

**Universidad de las Ciencias Informáticas**

**Facultad 6**



**Título: “Alas BioSyS: Proceso de Pruebas de Liberación de la Facultad 6”**

**Trabajo de Diploma para optar por el Título de  
Ingeniero en Ciencias Informáticas**

**Autores:** Yuneisi Castillo Tamayo  
Yulieska Ramos Fernández

**Tutores:** Ing. Yunet González Mulet  
Ing. Asnier Enrique Góngora Rodríguez

Ciudad de La Habana, junio de 2009  
“Año del 50 Aniversario del triunfo de la Revolución”



*“Hay una regla para los industriales que es: haz los productos de la mejor calidad posible al menor costo posible.”*

*Henry Ford*

## **DECLARACIÓN DE AUTORÍA**

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los    días del mes de junio del año 2009.

---

Yuneisi Castillo Tamayo (Autor)

---

Yulieska Ramos Fernández (Autor)

---

Ing. Yunet González Mulet (Tutor)

---

Ing. Asnier Enrique Góngora Rodríguez (Tutor)

## **DATOS DE CONTACTO**

Ingeniera: Yunet González Mulet

Universidad de Ciencias Informáticas, La Habana, Cuba.

E-mail: ygonzalezmu@uci.cu

Ingeniero: Asnier Enrique Góngora Rodríguez

Universidad de Ciencias Informáticas, La Habana, Cuba.

E-mail: agongora@uci.cu

## Agradecimientos

*A nuestro eterno Comandante en Jefe Fidel y a La Revolución Cubana por ser paradigma, guía y permitimos estudiar en esta Universidad de Excelencia.*

*A nuestros profesores de La UCI por formarnos como profesionales y jóvenes dignos de estos tiempos.*

*A nuestros tutores por su asesoría y su gran aporte.*

*A los profesores Maypher, Martha y Heydi por su colaboración sistemática en la realización de este trabajo.*

*A nuestro compañero de proyecto Edel por su apoyo y su ayuda incondicional.*

*A todas aquellas personas que colaboraron en la realización de este trabajo.*

*A nuestros compañeros de clases.*

## Dedicatoria

*De Yulieska:*

*A mi mamá por todo su amor, su dedicación, por ayudarme a llegar este momento por su apoyo incondicional y su confianza en mí.*

*A mi hermana por estar siempre a mi lado, apoyándome en todo, por su amor incondicional, por ser siempre el ejemplo que me impulsó a seguir hacia delante.*

*A mi papá, por desde pequeña ayudarme a llegar a ser quien soy.*

*A mi abuelita por quererme tanto por esperar siempre mi regreso con tantas ansias y todas las cosas ricas que me gustan.*

*A mi tío por su apoyo, su amor, por ser a su manera como otro papá para mí.*

*A mi novio Alaen por su amor, su apoyo incondicional y por estar siempre cuidando de mí.*

*A mi abuela, a mi tía y a mis primos gracias por estar siempre allí de una forma u otra.*

*A mi sobrinito(a) que está por llegar para que le sirva de ejemplo y guía.*

*A mis amigos a los de antes que a pesar de la distancia siguen allí, a los que conocí aquí con los que he compartido los momentos buenos y en los no tan buenos que se nos han presentado durante estos cinco años en nuestra casa grande la UCI.*

*A todos los que de una forma u otra me han ayudado a llegar hasta aquí, por su confianza, su apoyo, su amor*

*Muchas Gracias*

*De Yuneisi:*

*A mi mamá Bárbara y mi papá Antonio por su amor, dedicación y apoyo constante, por enseñarme a no creer en imposibles e inspirarme con su ejemplo y por depositar tanta confianza en mí, nunca los defraudaré.*

*A mi hermano Juvier y a toda mi familia por ser tan especial y apoyarme en los momentos difíciles.*

*A mi novio Alfredo por su apoyo y dedicación, por el amor y la protección que me ha dado en todos estos años.*

*A mi padrastro Miguel por su cariño, apoyo y amor incondicional.*

*A mi tía Mileydis por quererme y cuidarme como si fuera mi propia madre, por apoyarme y ayudarme a seguir adelante.*

*A mis amigos de La UCI por regalarme su valiosa amistad en estos cinco inolvidables años, siempre los llevaré en mi corazón.*

*A mis compañeros de aulas que día tras día compartimos momentos inolvidables.*

*A Diosdado y a su familia por su amistad desinteresada y su apoyo en los momentos difíciles.*

*A todos, muchas gracias.*

## **Resumen**

En la actualidad se ha incrementado enormemente el interés general por el concepto calidad, pues los clientes son cada vez más exigentes y no aceptan un producto que no satisfaga sus expectativas. Es por ello que, en la actualidad, cada vez son más las empresas que destinan mayores recursos para cumplir o llegar a satisfacer lo que este término implica.

En la facultad 6 de la Universidad de las Ciencias Informáticas, se ha desarrollado una aplicación que incluye herramientas tales como: modelación, edición de ecuaciones diferenciales, simulación y análisis de los resultados de las simulaciones usando técnicas de Inteligencia Artificial, las cuales facilitan y apoyan las investigaciones de los científicos en el área de la Biología de Sistemas y aporta la posibilidad de almacenar toda la información que se desee acerca del sistema biológico estudiado.

En este trabajo se desarrolla un análisis de los principales temas relacionados a la calidad y la documentación referente al proyecto. Se diseñan los casos de prueba y se definen los pasos a seguir para desarrollar las pruebas de calidad. Además se crea un plan de pruebas para garantizar el orden de este proceso. Se documentan las no conformidades encontradas en los procesos de prueba y se realizan varias iteraciones para poder corregir todos los errores encontrados y comparar los resultados encontrados con los esperados por el software, quedando el producto listo para ser entregado al grupo de calidad de la universidad y logrando un producto con un mínimo de errores.

## Índice

<b>INTRODUCCIÓN</b> .....	<b>1</b>
<b>CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA</b> .....	<b>5</b>
1.1 CALIDAD DE SOFTWARE .....	5
1.2 LAS PRUEBAS EN LA CALIDAD DEL SOFTWARE.....	6
1.3 EL PROCESO DE PRUEBA .....	9
1.3.1 Niveles de pruebas .....	9
1.4 MÉTODOS DE PRUEBAS .....	12
1.4.1. Prueba de caja blanca .....	12
1.4.2. Pruebas de caja negra.....	15
1.5. PLAN DE PRUEBAS. ....	17
1.6 HERRAMIENTAS PARA EL DESARROLLO DE LAS PRUEBAS DE SOFTWARE .....	18
1.7 DISEÑO DE CASOS DE PRUEBA.....	20
1.8 METODOLOGÍA OPENUP .....	21
1.8.1 Rol de probador .....	22
1.9 CONCLUSIONES DEL CAPÍTULO.....	23
<b>CAPÍTULO 2: DISEÑO Y APLICACIÓN DE LAS PRUEBAS DE LIBERACIÓN A BIOSYS</b> .....	<b>25</b>
2.1 CARACTERÍSTICAS DEL SISTEMA A PROBAR .....	25
2.2 REQUERIMIENTOS NECESARIOS PARA REALIZAR LAS PRUEBAS .....	27
2.3 ESTRATEGIA DE PRUEBAS. ....	28
2.4 ENTORNO DE PRUEBA .....	30
2.5 CRONOGRAMA DE PRUEBA.....	31
2.6 CASOS DE PRUEBA .....	33
2.6.1 Pruebas de Caja Negra .....	33
2.6.2 Pruebas de Integración.....	67
2.7 PRUEBAS DE CARGA Y ESTRÉS.....	68
2.7.1 Simulador de conexiones MySQL .....	69
2.7.2 Evaluación de atributos de calidad.....	71
2.8. CONCLUSIONES DEL CAPÍTULO .....	72
<b>CAPÍTULO 3: RESULTADOS DE LAS PRUEBAS REALIZADAS</b> .....	<b>73</b>
3.1. ANÁLISIS DE LOS RESULTADOS OBTENIDOS EN LAS PRUEBAS A LA DOCUMENTACIÓN .....	73

3.2. ANÁLISIS DE LOS RESULTADOS OBTENIDOS EN LAS PRUEBAS FUNCIONALES .....	77
3.3. ANÁLISIS DE LOS RESULTADOS OBTENIDOS EN LAS PRUEBAS DE INTEGRACIÓN.....	78
3.4. ANÁLISIS DE LOS RESULTADOS OBTENIDOS EN LA PRUEBA DE SISTEMA.....	78
3.4.1 Análisis de los resultados obtenidos en la prueba de Carga y Estrés realizada con estudiantes reales.....	79
3.4.2. Análisis de los resultados obtenidos en las pruebas realizadas con el Simulador de Conexiones MySQL.....	81
3.4.3. Análisis de los resultados obtenidos en la evaluación de los atributos de calidad .....	82
3.5. ANÁLISIS DE LAS NO CONFORMIDADES MÁS SIGNIFICATIVAS ENCONTRADAS EN LA DOCUMENTACIÓN	83
3.6. ANÁLISIS DE LAS NO CONFORMIDADES MÁS SIGNIFICATIVAS ENCONTRADAS EN LA APLICACIÓN .....	84
3.7. CONCLUSIONES DEL CAPÍTULO.....	84
<b>CONCLUSIONES.....</b>	<b>86</b>
<b>RECOMENDACIONES.....</b>	<b>87</b>
<b>REFERENCIAS BIBLIOGRÁFICAS.....</b>	<b>88</b>
<b>BIBLIOGRAFÍA.....</b>	<b>89</b>
<b>ANEXOS.....</b>	<b>92</b>
<b>GLOSARIO DE TÉRMINOS.....</b>	<b>132</b>

## **Índice de figuras**

Figura 1. Modelo en V. Calidad en el ciclo de vida del software. ....	7
Figura 2. Grafos de flujo. ....	13
Figura 3. Representación de bucles simples, anidados, concatenados y estructurados. ....	14
Figura 4. Técnicas de diseño de casos de pruebas. ....	21
Figura 5. Capas de OpenUP ....	22
Figura 6. Representación de las actividades y artefactos relacionados con el probador. ....	23

## **Introducción**

Las metodologías de desarrollo de software están cobrando cada vez más importancia en la gestión de las empresas y en el desarrollo de productos pasando de ser aspecto auxiliar a ser clave en la competitividad de las empresas y en el éxito de los productos.

Todas las metodologías y herramientas tienen un único fin, producir software de gran calidad. Para lograr esto se rigen por normativas, métricas, indicadores de la calidad que guían y controlan el proceso de desarrollo, como resultado de seguirlo a cabalidad se tendrá un software apto para satisfacer las necesidades tanto las expresadas por el cliente como las implícitas.

Uno de los principales problemas que presenta la industria del software en la actualidad es el tema relacionado con la calidad, el cual se debe al repentino crecimiento de la complejidad de los productos y en gran medida se debe también a que el personal no está lo suficientemente calificado. Por esta razón se ha incrementado enormemente el interés general respecto a este tema, pues los clientes son cada vez más exigentes y no aceptan un producto que no satisfaga sus expectativas. Las empresas se están enfrentando a un público que, lejos de ser conformista, espera de su proveedor resultados con altos niveles de calidad.

La obtención de un software con calidad implica la utilización de metodologías o procedimientos estándares para el análisis, diseño, programación y prueba del software que permitan uniformar la filosofía de trabajo, en aras de lograr una mayor confiabilidad y facilidad de prueba, a la vez que eleven la productividad, tanto para la labor de desarrollo como para el control de la calidad del software.

En nuestro país el desarrollo de la industria del software es una tarea de gran prioridad debido a la alta perspectiva económica que posee. La situación actual y las perspectivas de la industria de software cubana están caracterizadas por el trabajo que se viene realizando en materia de capacitación y en la inserción en el mercado nacional como internacional. Cursos de certificación nacional en herramientas informáticas de punta, cursos de preparación emergente en herramientas específicas de desarrollo de software, cursos emergentes para programadores para dar respuestas a oportunidades de negocios en mercados identificados, plan de preparación en aquellas herramientas informáticas y en los idiomas necesarios, que permitan dar una respuesta rápida y con calidad a las demandas puntuales que se vayan identificando. A pesar de ello, los resultados alcanzados no cubren

las expectativas, ya que la productividad es baja, la cantidad real de recursos a consumir es impredecible y el trabajo realizado casi nunca tiene la calidad y profesionalidad requerida. Los proyectos se terminan demasiado tarde y los beneficios que pudieran obtenerse al utilizar los mejores métodos e instrumentos en las distintas etapas, no se detectan en este medio indisciplinado y caótico de desarrollo.

Para poder dar solución a estos problemas no es suficiente con la aplicación de nuevas metodologías y tecnologías, es necesario también aprender a administrar procesos de software para que se realicen con calidad. Todo esto requiere del establecimiento de sistemas de calidad y por tanto definir rigurosamente los procesos de software que se llevan a cabo en la empresa.

En la facultad 6 de la Universidad de las Ciencias Informáticas se está desarrollando una aplicación, a petición del Centro de Inmunología Molecular (CIM), que contiene algoritmos y herramientas que facilitan y apoyan las investigaciones de los científicos en el área de la Biología de Sistemas. Con el uso de las herramientas incluidas en como son: modelación, edición de ecuaciones diferenciales, simulación y análisis de los resultados de las simulaciones usando técnicas de Inteligencia Artificial; se hace posible la realización de un estudio más completo de los sistemas biológicos al poder efectuar una investigación que implica todo el flujo de los procesos necesarios en este tipo de trabajo, y se agrega además la posibilidad de almacenar toda la información que se desee acerca del sistema biológico estudiado, pero este producto no ha sido probado más allá del uso de sus propios desarrolladores por lo que no se puede asegurar que sea lo suficientemente confiable y cumpla con todo los requisitos solicitados por el cliente.

Luego de analizar la problemática referente al tema de la calidad de los productos de software se identifica como **problema científico**: ¿Cómo verificar que el Software de Simulación y Análisis de Sistemas Biológicos cumple con la calidad establecida? Para lo cual se propone realizar un adecuado diseño de pruebas, que permita conocer los errores, trabajar en base a estos logrando así un sistema más estable y eficiente.

El problema planteado se enmarca en el **objeto de estudio**: Proceso de pruebas de calidad de software.

El objeto de estudio delimita el **campo de acción**: La aplicación de pruebas de Liberación al Software de Simulación y Análisis de Sistemas Biológicos en la facultad 6.

Para dar solución al problema se define como **objetivo general**: Desarrollar las pruebas de liberación al BioSyS verificando así el cumplimiento de los requerimientos de software establecidos.

Del cual se derivan los siguientes **objetivos específicos**:

- Aplicar pruebas a la documentación de BioSyS.
- Realizar Diseños de Casos de Pruebas.
- Desarrollar pruebas de Caja Negra.
- Aplicar pruebas de sistema.
- Documentar todo el proceso de pruebas a desarrollar.

Para lograr este objetivo, se realizarán las siguientes **tareas de investigación**:

1. Estudio de los aspectos importantes vinculados a la calidad del Software.
2. Selección de las pruebas de calidad a realizar.
3. Realización del Plan de Pruebas.
4. Revisión del documento Especificación de Requisitos.
5. Revisión del documento Descripción de Casos de Uso.
6. Revisión del documento Manual de Usuario.
7. Revisión del documento Manual de Instalación.
8. Realización de diseños de Casos de Pruebas.
9. Ejecución de las pruebas diseñadas.
10. Realización de pruebas de carga y estrés.
11. Confección del documento de No Conformidades.

El trabajo consta de introducción, tres capítulos, conclusiones, recomendaciones, referencia bibliográfica, bibliografía, glosario de términos y anexos.

En el **Capítulo 1 Fundamentación Teórica** se hace una descripción de los aspectos más importantes vinculados a la calidad del software. Se exponen conceptos, métodos, procedimientos y pruebas usados en la actualidad y se incluyen las pruebas que se realizan para validar la calidad del Software para la Simulación y Análisis de Sistemas Biológicos, así como la metodología a utilizar.

En el **Capítulo 2 Diseño y aplicación de las pruebas de liberación a BioSyS** se hace una breve descripción del producto partiendo de la elaboración de un plan de pruebas, se presenta el diseño de las mismas y se aplican las pruebas a los módulos de modelación, edición de ecuaciones diferenciales, simulación y análisis.

En el **Capítulo 3 Resultados de las pruebas realizadas** se describen los resultados obtenidos después de aplicarse las pruebas a los módulos de funcionalidades comunes, edición de ecuaciones diferenciales, simulación y análisis, y se hace una breve descripción de las no conformidades encontradas.

# Capítulo 1

## Fundamentación Teórica

En este capítulo se realiza un resumen de los principales aspectos que se relacionan con la calidad. Se brinda información acerca de las pruebas que le son realizadas al software y se hace alusión a la metodología empleada así como los métodos de prueba más utilizados

### 1.1 Calidad de software

En la actualidad el desarrollo de software se ha convertido en el principal aliado de las organizaciones, pues el desarrollo y supervivencia de estas depende en gran medida de los sistemas informáticos. Es por ello que se hace tanto énfasis en la necesidad de construir un software de buena calidad.

Muchas veces parece que el software de calidad es aquél que brinda lo que se necesita con adecuada velocidad de procesamiento. Sin embargo es mucho más, tiene que ver con la corrección, pero también con usabilidad, costo, consistencia, confiabilidad, compatibilidad, utilidad, eficiencia y cumplimiento de los estándares. Todos estos aspectos de la calidad pueden ser sometidos a pruebas que determinen la calidad incluso la documentación para el usuario debe ser probada.

Según la Organización Internacional para la Estandarización (ISO) en su Norma de Calidad ISO 9000:2000: calidad es el "grado en que un conjunto de características inherentes cumple con los requisitos", lo cual quiere decir que es el nivel de satisfacción que presenta un producto en cuanto a una serie de criterios predefinidos. [1]

Según Pressman es "Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados y con las características implícitas que se esperan de todo software desarrollado profesionalmente". [2]

Existen tres puntos importantes de la definición de calidad de software:

- Los requerimientos del software son los fundamentos desde los que se mide la calidad.
- Los estándares específicos definen un conjunto de criterios de desarrollo que guían la forma de aplicación de la ingeniería de software.
- Existen requerimientos implícitos que no se mencionan.

## 1.2 Las pruebas en la calidad del software

De acuerdo a la IEEE el concepto de prueba se define como: “Una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente”. [3]

Pressman plantea:” Las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación.”[2]

Un concepto más específico dado por algunos desarrolladores de software es que las pruebas son: “Cualquier intento de demostrar que el software tiene propiedades por debajo de la calidad requerida”. [4]

Las técnicas para encontrar problemas en un programa son extensamente variadas y van desde el uso del ingenio por parte del personal de prueba hasta herramientas automatizadas que ayudan a aliviar el peso, el costo y el tiempo de esta actividad. Existen cuatro elementos que son importantes a la hora de definir las pruebas, ellos son confiabilidad, costo, tiempo y calidad. En la medida que se deseen pruebas confiables y un software de calidad, el tiempo y el costo se incrementarán.

La fase de pruebas es una de las más costosas del ciclo de vida software. En sentido estricto, deben realizarse pruebas de todos los artefactos generados durante la construcción de un producto, lo que incluye especificaciones de requisitos, casos de uso, diagramas de diversos tipos y, por supuesto, el código fuente y el resto de productos que forman parte de la aplicación. Obviamente, se aplican diferentes técnicas de prueba a cada tipo de producto software.

En la figura 1 se representa el modelo V, este es un modelo que representa todo el proceso de calidad a lo largo de todo el ciclo de vida del software, este modelo es una evolución más robusta y efectiva que el modelo cascada, en el cual se mejoran (iteraciones entre fases, mucho mas flexible, planifica pruebas, etc.); se aumentan (fases como: la verificación y la validación) y se corrigen fases que nos permiten producir un software de mayor calidad

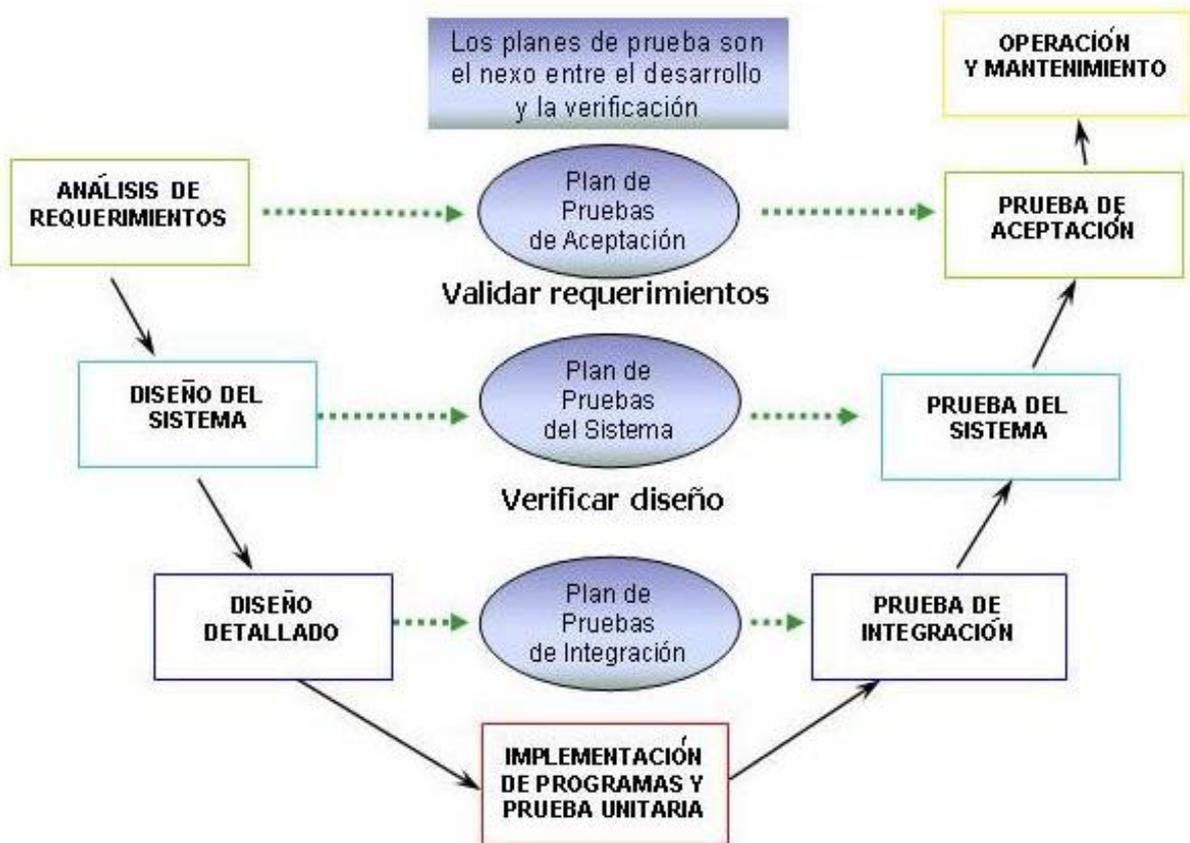


Figura 1. Modelo en V. Calidad en el ciclo de vida del software.

### Objetivos de las Pruebas de Calidad

Los objetivos fundamentales que se persiguen a la hora de realizar las pruebas son:

- Descubrir un error con un mínimo de costo y tiempo.
- Verificar que el producto cumple con los requisitos establecidos por el cliente.
- Ofrecer un producto altamente confiable.

Estos objetivos cambian la visión acerca de que una prueba exitosa es aquella en la que no se descubren errores. El objetivo principal es diseñar casos de pruebas que sometan al software a rigurosas pruebas lo que permita que reflejen diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo.

Si las pruebas se llevan a cabo con éxito se descubrirán errores en el software, dándole a este una mayor fiabilidad. Debemos de tener en cuenta que el proceso de prueba no asegura la ausencia de defectos solo demuestra que existen defectos en el software.

### Principios de Pruebas

Antes de realizar un diseño de caso de prueba efectivo se deben de tener en cuentas los principios básicos que guían las pruebas del software:

- A todas las pruebas se les debería poder hacer un seguimiento hasta los requisitos de los clientes (trazabilidad).
- Las pruebas deberían planificarse antes de que empiecen.
- El principio de Pareto es aplicable a la prueba del software (“donde hay un defecto, hay otros”).
- Las pruebas deberían empezar por “lo pequeño” y progresar hacia “lo grande”.
- No son posibles las pruebas exhaustivas.
- Para ser más efectivas, las pruebas deberían ser conducidas por un equipo independiente

### Estrategias de Pruebas

Una estrategia de prueba del software integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que dan como resultado una correcta construcción del software, también proporciona una guía a seguir para el desarrollo del software. Cualquier estrategia de prueba debe tener presente la planificación de la prueba, el diseño de casos de prueba, la ejecución de las pruebas y evaluación de los datos resultantes.

Una estrategia de prueba del software debe ser suficientemente flexible para promover la creatividad y la adaptabilidad necesarias para adecuar la prueba a todos los grandes sistemas basados en software. Al mismo tiempo, la estrategia debe ser suficientemente rígida para promover un seguimiento razonable de la planificación y la gestión a medida que progresa el proyecto.

La planificación de la Estrategia de Prueba puede reducir significativamente el esfuerzo necesario para el desarrollo de las pruebas adecuadas, reducir el tiempo de realización y ejecución de las mismas y disminuir los altos costos que se generan.

### 1.3 El proceso de prueba

El proceso de prueba de un software consta de varias etapas dentro de ellas las más importantes son:

- Inspección del análisis (verifica si se cometieron errores o falla en la etapa de análisis).
- Inspección del diseño (debe ser completo y eficiente).
- Inspección del código (observar el entendimiento y facilidad del código).
- Pruebas unitarias (probar cada método de las clases implementadas por separado).
- Pruebas de integración (probar todas las clases, verificando que compaginen entre sí).
- Pruebas de validación de requerimientos (verificar que cumple con todos los requerimientos exigidos por el cliente).

#### 1.3.1 Niveles de pruebas

##### Prueba unitaria o de unidad

Constituye una forma de probar el correcto funcionamiento de un módulo de código. El objetivo de las pruebas unitarias es aislar cada parte del programa y mostrar que las partes individuales funcionan correctamente. Para que una prueba unitaria sea buena debe ser automatizable, completa, reutilizable, independiente y profesional. Estas pruebas por si solas no descubrirán todos los errores del código

En este tipo de pruebas se busca probar elementos como la interfaz del módulo, para asegurar que la información fluye de forma adecuada hacia y desde la unidad de programa que está siendo probada. Se examinan las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente conservan su integridad durante todos los pasos de ejecución del algoritmo. Se prueban las condiciones límite para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento. Se ejercitan todos los caminos independientes (caminos básicos) de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez y finalmente, se prueban todos los caminos de manejo de errores.

##### Prueba integral o prueba de integración

Es aquella que se realiza una vez que se han aprobado las pruebas unitarias, se llevan a cabo durante la construcción del sistema. La misma es una técnica sistemática para construir la estructura del programa mientras al mismo tiempo, se lleva a cabo pruebas para detectar errores asociados con la interacción. El objetivo es tomar los módulos probados en unidad y estructurar un programa que esté de acuerdo con el que dicta el diseño.

