UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

TESIS

SISTEMA DE GESTIÓN DE REPORTES PARA LOS LABORATORIOS DE IDIOMAS DE LA ENP PLANTEL 9

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

PRESENTA:

VALDIVIA RAMOS JORGE IVÁN

DIRECTORA:

M.C. MARÍA JAQUELINA LÓPEZ BARRIENTOS

CIUDAD UNIVERSITARIA

Octubre, 2013.

Agradezco a Dios por darme el regalo de la vida.

A mi madre Teresa, a mi padre Jorge y a mi hermano Hugo Noel, por todo su amor, apoyo y confianza.

A la Universidad Nacional Autónoma de México por darme la oportunidad de ser uno de sus alumnos y por guiarme y educarme para llegar a ser un profesional responsable y comprometido con el cambio social.

A mis profesores de bachillerato y de la Facultad de Ingeniería que me han enseñado a ser un ingeniero honesto y capaz.

A la M. C. María Jaquelina López Barrientos por asesorarme y guiarme en la elaboración de esta tesis.

A los profesores Juan Carreón, Alejandra Vargas, Cintia Quezada y Alfonso Reyes, por sus observaciones y recomendaciones en la elaboración de este trabajo.

Al plantel 9 de la Escuela Nacional Preparatoria, primero por formarme en la educación media superior, y después por permitirme regresar para llevar a cabo este proyecto de tesis. En especial agradezco a los integrantes del equipo de trabajo de la Mediateca: Maestra Perla, Maestro Jordan; a las profesoras Marianita, Ruth y Diana; a las técnicas Lupita y Anita.

A los alumnos y profesores que participaron en el desarrollo de las pruebas.

A mis hermanos de vida Oscar y Francisco, por su amistad incondicional desde la infancia.

A Karen por apoyarme en esta última etapa.

A Dulce y Ana por sus invaluables enseñanzas y cariño.

A toda la Familia Valdivia, en especial a mis abuelos Lupita y Jorge; a mis tíos Irma, Eduardo y Arnulfo; a mis primos Alejandro y Liz.

A toda la Familia Ramos, en especial a mis abuelos Timo y Amado, a mi tío Gabriel, a mi tía Lupita; a mis primos Magy, Paty, Lalo, Chicho, Yolotzin, Chayo, Benjamín, Nelly, Froy, Lupita, Víctor, Amado, César, Carlos, Ángela y Ana.

A mis hermanos de vida Herme, Richi, Sanz, Vase y Ramms; a todos los pugas y X's.

A toda la comunidad de la Parroquia de Fátima, en especial a los sacerdotes Agustín, Jorge y José; a mis amigos Samuel, Rebeca, Raquel, Juan, Julio, Mónica, Gema, Perla, Betty, Arturo, Gilberto, Fernanda, Pedro, Frutsi, Karla, Michelle, Sr. Agustín, Sra. Anita y Lalito.

A mis amigos Javier, Patyy, Gaby, Mónica, Montse, Naomi, Nahomi, Dulce María, Fátima y Cristian por ayudarme en el desarrollo de las pruebas de Contenido.

A la sociedad mexicana que por medio de su esfuerzo y contribución mantienen a la UNAM como una institución pública.

A los hombres y mujeres libres que buscan un mundo más digno y justo para todos.

“Pronto, los hermanos no se batirán con sus hermanos; los niños ya no serán privados del sol, ni alejados del verdor de los campos; ya no está lejano el día en que ha de haber un pan para cada boca, un lecho para cada cabeza, felicidad para cada corazón.

Y ese será el triunfo de vuestra acción y de la mía, mis compañeros y amigos.”

-Bartolomeo Vanzetti

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Introducción

La Escuela Nacional Preparatoria (ENP) es una institución de carácter público y modelo educativo de la enseñanza media superior, que responde satisfactoriamente a los retos y demandas de la sociedad en su conjunto. Forma parte del sistema educativo mexicano y es uno de los dos sistemas de bachillerato de la UNAM.

La ENP cuenta con la infraestructura necesaria para el desarrollo y atención de la comunidad preparatoriana, donde actualmente asisten a sus nueve planteles cerca de 48,000 alumnos y 2,400 profesores.

El plan de estudios del Bachillerato en la ENP contempla una formación académica de enseñanza de una lengua extranjera, teniendo tres idiomas como opción para la comunidad escolar: inglés, francés e italiano.

Para una formación integral de los estudiantes, se implementaron desde el año 2009 los "Laboratorios de Idiomas" en la ENP Plantel 9 "Pedro de Alba", los cuales tienen como objetivo la practica interactiva de cada lengua estudiada, utilizando equipo informático, como audífonos, micrófonos, teclado, y software especializado para la práctica de alguna de las tres lenguas consideradas en el plan de estudios.

En el plantel 9 de la ENP se cuenta con dos "Laboratorios de Idiomas" que cuentan con las siguientes características:

- Un equipo Servidor
- Una Computadora Personal utilizada por el profesor
- Treinta clientes ligeros utilizados por los alumnos.

La situación actual de los Laboratorios se describe a continuación.

En los laboratorios de Idiomas se da servicio a 39 grupos de alumnos divididos cada uno de ellos en dos secciones, llamadas "Sección A" y "Sección B" con el fin de que en cada sección estén inscritos como máximo 30 alumnos, siendo ésta la capacidad máxima de usuarios que puede atender cada laboratorio.

El horario de servicio en ambos laboratorios inicia a las 7:00 horas, y concluye a las 21:10 horas de lunes a viernes, dividido en dos turnos que son descritos a continuación:

- Turno Matutino, de 7:00 a 14:30 horas. Brinda atención a 78 Secciones.
- Turno Vespertino, de 14:30 a 21:10 horas. Brinda atención a 80 Secciones.

Cada turno cuenta con nueve módulos de clase de 50 minutos en los que los alumnos toman sus clases, y con base en este modelo la Dirección General de la Preparatoria, al inicio de cada ciclo escolar, asigna los horarios correspondientes tanto para los salones como para los laboratorios de idiomas. En la asignación de horarios, se contemplan 5 módulos de clase a la semana, para cada laboratorio, indistintamente de los turnos, los cuales son dedicados exclusivamente al mantenimiento y soporte técnico.

El soporte técnico que se le da a los dos laboratorios es en su mayoría atendiendo las eventualidades detectadas que son reportadas por los estudiantes y profesores en el transcurso de las clases. Es por esto que la gestión de la información de los reportes se vuelve un factor importante para mantener los laboratorios funcionando en una forma adecuada y satisfactoria.

La metodología de la realización de los reportes se realiza de la siguiente manera:

1. El técnico encargado de los laboratorios debe pasar al laboratorio de idiomas en cada inicio de la clase.
2. El técnico debe esperar a que los alumnos enciendan los equipos computacionales y comiencen a utilizarlos para detectar cualquier anormalidad.
3. Los alumnos reportan de manera verbal al técnico los problemas que se presentan.
4. Si el problema se puede resolver de manera directa, el técnico procede a evaluar y resolver el problema de manera inmediata.
5. El profesor reporta al técnico de manera verbal los problemas que haya detectado y formaliza de manera escrita dicho reporte en una bitácora destinada para tal fin.
6. Los problemas que no se puedan resolver de manera directa se resuelven en los tiempos destinados para el mantenimiento.

Los problemas que surgen a partir de la metodología de reportes utilizada son los siguientes:

- No se lleva un registro de ningún tipo de los problemas detectados por los alumnos en los laboratorios.
- El registro escrito que realizan los profesores es ambiguo y carece de una metodología en el manejo y la gestión de reportes de equipos informáticos, lo que dificulta la interpretación del problema por parte del técnico.
- La presencia del técnico en el laboratorio distrae la atención de los alumnos, y se pierde continuidad en la clase.
- Se vuelve complicado el deslindar responsabilidades en el caso de que suceda algún agravio por parte de algún usuario hacia los equipos de cómputo o al inmobiliario, ya que no existe un registro de los reportes y es difícil analizar en qué momento se realizó el acto de agravio y quien lo realizó.
- El análisis de un mantenimiento proactivo para minimizar fallos y prever contingencias es ineficiente, ya que no hay un registro de datos de los tipos de fallos más comunes que se presentan en los Laboratorios de Idiomas.

Entonces, con base en la problemática descrita anteriormente se plantea el objetivo del presente trabajo de tesis el cual es:

Desarrollar e implementar un sistema de administración de reportes, que sean generados por los usuarios de los laboratorios de idiomas de manera ágil, concreta y eficiente, lo que permita gestionar de manera dinámica y efectiva el mantenimiento y soporte técnico necesario para aprovechar al máximo los recursos informáticos disponibles en los Laboratorios de Idiomas, de

manera tal que se permita ejercer una actividad académica plena, y que responda a las necesidades de la comunidad escolar, optimizando tiempos y maximizando el aprovechamiento académico.

Para lo cual se llevaron a cabo las siguientes actividades:

- Recopilar de manera automatizada y controlada la información del funcionamiento de los equipos computacionales que se encuentran en los laboratorios por medio de reportes generados por los usuarios, que permita una atención oportuna por parte de los técnicos para resolver fallas técnicas.
- Desarrollar mecanismos de seguridad efectivos que permitan deslindar responsabilidades en caso que se dé algún agravio por parte de los usuarios de los laboratorios al equipo informático y/o mobiliario.
- Implementar una manera concreta, sencilla y rápida de elaborar reportes por parte de los de los usuarios, lo que permita aprovechar al máximo el tiempo que se dispone del uso de los laboratorios en las actividades académicas correspondientes.
- Desarrollar el sistema en un esquema de código abierto, multiplataforma y de portabilidad, que facilite la escalabilidad, la interoperabilidad, y la implementación, para su uso posterior en algún otro centro de computo de la UNAM donde se requiera.

Así, en el capítulo 1 se plantea el marco teórico, en el que se describen las bases teóricas que sustentan el proyecto desarrollado; en el capítulo 2 se desarrolla el análisis de la problemática, en donde se descubren, analizan, documentan y verifican los requerimientos del sistema de gestión de reportes planteado; en el capítulo 3 se describe el diseño del sistema propuesto, que define la arquitectura de software y las interfaces entre los componentes. También se describen los componentes en un nivel del detalle que permiten su construcción; en el capítulo 4 se aborda el desarrollo del sistema propuesto en el que se documenta la construcción detallada de este, a través de una combinación de codificación, comprobación y depuración; en el capítulo 5 se tratan las pruebas, donde se definen las actividades para probar el sistema propuesto, encontrar errores en el mismo y corregirlos; en el capítulo 6 se presentan los resultados donde se mencionan los objetivos alcanzados por el proyecto. Finalmente, se presentan las conclusiones que se obtuvieron.

Capítulo 1. Marco Teórico

1.1 ANALISIS DE REQUERIMIENTOS

Sommerville (Sommerville, 2005) explica que los requerimientos para un sistema son la descripción de los servicios proporcionados por el mismo y sus restricciones operativas. Estos requerimientos deben reflejar las necesidades que tienen los clientes para apoyarse en un sistema que les ayude a resolver algún problema como el control de un dispositivo, hacer un pedido o encontrar información. El proceso de descubrir, analizar, documentar y verificar estos servicios y restricciones se denomina ingeniería de requerimientos (RE).

El objetivo del análisis de requerimientos es entonces el de determinar con detalle y sin ambigüedad las necesidades de información de una organización para proporcionar una alternativa de solución que las satisfaga de manera conveniente.

Peña (Peña, 2006) menciona las consideraciones necesarias para la definición de la alternativa de solución:

Tipos de alternativas: Definición de la propuesta de solución en función de la naturaleza del requerimiento y los recursos disponibles, ya sea modificando sustancial o parcialmente un sistema actual, el desarrollo de uno nuevo o aprovechando nuevas tecnologías que sustituyan o refuercen mecanismos existentes.

Definición de la alternativa: Elaborar de manera consistente y fundamentada cada propuesta de solución, describiendo las ventajas, desventajas, implicaciones y cambios estructurales de organización que impacten en ámbitos como procedimientos, manejo de información, recursos necesarios y tiempos de implementación que surgirían al implementar cada una.

Establecimiento de criterios de decisión: Determinar los aspectos a satisfacer para la propuesta de solución, como son el plazo de terminación, economía del proyecto, etc. Los criterios se establecen en base a las prioridades y los recursos disponibles.

Comparación de alternativas: Se determina el grado de satisfacción para cada opción, para analizar y distinguir a la que ofrezca mayores ventajas.

Elección: Al evaluar las diferentes alternativas se selecciona aquella que proporciona mayores ventajas y cuya producción sea factible en los términos de tiempos y costos esperados.

1.1.1 RECOPILACIÓN DE INFORMACIÓN

La recopilación de información se refiere al uso de diferentes técnicas y herramientas que son usadas por el analista para desarrollar los sistemas de

información, tales como la entrevista, la encuesta, el cuestionario, la observación, el diagrama de flujo y el diccionario de datos. A continuación se describen las técnicas utilizadas en el desarrollo del proyecto en cuestión.

Una de las técnicas utilizadas es la entrevista, que es el intercambio de información cara a cara entre el analista y el personal de una organización. En consecuencia, tanto el proceso de entrevista como el lugar donde se realiza pueden ser desde muy formal hasta casual. Para llevar a cabo una entrevista, Peña (Peña, 2006) menciona las siguientes recomendaciones:

Para preparar la entrevista:

- Definir el objetivo.
- Establecer los temas.
- Elegir la fuente: área administrativa, persona a entrevistar.
- Seleccionar documentos.
- Planear la entrevista: procedimiento, vigor, tiempo y material.

Para desarrollar la entrevista:

- Explicar los siguientes puntos: identificación personal, propósito de la entrevista, cuál es el proyecto y cuál será la contribución que hará al proyecto el entrevistado.
- Asegurarse de entender correctamente las actividades y responsabilidades del entrevistado.
- Si el entrevistado toma decisiones, es importante hacer un modelo representativo (qué decisiones son hechas y cómo interviene él).
- Hacer preguntas específicas (cuánto) (s).
- Evitar tecnicismos.
- Aprender a escuchar.
- Evitar la divagación y la desviación.
- Buscar opciones, ideas y sugerencias.
- Tomar nota de puntos relevantes.
- Evitar juicios sobre el valor o impresión de los datos recibidos.

Al terminar la entrevista:

- Verificar las notas.
- Revisar si hay algo confuso (escrito y/o mental).
- Repasar el plan.
- Cotejar el objetivo.

- Aclarar aspectos relacionados con la entrevista.

Observaciones:

- Es de suma importancia que las relaciones usuario-analista sean óptimas.
- El analista en la mayoría de los casos, tendrá el número y tiempo restringidos de las entrevistas.

Otra de las técnicas utilizadas es el cuestionario, que es un documento donde se recibe información de interés como por ejemplo: opciones sobre el funcionamiento del sistema, identificación de aspectos que requerirán un mayor estudio, auditoria de resultados, nuevos requerimientos, etc. El analista debe identificar la información que desea conocer para estructurar las preguntas que conduzcan a las respuestas deseadas y adecuar el cuestionario al tipo de individuo que lo llenará. Algunos detalles convenientes de observar, durante el diseño de cuestionarios, son los siguientes:

- Explicar el propósito y uso del cuestionario.
- Proporcionar las instrucciones necesarias para su llenado.
- Indicar el tiempo de llenado y retorno del documento.
- Formular preguntas claras.
- Especificar el tipo de respuesta: abierta o cerrada.
- Las respuestas estarán en el formato adecuado, su tabulación puede ser manual o mecánica.
- Si la respuesta no pudo ser contestada, proporcionar un espacio dedicado a las observaciones.
- Incluir una sección dedicada a verter opciones, críticas y comentarios.

Algunas limitaciones que se presenten en el uso de cuestionarios son las siguientes:

- Es difícil estructurar adecuadamente su diseño, cuando se desconocen las respuestas por obtener (cuestionarios abiertos).
- No es posible hacer aclaraciones al momento de llenar y devolver el cuestionario.
- Generalmente se asigna baja prioridad e importancia a su llenado.

Una técnica más es la observación, que es el acto de presenciar los acontecimientos en el momento y lugar en que suceden, con el propósito de identificar, aclarar y/o confirmar ciertos aspectos de un ambiente determinado. Para

obtener mejores resultados en la observación, existen las siguientes recomendaciones:

- Identificar el ambiente y situación a observar.
- Estimar el tiempo que necesitará.
- Seleccionar el material de apoyo.
- Conducir la observación
- Explique a las partes que serán observadas, qué será hecho y por qué.
- Familiarizarse con el ambiente de observación, así como con sus componentes.
- Tomar notas, checar el tiempo periódicamente.
- Si se interactúa con las personas observadas, preguntar, aclarar aspectos relevantes en el momento y de manera apropiada.

Cuando se termine:

- Organizar el material.
- Revisar los resultados y objetivos propuestos.

Observaciones:

- Las actividades de "Toma de Decisiones" requieren de un estudio más profundo y por medio de otras técnicas.
- Consume mucho tiempo, pues sólo se podrán apreciar una parte del total de actividades que se realizan.
- A la mayoría de personas no les gusta ser observadas.

Otras técnicas adicionales son la recopilación de documentos y el muestreo.

La recopilación de documentos es una fuente de información útil. Son precisamente los documentos tales como: manuales, reportes, hojas de codificación y estadísticas con los que el analista puede obtener una imagen de lo que se hizo, se hace y espera hacer, así como otros aspectos de interés (estructura orgánica, recursos, políticas, etc.).

Si el analista hace uso de la documentación disponible, es muy probable que alcance mayor éxito en sus entrevistas y observaciones posteriores preocupándose por corroborar y aclarar algunos aspectos. Dentro de las limitaciones en el uso de los documentos estarían el grado de actualización y el nivel de contenido y claridad.

Por otra parte, el muestreo es una actividad encargada de colectar y acumular datos relacionados con un problema determinado, que puede ser inmedible o requiere de gran cantidad de esfuerzo para cuantificarlo. Por ejemplo, si se desea conocer el tiempo necesario en atender 10,000 órdenes de clientes, se pudiera tomar el tiempo requerido solamente para 200 y obtener el resultado deseado mediante la proporción correspondiente.

La muestra elegida deberá ser representativa en función a la cantidad y variedad del universo posible del caso de estudio.

1.1.2 REQUERIMIENTOS DE USUARIO Y DE SISTEMA

Los requerimientos del usuario son declaraciones en lenguaje natural y en diagramas de los servicios que se espera que el sistema desarrolle y de las limitaciones bajo las cuales debe funcionar. Describen los requerimientos tanto funcionales como no funcionales de tal manera que puedan ser comprendidos por los usuarios del sistema que no posean un conocimiento técnico detallado. Especifican únicamente el comportamiento externo del sistema, evitando en la medida de lo posible tecnicismos y características complejas del diseño del sistema. Deben redactarse usando el lenguaje natural, empleando representaciones y diagramas intuitivos y sencillos que faciliten su comprensión.

Sin embargo, pueden surgir diversos problemas cuando se redactan en lenguaje natural, como son la falta de claridad, la confusión de requerimientos y la conjunción de requerimientos.

Por su parte, los requerimientos del sistema son descripciones más detalladas de los requerimientos del usuario. Sirven como base para definir el contrato de la especificación del sistema y, por lo tanto, debe ser una especificación completa y consistente del sistema. Son utilizados por los ingenieros de software como el punto de partida para el diseño del sistema.

Los requerimientos del sistema establecen con detalle los servicios y restricciones del sistema. El documento de requerimientos del sistema debe ser preciso. Éste sirve, además, como un contrato entre el comprador del sistema y el desarrollador del software.

La especificación de requerimientos del sistema incluye diferentes modelos del sistema como el de objetos o el de flujo de datos. En principio, los requerimientos del sistema deberán establecer lo que éste hará y no la manera en que se implementará. Sin embargo, en el nivel de detalle requerido para especificar el sistema completamente, es casi imposible excluir toda la información de diseño.

Una especificación del diseño del software es una descripción abstracta del diseño del software, que es una base para un diseño e implementación detallados; agrega detalle a la especificación de requerimientos del sistema.

1.1.3 REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES

Los requerimientos se plantean comúnmente en funcionales y no funcionales ya que permiten dividir el sistema en unidades más pequeñas para que sean manejables.

Los requerimientos funcionales son declaraciones de los servicios que debe proveer el sistema, de manera que reaccione a entradas particulares y el cómo debe de comportarse dada ciertas situaciones. En ocasiones, los requerimientos funcionales también pueden declarar de manera explícita lo que el sistema no debe hacer. La redacción de estos requerimientos depende del tipo de sistema desarrollado, de los usuarios y del enfoque de la organización. Cuando se describen los requerimientos de usuario se redactan de manera abstracta, sin embargo, los requerimientos de sistema se describen con detalle en función de éste, con las respectivas entradas, salidas, excepciones, etcétera. Un requisito funcional define el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica.

Los requerimientos no funcionales son restricciones de los servicios que ofrece el sistema. Incluyen restricciones de tiempo, restricciones sobre el proceso de desarrollo y sobre los estándares. Estas restricciones generalmente se aplican al sistema en su totalidad. Estos requerimientos, como su nombre lo sugiere, no se refieren directamente a las funciones específicas del sistema, sino a las propiedades emergentes como la fiabilidad, tiempo de respuesta y capacidad de almacenamiento. También, definen restricciones como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces. A menudo, estos requerimientos son más críticos que los requerimientos funcionales particulares. Los usuarios del sistema normalmente pueden encontrar formas de trabajar alrededor de una función del sistema que realmente no cumple sus necesidades, sin embargo, el incumplimiento de un requerimiento no funcional puede significar que el sistema entero sea inutilizable.

1.2 DISEÑO DE SOFTWARE

El diseño del software tiene como propósito general el establecer los aspectos lógicos y físicos del sistema, modelar la organización, definir la representación de datos, entradas, funciones y módulos que componen el sistema, considerando los recursos y las

limitaciones determinadas para satisfacer adecuadamente las especificaciones definidas en el análisis de requerimientos.

1.2.1 DISEÑO ARQUITECTÓNICO

Los sistemas de información grandes se descomponen en subsistemas que proveen algún conjunto de servicios relacionados. Al proceso de diseño que identifica estos subsistemas y define un marco de control y comunicación entre ellos se le llama diseño arquitectónico.

El resultado de este proceso de diseño es una descripción de la arquitectura del software.

La arquitectura del sistema incide en el rendimiento, solidez, grado de distribución y mantenibilidad de un sistema. Sommerville (Sommerville, 2005) afirma que el estilo y estructura particulares elegidos para una aplicación puede, por lo tanto, depender de los requerimientos no funcionales del sistema, los que se describen a continuación:

1. Rendimiento. Si el rendimiento es un requerimiento crítico, la arquitectura debería diseñarse para identificar las operaciones críticas en un pequeño número de subsistemas, con tan poca comunicación como sea posible entre estos subsistemas.
2. Seguridad. Si la seguridad es un requerimiento crítico, debería usarse una arquitectura estructurada en capas, con los recursos más críticos protegidos en las capas más internas y aplicando una validación de seguridad de alto nivel en dichas capas.
3. Protección. Si la protección es un requerimiento crítico, la arquitectura debería diseñarse para que las operaciones relacionadas con la seguridad se localizaran en un único subsistema o en un pequeño número de subsistemas.
4. Disponibilidad. Si la disponibilidad es un requerimiento crítico, la arquitectura debería diseñarse para incluir componentes redundantes y para que sea posible reemplazar y actualizar componentes sin detener el sistema.
5. Mantenibilidad. Si la mantenibilidad es un requerimiento crítico, la arquitectura del sistema debería diseñarse usando componentes independientes que puedan modificarse con facilidad. Los generadores de los datos deberían separarse de los consumidores y deberían evitarse las estructuras de datos compartidas.

El diseño arquitectónico es un proceso creativo en el que se intenta establecer una organización del sistema que satisfaga los requerimientos funcionales y no funcionales del propio sistema. Debido a que es un proceso creativo, las actividades dentro del proceso difieren radicalmente dependiendo del tipo de sistema a desarrollar, el conocimiento y la experiencia del arquitecto del sistema, y los requerimientos específicos del mismo. Es, por tanto, más útil pensar en el proceso de diseño arquitectónico desde una perspectiva de decisión en lugar de una perspectiva de actividades. Durante el proceso de diseño arquitectónico los arquitectos del sistema tienen que tomar varias decisiones fundamentales que afectan profundamente al sistema y a su proceso de desarrollo. Basándose en su conocimiento y experiencia, los arquitectos del sistema tienen que responder a las siguientes cuestiones fundamentales:

1. ¿Existe una arquitectura de aplicación genérica que pueda actuar como una plantilla para el sistema que se está diseñando?
2. ¿Qué estilo o estilos arquitectónicos son apropiados para el sistema?
3. ¿Cuál será la aproximación fundamental utilizada para estructurar el sistema?
4. ¿Cómo se descompondrán en módulos las unidades estructurales del sistema?
5. ¿Qué estrategia se usará para controlar el funcionamiento de las unidades del sistema?
6. ¿Cómo se evaluará el diseño arquitectónico?
7. ¿Cómo debería documentarse la arquitectura del sistema?

