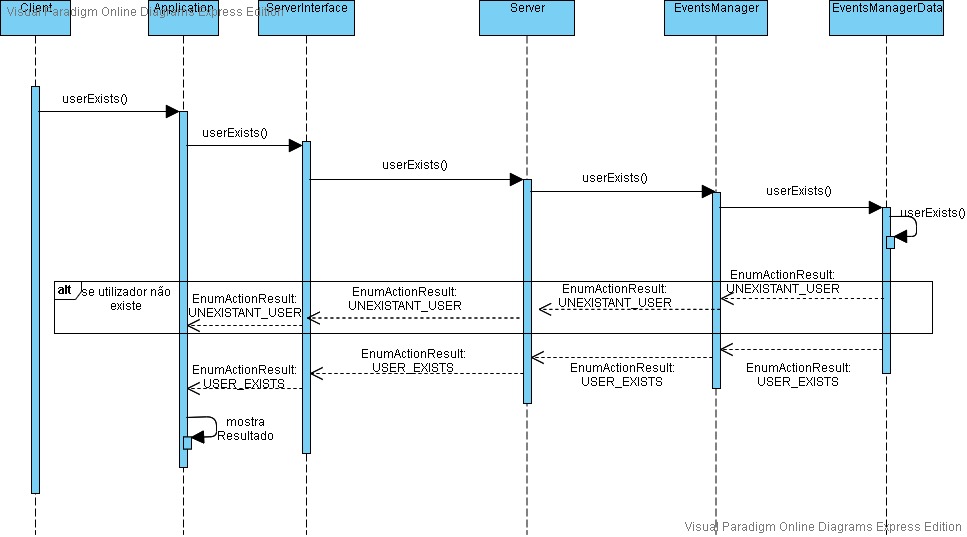
**GetEvents**

****

**RecoverPassword**

****

**UserExists**

****

### **Documentação Javadoc**

Foi também produzida a documentação relativa ao código fonte de Java da parte do servidor para possibilitar a melhor leitura do mesmo.

Aqui em baixo encontra-se o link para a pasta com os ficheiros .html produzidos.

[JAVADOC](https://drive.google.com/open?id=1SvDhQEIVJa3GPlZDKUIQNTYmYrD8eHOd)