Facultad de Ingeniería - UDELAR

Proyecto de grado
Ingeniería de sistemas

Herramienta computacional para el cálculo de confiabilidad estructural de tuberías metálicas fisuradas

Dario Britos, Victor Díaz

Tutores
Dr.Ing. Héctor Cancela
MSc. Rodolfo Mussini

23 de octubre de 2018

Dedicado a nuestros amigos, nuestra familia, y en lo personal (Dario Britos) a mi madre, Martha Vanoli.

Agradecimientos

A quienes de alguna forma u otra fueron y son parte de nuestro desarrollo personal y profesional-

Siglas

API API 579-1/ASME FFS-1. 5, 31

API American Petroleum Institute. 5

AWS Amazon Web Services. 60–62, 64

FAD Failure Assessment Diagram. 15–17, 19, 31, 70

GPL General Public License. 49

MVC Model-View-Controller. 44

NEA Nuclear Energy Agency. 6

ORM Mapeo objeto-relacional. 48

OSS Open Source Software. 45

Resumen

Este trabajó esta centrado en el desarrollo de una plataforma web dedicada al cálculo de confiabilidad estructural de tuberías metálicas fisuradas. Comenzamos con el análisis del estado del arte en base a relevar diferentes sistemas utilizados en el mercado para cálculos de esta índole. Se realiza un relevamiento de requerimientos para el sistema a desarrollar, basado en las características positivas de los sistemas analizados y en características deseables acordadas con los tutores. Se diseño una arquitectura orientada a micro servicios, la cual es escalable horizontalmente. El desarrollo esta basado en tecnologías como Angular, Java, Spring y NetflixOSS y se realizo una implantación productiva en servidores de Facultad de Ingeniería. Además para propósitos académicos en instancias de desarrollo y pruebas, se implanto el producto en la plataforma AWS.
Se entrega además de este documento, un anexo donde se detallan los pasos para una nueva implantación, código fuente y scripts de creación de base de datos.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

INTRODUCCIÓN

1.1. Contexto y Motivación

Todos los sistemas de tuberías deben asegurar:

(i) Seguridad - el sistema debe poseer un valor de probabilidad de falla estructural aceptable-mente bajo (o alta confiabilidad estructural)

(ii) Cumplimiento con códigos y legislación - el sistema debe satisfacer leyes locales y nacionales

(iii) Seguridad de abastecimiento - el sistema debe suministrar en forma continua el producto que circula por su interior

En referencia al punto (i), existen dos grandes métodos para poder cuantificar la confiabilidad estructural de componentes y sistemas de ingeniería. El primer método es el experimental o basado en ensayos y el segundo es el computacional. Este último método, es generalmente usado cuando la ejecución experimental no es factible en términos económicos y/o de ingeniería. La confiabilidad estructural computacional es particularmente valiosa cuando se pretenden analizar grandes sistemas tales como edificios, centrales nucleares y avanzados vehículos espaciales, donde se vuelve imposible construir idénticas réplicas para su ensayo.

1.2. Introducción al trabajo realizado

El presente trabajo busca realizar una contribución a la resolución de la problemática planteada, y por la motivación de desarrollar una nueva herramienta acorde a los tiempos que corren, emprendimos este proyecto con el objetivo de dar el puntapié inicial de una plataforma dedicada al calculo probabilístico de fallas

en componentes mecánicos fisurados.
Desde el comienzo esta como objetivo, el dejar un camino recorrido, para que en un futuro, sea continuado por otros estudiantes, investigadores, o aquellas personas que se sientan motivadas a extender el aporte que realizamos en este trabajo.

1.2.1. Organización del documento

Acompañados por el aporte de conocimiento de los tutores en el área, nos encaminamos en un relevamiento del estado del arte y a este le dedicamos el capítulo 2, el cual concluimos con un listado de características deseables para un sistema de este tipo.
En el capítulo 3 realizamos el análisis de requerimientos para el desarrollo de una plataforma web dedicada al calculo de confiabilidad estructural de tuberías metálicas fisuradas. El capítulo comienza con análisis funcional y no funcional, continua con un listado de casos de uso formal y finaliza con la descripción de los escenarios de fisuras de tuberías que el sistema resolverá.
Continuamos con la descripción de la solución propuesta en el capítulo 4, el cual describe el producto que proponemos y la arquitectura del mismo.
El capítulo 5 esta dedicado a la implementación de la plataforma. Tiene dos partes bien marcadas, la primera mitad esta dedicada a las tecnologías utilizadas y a las descartadas; la segunda mitad a la implementación de los diferentes componentes de la arquitectura.
Dedicamos el capítulo 6 a la implantación de la plataforma. En la primera parte se explica la implantación durante el desarrollo en la infraestructura de Amazon Web Services y en la segunda parte se hace referencia a la implantación productiva. Finalizamos el capítulo con las pautas para generar una nueva implantación.
Se describe el producto final logrado en el capítulo 7, tanto los entregables como las características funcionales de la plataforma. Se cierra el documento con el capítulo 8, en el cual brindamos conclusiones sobre el trabajo realizado y propuestas de trabajo a futuro.

Capítulo 2

ESTADO DEL ARTE

2.1. Introducción

En este capítulo se presentará el relevamiento del estado del arte realizado, considerando tanto el ámbito académico como en el industrial. El desarrollo de un sistema de esta índole lo requiere, para tener un panorama de las lineas de trabajo por las que se ha avanzado, las falencias que puedan existir, las buenas practicas y las características que hacen a estos sistemas. Relevar sistemas existentes brindara visión para la planificación y selección de características a incluir en nuestra solución, así como aquellas que sea mejor evitar. Comenzaremos por establecer cuales fueron los objetivos del relevamiento en la sección 2.2, prosiguiendo con el relevamiento de la información académica y de la industria en general en la sección 2.3. Dada la información relevada de la industria en general se decidió agregar información acerca las normas o procedimientos utilizados actualmente, subsecciones 2.3.1 y 2.3.2. Concluido el relevamiento se presentarán las características deseables de un sistema de esta índole en la sección 2.4. Por ultimo se brindan las conclusiones del relevamiento en 2.5.

2.2. Objetivos

Comenzaremos listando los objetivos planteados para esta etapa, de modo de lograr un relevamiento que cubra los puntos de interés del proyecto:

- Relevamiento de información existente, referente a sistemas que resuelven este tipo de cálculos.
- Confeccionar un listado de sistemas existentes, describiendo sus características así como toda la información relevante que se disponga.

- Experimentar con los sistemas encontrados, de los cuales se pueda disponer ya sea una versión completa o parcial, documentando la experiencia del lado de usuario y las funcionalidades que se relacionen a nuestro trabajo.
- Realizar un estudio comparativo de cualidades y características que aporten a este trabajo.
- Evaluar ventajas y desventajas detectadas, así como características relevantes que puedan o no, aplicar a nuestro producto final.

2.3. Relevamiento de información

Focalizados en un relevamiento en primera instancia de artículos académicos, como ser publicaciones científicas o resúmenes de conferencias. Encontramos que la información relacionada con los sistemas utilizados para analizar problemas de fractura en materiales metálicos, refiere a comparaciones entre sistemas particulares o simplemente son mencionados como herramientas usadas, sin dar mayores detalles de su arquitectura, construcción o funcionalidades de los mismos, que es en lo que estamos focalizados. Un ejemplo de trabajo relacionado a benchmarking es [8]. Para realizar este relevamiento utilizamos los portales Timbó (`http://www.timbo.org.uy/`) y Google Académico (`https://scholar.google.com.uy/`). Dado estos resultados decidimos ampliar la búsqueda al ámbito de la industria en general. Como resultado destacamos los siguientes sistemas:

Sistema	Desarrollado por	Industria	Daño que atacan	Modalidad	Versión	Norma
PC-PRAISE	NEA [5]	Nuclear	Fracturas y fuga de refrigeración de tuberías	Probabilístico		
PRO-LOCA	Battelle [14]	Nuclear	Fracturas y fugas de tuberías de refrigeración nucleares	Probabilístico	4.0	
Signal FFS	Quest Integrity [3]	General	Análisis de la mecánica de la fractura	Probabilístico	5.0.4	API579 BS970
FEA Crack	Quest Integrity [3]	General	Análisis de la mecánica de la fractura	Determinista	3.0	API579
PC-CRACK	Structural Integrity Asociates INC [13]	Nuclear	Crecimiento de la grieta, tamaño crítico.		4.1	
VINDIO	Inesco ingenieros [15]	General	Análisis de la mecánica de la fractura	Determinista	1.1	
CRACK WISE	Twisoftware [10]	General	-	Determinista	5.0	BS970

Cuadro 2.1: Sistemas relevados

Se desprende del relevamiento que:

- Métodos de Monte Carlo es el enfoque elegido por la mayoría de los sistemas evaluados.
- La industria de la energía nuclear es la mas involucrada en sistemas de este tipo.
- Los sistemas relevados estudian probabilidad de falla de fracturas y el crecimiento de las mismas.

Algo que es importante destacar es que no se encontró registro de un Sistema desarrollado bajo licencias de código abierto.

Continuando con el relevamiento, vamos a dedicar los siguientes párrafos para detallar aspectos, a nuestro criterio relevantes, del análisis de los sistemas. Para eso creemos necesario enumerar algunas normas, estándares internacionales de procedimiento para la evaluación de tuberías, las cuales están relacionadas con los sistemas que vamos a listar.

2.3.1. Normas

Una norma es un conjunto de pautas y recomendaciones que se imponen o se adoptan para dirigir la conducta o la correcta realización de una acción o el correcto desarrollo de una actividad. Lo que listamos a continuación son normas establecidas por entidades ligadas al estudio de la mecánica de la fractura, las cuales son adoptadas por varios de los sistemas que se analizarán más adelante. Que un software se base en una norma, implica que al momento de realizar sus cálculos, sigue las pautas y recomendaciones que la norma impone.

API 579 / ASME FFS 1 [6][1]

Establecida por American Petroleum Institute, la API 579 es una norma con la cual se garantiza la integridad estructural de un componente en servicio, que puede llegar a tener una falla o que está operando en condiciones que podrían generar una falla. Esta norma brinda pautas para realizar la evaluación Fitness For Services (FFS) en equipamiento presurizado, y así tener una herramienta para la toma de decisiones respecto al remplazo del componente. La API primero publicó API 579-1/ASME FFS-1 como una práctica recomendada en el 2000, para posteriormente en 2007 publicarla en conjunto con la American Society of Mechanical Engineers (ASME) como una norma.

BS 7910[7]

La British Standards de BSI Group establece la norma BS 7910, que es una guía de métodos para evaluar la aceptabilidad de defectos en estructuras metálicas. La aceptabilidad de estos defectos se basa en el principio de Fitness For Services. Sus métodos pueden ser aplicados tanto en la etapa del diseño, construcción o mantenimiento de la estructura.

2.3.2. Sistemas existentes

PC-PRAISE

PC-PRAISE [5] es un software probabilístico acerca de la mecánica de la fractura, fue desarrollado para ser ejecutado en computadoras personales IBM con el fin de estimar la probabilidad de fugas o roturas de tuberías de refrigeración de centrales nucleares.
Según la Nuclear Energy Agency (NEA), PC-PRAISE para calcular esto, considera las condiciones iniciales y el crecimiento de los defectos de las fisuras de las soldaduras en las tuberías. Las condiciones iniciales son basadas en los resultados de estudios de laboratorio y observaciones de campo de los materiales de tuberías austeníticas (por lo relevado, las cañerías elaboradas con acero austenitico tienen ciertas propiedades que las adecuan al uso en este tipo de sistemas) que son utilizadas bajo condiciones de reactor de agua hirviendo. Dados tales resultados se cuantifica e incorpora la dispersión al modelo probabilístico. Respecto al análisis del crecimiento de la fisura se basa en principios determinísticos de la mecánica de la fractura, donde algunas de las entradas se consideran variables aleatorias. Para generar los resultados de confiabilidad de la soldadura, calcula mediante una simulación con el método de Monte Carlo con muestreo estratificado sobre el tamaño inicial de la grieta. Como características se resalta la posibilidad de tratar de forma independiente dos dimensiones del crecimiento de la grieta, en dirección de la profundidad y la anchura. También se resalta el tratamiento estadístico de las diversas variables asociadas con el modelo de la mecánica de la fractura. Respecto a limitaciones o restricciones se destaca que para un gran número de repeticiones utilizadas en el esquema de simulación de Monte Carlo, el tiempo de cálculo puede llegar a ser prohibitivo, se debe considerar que PC-PRAISE es un software de principios de los noventa, surge del interés de facilitar su uso dado que el programa original (PRAISE) se ejecutaba en mainframes y la idea era que se pudiera ejecutar en pc de escritorio y así abaratar costos.

Se solicitó a la NEA el paquete correspondiente para poder instalarlo y probarlo, y como Uruguay no es miembro de NEA Data Bank el mismo fue denegado.

PRO-LOCA [9]

Desarrollado por la Nuclear Regulatory Commission (NRC), para hacer frente a fallas en tuberías metálicas, es el sucesor de PRAISE. Su desarrollo fue impulsado por la necesidad atacar los accidentes LOCA (Accidentes de Perdida de Refrigerante, por sus siglas en ingles). Basa sus cálculos en simulación de Monte Carlo, y además ofrece la posibilidad de utilizar métodos probabilísticos discretos, como por ejemplo, muestreos por importancia. Desde el 2008, el sistema permite variar el tipo de distribución con el que realizara los cálculos. Las opciones brindadas son:

- Uniforme
- Normal
- Lognormal
- Weibull
- Exponencial
- De valor extremo tipo 2

Provee una librería de materiales, con propiedades pre-cargadas para la ejecución de cálculos. También permite el trabajo sobre soldaduras de diferentes materiales. La librería proviene de un sistema de datos externo a PRO-LOCA, que gestiona y permite el ingreso de nuevos materiales. Cabe destacar que cambios en este sistema requieren que la librería de materiales de PRO-LOCA sea actualizada, dado que la misma no está sincronizada con este sistema, para el cual no se brinda un nombre en la documentación oficial. PRO-LOCA además de esta librería, permite al usuario ingresar materiales manualmente. De la documentación oficial se extrae la siguiente imagen, donde se puede observar el tipo de interfaz gráfica que ofrece el sistema, la cual ofrece un menú superior de estilo “Windows Forms” y expone los datos con un estilo de consola.

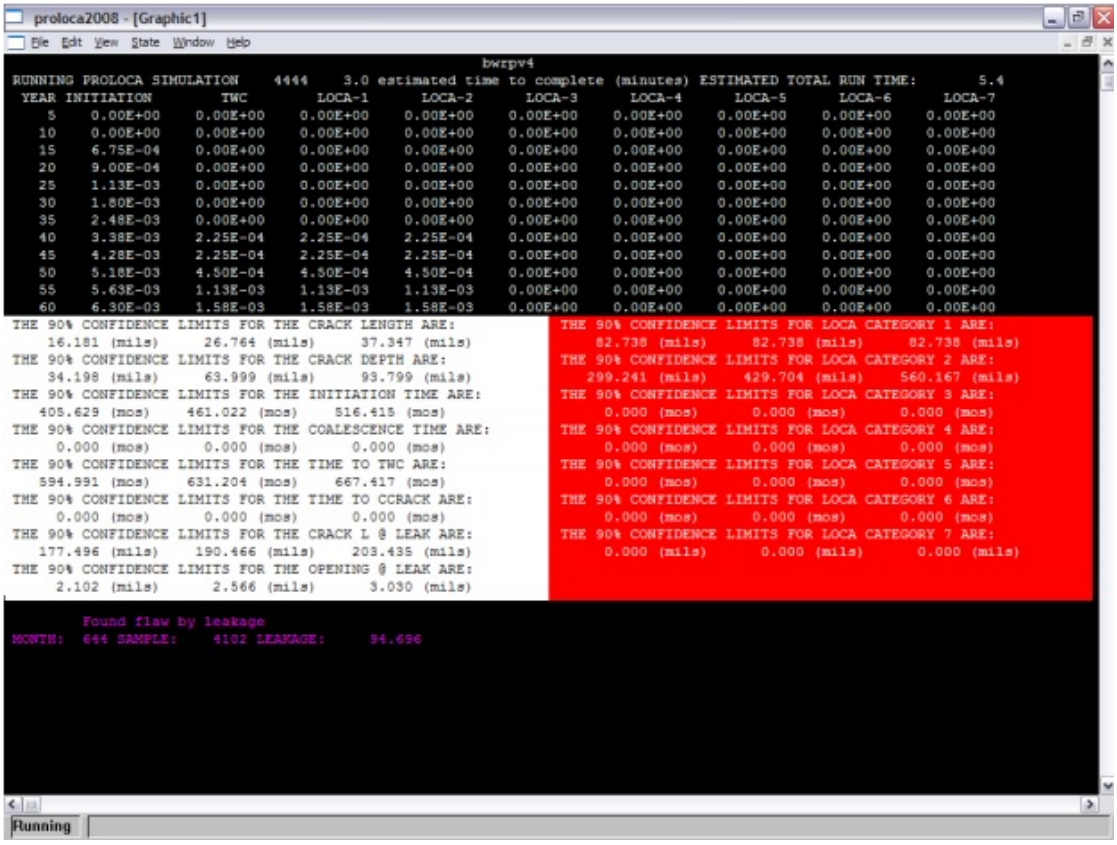

Figura 2.1: Interfaz gráfica de Pro-Loca

Signal Fitness-for-Service Software - FEACrack

Signal FFS es un software desarrollado por Quest Integrity [3], diseñado para analizar la aptitud para el propósito (fitness-for-purpos) y la mecánica de la fractura de equipamientos fijos o rotativos. Ofrece la posibilidad a sus usuarios de reducir riesgos operacionales y de seguridad asociados con el equipamiento, a través del cumplimiento de las normas de la industria. Se destaca que dada la adhesión a las normas API 579/ASME-FFS:

- Se facilita la realización del FFS.
- Provee ayuda en la toma de decisiones, como por ejemplo, sugerencias sobre reemplazo o reparación de elementos.
- Facilidad en la curva de aprendizaje para su uso, tanto por cumplir la norma como por la guía del usuario e interfaces intuitivas.