Las arquitecturas de los sistemas pueden ser bastante genéricas, tales como la arquitectura de los sistemas de gestión de información, o pueden ser mucho más específicas, por ejemplo, las aplicaciones de líneas de productos son aplicaciones construidas sobre una arquitectura base con variantes que satisfacen los requerimientos específicos del cliente.

Cuando se diseña la arquitectura de un sistema, se debe decidir qué tiene en común ese sistema con clases de aplicaciones más amplias, y determinar en qué medida el conocimiento de esas arquitecturas de aplicaciones se puede reutilizar.

1.2.2 EL MODELO DE CAPAS

El modelo de capas de una arquitectura organiza el sistema en capas, cada una de las cuales proporciona un conjunto de servicios.

Cada capa puede pensarse como una máquina abstracta cuyo lenguaje máquina se define por los servicios proporcionados por la capa misma. Este “lenguaje” se usa

para implementar el siguiente nivel de la máquina abstracta. Por ejemplo, una forma usual de implementar un lenguaje es definir un “lenguaje máquina” ideal y compilar el lenguaje para convertirlo en código para esta máquina. A continuación, un paso de traducción adicional convierte este código de máquina abstracta en código de máquina real.

La aproximación por capas soporta el desarrollo incremental de sistemas. A medida que se desarrolla una capa, algunos de los servicios proporcionados por esa capa pueden estar disponibles para los usuarios. Esta arquitectura también soporta bien los cambios y es portable.

En la medida en la que su interfaz permanezca sin cambios, una capa puede reemplazarse por otra capa equivalente. Además, cuando las interfaces de la capa cambian o se añaden nuevas facilidades a una capa, solamente se ve afectada la capa adyacente. Debido a que los sistemas por capas localizan las dependencias de la máquina en las capas más internas, es mucho más fácil proporcionar implementaciones multiplataforma de las aplicaciones de un sistema. Únicamente las capas más internas dependientes de la máquina necesitan ser reimplementadas para tener en cuenta las facilidades de un sistema operativo o base de datos diferente.

1.2.3 DISEÑO ESTRUCTURADO

El diseño estructurado es el proceso de decisión de componentes y la interconexión entre los mismos para la solución de un problema específico. Este enfoque de diseño define las relaciones entre los principales elementos estructurales del sistema. El objetivo principal es desarrollar la estructura de sistema de forma modular y representar las relaciones de control entre los módulos.

El diseño estructurado transforma los elementos estructurales en una descripción procedimental del software. El diseño se realiza después de que se ha definido la estructura del sistema y de los datos, definiendo los algoritmos de procesamiento necesarios. Al concluir el diseño se genera el código fuente, y se realizan las pruebas correspondientes para la integración y la validación del sistema de software.

Los principios utilizados por el diseño estructurado se definen en los siguientes conceptos:

a) Abstracción: Consiste en aislar un elemento de su contexto o del resto de los elementos que lo acompañan, lo que permite disponer de las características de un objeto requerido independientemente de los detalles irrelevantes de bajo nivel. Conforme se desplaza por diferentes niveles de abstracción, se trabaja para crear

abstracciones de datos y de procedimientos. Una abstracción procedimental es una determinada secuencia de instrucciones que tienen una función limitada y específica.

b) Refinamiento sucesivo: Es una primera estrategia de diseño descendente. La arquitectura de un programa se desarrolla en niveles sucesivos de refinamiento de los detalles procedimentales. Se desarrolla una jerarquía descomponiendo una declaración macroscópica de una función de una forma sucesiva, hasta que se llega a las sentencias del lenguaje de programación.

c) Modularidad: Cuando la arquitectura implica modularidad, el software se divide en componentes con nombres y ubicaciones determinados, que se denominan módulos, y que se integran para satisfacer los requerimientos del problema.

d) Arquitectura del software: Se refiere a la estructura jerárquica de los componentes procedimentales (módulos) y a la estructura de datos.

e) Jerarquía de control: También denominada estructura de programa, representa la organización, frecuentemente jerárquica, de los módulos e implica una jerarquía de control. No representa aspectos procedimentales del software, como lo son las secuencias de procesos o la repetición de operaciones.

f) Estructura de datos: Es una representación de la relación lógica existente entre los elementos individuales de datos. Debido a que la estructura de la información afectará invariablemente al diseño procedimental final, la estructura de datos es tan importante como la estructura del programa en la representación de la arquitectura del software.

g) Procedimientos del software: La estructura del programa define la jerarquía de control, independientemente de las decisiones y secuencias de procesamiento. El procedimiento del software se enfoca sobre los detalles de procesamiento de cada módulo individual.

h) Ocultamiento de la información: Principio que sugiere que los módulos se han de caracterizar por decisiones de diseño que los oculten unos a otros. Los módulos se especifican y diseñan de manera que la información (procedimientos y datos) contenida dentro de un módulo sea accesible a otros módulos únicamente a través de las interfaces formales establecidas para cada módulo.

1.3 DESARROLLO DE SOFTWARE

Dentro del desarrollo del software existen varios modelos de procesos de desarrollo de software, también llamados ciclos de vida del desarrollo del software. Existen diversos modelos concretos para el establecimiento de un proceso de desarrollo de software, cada uno de los cuales describe un enfoque especifico para diferentes actividades que se desenvuelven durante el proceso.

A continuación se describen los elementos presentes en el desarrollo de software que se tomarán como base teórica para la elaboración del presente trabajo, siendo estos el proceso de desarrollo incremental, los paradigmas de programación y la reutilización del software.

1.3.1 PROCESO DE DESARROLLO INCREMENTAL

En el desarrollo del presente proyecto se utilizó el modelo de proceso de desarrollo incremental. Tomando como referencia el modelo de desarrollo en cascada, este es el más básico de todos los modelos, sirviendo como base de construcción para los demás modelos de desarrollo, incluido el modelo de desarrollo incremental. La visión del modelo cascada del desarrollo de software es simple; dice que el desarrollo de software puede ser implementado por medio de una secuencia simple de fases. Cada fase tiene un conjunto de metas concretas y definidas, y las actividades dentro de una fase contribuyen a la satisfacción de metas de esa fase o quizás a una subsecuencia de metas de la fase.

En una visión genérica, el proceso de producción de software se divide en cuatro partes: Análisis, Diseño, Codificación y Prueba. Dentro de esta perspectiva de desarrollo se usa el principio de trabajo en cadena, que consiste en mantener al cliente en constante contacto con los resultados obtenidos para cada etapa o incremento. De esta manera es el mismo cliente quien incluye o desecha elementos al final de cada incremento a fin de que el sistema de software se adapte mejor a sus necesidades. El proceso se repite hasta la culminación del producto completo, logrando con esto reducir el tiempo de entrega de manera considerable.

En el modelo incremental el sistema se desarrolla para satisfacer un subconjunto de requerimientos especificados y en las posteriores versiones se incrementa el sistema con nuevas funcionalidades que satisfagan más requerimientos.

Las principales características de este modelo son:

- Combinación de elementos del modelo en cascada con la perspectiva interactiva de construcción de prototipos.
- Cada secuencia lineal produce un producto operacional con cada incremento de la misma forma que progresa el tiempo en el calendario.

- Como un resultado de evaluación y/o utilización se desarrolla un plan para el incremento siguiente, este proceso se repite hasta llegar al producto completo.
- Los primeros incrementos se pueden implementar con menos recursos.

Entre las ventajas de implementación de este modelo se mencionan las siguientes:

- Construir un sistema pequeño es siempre menos riesgoso que construir un sistema grande.
- Al ir desarrollando parte de las funcionalidades, es más fácil determinar si los requerimientos planeados para los niveles subsiguientes son correctos.
- Si un error importante es realizado, sólo la última iteración necesita ser descartada y utilizar el incremento previo.

1.3.2 PARADIGMAS DE PROGRAMACIÓN

Los paradigmas de programación proveen y determinan la visión y métodos de un programador en la construcción de un programa o subprograma. La diversificación de paradigmas da como resultado diferentes estilos de programación y diferentes formas de establecer la solución de problemas.

A continuación se describen cuatro paradigmas que fueron utilizados en la construcción del sistema desarrollado en el presente trabajo, que son: la programación estructurada, la programación modular, la programación por capas y la programación orientada a objetos. Cabe mencionar que existen más paradigmas en la Ingeniería de Software que no son tratados en el presente trabajo.

A) Programación Estructurada

La programación estructurada es un paradigma de programación que consiste en construir programas de fácil comprensión. Es especialmente útil, cuando se necesitan realizar correcciones o modificaciones después de haber concluido un programa o aplicación. Al utilizar la programación estructurada, es mucho más sencillo entender la codificación del programa, que se habrá hecho en diferentes secciones.

El paradigma de programación estructurada se basa en la metodología de desarrollo de programas de refinamiento sucesivo: Se plantea una operación como un todo y se divide en segmentos más sencillos o de menor complejidad, una vez terminado todos los segmentos del programa, se procede a unificar las aplicaciones realizadas por el grupo de programadores. Si se ha utilizado adecuadamente la programación estructurada, esta integración debe ser sencilla y no presentar problemas al integrar la misma, y de presentar algún problema, será rápidamente detectable para su corrección.

La representación gráfica de la programación estructurada se realiza, generalmente, a través de diagramas de flujo, el cual representa el programa con sus entradas, procesos y salidas. La programación estructurada propone segregar los procesos en estructuras lo más simple posibles, las cuales se conocen como secuencia, selección e iteración, que están disponibles en todos los lenguajes modernos de programación imperativa en forma de sentencias, y que, combinando esquemas simples con ellas, se pueden llegar a construir sistemas amplios y complejos pero de fácil entendimiento.

Para la solución de un problema en particular, se inicia considerando las funciones que tiene que cumplir el programa en general y después se van dividiendo estas funciones en subfunciones más pequeñas hasta llegar al caso último o más particular y que ya no se pueda subdividir en casos más pequeños. Una vez que ya se tiene el problema dividido de lo general a lo particular, se empiezan a programar estas pequeñas funciones particulares o módulos.

La modificación de los módulos es más fácil y se pueden referenciar cuantas veces se requiera, con lo que se ahorra tiempo en la programación.

Un programa tiene un diseño estructurado si cumple las dos siguientes condiciones:

- El teorema de Estructura.
- Está debidamente documentado

La programación estructurada hace uso de tres estructuras básicas de control que se definen formalmente como: Estructura Secuencial, Estructura Selectiva y Estructura Repetitiva.

Con la programación estructurada, elaborar programas de computadora sigue siendo una labor que demanda esfuerzo, creatividad, habilidad y cuidado. Sin embargo, con este estilo podemos obtener las siguientes ventajas:

- Los programas son más fáciles de entender. Un programa estructurado puede ser leído en secuencia, de arriba hacia abajo, sin necesidad de estar saltando de un sitio a otro en la lógica, lo cual es típico de otros paradigmas de programación.
- Se logra una reducción del esfuerzo en las pruebas. El seguimiento de las fallas o depuración se facilita debido a la lógica más visible, de tal forma que los errores se pueden detectar y corregir más fácilmente.
- Se crean programas más sencillos y más rápidos.

B) Programación Modular

La programación modular consta de varias secciones dividas de forma que interactúan a través de llamadas a procedimientos, que integran el programa en su totalidad.

En la programación modular, el programa principal coordina las llamadas a los módulos secundarios y pasa los datos necesarios en forma de parámetros.

A su vez, cada módulo puede contener sus propios datos y llamar a otros módulos o funciones. La programación modular consta de varias secciones dividas de forma tal que interactúan a través de llamadas a procedimientos que, a su vez, integran el programa en su totalidad.

En la programación modular, el programa principal coordina las llamadas a los módulos secundarios y pasa los datos necesarios en forma de parámetros.

A su vez, cada módulo puede contener sus propios datos y llamar a otros módulos o funciones.

C) Programación por Capas

La programación por capas es un estilo de programación en el que el objetivo principal es la separación de la lógica de negocios de la lógica de diseño, un ejemplo básico de esto es separar la capa de datos de la capa de presentación al usuario.

La ventaja principal de este paradigma es que el desarrollo se puede llevar a cabo en varios niveles y en caso de que haya algún cambio sólo se revisa al nivel requerido sin tener que revisar entre código mezclado. Además, permite distribuir el trabajo de creación de una aplicación por niveles, de este modo, cada grupo de trabajo está totalmente abstraído del resto de los demás niveles, simplemente es necesario conocer la API que existe entre los niveles.

En el diseño de sistemas informáticos se suele usar las arquitecturas multinivel o programación por capas. En dichas arquitecturas a cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).

El diseño más común actualmente es el diseño en tres capas que consiste en lo siguiente:

1. Capa de presentación: es la que ve el usuario, presenta el sistema al usuario, le comunica la información y captura la información del usuario dando un mínimo de proceso (realiza un filtrado previo para comprobar

que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio.

2. Capa de negocio: es donde residen los programas que se ejecutan, recibiendo las peticiones del usuario y enviando las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) pues es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él. En la Capa de Negocio se encuentra la responsabilidad de traducir los procedimientos deseados en funciones dentro del sistema en desarrollo.

3. Capa de datos: es donde residen los datos. Está formada por uno o más gestor de bases de datos que realiza todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Todas estas capas pueden residir en una única computadora, si bien lo más usual es que haya una multitud de computadoras donde reside la capa de presentación (equivalente a los clientes de la arquitectura cliente/servidor). Las capas de negocio y de datos pueden residir en la misma computadora o bien, si el crecimiento de las necesidades lo requiere, cada capa puede residir en diferentes computadoras. Si por el contrario fuese la complejidad en la capa de negocio lo que obligase a la separación, esta capa de negocio podría residir en una o más computadoras que realizarían solicitudes a una única base de datos. En sistemas muy complejos se llega a tener una serie de computadoras sobre la que reside la capa de datos, y otra serie de computadoras sobre la que reside la base de datos.

En una arquitectura de tres niveles, los términos "capas" y "niveles" no significan lo mismo ni son similares.

El término "capa" hace referencia a la forma como una solución es segmentada desde el punto de vista lógico:

Presentación/ Lógica de Negocio/ Datos.

En cambio, el término "nivel", corresponde a la forma en que las capas lógicas se encuentran distribuidas de forma física. Por ejemplo:

- Una solución de tres capas (presentación, lógica, datos) que residen en una sola computadora (presentación + lógica + datos). Se dice que la arquitectura de la solución es de tres capas y un nivel.

- Una solución de tres capas (presentación, lógica, datos) que residen en dos computadoras distintas (presentación + lógica, lógica + datos). Se dice que la arquitectura de la solución es de tres capas y dos niveles.
- Una solución de tres capas (presentación, lógica, datos) que residen en tres computadoras (presentación, lógica, datos). Se dice que la arquitectura de la solución es de tres capas y tres niveles.

D) Programación Orientada a Objetos

La Programación Orientada a Objetos es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de computadora. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo, y encapsulamiento. Su uso se popularizó a principios de la década de 1990. Actualmente son muchos los lenguajes de programación que soportan la orientación a objetos.

El término de Programación Orientada a Objetos indica más una forma de diseño y una metodología de desarrollo de software que un lenguaje de programación, ya que en realidad se puede aplicar el Diseño Orientado a Objetos a cualquier tipo de lenguaje de programación.

Como su mismo nombre lo indica, la programación orientada a objetos se basa en la idea de un objeto, que es una combinación de variables locales y procedimientos llamados métodos que juntos conforman una entidad de programación.

La programación Orientada a objetos es una forma especial de programar, más cercana a cómo se expresan las cosas en la vida real que otros tipos de programación. Intenta simular el mundo real a través del significado de objetos que contienen características y funciones. Los lenguajes orientados a objetos se clasifican como lenguajes de quinta generación.

1.3.3 REUTILIZACIÓN DEL SOFTWARE

El proceso de diseño en la mayoría de las disciplinas de Ingeniería se basa en la reutilización de sistemas o componentes existentes. Los ingenieros mecánicos o eléctricos no especifican normalmente un diseño en el que cada componente tenga que ser fabricado de una forma especial. Basan su diseño en componentes que han sido utilizados y probados en otros sistemas.

Estos componentes no son solamente pequeños componentes, como tuercas y válvulas sino que incluyen subsistemas mayores, como motores, condensadores o turbinas.

La Reutilización de Software es una estrategia de Ingeniería de Software comparable en la que el proceso de desarrollo es adaptado a la reutilización de software existente. Si bien los beneficios de la reutilización han sido reconocidos durante muchos años sólo en la última década ha existido una transición gradual desde el desarrollo del software original hasta el desarrollo basado en reutilización. La tendencia hacia el desarrollo basado en reutilización viene dada como respuesta a las demandas de una menor producción de software y de menores costos de mantenimiento, de una entrega más rápida de los sistemas y del incremento en la calidad del software. Cada vez más compañías ven su software como un activo valioso y están promocionando la reutilización para incrementar sus beneficios en las inversiones de software.

La Ingeniería del software basada en reutilización es una aproximación del desarrollo que intenta maximizar la reutilización del software existente. Las unidades de software que se reutilizan pueden ser de tamaños totalmente diferentes, por ejemplo:

1. Reutilización de sistemas de aplicaciones. La totalidad de un sistema de aplicaciones puede ser reutilizada incorporándolo sin ningún cambio en otros sistemas, configurando la aplicación para diferentes clientes o desarrollando familias de aplicaciones que tienen una arquitectura común pero que son adaptadas a clientes particulares.
2. Reutilización de componentes. La reutilización de componentes de una aplicación varía en tamaño y va desde subsistemas hasta objetos simples. Por ejemplo, un sistema de emparejamiento de patrones desarrollado como parte de un sistema de procesamiento de textos puede ser reutilizado en un sistema de gestión de base de datos.
3. Reutilización de objetos y funciones. Pueden reutilizarse componentes de software que implementan una única función, como por ejemplo una función matemática o una clase de objetos. Esta forma de reutilización, basada en bibliotecas estándar, ha sido habitual en los cuarenta últimos años. Están disponibles muchas bibliotecas de funciones y clases para diferentes tipos de aplicaciones y plataformas de desarrollo. Éstas pueden utilizarse fácilmente enlazándolas con código de otras aplicaciones. En áreas como algoritmos matemáticos y gráficos, donde se necesita a un experto específico para desarrollar objetos y funciones, ésta es una aproximación particularmente efectiva.

Los sistemas y componentes de software son entidades reutilizables específicas, pero su naturaleza específica significa a veces que el costo de modificarlos para una nueva situación resulta elevado. Una forma complementaria de reutilización es la reutilización de conceptos, en la que, en lugar de reutilizar un componente, la entidad

reutilizada es más abstracta y se diseña para ser configurada y adaptada a una variedad de situaciones. La reutilización de conceptos puede incluirse en aproximaciones tales como patrones de diseño, productos de sistemas configurables y generadores de programas. El proceso de reutilización, cuando se reutilizan los conceptos, incluye una actividad de instanciación en la que los conceptos abstractos se configuran para una situación concreta.

1.4 PRUEBAS DE SOFTWARE

Un sistema de pruebas tiene como objetivo principal determinar situaciones en donde, bajo el entorno del sistema, algo ocurre cuando no tiene que ocurrir y viceversa. La ejecución del sistema de pruebas implica la operación del sistema a analizar bajo un entorno de condiciones controladas y su respuesta a dicho entorno para su consiguiente evaluación. Las condiciones controladas deben incluir tanto situaciones normales como anormales para un análisis amplio y que corresponda con los requerimientos de operación que se plantearán en el Capítulo 2.

Existen dos tipos principales de pruebas, que son definidas como: las pruebas de caja blanca y las pruebas de caja negra.

Las pruebas de caja blanca se centran en los detalles de procedimiento del software, por lo que su diseño está fuertemente ligado al código fuente. En ellas se establecen distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y verificar que se devuelven los valores de salida esperados.

Por su parte, las pruebas de caja negra se centran en lo que se espera de un módulo, es decir, intentan encontrar casos en que el módulo no entregue respuestas correspondientes con su especificación. 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.

Las pruebas de caja negra no consideran la codificación dentro de sus parámetros de evaluación, es decir, que no están basadas en el conocimiento del diseño interno del programa. Estas pruebas se enfocan en los requerimientos establecidos y en la funcionalidad del sistema.

Si consideramos el proceso desde el punto de vista procedimental, la prueba, en el contexto de la ingeniería del software, realmente es una serie de cuatro pasos que se llevan a cabo secuencialmente, que son:

- Prueba de unidad
- Prueba de integración

- Prueba de alto nivel
- Prueba de validación

En las pruebas de unidad, la prueba se centra en cada módulo individualmente, asegurando que funcionan adecuadamente como una unidad. La prueba de unidad hace un uso intensivo de las técnicas de prueba de caja blanca, ejercitando caminos específicos de la estructura de control del módulo para asegurar un alcance completo y una detección máxima de errores. A continuación, se deben ensamblar o integrar los módulos para formar el paquete de software completo.

Por otra parte, la prueba de integración se dirige a todos los aspectos asociados con el doble problema de verificación y de construcción del programa. Durante la integración, las técnicas que más prevalecen son las de diseño de casos de prueba de caja negra, aunque se pueden llevar a cabo algunas pruebas de caja blanca con el fin de asegurar que se cubren los principales caminos de control.

Después de que el software se ha integrado, se dirigen un conjunto de pruebas de alto nivel. Se deben comprobar los criterios de validación (establecidos durante el análisis de requerimientos). Finalmente la prueba de validación proporciona una seguridad final de que el software satisface todos los requerimientos funcionales, los requerimientos de comportamiento y los requerimientos de rendimiento.

Durante la validación se usan exclusivamente técnicas de prueba de caja negra. El software, una vez validado, se debe combinar con otros elementos del sistema como lo es el hardware o las bases de datos. La prueba del sistema verifica que cada elemento encaja de forma adecuada y que se alcanza la funcionalidad y el rendimiento del sistema total.

1.4.1 TIPOS DE PRUEBAS

A) Pruebas de contenido

Las pruebas de contenido verifican que el contenido del sistema sea coherente y consistente, verificando también que las palabras usadas para transmitir una idea al usuario sean las correctas y que la idea trasmitida sea la misma. Para realizar estas pruebas se establecen diferentes casos de uso en donde los usuarios, a través de la interfaz del sistema, logren ciertos objetivos establecidos dentro de un intervalo de tiempo determinado.

B) Pruebas funcionales

Las pruebas de funcionalidad examinan si el sistema cubre los requerimientos de funcionamiento conforme a las especificaciones del diseño. Estas pruebas deben

verificar si el sistema realiza correctamente todas las funciones requeridas, además de verificar la validación de los datos y debe realizar pruebas de comportamiento bajo diferentes escenarios.

C) Pruebas de integración

El proceso de la integración del sistema implica construir éste a partir de sus componentes y probar el sistema resultante para encontrar problemas que pueden surgir debido a la integración de los componentes. Los componentes que se integran pueden ser componentes comerciales, componentes reutilizables que han sido adaptados a un sistema particular, o componentes nuevos desarrollados. Para muchos sistemas grandes, es probable que se usen los tres tipos de componentes. Las pruebas de integración comprueban que estos componentes realmente funcionan juntos, son llamados correctamente y transfieren los datos correctos en el tiempo preciso a través de sus interfaces.

La integración del sistema implica identificar grupos de componentes que proporcionan alguna funcionalidad del sistema e integrar éstos añadiendo código para hacer que funcionen conjuntamente. Algunas veces, primero se desarrolla el esqueleto del sistema en su totalidad, y se le añaden los componentes. Esto se denomina integración descendente. De forma alternativa, pueden integrarse primero los componentes de infraestructura que proporcionan servicios comunes, tales como el acceso a bases de datos y redes, y a continuación pueden añadirse los componentes funcionales. Ésta es la integración ascendente. En la práctica, para muchos sistemas, la estrategia de integración es una mezcla de ambas, añadiendo en incrementos componentes de infraestructura y componentes funcionales. En ambas aproximaciones de integración, normalmente tiene que desarrollarse código adicional para simular otros componentes y permitir que el sistema se ejecute.

Las técnicas utilizadas para las pruebas de integración son las que se describen a continuación:

- Técnica top-down: Se empieza con los módulos de nivel superior, y se verifica que estos módulos llaman a los de nivel inferior de manera correcta, con los parámetros correctos.
- Técnica bottom-up: Se empieza con los módulos de nivel inferior, y se verifica que estos llaman a los de nivel superior de manera correcta, con los parámetros correctos.

La principal dificultad que surge durante las pruebas de integración es la localización de los errores. Existen interacciones complejas entre los componentes del

sistema, y cuando se descubre una salida anómala, puede resultar difícil identificar dónde ha ocurrido el error. Para hacer más fácil la localización de errores, siempre debería utilizarse una aproximación incremental para la integración y pruebas del sistema. Inicialmente, debería integrarse una configuración del sistema mínima y probar este sistema. A continuación, deberían añadirse componentes a esta configuración mínima y probar después de añadir cada incremento.

