**DOCUMENTO ARQUITECTURA DE LA SOLUCION**

**Arquitectura Toures Balones**

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Descripción de la modificación** |
| 1.0 | Noviembre 16 2019 | Documento inicial |
| 2.0 | Noviembre 23 2019 | Documento final |

Tabla de Contenido

1. Introducción 4

1.1 Visión 4

1.2 Objetivo 4

1.3 Definiciones, Acrónimos y Abreviaturas 4

1.4 Referencias 4

1.5 Alcance 4

2. Drivers de Arquitectura 5

2.1 Objetivos de la arquitectura de la solución 5

2.2 Requerimientos Funcionales Significativos 5

2.3 Restricciones 5

2.4 Atributos de calidad 5

**2.4.1** **Descripción de atributos de calidad** 5

3. Contexto 6

3.1 Fundamentos de la solución 6

3.2 Tecnología seleccionada 8

4. Vistas de arquitectura 9

4.1 Escenario 1: CRUD de Productos 9

4.1.1 Vista de procesos 9

4.1.2 Vista de datos 9

4.1.3 Vista de desarrollo 10

4.2 Escenario 2: CRUD campañas 10

4.2.1 Vista de procesos 10

4.2.2 Vista de datos 10

5. Riesgos de la arquitectura 12

# Introducción

Para la compañía Toures Balón es de vital importancia que la funcionalidad de aplicación se encuentre alienada con las intenciones de crecimiento y con el negocio en general, pero también tiene una relevancia importante que los requerimientos no funcionales y las especificaciones sobre es desempeño, escalabilidad e integridad de la aplicación cuenten con estándares altos de calidad. Este documento muestra las diferentes vistas del proyecto que son requeridas por cada interesado, se pretende llegar al mayor numero de personas posible con las vistas acá expuestas.

## Visión

Esta propuesta de arquitectura soportará las necesidades de Toures Balón, las estrategias con mayor impacto del negocio se impulsarán con diferentes tecnologías, lenguajes de programación y patrones de software. [*Sentar la visión sobre el proyecto para eliminar ambigüedades*]

## Objetivo

Identificar los escenarios necesarios que soportan las estrategias principales del negocio, y cómo en estos escenarios la arquitectura empresarial pueden aportar valor agregado para impulsar e incrementar la productividad de cada proceso de negocio.

## Definiciones, Acrónimos y Abreviaturas

* **XML** Extensible Markup Language (Lenguaje de Marcado Extensible)
* **JSON** Formato de texto sencillo para el intercambio de datos.
* **HTTP** Protocolo de transferencia de hipertexto es el protocolo de comunicación que permite las transferencias de información en la World Wide Web
* **REST** La transferencia de estado representacional
* **SLA** (Service Level Agreement << Acuerdo de Niveles de Servicio >>).
* **RAM** (Memoria de Acceso Aleatorio).
* **CPU** (Unidad Central de Proceso).
* **B2C** Negocio a consumidor o B2C (del inglés business-to-consumer) se refiere a la estrategia que desarrollan las empresas comerciales para llegar directamente al cliente o consumidor final.
* **Métricas** Elementos de medición para supervisar el rendimiento del sistema y evaluar su estado.
* **Request** Solicitud directa a un servicio para iniciar un intercambio de datos.
* Response Respuesta dada por un servicio.
* **Troughput** La tasa de transferencia efectiva es el volumen de trabajo o de información neto que fluye a través de un sistema, como puede ser una red de computadoras.
* **IP** (Protocolo de internet) un número que identifica, de manera lógica y jerárquica, a una Interfaz en red de un dispositivo que utilice el protocolo

## Referencias

[1]. Cloud computer pattern, https://patterns.arcitura.com/cloud-computing-patterns.

[2]. Amdahl’s law, https://en.wikipedia.org/wiki/Amdahl%27s\_law.

[3]. SOA Patterns, <https://patterns.arcitura.com/soa-patterns>.

## Alcance

Este documento refleja las decisiones de arquitectura para la compañía Toures Balón, un demo simulando un par de sus funcionalidades será entregado con el fin de dar a conocer como se materializan algunas de las decisiones de arquitectura.

# Drivers de Arquitectura

## Objetivos de la arquitectura de la solución

Proveer como entidad líder en desarrollo una solución de arquitectura empresarial que permita soportar la estrategia de negocio a corto, mediano y largo plazo, brindando disponibilidad e integridad de la información de los servicios brindados por Toures Balones.