Esta prueba se puede plantear desde un punto de vista estructural o funcional. Las pruebas estructurales de integración son similares a las pruebas de caja blanca; pero trabajan a un nivel conceptual superior. En lugar de referirnos a sentencias del lenguaje, nos referiremos a llamadas entre módulos. Se trata pues de identificar todos los posibles esquemas de llamadas y ejercitarlos para lograr una buena cobertura de segmentos o de ramas.

Las pruebas funcionales de integración son similares a las pruebas de caja negra. Este tipo de prueba es fundamental para nuestro trabajo por que es utilizado en el desarrollo del mismo. Aquí trataremos de encontrar fallos en la respuesta de un módulo cuando su operación depende de los servicios prestados por otro(s) módulo(s). Según nos vamos acercando al sistema total, estas pruebas se van basando más y más en la especificación de requisitos del usuario.

#### **Tipos de integración**

- **Integración descendente:** es una estrategia de integración incremental a la construcción de la estructura de programas, en el cual se integran los módulos moviéndose en dirección hacia abajo por la jerarquía comenzando por el control principal (Programa principal).
- **Integración ascendente:** es donde la construcción del diseño empieza desde los módulos más bajos hacia arriba (módulo principal), el procesamiento requerido de los módulos subordinados siempre está disponible y elimina la necesidad de resguardo. La sección de una estrategia de integración depende de las características depende de las características del software y, a veces, del plan del proyecto, en algunos de los casos se puede combinar ambas estrategias.[5]

#### **Pruebas de Sistema**

El software ya validado se integra con el resto del sistema donde algunos tipos de pruebas a considerar son:

- **Rendimiento:** donde se determinan los tiempos de respuesta, el espacio que ocupa el módulo en disco o en memoria, el flujo de datos que genera a través de un canal de comunicaciones, etc.
- **Resistencia:** se establece hasta donde puede soportar el programa determinadas condiciones extremas.
- **Robustez:** donde se muestra la capacidad del programa para soportar entradas incorrectas.
- **Seguridad:** en las que se establecen los niveles de permiso de usuarios, las operaciones de acceso al sistema y acceso a datos.

- **Usabilidad:** en las que se determina la calidad de la experiencia de un usuario en la forma en la que éste interactúa con el sistema, se considera la facilidad de uso y el grado de satisfacción del usuario.
- **Instalación:** donde se definen las operaciones de arranque y actualización del software.
- **Carga y estrés:** es la prueba en la que se verifica que el software puede soportar un número n de conexiones concurrentes sin que se afecte su rendimiento.

### Pruebas de Validación

Son el proceso de revisión que válida que el sistema de software producido cumpla con las especificaciones y que cumpla su cometido. Son normalmente una parte del proceso de pruebas de software de un proyecto, que también utilizan técnicas tales como evaluaciones, inspecciones, y tutoriales. La validación es el proceso de comprobar si lo que se ha especificado es lo que el usuario realmente quería.

**Existen diferentes tipos de prueba de validación, entre ellos están:**

- **Pruebas de aceptación** que son las desarrolladas por el cliente.
- **Pruebas alfa** que son realizadas por el usuario con el desarrollador como observador en un entorno controlado.
- **Pruebas beta** que son realizadas por el usuario en su entorno de trabajo y sin observadores.

Existen dos tipos de requerimientos que deben tomarse en cuenta en la planificación de las pruebas de validación. Los primeros son los requerimientos funcionales, tomados a partir del modelo de casos de uso como se mencionó anteriormente. Los segundos son los llamados no-funcionales, entre los cuales podemos mencionar:

- Desempeño, que tiene que ver con el espacio, tiempo de ejecución y otros recursos del sistema que requiera el software para ejecutarse.
- Documentación, la cual debe estar correcta, inteligible, coherente y completa.
- Volumen de datos, el cual se refiere a la cantidad de carga que puede y debe soportar el sistema.
- Demanda pico, que está relacionado con la tensión o stress que debe soportar el sistema en intervalos de tiempo.
- Recuperabilidad, asociado a la capacidad de respaldo del sistema ante cualquier error de ejecución; por ejemplo, si el sistema maneja archivos de datos que modifica constantemente debe constar de módulos que se encarguen de guardar siempre los archivos antes de

utilizarlos; otro ejemplo de recuperabilidad podría ser la capacidad del sistema para restablecer las condiciones finales antes del error de ejecución. Usabilidad, pues hay que considerar ergonomía del sistema, es decir, facilidad de uso; por ejemplo, se debe tomar en cuenta las condiciones en las cuales el usuario tendrá que trabajar con el sistema para que su jornada no se le haga pesada sino más bien agradable.

### Pruebas de Aceptación

Estas pruebas le permite al cliente validar y verificar todos los requisitos .Las realiza el usuario final en lugar del responsable del desarrollo del sistema.

Entre sus características se encuentran que para poderse desarrollar debe estar presente el usuario, que es un aprueba que está enfocada hacia la prueba de los requisitos de usuario especificados y que está considerada como la fase final del proceso para crear una confianza en que el producto es el apropiado para su uso en explotación.

#### **1.4 Métodos de pruebas**

Existen varios métodos de prueba entre las más importantes se encuentran la prueba de Caja Blanca y la prueba de Caja Negra.

Estas pruebas permiten probar cada una de las condiciones existentes en el programa, identificar claramente las entradas, salidas y estudiar las relaciones que existen entre ellas, permitiendo así maximizar la calidad de las pruebas y en dependencia del resultado se constará con un sistema más estable y confiable. En este trabajo se hace un estudio de ambos pero centrándonos fundamentalmente en los de caja negra que son los que se utilizarán en nuestro proceso de pruebas de Liberación.

##### **1.4.1. Prueba de caja blanca**

La prueba de caja blanca del software se basa en el minucioso examen de los detalles procedimentales. Se comprueban los caminos lógicos del software proponiendo casos de prueba que:

- Garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada módulo.
- Ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa.
- Ejecuten todos los bucles en sus límites y con sus límites operacionales.
- Ejerciten las estructuras internas de datos para asegurar su validez.

Los errores lógicos y las suposiciones incorrectas son inversamente proporcionales a la probabilidad de que se ejecute un camino del programa.

Existen diferentes métodos de pruebas de Caja Blanca, entre ellos se encuentran:

### **Método de prueba de Camino Básico**

El método del camino básico permite obtener una medida de la complejidad de un diseño procedimental, y utilizar esta medida como guía para la definición de una serie de caminos básicos de ejecución, diseñando casos de prueba que garanticen que cada camino se ejecuta al menos una vez. A continuación se muestra la notación del grafo de flujo o grafo del programa.

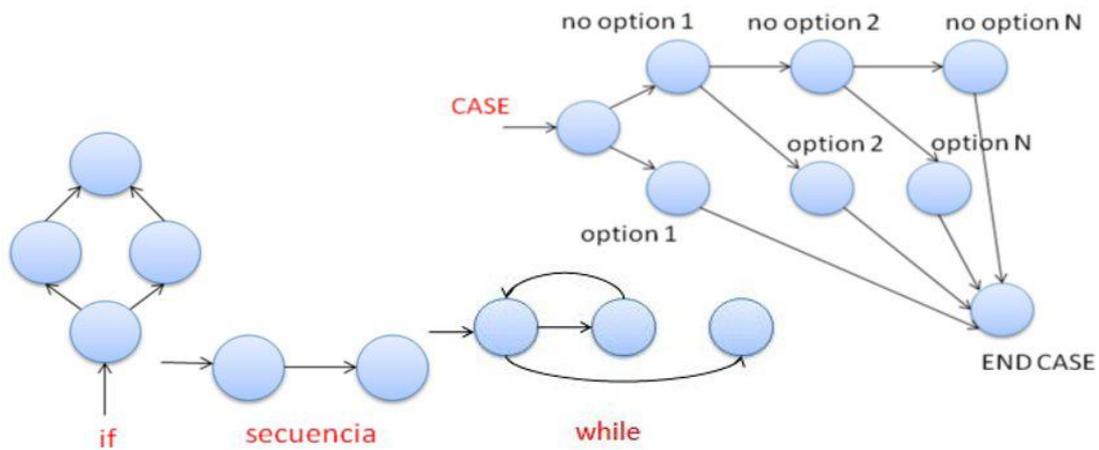


Figura 2. Grafos de flujo.

En el grafo de flujo:

- Cada nodo representa una o más sentencias procedimentales.
- Un solo nodo puede corresponder a una secuencia de pasos del proceso y a una decisión.
- Las flechas (aristas) representan el flujo de control.

Cualquier representación del diseño procedimental se puede traducir a un grafo de flujo. Si en el diseño procedimental se utilizan condiciones compuestas, la generación del grafo de flujo tiene que descomponer las condiciones compuestas en condiciones sencillas.

### **Complejidad Ciclomática**

Es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Establece el número de caminos independientes.

La Complejidad Ciclomática de un grafo de flujo  $V(G)$  establece el número de caminos independientes.

Puede calcularse de tres formas alternativas:

- El número de regiones del grafo de flujo.
- $V(G) = A - N + 2$ , donde A es el número de aristas y N es el número de nodos.
- $V(G) = P + 1$ , donde P es el número de nodos predicado.

Conjunto básico:

Es un conjunto de caminos independientes

Derivación de casos de prueba:

Pasos para realizar las pruebas:

- A partir del diseño o del código fuente, dibujar el grafo de flujo asociado.
- Se calcula la complejidad ciclomática del grafo.
- Se determina un conjunto básico de caminos independientes.
- Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

**Método de prueba de Bucles**

Los bucles son la piedra angular de la inmensa mayoría de los algoritmos implementados en software, sin embargo, les prestamos normalmente poca atención cuando llevamos a cabo las pruebas del software. Esta prueba se centra exclusivamente en la validez de las construcciones de bucles. Se pueden definir cuatro clases diferentes: bucles simples, bucles concatenados, bucles anidados y bucles no estructurados. Estos bucles se muestran en la figura 3.

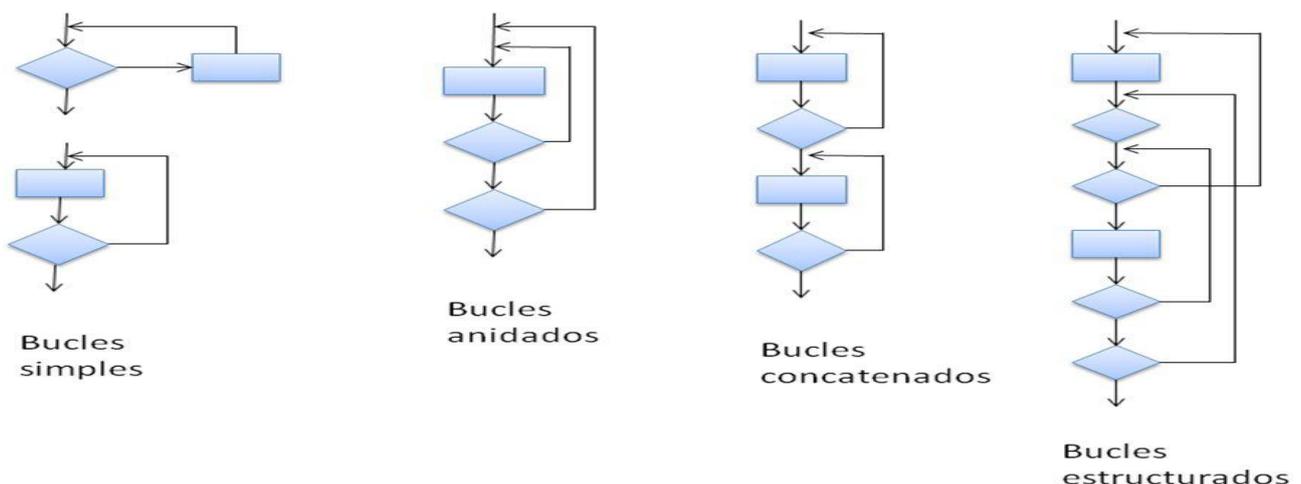


Figura 3. Representación de bucles simples, anidados, concatenados y estructurados.

### Bucles simples

Se les aplica el siguiente conjunto de pruebas, donde  $n$  es el número máximo de pasos permitidos por el bucle:

- Pasar por alto totalmente el bucle.
- Pasar una sola vez por el bucle.
- Pasar dos veces por el bucle.
- Hacer  $m$  pasos por el bucle con  $m < n$ .
- Hacer  $n - 1$ ,  $n$  y  $n+1$  pasos por el bucle.

### Bucles anidados

El método anterior no es factible para los bucles anidados ya que traería como consecuencia un aumento en el nivel de anidamiento y aun número impracticables de pruebas. Para evitar esto seguimos los siguientes pasos

- Iniciamos el proceso por el bucle más interior y los demás bucles con sus valores mínimos.
- Realizamos las pruebas de bucles simples para el bucle más interior, mientras se mantienen en los bucles externos en sus valores mínimos.
- Continuar hacia fuera, llevando a cabo pruebas para el siguiente bucle, con todos los bucles externos en sus valores mínimos y los demás bucles anidados en sus valores típicos.
- Terminar cuando se hayan probado todos los bucles.

### Bucles concatenados

Estos se pueden probar mediante el método definido para los bucles simples mientras todos los bucles sean independientes. Si los bucles no son independientes se recomienda usar el método aplicado a los bucles anidados.

### Bucles no estructurados

Este tipo de bucles, siempre que sea posible, deben ser rediseñados para que sean compatibles con las construcciones de programación estructurada.

#### **1.4.2. Pruebas de caja negra**

En este caso se busca probar:

- Las funciones realizadas por el sistema.
- El cumplimiento de los objetivos del sistema.

- Las reacciones del sistema ante los estímulos exteriores.
- Las transacciones manejadas por el sistema.

Conociendo el funcionamiento del producto, se desarrollan pruebas que aseguren que todas las piezas encajan, que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado adecuadamente.

Las pruebas de caja negra son complementarias a la de caja blanca y se centran en lo que se espera de un módulo, es decir, intentan encontrar casos en que el módulo no se atiene a su especificación (requisitos funcionales). Por ello se denominan pruebas funcionales, y el probador se limita a suministrarle datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el módulo por dentro.

Estas pruebas están especialmente indicadas en aquellos módulos que van a ser interfaz con el usuario y se apoyan en la especificación de requisitos del módulo. Permiten obtener conjuntos de condiciones de entrada que ejecuten todos los requisitos funcionales de un programa.

Lo fundamental en este tipo de pruebas es encontrar el subconjunto de todas las entradas posibles del programa esto es muy complejo, para conseguirlo se siguen diferentes criterios:

- Métodos de prueba basados en grafos.
- Métodos de prueba partición de equivalencia.
- Métodos de prueba de análisis de valores límite.
- Métodos de prueba de la tabla ortogonal.
- Métodos de prueba adivinando el error.

### **Método de prueba Partición de Equivalencia**

Se divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada.

- Se define una clase de equivalencia válida y dos no válidas cuando:
  - Una condición de entrada especifica un rango
  - Una condición de entrada requiere un valor específico.

- Se define una clase de equivalencia válida y una no válida cuando:
  - Una condición de entrada especifica un miembro de un conjunto.
  - Una condición de entrada es lógica.

En el proceso de prueba de liberación al BioSyS se utiliza este método de prueba para reducir el posible conjunto de casos de prueba en uno más pequeño, un conjunto manejable que evalúe bien el software.

### **Método de prueba Análisis de Valores Límite**

Los errores tienden a darse más en los límites del campo de entrada que en el centro. Por ello, se ha desarrollado el análisis de valores límites como técnica de prueba. El análisis de valores límite lleva a una elección de casos de prueba que ejerciten los valores límite.

Esta prueba es una técnica de diseño de casos de prueba que completa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el análisis de valores límite lleva a la elección de casos de prueba en los extremos de la clase. En lugar de centrarse solamente en las condiciones de entrada, el análisis de valores límite obtiene casos de prueba también para el campo de salida.

Condiciones sublímite: Las condiciones límite normales son las más obvias de descubrir. Estas son definidas en la especificación o son evidentes al momento de utilizar el software. Algunos límites, sin embargo, son internos al software, no son necesariamente aparentes al usuario final pero aún así deben ser probadas por el probador. Estas son conocidas como condiciones sublímites o condiciones límite internas. Una condición sublímite común es la tabla de caracteres ASCII, por ejemplo, si se está evaluando una caja de texto que acepta solamente los caracteres AZ y az, se debe incluir los valores en la partición inválida justo «debajo de» y «encima de» esos caracteres de la tabla ASCII.

Debido a que los errores tienden a darse más en los límites del campo de entrada que en el centro, se hizo necesario hacer uso de esta técnica en el proceso de prueba de liberación al BioSyS.

### **1.5. Plan de pruebas.**

El propósito del plan de pruebas es dejar de forma explícita el alcance, el enfoque, los recursos requeridos, el calendario, los responsables y el manejo de riesgos de un proceso de pruebas. [6]

Los objetivos que se persiguen con el Plan de Pruebas son:

- Identificar los elementos de pruebas.
- Describir y recomendar las estrategias de las pruebas a ser empleadas.
- Definir el cronograma de las pruebas.

El plan de pruebas nace en las etapas tempranas del proyecto, donde los requisitos del sistema aportan información crucial para la confección del mismo. El propósito del mismo es explicitar el alcance, enfoque, recursos requeridos, calendario, responsables y manejo de riesgos de un proceso de pruebas.

Para Crear un plan de pruebas debemos valorar:

- El alcance de la aplicación.
- La complejidad de sus procesos.
- Plataformas en las que deberemos probar.
- Conocimientos y formación de quienes ejecutarán las pruebas.
- Normativas legales aplicables.

## **1.6 Herramientas para el desarrollo de las pruebas de software**

### Listas de chequeos:

Se entiende por lista de chequeo a un listado de preguntas, en forma de cuestionario que sirve para verificar el grado de cumplimiento de determinadas reglas establecidas con un fin determinado y deben tenerse presente desde el inicio de un proyecto. Las preguntas sirven como una guía que obliga a quien las contesta a reflexionar sobre el nivel de acatamiento de determinados requisitos. La lista de chequeos enumera una serie de ítems que deberían verificarse uno a uno para asegurarnos de lograr el producto final con un nivel de calidad previamente aceptado.

Estas listas, correctamente diseñadas y utilizadas de manera responsable, sirven para controlar el nivel de calidad de un producto y evitan que contengan errores atribuibles a simples descuidos.

En el proceso de pruebas al producto BioSyS se hará uso de esta herramienta para chequear el documento de Especificación de Requisitos para verificar que se hayan documentado todos los requisitos correctamente y el documento de Especificación de Casos de Uso para revisar que se le haya dado respuesta a todos los requisitos y que las descripciones de casos de usos sean íntegras y precisas. También se hará uso de la lista de chequeos para hacer una evaluación de los atributos de calidad en las pruebas de sistema.

### Casos de pruebas:

Se entiende como caso de prueba al conjunto de condiciones o variables bajo las cuáles se determinará si el requisito de una aplicación es parcial o completamente satisfactorio, tienen como propósito comprobar que todos los requisitos de una aplicación son revisados y especificar una forma de probar el sistema, incluyendo las entradas con las que se ha de probar, los resultados esperados y las condiciones bajo las que ha de probarse. Estos reflejan trazabilidad con los casos de uso, las especificaciones suplementarias de requerimientos y diseño del sistema, garantizando que los procedimientos de pruebas sean compatibles con las necesidades de los clientes.

Lo que caracteriza un escrito formal de caso de prueba es que hay una entrada conocida y una salida esperada, los cuales son formulados antes de que se ejecute la prueba. La entrada conocida debe probar una precondition y la salida esperada debe probar una postcondición.

Los casos de pruebas para las pruebas de caja blanca permiten examinar la estructura interna del programa. Se diseñan casos de prueba para examinar la lógica del programa. Es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar casos de prueba que garanticen que:

- Se ejercitan todos los caminos independientes de cada módulo.
- Se ejercitan todas las decisiones lógicas.
- Se ejecutan todos los bucles.
- Se ejecutan las estructuras de datos internas.

Los casos de pruebas para la prueba de caja negra se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa y pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma adecuada.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

En el proceso de pruebas al producto BioSyS se hará uso de esta herramienta para la realización de las pruebas funcionales con el método de prueba de caja negra.

## Simulador de Conexiones MySQL

El Simulador de Conexiones MySQL es una aplicación implementada en el lenguaje de programación C# que permite simular un número de conexiones concurrentes y peticiones a una base de datos, y trabajar al mismo tiempo sobre una funcionalidad determinada.

Dicha aplicación permite conocer el tiempo que se tomará el sistema para dar respuesta a  $n$  cantidad de usuarios haciendo peticiones sobre una misma consulta en la base de datos (BD); por lo que será un buen medidor de la capacidad que tiene el sistema para soportar determinada carga de conexiones sin llegar a perder eficiencia y rapidez durante su funcionamiento.

Esta herramienta va a ser utilizada en el proceso de pruebas al BioSyS para realizar las pruebas de carga y estrés.

### **1.7 Diseño de casos de prueba**

Cualquier proceso de ingeniería puede ser probado de dos formas:

- Se pueden llevar a cabo pruebas que demuestren que cada función es completamente operativa (Caja negra).
- Se pueden desarrollar pruebas que aseguren que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada (Caja blanca).

Para realizar estas pruebas se emplean los diseños de casos de pruebas. A la hora de realizar los diseños de casos de pruebas se debe tener en cuenta una regla básica: cubrir todas las posibilidades en el menor número de casos de pruebas posibles. Se deben de diseñar pruebas que tengan la mayor probabilidad de encontrar el mayor número de errores con la mínima cantidad de esfuerzo y tiempo posible.

Existen dos técnicas de diseños de casos de pruebas, está la técnica de caja blanca y la técnica de caja negra. Estas técnicas están representadas en la figura 4.

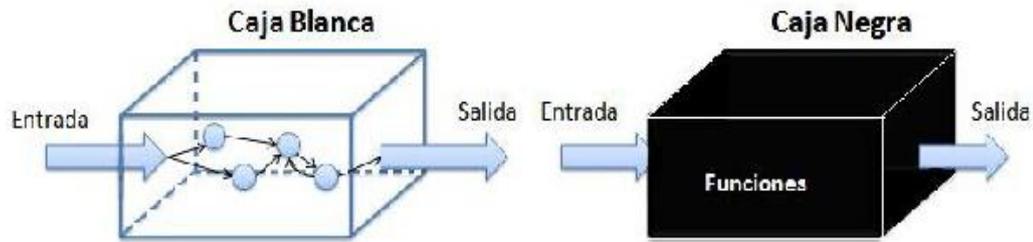


Figura 4. Técnicas de diseño de casos de pruebas.

Existen tres enfoques para el diseño de casos de prueba:

- El enfoque estructural o de caja blanca, en el que conociendo el funcionamiento del producto, se pueden desarrollar pruebas que aseguren que todas las piezas encajan, o sea, que la operación interna se ajusta a las especificaciones y que todos los componentes internos se comprueban de forma adecuada.
- El enfoque funcional o de caja negra, en el que se plantea que conociendo la función específica para la que fue diseñado el producto, se pueden llevarse a cabo pruebas que demuestren que cada función es completamente operativa y al mismo tiempo buscando errores en cada función.

### 1.8 Metodología OpenUP

Es un proceso de desarrollo iterativo del software que es mínimo, completo, y extensible. El proceso es mínimo en que solamente el contenido fundamental es incluido; es completo en que puede ser manifestado como todo el proceso para construir un sistema; extensible en que se puede adaptar según lo necesitado. Como metodología de desarrollo es conducida por el principio de colaboración para alinear intereses y para compartir su comprensión. Esta metodología será la que se utilizará en nuestro trabajo por las ventajas que ofrece y se describen a continuación.

Esta metodología tiene como principios:

- Colaborar para sincronizar intereses y compartir conocimiento.
- Equilibrar las prioridades para maximizar el beneficio obtenido por los interesados en el proyecto.
- Centrarse en la arquitectura de forma temprana para minimizar el riesgo y organizar el desarrollo.
- Desarrollo evolutivo para obtener retroalimentación y mejoramiento continuo.

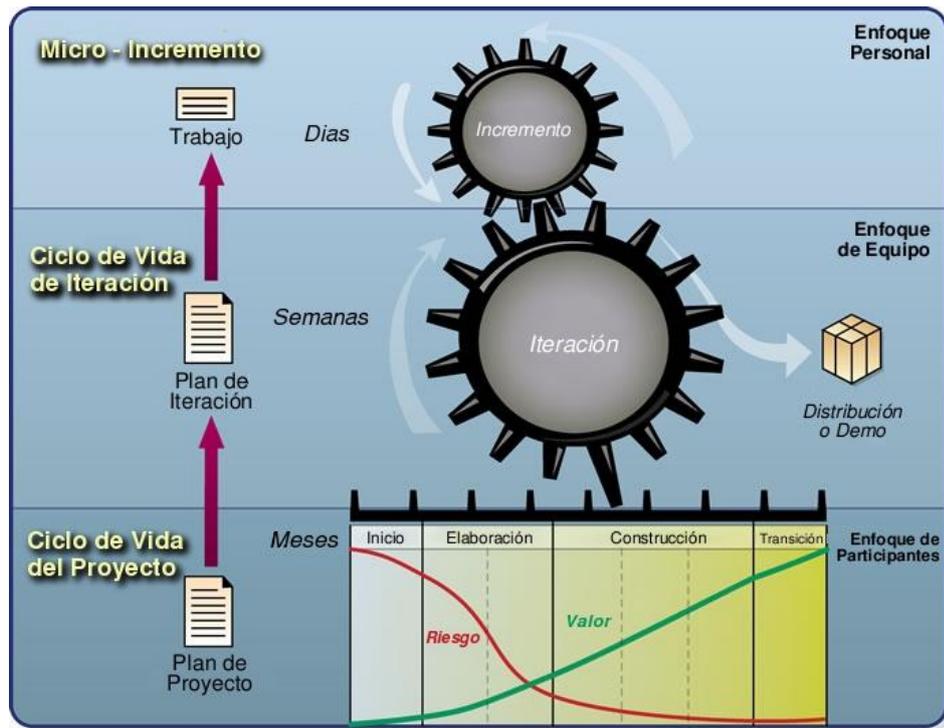


Figura 5. Capas de OpenUP

A nivel personal, los microincrementos, representan los resultados obtenidos en pocas horas o pocos días de trabajo de tal forma que el progreso puede ser visualizado efectivamente cada día. Los integrantes del equipo de desarrollo de forma abierta comparten su progreso diario el cual incrementa la visibilidad en el trabajo, la confianza y el trabajo en equipo.

El proyecto en general se divide en iteraciones, las cuales son planificadas en un intervalo definido de tiempo que no superan las pocas semanas. El OpenUP estructura el ciclo de vida de un proyecto en cuatro fases: concepción, elaboración, construcción y transición. El ciclo de vida del proyecto provee a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos. (Ver figura 5)

### 1.8.1 Rol de probador

El probador en la metodología de desarrollo OpenUP es responsable de las actividades principales del esfuerzo de las pruebas. Estas actividades incluyen identificar, definir, implementar y dirigir las pruebas

necesarias, como también verificar los resultados de las pruebas y analizar los resultados y reparar los errores de ejecución. Ver figura 6.