D) Pruebas de rendimiento

Para sistemas de tiempo real, es inaceptable el software que proporciona las funciones requeridas pero no se ajusta a los requerimientos de rendimiento. La prueba de rendimiento está diseñada para probar el rendimiento del software en tiempo de ejecución dentro del contexto de un sistema integrado.

La prueba de rendimiento se da durante todos los pasos del proceso de la prueba. Incluso al nivel de unidad, se debe asegurar el rendimiento de los módulos individuales a medida que se llevan a cabo las pruebas de caja blanca. Sin embargo, hasta que no están completamente integrados todos los elementos del sistema no se puede asegurar realmente el rendimiento del sistema.

Las pruebas de rendimiento, frecuentemente, requieren instrumentación tanto de software como de hardware. Es decir, muchas veces es necesario medir la utilización de recursos (por ejemplo, ciclos de procesador), de un modo exacto. La instrumentación externa puede monitorizar los intervalos de ejecución, los sucesos ocurridos (por ejemplo, interrupciones) y muestras de los estados de la máquina en un funcionamiento normal. Instrumentando un sistema, el encargado de la prueba puede descubrir situaciones que lleven a degradaciones y posibles fallos del sistema.

E) Pruebas de seguridad

Cualquier sistema basado en computadora que maneje información sensible o lleve a cabo acciones que puedan perjudicar (o beneficiar) impropiamente a las personas es un posible objetivo para entradas impropias o ilegales al sistema.

La prueba de seguridad intenta verificar que los mecanismos de protección incorporados en el sistema lo protegerán de accesos impropios.

Durante la prueba de seguridad, el responsable de la prueba desempeña el papel de un individuo que desea entrar en el sistema. Debe intentar conseguir las claves de acceso por cualquier medio, puede atacar al sistema con software a la medida, diseñado para romper cualquier defensa que se haya construido, debe bloquear el sistema, negando así el servicio a otras personas, debe producir a propósito errores

del sistema, intentando acceder durante la recuperación o debe curiosear en los datos sin protección, intentando encontrar la clave de acceso al sistema, etc.

Con tiempo y recursos suficientes, una buena prueba de seguridad terminará por acceder al sistema. El papel del diseñador del sistema es hacer que el costo de la entrada ilegal sea mayor que el valor de la información obtenida.

Capítulo 2. Análisis de la Problemática

El Sistema de Gestión de Reportes para los Laboratorios de Idiomas (SIRLI) contribuye a simplificar y mejorar la calidad de los servicios prestados en los laboratorios de idiomas. El sistema ofrece gestión y consulta de información proporcionada por los mismos usuarios de los laboratorios para poder responder de manera oportuna y organizada a las necesidades de soporte técnico, mantenimiento y planeación de actividades por parte de la Coordinación de la Mediateca y de los técnicos responsables de estas aulas habilitadas para el uso de material multimedia. En la actualidad en los laboratorios de idiomas de otros planteles de bachillerato de la UNAM se sigue utilizando el sistema de reportes de manera manual, esto es, de manera oral y presencial por parte de los usuarios hacia los técnicos, lo cual no se toma como un sistema perjudicial, sino como punto inicial de referencia para saber cuáles son las funciones que se deben implementar en el sistema.

En el análisis de requerimientos se planteó estudiar la problemática actual que tienen los usuarios con respecto a la calidad de los servicios prestados en los laboratorios, con el fin de satisfacer sus necesidades, en función de requerimientos medibles, comprobables, sin ambigüedades o contradicciones.

2.1 RECOPILACIÓN DE INFORMACIÓN DE USUARIOS FINALES

Tomando en cuenta el propósito del SIRLI, se definieron cuatro tipos de usuarios finales, los cuales tienen interacción con el sistema de manera directa:

- Alumnos
- Profesores
- Técnicos
- Coordinadores

Para definir los requerimientos de usuario y el tipo de arquitectura e interfaz que debe implementar el SIRLI se procedió a un levantamiento de información de los tipos de usuarios definidos, utilizando diferentes métodos de recopilación de información:

- Encuesta para los alumnos
- Encuesta para los profesores
- Entrevista para los técnicos y el Coordinador de Laboratorios Multimedia

Para los alumnos se diseñó un formato de encuesta en el que se les cuestionó acerca de la calidad de los equipos de cómputo disponibles en los laboratorios, el estado en el que se encuentran regularmente los laboratorios al iniciar su clase, el momento durante la clase que destinan a revisar el estado de los equipos que utilizan, los incidentes y fallas que suceden con mayor frecuencia en los laboratorios, y la cantidad de incidentes que no han podido reportar a lo largo del ciclo escolar a los técnicos. La encuesta se realizó de manera anónima con el fin de motivar al alumno a llenarla de manera sincera y sin presiones por parte de los profesores, técnicos y coordinador de los laboratorios. Se realizaron doscientas encuestas para alumnos, de las cuales se descartaron dos de ellas

porque no contestaron ningún reactivo. El formato de la encuesta para alumnos se muestra en el Anexo 1.

El cuestionario para los profesores se diseñó con el objetivo de conocer, de manera explícita, las necesidades actuales de los laboratorios con respecto al servicio y mantenimiento de los laboratorios por parte de los técnicos y de la coordinación, además de tomar en cuenta su experiencia en el laboratorio y sus propuestas para implementarlas en el SIRLI. Al profesor se le cuestionó acerca del tiempo de respuesta por parte de los técnicos para resolver fallas, el tiempo que considera necesario que se tome para que el alumno verifique el estado de su equipo y los problemas más recurrentes que suceden en los laboratorios. Para la realización de la encuesta se implementó una aplicación sencilla desarrollada en el lenguaje de programación "Visual Basic" que consiste en un formulario y se instaló en los equipos que los profesores utilizan en su clase en ambos laboratorios. El formato de la encuesta para profesores se muestra en el Anexo 2. La respuesta de los profesores se almacena en un archivo secuencial, que se muestra en el Anexo 3.

El formato de entrevista para los técnicos y el Coordinador de los Laboratorios está diseñado con el fin de conocer de manera directa los requerimientos de implementación y las funciones que el SIRLI debe desempeñar, para lo que se organizó una junta en la que participaron los dos técnicos responsables, un profesor representante y el coordinador de los laboratorios. Los formatos de las entrevistas tanto para los técnicos y para el coordinador se muestran en el Anexo 4.

2.2 ANÁLISIS DE REQUERIMIENTOS DE LOS USUARIOS FINALES

Los resultados se analizaron de acuerdo a las respuestas que dio cada grupo de estudio:

2.2.1 ALUMNOS

Pregunta 1. ¿Cómo calificarías la calidad de los equipos de cómputo de los laboratorios de idiomas?

Como lo muestra la **figura 2.1**, los alumnos tienen en general una buena imagen de la calidad de los equipos del laboratorio, indicando que los problemas que se llegan a presentar se pueden resolver de manera directa y rápida por parte de los técnicos. El SIRLI debe incidir directamente en mejorar la percepción de los alumnos con respecto a la calidad de los equipos, por lo que debe mostrar estadísticas de los reportes que los técnicos y la coordinación puedan utilizar para implementar una adecuada planeación de los mantenimientos y el soporte técnico que se da en los laboratorios.

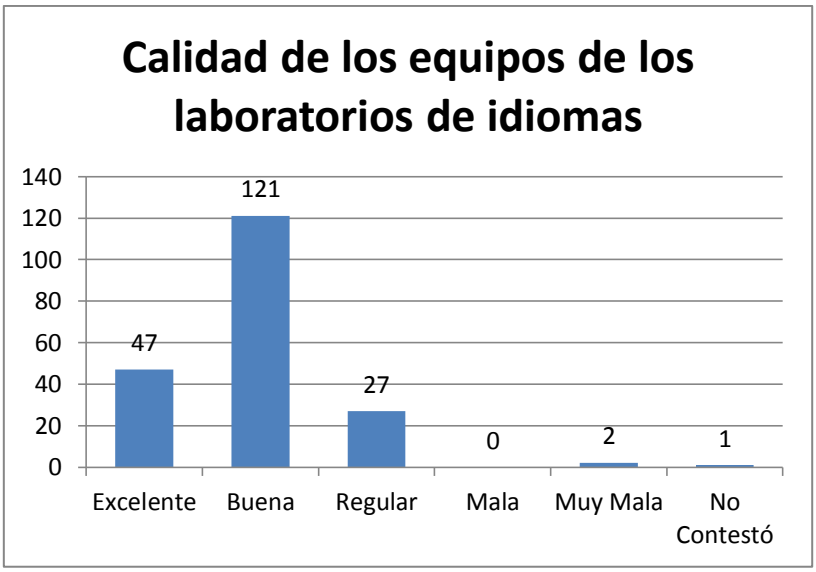


Figura 2.1. Resultados de las respuestas de los alumnos con respecto a su opinión en cuanto a la calidad de los equipos con los que cuentan los laboratorios.

Pregunta 2. ¿Generalmente cuál es la condición en la que encuentras el equipo de cómputo al momento de ingresar al laboratorio (computadora, audífonos, mesa, monitor, etc.)?

Como lo indica la **figura 2.2**, el estado en el que encuentran los alumnos los equipos al iniciar es generalmente aceptable, sin embargo, dada la importancia de estos en el aprendizaje de idiomas, se debe tener un adecuado plan de mantenimiento, tanto proactivo como reactivo, para asegurar que todos los equipos estén en condiciones de ser utilizados por la comunidad escolar. El SIRLI debe implementar estadísticas que contribuyan a este objetivo.

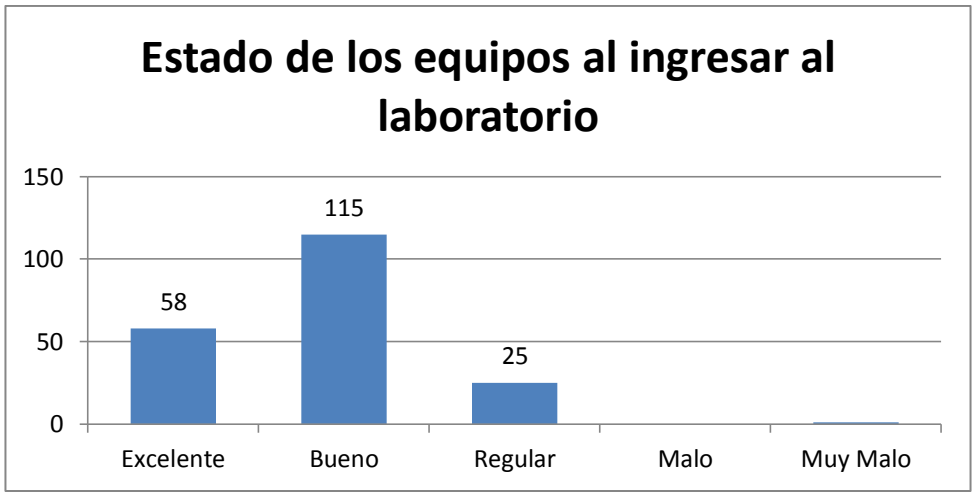


Figura 2.2. Resultados de las respuestas de los alumnos con respecto a su opinión en cuanto al estado de los equipos cuando ingresan a los laboratorios.

Pregunta 3. ¿En qué momento revisas el estado del equipo que utilizas?

De acuerdo con la **figura 2.3**, el resultado indica que los alumnos generalmente se preocupan por revisar minuciosamente el estado de sus equipos sólo al iniciar la clase, lo que deja abierta la posibilidad de que en el transcurso de la clase, si llega a haber algún problema o falla, no se crea necesario revisar o reportar el incidente, implicando que el alumno que utilice dicho equipo en la clase siguiente pierda tiempo, propiciando que no se aproveche adecuadamente el equipo del laboratorio. Por tanto, el sistema debe contar con la opción de generar el reporte de manera directa y cuando ocurra el fallo, lo que permita la atención oportuna de los técnicos, aun cuando el alumno ya haya realizado un reporte durante esa misma hora de clase.

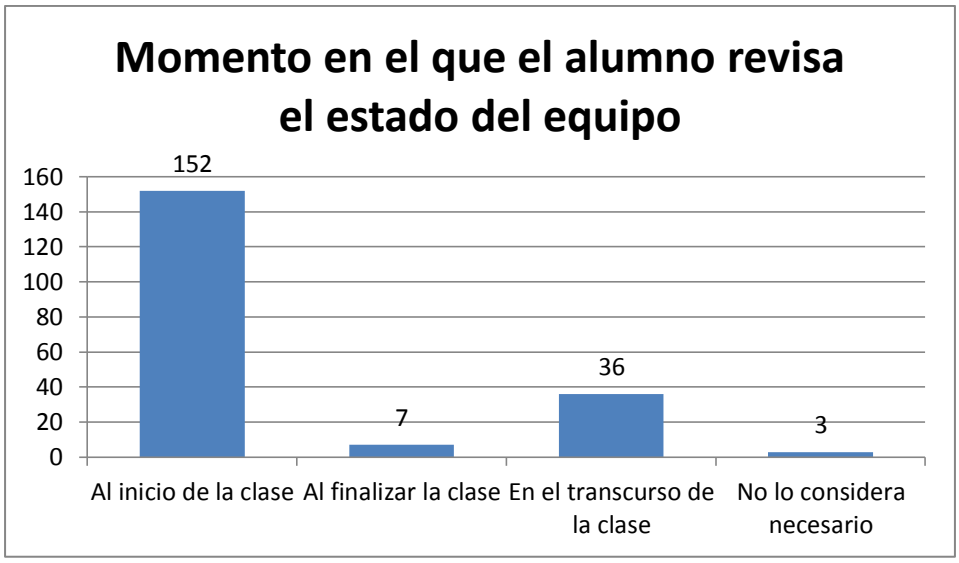
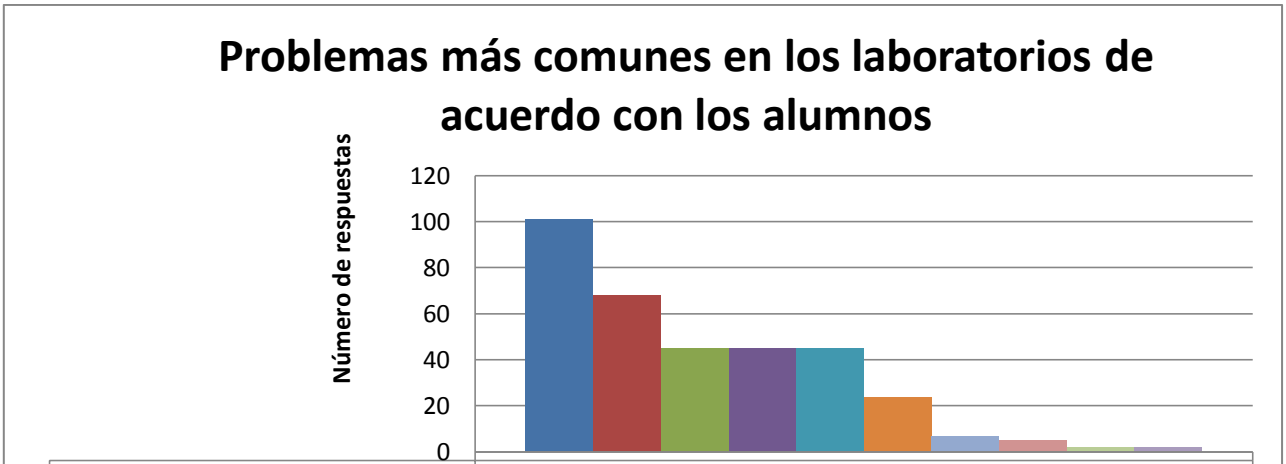


Figura 2.3. Resultados de las respuestas de los alumnos con respecto a su opinión en cuanto al estado de los equipos cuando ingresan a los laboratorios.

Pregunta 4. Indica los tres problemas que más te han ocurrido en el laboratorio.

Como lo indica la **figura 2.4**, las respuestas de los alumnos nos indican tipos de errores más comunes que se presentan en los laboratorios. Este dato permitirá definir los valores iniciales en el sistema. Debido a la diversidad de problemáticas que se pueden presentar, se diseñará un módulo especial en que los técnicos podrán personalizar el tipo de errores que podrán reportar los alumnos.



No se oye un lado de los audífonos	101
Esta rayada alguna parte del equipo	68
No enciende el monitor	45
No funcionan los audífonos	45
El mouse no reacciona	45
Problemas con algún programa	24
No funciona el teclado	7
No funciona el micrófono	5
No enciende el CPU	2
Daños al mobiliario	2

Figura 2.4. Resultados de la respuestas de los alumnos con respecto a la cantidad de problemas más frecuentes que ocurren en los laboratorios.

Pregunta 5. ¿Cuántas veces te ha ocurrido que el equipo falle repentinamente y ya no pudiste reportarlo al Técnico del Laboratorio?

Los datos de la **figura 2.5** nos indican que, aunque la mayoría de las veces se reportan las incidencias, existe un porcentaje considerable de alumnos (27.7% de los encuestados) que por diversas razones, al menos una vez durante el ciclo escolar, no pudieron reportar alguna incidencia o fallo, lo que provoca que el mantenimiento no sea efectivo, y en consecuencia el nivel de incidencias aumenta. Por tanto el esquema de llenado de reportes que se implemente del sistema debe propiciar que el alumno se familiarice con su uso, además debe ser sencillo, concreto y rápido en su llenado.

Figura 2.5. Resultados de la respuestas de los alumnos con respecto a la cantidad de fallas que no han podido reportar a los técnicos durante su clase en los laboratorios en el transcurso del ciclo escolar 2011-2012.

2.2.2 Profesores

Pregunta 1. ¿Generalmente cuántas clases se tardan los técnicos en resolver los problemas reportados en los laboratorios?

Como lo muestra la **figura 2.6**, los profesores comentan que los técnicos resuelven de forma rápida los incidentes cuando son reportados. El problema radica cuando el alumno reporta alguna incidencia, y aun cuando sea mínimo el tiempo empleado por el técnico para resolverla, es tiempo que pierde el alumno en clase, atrasándolo con respecto a los demás. Por medio del SIRLI se debe poder planificar el tiempo empleado para atender los reportes de una forma más eficiente, de manera que afecte lo menos posible el desarrollo de la clase.

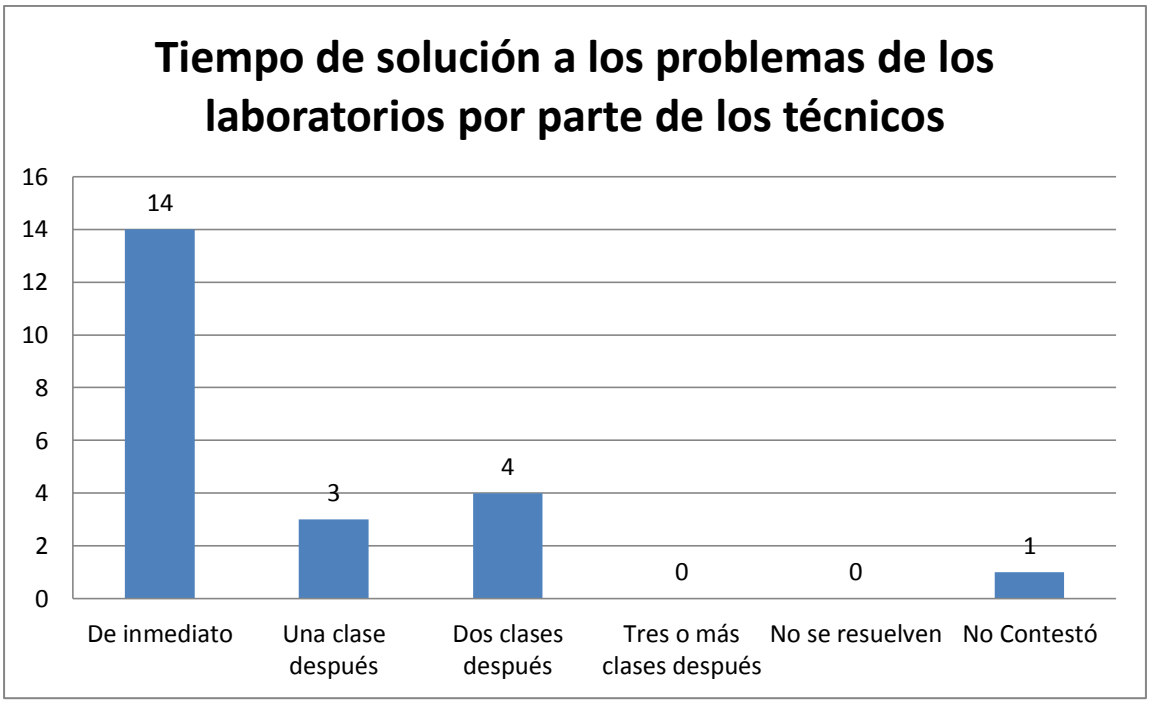


Figura 2.6. Resultados de la respuestas de los profesores con respecto a su opinión en cuanto al tiempo de respuesta de los técnicos para atender problemas en los laboratorios.

Pregunta 2. ¿Cuánto tiempo dedican sus alumnos a verificar el estado de sus equipos al iniciar la clase?

Según muestran los datos de la **figura 2.7**, para los profesores es muy importante que los alumnos dediquen unos minutos para la revisión de sus equipos con los que van a trabajar en la clase, siendo importante que utilicen el menor tiempo posible para aprovechar adecuadamente la sesión de clase. Uno de los objetivos del SIRLI es tener la característica de la elaboración de los reportes en aproximadamente un minuto, permitiendo al profesor elaborar su clase sin interrupciones y aprovechando al máximo el tiempo disponible de su clase.

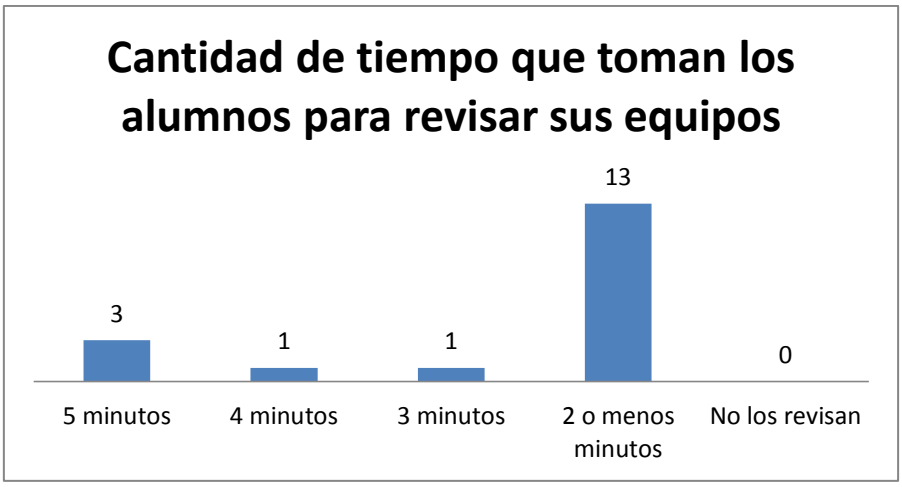


Figura 2.7. Resultados, en porcentajes, de las respuestas de los profesores con respecto a su opinión en cuanto al tiempo que deben tomar los alumnos cuando ingresan a los laboratorios para revisar sus equipos.

Pregunta 3. ¿Le gustaría poder llevar un registro electrónico (para alumnos y profesores) de las incidencias de clase en el que se puedan generar reportes en cualquier momento de la clase acerca del estado y la condición de los equipos del laboratorio?

Según la **figura 2.8**, la mayor parte de los profesores concuerdan en que si es necesario implementar un sistema electrónico para la generación de reportes. Es significativo también tomar en cuenta a los profesores que no están de acuerdo en implementar dicho sistema, pudiendo deberse a factores como la poca experiencia en el uso de software o la desconfianza que tienen de poder manejar el sistema correctamente. En este sentido el SIRLI debe contar con una interfaz sencilla y amigable, que motive su uso y permita desarrollar al docente la confianza en el uso de software dentro de sus hábitos académicos.

