

**PROYECTO SpaceFight**

David Rodríguez

Oscar Pinto

Martín Cantor

Alejandra Gómez

Luis Daniel Benavides

Arquitecturas de Software

11 de Diciembre del 2018

TABLE OF CONTENTS

1 Introduction

1.1 Document overview

1.2 Abbreviations and Glossary

1.2.1 Abbreviations

1.2.2 Glossary

1.3 References

1.3.1 Project References

1.3.2 Standard and regulatory References

1.4 Conventions

2 Architecture

2.1 Architecture overview

2.2 Logical architecture overview

2.2.1 Software Component 1st Level description

2.2.2 Software Component 2nd Level description

2.2.3 Software Component 3rd Level description

2.3 Physical architecture overview

2.3.1 Hardware Component description

2.4 Software COTS

3 Dynamic behaviour of architecture

3.1 Workflow

Introduction

Document overview

La finalidad de este documento es documentar y describir la arquitectura realizada para el proyecto, el cual consistió en el desarrollo de una plataforma web que permita a los usuarios jugar con otras personas, según la sala que escojan.

La aplicación cuenta con otra versión, donde funciona bajo un entorno escalable, y con lo cual se pueda evitar que tenga puntos únicos de fallo a causa del servidor, todo esto, logrado a través del despliegue en Heroku y una conexión con dos servidores en AWS (entorno IAAS) y un entorno principal, lo cual permite contar también con conexión a una base de datos NoSQL en la nube, RabbitMQ y REDIS también fueron utilizados como componentes XAAS.

Abbreviations and Glossary

Abreviaciones:

AWS: Amazon Web Services

Glosario:

REST: Estilo Arquitectónico

MOM: Intermediario de Mensajería

PAAS: Plataforma Como Servicio

XAAS: Todo/Cualquiera Como Servicio (e.g. PaaS, IaaS, SaaS)

Project references

Conventions

Diagrama de clases.

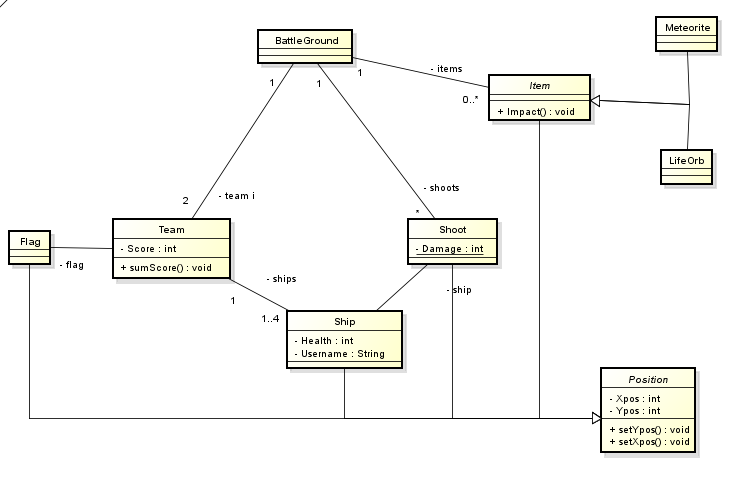
Diagrama de actividad.

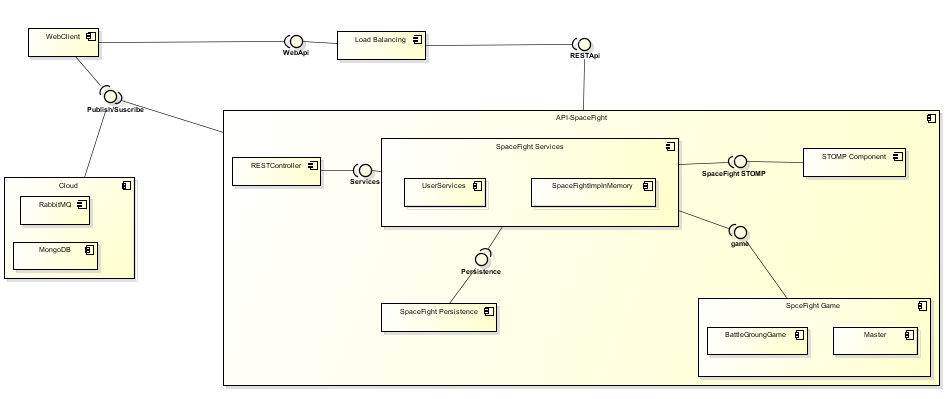
Diagrama de componentes.

Diagrama de despliegue.