Figura 6. Representación de las actividades y artefactos relacionados con el probador.

La persona que vaya a asumir este rol debe tener conocimiento de tendencias de pruebas y técnicas, habilidades para el diagnóstico y solución de problemas, debe tener además conocimiento del sistema o aplicación que está siendo probada y poseer algunos conocimientos de redes y arquitectura de sistemas.

Este rol es principalmente responsable por las siguientes tareas:

- Identificar las pruebas que se requiere llevar a cabo.
- Identificar el acercamiento más apropiado para implementar una prueba dada.
- Implementar pruebas individuales.
- Preparar y ejecutar las pruebas.
- Registrar resultados y verificar que las pruebas hayan sido ejecutadas.
- Análisis y recuperación de errores de ejecución.
- Comunicar los resultados de las pruebas al equipo.

## 1.9 Conclusiones del capítulo

Luego de haber realizado un estudio detallado acerca de la calidad y las pruebas de software, podemos concluir diciendo que para lograr un producto de software de alta calidad y confiabilidad, necesitamos que el producto esté libre de errores, por lo que se ha decidido aplicar las siguientes pruebas por ser las más adecuadas para este software: prueba modulares, de integración ascendente,

de carga y estrés , además se implementa el método de prueba de Caja Negra y dentro de este se utilizan las técnicas de partición de equivalencia y valor límite. Para la realización de estas pruebas se utilizan herramientas tales como las lista de chequeo, casos de prueba y para las pruebas de sistema se utiliza el Simulador de conexiones MySQL. Todo el proceso es guiado por la metodología de desarrollo Open Up.

## Capítulo **2**

### **Diseño y aplicación de las pruebas de liberación a BioSyS**

En el presente capítulo se expone el diseño y la aplicación de las pruebas de liberación al software alas BioSyS desarrollado por la Facultad 6 a petición del Centro de Inmunología Molecular (CIM). En el mismo se hace una descripción del sistema a probar, se determina la estrategia de prueba a seguir a lo largo del proceso de prueba, así como las técnicas, métodos y herramientas a utilizar.

#### **2.1 Características del sistema a probar**

Las características a probar pueden cambiar, el grado de complejidad y el nivel detalle están en dependencia del tipo de aplicación a probar así como los objetivos que se quiera evaluar. Existen particularidades que se deben tener presente permanentemente en el momento de aplicar las pruebas de software. Por ejemplo, verificar si la aplicación cumple con todas las funcionalidades requeridas, si cuando se ejecuta el programa corre debidamente y sin errores, si resulta sencillo a la hora de su utilización y si el usuario logra comprender como funciona el sistema fácilmente. Además se tiene que verificar que el software desarrollado sea confiable, flexible, este completo y con una documentación bien detallada. Igualmente se pueden medir elementos del código fuente para comprobar en que medida el código se encuentra bien estructurado y pueda ser reutilizable, ya que son características que aseguran la perdurabilidad del software.

#### **BioSyS: Software de Simulación y Análisis de Sistemas Biológicos. Características del sistema**

BioSyS es un producto que permite crear modelos de sistemas biológicos que puedan ser descritos mediante Sistemas de Ecuaciones Diferenciales (SED), brinda a los usuarios la posibilidad de resolver los sistemas de ecuaciones diferenciales para conjuntos de condiciones iniciales y parámetros previamente especificados lo que es la base para la realización de estudios cuantitativos de los sistemas biológicos en cuestión. Muestra al usuario las gráficas de las dinámicas de las poblaciones de una simulación que representa el resultado de los análisis que va haciendo. A partir de las simulaciones almacenadas en la Base de Datos (BD) se puede realizar el análisis de clúster, teniendo en cuenta el estado final, es decir, agrupa el volumen de simulaciones de acuerdo a los estados finales a los que tiende el sistema. Le permite al usuario clasificar de acuerdo a sus criterios (manualmente) las simulaciones realizadas, que se muestran aleatoriamente. También posibilita la creación de modelos que pueden ser utilizados para clasificar nuevas simulaciones y para observar las relaciones

lógicas que llevan a un estado o el otro y finalmente toda la información generada en los estudios realizados puede ser almacenada en una BD, facilitando que no se tenga que consumir tiempo innecesariamente haciendo estudios que ya se han realizado.

El proyecto está organizado en cuatro módulos:

El módulo del **Editor de Ecuaciones** cuenta con seis casos de usos:

- Gestionar Edición de Sistemas Ecuaciones Diferenciales (SED)
- Gestionar Biblioteca::
- Analizar Dimensionalidad SED:
- Exportar SED
- Cargar modelo matemático
- Salvar modelo matemático

Este módulo permite conocer si las ecuaciones escritas cuentan con términos cuyas dimensiones no son correctas, exportar sistema de ecuaciones diferenciales a diferentes formatos, la creación y modificación del SED en estudio, ademasen el se pueden encontrar funciones matemáticas y ecuaciones previamente creadas y cuenta con la posibilidad de guardar nuevas funciones y ecuaciones para su posterior utilización.

El módulo de **Simulación** cuenta con cuatro casos de usos:

- Editar Preferencias:
- Autenticar en la GRID
- Chequear estado de las ejecuciones
- Gestionar Simulación

Este módulo brinda la oportunidad de definir las opciones para realizar la simulación aunque primeramente para ello verifica que el usuario tiene los privilegios necesarios para acceder a la base de datos, así como posibilita que el usuario edite las preferencias que le brinda la aplicación de acuerdo a sus necesidades. Permite el estudio del comportamiento común para la realización de simulaciones locales y distribuidas.

El módulo de **Análisis** cuenta con seis casos de usos:

- Mostrar Dinámicas Aleatorias
- Realizar Clusters:

- Realizar Análisis de las Clasificaciones:
- Realizar análisis por Reglas
- Editar Reglas
- Realizar análisis de bifurcaciones.

En este módulo se puede analizar las salidas gráficas de las simulaciones donde se representan las Series Temporales. Permite el análisis de las simulaciones que se encuentran en un fichero, agrupándolas por tres tipos de algoritmos de clusters, así como clasificar simulaciones a partir de modelos generados de simulaciones previamente clasificadas y definir reglas y grupos de reglas.

El módulo de **Funcionalidades Comunes** cuenta con cuatro casos de uso:

- Gestionar conexión.
- Gestionar Sistema Biológico.
- Gestionar Modelo Matemático.
- Gestionar copias de valores de las tablas.

En este módulo se recogen todas las funcionalidades comunes entre todos los módulos, como es por ejemplo la funcionalidad de copiar valores de una tabla hacia otra, también está la de gestionar conexión, esta funcionalidad es común entre todos los módulos, pues para realizar alguna funcionalidad el usuario debe de estar autenticado en el sistema.

## **2.2 Requerimientos necesarios para realizar las pruebas**

Para la realización de las pruebas se necesitó de algunos recursos dentro de los cuales se pueden mencionar los siguientes:

Como mínimo 2 PC que como **requerimiento de software** deben tener:

- La máquina virtual de java versión 1,5 o superior.
- El sistema operativo a instalar en la PC donde se ejecuta el sistema es sin restricciones, por ser un software multiplataforma.
- El Sistema Operativo a instalar en las PC clientes del T-arenal puede ser cualquier distribución de Linux.
- Asistente matemático MatLab.
- Cliente del T-arenal instalado en cada una de las PCs que se utilizarán para la simulación distribuida.

**Especificaciones de Hardware:**

- Procesadores Pentium IV o superior.
- 256 MB de RAM o superior.
- 40 GB de disco duro o superior.

**2.3 Estrategia de pruebas.**

Una **estrategia de pruebas** es un conjunto de acciones determinadas de forma anticipada que se llevan a cabo para lograr un determinado objetivo, es el proceso mediante el cual se espera alcanzar un estado futuro.

Una estrategia integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que dan como resultado una correcta construcción del software.

La estrategia que se llevó a cabo durante todo este proceso de pruebas está compuesta por cuatro pasos fundamentales:

**Paso 1:**

**Revisión de la documentación** (Manual de usuario, Manual de instalación, Especificación de requisitos, Especificación de Casos de Uso): En el caso particular de las especificaciones de Requisitos y Casos de Uso se utilizaron las listas de chequeo propuestas por la dirección de calidad de software de la universidad (Anexo 1 y 2)

**Paso 2:**

**Pruebas a la aplicación:** Se llevó a cabo un proceso de prueba exhaustivo donde se realizaron en la pruebas modulares para verificar que cada elemento encaja de forma adecuada alcanzando así la funcionalidad y el rendimiento total; y a su vez verificar que la aplicación satisface todos los requisitos funcionales, de comportamiento y de rendimiento (Anexo 3) donde prevalecieron los diseños de casos de pruebas de caja negra, mediante la utilización de la técnica de partición equivalente y análisis de valores límites, la cual igualmente se utilizó en las pruebas de integración para asegurarnos de que los cuatro módulos de la aplicación lograban un correcto funcionamiento utilizando las herramientas que se proporcionan entre sí, donde se hizo uso de los diseños de casos de pruebas que vinculan funciones entre los diferentes módulos y de esta forma detectar las fallas de interacción entre las distintas clases que componen al sistema.

**Paso 3:**

**Pruebas de sistema:** La prueba del sistema está constituida por una serie de pruebas diferentes cuyo propósito primordial es ejercitar profundamente el sistema. Aunque cada prueba tiene un propósito diferente, todas trabajan para verificar que se han integrado adecuadamente todos los elementos del sistema y que realizan las funciones apropiadas. En este trabajo se aplicaron pruebas de carga y estrés, así como la aplicación de una lista de chequeo propuesta por el grupo de calidad de la facultad para valorar el estado de algunos atributos de calidad presentes en la aplicación.

Esta etapa de pruebas se va a dividir en tres etapas.

- Pruebas de forma manual con estudiantes reales.
- Pruebas con el Simulador de Conexiones MySQL.
- Pruebas de los atributos de calidad, con una lista de chequeos.

En la **primera etapa** para guiar a los estudiantes en la realización de la prueba se confeccionó un manual (Anexo 4) donde se detallaron cada una de las funcionalidades a probar. En este manual se indicaron los datos reales que deben ser insertados en cada uno de los casos y se explicó paso por paso cómo se debía ir ejecutando la prueba.

Con esta prueba se comprobó el funcionamiento de las operaciones más significativas, se considerando como las más significativas aquellas que serán más utilizadas por el cliente y las más complicadas.

En la **segunda etapa** se hizo uso de la herramienta Simulador de Conexiones MySQL. Con esta herramienta se simuló un número **n** de conexiones concurrentes y peticiones a la base de datos, y se trabajó al mismo tiempo sobre una funcionalidad determinada con el objetivo de comprobar la capacidad que tiene el sistema para soportar determinada carga de conexiones sin llegar a perder eficiencia y rapidez durante su funcionamiento.

Para utilizar el Simulador de Conexiones MySQL se debe introducir primeramente la cadena de conexión referente a la base de datos, donde se especifica la dirección de ésta, el puerto de conexión y datos para realizar la autenticación, la consulta que se desee ejecutar, la cantidad de conexiones que se van a realizar y por último la cantidad de peticiones de la consulta por cada una de estas conexiones a la base de datos.

Con esta prueba se comprobó el funcionamiento de las operaciones más significativas, considerando como las más significativas aquellas que serán más utilizadas por el cliente y las más complicadas. Las funcionalidades probadas responden a los siguientes CU.

**Módulo de Funcionalidades comunes:**

- Gestionar conexión

**Módulo Simulador:**

- Gestionar Simulación.

**Módulo Análisis.**

- Realizar Clusters.
- Realizar Análisis de Clasificación.
- Realizar Análisis de Bifurcaciones.
- Mostrar Dinámicas Aleatorias

En la **tercera etapa** se hizo una evaluación de los atributos de calidad haciendo uso de una lista de chequeo (Anexo 5) en la que vienen registradas una serie de interrogantes que ayudan a evaluar cada uno de estos atributos.

## **2.4 Entorno de prueba**

Para la aplicación de las pruebas se tuvo en cuentas los requerimientos de hardware y software necesarios para un correcto funcionamiento de la aplicación. Estos fueron:

Hardware:

- Una PC con procesador Pentium IV.
- 1 GB de memoria RAM.
- 160 GB de Disco Duro.

Software:

- La máquina virtual de java versión 1.6.
- Una PC con Linux (Ubuntu).
- Asistente matemático MatLab.

## 2.5 Cronograma de prueba

Durante la realización del plan de pruebas se creó un cronograma de pruebas que sirvió de guía para la organización del trabajo y lograr un mejor aprovechamiento del tiempo. Este sufrió algunos cambios en el transcurso de las pruebas pero fue actualizado y ajustado a las nuevas condiciones que fueron surgiendo.

Actividad	Fecha de inicio	Fecha de culminación	Responsable
Aceptación y firma del plan de pruebas	28/02/2009	28/02/2009	Yulieska, Yuneisi y Yunet
<b>Manual de usuario</b>			
Revisión primera iteración	15/01/2009	25/01/2009	Yuneisi
Respuesta a las NC de la primera iteración	25/01/2009	02/02/2009	Yuneisi
Revisión segunda iteración	03/03/2009	06/03/2009	Yuneisi
Respuesta a las NC de la segunda iteración	06/03/2009	12/03/2009	Yuneisi
Revisión tercera iteración	12/03/2009	20/03/2009	Yuneisi
Respuesta a las NC de la tercera iteración	21/03/2009	30/03/2009	Yuneisi
<b>Manual de instalación</b>			
Revisión primera iteración	19/01/2009	23/01/2009	Yulieska
Respuesta a las NC de la primera iteración	23/01/2009	20/03/2009	Yulieska
Revisión segunda iteración	4/03/2009	10/03/2009	Yulieska
Respuesta a las NC de la segunda iteración	10/03/2009	16/03/2009	Yulieska
Revisión tercera iteración	17/03/2009	20/03/2009	Yulieska
<b>Documento de especificación de requisitos</b>			
Revisión primera iteración	12/03/2009	16/03/2009	Yulieska
Respuesta a las NC de la primera iteración	17/03/2009	29/03/2009	Yulieska

iteración			
Revisión segunda iteración	4/04/2009	10/04/2009	Yulieska
Respuesta a las NC de la segunda iteración	11/03/2009	20/04/2009	Yulieska
Revisión tercera iteración	20/04/2009	27/04/2009	Yulieska
<b>Documento de descripción de las CU</b>			
Revisión primera iteración	23/03/2009	31/03/2009	Yuneisi
Respuesta a las NC de la primera iteración	01/03/2009	10/04/2009	Yuneisi
Revisión segunda iteración	12/04/2009	16/04/2009	Yuneisi
Respuesta a las NC de la segunda iteración	17/04/2009	23/04/2009	Yuneisi
Revisión tercera iteración	24/04/2009	27/04/2009	Yuneisi
Respuesta a las NC de la tercera iteración	27/04/2009	30/04/2009	Yuneisi
<b>Aplicación</b>			
Confección de los DCP módulo Análisis	06/03/2009	12/03/2009	Yuneisi y Yulieska
Confección de los DCP módulo Simulación	28/03/2009	5/04/2009	Yulieska
Confección de los DCP módulo Editor	29/03/2009	06/04/2009	Yuneisi
Revisión de la aplicación primera iteración	06/04/2009	10/04/2009	Yuneisi y Yulieska
Respuesta a las NC de la primera iteración	11/04/2009	17/04/2009	Yulieska y Yuneisi
Revisión de la aplicación segunda iteración	17/04/2009	21/04/2009	Yulieska y Yuneisi
Respuesta a las NC de la segunda iteración	21/04/2009	27/04/2009	Yulieska y Yuneisi
Revisión de la aplicación tercera iteración	28/04/2009	02/05/2009	Yulieska y Yuneisi
Respuesta a las NC de la tercera iteración	02/05/2009	07/05/2009	Yulieska y Yuneisi

Ejecución de las pruebas de integración	02/05/2009	07/05/2009	Yulieska y Yuneisi
Ejecución de las pruebas de carga y estrés	07/05/2009	07/05/2009	Yulieska y Yuneisi

## 2.6 Casos de Prueba

El objetivo de un Caso de Prueba es especificar una forma de probar el software, donde se deben de incluir los datos de entrada y la respuesta que se espera recibir del sistema. Estos son de gran ayuda para validar y verificar pues con un buen caso de prueba se asegura encontrar la mayor cantidad de defectos posibles dándole un mayor grado de fiabilidad y confiabilidad al producto. Para la realización de estos diseños de casos de pruebas se utilizó la plantilla propuesta por el grupo de calidad de la universidad de la cual se hace una descripción en el anexo 12.

### 2.6.1 Pruebas de Caja Negra

Para la realización de los diseños de casos de pruebas se utilizó el método de pruebas de Caja Negra específicamente la técnica de partición equivalente y análisis de valores límites. Se probó la efectividad en las funcionalidades y validez de las entradas de datos de la interfaz en cada módulo.

Otro punto que debemos de tener en cuenta en la prueba de Caja Negra es comprobar la existencia y funcionalidad de todos los requerimientos funcionales definidos (Anexo 3), que debe de cumplir el sistema según la petición del cliente a demás de verificar que la interfaz se encuentre libre de errores.

El diseño de estas pruebas se realizó basándose en las Especificaciones de Casos de Uso que se encuentran en el expediente del proyecto.

A continuación se muestran algunos de los DCP más significativos por módulos:

#### **Módulo de Simulación**

DCP del CU Gestionar Simulación

#### **Descripción General**

Resolver un modelo matemático que describa a un sistema biológico.

#### **Condiciones de Ejecución:**

Debe haber autenticado en el sistema, introducido los valores de las condiciones iniciales.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Definir parámetros necesarios	EC 1.1: Insertar los valores correctos para la simulación.	Insertar los valores necesarios para la simulación satisfactoriamente.	<ul style="list-style-type: none"> <li>• El Investigador selecciona el modelo que quiere simular y toma la opción para simular, luego define los valores de los parámetros:</li> <li>• Parameters/Initial Conditions, Initial Time, Final Time, Relative Tolerance, Absolute Tolerance, Numerical Method, Integration step, Type of simulation.</li> <li>• Selecciona el tipo de simulación.</li> <li>• Se comienza a simular el modelo matemático cargado.</li> </ul>
	EC 1.2: Insertar Valores no válidos para la Simulación	El Sistema no puede simular el modelo porque no cuenta con los datos correctos.	<ul style="list-style-type: none"> <li>• El Investigador selecciona el modelo que quiere simular y toma la opción para simular, luego define los valores de los parámetros.</li> <li>• El sistema lanza un mensaje indicando que hay campos llenados con valores incorrectos.</li> </ul>
	EC 1.3: Al Insertar se dejó campos en blancos.	El sistema no puede simular el modelo porque no tiene todos los datos que necesita.	<ul style="list-style-type: none"> <li>• El Investigador selecciona el modelo que quiere simular y toma la opción para simular, luego define los valores de los parámetros.</li> <li>• El sistema lanza un mensaje indicando que hay campos llenados con valores incorrectos.</li> </ul>

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 2: Realizar Simulación Local	EC 2.1: Se simula el modelo matemático.	La simulación ocurre normalmente a partir del modelo matemático cargado previamente.	<ul style="list-style-type: none"> <li>• Se visualiza el progreso de simulación y activa opciones a través de dos botones “Stop” y “Pause”.</li> <li>• No se selecciona ninguna de las opciones y termina el proceso de simulación satisfactoriamente mostrando al finalizar los resultados.</li> </ul>
	EC 2.2	Se Pausa el Proceso de simulación	Se detiene la simulación al presionar “Pause” y se reanuda al presionar “Continue”.
	EC 2.3: Se Detiene el Proceso de simulación	Se detiene el proceso de Simulación.	<ul style="list-style-type: none"> <li>• Se visualiza el progreso de simulación y activa opciones a través de dos botones “Stop” y “Pause”.</li> <li>• El usuario selecciona la opción de “Stop” cancelando así el proceso de Simulación..</li> </ul>

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC3: Realizar Simulación Distribuida	EC:3.1 : Se simula el modelo matemático	Se simula el modelo matemático de forma distribuida mediante la utilización de la GRID.	<ul style="list-style-type: none"> <li>• El usuario se autentica en la Grid.</li> <li>• Envía los datos para la ejecución de las simulaciones al servidor de la Grid, se comienza a simular el modelo matemático cargado.</li> <li>• Para finalizar se almacenan los resultados de las simulaciones en la Base de Datos cargado.</li> <li>• Para finalizar se almacenan los resultados de las simulaciones en la Base de Datos.</li> </ul>

DCP del CU Chequear estado de la Ejecución

### Descripción General

El investigador podrá chequear el avance de la realización de las simulaciones en la Grid.

### Condiciones de Ejecución:

El investigador deberá estar autenticado en la Grid.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
----------------------	--------------------------	---------------------------------	---------------

SC 1: Chequear ejecuciones en la GRID	EC 1.1: Ver estado de las ejecuciones.	Se muestra como se va desarrollando las ejecuciones.	<ul style="list-style-type: none"> <li>• Se selecciona en el menú View la opción Executions.</li> <li>• Se muestra las ejecuciones existentes.</li> <li>• Si desea ver una ejecución, para ello da doble clic sobre ella y se muestra una interfaz con el estado de la ejecución seleccionada.</li> </ul>
	EC 1.2: Detener las ejecuciones.	Detener una ejecución que se encuentra corriendo.	Estando en la interfaz donde se muestra el estado de la ejecución se selecciona la que se desea detener y presiona el botón "Stop".
	EC 1.3: Ver soluciones	Muestra las soluciones que ha dado el sistema.	En el menú View se selecciona la opción Solutions y se muestra una interfaz con las soluciones.
	EC 1.4: Descargar soluciones	Descarga las soluciones que generó el sistema.	<ul style="list-style-type: none"> <li>• En la interfaz que muestra un conjunto de soluciones selecciona una solución que desea descargar y presiona el botón "Download".</li> <li>• Se muestra un navegador de archivo para que el investigador seleccione dónde descargar.</li> </ul>
	EC 1.5: Eliminar soluciones.	Elimina una de las soluciones generadas por el sistema.	<ul style="list-style-type: none"> <li>• En la interfaz que muestra un conjunto de soluciones selecciona una solución que desea eliminar y presiona el botón "Delete". Se muestra un mensaje de confirmación. Se presiona "Yes" se elimina si presiona "No" se cancela</li> </ul>

			eliminación.
--	--	--	--------------

**Módulo de Análisis**

DCP del CU Mostrar Dinámicas Aleatorias

**Descripción General**

Permite analizar las salidas gráficas de las simulaciones donde se representan las Series Temporales.

**Condiciones de Ejecución:**

Deben existir simulaciones del modelo seleccionado.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Mostrar Dinámicas Aleatorias	EC 1.1: Mostrar Dinámicas Aleatorias	Permite analizar las dinámicas correspondientes a los resultados obtenidos de un grupo de simulaciones	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar la dinámica de población dando clic derecho sobre el modelo que desea analizar y seleccionando la opción Analysis/Show dynamic random.</li> <li>• El sistema verifica que existen simulaciones creadas.</li> <li>• El sistema ejecuta el algoritmo que crea los resultados.</li> <li>• El sistema envía estos resultados a un algoritmo que se encarga de generar la gráfica con la</li> </ul>

			<p>herramienta JfreeChart.</p> <ul style="list-style-type: none"> <li>• El sistema muestra la gráfica con los resultados de las simulaciones, donde se observa el comportamiento de las Series Temporales por Población.</li> </ul>
	<p>EC 1.2: Mostrar Dinámicas Aleatorias sin simulaciones existentes.</p>	<p>No permite analizar las dinámicas porque no se han realizado simulaciones.</p>	<ul style="list-style-type: none"> <li>• El sistema desactiva la opción para mostrar la dinámica de población que se despliega al dar clic derecho sobre el modelo que se desea analizar y seleccionar la opción Analysis/Show dynamic random.</li> </ul>

DCP del CU Realizar Análisis de la Clasificación

**Descripción General**

Permite clasificar las simulaciones que se encuentran en la base de datos usando un modelo ya clasificado

**Condiciones de Ejecución:**

El modelo debe tener simulaciones

<b>Nombre de la sección</b>	<b>Escenarios de la sección</b>	<b>Descripción de la funcionalidad</b>	<b>Flujo Central</b>
<p>SC 1: Realizar Clasificaciones.</p>	<p>EC 1.1: Realizar Clasificaciones.</p>	<p>Permite clasificar simulaciones a partir de modelos generados de simulaciones</p>	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar clasificaciones dando clic derecho sobre el modelo que desea analizar y</li> </ul>

		previamente clasificadas.	<p>seleccionando la opción Analysis/Classify</p> <ul style="list-style-type: none"> <li>• El sistema muestra la interfaz de clasificaciones para que usuario introduzca los datos.</li> <li>• El usuario introduce los datos y presiona el botón de ejecución que se encuentra en la parte superior izquierda de la interfaz. (Ver anexo 1)</li> <li>• El sistema muestra el resultado de las simulaciones clasificadas en una gráfica de clusters.</li> </ul>
	EC 1.2: Realizar Clasificaciones sin simulaciones existentes.	No se realiza la clasificación porque no existen simulaciones en ese modelo.	<ul style="list-style-type: none"> <li>• El sistema desactiva la opción para realizar clasificaciones que se despliega al dar clic derecho sobre el modelo que se desea analizar y seleccionar la opción Analysis/Classify</li> </ul>
	EC 1.3: Realizar Clasificaciones con valores inválidos.	No se realiza la clasificación porque los valores entrados son incorrectos	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar clasificaciones dando clic derecho sobre el modelo que desea analizar y seleccionando la opción Analysis/Classify</li> <li>• El sistema muestra la interfaz de clasificaciones para que usuario introduzca los datos.</li> <li>• El usuario introduce los datos y</li> </ul>

			<p>presiona el botón de ejecución que se encuentra en la parte superior izquierda de la interfaz.(Ver anexo 1)</p> <ul style="list-style-type: none"> <li>• El sistema verifica que los campos son inválidos y muestra un mensaje solicitando que se entren los datos correctamente.</li> </ul>
	<p>EC 1.4: Realizar Clasificaciones con campos vacíos.</p>	<p>No se realiza la clasificación porque no fueron llenados todos los campos.</p>	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar clasificaciones dando clic derecho sobre el modelo que desea analizar y seleccionando la opción Analysis/Classify</li> <li>• El sistema muestra la interfaz de clasificaciones para que usuario introduzca los datos.</li> <li>• El usuario introduce los datos y presiona el botón de ejecución que se encuentra en la parte superior izquierda de la interfaz.(Ver anexo 1)</li> <li>• El sistema verifica que hay campos vacíos y muestra un mensaje solicitando que se llenen todos los campos.</li> </ul>
	<p>EC 1.5: Realizar Clasificaciones sin cargar el archivo arff</p>	<p>No se realiza la clasificación porque no se ha cargado el</p>	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar clasificaciones dando clic derecho sobre el modelo que</li> </ul>

		resultado del análisis por cluster	<p>desea analizar y seleccionando la opción Analysis/Classify</p> <ul style="list-style-type: none"> <li>• El sistema muestra la interfaz de clasificaciones para que usuario introduzca los datos.</li> <li>• El usuario inserta los datos dejando sin seleccionar el archivo arff</li> <li>• EL sistema no activa el botón de ejecución que se encuentra en la parte superior de la interfaz.(Ver anexo 1)</li> <li>•</li> </ul>
SC 2: Mostrar árbol de reglas	EC 2.1: Mostrar árbol de reglas	El sistema visualiza el árbol de reglas correspondiente al modelo cargado, permitiendo salvarlo como una imagen.	<ul style="list-style-type: none"> <li>• El investigador selecciona el archivo arff perteneciente a el modelo seleccionado y presiona el botón Tree View.</li> <li>• El sistema visualiza el árbol de reglas correspondiente al modelo cargado, permitiendo salvarlo como una imagen.</li> </ul>
	EC 2.2: Mostrar árbol de reglas sin cargar fichero.	No se muestra el árbol de reglas correspondiente porque no hay ningún fichero cargado.	<ul style="list-style-type: none"> <li>• El sistema mantiene el botón Tree View desactivado.</li> </ul>
	EC 2.3: Mostrar árbol de reglas con fichero incorrecto.	No se muestra el árbol de reglas correspondiente al modelo porque el	<ul style="list-style-type: none"> <li>• El usuario selecciona el archivo arff y presiona el botón Tree View.</li> </ul>

		archivo seleccionado es incorrecto.	<ul style="list-style-type: none"> <li>• El sistema verifica que el fichero no pertenece al modelo en cuestión y muestra un mensaje informando que el modelo es incorrecto.</li> </ul>
--	--	-------------------------------------	--

DCP del CU Realizar Análisis por Reglas

**Descripción General**

Permite realizar análisis de un conjunto de simulaciones variando dos parámetros con reglas lógicas definidas por el investigador.