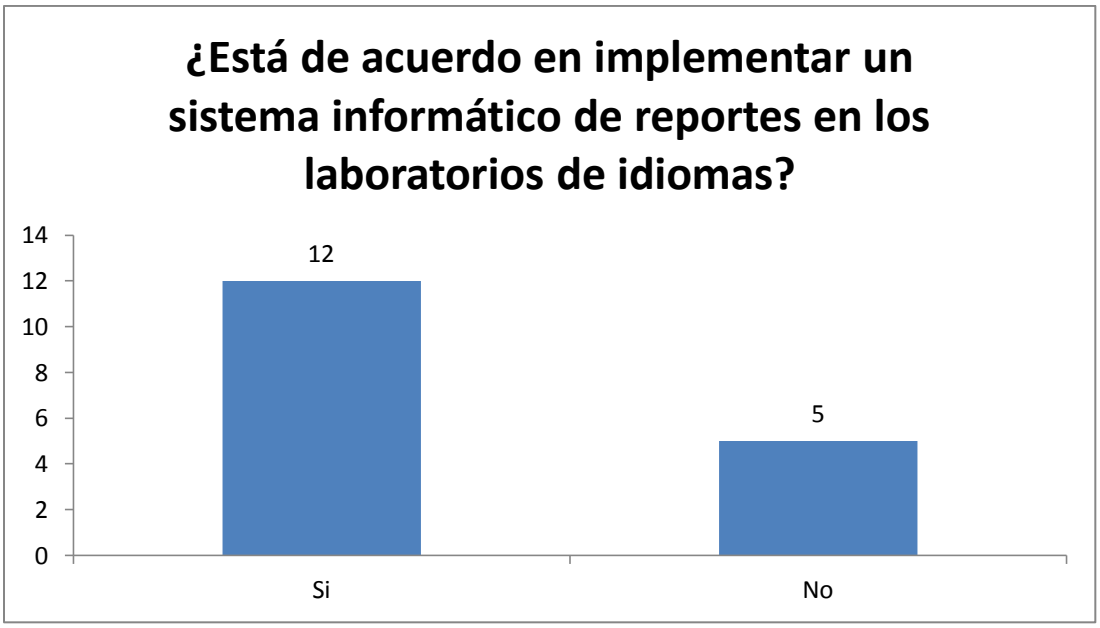


Figura 2.8. Resultados, en porcentajes, de las respuestas de los profesores con respecto a su opinión en cuanto al tiempo que deben tomar los alumnos cuando ingresan a los laboratorios para revisar sus equipos.

Pregunta 4. Enliste los problemas que más ocurren en el laboratorio.

Se enlistan a continuación los problemas que los profesores identifican en los laboratorios:

- Fallas en la red
- Laboratorios desaseados
- Fallas en los audífonos
- Micrófono no registra apropiadamente la voz
- Equipos que se pasman o dejan de funcionar
- Problemas relacionados al software “Tell Me More”
- Monitores desactivados
- Diademas rotas
- Fallas en proyección de video

Los problemas que enlistan los profesores son semejantes a los que reportan los alumnos. En este punto se reafirma la necesidad de implementar en el SIRLI la característica de personalizar la interfaz de usuario en cuanto a las opciones de reportes, de manera que sea adaptable a las necesidades propias de académicos, alumnos y técnicos.

2.2.3 TÉCNICOS Y COORDINADOR DE LOS LABORATORIOS.

Para el análisis de requerimientos con base en las necesidades de la Coordinación de Laboratorios de Idiomas y de los técnicos correspondientes, se procedió a la realización de una entrevista con la presencia de los dos técnicos responsables de los laboratorios y el coordinador de la Mediateca del plantel, en la que se especifican de manera concreta las características que se desean que contemple el SIRLI, con base en su experiencia en el trato de reportes y con la interacción con los usuarios de los laboratorios. Se enlistan a continuación las solicitudes para implementación.

El SIRLI debe:

- Ser portable
- Desarrollarse en un enfoque de Software Libre
- Tener módulos de usuario claramente definidos, de acuerdo a su jerarquía
- Implementar una interfaz sencilla, amigable, interactiva e intuitiva.
- Consumir bajos recursos en su implementación, con base en las características propias del servidor y de los clientes ligeros con los que cuenta el laboratorio.
- Ser sencillo de instalar e implementar.
- Contar con la documentación correspondiente.
- Mostrar datos estadísticos de los reportes (índice de reportes, alumnos que reportan por clase, etcétera).

Es necesario mencionar, que la Coordinación y los técnicos solicitan de manera explícita, y como un requisito del sistema, que la información de inicio de sesión de los alumnos en el sistema sea en referencia directa al número de cuenta UNAM de cada alumno, pues argumentan que los alumnos no tienen el habito de resguardar correctamente sus contraseñas, y el hecho de atender a cada alumno que pierda sus datos de inicio de sesión en el sistema haría más compleja y problemática la administración del mismo.

2.3 ESPECIFICACIÓN DE LOS PERFILES DE USUARIO

El SIRLI debe contar con una jerarquización de usuarios clara y concreta, que le otorgue la capacidad de implementar los servicios de seguridad necesarios para asegurar la información que maneja. Para este fin, contando con el apoyo de la Coordinación de Mediateca, se definen los siguientes perfiles de usuario:

Alumno: Usuario capaz de generar reportes. Sólo podrá ingresar al sistema en el módulo de generación de reportes de alumno. Podrá visualizar sus propios reportes, pero no los de los demás usuarios. Debe contar con un nombre de usuario y contraseña propios, otorgados directamente por la Coordinación de Mediateca.

Profesor: Usuario capaz de generar reportes. Sólo podrá ingresar al sistema en el módulo de generación de reportes de profesor. Puede visualizar sus propios reportes y los de sus grupos únicamente. Debe contar con nombre de usuario y contraseña propios, otorgados directamente por la Coordinación de Mediateca.

Técnico: Usuario capaz de hacer consultas de los reportes a la Base de Datos, agregar y eliminar reportes, agregar y eliminar usuarios ya sean alumno o profesor. Sólo podrán ingresar al sistema en el módulo de Técnico. Debe contar con nombre de usuario y contraseña propios, otorgado directamente por la Coordinación de Mediateca.

Coordinador: Usuario capaz de realizar las mismas funciones que el usuario Técnico, además de agregar y eliminar técnicos. Son los responsables de otorgar nombres de usuario y contraseñas a cada técnico. Sólo podrán ingresar al sistema en el módulo de Coordinador.

2.4 DEFINICIÓN DE LOS REQUERIMIENTOS DEL SISTEMA

2.4.1 REQUERIMIENTOS FUNCIONALES

En cuanto a requerimientos funcionales el SIRLI debe administrar un volumen considerable de reportes, tomando en consideración elementos clave como el grupo, la fecha, la hora, el profesor y el contenido del reporte, por lo que debe contar con una interfaz de consulta muy potente y eficiente sobre todo para los módulos de los técnicos y de coordinador.

Para cumplir con el objetivo de generación de reportes por parte del alumno en un tiempo mínimo, establecido en la introducción de este trabajo, el módulo del usuario Alumno debe contar con una interfaz intuitiva, sencilla, amigable, y con un formulario simple que mediante el uso de menús simplificados pueda ser llenado por el usuario de una manera rápida y concreta.

Otro requerimiento funcional es la ubicación de los servicios que provean al SIRLI de las características necesarias para su funcionamiento, como lo son el sistema gestor de la base de datos y el servidor Web con soporte del lenguaje PHP. El sistema de base de datos del SIRLI reside en el servidor local de cada laboratorio, que cuenta con los servicios PHP y MySQL, permitiendo a la interfaz del SIRLI desplegarse en el navegador web del cliente. Esto le da al SIRLI la oportunidad de no tener limitantes de plataforma. Estos servicios son utilizados ya que los servidores de los laboratorios cuentan con estas herramientas instaladas y en funcionamiento, además de que permiten el desarrollo e implementación del SIRLI.

El sistema debe ser capaz de identificar al usuario por medio de un nombre de usuario y una contraseña, permitiendo también determinar los datos de usuario, estadísticas de reporte individual y por grupo. Los datos de los alumnos, así como su nombre de usuario y contraseña deben ser importados al sistema mediante una base de datos que el plantel proporcione a la Coordinación de Mediateca.

2.4.2 REQUERIMIENTOS NO FUNCIONALES

Los requerimientos no funcionales, que se enfocan en el diseño o la implementación del SIRLI son:

A) Requerimientos de entorno.

El SIRLI requiere estar alojado en un servidor que tenga correctamente instalado un servidor web, en el que se implemente la base de datos del mismo sistema, y que permita la utilización de scripts php.

B) Requerimientos del servidor web.

Es requisito indispensable tener instalado un servidor web que cuente con los servicios de PHP y MySQL, por lo que se toma como punto de inicialización la instalación de la herramienta **AppServe**, que al ser de código abierto, cumple con el enfoque de desarrollo del proyecto y facilita la instalación de **Apache**, **MySQL** y **PHP** configurando estas aplicaciones en forma automática. Además, como agregado, incorpora **phpMyAdmin** para el manejo de MySQL, lo que facilita la creación y la administración de la base de datos mencionada.

Para el desarrollo del SIRLI se toma como referencia AppServ en su versión 2.5.10, instalando y configurando las siguientes herramientas con sus respectivas versiones:

- Apache 2.2.8
- PHP 5.2.6
- MySQL 5.0.51b
- phpMyAdmin 2.10.3

C) Requerimientos de la interfaz de usuario.

El sistema SIRLI se desarrolla en un ambiente modular, definido por la jerarquía de usuario y las políticas de privilegios que cada tipo de usuario posee. Cada usuario posee su propio módulo, que manejará de manera interna los privilegios y las restricciones que éste tiene con respecto al sistema y a la base de datos. Toda la interacción con la base de datos dispone de mensajes que el sistema despliega dependiendo del tipo de acción por parte del usuario y las restricciones que éste tiene con respecto al sistema.

Capítulo 3. Diseño del SIRLI

Con base en el análisis de requerimientos dado en el capítulo anterior, se modelaron las estructuras lógicas del sistema a desarrollar y se establecieron las características de funcionamiento en las que se fundamenta el sistema, de manera que cumpliera con los objetivos definidos en la introducción del presente trabajo.

3.1 ESTRUCTURA GENERAL DEL SIRLI

3.1.1 ARQUITECTURA FÍSICA

La descripción de la arquitectura física, sus interconexiones, el software y hardware que se encuentra en cada equipo de los laboratorios estudiados se representa en el **diagrama** de distribución modelado en la **figura 3.1.**

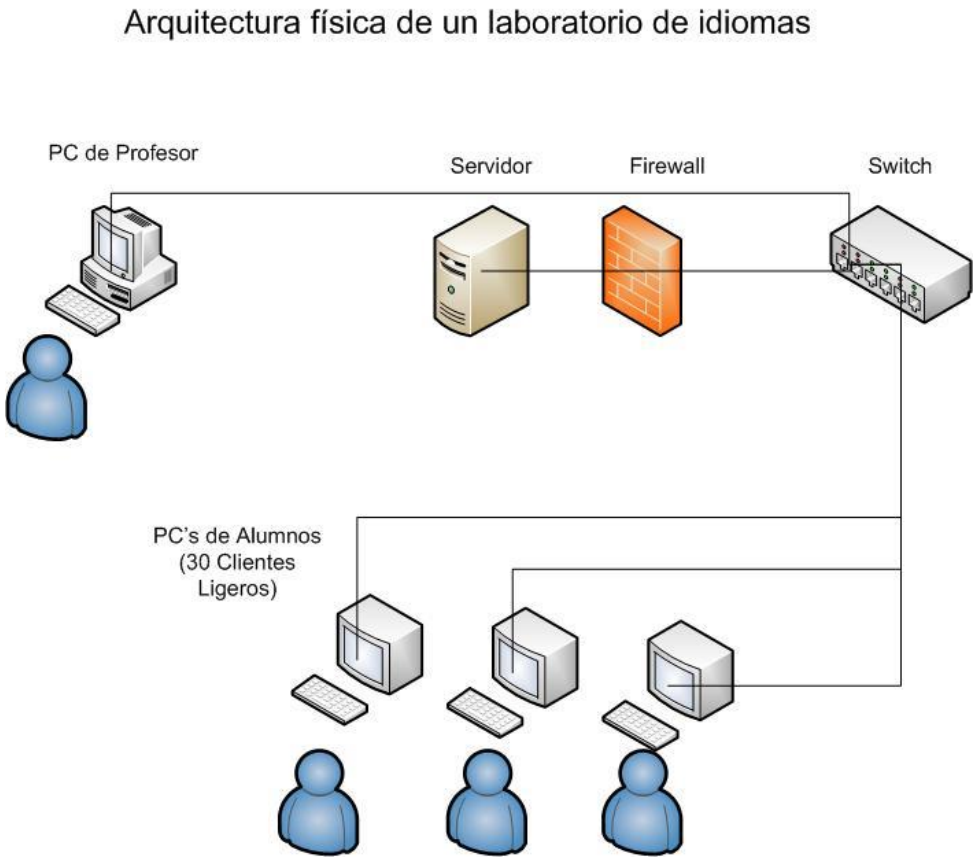


Figura 3.1. Diagrama de distribución de la arquitectura de red de los laboratorios de idiomas.

Cabe mencionar que en el diagrama de distribución se omiten ciertos elementos físicos que están presentes en los laboratorios pero no son elementos que influyan en la comunicación en red, como lo son el proyector VGA o el reproductor DVR utilizados para reproducir contenido multimedia.

En el diseño del sistema se tomó en cuenta el tipo de lógica de red en la cual se desempeña, siendo determinante en el tipo de solución a desarrollar.

Dado el modelo de red que se tiene en los laboratorios se determinó que los módulos de operación del SIRLI se desempeñen en un modelo Cliente/Servidor, estableciendo los scripts de operación en el servidor, y colocando solamente accesos directos a estos scripts en función del perfil de usuario en el que se desenvuelve éste dentro del laboratorio.

3.1.2 PERFILES DE USUARIOS

Tomando como base de jerarquización de usuarios la planteada en el análisis de requerimientos, se definieron cuatro perfiles de usuario, cada uno con funciones particulares de cada perfil. Las funciones principales de cada usuario se muestran de la **tabla 2.1** a la **tabla 2.4**, en orden descendente de acuerdo a su nivel de jerarquía.

Tabla 3.1. Descripción de funciones del nivel de usuario "Coordinador".

Funciones de Coordinador	Descripción
Edición de datos de usuario propios	Permite al coordinador cambiar el nombre de usuario y contraseña. Es recomendable cambiar este último valor en el primer uso del sistema.
Inicialización del sistema	Elimina la información de la base de datos, restableciendo las variables del sistema. Se debe de usar al término de cada ciclo escolar posteriormente a la generación de un respaldo de la información.
Configuración del sistema	Establece los parámetros del sistema, como los horarios de los grupos, la asociación de grupo-profesor y los valores que pueden tener los reportes.
Administración de Usuarios "Técnico"	Permite modificar los datos individuales de los usuarios definidos con el nivel de usuario "Técnico".
Administración de Usuarios "Profesor"	Permite modificar los datos individuales de los usuarios definidos con el nivel de usuario "Profesor".
Administración de Usuarios "Alumno"	Permite modificar los datos individuales de los usuarios con el nivel de usuario "Alumno".
Consultas a la Base de Datos	Genera consultas directas a la base de datos, con un nivel de privilegios de jerarquía máxima. La consulta realizada genera también un reporte visual en función de parámetros definidos en la misma consulta.

Tabla 3.1. (Continuación) Descripción de funciones del nivel de usuario "Coordinador".

Agregar Avisos al Sistema	Genera un mensaje que se puede visualizar en el sistema por todos los usuarios.
Generar respaldo de la información	Genera un respaldo total de la información contenida en la base datos. Se debe de ejecutar por lo menos al finalizar el ciclo escolar en curso.
Crear reporte	Permite la creación de un reporte con valores definidos por el mismo usuario.

Tabla 3.2. Descripción de funciones del nivel de usuario "Técnico".

Funciones de Técnico	Descripción
Edición de datos de usuario propios	Permite modificar los datos personales de usuario, como lo son el nombre de usuario y su contraseña.
Inicialización del sistema	Modifica diversos valores en el sistema, como son los horarios de las clases y los tipos de errores.
Configuración del sistema	Establece los parámetros del sistema, como los horarios de los grupos, la asociación de grupo-profesor y los valores que pueden tener los reportes.
Administración de Usuarios "Profesor"	Permite modificar los datos individuales de los usuarios definidos con el perfil "Profesor".
Administración de Usuarios "Alumno"	Permite modificar los datos individuales de los usuarios definidos con el perfil "Alumno".
Consultas a la Base de Datos	Genera consultas directas a la base de datos, con un nivel de privilegios de jerarquía máxima. La consulta realizada genera también un reporte visual en función de parámetros definidos en la misma consulta.
Agregar Avisos al Sistema	Genera un mensaje que se puede visualizar en todo el sistema, tanto para profesores como para alumnos por separado.
Generar respaldo de la información	Genera un respaldo total de la información contenida en la base de datos. Se debe de ejecutar por lo menos al finalizar el ciclo escolar en curso.
Crear reporte	Permite la creación de un reporte con valores definidos por el mismo usuario.

Tabla 3.3. Descripción de funciones del nivel de usuario "Profesor".

Funciones de Profesor	Descripción
Crear reporte	Permite la creación de un reporte con valores definidos por el mismo usuario.
Consultas a la Base de Datos	Genera consultas directas a la base de datos, con un nivel de privilegios de jerarquía mínima, la cual no permite modificación de los valores de la base de datos salvo la de añadir reportes generados por el mismo usuario. La consulta realizada genera también un reporte en función de parámetros definidos en la misma consulta, siendo válidos sólo para sus propios grupos o del mismo día de otros profesores.

Tabla 3.4. Descripción de funciones del nivel de usuario "Alumno".

Funciones de Alumno	Descripción
Crear reporte	Permite la creación de un reporte con valores definidos por el mismo usuario.
Consultas a la Base de Datos	Genera consultas directas a la base de datos, con un nivel de privilegios de jerarquía mínima, la cual no permite modificación de los valores de la base de datos, salvo la de añadir reportes generados por el mismo usuario. Cada alumno solo podrá ver sus propios reportes realizados anteriormente.

3.1.3 ESTRUCTURA GENERAL

Partiendo de lo general a lo particular, tomando en cuenta en todo momento que el objetivo es diseñar un sistema modular, se plantea en primera instancia un diagrama de caja negra del SIRLI, en función de las entradas y salidas del sistema planteado, descritos en la **figura 3.2.** En el esquema se aprecian los módulos generales del SIRLI, planteando al Proceso de Negocio como aquel que realiza las funciones del SIRLI respecto a las entradas que le sean suministradas por los módulos correspondientes dada la actividad que se desee desempeñar por parte del usuario.

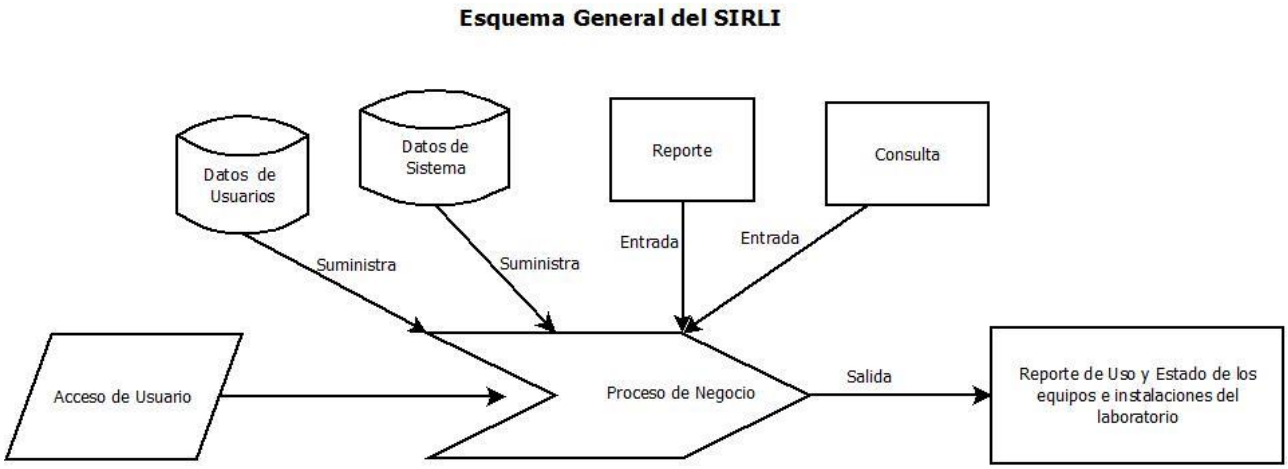


Figura 3.2. Esquema General del SIRLI

3.2 DISEÑO POR CAPAS

Para el desarrollo del SIRLI, tomando en cuenta las características físicas de los laboratorios, su entorno de red y los requerimientos establecidos por el usuario final, se determina el paradigma de diseño y desarrollo “por capas”.

A continuación se describen las capas de desarrollo y la arquitectura del SIRLI.

3.2.1 CAPA DE PRESENTACIÓN

Se contempló utilizar en esta capa herramientas de diseño tales como CSS y HTML, al igual que una interfaz simplificada, con colores de alto contraste que favorecieron a una mejor experiencia para el usuario en términos de facilidad de comprensión y uso del sistema.

La estructura básica para el desarrollo de la interfaz del SIRLI se muestra en la **figura 3.3.**

Cabe mencionar que se considera utilizar en la medida de lo posible, de acuerdo con la Coordinación de la Mediateca, imágenes institucionales de la UNAM, la ENP, el Plantel 9 y de la misma Mediateca, por ser un sistema de carácter institucional.

Figura 3.3. Modelado de la interfaz gráfica del SIRLI

3.2.2 CAPA DE NEGOCIO

En el diseño de la capa de negocio, conforme al análisis de requerimientos realizado en el capítulo anterior, se definieron los siguientes módulos funcionales:

- Sistema de Acceso: Verifica los datos de usuario para otorgar acceso al sistema.
- Administración de Usuarios: Gestiona la información de los usuarios.
- Búsqueda de reportes: Realiza el filtrado de reportes en función de los parámetros definidos por el usuario.
- Estadísticas de reportes: Realiza la selección de los reportes que cumplan con los parámetros definidos por el usuario y con ellos genera la gráfica correspondiente y la muestra en el sistema.
- Inicialización del sistema: Inicializa los parámetros del sistema a su estado por defecto, permitiendo generar antes un respaldo de la información completa del sistema.

- Administración de grupos: Gestiona la información de los grupos y la relación con el profesor correspondiente.
- Administración de horarios: Gestiona la información de los horarios y su relación con los grupos dados de alta en el sistema.
- Administración de parámetros de reportes: Maneja la información de los reportes que los usuarios pueden realizar, como lo son las opciones de dispositivos o de tipos de daños de sus equipos.
- Manejador de Noticias: Gestiona la información referente a las noticias y avisos que envía la coordinación de los laboratorios a los profesores y alumnos.
- Cierre de sesión: Realiza la función de cierre de sesión de usuarios.

Para la navegación del usuario a través de los distintos módulos se plantea un menú basado en CSS el cual sea multinivel para facilitar la interacción con el sistema.

Para cumplir con el requisito de código basado en un software libre y suficientemente robusto para cumplir con las características técnicas del sistema se decidió utilizar el lenguaje de programación PHP. Este lenguaje permite también la integración y comunicación con el sistema gestor de base de datos. Una ventaja adicional de PHP es la posibilidad de utilizar código embebido de este lenguaje en páginas basadas en HTML en el cual está basado el SIRLI. Se definen los bloques de la capa de negocio en la **figura 3.4.**

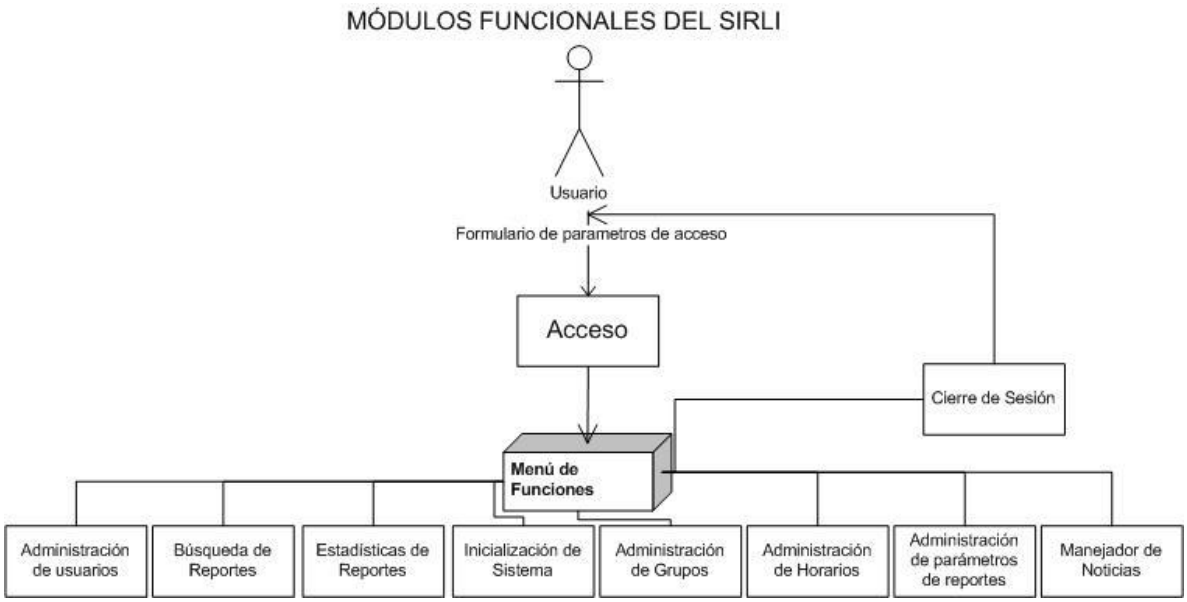


Figura 3.4. Módulos Funcionales del SIRLI

3.2.3 CAPA DE ADMINISTRACIÓN DE DATOS

El Sistema Gestor de Base de Datos seleccionado para actuar como el administrador de datos del SIRLI es MySQL, ya que, como se mencionó en el apartado anterior, permite una total integración con PHP y cumple con las necesidades lógicas en las que el sistema se desenvuelve.

