UNIVERSIDAD NACIONAL DE CAJAMARCA
FACULTAD DE INGENIERÍA
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS

“IMPACTO DE LA IMPLEMENTACIÓN DEL MÓDULO DE CENTRO QUIRÚRGICO EN LA PROGRAMACIÓN DE SALA DE OPERACIONES EN EL INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA, LIMA”

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

AUTOR:

Bach. Ronald Enrique Chicoma Roca

ASESOR:

Dr. Ing. Manuel Enrique Malpica Rodríguez

CAJAMARCA – PERÚ

2025

CONSTANCIA DE INFORME DE ORIGINALIDAD

- FACULTAD DE INGENIERÍA –

1. **Investigador: Ronald Enrique Chicoma Roca**
   DNI: 42729088
   Escuela Profesional: Ingeniería de Sistemas

2. **Asesor:** Manuel Enrique Malpica Rodríguez
   Facultad: Ingeniería

3. **Grado académico o título profesional**

   ☐Bachiller ■Título profesional ☐Segunda especialidad

   ☐Maestro ☐Doctor

4. **Tipo de Investigación:**

   ■ Tesis ☐ Trabajo de investigación ☐ Trabajo de suficiencia profesional

   ☐ Trabajo académico

5. **Título de Trabajo de Investigación:**

   IMPACTO DE LA IMPLEMENTACIÓN DEL MÓDULO DE CENTRO QUIRÚRGICO EN LA PROGRAMACIÓN DE SALA DE OPERACIONES EN EL INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA, LIMA

6. **Fecha de evaluación:** 11/08/2025
7. **Software antiplagio:** ■ **TURNITIN** ☐ **URKUND (OURIGINAL)** ***(*)***
8. **Porcentaje de Informe de Similitud:** 12%
9. **Código Documento:** 3117:481564006
10. **Resultado de la Evaluación de Similitud:**

    ■ **APROBADO** ☐ **PARA LEVANTAMIENTO DE OBSERVACIONES O DESAPROBADO**

Fecha Emisión: Cajamarca 12 de agosto del 2025.

Firmado digitalmente por: BAZAN DIAZ Laura Sofia FAU 20148258601 soft
Motivo: En señal de conformidad
Fecha: 12/08/2025 14:00:48-0500

FIRMA DEL ASESOR	***UNIDAD DE INVESTIGACIÓN FI***
Manuel Enrique Malpica Rodríguez	
DNI: 26707158	

ACTA DE SUSTENTACIÓN PÚBLICA DE TESIS.

TITULO : *IMPACTO DE LA IMPLEMENTACIÓN DEL MÓDULO DE CENTRO QUIRÚRGICO EN LA PROGRAMACIÓN DE SALA DE OPERACIONES EN EL INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA, LIMA.*

ASESOR : *Dr. Ing. Manuel Enrique Malpica Rodríguez.*

En la ciudad de Cajamarca, dando cumplimiento a lo dispuesto por el Oficio Múltiple Nº 0492-2025-PUB-SA-FI-UNC, de fecha 15 de agosto de 2025, de la Secretaría Académica de la Facultad de Ingeniería, a los ***veinticinco días del mes de agosto de 2025***, siendo las dieciséis horas (04:00 p.m.) en la Sala de Audiovisuales (Edificio 4K) de la Escuela Profesional de Ingeniería de Sistemas, de la Facultad de Ingeniería se reunieron los Señores Miembros del Jurado Evaluador:

Presidenta : Dra. Ing. Sandra Cecilia Rodríguez Avila.
Vocal : Dr. Ing. Jaime Amador Meza Huamán.
Secretario : Dr. Ing. Roger Manuel Sánchez Chávez.

Para proceder a escuchar y evaluar la sustentación pública de la tesis titulada: *IMPACTO DE LA IMPLEMENTACIÓN DEL MÓDULO DE CENTRO QUIRÚRGICO EN LA PROGRAMACIÓN DE SALA DE OPERACIONES EN EL INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA, LIMA*, presentado por el Bachiller en Ingeniería de Sistemas *RONALD ENRIQUE CHICOMA ROCA*, asesorado por el Dr. Ing. Manuel Enrique Malpica Rodríguez, para la obtención del Título Profesional

Los Señores Miembros del Jurado replicaron al sustentante debatieron entre sí en forma libre y reservada y lo evaluaron de la siguiente manera:

EVALUACIÓN PRIVADA : ...07.00... PTS.
EVALUACIÓN PÚBLICA : ...10.00... PTS.
EVALUACIÓN FINAL : ...17.00... PTS .....DIECISIETE..... (En letras)

En consecuencia, se lo declara .....APROBADO..... con el calificativo de .....DIECISIETE.....
acto seguido, el presidente del jurado hizo saber el resultado de la sustentación, levantándose la presente a las .....17:45..... horas del mismo día, con lo cual se dio por terminado el acto, para constancia se firmó por quintuplicado.

Dra. Ing. Sandra Cecilia Rodríguez Avila.
Presidenta

Dr. Ing. Jaime Amador Meza Huamán.
Vocal

Dr. Ing. Roger Manuel Sánchez Chávez.
Secretario

Dr. Ing. Manuel Enrique Malpica Rodríguez.
Asesor

AGRADECIMIENTO

En primer lugar, agradezco a Dios, por concederme salud y fortaleza para alcanzar mis metas. Asimismo, expreso mi sincero agradecimiento a los docentes de la Universidad Nacional de Cajamarca, y en particular a los de la Escuela Académico Profesional de Ingeniería de Sistemas, por su valiosa contribución a mi formación profesional. Mi gratitud también al Ingeniero Manuel Malpica, cuyo apoyo permanente fue fundamental para la realización de este trabajo. Del mismo modo, agradezco al personal del hospital, especialmente a quienes brindaron retroalimentación para mejorar la propuesta, y a mis compañeros de la institución, por sus ánimos y motivación durante el desarrollo de esta investigación.

DEDICATORIA

Dedico esta tesis a mis hijos, Andrew, Sofía y Hannah, cuya presencia me infunde la fuerza diaria para seguir creciendo profesionalmente. Asimismo, la dedico a mi madre, Herlinda, y a mis hermanas, Margarita y Rosa, por su constante apoyo y ánimo, fundamentales para la culminación de mis estudios.

CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La presente investigación tuvo como objetivo determinar el impacto de la implementación de un módulo de Centro Quirúrgico en la programación de salas de operaciones de un hospital pediátrico de alta complejidad. El problema identificado fue la gestión manual y fragmentada de las solicitudes quirúrgicas, lo que ocasionaba demoras, errores administrativos y baja satisfacción entre el personal involucrado. Para dar solución a esta problemática, se desarrolló e implementó un módulo quirúrgico integrado al sistema SISGALENPLUS, utilizando la metodología de diseño en cascada y tecnologías como Visual Basic 6.0 y SQL Server. Se empleó un diseño preexperimental con medición pretest y postest, evaluando tres dimensiones clave: tiempo, satisfacción y calidad. Los resultados evidenciaron una reducción promedio del 51 % en los tiempos operativos, un incremento del 83.4 % en la satisfacción del personal médico y administrativo, y una mejora del 108 % en los indicadores de calidad del proceso quirúrgico. Estos hallazgos permiten concluir que la implementación del módulo tuvo un impacto positivo en la programación de salas de operaciones, optimizando el uso de recursos, fortaleciendo la trazabilidad y elevando la eficiencia en la atención quirúrgica. El estudio confirma que el uso de herramientas tecnológicas bien diseñadas constituye una estrategia efectiva para mejorar la gestión hospitalaria en contextos complejos.

Palabras Claves: proceso, metodología cascada, sistema hospitalario, programación de salas de operaciones.

ABSTRACT

The purpose of this research was to determine the impact of implementing a Surgical Center module on operating room scheduling at a highly complex pediatric hospital. The identified problem was the manual and fragmented management of surgical requests, which led to delays, administrative errors, and low satisfaction among the staff involved. To address this issue, a surgical module integrated into the SISGALENPLUS system was developed and implemented using the cascade design methodology and technologies such as Visual Basic 6.0 and SQL Server. A pre-experimental design with pretest and posttest measurements was used, evaluating three key dimensions: time, satisfaction, and quality. The results showed an average 51% reduction in operating times, an 83.4% increase in the satisfaction of medical and administrative staff, and a 108% improvement in quality indicators of the surgical process. These findings suggest that the implementation of the module had a positive impact on operating room scheduling, optimizing resource use, strengthening traceability, and increasing the efficiency of surgical care. The study confirms that the use of well-designed technological tools is an effective strategy for improving hospital management in complex settings.

Key Words: process, waterfall methodology, hospital system, operating room scheduling.

CAPÍTULO I. INTRODUCCIÓN

La cirugía viene siendo un componente esencial de la asistencia sanitaria en todo el mundo, donde se calcula que se realizan cada año 234 millones de operaciones de cirugía mayor, siendo la cirugía el único método que puede mitigar discapacidades y reducir el riesgo de muerte por afecciones comunes. En respuesta a esta preocupación la OMS formuló la seguridad de las prácticas quirúrgicas como el segundo Reto Mundial relacionado a la seguridad del paciente [1]. En Europa muestran la importancia que tiene una correcta gestión en un centro quirúrgico, donde el indicador de cancelación de cirugías programadas es un factor fundamental comparando las causas por motivos de cirujano, anestesiología, administrativa y paciente. La cancelación de cirugías programadas genera una pérdida de recursos de preparación pre operatoria, desperdiciando tiempo en quirófano y reduciendo la eficiencia del mismo [2]. Otro estudio realizado explica que el centro quirúrgico es un centro crítico financiero, muestra la mala planificación de programación de quirófanos debido a la aleatoriedad de tiempos de uso de quirófano, en no clasificar la programación de cirugías para pacientes ambulatorios y pacientes hospitalizados, desaprovechando el rendimiento óptimo del quirófano [3]. Estudio en España, Sevilla se encontró que hubieron más de 670 000 cirugías en lista de espera, con una realización de cirugía después de 100 días de su programación [4].

Dentro de América Latina, presentan problemas en la gestión oportuna de información, donde se cuenta con un conjunto de sistemas independientes que apoyan a cada proceso, pero al generar los indicadores globales se convierte en desafío para dicha consolidación [5]. Otro estudio muestra que la actividad quirúrgica, es fundamental, porque representa las dos terceras partes de los ingresos. La alta demanda debido a factores como esperanza de vida y una alta tasa de afiliaciones a sistemas de salud, representan una problemática en el sector salud [6]. Entre las cancelaciones de cirugías por causas administrativas como error de programación quirúrgica, falta de consentimiento informado, falta de trámites, autorizaciones de cirugía y la falta de camas, representan un 44.2% de las causas totales [7].

A nivel nacional en Arequipa, se señala que la principal causa de suspensión fue por recursos humanos con un 37.2%, mientras que el factor de tiempo quirúrgico excedido fue de 33.33%, siendo así, la pérdida de tiempo en la programación de quirófanos, sin mencionar que esto afecta al paciente [8]. En Chiclayo, en un estudio de tiempo de espera quirúrgica, señala, que, se encontraron 551 pacientes en lista de espera y solo el 24.7% se realizaron la cirugía, el 28% ya

no requirió de la cirugía, el 40.1% se realizó otra cirugía y el 6.9% tuvieron exámenes vencidos. Este resultado advierte, que la existencia de lista de espera quirúrgica representa un gran problema de los sistemas de salud que repercute en la satisfacción del usuario, evitando así futuras complicaciones de la enfermad y/o retraso en la recuperación del paciente [9]. De acuerdo al contexto Internacional y a nivel nacional, resaltan la necesidad de mejorar la eficiencia en los centros quirúrgicos, tanto en la gestión de recursos humanos como en la planificación y programación de cirugías, con el objetivo de brindar una atención de calidad, reducir los tiempos de espera y satisfacer las necesidades de los pacientes.

La institución donde se realizó esta investigación, no es ajena a la problemática descrita sobre la eficiencia en quirófano. La institución no contaría con un módulo de centro quirúrgico para la gestión de sala de operaciones, volviéndose un punto crítico por ser un centro hospitalario pediátrico quirúrgico de alta complejidad. Al no contar con dicho módulo, los médicos de especialidades realizan las solicitudes de sala de operaciones y otros formatos de forma manual apoyándose de plantillas en archivo Excel, luego de esto, las solicitudes se envían a la unidad de centro quirúrgico luego de ser aprobadas. En la unidad de centro quirúrgico y anestesiología, las actividades toman mucho tiempo y recursos de personal administrativo para generar la programación quirúrgica diaria, debido a que tiene que digitar todos los datos, calcular tiempos de uso de sala, desencadenando demora en todos los servicios involucrados como enfermería, farmacia, banco de sangre, entre otros. Al no contar con un módulo de centro quirúrgico integrado al sistema, no permite consultar la información quirúrgica del paciente en línea, la situación actual de listas de espera, información de las causas de cancelación de cirugía para toma de decisiones. Por lo tanto, revisando el contexto institucional se formuló la siguiente pregunta: ¿Cuál es el impacto de la implementación del módulo de Centro Quirúrgico en la programación de sala de operaciones en el Instituto Nacional de Salud Del Niño San Borja, Lima?, conduciendo a la siguiente hipótesis: la implementación del módulo de Centro Quirúrgico impacta en la mejora de la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima.

Esta investigación se justifica desde el punto de vista práctico ya que una vez terminada va a permitir automatizar el proceso de programación de sala de operaciones en centro quirúrgico para beneficiar al personal médico y administrativo de la institución, reduciendo el tiempo de creación de formatos de solicitudes al llenar automáticamente los datos del paciente, evitando errores de digitación, generando la programación diaria oportunamente. Desde el punto de vista

académico aportó el diseño y desarrollo de un módulo de centro quirúrgico para el sistema SisGalenPlus, como referente para la mejora de procesos de otros establecimientos de salud. Desde el punto social el módulo de centro quirúrgico, contribuyó en mejorar la satisfacción del paciente, al contar con un sistema de centro quirúrgico que garantice la seguridad, la calidad y la eficiencia en la atención, el cual repercute de manera positiva ayudando a la sociedad en el cuidado de la salud del paciente.

Como alcance de esta investigación se planteó desarrollar interfaces de administración de solicitudes de sala de operaciones programadas y de emergencia, de aprobaciones de solicitudes, programación de salas, y administración de reportes operatorios y suspensiones. El módulo fue desarrollado con Microsoft Visual Basic 6.0, el repositorio de datos usado fue SQL Server 2012 y para el modelado se usó UML (Unified Modeling Language) usando la metodología de desarrollo cascada. Para ello se planteó la investigación teniendo como objetivo principal: en determinar el impacto de la implementación del módulo de Centro Quirúrgico en la programación de sala de operaciones en el Instituto Nacional de Salud Del Niño San Borja, Lima y como objetivos específicos: conocer el impacto de la implementación del módulo de Centro Quirúrgico en la satisfacción del personal médico y administrativo del centro quirúrgico en el proceso de la programación de sala de operaciones, estudiar el impacto de la implementación del módulo de Centro Quirúrgico en el tiempo de las actividades que realizan para cumplir con el proceso de programación de salas de operaciones y determinar el impacto del módulo de Centro Quirúrgico en la calidad de las actividades del proceso de centro quirúrgico para la programación de sala de operaciones.

El presente informe está organizado por 5 capítulos: **Capítulo I**: Abarca diversos aspectos cruciales de la investigación, comenzando por la exposición del problema. Además, se formula la hipótesis que guiará el estudio, se presenta la justificación de la investigación y se delimitan los alcances y objetivos que se persiguen. **Capítulo II**: Se centra en el marco teórico que sustenta la investigación. Aquí se analizan los antecedentes relevantes del tema, se exponen las bases teóricas fundamentales y se definen los términos básicos relacionados en el proceso, como con el desarrollo de software. **Capítulo III**: Detalla los materiales y métodos empleados durante el desarrollo del proyecto de investigación. Se proporciona una explicación clara y coherente de cómo se llevó a cabo el tratamiento de los datos para obtener los resultados deseados. **Capítulo IV**: Se presentan de manera detallada los resultados obtenidos en la investigación. Se realiza un análisis exhaustivo y una discusión profunda sobre los hallazgos encontrados.

Capítulo V: Incluye las conclusiones derivadas de los objetivos planteados en la investigación. Además, se ofrecen recomendaciones, con el propósito de ampliar y enriquecer los conocimientos sobre el tema de investigación.

CAPÍTULO II. MARCO TEÓRICO

2.1 Antecedentes teóricos de la investigación

En este apartado se presentan las investigaciones relacionadas con las variables en estudio, a nivel internacional, nacional y local.

2.1.1 A nivel internacional

Desta, et al [2], en su estudio, plantearon como objetivo evaluar las causas de cancelación de cirugías programadas en el día previsto, utilizando estadística descriptiva analítica. Como resultado, se identificó que, de un total de 462 pacientes programados, el 31,6 % de las cirugías fueron canceladas. Del total de procedimientos cancelados, el 20,5 % se debió a una inadecuada programación, mientras que el 8,9 % obedeció a la falta de disponibilidad de cirujanos. El estudio concluye que es necesario reducir la tasa de cancelaciones quirúrgicas, ya que la mayoría de estas pudieron haberse evitado. Para ello, se recomienda una adecuada programación, una comunicación efectiva entre el equipo multidisciplinario quirúrgico, una gestión eficiente de la disponibilidad de cirujanos y la mejora en la utilización de los quirófanos. Este estudio aporta a la presente investigación al permitir identificar las variables asociadas a la cancelación de cirugías, las cuales pueden ser integradas al modelo propuesto para la implementación del módulo correspondiente.

En Bélgica, Wang, et al [3], en su artículo, propusieron como objetivo incorporar una nueva clasificación para evaluar la eficiencia del centro quirúrgico. Para ello, se realizó un análisis en el que intervinieron variables relacionadas con el uso del quirófano según dos tipos de procedencia de los pacientes: ambulatorios y hospitalizados. Como resultado, se evidenció un aumento significativo en el número de cirugías ambulatorias, pasando de 20 procedimientos diarios en 1990 a más de 200 por día en el año 2019. Este incremento permitió observar diferencias en el uso del quirófano entre ambos tipos de pacientes. El estudio señala que los procedimientos ambulatorios, por su naturaleza, presentan menores tasas de infecciones adquiridas y una menor tasa de cancelación. En consecuencia, se concluyó que la cirugía ambulatoria es más rentable en comparación con la cirugía para pacientes hospitalizados, dado que no requiere internamiento o, en su caso, este es de corta duración. Este antecedente aporta a la presente investigación al considerar la procedencia del paciente como un criterio relevante para la programación de salas quirúrgicas, sugiriendo su clasificación en cirugías ambulatorias y cirugías para pacientes hospitalizados.

En Alabama EE UU, Ahmed y Ali [10], en su investigación, tuvieron como objetivo demostrar la importancia de considerar la preferencia del paciente en la selección del cirujano, relacionándola con el nivel de satisfacción del mismo. Los resultados indicaron que el 70 % de los pacientes son asignados a cirujanos con una alta tasa de preferencia, mientras que menos del 5 % son asignados a cirujanos con baja tasa de preferencia. Esta preferencia se basa en diversos factores, tales como la experiencia profesional, el sexo, el origen, la etnia y la reputación del cirujano. En consecuencia, los autores concluyen que establecer la preferencia del paciente como una variable en la asignación del cirujano contribuye al aumento de la satisfacción del paciente y a la disminución de la tasa de reingreso a cirugía dentro de los 30 días posteriores al procedimiento. Este antecedente aporta a la presente investigación la inclusión de la variable "preferencia del paciente", como un factor determinante para alcanzar una alta tasa de satisfacción en los procesos quirúrgicos.

En Cuba Cantera, et al [11], en su artículo, propusieron como objetivo implementar un módulo web que gestione el proceso quirúrgico. Para ello, se utilizó la metodología de desarrollo Programación Extrema (Extreme Programming, XP), realizando un análisis detallado de todos los formatos utilizados en dicho proceso. Como resultado principal, se desarrollaron dos componentes clave: el anuncio operatorio (documento previo a la cirugía, equivalente a una solicitud quirúrgica) y el informe operatorio (documento posterior a la cirugía, equivalente a un reporte operatorio), modelando la lógica necesaria para generar un informe completo y representarlo en un modelo físico. Los autores concluyen que el desarrollo de este sistema proporciona una arquitectura escalable, ofrece información tangible sobre el proceso quirúrgico y permite utilizar dicha información con fines investigativos. Este antecedente aporta a la presente investigación al describir el proceso de gestión documental en cirugía, desarrollado mediante una metodología de programación ágil y aplicado en un módulo informático.

2.1.2 A nivel nacional

En Lima, Villar [12], En Lima, Villar [12], en su tesis desarrolló el objetivo de implementar un sistema web para automatizar los procesos de las intervenciones quirúrgicas, enfatiza la mejora de registro, verificación y seguimiento de las programaciones quirúrgicas, aplica el método de observación y estudio cuantitativo. Da como resultados el tiempo que demora en registrar al paciente en la intervención quirúrgica con un promedio de 6.23 minutos sin sistema y 2.35 minutos con la implementación del sistema; para la verificación de la programación quirúrgica un promedio de 27.10 minutos sin sistema y 2.71 minutos con sistema; en cuanto al seguimiento

de información quirúrgica del paciente con un promedio de 34.36 minutos sin sistema y 3.09 minutos usando el sistema. Con esto concluye que se logró disminuir los tiempos de registro de pacientes en las intervenciones quirúrgicas logrando un 63.04% de eficiencia, los tiempos de verificación de datos quirúrgicos del paciente con una mejora de 24.39 minutos y para los tiempos de seguimiento de intervenciones de paciente con un 91.01% de eficiencia. Este antecedente aporta a la investigación, ya que proporciona el modelo de planificación, recolección y comparación de resultados antes y después de la implementación de un sistema o módulo de centro quirúrgico.

En Lima, Arias [13] en su tesis, tuvo como objetivo determinar cuál es el factor que genera un mayor tiempo de espera en una intervención quirúrgica, clasificando dicho tiempo en una escala que contempla las categorías: esperado, retardo moderado, retardo y retardo excesivo. Como resultado, se observó que, durante el periodo de enero y febrero de 2017, los factores administrativos generaron un 40,2 % de retardo excesivo y un 33,7 % de retardo, superando en porcentaje a los factores asistenciales en la lista de espera. Se concluyó que el tiempo promedio de espera por factores administrativos fue de 49,26 días, cifra superior al tiempo promedio asociado a factores asistenciales, que fue de 30,75 días. Este antecedente aporta a la presente investigación al ofrecer un enfoque cuantitativo para medir el tiempo de espera en función de factores administrativos y asistenciales, con miras a mejorar la eficiencia en la gestión de los procesos del centro quirúrgico.

Finalmente, otro estudio realizado en Lima por Anchante [14], planteó como objetivo identificar las razones que llevan al cirujano a suspender cirugías, evaluando la relación entre factores médicos y administrativos. Para ello, se utilizó una metodología observacional, retrospectiva y longitudinal. Como resultado, se registró que en el año 2018 se realizaron 2595 cirugías, de las cuales 232 fueron suspendidas; en el año 2017, el porcentaje de operaciones suspendidas fue del 8,56 % del total de procedimientos. Se concluyó que, en lo referente a los factores administrativos, las causas más frecuentes de suspensión fueron el exceso de programación y la falta de materiales quirúrgicos. Este antecedente aporta a la presente investigación al proponer un enfoque para relacionar los factores administrativos y médicos con la suspensión de cirugías, lo que permite una identificación adecuada de variables clave para la implementación del módulo de gestión del centro quirúrgico.

2.2 Bases Teóricas (Marco Teórico)

2.2.1 Sistema de información

Un sistema de información (SI), se entiende como un conjunto de componentes que se utilizan para procesar datos con el fin de producir información. Los datos que se ingresan en el sistema se procesan y se convierten en información que se presenta como salida. Estos sistemas pueden ser manuales o computarizados, dependiendo si hay o no intervención de computadoras en el proceso. Los componentes de los sistemas de información incluyen personas, datos, soportes de datos, máquinas, procedimientos, programas, controles, formularios, reglas, entre otros [15].

De manera más específica, un sistema de información de salud (HIMS) se define como un sistema de información creado para administrar información médica de pacientes, la gestión hospitalaria y apoyo a toma de decisiones en salud. Su enfoque también involucra datos vinculados a proveedores y organizaciones médicas, trabajando en conjunto para mejorar la atención del paciente e impulsar investigaciones, dado que estos sistemas manejan datos sensibles y vastos, la seguridad es una preocupación primordial, como se aprecia en la Fig. 1 [16].

Dentro de un HIMS se define cuatro grandes sub sistemas como sistema de registro médico (historia clínica del paciente), sistemas departamentales (módulo de laboratorio, imágenes, farmacia, centro quirúrgico y otros especializados como cardiología, neumología, neurología), sistema financiero, y sistemas fundamentales.

Esta investigación de todo un sistema de información en salud HISM se enfoca en el módulo de centro quirúrgico que representa a ***Surgery/Endoscopy***, dentro de los sistemas departamentales como se aprecia en la Fig. 1.

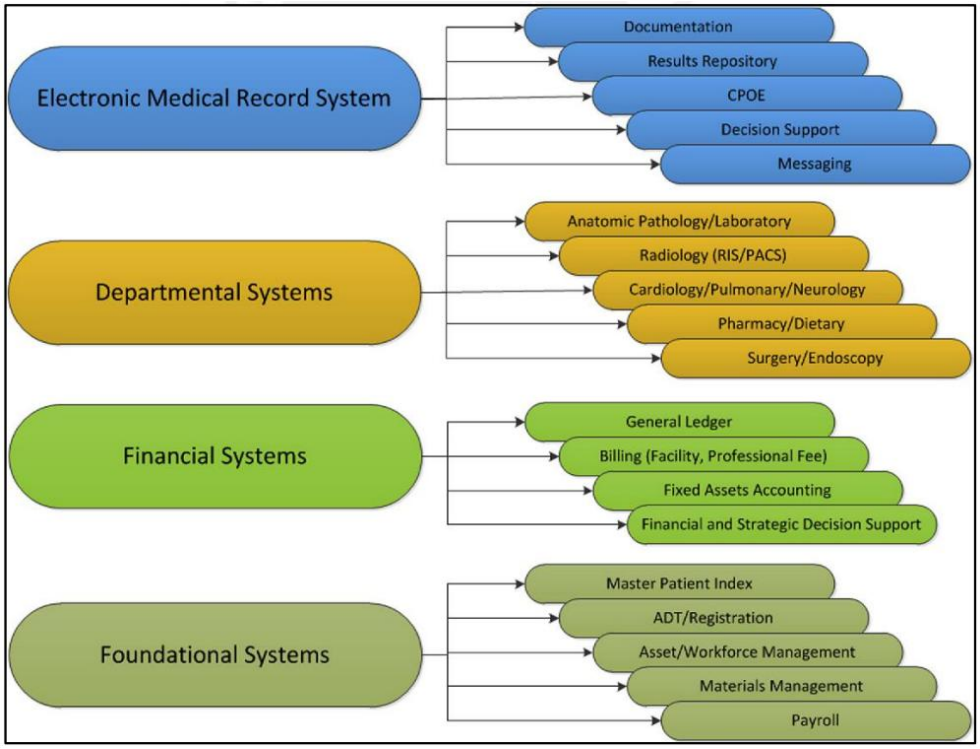


Fig. 1: Módulos de HIMS

Dado el contexto anterior se entiende como módulo de sistema de información, que a su vez puede estar formado por otros pequeños sistemas, a los que se los conoce como módulos, haciendo referencia a la modularidad definida como la capacidad de un sistema de ser dividido en varias partes que trabajan juntas para lograr un objetivo común, donde cada parte se denomina módulo y opera independientemente del resto de los componentes del sistema [17].

De manera más especializada se tiene al módulo de centro quirúrgico que se entiende como un subsistema de un sistema de información mucho más grande llamado: sistema de información en salud (HIMS), el cual se encarga de la gestión asistencial de un hospital o centro médico, integrando a los procesos de farmacia, consulta externa, hospitalización, emergencia, laboratorio, imágenes, facturación, tesorería entre otros. Por lo tanto, un módulo de centro quirúrgico se define a través de su capacidad de realizar la programación de cirugías de forma sencilla y eficiente, permitiendo la asignación de recursos, personal y equipamiento necesario para cada cirugía [18]; así mismo, tener la capacidad de seguimiento en tiempo real de las cirugías, gestión de los resultados de las cirugías como reportes operatorios, incluyendo las suspensiones, complicaciones y reacciones adversas; sin dejar de mencionar la capacidad de integrarse con otros módulos como facturación, farmacia, entre otros; además debe tener la capacidad de cumplir con la seguridad y privacidad de la información del paciente, protegiendo los datos sensibles, gestionando permisos y accesos.

Ahora, para el desarrollo de un módulo, es necesario seguir una metodología de desarrollo de software. En la actualidad, las metodologías son consideradas como un requisito fundamental para cualquier proyecto de desarrollo de software. Son necesarias tanto para el desarrollo efectivo del software como para la documentación del proyecto y la rendición de cuentas de los resultados obtenidos. En la etapa inicial del proyecto, es fundamental identificar la metodología de desarrollo de software que se adapte mejor a las necesidades del proyecto para lograr los mejores resultados, de modo que una metodología de desarrollo de software es un conjunto de técnicas y métodos que permiten abordar de manera sistemática y abierta cada actividad del ciclo de vida; su objetivo es optimizar el proceso y el producto de software, proporcionando métodos para guiar la planificación, el desarrollo y definir qué hacer, cómo hacerlo y cuándo hacerlo, durante todo el proceso de desarrollo y mantenimiento del proyecto [19]. Por lo tanto, enfocándose en la metodología en cascada, caracterizada por un orden riguroso de las etapas del proceso, donde que las etapas comienzan después de que se haya completado la etapa anterior y se realiza una revisión al final de cada etapa para determinar si el proyecto está listo para avanzar a la siguiente fase. Este modelo fue el primero en su tipo y sirve como base para otros modelos de ciclo de vida. Su principal problema es que no se pueden esperar especificaciones iniciales completas ya que los usuarios pueden cambiar de opinión sobre las características del software [19].

Esta metodología se desarrolla en una serie de fases como se observa en la figura 2 [19]; estas fases implican actividades particulares en cada una de ellas, las mimas que se describen a continuación: **análisis del sistema**: se realiza un análisis del sistema en el que el módulo se integrará o interactuará, así mismo se consideran aspectos como la infraestructura, los componentes del sistema, las interfaces y la arquitectura en general; además, se analiza cómo el módulo se ajusta a los objetivos y necesidades generales de un sistema más grande; **análisis de requisitos**: se recopilan y documentan los requisitos del módulo de sistema, esto implica comprender las necesidades y expectativas de los usuarios y se define claramente qué funcionalidades y características debe tener dicho módulo; **diseño**: una vez que los requisitos se han definido, se pasa a la fase de diseño donde se crea un diseño detallado del módulo, incluyendo su arquitectura, estructura de base de datos y cómo interactuará con otros componentes del sistema; i**mplementación**: el diseño se lleva a cabo mediante la codificación, se escriben el código y los algoritmos necesarios para desarrollar el módulo de sistema, aquí es importante seguir las pautas del diseño y las mejores prácticas de codificación para garantizar la calidad y la coherencia y escalabilidad; **pruebas**: una vez que se ha completado la

implementación, se procede a las pruebas, consiste en verificar y validar el módulo en busca de errores y asegurarse de que funcione según lo previsto, las pruebas pueden ser de diferentes tipos, como pruebas unitarias, de integración y de sistema; **mantenimiento**: después del despliegue, se entra en la fase de mantenimiento, aquí se realizan correcciones de errores y actualizaciones según sea necesario, también se pueden realizar mejoras y optimizaciones en función del uso de los usuarios [19] como se muestra en la Fig. 2.

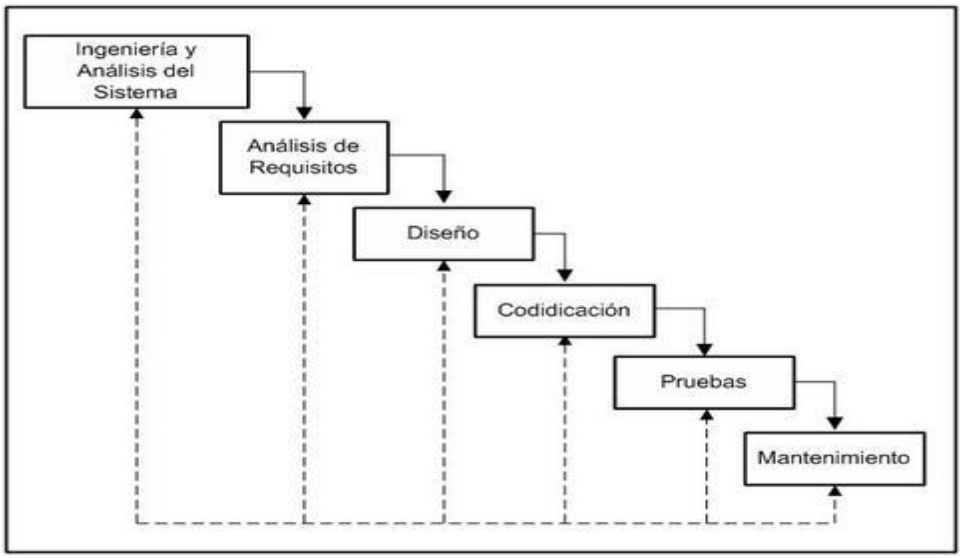


Fig. 2: Fases de metodología cascada

Para la fase de implementación en esta investigación se realiza el desarrollo del módulo en el sistema Integral SISGALENPLUS, éste es un sistema de gestión hospitalaria diseñado para mejorar la eficiencia en los procesos de hospitales, como consultas, hospitalización y facturación, su objetivo es facilitar la toma de decisiones gerenciales al registrar información clínica y administrativa precisa, también contribuye a reducir los tiempos de espera para los pacientes [20]. Este sistema fue desarrollado con el lenguaje VB6 para la interface de usuario y como repositorio de datos SQL server.

Este lenguaje fue creado por la empresa Microsoft, Visual Basic 6.0 (su IDE se aprecia en la figura 3 [21]) es un lenguaje de programación visual de cuarta generación, lo que significa que muchas tareas se pueden realizar sin necesidad de escribir código, utilizando operaciones gráficas con el ratón en la pantalla, es un lenguaje basado en objetos, aunque no es un lenguaje orientado a objetos en el sentido de lenguajes como C++ o Java, utiliza objetos con propiedades y métodos, pero carece de los conceptos de herencia y polimorfismo que son característicos de los lenguajes orientados a objetos como Java y C++ [21]; además, presenta un entorno intuitivo para el usuario como se observa en la Fig. 3.

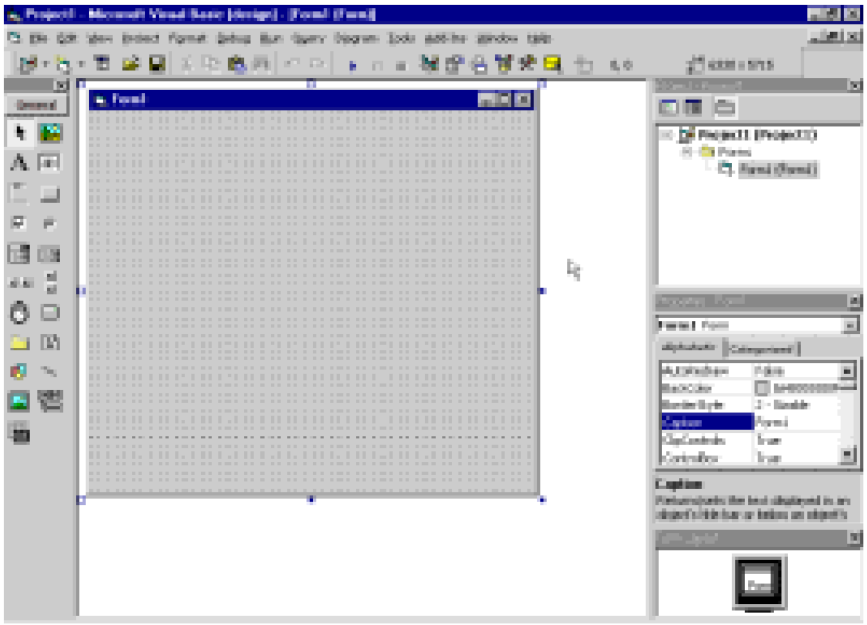


Fig. 3: Entorno Visual Basic 6.0

Para la programación en Visual Basic se cuenta con las características que se presentan en la Tabla I.

Tabla I: Características de Visual Basic 6.0

Características de Visual Basic 6.0	
Elementos	▪ Formularios: son ventanas de usuario que proporcionan una interfaz gráfica para que los usuarios interactúen con la aplicación, pueden contener controles, y responder a eventos, como clics de botón y entrada de teclado. ▪ Controles de usuario: son interfaces que contienen controles y lógica específica, pueden ser diseñarlos y personalizarlos, estos pueden encapsular funcionalidades y crear componentes que se pueden arrastrar y soltar en los formularios, lo que facilita la reutilización de código. ▪ Módulos: son archivos que contienen código y se utilizan para organizar y agrupar funciones, procedimientos y variables que se pueden utilizar en todo el proyecto. ▪ Módulos de clase: se utilizan para definir clases y objetos personalizados, cada módulo de clase puede contener propiedades, métodos y eventos personalizados que definen el comportamiento de los objetos basados en esa clase, se pueden crear múltiples instancias de objetos basados en la misma clase. ▪ Diseñadores: son componentes que se utilizaba para generar informes y documentos basados en datos, es una característica útil para crear informes simples dentro de la aplicación.
Tipo de datos	▪ **Integer:** se utiliza para números enteros con signo en el rango de -32,768 a 32,767. ▪ **Long:** es un tipo de datos entero de mayor tamaño que Integer, y se utiliza para números enteros mucho más amplio, desde -2,147,483,648 hasta 2,147,483,647. ▪ **Double:** se utilizan para números de punto flotante (números con decimales). ▪ **String:** se utiliza para texto y cadenas de caracteres. ▪ **Boolean:** se utiliza para representar valores lógicos, es decir, True o False. ▪ **Date:** se utiliza para fechas y horas.