* Reutilizar los recursos ya existentes de la compañía en la nueva propuesta arquitectónica presentada por Mote E’ Queso.
* Garantizar a través de la solución de arquitectura tiempos de respuesta conforme a las necesidades del negocio durante las visitas al catalogo y compras de los usuarios.
* Rediseñar los procesos de negocio para el pago electrónico, desplegando una solución que permita respaldar las estrategias del negocio.

## Requerimientos Funcionales Significativos

A continuación, se presentan los requerimientos más significativos y retadores a la hora de tomar las decisiones de arquitectura:

* Automatización de procesos en su máxima expresión.
* El sistema debe estar en capacidad de responder ante temporadas altas, Black Friday, Ciber Monday, entre otros.
* Tener capacidad de agregar nuevos proveedores/convenios/alianzas sin traumatismo.
* Las reglas de negocio deben ser modificadas sin redesplegar el sistema.
* Recolección de analíticas de los usuarios y servidores para poder tomar decisiones futuras.
* Implementar el mecanismo de subasta inversa para todos los proveedores.

## Restricciones

* Reutilización de bases de datos existente, SQLServer y Oracle.
* Lenguajes de programación principal: Java y .Net Core.
* El negocio espera que el catalogo de ventas no se presente al usuario en menos de 1.5 segundos ya que estudios demuestran que tiempos mayores a estos disminuirían las ventas en los diferentes productos.
* El negocio espera que el resto de solicitudes se presente al usuario en menos de 3 segundos ya que estudios demuestran que tiempos mayores a estos disminuirían las ventas en los diferentes productos.

## Atributos de calidad

### **Descripción de atributos de calidad**

La empresa Toures Balon lleva varios años en el mercado y no dejará a la suerte los requerimientos de la compañía, por lo cual ha establecido cuales deben ser los atributos de calidad que el negocio espera y por lo tanto la arquitectura debe ser soporte para las estrategias que ellos pretenden implementar a corto, mediano y largo plazo. A continuación, se describe uno a uno de los requerimientos que se extraen del negocio.

#### Seguridad

**Seguridad\_001**: La comunicación entre los servicios, aunque sea interna, requiere utilización de comunicación TSL para la protección de mensajes, ya que los datos que se transmiten son sensibles.

**Seguridad\_002**: Todo el tratamiento y procesamiento de la información de pagos debe cumplir con la norma PCI.

#### Disponibilidad / Escalabilidad

**Disponibilidad\_001**: se espera disponibilidad del 99.999 del sistema de catálogos y compra de productos.

**Disponibilidad\_002**: se espera disponibilidad del 99.999 del sistema de productos.

**Disponibilidad\_003**: el resto de los sistemas se requieren que estén disponibles, pero pueden tolerar fallos.

#### Escalabilidad

**Escalabilidad\_001**: El sistema debe estar en capacidad de incrementar de forma automática la carga que soporta en temporadas altas, esto quiere decir que ante cierta carga o cuando este llegando al punto de estrés, aumente elásticamente sus capacidades.

**Escalabilidad\_002**: Definir una arquitectura escalable que soporte la integración de nuevos proveedores con el menor impacto para el sistema.

#### Interoperabilidad

**Interoperabilidad\_001**: El servicio de pagos debe soportar pagos por VISA, MasterCard, AMEX, Discover y PayPal; para el caso de Colombia también PSE

**Interoperabilidad**\_**002**: Aplicación multiplataforma que pueda ejecutarse en computadores, tabletas y celulares (Omnicanalidad).

**Interoperabilidad**\_**003**: Arquitectura orientada para ser desplegada en la nube.

#### Performance

**Performance\_001**: Las peticiones a consultas de producto no deben ser superiores a 1.5 segundos.

**Performance\_002**: La solución debe tener tiempos de respuesta menor a 3 segundo para todas las solicitudes restantes.

**Performance\_003**: Implementación clúster de réplica de base de datos para módulos de consulta y analítica (reportes).

#### Mantenibilidad/Manejabilidad

**Mantenibilidad\_001**: Las funcionalidades de negocio serán implementados a través de microservicios, de esta forma garantizar la mantenibilidad de la aplicación.