El diccionario de datos de las tablas con las que trabaja el sistema es necesario ya que permite el análisis detallado de los datos, el flujo de la información y de los procesos que conforman el sistema el cual se presenta en la **figura 3.5** y la cardinalidad de las relaciones entre campos de las tablas de la base de datos se refieren a la forma en cómo los datos son procesados por el sistema (véase **figura 3.6**).

3.3 SEGURIDAD DEL SISTEMA

El SIRLI contempla mecanismos de seguridad para proteger la información almacenada en la base de datos.

La seguridad que el sistema requiere se da en tres tipos de orden:

- Lógica
- Por nivel de usuario
- Física

Para ello se establecieron, en primera instancia, en la capa de presentación algoritmos de validación de consultas SQL antes de aplicarse en la base de datos, con el fin de evitar ataques del tipo de Inyección SQL, que son muy comunes en la actualidad. Estos algoritmos se basan principalmente en bloquear caracteres especiales utilizados en sentencias SQL que pudieran comprometer la información almacenada.

Para el resguardo seguro de las contraseñas se utilizó un algoritmo de reducción criptográfico, en este caso el algoritmo de uso es Message-Digest Algorithm 5 o MD5. Este algoritmo permite que la contraseña pueda contener un número variable de caracteres, ya que se toma el vector de datos y lo resume en un bloque de 128 bits. De esta manera, las contraseñas de ingreso al sistema por parte de usuarios de jerarquía superior desde profesores hasta coordinadores quedan cifradas y resguardadas.

Otro factor que se contempló para la seguridad del sistema se estableció en el control de acceso por nivel de usuario, delimitando para cada usuario el acceso a las tablas de la base de datos. El diseño del sistema contempló que hasta los usuarios de jerarquía más baja (los alumnos) pudieran añadir información a la tabla de reportes, pero en el control de acceso a la base de datos se delimitaron las restricciones correspondientes para que estos usuarios solo ingresaran información en la tabla dedicada para ello, pero siendo inaccesibles para ellos las demás tablas de la base de datos.

Diccionario de Datos BDSIRLI

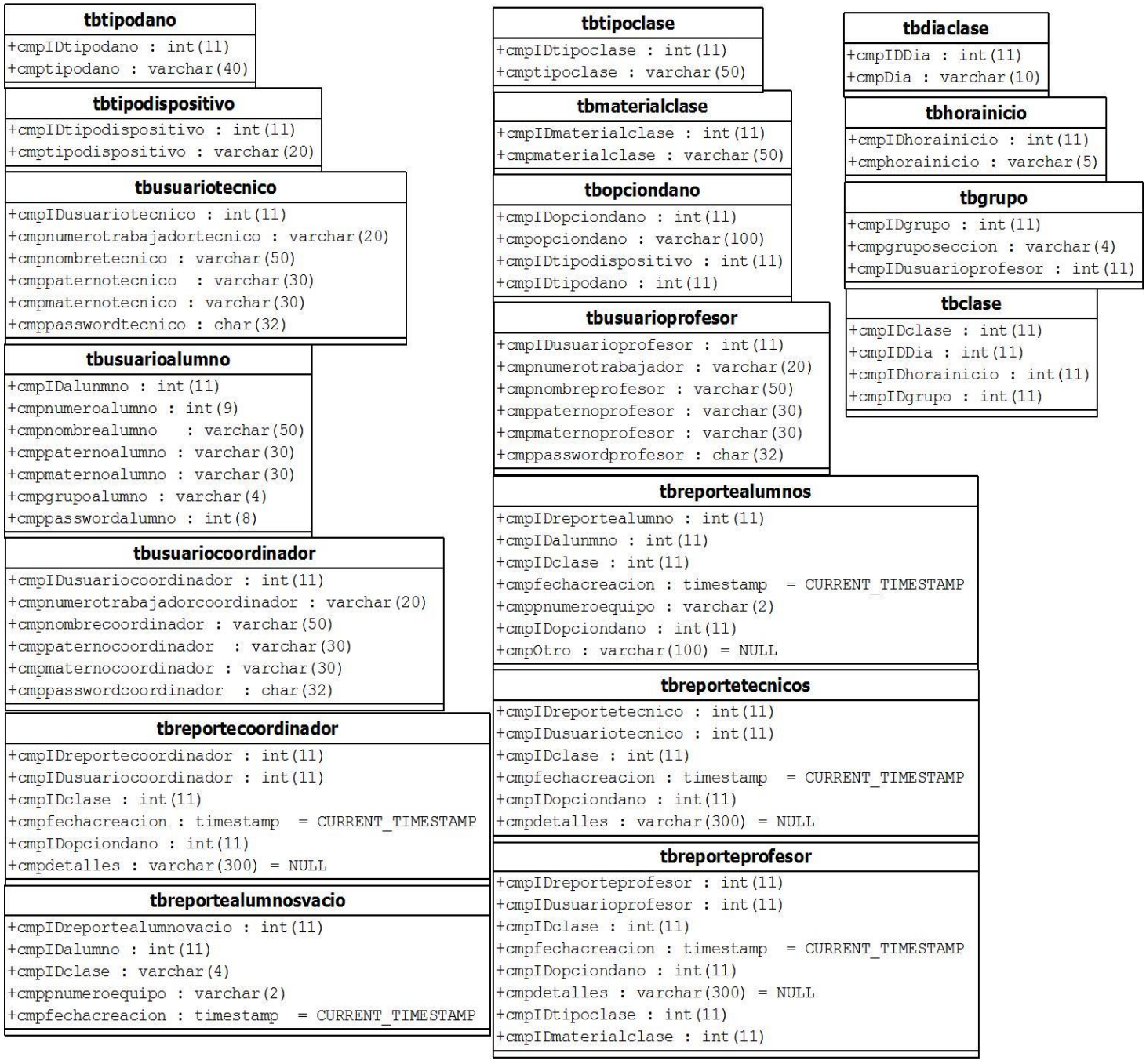


Figura 3.5. Diccionario de de Datos del SIRLI

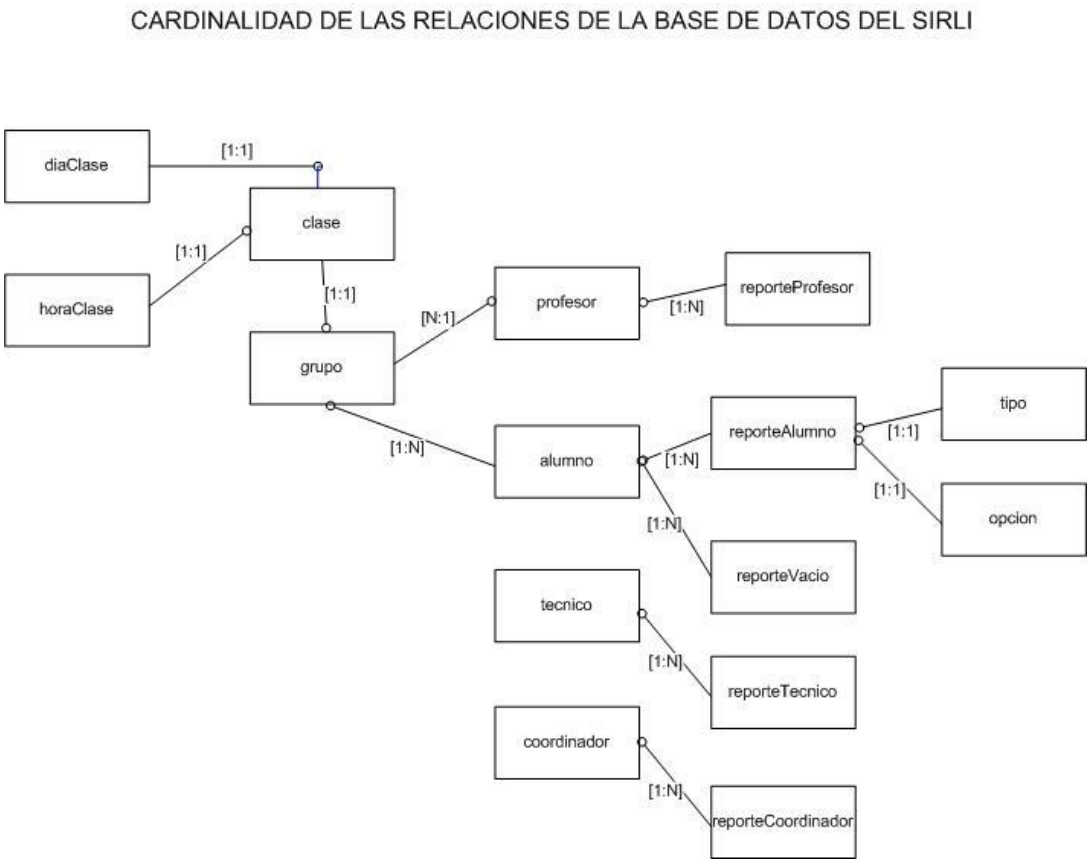


Figura 3.6. Cardinalidad de las tablas en la base de datos del sistema.

3.4 DISEÑO DEL PROCESO DE INSTALACIÓN

El diseño del SIRLI contempló el requerimiento de portabilidad, que establece la facilidad de instalación y configuración. Para lograr este objetivo se establecieron tres puntos básicos en los que consiste la instalación y la inicialización del sistema:

a) Configuración del Equipo Servidor: Para la configuración del Equipo Servidor bastó con la instalación guiada de la herramienta AppServ en su versión 2.5.10, que integra la configuración automatizada de Apache 2.2.8, PHP 5.2.6, MySQL 5.0.51b, y phpMyAdmin 2.10.3.

b) Creación de la Base de Datos: Se proporcionó un script PHP que permitiera, con su sola ejecución, inicializar la base de datos correspondiente al SIRLI, siendo necesario que se proporcionaran los datos de usuario y contraseña del sistema MySQL que correspondan al Administrador.

c) Inicialización del sistema: Dentro de los módulos funcionales para el perfil de usuario "Coordinador", se encuentra el módulo de Inicialización del SIRLI, el cual permite, de una manera segura, la restauración a valores por defecto del sistema.

En caso de que se desee desinstalar el sistema, basta con eliminar la base de datos correspondiente utilizando la herramienta phpMyAdmin desde el mismo Equipo Servidor, y borrar manualmente los archivos alojados en la carpeta del Servidor Web configurado al instalar la herramienta AppServ.

Los procesos de instalación y desinstalación del SIRLI están descritos en el Anexo 5.

Capítulo 4. Desarrollo del SIRLI

El diseño del SIRLI se realizó en el capítulo anterior, mismo que corresponde a un diseño estructurado, y en el presente capítulo, dicho diseño se llevó a la práctica mediante la metodología planteada por el Modelo de Desarrollo Incremental, dividiendo el desarrollo en módulos y considerando en cada módulo objetivos y especificaciones propias en función de las necesidades identificadas, de manera que, al unificarse todos los módulos y satisfacer los objetivos generales del sistema descritos en el análisis de requerimientos, se logró implementar un sistema integral de manejo de información del estado y funcionamiento de los Laboratorios de Idiomas. Como ya se mencionó en el capítulo 1, los paradigmas de programación establecidos para el desarrollo del proyecto son la programación estructurada, programación modular, programación por capas y programación orientada a objetos.

En la programación del SIRLI se establecieron como factores de calidad los siguientes elementos:

- *Corrección*. SIRLI debe realizar cada una de sus funciones de acuerdo al diseño establecido en las fases previas al desarrollo. Para evaluar este aspecto se definieron en los análisis previos especificaciones claras y completas, que debieron ser satisfechas en el momento de las pruebas, y en caso de no ser así, se analizó de manera exhaustiva la función en cuestión y se corrigió lo que fuera pertinente.

- *Claridad*. SIRLI se desarrolló con una perspectiva de código claro y bien documentado, tanto en sí mismo con el uso de comentarios, en los diagramas, en el análisis de diseño y en la documentación de desarrollo propia de esta fase. En este aspecto se tuvo en cuenta que uno de los objetivos principales de este proyecto es el escalamiento y actualizaciones realizadas por los mismos usuarios especializados (técnicos del plantel), por lo que el código se generó con sencillez y claridad en las sentencias declaradas en cada script del sistema.

- *Eficiencia*. SIRLI se desarrolló bajo el esquema de gestión de recursos eficiente, que por la naturaleza del sistema, que es bajo el paradigma Cliente-Servidor, se establecieron como recursos principales del sistema los mismos con los que cuenta el Servidor, sin dejar de lado los propios de cada cliente. En el análisis de la eficiencia del SIRLI se analizó con particularidad las consultas a la base de datos, siendo eficiente en un orden del número de 500,000 reportes almacenados en el sistema. En este sentido también se tomó en cuenta la necesidad de inicializar el sistema en cada inicio de ciclo escolar, teniendo para ello una herramienta en la interfaz para el Coordinador del Laboratorio y para los Técnicos, con los respectivos mecanismos de seguridad (Control de Acceso y Autentificación).

- *Portabilidad*. SIRLI es un sistema portable, ya que la arquitectura del sistema está basada en el lenguaje interpretado PHP, siendo el sistema independiente de la plataforma en la que se ejecuta. El SIRLI también permite portabilidad aun en casos de modificaciones del Sistema Operativo de las estaciones de trabajo, modificaciones de Hardware y actualizaciones de los dispositivos propios de la red de trabajo.

Tomando como elementos primordiales en el desarrollo del SIRLI los descritos anteriormente, se procedió a la codificación, que es el punto medular del proyecto.

A continuación se describe cada uno de los módulos que componen al SIRLI. La nomenclatura a utilizar para nombrar a los scripts PHP y HTML de dichos módulos tiene la siguiente estructura: "*nombre del módulo o función" "_" "nombre del perfil de usuario" "." "extensión del archivo*". Por ejemplo:

index_alumno.php – Hace referencia al módulo de inicio del perfil de usuario Alumno.

4.1 ESTRUCTURA DE ARCHIVOS DEL SIRLI

Partiendo del requerimiento del SIRLI como un Sistema Modular se propuso la estructura de archivos descrita en la **figura 4.1.**

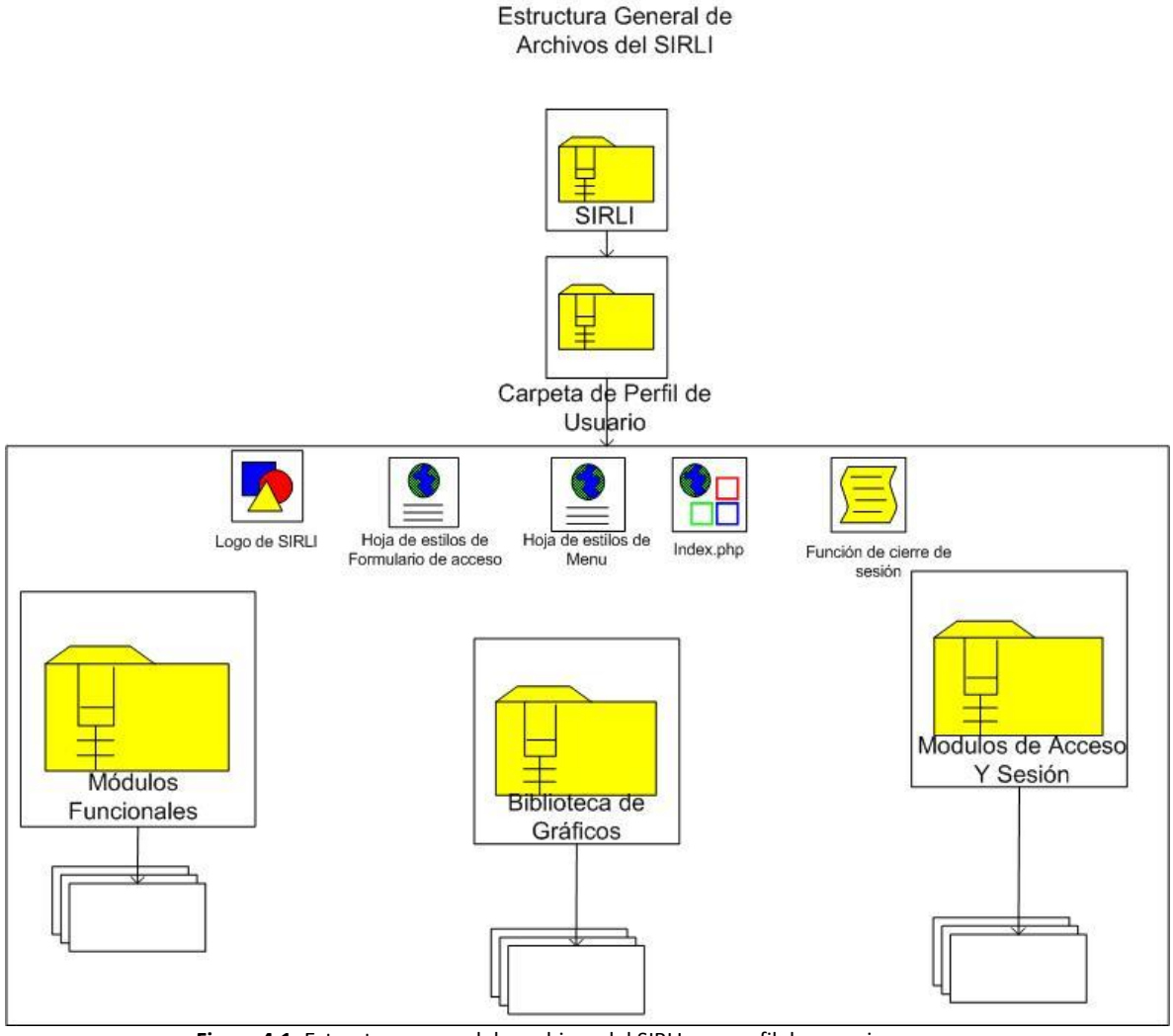


Figura 4.1. Estructura general de archivos del SIRLI por perfil de usuario.

Se decidió implementar este tipo de arquitectura modular porque permite una interpretación sencilla y práctica de la estructura y funcionamiento del sistema. Cada elemento cuenta con su respectiva organización, siendo fácilmente modificable desde el punto de vista de actualizaciones por parte de usuarios especializados, para su correspondiente escalamiento cuando se requiera.

En este tipo de desarrollo es importante tomar en cuenta que cada Perfil de Usuario cuenta con sus respectivos archivos de ejecución, por lo que si se hace una modificación en algún archivo sólo será válido para el perfil de usuario correspondiente a dicho script modificado.

En la implementación y codificación de los scripts del sistema se tomó como una medida de seguridad el acceso a los mismos, ya que cada script está protegido por medio de Sesiones propias del lenguaje de programación PHP para que, aun cuando algún usuario de privilegios inferiores (Profesor o Alumno) conozca la ruta en la que se encuentra el archivo en el Equipo Servidor, éste no pueda tener ningún tipo de acceso al código fuente correspondiente.

4.2 ESTRUCTURA DE LA BASE DE DATOS

A continuación se describe la estructura particular de la base de datos del SIRLI. El diagrama entidad-relación de la base de datos del SIRLI se muestra en la **figura 4.2.** Como se puede apreciar en el modelado físico, algunas relaciones necesarias para el funcionamiento del sistema se realizaron por medio del intercambio de datos de sesión, propias de los scripts PHP que lo componen, específicamente en los datos de usuario almacenados en las sesiones cuando se ingresa correctamente. Esto se hizo con la finalidad de ahorrar tiempo en consultas SQL lo que representa una mayor velocidad de respuesta y manejo de datos en el sistema.

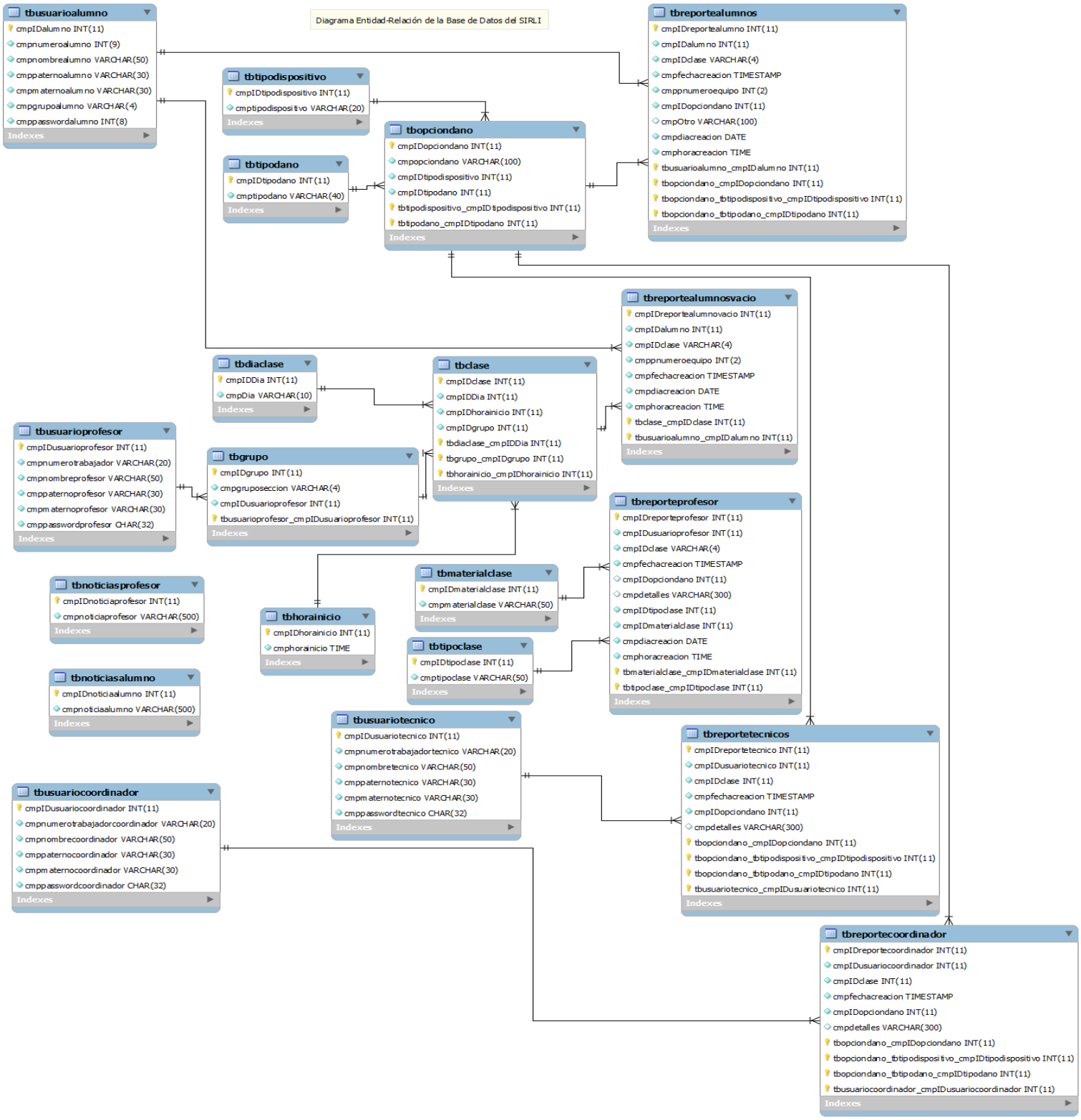


Figura 4.2. Diagrama Entidad-Relación de la Base de Datos del SIRLI.

4.3 INICIO Y CIERRE DE SESIÓN

Para iniciar sesión en el SIRLI, además del archivo de inicio correspondiente a cada tipo de usuario (para el caso del perfil Alumno, es index_alumno.php), son necesarios dos archivos más que contienen la información de acceso al servidor de base de datos y la función de verificación de identidad propia del sistema. Estos archivos se encuentran direccionados en la carpeta llamada *"Admin"* correspondiente para cada perfil de usuario. La información recibida por el formulario de acceso es comparada en una consulta SQL a la tabla de datos de usuario, previa codificación con el algoritmo MD5 de la contraseña proporcionada para que ésta no se envíe como dato en claro por el medio de transferencia. Si el proceso encuentra validez de datos entonces el sistema procede a dar el acceso con los respectivos privilegios del perfil del usuario autenticado.