Características de Visual Basic 6.0	
	▪ **Currency:** se utiliza para valores monetarios y proporciona precisión en los cálculos financieros. ▪ **Object:** se utiliza para hacer referencia a objetos COM y objetos personalizados en VB6. ▪ **Variant:** es un tipo de datos que puede contener cualquier tipo de dato, lo que lo hace muy versátil pero menos eficiente que los tipos de datos específicos. ▪ **Arrays:** admite matrices de varios tipos de datos, como enteros, cadenas, etc. Puede declarar matrices unidimensionales y multidimensionales. ▪ **Enumeraciones:** permiten definir un conjunto de valores constantes con nombres significativos.
Variables y constantes	▪ Variables locales se declaran dentro de un procedimiento o función y solo son accesibles dentro de ese procedimiento o función Dim ml_VariableEntero As Integer Dim ms_VariableString As String ▪ Variables globales se declaran fuera de cualquier procedimiento o función y son accesibles desde cualquier parte del módulo o formulario definido. Public ml_idUsuario As Integer Public ms_nombreUsuario As String ▪ Constantes se declaran con un valor especifico, se necesita la palabra Const y genera error cuando se intenta cambiar, también existen constantes propias del lenguaje ejemplo, vbNewLine Const MiConstante = 1000 Public Const MiConstanteString = "Modulo" Private Const MiConstanteEntero As Integer = 10 Const ConstanteString = "", ConstantePI As Double = 3.1416
Funciones y procedimientos	▪ Procedimientos es una sentencia de código independiente que, una que es invocado, ejecuta un número determinado de instrucciones, sin necesidad de devolver ningún valor. Sub nombreProcedimiento (parám1 as long)

<table>
<tr><th colspan="2">Características de Visual Basic 6.0</th></tr>
<tr><td></td><td>[sentencias]
[Exit Sub]
[sentencias]
End Sub
▪ Funciones puede ser utilizada en una expresión porque tiene un valor de retorno.
Function nombreFuncion (parám1 as long) As Long [sentencias]
[nombreFuncion = expresión]
Exit Function
[sentencias]
[nombreFuncion = expresión]
End Function</td></tr>
<tr><td>Sentencias de control</td><td>▪ Sentencia IF: sentencia condicional
If condicion1 Then
sentencias1
ElseIf condicion2 Then
sentencias2
Else
sentencia-n
End If
▪ Sentencia SELECT CASE: sentencia condicional
Select Case expresión
Case etiq1
[sentencias1]
Case etiq2
[sentencias2]
Case Else
sentencias
End Select
▪ Sentencia FOR ... NEXT: sentencia iterativa
For variable = expresion1 To expresion2 [Step expresion3]
[sentencias]</td></tr>
</table>

Características de Visual Basic 6.0	
	Exit For [sentencias] Next [variable] ▪ Sentencia DO ... LOOP: sentencia iterativa Do [{While/Until} condicion] [sentencias] [Exit Do] [sentencias] Loop
Operadores	▪ Operadores aritméticos Exponenciación: ^ Cambio de signo: - Multiplicación, división: *, / Resto de una división: Mod Suma y resta: +, - ▪ Concatenación Concatenar o enlazar: &, + ▪ Relacional Igual a: = Distinto: <> Menor que / menor o igual que: <, <= Mayor que / mayor o igual que: >, >= ▪ Otros Comparar dos expresiones de caracteres: Like Comparar dos referencias a objetos: Is ▪ Lógico Negación: Not Y: And O: Or

Siguiendo con el desarrollo en esta investigación se utiliza como Sistema Gestor de BD a SQL Server 2012, siendo el repositorio de datos para el sistema SISGALENPLUS.

SQL (Structured Query Language), es un sistema de gestión de bases de datos relacionales, es un lenguaje de consultas estructurados en la cual utiliza comandos los cuales son instrucciones para interactuar con la base de datos, estos comandos se pueden categorizar mediante cinco conceptos, las cuales son **DDL**: conjunto de comandos de definición de datos como por ejemplo CREATE, DROP, ALTER; **DML:** comandos para la manipulación de datos como por ejemplo INSERT, UPDATE, DELETE; **TCL:** comandos para controlar transacciones como lo son COMMIT, ROLLBACK, SAVEPOINT; **DCL:** comandos para dar y quitar permisos y roles como por ejemplo GRANT, REVOQUE; **DQL:** comando que se utilizan para la consulta de datos con SELECT el cual permite consultar registros, calcular datos, mesclar datos de varias tablas, agrupar datos entre otros [22] como se observa en la Fig. 4.

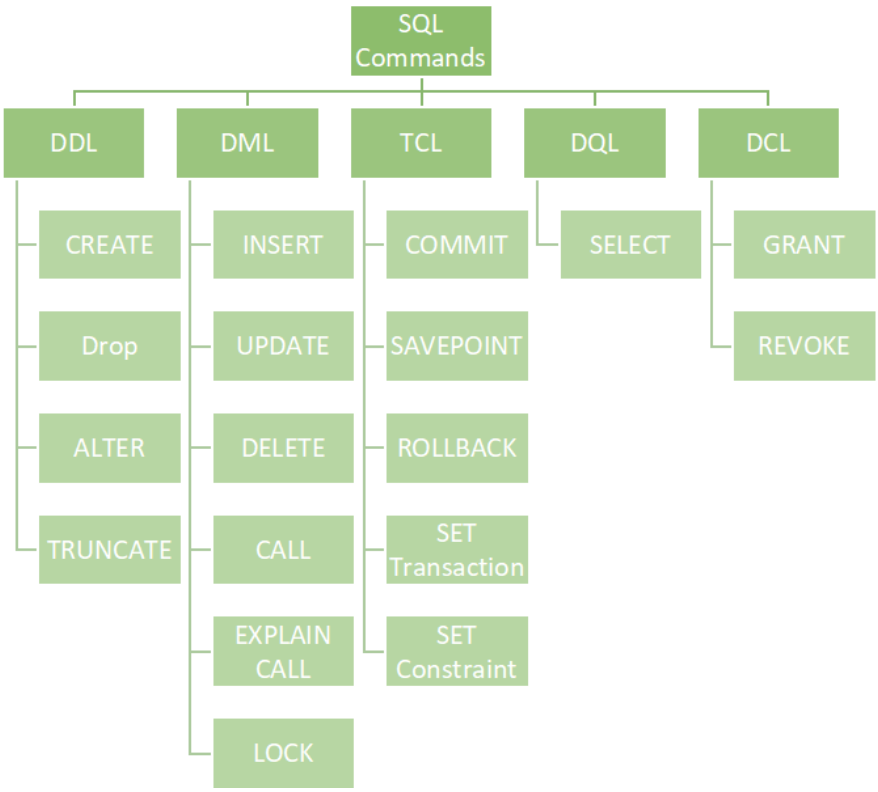


Fig. 4: Categorías de comandos en SQL

Para el desarrollo del módulo de centro quirúrgico se utilizan comandos, los más usados en esta fase, se explican en la Tabla II.

Tabla II: Características de SQL Server 2012

Características de SQL Server 2012	
Tipo de datos	Los tipos de datos en SQL se usa para almacenar de una forma óptima los datos de acuerdo a su naturaleza, podemos mencionar los más usados: ▪ **Bit**: para valores únicos 0 y 1, se usará para valores de ampliación de TRUE o FALSE. ▪ **Char**: para valores de cadena, se usa cuando la longitud del valor de dato es única. ▪ **DateTime:** para valores de fecha y hora. ▪ **Decimal, Numeric: s**e usa para valores con números decimales. ▪ **Int:** se usa para valores de números enteros. ▪ **VarChar:** para valores de cadena con longitud variable aproximada. ▪ **Date:** se usa para datos de fecha sin hora. ▪ **Varbinary:** se usa para guardar archivos, imágenes.
Tablas	Son objetos que organizan sus datos mediante filas y columnas, cada fila representa un registro y cada columna un campo con tipo de dato. Mediante SQL se pueden crear tablas y estas pueden ser de clasificadas como **heap** y **cluster** definida la última por utilizar índices [22]. Se puede mencionar tipo de tablas como tablas permanentes, tablas temporales, tablas como variable y otras como tablas particionadas. ▪ Tablas permanentes: se crean de forma permanente en la base de datos, los datos se conservan incluso al cerrar la sesión, a continuación, se describe la sintaxis para crear tabla con SQL: CREATE TABLE Nombre DeTabla([Identificador] [int] IDENTITY(1,1) NOT NULL, [CampoN] [varchar](250) NULL, CONSTRAINT [PK_NombreTabla]

```
        PRIMARY KEY CLUSTERED ([Identificador] ASC )
        WITH (
                PAD_INDEX = OFF,
                IGNORE_DUP_KEY = OFF,
                ALLOW_ROW_LOCKS = ON,
                ALLOW_PAGE_LOCKS = ON,
                FILLFACTOR = 100
                ) ON [PRIMARY]
    )
```

Donde:

- [Identificador] [int] IDENTITY(1,1) NOT NULL,

  Identificador: es el nombre del campo**, int**: Define el tipo de dato entero de la columna, **IDENTITY (1,1):** Indica que se generará automáticamente un valor único para cada fila cuando se inserte un nuevo registro, aquí los valores comienzan en 1 y se incrementan en 1 para cada nueva fila, **NOT NULL**: Indica que esta columna no puede contener valores nulos.

- [CampoN] [varchar](250) NULL,

  CampoN: define el nombre del enésimo campo**, varchar(250):** Define el tipo de datos de la columna como una cadena de longitud variable con una longitud máxima de 250 caracteres, **NULL**: Indica que esta columna puede contener valores nulos.

- CONSTRAINT [PK_NombreTabla]
  PRIMARY KEY CLUSTERED ([Identificador] ASC)

CONSTRAINT [PK_ NombreTabla]: Define la restricción de clave primaria **llamada** "PK_ NombreTabla", **PRIMARY KEY CLUSTERED([Identificador] ASC):** Indica que la columna [**Identificador**] es la clave primaria de la tabla. La cláusula **CLUSTERED** especifica que la clave primaria se almacenará de manera ordenada en el almacenamiento, mejorando el

rendimiento, **ASC**: Indica que la clave primaria se ordena en orden ascendente.

- WITH (

  PAD_INDEX = OFF,

  IGNORE_DUP_KEY = OFF,

  ALLOW_ROW_LOCKS = ON,

  ALLOW_PAGE_LOCKS = ON,

  FILLFACTOR = 100

) ON [PRIMARY]

Esta parte define las opciones de almacenamiento para el índice y estas opciones especifican cómo se deben almacenar los datos para mejorar el rendimiento y la eficiencia de almacenamiento.

- PAD_INDEX = OFF Indica si se debe realizar el relleno de índice cuando no hay suficientes valores para llenar una página de datos.
- IGNORE_DUP_KEY = OFF: Indica si se deben ignorar las claves duplicadas.
- ALLOW_ROW_LOCKS = ON: Indica si se deben permitir los bloqueos de fila.
- ALLOW_PAGE_LOCKS = ON: Indica si se deben permitir los bloqueos de página.
- FILLFACTOR = 100: Indica el nivel de llenado del índice.

- **Tablas temporales local,** se usa para almacenar datos de forma temporal en la ejecución de un conjunto de instrucciones, su alcance se encuentra en la sesión del usuario que lo crea. Su sintaxis es igual que las tablas permanentes, pero llevan en el nombre el símbolo: #

  CREATE TABLE #NombreDeTabla (

  Campo1 TipoDato,

  Campo2 TipoDato,

  --Otros campos y restricciones

```
);
```

- **Tablas temporales global,** se usa para almacenar datos de forma temporal, su alcance se encuentra en todas las sesiones del usuario, y estos datos persisten hasta la última conexión que hace referencia a este tipo de tabla. Su sintaxis es igual que las tablas permanentes, pero llevan en el nombre el símbolo: ##

```
CREATE TABLE ##NombreDeTabla (
    Campo1 TipoDato,
    Campo2 TipoDato,
    -- Otros campos y restricciones
);
```

- **Tablas como variable:** son variables de tipo Tabla se usa para almacenar datos temporales dentro de un conjunto de instrucciones por usuario, se usan para almacenar pequeñas cantidades de datos, su sintaxis es la siguiente:

```
DECLARE @NombreDeTabla TABLE (
    Campo1 TipoDato,
    Campo2 TipoDato
    -- Otros campos y restricciones
);
```

- **Tablas de expresión común:** también SQL nos proporciona las tablas CTEs, que son expresiones de tablas temporales para resultados complejos, donde se necesita una o más tablas temporales y estas tengan dependencias o relaciones entre ellas **Fuente especificada no válida.**, su sintaxis es la siguiente:

```
;WITH
 CTE1 AS (SELECT * FROM Tabla1 WHERE Condiciones)
,CTE2 AS (SELECT * FROM Tabla2 WHERE Condiciones)
SELECT * FROM CTE1, CTE2 WHERE Condiciones
```

Características de SQL Server 2012	

Procedimientos almacenados de usuario	Los procedimientos almacenados del usuario son programas que ejecutan un conjunto de instrucciones SQL, se caracterizan por no devolver un valor, a diferencia de las funciones, su sintaxis es la siguiente:
	CREATE PROCEDURE NombreProcedimiento
	@ParametroEntrada1 TipoDato,
	@ParametroEntrada2 TipoDato = ValorDefecto,
	@ParametroSalida TipoDato OUTPUT
	AS
	BEGIN
	-- Cuerpo del procedimiento almacenado
	[Sentencia_1]
	[Sentencia_2]
	[Sentencia_N]
	--Asignación de valor calculado a parámetro de salida
	SET @ParametroSalida = ValorParaParametroDeSalida;
	[Sentencias…]
	END;
	Su invocación:
	DECLARE @Param3 TipoDato;
	EXEC NombreProcedimiento ValorParam1, ValorParam2, @Param3 OUTPUT;
	EXEC NombreProcedimiento ValorParam1, @Param3 OUTPUT;
	Donde: se aprecian tres parámetros que tiene el procedimiento, el primer parámetro es de entrada definida solo por su tipo de dato, el segundo parámetro es también de entrada pero tiene un valor por defecto, significa que a la hora de invocar el procedimiento, este parámetro no es obligatorio, ya que si no se envía el valor, toma el valor por defecto, luego tenemos el tercer parámetro que viene a ser un parámetro de salida, significa que

<table>
<tr><th colspan="2">Características de SQL Server 2012</th></tr>
<tr><td></td><td>cuando el usuario invoque al procedimiento, SQL le devolverá el valor calculado mediante este parámetro, un ejemplo muy común es cuando el procedimiento realiza una inserción y este devuelve en este parámetro el id del registro insertado.</td></tr>
<tr><td>Funciones</td><td>

Las funciones en SQL son programas que ejecutan un conjunto de instrucciones y devuelven un valor o una tabla, se pueden clasificar como funciones escalares, funciones de tabla, funciones de Agregado.

CREATE FUNCTION NombreFuncion (@Parametro TipoDato)
RETURNS TipoDato
AS
BEGIN
[Instrucción 1]
[Instrucción 2]
[Instrucción N]
--retorno de la función de acuerdo al tipo de dato declarado:
RETURN ValorCalculado;
END;
Donde:

En funciones escalares NombreFuncion: es el nombre de la función, @Parametro con su tipo de dato, puede ser n parámetros, RETURNS TipoDato, aquí se declara que tipo de dato retornara la función, y la última sentencia RETURN ValorCalculado: es valor que retorna la función luego de ejecutar todas las sentencias.
Las funciones de tabla en RETURNS especifican la tabla.
Las funciones de Agregado son funciones nativas de SQL como SUM, COUNT, MAX, MIN, AVG

</td></tr>
</table>

Para el modelamiento del diseño de sistema en esta investigación se usa UML (Lenguaje de Modelado Unificado). UML es una herramienta estándar de modelado para diseñar el software antes de su implementación, en un lenguaje que permite representar de una manera grafica el sistema por el equipo de desarrollo y diferentes usuarios apoyándose de los diferentes diagramas que contiene, mediante la notación UML como se observa en la Fig. 5, se puede especificar las características y funcionalidades que tendrá el sistema, de modo que esto sirve como base para la construcción del sistema, y la documentación del mismo. Este lenguaje UML es un estándar respaldado por la OMG (Estandards Development Organization), dentro de su composición lo clasifica en 3 bloques de construcción como elementos, relaciones y diagramas [23].

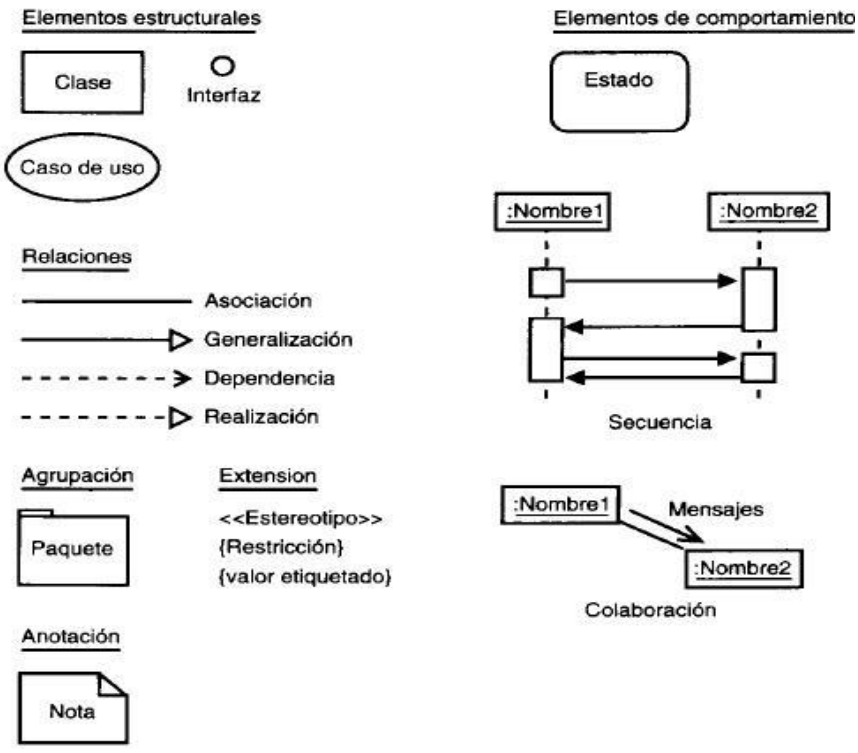


Fig. 5: Elementos UML

Dentro del bloque elementos, UML cuenta con elementos estructurales: estos elementos se utilizan para representar la estructura estática de un sistema y cómo sus componentes se relacionan entre sí, incluyen clases, objetos, interfaces, componentes. Por ejemplo, las clases representan las entidades en un sistema, los objetos son instancias específicas de esas clases, y las interfaces definen contratos que las clases deben seguir, elementos de comportamiento: estos elementos se utilizan para modelar el comportamiento dinámico de un sistema, es decir, cómo interactúan sus componentes en tiempo de ejecución, estos elementos incluyen casos de uso,

actividades, diagramas de secuencia y diagramas de estado. Por ejemplo, los casos de uso del negocio describen las interacciones entre el sistema y los actores externos, elementos de agrupación: estos elementos se utilizan para organizar y estructurar otros elementos en diagramas UML, incluyen paquetes y sub paquetes. Por ejemplo, un paquete puede contener clases relacionadas en un sistema [24].

Dentro del bloque relaciones incluyen agregación y composición: son aquellas que representan relaciones de "todo-parte" entre clases u objetos, la composición es una forma más fuerte de agregación y sugiere una relación más estrecha y duradera entre el todo y las partes; realización o implementación: indican que una clase realiza una interfaz, implementando los métodos definidos en esa interfaz; asociación de inclusión: usada en diagramas de casos de uso, muestra cómo un caso de uso incluye otro caso de uso en su flujo; asociación de extensión: también utilizada en diagramas de casos de uso, representa cómo un caso de uso puede extender otro caso de uso en condiciones específicas [24].

Dentro de este bloque se define como la representación gráfica de un sistema o parte de un sistema que utiliza símbolos y anotaciones específicas para describir la estructura, el comportamiento y las interacciones de los elementos que componen el sistema, se puede mencionar los diagramas de clases: la cual representa la estructura estática de un sistema, mostrando las clases, sus atributos, métodos y las relaciones, entre ellas están centrados en la modelización de las clases y sus relaciones para entender la estructura de un sistema, diagrama de casos de uso, los cuales describen las interacciones entre los actores externos y el sistema, identificando los casos de uso del sistema, se utilizan para definir los requisitos funcionales desde la perspectiva del usuario y cómo interactúa con el sistema, diagrama de estados, estos modelan el comportamiento de un objeto o sistema en términos de estados, transiciones y eventos que provocan cambios de estado, centrados en cómo los objetos cambian de estado en diferentes circunstancias, diagrama de componentes: los cuales muestran cómo los componentes del sistema están organizados y cómo interactúan [24].

Por otro lado, en ingeniería de software, la arquitectura cliente-servidor multicapa (o n-capas) es un patrón arquitectónico que separa físicamente o lógicamente la presentación, la lógica de negocio y la gestión de datos, permitiendo que cada capa se ejecute o se despliegue de forma independiente para mejorar la escalabilidad, mantenibilidad, seguridad y reutilización del sistema. Estas capas permiten, por ejemplo, que sólo la lógica de negocio interactúe directamente con la base de datos, mientras que la capa de presentación permanece aislada de

detalles internos, reduciendo acoplamiento y aumentando la robustez del sistema frente a cambios futuros y necesidades de integración o evolución [25].

Otro punto importante a destacar es el Control de Acceso Basado en Roles (RBAC, por sus siglas en inglés) es un modelo de seguridad que organiza el acceso a los recursos del sistema mediante la asignación de derechos y permisos a roles, a los cuales luego se vinculan los usuarios. De esta manera, un usuario adquiere permisos exclusivamente a través de su rol asignado, lo que simplifica significativamente la gestión de privilegios, especialmente en sistemas con múltiples perfiles y escenarios de uso compartido [26]. Además, el modelo RBAC favorece la jerarquía de roles, donde los roles de nivel superior heredan los permisos de los roles subordinados, proporcionando flexibilidad, escalabilidad y control del principio de mínima autoridad. El estándar unificado propuesto por NIST define variantes como RBAC básico, jerárquico y restringido, consolidando así un esquema estructurado para implementaciones seguras en entornos empresariales complejos, este enfoque es especialmente adecuado en sistemas hospitalarios modulares como el módulo de centro quirúrgico, donde distintos perfiles (médico solicitante, cirujano, anestesiólogo, administrativos, enfermería y dirección) requieren acceso diferenciado y controlado sobre funcionalidades específicas [27].

En el ámbito de la calidad de software, existen varios estándares ISO relevantes, para medir la calidad de un módulo de sistema; por lo que se puede mencionar: ISO/IEC 25010: La ISO/IEC 25010, está definida por ocho dimensiones de calidad de software como se muestra en la Fig. 6, aplicado a un módulo de salud, se deben considerar funcionalidad: se refiere a la capacidad del módulo de salud para realizar sus funciones correctamente. En un módulo de salud, esto implica que debe ser capaz de gestionar registros médicos, realizar diagnósticos, ofrecer recomendaciones de tratamiento y otras tareas relacionadas con la atención médica de manera precisa y eficaz; fiabilidad: se refiere a la capacidad del módulo para mantener un rendimiento consistente y confiable. En un contexto de salud, la fiabilidad es crítica, ya que los errores o malentendidos pueden tener graves consecuencias para la salud de los pacientes; usabilidad: se refiere a la facilidad con la que los usuarios, como médicos y pacientes, pueden interactuar con el módulo de salud. Debe ser intuitivo, fácil de aprender y usar, para que los profesionales médicos puedan aprovecharlo al máximo; eficiencia: se refiere a la capacidad del módulo para responder rápidamente a las solicitudes de los usuarios y procesar datos médicos de manera eficiente; seguridad: es de suma importancia para proteger los datos médicos sensibles y garantizar que solo las personas autorizadas tengan acceso a la información médica;

mantenibilidad: se relaciona con la capacidad de realizar cambios y mejoras en el módulo de manera eficiente y sin interrupciones en la atención médica. Los sistemas de salud evolucionan constantemente, por lo que es esencial que el módulo sea fácilmente adaptable; compatibilidad: el módulo debe ser compatible con otros sistemas y estándares de salud, lo que facilita la interoperabilidad y la integración con otros sistemas médicos; portabilidad: se refiere a los datos obtenidos del sistema deben ser intercambiables entre otros sistemas, así mismo su configuración debe personalizable desde cualquier punto de acceso [28].

Fig. 6: ISO ISO/IEC 25010

De las cuales se evalúa las siguientes sub características para medir el grado de satisfacción del producto completitud funcional: se refiere a la capacidad para proporcionar todas las funciones necesarias para abordar las necesidades médicas y administrativas, la completitud funcional implica que la aplicación debe incluir todas las funciones requeridas para el registro de pacientes, diagnóstico, tratamiento, gestión de historiales médicos, programación, facturación y cualquier otra función esencial para el flujo de trabajo de atención médica; corrección funcional: se relaciona con la capacidad del módulo para realizar sus funciones de manera precisa y sin errores. En un entorno médico, la corrección es fundamental, ya que incluso pequeños errores pueden tener graves consecuencias, debe garantizar que los cálculos y registros sean precisos y fiables; pertinencia funcional: se centra en la relevancia de las funciones del módulo para los usuarios y los procesos médicos, debe adaptarse a las necesidades específicas de los profesionales de la salud y los pacientes, evitando funciones innecesarias que puedan complicar la usabilidad; capacidad de entender fácilmente: se refiere a la claridad y la comprensión de la interfaz de usuario y la información presentada en el módulo, la cual debe ser fácil para los usuarios entender cómo utilizar el software y comprender la información que se muestra; capacidad de aprender fácilmente: se relaciona con la facilidad con la que los usuarios pueden aprender a utilizar el módulo, el cual debe ser intuitivo y requerir un tiempo

mínimo de capacitación para que los profesionales médicos puedan incorporarlo a sus flujos de trabajo sin dificultad; capacidad de operar fácilmente: se refiere a la facilidad con la que los usuarios pueden interactuar con el módulo de salud y llevar a cabo sus tareas de manera eficiente, debiendo minimizar la necesidad de pasos innecesarios o complicados, permitiendo a los usuarios realizar acciones como ingresar datos, consultar registros o generar informes de manera sencilla y rápida [28].

2.2.2 Programación de sala de operaciones

La programación de las salas de operaciones es un proceso clave en la gestión del centro quirúrgico, ya que garantiza que los procedimientos se realicen de manera eficiente y dentro de los plazos establecidos. Este proceso abarca desde la solicitud de la sala de operaciones hasta la realización de la cirugía en el paciente, permitiendo, además, mejorar el uso de los quirófanos al organizar los tiempos disponibles de manera efectiva. Es fundamental comprender los conceptos importantes que intervienen en este proceso, comenzando por el diagnóstico. El diagnóstico es el acto mediante el cual el profesional de la salud identifica una enfermedad a partir de los síntomas del paciente, apoyándose en diversas herramientas y exámenes auxiliares [29]. Según su clasificación, el diagnóstico puede ser:

- **Definitivo o final**: Establecido con datos clínicos precisos, respaldados por exámenes complementarios.
- **Presuntivo:** Evaluación provisional basada en los antecedentes médicos del paciente.
- **Repetitivo:** Relacionado con la atención continua de un diagnóstico ya identificado y registrado previamente [30].

El proceso comienza con la solicitud de sala de operaciones u Orden de Intervención Quirúrgica, que es un documento que describe los detalles del paciente, como nombres, apellidos, número de historia clínica, edad, sexo, servicio actual, el procedimiento quirúrgico previsto (incluyendo el tiempo estimado), datos del médico cirujano, tipo de anestesia, y plan de cuidados postoperatorios [30]. El procedimiento quirúrgico está incluido en el Catálogo de Procedimientos Médicos y Sanitarios del Sector Salud (CPMS), aprobado por resolución ministerial 1044-2020-MINSA. En cuanto a la anestesia y el riesgo anestesiológico, este se clasifica según los niveles de la "American Society of Anesthesiology", que van desde el nivel I (paciente sin alteraciones orgánicas) hasta el nivel V (paciente con pocas probabilidades de sobrevivir) [31]**.**

Además de la solicitud, el consentimiento informado es otro documento esencial, donde se detallan el procedimiento quirúrgico y los posibles riesgos y beneficios. El paciente debe firmarlo antes de la cirugía para confirmar que comprende y acepta el tratamiento [30]. Tras las aprobaciones correspondientes, la solicitud se transforma en la programación de la sala de operaciones, un listado de las cirugías programadas, que incluye detalles del paciente, quirófano asignado, hora de inicio, médico cirujano y anestesiólogo [30]. Esta programación se realiza con un día hábil de antelación y se comunica a las áreas implicadas.

El término cirugía proviene del latín "chirurgia" y, según la RAE, se refiere a una parte de la medicina que tiene por objetivo curar enfermedades mediante operaciones. Hoy en día, gracias a los avances tecnológicos, se emplean métodos como el láser o la radiación para realizar intervenciones menos invasivas, que incluso evitan el uso de suturas.

Las cirugías se pueden clasificar según su gravedad:

- **Cirugía mayor:** Implica sedación general y afecta órganos principales.
- **Cirugía menor:** No compromete órganos principales y puede realizarse con anestesia local, regional o general [32].

Clasificándose por la urgencia:

- **Cirugía de emergencia:** Debe realizarse inmediatamente debido a que la vida del paciente está en riesgo.
- **Cirugía de urgencia:** Se debe realizar en unas horas, aunque el paciente no está en peligro inmediato.
- **Cirugía programada o electiva:** Puede postergarse para realizar estudios previos que optimicen los resultados, ya que no representa un peligro inmediato para la vida del paciente.
- **Cirugía por reingreso:** Se refiere a una reintervención antes del alta del paciente [32].

La anestesia, que se clasifica en local, regional o general, es fundamental para garantizar la ausencia de dolor durante el procedimiento. La anestesia local se aplica en la zona del cuerpo donde se realizará la intervención; la anestesia regional, que afecta una parte mayor del cuerpo, como una extremidad; y la anestesia general, que sumerge al paciente en un estado de inconsciencia total [32]. El quirófano o sala de operaciones es el espacio donde se llevan a cabo los procedimientos quirúrgicos, éste debe estar diseñado y equipado para garantizar la seguridad

y eficiencia durante las intervenciones. Entre el equipo especializado se encuentran el cirujano principal, el cirujano ayudante, el anestesiólogo, el instrumentista y el circulante, cada uno desempeñando funciones críticas [30]. Al finalizar la cirugía, se elabora el reporte operatorio, que es un documento que detalla la duración de la intervención, los procedimientos realizados, los materiales utilizados y cualquier complicación o reacción adversa que haya ocurrido durante la operación. También se registra el nombre del cirujano y el equipo quirúrgico [33]. Finalmente, el rendimiento del quirófano se mide mediante indicadores clave como el rendimiento de la sala de operaciones. Se considera óptimo un rendimiento del 80% al 85% (incluyendo tiempos inactivos), mientras que un rendimiento entre el 65% y 70% (sin tiempos inactivos) es aceptable. Si el rendimiento es inferior al 60%, indica un uso ineficiente de los recursos, mientras que rendimientos superiores al 75% pueden generar sobrecarga de trabajo en el personal, lo que aumenta el riesgo de errores [34]. La Fig. 7, muestra la sala de operaciones con equipo multidisciplinario.

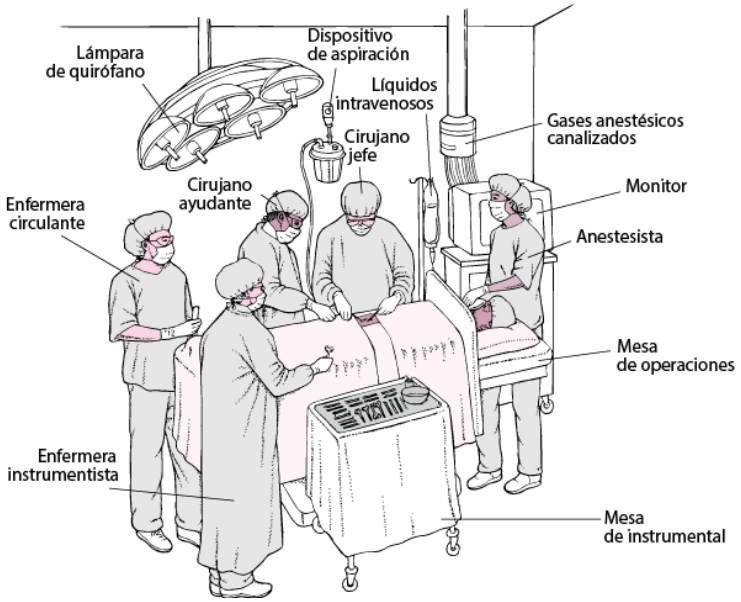


Fig. 7: Sala de operación con equipo multidisciplinario

2.3 Definición de términos básicos

- **ADO:** tecnología de acceso a datos en aplicaciones de Microsoft que permite interactuar con bases de datos [22].
- **Alcance**: límites y objetivos definidos para un proyecto o sistema, detallando qué incluirá y qué no [15].
- **Análisis de datos**: proceso de examinar, limpiar y modelar datos para obtener información útil y tomar decisiones [22].

- **Base de datos**: conjunto organizado de datos almacenados accesibles y gestionables mediante software [22].
- **Calidad de software**: grado en que un software cumple con los requisitos funcionales y no funcionales esperados [35].
- **CIE10**: clasificación internacional de enfermedades, utilizada para codificar diagnósticos médicos [36].
- **CPMS**: clasificador de procedimientos médicos en salud, usado para estandarizar procedimientos quirúrgicos [30].
- **Circulante**: personal que apoya en el quirófano sin participar directamente en la esterilidad del procedimiento [30].
- **Eficiencia**: relación entre los recursos utilizados y los resultados obtenidos en un sistema o proceso [28].
- **Establecimiento de salud**: institución encargada de brindar servicios de atención médica a la población [30].
- **Fiabilidad**: capacidad del sistema para funcionar correctamente bajo condiciones específicas durante un periodo determinado [28].
- **Historia clínica**: documento que registra la información médica y antecedentes de un paciente de forma ordenada y confidencial [36].
- **Integración**: proceso de combinar componentes o sistemas para que trabajen conjuntamente de manera coordinada [28].
- **ISO**: Organización Internacional de Normalización que desarrolla estándares para garantizar calidad, seguridad y eficiencia [28].
- **Instrumentista**: profesional que asiste al cirujano entregando los instrumentos necesarios durante una operación [30].
- **Interoperabilidad**: capacidad de un sistema para interactuar y compartir información con otros sistemas de manera eficiente [28].
- **Módulo de sistema**: componente independiente de un sistema que realiza funciones específicas dentro de un sistema mayor [16].
- **Nivel de establecimiento**: clasificación de un establecimiento de salud según la complejidad de los servicios que ofrece [30].
- **Proceso**: secuencia de actividades secuenciales y relacionadas que transforman insumos en resultados específicos [35].
- **Redundancia:** duplicación de componentes o procesos para garantizar la disponibilidad y confiabilidad del sistema [22].

- **Requerimiento**: necesidad o condición que un sistema debe cumplir para lograr sus objetivos [19].
- **Satisfacción del usuario:** nivel de cumplimiento de las expectativas del usuario respecto a un producto o servicio [28].
- **Seguridad informática**: medidas aplicadas para proteger los sistemas y datos contra accesos no autorizados o ciberataques [15].
- **Trazabilidad**: seguimiento del historial, ubicación y uso de datos o procesos dentro de un sistema [28].
- **Usabilidad**: facilidad con la que los usuarios pueden interactuar con un sistema para alcanzar sus objetivos [28].

CAPÍTULO III. MATERIALES Y MÉTODOS

La presente investigación se desarrolló en el Instituto Nacional de Salud Del Niño – San Borja de la ciudad de Lima Perú. Los datos obtenidos fueron en base a herramientas de investigación y fueron proporcionadas por el Instituto relacionado a los procesos de centro quirúrgico. Esta investigación se realizó en un periodo de 4 meses desde febrero 2023 hasta agosto 2023, debido a la necesidad de contar de un módulo de centro quirúrgico para la programación de salas de operación en el sistema SISGALENPLUS.

Descripción de la Institución

El Instituto Nacional de Salud del Niño - San Borja, pertenece a la red de Salud del Ministerio de Salud del Perú, especializado en la atención del paciente pediátrico y adolescente referidos de otros hospitales, en cirugías y patologías altamente complejas, esta categorizada y acreditada por MINSA con nivel III.2, El Instituto Inicio sus actividades desde el año 2013.

Ubicación geográfica

La dirección del Instituto se encuentra en: Av. Agustín de la Rosa Toro nro. 1399 urb. Jacaranda II (av. Javier prado este 3101) Lima - Lima - San Borja, como se observa en la Fig. 8.

Fig. 8: Ubicación geográfica del Instituto Nacional de Salud del Niño San Borja

3.1 Procedimiento

Esta investigación fue de tipo aplicada [37], ya que su propósito fue generar conocimiento y poner en práctica la teoría para resolver problemas de la vida real. En este caso, se recopiló información sobre la problemática en la sala de operaciones y se aplicó al desarrollo de un módulo de centro quirúrgico. En cuanto al nivel, fue de tipo explicativo [38], ya que buscó analizar las causas y efectos de la problemática identificada en la programación de la sala de operaciones. Este tipo de investigación permite identificar relaciones de causalidad entre variables, explicando cómo y por qué la implementación del módulo de centro quirúrgico influye en la programación de cirugías. A través del análisis de datos recopilados mediante encuestas y observaciones, se pudo determinar el impacto del sistema en la mejora del proceso quirúrgico. Respecto al diseño, esta investigación optó por un enfoque experimental [39], específicamente preexperimental, en el que se identificaron y analizaron las variables independientes para observar su efecto sobre las variables dependientes. En este caso, la variable independiente fue el desarrollo del módulo del centro quirúrgico, el cual tuvo un impacto sobre la variable dependiente, que fue la programación de la sala de operaciones. Finalmente, el método utilizado fue el inductivo [39], apropiado para diseños experimentales, ya que se enfocó en la observación y recopilación de datos para formular hipótesis o teorías.

En esta investigación, se realizaron observaciones y análisis de datos para evaluar el impacto de la variable independiente sobre la dependiente.

Para garantizar la alineación entre el problema, los objetivos y la hipótesis de la investigación, se elaboró una matriz de consistencia (9), la cual permitió estructurar las variables de estudio y su relación con los objetivos de la investigación. Asimismo, se desarrolló la operacionalización de variables (Anexo 8), en la que se definieron las dimensiones e indicadores de cada variable, así como las técnicas e instrumentos de recolección de datos utilizados en el estudio.

La población de estudio de la presente investigación estuvo representada por el personal encargado de la programación de la sala de operaciones del INSNSB, compuesto por 38 profesionales como médicos tratantes, médicos cirujanos y personal administrativo que trabajaba en el centro quirúrgico. En cuanto a la muestra, dado que la programación de la sala de operaciones es una unidad que trabajaba con personal continuamente ocupado, se consideró una muestra por conveniencia de 38 profesionales de la salud, los mismos que estuvieron más involucrados con el área, ya que, de acuerdo con las estadísticas, ellos realizaban la mayor parte

de cirugías y solicitudes, por lo que la unidad de análisis estuvo formada por cada uno de los profesionales encargados de la programación de la sala de operaciones del centro quirúrgico del INSNSB.