**Mantenibilidad\_002**: Usar aplicación en contenedores hace que realizar despliegues o trabajar en servicios específicos del sistema sea mas fácil a la hora de desarrollar y desplegar cualquier nueva funcionalidad.

#### Reusabilidad

**Reusabilidad\_001**: la implementación de microservicios nos brinda la posibilidad de reutilizar funcionalidades responsables de una parte del negocio, en nuevas implementaciones de la empresa.

#### Facilidad de soporte

**Facilidad\_Soporte\_001**:

#### Usabilidad

**Usabilidad\_001**: Interfaz orientada a permitir un seguimiento paso a paso para la compra y pago de ordenes una vez se encuentren el carrito de compras, evitando la fricción entre el usuario y proceso de compra.

# Contexto

Toures Balón es una compañía con varios años de experiencia en la industria de organización de eventos deportivos, se ha destacado por tener los mejores precios del mercado y siempre tiene convenios con los mejores hoteles, empresas transportadoras y clubes deportivos, debido a eso y a una renovación tecnológica que se encuentran apostándole, han decidido contratar a Mote e’ Queso Company para desarrollar esta propuesta.

## Fundamentos de la solución

Objetivo Arquitectónico para implementar:

**SOA (arquitectura orientada a servicios)**

La solución propone una arquitectura basada en microservicios con la cual se pueda realizar desacoplamiento entre los servicios, permitir escalar de forma que la carga sea mayor en momentos de mayor concurrencia de usuario, fácil adaptación ante anomalías de eventos como Black Friday, 11/11, entre otros ante los cuales el sistema debe responder de forma oportuna y sin degradarse

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ítem** | **Problema** | **Fuerza** | **Consecuencia** | **Descripción** |
| Decisión\_001 Microservicios [3] | Escalabilidad limitada, monolíticos. Redesplegar nuevas versiones de un servicio que se encuentra dentro de un monolítico. | Escalabilidad.  Seguridad.  Disponibilidad.  Reusabilidad. | Los servicios pueden ser desarrollados y evolucionados de forma independiente. Aumenta el uso de memoria y red cuando se comunica con otros servicios. | Al implementar este patrón se pretende tener mayor granularidad, minimizando el tiempo de nuevas actualizaciones, desarrollos, separando puntos de falla, entre otros y sobre todo poder soportar al negocio en lograr su objetivo de disponibilidad y en caso de temporadas altas escalar los servicios requeridos. |
| Decisión\_002 Modelo Vista-vista Modelo | Implementación a nivel de interfaz | Desacoplamiento.  Reusabilidad.  Eficiencia. | Mantenibilidad alta. | Al desacoplar el back-end del front, se brinda la posibilidad al negocio de interactuar fácilmente y sin traumas con los diferentes servicios de Toures Balón, así como también apoyar el requerimiento de la fácil integración con múltiples proveedores. |
| Decisión\_003 Capas | Aplicaciones monolíticas.  Acceso descontrolado entre capas. | Desacoplamiento de componentes.  Mayor gobierno de aplicaciones.  Escalabilidad. | Mantenibilidad alta. | Para mantener un nivel de abstracción entre los componentes se proponen servicio a la siguiente capa. |
| Decisión\_004 Encolamiento asíncrono | Desempeño comprometido en el cliente, cuando el usuario invoca el servicio los recursos de las maquinas son bloqueados. | Desempeño.  Fiabilidad.  Escalabilidad.  Recursos. | Manejo de acuse de recibido requiere mayor esfuerzo.  Transacciones atómicas no son posibles. | Se utilizará Rabbit MQ para encolar las peticiones que se requieran manejar de forma asíncrona y se liberaran de esta forma los recursos tanto del usuario como del lado del servidor. |
| Decisión\_005 Implementación redundante | Componentes del alto impacto en la solución y que no cuentan con redundancia. | Redundancia.  Alta disponibilidad. | La complejidad para gobernar las aplicaciones incrementa. | Para evitar un punto único de fallo, se implementará este patrón. |
| Decisión\_006 Inventario de dominio | La compañía tiene múltiples dominios de negocio, se debe identificar como trabajar de forma estándar para mejorar el desempeño. | Gobierno.  Estandarización. | Desempeño si los dominios utilizan diferentes estándares, por lo que se debe realizar transformación. | Ya que la compañía posee diferentes frentes ante el negocio, lo mejor es iniciar la estandarización por dominios. |
| Decisión\_007  Centralización de políticas | Aplicación de políticas y contratos individuales. | Consistencia de políticas.  Gobierno. | Desempeño.  Dependencia en las tecnologías propietarias. | Mantener las mismas políticas a través de todos los servicios con el fin de estandarizarlas. |
| Decisión\_008 Servicio de escalamiento automático [1] | Nuevos clientes pretenden consumir el servicio, pero el sistema esta a tope. | Escalabilidad.  Alta disponibilidad.  Recursos.  Desempeño. | Costos.  Ley de Amdalh. [2] | Mantener la disponibilidad y escalabilidad del sistema, este patrón, cuando la aplicación es desplegada en la nube, nos facilitará rápidamente ante eventos de incremento de carga. |
| Decisión\_009 Encapsulamiento de errores | Los errores no se controlan y se presenta gran parte de la arquitectura y como es la implementación al usuario final. | Seguridad.  Usabilidad.  Fiabilidad. | Rastrear los errores se dificulta, porque al usuario no se le proporciona mucha información en los errores. | Evitar que los usuarios no vean como esta implementada la solución y evitar que la aplicación tenga errores visuales o ciclos infinitos, haciendo el uso un problema para el usuario. |
| Decisión\_010 Contenedor [3] | Servicios desplegados en servidores o maquinas virtuales suelen consumir grandes recursos a nivel de SO, Hardware. | Escalabilidad.  Alto rendimiento | Impone requerimientos en infraestructura tecnológica adicionales y aumenta la complejidad de administración. | En la forma de despliegue tradicional y cuando se construyen aplicaciones sobre sistemas bare metal o virtualizados, se consumen muchos recursos. |
|  |  |  |  |  |