Los datos del usuario son almacenados a partir de ese momento en las variables globales de sesión propias del lenguaje PHP, y dado que ya se tiene la información del usuario no es necesario volver a hacer una consulta SQL. Con respecto a este punto, las relaciones lógicas de la base de datos se simplificaron para este caso en particular, los datos del usuario permanecen hasta que se cierre sesión o las mismas caduquen según la configuración del servidor PHP en el que esté implementado el sistema.

En la **figura 4.3** se observa la pantalla de inicio de sesión del SIRLI.

Figura 4.3. Pantalla de Inicio de Sesión del SIRLI.

4.4 CONFIGURACIÓN E INICIALIZACIÓN

El SIRLI cuenta con un modelo de base de datos flexible, por lo que sus valores de inicio y de operación son modificables y configurables según el requerimiento propio en un determinado caso de cambio de políticas en la organización. Para que este cambio no genere problemas en cuanto a integridad de los datos es recomendable realizarlos al inicio del ciclo de operación, en este caso particular del ciclo escolar. Si fuera necesario realizar algún cambio se recomienda hacer un respaldo completo de la información, e inicializar el sistema a sus valores por defecto, o también se puede realizar tomando en consideración las medidas necesarias de integridad de la información correspondientes a dicho cambio.

Para realizar los cambios de valores internos del SIRLI es necesario acceder como Técnico o Coordinador, y en el menú acceder a la opción llamada "*Sistema*". La pantalla correspondiente se puede observar en la **figura 4.4.** En dicha opción se pueden realizar cambios a los valores del sistema como son los horarios, asignación de grupos, opciones de reportes y tipos de daño registrados para los equipos del laboratorio. Se debe tener especial consideración si se elimina alguna opción, puesto que esto causaría que algunos datos en las estadísticas o en las consultas sean erróneos debido a incongruencia en la asociación de información. Agregar o modificar valores no representa problemas de este tipo.

Figura 4.4. Pantalla de menú de configuración del SIRLI.

El módulo de Inicialización permite al SIRLI regresar a sus valores por defecto, borrando los registros de los reportes y de los usuarios. Desde este apartado también es posible generar un respaldo general de la información, para su almacenamiento en un medio externo. Este módulo posee la característica de confirmación exhaustiva por parte del usuario además de solicitar nuevamente los datos de sesión, debido al alto impacto que se puede generar al sistema.

Este módulo sólo es accesible por usuarios con el perfil de Técnico o Coordinador, los cuales tienen los privilegios correspondientes en el manejador de base de datos.

En la **figura 4.5** se observa la pantalla del módulo de Inicialización del SIRLI para el perfil de usuario de Coordinador.

Figura 4.5. Pantalla de Inicialización del SIRLI en el Perfil de Usuario de Coordinador.

4.5 EDICIÓN DE USUARIOS

El módulo de edición de datos de usuario se encuentra limitado a los perfiles de usuario de Técnico y Coordinador. El usuario Coordinador es el único capaz de generar tanto un nuevo usuario Coordinador, como también usuarios del perfil Técnico, siendo indispensable que cuando se inicialice el sistema o se use por

primera vez el Coordinador sea quien genere nuevos usuarios. Para el primer uso se facilita en la documentación correspondiente un nombre de usuario y una contraseña, datos que en el momento de utilizar deberán cambiar por los propios y volver a ingresar al sistema. Esto se hace con el fin de fortalecer los mecanismos de seguridad de control de acceso y autenticación propios del SIRLI.

El usuario con perfil de Técnico es capaz de modificar o eliminar datos de usuarios con jerarquía organizacional inferior, no así sus propios datos o de otro usuario de su mismo perfil.

Los usuarios con perfil de Profesor o Alumno no tienen acceso a este módulo, pero sí les es posible ver los datos con los que están dados de alta en el sistema, y en caso de notar algún error notificarlo a la Coordinación de los Laboratorios de Idiomas para su corrección, situación que sí le es posible solucionar a un usuario Técnico.

La modificación por error en algún dato no representa riesgo para la integridad del sistema, no así el borrar directamente a usuarios, ya que los reportes generados por este usuario quedarían sin referencia lógica en la base de datos.

En la **figura 4.6** se observa la pantalla del módulo de Edición de Usuarios del SIRLI para el perfil de usuario de Coordinador.

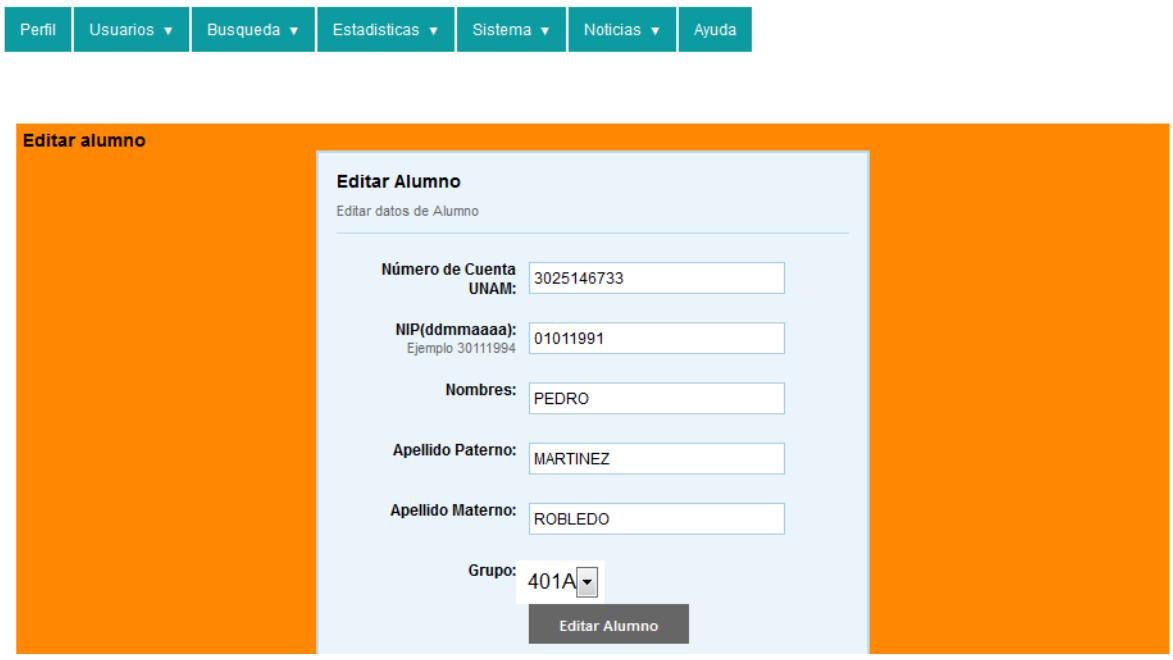


Figura 4.6. Pantalla de Edición de Usuarios del SIRLI en el Perfil de Usuario de Coordinador.

4.6 ELABORACIÓN DE REPORTES

El registro de los reportes de los usuarios del tipo Coordinador, Técnico y Profesor tienen una tabla dedicada para este propósito en la base de datos, almacenando los datos de usuario, fecha de creación, grupo, tipo de daño, y un campo abierto limitado a 100 caracteres para detallar la incidencia. Se toma en consideración que la cantidad de reportes de este tipo es considerablemente menor a la que generen los usuarios del tipo Alumno, dada que es mayor la cantidad de usuarios que representa este último.

Por su parte, la elaboración de reportes para los alumnos se divide en dos tipos:

- Reporte Vacío
- Reporte Completo

El tipo de reporte “Vacío” tiene como finalidad registrar los reportes de los usuarios que no tienen ningún problema en su estación de trabajo en una sesión de clase determinada. El reporte contiene la información del usuario, el grupo, el número de equipo, la fecha y hora en la que se realizó el reporte, utilizando una cantidad menor de información que requeriría el reporte completo, optimizando espacio de almacenamiento en el equipo servidor y siendo un filtrado distintivo básico para eficiencia del SIRLI en el momento de evaluar el estado de los equipos informáticos en el análisis y elaboración de los planes de mantenimiento de los Laboratorios Multimedia.

El tipo de reporte “Completo” registra en una tabla dedicada en la base de datos del sistema la información del usuario, el grupo, el número de equipo, la fecha, hora, y el tipo de daño en su estación de trabajo que desea reportar. Este tipo de reporte contiene información específica que permite documentar con datos puntuales y específicos el tipo de daño que presenta la estación de trabajo, permitiendo al técnico analizar y planificar el mantenimiento correspondiente al conjunto de problemas presentados en el laboratorio.

En la **figura 4.7** se observa la pantalla del módulo de Elaboración de Reportes de Alumnos del SIRLI.

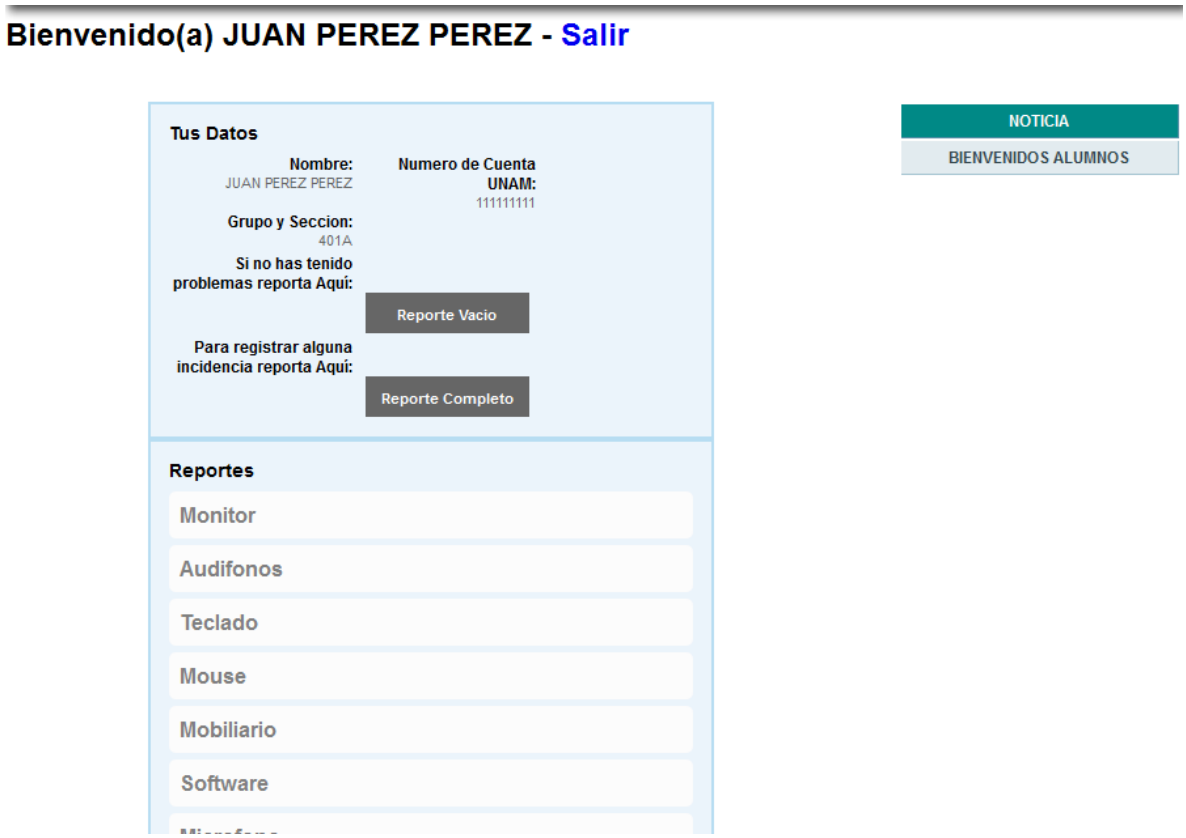
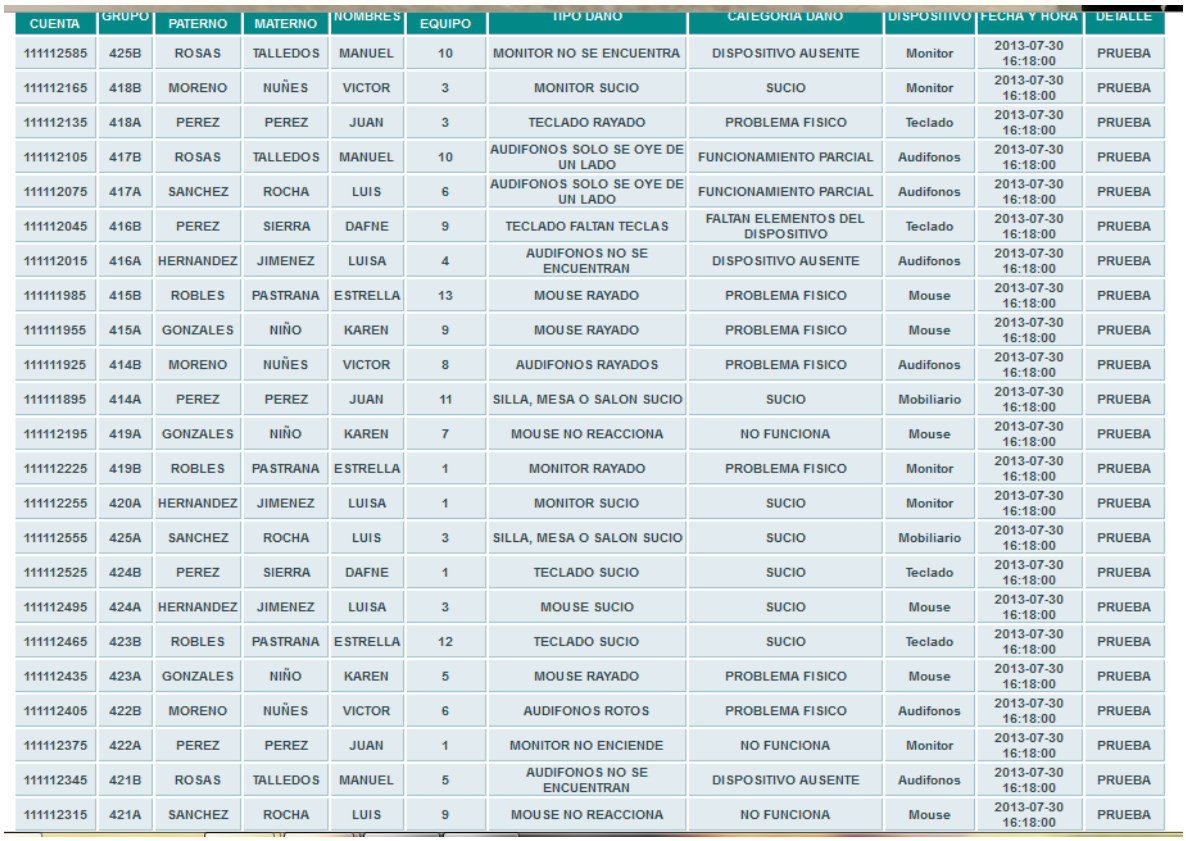


Figura 4.7. Pantalla de Elaboración de Reportes de Alumnos del SIRLI.

4.7 CONSULTA DE REPORTES

El módulo de consulta de reportes genera una tabla HTML con los datos de los reportes que cumplan con los criterios de búsqueda especificados por el usuario en el formulario desarrollado para tal fin. Para este módulo se limita el número de reportes máximo que muestra el SIRLI en 100, permitiendo que, si en una consulta el número de reportes coincidentes es grande, se refine el criterio de selección para obtener datos precisos y de esta forma generar información útil y significativa para el usuario, en lugar de mostrar una lista demasiado grande que no permita una correcta interpretación de los datos.



CUENTA	GRUPO	PATERNO	MATERNO	NOMBRES	EQUIPO	TIPO DAÑO	CATEGORIA DAÑO	DISPOSITIVO	FECHA Y HORA	DETALLE
111112585	425B	ROSAS	TALLEDOS	MANUEL	10	MONITOR NO SE ENCUENTRA	DISPOSITIVO AUSENTE	Monitor	2013-07-30 16:18:00	PRUEBA
111112165	418B	MORENO	NUÑES	VICTOR	3	MONITOR SUCIO	SUCIO	Monitor	2013-07-30 16:18:00	PRUEBA
111112135	418A	PEREZ	PEREZ	JUAN	3	TECLADO RAYADO	PROBLEMA FISICO	Teclado	2013-07-30 16:18:00	PRUEBA
111112105	417B	ROSAS	TALLEDOS	MANUEL	10	AUDIFONOS SOLO SE OYE DE UN LADO	FUNCIONAMIENTO PARCIAL	Audifonos	2013-07-30 16:18:00	PRUEBA
111112075	417A	SANCHEZ	ROCHA	LUIS	6	AUDIFONOS SOLO SE OYE DE UN LADO	FUNCIONAMIENTO PARCIAL	Audifonos	2013-07-30 16:18:00	PRUEBA
111112045	416B	PEREZ	SIERRA	DAFNE	9	TECLADO FALTAN TECLAS	FALTAN ELEMENTOS DEL DISPOSITIVO	Teclado	2013-07-30 16:18:00	PRUEBA
111112015	416A	HERNANDEZ	JIMENEZ	LUISA	4	AUDIFONOS NO SE ENCUENTRAN	DISPOSITIVO AUSENTE	Audifonos	2013-07-30 16:18:00	PRUEBA
111111985	415B	ROBLES	PASTRANA	ESTRELLA	13	MOUSE RAYADO	PROBLEMA FISICO	Mouse	2013-07-30 16:18:00	PRUEBA
111111955	415A	GONZALES	NIÑO	KAREN	9	MOUSE RAYADO	PROBLEMA FISICO	Mouse	2013-07-30 16:18:00	PRUEBA
111111925	414B	MORENO	NUÑES	VICTOR	8	AUDIFONOS RAYADOS	PROBLEMA FISICO	Audifonos	2013-07-30 16:18:00	PRUEBA
111111895	414A	PEREZ	PEREZ	JUAN	11	SILLA, MESA O SALON SUCIO	SUCIO	Mobiliario	2013-07-30 16:18:00	PRUEBA
111112195	419A	GONZALES	NIÑO	KAREN	7	MOUSE NO REACCIONA	NO FUNCIONA	Mouse	2013-07-30 16:18:00	PRUEBA
111112225	419B	ROBLES	PASTRANA	ESTRELLA	1	MONITOR RAYADO	PROBLEMA FISICO	Monitor	2013-07-30 16:18:00	PRUEBA
111112255	420A	HERNANDEZ	JIMENEZ	LUISA	1	MONITOR SUCIO	SUCIO	Monitor	2013-07-30 16:18:00	PRUEBA
111112555	425A	SANCHEZ	ROCHA	LUIS	3	SILLA, MESA O SALON SUCIO	SUCIO	Mobiliario	2013-07-30 16:18:00	PRUEBA
111112525	424B	PEREZ	SIERRA	DAFNE	1	TECLADO SUCIO	SUCIO	Teclado	2013-07-30 16:18:00	PRUEBA
111112495	424A	HERNANDEZ	JIMENEZ	LUISA	3	MOUSE SUCIO	SUCIO	Mouse	2013-07-30 16:18:00	PRUEBA
111112465	423B	ROBLES	PASTRANA	ESTRELLA	12	TECLADO SUCIO	SUCIO	Teclado	2013-07-30 16:18:00	PRUEBA
111112435	423A	GONZALES	NIÑO	KAREN	5	MOUSE RAYADO	PROBLEMA FISICO	Mouse	2013-07-30 16:18:00	PRUEBA
111112405	422B	MORENO	NUÑES	VICTOR	6	AUDIFONOS ROTOS	PROBLEMA FISICO	Audifonos	2013-07-30 16:18:00	PRUEBA
111112375	422A	PEREZ	PEREZ	JUAN	1	MONITOR NO ENCIENDE	NO FUNCIONA	Monitor	2013-07-30 16:18:00	PRUEBA
111112345	421B	ROSAS	TALLEDOS	MANUEL	5	AUDIFONOS NO SE ENCUENTRAN	DISPOSITIVO AUSENTE	Audifonos	2013-07-30 16:18:00	PRUEBA
111112315	421A	SANCHEZ	ROCHA	LUIS	9	MOUSE NO REACCIONA	NO FUNCIONA	Mouse	2013-07-30 16:18:00	PRUEBA

Figura 4.8. Pantalla de Consulta de Reportes de Alumnos del SIRLI.

Este módulo permite llevar un seguimiento de cada dispositivo del laboratorio, ya que el formulario de búsqueda contempla parámetros de búsqueda concretos y específicos, lo que permite una localización precisa de información de interés para los coordinadores y técnicos, como el historial de incidencias por dispositivo o grupo de dispositivos, grupo, alumno, profesor, fechas, horas, tipo y categoría de daño. También permite llevar un control específico de los alumnos que han reportado el estado de sus equipos, documentando y facilitando el seguimiento de reportes para atender de manera oportuna, sistemática y puntual cada incidencia reportada.

En la **figura 4.8** se observa la pantalla del módulo de Consulta de Reportes de Alumnos del SIRLI.

4.8 GENERADOR DE ESTADÍSTICAS

Para la generación de este módulo se utiliza como herramienta de generación de gráficos la librería Open Flash Chart la cual cuenta con licencia LGPL, esto permite que el SIRLI siga cumpliendo con la especificación de ser completamente de código abierto. La herramienta permite graficar arreglos de datos con formato JSON el cual es flexible y ligero, y permite organizar datos obtenidos de una

consulta SQL, para que, con ayuda de la biblioteca PHP de esta herramienta y con la correspondiente generación de código que permita obtener dichos datos, grafique y muestre la asociación de valores y relaciones que se le indiquen en la interfaz del SIRLI, requiriendo para ello tener instalado en el navegador web cliente el software Adobe Flash Player, el cual está considerado en el Análisis de Requerimientos como software con el que cuentan los equipos de los Laboratorios de Idiomas.

El SIRLI es capaz de graficar y mostrar valores y asociaciones de incidencias en los Laboratorios, lo que permite a los Técnicos visualizar información que de otro modo sería compleja analizar, y así estos puedan realizar un análisis correcto en cuanto a la planeación del mantenimiento a los equipos computacionales, permitiendo también modificar el enfoque de un mantenimiento reactivo a uno proactivo, disminuyendo el número de incidencias en los equipos y contribuyendo en elevar la calidad de atención y servicio prestada en los Laboratorios de Idiomas, incidiendo directamente en el proceso de formación académica de la comunidad estudiantil.

En la **figura 4.9** se observa la pantalla del módulo de Generación de Estadísticas del SIRLI.

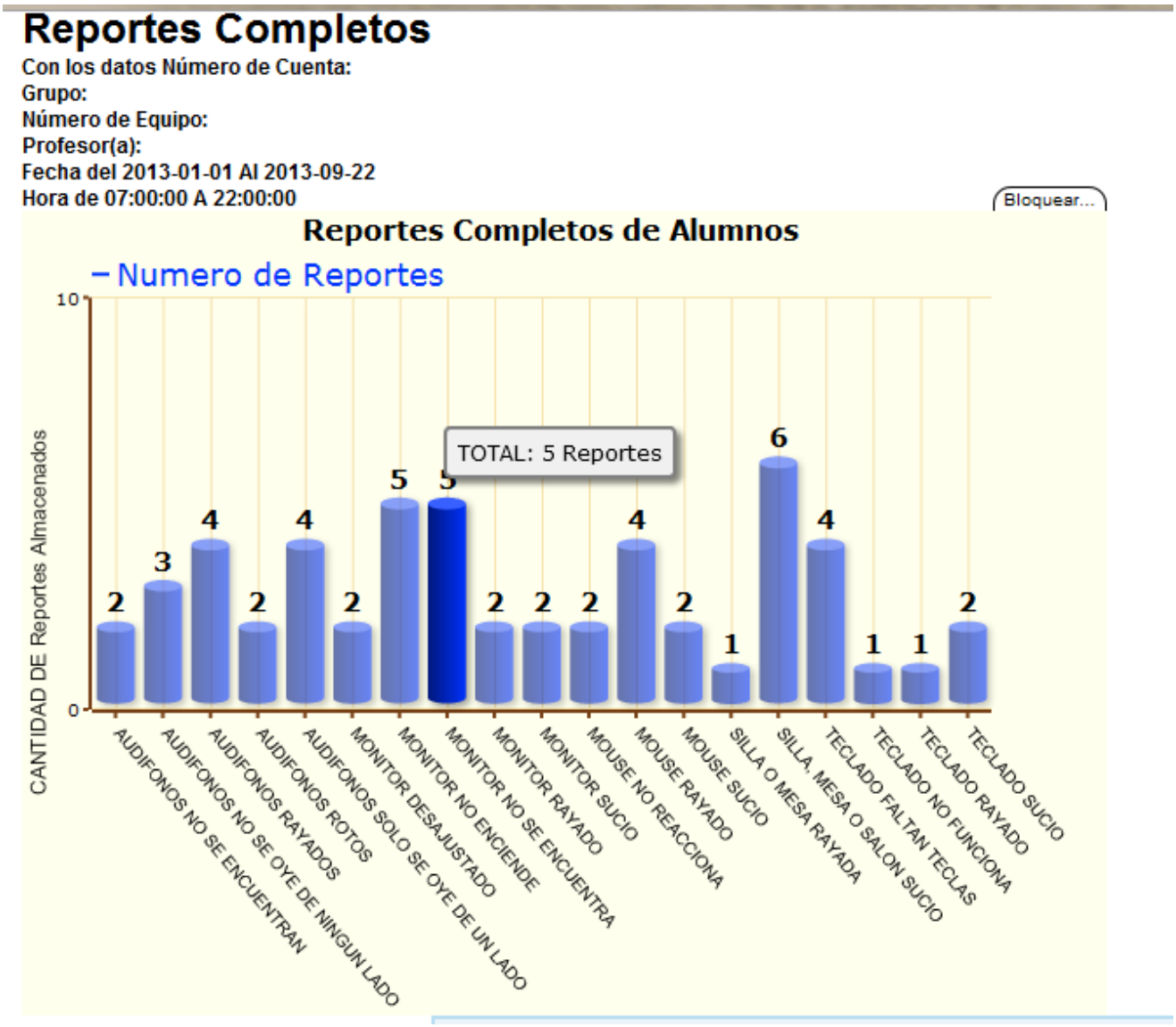


Figura 4.9. Pantalla de Generación de Estadísticas del SIRLI.