En cuanto a las técnicas e instrumentos que permitieron recopilar información, fueron de diferentes tipos, considerando las características y objetivos establecidos, con el fin de contar con información adecuada [38]. En esta investigación se emplearon técnicas como la encuesta, la observación y el análisis documental. Para la recolección de datos de la variable independiente se utilizó como técnica la encuesta con su respectivo cuestionario (Anexo 1), de manera similar para la variable dependiente se utilizó la observación con su ficha de observación (Anexo 3), el análisis de documental con su respectiva ficha de cotejo (Anexo 4) y la encuesta con su respectivo cuestionario (Anexo 2). Asimismo, cada uno de los instrumentos fueron validados a través del juicio de expertos, dichos instrumentos de validación (Anexo 5); por otro lado, para los cuestionarios se evaluó su confiabilidad a través del indicador de Alfa Cronbach para el cuestionario de la variable independiente (Anexo 6) y para el cuestionario de la variable dependiente (Anexo 7).

Para el proceso de recolección de datos de la variable independiente se creó un cuestionario (Anexo 1) que evalúa la calidad del módulo del centro quirúrgico desarrollado. Las preguntas del cuestionario fueron elaboradas considerando la ISO 25010 [28] evaluando su completitud funcional, corrección funcional, pertinencia funcional, capacidad de entendimiento, capacidad de aprendizaje y capacidad de operación, para cada uno de los indicadores se plantearon varias preguntas las mismas que trabajaron con una escala de Likert. Una vez desarrollado el cuestionario, se evaluó la confiabilidad del mismo obteniendo un Alfa de Cronbach del 0.93 (Anexo 6) para posteriormente ser validado por juicio de expertos, este cuestionario fue aplicado a profesionales conocedores de la calidad del software los mismos que llenaron el cuestionario.

Para el proceso de recolección de datos de la variable dependiente se creó un cuestionario (Anexo 2) que evalúa el nivel de satisfacción del usuario en la programación de sala de operaciones. Las preguntas del cuestionario fueron elaboradas considerando la experiencia en uso de herramientas para realizar la programación de salas de operación, para cada uno de los indicadores se plantearon varias preguntas las mismas que trabajaron con una escala de Likert. Luego el cuestionario fue evaluado el grado de confiabilidad obteniendo un Alfa de cronbach

del 0.79 (Anexo 7) para posteriormente ser validado por juicio de expertos, este cuestionario fue aplicado a profesionales conocedores del proceso de centro quirúrgico.

Para la ficha de observación aplicada a la variable dependiente para medir el tiempo de registro de forma manual y aplicando el nuevo módulo haciendo uso de pre test o post test. De cada indicador definido en la matriz de operacionalización de variables. Esta ficha de observación (Anexo 3) el cual observa los tiempos que demora en realizar cada actividad en la solicitud, aprobación, programación de salas de operaciones. Esta ficha ha sido validada por juicio de expertos, por profesionales conocedores en el proceso de Centro Quirúrgico.

Así mismo, esta investigación hace uso de una ficha de cotejo aplicada a la variable dependiente para medir la calidad de los datos obtenidos por el módulo de centro quirúrgico, como la fiabilidad de los datos correctos, fiabilidad del doble registro de aprobación, fiabilidad de inalterabilidad de la información de programación de sala de operaciones en el tiempo. Esta ficha de cotejo (Anexo 4) ha sido validada por juicio de expertos, por profesionales conocedores de la gestión de la Calidad, se termina indicando el tratamiento realizado con los datos obtenidos, utilizando hoja de cálculo Google e indicar la prueba estadística para contrastar la hipótesis (Prueba Z) utilizando un software estadístico.

Para el desarrollo de este módulo de centro quirúrgico, se realizó el diseño bajo la metodología cascada, para ello se utilizó el programa Microsoft Visual Basic 6.0 y base de datos SQL Server, tecnología en la que está desarrollada el sistema institucional SISGALENPLUS. En ese sentido se aplicó la metodología desde el **Análisis del sistema**, en esta fase se realiza un análisis del sistema en el que el módulo se integrará o interactuará, así mismo se consideran aspectos como la infraestructura, los componentes del sistema, las interfaces y la arquitectura en general, además, se analiza cómo el módulo se ajusta a los objetivos y necesidades generales de un sistema más grande, **análisis de requisitos**: en esta fase se recopilan y documentan los requisitos del módulo de sistema, esto implica comprender las necesidades y expectativas de los usuarios y se define claramente qué funcionalidades y características debe tener dicho módulo; **diseño**: una vez que los requisitos se han definido, se pasa a la fase de diseño donde se crea un diseño detallado del módulo, incluyendo su arquitectura, estructura de base de datos y cómo interactuará con otros componentes del sistema; i**mplementación**: en esta fase, el diseño se lleva a cabo mediante la codificación, se escriben el código y los algoritmos necesarios para desarrollar el módulo de sistema, aquí es importante seguir las pautas del diseño y las mejores prácticas de codificación para garantizar la calidad y la coherencia y escalabilidad;

pruebas: una vez que se ha completado la implementación, se procede a las pruebas, consiste en verificar y validar el módulo en busca de errores y asegurarse de que funcione según lo previsto, las pruebas pueden ser de diferentes tipos, como pruebas unitarias, de integración y de sistema; **despliegue**: una vez que el módulo ha pasado con éxito las pruebas, se puede desplegar en el entorno de producción, procediendo a instalar y configurar el módulo en el que estará en funcionamiento; **mantenimiento**: después del despliegue, se entra en la fase de mantenimiento, aquí se realizan correcciones de errores y actualizaciones según sea necesario, también se pueden realizar mejoras y optimizaciones en función del uso de los usuarios aplicando la metodología elegida para el desarrollo del módulo de centro quirúrgico, se procede a definir las fases que se van a considerar para la implementación de dicho proyecto.

3.1.1 Fase: Análisis del sistema

El Instituto Nacional de Salud del Niño San Borja cuenta con el sistema SisGalenPlus para la gestión hospitalaria de la mayoría de sus procesos, por lo que el módulo de centro quirúrgico formara parte de este sistema, el cual se debe considerar lo siguiente de forma obligatoria.

Descripción del Sistema SisGalenPlus

El Sistema Integrado de Gestión Hospitalaria SisGalenPlus ha sido diseñado con el propósito de apoyar a los establecimientos de salud en el correcto registro de información clínica y administrativa, así como en la generación de información gerencial que permita una adecuada toma de decisiones. Además, facilita la mejora de la gestión hospitalaria mediante la digitalización de la información, la transparencia en los procedimientos y un control más eficiente de los recursos hospitalarios. Este sistema fue desarrollado por la Agencia de los Estados Unidos para el Desarrollo Internacional (USAID), en el marco de un convenio de asistencia técnica con el Ministerio de Salud del Perú, con el fin de optimizar los sistemas de información hospitalaria. El software fue transferido gratuitamente al Perú a través de la Carta de Ejecución N.° 527-0423-MOH-4 en el año 2014 y, en 2015, mediante la Resolución Ministerial N.° 696-2015/MINSA, el Ministerio de Salud formalizó la transferencia de propiedad de los derechos de autor del software.

Entre sus módulos principales se encuentran los de hospitalización, consulta externa, emergencia, programación médica y citas, farmacia y facturación.

Integración del módulo de centro quirúrgico

El módulo de centro quirúrgico se integrará completamente con otros módulos del sistema SisGalenPlus, tales como los módulos de hospitalización, consulta externa y emergencia, con el fin de facilitar el flujo de información clave del paciente. Estos módulos proporcionarán datos esenciales como el diagnóstico, sexo, edad, número de historia clínica, número de cama, entre otros, lo cual permitirá un registro inicial eficiente en la solicitud de sala de operaciones. La integración también incluirá el módulo de facturación, de modo que, una vez registrado el reporte operatorio —documento que detalla la cirugía o procedimiento realizado—, esta información se cargará automáticamente en la cuenta corriente del paciente, permitiendo así una correcta facturación de los costos asociados a la intervención quirúrgica.

Además, el módulo de centro quirúrgico tendrá acceso directo al catálogo institucional de procedimientos médico-quirúrgicos. Esta integración permitirá que el personal médico seleccione con precisión el procedimiento que se llevará a cabo durante la operación, asegurando la concordancia entre el acto médico y el registro administrativo.

Infraestructura del sistema

El sistema SisGalenPlus presenta una arquitectura cliente-servidor, en la cual el servidor centralizado maneja las operaciones del sistema, mientras que los clientes (computadoras en distintas áreas del instituto) operan mediante la instalación local del software. El diseño modular de SisGalenPlus permite que sea escalable, lo que significa que se pueden añadir nuevos módulos o funcionalidades sin necesidad de una reestructuración significativa del sistema.

Para garantizar el correcto despliegue y funcionamiento del sistema, es necesario que las estaciones de trabajo y el servidor central cumplan con ciertos requisitos técnicos mínimos, tales como:

Servidor central: Procesador de alto rendimiento, almacenamiento de alta capacidad para manejar grandes volúmenes de datos y redundancia en los sistemas de almacenamiento (RAID).

Estaciones de trabajo (clientes): Computadoras con capacidad suficiente para ejecutar VB6, conectividad de red estable y rápida, así como dispositivos periféricos como impresoras y lectores de códigos de barras.

Ambiente de desarrollo

En cuanto a los estándares establecidos en el sistema SisGalenPlus para el desarrollo de nuevos módulos, estos deben regirse por el documento interno del área de desarrollo de software del INSNSB, denominado “Definición de Estándares de Clases y Formularios en el Desarrollo de Visual Basic SISGALENPLUS”, el cual contempla las siguientes especificaciones:

Arquitectura N - Capas

Bajo estos lineamientos, el sistema adopta una arquitectura lógica en capas. Aunque la solución completa contempla 14 capas/proyectos, para este módulo se emplean 6 capas:

Capa COMÚN: Clases entidad/DTO que transportan información entre capas.

Capa DATOS: Acceso a datos mediante ADO y procedimientos almacenados, estas clases contienen los métodos de insertar, modificar, consultar y eliminar para cada entidad.

Capa ENTIDADES: Contiene las clases de configuración, funciones de conexión, innumerables.

Capa FACTURACIÓN: Servicios transversales de cálculo/generación de cargos y autorizaciones según financiamiento

Capa NEGOCIOS: Contiene las clases de reglas del negocio para cada proceso, para esta investigación se creó la clase ReglasQx.cls

Capa SISGALENPLUS (Presentación): Contienen los formularios y controles de usuario que permiten la funcionalidad con los usuarios del sistema

Esta separación de responsabilidades asegura que los cambios que e realice en una capa tengan un impacto mínimo en las demás capas, asimismo mejora la organización del código y facilita la futura integración de nuevas funcionalidades al sistema.

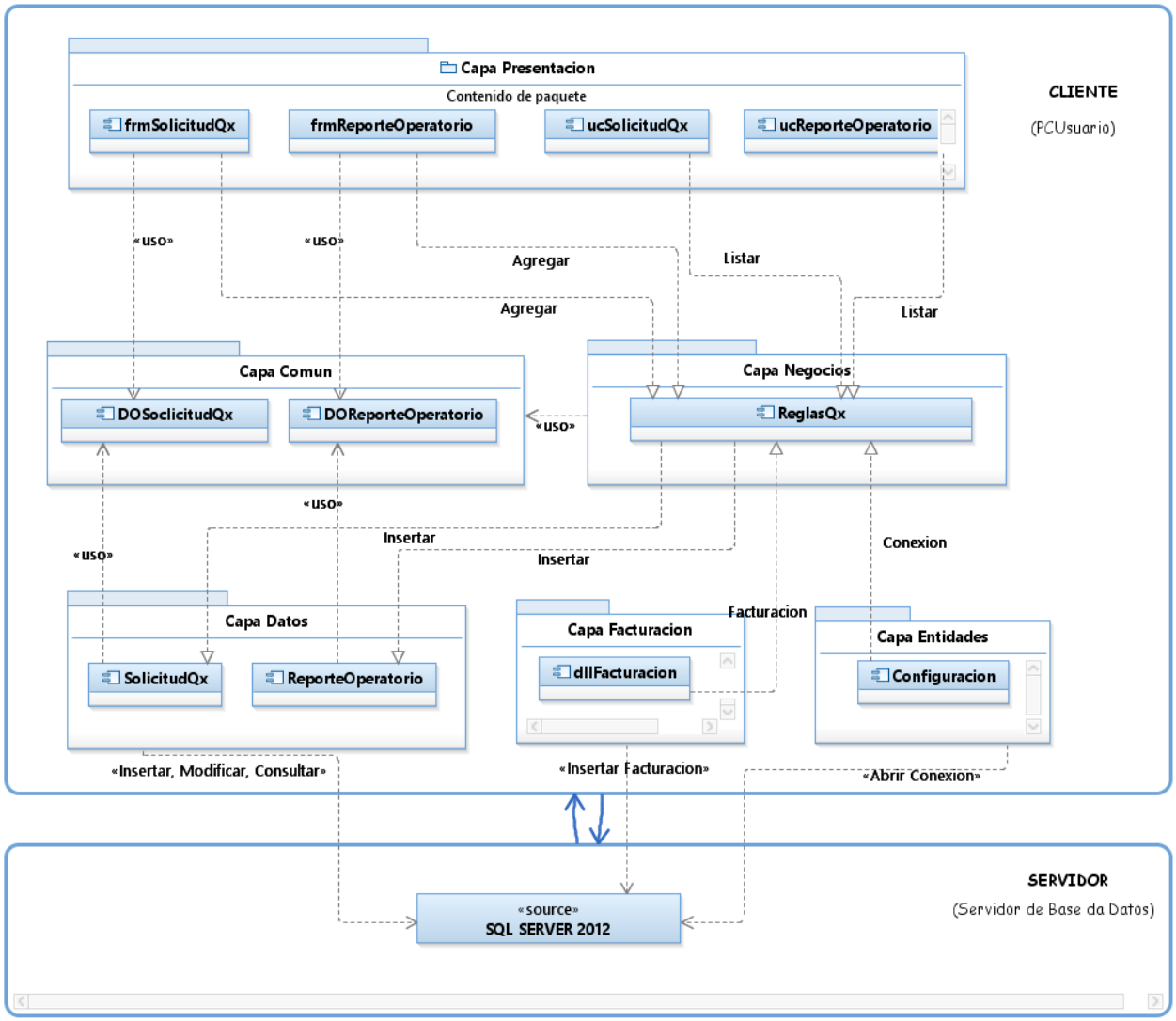


Fig. 9 Diagrama Arquitectura N- Capas SISGALENPLUS

Estándares para clases del sistema

El sistema se encuentra desarrollado en n capas, por lo cual es fundamental manejar la información atravesando dichas capas mediante clases. Se han identificado tres tipos de clases necesarias para el correcto funcionamiento de los módulos desarrollados en SisGalenPlus:

Clases Data Object:

Clases entidad que representan una tabla de la base de datos. El nombre de la clase debe tener el prefijo DO seguido del nombre de la tabla (ej. DONombreTabla.cls). Las propiedades de la

clase deben tener los nombres de los campos de la tabla de base de datos con los prefijos correspondientes, tal como se indica en la Tabla III.

Tabla III: Nomenclatura de variable por tipo de datos en Visual Basic 6.0

Tipo de Dato DB	Tipo dato VB6	Prefijo	Ejemplo
VARCHAR	String	ms_	ms_nombre
NVARCHAR	String	ms_	ms_descripcion
INTEGER	Long	ml_	ml_idusuario
NUMERIC	Double	ml_	ml_dispensado
DECIMAL	Double	ml_	ml_cantidad
MONEY	Currency	mc_	mc_Total
DATE	DateTime	mda_	md_fechaprogramada
DATETIME	DateTime	mda_	mda_fecharegistro
BYTE	Variant	ms_	ms_archivo
BIT	Boolean	mb_	mb_indEgresoFisico

Adicionalmente todas las clases deben de contener la propiedad ***idUsuarioAuditoria***, ya que esto servirá para contener la información del usuario que se encuentra usando la clase

Clases data ADO

Clases ADO contiene como mínimo todas las acciones principales CRUD, conectándose a la base de datos.

- Las clases principales no deben instanciar la conexión, ya que esta, debe ser instanciada desde el proceso de llamado a la clase
- Las clases de conectaran mediante ADO y StoredProcedure versionados, no se permite sentencias SQL en la ejecución de comando
- Nombre de clase no debe tener prefijo: NombreTabla.cls

Las clases debe contener obligatorio las siguientes propiedades que se indican en la Tabla IV.

Tabla IV: Propiedades de una clase ADO

Propiedades de una clase ADO
Property Let **MensajeError**(sValue As String) ms_MensajeError = sValue End Property Property Get **MensajeError**() As String MensajeError = ms_MensajeError End Property Property Set **Conexion**(oValue As ADODB.Connection) Set mo_Conexion = oValue End Property Property Get **Conexion**() As ADODB.Connection Set Conexion = mo_Conexion End Property

Debe contener como mínimo los siguientes métodos y funciones como se indica en la Tabla V.

Tabla V: Funciones y métodos mínimos en clase ADO Visual Basic 6.0

Nombre de función o procedimiento	Descripción
`Function Insertar(ByVal oTabla As DOAtencion) As Boolean`	Método que inserta un objeto en la base de datos, ejecutando un **storedProcedure**, adicionalmente se debe agregar el parámetro de **IdusuarioAuditoria**
`Function Modificar (ByVal oTabla As DOAtencion) As Boolean`	Método que modifica un objeto en la base de datos, ejecutando un **storedProcedure**, adicionalmente se debe agregar el parámetro de **IdusuarioAuditoria**
`Function Eliminar (ByVal oTabla As DOAtencion) As Boolean`	Método que anula o elimina un objeto en la base de datos, ejecutando un **storedProcedure**, adicionalmente se debe agregar el parámetro de **IdusuarioAuditoria**
`Function SeleccionarPorId(ByVal oTabla As DOAtencion) As Boolean`	Método que lista un objeto en la base de datos, enviado el identificador y ejecutando un **storedProcedure**,
`Sub CargaCampos(ByRef oTabla As DOAtencion, oRecordset As Recordset)`	Método interno de la clase que asignar un recordset de base de datos al objeto

Clases de reglas de negocio

Clases de negocio que implementan las clases Entidad y clases ADO validando las reglas de negocio.

- Estas clases deben tener el prefijo Reglas y el nombre del proceso: ReglasFarmacia.cls
- Los métodos de las clases son instanciados desde formularios, Controles de usuario u otras clases de los proyectos de Interface
- Las clases deben instanciar las conexiones, abrir y asignar a las clases ADO, así mismo deben de cerrar las conexiones.

- Si el proceso involucra muchas clases de inserción y o modificaciones, se debe crear las transacciones en la conexión y así mismo la validación de la misma con commit o rollback
- Debe tener como mínimo la propiedad ***MensajeError***, la cual recibirá el mensaje de error y se enviará este a la capa de presentación.
- Ejemplo de método de clase ReglasFarmacia, como se indica en la Tabla VI.

Tabla VI: Funciones y métodos en una Clase Negocio

Utilización de funciones o métodos en clases de negocio
Este método realiza el proceso de realizar una venta en farmacia, `Function` **`AgregaDatosDeVentaDirecta`**`(oDoMovimiento As DoFarmMovimiento, oDoFarmMovimientoVentas As DoFarmMovimientoVentas, oRsDetalleProductos As ADODB.Recordset …)As Boolean`
Este método realiza el proceso de realizar una modificación en venta en farmacia, `Function` **`ModificaDatosVentaDirecta`**`(oDoMovimiento As DoFarmMovimiento, oDoFarmMovimientoVentas As DoFarmMovimientoVentas, oRsDetalleProductos As ADODB.Recordset…) As Boolean`

Estándares de formularios

Clases asociadas a formulario de mantenimiento y controles de usuario que interactúa con el usuario. En clases de formulario de mantenimiento deben contener como mínimo los siguientes métodos que se indican en la Tabla VII.

Tabla VII: Funciones y métodos mínimos en un formulario de mantenimiento

Método o Función	**Descripción**
`Sub` **`CargarDatosAlFormulario`**`()`	Este método se aplica al cargar el formulario, aquí se carga la los datos comunes como listas dependientes u otros, además dependiendo de la opción (Agregar, Modificar, Eliminar) deriva a los métodos **cargaAlosControlesNuevo** para la opción de **a**gregar o **CargarDatosALosControles** para opciones de modificar, consultar, eliminar.

Método o Función	Descripción
`Sub **cargaAlosControlesNuevo**()`	Este método carga los datos por defecto dependiendo de las reglas del negocio para un nuevo registro.
`Sub **CargarDatosALosControles**()`	Este método carga desde la base de datos y asigna a los controles.
`Function **ValidarDatosObligatorios**() As Boolean`	Este método Valida los controles con datos obligatorios propios del proceso.
`Function **ValidarReglas**() As Boolean`	Este método Valida reglas del propio del proceso.
`Sub **CargaDatosAlObjetosDeDatos**()`	Este método es ejecutado después de validar los métodos anteriores el cual el objeto es asignado por los valores de los controles.
`Function **AgregarDatos**() As Boolean`	Método para agregar un registro.
`Function **ModificarDatos**() As Boolean`	Método para modificar un registro.
`Function **EliminarDatos**() As Boolean`	Método para eliminar un registro.
`Sub **LimpiarFormulario**()`	Método que es ejecutado para limpiar datos de los controles y variables globales.
`Sub **ConfiguraPermisos**()`	Método para validar el permiso propio del formulario-

Nombre de formulario con prefijo **frm** . Ejemplo de clase de formulario para mantenimiento: **frmSolicitudQx.frm** como se indica en la Tabla VIII.

Tabla VIII: Estructura de métodos y funciones dentro de un formulario

Estructura de métodos y funciones dentro de un formulario	

`Dim mi_Opcion As sghOpciones`
 	
`**Property Let Opcion(iValue As sghOpciones)**`
`   mi_Opcion = iValue`
`**End Property**`
 	
`**Property Get Opcion() As sghOpciones**`
	`**Function ModificarDatos() As Boolean**`
	`  …`
	`**End Function**`
	
	`**Function EliminarDatos() As Boolean**`
	`…`
	`**End Function**`
	
	`**Sub LimpiarFormulario()**`

```
   Opcion = mi_Opcion
End Property

Sub CargarDatosAlFormulario()
…
ConfiguraPermisos
 Select Case mi_Opcion
     Case sghAgregar
         cargaAlosControlesNuevo
     Case sghModificar
         CargarDatosALosControles
     Case sghConsultar
         CargarDatosALosControles
     Case sghEliminar
         CargarDatosALosControles
 End Select
…
End Sub

Sub cargaAlosControlesNuevo()
…

End Sub

Sub CargarDatosALosControles()
 …
End Sub

Private Sub Form_Load()
    …
    CargarDatosAlFormulario
    …
End Sub

Function  ValidarDatosObligatorios()  As
Boolean
      …
End Function

Function ValidarReglas() As Boolean
      …
End Function


Sub CargaDatosAlObjetosDeDatos()
…
End Function
```

```
    …
End Sub

Sub ConfiguraPermisos()
   …
End Sub

Private Sub btnAceptar_Click()

   Select Case mi_Opcion
   Case sghAgregar
       If    ValidarDatosObligatorios()
Then
           If ValidarReglas() Then
               If AgregarDatos() Then
                       …

LimpiarFormulario
              End If
           End If
       End If
   Case sghModificar
       If    ValidarDatosObligatorios()
Then
           If ValidarReglas() Then
               If ModificarDatos() Then
                …
               End If
           End If
       End If
   Case sghEliminar
           If ValidarReglas() Then
               If EliminarDatos() Then
                    …
               End If
           End If
   End Select

End Sub

Private Sub btnCancelar_Click()
   ..
End Sub

Private Sub btnImprimirReporte_Click()
…
End Sub
```

Clases asociadas a formulario para búsqueda y controles que interactúa con el usuario. En clases de formulario de Búsqueda deben contener como mínimo los siguientes métodos, como se indica en la Tabla IX.

Tabla IX: Lista de métodos mínimos formulario de búsqueda

Nombre Método	Descripción
Public Sub **RealizarBusqueda**()	Método que realiza la búsqueda.
Public Sub **LimpiarFiltro**()	Método que limpia los controles de búsqueda y asigna valores por defecto.
Private Sub **btnBuscar_Click**()	Acción del botón buscar que realiza la búsqueda validando campos obligatorios.
Private Sub **btnLimpiar_Click**()	Acción del botón que llama al método **LimpiarFiltro.**
Private Sub **btnAceptar_Click**()	Acción del botón que devuelve el registro buscado y seleccionado y cierra la ventana.
Private Sub **btnCancelar_Click**()	Acción del botón que cierra la ventana.
Sub AdministrarKeyPreview(KeyCode As Integer)	Método que es invocado para la accesibilidad de teclas.

Nombre de formulario con sufijo búsqueda. Ejemplo de clase de formulario para búsqueda: medicosBusqueda.frm, como se muestra en la Tabla X.

Tabla X: Estructura de métodos de un formulario búsqueda

Estructura de métodos y funciones dentro de un formulario	
Public Sub RealizarBusqueda() … End Sub Public Sub LimpiarFiltro() … End Sub Private Sub btnBuscar_Click() *RealizarBusqueda* End Sub Private Sub btnLimpiar_Click() *LimpiarFiltro* End Sub	Sub AdministrarKeyPreview(KeyCode As Integer) Select Case KeyCode Case vbKeyF6 *btnBuscar_Click* Case vbKeyF7 *btnLimpiar_Click* End Select End Sub Private Sub btnCancelar_Click() .. End Sub Private Sub btnAceptar_Click() … End sub

Estándares para objetos de base de datos

Entre los estándares para la creación de objetos base de datos como tablas, estas deberán contener los siguientes campos en tablas de movimiento, como se indica en la Tabla XI.

Tabla XI: Campos mínimos de tablas para registro de auditoria

Nombre de Campo	Tipo de Dato
sg_Log_Cre_feh	datetime
sg_Log_Cre_Usuario	int
sg_Log_Cre_Equipo	Varchar (20)
sg_Log_Mod_Feh	datetime
sg_Log_Mod_Usuario	int
sg_Log_Mod_Equipo	varchar(20)

En la creación de **stored procedure** para ser invocados por el sistema, los nombres de estos objetos deberán contener el siguiente esquema, como se indica en la Tabla XII.

Tabla XII: Estructura de starded procedures para versionamiento

Estructura de Stored Procedure	Ejemplo de Uso
Usp_select_NombreTablaAccion_fecha	usp_select_MovDiagnosticosDetalleFiltrar_17012023
Usp_delete_NombreTablaAccion_fecha	usp_delete_MovDiagnosticosEliminar_17012023
Usp_insert_NombreTablaAccion_fecha	usp_insert_MovDiagnosticosAgregar_17012023
Usp_update_NombreTablaAccion_fecha	usp_update_MovDiagnosticosModificar_17012023

Seguridad del sistema

El sistema SisGalenPlus cuenta con una estructura de seguridad definida mediante roles, permisos, tipos de cargo y área de trabajo, los cuales son asignados a usuarios del sistema para el uso correcto de sus funciones, esto facilita una mejor gestión de accesos para los diferentes usuarios que interactúan con el sistema. La seguridad del módulo de centro quirúrgico dentro del sistema SisGalenPlus está basada en una estructura de roles y permisos que permite un

control detallado de las acciones que pueden realizar los usuarios según su perfil y funciones dentro del INSNSB, como se presenta en la Fig. 10.

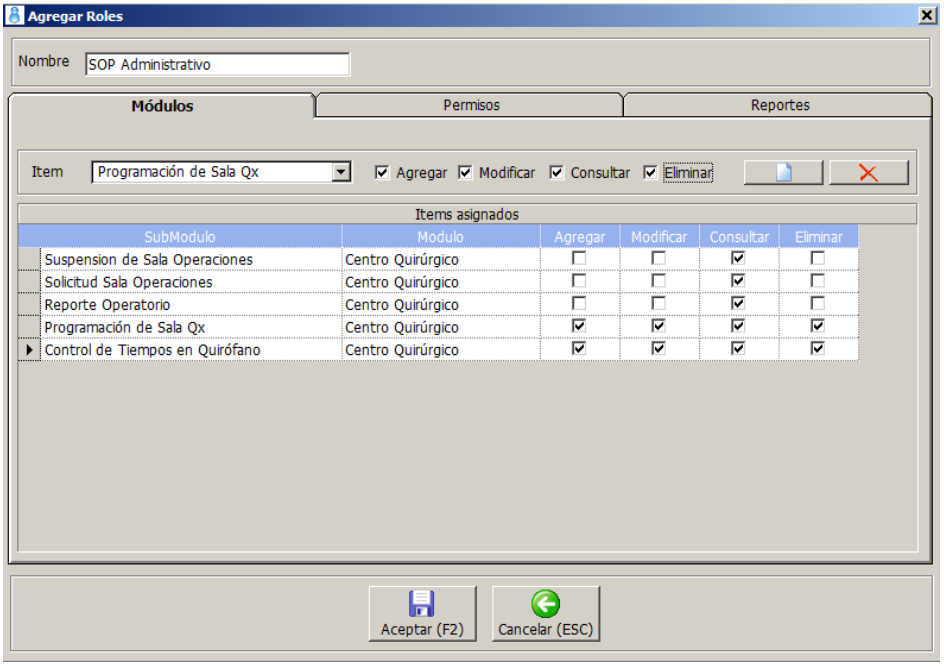


Fig. 10: Formulario de asignación de roles a usuario

Modelo de roles y permisos

La implementación de seguridad sigue un modelo basado en roles (RBAC - Role-Based Access Control), donde cada usuario es asignado a uno o más roles, los cuales definen el conjunto de permisos asociados a cada rol. Estos permisos especifican las operaciones que pueden realizar sobre los diferentes módulos y funcionalidades del sistema, incluyendo el centro quirúrgico.

Los roles se definen para diferentes tipos de usuarios, como médicos, enfermeros, administrativos por los cuales tendrán acceso a diferentes opciones del sistema con accesos permitidos como es Agregar, Modificar, Consultar, Eliminar. Cada rol tendrá un conjunto de permisos que les habilitará para realizar determinadas acciones dentro del sistema, así mismo cada rol tendrá asignado un conjunto de reportes específicos permitidos, como se muestra en la Fig. 11.

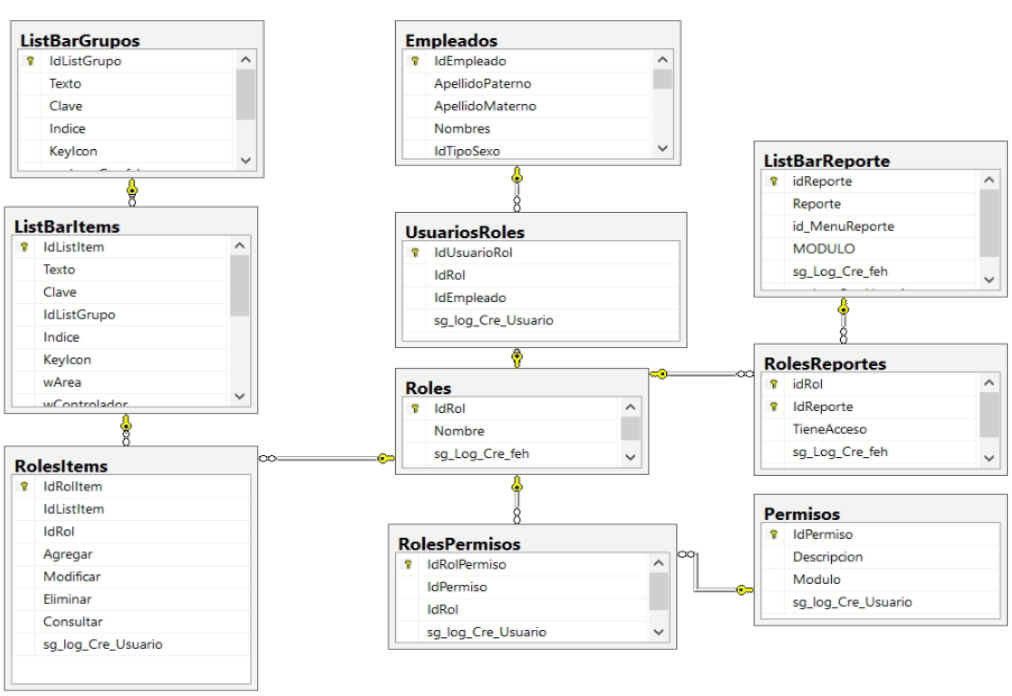


Fig. 11 Modelo de Seguridad en SisGalenPlus

Auditoría y seguimiento

Cada acción realizada en el sistema puede ser auditada mediante la asignación de roles, permitiendo a los administradores del sistema revisar las operaciones realizadas por cada usuario. Esto ayuda a garantizar la integridad y seguridad de la información registrada en el módulo de centro quirúrgico.

3.1.2 Fase: Análisis de requisitos

Alcance del proyecto

El Instituto Nacional de Salud del Niño San Borja tiene la necesidad de automatizar y mejorar la gestión de los procedimientos quirúrgicos a través del desarrollo de un módulo de Centro Quirúrgico integrado en el sistema de información hospitalaria existente, SisGalenPlus.

Este módulo permitirá a los usuarios registrar, gestionar y acceder a información importante como diagnósticos de cirugía realizados, programación de cirugías, personal involucrado, y generar reportes operatorios. El sistema sigue una arquitectura cliente-servidor y se integrará con los módulos de hospitalización, consulta externa y emergencia para obtener datos

esenciales del paciente. La base de datos en SQL Server 2012 será utilizada para el almacenamiento de información, y el desarrollo se realizará en un entorno de VB6.

3.1.2.1 Modelado del negocio

El hospital se enfrenta a la necesidad de agilizar y controlar de manera más efectiva el proceso quirúrgico. Actualmente, los datos se registran manualmente, lo que genera un esfuerzo duplicado, retrasos y mayor posibilidad de errores. Además, el control de inventario y recursos quirúrgicos no está mejorado, lo que afecta la programación eficiente de cirugías.

3.1.2.2 Procesos identificados

Proceso de solicitud de sala de operaciones

Para realizar una solicitud de sala de operaciones, se toma en cuenta dos situaciones, la cuales son programadas o de emergencia, en cada una de ellas el médico realiza un examen pre quirúrgico (ordenes de laboratorio, imágenes, procedimiento), luego genera interconsulta para Evaluación Anestesiológica, por consiguiente si el paciente es apto para cirugía, se valida disponibilidad de los insumos de farmacia necesarios por especialidad, valida disponibilidad de sala y luego se procede a realizar la solicitud de sala de operaciones, esto se realiza mediante un formato pre impreso que proporciona la institución el cual consta de dos modelos uno para sala de cirugía y otro para sala de procedimientos, por lo que deben tener en stock en el consultorio de estos formatos, al digitarse a mano, conlleva tiempo escribir todos los datos del paciente, y si se encuentra hospitalizado, escribir la información del servicio actual del paciente, en cuanto a los procedimientos a realizarse no lleva los códigos CPMS (Clasificación Peruana de Procedimientos Médicos en Salud) establecidos en el catálogo Institucional, así como los códigos CIE10 que define los diagnósticos del paciente así como su clasificación de presuntivo o definitivo. Así mismo se suman otros formatos como el formato Check List de cirugía, consentimiento informado, orden de hospitalización en caso sea paciente ambulatorio como se muestra en la Fig. 12.

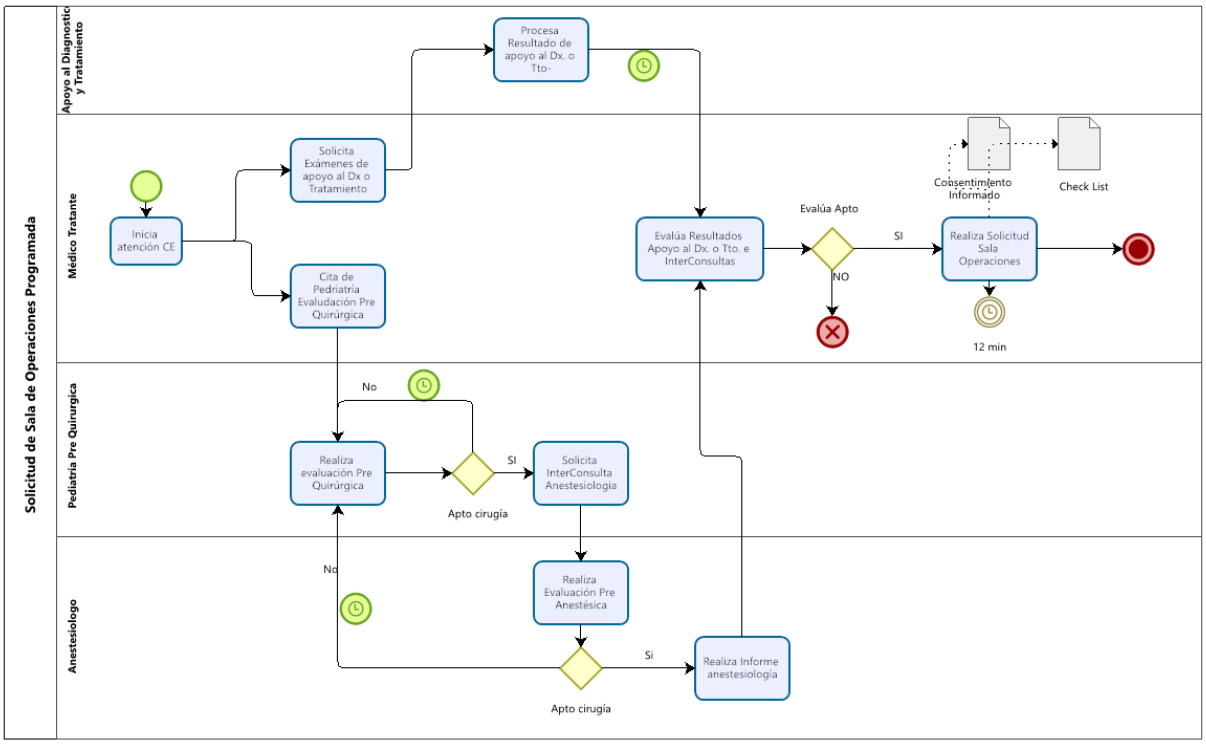


Fig. 12: Proceso de solicitud de sala de operaciones

Proceso de aprobación de realización de cirugías o procedimientos

Cada especialidad médica tiene un personal administrativo encargado de la programación de cirugías, quien presenta la lista preliminar de intervenciones a la jefatura de la especialidad correspondiente para una primera revisión. Una vez aprobadas internamente, estas listas son enviadas a la dirección del hospital, que realiza la aprobación final para todas las cirugías programadas de las diferentes especialidades. Una vez aprobadas, se derivan al área de centro quirúrgico para la programación definitiva del día hábil siguiente, como se observa en la Fig. 13.