Además realiza la evaluación de las grietas según el procedimiento de la norma británica BS 7910.

Para representar y facilitar el análisis de la mecánica de la fractura el mismo proveedor de software ofrece el sistema FEACrack, con el cual se pueden realizar gráficas de malla tridimensionales de la fisura, para armarlas el software ofrece una biblioteca con geometrías estructurales y formas de grieta. Es capaz de realizar el análisis de la fatiga, dada una entrada finita de valores. Este software funciona en un entorno Windows y según promociona el proveedor es de fácil aprendizaje y uso.

Sobre este sistema solo se obtuvo la información disponible en su sitio web [3], la cual tiene un alto grado de intención de venta. Al contactarnos con ellos para obtener la versión de prueba que ofrecen, así como también solicitar más información acerca de su sistema, no se obtuvo una respuesta positiva.

PC-CRACK

Desarrollado por Structural Integrity Asociates INC [13], la cual afirma que el software ha sido líder entre aquellos dedicados a las mecánicas de la fractura. Según reza en su web:
El software analiza y predice el comportamiento de las fallas, incluyendo el cálculo de las tasas de crecimiento de grietas y sus tamaños críticos, para recipientes a presión, tuberías, turbinas de vapor y estructuras, con visualización inmediata de los resultados del análisis. Las aplicaciones de PC-CRACK incluyen evaluaciones de fallas de la Sección XI del Código ASME, así como diseño de superposición de soldadura.
La sección XI del Código ASME estipula reglas para la inspección en servicio de componentes de plantas nucleares.

Tienen disponible un archivo de descarga, con el instalador de la versión 4.0 BETA. La cual se entrega en modo de "Demo". Esta permite navegar por todas las opciones, pero habilita la realización de cálculos solo para un conjunto reducido de escenarios. El software esta ensamblado para ejecutar en sistemas operativos Windows. En su carpeta de instalación se pueden ver archivos ejecutables .exe, archivos de librería dinámica .dll, archivos de icono .ico y una guía de usuario, la cual no está disponible en la web. Ofrece una interfaz simple, basada en Windows Forms, con un menú superior donde se ofrecen todas las opciones que brinda el software.

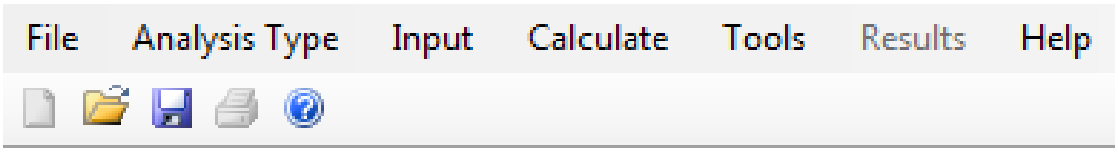

Figura 2.2: Menú principal de Pc-Crack

A continuación se listan funcionalidades a destacar.

1. Guardado y restauración de análisis.
   Como primera funcionalidad a destacar, se tiene la capacidad de guardar y restaurar análisis, así como guardar e imprimir los resultados de la ejecución de un análisis. El sistema guarda los análisis en archivos con extensión .pcf. Al observar uno de esos archivos más detenidamente, vemos que contiene los parámetros de entrada del análisis en texto plano:

```
Sample Problem 1
0,0
1
301
0.1
0.5,2.5
False,0.8,0.05,0,0
2  //Total number of cases
Pressure Stress
0,5,0,0,0
Residual Stress
3
File-
6   //number of x-y pairs
0,1
0.1,12
0.2,11
0.3,7
0.4,-5
0.5,-11
0
0,0,0
```

Figura 2.3: Vista de archivo de análisis guardado

2. Manejo de sistemas de unidades
   Se puede elegir entre el sistema internacional (SI), y el US (United States customary units)

3. Sistema de Asistencia (Wizard)
   Al momento de crear un análisis, se brinda la opción de seguir un wizard, para la configuración de un análisis. Este wizard no es mas que una concatenación de formularios, sin ninguna leyenda particular, el cual es de ayuda para no olvidarse de configurar ningún parámetro.

4. Tipos de materiales
   El sistema permite calcular con diferentes tipos de materiales precargados, incluso permite cargar materiales manualmente, dando al usuario la posibilidad de ingresar los parámetros descriptivos del mismo. La complejidad de esta sección excede nuestro alcance, por lo que no nos vamos a extender en este punto.

5. Categorías de fractura
   Para los tipos de análisis Linear Elastic Fracture Mechanics (LEFM), el sistema da la opción de elegir diferentes categorías de fractura. Se listan a continuación todos y se mostrara el ejemplo de uno a modo de visualización.

   100-Standard Specimens
   101-Standard Compact Tension Specimen
   102-Center Cracked Plate Under Remote Tension
   103-Three-Point Bend Specimen
   200-Plates
   201-Continuous Surface Crack in Half Space
   202-Single Edge Cracked Plate under Tension and Bending
   203-Single Edge Cracked Plate under Arbitrary Loading
   204-Center Cracked Plate Under Arbitrary Loading
   205-Semi-Elliptical Surface Cracked Plate (ASME)
   206-Semi-Elliptical Surface Cracked Plate Under Arbitrary Loading (API 579)
   207-Edge Cracked Disk
   208-Elliptical Subsurface Crack under Tension and Bending (ASME)
   209-Quarter-Elliptical Crack in Plate
   210-Double Edge Cracked Plate Under Arbitrary Loading
   300-Hollow Cylinders
   301- Full-Circumferential Crack in Cylinder on The Inside Surface
   302- Full-Circumferential Crack in Cylinder on The Outside Surface
   303- Longitudinal Crack in Cylinder on The Inside Surface
   304- Longitudinal Crack in Cylinder on The Outside Surface
   305- Semi-Elliptical Longitudinal Crack in Cylinder on The Inside Surface (API 579)

306- Semi-Elliptical Circumferential Crack in Cylinder on The Inside Surface (API 579)
307- Semi-Elliptical Longitudinal Crack in Cylinder on The Inside Surface (Zahoor)
308- Semi-Elliptical Circumferential Crack in Cylinder on The Inside Surface (Zahoor)
309- Semi-Elliptical Circumferential Crack in Cylinder on The Inside Surface (Chapuliot)
310-Through-Wall Axial Crack in Pressurized Cylinder
311-Through-Wall Circumferential Crack in Cylinder Under Tension And Bending
400-Solid Cylinders
401-Exterior Circumferential Crack under Tension and Bending
402-Interior Penny-Shaped Crack under Tension and Bending
403-Edge Crack under Tension and Bending
500-Holes
502-Crack from Hole in Half Plate
503-Edge Crack at Lug Hole
504-Edge Crack at Hole in Finite Plate
600-Nozzles
601-Nozzle Corner Crack
700-Welds
701-Cruciform Crack at Weld
702-Crack at Weld in T-Joint

Nos vamos a detener en la categoría ***"304- Longitudinal Crack in Cylinder on The Outside Surface"***, la cual refiere a un caso de estudio similar al de este proyecto académico. No se pueden obtener resultados de la ejecución de este caso dado que la versión Beta lo tiene restringido.

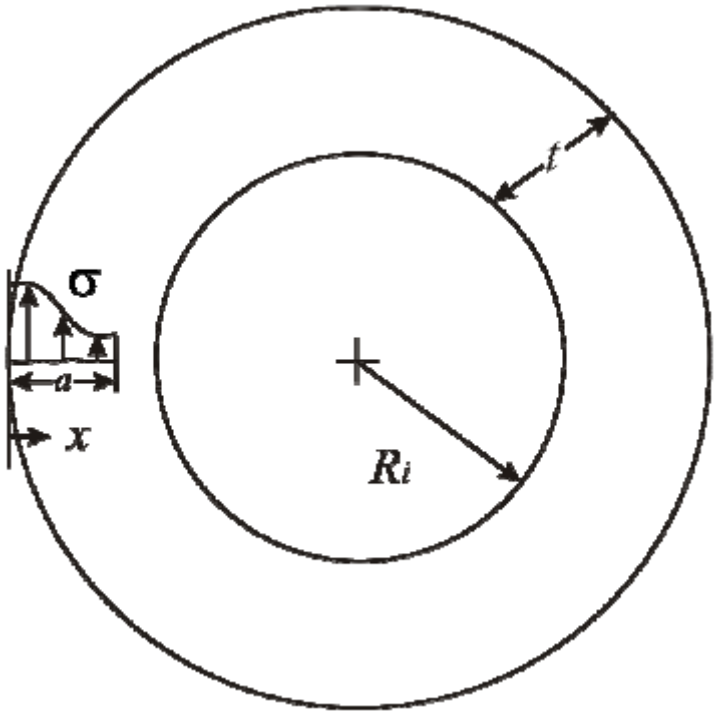

Figura 2.4: Longitudinal Crack in Cylinder on The Outside Surface

El cálculo para este tipo de fractura, requiere las siguientes entradas:
Ri – Radio interno del tubo
a – Profundidad de la grieta
t – Espesor del tubo
También impone las siguientes restricciones:

$$0 < a/t \leq 0{,}8 \tag{2.1}$$

$$0 < Ri/t \leq 1000 \tag{2.2}$$

6. Procesamiento batch
   Se brinda la opción de correr varios análisis previamente guardados en modo batch, donde los resultados son volcados a archivos con extensión .rpt, con el mismo nombre que los archivos de entrada.

VINDIO

Es un sistema desarrollado por Inesco Ingenieros [15], una empresa española fundada en el año 2005, la cual se presenta a si misma en su pagina web con las

siguientes palabras:

INESCO INGENIEROS es una empresa de ingeniería especializada en el ámbito de la integridad estructural, la ingeniería mecánica y la ingeniería de materiales, lo cual le permite prestar servicios avanzados en estos campos a sectores diversos como el de la energía nuclear, las energías renovables, la automoción, la aeronáutica o la ingeniería civil.

Antes de entrar en el detalle de lo que pudimos observar del sistema, vamos a reparar en el producto entero que esta empresa vende, y la manera en la que lo presenta.
Como tarjeta de presentación y punto de entrada tienen su pagina web, la cual vemos es una pagina prolija, dividida por secciones de interés, y un detalle no menor es que está internacionalizada a los lenguajes Español, Inglés y Portugués. Esta pagina esta desarrollada en php, y no presenta un estilo moderno, a pesar de mostrarse de manera correcta en diferentes resoluciones de pantalla. Su diseño no es completamente auto ajustable, y al acceder desde un dispositivo móvil, la pagina no distorsiona su contenido, por lo que es navegable desde este tipo de dispositivos. Al pie de pagina la sección de derecho de autor hace referencia al año 2013, lo cual es congruente con su diseño no tan moderno.

Entrando ya en la sección dedicada a su sistema Vindio, a la cual le dedican una sección de la pagina web de la empresa [16], se observan las siguientes características a destacar:

1. Logo del sistema

2. Párrafo de venta, con las características mas importantes del sistema

3. Vídeos tutoriales con los diferentes modos de uso del sistema

4. Integración con redes sociales
   Mas específicamente con Twitter, donde se muestran todos los Tweets realizados por la cuenta @VindioSoftware. Algo a destacar es que su publicación mas reciente, es del año 2013, mismo año que aparece en el pie de pagina del sitio web. Al parecer este punto no dio los frutos esperados y se dejo de mantener.

5. Proceso de compra y generación de licencia
   Ofrecen una sección de compra, que se maneja con un carro de compra y un proceso de checkout, donde el cliente se registra en caso de no estar registrado, y abona las diferentes opciones de venta del sistema que ellos ofrecen:

a) Vindio 1.1 995.00 Euros

b) Vindio Educacional (10 licencias incluidas + Case-Studies) 1,345.00 Euros

c) Vindio Educacional (25 licencias incluidas + Case-Studies) 1,745.00 Euros

Luego de abonar, se procede a la descarga del sistema. Para tenerlo operativo es necesario realizar pasos adicionales:

a) Descargar un archivo ejecutable, que obtiene datos de la PC donde se va a instalar el sistema

b) Ejecutar dicho programa y guardar el archivo generado.

c) Enviar el archivo guardado a la casilla de correo electrónico: vindio@inescoingenieros.com

d) Luego recibiremos un archivo de licencia, el cual debe ser guardado en la carpeta de instalación del sistema.

6. Blog del sistema
   Ofrecen una sección de blog en la web [2], donde se deja ver la intención de publicar artículos por categoría, al igual que el manejo de redes sociales, todos ellos actualizados por ultima vez en el 2013.

Luego de haber hecho una reseña de puntos a destacar de el producto que venden y como lo venden, vamos a entrar en detalle en lo que al sistema refiere. No se encuentra documentación oficial del sistema. El material al que se tiene acceso son: los vídeo tutóriales y las entradas en el blog de la pagina.

Vindio es un sistema DETERMINISTA, con una interfaz basada en Windows Forms, al cual no se tiene acceso salvo que se pague alguna de las opciones de compra que ellos ofrecen y que enumeramos mas arriba. Se basa en los procedimientos FFS y BS790, y ofrece 4 modalidades de cálculo:

1. Evaluación
   Permite obtener los puntos solución en el Failure Assessment Diagram (FAD), para condiciones geométricas de la fisura, del material y cargas dadas.

2. Búsqueda
   Permite buscar el valor critico de carga y tamaño de una fisura, mediante una iteración del cálculo moviendo la variable dentro de un rango.

3. Barrido
   Permite analizar una serie de valores de carga o de tamaño de fisura entre los limites inferior y superior que indique el usuario.

4. Propagación
   Permite realizar análisis de propagación de fisuras en el tiempo

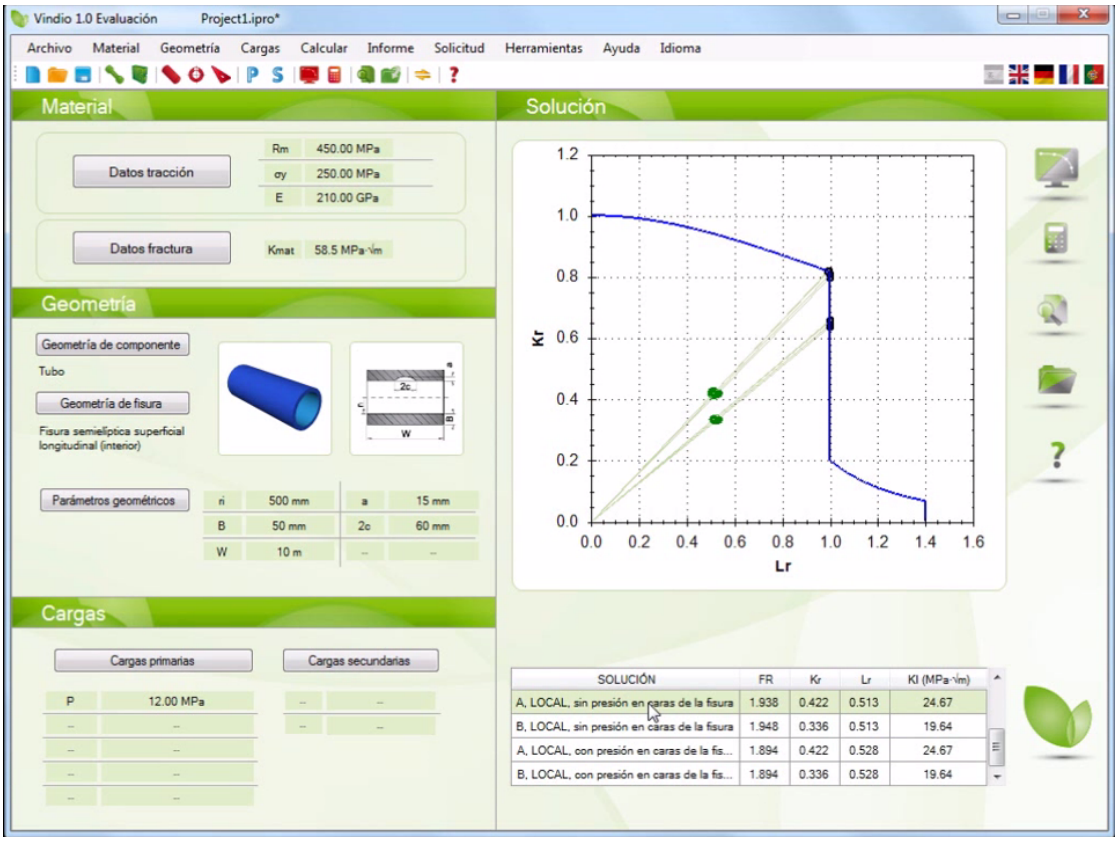

Figura 2.5: Interfaz gráfica de Vidio

A continuación se listan funcionalidades a destacar.

1. Internacionalización múltiple
   Vindio provee internacionalización a 5 lenguajes: Español, Ingles, Alemán, Francés y Portugués.

2. Imágenes descriptivas de figuras y fuerzas y gráficas
   Como se observa en la [figura 2.5], y en todo el proceso de configuración de un caso para calcular, el sistema muestra múltiples imágenes descriptivas de la fractura, componente, y diagramas FAD.

3. Base de datos de materiales categorizados y con filtros
   Al igual que otros sistemas estudiados anteriormente, ofrece una base de datos de materiales, y un filtro de búsqueda en función de las propiedades de los materiales. También mantiene una categorización de los materiales.

4. Carga de geometría de fisura
   Permite cargar la geometría de la fisura a partir de archivos. Se desconoce que aplicación genera este tipo de archivos.

5. Guardado de evaluaciones
   Al igual que sistemas analizados anteriormente, Vindio permite el guardado de evaluaciones.

6. FAD al cargar el material
   Al momento en que se selecciona el material, el sistema muestra su FAD. Esta característica engloba una mas general, y es que el sistema va mostrando imágenes descriptivas a medida que se cargan los inputs.

7. Inputs en categorías
   Divide los inputs en tres categorías: de material, de geometría y de cargas. Se pueden observar en la [figura 2.5]

8. Evaluación de fisuras múltiples
   Permite la carga de fisuras que están cercanas en el espacio, y en función de las normas, las evalúa como fisuras independientes, o como una fisura única.

9. Impresión de reporte de cálculo en pdf
   Luego de ejecutado un cálculo, ofrece la opción de obtener una versión para imprimir, la cual contiene inputs, imágenes, gráficas y tablas de resultados.