Capítulo 5. Pruebas

En este capítulo se describen las pruebas realizadas al sistema para localizar errores no detectados en la fase de desarrollo del SIRLI.

Durante el proceso de desarrollo se implementaron pruebas de caja blanca individualmente para cada variable utilizada, a manera de ir corrigiendo errores en la codificación. Por consiguiente, cada módulo se encuentra verificado en cuanto a las variables programadas para cada función, por lo que este capítulo describe las pruebas de caja negra correspondientes.

Las pruebas de caja negra descritas a continuación se implementaron tanto en el entorno de desarrollo como en los servidores de operación en los cuales se alojará el SIRLI, contando con la presencia de los técnicos del plantel, y presentando los resultados al coordinador de Mediateca quien tiene a su cargo la responsabilidad de mayor jerarquía en los laboratorios.

Para la planeación de las pruebas que se aplicaron al sistema evaluador, se integraron los distintos tipos de pruebas que se explicarán en detalle a continuación.

5.1 CONTENIDO

En las pruebas de contenido, se verificó que las palabras usadas en la interfaz del sistema para transmitir una idea al usuario fueran las adecuadas y que la idea transmitida fuera la misma. En el desarrollo de esta prueba también se analizó la navegabilidad del SIRLI, esto es, la facilidad para desplazarse entre las diferentes funciones del sistema, tomando como punto de referencia el Menú correspondiente. En la puesta en práctica de estas pruebas se analizaron cuatro casos de uso, en las cuales el Usuario buscó cumplir con una tarea específica dentro del sistema sin ningún tipo de asistencia por parte del desarrollador.

Para las pruebas funcionales del perfil de usuario Alumno, se analizó la función de “Reporte Completo”, en la que el usuario tiene como objetivo realizar un reporte dentro del módulo correspondiente en el SIRLI de un supuesto daño en su equipo. Esta prueba se realizó en un aula externa a los laboratorios de idiomas, con 15 usuarios que se encuentran en el rango de edad de entre 15 a 18 años (semejantes a la edad de los alumnos de la preparatoria que utilizarán el SIRLI como usuarios finales).

Para las pruebas funcionales del perfil de Usuario Profesor se contó con el apoyo de un profesor de la preparatoria que utilizó el sistema para ver los reportes de sus alumnos y sus datos generales de usuario. Esta prueba se realizó en un aula externa a los laboratorios de idiomas. Lo que se esperaba del profesor que realizara esta prueba es que él fuera capaz de realizar un reporte del estado de su equipo y desconectarse del SIRLI en poco tiempo.

Las pruebas funcionales para el perfil de Usuario “Técnico” y “Coordinador” se realizaron de manera simultánea con los dos técnicos responsables de los

laboratorios y el coordinador de los mismos. Dado que estos son usuarios finales del SIRLI se estableció que durante las pruebas el desarrollador y los usuarios comentaron directamente las funciones, probando diferentes parámetros y utilizando todas las funciones propias del perfil de usuario, por lo que no se estableció un tiempo máximo para realizarlas.

5.2 INTEGRACIÓN

Como ya se mencionó, el conjunto de pruebas de integración verifican que cada elemento encaja de forma adecuada y que se alcanza la funcionalidad y el rendimiento del sistema total. La prueba del sistema está constituida por una serie de pruebas diferentes cuyo propósito primordial es ejercitar profundamente el sistema.

El parámetro de jerarquía utilizado es la invocación de un script a otro de acuerdo a su posición en el árbol de la aplicación, donde se tiene como punto de partida los archivos de index_*"perfil_de_usuario_correspondiente"*.php como elemento base. Las jerarquías de scripts por niveles se encuentran de la **figura 5.1** a la **figura 5.4**.

Durante las pruebas de contenido los usuarios utilizaron diversos módulos en su interacción con el sistema, probando cada perfil de usuario y haciendo un uso exhaustivo del sistema.

Figura 5.1. Jerarquía de scripts para módulos de usuario Coordinador.

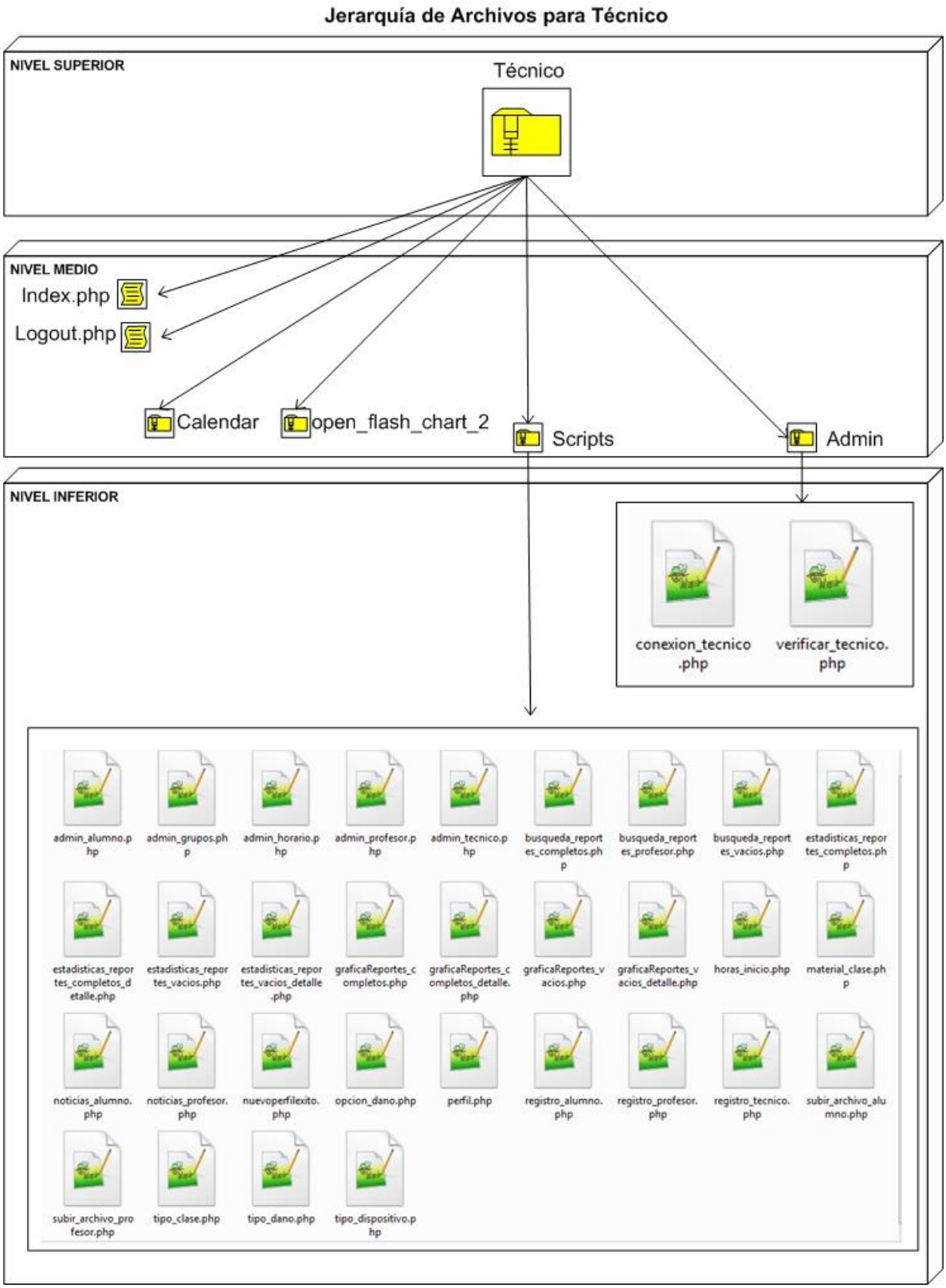


Figura 5.2. Jerarquía de scripts para módulos de usuario Técnico.

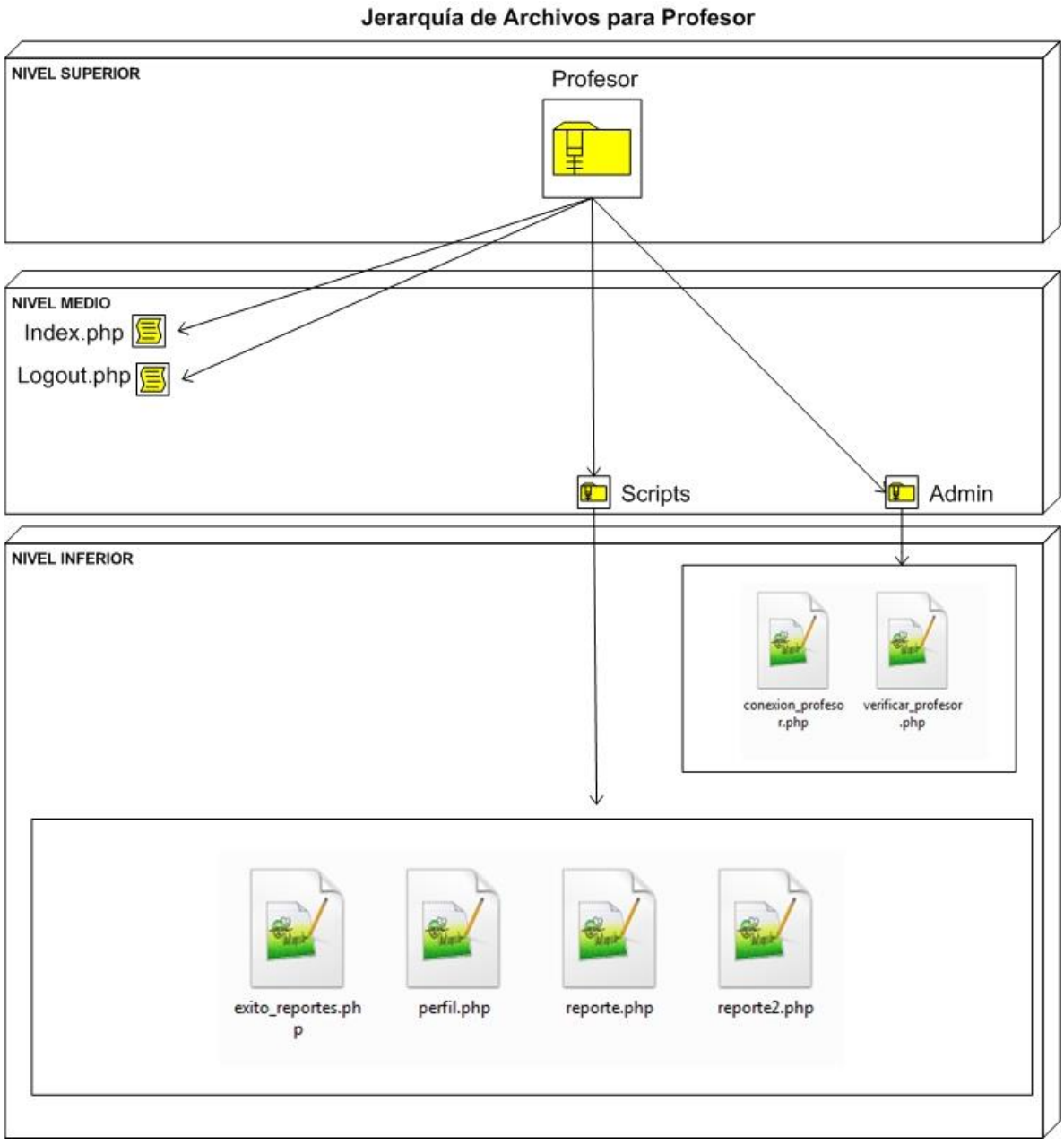


Figura 5.3. Jerarquía de scripts para módulos de usuario Profesor.

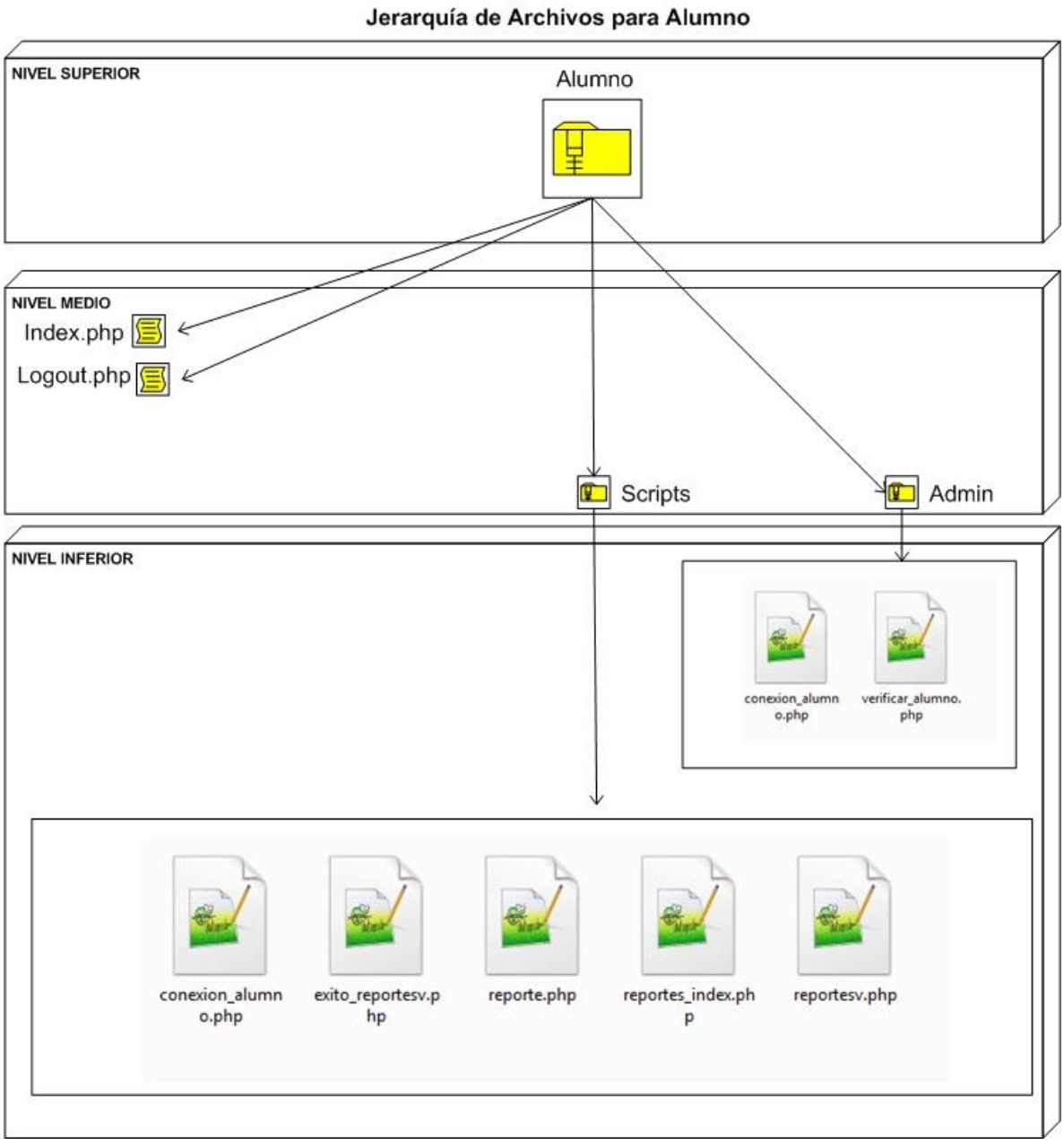


Figura 5.4. Jerarquía de scripts para módulos de usuario Alumno.

5.3 RENDIMIENTO

El objetivo de las pruebas de rendimiento es verificar y validar los requerimientos de rendimiento que se han especificado en análisis de requerimientos.

La técnica empleada para la prueba de rendimiento consistió en modificar la base de datos, incrementar el número de transacciones entre la base de datos y los scripts con sentencias SQL y validar el tiempo de respuesta en los scripts. En este caso el tiempo de respuesta esperado se estableció en 5 segundos como máximo. Este valor se estableció como parámetro óptimo de eficiencia para un entorno de volumen de datos de 250000 reportes completos, 2850 usuarios totales y 30 usuarios múltiples como base, de acuerdo al análisis de requerimientos.

Se desarrollaron las siguientes actividades en la aplicación de las pruebas de rendimiento:

- Se utilizaron los procedimientos de prueba desarrollados para las pruebas de integración.
- Se modificaron los archivos de datos (para incrementar el número de transacciones) y los scripts para incrementar el número de veces que ocurre cada transacción.
- Se ejecutaron los scripts en una máquina y deben ser repetidos con múltiples clientes (virtuales y actuales). Los clientes utilizados son ejecutados bajo entornos con sistemas operativos Windows y Linux/Unix para garantizar la operatividad del SIRLI en múltiples tipos de sistemas.

5.4 SEGURIDAD

Se definieron los objetivos de las pruebas en dos niveles de seguridad para todos los perfiles de usuario:

- Nivel de seguridad de la aplicación: Verifica que un actor sólo pueda acceder a las funciones y datos que su usuario tiene permitido.
- Nivel de Seguridad del Sistema: Verificar que sólo los actores con acceso al sistema y a la aplicación están habilitados para accederla.

El objetivo de esta prueba es evaluar el funcionamiento correcto de los controles de seguridad del sistema para asegurar la integridad y confidencialidad de los datos. El foco principal es probar la vulnerabilidad del sistema frente a accesos o manipulaciones no autorizadas. Una manera de encontrar esos casos de prueba es estudiar problemas conocidos de seguridad en sistemas similares y

tratar de mostrar la existencia de problemas parecidos en el sistema que se examina.

Algunas consideraciones de prueba son:

- Controles de acceso físico
- Existencia de datos confidenciales en reportes y pantallas
- Controles manuales, incluyendo aquellos para autorización y aprobación, formularios, documentación numerada y transmisión de datos.
- Controles automáticos, incluyendo aquellos para edición de datos, errores del operador, acceso a datos elementales y acceso a funciones.

Las técnicas empleadas en las pruebas se describen a continuación:

- Funciones / Seguridad de Datos: Identificar cada tipo de usuario y las funciones y datos a los que se debe autorizar.
- Crear pruebas para cada tipo de usuario y verificar cada permiso, creando transacciones específicas para cada tipo de usuario.
- Modificar tipos de usuarios y volver a ejecutar las pruebas. En cada caso, verificar si los datos o funciones adicionales quedan correctamente permitidos o denegados.
- Acceso al sistema.

Como consideración especial se tomó en cuenta que el acceso al sistema debe ser revisado y discutido con los administradores de la red y/o del sistema, en este caso los técnicos de los laboratorios. Esta prueba puede no ser requerida como tal, sino como una función de los técnicos del plantel.

Capítulo 6. Resultados

6.1 RESULTADOS DE LAS PRUEBAS

A continuación se presentan los resultados obtenidos para las pruebas de contenido, integración, rendimiento y seguridad.

A) Pruebas de contenido

Los resultados de la prueba de Contenido para los alumnos se muestran en la **tabla 6.1.**

Tabla 6.1. Resultados de la Prueba de Contenido para el Perfil de Usuario "Alumno".

Prueba	Tiempo (s)	Resultado
1	180	Incompleto
2	78	Completo
3	50	Completo
4	53	Completo
5	45	Completo
6	37	Completo
7	40	Completo
8	60	Completo
9	47	Completo
10	40	Completo
11	30	Completo
12	49	Completo
13	43	Completo
14	45	Completo
15	53	Completo

Los resultados muestran que sólo un usuario de entre quince no pudo completar la prueba satisfactoriamente.

Los resultados de las pruebas para el profesor muestran que el profesor la realizó satisfactoriamente con un tiempo de 58 segundos, comentando haber entendido correctamente el funcionamiento del sistema.

Los resultados de las pruebas de contenido muestran que el nivel de facilidad de interacción con el SIRLI es satisfactorio, ya que las diferentes acciones puestas a prueba en todos los perfiles de usuario pudieron realizarse de manera exitosa y con un tiempo de realización dentro del intervalo estimado.

B) Pruebas de integración

Los resultados de esta prueba muestran una correcta integración de los módulos para cada perfil de usuario, detectando que no existe ninguna referencia extraviada en el árbol estructural del sistema.

Cabe mencionar que en el transcurso de las pruebas se encontró que el script de calendario utilizado en los formularios para determinar fechas presenta un error de posicionamiento de elementos para fechas posteriores al año 2060, por lo que se establece como fecha máxima de esta herramienta el año 2059 dentro de sus parámetros internos. Al corregir dicho valor se comprueba que este script opera sin problemas.

En todos los casos las referencias e interacción entre módulos mostraron una integración completa y sin referencias pérdidas o nulas en el SIRLI. Como complemento y de acuerdo al árbol estructural se verificó que la interacción entre módulos es completa en ambos sentidos, de nivel superior a nivel inferior y viceversa.

C) Pruebas de rendimiento

Los resultados de las pruebas se indican para cada tipo de transacción:

- Transacción Única: Se completaron las pruebas sin ninguna falla y dentro del tiempo esperado o requerido por transacción.
- Múltiples transacciones: múltiples usuarios. Se completaron las pruebas de los scripts sin ninguna falla y dentro del tiempo esperado. Cabe mencionar que el parámetro de número de múltiples usuarios es de 30 por el número de alumnos máximo que utilizan cada laboratorio en cada sesión.

Los tiempos de respuesta obtenidos en las pruebas de “búsqueda de reportes completos” se describen en la **tabla 6.2**. El script PHP que realiza esta función delimita a un máximo de 100 registros como resultado máximo, para mantener la eficiencia del tiempo de respuesta del SIRLI en un máximo de un segundo. Además esta limitación se establece en función del requerimiento de mantener al SIRLI simplificado en su operación y respuesta, considerando que una solicitud que arroje más de 100 resultados debería de ser más específica en sus parámetros de búsqueda por parte del usuario para obtener información significativa en la consulta solicitada.

Tabla 6.2. Resultados de la Prueba de Rendimiento para la función "Búsqueda de Reportes Completos".

Número de Reportes Insertados	Tiempo 1 (s)	Tiempo 2 (s)	Tiempo 3 (s)	Tiempo promedio (s)
100	0.0089	0.0094	0.0064	0.0082

Para realizar la prueba del tiempo de respuesta del SIRLI en la función "Estadísticas de Reportes Completos" se insertó un número de reportes determinados para documentar el tiempo de respuesta en cada caso. En esta prueba se tomó en cuenta que el SIRLI está diseñado para trabajar con un volumen de datos de 250,000 reportes como máximo para cada uno de los dos tipos de reporte (500,000 reportes en total), de acuerdo al análisis de requerimientos. Los tiempos de respuesta obtenidos en esta prueba se describen en la **tabla 6.3**.

El resultado de ambas pruebas muestra que el SIRLI responde de manera satisfactoria para el tiempo establecido como parámetro de eficiencia de menos de cinco segundos en un entorno de volumen de datos máximo esperado.

Tabla 6.3. Resultados de la Prueba de Rendimiento para la función "Estadísticas de Reportes Completos".

Número de Reportes Insertados	Tiempo 1 (s)	Tiempo 2 (s)	Tiempo 3 (s)	Tiempo promedio (s)	Tamaño (KB)
100	0.0089	0.0090	0.0047	0.0075	0.0007
1000	0.0409	0.0417	0.0397	0.0408	0.0746
10000	0.2371	0.2369	0.2337	0.2359	0.623
100000	2.3354	2.2682	2.2140	2.2725	5,731
250000	4.7404	4.9698	4.7237	4.8113	14,806

D) Pruebas de seguridad

Las pruebas de seguridad del sistema garantizaron que solamente aquellos usuarios autorizados a acceder al sistema fueron capaces de ejecutar las funciones del sistema a través de los mecanismos apropiados. Por ejemplo, cada usuario con perfil "Técnico" aún cuando está autorizado a editar nuevas cuentas del mismo perfil o inferiores, sólo el Coordinador (nivel de jerarquía máximo) puede borrarlas. Para la seguridad a nivel de datos, la prueba garantizó que un usuario "Profesor" puede ver toda la información de sí mismo y de los reportes generados

por sus alumnos en su clase, incluyendo gráficas; sin embargo, el usuario "Alumno" solamente puede generar reportes a su nombre, sin posibilidad de ver o borrar los propios o de otros usuarios con su mismo nivel de privilegios.