**Condiciones de Ejecución:**

Debe existir un modelo matemático.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Análisis por reglas	EC 1.1: Análisis por reglas.	Permite realizar análisis de un conjunto de simulaciones variando dos parámetros con reglas lógicas definidas por el investigador.	<ul style="list-style-type: none"> <li>• El investigador presiona el clic derecho sobre el modelo matemático y selecciona la opción Analysis/Analysis by Rules.</li> <li>• El investigador, define los valores necesarios para realizar el análisis y especifica las reglas para el análisis y presiona el botón Run.</li> <li>• El sistema verifica la validez de los datos y comienza a hacer el análisis, mostrando el progreso y la grafica de regiones. Le da la posibilidad al investigador de pausar o detener el análisis.</li> </ul>

	<p>EC 1.2: Análisis por reglas con valores incorrectos.</p>	<p>No se realiza el análisis por reglas porque los valores entrados por el investigador son incorrectos.</p>	<ul style="list-style-type: none"> <li>• El investigador presiona el clic derecho sobre el modelo matemático y selecciona la opción Analysis/Analysis by Rules.</li> <li>• El investigador, define los valores necesarios para realizar el análisis y especifica las reglas para el análisis y presiona el botón Run.</li> <li>• El sistema verifica que los datos son incorrectos y muestra un mensaje de error.</li> </ul>
	<p>EC 1.3: Análisis por reglas con campos vacíos</p>	<p>No se realiza el análisis por reglas porque no fueron entrados todos los valores.</p>	<ul style="list-style-type: none"> <li>• El investigador presiona el clic derecho sobre el modelo matemático y selecciona la opción Analysis/Analysis by Rules.</li> <li>• El investigador, define los valores necesarios para realizar el análisis y especifica las reglas para el análisis y presiona el botón Run.</li> <li>• El sistema verifica faltan campos por llenar y muestra un mensaje de error.</li> </ul>
	<p>EC 1.4: Análisis por reglas sin reglas definidas.</p>	<p>No se realiza el análisis por reglas porque el investigador no ha definido las reglas.</p>	<ul style="list-style-type: none"> <li>• El investigador presiona el clic derecho sobre el modelo matemático y selecciona la opción Analysis/Analysis by Rules.</li> <li>• El investigador, define los</li> </ul>

			<p>valores necesarios para realizar el análisis y presiona el botón Run.</p> <ul style="list-style-type: none"> <li>• El sistema verifica que no se ha editado ninguna regla y muestra un mensaje informando que se deben definir las reglas lógicas.</li> </ul>
SC 2: Cargar Modelo	EC 2.1: Cargar Modelo	El sistema carga el fichero seleccionado, y genera el modelo.	<ul style="list-style-type: none"> <li>• El investigador presiona el botón "Load Model" para cargar el modelo.</li> <li>• El sistema muestra un navegador de archivos y el investigador selecciona el fichero de las simulaciones ya clasificadas.</li> <li>• El sistema carga el fichero seleccionado y genera el modelo.</li> </ul>
	EC 2.2: Cargar Modelo con fichero incorrecto.	No se carga el modelo porque el archivo seleccionado es incorrecto.	<ul style="list-style-type: none"> <li>• El investigador presiona el botón "Load Model" para cargar el modelo.</li> <li>• El sistema muestra un navegador de archivos y el investigador selecciona el fichero de las simulaciones ya clasificadas.</li> <li>• El sistema verifica que el fichero seleccionado es incorrecto y muestra un mensaje de error.</li> </ul>

SC 3: Mostrar árbol de reglas.	EC 3.1: Mostrar árbol de reglas	Se muestra el árbol de reglas del modelo cargado.	<ul style="list-style-type: none"> <li>• El investigador presiona el botón "Load Model" para cargar el modelo.</li> <li>• El sistema muestra un navegador de archivos y el investigador selecciona el fichero de las simulaciones ya clasificadas.</li> <li>• El sistema carga el fichero seleccionado y genera el modelo.</li> <li>• El investigador presiona el botón "Tree View" para visualizar el árbol de reglas.</li> <li>• El sistema muestra el árbol de las reglas inferidas.</li> </ul>
	EC 3.2: Mostrar árbol de reglas sin cargar fichero	No se muestra el árbol de reglas porque no hay fichero cargado.	<ul style="list-style-type: none"> <li>• El sistema no activa el botón Tree View.</li> </ul>

DCP del CU Realizar Análisis de la Clasificación

**Descripción General**

Permite realizar análisis de las simulaciones agrupándolas por tres tipos de algoritmos de clusters.

**Condiciones de Ejecución:**

El modelo debe tener simulaciones existentes.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Realizar Clusters.	EC 1.1: Realizar Clusters.	Permite realizar el análisis de las	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar</li> </ul>

		<p>simulaciones agrupándolas por tres tipos de algoritmos de clusters.</p>	<p>clusters dando clic derecho sobre el modelo que desea analizar y seleccionando la opción Analysis/Analysis of Clusters</p> <ul style="list-style-type: none"><li>• El sistema verifica que existen simulaciones creadas y muestra la interfaz para realizar clusters.</li><li>• El investigador llena los campos y presiona el botón verde de ejecución que se encuentra en la parte superior de la interfaz.</li><li>• El sistema verifica que los campos estén correctos y genera las instancias con los resultados de las simulaciones en el tiempo dado.</li><li>• El sistema muestra la interfaz de clusters del weka.</li><li>• El usuario selecciona el algoritmo de cluster a usar y procede a realizar los clusters.</li><li>• El sistema muestra una gráfica con las simulaciones agrupadas por clusters según el algoritmo seleccionado</li></ul>
--	--	--	---

	<p>EC 1.2: Realizar Clusters sin simulaciones existentes.</p>	<p>No se realiza el análisis de clusters porque no existen simulaciones en ese modelo.</p>	<ul style="list-style-type: none"> <li>• El sistema desactiva la funcionalidad para realizar clusters que se despliega al dar clic derecho sobre el modelo que desea analizar y seleccionar la opción Analysis/Analysis of Clusters</li> </ul>
	<p>EC 1.3: Realizar Clusters con campos inválidos.</p>	<p>No se realiza el análisis de clusters porque los campos especificados no son válidos.</p>	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar clusters dando clic derecho sobre el modelo que desea analizar y seleccionando la opción Analysis/Analysis of Clusters</li> <li>• El sistema verifica si existen simulaciones creadas y muestra la interfaz para realizar clusters.</li> <li>• El investigador llena los campos y presiona el botón verde de ejecución que se encuentra en la parte superior de la interfaz.</li> <li>• El sistema verifica que los campos no están correctos y muestra un mensaje solicitando que se llenen bien los campos</li> </ul>

	<p>EC 1.4: Realizar Clusters con campos vacíos</p>	<p>No se realiza el análisis de clusters porque hay campos vacíos.</p>	<ul style="list-style-type: none"> <li>• El investigador escoge la funcionalidad para realizar clusters dando clic derecho sobre el modelo que desea analizar y seleccionando la opción Analysis/Analysis of Clusters</li> <li>• El sistema verifica si existen simulaciones creadas y muestra la interfaz para realizar clusters.</li> <li>• El investigador llena los campos y presiona el botón verde de ejecución que se encuentra en la parte superior de la interfaz.</li> <li>• El sistema verifica si hay campos vacíos y muestra un mensaje solicitando que se deben llenar todos los campos</li> </ul>
<p>SC 2: Insertar nueva variable</p>	<p>EC 2.1: Insertar nueva variable</p>	<p>Se inserta una nueva variable en la lista de variables a analizar.</p>	<ul style="list-style-type: none"> <li>• El investigador desea adicionar nueva variable y para ello introduce la nueva variable (“New Variable”) y presiona el botón “Add”.</li> <li>• El sistema adiciona la variable a la lista de variables a analizar.</li> </ul>

	EC 2.2: Insertar variable no existente en el modelo.	No se inserta la variable en la lista de variables a analizar porque no existe en el modelo.	<ul style="list-style-type: none"> <li>• El investigador desea adicionar nueva variable y para ello introduce la nueva variable (“New Variable”) y presiona el botón “Add”.</li> <li>• El sistema verifica que la variable no existe en le modelo y muestra un mensaje de error informando que la expresión es incorrecta.</li> </ul>
SC 3: Eliminar variable.	EC 3.1: Eliminar variables existentes.	Se elimina la variable seleccionada de la lista de variables a ser analizadas.	<ul style="list-style-type: none"> <li>• El investigador seleccionó la variable que desea eliminar y presiona el botón (“Delete”).</li> <li>• El sistema elimina la variable de la lista de variables a analizar.</li> </ul>
	EC 3.2: Eliminar última variable.	No se elimina la variable porque debe existir al menos una para realizar el análisis.	<ul style="list-style-type: none"> <li>• El investigador seleccionó la variable que desea eliminar y presiona el botón (“Delete”).</li> <li>• El sistema muestra un mensaje informando que debe existir al menos una variable para hacer el análisis</li> </ul>

DCP del CU Realizar Análisis de Bifurcaciones

**Descripción General**

Permitirá realizar análisis de bifurcaciones de un modelo matemático dado, es decir, el comportamiento de las poblaciones dada la variación de un parámetro.

**Condiciones de Ejecución:**

Debe existir el modelo matemático

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Definir parámetros necesarios	EC 1.1: Insertar los valores correctos	Se insertan los valores necesarios	<ul style="list-style-type: none"> <li>• El Investigador define los valores de los parámetros:</li> <li>• Parameters/Initial Conditions, Initial Time, Final Time, Relative Tolerance, Absolute Tolerance, Numerical Method, Integration step, Type of Bifurcation.</li> <li>• Selecciona el tipo de Bifurcación</li> <li>• Presiona el botón que permite realizar la bifurcación</li> </ul>
	EC 1.2: Insertar Valores no válidos	El sistema no inserta los valores porque no cuenta con los datos correctos.	<ul style="list-style-type: none"> <li>• El Investigador define los valores de los parámetros.</li> <li>• El sistema lanza un mensaje indicando que hay campos con valores incorrectos.</li> </ul>
	EC 1.3: Al Insertar se dejó campos en blancos.	El sistema no tiene todos los datos que necesita.	<ul style="list-style-type: none"> <li>• El Investigador define los valores de los parámetros.</li> <li>• El sistema lanza un mensaje indicando que hay campos con valores incorrectos.</li> </ul>

<p>SC 2: Realizar Bifurcación Local</p>	<p>EC 2.1: Se realiza la bifurcación local</p>	<p>Se realiza el análisis normalmente</p>	<ul style="list-style-type: none"> <li>• El investigador selecciona el tipo de bifurcación Local y presiona el botón Run</li> <li>• Se visualiza el análisis, mostrando el progreso y la gráfica de bifurcaciones, dándole la posibilidad al investigador de detener o pausar el análisis,</li> <li>• No se selecciona ninguna de las opciones y termina el análisis satisfactoriamente mostrando al finalizar los resultados.</li> </ul>
	<p>EC 2.2 Se Pausa el análisis</p>	<p>Se pausa el análisis</p>	<ul style="list-style-type: none"> <li>• Se detiene el análisis al presionar "Pause" y se reanuda al presionar "Continue".</li> </ul>
	<p>EC 2.3: Se Detiene el análisis.</p>	<p>Se detiene el análisis</p>	<ul style="list-style-type: none"> <li>• Se visualiza el progreso de análisis y activa opciones a través de dos botones "Stop" y "Pause".</li> <li>• El usuario selecciona la opción de "Stop" cancelando así el análisis.</li> </ul>

<p>SC3: Realizar Bifurcación Distribuida</p>	<p>EC:3.1 : Se realiza la bifurcación distribuida</p>	<p>Se realiza la bifurcación de forma distribuida mediante la utilización de la GRID.</p>	<ul style="list-style-type: none"> <li>• El investigador selecciona el tipo de bifurcación Local y presiona el botón Run</li> <li>• El usuario se autentica en la Grid.</li> <li>• Se envían los datos para la ejecución del análisis de bifurcaciones al servidor de la Grid, se comienza a hacer el análisis.</li> <li>• Para finalizar se almacena los resultados de las simulaciones en la Base de Datos y guarda una imagen con la gráfica de las bifurcaciones.</li> </ul>
<p>SC4: Realizar Bifurcación con más de parámetro variado</p>	<p>EC: 4.1: Realiza bifurcación con más de un parámetro variado.</p>	<p>No se permite variar dos parámetros.</p>	<ul style="list-style-type: none"> <li>• El investigador luego de haber variado un parámetro trata de variar otro.</li> <li>• El sistema muestra un mensaje de error informando que solo se permite variar un parámetro.</li> </ul>

### Módulo de Editor de Ecuaciones

#### DCP del CU Analizar dimensionalidad del SED

##### Descripción General

Permite conocer si las ecuaciones escritas cuentan con términos cuyas dimensiones no son correctas. Permite asociar variables del SED con dimensiones, eliminar dimensiones del sistema e insertar nuevas dimensiones al sistema

**Condiciones de Ejecución:**

Debe existir un sistema de ecuaciones creado.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Crear nueva dimensión	EC 1.1: Crear dimensión	Inserta nuevas dimensiones al sistema.	<ul style="list-style-type: none"> <li>• El investigador selecciona la opción para crear una nueva dimensión accionando el menú Tools/Dimensions/Add new.</li> <li>• El sistema muestra un diálogo con una región donde se introduce la fórmula de la dimensión y un campo para definirle un nombre.</li> <li>• El investigador construye la fórmula de la dimensión y especifica el nombre con el que desea identificar dicha dimensión y presiona el botón "Insert".</li> <li>• El sistema verifica que los campos no estén vacíos y que no exista otra dimensión con el mismo nombre.</li> <li>• El sistema adiciona la nueva dimensión.</li> </ul>
	EC 1.2: Crear dimensión con nombre existente.	No inserta la dimensión al sistema.	<ul style="list-style-type: none"> <li>• El investigador selecciona la opción para crear una nueva dimensión accionando el menú Tools/Dimensions/Add new.</li> <li>• El sistema muestra un diálogo</li> </ul>

			<p>con una región donde se introduce la fórmula de la dimensión y un campo para definirle un nombre.</p> <ul style="list-style-type: none"> <li>• El investigador especifica los datos necesarios insertando un nombre ya existente en la base de datos y presiona el botón "Insert".</li> <li>• El sistema verifica que el nombre entrado por teclado ya existe y muestra un mensaje informando que ya existe una dimensión con dicho nombre.</li> </ul>
	<p>EC 1.3: Crear dimensión con campos vacíos.</p>		<ul style="list-style-type: none"> <li>• El investigador selecciona la opción para crear una nueva dimensión accionando el menú Tools/Dimensions/Add new.</li> <li>• El sistema muestra un diálogo con una región donde se introduce la fórmula de la dimensión y un campo para definirle un nombre.</li> <li>• El investigador llena los campos y presiona el botón "Insert".</li> <li>• El sistema verifica que faltan campos por llenar y muestra un mensaje informando que debe llenar los campos.</li> </ul>

SC 2: Editar dimensiones	EC 2.1: Eliminar dimensión	Permite eliminar dimensiones del sistema.	<ul style="list-style-type: none"><li>• El investigador selecciona la opción para editar una dimensión accionando el menú Tools/Dimensions/Edit dimensions.</li><li>• El sistema muestra un diálogo que contiene una lista desplegable con todas las dimensiones que ha introducido el usuario y debajo de la lista un botón de eliminar llamado "Delete Selected".</li><li>• El usuario selecciona una dimensión de las que aparecen en la lista desplegable y presiona el botón "Delete Selected".</li><li>• El sistema elimina la dimensión seleccionada.</li></ul>
--------------------------	----------------------------	---	--

	EC 2.2: Editar dimensión	Permite editar las dimensiones ya existentes en el sistema	<ul style="list-style-type: none"><li>• El investigador selecciona la opción para editar una dimensión accionando el menú Tools/Dimensions/Edit dimensions.</li><li>• El sistema muestra un diálogo que contiene una región donde se muestra la fórmula de la dimensión que está seleccionada.</li><li>• El usuario modifica la fórmula de la dimensión seleccionada.</li><li>• El sistema muestra los cambios que realiza el usuario sobre la fórmula.</li></ul>
--	--------------------------	--	---

	EC 2.3: Eliminar dimensión con la lista de dimensiones vacía.	No elimina ninguna dimensión porque la lista de dimensiones está vacía.	<ul style="list-style-type: none"><li>• El investigador selecciona la opción para editar una dimensión accionando el menú Tools/Dimensions/Edit dimensions.</li><li>• El sistema muestra un diálogo que contiene una lista desplegable con todas las dimensiones que ha introducido el usuario y debajo de la lista un botón de eliminar llamado "Delete Selected".</li><li>• El usuario presiona el botón "Delete Selected" estando la lista de dimensiones vacía.</li><li>• El sistema muestra un mensaje informando que la lista de dimensiones ya está vacía.</li></ul>
--	---	---	---

<p>SC 3: Asociar variables con dimensiones</p>	<p>EC 3.1: Asociar variables del SED con dimensiones.</p>	<p>Asocia los valores del sistema de ecuaciones diferenciales con las dimensiones.</p>	<ul style="list-style-type: none"> <li>• El investigador selecciona la opción para asociar las variables con las dimensiones accionando el menú System/Dimensions/Assign dimensions.</li> <li>• El sistema muestra un diálogo con la lista de las variables que no tienen dimensiones asignadas, otra lista con las dimensiones existentes y otra lista para poner las asociaciones entre variables y dimensiones.</li> <li>• El investigador selecciona una variable en lista de variables y selecciona una dimensión en la lista de dimensiones y presiona el botón "Add".</li> <li>• El sistema adiciona la variable con la dimensión asociada a lista de asociaciones y la elimina la variable asociada de la lista de variables.</li> </ul>
--	---	--	--

	<p>EC 3.2: Asociar variables del SED a las dimensiones sin dimensiones creadas.</p>	<p>No asigna las variables con las dimensiones porque no hay dimensiones creadas en el sistema.</p>	<ul style="list-style-type: none"> <li>• El investigador selecciona la opción para asociar las variables con las dimensiones accionando el menú System/Dimensions/Assign dimensions.</li> <li>• El sistema muestra la ventana de asignaciones con la opción Adimensioned</li> <li>• El investigador puede asignar la variable con esta opción o puede ir a la opción de crear nueva dimensión.</li> </ul>
<p>SC 4: Eliminar asociación entre variable y dimensión.</p>	<p>EC 4.1: Eliminar asociación entre variable y dimensión.</p>	<p>Elimina la asignación entre una variable y una dimensión seleccionada de la lista de asignaciones</p>	<ul style="list-style-type: none"> <li>• El investigador selecciona la opción para asociar las variables con las dimensiones accionando el menú System/Dimensions/Assign dimensions.</li> <li>• El sistema muestra la ventana de asignaciones de dimensiones.</li> <li>• El investigador selecciona la asignación que desea eliminar de la lista de asignaciones y presiona la tecla Delete del teclado.</li> <li>• -El sistema elimina la asignación de la lista de asignaciones y regresa la variable a la lista de variables</li> </ul>

DCP del CU Gestionar Edición de SED

**Descripción General**

Permite el trabajo de creación y modificación del SED en estudio. Brinda la posibilidad de adicionar nuevas ecuaciones, insertar identificadores o constantes numéricas, insertar operaciones matemáticas, insertar fórmulas desde biblioteca de expresiones y eliminar elementos del SED.

**Condiciones de Ejecución:**

Ninguna

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Adicionar nueva ecuación	EC 1.1: Adicionar nueva ecuación	Se adiciona una nueva ecuación en el modelo	<ul style="list-style-type: none"> <li>• Se da clic derecho sobre el modelo que se desea editar.</li> <li>• Se selecciona la opción Edit Model.</li> <li>• El sistema muestra la interfaz gráfica del editor.</li> <li>• En la barra de menús se selecciona la opción</li> <li>• System/Equations/New equation</li> <li>• El sistema adiciona la nueva ecuación en el área de trabajo.</li> </ul>
	EC 1.2: Adicionar nueva ecuación diferencial.	Se adiciona una nueva ecuación diferencial en el modelo.	<ul style="list-style-type: none"> <li>• Se da clic derecho sobre el modelo que se desea editar.</li> <li>• Se selecciona la opción Edit Model.</li> <li>• El sistema muestra la interfaz gráfica del editor.</li> <li>• En la barra de menús se selecciona la opción</li> <li>• System/Equations/New differential equation</li> <li>• El sistema adiciona la nueva</li> </ul>

			ecuación diferencial en el área de trabajo.
	EC 1.3: Adicionar nueva función.	Se adiciona una nueva función en el modelo.	<ul style="list-style-type: none"> <li>• Se da clic derecho sobre el modelo que se desea editar.</li> <li>• Se selecciona la opción Edit Model.</li> <li>• El muestra la interfaz gráfica del editor.</li> <li>• En la barra de menús se selecciona la opción</li> <li>• System/Equations/New function</li> <li>• El sistema adiciona la nueva función en el área de trabajo.</li> </ul>
SC 2: Insertar identificadores o constantes numéricas.	EC 2.1: Insertar identificadores o constantes numéricas.	Se inserta identificadores o constantes numéricas	<ul style="list-style-type: none"> <li>• El investigador selecciona el elemento donde desea insertar el texto.</li> <li>• El sistema pasa el foco del teclado al elemento seleccionado.</li> <li>• El investigador presiona la tecla asociada al caracter que desea insertar.</li> <li>• El sistema verifica que el carácter entrado por teclado es correcto.</li> <li>• El sistema adiciona en la región correspondiente al elemento seleccionado el caracter entrado por teclado</li> </ul>
	EC 2.2: Insertar identificadores o	No se inserta el identificador o la	<ul style="list-style-type: none"> <li>• El investigador selecciona el elemento donde desea insertar</li> </ul>

	constantes numéricas con dos puntos.	contante numérica.	<p>el texto.</p> <ul style="list-style-type: none"> <li>• El sistema pasa el foco del teclado al elemento seleccionado.</li> <li>• El investigador presiona la tecla asociada al caracter de punto</li> <li>• El sistema verifica que en ese identificador o constante numérica ya existe un punto y no inserta dicho carácter.</li> </ul>
	EC 2.3: Insertar identificadores o constantes numéricas insertando una letra detrás de un número.	No se inserta el identificador o la constante numérica.	<ul style="list-style-type: none"> <li>• El investigador selecciona el elemento donde desea insertar el texto.</li> <li>• El sistema pasa el foco del teclado al elemento seleccionado.</li> <li>• El investigador presiona la tecla asociada a una letra.</li> <li>• El sistema verifica que en ese identificador o constante numérica ya existe un número y no inserta dicho caracter.</li> </ul>
SC 3: Insertar operaciones matemáticas	EC 3.1: Insertar operaciones matemáticas	Se inserta una operación matemática en el elemento deseado de la ecuación.	<ul style="list-style-type: none"> <li>• El investigador selecciona el elemento donde desea insertar la operación matemática.</li> <li>• El sistema pasa el foco del teclado al elemento seleccionado.</li> <li>• El investigador presiona el botón asociado a la operación matemática que desea</li> </ul>

			<p>insertar.</p> <ul style="list-style-type: none"> <li>• El sistema inserta el elemento representativo de la operación escogida por el investigador.</li> </ul>
	<p>EC 3.2: Cambiar el signo de las constantes numéricas, identificadores u operaciones matemáticas.</p>	<p>Se cambia el signo delante de las constantes numéricas, identificadores y operaciones matemáticas, en caso de que el elemento sea positivo lo pone negativo y viceversa.</p>	<ul style="list-style-type: none"> <li>• El investigador inserta el elemento al que le quiere poner el signo negativo y lo selecciona.</li> <li>• El sistema pasa el foco del teclado al elemento seleccionado.</li> <li>• El investigador presiona el botón asociado al cambio de signo que se encuentra al final en la pestaña de operaciones matemáticas estándar.</li> <li>• El sistema cambia el signo del elemento seleccionado.</li> </ul>
<p>SC 4: Insertar desde biblioteca de expresiones.</p>	<p>EC 4.1: Insertar elemento desde biblioteca de expresiones.</p>	<p>Inserta elementos en el modelo desde la biblioteca de expresiones.</p>	<ul style="list-style-type: none"> <li>• El investigador selecciona la región donde desea insertar el elemento.</li> <li>• El sistema pasa el foco a la región seleccionada.</li> <li>• El investigador selecciona la categoría donde se encuentra el elemento que desea insertar y presiona el botón asociado a dicho elemento desde la biblioteca de expresiones.</li> <li>• El sistema sustituye la región que fue seleccionada por la expresión que representa el</li> </ul>

			botón presionado.
	EC 4.2: Insertar ecuación desde biblioteca de expresiones.	Inserta una ecuación en el modelo desde la biblioteca de expresiones.	<ul style="list-style-type: none"> <li>• El investigador selecciona la región donde desea insertar la función.</li> <li>• El sistema pasa el foco a la región seleccionada.</li> <li>• El investigador selecciona la categoría donde se encuentra la ecuación que desea insertar y presiona el botón asociado a dicha ecuación desde la biblioteca de expresiones.</li> <li>• El sistema adiciona la ecuación seleccionada al final, después de todas las ecuaciones.</li> </ul>
SC 5: Eliminar elementos del SED	EC 5.1: Eliminar identificador o constante numérica.	Se elimina el identificador o constante numérica que seleccione el investigador.	<ul style="list-style-type: none"> <li>• El investigador selecciona la constante o identificador a eliminar.</li> <li>• El sistema pasa el foco de teclado al elemento seleccionado.</li> <li>• El investigador acciona el clic derecho situado sobre el elemento seleccionada y escoge la opción "Delete".</li> <li>• El sistema sustituye el elemento seleccionado por una casilla en blanco.</li> </ul>
	EC 5.2: Eliminar identificador o constante numérica	Se elimina el identificador o constante	<ul style="list-style-type: none"> <li>• El investigador selecciona la constante o identificador a eliminar.</li> </ul>

	vinculada con expresión de suma o resta.	numérica y la expresión de suma o resta la cual está vinculada.	<ul style="list-style-type: none"> <li>• El sistema pasa el foco de teclado al elemento seleccionado.</li> <li>• El investigador acciona el clic derecho situado sobre el elemento seleccionada y escoge la opción "Delete".</li> <li>• El sistema elimina el elemento seleccionado y la expresión de suma o resta a la cual está vinculado dicho elemento.</li> </ul>
	EC 5.3: Eliminar identificador o constante numérica vinculada con expresión distinta a las de suma y resta.	Se sustituye el identificador o contante numérica seleccionado por una casilla en blanco.	<ul style="list-style-type: none"> <li>• El investigador selecciona la constante o identificador a eliminar.</li> <li>• El sistema pasa el foco de teclado al elemento seleccionado.</li> <li>• El investigador acciona el clic derecho situado sobre el elemento seleccionada y escoge la opción "Delete".</li> <li>• El sistema sustituye la constante o identificador que se desea eliminar por una casilla en blanco.</li> </ul>
	EC 5.4: Eliminar expresión matemática del SED	Se elimina la expresión matemática que seleccione el investigador junto con la constante numérica o	<ul style="list-style-type: none"> <li>• El investigador selecciona la expresión a eliminar.</li> <li>• El sistema pasa el foco de teclado a la región seleccionada.</li> <li>• El investigador acciona el clic derecho situado sobre dicha</li> </ul>

		identificador a la cual está vinculada.	<p>región y escoge la opción "Delete".</p> <ul style="list-style-type: none"> <li>• El sistema elimina la expresión seleccionada y el identificador o constante numérica a la cual está vinculada.</li> </ul>
	EC 5.5: Eliminar ecuación del SED	Elimina la ecuación que el investigador seleccione.	<ul style="list-style-type: none"> <li>• El investigador selecciona la ecuación o función a eliminar.</li> <li>• El sistema pasa el foco de teclado a la región seleccionada.</li> <li>• El investigador acciona el clic derecho situado sobre dicha región y escoge la opción "Delete".</li> <li>• El sistema elimina la ecuación o función seleccionada.</li> </ul>

Para consultar el resto de los diseños de casos de prueba, remitirse al Expediente de Proyecto.