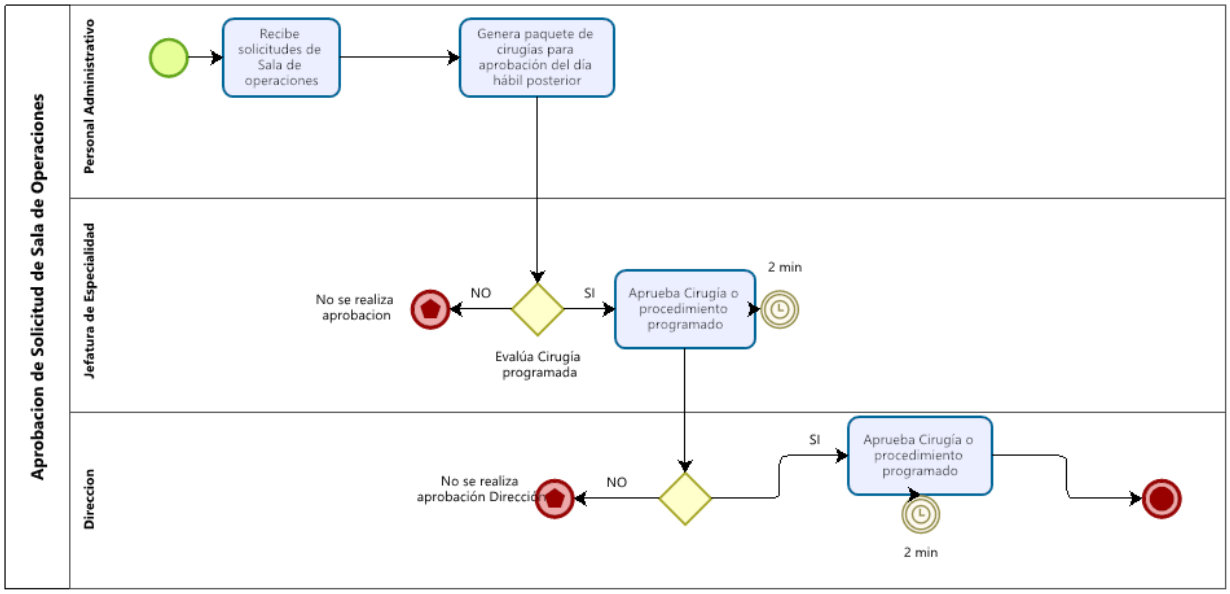


Fig. 13: Proceso de aprobación de solicitud de sala de operaciones

Proceso de programación de sala de operaciones

El responsable de programación recibe las solicitudes de sala de operaciones en formato físico programadas para el día hábil siguiente y transcribe las solicitudes hacia un archivo Excel, el cual toma tiempo digitar los datos del paciente, los procedimientos y diagnósticos de la solicitud, lo cual no solo consume tiempo sino que es propenso a errores en la digitación de la información del paciente, procedimientos, y diagnósticos. Adicionalmente, la asignación de tiempos entre cirugías se realiza manualmente, teniendo en cuenta el tiempo requerido para la limpieza y preparación de la sala, dentro de la programación, el responsable también debe asignar anestesiólogos, asegurando que no haya conflictos de horario entre los profesionales, al término de registro, todas la solicitudes en la programación, se imprime y son derivadas a la

jefatura de centro quirúrgico para ser revisada y validada, para luego realizar la publicación mediante correo electrónico a las áreas pertinentes, como farmacia, jefaturas de especialidades, dirección, central de esterilización, enfermería de sala de operaciones, como se observa en la Fig. 14.

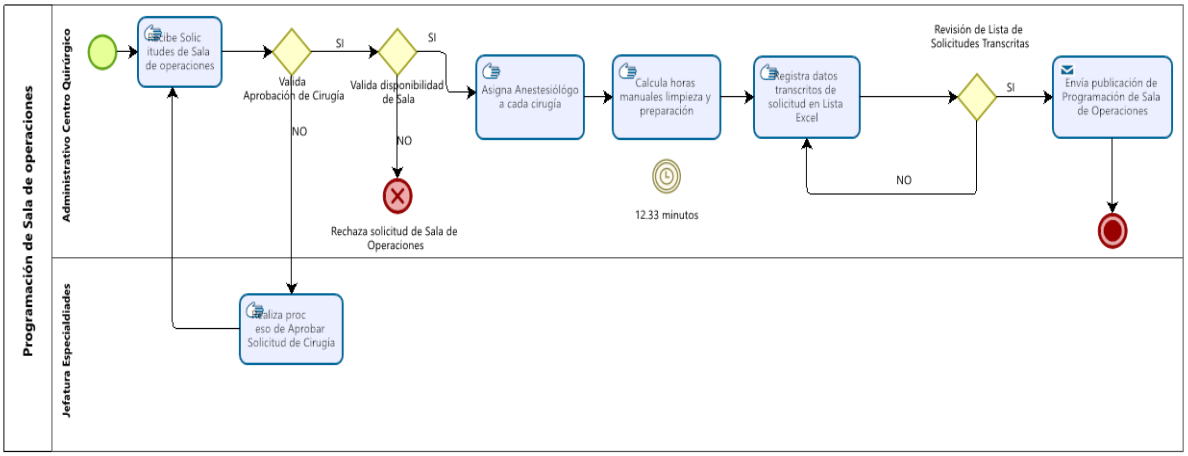


Fig. 14: Proceso de programación de sala de operaciones

Proceso registro de ejecución de la cirugía

Después de cada cirugía, el médico cirujano principal, mediante un formato pre impreso del Reporte Operatorio escribe los datos del paciente, fecha y hora de la cirugía, la cirugía realizada incluyendo los procedimientos, diagnósticos, hallazgos e incidentes, lo que puede provocar retrasos y errores en el registro; así mismo, el médico registra al personal involucrado de la cirugía como enfermero circulante, instrumentista, medico anestesiólogo, medico ayudante, posteriormente se agrega manualmente a la historia clínica del paciente, como se muestra en la Fig. 15.

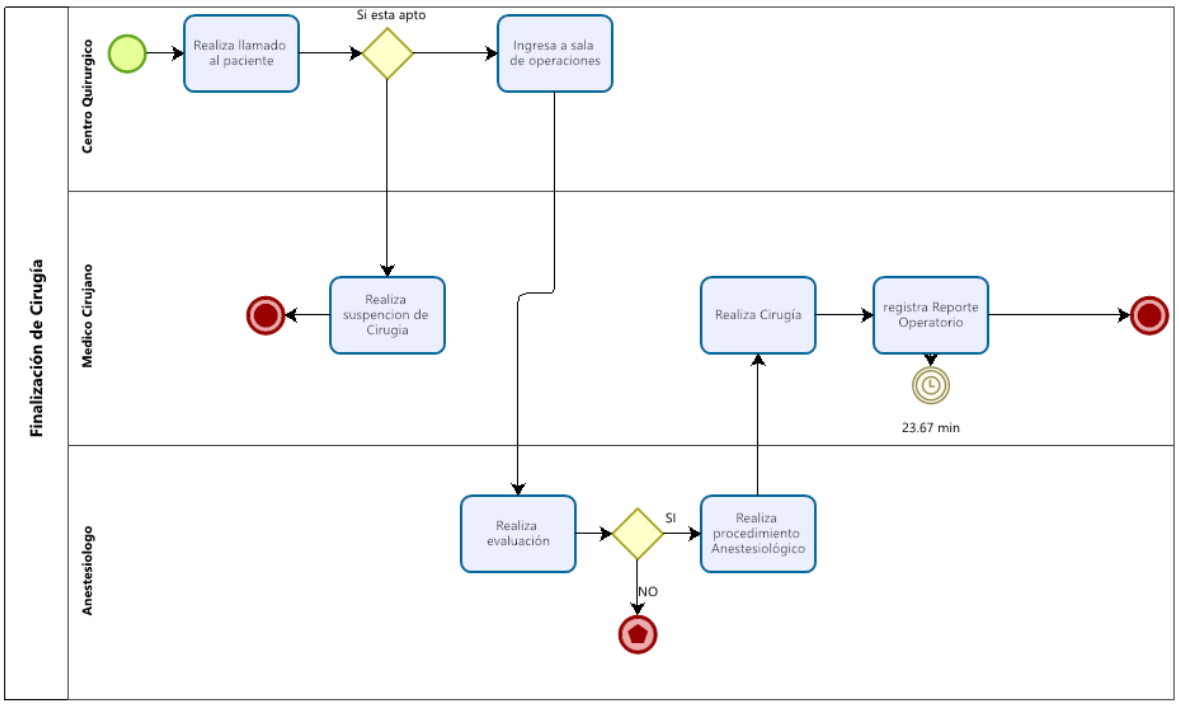


Fig. 15: Proceso de realización de cirugía

3.1.2.3 Identificación de requerimientos

Para solucionar los problemas identificados, se propone implementar el módulo de centro quirúrgico el cual se integrará con los módulos de hospitalización, consulta externa, emergencia, facturación y farmacia, permitiendo un flujo de información continuo, por lo que se han identificado los siguientes requerimientos funcionales y no funcionales

Requerimientos funcionales:

- **RF01** Generar Solicitud de Sala de Operaciones
- **RF02** Aprobar Solicitud de sala de Operaciones
- **RF03** Crear la Programación de Sala de Operaciones
- **RF04** Cerrar Programación de Sala de Operaciones
- **RF05** Registrar el Reporte Operatorio
- **RF06** Registrar Suspensión de Sala de Operaciones
- **RF07** Registros adicionales de traslados a centro quirúrgico como horas de llamado, hora de ingreso, hora de salida a quirófano, tiempo de recuperación
- **RF08** Integración a la facturación del Reporte Operatorio
- **RF09** Integración con órdenes de Anatomía Patológica

- **RF10** Consulta inmediata de las cirugías desde los módulos de hospitalización, emergencia y consulta externa.
- **Requerimientos no funcionales:**
- **RNF01** Alta disponibilidad del sistema para asegurar que el módulo esté operativo en todo momento.
- **RNF02** Seguridad en el acceso a los datos, asegurando que solo el personal autorizado pueda acceder a la información quirúrgica.
- **RNF03** Escalabilidad del sistema, permitiendo futuras expansiones e integraciones.
- **RNF04**: El sistema debe implementarse bajo una arquitectura Cliente-Servidor Multicapa, asegurando separación de responsabilidades entre capa de presentación, lógica de negocio y acceso a datos.

3.1.2.4 Modelado de casos de uso de negocio

De acuerdo a los requerimientos funcionales y no funcionales se realizó el modelo de casos de uso del negocio para identificar y estructurar las iteraciones entre los actores del negocio y los procedimientos realizados en la investigación.

Actores del negocio

En el diagrama de actores del negocio, como se observa en la Fig. 14, se identificaron las entidades externas que interactúan con el sistema del módulo de centro quirúrgico. Estos actores incluyen (ver Fig. 16).

- **AN1 Paciente**: representa al beneficiario final de los servicios quirúrgicos, ya sea ambulatorio, de emergencia u hospitalizado.
- **AN2 Familiar del paciente**: actúa como representante del paciente, en especial en la toma de decisiones.
- **AN3 Aseguradora**: representa a las entidades encargadas de financiar los procedimientos quirúrgicos según el tipo de aseguramiento.
- **AN4 Entidad reguladora**: supervisora del cumplimiento de las normativas de salud relacionadas con el proceso quirúrgico.

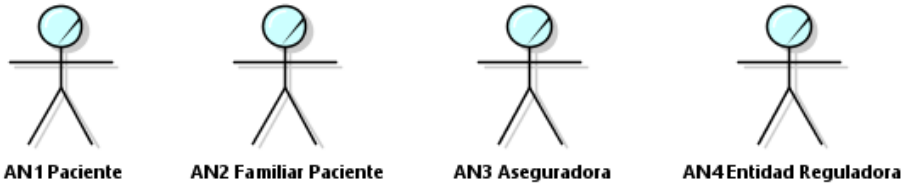


Fig. 16: Actores del negocio

Casos de uso del negocio

El diagrama de casos de uso del negocio, como se muestra en la Fig. 17, establece las principales actividades que conforman el flujo del módulo quirúrgico. Cada caso de uso fue definido en función de los objetivos del negocio y las interacciones necesarias para alcanzar los resultados esperados.

- **CUN1 Solicitud de cirugía**: permite realizar la solicitud del procedimiento quirúrgico, incluyendo diagnóstico y procedimientos a realizar.
- **CUN2 Aprobación de cirugía**: permite la validación y autorización de la solicitud de cirugía por parte de las jefaturas y dirección.
- **CUN3 Programación de cirugía:** consiste en la asignación de horarios, salas, y personal anestesiólogo involucrado en el procedimiento.
- **CUN4 Notificación de programación de cirugía:** asegura la comunicación del cronograma a las áreas involucradas como farmacia, anestesiología y especialidades médicas.
- **CUN5 Registro de reporte operatorio y suspensiones**: implica documentar los procedimientos realizados o los motivos de suspensión quirúrgica, además de actualizar la información en el sistema.

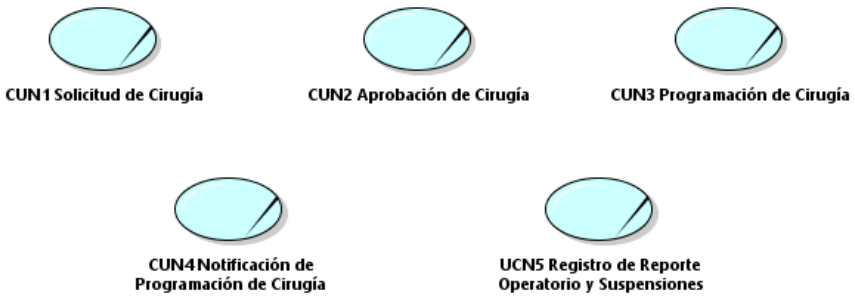


Fig. 17: Casos de uso del negocio

Objetivos del negocio

El diagrama de objetivos del negocio, como se observa en la Fig. 18, los principales propósitos estratégicos que se buscan alcanzar mediante la implementación del módulo quirúrgico son:

- **ON1** Mejorar la eficiencia operativa: permite reducir tiempos y mejorar el uso de los recursos quirúrgicos.
- **ON2** Mejorar la calidad de atención: permite garantizar un servicio oportuno, seguro y centrado en las necesidades del paciente.
- **ON3** Garantizar la transparencia y trazabilidad: permite asegurar que todas las actividades quirúrgicas sean monitoreadas y auditables.

Fig. 18: Objetivos del negocio

Diagrama de caso de uso del negocio

El diagrama general de casos de uso del negocio, como se muestra en la Fig. 19 establece las interacciones entre los actores principales del negocio y los casos de uso del negocio identificados en el módulo de centro quirúrgico.

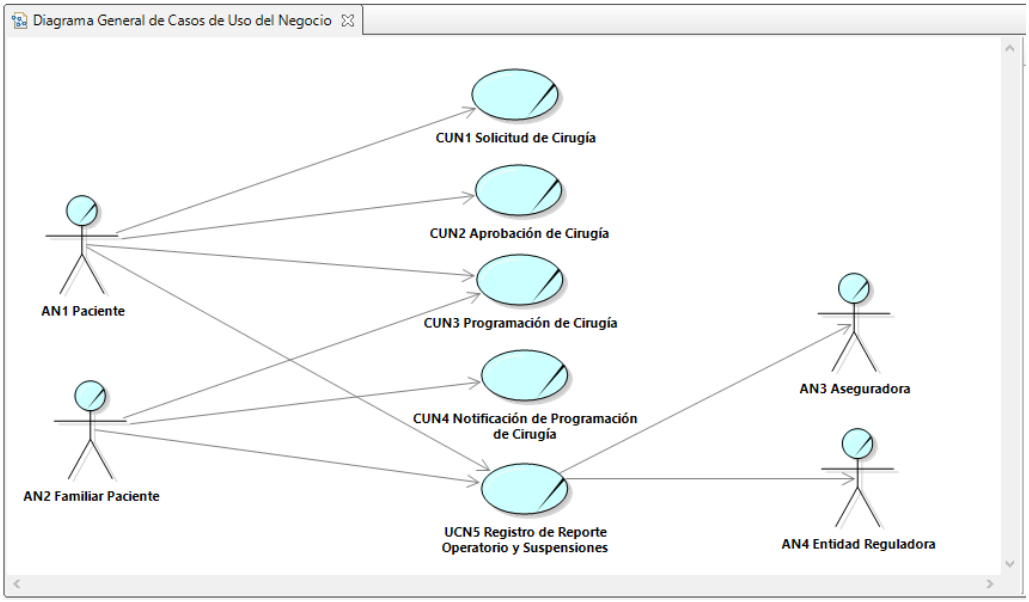


Fig. 19: Diagrama de casos de uso del negocio

3.1.3 Fase: Diseño

3.1.3.1 Diagramas de caso de uso del sistema

Actores del sistema

En la Fig. 20, se observa el diagrama de actores del sistema, identifica las personas y/o entidades que interactúan directamente con el sistema, desempeñando roles específicos en los procesos quirúrgicos. Los actores definidos son:

- **Médico solicitante:** actor encargado de registrar las solicitudes de cirugía desde la consulta externa, hospitalización o emergencias.
- **Jefatura médica:** actor encargado de validar y aprobar las solicitudes de sala de operaciones, asegurando el primer filtro para poder realizar la programación de dicha solicitud.
- **Dirección del hospital:** actor que Supervisa y da la aprobación final de todas las cirugías, programadas, previamente autorizadas por los jefes de servicio, para pacientes ambulatorios y hospitalizados.
- **Administrativo de centro quirúrgico:** actor responsable de programar las salas quirúrgicas asignando un horario, recursos y por consiguiente notificar al personal.
- **Médico cirujano:** Actor que realiza los procedimientos o cirugías y registra los reportes operatorios o suspensiones de sala de operaciones
- **Enfermería de centro quirúrgico:** actor que apoya en la gestión de insumos, y registro de datos adicionales de las cirugías.
- **Médico anestesiólogo:** actor que participa en la programación y ejecución de las cirugías, aportando evaluaciones prequirúrgicas y asistencia anestesiológica durante el procedimiento.

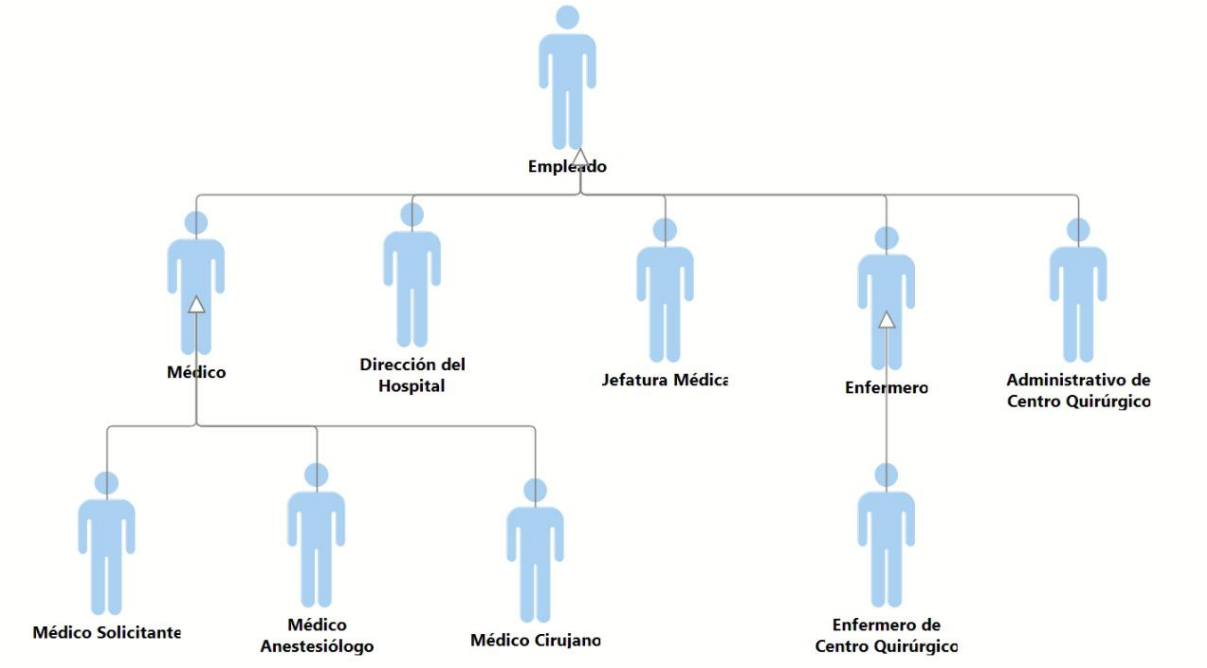


Fig. 20: Actores del sistema

Casos de uso del sistema

En la Fig. 21, se observa el diagrama de casos de uso del sistema el cual establece las funcionalidades implementadas en el módulo de centro quirúrgico:

- **Registrar solicitud de sala de operaciones:** permite al médico solicitante ingresar la información necesaria para solicitar la programación de una cirugía.
- **Aprobar solicitud de cirugía:** proceso de revisión y autorización de las solicitudes por parte de las jefaturas y dirección.
- **Programar cirugía:** permite al personal administrativo realizar la gestión de recursos, horarios y asignación de personal médico para cada procedimiento quirúrgico.
- **Cerrar día programación de cirugía:** asegura que el cronograma quirúrgico sea comunicado a todas las áreas involucradas.
- **Registrar reporte operatorio:** permite al Cirujano el registro del procedimiento realizado.
- **Registrar suspensión de cirugía:** permite al personal involucrado documentar las razones y responsables de las cirugías canceladas.

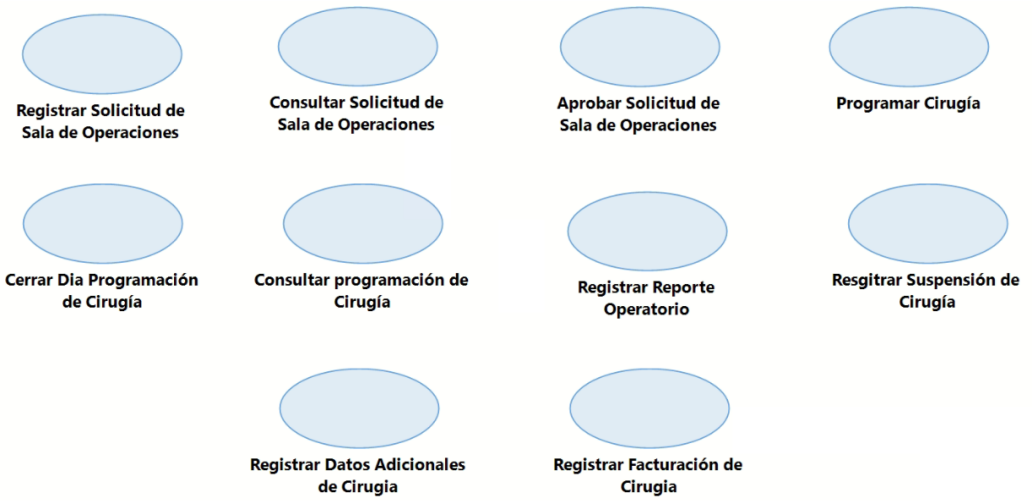


Fig. 21: Casos de uso del sistema

Diagrama de casos de uso del sistema

En la Fig. 22 se observa el diagrama general de casos de uso del sistema, representa las interacciones específicas entre los actores del sistema y las funcionalidades del módulo quirúrgico, comprendiendo las dependencias entre actores y casos de uso del sistema.

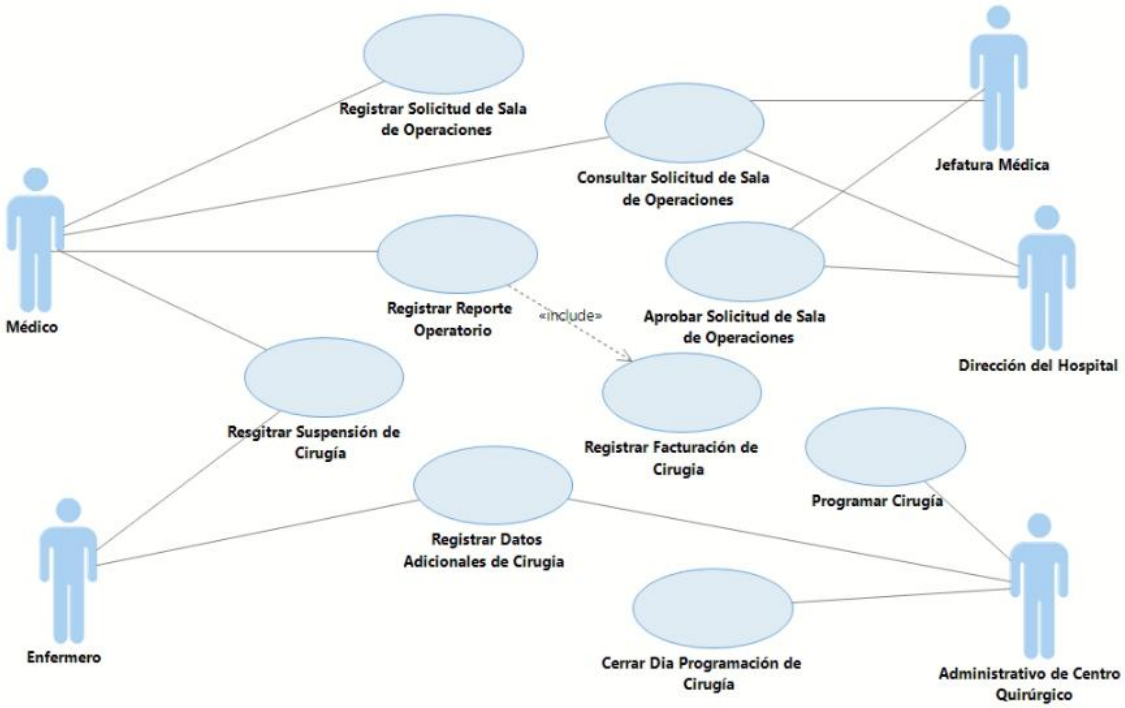


Fig. 22: Diagrama de casos de uso del sistema

Rol Jerárquico	Rol Específico	Accede al Sistema	Módulo de acceso principal	Descripción de funciones clave
Empleado	—	No (abstracto)	—	Rol conceptual, no interactúa directamente
Médico	Médico solicitante	si	Solicitud de Cirugía	Registra solicitudes de cirugía; consulta estado de programación
Médico	Médico cirujano	Si	Cirugía / Reporte operatorio	Registra reporte operatorio, suspensiones, y genera facturación
Médico	Médico anestesiólogo	Si	Datos Preoperatorios	Ingresa evaluación anestésica; consulta solicitud
Enfermero	Enfermería quirúrgica	Si	Sala de operaciones	Registra insumos, datos postoperatorios, apoyo en suspensión
Administrativo	Administrativo de centro quirúrgico	Si	Programación Quirúrgica	Programa cirugías, asigna salas y turnos, cierra programación diaria
Administrativo	Jefatura médica	Si	Validación de Solicitudes	Revisa y aprueba solicitudes quirúrgicas (primer filtro)
—	Dirección del hospital	si	Validación Final / Aprobación	Otorga aprobación institucional de cirugías, accede a reportes de programación

Arquitectura de análisis del negocio

En la Fig. 23 se observa el diagrama de la arquitectura de análisis del negocio, separa los procesos en dos capas:

- **Capa general**: incluye funciones transversales, como consultas de pacientes, diagnósticos y especialidades del sistema SisGalenplus, utilizadas para otros módulos del sistema.

- **Capa específica**: incluye procesos y funciones propios del módulo de centro quirúrgico, como la gestión de solicitudes, programación, y registro de reportes operatorios.

Esta arquitectura modular permite una visión clara y eficiente entre funcionalidades generales y específicas, asegurando que el sistema sea escalable y adaptable a futuros cambios en el módulo.

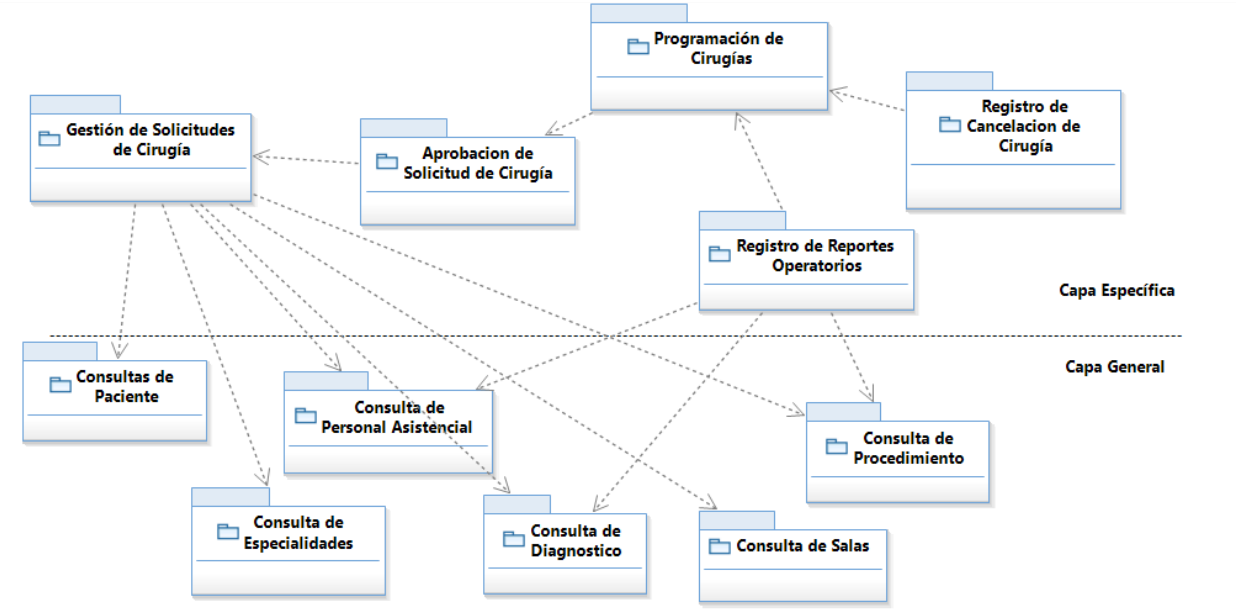


Fig. 23: Arquitectura de análisis del negocio

Análisis relacional de casos de uso

En la Fig. 24, se muestra el análisis relacional de casos de uso, detallándose cómo cada funcionalidad del sistema se organiza en paquetes, agrupando casos de uso relacionados para simplificar la estructura y garantizar el modularidad del sistema. De los cuales los paquetes identificados son los siguientes.

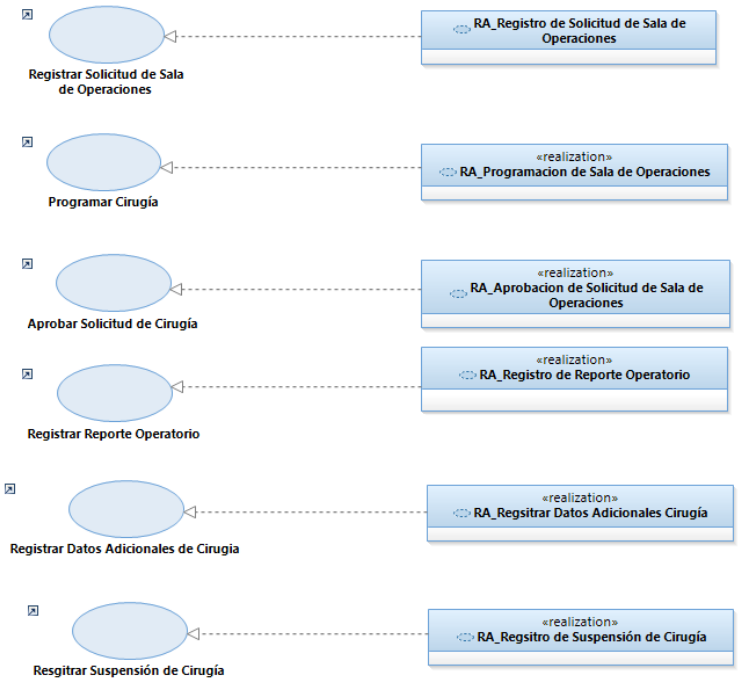


Fig. 24: Análisis relacional de casos de uso

Análisis relacional de paquete de consulta de personal asistencial

En la Fig. 25, se muestra el análisis relacional de este paquete, se enfoca en las interacciones específicas relacionadas con la consulta y selección del personal asistencial que participará en las cirugías. Teniendo como objetivo de garantizar que el personal médico y asistencial esté asignado para cada procedimiento quirúrgico, mejorando la calidad del servicio y minimizando errores en la programación.

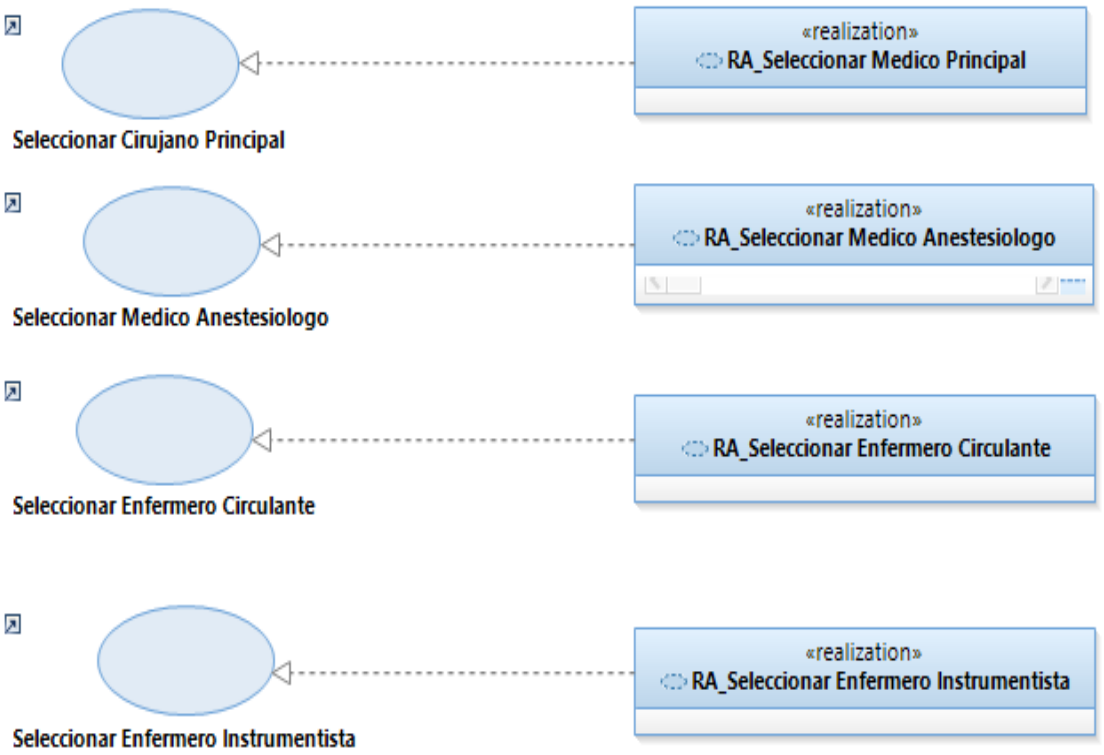


Fig. 25: Análisis relacional de consulta de personal asistencial

Diagrama de clases de análisis: realización de seleccionar médico anestesiólogo

En la Fig. 26, se observa el diagrama de clases de análisis, modela las clases y relaciones necesarias para implementar la funcionalidad para seleccionar un médico anestesiólogo en el módulo quirúrgico. Establece y organiza los elementos en la selección, identificando las responsabilidades y atributos de cada clase.

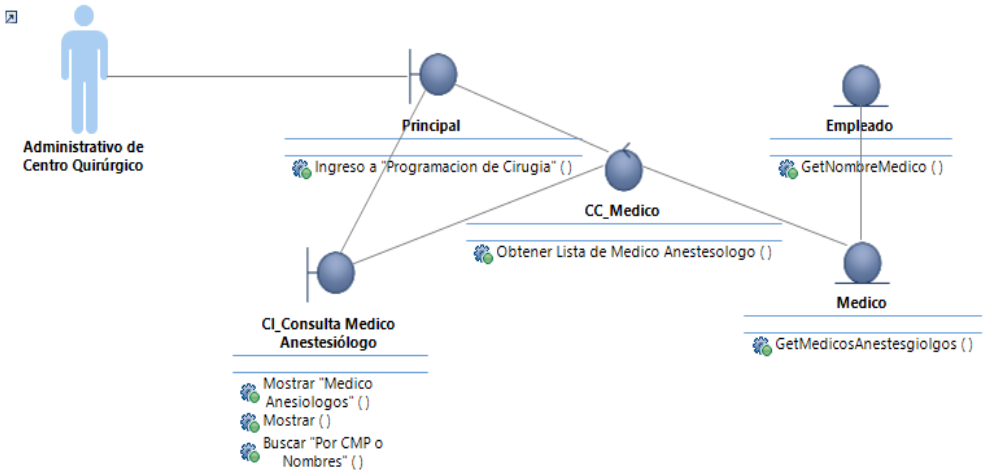


Fig. 26: Diagrama de clases de análisis: seleccionar médico anestesiólogo

Diagrama de secuencia: seleccionar médico anestesiólogo

En la Fig. 27, se muestra el diagrama de secuencia, asimismo, detalla la interacción temporal entre las clases y actores para completar la funcionalidad de seleccionar un médico anestesiólogo en proceso de programación de sala de operaciones.

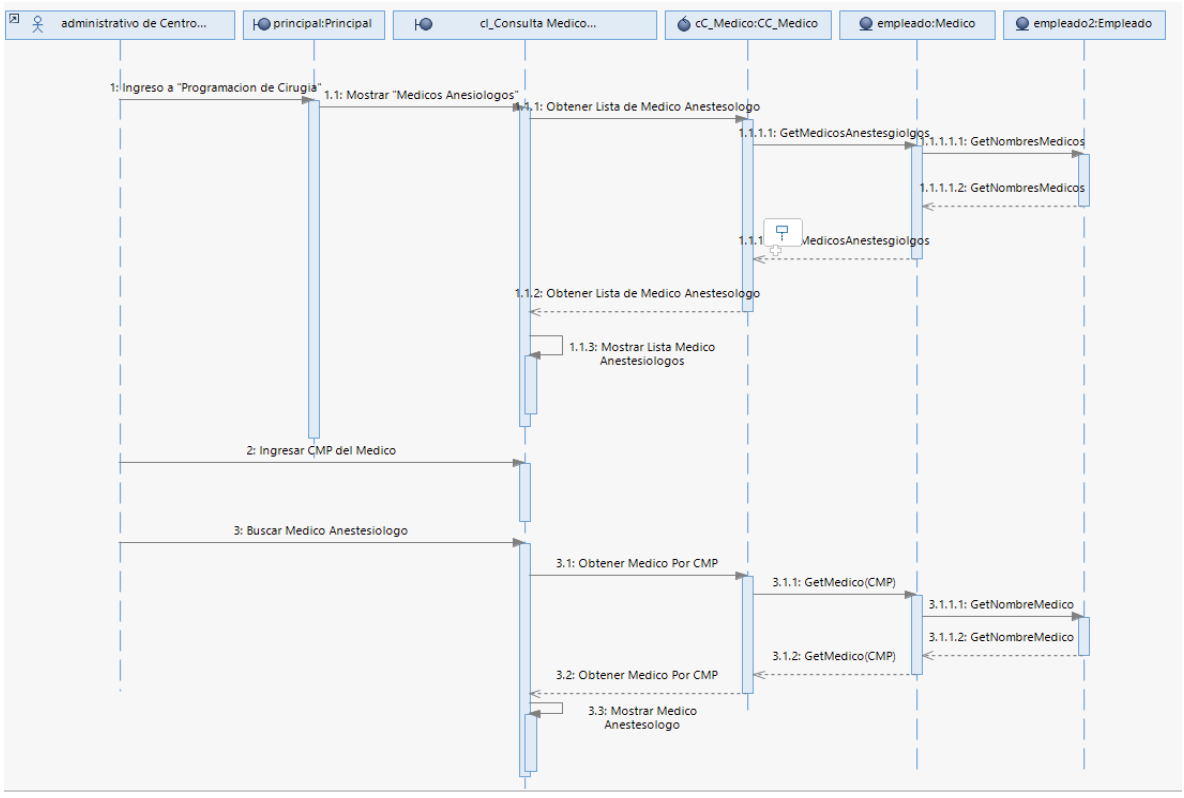


Fig. 27: Diagrama de secuencia: seleccionar médico anestesiólogo

3.1.3.2 Especificación de caso de uso del sistema

CU01: Registrar solicitud de sala de operaciones

Se inicia cuando el médico tratante del paciente solicita una sala de operación para una cirugía o procedimiento, como se puede apreciar en la Fig. 28.