Architecture

Logical architecture





Software Component 1st Level description

Web Client: Este componente consiste en la abstracción del elemento del uso de un web browser por el cliente que va a usar la aplicación.

Load Balancing (NginX) : Su objetivo como componente es distribuir el trabajo bajo varias plataformas, de manera que se optimice el tiempo de respuesta y evitar el punto único de fallo, ya que alguna de las plataformas puede fallar.

Software Component 2nd Level description

REST Controller: El controlador recibe las peticiones y haciendo uso de los servicios ofrecidos y delegando las acciones necesarias

Persistencia: efectúa cambios en los datos respecto a las acciones que realiza el usuario con respecto a los movimientos, disparos e items de una nave y del campo.

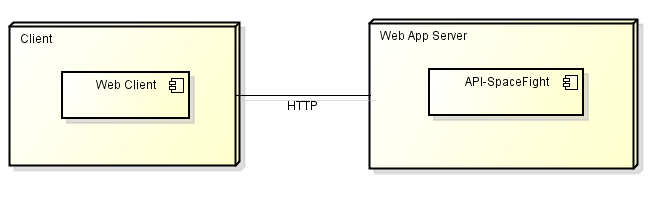
Software Component 3rd Level description

MongoDB: Es el componente que se está usando para la escalabilidad de la aplicación para persistir los datos del usuario y de la plataforma en un repositorio.

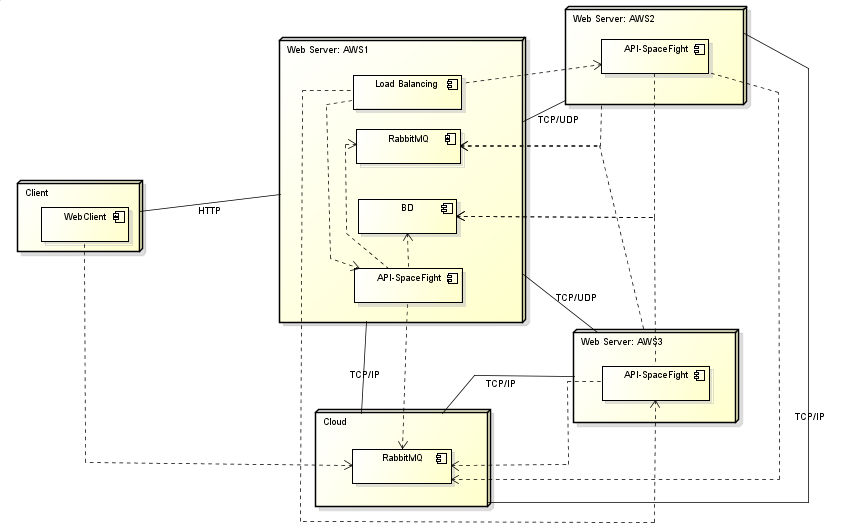
BD-RabbitMQ: Al igual que MongoDB se usa como un XaaS, para controlar las peticiones de los usuarios a un enfoque publicar.

Physical architecture overview

###Deployment v1



###Deployment v2



Description

Web Client: Este componente consiste en la abstracción del elemento del uso de un web browser por el cliente que va a usar la aplicación

Load Balancing: Su objetivo como componente es distribuir el trabajo bajo varias plataformas, de manera que se optimice el tiempo de respuesta y evitar el punto único de fallo, ya que alguna de las plataformas puede fallar

BD-REDIS BSD (cache): Como componente redis actúa como una estructura de almacenamiento de datos memoria

Máquinas virtuales: Contienen los diferentes componentes descritos anteriormente, se encuentran en AWS

COTS:

MongoDB (MongoDB Inc.): Es el componente que se está usando para la escalabilidad de la aplicación para persistir los datos del usuario en un repositorio con base de datos orientada a datos, es un servicio open source, que está enfocado como una base de datos orientada a datos.

RabbitMQ: Al igual que MongoDB se usa como un XaaS, para controlar las peticiones de los usuarios a un enfoque Publish-Subscribe, así mismo también es un servicio opensource, del tipo de broker de mensajes que traduce los mensajes entre receptor y emisor incluso si manejan diferentes protocolos de mensajería.

hace uso del protocolo AMQP.

REDIS BSD: Como componente redis actúa como una estructura de almacenamiento de datos en memoria.

Load Balancing (NginX): Su objetivo como componente es distribuir el trabajo bajo varias plataformas, de manera que se optimice el tiempo de respuesta y evitar el punto único de fallo, ya que alguna de las plataformas puede fallar.

Workflow

##Activity