## 2.6.2 Pruebas de Integración

Los DCP utilizados para realizar la prueba de integración se nombran a continuación:

### 1. DCP del CU Mostrar Dinámicas Aleatorias:

Este caso de prueba vincula el módulo Simulador con el de Análisis, pues el análisis de dinámicas se le realiza a simulaciones hechas en el módulo Simulador que luego son guardadas en la BD.

### 2. DCP del CU Realizar Análisis de clasificación:

Este caso de prueba vincula el módulo Simulador con el de Análisis, pues las clasificaciones se les realizan a simulaciones hechas en el módulo Simulador que luego son guardadas en la BD.

### 3. DCP del CU Realizar Clusters:

Este caso de prueba vincula el módulo Simulador con el de Análisis, pues las agrupaciones por algoritmos de clusters se les realizan a simulaciones hechas en el módulo Simulador que luego son guardadas en la BD.

#### **4. DCP del CU Realizar Análisis por reglas:**

Este caso de prueba vincula los módulos Editor y Funcionalidades Comunes con el de Análisis, pues los análisis por reglas se realizan solamente cuando hay modelos creados, y los modelos se crean en el módulo Funcionalidades Comunes y se les ponen ecuaciones y se guardan en la BD en el módulo Editor.

#### **5. DCP del CU Realizar análisis de Bifurcaciones:**

Este caso de prueba vincula los módulos Editor y Funcionalidades Comunes con el de Análisis, pues los análisis de bifurcaciones se realizan solamente cuando hay modelos creados, y los modelos se crean en el módulo Funcionalidades Comunes y se les ponen ecuaciones y se guardan en la BD en el módulo Editor.

Para ver estos diseños de casos de prueba remitirse al expediente de proyecto.

## **2.7 Pruebas de carga y estrés**

Para la realización de esta prueba se hizo uso de 25 computadoras con Microprocesador Pentium IV, Sistema Operativo Windows, con 1 GB de memoria RAM y 160 GB de Disco Duro. Con esta prueba se comprobó el funcionamiento de las operaciones más significativas, considerando como las más significativas aquellas que serían más utilizadas por el cliente y las más complicadas.

La prueba se realizó en el Laboratorio Industrial de Pruebas de Software (LIPS) con ayuda de estudiantes reales. Para guiar a los estudiantes en la realización de la prueba se confeccionó un manual donde se detallaron cada una de las funcionalidades a probar. En este manual se indicaron los datos reales que debían ser insertados en cada uno de los casos y se explicó paso por paso cómo se debía ir ejecutando la prueba. Para ver este manual remitirse al anexo 4.

Los casos de uso que se probaron se nombran a continuación (Para ver estos CU remitirse al documento de Especificación de Casos de Uso en el expediente de proyecto).

### **Módulo de funcionalidades comunes:**

- Gestionar conexión
- Gestionar Sistema Biológico.
- Gestionar Modelo Matemático.

**Módulo editor de ecuaciones:**

- Cargar Modelo Matemático.
- Salvar Modelo Matemático.
- Gestionar Edición SED.
- Exportar SED.

**Módulo simulador:**

- Gestionar Simulación.
- Autenticar en la Grid.

**Módulo análisis.**

- Realizar Clusters.
- Realizar Análisis de Clasificación.
- Realizar Análisis de Bifurcaciones.
- Mostrar Dinámicas Aleatorias

Además de la prueba antes mencionada, también se realizó la prueba con una herramienta llamada Simulador de Conexiones MySQL que trabaja directamente sobre la BD.

**2.7.1 Simulador de conexiones MySQL**

Las pruebas de estrés llevan al sistema a un punto máximo para poder medir sus capacidades y las condiciones en las cuales trabaja realizando una cantidad definida de peticiones y procesos. Para realizar dichas pruebas a la base de datos se utilizó el Simulador de Conexiones MySQL, esta es una herramienta implementada en C# que simula un número de conexiones y peticiones a la base de datos indicado por el usuario. Luego de establecida esta cantidad *n* de conexiones entonces se procede a la ejecución de las pruebas para valorar el comportamiento del sistema frente a diferentes números de conexiones.

Para la realización de la prueba, como para todas las pruebas realizadas a la aplicación, se creó un escenario de pruebas lo más semejante posible al ambiente para el cual fue diseñado el software.

Se utilizó una computadora con procesador Pentium D, con 2 GB de RAM y 40 GB de disco duro a 2.8 GHz.

Para realizar la prueba se introdujo primeramente la cadena de conexión referente a la base de datos, donde se especificó la dirección de ésta, puerto de conexión, y datos para realizar la autenticación, esta cadena quedó de la siguiente manera:

```
DATABASE=biosys;DSN=prueba;OPTION=0;PWD=lagardere;PORT=0;SERVER=localhost;UID=root
```

Luego se introdujo la consulta y seguidamente la cantidad de conexiones que se querían realizar junto a la cantidad de peticiones de la consulta por cada una de las conexiones a la base de datos.

En la prueba se ejecutaron 5 consultas con una conexión a la base de datos, luego se ejecutaron las mismas consultas con 20 conexiones y después cuando habían 30 conexiones establecidas, realizando 10 peticiones a la base de datos en cada caso.

Las consultas utilizadas para hacer las pruebas se consideran unas de las que más probabilidad tienen de ser utilizadas. Estas consultas son mostradas a continuación:

1. Esta es una consulta que devuelve los resultados de una simulación para luego graficarla.

```
"SELECT resultados.Tiempo,resultados.Id_Poblacion FROM resultados WHERE resultados.Id_Simulacion = '106_107_117_131_137"
```

2. Esta es una consulta que devuelve los resultados de una simulación dado un tiempo determinado, la misma se utiliza para dar soporte a la funcionalidad de análisis de clusters.

```
"SELECT resultados.Tiempo, resultados.Id_Poblacion FROM resultados WHERE resultados.Id_Simulacion = '1_2_3_1_2_3_4_5_6_7_8_9_10_11_12' AND resultados.Tiempo = '0'ORDER BY Id_Poblacion"
```

- 3 Esta consulta devuelve los resultados de una simulación en el rango de tiempo de 1 a 10 y se utiliza para graficar una simulación.

```
"select resultados.Id_Poblacion, resultados.Tiempo from resultados where Id_Simulacion='1_2_3_1_2_3_4_5_6_7_8_9_10_11_12' and Tiempo>=0 and Tiempo<=10 order by Tiempo,Id_Poblacion"
```

4 Esta consulta devuelve una cantidad de id de simulaciones aleatorias.

```
"SELECT simulacion.Id_Simulacion FROM simulacion WHERE Id_Modelo =2 order by rand()  
limit 10"
```

5 Devuelve una simulación aleatoria.

```
"SELECT simulacion.Id_Simulacion FROM simulacion WHERE Id_Modelo =1 order by rand()  
LIMIT 1"
```

Se realizaron tres iteraciones de prueba, la primera para obtener el tiempo de respuesta del sistema al ejecutar las consultas para 1 clientes, la segunda para 20 clientes porque el entorno para el cual el software fue creado el producto cuenta con aproximadamente 20 computadoras para trabajar como clientes, y la tercera para 30 clientes, para asegurar que el sistema trabajará con igual eficiencia en caso de aumentar en un futuro esa cantidad de computadoras.

### 2.7.2 Evaluación de atributos de calidad

El grupo de calidad de la facultad propuso una lista chequeo como herramienta para evaluar algunos atributos de calidad que deben estar presente en la aplicación.

Los atributos de calidad de software que fueron evaluados se muestran a continuación:

**Seguridad:** Donde se comprobó la capacidad del sistema para resistir intentos de uso y negación de servicios a usuarios no autorizados sin restar servicios a los usuarios autorizados.

**Portabilidad:** Se verificó la habilidad del sistema para ejecutarse en diferentes ambientes (hardware, software, o una combinación de ambos).

**Eficiencia:** Se chequeó como la aplicación hacia uso de los recursos, incluyen memoria y tiempos de ejecución, en general los usuarios finales miden este atributo con la rapidez con que el programa le permite realizar sus actividades

**Confiabilidad:** Se verificó la habilidad del programa para cumplir con todas sus funciones de forma constante teniendo un largo periodo de tiempo entre fallas.

**Usabilidad:** Donde se comprobó la capacidad del sistema para que el usuario pueda aprender a usar la interfaz del sistema fácilmente.

## **2.8. Conclusiones del capítulo**

En este capítulo se expuso una breve descripción de la aplicación diseñada para el análisis y la simulación de sistemas biológicos: alas BioSyS, se describió los aspectos fundamentales del plan de prueba como son la estrategia de prueba y el cronograma de pruebas a seguir durante todo el proceso. Se diseñaron los caso de pruebas que se le aplicaron a los cuatro módulos en varias iteraciones con el fin de descubrir el máximo de errores los que luego fueron enviado al grupo de desarrolladores como no conformidades.

# Capítulo **3**

## Resultados de las pruebas realizadas

En este capítulo se hace un análisis de los resultados obtenidos después de haber diseñado y aplicado las pruebas al BioSyS, exponiendo detalladamente cada uno de los defectos y dificultades detectadas en cada una de las iteraciones. Para ello se detallan las no conformidades encontradas en cada una de las pruebas que fueron aplicadas.

### 3.1. Análisis de los resultados obtenidos en las pruebas a la documentación

Para la realización de las pruebas a la documentación primeramente se hizo una investigación detallada de lo que en realidad quería el cliente y de las características que debía tener el producto y seguidamente se procedió a la realización de las pruebas. Se utilizó una PC con Microprocesador Pentium IV, Sistema Operativo Linux, con 1 GB de memoria RAM y 160 GB de Disco Duro.

Se le realizaron tres iteraciones de prueba a cada uno de los documentos. Estos documentos fueron:

- Manual de Instalación.
- Manual de Usuarios.
- Especificación de requisitos.
- Especificación de Casos de Uso.

#### Análisis de los resultados obtenidos en las pruebas al Manual de Instalación.

Las tres iteraciones de pruebas realizadas al manual de instalación arrojaron los siguientes resultados:

Iteración	No Conformidades significativas	No Conformidades no significativas	Recomendaciones	Total de no conformidades	No procedieron
Prime ra iteración	6	0	1	7	0
Segunda iteración	7	1	0	8	2
Tercera iteración	0	0	0	0	0

A continuación se hace un análisis de los errores encontrados en el Manual de Instalación en cada una de sus iteraciones de pruebas.

Análisis de los resultados obtenidos en las pruebas al manual de instalación en la primera iteración

En la primera iteración de pruebas realizada al Manual de Instalación resultaron encontrados 7 errores, de ellos 6 fueron significativos y uno fue una recomendación, de esos 6 errores significativos 2 fueron clasificados como problemas de redacción, 3 como problemas de ortografía y uno como error técnico. Todos los errores fueron resueltos.

Análisis de los resultados obtenidos en las pruebas al manual de instalación en la segunda iteración

En las segundas pruebas aplicadas al documento fueron encontrados 8 errores, de ellos uno fue no significativo y 7 fueron significativos. De los 7 errores significativos uno fue por error de ortografía, uno por problemas de redacción y 5 por errores técnicos. De estos errores 2 no procedieron y 6 fueron resueltos satisfactoriamente.

Análisis de los resultados obtenidos en las pruebas al manual de instalación en la tercera iteración

Luego de haber terminado la segunda iteración de pruebas se hizo una revisión completa del documento y no se encontraron más errores por lo que se declaró el documento listo para ser liberado.

**Análisis de los resultados obtenidos en las pruebas al manual de usuarios**

Como resultado a las pruebas aplicadas al manual de usuarios se obtuvieron los siguientes resultados:

Iteración	No Conformidades significativas	No Conformidades no significativas	Recomendaciones	Total de no conformidades	No procedieron
Primera iteración	29	0	0	29	1
Segunda iteración	40	0	0	40	13
Tercera iteración	19	0	0	19	0

Análisis de los resultados obtenidos en las pruebas al manual de usuarios en la primera iteración

En la primera iteración de las pruebas realizadas al Manual de Usuarios fueron encontradas 29 no conformidades significativas de las cuales 16 fueron problemas de ortografía, 6 problemas de

redacción y 7 por errores técnicos. De estos problemas, una no conformidad no procedió, 17 fueron resueltas y 11 quedaron pendientes para la segunda iteración.

Análisis de los resultados obtenidos en las pruebas al manual de usuarios en la segunda iteración

En las segundas pruebas aplicadas al documento fueron encontrados 40 errores todos significativos, de ellos 15 fueron errores de ortografía, 3 problemas de redacción y 22 errores técnicos. De estos errores junto con los 11 que quedaron pendientes en la primera iteración 13 no procedieron, 5 quedaron pendientes para la siguiente iteración y se le dio respuestas a 33 no conformidades.

Análisis de los resultados obtenidos en las pruebas al manual de usuarios en la tercera iteración

En esta prueba se encontraron 19 no conformidades todos significativos, de ellos 3 fueron errores de redacción, 7 problemas de ortografía y 9 errores técnicos. Todos los errores encontrados en esta iteración y los que quedaron pendientes en la iteración anterior fueron resueltos correctamente.

Luego de haber realizado estas tres iteraciones se hizo otra revisión al documento y no se encontraron más errores.

**Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Requisitos**

De las pruebas realizadas al documento de Especificación de Requisitos se obtuvieron los siguientes resultados:

Iteración	No Conformidades significativas	No Conformidades no significativas	Recomendaciones	Total de no conformidades	No procedieron
Primera iteración	20	0	0	20	4
Segunda iteración	4	2	0	6	1
Tercera iteración	0	0	0	0	0

Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Requisitos en la primera iteración

En la primera iteración de las pruebas realizadas al documento de Especificación de Requisitos fueron encontradas 20 no conformidades significativas de las cuales 2 fueron problemas de ortografía, 5

problemas de redacción y 13 por errores técnicos. De estos problemas 4 no procedieron y 16 fueron resueltos.

Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Requisitos en la segunda iteración

En esta prueba se encontraron 6 no conformidades, de ellas 4 fueron significativas y 2 no significativas. De las 4 no conformidades significativas 2 se clasificaron como errores técnicos y 2 como problemas de formato del documento. Todos los errores encontrados en esta iteración fueron resueltos correctamente excepto uno que no procedió.

Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Requisitos en la tercera iteración

Luego de haber terminado la segunda iteración de pruebas se hizo una revisión completa del documento y no se encontraron más errores por lo que se declaró el documento listo para ser liberado.

**Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Casos de Uso**

Al aplicar las pruebas al documento de Especificación de Casos de Uso en sus tres iteraciones se obtuvieron los siguientes resultados:

Iteración	No Conformidades significativas	No Conformidades no significativas	Recomendaciones	Total de no conformidades	No procedieron
<b>Primera iteración</b>	115	0	1	116	6
<b>Segunda iteración</b>	29	2	0	31	4
<b>Tercera iteración</b>	19	1	0	20	3

A continuación se hace un análisis de los errores encontrados por iteraciones.

Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Casos de Uso en la primera iteración

Al realizar la primera iteración de prueba al documento de Especificación de Casos de Uso se encontró un total de 116 errores, de los cuales uno fue una recomendación y 115 fueron no conformidades significativas. De estas 115 no conformidades significativas 68 fueron clasificadas como errores técnicos, 14 como errores de ortografía y 33 como errores de redacción. Todos estos problemas fueron

enviados al analista el cual dio respuestas a 100 de estos errores y determinó que 6 de ellos no procedían. Quedaron pendientes 10 errores para ser resueltos en la segunda iteración.

Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Casos de Uso en la segunda iteración

La segunda iteración de las pruebas al documento de Especificación de Casos de Uso arrojó 31 no conformidades de las cuales 2 fueron no significativas y 29 significativas. De las 29 no conformidades significativas 3 fueron por problemas de redacción y 26 por errores técnicos. Como respuesta a estos problemas el analista resolvió 35 de ellos, determinó que 4 no procedían y dejó sin responder 2 no conformidades las cuales quedaron pendientes para la siguiente iteración.

Análisis de los resultados obtenidos en las pruebas al documento de Especificación de Casos de Uso en la tercera iteración

De las 20 no conformidades encontradas en la tercera iteración una fue no significativa y 19 significativas. De las 19 conformidades significativas 15 fueron clasificadas como errores técnicos, 3 problemas de redacción y un error de ortografía. De estos problemas 3 no procedieron y el resto fue resuelto correctamente junto con los errores que quedaron pendientes en la iteración anterior.

Luego de haber realizado estas tres iteraciones se hizo otra revisión al documento y no se encontraron más errores por lo que se declaró el documento listo para ser liberado.

**3.2. Análisis de los resultados obtenidos en las pruebas funcionales**

Las pruebas aplicadas fueron diseñadas por dos probadoras del equipo de desarrollo. Se utilizó una PC con Microprocesador Pentium IV, Sistema Operativo Linux, con al menos 1 GB de memoria RAM y 160 GB de Disco Duro. Para cada caso de uso se confeccionó un caso de prueba compuesto por clases de equivalencia con datos tanto válidos como inválidos y obteniendo los siguientes resultados en cada una de las iteraciones realizadas:

Iteración	No Conformidades significativas	No Conformidades no significativas	Recomendaciones	Total de no conformidades	No procedieron
Prime ra iteración	23	1	3	27	4
Segunda iteración	10	2	0	12	3

<b>Tercera iteración</b>	5	0	1	6	3
--------------------------	---	---	---	---	---

#### Análisis de los resultados obtenidos en las pruebas funcionales en la primera iteración

En la primera iteración de las 23 no conformidades significativas encontradas 19 fueron por problemas de funcionalidad de la aplicación, 2 por problemas de correspondencia con la documentación y 2 por problemas de validación de los datos. Estos problemas fueron documentados y entregados al equipo de desarrollo y todas fueron respondidas correctamente.

#### Análisis de los resultados obtenidos en las pruebas funcionales en la segunda iteración

De las diez no conformidades significativas encontradas en la segunda iteración, 9 resultaron ser por problemas de funcionalidad de la aplicación y una por problemas de correspondencia con la documentación. En las respuestas del equipo de desarrollo para dar solución a estos problemas, 3 no conformidades no procedieron, 4 fueron resueltas y 5 quedaron pendientes para la tercera iteración.

#### Análisis de los resultados obtenidos en las pruebas funcionales en la tercera iteración

En la tercera iteración se encontraron 6 nuevos errores de los cuales 5 eran significativos por problemas de funcionalidad de la aplicación y uno fue una recomendación. En el documento de no conformidades de la tercera iteración se documentaron esas 6 no conformidades más las 5 que quedaron pendientes de la iteración anterior. En las respuestas del equipo de desarrollo para dar solución a estos problemas 3 no procedieron y las 8 restantes fueron resueltas eficientemente.

### **3.3. Análisis de los resultados obtenidos en las pruebas de Integración**

Las pruebas de integración se ejecutaron en 1 PC con Microprocesador Pentium IV, Sistema Operativo Linux, con al menos 1 GB de memoria RAM y 160 GB de Disco Duro. Para realizar las pruebas de integración se hizo uso de los diseños de casos de pruebas que vinculaban funciones entre los diferentes módulos y de esta forma detectar las fallas de interacción entre las distintas clases que componen al sistema.

Las pruebas de integración realizadas a la aplicación no arrojaron no conformidades.

### **3.4. Análisis de los resultados obtenidos en la prueba de Sistema**

Esta etapa de pruebas se dividió en tres etapas, estas fueron:

- Las pruebas se harán de forma manual con estudiantes reales.

- Las pruebas se harán con el Simulador de Conexiones MySQL.
- Las pruebas se harán con una lista de chequeos.

A continuación se hace un resumen de los resultados de estas pruebas.

### **3.4.1 Análisis de los resultados obtenidos en la prueba de Carga y Estrés realizada con estudiantes reales.**

Primeramente se planeó que los estudiantes hicieran las pruebas todos juntos y luego separarlos en dos grupos para que el primer grupo comenzara a hacer las pruebas nuevamente y luego comenzara a hacerlas el segundo grupo, para así lograr un bucle, pero esta vez probando nuevas funcionalidades y nuevos datos de entrada.

En un principio se autenticaron 20 estudiantes para comprobar que la aplicación soportara al menos las veinte conexiones que debe aguantar como promedio y seguidamente se hicieron 5 conexiones más para probar las funcionalidades más significativas. La aplicación aunque un poco lenta a la hora de dar respuesta permitió las 20 conexiones primeras más las 5 que se hicieron después.

Luego de haber probado que la base de datos soportaba todas las conexiones se comenzaron a hacer las pruebas a las funcionalidades con ayuda del manual, ya con 25 conexiones la aplicación sólo permitió realizar la primera inserción de un sistema biológico que fue la primera función que se probó, al realizar estas inserciones algunas computadoras se bloquearon y fue necesario cerrarlas y abrirlas nuevamente, en otros casos cuando se actualizó no se mostraron todos los sistemas biológicos que habían sido insertados en ese momento y en 4 computadoras se mostró un mensaje error que no había salido antes. Anexo 6

La segunda función a probar fue la creación de un modelo matemático e inserción de ecuaciones matemáticas, en esta prueba la aplicación en algunas computadoras mostró mensajes indicando errores en la construcción de las ecuaciones, error que no se mostró cuando la prueba fue hecha por las probadoras haciendo uso de una sola computadora (anexo7), otras computadoras se bloquearon y al intentar actualizar la aplicación se mostró un mensaje en blanco (anexo 8) computadoras. Luego cuando se intentó nuevamente establecer conexión la aplicación mostró un mensaje de error negando dicha conexión por lo que fue necesario suspender la prueba.

Otro problema encontrado en el sistema fue que se mostró un mensaje de error que no siempre se muestra incluso bajo las mismas condiciones en que se ejecuta en otras ocasiones, a este problema no se le ha podido dar solución porque los desarrolladores no han logrado ver el mensaje para determinar a qué se debe.

En la segunda iteración de pruebas de carga y estrés de forma manual se hizo con 20 estudiantes esta vez con un nuevo servidor de base de datos. Al parecer los problemas que se presentaron en la

primera iteración se debieron a problemas de conexión porque en esta nueva iteración se obtuvieron muy buenos resultados. En esta se hizo la misma prueba pero esta vez con los siguientes datos de conexión a la BD:

DB"Server: PostgreSQL

Server IP: 10.7.19.201

User: biosys

DB' name: biosys

Port: 5900

Password: se especificó a la hora de realizar la prueba.

En esta prueba los tiempos de respuestas del sistema fueron bastante buenos y la aplicación soportó de manera eficiente las 20 conexiones establecidas. Se comprobó además que la aplicación no permite borrar sistemas o modelos biológicos que están siendo utilizados por otros usuarios, en este caso no permite borrarlo pero muestra un mensaje de error que no refleja correctamente la razón del mensaje de error planteado. Se encontró nuevamente el problema de los mensajes que se muestran en algunas ocasiones y en otras no pero esta vez en conjunto con el líder del proyecto se determinó la manera de darle solución.

Debido a que la segunda iteración de esta prueba fue realizada solamente con los 20 estudiantes requeridos se determinó realizar una tercera iteración con no menos de 25 estudiantes.

La tercera iteración se llevó a cabo bajo las mismas condiciones que las pruebas anteriores, en esta se conectaron 25 estudiantes que comenzaron a trabajar simultáneamente en las actividades descritas en el manual de pruebas anteriormente citado, por problemas técnicos un estudiante se vio obligado a salir cuando el resto de su compañeros ya se encontraban ejecutando funcionalidades dentro de BioSyS al intentar una nueva conexión la base de datos no la aceptó y mostró el mensaje del anexo # 9, a partir de ese momento no se aceptó ninguna nueva conexión por lo que la prueba continuó con 24 estudiantes, cuyo valor de conexiones es aceptable porque en los requisitos se define que la aplicación acepta un promedio de 20 conexiones y se logró un valor superior. El servidor de la GRID se no estaba disponible por lo que no se pudo probar a fondo la simulación distribuida pero en esta sección cuando aparecía la ventana de autenticación y el usuario se logueaba la aplicación mostraba un cartel (Anexo 10) indicando que un parámetro no estaba variando, cuando todos los parámetros variaban correctamente y lo que realmente ocurría era un problema de conexión, por lo que se le hizo llegar al equipo de desarrolladores la no conformidad para que en esos casos se mostrara un mensaje más ilustrativo del verdadero problema que se había presentado. En esta iteración se comprobó que hubieran sido resueltas las no conformidades encontradas en iteraciones anteriores.