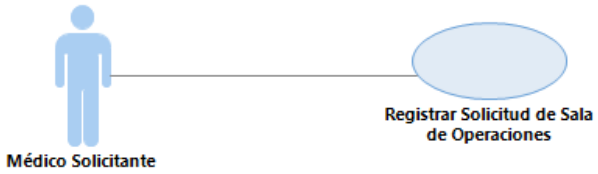


Fig. 28: Especificación CU01 Registro de solicitud de sala de operaciones

En la Tabla XIII se describe las especificaciones del registro de solicitud de sala de operaciones.

Tabla XIII: CU01 Registrar solicitud de sala de operaciones

CU01 Registrar solicitud de sala de operaciones	
ID	**CU01**
Nombre	Registrar Solicitud de Sala de Operaciones
Descripción	Permite al médico registrar una solicitud de cirugía indicando diagnóstico, procedimiento, sala, y médicos involucrados.
Actor	Médico Solicitante
Precondiciones	El médico debe haber iniciado sesión en el sistema.
	El paciente debe estar registrado en el sistema.
Postcondiciones	La solicitud queda registrada con estado " Pendiente de Aprobación ".
Flujo Normal	1. El médico accede al módulo "Solicitudes de Sala de Operaciones".
	2 al agregar una nueva solicitud el medico puede buscar por Nro. de Historia o búsqueda por otros datos como apellidos, nombres, documento
	2. Selecciona el paciente desde el listado.
	3. Completa los datos requeridos en la pantalla:
	Diagnóstico, procedimiento, sala, fecha y hora programada, médico principal y ayudantes, servicio que opera, especialidad, resumen de diagnóstico médico, resumen de cirugía a realizar.
	4. Confirma los datos seleccionando "Aceptar".
	5. El sistema guarda la solicitud y muestra un mensaje de confirmación.

CU01 Registrar solicitud de sala de operaciones	
Sub flujos	**Registrar Procedimiento sin Costo**
	1. Al realizar la solicitud coordina con el área de costos para solicitar código equivalente o adición del procedimiento por el área de Costos
	2. Nuevamente se consulta la lista de procedimientos
	3. El sistema muestra el nuevo procedimiento para asignar a la solicitud
	Modificar Datos Antes de Guardar:
	1. Si se requiere modificar algún dato ingresado, el médico selecciona "Modificar".
	2. Ajusta los valores en los campos correspondientes.
	3. Confirma los cambios seleccionando "Aceptar".
Flujo Alternativo	1. Si faltan datos obligatorios, el sistema muestra un mensaje indicando los campos faltantes.
	2. Si el paciente no cuenta con una atención con esta abierto, se debe coordinar con área de seguros para la apertura de ser el caso, de lo contrario se necesitará crear otra atención valida.
	3. La solicitud se puede realizar desde la atención de consulta externa, emergencia u hospitalización permitiendo tener los datos de la atención vigente en el formulario

CU02: Aprobar solicitud de cirugía

En el proceso de aprobación de cirugía se da doble aprobación, por parte de la jefatura del servicio y la segunda por parte de dirección como se muestra en la Fig. 29.

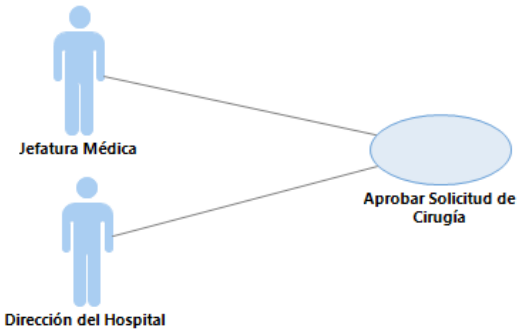


Fig. 29: Especificación CU02 aprobación solicitud de cirugía

En la Tabla XIV se describen las especificaciones de aprobación de solicitud de sala de operaciones.

Tabla XIV: CU01 Aprobar solicitud de sala de operaciones

CU01 Aprobar solicitud de sala de operaciones	
ID	**CU02**
Nombre	Aprobar Solicitud de Sala de Operaciones
Descripción	Permite a la jefatura médica y la dirección aprobar solicitudes quirúrgicas para programación.
Actor	Jefatura Médica, Dirección del Hospital
Precondiciones	- La solicitud debe estar registrada y en estado "Pendiente de Aprobación".
Postcondiciones	- La solicitud cambia a estado “Pre Aprobada”. O “Aprobada” dependiente del usuario.
Flujo Normal	1. El usuario accede al módulo "Solicitudes de Cirugía". 2. Selecciona una solicitud desde la lista, donde se muestran los siguientes campos: **Estado de la Solicitud** (Solicitado, Programado, Suspendido, etc.). **Paciente** (Nombre, Apellido, Historia Clínica). **Servicio Quirúrgico y Diagnóstico**. **Fecha y Hora Solicitada**. 3. Revisa los datos y selecciona el estado “Pre Aprobar” para el caso de Jefatura de Servicio y estado “Aprobado” para el caso de Dirección Médica. 4. El sistema actualiza el estado de la solicitud.
Flujo Alternativo	1. Si la solicitud no cumple los criterios, el usuario puede no “aprobar” o “pre aprobar”

CU03: Programar sala de operaciones

De acuerdo al requerimiento funcional se establece que el usuario administrativo de centro quirúrgico es quien programa las cirugías a partir de las solicitudes aprobadas por dirección, este caso está representado mediante la Fig. 30.

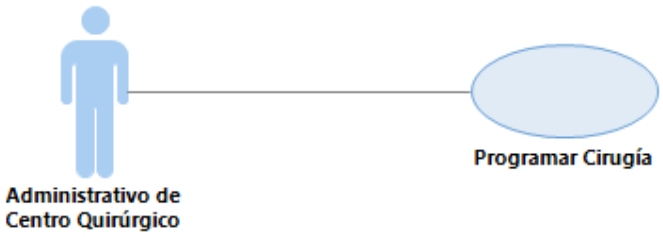


Fig. 30: Especificación CU03 Registro de programación de sala de operaciones

En la Tabla XV se describen las especificaciones de programación de solicitud de sala de operaciones.

Tabla XV: CU03 Programar sala de operaciones

CU03 Programar sala de operaciones	
ID	**CU03**
Nombre	Programar Sala de Operaciones
Descripción	Permite al personal administrativo programar las cirugías aprobadas asignando sala, médicos anestesiólogos y horarios.
Actor	Personal Administrativo del Centro Quirúrgico
Precondiciones	La solicitud debe estar aprobada.
Postcondiciones	La cirugía queda programada y registrada en el sistema.
Flujo Normal	1. El usuario accede al módulo “Programación de Sala de Operaciones”. 2. Selecciona una solicitud desde la ventana de búsqueda o por número de solicitud, donde se muestran: **Paciente y servicio quirúrgico**. **Fecha y hora inicial** (editable). **Sala quirúrgica y especialidad**. **Equipo quirúrgico asignado,**

CU03 Programar sala de operaciones	
	3. Completa los datos necesarios y selecciona "Aceptar". 4. El sistema guarda la programación. 5. El formulario limpia los controles para un nuevo registro de Sala de Operaciones
Flujo Alternativo	1. Si no hay disponibilidad de recursos, no se realiza la programación de Sala de Operaciones.

CU04: Cerrar programación de cirugía

Este proceso se da al término del registro de las programaciones de cirugía en sala de operaciones, se puede apreciar en la Fig. 31.

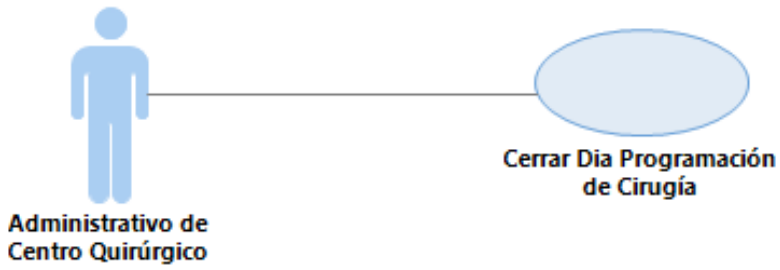


Fig. 31: Especificación CU04 Cerrar día de programación de sala de operaciones

En la Tabla XVI se describe las especificaciones de proceso de cerrar día de programación de sala de operaciones.

Tabla XVI: CU04 Cerrar día de programación de sala de operaciones

CU04 Cerrar día de programación de sala de operaciones	
ID	**CU02**
Nombre	Cerrar Programación de Dia de Sala de Operaciones
Descripción	Mientras que el Dia de Programación de Sala de operaciones este abierto, se permitirán registrar programaciones de salas de operaciones, este proceso realiza el cierre de Dia para emitir el documento de programación y notificar a las áreas involucradas.
Actor	Administrativo de Centro Quirúrgico
Precondiciones	- Las cirugías deben estar programada.
Postcondiciones	- Los actores involucrados reciben las notificaciones correctamente.

CU04 Cerrar día de programación de sala de operaciones	
	- No se permitirá registra nuevas programaciones de cirugía para el día correspondiente
Flujo Normal	1. El usuario termina de registrar las cirugías programadas. 2. Realiza el cierre. consultando la fecha de cierre 3. Se genera un reporte en formato Excel con información de la programación de Sala de Operaciones: **Fecha y hora de la cirugía**. **Sala quirúrgica asignada**. **Especialidad Quirúrgica**. **Equipo médico asignado (cirujano y anestesiólogo)**. 4. Las notificaciones se entregan a los usuarios responsables por correo.
Flujo Alternativo	1. Antes de notificar el usuario puede abrir el día de programación par adicionar alguna cirugía o retirarla

CU05: Registrar reporte operatorio

Este proceso se inicia al término de la realización de la cirugía por parte el médico cirujano principal, este caso de uso desencadena otro caso el cual genera la facturación a la cuenta del paciente, como se aprecia en la Fig. 32.

Fig. 32: Especificación CU05 Registrar reporte operatorio

En la Tabla XVII se describen las especificaciones de registro de reporte operatorio.

Tabla XVII: CU05 Registrar reporte operatorio

CU05 Registrar reporte operatorio	
ID	**CU05**
Nombre	Registrar Reporte Operatorio
Descripción	Permite al cirujano registrar los detalles del procedimiento quirúrgico realizado, incluyendo diagnóstico, equipo quirúrgico, procedimiento, incidencias, complicaciones.
Actor	Cirujano
Precondiciones	La cirugía debe estar programada o de emergencia
	El cirujano debe haber iniciado sesión.
Postcondiciones	El reporte operatorio queda registrado en el sistema.
Flujo Normal	1. El cirujano accede al módulo "Reporte Operatorio".
	2. Selecciona una cirugía programada desde la lista.
	3. Completa los campos requeridos:
	- **Procedimiento Realizado, Diagnóstico Final, Equipo Quirúrgico, Complicaciones**.
	4. Confirma los datos seleccionando "Aceptar".
	5. El sistema guarda el reporte operatorio.
	6. El sistema genera facturación en cuenta de paciente.
Flujo Alternativo	**Medico Programado diferente a medico realiza cirugía:**
	1. Si el medico que realiza la cirugía no es el medico programado, debe solicitar el cambio en centro quirúrgico
	2. Una vez realizado el cambio el medico vuelve a consulta las cirugías asignadas para generar el reporte Operatorio
	Por Validación de datos obligatorios
	1. Si falta algún dato obligatorio, el sistema muestra mensajes de error indicando los campos a completar.

CU06: Registrar suspensión de sala de operaciones

Este acto sucede cuando la cirugía programada en sala de operaciones se suspende y esto puede ser registrado por personal del equipo de la cirugía programada como puede ser el médico cirujano, enfermero de centro quirúrgico o medico anestesiólogo asignado, como se aprecia en la Fig. 33.

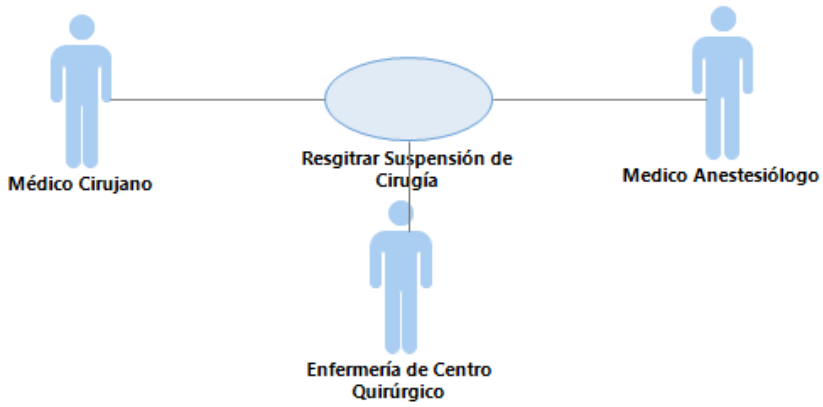


Fig. 33: Especificación CU06 Registrar suspensión de sala de operaciones

En la Tabla XVIII se describe las especificaciones de suspensión de sala de operaciones.

Tabla XVIII: CU06 Registrar suspensión de sala de operaciones

CU06 Registrar suspensión de sala de operaciones	
ID	**CU06**
Nombre	Registrar suspensión de sala de operaciones
Descripción	Permite registrar suspensiones quirúrgicas indicando el motivo, fecha, hora, tipo de suspensión y el responsable.
Actor	Cirujano, anestesiólogo
Precondiciones	- La cirugía debe estar programada.
Postcondiciones	- La cirugía queda registrada como "Suspendida" en el sistema.
Flujo Normal	1. El usuario accede al módulo. 2. Selecciona la cirugía programada desde la lista. 3. Completa los siguientes campos: **Fecha y hora de suspensión**. **Motivo de suspensión**. **Tipo de suspensión (médica, administrativa, etc.)**. **Responsable**. 4. Confirma seleccionando "Aceptar". 5. El sistema guarda la suspensión y actualiza el estado de la cirugía.
Sub flujos	**Modificar datos antes de confirmar**: 1. Si es necesario corregir un dato, el usuario selecciona "Modificar". 2. Ajusta los campos requeridos. 3. Confirma seleccionando "Guardar Cambios".
Flujo Alternativo	1. Si faltan datos obligatorios, el sistema muestra mensajes de error y solicita su corrección.

CU07: Consultar solicitudes de cirugía

Ese caso de uso se aplica en el requerimiento de consulta de las solicitudes realizadas al paciente como se muestra en la Fig. 34.

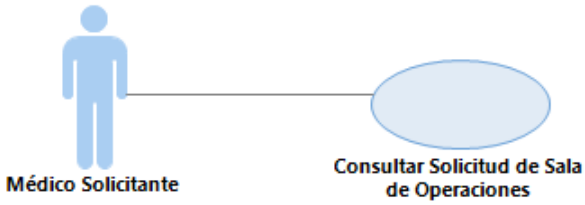


Fig. 34: Especificación CU07 Consultar solicitudes de cirugía

En la Tabla XIX se describe las especificaciones de consulta de solicitud de sala de operaciones

Tabla XIX: CU07 Consultar solicitudes de cirugía

CU07 Consultar solicitudes de cirugía	
ID	**CU07**
Nombre	Consultar solicitudes de sala de operaciones
Descripción	Permite al usuario visualizar solicitudes de cirugía según criterios de búsqueda como paciente, médico, estado de la solicitud, o fecha.
Actor	Médico solicitante, personal administrativo
Precondiciones	El usuario debe haber iniciado sesión.
Postcondiciones	Se muestra una lista de solicitudes que coinciden con los filtros aplicados.
Flujo Normal	1. El usuario accede al módulo “Solicitudes de Sala de Operaciones”. 2.Introduce criterios de búsqueda en los siguientes campos: Paciente (nombre, historia clínica). Fecha (inicio y fin). Estado de la solicitud (solicitado, aprobado, programado, etc.). Servicio quirúrgico y especialidad. 3. El sistema muestra una lista con columnas: Estado de la Solicitud. Diagnóstico y procedimiento planeado. Médico solicitante y fecha programada. 4. El usuario selecciona una solicitud para ver detalles.
Sub flujos	**Exportar Resultados**: 1. El usuario selecciona "Exportar" en el menú de opciones. 2. Selecciona el formato de exportación (Excel). 3. El sistema genera y descarga el archivo con los datos filtrados.

CU07 Consultar solicitudes de cirugía	
	Limpiar filtros: 1. El usuario selecciona "Limpiar Filtros". 2. El sistema resetea los campos de búsqueda.
Flujo Alternativo	1. Si no hay coincidencias, el sistema muestra un mensaje indicando que no se encontraron resultados.

CU08: Registrar datos adicionales de cirugía

Este proceso se realiza al término del día de sala de operaciones, estas anotaciones por ser de carácter rápido en centro quirúrgico, se pasan al sistema para términos estadísticos, como son registros de horas de entrada y salida de centro quirúrgico, esto puede ser registrado por el personal de enfermería que realiza las anotaciones o el personal administrativo de Centro Quirúrgico, como se aprecia en la Fig. 35.

Fig. 35: Especificación CU07 Registrar datos adicionales de cirugía

En la Tabla XX se describen las especificaciones de registrar datos adicionales de cirugía.

Tabla XX: CU08 Registrar datos adicionales de cirugía

CU08 Registrar datos adicionales de cirugía	
ID	**CU08**
Nombre	Registrar datos adicionales de cirugía
Descripción	Permite al personal registrar información adicional sobre la cirugía, como horarios de ingreso, salida y traslado.
Actor	Enfermería, personal administrativo
Precondiciones	La cirugía debe estar en proceso.
Postcondiciones	Los datos adicionales quedan registrados y vinculados a la cirugía.
Flujo Normal	1. El usuario accede al módulo "Datos Adicionales de Cirugía". 2. Selecciona la cirugía desde la lista.

CU08 Registrar datos adicionales de cirugía	
	3. Completa los siguientes campos: **Hora de llamado, ingreso y salida del quirófano**. **Tiempo de recuperación en URPA**. **Observaciones**. 4. Confirma seleccionando "Aceptar". 5. El sistema guarda los datos ingresados.
Sub flujos	**Editar datos registrados**: 1. Si se detecta un error, el usuario selecciona "Editar". 2. Corrige los datos ingresados. 3. Guarda los cambios seleccionando "Aceptar". **Marcar datos como pendientes**: 1. Si un dato no está disponible en el momento, el usuario marca "Pendiente". 2. El sistema registra el estado pendiente y permite completarlo más tarde.
Flujo Alternativo	1. Si los horarios ingresados son inconsistentes, el sistema alerta al usuario para corregirlos.

3.1.3.3 Diagrama de clases

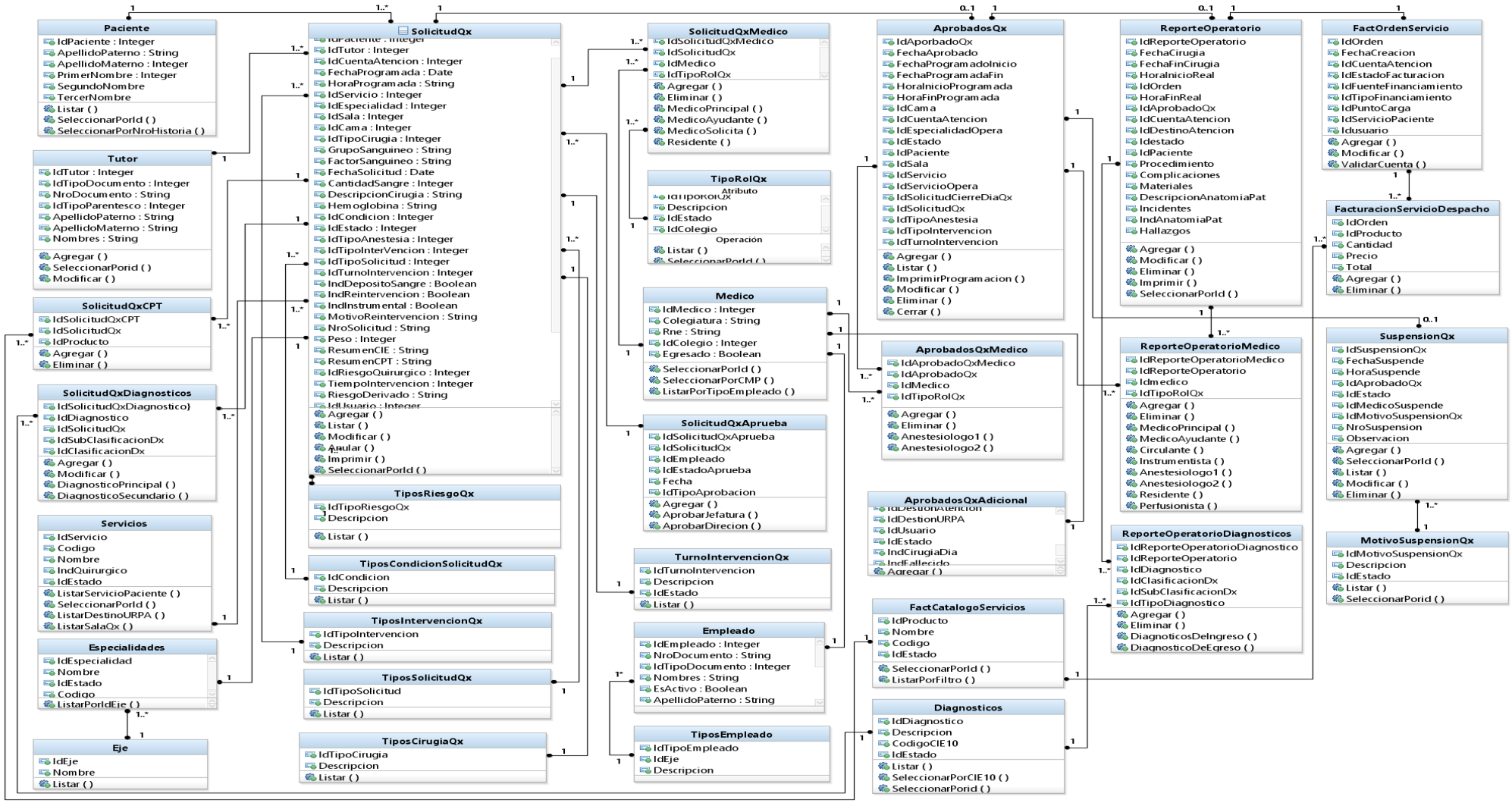


Fig. 36: Diagrama de clases

3.1.3.4 Prototipos

Prototipo de registro de solicitud de sala de operaciones

Para la interface de registro de solicitud de sala de operaciones, se considera los campos obligatorios para la solicitud de sala de operaciones de acuerdo al diagrama de clases, debe mostrar datos generales del paciente como nombres y apellidos, Nro. de historia clínica, servicio de procedencia y cama si fuese hospitalizado, el sexo del paciente, edad, peso, tipo de seguro (particular, asegurado SIS, entre otros), como datos obligatorios en la solicitud de sala de operaciones, se establece la fecha programada de cirugía, hora programada de cirugía, tiempo de la cirugía, tipo de intervención como (programada o emergencia), tipo de cirugía como (mayor, menor o procedimiento), diagnostico CIE10, procedimientos o cirugías a realizar, seleccionar médico cirujano, como la Fig. 37.

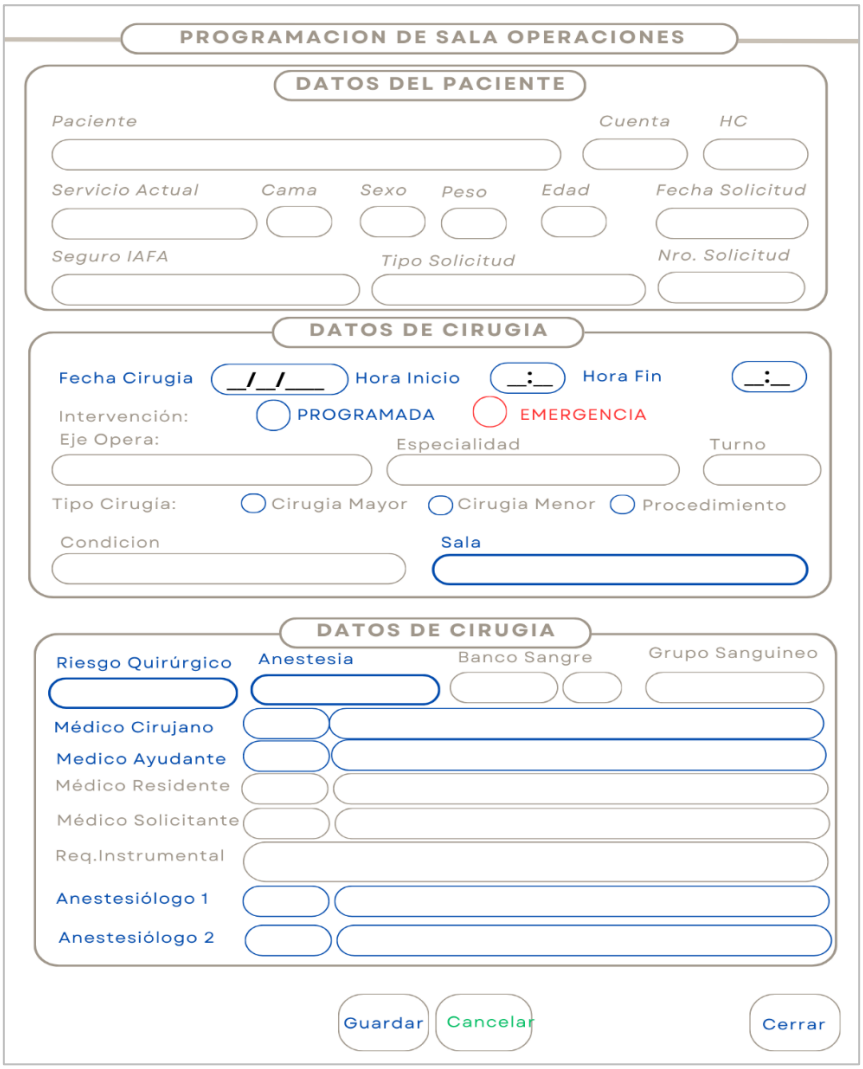


Fig. 37: Prototipo de programación de sala de operaciones

Prototipo de registro de reporte operatorio

Par la interface del registro de reporte operatorio, se considera los campos obligatorios de acuerdo al diagrama de clases, debe mostrar datos generales del paciente, como datos obligatorios en el reporte operatorio, se establece la Fecha de cirugía, hora inicio cirugía, hora fin de cirugía, seleccionar medico intervencionista, circulante, diagnósticos y procedimientos post quirúrgicos, descripción del procedimiento realizado, incidencias, complicaciones, materiales usados, destino del paciente, como en la Fig. 38.

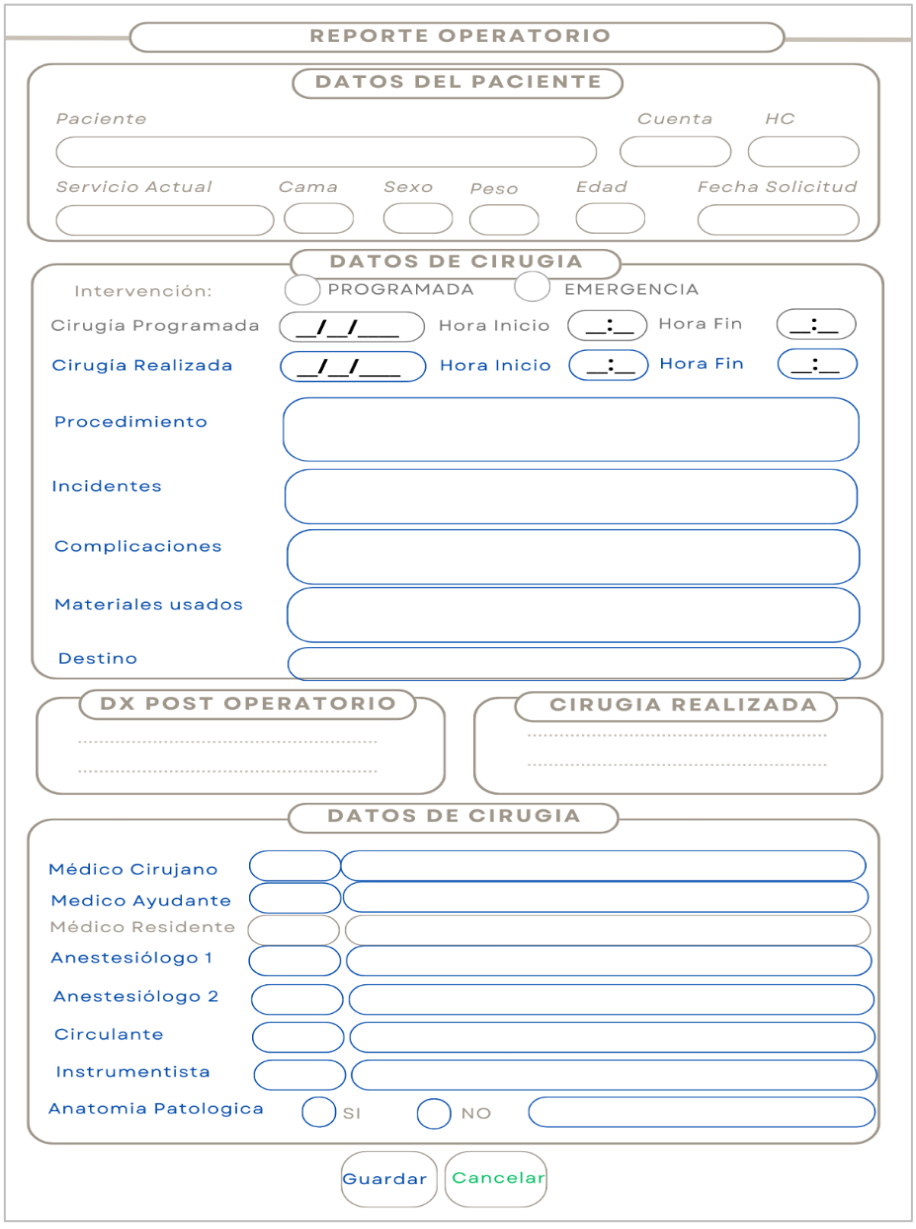


Fig. 38: Prototipo de registro de reporte operatorio

Prototipo de registro de datos adicionales de cirugía

Para la interface del registro de datos adicionales de cirugía, se considera los campos obligatorios de acuerdo al diagrama de clases, debe mostrar datos generales del paciente, como datos obligatorios en el formulario, se establece hora de llamado al paciente, hora de ingreso, hora de ingreso a sala, hora de ingreso a anestesiología, hora de ingreso a cirugía, hora de salida de cirugía, hora de salida de anestesiología, como en la Fig. 39.

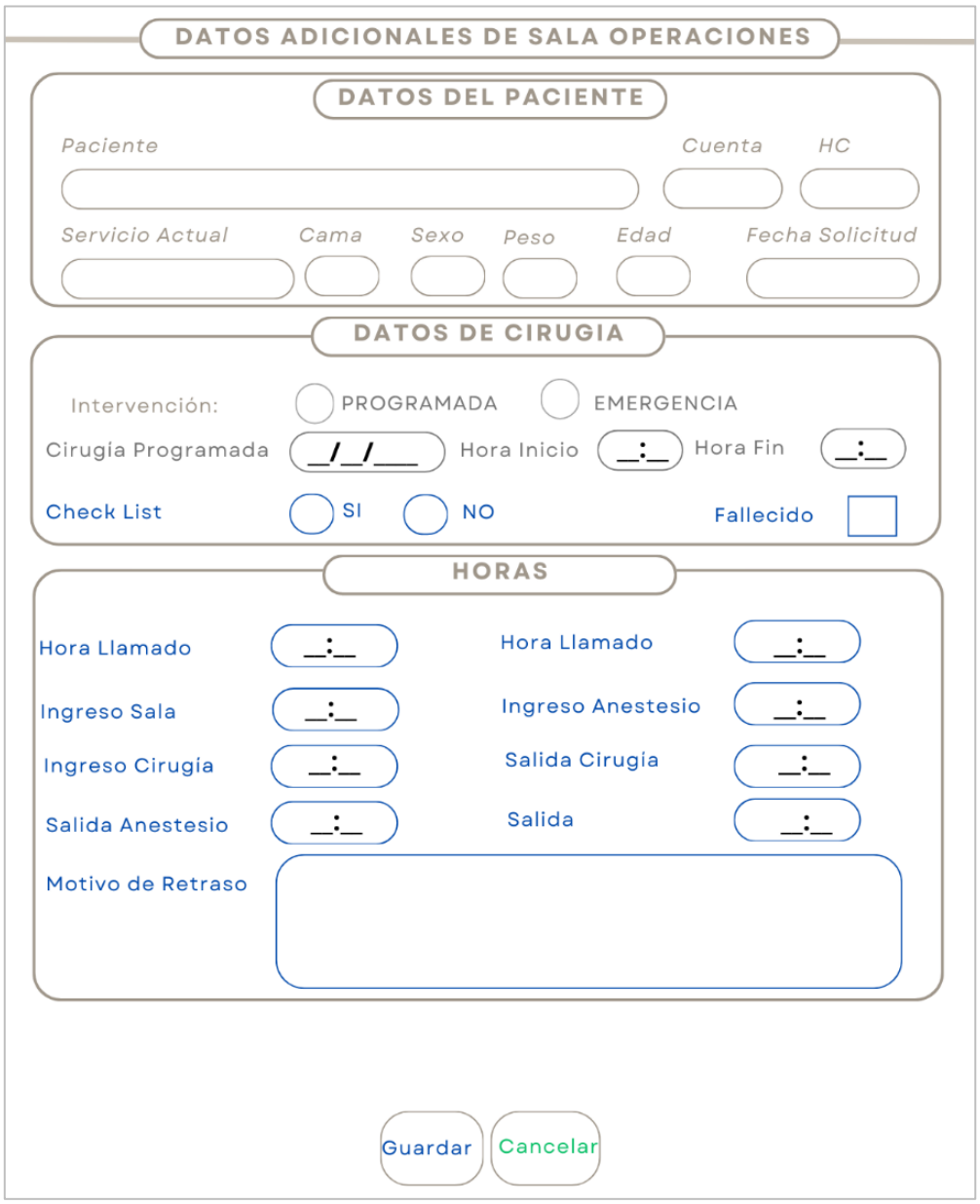


Fig. 39: Prototipo de registro de datos adicionales de cirugía

Prototipo de registro de suspensión de sala de operaciones

Par la interface del registro de Suspensión de cirugía, se considera los campos obligatorios de acuerdo al diagrama de clases (Fig. 36), debe mostrar datos generales del paciente, como datos obligatorios en el formulario se establece fecha de suspensión, hora de suspensión, tipo de suspensión, motivo de suspensión, responsable de suspensión de cirugía, como la Fig. 40.

Fig. 40: Prototipo de registro de suspensión de sala de operaciones

3.1.4 Fase: Implementación

En esta fase se realizó la construcción de la base de datos, se establecieron relaciones con tablas centrales del sistema, como, por ejemplo: pacientes.

En la Fig. 41 se estable el modelo de base de datos del módulo de Centro Quirúrgico.

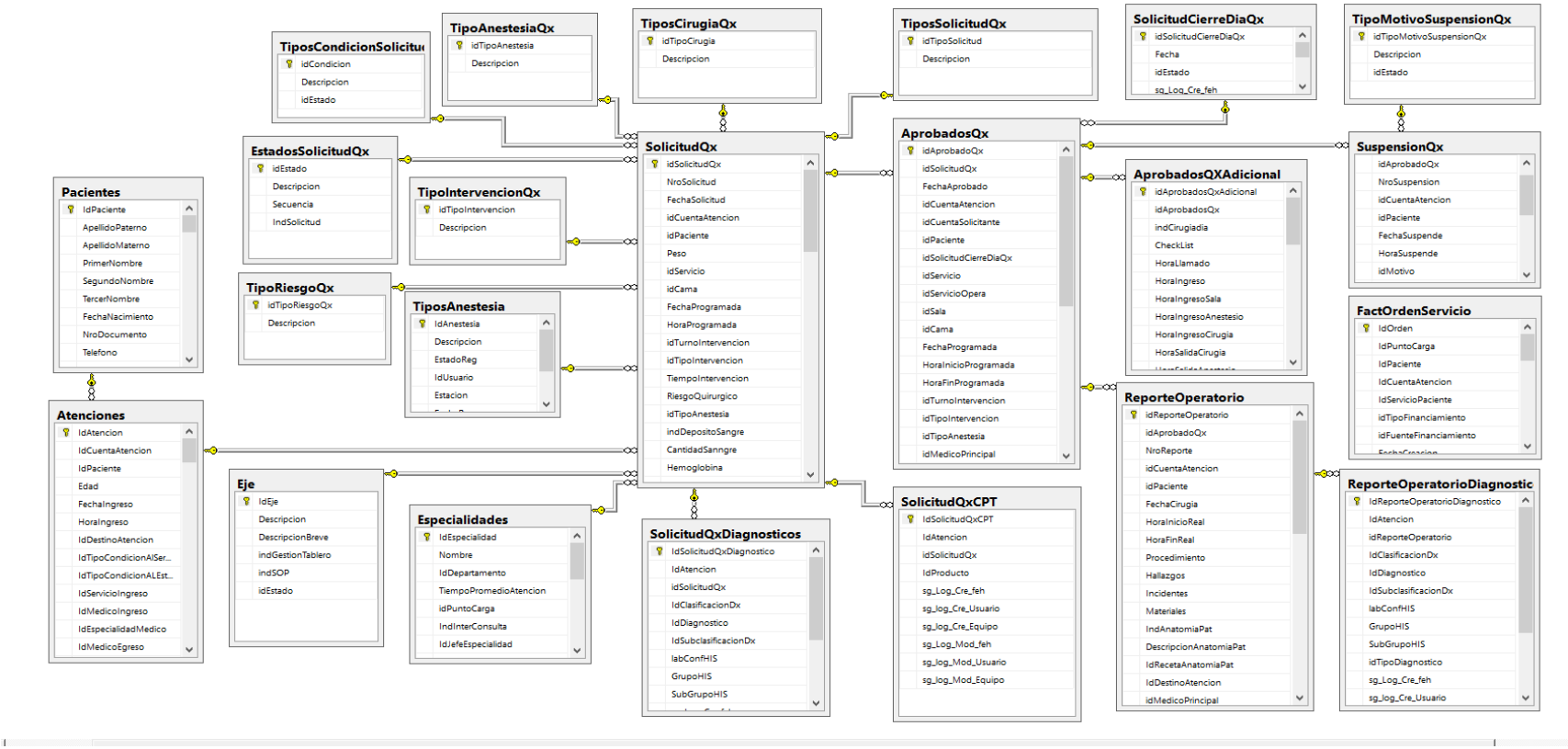


Fig. 41: Modelo de base de datos

También en esta fase se aplicaron los requerimientos mediante los casos de uso, teniendo en cuenta el marco de trabajo recopilado en **la Fase 1: análisis del sistema**, donde se establece los estándares, nomenclaturas de las clases, métodos, funciones, estructuras en las interfaces para el desarrollo. A continuación, se visualiza la implementación de los casos de uso identificados en la fase de diseño.