Los resultados de las pruebas indican que para cada tipo de usuario conocido, las funciones y datos apropiados y todas las transacciones funcionan como se esperaba.

6.2 ESQUEMA DE REPORTES ACTUAL

A partir de la implementación del SIRLI se ha puesto en marcha un nuevo protocolo de generación de reportes por parte de los usuarios de los laboratorios de idiomas. Partiendo del esquema anterior al SIRLI, en donde no había una documentación precisa y referencial de las fallas en la operación de los laboratorios de idiomas, en la actualidad se cuenta con un esquema completamente nuevo y eficiente, capaz de soportar un volumen de datos adecuado en función de la cantidad de información que se genera en cada ciclo escolar.

El nuevo esquema de reportes permite, además, un análisis gráfico del comportamiento y funcionalidad de los laboratorios en tiempo real y pudiendo ser diferenciado el análisis a partir de rangos de fechas determinados por los mismos administradores del sistema (Coordinador y Técnicos), permitiendo, además, idear un plan de mantenimiento proactivo para mantener cada laboratorio multimedia en un intervalo de operatividad óptimo que responda a las necesidades docentes de la institución.

La facilidad de reportar por parte de los alumnos y profesores hace al SIRLI un sistema de fácil comprensión y manipulación, permitiendo a los usuarios proporcionar la información del estado de los equipos en un tiempo de aproximadamente un minuto para los reportes detallados, haciendo que el tiempo de revisión por parte de los alumnos sea corto y permitiendo al docente concentrar el tiempo de clase en la actividad académica, además de que ya no es necesario que el Técnico interrumpa la clase, incidiendo directamente en la concentración del alumno en la clase.

El alumno cuenta con la opción de reportar en cualquier momento durante la clase, pudiendo ser inmediatamente después de encontrar algún problema en su equipo, o hasta el final de la clase, teniendo también la obligación de reportar si el sistema funcionó correctamente durante el transcurso de la clase, contando con un apartado especifico en el SIRLI para este fin.

Una característica nueva para el profesor es contar con un sistema que almacene los reportes que como docente él y sus pares generan en cada clase, con

el fin de llevar un registro de la funcionalidad de los laboratorios para un posible análisis por parte del Coordinador, con el propósito de mejorar la experiencia del docente en el uso de la tecnología con la que se cuenta en estos espacios multimedia.

El SIRLI también permite documentar el uso que se le da a los laboratorios por parte de los profesores, permitiendo además analizar y valorar el uso de cierto software aplicativo para futuros cambios o adquisiciones de los mismos.

6.3 GENERACIÓN DE ESTADÍSTICAS Y DOCUMENTACIÓN DEL ESTADO DE LOS LABORATORIOS

Una característica fundamental en el SIRLI es la generación de estadísticas diferenciadas en tiempo real del estado de los equipos.

La generación de estadísticas de manera visual y con relaciones cruzadas y filtradas permite a los Técnicos y al Coordinador contar con la información concreta y suficiente para analizar el funcionamiento de los equipos computacionales, permitiendo realizar proyecciones a corto y mediano plazo para la planeación de las etapas de mantenimiento proactivo en los laboratorios.

La característica de búsqueda específica o general de reportes permite documentar a manera de tabla dinámica HTML los reportes generados por rangos de fecha y opciones específicas como tipo de dispositivo, categoría de daño, grupo, alumno, número de equipo, entre otros. Esta característica es una de las herramientas más detalladas del SIRLI, ya que permite conocer el estado de cada equipo o incluso de cada dispositivo durante un rango de fechas establecidas desde la búsqueda, pudiendo además tener la historia del dispositivo desde el comienzo del ciclo escolar, dando una opción de análisis completo y suficiente en el mantenimiento de cada dispositivo y de cada equipo.

Una opción adicional es la de consultar los reportes en donde el alumno no encontró ningún problema en su equipo durante el transcurso de la clase, llamados dentro del sistema como “Reportes Vacios”. El análisis general de estos reportes y su comparación directa con los reportes de daño o falla permiten obtener un panorama completo del comportamiento y eficiencia de los laboratorios, otorgando parámetros para la valoración del uso general que se tiene en estos espacios, para potencializar y maximizar el uso de dichas herramientas informáticas en beneficio de la actividad académica.

La capacidad de filtrado y la respuesta ágil de las sentencias SQL declaradas en los scripts del sistema hacen del SIRLI un sistema robusto en su capacidad de respuesta, y eficaz y sencillo en su uso y operación por parte del usuario, independientemente de su nivel de jerarquía de privilegios.

6.4 CAMBIO DE POLÍTICAS DE SERVICIO EN LOS LABORATORIOS DE IDIOMAS

La implementación del SIRLI implica un análisis y revisión de las políticas de seguridad y operación de los Laboratorios de Idiomas, ya que ahora la responsabilidad de la generación de reportes es mayor para los alumnos que en el esquema anterior de reportes. Anteriormente la responsabilidad de reportar era compartida con el técnico, ya que éste lo registraba manualmente en una libreta, y podía darse el caso de que el alumno alegara que reportó una falla en su equipo y que al técnico se le hubiera olvidado registrarla. Ahora el SIRLI permite delegar responsabilidades y delimitarlas de forma clara y concreta, aun en el caso de que el alumno no haya encontrado falla alguna en el transcurso de la clase es su obligación registrarlo en el reporte correspondiente.

Es necesario evaluar y valorar el Sistema de Acceso al SIRLI por parte de los Alumnos, que en la actualidad tienen como clave de acceso su NIP, esto es, su fecha de nacimiento con el formato *ddmmaaaa*, ya que un uso malintencionado por parte de algún alumno con la información de un número de cuenta y un NIP que no le corresponda puede generar información “basura” dentro del sistema, entorpeciendo la funcionalidad y falseando los datos generados por el SIRLI. Una medida empleada para contrarrestar esta posible actividad es la de establecer dentro del reglamento interno de los laboratorios sanciones claras y contundentes a los usuarios que falseen información o utilicen el SIRLI de manera indebida.

Conclusiones

1. El SIRLI es un sistema de código abierto, libre y eficiente, que responde a las necesidades específicas de la automatización de reportes por parte de los usuarios de los Laboratorios de Idiomas. Es un sistema robusto en su lógica y sencillo en su interfaz y usabilidad, que lo hace amigable y útil para el usuario final, otorgando información íntegra, completa y concreta para identificar y planear esquemas de mantenimiento de los sistemas computacionales que componen a los laboratorios multimedia.

2. La portabilidad y la característica de software libre hacen del SIRLI un sistema portable y rentable, requiriendo especificaciones muy sencillas para su instalación y operación en una red de computadoras, pudiendo ser ejecutado bajo distintos entornos y sistemas operativos, siendo necesario para los equipos clientes disponer de un explorador web para poder utilizarlo.

3. El código fuente en el que ha sido desarrollado, basado principalmente en PHP, MySQL y HTML, permite la escalabilidad directa para implementar mejoras de funcionamiento y lógica de programación. El uso de hojas de estilo en cascada (CSS) da la oportunidad de configurar, de manera sencilla y de acuerdo a las políticas institucionales, su interfaz gráfica de forma eficaz y sencilla, permitiendo la integración total del sistema a los servicios web que la institución ofrece.

4. El análisis del estado del arte antes de la implementación del SIRLI y su comparación con el esquema actual muestran un mejoramiento de la calidad del servicio prestado en los laboratorios, mostrando resultados directos en la velocidad de respuesta de los técnicos en los problemas que se generan en los equipos, en la planeación de los mantenimientos, y en la velocidad y sencillez de la generación de reportes por parte de alumnos y profesores.

5. Los beneficios obtenidos en la realización del SIRLI han sido la recopilación de información de los usuarios de los laboratorios que evalúan la calidad de los servicios prestados antes de la implementación del sistema.

6. Otro aspecto de relevancia es la revisión y actualización de las políticas de servicio y uso en los laboratorios, donde hasta ese momento no se contaba con medidas de delegación de responsabilidades por un mal uso de estos espacios académicos.

7. El SIRLI, al ser un sistema de código abierto, con una codificación modular, estructurada y referencial, permite que sea un sistema sencillo en su actualización y escalamiento, siendo posible, gracias a sus características propias de la arquitectura Cliente/Servidor, su implementación para su uso a

distancia. Esto permite también que pueda ser utilizado en diversos espacios académicos en donde se necesiten gestionar las incidencias de sus equipos computacionales. Una proyección a futuro es la centralización de la información de los espacios computacionales de la UNAM, con el fin de controlar y gestionar de manera sistematizada y eficaz la información de estos espacios para analizar y elaborar planes de mantenimiento proactivo y reactivo para mantenerlos funcionando correctamente y que se puedan ofrecer servicios de calidad en beneficio de la comunidad escolar.

8. El manejo de contraseñas seguras para el acceso al SIRLI por parte de los Alumnos sigue una metodología de seguridad limitada, ya que un usuario con poca información puede causar serios problemas en la veracidad de la información manejada en el sistema. Una propuesta para mejorar la seguridad en este aspecto es la generación de claves seguras y automatizadas al inicio de cada ciclo escolar para cada alumno, donde éste tenga la opción de cambiar su contraseña generada por una que el mismo especifique, previa valoración de características seguras de contraseñas, para lograr un mayor control en el acceso.

9. Una característica de proyección a futuro consiste en programar y automatizar el análisis estadístico generado en el sistema, de manera que los datos arrojados en cada consulta generen además un análisis completo y funcional que permita proyectar el comportamiento del sistema en un momento dado, para implementar los planes de mantenimiento proactivo de manera directa, y considerando parámetros que el análisis estadístico y probabilístico brindarían para una mayor comprensión del estado de los laboratorios.

10. Otra característica considerable para una futura actualización es centralizar la información de cada laboratorio de Idiomas de cada plantel de la UNAM en un servidor dedicado para tal fin, con la idea de mantener un control de todos los laboratorios desde un mismo lugar, y aligerar la carga en los servidores locales, con el objetivo de que estos sean utilizados para un fin concreto, optimizando tiempos de análisis y lectura de la información, automatizando la entrega de reportes resumidos a cada plantel de los estados de sus laboratorios y tener la documentación del uso que se le da a cada laboratorio por parte de cada profesor y plantel, a fin de proponer nuevos esquemas de aprovechamiento académico en beneficio de la comunidad escolar.

11. Una consideración de implementación a futuro es la actualización de la interfaz gráfica por medio de frameworks php y javascript bajo el esquema de Software Libre, con el fin de mejorar la navegabilidad del SIRLI y automatizar el aprendizaje de operación del sistema desde el mismo.

Referencias

Abad Londoño, J. (2005, 06 de Abril). *Ingeniería de Software: tipos de pruebas de software.* Recuperado el 12 de agosto de 2013, de http://ing-sw.blogspot.mx/2005/04/tipos-de-pruebas-de-software.html

· AppServNetwork (2013). *AppServ Open Project.* Recuperado el 2 de febrero de 2013, de www.appservnetwork.com/index.php

· Arrúa, M. (2008, 26 de Abril). *Evitar SQL Injection en aplicaciones PHP | Martin Arrua.* Recuperado el 5 de septiembre de 2013, de http://www.martinarrua.com/evitar-sql-injection-en-aplicaciones-php/

· Briseño Díaz, M. (s. f.). *Mecanismos de seguridad informática.* Recuperado el 18 de agosto de 2013, de http://redyseguridad.fi-p.unam.mx/proyectos/buenaspracticas/mecanismos%20de%20seguridad.html

· Ecma International (2013). *The JSON Data Interchange Format.* Recuperado el 10 de septiembre de 2013, de http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

· Glazebrook, J. (s. f.). *Open Flash Chart – Home.* Recuperado el 2 de febrero de 2013, de http://teethgrinder.co.uk/open-flash-chart-2/

· Mañas, J. (1994, 16 de Marzo). *Prueba de Programas.* Recuperado el 17 de agosto de 2013, de http://www.lab.dit.upm.es/~lprg/material/apuntes/pruebas/testing.htm

· Manzor, S. (2013, 18 de Febrero). *Métricas de pruebas rendimiento de aplicaciones web.* Recuperado el 9 de septiembre de 2013, de http://es.agileload.com/agileload//blog/2013/02/18/web-applications-performance-testing-metrics

· Ochoa, J. (2011, 23 de Octubre). *Como Hacer un Menú Desplegable Multinivel Usando solo CSS | WebTursos.* Recuperado el 4 de marzo de 2013, de http://web.tursos.com/como-hacer-un-menu-desplegable-multinivel-usando-solo-css/

· Oracle Corporation (2010). *Distribución IPQoS (Guía de administración del sistema: servicios IP).* Recuperado el 10 de abril de 2013, de http://docs.oracle.com/cd/E19957-01/820-2981/ipqos-config-planning-12/index.html

- Peña Ayala, Alejandro. *Ingeniería de Software: Una Guía para Crear Sistemas de Información* [En línea]. México: Instituto Politécnico Nacional, 2006. [Consulta: 17 septiembre 2013]. Disponible en: http://www.wolnm.org/apa/articulos/Ingenieria_Software.pdf

- Pressman, Roger S. *Ingeniería del Software, un enfoque práctico.* Quinta Edición. España: McGrawhill, 1998.

- Sollutia (2011, 04 de Febrero). *Fase de pruebas en aplicaciones Web | Sollutia: Diseño web y soluciones de Internet.* Recuperado el 12 de septiembre de 2013, de http://blog.sollutia.com/2011/02/fase-de-pruebas-en-aplicaciones-web/

- Sommerville, Ian. *Ingeniería del Software.* Séptima Edición. Madrid: Pearson Educación, 2005

- Torossi, G. (s. f.). *Diseño Estructurado.* Recuperado el 9 de junio de 2013, de http://www.ecomchaco.com.ar/utn/disenodesistemas/apuntes/de/dise%C3%B1o_estructurado.htm

- Vásquez, J. & Rubio, D. (s. f.). *Modelo incremental o evolutivo - Programación Estructurada.* Recuperado el 1 de septiembre de 2013, de https://sites.google.com/site/programacion1electronica/metodologias-de-desarrollo-de-software/modelo-incremental-o-evolutivo

- Vendan, M. (2005, 11 de Agosto). *Guía del Desarrollo de Software - Fundamentos del Análisis de Requerimientos – Wikilearning.* Recuperado el 22 de febrero de 2013, de http://www.wikilearning.com/curso_gratis/guia_del_desarrollo_de_software-fundamentos_del_analisis_de_requerimientos/3471-2

Anexos

Anexo 1.

ENCUESTA DE SERVICIOS Y ATENCION EN LOS LABORATORIOS DE IDIOMAS
UNAM ENP PLANTEL 9 "PEDRO DE ALBA"

INSTRUCCIONES: LLENA EL SIGUIENTE CUESTIONARIO. TOMA EN CUENTA QUE LA INFORMACION QUE PROPORCIONES NOS AYUDARA A MEJORAR LA CALIDAD DEL SEVICIO PROPORCIONADO EN LOS LABORATORIOS.

GRUPO_____

1. ¿Cómo calificarias la calidad de los equipos de cómputo de los laboratorios de idiomas?

☐ EXCELENTE ☐ BUENO ☐ REGULAR ☐ MALO ☐ MUY MALO

2. ¿Generalmente cuál es la condición en la que encuentras el equipo de cómputo al momento de ingresar al laboratorio (COMPUTADORA, AUDIFONOS, MESA, MONITOR, ETC)?

☐ EXCELENTE ☐ BUENO ☐ REGULAR ☐ MALO ☐ MUY MALO

3. En qué momento revisas el estado del equipo que utilizas:

☐ AL INICIAR LA CLASE ☐ AL FINALIZAR LA CLASE ☐ MIENTRAS LO VOY UTILIZANDO

☐ NUNCA

4. Indica los tres problemas que más te han ocurrido en el laboratorio:

- ☐ No enciende el monitor.
- ☐ No funciona el teclado.
- ☐ No enciende el CPU.
- ☐ El mouse no reacciona.
- ☐ No se oye un lado de los audifonos.
- ☐ No se oyen los dos lados de los audifonos.
- ☐ Esta rayada alguna parte del equipo.
- ☐ Un software no funciona como deberia (TELL ME MORE, VIDEO DIGITAL, ETC)

Otro: _____

5. ¿Cuántas veces te ha ocurrido que el equipo falla de repente y ya no has podido reportarlo al Técnico del Laboratorio?

☐ MAS DE 5 VECES ☐ ENTRE 5 Y 2 VECES ☐ 1 VEZ ☐ NUNCA

Anexo 2.

ENCUESTA DE SERVICIOS Y ATENCION EN LOS LABORATORIOS DE IDIOMAS
UNAM ENP PLANTEL 9 "PEDRO DE ALBA"

INSTRUCCIONES: LLENE EL SIGUIENTE CUESTIONARIO. LA INFORMACION QUE PROPORCIONE NOS AYUDARA A MEJORAR LA CALIDAD DEL SEVICIO PROPORCIONADO EN LOS LABORATORIOS.

¿Cómo calificaría el tiempo de respuesta de los técnicos para resolver problemas en los laboratorios?
☐ INMEDIATO ☐ MUY RÁPIDO ☐ RÁPIDO ☐ NORMAL ☐ LENTO ☐ MUY LENTO

¿Cuánto tiempo le parece adecuado que los alumnos dediquen para revisar el estado de los equipos?

☐ Minutos.

¿Le gustaría poder llevar un registro electrónico (para alumnos y profesores) de las incidencias de clase en el cual usted pueda generar reportes en cualquier momento de la clase acerca del estado y la condición de los equipos del laboratorio? _____

Enliste los problemas que más ocurren en el laboratorio:

Anexo 3.

//ARCHIVO SECUENCIAL CON LAS RESPUESTAS DE LOS PROFESORES
//EN EL LABORATORIO B-107

"516"
"B"
"De Inmediato"
"3 minutos"
"No hay Internet"
"Está sucio"
""
"Si"

"420"
"B"
"De Inmediato"
"2 minutos o menos"
"Audífonos que solo se escuchan en un auricular."
"Micrófonos que no registran la voz."
"Computadoras que se pasman o congelan la actividad."
"Si"

"405"
"B"
"De Inmediato"
"2 minutos o menos"
"Algunos de los equipos contienen problemas en los audífonos (no funcionan o están en mal estado)"
"En alguna ocasión el programa se apaga solo. "
"Algunas veces el programa se traba o es difícil el acceso a éste. "
"Si"

"519"
"B"
"De Inmediato"
"2 minutos o menos"
"Falta de limpieza"
"Las diademas a veces no funcionan bien."
"No todos los grupos de quinto grado tienen acceso al laboratorio."
"Si"

"461"
"A"
"De Inmediato"
"2 minutos o menos"
"Se traban las máquinas"
"El monitor está desactivado"

"Los auriculares no funcionan"
"No"

"467"
"A"
"De Inmediato"
"3 minutos"
"1.-Los alumnos no pueden escuchar. Audífonos en mal estado. Alguien los rompió."
"2.-Los alumnos no pueden entrar porque la computadora se apagó. (No tan frecuente)"
"3.-El DVD está descompuesto o el visualizador. (No tan frecuente)"
"Si"

"511"
"A"
"De Inmediato"
"5 minutos"
"Se congelan las computadoras cuando envió video"
"Los audífonos no sirven"
"El micrófono no sirve"
"No"

"509"
"A"
"Una Clase"
"2 minutos o menos"
"Diademas en mal estado"
"El laboratorio está sucio"
""
"Si"

"507"
"B"
"De Inmediato"
"unos minutos"
"No hay Internet"
"no hacen la limpieza, está sucio el salón siempre"
""
"Si"

"516"
"B"
"Tres minutos"
"No funcionan las diademas"
"Micrófonos que no registran la voz."
"Computadoras que se traban cuando se trabaja en TELL ME MORE"
"Si"

"419"
"B"
"Rápido"
"2 minutos o menos"
"que cambien las diademas"
"En alguna ocasión el programa se apaga solo"
"Hay veces que nos es difícil trabar una máquina"
"Si"

"559"
"B"
"Inmediato"
"unos minutos"
"Hay que hacer la limpieza con más constancia"
"Los micrófonos no funcionan bien."
""
"Si"

//ARCHIVO SECUENCIAL CON LAS RESPUESTAS DE LOS PROFESORES
//EN EL LABORATORIO B-106

"414"
"A"
"De Inmediato"
"3 minutos"
"El mouse y los audífonos están fuera de lugar"
"La falta de limpieza del laboratorio"
"Los audífonos se descomponen con facilidad y muy seguido."
"No"

""
"A"
"De inmediato"
"2 minutos o menos"
"Audífonos en mal estado"
"Equipo fuera de lugar o monitores encendidos"
"Falta de limpieza del equipo"
"Si"

"513"
"A"
"Dos Clases"
"3 minutos"
"Fallas técnicas en las PCs"
"Laboratorio desaseado"
""
"No"

"514"
"B"
"De Inmediato"
"2 minutos o menos"
"El sistema saca a los alumnos del programa en el que están."
"Los alumnos no pueden escuchar las instrucciones desde el servidor y tampoco pueden escuchar bien mediante las diademas por desgaste y mal funcionamiento."
"En ocasiones los micrófonos no reconocen las voces de los estudiantes para realizar los ejercicios en TELL ME MORE"
"Si"

"460"
"B"
"De Inmediato"
"4 minutos"
"Todo bien"
""
""
"No"

Anexo 4.

Preguntas para la Coordinación de los Laboratorios de Idiomas.

1. ¿Cuántos académicos imparten clase en los laboratorios?

2. ¿Cuántos alumnos utilizan el laboratorio?

3. Describa por pasos la metodología en la cual los profesores elaboran referencias con respecto a fallos técnicos de los equipos de cómputo utilizados en los laboratorios.

4. ¿Qué tipo de información debería generar el sistema? (Ejemplo: Graficas, Estadísticas, Índice de incidencias, etc.) Enliste.

5. Elabore una lista de perfiles de usuarios que utilizarían el sistema (profesores, alumnos, técnicos, etc.)

6. ¿Cuál cree que sea el tipo de software en el cual se deba desarrollar el sistema?
   (Software Libre o Software Comercial)

Preguntas para Técnicos de los Laboratorios de Idiomas

1. Enliste los problemas que son reportados con mayor frecuencia por los alumnos:

2. Enliste los problemas que son reportados con mayor frecuencia por los profesores:

3. ¿Cree necesaria la implementación de un sistema de software que gestione los reportes de alumnos y de profesores? ___ ¿Por qué?

4. ¿Qué tipo de información debe arrojar el sistema? (Ejemplo: Graficas, Estadísticas, Índice de incidencias, etc.)Enliste.

5. ¿Qué otras características se deberían incluir en el sistema?

Anexo 5.

INSTALACIÓN

1. Copiar en la carpeta de acceso al servidor php el archivo SIRLI.zip y descomprimirlo. Esto creara la Carpeta "SIRLI" con los archivos php del sistema.

2. Descargar el paquete AppServ 2.5.10 desde la página oficial: http://www.appservnetwork.com.

3. Instalar Appserve en el equipo Servidor.

4. Ingresar, con algún explorador Web, al Servidor Local: localhost

5. Ingresar en el enlace "phpMyAdmin Database Manager Version 2.9.2" y acceder a phpMyAdmin como Súper usuario.

6. Seleccionar "Importar", dentro del apartado "Archivo a importar" seleccionar el botón "Examinar..." y ubicar el archivo "SIRLI.sql" que se encuentra dentro de la carpeta SIRLI previamente descomprimida.

7. Verificar que se haya importado correctamente la base de datos "SIRLIBD"

8. Ingresar desde un explorador web a: "nombre_equipo_servidor"/SIRLI/COORDINADOR/index_coordinador.php".

9. Si puede ver la pantalla de inicio de sesión del SIRLI y acceder con el nombre de usuario y la contraseña adecuada, el SIRLI se ha instalado correctamente.

DESINSTALACIÓN

1. Ingresar, con algún explorador Web, al Servidor Local: localhost

2. Ingresar en el enlace "phpMyAdmin Database Manager Version 2.9.2" y acceder a phpMyAdmin como Súper usuario.

3. En el apartado "Base de datos" seleccionar "SIRLIBD".

4. Seleccionar el apartado "Eliminar" y Confirmar la acción.

5. Con el explorador de archivos, buscar en la ubicación de archivos del servidor la carpeta "SIRLI". Eliminar la carpeta.

6. El SIRLI se ha desinstalado del equipo.