CrackWISE

Desarrollado por TWI Software development and training [10], una empresa dedicada al software del rubro integridad estructural desde hace mas de 30 años. Forma parte de una terna de sistemas dedicados al manejo de integridad estructural, en la cual lo acompañan RiskWise [11] e IntegritiWISE [12].

Actualmente en su versión 5.0, cuenta con un instalador para sistemas Windows, y tienen disponible una versión de prueba con las características del software limitadas.
Es un sistema DETERMINISTA, con una interfaz amigable y bien estructurada. Secciona la configuración de un cálculo en "pasos", donde el usuario selecciona

desde el tipo de material hasta el tipo de fractura entre otros.
Dispone de una barra de herramientas superior, donde se destaca una sección llamada "Toolkit", donde se ofrecen funcionalidades extra, como por ejemplo, conversión entre unidades.

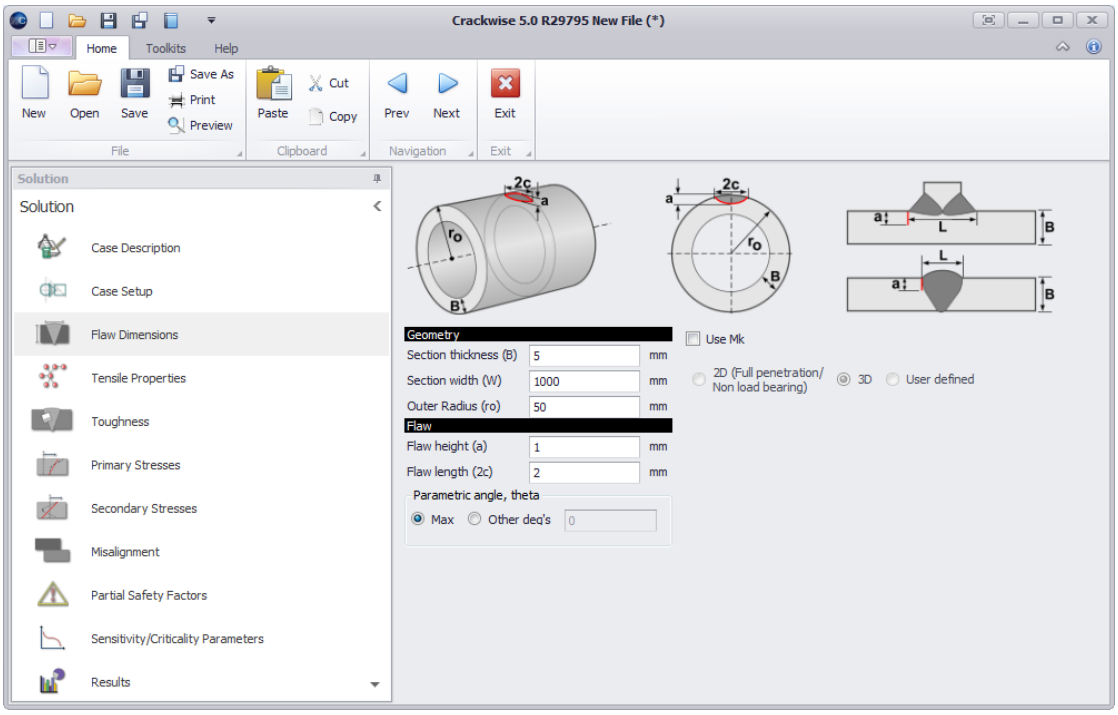

Figura 2.6: Interfaz gráfica de CrackWise

A continuación se listan funcionalidades a destacar.

1. Interfaz
   Su interfaz es amigable y esta estructurada en pasos para la configuración de un caso a calcular. Cada paso recopila información y parámetros de cierto tipo, por ejemplo : Dimensiones de la fractura, Stress, Propiedades del material, etc. En cada paso se controla las pre condiciones de los datos ingresados, dando ayuda textual para los parámetros que no cumplen las pre condiciones.

2. Guardado de evaluaciones
   Las evaluaciones pueden ser guardadas y restauradas desde unidades físicas.

3. Múltiple sistema de unidades
   Permite manejar los sistemas de unidades Internacional y Estadounidense.

4. Imágenes descriptivas paso a paso
   En cada paso de la configuración del cálculo, se muestran imágenes descriptivas de la configuración del mismo, así como diagramas FAD.

5. Impresión de reporte
   Genera una versión para imprimir los resultados de un cálculo, con imágenes de la geometría de la fractura, diagramas FAD, e inputs ingresados.

6. Toolkit
   Provee una sección con funcionalidades extra a la de un cálculo de fractura. Entre ellas un conversor de unidades.

7. Barrido
   Permite seleccionar una de las variables de la configuración de la fractura, y ejecutar un barrido de cálculos, moviendo la variable dentro de un rango.

8. Referencia a las normas
   En la sección de ayuda, en la barra de herramientas, disponibiliza un acceso a las normas en un formato que se desconoce, dado que está disponible para la versión paga.

2.4. Características deseables de un sistema de cálculo de probabilidades de fracturas

Uno de los objetivos principales de esta sección, es tener visión de características deseables en un sistema de esta índole, así como atenuar los puntos débiles observados en este tipo de sistemas. Por lo que vamos a cerrar esta fase con un listado de características deseables (algunas exceden el alcance de este proyecto), las cuales se desprenden de características existentes en los sistemas analizados y también de los puntos débiles de los mismos.

1. Guardado y restauración de análisis
   Esta característica esta en alguno de los sistemas analizados, y creemos que es fundamental a la hora de utilizar un software de esta índole. Permitir al usuario tener una lista de casos guardados, los cuales pueda volver a ejecutar en cualquier momento, con la posibilidad de variar alguno de los parámetros sin tener que reconfigurar todo el análisis desde cero.

2. Librería de materiales
   Nuestro proyecto ataca un tipo de fisura con un tipo dado de material, para el cual están dados los parámetros. Sin embargo una herramienta de este tipo es de gran importancia para un alcance mayor del sistema. Esta característica agregaría un grado de complejidad al desarrollo de los cálculos, dado que las formulas dejarían de ser estáticas y pasarían a ser dependientes de más variables. En uno de los sistemas utilizados, esta característica esta disponible pero tiene problemas de actualización dado que utilizaba valores extraídos de un sistema externo, con una manera forzosa de sincronización. Es deseable tener también, una categorización y búsqueda de materiales para una mejor utilización de los datos.

3. Ingreso de materiales manualmente
   Esta característica va de la mano de la anterior, y creemos que aporta valor que el usuario pueda experimentar con los valores predefinidos para los materiales, dado que para este tipo de cálculos se utilizan fórmulas que no están 100 % probadas, y que van variando con el tiempo debido a ajustes. Creemos que la posibilidad de experimentar siempre aporta al desarrollo y mejora de los métodos utilizados.

4. Variación de las distribuciones de probabilidad
   Esta característica, como la anterior, aporta a la experimentación por parte del usuario.

5. Variación de métodos probabilísticos
   Ídem al anterior

6. Múltiples tipos de fracturas
   Varios de los sistemas estudiados ofrecen una amplia gama de tipos de fractura para trabajar. Aunque queda fuera del alcance de nuestro proyecto, hacer un sistema que sea extensible en ese sentido, estaría alineado con esta característica.

7. Alinearse con estándares
   Así como manejar tipos de fracturas, es deseable alinearse con estándares, de forma de tener una herramienta alineada con la comunidad de investigación y de la industria. Nuestro caso de estudio se basa en parte en la API 579 (por mas detalle ver capítulo 3).

8. Manual de usuario con ejemplos prácticos
   A la hora de relevar sistemas existentes, y entender de alguna forma su funcionamiento y funcionalidades, notamos una gran diferencia entre los que brindaban un manual de usuario claro y con ejemplos prácticos de ejecución, aunque luego en sus versiones beta no permitieran ejecutarlos. Un manual de usuario claro es fundamental para que la herramienta sea aceptada por los usuarios finales. Además, tener dicho material accesible al público, es fundamental para aumentar la cantidad de usuarios del sistema.

9. Página web del producto
   En la actualidad una página web es una carta de presentación. Hay múltiples tecnologías para desarrollar páginas estáticas rápidamente, sin un esfuerzo mayúsculo. Creemos que la relación tiempo invertido / valor agregado, que da una página web para un producto es muy favorable. Esta característica va de la mano de la anterior. Tener el manual de usuario disponible, ya sea para descargar o para visualizar en la página web del producto contribuye a la accesibilidad del mismo.

10. Internacionalización
    Más arriba dedicamos un párrafo a la distribución geográfica de los desarrollos existentes y de la aplicación de los mismos. Y de ese apartado se deduce directamente, que un sistema de estas características tiene que estar internacionalizado, así como la página web del producto en caso de que esta exista.

11. Múltiples sistemas de unidades
    Del punto anterior se desprende el actual, dado que en diferentes regiones se utilizan diferentes sistemas de unidades.

12. Sistema de Asistencia (Wizard)
    Uno de los sistemas analizados brindaba un wizard, el cual marcaba los

pasos a realizar para llevar a cabo un cálculo. Entendemos que una buena documentación con ejemplos claros es aceptable para una herramienta de este tipo y también esta es una característica que puede llegar a aportar a la usabilidad y entendimiento del sistema.

13. Procesamientos por lote
    Es una característica que solo observamos en uno de los sistemas analizados, y entendemos que aporta valor a la hora de ahorro de tiempo, dado que con una simple instrucción se podrían llegar a correr varios casos, para luego realizar comparaciones y/o correcciones en los parámetros de entrada. También proyectando un escenario de uso, el usuario podría recolectar información de varias fracturas y luego ingresar al sistema, cargarlas todas, y ejecutar el cálculo para todas ellas en un solo paso.

14. Multi plataforma
    En la actualidad, existen múltiples sistemas operativos, múltiples tipos de dispositivos, tanto de escritorio, web como Mobile. Tener una aplicación que soporte ejecución multi plataforma es una característica fundamental, de modo que el entorno de ejecución o el hardware con el que disponga una persona o empresa, no sea impedimento para ejecutarla.

15. Interfaz adecuada a la época actual, aprovechando tecnologías para graficado en 2D y 3D.
    Ninguno de los sistemas evaluados tiene una interfaz actualizada, y muy pocos llegan a tener una amigable al usuario. Actualmente hay múltiples tecnologías para el diseño de interfaces gráficas, en los diferentes tipos de plataformas. Tener una interfaz amigable e intuitiva es una característica fundamental en los tiempos que corren.

16. Versión para imprimir de un cálculo
    Dada la ejecución de un cálculo, obtener una versión para imprimir, por ejemplo en formato pdf.

17. Barrido
    Configurar un cálculo, y asignar a una de las variables un rango, para luego ejecutar el cálculo con valores de esa variable tomados de ese rango.

2.5. Conclusiones

Una vez finalizado el relevamiento del estado del arte, se tiene el conocimiento necesario acerca de los avances en el área, las necesidades y las características deseables, para el sistema a construir. Realizado este relevamiento se empezaran

a definir los requerimientos del sistema, los mismos se detallaran en el siguiente capítulo.

Capítulo 3

REQUERIMIENTOS

3.1. Introducción

En este capítulo se definirán y describirán los requerimientos funcionales y no funcionales del sistema, los mismos se desprenden de la propuesta del proyecto de grado, el relevamiento del estado del arte y reuniones con los tutores de proyecto para tal fin. Para especificar las funcionalidades relevantes a brindar se detallan los casos de uso correspondientes y de los escenarios que el sistema podrá resolver se explicara que fin tienen y como se resuelven. Entiéndase por escenario como el conjunto de valores de entrada y tipo de problema a resolver con su determinado procedimiento. Todas estas especificaciones son el punto de partida para la construcción del sistema y en la correctitud de ellas se basa la correctitud del proyecto. En la sección 3.2 se definirán los requerimientos funcionales y en la sección 3.3 los no funcionales, posteriormente se presentaran los casos de uso relevantes en 3.4 continuando con la descripción de los escenarios que podrá resolver el sistema en las secciones 3.5 y 3.6. Finalizando con las correspondientes conclusiones del capítulo en 3.7

3.2. Requerimientos Funcionales

Modularización de Escenario.

El sistema deberá soportar el manejo de múltiples escenarios, si bien el alcance del proyecto es la resolución de un único escenario, es deseable que sea de fácil incorporación uno nuevo.

Librería de Materiales.

El sistema permitirá el ingreso de materiales, con sus correspondientes propiedades y los mismos podrán ser usados en los diversos escenarios del sistema en caso de aplicar.

Ingreso de entradas simples para el Escenario.

El sistema permitirá ingresar los valores de entrada para los diversos escenarios de forma sencilla a través de una interfaz gráfica.

Salida de resultados del Escenario.

El sistema mostrara los resultados obtenidos del calculo de un escenario, incluyendo la probabilidad de falla y la varianza. También se mostrara a modo informativo los valores de entrada del escenario.

Almacenamiento de Escenarios (Identificación de Escenario).

Los escenarios creados serán almacenados y estarán accesibles para futuras consultas.

Adecuación de la Precisión.

El sistema permitirá realizar ajustes en la precisión a utilizar para la resolución de los escenarios.

Administrador de la Semilla.

El sistema permitirá administrar la semilla a utilizar para los escenarios que la requieran.

Administración de Usuarios.

El sistema permitirá realizar el alta, baja y modificación de usuarios. Los mismos estar autenticados y dispondrán de una sesión.

3.3. Requerimientos No Funcionales

Disponibilidad 24 7.

El sistema deberá estar disponible para su uso las 24 horas del día los 7 días de la semana.

Capa de Servicios Autenticada.

Los servicios brindados por el sistema solo responderán ante solicitudes autenticadas.

Escalabilidad en la disponibilidad de resolución de Escenarios.

El sistema deberá ser capaz de ampliar su disponibilidad de resolución de escenarios de forma sencilla.

Resolución de Escenarios en el Servidor.

Los escenarios serán resueltos del lado del servidor.

Internacionalización del Idioma.

El sistema brindara la opción al usuario de elegir que idioma utilizar, en un principio se dispondrá de español e ingles, y se permitirá el ingreso de nuevos idiomas de forma sencilla.

Internacionalización del Sistema de Unidades.

El sistema brindara la opción al usuario de elegir que sistema de unidades utilizar.

3.4. Casos de Uso

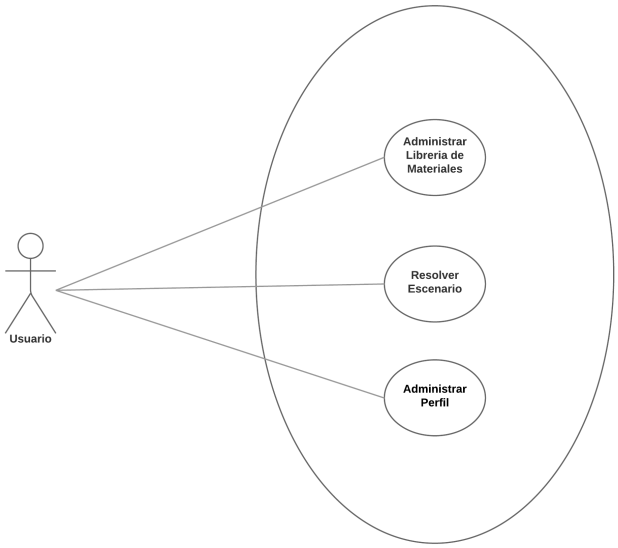

Figura 3.1: Casos de Uso

A continuación se describen los casos de uso así como sus sub casos, en el diagrama se omiten por ser triviales y no agregar claridad.

DESCRIPCIÓN DE CASOS DE USO	
Nombre	Administrar Librería de Materiales
Identificador	CU1
Actores	Usuario
Función	Administrar la librería de materiales del sistema.
Descripción	Permite al usuario dar de alta, modificar o eliminar materiales.
Referencias	-

Cuadro 3.1: Administrar Librería de Materiales

DESCRIPCIÓN DE CASOS DE USO	
Nombre	Resolver escenario
Identificador	CU2
Actores	Usuario
Función	Resolver escenario.
Descripción	Permite al usuario solicitar la resolución de un escenario. Para esto el usuario selecciona un tipo de escenario a resolver, indica los parámetros de entrada a usar, precisión, semilla y solicita su resolución. Luego de resuelto el mismo sera guardado.
Referencias	-

Cuadro 3.2: Resolver escenario

DESCRIPCIÓN DE CASOS DE USO	
Nombre	Alta Material
Identificador	CU3
Actores	Usuario
Función	Alta material en el sistema
Descripción	Permite al usuario dar de alta un nuevo material en el sistema, para esto deberá ingresar sus propiedades con sus correspondientes valores.
Referencias	CU1

Cuadro 3.3: Alta Material

DESCRIPCIÓN DE CASOS DE USO	
Nombre	Modificación Material
Identificador	CU4
Actores	Usuario
Función	Modificación de material en el sistema
Descripción	Permite al usuario modificar materiales en el sistema, para esto podrá ingresar nuevas propiedades del material con sus correspondientes valores, así como editar las existentes.
Referencias	CU1

Cuadro 3.4: Modificación Material

DESCRIPCIÓN DE CASOS DE USO	
Nombre	Baja Material
Identificador	CU5
Actores	Usuario
Función	Baja de material en el sistema
Descripción	Permite al usuario dar de baja un material en el sistema, para esto el usuario selecciona el material a dar de baja y confirma la misma.
Referencias	CU1

Cuadro 3.5: Baja Material

DESCRIPCIÓN DE CASOS DE USO	
Nombre	Administrar Perfil
Identificador	CU6
Actores	Usuario
Función	Administrar Perfil.
Descripción	Permite al usuario editar sus datos así como sus preferencias.
Referencias	-

Cuadro 3.6: Administrar Perfil

3.5. Confiabilidad estructural de tuberías fisuradas basada en el diagrama de evaluación de falla

Se requiere implementar la resolución del escenario confiabilidad estructural de tuberías fisuradas basada en el diagrama de evaluación de falla, para esto se establece como guía y referencia la publicación Failure assessment of Cracked Pipes Based on Failure Assessment Diagram Considering Random and Fuzzy Uncertainties [17]. A continuación detallaremos los aspectos mas relevantes sobre la publi-

cación. En la misma se propone un modelo de calculo basado en el diagrama de evaluación de fallas (FAD) de la API 579-1/ASME FFS-1 (API) y considerando la combinación de la teoría de probabilidad con la teoría de la posibilidad para tratar variables confusas y aleatorias. Esto según los autores, dado que en la práctica existen dos tipos de incertidumbres las aleatorias o las confusas, siendo que algunos parámetros son de una u otra naturaleza. Para resolver esto los autores de la publicación proponen el procedimiento de evaluación de fallas de fisuras de la siguiente manera. Toman como limite de estado la función $g(x) = f(L_r) - K_r$, donde.

$$f(L_r) = \begin{cases} (1 - 0{,}14L_r^2)[0{,}3 + 0{,}7exp(-0{,}65L_r^6)], L_r \leqslant L_r^{max} \\ 0, L_r > L_r^{max} \end{cases}$$

Donde L_r es la relación de carga, K_r es la relación de tenacidad y $f(L_r)$ es la curva de evaluación de fallas, si la evaluación es $g(x) > 0$ la tubería es segura, de lo contrario significa que ocurre una falla. Entonces la probabilidad de falla de la tubería la expresan como:
$P_f = P(g(X) \leqslant 0)$,
A partir de esto y de acuerdo al procedimiento establecido en la API, se puede escribir la probabilidad como una integral basada en la función de densidad de probabilidad conjunta de L_r y K_r con esto los autores de la publicación llegan a resolver el calculo de la probabilidad de falla utilizando el método de Monte Carlo. Para mas detalle sobre como se llega al calculo concreto, se puede consultar la publicación original [17].