3.1.4.1 Implementación CU01 Registro de solicitud de sala de operaciones

Las clases Data Object se generan en la capa SIGHComun del Sistema SisGalenPlus, tienen como prefijo: "DO", estas clases mapean todos los campos de las tablas de base de datos, a continuación, se detallan las clases que pertenecen al **CU01**:

DOSolicitudQx: clase que contiene todos los campos de la tabla SolicitudQx de la base de datos, como se muestra en la Tabla XXI.

Tabla XXI: Clase Data Object DoSolicitudQx

Clase: DOSolicitudQx.cls

```
'----------------------------------------------------------
'Solicitudes de Centro Quirurgico
'----------------------------------------------------------

Option Explicit
Private ml_idUsuarioAuditoria As Long
Private ml_idSolicitudQx As Long
Private md_FechaSolicitud As Date
Private ml_idCuentaAtencion As Long
Private ml_idPaciente As Long
Private mc_Peso As Currency
Private ml_idServicio As Long
Private ml_idCama As Long
Private md_FechaProgramada As Date
Private ms_HoraProgramada As String
Private ml_idTipoIntervencion As Long
Private mc_TiempoIntervencion As Currency
Private ml_RiesgoQuirurgico As Long
Private ml_idTipoAnestesia As Long
Private ms_GrupoSanguineo As String
Private ms_FactorSanguineo As String
Private ml_idMedicoPrincipal As Long

Private ml_idMedicoAyudante1 As Long
Private ml_idMedicoAyudante2 As Long
Private ms_Equipo As String
Private mb_IndArcoEnC As Boolean
Private ml_idMedicoSolicitante As Long
Private ml_idTipoCirugia As Long
Private ml_idTipoParentesco As Long
Private ml_IdTipoDocumentoTutor As Long
Private ms_NroDocumentoTutor As String
Private ms_ApellidoPaternoTutor As String
Private ms_ApellidoMaternoTutor As String
Private ms_NombresTutor As String
Private ml_IdEspecialidadOpera As Long
Private ms_ResumenCIE As String
Private ms_ResumenCPT As String
Private ml_idEstado As Long
Private mb_IndReintervencion As Boolean
Private ms_MotivoReintervencion As String
Private ml_IdUsuario As Long
Private md_FechaCreacion As Date
Private ml_IdUsuarioModificacion As Long
Private md_FechaModificacion As Date

Property Let IdUsuarioAuditoria(lValue As Long)
   ml_idUsuarioAuditoria = lValue
End Property
Property Get IdUsuarioAuditoria() As Long
   IdUsuarioAuditoria = ml_idUsuarioAuditoria
End Property
Property Let idSolicitudQx(lValue As Long)
   ml_idSolicitudQx = lValue
End Property
Property Get idSolicitudQx() As Long
   idSolicitudQx = ml_idSolicitudQx
End Property
Property Let FechaSolicitud(dValue As Date)
   md_FechaSolicitud = dValue
End Property
Property Get FechaSolicitud() As Date
   FechaSolicitud = md_FechaSolicitud
End Property

Property Let idCuentaAtencion(lValue As Long)
   ml_idCuentaAtencion = lValue
End Property
Property Get idCuentaAtencion() As Long
   idCuentaAtencion = ml_idCuentaAtencion
End Property
Property Let idPaciente(lValue As Long)
   ml_idPaciente = lValue
End Property
Property Get idPaciente() As Long
   idPaciente = ml_idPaciente
End Property
Property Let Peso(cValue As Currency)
   mc_Peso = cValue
End Property
Property Get Peso() As Currency
   Peso = mc_Peso
End Property
Property Let idServicio(lValue As Long)
   ml_idServicio = lValue
End Property

Property Get idServicio() As Long
   idServicio = ml_idServicio
End Property
Property Let idCama(lValue As Long)
   ml_idCama = lValue
End Property
Property Get idCama() As Long
   idCama = ml_idCama
End Property
Property Let FechaProgramada(dValue As Date)
   md_FechaProgramada = dValue
End Property
Property Get FechaProgramada() As Date
   FechaProgramada = md_FechaProgramada
End Property
Property Let HoraProgramada(sValue As String)
   ms_HoraProgramada = sValue
End Property
Property Get HoraProgramada() As String
   HoraProgramada = ms_HoraProgramada
End Property

Property Get idTipoAnestesia() As Long
   idTipoAnestesia = ml_idTipoAnestesia
End Property
Property Let GrupoSanguineo(sValue As String)
   ms_GrupoSanguineo = sValue
End Property
Property Get GrupoSanguineo() As String
   GrupoSanguineo = ms_GrupoSanguineo
End Property
Property Let FactorSanguineo(sValue As String)
   ms_FactorSanguineo = sValue
End Property
Property Get FactorSanguineo() As String
   FactorSanguineo = ms_FactorSanguineo
End Property
Property Let idMedicoPrincipal(lValue As Long)
   ml_idMedicoPrincipal = lValue
End Property
Property Get idMedicoPrincipal() As Long
   idMedicoPrincipal = ml_idMedicoPrincipal
End Property
Property Let idMedicoAyudante1(lValue As Long)
   ml_idMedicoAyudante1 = lValue
End Property
Property Get idMedicoAyudante1() As Long
   idMedicoAyudante1 = ml_idMedicoAyudante1
End Property

Property Let idTipoIntervencion(lValue As Long)
   ml_idTipoIntervencion = lValue
End Property
Property Get idTipoIntervencion() As Long
   idTipoIntervencion = ml_idTipoIntervencion
End Property
Property Let TiempoIntervencion(cValue As Currency)
   mc_TiempoIntervencion = cValue
End Property
Property Get TiempoIntervencion() As Currency
   TiempoIntervencion = mc_TiempoIntervencion
End Property
Property Let RiesgoQuirurgico(lValue As Long)
   ml_RiesgoQuirurgico = lValue
End Property
Property Get RiesgoQuirurgico() As Long
   RiesgoQuirurgico = ml_RiesgoQuirurgico
End Property
Property Let idTipoAnestesia(lValue As Long)
   ml_idTipoAnestesia = lValue
End Property
```

DOSolicitudQxCPT: clase que contiene todos los campos de la tabla SolicitudQxCPT de la base de datos para el registro de los procedimientos quirúrgicos relacionados a la solicitud de sala de operaciones, como se observa en la Tabla XXII.

Tabla XXII: Clase Data Object DoSolicitudQxCPT

Clase: DOSolicitudQxCPT.cls

```
'--------------------------------------------------------------------------------
'      Programa:  Clase para capa de estructura de la tabla
'      procedimientos cirugia de solicitudes Qx
'--------------------------------------------------------------------------------
Option Explicit

Dim ml_Auditoria As Long
Dim ml_idProducto As Long
Dim ml_IdSolicitudQxCPT As Long
Dim ml_IdAtencion As Long
Dim ml_idSolicitudQx  As Long
Property Let IdUsuarioAuditoria(lValue As Long)
  ml_Auditoria = lValue
End Property
Property Get IdUsuarioAuditoria() As Long
  IdUsuarioAuditoria = ml_Auditoria
End Property
Property Let idProducto(lValue As Long)
  ml_idProducto = lValue
End Property
Property Get idProducto() As Long
  idProducto = ml_idProducto
End Property

Property Let idProducto(lValue As Long)
  ml_idProducto = lValue
End Property
Property Get idProducto() As Long
  idProducto = ml_idProducto
End Property
Property Let IdSolicitudQxCPT(lValue As Long)
  ml_IdSolicitudQxCPT = lValue
End Property
Property Get IdSolicitudQxCPT() As Long
  IdSolicitudQxCPT = ml_IdSolicitudQxCPT
End Property
Property Let IdAtencion(lValue As Long)
  ml_IdAtencion = lValue
End Property
Property Get IdAtencion() As Long
  IdAtencion = ml_IdAtencion
End Property
Property Let idSolicitudQx(lValue As Long)
  ml_idSolicitudQx = lValue
End Property
Property Get idSolicitudQx() As Long
  idSolicitudQx = ml_idSolicitudQx
End Property
```

DOSolicitudQxDiagnostico: clase que contiene todos los campos de la tabla SolicitudQxDiagnosticos de la base de datos para el registro de los diagnósticos CIE10 a la solicitud de sala de operaciones, como se observa en la Tabla XXIII.

Tabla XXIII: Clase Data Object DoSolicitudQxDiagnostico

Clase: DOSolicitudQxDiagnostico.cls

```
'--------------------------------------------------------------------------------
'      Programa:  Clase para capa de estructura de la
'      tabla diagnosticos de solicitudes Qx
'--------------------------------------------------------------------------------
Option Explicit

Dim ml_Auditoria As Long
Dim ml_IdSubClasificacionDX As Long
Dim ml_IdClasificacionDx As Long
Dim ml_IdDiagnostico As Long
Dim ml_IdSolicitudQxDx As Long
Dim ml_IdAtencion As Long
Dim ms_LabConfHIS As String
Dim ml_GrupoHIS As Long
Dim ml_SubGrupoHIS As Long
Dim ml_idSolicitudQx  As Long

Property Get GrupoHIS() As Long
  GrupoHIS = ml_GrupoHIS
End Property
Property Let GrupoHIS(lValue As Long)
  ml_GrupoHIS = lValue
End Property
Property Get SubGrupoHIS() As Long
  SubGrupoHIS = ml_SubGrupoHIS
End Property

Property Get IdUsuarioAuditoria() As Long
  IdUsuarioAuditoria = ml_Auditoria
End Property
Property Let IdSubClasificacionDX(lValue As Long)
  ml_IdSubClasificacionDX = lValue
End Property
Property Get IdSubClasificacionDX() As Long
  IdSubClasificacionDX = ml_IdSubClasificacionDX
End Property
Property Let IdClasificacionDx(lValue As Long)
  ml_IdClasificacionDx = lValue
End Property
Property Get IdClasificacionDx() As Long
  IdClasificacionDx = ml_IdClasificacionDx
End Property
```

Clase: DOSolicitudQxDiagnostico.cls

```
Property Let SubGrupoHIS(lValue As Long)
  ml_SubGrupoHIS = lValue
End Property
Property Get labConfHIS() As String
  labConfHIS = ms_LabConfHIS
End Property
Property Let labConfHIS(sValue As String)
  ms_LabConfHIS = sValue
End Property
Property Let IdUsuarioAuditoria(lValue As Long)
  ml_Auditoria = lValue
End Property
```

```
Property Let IdDiagnostico(lValue As Long)
  ml_IdDiagnostico = lValue
End Property
Property Get IdDiagnostico() As Long
  IdDiagnostico = ml_IdDiagnostico
End Property
Property Let IdSolicitudQxDiagnostico(lValue As Long)
  ml_IdSolicitudQxDx = lValue
End Property
Property Get IdSolicitudQxDiagnostico() As Long
  IdSolicitudQxDiagnostico = ml_IdSolicitudQxDx
End Property
Property Let IdAtencion(lValue As Long)
  ml_IdAtencion = lValue
End Property
Property Get IdAtencion() As Long
  IdAtencion = ml_IdAtencion
End Property
Property Let idSolicitudQx(lValue As Long)
  ml_idSolicitudQx = lValue
End Property
Property Get idSolicitudQx() As Long
  idSolicitudQx = ml_idSolicitudQx
End Property
```

Las clases ADO se generan en la capa SIGHDatos del sistema SisGalenPlus, estas clases contienen los métodos que se conectan con la base de datos. A continuación se detallan las clases que pertenecen al **CU01**:

SolicitudQx: clase que contiene todos los métodos para insertar, modificar, eliminar y seleccionar los datos de SolicitudQx de la base de datos, como se muestra en la Tabla XXIV.

Tabla XXIV: Clase ADO SolicitudQx

Clase: DOSolicitudQxCPT.cls

Variables y propiedades de la clase SolicitudQx

```
'-------------------------------------------------------------------------
'      Programa:  Clase para capa de Mantenimiento de la tabla SolicitudQx
'-------------------------------------------------------------------------
Option Explicit
Dim mo_Conexion As ADODB.Connection
Dim ms_MensajeError As String
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Set Conexion(oValue As ADODB.Connection)
  Set mo_Conexion = oValue
End Property
Property Get Conexion() As ADODB.Connection
  Set Conexion = mo_Conexion
End Property
```

Método para insertar un registro en la base de datos

```
Function Insertar(ByVal oTabla As doSolicitudQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter
  Insertar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
      .CommandText = "usp_insert_SolicitudQxAgregar_12082024"
Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamOutput, 0): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@NroSolicitud", adInteger, adParamOutput, 0): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@FechaSolicitud ", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaSolicitud = 0, Null, oTabla.FechaSolicitud)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idCuentaAtencion", adInteger, adParamInput, 0, IIf(oTabla.idCuentaAtencion = 0, Null, oTabla.idCuentaAtencion)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idPaciente", adInteger, adParamInput, 0, IIf(oTabla.IDPaciente = 0, Null, oTabla.IDPaciente)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@Peso", adDouble, adParamInput, 0, oTabla.Peso): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idServicio", adInteger, adParamInput, 0, IIf(oTabla.idServicio = 0, Null, oTabla.idServicio)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idCama", adInteger, adParamInput, 0, IIf(oTabla.idCama = 0, Null, oTabla.idCama)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@FechaProgramada", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaProgramada = 0, Null, oTabla.FechaProgramada)): .Parameters.Append oParameter
```

Clase: DOSolicitudQxCPT.cls

```
Set oParameter = .CreateParameter("@HoraProgramada", adVarChar, adParamInput, 5, oTabla.HoraProgramada): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idTurnoIntervencion", adInteger, adParamInput, 0, IIf(oTabla.idTurnoIntervencion = 0, Null, oTabla.idTurnoIntervencion)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idTipoIntervencion", adInteger, adParamInput, 0, IIf(oTabla.idTipoIntervencion = 0, Null, oTabla.idTipoIntervencion)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@TiempoIntervencion", adDouble, adParamInput, 0, oTabla.tiempoIntervencion): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@RiesgoQuirurgico", adInteger, adParamInput, 0, IIf(oTabla.RiesgoQuirurgico = 0, Null, oTabla.RiesgoQuirurgico)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idTipoAnestesia", adInteger, adParamInput, 0, IIf(oTabla.idTipoAnestesia = 0, Null, oTabla.idTipoAnestesia)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@indDepositoSangre", adBoolean, adParamInput, 0, IIf(oTabla.indDepositoSangre = True, 1, 0)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@CantidadSanngre", adDouble, adParamInput, , oTabla.CantidadSanngre): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@Hemoglobina", adVarChar, adParamInput, 10, oTabla.Hemoglobina): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@GrupoSanguineo", adVarChar, adParamInput, 5, oTabla.GrupoSanguineo): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@FactorSanguineo", adVarChar, adParamInput, 10, oTabla.FactorSanguineo): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idMedicoPrincipal", adInteger, adParamInput, 0, IIf(oTabla.idMedicoPrincipal = 0, Null, oTabla.idMedicoPrincipal)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idMedicoAyudante1", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante1 = 0, Null, oTabla.idMedicoAyudante1)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idMedicoAyudante2", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante2 = 0, Null, oTabla.idMedicoAyudante2)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@Instrumental ", adVarChar, adParamInput, 200, IIf(oTabla.Instrumental = "", Null, oTabla.Instrumental)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@ResumenCIE", adVarChar, adParamInput, 800, IIf(oTabla.ResumenCIE = "", Null, oTabla.ResumenCIE)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@ResumenCPT", adVarChar, adParamInput, 800, IIf(oTabla.ResumenCPT = "", Null, oTabla.ResumenCPT)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idMedicoSolicitante", adInteger, adParamInput, 0, IIf(oTabla.idMedicoSolicitante = 0, Null, oTabla.idMedicoSolicitante)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IdEstado", adInteger, adParamInput, 0, IIf(oTabla.IdEstado = 0, Null, oTabla.IdEstado)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idServicioOpera", adInteger, adParamInput, 0, IIf(oTabla.idServicioOpera = 0, Null, oTabla.idServicioOpera)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idSala", adInteger, adParamInput, 0, IIf(oTabla.idSala = 0, Null, oTabla.idSala)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idTipoSolicitud", adInteger, adParamInput, 0, IIf(oTabla.idTipoSolicitud = 0, Null, oTabla.idTipoSolicitud)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idTipoCirugia", adInteger, adParamInput, 0, IIf(oTabla.idTipoCirugia = 0, Null, oTabla.idTipoCirugia)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@DescripcionCirugia", adVarChar, adParamInput, 400, oTabla.DescripcionCirugia): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@RiesgoDerivado", adVarChar, adParamInput, 200, oTabla.RiesgoDerivado): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idTipoDocumentoTutor", adInteger, adParamInput, 0, IIf(oTabla.IdTipoDocumentoTutor = 0, Null, oTabla.IdTipoDocumentoTutor)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idTipoParentesco", adInteger, adParamInput, 0, IIf(oTabla.idTipoParentesco = 0, Null, oTabla.idTipoParentesco)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@NroDocumentoTutor", adVarChar, adParamInput, 20, oTabla.NroDocumentoTutor): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@apellidoPaternoTutor", adVarChar, adParamInput, 50, oTabla.ApellidoPaternoTutor): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@apellidoMaternoTutor", adVarChar, adParamInput, 50, oTabla.ApellidoMaternoTutor): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@nombresTutor", adVarChar, adParamInput, 50, oTabla.NombresTutor): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idEspecialidadOpera", adInteger, adParamInput, 0, IIf(oTabla.idEspecialidadOpera = 0, Null, oTabla.idEspecialidadOpera)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@idCondicion", adInteger, adParamInput, 0, IIf(oTabla.idCondicion = 0, Null, oTabla.idCondicion)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IndClEvaluacionPediatrica", adBoolean, adParamInput, 0, IIf(oTabla.IndClEvaluacionPediatrica = True, 1, 0)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IndClConcentimiento", adBoolean, adParamInput, 0, IIf(oTabla.IndClConcentimiento = True, 1, 0)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IndClBancoSangre", adBoolean, adParamInput, 0, IIf(oTabla.IndClBancoSangre = True, 1, 0)): .Parameters.Append oParameter


Set oParameter = .CreateParameter("@IndClCompatibilidadSangre", adBoolean, adParamInput, 0, IIf(oTabla.IndClCompatibilidadSangre = True, 1, 0)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IndClRequiereInsumo", adBoolean, adParamInput, 0, IIf(oTabla.IndClRequiereInsumo = True, 1, 0)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IndClRequiereCamaUCI", adBoolean, adParamInput, 0, IIf(oTabla.IndClRequiereCamaUCI = True, 1, 0)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IndClResultadoIsopado", adInteger, adParamInput, 0, IIf(oTabla.IndClResultadoIsopado = 0, Null, oTabla.IndClResultadoIsopado)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IndReintervencion", adBoolean, adParamInput, 0, IIf(oTabla.IndReintervencion = True, 1, 0)): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@MotivoReintervencion", adVarChar, adParamInput, 200, oTabla.MotivoReintervencion): .Parameters.Append oParameter
Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParameter

    .Execute
      oTabla.IdSolicitudQx = .Parameters("@IdSolicitudQx")
      oTabla.NroSolicitud = .Parameters("@NroSolicitud")
  End With

  Insertar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface de acceso a datos"
Exit Function
End Function
```

Modelo para modificar un registro en la base de datos

```
Function Modificar(ByVal oTabla As doSolicitudQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter
   Modificar = False
   With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
        .CommandText = "usp_update_SolicitudQxModificar_12082024"
     Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, oTabla.IdSolicitudQx): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@NroSolicitud", adInteger, adParamInput, 0, oTabla.NroSolicitud): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@FechaSolicitud ", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaSolicitud = 0, Null, oTabla.FechaSolicitud)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idCuentaAtencion", adInteger, adParamInput, 0, IIf(oTabla.idCuentaAtencion = 0, Null, oTabla.idCuentaAtencion)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idPaciente", adInteger, adParamInput, 0, IIf(oTabla.IDPaciente = 0, Null, oTabla.IDPaciente)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@Peso", adDouble, adParamInput, 0, oTabla.Peso): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idServicio", adInteger, adParamInput, 0, IIf(oTabla.idServicio = 0, Null, oTabla.idServicio)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idCama", adInteger, adParamInput, 0, IIf(oTabla.idCama = 0, Null, oTabla.idCama)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@FechaProgramada", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaProgramada = 0, Null, oTabla.FechaProgramada)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@HoraProgramada", adVarChar, adParamInput, 5, oTabla.HoraProgramada): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idTurnoIntervencion", adInteger, adParamInput, 0, IIf(oTabla.idTurnoIntervencion = 0, Null, oTabla.idTurnoIntervencion)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idTipoIntervencion", adInteger, adParamInput, 0, IIf(oTabla.idTipoIntervencion = 0, Null, oTabla.idTipoIntervencion)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@TiempoIntervencion", adDouble, adParamInput, 0, oTabla.tiempoIntervencion): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@RiesgoQuirurgico", adInteger, adParamInput, 0, IIf(oTabla.RiesgoQuirurgico = 0, Null, oTabla.RiesgoQuirurgico)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idTipoAnestesia", adInteger, adParamInput, 0, IIf(oTabla.idTipoAnestesia = 0, Null, oTabla.idTipoAnestesia)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@indDepositoSangre", adBoolean, adParamInput, 0, IIf(oTabla.indDepositoSangre = True, 1, 0)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@CantidadSanngre", adDouble, adParamInput, , oTabla.CantidadSanngre): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@Hemoglobina", adVarChar, adParamInput, 10, oTabla.Hemoglobina): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@GrupoSanguineo", adVarChar, adParamInput, 5, oTabla.GrupoSanguineo): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@FactorSanguineo", adVarChar, adParamInput, 10, oTabla.FactorSanguineo): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idMedicoPrincipal", adInteger, adParamInput, 0, IIf(oTabla.idMedicoPrincipal = 0, Null, oTabla.idMedicoPrincipal)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idMedicoAyudante1", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante1 = 0, Null, oTabla.idMedicoAyudante1)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@idMedicoAyudante2", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante2 = 0, Null, oTabla.idMedicoAyudante2)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@Instrumental ", adVarChar, adParamInput, 200, IIf(oTabla.Instrumental = "", Null, oTabla.Instrumental)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@ResumenCIE", adVarChar, adParamInput, 800, IIf(oTabla.ResumenCIE = "", Null, oTabla.ResumenCIE)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@ResumenCPT", adVarChar, adParamInput, 800, IIf(oTabla.ResumenCPT = "", Null, oTabla.ResumenCPT)): .Parameters.Append oParameter
```

Clase: DOSolicitudQxCPT.cls

```
    Set oParameter = .CreateParameter("@idMedicoSolicitante", adInteger, adParamInput, 0, IIf(oTabla.idMedicoSolicitante = 0, Null, oTabla.idMedicoSolicitante)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IdEstado", adInteger, adParamInput, 0, oTabla.IdEstado): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idServicioOpera", adInteger, adParamInput, 0, IIf(oTabla.idServicioOpera = 0, Null, oTabla.idServicioOpera)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idSala", adInteger, adParamInput, 0, IIf(oTabla.idSala = 0, Null, oTabla.idSala)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idTipoSolicitud", adInteger, adParamInput, 0, IIf(oTabla.idTipoSolicitud = 0, Null, oTabla.idTipoSolicitud)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idTipoCirugia", adInteger, adParamInput, 0, IIf(oTabla.idTipoCirugia = 0, Null, oTabla.idTipoCirugia)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@DescripcionCirugia", adVarChar, adParamInput, 400, oTabla.DescripcionCirugia): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@RiesgoDerivado", adVarChar, adParamInput, 200, oTabla.RiesgoDerivado): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IdTipoDocumentoTutor", adInteger, adParamInput, 0, IIf(oTabla.IdTipoDocumentoTutor = 0, Null, oTabla.IdTipoDocumentoTutor)): .Parameters.Append oParamete
    Set oParameter = .CreateParameter("@idTipoParentesco", adInteger, adParamInput, 0, IIf(oTabla.idTipoParentesco = 0, Null, oTabla.idTipoParentesco)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@NroDocumentoTutor", adVarChar, adParamInput, 20, oTabla.NroDocumentoTutor): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@apellidoPaternoTutor", adVarChar, adParamInput, 50, oTabla.ApellidoPaternoTutor): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@apellidoMaternoTutor", adVarChar, adParamInput, 50, oTabla.ApellidoMaternoTutor): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@nombresTutor", adVarChar, adParamInput, 50, oTabla.NombresTutor): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idEspecialidadOpera", adInteger, adParamInput, 0, IIf(oTabla.idEspecialidadOpera = 0, Null, oTabla.idEspecialidadOpera)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idCondicion", adInteger, adParamInput, 0, IIf(oTabla.idCondicion = 0, Null, oTabla.idCondicion)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClEvaluacionPediatrica", adBoolean, adParamInput, 0, IIf(oTabla.IndClEvaluacionPediatrica = True, 1, 0)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClConcentimiento", adBoolean, adParamInput, 0, IIf(oTabla.IndClConcentimiento = True, 1, 0)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClBancoSangre", adBoolean, adParamInput, 0, IIf(oTabla.IndClBancoSangre = True, 1, 0)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClCompatibilidadSangre", adBoolean, adParamInput, 0, IIf(oTabla.IndClCompatibilidadSangre = True, 1, 0)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClRequiereInsumo", adBoolean, adParamInput, 0, IIf(oTabla.IndClRequiereInsumo = True, 1, 0)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClRequiereCamaUCI", adBoolean, adParamInput, 0, IIf(oTabla.IndClRequiereCamaUCI = True, 1, 0)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClResultadoIsopado", adInteger, adParamInput, 0, IIf(oTabla.IndClResultadoIsopado = 0, Null, oTabla.IndClResultadoIsopado)): .Parameters.Append oParame
    Set oParameter = .CreateParameter("@IndReintervencion", adBoolean, adParamInput, 0, IIf(oTabla.IndReintervencion = True, 1, 0)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@MotivoReintervencion", adVarChar, adParamInput, 200, oTabla.MotivoReintervencion): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idMedicoResidente", adInteger, adParamInput, 0, IIf(oTabla.idMedicoResidente = 0, Null, oTabla.idMedicoResidente)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idPerfusionista1", adInteger, adParamInput, 0, IIf(oTabla.idPerfusionista1 = 0, Null, oTabla.idPerfusionista1)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idPerfusionista2", adInteger, adParamInput, 0, IIf(oTabla.idPerfusionista2 = 0, Null, oTabla.idPerfusionista2)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IndClEvaluacionPreAnestesica", adBoolean, adParamInput, 0, IIf(oTabla.IndClEvaluacionPreAnestesica = True, 1, 0)): .Parameters.Append oParameter

    .Execute
  End With

  Modificar = True


  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface de acceso a datos"
End Function
```

Método para Eliminar un registro en la base de datos

```
Function Eliminar(ByVal oTabla As doSolicitudQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Eliminar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_delete_SolicitudQxEliminar_03092018"
      Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQx = 0, Null, oTabla.IdSolicitudQx)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria)
    .Parameters.Append oParameter
    .Execute
  End With

  Eliminar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface de acceso a datos"
Exit Function
End Function
```

Método para seleccionar un registro en la base de datos

```
Function SeleccionarPorId(ByVal oTabla As doSolicitudQx) As Boolean
On Error GoTo ManejadorDeError
Dim oRecordset As New ADODB.Recordset
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  SeleccionarPorId = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_select_SolicitudQxSeleccionarPorId_12082024"
    Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, oTabla.IdSolicitudQx): .Parameters.Append oParameter
    Set oRecordset = .Execute
  End With


    oTabla.idTurnoIntervencion = IIf(IsNull(oRecordset!idTurnoIntervencion), 0, oRecordset!idTurnoIntervencion)
    oTabla.idTipoIntervencion = IIf(IsNull(oRecordset!idTipoIntervencion), 0, oRecordset!idTipoIntervencion)
    oTabla.tiempoIntervencion = IIf(IsNull(oRecordset!tiempoIntervencion), 0, oRecordset!tiempoIntervencion)
    oTabla.RiesgoQuirurgico = IIf(IsNull(oRecordset!RiesgoQuirurgico), 0, oRecordset!RiesgoQuirurgico)
    oTabla.idTipoAnestesia = IIf(IsNull(oRecordset!idTipoAnestesia), 0, oRecordset!idTipoAnestesia)
    oTabla.indDepositoSangre = IIf(IsNull(oRecordset!indDepositoSangre), False, IIf(oRecordset!indDepositoSangre = 1, True, False))
    oTabla.CantidadSanngre = IIf(IsNull(oRecordset!CantidadSanngre), 0, oRecordset!CantidadSanngre)

  If Not (oRecordset.EOF And oRecordset.BOF) Then
    SeleccionarPorId = True
    oTabla.IdSolicitudQx = IIf(IsNull(oRecordset!IdSolicitudQx), 0, oRecordset!IdSolicitudQx)
    oTabla.idCuentaAtencion = IIf(IsNull(oRecordset!idCuentaAtencion), 0, oRecordset!idCuentaAtencion)
    oTabla.FechaSolicitud = IIf(IsNull(oRecordset!FechaSolicitud), 0, oRecordset!FechaSolicitud)
    oTabla.NroSolicitud = IIf(IsNull(oRecordset!NroSolicitud), 0, oRecordset!NroSolicitud)
    oTabla.IDPaciente = IIf(IsNull(oRecordset!IDPaciente), 0, oRecordset!IDPaciente)
    oTabla.Peso = IIf(IsNull(oRecordset!Peso), 0, oRecordset!Peso)
    oTabla.idServicio = IIf(IsNull(oRecordset!idServicio), 0, oRecordset!idServicio)
    oTabla.idCama = IIf(IsNull(oRecordset!idCama), 0, oRecordset!idCama)
    oTabla.FechaProgramada = IIf(IsNull(oRecordset!FechaProgramada), 0, oRecordset!FechaProgramada)
    oTabla.HoraProgramada = IIf(IsNull(oRecordset!HoraProgramada), "", oRecordset!HoraProgramada)
```

Clase: DOSolicitudQxCPT.cls

```
    oTabla.Hemoglobina = IIf(IsNull(oRecordset!Hemoglobina), "", oRecordset!Hemoglobina)
    oTabla.GrupoSanguineo = IIf(IsNull(oRecordset!GrupoSanguineo), "", oRecordset!GrupoSanguineo)
    oTabla.FactorSanguineo = IIf(IsNull(oRecordset!FactorSanguineo), "", oRecordset!FactorSanguineo)
    oTabla.idMedicoPrincipal = IIf(IsNull(oRecordset!idMedicoPrincipal), 0, oRecordset!idMedicoPrincipal)
    oTabla.idMedicoAyudante1 = IIf(IsNull(oRecordset!idMedicoAyudante1), 0, oRecordset!idMedicoAyudante1)
    oTabla.idMedicoAyudante2 = IIf(IsNull(oRecordset!idMedicoAyudante2), 0, oRecordset!idMedicoAyudante2)
    oTabla.idMedicoSolicitante = IIf(IsNull(oRecordset!idMedicoSolicitante), 0, oRecordset!idMedicoSolicitante)
    oTabla.Instrumental = IIf(IsNull(oRecordset!Instrumental), "", oRecordset!Instrumental)
    oTabla.ResumenCIE = IIf(IsNull(oRecordset!ResumenCIE), "", oRecordset!ResumenCIE)
    oTabla.ResumenCPT = IIf(IsNull(oRecordset!ResumenCPT), "", oRecordset!ResumenCPT)
    oTabla.IdEstado = IIf(IsNull(oRecordset!IdEstado), 1, oRecordset!IdEstado)
    oTabla.idServicioOpera = IIf(IsNull(oRecordset!idServicioOpera), 0, oRecordset!idServicioOpera)
    oTabla.idSala = IIf(IsNull(oRecordset!idSala), 0, oRecordset!idSala)
    oTabla.idTipoSolicitud = IIf(IsNull(oRecordset!idTipoSolicitud), 0, oRecordset!idTipoSolicitud)
    oTabla.idTipoCirugia = IIf(IsNull(oRecordset!idTipoCirugia), 0, oRecordset!idTipoCirugia)
    oTabla.DescripcionCirugia = IIf(IsNull(oRecordset!DescripcionCirugia), "", oRecordset!DescripcionCirugia)
    oTabla.RiesgoDerivado = IIf(IsNull(oRecordset!RiesgoDerivado), "", oRecordset!RiesgoDerivado)
    oTabla.IdTipoDocumentoTutor = IIf(IsNull(oRecordset!IdTipoDocumentoTutor), 0, oRecordset!IdTipoDocumentoTutor)
    oTabla.idTipoParentesco = IIf(IsNull(oRecordset!idTipoParentesco), 0, oRecordset!idTipoParentesco)
    oTabla.NroDocumentoTutor = IIf(IsNull(oRecordset!NroDocumentoTutor), "", oRecordset!NroDocumentoTutor)
    oTabla.ApellidoPaternoTutor = IIf(IsNull(oRecordset!ApellidoPaternoTutor), "", oRecordset!ApellidoPaternoTutor)
    oTabla.ApellidoMaternoTutor = IIf(IsNull(oRecordset!ApellidoMaternoTutor), "", oRecordset!ApellidoMaternoTutor)
    oTabla.NombresTutor = IIf(IsNull(oRecordset!NombresTutor), "", oRecordset!NombresTutor)
    oTabla.idEspecialidadOpera = IIf(IsNull(oRecordset!idEspecialidadOpera), 0, oRecordset!idEspecialidadOpera)
    oTabla.idCondicion = IIf(IsNull(oRecordset!idCondicion), 0, oRecordset!idCondicion)
    oTabla.IndClEvaluacionPediatrica = IIf(IsNull(oRecordset!IndClEvaluacionPediatrica), False, IIf(oRecordset!IndClEvaluacionPediatrica = 1, True, False))
    oTabla.IndClConcentimiento = IIf(IsNull(oRecordset!IndClConcentimiento), False, IIf(oRecordset!IndClConcentimiento = 1, True, False))
    oTabla.IndClBancoSangre = IIf(IsNull(oRecordset!IndClBancoSangre), False, IIf(oRecordset!IndClBancoSangre = 1, True, False))
    oTabla.IndClCompatibilidadSangre = IIf(IsNull(oRecordset!IndClCompatibilidadSangre), False, IIf(oRecordset!IndClCompatibilidadSangre = 1, True, False))
    oTabla.IndClRequiereInsumo = IIf(IsNull(oRecordset!IndClRequiereInsumo), False, IIf(oRecordset!IndClRequiereInsumo = 1, True, False))
    oTabla.IndClRequiereCamaUCI = IIf(IsNull(oRecordset!IndClRequiereCamaUCI), False, IIf(oRecordset!IndClRequiereCamaUCI = 1, True, False))
    oTabla.IndClResultadoIsopado = IIf(IsNull(oRecordset!IndClResultadoIsopado), 0, oRecordset!IndClResultadoIsopado)
    oTabla.IndReintervencion = IIf(IsNull(oRecordset!IndReintervencion), 0, IIf(oRecordset!IndReintervencion = 1, 1, 0))
    oTabla.MotivoReintervencion = IIf(IsNull(oRecordset!MotivoReintervencion), "", oRecordset!MotivoReintervencion)
    oTabla.idMedicoResidente = IIf(IsNull(oRecordset!idMedicoResidente), 0, oRecordset!idMedicoResidente)


    oTabla.idPerfusionista1 = IIf(IsNull(oRecordset!idPerfusionista1), 0, oRecordset!idPerfusionista1)
    oTabla.idPerfusionista2 = IIf(IsNull(oRecordset!idPerfusionista2), 0, oRecordset!idPerfusionista2)
    oTabla.IndClEvaluacionPreAnestesica = IIf(IsNull(oRecordset!IndClEvaluacionPreAnestesica), False, IIf(oRecordset!IndClEvaluacionPreAnestesica = 1, True, False))

  Else
    SeleccionarPorId = False
  End If

  oRecordset.Close
  ms_MensajeError = ""
Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface de acceso a datos"
Exit Function
End Function
```

SolicitudQxCPT: clase que contiene todos los métodos para insertar, modificar, eliminar y seleccionar los datos de SolicitudQxCPT de la base de datos, como se muestra en la Tabla XXV.