## Tecnología seleccionada

Bajo los requerimientos de la compañía se ha decidido implementar la solución con la siguiente tecnología:

|  |  |
| --- | --- |
| **Aplicación** | **Descripción** |
| Oracle database | Aprovechando la tecnología ya adquirida por la compañía |
| SQLServer | Aprovechando la tecnología ya adquirida por la compañía |
| Docker | Tecnología para enjaular aplicaciones |
| Sprig boot | Framework web para microservicios |
| JEE | Java Enterprise Edition soportando spring boot |
| Python | Lenguaje de programación complementario para la autenticación |
| .Net Core | Lenguaje de programación secundario para soportar los diferentes servicios |
| JWT | Estándar para la creación de tokens de acceso que permiten la propagación de identidad y privilegios o claims |

# Vistas de arquitectura

## Escenario 1: CRUD de Productos

### Vista de procesos

A picture containing screenshot

Description automatically generated

Figure 1 Vista de procesos CRUD productos

[1. ]

### Vista de datos

A screenshot of a cell phone

Description automatically generated

Figure 2 Vista de datos Productos

### Vista de desarrollo

A screenshot of a cell phone

Description automatically generated

Figure 3 Vista de desarrollo Productos

## Escenario 2: CRUD campañas

### Vista de procesos

A screenshot of a cell phone

Description automatically generated

Figure 4 Vista de procesos Campaña

### Vista de datos

**Clientes y Ordenes:**

Imagen que contiene captura de pantalla

Descripción generada automáticamente

Productos:

Imagen que contiene captura de pantalla

Descripción generada automáticamente

# Riesgos de la arquitectura

|  |  |  |  |
| --- | --- | --- | --- |
| **Riesgos** | **Estrategia de mitigación** | **Plan de contingencia** | **Impacto** |
| Perdida de información por reutilización de bases de dato ya existente. | Implementacion de elasticsearch como motor de búsquedas | Elasticsearch no es un motor de base de datos pero puede servir como guía para restaurar la información. | Perdidad de información importante para el negocio. |
| Perdida de información por problemas en las bases de datos ya existente. | Backups periódicos de base de datos | Restauración del backup mas inmediato | Perdida de información hasta el backup restaurado |
| Integración con nuevos proveedores. | Contratos y API para facilitar la integración. | Adaptar un Middleware para casos especiales. | Tiempo de desarrollo aumenta, complejidad, mantenibilidad. |
| Caída de servicios y mantenimiento de estos | Utilizar un orquestador de servicios y health check para el estatus de cada uno generando un monitor de servicios | Balanceo de servicios y orquestador | Caída parcial de las capacidades y disponibilidad del negocio. |