Luego de analizada la publicación, llegamos a la conclusión de que omitieron explícitamente la formula para hallar el valor de bending stress. Luego de alertado esto a los tutores, Rodolfo Mussini, nos brindo una formula alternativa para hallar este valor:

$$\sigma_b = \left[\frac{PR_0^2}{R_0^2 - R_i^2}\right] * \left[\frac{t}{R_i} - \frac{3}{2}\left(\frac{t}{R_i}\right)^2 + \frac{9}{5}\left(\frac{t}{R_i}\right)^3\right]$$

Además, se encontró una inconsistencia en la formula A2 de la publicación de referencia. La formula corregida se muestra a continuación:

$$K_1 = \sqrt{\pi \frac{a}{1000}}\left(\sigma_m f_m + \sigma_b f_b\right)$$

Siguiendo con los requerimientos, se establece que tanto la semilla como la precisión del escenario sean ingresadas por el usuario, y que los parámetros de entrada puedan ser valores tantos concretos como probabilísticos, indicando su distribución y parámetros, estos parámetros de entrada según lo especificado en la publicación. Respecto al resultado de este escenario, se requiere que se muestre la probabilidad de falla resultante, y que con la misma se visualicen los parámetros

de entrada.
Mostramos a continuación una imagen del escenario en cuestión:

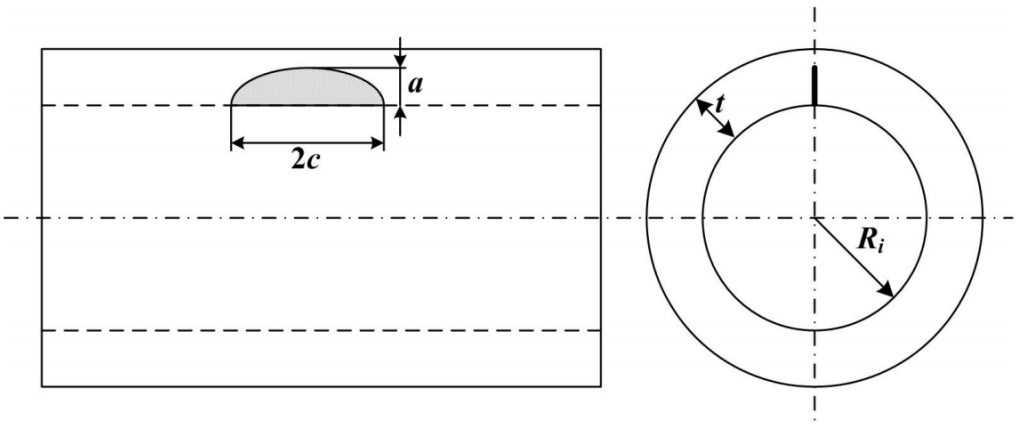

Figura 3.2: Confiabilidad estructural de tuberías fisuradas basada en el diagrama de evaluación de falla

3.6. Ejemplo de confiabilidad estructural de barra de sección circular sometida a tracción

Se requiere implementar el escenario de confiabilidad estructural de barra de sección circular sometida a tracción, con el fin de disponer un segundo escenario simple y de fácil interpretación, para demostrar la correctitud de la generación de números aleatorios así como las simulaciones de las diversas distribuciones ofrecidas. En este escenario se podrán ingresar tanto sus parámetros de entrada como la semilla y precisión, respecto al resultado sera análogo al escenario anterior. Este escenario consiste en resolver la confiabilidad estructural de barras de sección circular sometidas a tracción, esto implica la comparación de dos fuerzas colineales opuestas (T y T_y) y según la resultante de estas, determinar la probabilidad de falla, para esto se utiliza el siguiente calculo para hallar T:

$$T = \frac{\vec{F}}{A_0}$$

con

$$A_0 = \pi\left(\frac{d}{2}\right)^2$$

En donde los parámetros son:

- Diámetro de la barra, d
- Carga a la que es sometida, $\vec{F}$
- Tensión de fluencia del material, T_y

Sabiendo que si se mantiene la relación $T \leq T_y$ el escenario es seguro, y que la probabilidad de falla surge del conteo de casos seguros de la muestra generada.

Mostramos a continuación una imagen del escenario en cuestión:

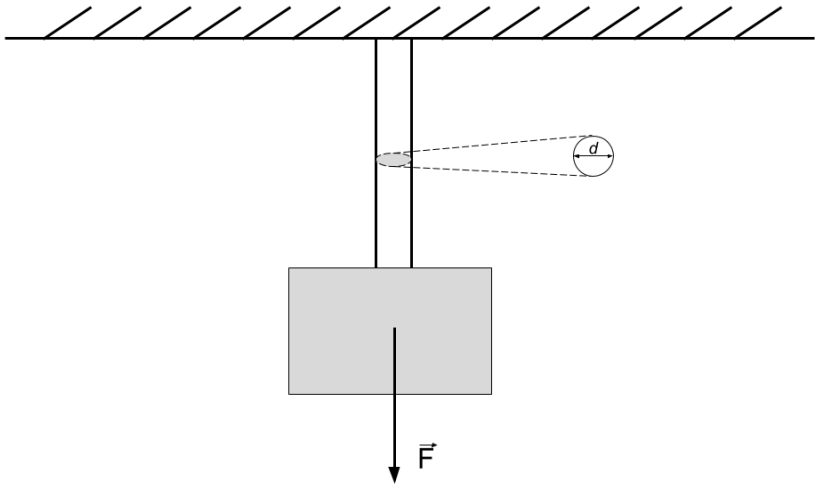

Figura 3.3: Confiabilidad estructural de barra de sección circular sometida a tracción

3.7. Conclusiones

Una vez expuestos todos los requerimientos relevados, sugeridos y pactados con los tutores, se tiene la primera definición técnica, que nos permite tener los requerimientos completos y sin ambigüedades. De los requerimientos el concerniente a la resolución del escenario de la sección 3.5 requirió como se expresa previamente apoyo por parte de los tutores, para obtener información necesaria para su resolución posteriormente. Pudiendo así darle un cierre a lo solicitado y a lo comprometido a alcanzar, pasando así a el análisis y diseño de la solución a implementar, sobre la misma se detallara en el siguiente capítulo.

Capítulo 4

SOLUCIÓN PROPUESTA

4.1. Introducción

En este capítulo se describirá la solución propuesta y acordada con los tutores del proyecto, describiendo tanto características generales del sistema, las concretas de los componentes del ecosistema, sus interacciones y su rol. Comenzaremos por proponer las características generales del producto en la sección 4.2 para continuar indicando la arquitectura que tendrá la solución en la sección 4.3, detallando los componentes de la misma en la subsección 4.3.1. Finalizando con las correspondientes conclusiones del capítulo en 4.4

4.2. Producto

Se propone implementar una plataforma web, respaldada por un ecosistema de aplicaciones, con responsabilidades distribuidas, siguiendo una arquitectura orientada a micro servicios [4], la cual permitirá escalar horizontalmente sus componentes de forma dinámica. El producto sera instalable tanto en un servidor individual como en un entorno de nube, y se apoyara en un servidor de base de datos relacional. La plataforma web se adaptara a los diferentes tamaños de pantalla, incluyendo los dispositivos móviles.

Se apuntara a:

- Un esquema de micro servicios RESTful.
- Arquitecturas simples con bajo acoplamiento.
- Responsabilidades de los micro servicios bien definidas.
- Proceso simple de instalación y puesta en producción

- Uso de tecnologías actuales con proyección a futuro.
- Minimizar esfuerzos en trabajos futuros para la inserción de nuevos tipos de escenarios de calculo.
- Dejar un panorama claro de mejoras y trabajos a futuro, de modo de incrementar las funcionalidades del ecosistema.
- Dejar una instalación productiva para uso de los interesados.

4.3. Arquitectura

Como se menciona en la sección anterior, la arquitectura diseñada está orientada a un esquema de micro servicios, con interfaces RESTful y respaldo en un servidor de base de datos relacional.
En el siguiente esquema se muestran los micro servicios del ecosistema.

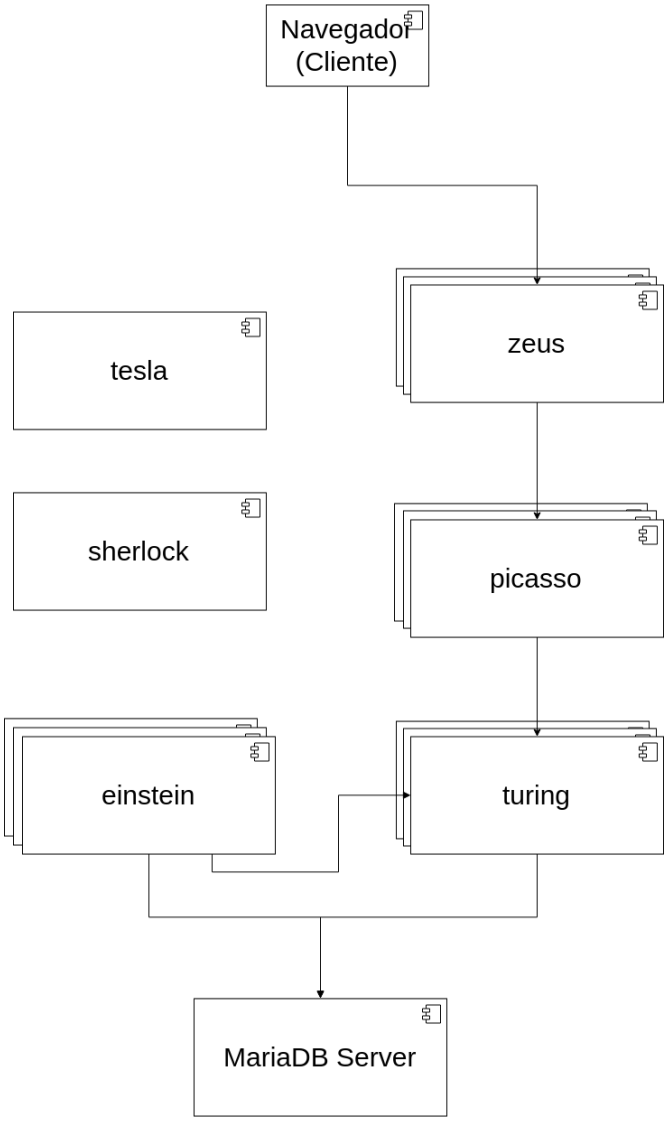

Figura 4.1: Diagrama de micro servicios

4.3.1. Componentes

***Nota**: En la sección 5.2.2 se detallan tecnologías en las cuales se basan los componentes de nuestra arquitectura.*

Tesla

Tesla es nuestro servidor de configuraciones, basado en el Config Server de Spring. Cada componente de la arquitectura, al momento de arranque, consultara

con este servidor su archivo de propiedades. Dichos archivos están alojados en un repositorio git, en la plataforma GitLab. De esta forma, propiedades criticas, como credenciales de bases de datos, no están en el código fuente, sino que son solicitadas al servidor de configuraciones al momento de arranque. Este componente debe ser el primero en activarse al momento de poner en producción la arquitectura.

Sherlock

Nuestro servidor de descubrimiento. Basado en Eureka de SpringCloud, será quien gestione las instancias activas de los diferentes micro servicios de la arquitectura. Este servidor provee un panel donde se puede ver el estado actual de los componentes. En nuestra solución optamos por tener solo un servidor de descubrimiento, aunque la tecnología permite armar un cluster con N servidores de este tipo y de esa manera escalar.

Zeus

Zeus es nuestro servidor de arista, encargado de redirigir todos las llamadas a servicio desde afuera de la red interna, hacia el micro servicio que corresponda. En nuestra solución, es un pasa manos directo con picasso. En una instalación rigurosa de nuestra arquitectura, este servidor debería ser el único que expone un puerto hacia afuera de la red, todos los demás deben ser internos a la misma. Zeus utiliza a Ribbon como balanceador de carga, el cual utiliza nuestro servidor de descubrimiento para localizar una de las N instancias de picasso que puedan estar activas.

Picasso

Este componente contiene nuestra aplicación web. En una instalación rigurosa, solo recibe pedidos desde nuestro servidor de arista, y no tiene ningún puerto expuesto hacia la red externa. Interactúa directamente con turing, y se apoya en tesla y sherlock para su ejecución y escalamiento horizontal. Provee una interfaz web para el cliente final, la cual tiene como soporte una interfaz REST, para el manejo de recursos.

Turing

Este componente esta dedicado a la gestión de Usuarios, Escenarios y Materiales. Estos son los recursos que expone y mantiene mediante una interfaz REST, apoyándose en una base de datos relacional que se vera en los siguientes puntos. Al

igual que picasso, se apoya en tesla y sherlock para su ejecución y escalamiento horizontal. Dado que solo se encarga de mantener los recursos, la lógica de ejecución de escenarios se la delega a einstein.

Einstein

El tercer pilar de nuestra arquitectura esta dedicado al calculo de las probabilidades de falla de los diferentes escenarios que los usuarios ingresan. Es un componente dedicado únicamente a la ejecución de cálculos, los cuales luego de finalizados, los reporta a turing. Al igual que los dos anteriores, interactúa con tesla y sherlock para su correcta ejecución en un entorno escalable.

Database

Este componente es una instalación de un MariaDB (`mariadb.org`). Optamos por utilizar una instalación única, aunque la tecnología permite crear un cluster con alta disponibilidad.

Diagrama de arquitectura con tecnologías Spring Cloud

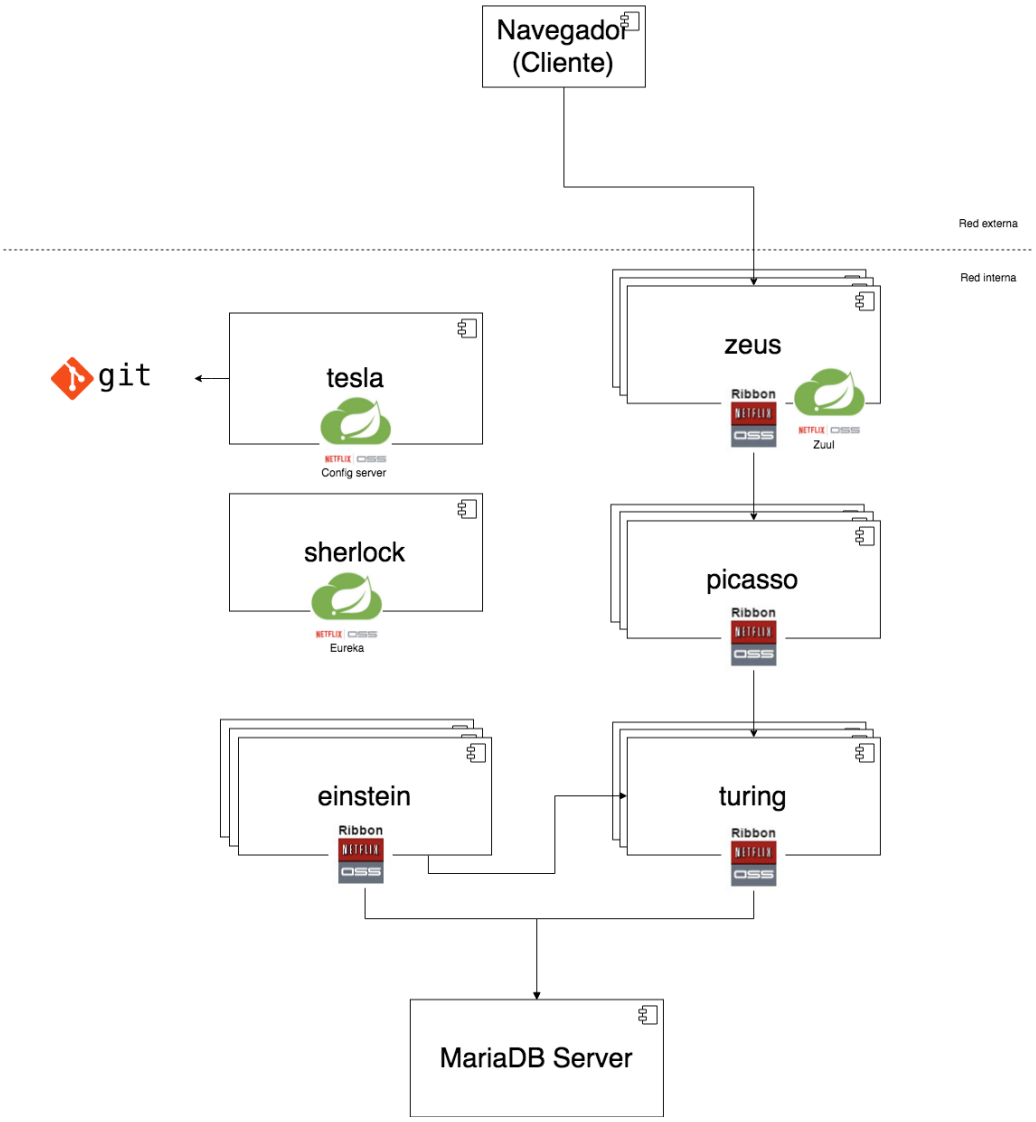

Figura 4.2: Diagrama de micro servicios y las tecnologías de Spring Cloud

4.4. Conclusiones

Con lo visto hasta el momento en los capítulos anteriores y lo expresado en este, se llega al diseño de la Solución Propuesta, se definió la arquitectura que se tendrá, sus características, los componentes que la integraran y con que rol participaran de la solución. En el siguiente capítulo describiremos el relevamiento de tecnologías

realizado, y como se implemento con ellas el diseño propuesto en este capítulo.