Tabla XXV: Clase ADO SolicitudQxCPT

Clase ADO SolicitudQxCPT

Variables y propiedades de la clase SolicitudQxCPT

```
'-------------------------------------------------------------------------
'      Programa:  Clase para capa de Mantenimiento de la tabla SolicitudQxCPT
'-------------------------------------------------------------------------
Option Explicit
Dim mo_Conexion As ADODB.Connection
Dim ms_MensajeError As String
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Set Conexion(oValue As ADODB.Connection)
  Set mo_Conexion = oValue
End Property
Property Get Conexion() As ADODB.Connection
  Set Conexion = mo_Conexion
End Property
```

Método para insertar un registro en la base de datos de SolicitudQxCPT

```
Function Insertar(ByVal oTabla As DOSolicitudQxCPT) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Insertar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
     .CommandText = "usp_insert_SolicitudQxCPTAgregar_03092024"
     Set oParameter = .CreateParameter("@IdSolicitudQxCPT", adInteger, adParamOutput, 0): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdProducto", adInteger, adParamInput, 0, IIf(oTabla.idProducto = 0, Null, oTabla.idProducto)): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdAtencion", adInteger, adParamInput, 0, IIf(oTabla.IdAtencion = 0, Null, oTabla.IdAtencion)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQx = 0, Null, oTabla.IdSolicitudQx)): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParameter
    .Execute
      oTabla.IdSolicitudQxCPT = .Parameters("@IdSolicitudQxCPT")
  End With

  Insertar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface c
End Function
```

Método para modificar un registro en la base de datos de SolicitudQxCPT

Clase ADO SolicitudQxCPT

```
Function Modificar(ByVal oTabla As DOSolicitudQxCPT) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Modificar = False
  With oCommand
     .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_update_SolicitudQxCPTModificar_03092018"
       Set oParameter = .CreateParameter("@IdSolicitudQxCPT", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQxCPT = 0, Null, oTabla.IdSolicitudQxCPT)): .Parameters.Append oF
       Set oParameter = .CreateParameter("@IdProducto", adInteger, adParamInput, 0, IIf(oTabla.idProducto = 0, Null, oTabla.idProducto)): .Parameters.Append oParameter
        Set oParameter = .CreateParameter("@IdAtencion", adInteger, adParamInput, 0, IIf(oTabla.IdAtencion = 0, Null, oTabla.IdAtencion)): .Parameters.Append oParameter
       Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQx = 0, Null, oTabla.IdSolicitudQx)): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria)
    .Parameters.Append oParameter
    .Execute
  End With

  Modificar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
Exit Function
End Function
```

Método para seleccionar un registro en la base de datos de SolicitudQxCPT

```
Function SeleccionarPorId(ByVal oTabla As DOSolicitudQxCPT) As Boolean
On Error GoTo ManejadorDeError
Dim oRecordset As New ADODB.Recordset
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  SeleccionarPorId = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_select_SolicitudQxCPTSeleccionarPorId_03092024"
      Set oParameter = .CreateParameter("@IdSolicitudQxCPT", adInteger, adParamInput, 0, oTabla.IdSolicitudQxCPT): .Parameters.Append oParameter
    Set oRecordset = .Execute
  End With

  If Not (oRecordset.EOF And oRecordset.BOF) Then
     SeleccionarPorId = True
     oTabla.idProducto = IIf(IsNull(oRecordset!idProducto), 0, oRecordset!idProducto)
     oTabla.IdSolicitudQxCPT = IIf(IsNull(oRecordset!IdSolicitudQxCPT), 0, oRecordset!IdSolicitudQxCPT)
     oTabla.IdAtencion = IIf(IsNull(oRecordset!IdAtencion), 0, oRecordset!IdAtencion)
     oTabla.IdSolicitudQx = IIf(IsNull(oRecordset!IdSolicitudQx), 0, oRecordset!IdSolicitudQx)

  Else
     SeleccionarPorId = False
  End If

  oRecordset.Close
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
End Function
```

Método para actualizar una colección de registro en la base de datos de SolicitudQxCPT

```
Function ActualizarCPTSolicitudQx(oCPT As Collection, lIdSolicitudQx As Long) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter
Dim sSQL As String
Dim oDOSolicitudQxCPT As New DOSolicitudQxCPT

  ActualizarCPTSolicitudQx = False
  With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
     .CommandTimeout = 150
     .CommandText = "usp_delete_SolicitudQxCPTEliminarXIdSolicitudQx_03092018"
     Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, lIdSolicitudQx): .Parameters.Append oParameter
     .Execute
  End With
  If Not oCPT Is Nothing Then
      For Each oDOSolicitudQxCPT In oCPT
        oDOSolicitudQxCPT.IdSolicitudQx = lIdSolicitudQx
        If Not Insertar(oDOSolicitudQxCPT) Then
          Exit Function
        End If
      Next
  End If
  ActualizarCPTSolicitudQx = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  MsgBox Err.Description
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
End Function
```

Método para eliminar un registro en la base de datos de SolicitudQxCPT

```
Function Eliminar(ByVal oTabla As DOSolicitudQxCPT) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Eliminar = False
  With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
     .CommandText = "usp_delete_SolicitudQxCPTEliminar_03092024"
       Set oParameter = .CreateParameter("@IdSolicitudQxCPT", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQxCPT = 0, Null, oTabla.IdSolicitudQxCPT)): .Parameters.Append o
     Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria)
     .Parameters.Append oParameter
     .Execute
  End With

  Eliminar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
End Function
```

SolicitudQxDiagnostico: clase que contiene todos los métodos para insertar, modificar, eliminar y seleccionar los datos de SolicitudQxDiagnostico de la base de datos, como se observa Tabla XXVI.

Tabla XXVI: Clase ADO SolicitudQxDiagnostico

Variables y propiedades de clase SolicitudQxDiagnostico

```
'-------------------------------------------------------------------
'     Programa:  Clase para capa de Mantenimiento de la tabla SolicitudQxDiagnostico
'-------------------------------------------------------------------
Option Explicit
Dim mo_Conexion As ADODB.Connection
Dim ms_MensajeError As String
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Set Conexion(oValue As ADODB.Connection)
  Set mo_Conexion = oValue
End Property
Property Get Conexion() As ADODB.Connection
  Set Conexion = mo_Conexion
End Property
```

Método para insertar un registro en la base de datos de SolicitudQxDiagnostico

```
Function Insertar(ByVal oTabla As DOSolicitudQxDiagnosticos) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Insertar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
     .CommandText = "[usp_insert_SolicitudQxDiagnosticosAgregar_23112024]"
      Set oParameter = .CreateParameter("@IdSolicitudQxDiagnostico", adInteger, adParamOutput): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdDiagnostico", adInteger, adParamInput, 0, IIf(oTabla.IdDiagnostico = 0, Null, oTabla.IdDiagnostico)): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdSubclasificacionDx", adInteger, adParamInput, 0, IIf(oTabla.IdSubclasificacionDx = 0, Null, oTabla.IdSubclasificacionDx)): .Parameters.A
      Set oParameter = .CreateParameter("@IdClasificacionDx", adInteger, adParamInput, 0, IIf(oTabla.IdClasificacionDx = 0, Null, oTabla.IdClasificacionDx)): .Parameters.Append oPa
      Set oParameter = .CreateParameter("@IdAtencion", adInteger, adParamInput, 0, IIf(oTabla.IdAtencion = 0, Null, oTabla.IdAtencion)): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQx = 0, Null, oTabla.IdSolicitudQx)): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParameter
      .Execute
      oTabla.IdSolicitudQxDiagnostico = .Parameters("@IdSolicitudQxDiagnostico")
  End With

  Insertar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
End Function
```

Método para modificar un registro en la base de datos de SolicitudQxDiagnostico

```
Function Modificar(ByVal oTabla As DOSolicitudQxDiagnosticos) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Modificar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "[usp_update_SolicitudQxDiagnosticosModificar_23112024]"
       Set oParameter = .CreateParameter("@IdSolicitudQxDiagnostico", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQxDiagnostico = 0, Null, oTabla.IdSolicitudQxDiagnostico)): .
      Set oParameter = .CreateParameter("@IdDiagnostico", adInteger, adParamInput, 0, IIf(oTabla.IdDiagnostico = 0, Null, oTabla.IdDiagnostico)): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdSubclasificacionDx", adInteger, adParamInput, 0, IIf(oTabla.IdSubclasificacionDx = 0, Null, oTabla.IdSubclasificacionDx)): .Parameters.A
      Set oParameter = .CreateParameter("@IdClasificacionDx", adInteger, adParamInput, 0, IIf(oTabla.IdClasificacionDx = 0, Null, oTabla.IdClasificacionDx)): .Parameters.Append oPa
      Set oParameter = .CreateParameter("@IdAtencion", adInteger, adParamInput, 0, IIf(oTabla.IdAtencion = 0, Null, oTabla.IdAtencion)): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQx = 0, Null, oTabla.IdSolicitudQx)): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria)
    .Parameters.Append oParameter
    .Execute
  End With

  Modificar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
End Function
```

Método para eliminar un registro en la base de datos de SolicitudQxDiagnostico

```
Function Eliminar(ByVal oTabla As DOSolicitudQxDiagnosticos) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Eliminar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_delete_SolicitudQxDiagnosticosEliminar_03092024"
       Set oParameter = .CreateParameter("@IdSolicitudQxDiagnostico", adInteger, adParamInput, 0, IIf(oTabla.IdSolicitudQxDiagnostico = 0, Null, oTabla.IdSolicitudQxDiagnostico)): .I
    Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria)
    .Parameters.Append oParameter
    .Execute
  End With

  Eliminar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
End Function
```

Método para seleccionar un registro en la base de datos de SolicitudQxDiagnostico

```
Function SeleccionarPorId(ByVal oTabla As DOSolicitudQxDiagnosticos) As Boolean
On Error GoTo ManejadorDeError
Dim oRecordset As New ADODB.Recordset
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  SeleccionarPorId = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_select_SolicitudQxDiagnosticosSeleccionarPorId_03092024"
      Set oParameter = .CreateParameter("@IdSolicitudQxDiagnostico", adInteger, adParamInput, 0, oTabla.IdSolicitudQxDiagnostico): .Parameters.Append oParameter
    Set oRecordset = .Execute
  End With

  If Not (oRecordset.EOF And oRecordset.BOF) Then
     SeleccionarPorId = True
     oTabla.IdSubclasificacionDx = IIf(IsNull(oRecordset!IdSubclasificacionDx), 0, oRecordset!IdSubclasificacionDx)
     oTabla.IdClasificacionDx = IIf(IsNull(oRecordset!IdClasificacionDx), 0, oRecordset!IdClasificacionDx)
     oTabla.IdDiagnostico = IIf(IsNull(oRecordset!IdDiagnostico), 0, oRecordset!IdDiagnostico)
     oTabla.IdSolicitudQxDiagnostico = IIf(IsNull(oRecordset!IdSolicitudQxDiagnostico), 0, oRecordset!IdSolicitudQxDiagnostico)
     oTabla.IdAtencion = IIf(IsNull(oRecordset!IdAtencion), 0, oRecordset!IdAtencion)
     oTabla.GrupoHIS = IIf(IsNull(oRecordset!GrupoHIS), 0, oRecordset!GrupoHIS)
     oTabla.SubGrupoHIS = IIf(IsNull(oRecordset!SubGrupoHIS), 0, oRecordset!SubGrupoHIS)
     oTabla.IdSolicitudQx = IIf(IsNull(oRecordset!IdSolicitudQx), 0, oRecordset!IdSolicitudQx)
  Else
     SeleccionarPorId = False
  End If
  oRecordset.Close
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
End Function
```

Método para actualizar una colección de registros en la base de datos de SolicitudQxDiagnostico

```
Function ActualizarDiagnosticosSolicitudQx(oDiagnosticos As Collection, lIdSolicitudQx As Long) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter
Dim sSQL As String
Dim oDOSolicitudQxDiagnostico As New DOSolicitudQxDiagnosticos
  ActualizarDiagnosticosSolicitudQx = False
  With oCommand
      .CommandType = adCmdStoredProc
      Set .ActiveConnection = mo_Conexion
      .CommandTimeout = 150
      .CommandText = "usp_delete_DiagnosticosEliminarXIdSolicitudQx_03092024"
      Set oParameter = .CreateParameter("@IdSolicitudQx", adInteger, adParamInput, 0, lIdSolicitudQx): .Parameters.Append oParameter
      .Execute
  End With

  If Not oDiagnosticos Is Nothing Then
      For Each oDOSolicitudQxDiagnostico In oDiagnosticos
         oDOSolicitudQxDiagnostico.IdSolicitudQx = lIdSolicitudQx
         If Not Insertar(oDOSolicitudQxDiagnostico) Then
            Exit Function
         End If
      Next
   End If
  ActualizarDiagnosticosSolicitudQx = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  MsgBox Err.Description

  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface
Exit Function
End Function
```

Para el registro de solicitud de sala de operación, se estableció el formulario que se observa en la Fig. 42, para el registro de solicitud de sala de operaciones, lo siguiente:

- Datos generales del paciente: consultar N° HC y Cuenta, apellidos y nombres, edad, sexo y Peso
- Detalles de la solicitud: registro servicio y especialidad, tipo de intervención, tipo de cirugía, sala asignada, fecha, hora, duración de cirugía y turno.
- Diagnóstico y procedimiento: registro de diagnósticos principales o secundarios y registro de procedimientos de cirugía que se realizara.
- Asignación de personal médico: registro de médico principal y ayudantes
- Información adicional: resumen de cirugía, resumen de diagnóstico, indicador con unidades de sangre, grupo sanguíneo y factor RH, riesgo quirúrgico, tipo de anestesia.

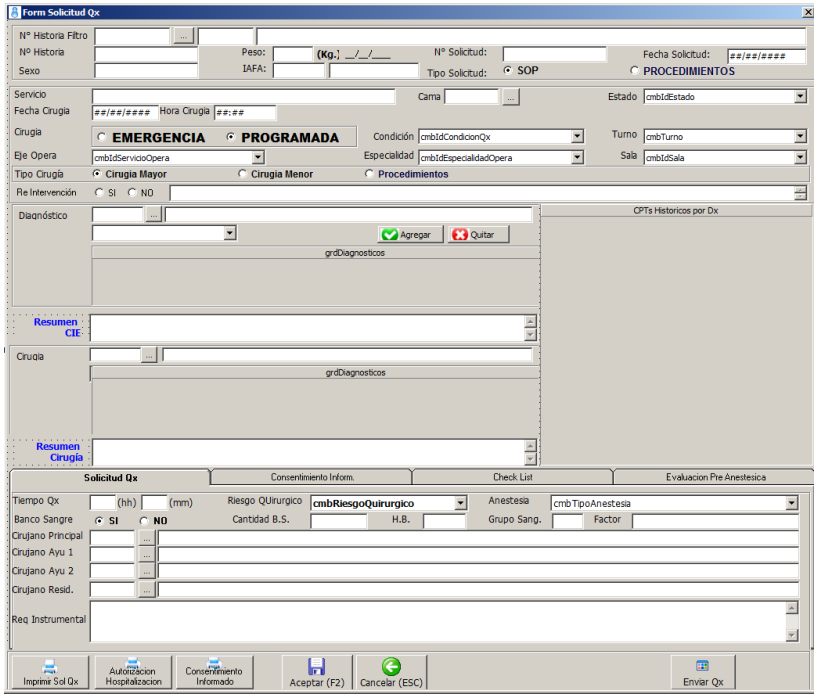


Fig. 42: Formulario de registro de solicitud de sala de operaciones

Aquí se muestra el código del formulario de acuerdo al estándar establecido, en la Fig. 43 se muestra la declaración de variables globales utilizadas en este formulario.

```
Option Explicit

Dim lMostrarAlerta As Boolean
Dim mo_Teclado As New sighentidades.Teclado
Dim mo_Formulario As New sighentidades.formulario
Dim oSolicitudQx As New SIGHDatos.SolicitudQx
Dim doSolicitudQx As New SIGHComun.doSolicitudQx
Dim ml_idTipoServicio As Long
Dim ml_IDUsuario As Long
Dim ms_MensajeError As String
Dim mi_Opcion As sghOpciones
Dim mb_ExistenDatos As Boolean
Dim ml_id As Long
Dim ml_idSolicitudQx As Long
Dim lcBuscaParametro As New SIGHDatos.Parametros
Dim mo_AdminComun As New ReglasComunes
Dim mo_atencion As New SIGHDatos.Atenciones
```

Fig. 43: Declaración de variables formulario de solicitud de sala de operaciones

En la Fig. 44 se muestra las propiedades del formulario utilizadas para el registro de solicitud de sala de operaciones.

```
Property Let idTipoIntervencion(lValue As Long)
   ml_idTipoIntervencion = lValue
End Property
Property Let idCuentaAtencion(iValue As Long)
  ml_idCuentaAtencion = iValue
End Property
Property Get idCuentaAtencion() As Long
  idCuentaAtencion = ml_idCuentaAtencion
End Property
Property Let lcNombrePc(lValue As String)
  mo_lcNombrePc = lValue
End Property
Property Let lnIdTablaLISTBARITEMS(lValue As Long)
  mo_lnIdTablaLISTBARITEMS = lValue
End Property
Property Let Opcion(iValue As sghOpciones)
  mi_Opcion = iValue
End Property
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Let IDUsuario(lValue As Long)
  ml_IDUsuario = lValue
End Property
Property Get id() As Long
  id = ml_id
End Property
```

Fig. 44: Propiedades del formulario de solicitud de sala de operaciones

En la Tabla XXVII se muestran los principales métodos y funciones aplicados en el formulario de solicitud de sala de operaciones.

Tabla XXVII: Codificación en formulario de solicitud de sala de operaciones

Evento Load de Formulario	Metodo para cargar datos al formulario	Metodo para inicializar registro nuevo

Private Sub Form_Load()
 	
mo_Formulario.HabilitarDeshabilitar txtNroHistoriaBusqueda, False
Select Case mi_Opcion
Case sghAgregar
Me.Caption = "Agregar Solicitud Qx"
mo_Formulario.HabilitarDeshabilitar txtNroHistoriaBusqueda, True
cmdBuscaCuentaPorApellidos.Enabled = True
Case sghModificar
Me.Caption = "Modificar SOlicitud Qx"
Case sghConsultar
Me.Caption = "Consultar Solicitud Qx"
Case sghEliminar
Me.Caption = "Eliminar Solicitud Qx"
End Select
 	
CargarDatosAlFormulario
 	
End Sub
	Sub CargarDatosAlFormulario()
	
	cargaComboboxes
	
	Select Case mi_Opcion
	Case sghAgregar
	cargaAlosControlesNuevo
	If ml_idTipoIntervencion = 1 Then
	rbProgramada.Value = True
	FrameTipoIntervencion.Enabled = False
	ElseIf ml_idTipoIntervencion = 2 Then
	rbEmergencia.Value = True
	FrameTipoIntervencion.Enabled = False
	End If
	Case sghModificar
	CargarDatosALosControles
	ConfiguraPermisos
	mo_Formulario.HabilitarDeshabilitar txtNcuenta, False
	Case sghConsultar
	CargarDatosALosControles
	mo_Formulario.HabilitarDeshabilitar txtNcuenta, False
	Me.btnAceptar.Enabled = False
	Case sghEliminar
	CargarDatosALosControles
	mo_Formulario.HabilitarDeshabilitar txtNcuenta, False
	End Select
	End Sub
		Sub cargaAlosControlesNuevo()
		txtNcuenta.Text = ""
		txtNPaciente.Text = ""
		txtFechaSolicitud.Text = lcBuscaParametro.RetornaFechaServidorSQL
		txtNroSolicitud.Text = ""
		
		If ml_idTipoIntervencion <> 2 Then
		If Weekday(lcBuscaParametro.RetornaFechaServidorSQL) = 7 Then
		txtFechaProgramada.Text = DateAdd("d", 2, lcBuscaParametro.RetornaFecha
		Else
		txtFechaProgramada.Text = DateAdd("d", 1, lcBuscaParametro.RetornaFecha
		End If
		ElseIf ml_idTipoIntervencion = 2 Then
		txtFechaProgramada.Text = DateAdd("d", 0, lcBuscaParametro.RetornaFechaSe
		End If
		
		mo_cmbIdCondicionQx.BoundText = 1
		
		Me.ucDiagnosticoSolicitudQx1.TipoDiagnostico = sghHospitalizacionAtencion
		Me.ucDiagnosticoSolicitudQx1.inicializar
		Me.ucDiagnosticoSolicitudQx1.IDUsuario = ml_IDUsuario
		Me.ucDiagnosticoSolicitudQx1.CargarDiagnosticosAlObjetoDatos mo_Diagnostico
		Me.ucDiagnosticoSolicitudQx1.ConfigurarComboBoxes
		
		Me.ucSolicitudQxProcedimientos1.inicializar
		Me.ucSolicitudQxProcedimientos1.IDUsuario = ml_IDUsuario
		Me.ucSolicitudQxProcedimientos1.CargarDiagnosticosAlObjetoDatos mo_CPTs
		
		End Sub

Metodo que carga datos a los controles	**Metodo para cargar listas desplegables**	**Evento boton Aceptar del formulario**

Metodo validar reglas del registro

```
Function ValidarReglas() As Boolean
 Dim sMensaje As String
 ValidarReglas = False

  If Len(txtResumenCIE.Text) < 30 Then
    sMensaje = sMensaje & vbNewLine & " El Resumen de CIE debe contar con al menos
  End If
  If Len(txtResumenCPT.Text) < 30 Then
    sMensaje = sMensaje & vbNewLine & " El Resumen de CPT debe contar con al meno
  End If
  If Not (txtGrupoSanguineo.Text = "A" Or txtGrupoSanguineo.Text = "B" Or txtGrupoSangu
    sMensaje = sMensaje & vbNewLine & " Grupo Sanguineo solo puede ser A,B,AB,O "
  End If
  If Not EsFecha(Me.txtFechaProgramada, "DD/MM/AAAA") Then
     sMensaje = sMensaje & vbNewLine & "La fecha Programada ingresada no es válida"
     txtFechaProgramada = sighentidades.FECHA_VACIA_DMY
  End If
  If Not EsFecha(Me.txtFechaSolicitud, "DD/MM/AAAA") Then
     sMensaje = sMensaje & vbNewLine & "La fecha Solicitud ingresada no es válida"
     txtFechaSolicitud = sighentidades.FECHA_VACIA_DMY
  End If
  If Not EsHora(Me.txtHoraProgramada.Text) Then
     sMensaje = sMensaje & vbNewLine & "La HORA Programada no es válida"
     txtHoraProgramada.Text = sighentidades.HORA_VACIA_HM
  End If
  If val(Me.txtTiempoIntervencion.Text) > 12 Then
    sMensaje = sMensaje + "Tiempo Intervencion no debe exceder 12 horas" + Chr(13)
  End If
  If val(Me.txtMinutoIntervencion.Text) > 59 Then
    sMensaje = sMensaje + "Minutos de Intervencion no debe exceder 59 minutos" + Chr(
  End If
  If sMensaje <> "" Then
    MsgBox sMensaje, vbInformation, Me.Caption
    Exit Function
  End If
 ValidarReglas = True
End Function
```

Metodo para validar datos obligatorios

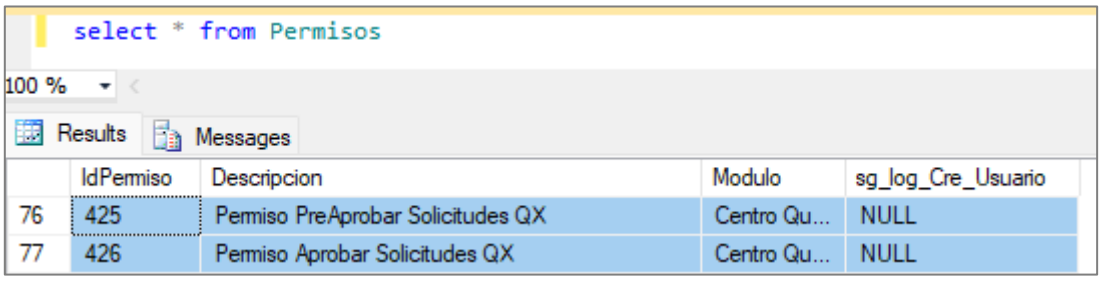
```
Function ValidarDatosObligatorios() As Boolean
  Dim sMensaje As String
  ValidarDatosObligatorios = False

  If Trim(Me.txtNcuenta.Text) = "" Then
     sMensaje = sMensaje + "Ingrese el Nro. de Cuenta (corto)" + Chr(13)
  End If
  If Trim(Me.txtHoraProgramada.Text) = "" Then
     sMensaje = sMensaje + "Ingrese HORA programada" + Chr(13)
  End If
  If val(mo_cmbIdServicioOpera.BoundText) = 0 Then
     sMensaje = sMensaje + "Ingrese Servicio que Opera " + Chr(13)
  End If
  If val(mo_cmbTipoAnestesia.BoundText) = 0 Then
     sMensaje = sMensaje + "Ingrese Tipo Anestesia" + Chr(13)
  End If
  If val(Me.txtTiempoIntervencion.Text) + val(Me.txtMinutoIntervencion.Text) = 0 Then
     sMensaje = sMensaje + "Ingrese Tiempo Intervencion" + Chr(13)
  End If
  If val(Me.txtDNIMedicoPrincipal.Tag) = 0 Then
     sMensaje = sMensaje + "Ingrese Medico Principal " + Chr(13)
  End If

   Me.ucDiagnosticoSolicitudQx1.idAtencion = ml_IdAtencion
   Set mo_Diagnosticos = Nothing
   Me.ucDiagnosticoSolicitudQx1.CargarDiagnosticosAlObjetoDatos mo_Diagnosticos
   If mo_Diagnosticos.Count = 0 Then
      sMensaje = sMensaje + "Ingrese el Diagnosticos "
      Exit Function
   End If

   If sMensaje <> "" Then
     MsgBox sMensaje, vbInformation, Me.Caption
     Exit Function
   End If
   ValidarDatosObligatorios = True

End Function
```

Metodo para cargar datos para envio a bd

```
Sub CargaDatosAlObjetosDeDatos()

  With doSolicitudQx
     .idCuentaAtencion = Me.txtNcuenta.Text
     .idPaciente = val(Me.txtNPaciente.Text)
     .NroSolicitud = val(txtNroSolicitud.Text)
     .FechaSolicitud = Me.txtFechaSolicitud
     .FechaProgramada = Me.txtFechaProgramada.Text
     .HoraProgramada = Me.txtHoraProgramada.Text
     .Peso = val(Me.txtPeso.Text)
     .idServicio = Me.txtServicio.Tag
     .idCama = CLng(Me.txtCama.Tag)
     .idTurnoIntervencion = val(mo_cmbTurno.BoundText)
     .descripcioncirugia = txtDescripcionCirugia.Text
     .RiesgoDerivado = txtRiesgoDerivado.Text
     .idTipoDocumentoTutor = val(mo_cmbMadreTipoDocumento.BoundText)
     .IdTipoParentesco = val(mo_cmbParentesco.BoundText)
     .nroDocumentoTutor = txtMadreDocumento.Text
     .ApellidoPaternoTutor = Me.txtMadreApellidoP.Text
     .ApellidoMaternoTutor = Me.txtMadreApellidoM.Text
     .nombresTutor = Me.txtNombresMadre.Text
     .IdCondicion = val(mo_cmbIdCondicionQx.BoundText)
     .idEspecialidadOpera = val(mo_cmbIdEspecialidadOpera.BoundText)
     If (rbProgramada.Value) Then
        .idTipoIntervencion = 1
     Else
        .idTipoIntervencion = 2
     End If
  End With
  ucDiagnosticoSolicitudQx1.idAtencion = ml_IdAtencion
  Set mo_Diagnosticos = Nothing
  ucDiagnosticoSolicitudQx1.CargarDiagnosticosAlObjetoDatos mo_Diagnosticos

  ucSolicitudQxProcedimientos1.idAtencion = ml_IdAtencion
  Set mo_CPTs = Nothing
  ucSolicitudQxProcedimientos1.CargarDiagnosticosAlObjetoDatos mo_CPTs

End Sub
```

3.1.4.2 Implementación CU02 Aprobación de solicitud de sala de operaciones

Para esta implementación se realizó mediante permisos para aprobación de jefatura y aprobación de dirección, como se muestra en la Fig. 45.



```
select * from Permisos
```

100 %

Results | Messages

	IdPermiso	Descripcion	Modulo	sg_log_Cre_Usuario
76	425	Permiso PreAprobar Solicitudes QX	Centro Qu...	NULL
77	426	Permiso Aprobar Solicitudes QX	Centro Qu...	NULL

Fig. 45: Permisos en base de datos para aprobación de solicitud de sala de operaciones

Estos permisos fueron validados en el formulario de solicitud de sala de operaciones desde el método de configurar Permisos, para las validaciones de cambio de estado asignado a cada usuario correspondiente como jefaturas de cada servicio (Pre Aprobar) y dirección general (Aprobar), como se muestran en las Fig. 46 y Fig. 47.

```
Sub ConfiguraPermisos()
    Dim oRsPermisos As New Recordset
   Set oRsPermisos = mo_ReglasSeguridad.UsuariosRolesSeleccionarPermisosTodos(ml_IdUsuario)
   lbTIenePermisoAprobarSolicitud = False
   lbTIenePermisoPreAprobarSolicitud = False
   If oRsPermisos.RecordCount > 0 Then
     Do While Not oRsPermisos.EOF
       Select Case oRsPermisos.Fields!idpermiso
       Case 425    'permiso para pre aprobar solicitud qx
           lbTIenePermisoPreAprobarSolicitud = True
       Case 426    'permiso para pre aprobar solicitud qx
           lbTIenePermisoAprobarSolicitud = True
       End Select
       oRsPermisos.MoveNext
     Loop
   End If
   Set oRsPermisos = Nothing
End Sub
```

Fig. 46: Asignación de permisos para aprobación de solicitud de sala de operaciones

```
Function ValidarReglas() As Boolean
  ValidarReglas = False
  Dim sMensaje As String

  Select Case mi_Opcion
  Case sghModificar
        Select Case Me.cmbIdEstado.Text
        Case "PreAprobado"
              If Not lbTIenePermisoPreAprobarSolicitud Then
                 sMensaje = sMensaje & vbNewLine & _
                 "No está Autorizado para Pre Aprobar Solicitudes"
              End If
              If mo_reglasQx.ValidaPermisoPreAprobacion(ml_IdUsuario) <= 0 Then
                 sMensaje = sMensaje & vbNewLine & _
                 "No está Autorizado para Pre Aprobar Solicitudes del Servicio Indicado"
              End If
        Case "Aprobado"
              If Not lbTIenePermisoAprobarSolicitud Then
                 sMensaje = sMensaje & vbNewLine & "No está Autorizado para Aprobar Solicitudes"
              End If
        End Select
  End Select
  If sMensaje <> "" Then
     MsgBox sMensaje, vbInformation, Me.Caption
     Exit Function
  End If
  ValidarReglas = True
End Function
```

Fig. 47: Validación de permisos para aprobación de solicitud de sala de operaciones

3.1.4.3 Implementación CU03 Programación de sala de operaciones

Para la programación de Sala de operaciones se ha establecido una tabla en base de datos **AprobadosQx** que guarda la información de todas las cirugías programadas en centro quirúrgico a partir de las solicitudes aprobadas. Para este desarrollo se ha utilizado las clases DOAprobadosQx para mapear todos los campos de la base de datos correspondientes a la tabla, AprobadosQx para establecer los métodos con conexión a la base de datos

En la Tabla XXVIII se muestra la Clase DOAprobadosQx con las propiedades establecidas en el diagrama de base de datos.

Tabla XXVIII: Clase Data Object DOAprobadosQx

Variables internas

```
'Solicitudes Aprobadas de Centro Quirurgico
'
Option Explicit
Dim ml_Auditoria As Long
Dim ml_idAprobadoQx As Long
Dim mda_FechaAprobado As Date
Dim ml_idSolicitudQx As Long
Dim ml_NroSolicitud As Long
Dim ml_idCuentaSolicitante As Long
Dim ml_idCuentaAtencion As Long
Dim ml_idPaciente As Long
Dim ml_idServicio As Long
Dim ml_idServicioOpera As Long
Dim ml_idSala As Long
Dim ml_idCama As Long
Dim mda_FechaProgramada As Date
Dim mda_FechaProgramadaFin As Date
Dim ml_idSolicitudCierreDiaQx As Long
Dim ms_HoraInicioPrograma As String
Dim ms_HoraFinPrograma As String
Dim ml_idTurnoIntervencion As Long
Dim ml_TiempoIntervencion As Long
Dim ms_RiesgoQuirurgico As String
Dim ml_idTipoAnestesia As Long
Dim mb_indDepositoSangre As Boolean
Dim md_CantidadSanngre As Double
Dim ms_GrupoSanguineo As String
Dim ms_FactorSanguineo As String
Dim ml_idMedicoPrincipal As Long
Dim ml_idMedicoAyudante1 As Long
Dim ml_idMedicoAyudante2 As Long
Dim ms_Instrumental As String
Dim ml_idMedicoSolicitante As Long
Dim ml_idEstado As Integer
Dim ml_idMedicoAnestesiologo1 As Long
Dim ml_idMedicoAnestesiologo2 As Long
Dim ml_idTipoIntervencion As Integer
Dim ml_idInstrumentista As Long
Dim ml_idCirculante As Long
Dim ml_idEspecialidadOpera As Long
Dim ml_idMedicoResidente As Long
```

Propiedades de clase

```
Property Let idMedicoResidente(lValue As Double)
  ml_idMedicoResidente = lValue
End Property
Property Get idMedicoResidente() As Double
  idMedicoResidente = ml_idMedicoResidente
End Property
Property Let idEspecialidadOpera(lValue As Long)
  ml_idEspecialidadOpera = lValue
End Property
Property Get idEspecialidadOpera() As Long
  idEspecialidadOpera = ml_idEspecialidadOpera
End Property
Property Let IdUsuarioAuditoria(lValue As Long)
  ml_Auditoria = lValue
End Property
Property Get IdUsuarioAuditoria() As Long
  IdUsuarioAuditoria = ml_Auditoria
End Property
Property Let idAprobadoQx(lValue As Long)
  ml_idAprobadoQx = lValue
End Property
Property Get idAprobadoQx() As Long
  idAprobadoQx = ml_idAprobadoQx
End Property
Property Let FechaAprobado(lValue As Date)
  mda_FechaAprobado = lValue
End Property
Property Get FechaAprobado() As Date
Property Let FechaAprobado(lValue As Date)
  mda_FechaAprobado = lValue
End Property
Property Get FechaAprobado() As Date
  FechaAprobado = mda_FechaAprobado
End Property
Property Let idSolicitudQx(lValue As Long)
  ml_idSolicitudQx = lValue
End Property
Property Get idSolicitudQx() As Long
  idSolicitudQx = ml_idSolicitudQx
End Property
Property Let idSolicitudCierreDiaQx(lValue As Long)
  ml_idSolicitudCierreDiaQx = lValue
End Property
Property Get idSolicitudCierreDiaQx() As Long
  idSolicitudCierreDiaQx = ml_idSolicitudCierreDiaQx
End Property
Property Let NroSolicitud(lValue As Long)
  ml_NroSolicitud = lValue
End Property
Property Get NroSolicitud() As Long
  NroSolicitud = ml_NroSolicitud
End Property
Property Let idCuentaSolicitante(lValue As Long)
  ml_idCuentaSolicitante = lValue
End Property
Property Get idCuentaSolicitante() As Long
```

Propiedades de clase

```
Property Get idCuentaSolicitante() As Long
  idCuentaSolicitante = ml_idCuentaSolicitante
End Property
Property Let idCuentaAtencion(lValue As Long)
  ml_idCuentaAtencion = lValue
End Property
Property Get idCuentaAtencion() As Long
  idCuentaAtencion = ml_idCuentaAtencion
End Property
Property Let idPaciente(lValue As Long)
  ml_idPaciente = lValue
End Property
Property Get idPaciente() As Long
  idPaciente = ml_idPaciente
End Property
Property Let idServicio(lValue As Long)
  ml_idServicio = lValue
End Property
Property Get idServicio() As Long
  idServicio = ml_idServicio
End Property
Property Let idCama(lValue As Long)
  ml_idCama = lValue
End Property
Property Get idCama() As Long
  idCama = ml_idCama
End Property
Property Let idServicioOpera(lValue As Long)
Property Let idServicioOpera(lValue As Long)
  ml_idServicioOpera = lValue
End Property
Property Get idServicioOpera() As Long
  idServicioOpera = ml_idServicioOpera
End Property
Property Let idSala(lValue As Long)
  ml_idSala = lValue
End Property
Property Get idSala() As Long
  idSala = ml_idSala
End Property
Property Let FechaProgramada(lValue As Date)
  mda_FechaProgramada = lValue
End Property
Property Get FechaProgramada() As Date
  FechaProgramada = mda_FechaProgramada
End Property
Property Let HoraInicioProgramada(lValue As String)
  ms_HoraInicioProgramada = lValue
End Property
Property Get HoraInicioProgramada() As String
  HoraInicioProgramada = ms_HoraInicioProgramada
End Property
Property Let HoraFinProgramada(lValue As String)
  ms_HoraFinProgramada = lValue
End Property
Property Get HoraFinProgramada() As String
```

Así mismo en la siguiente Tabla XXIX se muestran los métodos principales para conectarse a la base de datos en la clase AprobadosQx.