### 3.4.2. Análisis de los resultados obtenidos en las pruebas realizadas con el Simulador de Conexiones MySQL

En las siguientes tablas se muestra el resultado de la prueba de estrés realizada a la base de datos, para ello se ejecutaron ver consultas con una conexión a la base de datos, luego se ejecutaron las mismas consultas con 20 conexiones y después cuando habían 30 conexiones establecidas, realizando peticiones a la base de datos.

<b>Consulta</b>	<b>Conexiones</b>	<b>Peticiones</b>	<b>Tiempo</b>
Consulta 1	1	10	281 ms
Consulta 2	1	10	421 ms
Consulta 3	1	10	406 ms
Consulta 4	1	10	15 ms
Consulta 5	1	10	31 ms

Pruebas de estrés en la segunda iteración:

<b>Consulta</b>	<b>Conexiones</b>	<b>Peticiones</b>	<b>Tiempo</b>
Consulta 1	20	10	303 ms
Consulta 2	20	10	732 ms
Consulta 3	20	10	723 ms
Consulta 4	20	10	515 ms
Consulta 5	20	10	609 ms

Pruebas de estrés en la tercera iteración:

<b>Consultas</b>	<b>Conexiones</b>	<b>Peticiones</b>	<b>Tiempo</b>
Consulta 1	30	10	592 ms
Consulta 2	30	10	635 ms
Consulta 3	30	10	785 ms
Consulta 4	30	10	765 ms
Consulta 5	30	10	906 ms

Puede decirse que aunque el tiempo de respuesta a las consultas es menor para un menor número de conexiones y peticiones, en ambos casos se encuentra en correspondencia con las condiciones establecidas para ejecutar las pruebas. Para estas condiciones la base de datos soporta 30 conexiones, siendo esto satisfactorio ya que una vez que el sistema se encuentre desplegado no va a exceder de 20 conexiones.

Se puede concluir con que los tiempos de respuestas obtenidos como resultado son tiempos de espera admisibles, son tiempos que no llegan a 1 segundo y se puede afirmar que el software BioSys no perderá calidad, ni eficiencia de trabajo en caso de aumentar la cantidad de computadoras trabajando como clientes del sistema.

En los Anexos se puede encontrar un grupo de imágenes resultantes de las pruebas para cada una de las consultas en sus tres iteraciones. (Anexo 11)

### 3.4.3. Análisis de los resultados obtenidos en la evaluación de los atributos de calidad

Para la evaluación de los atributos de calidad de software se hizo uso de una PC con Microprocesador Pentium IV, Sistema Operativo Linux, con 1 GB de memoria RAM y 160 GB de Disco Duro.

Para llevar a cabo esta evaluación se hizo uso de una lista de chequeos en la que estaban registradas una serie de preguntas que ayudarían a evaluar cada uno de los atributos de calidad. Estos atributos fueron evaluados en un rango de cero a cinco quedando de la siguiente manera:

Evaluación	Clasificación
0	No disponible
2 – 3	Parcialmente disponible
4	Disponible
5	Muy bien implementada

A continuación se hace un análisis de la evaluación de cada uno de estos atributos.

Aspecto	Evaluación
Seguridad	3 (Parcialmente disponible)
Portabilidad	5 (Muy bien implementada)
Eficiencia	4 (Disponible)
Confiabilidad	0 (No disponible)
Usabilidad	3 (Parcialmente disponible)

El atributo de **seguridad** del software fue evaluado con una nota de 3, nota con la que fue clasificado como parcialmente disponible. Uno de los problemas encontrados en el producto que afectan su seguridad es que hasta el momento no han sido borradas las contraseñas del equipo de prueba y desarrolladores y que no están instrumentados los auto chequeos que permitan la protección contra virus.

El atributo de **portabilidad** del software fue evaluado con 5 ya que es un software que se adapta a cualquier ambiente de manera eficiente.

El atributo de **eficiencia** alcanzó una evaluación de 4, nota con la que fue clasificado como un atributo disponible. El principal problema encontrado en la aplicación que disminuye su calidad con respecto a la eficiencia es que el sistema demora en dar respuestas ante determinadas acciones.

El atributo de calidad que más problema presentó en la aplicación fue la **confiabilidad** que fue evaluada con una nota de cero. La aplicación no se considera para nada confiable debido a que no se recupera ante fallas, demora mucho tiempo en dar respuesta a la hora de ejecutar determinadas funcionalidades además de que actualmente presenta problemas a los que no se pudo dar solución

El atributo de **usabilidad** del software fue evaluado con una nota de 3 puntos, nota con la que fue clasificado como parcialmente disponible. Entre los principales problemas encontrados en la aplicación que afectan su usabilidad se encuentran que el sistema no proporciona un soporte apropiado a los usuarios más novatos, otro problema es que el sistema en algunas ocasiones no asiste al usuario cuando hay más de una manera de realizar un procedimiento, otro problemas es que el sistema no actúa como correctos de errores y no permite al usuario interrumpir su tarea y terminar de realizarla más tarde además de que no permite deshacer y rehacer acciones. Otro problema no muy significativo pero si importante es que el sistema en algunas ocasiones no muestra textos descriptivos para guiar al usuario y otro bastante significativo es que el sistema no tiene implementada la ayuda al usuario.

### **3.5. Análisis de las no conformidades más significativas encontradas en la documentación**

Después de revisar toda la documentación correspondiente se puede concluir señalando que los errores más significativos encontrados en la documentación se debieron a que se violó en muchas ocasiones los principales aspectos que se deben tener en cuenta a la hora de documentar. Por ejemplo en el caso del documento de Especificación de CU ocurrió mucho que no se describieron todos los flujos alternativos del CU y en algunas descripciones no se detalló correctamente el flujo de eventos, además de que no se especificó en repetidas ocasiones dónde terminaban los CU. En el Manual de Usuarios ocurrió también que había funcionalidades en la aplicación que no se abordaron

en el documento y otro aspecto significativo fue que en este documento al igual que en el documento de Especificación de CU se encontraron varios errores de ortografía.

Es válido aclarar que no se pudo probar el formato de la documentación debido a que las pautas no le fueron entregadas al equipo de pruebas.

### **3.6. Análisis de las no conformidades más significativas encontradas en la aplicación**

Luego de aplicar el proceso de prueba a la aplicación se puede concluir señalando que uno de los problemas más significativos encontrados en la documentación durante la etapa de pruebas y que no se pudo dar solución debidamente es el problema de los mensajes de error. Los mensajes de error que son enviados desde el T- ARENAL no son entendibles para el usuario y a veces están vacíos.

A este problema no se le pudo dar solución completamente debido a que algunos de estos mensajes no siempre se muestran, hay veces en la que se ejecuta una acción y se muestra el mensaje de error vacío o que no se entiende y luego de un tiempo se ejecuta nuevamente la misma acción y no se muestra dicho mensaje por lo que no se pudo determinar por qué viene dado.

Otro problema se debe a que la aplicación es muy lenta a la hora de realizar algunas acciones como es el caso en el que se desconecta una conexión al servidor y se intenta conectar de nuevo. En este caso la aplicación se bloquea y no se puede hacer nada más. Otra de las veces en las que la aplicación se demora es cuando se guarda un sistema biológico o un modelo en la base de datos. Las respuestas que da el equipo de desarrollo a este problema es que se debe a problemas de conexión o congestión en las líneas.

### **3.7. Conclusiones del capítulo**

Luego de haber realizado el análisis de los resultados obtenidos al aplicar las pruebas diseñadas a la versión 1.0 del BioSyS, se puede resumir que en la aplicación se detectaron en total 45 no conformidades. De éstas no conformidades 38 fueron significativas y de ellas 33 fueron por problemas de funcionalidad de la aplicación, 3 por problemas de correspondencia con la documentación y 2 por problemas en la validación de datos de entrada. De todas las no conformidades 10 no procedieron y 35 fueron resueltas satisfactoriamente.

Las pruebas realizadas a la documentación arrojaron un total de 296 no conformidades, resultando encontradas en el Manual de Instalación 15 no conformidades de las cuales 13 fueron significativas, una no significativa y una recomendación, en el Manual de usuarios se encontraron 88 no conformidades, en el documento de Especificación de Requisitos 26 y en el de Especificación de Casos de Uso 167 no conformidades.

De manera general, haciendo una suma de los errores encontrados en la documentación y la aplicación se encontró un total de 341 no conformidades, de las cuales 326 resultaron significativas, 9 no significativas y 6 recomendaciones.

## **Conclusiones**

Con el estudio y las pruebas que se han llevado a cabo en el trabajo “alas BioSyS: Proceso de Pruebas de Liberación de la Facultad 6” se ha cumplido el objetivo propuesto al inicio del mismo el cual consistió en desarrollar las pruebas de liberación a BioSyS validando así el cumplimiento de los requerimientos de software establecidos. Con esto se logró un software con un mínimo de errores, de mayor calidad y a su vez se arribó a las siguientes conclusiones:

- Se aplicaron pruebas a la documentación garantizando que lleguen al usuario final libre de errores.
- Se logró probar la aplicación utilizando diseños de casos de prueba basados en el método de caja negra
- Se realizó la prueba de sistema de carga y estrés, además se hizo una evaluación de los atributos de calidad haciendo uso de una lista de chequeos.
- El proceso de prueba fue debidamente documentado en este trabajo así como en los documentos adjuntos a él.

## **Recomendaciones**

Luego de aplicar todo el proceso de pruebas al producto BioSyS del proyecto Software para la Simulación y Análisis de Sistemas Biológicos y demostrar la importancia que tiene la ejecución de las mismas en el desarrollo de software, se considera que todo esfuerzo por mejorar la calidad de un producto es poco por lo que se recomienda:

- Dar continuidad al proceso de pruebas aplicado al BioSyS ya que el sistema todavía presenta imprecisiones.
- Realizar otros tipos de pruebas al sistema como para que se pueda determinar claramente las fallas que ocasionalmente atacan al sistema.

## Referencias Bibliográficas

- [1] **ISO 9000**. Introducción a los Sistemas de Gestión de la Calidad ISO 9000, 2009. [Disponible en: <http://www.slideshare.net/jcfdezmxcal/introduccion-a-los-sistemas-de-gestion-de-la-calidad-iso-9000>]
- [2] **Pressman, Roger S**. Ingeniería del software. Un enfoque práctico. McGrawHill. 3ª Edición. 1993.
- [3] **IEEE**. IEEE Standard Glossary of Software Engineering Terminology. 1990.
- [4] **Cig\_Labs**. The Home of Groundbreaking Software Quality, 2002. [Disponible en: [http://www.cigitallabs.com/resources/definitions/software\\_testing.html](http://www.cigitallabs.com/resources/definitions/software_testing.html)]
- [5] -. Prueba de integración de software, 2008. [Disponible en: <http://vivenciasdiariaspropias.blogspot.com/2008/06/prueba-de-integracion-de-software.html>]
- [6] **Teruel, Alejandro**. Prueba repetibles y mantenibles, 2001.[Disponible en: <http://www ldc.usb.ve/~teruel/ci4713/clases2001/pruebasRep.html#bitacora>]
- [7] **Sarmiento, Alekxy y Cutiño, Elián**. LIMS de Calidad del Centro de Ingeniería Genética y Biotecnología: Análisis del Grupo de Recepción de Muestras. 2006.
- [8] **Shannon, Robert;Johannes, James D**. Systems simulation: the art and science. IEEE Transactions on Systems, Man and Cybernetics. (1976).
- [9] **ISO 9000**. NORMA INTERNACIONAL ISO (3.1.2), 2005.[Disponible en: [http://www.uaemex.mx/SGCUAEMex/pdf/normaiso9000\\_2005.pdf](http://www.uaemex.mx/SGCUAEMex/pdf/normaiso9000_2005.pdf)]
- [10] **Ljungberg, A**. International Journal of Physical Distribution & Logistics Management Vol.32 No4, 1994
- [11] **Ferry, César**. WEKA. [Disponible en: <http://users.dsic.upv.es/~cferri/weka/>]

## Bibliografía

- **Alvarado Morgan, Edgar M.** Pruebas Unitarias (Unit Testing), 2008. [Disponible en: [http://www.itbuilder.com.mx/blogs/edgar.alvarado/post/Niveles-de-Prueba-\(Levels-of-Testing\).aspx](http://www.itbuilder.com.mx/blogs/edgar.alvarado/post/Niveles-de-Prueba-(Levels-of-Testing).aspx)].
- **Collado, Manuel.** Pruebas de software. Técnicas de prueba del software. Estrategias de prueba del software, 2003. [Disponible en: [http://74.125.47.132/search?q=cache:FVfTzTB\\_kusJ:lmf.fi.upm.es/ftp/ed2/0203/Apuntes/pruebas.ppt+pruebas+de+integraci%C3%B3n+de+software&cd=8&hl=es&ct=clnk&gl=cu](http://74.125.47.132/search?q=cache:FVfTzTB_kusJ:lmf.fi.upm.es/ftp/ed2/0203/Apuntes/pruebas.ppt+pruebas+de+integraci%C3%B3n+de+software&cd=8&hl=es&ct=clnk&gl=cu)].
- **Cueva Lovelle, Juan Manuel.** Calidad del Software, 1999. [Disponible en: [http://74.125.45.132/search?q=cache:ffgcn5YnftoJ:gidis.ing.unlpam.edu.ar/downloads/pdfs/Calidad\\_software.PDF+calidad+de+software&hl=es&ct=clnk&cd=2&gl=cu](http://74.125.45.132/search?q=cache:ffgcn5YnftoJ:gidis.ing.unlpam.edu.ar/downloads/pdfs/Calidad_software.PDF+calidad+de+software&hl=es&ct=clnk&cd=2&gl=cu)].
- **C Grimán, Anna. Pérez, María y E Mendoza, Luis.** Estrategia de Pruebas para Software OO que garantiza Requerimientos No Funcionales. [Disponible en: [http://www.lisi.usb.ve/publicaciones/03%20evaluacion/evaluacion\\_15.pdf](http://www.lisi.usb.ve/publicaciones/03%20evaluacion/evaluacion_15.pdf)]
- **Dr. Polo Usaola, Macario.** Pruebas de Software. [Disponible en: <http://74.125.45.132/search?q=cache:2GbbS7Hj2jcJ:alarcos.infcr.uclm.es/doc/masi/doc/lec/parte5/polo-apuntesp5.pdf+pruebas+en+software&hl=es&ct=clnk&cd=7&gl=cu>].
- **Fernández Carrasco, Oscar M. García León, Delba y Beltrán Benavides, Alfa.** Un enfoque actual sobre la calidad del software, 1995.[Disponible en: <http://eprints.rclis.org/archive/00002231/01/aci05395.htm>].
- **Fernández Pons, Yanet.** Pruebas de Aceptación del Cliente, -. [Disponible en: <http://www.monografias.com/trabajos36/pruebas-de-aceptacion/pruebas-de-aceptacion.shtml>].
- **Ferry, César.** WEKA. [Disponible en: <http://users.dsic.upv.es/~cferri/weka/>]
- **ISO 9000:2000.** Sistema de Gestión de la Calidad - Principios y Vocabulario, 2000.
- **ISO 9000.** NORMA INTERNACIONAL ISO (3.1.2), 2005.[Disponible en: [http://www.uaemex.mx/SGCUAE Mex/pdf/normaiso9000\\_2005.pdf](http://www.uaemex.mx/SGCUAE Mex/pdf/normaiso9000_2005.pdf)]
- **Ljungberg, A.** International Journal of Physical Distribution & Logistics Management Vol.32 No4, 1994

- **Lugo Mijares, Damarys.** Pruebas en Programación Orientada a Objetos, 1999. [Publicado en: <http://www ldc.usb.ve/~teruel/ci3711/test3/index.html>].
- **Mauricio.** DISEÑO DE CASOS DE PRUEBA, 2006.[Disponible en: <http://my.opera.com/pelican0/blog/show.dml/564829>]
- **Mañas, José A.** Calidad de las Pruebas, 2003. [Disponible en: <http://www.lab.dit.upm.es/~lprg/material/apuntes/pruebas/calidad.htm>].
- **MCH.** Factores de la calidad del software, 2008. [Disponible en: <http://ckernel.blogspot.com/2008/02/factores-de-la-calidad-del-software.html>].
- **Méndez, Carlos.** Metodo de Pruebas Orientada a Objetos para el Ciclo de Vida Completo (FLOOT), 2007. [Disponible en: <http://www.ambyssoft.com/essays/flootSpanish.html>]
- **Noriega Quintana, Darcy Javier.** Calidad de Software, 1997.[Disponible en: <http://www.monografias.com/trabajos59/calidad-software/calidad-software.shtml>].
- **Raja Prado, Elena.** Casi todas las pruebas del software, 2007. [Disponible en: <http://www.sistedes.es/TJISBD/Vol-1/No-4/articulos/pris-07-raja-ctps.pdf>].
- **Rojas, Johanna y Barrios, Emilio.** Investigación sobre estado del arte en diseño y aplicación de pruebas de software, 2007. [Disponible en: <http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node28.html>].
- **Sarmiento, Aliksy y Cutiño, Elián.** LIMS de Calidad del Centro de Ingeniería Genética y Biotecnología: Análisis del Grupo de Recepción de Muestras. 2006.
- **Shannon, Robert;Johannes, James D.** Systems simulation: the art and science. IEEE Transactions on Systems, Man and Cybernetics. (1976).
- **Teruel, Alejandro.** Pruebas de sistemas orientados a objetos, 2001.[Disponible en: <http://www ldc.usb.ve/~teruel/ci3715/clases/testing2.html>]
- **Universidad Rey Juan Carlos.** Calidad de Software. [Disponible en: <http://74.125.45.132/search?q=cache:yeKzx8g4MLUJ:kybele.escet.urjc.es/Documentos/ISI/Calida>

d%2520del%2520software.ppt+aseguramiento+de+lacalidad+de+software&hl=es&ct=clnk&cd=5&gl=cu].

- -. Diseños de casos de pruebas, 2005. [Disponible en: <http://pruebasoftware.blogcindario.com/2005/10/00002-disenos-de-casos-de-prueba.html>].
- -. La prueba del software. [Disponible en: [http://eisc.univalle.edu.co/materias/Material\\_Desarrollo\\_Software/Pruebas.pdf](http://eisc.univalle.edu.co/materias/Material_Desarrollo_Software/Pruebas.pdf)].
- -. Open UP. [Disponible en: <http://www.openup.es/>].
- -. Prueba de integración de software, 2008. [Disponible en: <http://vivenciasdiariaspropias.blogspot.com/2008/06/prueba-de-integracin-de-software.html>].

## Anexos

**Anexo #1 Lista de chequeo Especificación de Requisitos**

Estructura del documento					
Peso	Indicadores a Evaluar	Eval	(NP)	Cantidad de elementos afectados	Comentarios
crítico	1. ¿Está el documento acorde con a la plantilla estándar del proyecto o del expediente de proyecto?				
crítico	2. ¿Contiene las secciones obligatorias definidas en el expediente? (Ver Expediente de Proyecto)				
Elementos definidos por la metodología					
Peso	Indicadores a Evaluar	Eval	(NP)	Cantidad de elementos afectados	Comentarios
crítico	○ ¿Están todos los requisitos redactados de forma simple y clara para aquellos que vayan a consultarlo en un futuro?				
	○ ¿Debería especificarse algún requisito con más detalle?				

	<ul style="list-style-type: none"> <li>○ ¿Debería especificarse algún requisito con menos detalles?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Todos los requisitos identificados se centran en lo que el sistema debe hacer y no como el sistema debe hacerlo?</li> </ul>				
crítico	<ul style="list-style-type: none"> <li>○ ¿Han sido abordadas e identificadas los valores de entradas y salidas?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Han sido incluidos las respuestas válidas y no válidas de los valores de entrada?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Se han identificado los requerimientos de software y de hardware?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Han sido identificadas las restricciones de diseño e implementación?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Han sido identificadas las restricciones de interfaz externa?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Los requerimientos de soporte y usabilidad se han identificados?</li> </ul>				

	<ul style="list-style-type: none"> <li>○ ¿Se han identificado los requerimientos de seguridad (confidencialidad, integridad, disponibilidad)?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Se puede verificar cada requisito? (Un requisito se dice que es verificable si existe algún proceso no excesivamente costoso por el cual una persona o una máquina pueda chequear que el software satisface dicho requerimiento, ejemplo la especificación del caso de uso).</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Se han enumerado los requisitos incluso los que se derivan de otros requisitos?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Se puede trazar cada requisito al origen en el entorno del problema, (caso de uso del negocio)?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Se han especificado todos los posibles cambios en los requisitos,</li> </ul>				

	incluyendo la probabilidad de cambio?				
	<ul style="list-style-type: none"> <li>○ ¿No aparece un mismo requisito en más de un lugar del documento de especificación?</li> </ul>				
crítico	<ul style="list-style-type: none"> <li>○ ¿No existe contradicción entre lo especificado por un requisito y lo especificado por otro?</li> </ul>				
	<ul style="list-style-type: none"> <li>○ ¿Existe correspondencia entre el modelo de caso de uso, las Especificaciones Suplementarias y las especificaciones de requerimientos?</li> </ul>				
<b>Semántica del documento</b>					
<b>Peso</b>	<b>Indicadores a Evaluar</b>	<b>Eval</b>	<b>(NP)</b>	<b>Cantidad de elementos afectados</b>	<b>Comentarios</b>
Crítico	<ul style="list-style-type: none"> <li>● ¿Ha identificado errores ortográficos?</li> </ul>				
Crítico	<ul style="list-style-type: none"> <li>● ¿Se entiende claramente lo que se ha especificado en el documento?</li> </ul>				
	<ul style="list-style-type: none"> <li>● ¿El número de página que aparece en el índice coincide con el contenido que se refleja realmente en dicha página?</li> </ul>				

	<ul style="list-style-type: none"><li>• ¿El total de páginas que aparecen en las reglas de confidencialidad coincide con el total de páginas que tiene el documento?</li></ul>				
--	--	--	--	--	--

**Anexo #2 Lista de chequeo Especificación de Casos de Uso**

<b>Estructura del documento</b>					
<b>Peso</b>	<b>Indicadores a Evaluar</b>	<b>Eval</b>	<b>(NP)</b>	<b>Cantidad de elementos afectados</b>	<b>Comentarios</b>
crítico	1. ¿Está el documento acorde con a la plantilla estándar del proyecto o del expediente de proyecto?				
crítico	2. ¿Contiene las secciones obligatorias definidas en el expediente? (Ver Expediente de Proyecto)				
<b>Elementos definidos por la metodología</b>					
<b>Peso</b>	<b>Indicadores a Evaluar</b>	<b>Eval</b>	<b>(NP)</b>	<b>Cantidad de elementos afectados</b>	<b>Comentarios</b>
<b>Aspectos Generales</b>					
	1. ¿Cada caso de uso registra claramente lo que el sistema debe hacer?				
crítico	2. ¿Están clasificado los casos de uso que definen la arquitectura básica del sistema?				
	3. ¿Se ha identificado los casos de uso que darán soporte y mantenimiento al sistema?				

crítico	4. ¿Se ha descrito con precisión todas las alternativas o excepciones?				
	5. ¿Están clasificados los casos de uso que sirven de apoyo a los caso de uso que cubren las principales funciones que el sistema debe realizar?				
	6. ¿Están clasificados los casos de uso que no son claves para la arquitectura?				
<b>Nombre del Caso de Uso</b>					
crítico	1. ¿Está en infinitivo y refleja de manera clara el objetivo del usuario sobre el sistema?				
crítico	2. ¿El nombre del caso de uso es único?				
	3. ¿El nombre del caso de uso es intuitivo?				
<b>Descripción</b>					
	1. ¿El resumen dice como se inicia, como termina y las operaciones principales que realiza el caso de uso?				
<b>Precondición</b>					

	1. ¿Se escribe una precondición si y solo si a partir de la ocurrencia de un suceso determinado comienza el caso de uso?				
	2. ¿La precondición es válida tanto para flujos básicos como flujos alternativos?				
<b>Poscondición</b>					
	1. ¿La poscondición plasma cambios que suceden en el sistema al terminarse de ejecutar el caso de uso?				
<b>Complejidad del CU</b>					
	1. ¿Se especifica la complejidad del caso de uso?				
<b>Forma de presentar la información</b>					
crítico	1. ¿Está descrito el caso de uso en presente?				
crítico	2. ¿Se describe de manera comprensible y detallada las acciones del actor frente al sistema? ¿Está lo más parecido a un manual de ayuda?				
<b>Actores del CU</b>					
crítico	1. ¿El Caso de Uso está relacionado con al menos				

	un actor?				
crítico	2. ¿Si hay dos actores interactuando con el caso de uso está generalizado en uno solo?				
crítico	3. ¿Si el caso de uso es abstracto (include, extend, generalización-especialización), no lo inicializa ningún actor?				
<b>Flujo Básico</b>					
	1. ¿Comienza diciendo “El caso de uso se inicia cuando el actor...”?				
	2. ¿Termina diciendo en un evento independiente “El caso de uso termina”?				
	3. ¿No existen abreviaturas?				
crítico	4. ¿Las partes del flujo de eventos que se repiten en otro caso de uso se especifican como un Caso de Uso incluido?				
crítico	5. ¿Si las alternativas que se describen casi nunca ocurren o son alternativas comunes a otros casos de uso se especifican como un Caso de Uso extendido?				
	6. ¿Si existe un proceso general y a partir de él se				

	especializan otros se especifican como una generalización/especialización?				
<b>Flujo Alterno</b>					
crítico	1. ¿Las alternativas o excepciones se reflejan como flujos alternos?				
	2. ¿En todos los CU que se introducen datos tienen un flujo alternativo donde el sistema valida la integridad de los datos que se introducen y muestra un mensaje en caso de que los datos estén incompletos?				
	3. ¿Los flujos alternativos se nombran con el número del paso que lo generó en el flujo básico, una letra, ordenados alfabéticamente que lo produjo?				
crítico	4. ¿En la sección flujos alternativos se describen todas las excepciones que existan por muy evidentes que parezcan?				
<b>Casos de uso incluidos y extendidos</b>					
crítico	1. ¿Al describir el caso de uso base se mencionan				

	todos los casos de Uso que Extienden, se incluyen o se generalizan del Caso de Uso?				
crítico	2. ¿La descripción de los Casos de Uso incluidos, extendidos y especializados se realiza aparte?				
<b>Navegabilidad</b>					
crítico	1. ¿La navegabilidad en los caso de uso de inclusión se inicia desde el caso uso base hasta el caso de uso incluido?				
crítico	2. ¿La navegabilidad en los caso de uso de extensión se inicia desde el caso uso extendido hasta el caso de uso base?				
	3. ¿La navegabilidad en la generalización/especialización se inicia desde el caso de uso especializado a al generalizado y se representa con una relación de herencia?				
<b>Relaciones</b>					
crítico	1. ¿Las relaciones de inclusión y extensión entre los caso de uso se han				

	representado con línea discontinua?				
<b>Información General</b>					
crítico	1. ¿El diagrama de casos de uso expresa en detalles y claramente lo que debe hacer el sistema?				
	2. Si la modelación de las interacciones con el sistema es muy extensa ¿ha empleado los paquetes de caso de uso?				
<b>Semántica del documento</b>					
<b>Peso</b>	<b>Indicadores a Evaluar</b>	<b>Eval</b>	<b>(NP)</b>	<b>Cantidad de elementos afectados</b>	<b>Comentarios</b>
Crítico	1. ¿Ha identificado errores ortográficos?				
Crítico	2. ¿Se entiende claramente lo que se ha especificado en el documento?				
	3. ¿El número de página que aparece en el índice coincide con el contenido que se refleja realmente en dicha página?				
	4. ¿El total de páginas que aparecen en las reglas de confidencialidad coincide con el total de páginas que tiene el				

	documento?				
--	------------	--	--	--	--

### **Anexo # 3 Especificación de Requisitos**

Funcionalidades básicas

**RB 1** Gestionar la información del sistema biológico que se esté estudiando.

**RB.1.1** Insertar sistema biológico a estudiar.

**RB.1.2** Seleccionar sistema biológico existente.

**RB.1.3** Eliminar un sistema biológico existente.

**RB.1.4** Modificar sistema biológico existente.

**RB 2** Gestionar modelo matemático.

**RB 2.1** Insertar el modelo matemático.

**RB 2.2** Eliminar el modelo matemático.

**RB 2.3** Modificar el modelo matemático.

**RB 2.4** Eliminar simulaciones

**RB 2.5** Mostrar información del modelo.

**RB 3** Gestionar copia de valores de las tablas

**RB 3.1** Copiar valores de las tablas.

**RB 3.2** Pegar copia de valores de las tablas.

**RB 3.3** Eliminar copia de valores de las tablas.

**RB 4** Gestionar conexión.

**RB.4.1** Realizar conexión.

**RB.4.2** Desconectar de la base de dato.

Funcionalidades del Graficador:

**RG 1.** Permitir explorar las simulaciones

**RG.2.** Gestionar propiedades del gráfico

**RG 2.1.** Cambio de las variables a graficar

**RG 2.2.** Gestionar Anotaciones**RG 2.2.1** Insertar nueva anotación.**RG 2.2.2** Editar anotación.**RG 2.2.3** Eliminar anotación.**RG 2.3.** Modificar propiedades de la escala**RG 2.4.** Cambiar color de las variables.**RG 2.5** Salvar imagen de la gráfica.**RG 3.** Clasificación.**RG 4.** Insertar valores para una nueva simulación.**RG 5.** Copiar valores de la gráfica en un tiempo específico.**RG 6.** Eliminar Simulación

## Funcionalidades del Editor de Ecuaciones

**RE 1.** Cargar Modelo matemático**RE 2.** Salvar Modelo Matemático**RE 3.** Gestionar SED**RE 3.1** Adicionar nuevas ecuaciones.**RE 3.2** Insertar literales o constantes numéricas.**RE 3.3** Insertar operaciones matemáticas.**RE 3.4** Insertar fórmulas desde biblioteca de expresiones.**RE 3.5.** Eliminar elementos del SED.**RE 4.** Gestionar la biblioteca de expresiones**RE 4.1** Insertar nuevas categorías a la biblioteca de expresiones.**RE 4.2** Eliminar categorías de la biblioteca de expresiones.**RE 4.3** Insertar nuevos elementos a la biblioteca de expresiones.**RE 4.4** Eliminar elementos contenidos en la biblioteca de expresiones.**RE 5.** Exportar SED a diferentes formatos.**RE 5.1** Exportar a MathML.**RE 5.2** Exportar a MATLAB.**RE 6.** Analizar la homogeneidad dimensional del SED.**RE 6.1** Insertar nuevas dimensiones al sistema.**RE 6.2** Eliminar dimensiones del sistema.**RE 6.3** Asociar variables del SED con dimensiones.