Capítulo 5

IMPLEMENTACIÓN

5.1. Introducción

En este capítulo se describirán las principales tecnologías utilizadas, se mencionan algunas de las tecnologías que podrían haberse utilizado, para estas se indicara el motivo por el cual se descarta o se da prioridad a otra. Esto se desprende del correspondiente análisis de herramientas necesarias, teniendo en cuenta los requerimientos funcionales, no funcionales y el diseño propuesto. Para la selección de herramientas se tomo el criterio de que sean actuales, de vanguardia y aquellas que maximicen el beneficio tanto para la implementación como el posterior desempeño. Una vez planteadas las tecnologías a utilizar se pasara a describir la implementación de los principales micro servicios Picasso, Turing y Einstein, los cuales contienen la lógica de negocio e interfaz de usuario de nuestra solución. Comenzaremos por detallar las tecnologías a utilizar agrupadas por área en la sección 5.2, siguiendo por las tecnologías descartadas en la sección 5.3. Pasando a la descripción de la implementación en 5.4 cerrando con las conclusiones en 5.5.

5.2. Tecnologías a utilizar

A continuación detallaremos las tecnologías a utilizar en nuestro proyecto, acompañadas de un fundamento acerca de su elección.

5.2.1. Tecnología Interfaz de Usuario

Dada la solución que vamos a implementar, nuestro cliente reside en un navegador web. Entre las tecnologías a utilizar de cara al cliente destacamos:

Bootstrap

Bootstrap (`getbootstrap.com`) es un entorno de trabajo libre, para el desarrollo de aplicaciones web, de manera rápida y fácil. Incluye plantillas basadas en html y css para tipografías, formularios, botones, tablas, navegación, modales y carruseles entre otros. Fue desarrollado por la empresa Twitter, y lanzado como producto código abierto en Agosto del 2011, estando en un estado de madurez considerable. El punto mas atractivo para nuestra solución, viene dado por la capacidad de ajustar sus componentes a los distintos tamaños de pantalla, lo cual es un requerimiento para nuestra solución.

Ventajas de la utilización de Bootstrap:

1. Fácil de usar
   Cualquier desarrollador con conocimiento básico de HTML y CSS puede comenzar a usar Bootstrap.

2. Características adaptables
   Los elementos de Bootstrap se ajustan dependiendo del tamaño de pantalla del dispositivo en el que están siendo visualizados.

3. Compatibilidad con múltiples navegadores
   Es compatible con la mayoría de los navegadores actuales (Chrome, Firefox, Internet Explorer, Safari, y Opera)

Angular

Angular (`https://angular.io/`) es una plataforma y entorno de trabajo para crear aplicaciones cliente en HTML CSS y Typescript. Desarrollada y mantenida por Google, se basa en el desarrollo de aplicaciones "Single web applications", esto es, aplicaciones web que son cargadas en una única página, y las navegaciones entre las diferentes secciones se da de forma dinámica.

Las principales características que nos llevaron a optar por esta tecnología son:

1. No precisa instalación por parte del cliente.

2. Permite un alto nivel de modularización y reutilización de componentes, por lo que agiliza el desarrollo.

3. Tiene integración con la mayoría de los IDEs de desarrollo, permitiendo así una mayor agilidad a la hora de desarrollar.

4. Brinda soluciones para seguridad entre aplicaciones e internacionalización.

5. Hay una extensa variedad de librerías que extienden la plataforma agregándole funcionalidades mas especificas. Por ejemplo ng2-charts, una librería para el renderizado web de gráficas.

Ng2 charts

Esta librería se integra con Angular, y lo nutre de directivas para el uso de chart.js (`www.chartjs.org`), la cual provee el armado de distintos tipos de gráficas adaptables al tamaño de pantalla del dispositivo.

5.2.2. Tecnología Servidor

Java

Java (`www.oracle.com/technetwork/java/javaee/overview/index.html`) sera la plataforma de desarrollo de software a utilizar. Es uno de los frameworks mas utilizados en el mundo a lo largo de los años. Entre sus puntos fuertes se destacan:

1. Una amplia comunidad de desarrolladores y empresas que la utilizan.
2. Gran cantidad de complementos o herramientas, enfocadas al desarrollo o la administración del código. Un claro ejemplo es Spring Cloud, la cual utilizamos ampliamente en nuestra solución.
3. Gran compatibilidad, el sistema podría ser migrado fácilmente.
4. Permite sistemas en capas, distribuidos y escalables.

Por mas detalle ver Java Tutorial (`docs.oracle.com/javaee/7/tutorial/title.htm`).

Spring Web MVC

Spring Web MVC es un entorno de trabajo que brinda una arquitectura Model-View-Controller, permitiendo flexibilidad a la hora de desarrollar dado el bajo acoplamiento que genera (`projects.spring.io/spring-framework/`). El entorno de trabajo de Spring provee un modelo de programación y configuración integral para aplicaciones basadas en Java. Es soporte de infraestructura a nivel de aplicación, permitiendo al desarrollador enfocarse en la lógica de negocio. Como se vio anteriormente, nos apoyaremos fuertemente en sus herramientas dedicadas al desarrollo de aplicaciones en entornos de nube (SpringCloud).

Spring Boot + Netflix OSS

La arquitectura de micro servicios surge como alternativa y trae una serie de ventajas respecto a las arquitecturas monolíticas. La principal es la posibilidad de crear sistemas basados en aplicaciones independientes de poca envergadura, que se despliegan de forma autónoma, que a su vez se pueden combinar, reutilizar o eliminar, sin que esa modificación afecte al resto de los componentes y sin que se produzca ninguna interrupción. Además la configuración individual nos permite tal flexibilidad, que incluso pueden estar escritos con diferentes lenguajes de programación.

Existen diversas tecnologías en el mercado dedicadas a la construcción de sistemas orientados a este tipo de arquitecturas. Dos de ellas, en las cuales basaremos fuertemente nuestra solución son: Spring Boot (`projects.spring.io/spring-boot/`) y Netflix OSS (`netflix.github.io/`), mas precisamente la implementación SpringCloud-Netflix OSS (`cloud.spring.io/spring-cl`). A esta altura nos centraremos solamente en SpringCloud-Netflix OSS, para un mejor entendimiento de la arquitectura planteada. Spring Boot será explicada en la sección de Implementación.

SpringCloud Netflix OSS

Netflix Open Source Software, es una colección de herramientas que la empresa Netflix, Inc. puso a disposición de la comunidad. Estas herramientas tienen como objetivo principal brindar soluciones para el desarrollo de aplicaciones orientadas a micro servicios, además de dar soporte para big data, persistencia de datos, seguridad de aplicaciones e interfaces de usuario.
Pivotal Software incorporó esta colección de herramientas a su producto Spring Boot, logrando un potente framework para el desarrollo de aplicaciones orientadas a micro servicios implantados en la nube.

A continuación listaremos los elementos principales de este conjunto de herramientas, que dan soporte a nuestra arquitectura.

El servidor de configuraciones

El servidor de configuraciones de Spring (`cloud.spring.io/spring-cloud-config/`), es un micro servicio dedicado al manejo de las configuraciones de los demás micro servicios de la arquitectura. Con su integración con GitHub, brinda entre sus ventajas el no tener configuraciones de aplicación en los códigos fuentes de las mismas. Por omisión este servidor atiende en el puerto 8888.

Eureka Discovery Service

Eureka Discovery Service (`cloud.spring.io/spring-cloud-netflix/multi/multi__service_discovery_eureka_clients.html`). Este componente es a nuestro criterio, el más importante, de cara al soporte a una arquitectura de micro servicios escalable. Este servicio atiende por omisión en el puerto 1111, y esta encargado de llevar un registro de todos los micro servicios que están ejecutando en el ecosistema. Cada micro servicio al iniciar, se debe registrar contra este servidor de descubrimiento, el cual lo incluirá en su listado de servicios activos. El servidor de descubrimiento es responsable de saber que micro servicios están activos y bajo que dirección url son accesibles.
Este componente junto con el siguiente (Ribbon) son los pilares de una arquitectura escalable horizontalmente.

Ribbon

El balanceador de carga, Ribbon (`cloud.spring.io/spring-cloud-netflix/multi/multi_spring-cloud-ribbon.html`), es el componente que junto con Eureka Discovery Service, permiten de manera sencilla, escalabilidad horizontal en una arquitectura. Como balanceador de carga, su función principal es distribuir las peticiones a un micro servicio dado, entre todas sus instancias activas, las cuales conoce de la consulta al servidor de descubrimiento (Eureka). Este balanceador entre sus variadas configuraciones, permite elegir la estrategia de balanceo, la cual por omisión es "Round Robin".

Zuul

Zuul (`cloud.spring.io/spring-cloud-netflix/multi/multi__router_and_filter_zuul.html`) es un servidor de arista encargado de redirigir los pedidos HTTP desde fuera de la arquitectura (por ejemplo desde un navegador web) hacia el interior de la misma (los micro servicios). Se le llama servidor de arista porque en una arquitectura bien configurada es el único con salida a la red externa; los demás servidores no necesitan dicha conectividad dado que el servidor de arista es quién la provee. Para nuestra arquitectura lo consideramos importante, dado que es nuestro balanceador de carga para los pedidos desde el cliente (navegador web) y nos permite escalar horizontalmente nuestra aplicación web (picasso). Zuul es una herramienta con una amplia variedad de configuraciones y restricciones en lo que refiere a la redirección de pedidos HTTP. En nuestro trabajo la usamos como un "pasa manos", entre el cliente y nuestro servidor web (picasso).

Algunas de las herramientas que optamos por no incluir en nuestra solución, dado

que entendemos que no aportan características criticas para nuestra arquitectura son:

Hystrix

En un entorno distribuido, inevitablemente algunas de las dependencias de servicios fallarán. Hystrix (`github.com/Netflix/Hystrix/wiki`) es una biblioteca que ayuda a controlar las interacciones entre estos servicios distribuidos al agregar lógica de tolerancia a fallas. Lo hace aislando los puntos de acceso entre los servicios, deteniendo las fallas en cascada a través de ellos y proporcionando opciones de respaldo, todo lo cual mejora la capacidad de recuperación general de su sistema. Este sistema de tolerancia a fallas entendimos que no era critico para nuestra solución, dado que es pequeña en cantidad de micro servicios y en principio no tendrá alta tasa de trafico. Si en un futuro la arquitectura se viera sobre cargada de trafico, utilizar esta herramienta es una buena opción.

Spring Cloud Sleuth

Spring Cloud Sleuth (`cloud.spring.io/spring-cloud-sleuth/single/spring-cloud-sleuth`
`html`). Una herramienta que permite seguir la traza de una llamada a un servicio, la cual requiere de la interacción de varios micro servicios de la arquitectura. Por el alcance y tamaño de nuestro proyecto, entendemos que podemos prescindir de esta herramienta.

5.2.3. Almacenamiento de datos

Maria DB

Maria DB (`mariadb.org`) es un servidor de base de datos relacional, cuyos datos son accesibles mediante un lenguaje de consulta estructurado (SQL). Este sistema surge de una bifurcación (fork) de MySQL, es desarrollado y mantenido por los desarrolladores que crearon MySQL, a diferencia de MySQL la fundación Maria DB garantiza que sea código abierto y es propuesto como una alternativa mejorada a MySQL. Es utilizado ya por importantes usuarios, es rápido, escalable y robusto. Las versiones mas recientes incluyen características para GIS y JSON, ambas de potencial ayuda para este proyecto y futuras mejoras o extensiones. Dado el enfoque que vemos de parte de los proveedores de este software y las mejoras y correcciones sobre el producto final y la garantía de ser código abierto, en constante ajuste mejora y corrección nos decidimos por ese sistema.

Hibernate

Hibernate (`hibernate.org/orm/`) es una herramienta Mapeo objeto-relacional (ORM), con la cual se simplificara el mapeo entre el modelo de objetos del sistema y los atributos de la base de datos relacional que usaremos. De Hibernate podríamos aprovechar también sus cargas parciales (Lazy), sus estrategias de búsquedas y bloqueos, todo esto con el fin de obtener un buen desempeño del sistema en relación a la base de datos.

5.2.4. Sistema operativo

Ubuntu 16.04.2 LTS

El sistema operativo recomendado para la ejecución del sistema es Ubuntu 16.04.2 LTS, (`wiki.ubuntu.com/XenialXerus/ReleaseNotes`) a esta decisión llegamos dadas las siguientes características:

1. Instalación a medida
   Al disponer de la opción de instalar del sistema únicamente lo que necesitaremos del sistema, lo hace más liviano pudiendo así aprovechar más el hardware disponible.

2. Costos
   Al estar basados en su totalidad o gran mayoría sobre licencias código abierto, se reducen costos o los mismo son nulos.

3. Comunidad de usuarios
   Al disponer de una gran comunidad de usuarios es bastante sencillo encontrar aportes ante los posibles sucesos que nos enfrentemos.

4. Tiempo de soporte de la versión
   Un punto a considerar el el tiempo de respaldo de la versión, esto es el tiempo a que se compromete el proveedor a brindar soporte y mantenimiento sobre esta versión, en este caso podrían ser hasta abril de 2021.

Además de las características mencionadas, destaca su estabilidad.

5.2.5. Tecnologías de control de versiones y puesta en producción

Poner en producción un sistema o una nueva versión del mismo, llevar un control del versionado, tanto de código fuente como de los ejecutables, son tareas del día a día en el desarrollo y mantenimiento de aplicaciones de mediano gran porte. Para esta tarea nos basamos en las siguientes tecnologías:

Git

Git es una herramienta de control de versionado de código utilizada ampliamente en el mercado. Hay varios proveedores de servicios git, entre ellos GitHub, el cual elegimos para nuestro trabajo. Con esta herramienta se puede realizar un complejo manejo de ramas y versionado de código. En nuestro proyecto no explotamos esta característica, dado que somos solo dos desarrolladores. Luego ya enfocados en la entrega del producto a los tutores, migramos nuestros repositorios a la instalación de GitLab provista por la Facultad de Ingeniería: **gitlab.fing.edu.uy**

Maven

Es una herramienta de gestión y construcción de proyectos, manejo de dependencias para soluciones basadas en Java. Entre sus funcionalidades destacadas están, la gestión de dependencias y versionado de las mismas, compilación y generación de ejecutables y puesta en producción de los mismos. Esta herramienta es extensible por medio de plugins, que agregan funcionalidades extra a las que trae el núcleo de Maven.

Docker

Docker es una plataforma abierta para automatizar el despliegue de aplicaciones en diferentes entornos, gestionando las dependencias de las mismas y minimizando los requerimientos de la máquina donde se va a realizar la instalación. Se maneja por medio de repositorios con versionado, donde se alojan los llamados "contenedores", estos contenedores pueden alojar desde un archivo de aplicación ejecutable hasta un sistema operativo. En nuestro caso, nuestros contenedores llevan los archivos ejecutables del micro servicio que corresponda, y la dependencia a la librería Java. Con un simple comando esta herramienta descarga el repositorio y sus dependencias y pone a ejecutar nuestra aplicación. Una de las grandes ventajas es que Docker da soporte para los principales sistemas operativos, por lo que nuestra solución podría correr en entornos Windows como Linux sin ningún problema.

5.3. Tecnologías Descartadas

A continuación nombraremos algunas tecnologías que se descartaron y un breve detalle del motivo.

1. MySQL
   Es un servidor de base de datos, relacional. El mismo se descarta por ser propiedad de un privado y si bien permite licenciamiento GPL, le vemos mayor

potencial a Maria DB el cual permite una configuración de alta disponibilidad, característica que un MySql no brinda.

2. Windows Server
   Servidores Windows no sólo se descartan por los costos de sus licencias, si no que también por el mantenimiento que requieren, su disponibilidad, su capacidad de manejar procesos o su velocidad de respuesta ante incidentes de seguridad.

3. Otras Servidores Linux
   Se descartan principalmente por los costos relacionados para acceder al soporte o consultoría.

4. .Net Framework
   A pesar de que tenemos varios años de experiencia en esta tecnología (desarrollando aplicaciones de escritorio), entendemos que nos deja ligados a un sistema operativo, además de que la experiencia en el desarrollo de una herramienta de este tipo la tenemos en Java con Spring.

5.4. La implementación

En esta sección describiremos las implementaciones de los micro servicios Picasso (interfaz de usuario), Turing (manejo de recursos), Einstein (lógica de cálculos matemáticos).