Funcionalidades del Simulador

**RS 1.** Editar Preferencias

**RS 2.** Definir valores necesarios para realizar simulaciones.

**RS 3.** Gestionar Simulación.

**RS 3.1** Realizar Simulación.

**RS 3.1.1** Realizar simulaciones locales.

**RS 3.1.2** Realizar simulaciones distribuidas.

**RS 3.1.3** Realizar Simulación utilizando la librería ODEtoJava.

**RS 3.1.4** Realizar Simulación utilizando el asistente matemático MatLab.

**RS 3.2** Chequear estado de las ejecuciones.

**RS 3.3** Detener las ejecuciones.

**RS 4.** Autenticar en la Grid.

Funcionalidades del Módulo de Análisis

**RA1.** Mostrar Dinámicas Aleatorias.

**RA2.** Realizar Clusters.

**RA2.1** Cargar las simulaciones de la BD.

**RA2.2** Agrupar simulaciones.

**RA2.3** Mostrar gráfica de Clusters.

**RA2.4** Guardar fichero del resultado de los Clusters.

**RA2.5** Mostrar dinámicas de una simulación dentro de un cluster.

**RA3.** Realizar Clasificaciones.

**RA3.1** Cargar simulaciones clasificadas.

**RA3.1.1** Generar modelo.

**RA3.1.2** Mostrar resultado en forma de árbol.

**RA3.2** Clasificar simulaciones a partir de un modelo.

**RA4.** Realizar Análisis por reglas

**RA4.1** Definir valores necesarios para realizar el análisis.

**RA4.2** Editar reglas.

**RA4.3** Realizar análisis.

**RA4.3.1** Realizar análisis con un grupo de nuevas reglas.

**RA 4.3.2** Realizar análisis con un grupo de reglas inferidas a partir de simulaciones ya clasificadas.

**RA5.** Realizar Análisis de Bifurcaciones.

#### **Anexo # 4: Manual de pruebas para carga y estrés**

##### ***Pruebas de estrés y carga a la aplicación BioSyS***

###### ***Objetivos:***

Probar que todos se conecten a la misma base de datos.

Probar que las principales funcionalidades aguanten usos simultáneos desde diferentes estaciones de trabajo.

###### ***Requisitos para hacer la prueba:***

Debe estar instalada en la PC la máquina virtual de java

###### ***Funcionalidades principales:***

###### ***Módulo de Funcionalidades comunes:***

- Gestionar conexión
- Gestionar Sistema Biológico.
- Gestionar Modelo Matemático.

###### ***Módulo Editor de Ecuaciones:***

- Cargar Modelo Matemático.
- Salvar Modelo Matemático.
- Gestionar Edición SED.
- Exportar SED.

###### ***Módulo Simulador:***

- Gestionar Simulación.
- Autenticar en la Grid.

###### ***Módulo Análisis.***

- Realizar Clusters.

- Realizar Análisis de Clasificación.
- Realizar Análisis de Bifurcaciones.
- Mostrar Dinámicas Aleatorias

### **Primera Etapa de Pruebas.**

#### **Prueba de carga:**

##### **Paso 1:**

En la capeta de la aplicación ir al archivo BioSys\_1.0.jar, y dar doble clic sobre él. Loguearse en la aplicación con los datos siguientes:

DB"Server: PostgreSQL.

Server IP: 10.7.19.202

User: biosys

DB' name: biosys

Port: 5900

Password: se da a la hora de la prueba

##### **Paso 2:**

Presionar el botón Connect cuando se de la orden.

#### **Prueba de estrés:**

##### **Paso 1:**

Dar clic derecho sobre el icono que representa la conexión a la base de datos, escoger la opción New System, entrar el nombre y la descripción y presionar botón Insert cuando se de la orden.

##### **Paso 2:**

Dar clic derecho sobre el icono que representa el sistema insertado y escoger la opción New Model.

##### **Paso 3:**

Insertar dos ecuaciones en el menú System/Equations/New differential equation. Ponerle los siguientes valores:

$$dx / dt = ax - bxy$$

$$dy / dt = - cy + dxy$$

**Paso 4:**

Ir al menú File/Export to.../Database y seleccionar como variables las letras **x** y **y**, el resto seleccionarlas como parámetros y presionar el botón Insert cuando se de la orden.

**Paso 5:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Simulate Model e insertar los valores de la tabla siguiente (solo los valores que aparecen con el fondo blanco):

Tabla 1					
	Initial Value	Vary	Final Value	No of po...	Variation
<b>x</b>	1	Marcar	4	10	Lineal
<b>y</b>	1	No marcar			Lineal
Tabla 2					
<b>a</b>	1	Marcar	4	10	Lineal
<b>t</b>	1	No marcar			Lineal
<b>d</b>	1	No marcar			Lineal
<b>c</b>	1	No marcar			Lineal
<b>b</b>	1	No marcar			Lineal

Initial Time	Final Time	Type of Simulation
1	10	Local

**Paso 6:**

Presionar el botón Run (Botón verde que se encuentra en la parte superior izquierda) o las teclas Control + R cuando se de la orden.

**Paso 7:**

Repetir en paso 5 pero cambiando el tipo de simulación (Type of Simulation) por la simulación Distributed y repetir el paso 6.

**Paso 8:**

En la ventana de autenticación que se muestra poner los siguientes datos.

User: edel

Password: se especifica a la hora de la prueba.

**Paso 9:**

Dar clic derecho sobre el modelo creado y seleccionar la opción Information cuando se de la orden.

**Paso 10:**

Dar clic derecho sobre el modelo y seleccionar la opción Analysis/Show dynamic random cuando se de la orden.

**Paso 11:**

Dar clic derecho sobre el modelo, seleccionar la opción Analysis/Analysis of Clusters y llenar solo los datos siguientes.

Time to be analysed	Type of analysis
1	All simulations

**Paso 12:**

Combinar las teclas Control + R cuando se de la orden, en la ventana que se muestra presionar el botón Start.

**Paso 13:**

En la ventana que se muestra a continuación presionar el botón Save y guardarlo en el escritorio.

**Paso 14:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Classyfi, entrar los datos de la tabla siguiente, seleccionar en el directorio el archivo que se guardó en el escritorio de los clusters .

Time to be analysed	Type of analysis
1	All simulations

Combinar las teclas Control +R cuando se de la orden.

**Paso 15:**

Dar clic derecho sobre el modelo, seleccionar la opción Analysis/Analysis of bifurcaciones.

**Paso 16:**

Entrar los datos de la tabla siguiente, solo los datos que están en blanco.

Tabla 1					
	Initial Value	Vary	Final Value	No of po...	Variation
<b>x</b>	1				Lineal
<b>y</b>	1				Lineal
T					
Tabla 2					
<b>a</b>	1	Marcar	2	9	Lineal
<b>t</b>	1	No marcar			Lineal
<b>d</b>	1	No marcar			Lineal
<b>c</b>	1	No marcar			Lineal
<b>b</b>	1	No marcar			Lineal

Initial Time	Final Time	Type of Simulation
1	10	Local

**Paso 17:**

Presionar las teclas Control + R cuando se de la orden.

**Paso 18:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Delete Simulations, en la ventana que se muestra a continuación seleccionar la opción All simulations y presionar el botón Delete Simulations y luego seleccionar Yes cuando se de la orden en la ventana de confirmación.

**Paso 19:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Delete Model y luego en la ventana de confirmación presionar el botón Yes.

**Paso 20:**

Cerrar la aplicación

**Segunda Etapa de Pruebas.****Indicaciones:**

Separarse en dos grupos con igual número de integrantes y desarrollar las pruebas cuando se le indique. Los dos equipos van a desarrollar la misma prueba pero con una diferencia de tiempo

**Prueba de carga:****Paso 1:**

En la capeta de la aplicación ir al archivo BioSys\_1.0.jar, y dar doble clic sobre él. Loguearse en la aplicación con los datos siguientes:

DB"Server: PostgreSQL.

Server IP: 10.7.19.202

User: biosys

DB' name: biosys

Port: 5900

Password: se da a la hora de la prueba

**Paso 2:**

Presionar el botón Connect cuando se de la orden.

**Prueba de estrés:****Paso 1:**

Dar clic derecho sobre el icono que representa la conexión a la base de datos, escoger la opción New System, entrar el nombre y la descripción y presionar botón Insert cuando se de la orden.

**Paso 2:**

Dar clic derecho sobre el icono que representa el sistema insertado y escoger la opción New Model.

**Paso 3:**

Insertar dos ecuaciones en el menú System/Equations/New differential equation. Ponerle los siguientes valores:

$$dx / dt = -ax + bxy$$

$$dy / dt = cy + dxy$$

**Paso 4:**

Ir al menú File/Export to.../Database y seleccionar como variables las letras **x** y **y**, el resto seleccionarlas como parámetros y presionar el botón Insert cuando se de la orden.

**Paso 5:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Simulate Model e insertar los valores de la tabla siguiente (solo los valores que aparecen con el fondo blanco):

Tabla 1					
	Initial Value	Vary	Final Value	No of po...	Variation
<b>x</b>	3	Marcar	6	10	Lineal
<b>y</b>	3	No marcar			Lineal
Tabla 2					
<b>a</b>	3	Marcar	6	10	Lineal
<b>t</b>	3	No marcar			Lineal

<b>d</b>	3	No marcar			Lineal
<b>c</b>	3	No marcar			Lineal
<b>b</b>	3	No marcar			Lineal

Initial Time	Final Time	Type of Simulation
1	13	Local

**Paso 6:**

Presionar el botón Run (Botón verde que se encuentra en la parte superior izquierda) o las teclas Control + R cuando se de la orden.

**Paso 7:**

Repetir en paso 5 pero cambiando el tipo de simulación (Type of Simulation) por la simulación Distributed y repetir el paso 6.

**Paso 8:**

En la ventana de autenticación que se muestra poner los siguientes datos.

User: edel

Password: se especifica a la hora de la prueba.

**Paso 9:**

Dar clic derecho sobre el modelo creado y seleccionar la opción Information cuando se de la orden.

**Paso 10:**

Dar clic derecho sobre el modelo y seleccionar la opción Analysis/Show dynamic random cuando se de la orden.

**Paso 11:**

Dar clic derecho sobre el modelo, seleccionar la opción Analysis/Analysis of Clusters y llenar solo los datos siguientes.

Time to be analysed	Type of analysis
1	All simulations

**Paso 12:**

Combinar las teclas Control + R cuando se de la orden, en la ventana que se muestra presionar el botón Start.

**Paso 13:**

En la ventana que se muestra a continuación presionar el botón Save y guardarlo en el escritorio.

**Paso 14:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Classyfi, entrar los datos de la tabla siguiente, seleccionar en el directorio el archivo que se guardó en el escritorio de los clusters .

Time to be analysed	Type of analysis
1	All simulations

Combinar las teclas Control +R cuando se de la orden.

**Paso 15:**

Dar clic derecho sobre el modelo, seleccionar la opción Analysis/Analysis of bifurcaciones.

**Paso 16:**

Entrar los datos de la tabla siguiente, solo los datos que están en blanco.

Tabla 1					
	Initial Value	Vary	Final Value	No of po...	Variation
<b>x</b>	1				Lineal
<b>y</b>	1				Lineal
Tabla 2					
<b>a</b>	1	Marcar	4	12	Lineal
<b>t</b>	1	No marcar			Lineal
<b>d</b>	1	No marcar			Lineal
<b>c</b>	1	No marcar			Lineal
<b>b</b>	1	No marcar			Lineal

Initial Time	Final Time	Type of Simulation
1	10	Local

**Paso 17:**

Presionar las teclas Control + R cuando se de la orden.

**Paso 18:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Delete Simulations, en la ventana que se muestra a continuación seleccionar la opción All simulations y presionar el botón Delete Simulations y luego seleccionar Yes cuando se de la orden en la ventana de confirmación.

**Paso 19:**

Dar clic derecho sobre el modelo creado, seleccionar la opción Delete Model y luego en la ventana de confirmación presionar el botón Yes.

**Paso 20:**

Cerrar la aplicación.

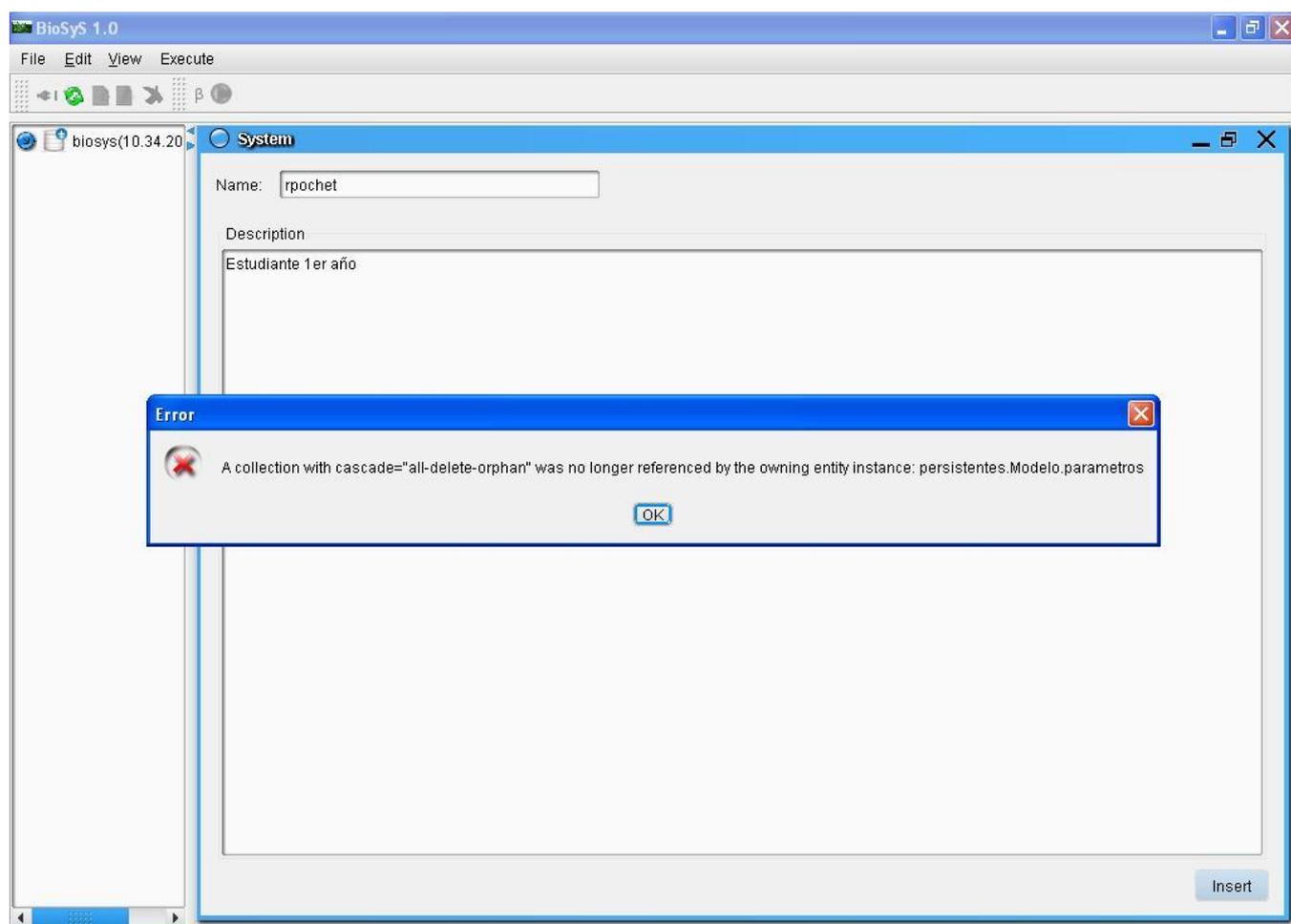
**Anexo #5 Lista de Chequeo de Atributos de Calidad**

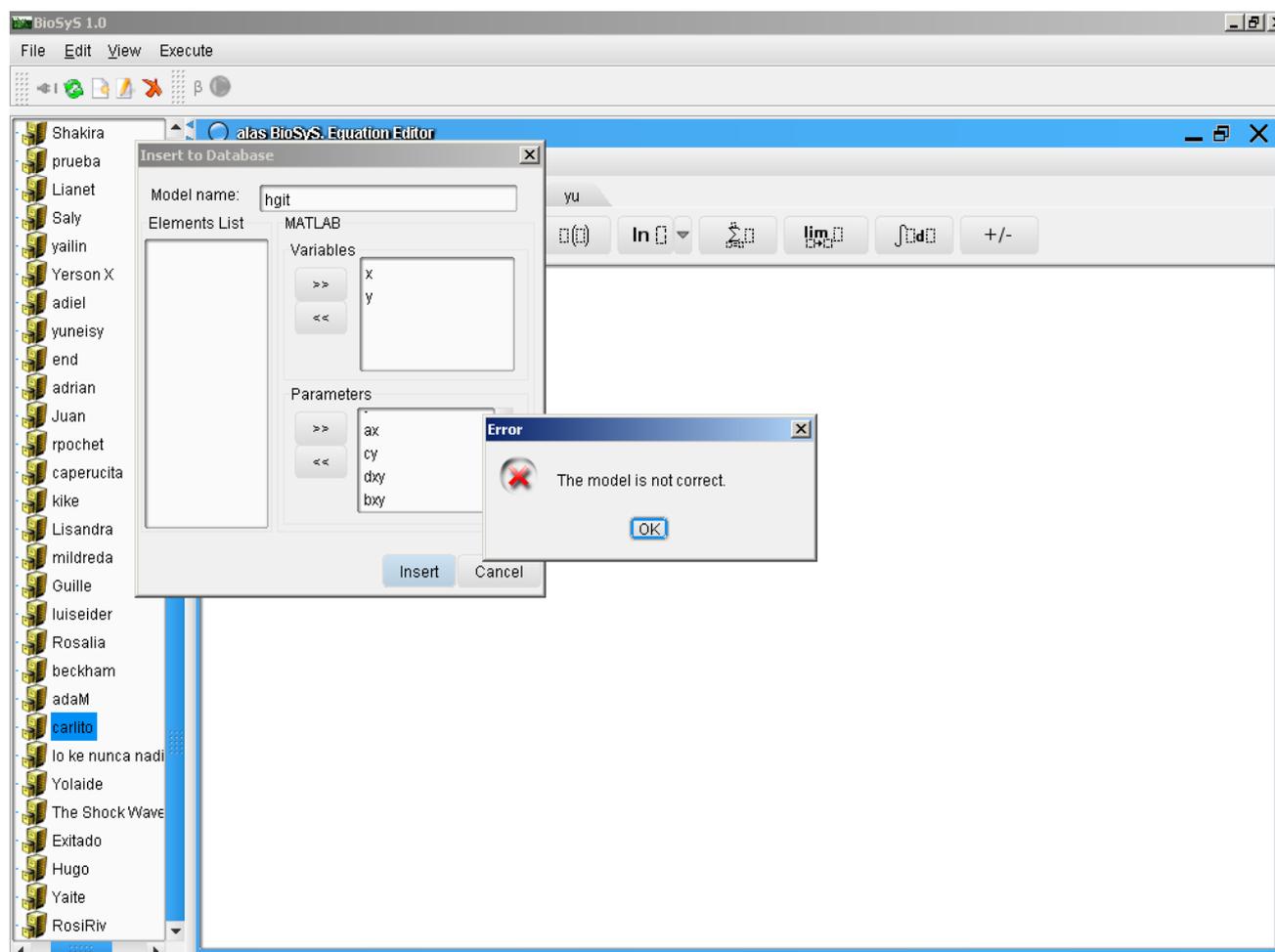
<b>Nivel</b>	<b>Evaluación</b>	<b>Eval</b>	<b>NP</b>	<b>Comentario</b>
!	¿Se consideran otros idiomas para las instrucciones en la pantalla de forma adecuada?			
!	¿Hay ausencia de términos en idiomas diferentes mezclados?			
!	¿Es simple el vocabulario utilizado?			
!	¿Se proporciona tiempo suficiente para realizar las entradas por teclado?			
!	¿Se posibilita la opción de incrementar el tamaño de los caracteres?			
!	¿Hay algún tipo de asistencia para los usuarios que hacen uso del sistema por primera vez?			
!	¿Resulta fácil instalar el software?			
!	¿Proporciona el sistema un soporte apropiado a los usuarios más novatos?			
!	¿Se entienden la interfaz y su contenido?			
!	¿Resulta fácil especificar un objeto o una acción?			
!	¿Resulta fácil entender el resultado de una acción?			
!	¿Asiste el sistema al usuario de forma efectiva proporcionando el modo más efectivo de hacer las tareas en caso de que no haya una única forma de hacerlas?			
!!	¿Está diseñada la interfaz para facilitar la realización eficiente de las tareas de la mejor forma posible?			
!	¿Es fácil de utilizar el sistema en la realización de tareas?			

!	¿Facilita la interfaz de usuario un uso eficiente (para interfaces basadas en pantallas)?			
!	¿Es apropiada la retro alimentación presentada por el sistema?			
!	¿Actúa el sistema como corrector de errores?			
!	¿Actúa el sistema en la prevención de errores?			
!!	¿Actúa el sistema en la información de los errores?			
!	¿Actúa el sistema en la corrección de errores?			
!	¿Se usan adecuadamente los modos de trabajo de los usuarios en el software?			
!	¿Se permite la utilización del ratón o el teclado?			
!	¿Permite al usuario interrumpir su tarea y continuar más tarde?			
!	¿Se utiliza mensajes y textos descriptivos?			
!!	¿Permite deshacer las acciones, e informar el estado?			
!!	¿Permite una cómoda navegación dentro del producto y una fácil salida de éste?			
!	¿Permite distintos niveles de uso del producto para usuarios con distintos niveles de experiencia?			
!	¿Se permite al usuario personalizar la interfaz?			
!	¿Se permite al usuario manipular directamente los objetos de la interfaz?			
!	¿Se reconoce todas las tareas del software en cualquier momento?			