5.4.1. Los puntos comunes

Los tres son aplicaciones Java, desde las cuales se generan archivos ejecutables ".jar". Gracias a la tecnología SpringBoot, dicho ejecutable levantara un servidor, en un puerto aleatorio (esto es decisión nuestra, para que sea posible levantar varias instancias de un mismo servicio en una misma maquina física). Aunque difieren en algunos de sus paquetes internos, tienen una arquitectura base común, la cual se muestra a continuación:

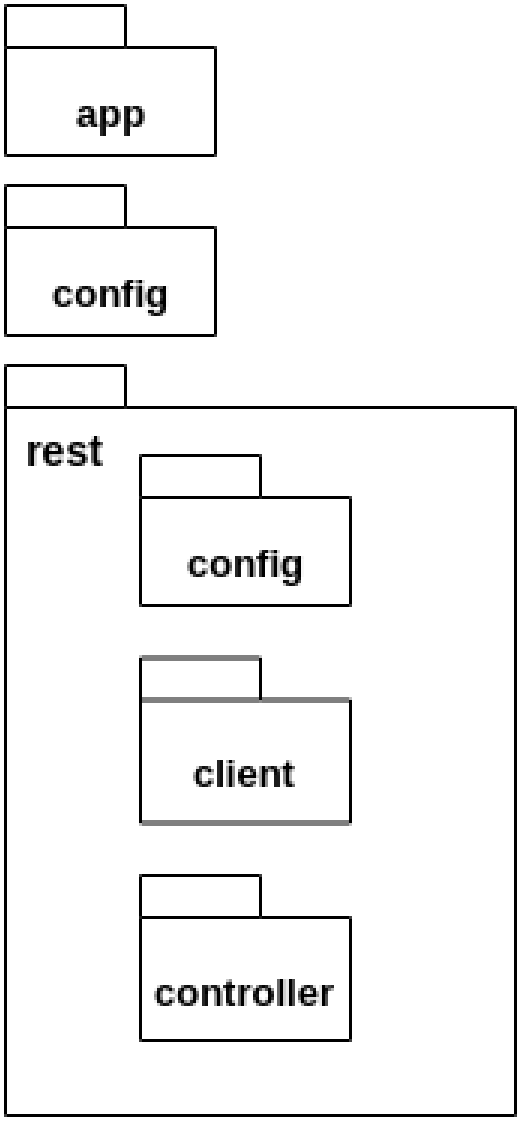

Figura 5.1: Arquitectura de paquetes comunes a los tres micro servicios principales

- App: En este paquete se encuentra la clase principal del proyecto, con sus correspondientes configuraciones según corresponda.
- Config: En este paquete se encuentran las clases dedicadas a configurar el contexto de Spring, si este es requerido.
- Rest: En este paquete se encuentran las clases dedicadas a exponer servicios REST (controller), a consumir servicios REST (client) y sus configuraciones en el paquete config.

Dado que Picasso esta orientado a brindar una plataforma web, su arquitectura interna difiere, con respecto a las de Turing y Einstein, las cuales coinciden en la mayoría de los puntos dados que son micro servicios que exponen interfaces REST y tienen acceso a datos.

5.4.2. Turing y Einstein

A los componentes comunes que se mostraron anteriormente, estos micro servicios suman los referentes a acceso a datos y capa de servicios.

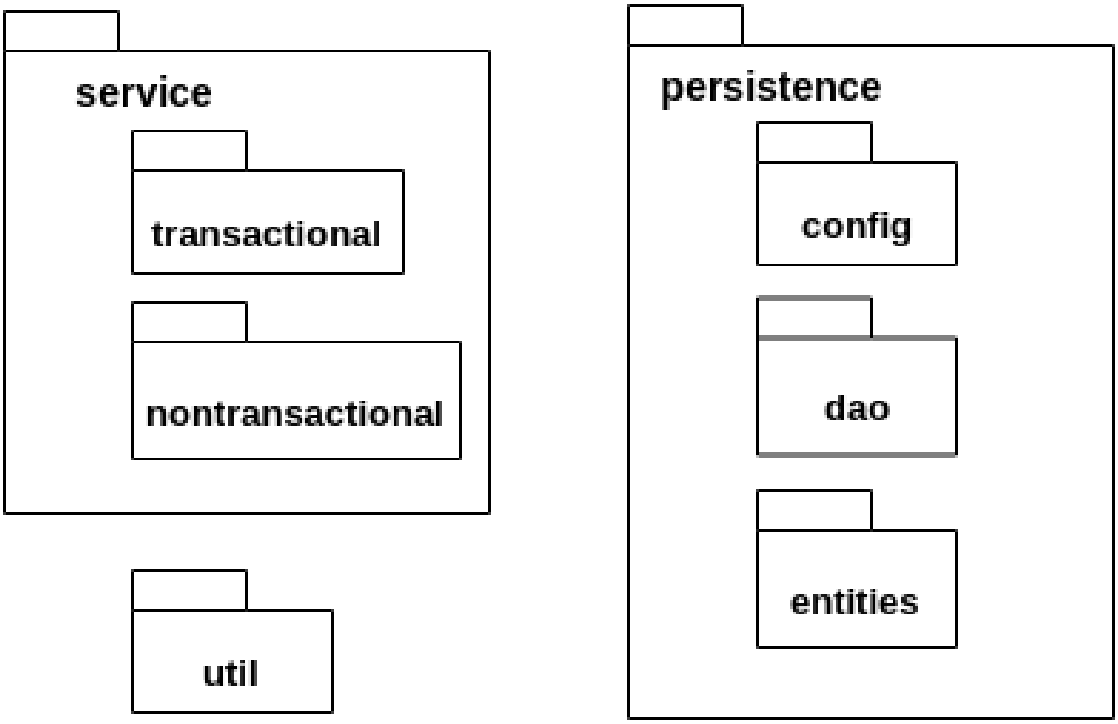

Figura 5.2: Arquitectura de paquetes específicos de Turing y Einstein

Service

En estos paquetes radica la lógica de negocio, ya sea en Turing la de manejo de recursos y en Einstein los cálculos matemáticos. Dado que "contra la base de datos"se maneja transaccionalidad, dividimos esta lógica transaccional (transactional) y no transaccional (nontransactional)

Persistence

Este paquete contiene lo referente a acceso a base de datos, desde la configuración de la misma (config), las entidades (entities), y los servicios que acceden a los datos directamente (dao).

Util

En este paquete se alojan clases cuyo cometido es ser de utilidad para la implementación de la lógica de negocio, y que por motivos de reutilización se ponen en un paquete aparte.

5.4.3. Picasso

Picasso es el que más difiere en lo que respecto a arquitectura interna, dado que además de albergar una página web Angular, sus servicios están orientados a servir a un cliente web.

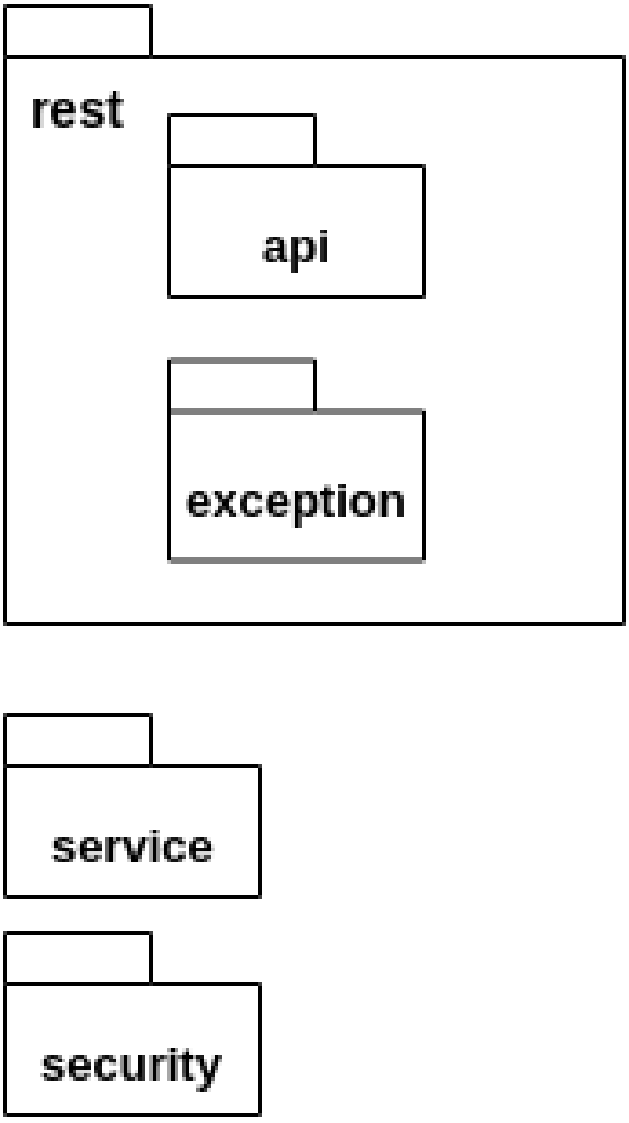

Figura 5.3: Arquitectura de paquetes específicos de Picasso

Rest

Dada la configuración común de paquetes especificada mas arriba, Picasso le agrega al paquete Rest, un componente de api, donde radican los objetos que se exponen via REST, y exception donde se declaran las diferentes excepciones a lanzar en función de los errores que puedan llegar a ocurrir. Este ultimo punto es critico, dado que la forma de comunicarse ante errores con el cliente web, es por medio de excepciones, que gracias al framework SpringMVC, son transformadas en respuestas http con su correspondiente código (véase la clase ForbiddenException,

en el proyecto Picasso).

Service

Al igual que en Turing y Einstein, aquí radica la lógica de negocio, y dado que Picasso no tiene acceso a base de datos, no hay diferencia de transaccionalidad.

Security

Dado que estamos en el contexto de una aplicación web, mantener una capa de seguridad es crucial ante ataques maliciosos. A esto se le suman los requerimientos de autenticación y autorización, cuya configuración también se aloja en este paquete.

5.4.4. La aplicación web

Hasta ahora, hemos visto un esquema de implementación de los tres micro servicios principales, todos ellos aplicaciones java, con una arquitectura interna común y partes mas especificas dependiendo de sus responsabilidades. Poco se ha dicho sobre la aplicación web, y donde radica, así como la forma de exponerse en el servidor.

La integración Angular - Java

Las páginas Angular no son más que un archivo html acompañado de varios archivos javascript y css. Estos archivos al ser expuestos en un puerto de un servidor, pueden ser accedidos mediante cualquier navegador, en forma de página web. Uno de nuestros objetivos, era lograr entregables fáciles de instalar y poner en producción, por lo que apuntamos a que la página entera esté contenida en el archivo ejecutable de Picasso, y cuando éste corriera, ya dejara operativa la página web asi como los servicios REST en los que se apoya.
Esto lo logramos gracias a la funcionalidad de SpringBoot de exponer recursos estáticos. El servidor que se levanta al ejecutar la aplicación servirá todos los archivos que estén en una carpeta nombrada "webapp". Por ende nuestra aplicación Angular reside allí. Entendemos que corresponde aclarar que este archivo html y sus correspondientes javascript y css, que conforman la página angular, son archivos compilados, y no son el código fuente de la misma. En el siguiente diagrama se muestra donde reside el código fuente de la página y el compilado, en función de la estructura principal del proyecto.

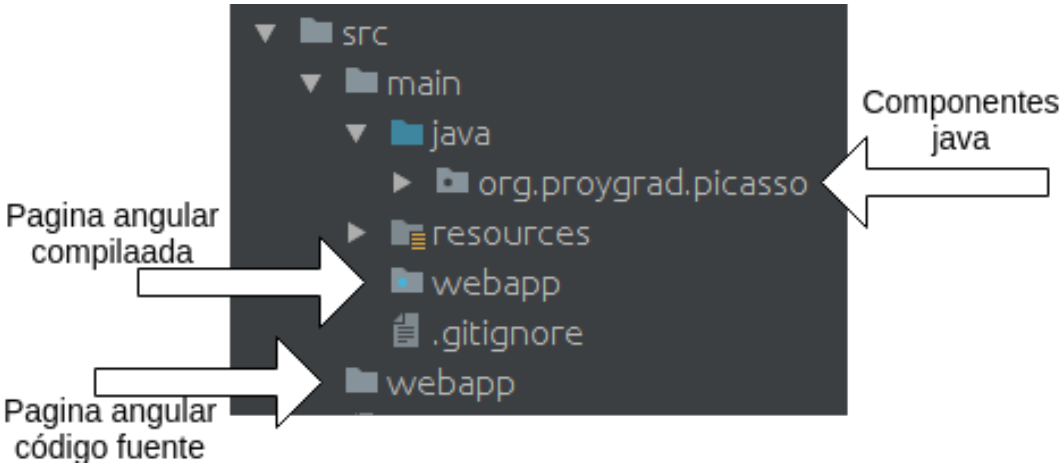

Figura 5.4: Arquitectura de paquetes específicos de Picasso

La estructura en el código fuente angular

Dado que no es objetivo de este trabajo explicar como es la estructura base de una página angular, ni los elementos que la componen, vamos a ir directamente a la arquitectura de nuestros componentes, los que hacen a la página web de nuestra solución. Esta estructura se encuentra dentro de src/webapp/src/app. Ilustramos la misma con una imagen para luego entrar en detalle.

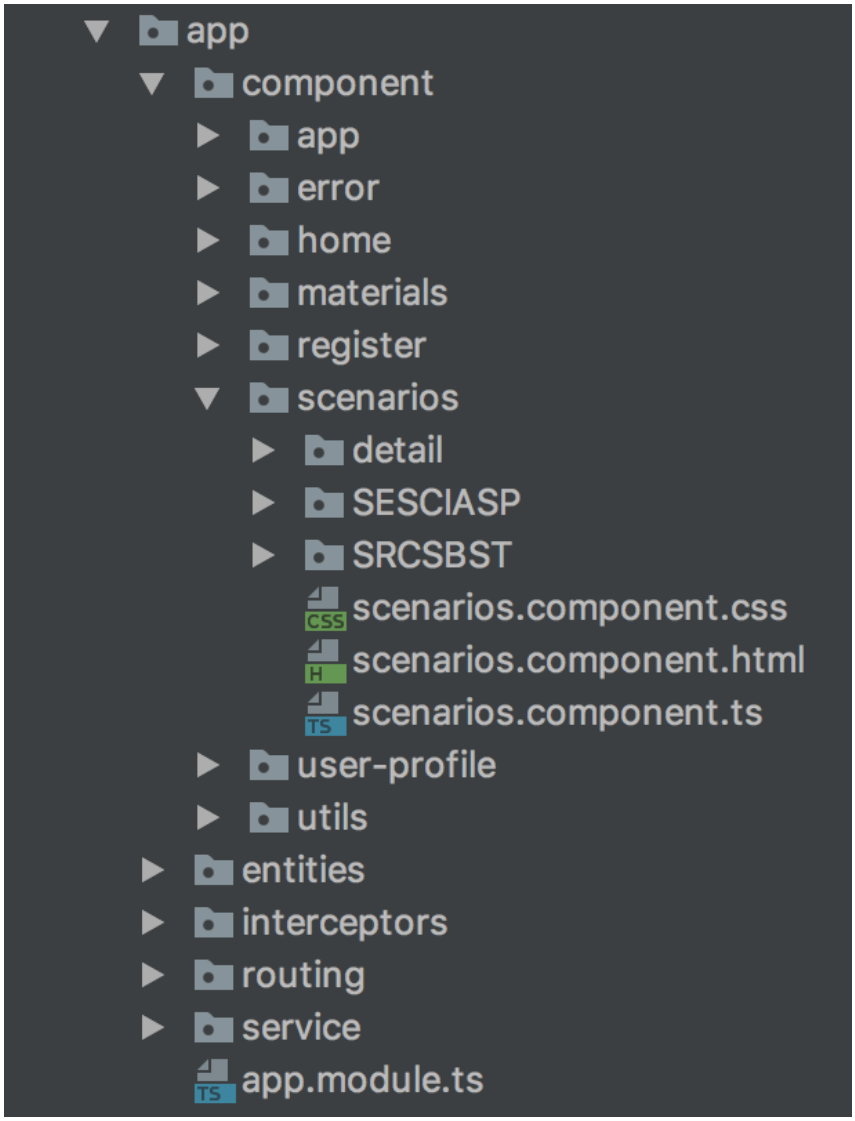

Figura 5.5: Arquitectura de componentes Angular

- Component.App
  Componente principal de la aplicación, el cual contendrá a los componentes específicos de cada sección.
- Component.Error
  Componentes para mostrar en caso de que ocurran errores inesperados.
- Component.Home
  Componente del panel principal de la aplicación

- Component.Materials
  Componente de la sección de gestión de materiales
- Component.Register
  Componente para el registro de un nuevo usuario
- Component.Scenarios
  Componente para seleccionar el escenario a calcular.
- Component.Scenarios.Detail
  Componente donde se muestra el resultado de un calculo de un escenario.
- Component.Scenarios.SESCIASP
  Componente donde se carga la configuración para un escenario de tipo: Confiabilidad estructural de tuberías fisuradas basada en el diagrama de evaluación de falla.
- Component.Scenarios.SRCSBST
  Componente donde se carga la configuración para un escenario de tipo: Confiabilidad estructural de barra de sección circular sometida a tracción.
- Component.User-Profile
  Componente de visualización y edición del perfil de usuario.
- Component.utils
  Este paquete contiene clases utilitarias que se utilizan a lo largo de toda la aplicación.
- Entities
  Contiene las clases entidad utilizadas en la aplicación.
- Interceptors
  Contiene interceptores angular de pedidos HTTP.
- Routing
  Contiene la configuración de las rutas de la aplicación angular.
- Service
  Contiene las clases que implementan las llamadas a los servicios REST.

5.5. Conclusiones

En este capítulo se detallaron las principales tecnologías utilizadas en la implementación de la Solución Propuesta, se nombraron tecnologías que podrían haber sido usadas, y por que motivo se descartaron. Luego de esto se paso a describir la implementación y como se usaron las principales tecnologías en los diferentes componentes. Con todo esto se deja claro como y donde se usaron las tecnologías que le dan un valor agregado a la Solución Propuesta. En el próximo capítulo sobre Implantación se describirá como esta implementación fue desarrollada y su puesta en producción.

Capítulo 6

IMPLANTACIÓN

6.1. Introducción

En este capítulo detallaremos la implantación de nuestra arquitectura, tanto en la etapa de desarrollo como en la versión productiva. Se darán pautas para la realización de una nueva implantación, esto dado el objetivo e intención de que este proyecto pueda continuar ampliándose. En la sección 6.2 se explicara lo referente a la implantación durante el desarrollo, se explicaran los servicios de AWS en la sección 6.2.1 y la implementación correspondiente en 6.2.2. Se decidió entregar parte de esta sección en un anexo, en el cual incluimos los detalles de la implantación productiva y un paso a paso para una nueva implantación. Por ultimo se brindan las conclusiones del capítulo en 6.4.

6.2. La implantación durante el desarrollo

Durante el desarrollo del proyecto, trabajamos con una instalación de nuestra arquitectura en la nube de Amazon Web Services (AWS) (`aws.amazon.com`). La decisión fue tomada en conjunto con los profesores, luego de consultarles opciones que nos pudiera brindar la facultad, tanto para la etapa de desarrollo como para una instalación definitiva del producto. La facultad tiene convenios académicos con Amazon y Microsoft. Con estos convenios no podríamos dejar nuestra solución instalada y productiva en sus servidores, porque excederían el ámbito académico. Se decidió por AWS para la instalación en etapas de desarrollo, dado que ofrece una cuenta estudiantil (AWS Educate `aws.amazon.com/es/education/awseducate`), con saldo disponible por un año para hacer uso de los servicios de la plataforma. Luego nos encontramos con que no estaban disponibles la totalidad de estos servicios, igualmente fue suficiente para los propósitos de nuestro trabajo, sin tener complicaciones.

6.2.1. Servicios de AWS

Antes de explicar la implantación de nuestra arquitectura en AWS, introduciremos los diferentes servicios utilizados de esta plataforma.

AWS EC2

Amazon Elastic Compute Cloud (`docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/concepts.html`) provee servicios para la creación y gestión de maquinas virtuales (llamadas instancias).
Para la creación de nuestras instancias optamos por utilizar las plantillas predefinidas que este servicio provee, mas específicamente una llamada t2.micro, las cuales constan de un procesador virtual y 1GB de memoria RAM. También optamos por utilizar la preinstalación de un sistema operativo. EC2 provee diferentes imágenes de sistema operativo, entre ellas Ubuntu Server 16.04 LTS.

A eso se le suman variadas opciones de configuración, como ser, creación de redes, reglas de entrada y de salida de paquetes de red, configuración de elasticidad de las instancias, grupos de seguridad, etc.

En nuestra implantación de desarrollo y solo con fines académicos, realizamos la configuración de una red segura, de modo que solo tengan acceso a internet los micro servicios que así lo requieran. Para eso, se crearon dos grupos de seguridad. El primero permite a la instancia cualquier tipo de trafico de red hacia la internet. Este grupo de seguridad fue aplicado en el servidor de configuraciones (Tesla), servidor de descubrimiento (Sherlock) y servidor de arista (Zeus). El segundo grupo de seguridad no permite trafico de entrada desde la internet, salvo para conexiones SSH, las cuales utilizamos para realizar las instalaciones. Este segundo grupo de seguridad fue aplicado a los micro servicios de lógica de negocio: Picasso, Turing y Einstein.

AWS RDS

Amazon Relational Database Service (`https://docs.aws.amazon.com/es_es/AmazonRDS/latest/UserGuide/Welcome.html`) es un servicio web que facilita las tareas de configuración, uso y escalado de bases de datos relacionales en la nube. Permite crear instancias y clusters de bases de datos de manera sencilla. Los tipos de bases de datos que soporta actualmente son:

- Amazon Aurora

- Mysql
- PostgreSql
- Microsoft SQL Server
- Oracle
- MariaDB

En nuestro caso optamos por una instancia de MariaDB

AWS S3

Amazon Simple Storage Service (`docs.aws.amazon.com/es_es/AmazonS3/latest/dev/Introduction.html`), provee servicios para alojar contenido estático y acceder al mismo de forma rápida desde redes internas y externas. Utilizamos esta tecnología para alojar las imágenes a ser utilizadas en nuestra pagina web.

6.2.2. Implantación en AWS

A continuación una imagen descriptiva de nuestra arquitectura y su relación con la tecnología AWS.

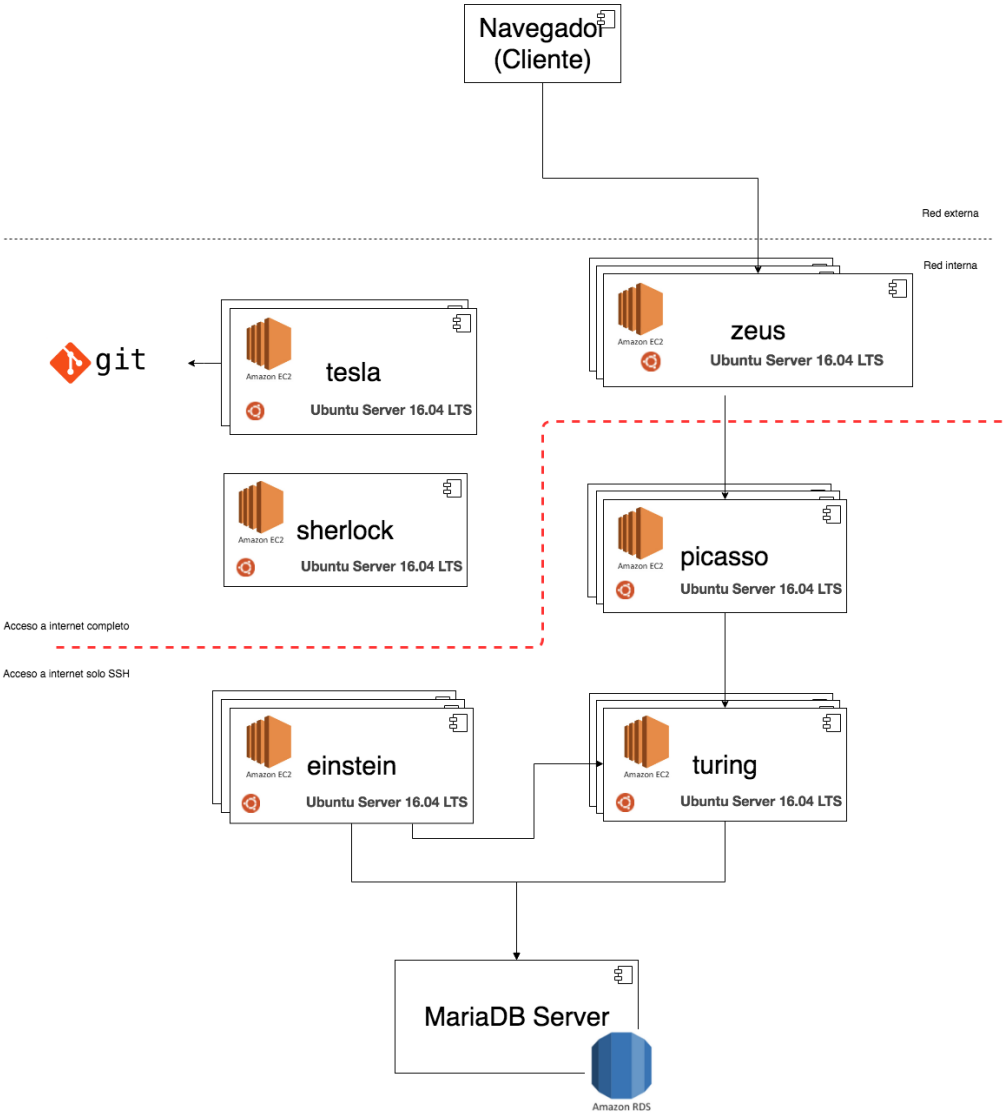

Figura 6.1: Diagrama de micro servicios y las tecnologías de AWS

En la imagen se muestra como aplicamos las tecnologías AWS a nuestra arquitectura. Se observa que todas las instancias de los micro servicios fueron instaladas en una instancia de EC2, la cual fue configurada con una imagen de Ubuntu Server 16.04. También se muestra que nuestro servidor de base de datos se basa en Amazon RDS. La linea punteada, hace referencia a la configuración de la red virtual, diferenciando dos zonas; la primera zona, integrada por tesla sherlock y zeus, tiene acceso de entrada y salida a internet; la segunda zona, integrada por Picasso, Turing, Einstein y la base de datos, tiene acceso restringido solo a la red interna. Algo que no se muestra en la imagen pero que también fue utilizado del conjunto de tec-

nologías AWS, es el almacenamiento de contenido estático de nuestra plataforma, como ser imágenes, para el cual se utilizo Amazon S3.

También a modo académico, llegamos a tener una configuración con múltiples instancias de los micro servicios:

- Zeus: 1 instancia
- Sherlock: 1 instancia
- Tesla: 1 instancia
- Picasso: 3 instancias
- Turing: 3 instancias
- Einstein: 3 instancias

De esta manera se pudieron apreciar las ventajas de tener un servidor de descubrimiento y un balanceador de carga que distribuía el trabajo entre las diferentes instancias múltiples de cada micro servicio.

6.3. La implantación productiva

La implantación productiva y el instructivo para una nueva implantación se puede ver en el documento denominado **Anexo de implantación**, el cual se puede encontrar en la carpeta de entrega.

6.4. Conclusiones

En este capítulo se presentan las instalaciones de desarrollo y productivas de nuestro trabajo. Se introducen las tecnologías AWS utilizadas para la instalación de desarrollo. Esta sección referencia al documento **Anexo de implantación**, en el cual se detalla la implantación productiva y los pasos para una nueva implantación. Entendemos que se logro un producto con un proceso de implantación simple, gracias a las tecnologías seleccionadas para ello.
Luego de implantado el producto, resta una introducción al mismo y un pasaje por sus características principales. El próximo capítulo esta dedicado al producto final, tanto la plataforma instalada en modo productivo, como los entregables de software.

Capítulo 7

PRODUCTO FINAL

7.1. Introducción

En este capítulo se describirán las características del producto final, para esto se decidió hacerlo de forma separada, por un lado las que refieren a los elementos entregados y por otro las características funcionales ya sea generales o las correspondientes al uso del sistema. Las características referentes a los elementos entregados se detallarán en 7.2, para luego seguir con la sección de las características funcionales en 7.3, teniendo dos subsecciones una para las características generales 7.3.1 y otra para las características en el uso 7.3.2. En 7.4 se detalla la validación algorítmica de los escenarios resueltos. Finalizamos con la sección de conclusiones 7.5.

7.2. Características de los elementos entregados

Respecto a los elementos entregados, estos refieren tanto a la documentación del proyecto, a los elementos que hacen tanto al entorno del sistema como al sistema en si. A continuación listaremos estos elementos junto a su descripción.

- Documentación: Este elemento contiene el informe final correspondiente al proyecto, el mismo se entregara tanto su versión impresa como su versión digital.
- Documentación-Anexo: Este elemento contiene los pasos para realizar una nueva implantación y los realizados en la implantación productiva correspondiente al proyecto, el mismo se entregara tanto su versión impresa como su versión digital.

- Documentación-Validación: Este elemento contiene la validación realizada para el escenario de calculo de confiabilidad estructural de tuberías fisuradas basada en el diagrama de evaluación de falla (3.5), el mismo se entregara en su versión digital.
- Script BD: El mismo contiene los comandos necesarios para configurar la Base de Datos utilizada por el sistema, en el mismo se definen los esquemas, tablas y usuarios necesarios para el uso normal del sistema.
- Código fuente: De cada micro-servicio del sistema se entregara el código fuente, por mas detalle sobre que elementos están comprendidos consultar el capítulo 4, sección 4.3.1. Cada micro-servicio tiene su correspondiente proyecto en el repositorio GITLab brindado por la facultad, se les brindaran permisos de administrador a los tutores del proyecto.

Cada uno de los elementos entregados fue diseñado con el fin de poder ser reutilizado y ampliado en sucesivos trabajos o ser de apoyo para tal fin.

7.3. Características funcionales

En esta sección detallaremos las principales características del producto en lo funcional, enfocándonos en la experiencia brindada al usuario final, dejaremos de lado en esta sección aquellas características ya descriptas o especificadas previamente, como ser las que refieren a la escalabilidad, performance o mas precisamente aquellas referidas a cuestiones técnicas. Teniendo en cuenta esto, dividiremos esta sección en dos partes, características generales del producto y características de uso del sistema.

7.3.1. Características generales del producto

En esta subsección detallaremos las principales características del producto en lo funcional, enfocándonos en la experiencia dada al usuario.

- El usuario puede utilizar la aplicación en una amplia variedad de navegadores actuales, con esto podrá disponer del sistema ya sea desde su celular, tablet o cualquier dispositivo capaz de ejecutar un navegador web. Para poder brindar esto el sistema es capaz de adaptarse a la resolución en la cual se esta visualizando, dependiendo de esto el sistema establece como mostrar sus diversas opciones, con el fin de garantizar la usabilidad del mismo.
- El poder de resolución de un escenario no dependerá del hardware que disponga el cliente, pues a su hardware solo se le requerirá el ingreso de datos

de entrada y recepción de datos de salida, lo realmente costoso computacionalmente hablando sera resuelto del lado del servidor.

- El usuario no deberá preocuparse por realizar conversiones de unidades, dado que los datos de entrada serán solicitados en el sistema de unidades de su preferencia, y si a pesar que estas no sean coincidentes con las utilizadas por el escenario el sistema se encargara de convertirlas para su correcta resolución.

7.3.2. Características de uso del sistema

A continuación listaremos las principales características de uso del sistema:

- Inicio de sesión: En esta pantalla se le permite al usuario iniciar sesión, en caso de no estar registrado el usuario podrá registrarse mediante la opción de "Sign Up". Una vez que inicie sesión o se haya registrado, podrá utilizar el sistema teniendo a su disposición de un "Menú Principal". El inicio de sesión del sistema es de la forma tradicional, sin mayores opciones que las básicas correspondientes a la interacción que se desea tener.

- Registro o "Sign Up": En esta pantalla se le permite al usuario registrarse, para esto al usuario se le solicitara: correo electrónico, contraseña, nombre, apellido, organización y preferencias de sistema de unidades como también de idioma a utilizar.

- Menú Principal: Es el principal menú del sistema, el mismo siempre estará accesible y disponible en la parte superior de la pantalla, en el se ofrecerán las funcionalidades a través de secciones concretas y claras (Escenarios, Materiales y Perfil), además se brinda la facilidad "Escenario Rápido" esta opción dirigirá al usuario de forma directa a su escenario preferido. Así como también la funcionalidad de cierre de sesión.

- Materiales: En esta sección, el usuario podrá ver un listado de los materiales que se ingresaron con sus propiedades, también podrá agregar materiales así como modificar o eliminar los ya incorporados. A la hora del ingreso de materiales se les definirá un nombre, meramente descriptivo y se le permitirá al usuario ir agregando propiedades, deberá ingresar al menos una para guardar el nuevo material. Si se desea modificar algún material solo se debe seleccionarlo, y como en el caso del ingreso se listaran las propiedades que se pueden ingresar nuevas, además de las propiedades ya cargadas permitiendo que estas puedan ser modificadas.

- Escenarios: En esta sección del sistema, el usuario podrá seleccionar un escenario a calcular, para esto lo seleccionara desde el listado de escenarios

que dispone el sistema. Una vez seleccionado que escenario se desea resolver basta con presionar el botón proceder para continuar con el ingreso de los valores de entrada del escenario, los mismos estarán agrupados según el tipo de dato que fueran, se identifican en la imagen del escenario, con el fin de facilitar el ingreso de los mismos, los que correspondan a materiales podrán cargarse de la biblioteca de materiales o en ese mismo momento, a medida que estos datos se ingresen serán validados individualmente, indicándole al usuario si los mismos son validos. Una vez ingresados todos los valores de entrada el usuario podrá solicitar la resolución del escenario mediante el botón comenzar escenario, aquí en este punto se realiza un nueva validación de los valores ingresados, pero esta vez respecto a las relaciones entre ellos, esto con el fin de evitar escenarios no validos que no podrían o no tendrían sentido resolver, si esto ocurriera se le indicarían las advertencias pertinentes al usuario, en caso contrario se procedería a resolver el escenario. Mientras el sistema resuelve el escenario se le mostrara al usuario una pantalla indicando que se esta en proceso, pudiendo ver los parámetros de entrada ingresados. Una vez resuelto el escenario, se le indicara al usuario el resultado del mismo.

- Perfil: En esta sección de Perfil, se muestran los datos ingresados por el usuario cuando se registro en el sistema o los que resulten de modificaciones previas. En esta sección el usuario podrá actualizar sus datos personales así como sus preferencias o su escenario rápido.

7.4. Validaciones

En esta sección se detallan las validaciones realizadas a los cálculos realizados por el sistema. En primera instancia se valido el escenario de confiabilidad estructural de barra de sección circular sometida a tracción (3.6) y por ultimo se realizaron validaciones al escenario de calculo de confiabilidad estructural de tuberías fisuradas basada en el diagrama de evaluación de falla (3.5), haciendo foco en la resolución de los cálculos y en los resultados intermedios.

7.4.1. Primera validación

Para esta validación se realiza en primer lugar, el calculo de confiabilidad estructural de barra de sección circular sometida a tracción, con los siguientes valores de entrada:

- Diámetro de la barra, $d = 10\,(mm)$ $(Determinista)$
- Carga a la que es sometida la barra, $\vec{F} = 19625\,(N)$ $(Determinista)$

- Tensión de fluencia del material, $T_y = 250\,(MPa)$ $(Determinista)$
- **Resultado esperado**, $Probabilidad\,de\,falla\; = 0$
  Observando el escenario planteado, las tensiones son iguales ($T = T_y$), por lo que dadas las formulas de falla en 3.6, se tiene un escenario seguro.

La probabilidad de falla obtenida con el sistema fue de 0.
El sistema siempre retorna una probabilidad de falla luego de ejecutar el método de Monte Carlo y en este caso, al ser todos los parámetros deterministas, no importa las iteraciones el resultado sera siempre el mismo.

En segundo lugar se realiza un calculo similar pero con las variables de tipo probabilísticas:

- Diámetro de la barra, $d = 10\,(mm)$ $(Determinista)$
- Carga a la que es sometida la barra, $\vec{F} = Normal[19625, 588{,}75](N)$
- Tensión de fluencia del material, $T_y = Normal[250, 7{,}5](MPa)$
- Precisión, 10^6

- **Resultado esperado**, $Probabilidad\,de\,falla\; = 0{,}5$
  Esto se deduce de que para el escenario planteado, en un caso probabilístico, se tiene que $T \simeq T_y$ (ver 3.6).