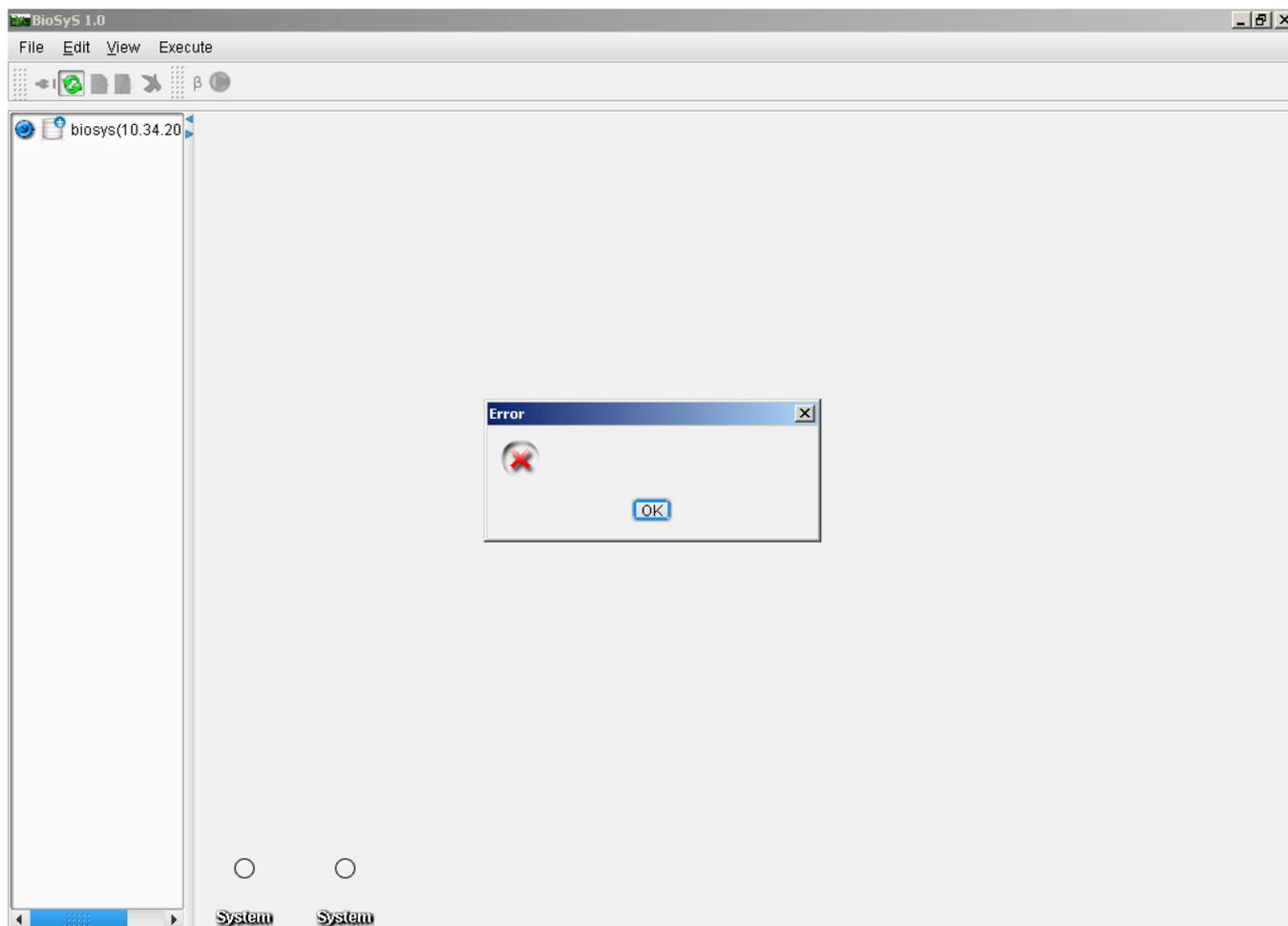
!!	¿Se proporciona información visual de dónde está el usuario, qué está haciendo y qué puede hacer a continuación?			
!	¿Proporciona funciones deshacer, rehacer?			
!	¿Existe atajos de teclado bien hechos?			
!	¿Se presenta al usuario la información que sólo necesita?			
!	¿Existen diferentes niveles de ayuda?			

#### **Anexo #6 Mensaje de error que se muestra al actualizar la aplicación (No siempre se muestra).**

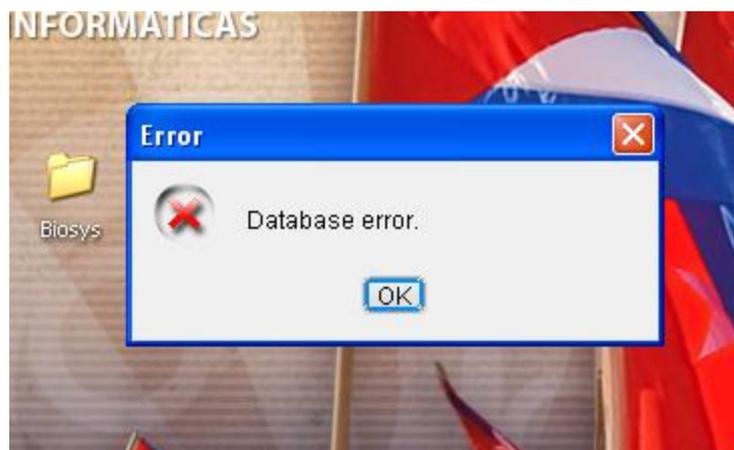


**Anexo #7 Mensaje de error que se muestra al insertar las ecuaciones en la base de datos.**

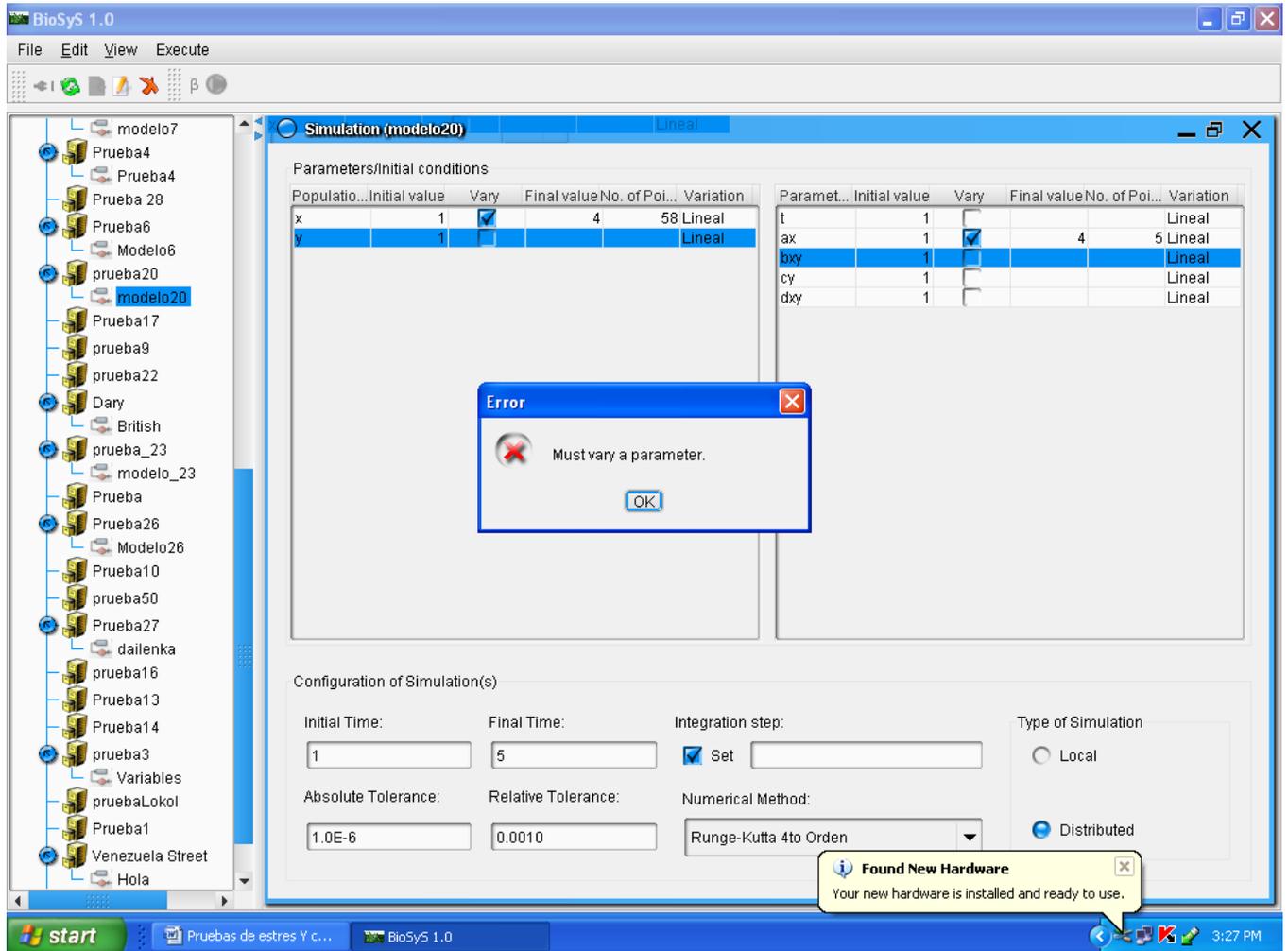
**Anexo #8 Mensaje que se muestra al actualizar la aplicación cuando está cargando.**



**Anexo #9 Mensaje de error que muestra la aplicación cuando rechazó la conexión a la Base de datos**



**Anexo #10 Mensaje de error que muestra la aplicación cuando hay un error en la conexión con la GRID**



**Anexo #1:****Prueba realizada a la primera consulta en sus tres iteraciones**

"SELECT resultados.Tiempo,resultados.Id\_Poblacion FROM resultados WHERE resultados.Id\_Simulacion = '106\_107\_117\_131\_137'"

**Para 1 conexión**

**Introducir datos**

**Cadena de conexión** sys;DSN=prueba;OPTION=0;PWD=lagardere;PORT=0;SERVER=localhost;UID=root

**Consulta** ltados.Id\_Poblacion from resultados where Id\_Simulacion='106\_107\_117\_131\_137'

**Conexiones** 1 **Peticiónes** 10

**Visualización de las conexiones**

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll

Tiempo: 281 ms

Simular Limpiar

**Para 20 conexiones**

**Introducir datos**

**Cadena de conexión** sys;DSN=prueba;OPTION=0;PwD=lagardere;PORT=0;SERVER=localhost;UID=root

**Consulta** cion FROM resultados \WHERE resultados.Id\_Simulacion = '106\_107\_117\_131\_137'

**Conexiones** 20 **Peticiónes** 10

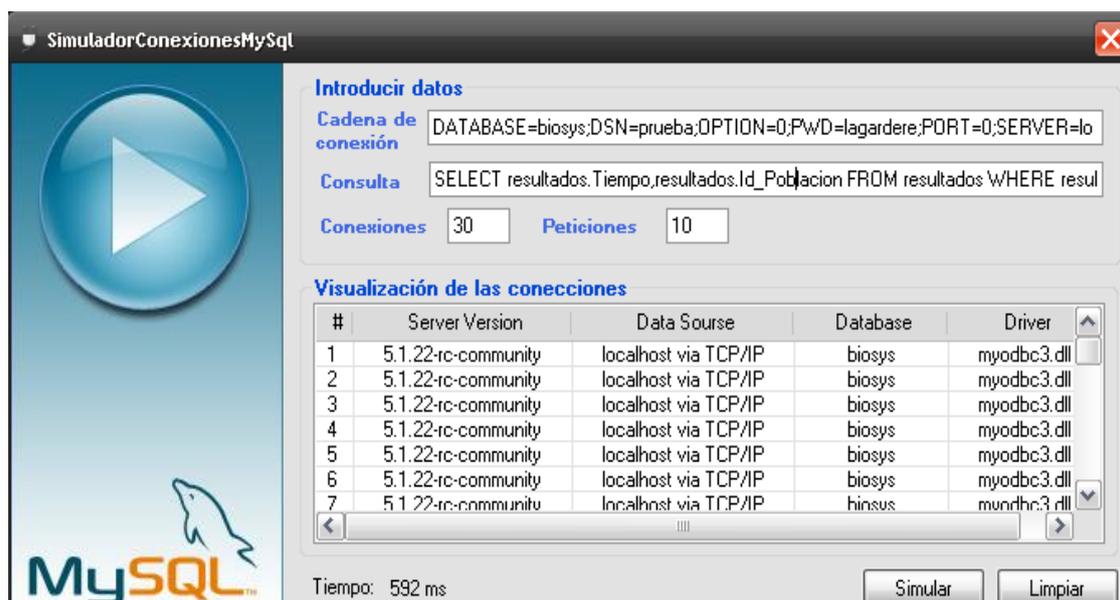
**Visualización de las conexiones**

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-community	localhost via TCP/IP	hinsus	mundhc3.dll

Tiempo: 303 ms

Simular Limpiar

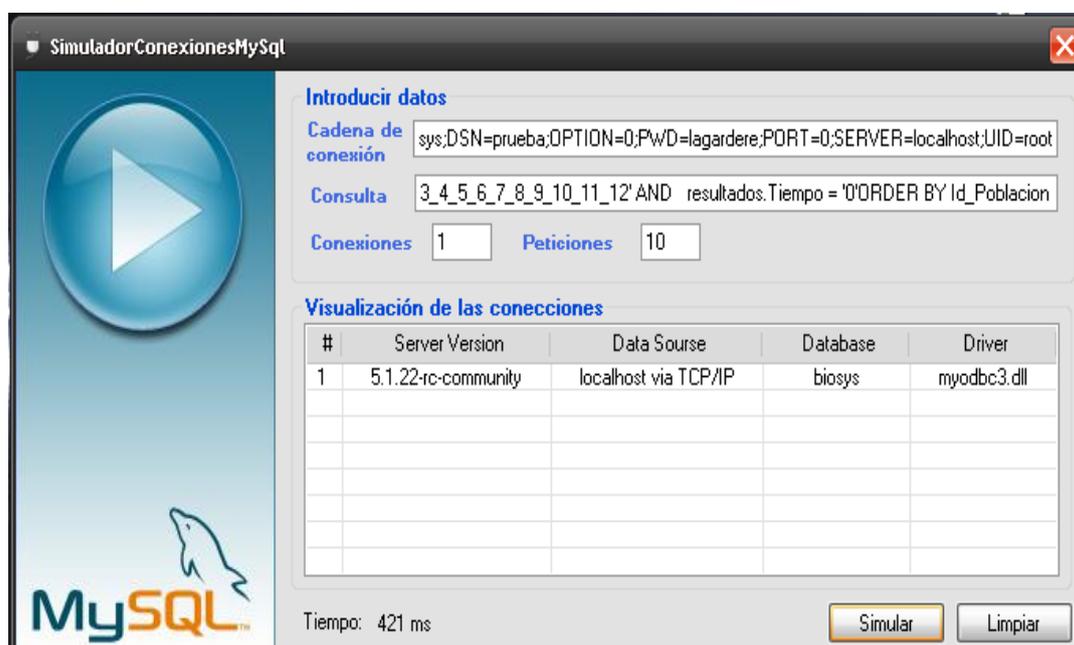
## Para 30 conexiones



## Prueba realizada a la segunda consulta en sus tres iteraciones

"SELECT resultados.Tiempo, resultados.Id\_Poblacion FROM resultados WHERE resultados.Id\_Simulacion = '1\_2\_3\_1\_2\_3\_4\_5\_6\_7\_8\_9\_10\_11\_12' AND resultados.Tiempo = '0'ORDER BY Id\_Poblacion"

## Para 1 conexión



## Para 20 conexiones

The screenshot shows the 'SimuladorConexionesMySQL' application window. On the left is a large blue play button and the MySQL logo. The main area is titled 'Introducir datos' and contains the following fields:

- Cadena de conexión:** DATABASE=biosys;DSN=prueba;OPTION=0;PWD=lagerdere;PORT=0;SERVER=lo
- Consulta:** SELECT resultados.Tiempo, resultados.Id\_Poblacion FROM resultados WHERE res
- Conexiones:** 20
- Peticiones:** 10

Below this is a section titled 'Visualización de las conexiones' containing a table with 7 rows and 5 columns:

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-community	localhost via TCP/IP	hinsus	mundhc3.dll

At the bottom, it shows 'Tiempo: 732 ms' and two buttons: 'Simular' and 'Limpiar'.

## Para 30 conexiones

The screenshot shows the 'SimuladorConexionesMySQL' application window. On the left is a large blue play button and the MySQL logo. The main area is titled 'Introducir datos' and contains the following fields:

- Cadena de conexión:** sys;DSN=prueba;OPTION=0;PWD=lagerdere;PORT=0;SERVER=localhost;UID=root
- Consulta:** 3\_4\_5\_6\_7\_8\_9\_10\_11\_12' AND resultados.Tiempo = '0'ORDER BY Id\_Poblacion
- Conexiones:** 30
- Peticiones:** 10

Below this is a section titled 'Visualización de las conexiones' containing a table with 7 rows and 5 columns:

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-community	localhost via TCP/IP	hinsus	mundhc3.dll

At the bottom, it shows 'Tiempo: 635 ms' and two buttons: 'Simular' and 'Limpiar'.

### Prueba realizada a la tercera consulta en sus tres iteraciones

*"select resultados.Id\_Poblacion, resultados.Tiempo from resultados where Id\_Simulacion='1\_2\_3\_1\_2\_3\_4\_5\_6\_7\_8\_9\_10\_11\_12' and Tiempo>=0 and Tiempo<=10 order by Tiempo,Id\_Poblacion"*

#### Para 1 conexión

The screenshot shows the 'SimuladorConexionesMySQL' application window. On the left is a large blue play button and the MySQL logo. The main area is titled 'Introducir datos' and contains the following fields:

- Cadena de conexión:** DATABASE=biosys;DSN=prueba;OPTION=0;PWD=lagardere;PORT=0;SERVER=lo
- Consulta:** select resultados.Id\_Poblacion, resultados.Tiempo from resultados where Id\_Simulaci
- Conexiones:** 1
- Peticiones:** 10

Below these fields is a table titled 'Visualización de las conexiones' with the following data:

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll

At the bottom, it shows 'Tiempo: 406 ms' and two buttons: 'Simular' and 'Limpiar'.

#### Para 20 conexiones

The screenshot shows the 'SimuladorConexionesMySQL' application window with the same configuration as above, but with 'Conexiones' set to 20. The 'Visualización de las conexiones' table now displays 7 rows:

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-community	localhost via TCP/IP	hinsus	mundhc3.dll

The 'Tiempo' at the bottom is now '723 ms'. The 'Simular' and 'Limpiar' buttons are also present.

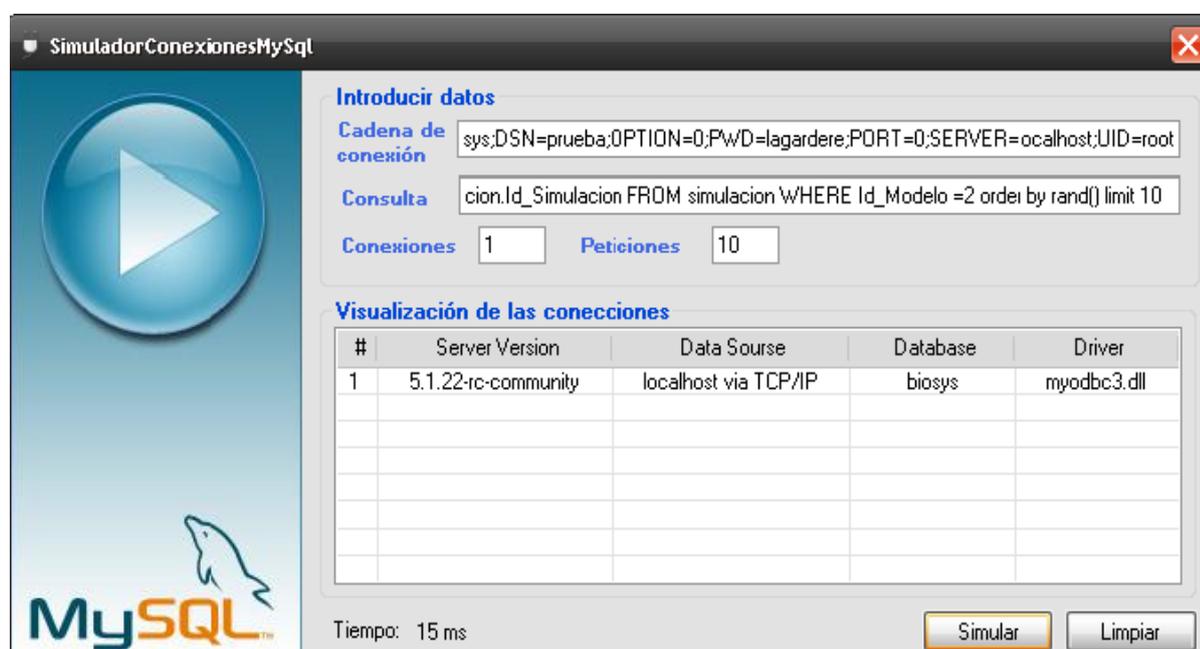
### Para 30 conexiones



### Prueba realizada a la cuarta consulta en sus tres iteraciones

"SELECT simulacion.Id\_Simulacion FROM simulacion WHERE Id\_Modelo =2 order by rand() limit 10"

### Para 1 conexión



## Para 20 conexiones

SimuladorConexionesMySQL

**Introducir datos**

**Cadena de conexión**

**Consulta**

**Conexiones**  **Peticiones**

**Visualización de las conexiones**

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-communitu	localhost via TCP/IP	hinsus	mundhc3.dll

Tiempo: 515 ms

## Para 30 conexiones

SimuladorConexionesMySQL

**Introducir datos**

**Cadena de conexión**

**Consulta**

**Conexiones**  **Peticiones**

**Visualización de las conexiones**

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-communitu	localhost via TCP/IP	hinsus	mundhc3.dll

Tiempo: 765 ms

**Prueba realizada a la quinta consulta en sus tres iteraciones**

"SELECT simulacion.Id\_Simulacion FROM simulacion WHERE Id\_Modelo =1 order by rand()  
LIMIT 1"

**Para 1 conexión**

The screenshot shows the 'SimuladorConexionesMySQL' application window. On the left is a large blue play button and the MySQL logo. The main area is titled 'Introducir datos' and contains the following fields:

- Cadena de conexión:** sys;DSN=prueba;OPTION=0;PWD=lagardere;PORT=0;SERVER=localhost;UID=root
- Consulta:** cion.Id\_Simulacion FROM simulacion WHERE Id\_Modelo =1 order by rand() LIMIT 1
- Conexiones:** 1
- Peticiones:** 10

Below this is a section titled 'Visualización de las conexiones' with a table:

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll

At the bottom, it shows 'Tiempo: 31 ms' and two buttons: 'Simular' and 'Limpiar'.

**Para 20 conexiones**

The screenshot shows the 'SimuladorConexionesMySQL' application window with the same configuration as above, but with 'Conexiones' set to 20. The 'Visualización de las conexiones' table now contains 20 rows:

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-community	localhost via TCP/IP	hinsus	mundhc3.dll

The 'Tiempo' at the bottom is now '609 ms'. The 'Simular' and 'Limpiar' buttons are still present.

## Para 30 conexiones

**Introducir datos**

**Cadena de conexión** `sys:DSN=prueba;OPTION=0;PWD=lagardere;PORT=0;SERVER=localhost;UID=root`

**Consulta** `cion.Id_Simulacion FROM simulacion WHERE Id_Modelo =1 order by rand() LIMIT 1`

**Conexiones**  **Peticiones**

**Visualización de las conexiones**

#	Server Version	Data Source	Database	Driver
1	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
2	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
3	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
4	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
5	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
6	5.1.22-rc-community	localhost via TCP/IP	biosys	myodbc3.dll
7	5.1.22-rc-community	localhost via TCP/IP	hinsus	myodbc3.dll

Tiempo: 906 ms

## Glosario de Términos

- **BD:** (Base de Datos). Colección de datos interrelacionados, variables en el tiempo, que pueden ser utilizados por uno o varios programas de aplicación.
- **Producto:** Es el resultado de una investigación al que finalmente se le establecen dos nombres: uno genérico y otro comercial. Está destinado al uso que su investigador haya propuesto y definido. [7]
- **Aplicación:** Es el programa informático diseñado como herramienta para permitir a un usuario realizar un o diversos tipos de trabajo.
- **Ecuaciones diferenciales:** Es la ecuación en la que intervienen derivadas de una o más funciones.
- **Simulación:** Es el proceso de diseñar un modelo de un sistema real y llevar a término experiencias con él, con la finalidad de comprender el comportamiento del sistema o evaluar nuevas estrategias dentro de los límites impuestos por un cierto criterio o un conjunto de ellos para el funcionamiento del sistema [8]
- **Inteligencia Artificial:** Es la rama de la ciencia informática dedicada al desarrollo de agentes racionales no vivos.
- **Biología de Sistemas:** Es una disciplina académica que pretende integrar diferentes niveles de información con el fin de entender cómo funcionan los sistemas biológicos.
- **No conformidades:** Es el incumplimiento de un requisito [9].
- **Proceso:** Un proceso es una red de actividades vinculadas ordenadamente las cuales se llevan a cabo repetidamente y que utilizan recursos e información para transformar insumos en productos abarcando desde el inicio del proceso hasta la satisfacción de las necesidades del cliente [10]
- **Iteración:** Repetición de un proceso.
- **CIM:** Centro de Inmunología Molecular.

- **ISO:** Organización Internacional para la Estandarización.
- **(IEEE):** Instituto de Ingenieros Eléctricos y Electrónicos. Esta es una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas.
- **Artefacto:** Es la obra manual realizada con un propósito o función técnica específica. En el caso de la ingeniería viene siendo todo el material tangible (Documentos, etc.) que se obtiene a lo largo de todo el proceso de trabajo de ingeniería.
- **Diagrama:** Representación gráfica en la que se muestran las relaciones entre las diferentes partes de un conjunto o sistema o los cambios de un determinado fenómeno.
- **Bucle:** Un bucle o ciclo, en programación, es una sentencia que se realiza repetidas veces a un trozo aislado de código, hasta que la condición asignada a dicho bucle deje de cumplirse.
- **BioSyS:** Software que fusiona algoritmos y herramientas capaces de facilitar y apoyar las investigaciones de los científicos en el área de la Biología de Sistemas.
- **MySQL:** Es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones.
- **SED:** Sistemas de Ecuaciones Diferenciales.
- **T-arenal:** Plataforma de cómputo distribuido.
- **MatLab:** Es un software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio.
- **DCP:** Diseños de casos de prueba.
- **Módulo:** Es un componente bueno autocontrolado de un sistema, el cual posee una interfaz bien definida hacia otros componentes; algo es modular si es construido de manera tal que se facilite su ensamblaje.
- **Bifurcación:** División o separación de una cosa en dos ramales; es un punto o lugar donde se produce una división.

- **Grid:** Es una tecnología innovadora que permite utilizar de forma coordinada todo tipo de recursos (entre ellos cómputo, almacenamiento y aplicaciones específicas) que no están sujetos a un control centralizado.
- **Modelo matemático:** Es uno de los tipos de modelos científicos, que emplea algún tipo de formulismo matemático para expresar relaciones, proposiciones sustantivas de hechos, variables, parámetros, entidades y relaciones entre variables y/o entidades u operaciones, para estudiar comportamientos de sistemas complejos ante situaciones difíciles de observar en la realidad.
- **Weka:** Es un conjunto de librerías JAVA para la extracción de conocimientos desde bases de datos [11].
- **Variable:** Es un símbolo que representa un elemento no especificado de un conjunto dado.
- **LIPS:** Laboratorio Industrial de Pruebas de Software.
- **C#:** Es uno de los lenguajes de programación más populares en informática y comunicaciones.
- **PostgreSQL:** Es un sistema de gestión de base de datos relacional orientada a objetos de software libre.
- **Atributo:** Es la cualidad que se adjudica o predica de un ser con sentido de identidad.
- **Pauta:** Norma o directriz para llevar a cabo una acción.