La probabilidad de falla obtenida fue de 0,5, con una varianza de $0{,}25 \times 10^{-5}$. Se observa que el resultado esperado se encuentra en el centro del intervalo de confianza dado por el resultado obtenido, por lo que se da por aprobada la validación.

Para el caso de las variables con distribución normal además se realizo el Test de Shapiro–Wilk, confirmando la correctitud de los números aleatorios generados.

7.4.2. Segunda validación

Para la validación de los cálculos para el escenario 3.5 se toman como entrada los valores y los resultados expresados en [17]:

- Espesor de la pared, $t = Normal[25, 5](mm)$
- Radio interno, $Ri = Normal[1500, 150](mm)$
- Largo de la grieta, $2c = Normal[80, 8](mm)$

- Profundidad de la grieta, $a = Normal[5,0,5](mm)$
- Tensión de fluencia, $LogNormal[248,24{,}8](MPa)$
- Tenacidad a la fractura, $LogNormal[329{,}7,32{,}97](mpa * m^{0{,}55})$
- Presión operativa, $P = 3{,}4(MPa)(Determinista)$
- Colapso plástico, $L_r^{max} = 1$
  se ingresa en primera instancia el valor 1 (indicado por los tutores dado que en [17] no se especifica). Con el fin de brindar una mayor precisión a la validación se solicito por parte de los tutores del proyecto a los autores de [17] el valor utilizado para L_r^{max}, pero no se obtuvo respuesta.
- Precisión, 10^6

- **Resultado esperado**, *Probabilidad de falla entre* 0,0002 *y* 0,0012

La probabilidad de falla obtenida fue de 0,5, con una varianza de $0{,}25 \times 10^{-5}$.

Validación de cálculos intermedios

Se tomo registro de cada iteración del Método de Monte Carlo, con las variables de entrada y los resultados intermedios. Estos fueron cotejados contra una hoja de calculo en la cual se implementaron las formulas referentes a la resolución del escenario. No se observan mayores diferencias en los resultados intermedios, mas que las generadas por redondeos de un orden despreciable.
La hoja de calculo se entrega junto con este documento.

Validación del resultado final

Respecto a la probabilidad de falla se observa que difiere de lo esperado, por lo cual se pasa a analizar el porque de este fenómeno.
Se llega a la conclusión que por la influencia del valor de L_r^{max} en el resultado de $f(L_r)$, y por ende en el FAD, el cual determina si una interacción corresponde a un escenario seguro o no, y como se tomo el valor 1 en base a la experiencia, optamos por experimentar con valores superiores. Obteniendo las siguientes probabilidades:

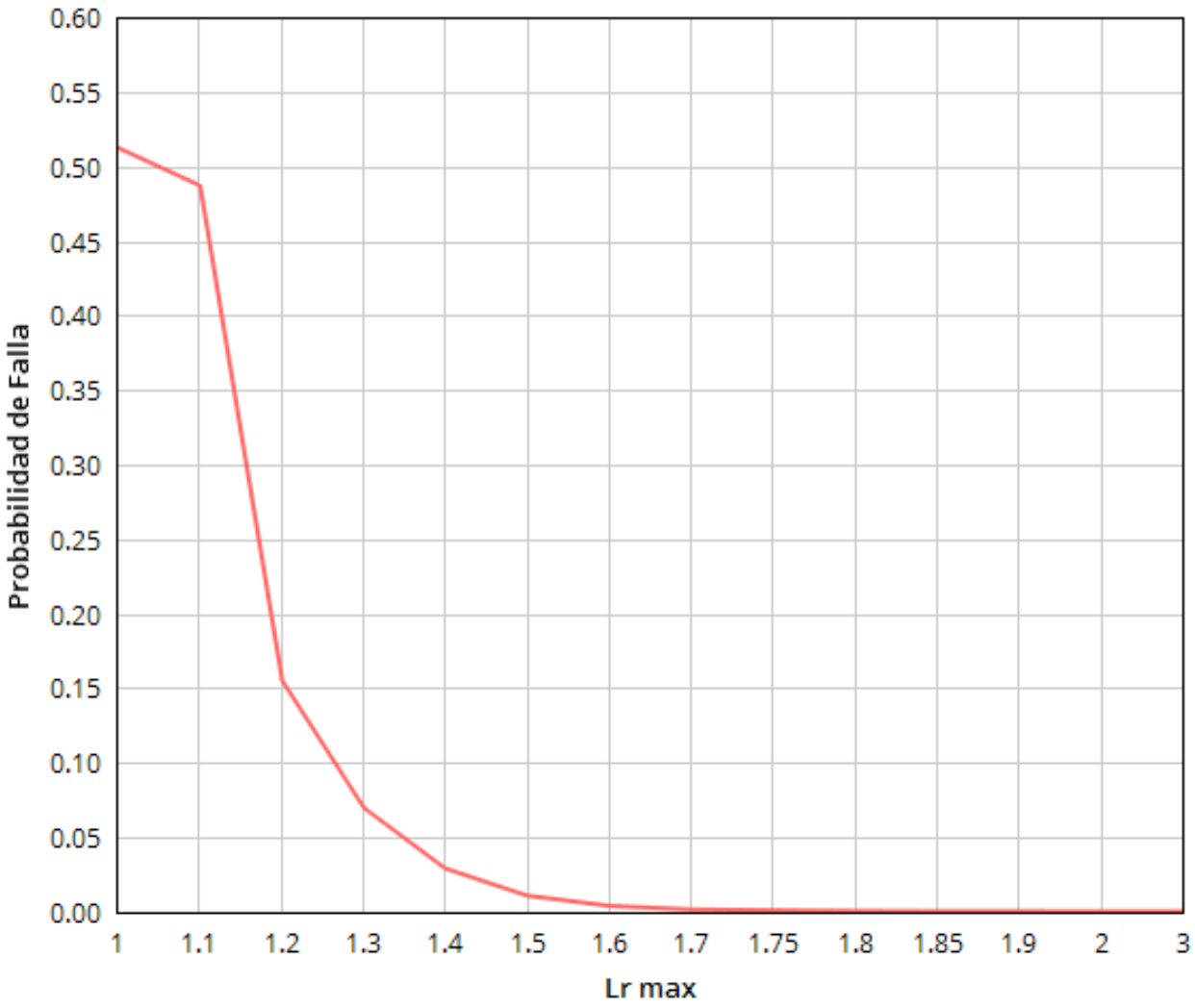

Figura 7.1: Experimentación con L_r^{max}

De lo observado en la anterior figura y la probabilidad esperada podemos ver que el valor para L_r^{max} podría ser alguno de los siguientes:

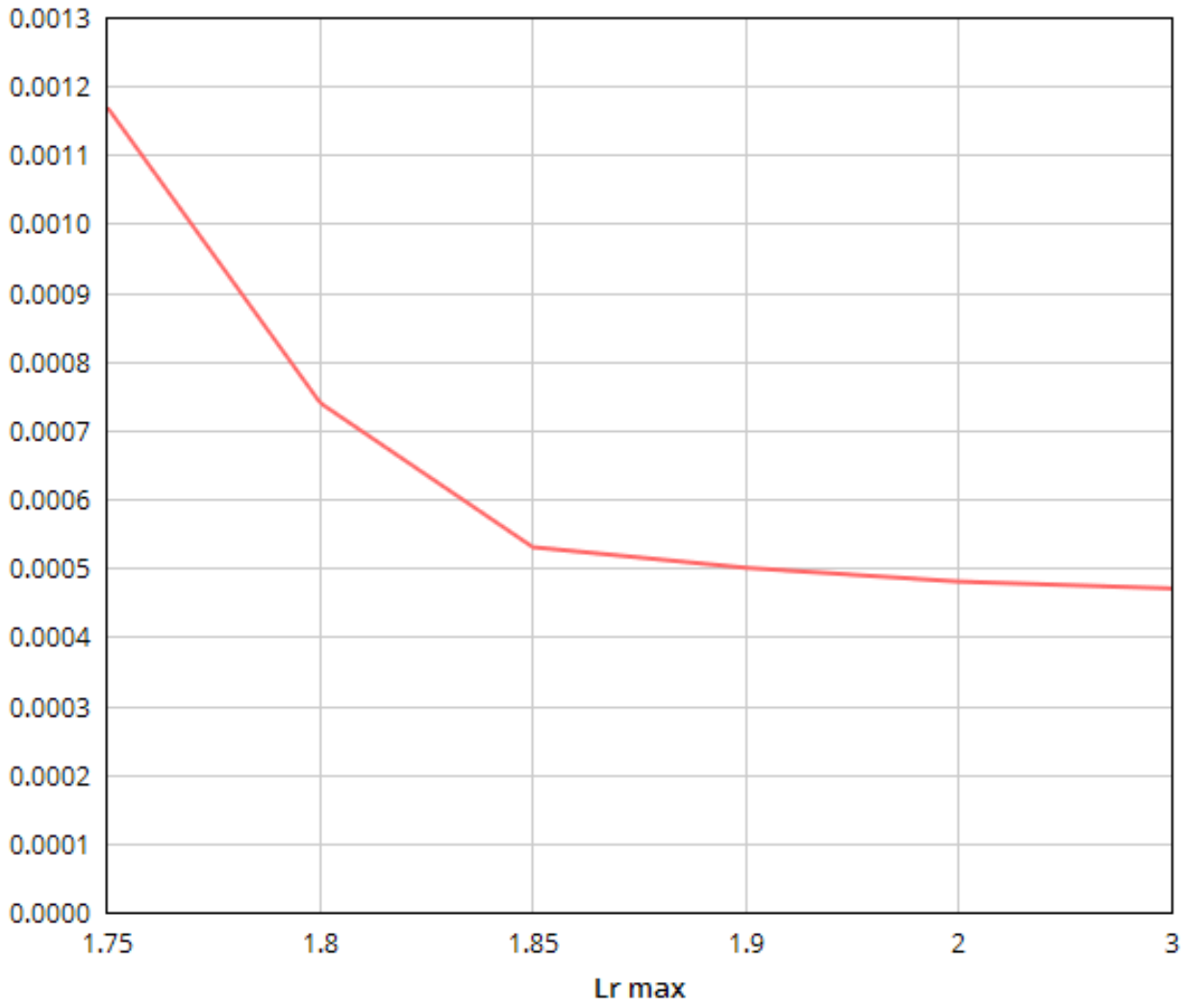

Figura 7.2: Experimentación con L_r^{max}, valores con probabilidad esperada

7.5. Conclusiones

En este capítulo se detallaron las características de los elementos entregados, se describieron las características funcionales del producto que hasta el momento no habían sido expresadas, se describieron las características de uso del sistema y se registran las validaciones realizadas. Culminado este capítulo se deja la idea de como es el producto entregado. En el siguiente capítulo se desarrollaran las conclusiones del trabajo, haciendo énfasis en la arquitectura, en la resolución de

los escenarios y se dedicara un apartado referente a trabajo a futuro.

Capítulo 8

CONCLUSIONES

8.1. Introducción

Es este capítulo se detallarán nuestras conclusiones respecto al producto alcanzado, con el fin de brindar una mayor claridad a la hora de exponerlas optamos por separarlas según sean concernientes a la arquitectura del sistema o respecto a la resolución de los escenarios.

8.2. El camino recorrido

Comenzamos este trabajo realizando un relevamiento del estado del arte guiado al principio por los tutores. Se realizo relevamiento en documentos académicos y en artículos con menos formalidad, como ser presentaciones, paginas web y foros. Logramos confeccionar una lista de características deseables para una plataforma dedicada a solventar la problemática tratada en este trabajo. Sobre esta base, se realizo un análisis de requerimientos, llegando a un consenso con los tutores, de un alcance para el producto. Se realizo un relevamiento que nos permitió elegir las tecnologías adecuadas a nuestro criterio, para aportar una plataforma productiva para los interesados y lista para ser extendida. Entendemos que para esto una arquitectura preparada para crecer es fundamental, y a eso apuntamos. Logramos una arquitectura orientada a micro servicios, con responsabilidades bien definidas, la cual tiene entre sus ventajas, una mayor flexibilidad a la hora de crecer. Por ejemplo, en trabajos posteriores, se pueden abordar mejoras y perfeccionamiento de la lógica de calculo, de la gestión de recursos y de la interfaz web, completamente en paralelo y bajo responsabilidad de diferentes equipos. Desde el comienzo del proyecto siempre estuvo como meta, dar el primer paso en la realización de una plataforma preparada para crecer. Entregamos una arquitectura la cual esta preparada para instalarse en entornos de nube, con el potencial de crecimiento que

esta infraestructura brinda.

Se trabajó con una instalación en entornos de cloud computing, más específicamente con las tecnologías de AWS (`aws.amazon.com`), y se entregó una instalación productiva en servidores de la facultad. Se detallan ambas instalaciones y se entrega un listado de pautas a seguir para una nueva implantación.

Este trabajo planteó desafíos nuevos desde el principio, desde la realización de un relevamiento de estado de arte, la selección de tecnologías acorde al producto que se quería entregar, las curvas de aprendizaje recorridas sobre esas tecnologías, las limitaciones de las mismas que nos llevaron a tener que tomar decisiones acorde y la complejidad del problema a resolver, el cual en su documentación original no está completo.

8.3. El trabajo a futuro

Dado el alcance del proyecto y teniendo en cuenta el análisis del estado del arte en sistemas de similares características, además del producto final alcanzado, entendemos que se abrieron varios caminos sobre los cuales se puede seguir avanzando. Con tal fin en esta sección recomendaremos aspectos en los que consideramos se pueden comenzar o continuar trabajando.

8.3.1. En el producto

- Escenarios: Agregar escenarios enriquecería la experiencia de usuario, además de potenciar la herramienta.
- Análisis de resultados: Brindar más opciones para el análisis de resultados, por ejemplo, el diagrama FAD.
- Exportación de cálculos: Brindar la posibilidad de extraer los resultados obtenidos, esto es brindar la opción de exportar datos ya sea a un formato establecido o simplemente a un archivo de texto.
- Paralelización: A la hora de resolver el escenario, se podrían incorporar diferentes estrategias de ser posible según cada calculo, con las cuales se puedan resolver el calculo en paralelo.
- Gestión y control de Usuarios: Se podría ampliar las funcionalidades respecto a gestión de usuarios, en caso que a futuro sea de interés, tanto controlar el registro de nuevo usuarios así como licencias y acceso a funcionalidades por roles.

8.3.2. En la arquitectura

- Paralelización de cálculo: Incorporar tecnologías que exploten el paralelismo de calculo local en una instancia, de modo de lograr un escalamiento vertical.
- Micro servicio de gestión de usuarios: En la arquitectura entregada, la gestión de usuarios queda en responsabilidad del microservicio Turing, el cual además es responsable de la gestión de los cálculos. En caso de que la plataforma tienda a crecer en usuarios y funcionalidades, se recomienda crear un nuevo micro servicio dedicado a la gestión de usuarios, y dejar en Turing únicamente la responsabilidad del manejo de cálculos.
- Cluster de servidor de descubrimiento: En la arquitectura entregada se tiene solo 1 servidor de descubrimiento. En caso de que en un futuro la cantidad de instancias de los microservicios se incremente, se recomienda crear un cluster de servidores de descubrimiento, para evitar latencia en las comunicaciones entre los microservicios.

8.4. El cierre

Concluimos este trabajo con la certeza de haber aportado nuestro granito de arena a la universidad y a la comunidad. Nos llevamos la experiencia y la madurez de encarar un trabajo de este porte y deseamos que sea el primero de muchos pasos sobre el camino que empezamos a transitar.

Bibliografía

[1] M. S. Attia. *Assessment of corrosion damage acceptance criteriain API579-ASME/1 code*. Springer Science Business Media Dordrecht, 2014.

[2] *Blog de noticias de la compañia desarrolladora de VINDIO, ultimo acceso 17/09/2018*. `http://inescoingenieros.com/en/blog-news/`.

[3] *Compañía QuestIntegrity, ultimo acceso 17/09/2018*. `www.questintegrity.com/`.

[4] *Estilo de arquitectura de microservicios, ultimo acceso 17/09/2018*. `docs.microsoft.com/es-es/azure/architecture/guide/architecture-styles/microservices`.

[5] *Informe sobre PC Praise según la NEA, ultimo acceso 17/09/2018*. `www.oecd-nea.org/tools/abstract/detail/ests0071/`.

[6] American Petroleum Institute. *Publications Programs and Services*. American Petroleum Institute, 2017.

[7] The British Standards Institution. *Guide to methods for assessing the acceptability of flaws in metallic structures*. BSI Standards Limited, 2013.

[8] J. Katsuyamaa. *Benchmark analysis on probabilistic fracture mechanics analysis codesconcerning fatigue crack growth in aged piping of nuclear powerplants*. International Journal of Pressure Vessels y Piping 117-118 (2014) 56e63, 2013.

[9] Paul Scott, Robert Kurth, Andrew Cox, Rick Olson y Dave Rudland. *Development of the PRO-LOCA Probabilistic Fracture Mechanics Code, MERIT Final Report*. Swedish radiation safety authority, 2010.

[10] *Sitio de los desarrolladores de CrackWISE, ultimo acceso 17/09/2018*. `www.twisoftware.com`.

[11] *Sitio de los desarrolladores de CrackWISE, ultimo acceso 17/09/2018*. `http://www.twisoftware.com/software/integrity-management-software/riskwise-for-process-plant/`.

[12] *Sitio de los desarrolladores de CrackWISE, ultimo acceso 17/09/2018.* `http://www.twisoftware.com/software/integrity-management-software/integriwise/`.

[13] *Sitio de los desarrolladores de PC-CRACK, ultimo acceso 17/09/2018.* `www.structint.com`.

[14] *Sitio de los desarrolladores de PRO LOCA, ultimo acceso 17/09/2018.* `www.battelle.org`.

[15] *Sitio de los desarrolladores de VINDIO, ultimo acceso 17/09/2018.* `www.inescoingenieros.com`.

[16] *Sitio de los desarrolladores de VINDIO, ultimo acceso 25/06/2017.* `www.inescoingenieros.com/vindio/`.

[17] W. Wu, G.X. Cheng, Y. Li, Q Zhou y H.J. Hu. *Failure Assessment of Cracked Pipes Based on Failure Assessment Diagram Considering Random and Fuzzy Uncertainties.* Elsevier Ltd, 2015.