Tabla XXIX: Clase ADO AprobadoQx

Declaracion de variables y propiedades de la clase

```
'-----------------------------------------------------------------------------
'      Programa:  Clase ADO de la tabla AprobadosQx
'-----------------------------------------------------------------------------
Option Explicit
Dim mo_Conexion As ADODB.Connection
Dim ms_MensajeError As String
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Set Conexion(oValue As ADODB.Connection)
  Set mo_Conexion = oValue
End Property
Property Get Conexion() As ADODB.Connection
  Set Conexion = mo_Conexion
End Property
```

Metodo para insertar un registro en la tabla AprobadosQx

```
Function Insertar(ByVal oTabla As DoAprobadosQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Insertar = False
  With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
      .CommandText = "usp_insert_AprobadoQxAgregar_27102024"
     .Execute
        oTabla.idAprobadoQx = .Parameters("@idAprobadoQx")
  End With

  Insertar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) _
  + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface de acceso a datos"
End Function
```

Metodo para modificar un registro en la tabla AprobadosQx

```
Function Modificar(ByVal oTabla As DoAprobadosQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Modificar = False
  With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion

     .CommandText = "usp_update_AprobadoQxModificar_27102024"

     .Execute
  End With
  Modificar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vb
End Function
```

Metodo para Eliminar un registro en la tabla AprobadosQx

```
Function Eliminar(ByVal oTabla As DoAprobadosQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Eliminar = False
  With oCommand

     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
      .CommandText = "usp_delete_AprobadoQxEliminar_28022024"
      Set oParameter = .CreateParameter("@IdAprobadoQx", adInteger, adParamInput, 0, IIf(oTabla.idAprobadoQx = 0, Null, oTabla.idAprobadoQx)): .I
      Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria)
      .Parameters.Append oParameter
     .Execute
  End With

  Eliminar = True
  ms_MensajeError = ""
ManejadorDeError:
   ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", v
End Function
```

Metodo para Seleccionar un registro en la tabla AprobadosQx

```
Function SeleccionarPorId(ByVal oTabla As DoAprobadosQx) As Boolean
On Error GoTo ManejadorDeError
Dim oRecordset As New ADODB.Recordset
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  SeleccionarPorId = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_select_AprobadoQxSeleccionarPorId_30102023"
      Set oParameter = .CreateParameter("@IdAprobadoQx", adInteger, adParamInput, 0, oTabla.idAprobadoQx): .Parameters.Append oParameter
    Set oRecordset = .Execute
  End With

  If Not (oRecordset.EOF And oRecordset.BOF) Then
     SeleccionarPorId = True
     oTabla.idAprobadoQx = IIf(IsNull(oRecordset!idAprobadoQx), 0, oRecordset!idAprobadoQx)
     oTabla.IdSolicitudQx = IIf(IsNull(oRecordset!IdSolicitudQx), 0, oRecordset!IdSolicitudQx)
     oTabla.idCuentaSolicitante = IIf(IsNull(oRecordset!idCuentaSolicitante), 0, oRecordset!idCuentaSolicitante)
     oTabla.idCuentaAtencion = IIf(IsNull(oRecordset!idCuentaAtencion), 0, oRecordset!idCuentaAtencion)
     oTabla.FechaAprobado = IIf(IsNull(oRecordset!FechaAprobado), 0, oRecordset!FechaAprobado)
     oTabla.IDPaciente = IIf(IsNull(oRecordset!IDPaciente), 0, oRecordset!IDPaciente)
     oTabla.idServicio = IIf(IsNull(oRecordset!idServicio), 0, oRecordset!idServicio)
      oTabla.FechaProgramada = IIf(IsNull(oRecordset!FechaProgramada), 0, oRecordset!FechaProgramada)
      oTabla.HoraInicioProgramada = IIf(IsNull(oRecordset!HoraInicioProgramada), "", oRecordset!HoraInicioProgramada)
      oTabla.HoraFinProgramada = IIf(IsNull(oRecordset!HoraFinProgramada), "", oRecordset!HoraFinProgramada)
      oTabla.idTipoIntervencion = IIf(IsNull(oRecordset!idTipoIntervencion), 0, oRecordset!idTipoIntervencion)
      oTabla.idTurnoIntervencion = IIf(IsNull(oRecordset!idTurnoIntervencion), 0, oRecordset!idTurnoIntervencion)
      oTabla.idTipoAnestesia = IIf(IsNull(oRecordset!idTipoAnestesia), 0, oRecordset!idTipoAnestesia)
      oTabla.idMedicoPrincipal = IIf(IsNull(oRecordset!idMedicoPrincipal), 0, oRecordset!idMedicoPrincipal)
      oTabla.idMedicoAyudante1 = IIf(IsNull(oRecordset!idMedicoAyudante1), 0, oRecordset!idMedicoAyudante1)
      oTabla.idMedicoAyudante2 = IIf(IsNull(oRecordset!idMedicoAyudante2), 0, oRecordset!idMedicoAyudante2)
      oTabla.idMedicoSolicitante = IIf(IsNull(oRecordset!idMedicoSolicitante), 0, oRecordset!idMedicoSolicitante)
      oTabla.IdEstado = IIf(IsNull(oRecordset!IdEstado), 1, oRecordset!IdEstado)
      oTabla.idMedicoAnestesiologo1 = IIf(IsNull(oRecordset!idMedicoAnestesiologo1), 0, oRecordset!idMedicoAnestesiologo1)
       oTabla.idMedicoAnestesiologo2 = IIf(IsNull(oRecordset!idMedicoAnestesiologo2), 0, oRecordset!idMedicoAnestesiologo2)
      oTabla.idCirculante = IIf(IsNull(oRecordset!idCirculante), 0, oRecordset!idCirculante)
      oTabla.idInstrumentista = IIf(IsNull(oRecordset!idInstrumentista), 0, oRecordset!idInstrumentista)
      oTabla.idEspecialidadOpera = IIf(IsNull(oRecordset!idEspecialidadOpera), 0, oRecordset!idEspecialidadOpera)
      oTabla.idMedicoResidente = IIf(IsNull(oRecordset!idMedicoResidente), 0, oRecordset!idMedicoResidente)
  Else
     SeleccionarPorId = False
  End If
  oRecordset.Close
  ms_MensajeError = ""
Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + _
  "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface de acceso a datos"
End Function
```

Para el desarrollo de las interfaces para la programación de sala de operaciones, se estableció el siguiente formulario, para que el personal administrativo quien registra, tenga la opción de confirmar la fecha de cirugía, hora inicio de cirugía, hora fin de cirugía, asignar anestesiólogo,

actualizar el tipo de anestesia y riesgo ASA sugerida por el anestesiólogo asignado, como se observa en la Fig. 48.

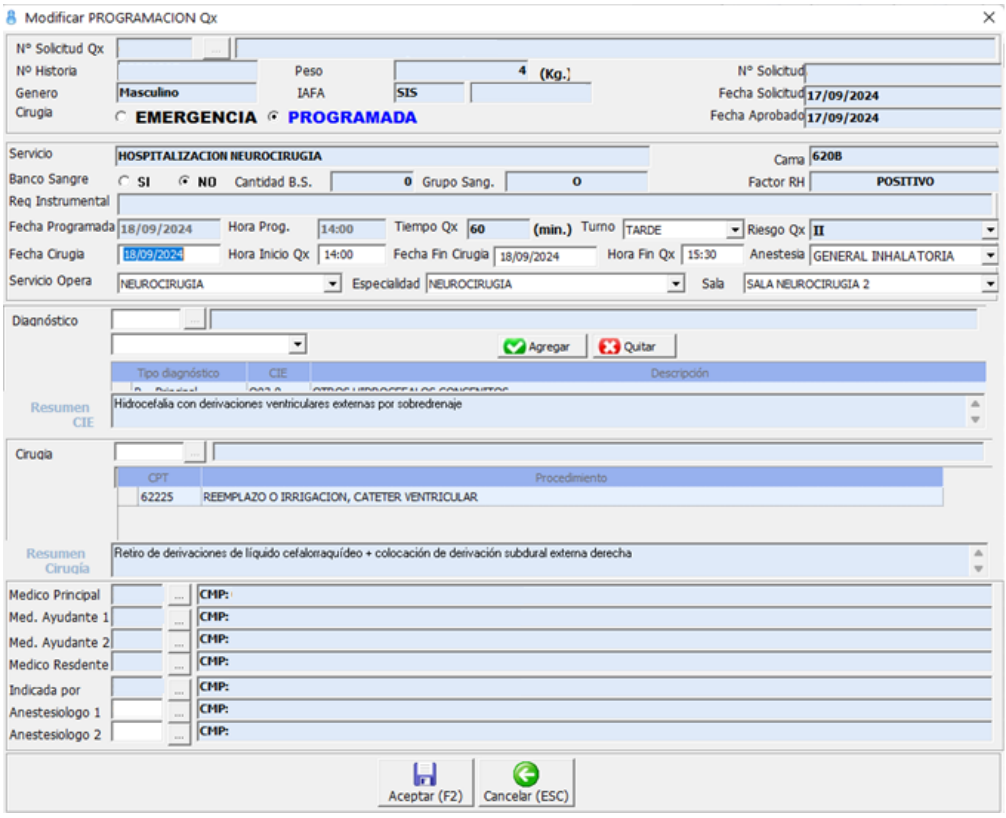


Fig. 48: Formulario de programación de sala de operaciones

En la Tabla XXX se presenta los principales variables, propiedades, métodos y funciones aplicados para la programación de sala de operaciones.

Tabla XXX: Codificación de formulario de programación de sala de operaciones

Variables Internas	Propiedades de clase	Propiedades de clase
'Formulario de Programacion de Sala de Operaciones Option Explicit Dim mo_Teclado As New sighentidades.Teclado Dim mo_Formulario As New sighentidades.formulario Dim oAprobadoQx As New sighdatos.AprobadoQx Dim odoAprobadoQx As New SIGHComun.DoAprobadosQx Dim ml_IdUsuario As Long Dim ms_MensajeError As String Dim mi_Opcion As sghOpciones Dim mb_ExistenDatos As Boolean Dim ml_Id As Long Dim lcBuscaParametro As New sighdatos.Parametros Dim mo_AdminComun As New ReglasComunes Dim mo_reglasQx As New SIGHNegocios.ReglasQx Dim mo_Apariencia As New sighentidades.GridInfragistic Dim mo_InIdTablaLISTBARITEMS As Long Dim mo_lcNombrePc As String Dim mo_Diagnosticos As New Collection Dim mo_CPTs As New Collection Dim mo_cmbTurno As New ListaDespleglable Dim mo_cmbTipoAnestesia As New ListaDespleglable Dim mo_cmbTipoRiesgoQuirurgico As New ListaDespleglable Dim mo_cmbIdServicioOpera As New ListaDespleglable Dim mo_cmbIdSala As New ListaDespleglable Dim ml_IdSolicitudQx As Long Dim ml_IdAprobadoQx As Long	Property Let lcNombrePc(lValue As String) mo_lcNombrePc = lValue End Property Property Let InIdTablaLISTBARITEMS(lValue As Long) mo_InIdTablaLISTBARITEMS = lValue End Property Property Let ExistenDatos(bValue As Boolean) mb_ExistenDatos = bValue End Property Property Get ExistenDatos() As Boolean ExistenDatos = mb_ExistenDatos End Property Property Let Opcion(iValue As sghOpciones) mi_Opcion = iValue End Property Property Get MensajeError() As String MensajeError = ms_MensajeError End Property Property Let IdUsuario(lValue As Long) ml_IdUsuario = lValue End Property Property Let Id(lValue As Long) ml_Id = lValue End Property	Sub CargarDatosAlFormulario() CargaComboBoxes Select Case mi_Opcion Case sghAgregar cargaAlosControlesNuevo Case sghModificar CargarDatosALosControles txtNSolicitud_LostFocus mo_Formulario.HabilitarDeshabilitar txtNSolicitud, False cmdBuscaCuentaPorApellidos.Enabled = False Case sghConsultar CargarDatosALosControles txtNSolicitud_LostFocus mo_Formulario.HabilitarDeshabilitar txtNSolicitud, False Me.btnAceptar.Enabled = False cmdBuscaCuentaPorApellidos.Enabled = False Case sghEliminar CargarDatosALosControles txtNSolicitud_LostFocus mo_Formulario.HabilitarDeshabilitar txtNSolicitud, False cmdBuscaCuentaPorApellidos.Enabled = False End Select End Sub

Metodo para inciar los controles para un nuevo registro

```
Sub cargaAlosControlesNuevo()
txtNSolicitud.Text = ""
txtNPaciente.Text = ""
txtFechaAProbado.Text = lcBuscaParametro.RetornaFechaServidorSQL
txtNroSolicitud.Text = ""

If Weekday(lcBuscaParametro.RetornaFechaServidorSQL) = 7 Then
txtFechaProgramada.Text = DateAdd("d", 2, lcBuscaParametro.RetornaFechaServidorSQL)
Else
txtFechaProgramada.Text = DateAdd("d", 1, lcBuscaParametro.RetornaFechaServidorSQL)
End If

Me.ucDiagnosticoSolicitudQx1.inicializar
Me.ucDiagnosticoSolicitudQx1.IdUsuario = ml_IdUsuario
Me.ucDiagnosticoSolicitudQx1.CargarDiagnosticosAlObjetoDatos mo_Diagnosticos
Me.ucDiagnosticoSolicitudQx1.ConfigurarComboBoxes

Me.ucSolicitudQxProcedimientos1.inicializar
Me.ucSolicitudQxProcedimientos1.IdUsuario = ml_IdUsuario
Me.ucSolicitudQxProcedimientos1.CargarDiagnosticosAlObjetoDatos mo_CPTs

End Sub
```

Evento Aceptar del formulario

```
Private Sub btnAceptar_Click()
  Select Case mi_Opcion
  Case sghAgregar
    If ValidarDatosObligatorios() Then
      If ValidarReglas() Then
        If AgregarDatos() Then
          MsgBox "Los datos se agregaron correctamente"
          LimpiarFormulario
        End If
      End If
    End If
  Case sghModificar
    If ValidarDatosObligatorios() Then
      If ValidarReglas() Then
        If ModificarDatos() Then
          MsgBox "Los datos se modificaron correctamente"
          Me.Visible = False
        End If
      End If
    End If
  Case sghEliminar
      If ValidarReglas() Then
        If EliminarDatos() Then
          MsgBox "Los datos se eliminaron correctamente"
          Me.Visible = False
        End If
      End If
  End Select
```

Metodo para cargar de base a datos a los controles del formulario

```
Sub CargarDatosALosControles()
  Dim rsap As New Recordset

   Set rsap = mo_reglasQx.AprobadoQxCompletoPorId(odoAprobadoQx.idAprobadoQx)
  If rsap.RecordCount > 0 Then
      txtNroSolicitud = rsap.Fields!NroSolicitud
      txtNSolicitud.Text = rsap.Fields!NroSolicitud
     txtFechaAProbado.Text = rsap.Fields!FechaAprobado
      txtServicio.Tag = rsap.Fields!idServicio
      txtHoraInicioProgramada.Text = rsap.Fields!HoraInicioProgramada
      txtHoraFinProgramada.Text = rsap.Fields!HoraFinProgramada
      txtFechaProgramada.Text = rsap.Fields!FechaProgramada
      txtFechaFinProgramada.Text = rsap.Fields!FechaProgramadaFin
      mo_cmbTurno.BoundText = rsap.Fields!idTurnoIntervencion
      mo_cmbTipoAnestesia.BoundText = rsap.Fields!idTipoAnestesia
      txtServicio.Tag = rsap.Fields!idServicio
      txtServicio.Text = rsap.Fields!ServicioSolicitante
      txtPeso.Text = rsap.Fields!Peso
      txtResumenCIE.Text = IIf(IsNull(rsap.Fields!ResumenCIE), "", rsap.Fields!ResumenCIE)
      txtResumenCPT.Text = IIf(IsNull(rsap.Fields!ResumenCPT), "", rsap.Fields!ResumenCPT)
      mo_cmbIdServicioOpera.BoundText = rsap.Fields!Idservicioopera
      mo_cmbIdEspecialidadOpera.BoundText = rsap.Fields!idEspecialidadOpera
      mo_cmbIdSala.BoundText = rsap.Fields!IdSala
      txtDNIMedicoPrincipal.Tag = IIf(IsNull(rsap.Fields!idMedicoPrincipal), 0, rsap.Fields!idMedicoPrincipal)
       txtDNIMedicoPrincipal.Text = "CMP: " & IIf(IsNull(rsap.Fields!cmpmep), "", rsap.Fields!cmpmep) & " " &
      txtMedicoPrincipal.Text = IIf(IsNull(rsap.Fields!cmpmep), "", rsap.Fields!cmpmep)
      txtDNIMedicoAyudante1.Tag = IIf(IsNull(rsap.Fields!idMedicoAyudante1), 0, rsap.Fields!idMedicoAyuda
```

Metodo para agregar, modificar y eliminar un registro

```
Function AgregarDatos() As Boolean
    Dim oConexion As New ADODB.Connection
   oConexion.Open sighentidades.CadenaConexion
   oConexion.CursorLocation = adUseClient

   CargaDatosAlObjetosDeDatos
   Set oAprobadoQx.Conexion = oConexion
   AgregarDatos = oAprobadoQx.Insertar(odoAprobadoQx)

   oConexion.Close
   Set oConexion = Nothing

Exit Function
ErrorManager:
   MsgBox oAprobadoQx.MensajeError
End Function
```

```
Function ModificarDatos() As Boolean
  Dim oConexion As New ADODB.Connection
    oConexion.Open sighentidades.CadenaConexion
    oConexion.CursorLocation = adUseClient
    Set oAprobadoQx.Conexion = oConexion

   CargaDatosAlObjetosDeDatos
   ModificarDatos = oAprobadoQx.Modificar(odoAprobadoQx)

   oConexion.Close
   Set oConexion = Nothing

Exit Function
ErrorManager:
   MsgBox oAprobadoQx.MensajeError
End Function
```

```
Function EliminarDatos() As Boolean
  Dim oConexion As New ADODB.Connection
    oConexion.Open sighentidades.CadenaConexion
    oConexion.CursorLocation = adUseClient
    Set oAprobadoQx.Conexion = oConexion

   CargaDatosAlObjetosDeDatos

   odoAprobadoQx.idEstado = 0
   EliminarDatos = oAprobadoQx.Eliminar(odoAprobadoQx)

  oConexion.Close
   Set oConexion = Nothing

Exit Function
ErrorManager:
   MsgBox oAprobadoQx.MensajeError
End Function
```

```
    txtDNIMedicoSolicitante.Tag = IIf(IsNull(rsap.Fields!idMedicoSolicitante), 0, rsap.Fields!idMedicoSolicitante)
    txtDNIMedicoSolicitante.Text = "CMP:" & IIf(IsNull(rsap.Fields!CmpSol), "", rsap.Fields!CmpSol) & " " & IIf(IsNull(rsap.Fie
    txtMedicoSolicitante.Text = IIf(IsNull(rsap.Fields!CmpSol), "", rsap.Fields!CmpSol)

    txtDNIMedicoAResidente.Tag = IIf(IsNull(rsap.Fields!idMedicoResidente), 0, rsap.Fields!idMedicoResidente)
    txtDNIMedicoAResidente.Text = "CMP:" & IIf(IsNull(rsap.Fields!cmpResidente), "", rsap.Fields!cmpResidente) & " " & IIf
    txtMedicoAResidente.Text = IIf(IsNull(rsap.Fields!cmpResidente), "", rsap.Fields!cmpResidente)

    txtDNIMedicoAnestesiologo1.Tag = IIf(IsNull(rsap.Fields!idMedicoAnestesiologo1), 0, rsap.Fields!idMedicoAnestesiolog
    txtDNIMedicoAnestesiologo1.Text = "CMP:" & IIf(IsNull(rsap.Fields!cmpane1), "", rsap.Fields!cmpane1) & " " & IIf(IsNull(
    txtMedicoAnestesiolo1.Text = IIf(IsNull(rsap.Fields!cmpane1), "", rsap.Fields!cmpane1)

    txtDNIMedicoAnestesiologo2.Tag = IIf(IsNull(rsap.Fields!idMedicoAnestesiologo2), 0, rsap.Fields!idMedicoAnestesiolog
    txtDNIMedicoAnestesiologo2.Text = "CMP: " & IIf(IsNull(rsap.Fields!cmpane2), "", rsap.Fields!cmpane2) & " " & IIf(IsNull
    txtMedicoAnestesiolo2.Text = IIf(IsNull(rsap.Fields!cmpane2), "", rsap.Fields!cmpane2)

    ucDiagnosticoSolicitudQx1.inicializar
    ucDiagnosticoSolicitudQx1.idSolicitudQx = rsap.Fields!idSolicitudQx
    ucDiagnosticoSolicitudQx1.CargarDatosDeDiagnosticos oConexion

    ucSolicitudQxProcedimientos1.inicializar
    ucSolicitudQxProcedimientos1.idSolicitudQx = rsap.Fields!idSolicitudQx
    ucSolicitudQxProcedimientos1.CargarDatosDeCpts oConexion

  End If
End Sub
```

Por consiguiente, se presenta la bandeja se programación de sala de operaciones que han sido registradas por el personal administrativo de centro quirúrgico, como se observa en la Fig. 49.

Fig. 49: Bandeja de programación de sala de operaciones

3.1.4.4 Implementación CU04 Cerrar programación de sala de operaciones

Este caso se utiliza una vez se hayan realizado la programación de todas las solicitudes aprobadas para el día siguiente hábil de Cirugía.

En la siguiente imagen se establece el proceso de cierre de Dia de Centro Quirúrgico, para casos de alguna corrección o adición de cirugías para esta fecha, tiene la opción de abrir Programación, como se observa en la Fig. 50.

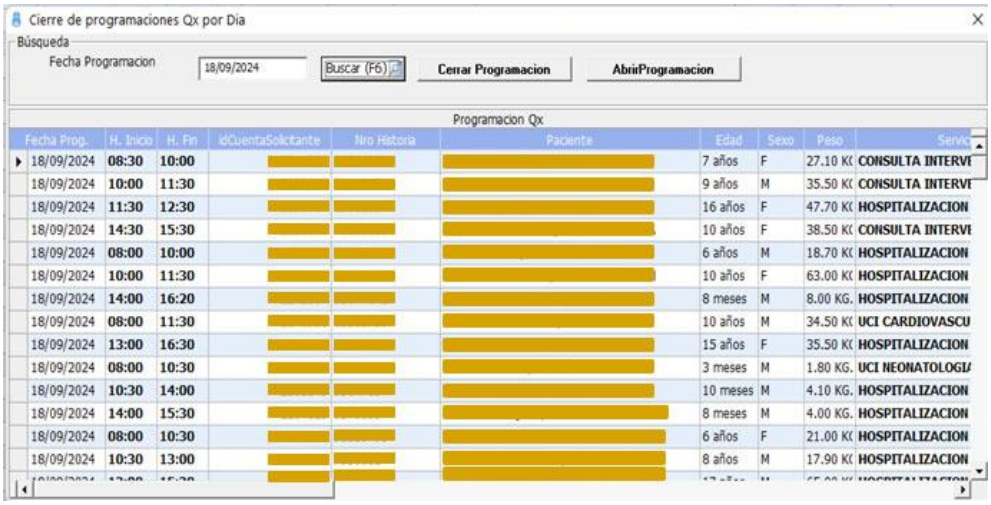


Fig. 50: Formulario de cierre de día de programación de sala de operaciones

Para poder ingresar a este formulario ese necesario que el usuario tenga el **permiso asignado** tal como se muestra en la Fig. 49.

```
Sub ConfiguraPermisos()
   'PERMISOS
   btnCerrarprogramacion.Enabled = False
   Dim oRsPermisos As New Recordset
   Set oRsPermisos = mo_ReglasSeguridad.UsuariosRolesSeleccionarPermisosTodos(ml_IdUsuario)

   If oRsPermisos.RecordCount > 0 Then
     Do While Not oRsPermisos.EOF
       Select Case oRsPermisos.Fields!idpermiso
       Case 427    'permiso para cerrar / abrir programaciòn qx
             btnCerrarprogramacion.Enabled = True
       End Select
       oRsPermisos.MoveNext
     Loop
   End If

   Set oRsPermisos = Nothing
End Sub
```

Fig. 51: Asignación de permisos para cerrar día de programación

En las Fig. 50 y Fig. 51 se muestra la creación de los procedimientos en la capa negocios **ReglasQx** para abrir y cerrar día de programación.

```
Function CierreDiaQx(lFechaInicio As Date, lidUsuario As Long) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter
Dim oConexion As New ADODB.Connection
Dim ms_MensajeError  As String
    CierreDiaQx = False
   ms_MensajeError = ""
   oConexion.CursorLocation = adUseClient
   oConexion.CommandTimeout = 300
   oConexion.Open sighEntidades.CadenaConexion
   With oCommand
      .CommandType = adCmdStoredProc
      Set .ActiveConnection = oConexion
      .CommandTimeout = 150
      .CommandText = "[usp_update_CierreDiaQx_28022024]"
      Set oParameter = .CreateParameter("@Fecha", adDBTimeStamp, adParamInput, 0, lFechaInicio): .Parame
     Set oParameter = .CreateParameter("@IdUsuario", adInteger, adParamInput, 0, lidUsuario): .Parameters.Ap
      .Execute
  End With
   CierreDiaQx = True
  oConexion.Close
  Set oConexion = Nothing
  Set oCommand = Nothing
Exit Function
ManejadorDeError:
  ms_MensajeError = err.Number & " " + err.Description:
End Function
```

Fig. 52: Función para cierre de día de programación

```
Function AbrirDiaQx(lFechaInicio As Date, lidUsuario As Long) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter
Dim oConexion As New ADODB.Connection
Dim ms_MensajeError  As String
    AbrirDiaQx = False
   ms_MensajeError = ""
   oConexion.CursorLocation = adUseClient
   oConexion.CommandTimeout = 300
   oConexion.Open sighEntidades.CadenaConexion
   With oCommand
      .CommandType = adCmdStoredProc
      Set .ActiveConnection = oConexion
      .CommandTimeout = 150
      .CommandText = "[usp_update_AbrirDiaQx_28022024]"
      Set oParameter = .CreateParameter("@Fecha", adDBTimeStamp, adParamInput, 0, lFechaInicio): .Parame
      Set oParameter = .CreateParameter("@IdUsuario", adInteger, adParamInput, 0, lidUsuario): .Parameters.Ap
      .Execute
  End With
   AbrirDiaQx = True
  oConexion.Close
  Set oConexion = Nothing
  Set oCommand = Nothing
Exit Function
ManejadorDeError:
  ms_MensajeError = err.Number & " " + err.Description:
End Function
```

Fig. 53: Función para apertura de día de programación

En la Fig. 52 se establece las acciones en el formulario para realizar el cierre de Dia de Programación de sala de operaciones y abrir día de programación de Sala para casos de adición y/o modificación de alguna programación.

```
Private Sub btnCerrarQx_Click()
Dim rsRespuesta As New Recordset
   grdAdmision.Caption = "Programacion Qx"
   If sighentidades.EsFecha(txtFI.Text, "DD/MM/AAAA") Then
      Call mo_reglasQx.CierreDiaQx(CDate(txtFI.Text + " 00:00:00"), ml_IdUsuario)
      RealizarBusqueda
      MsgBox "Se cerro el dia para programacion Qx"
      Me.Hide
   Else
      MsgBox "Favor de Ingrese una fecha valida"
   End If
Set rsRespuesta = Nothing
End Sub
```

```
Private Sub btnAbrirProgramacion_Click()
Dim rsRespuesta As New Recordset
   grdAdmision.Caption = "Programacion Qx"
   If sighentidades.EsFecha(txtFI.Text, "DD/MM/AAAA") Then
      Call mo_reglasQx.abrirDiaQx(CDate(txtFI.Text + " 00:00:00"), ml_IdUsuario)
      RealizarBusqueda
      MsgBox "Se abrio el dia para programacion Qx"
      Me.Hide
   Else
      MsgBox "Favor de Ingrese una fecha valida"
   End If
Set rsRespuesta = Nothing
End Sub
```

Fig. 54: Acciones de registro de cierre de programación de sala de operaciones

3.1.4.5 Implementación CU05 Registrar reporte operatorio

El registro del reporte operatorio se realiza después de la cirugía, el médico cirujano es el único que puede generar el registro, para este caso se estableció en base de datos la tablas ReporteOperatorio, ReporteOperatorioDiagnosticos, en tal sentido para el desarrollo se implementaron las clases DOReporteOperatorio.cls y DoReporteOperatorioDiag.cls, para mapear todos los campos de las tablas en base de datos, así mismo se implementaron las clases ADO como ReporteOperatorio.cls y ReporteOperatorioDiagnosticos.cls, encargadas de comunicarse con la base de datos para recibir y enviar información. A continuación, se muestra la clase **DOReporteOperatorio**, como se observa en la Tabla XXXI.

Tabla XXXI: Clase Data Object DOReporteOperatorio

Variables Internas

```
'---------------------------------------------
'Reporte Operatorio
'---------------------------------------------
Option Explicit
Dim ml_Auditoria As Long
Dim ml_idReporteOperatorio As Long
Dim ml_idAprobadoQx As Long
Dim ml_NroReporte  As Long
Dim ml_idCuentaAtencion As Long
Dim ml_idPaciente As Long
Dim mda_FechaCirugia As Date
Dim ms_HoraInicioReal As String
Dim ms_HoraFinReal As String
Dim ms_Procedimiento As String
Dim ms_Hallazgos As String
Dim ms_Incidentes As String
Dim ms_Materiales As String
Dim mb_indAnatomiaPat As Boolean
Dim ml_idRecetaAnatomiaPat As Long
Dim ml_idDestinoAtencion As Long
Dim ml_idMedicoPrincipal As Long
Dim ml_idMedicoAyudante1 As Long
Dim ml_idMedicoAyudante2 As Long
Dim ml_idMedicoAnestesiologo1 As Lo
Dim ml_idMedicoAnestesiologo2 As Lo
Dim ml_idInstrumentista As Long
Dim ml_idCirculante As Long
Dim ml_idEstado As Integer
```

Propiedades de la clase

```
Property Let IdUsuarioAuditoria(lValue As Long)
  ml_Auditoria = lValue
End Property
Property Get IdUsuarioAuditoria() As Long
  IdUsuarioAuditoria = ml_Auditoria
End Property
Property Let idReporteOperatorio(lValue As Long)
  ml_idReporteOperatorio = lValue
End Property
Property Get idReporteOperatorio() As Long
  idReporteOperatorio = ml_idReporteOperatorio
End Property
Property Let FechaCirugia(lValue As Date)
  mda_FechaCirugia = lValue
End Property
Property Get FechaCirugia() As Date
  FechaCirugia = mda_FechaCirugia
End Property
Property Let idAprobadoQx(lValue As Long)
  ml_idAprobadoQx = lValue
End Property
Property Get idAprobadoQx() As Long
  idAprobadoQx = ml_idAprobadoQx
End Property
Property Let NroReporte(lValue As Long)
  ml_NroReporte = lValue
End Property
Property Let idCuentaAtencion(lValue As Long)
  ml_idCuentaAtencion = lValue
End Property
Property Get idCuentaAtencion() As Long
  idCuentaAtencion = ml_idCuentaAtencion
End Property
Property Let idPaciente(lValue As Long)
  ml_idPaciente = lValue
End Property
Property Get idPaciente() As Long
  idPaciente = ml_idPaciente
End Property
Property Let HoraInicioReal(lValue As String)
  ms_HoraInicioReal = lValue
End Property
Property Get HoraInicioReal() As String
  HoraInicioReal = ms_HoraInicioReal
End Property
Property Let HoraFinReal(lValue As String)
  ms_HoraFinReal = lValue
End Property
Property Get HoraFinReal() As String
  HoraFinReal = ms_HoraFinReal
End Property
Property Let idRecetaAnatomiaPat(lValue As Long)
  ml_idRecetaAnatomiaPat = lValue
End Property
```

Propiedades de la clase

```
Property Let Procedimiento(lValue As String)
  ms_Procedimiento = lValue
End Property
Property Get Procedimiento() As String
  Procedimiento = ms_Procedimiento
End Property
Property Let Hallazgos(lValue As String)
  ms_Hallazgos = lValue
End Property
Property Get Hallazgos() As String
  Hallazgos = ms_Hallazgos
End Property
Property Let Incidentes(lValue As String)
  ms_Incidentes = lValue
End Property
Property Get Incidentes() As String
  Incidentes = ms_Incidentes
End Property
Property Let Materiales(lValue As String)
  ms_Materiales = lValue
End Property
Property Get Materiales() As String
  Materiales = ms_Materiales
End Property
Property Let idMedicoPrincipal(lValue As Double)
  ml_idMedicoPrincipal = lValue
End Property
Property Get idMedicoPrincipal() As Double
  idMedicoPrincipal = ml_idMedicoPrincipal
End Property
Property Let idMedicoAyudante1(lValue As Double)
  ml_idMedicoAyudante1 = lValue
End Property
Property Get idMedicoAyudante1() As Double
  idMedicoAyudante1 = ml_idMedicoAyudante1
End Property
Property Let idMedicoAyudante2(lValue As Double)
  ml_idMedicoAyudante2 = lValue
End Property
```

Variables Internas	Propiedades de la clase	Propiedades de la clase
	Property Get idRecetaAnatomiaPat() As Long idRecetaAnatomiaPat = ml_idRecetaAnatomiaF End Property Property Let IndAnatomiaPat(lValue As Boolean) mb_indAnatomiaPat = lValue End Property Property Get IndAnatomiaPat() As Boolean IndAnatomiaPat = mb_indAnatomiaPat End Property Property Let IdDestionAtencion(lValue As Long) ml_idDestinoAtencion = lValue End Property Property Get IdDestionAtencion() As Long IdDestionAtencion = ml_idDestinoAtencion End Property	Property Get idMedicoAnestesiologo1() As Double idMedicoAnestesiologo1 = ml_idMedicoAnestesiologo1 End Property Property Let idMedicoAnestesiologo2(lValue As Double) ml_idMedicoAnestesiologo2 = lValue End Property Property Get idMedicoAnestesiologo2() As Double idMedicoAnestesiologo2 = ml_idMedicoAnestesiologo2 End Property Property Let idInstrumentista(lValue As Double) ml_idInstrumentista = lValue End Property Property Get idInstrumentista() As Double idInstrumentista = ml_idInstrumentista End Property Property Let idCirculante(lValue As Double) ml_idCirculante = lValue End Property Property Get idCirculante() As Double idCirculante = ml_idCirculante End Property Property Let idEstado(lValue As Integer) ml_idEstado = lValue End Property Property Get idEstado() As Integer idEstado = ml_idEstado End Property

Seguidamente se presenta la clase ADO ReporteOperatorio con los principales métodos para conectarse a la base de datos como se muestra en la Tabla XXXII.

Tabla XXXII: Clase ADO reporte operatorio

Propiedades de la clase ReporteOperatorio

```
'--------------------------------------------------------------------------------
'       Programa:  Clase para capa Reporte Operatorio
'--------------------------------------------------------------------------------
Option Explicit
Dim mo_Conexion As ADODB.Connection
Dim ms_MensajeError As String
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Set Conexion(oValue As ADODB.Connection)
  Set mo_Conexion = oValue
End Property
Property Get Conexion() As ADODB.Connection
  Set Conexion = mo_Conexion
End Property
```

Metodo para insertar un registro en la tabla ReporteOperatorio

```
Function Insertar(ByVal oTabla As DoReporteOperatorio) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

Insertar = False
With oCommand
  .CommandType = adCmdStoredProc
  Set .ActiveConnection = mo_Conexion
  .CommandText = "usp_insert_ReporteOperatorioAgregar_09042024"
  Set oParameter = .CreateParameter("@idReporteOperatorio", adInteger, adParamOutput): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@IdAprobadoQx", adInteger, adParamInput, 0, IIf(oTabla.idAprobadoQx = 0, Null, oTabla.idAprobadoQx)): .Par
  Set oParameter = .CreateParameter("@NroReporte", adInteger, adParamInput, 0, IIf(oTabla.NroReporte = 0, Null, oTabla.NroReporte)): .Parameters.
  Set oParameter = .CreateParameter("@idCuentaAtencion", adInteger, adParamInput, 0, IIf(oTabla.idCuentaAtencion = 0, Null, oTabla.idCuentaAtenci
  Set oParameter = .CreateParameter("@idPaciente", adInteger, adParamInput, 0, IIf(oTabla.IDPaciente = 0, Null, oTabla.IDPaciente)): .Parameters.Ap
  Set oParameter = .CreateParameter("@FechaCirugia", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaCirugia = 0, Null, oTabla.FechaCirugia)): .F
  Set oParameter = .CreateParameter("@HoraInicioReal", adVarChar, adParamInput, 5, oTabla.HoraInicioReal): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@HoraFinReal", adVarChar, adParamInput, 5, oTabla.HoraFinReal): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@Procedimiento", adVarChar, adParamInput, 4000, oTabla.Procedimiento): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@Hallazgos", adVarChar, adParamInput, 2000, oTabla.Hallazgos): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@Incidentes", adVarChar, adParamInput, 2000, oTabla.Incidentes): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@Complicaciones", adVarChar, adParamInput, 2000, oTabla.Complicaciones): .Parameters.Append oParamet
  Set oParameter = .CreateParameter("@Materiales", adVarChar, adParamInput, 2000, oTabla.Materiales): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@indAnatomiaPat", adBoolean, adParamInput, 0, IIf(oTabla.IndAnatomiaPat = False, 0, 1)): .Parameters.Appen
  Set oParameter = .CreateParameter("@DescripcionAnatomiaPat", adVarChar, adParamInput, 4000, oTabla.DescripcionAnatomiaPat): .Parameters.A
  Set oParameter = .CreateParameter("@idRecetaAnatomiaPat", adInteger, adParamInput, 0, IIf(oTabla.idRecetaAnatomiaPat = 0, Null, oTabla.idRece
  Set oParameter = .CreateParameter("@idDestinoAtencion", adInteger, adParamInput, 0, IIf(oTabla.IdDestionAtencion = 0, Null, oTabla.IdDestionAter
  Set oParameter = .CreateParameter("@idMedicoPrincipal", adInteger, adParamInput, 0, IIf(oTabla.idMedicoPrincipal = 0, Null, oTabla.idMedicoPrinci
```

```
    Set oParameter = .CreateParameter("@idMedicoAyudante1", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante1 = 0, Null, oTabla.idMedicoA
    Set oParameter = .CreateParameter("@idMedicoAyudante2", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante2 = 0, Null, oTabla.idMedicoA
    Set oParameter = .CreateParameter("@idMedicoAnestesiologo1", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAnestesiologo1 = 0, Null, oTabla.i
    Set oParameter = .CreateParameter("@idMedicoAnestesiologo2", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAnestesiologo2 = 0, Null, oTabla.i
    Set oParameter = .CreateParameter("@idInstrumentista", adInteger, adParamInput, 0, IIf(oTabla.idInstrumentista = 0, Null, oTabla.idInstrumentista)): .
    Set oParameter = .CreateParameter("@idCirculante", adInteger, adParamInput, 0, IIf(oTabla.idCirculante = 0, Null, oTabla.idCirculante)): .Parameters.
    Set oParameter = .CreateParameter("@IdEstado", adInteger, adParamInput, 0, IIf(oTabla.IdEstado = 0, Null, oTabla.IdEstado)): .Parameters.Append
    Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParamete
  .Execute
     oTabla.idReporteOperatorio = .Parameters("@idReporteOperatorio")
End With

Insertar = True
ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbI
End Function
```

Metodo para modificar un registro en la tabla ReporteOperatorio

```
Function Modificar(ByVal oTabla As DoReporteOperatorio) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

Modificar = False
With oCommand
.CommandType = adCmdStoredProc
Set .ActiveConnection = mo_Conexion
.CommandText = "usp_update_ReporteOperatorioModificar_09042024"
   Set oParameter = .CreateParameter("@IdReporteOperatorio", adInteger, adParamInput, 0, oTabla.idReporteOperatorio): .Parameters.Append oPara
   Set oParameter = .CreateParameter("@IdAprobadoQx", adInteger, adParamInput, 0, IIf(oTabla.idAprobadoQx = 0, Null, oTabla.idAprobadoQx)): .Pa
   Set oParameter = .CreateParameter("@NroReporte", adInteger, adParamInput, 0, IIf(oTabla.NroReporte = 0, Null, oTabla.NroReporte)): .Parameters
   Set oParameter = .CreateParameter("@idCuentaAtencion", adInteger, adParamInput, 0, IIf(oTabla.idCuentaAtencion = 0, Null, oTabla.idCuentaAtenc
   Set oParameter = .CreateParameter("@idPaciente", adInteger, adParamInput, 0, IIf(oTabla.IDPaciente = 0, Null, oTabla.IDPaciente)): .Parameters.A
   Set oParameter = .CreateParameter("@FechaCirugia", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaCirugia = 0, Null, oTabla.FechaCirugia)): .F
   Set oParameter = .CreateParameter("@HoraInicioReal", adVarChar, adParamInput, 5, oTabla.HoraInicioReal): .Parameters.Append oParameter
   Set oParameter = .CreateParameter("@HoraFinReal", adVarChar, adParamInput, 5, oTabla.HoraFinReal): .Parameters.Append oParameter
   Set oParameter = .CreateParameter("@Procedimiento", adVarChar, adParamInput, 4000, oTabla.Procedimiento): .Parameters.Append oParameter
   Set oParameter = .CreateParameter("@Hallazgos", adVarChar, adParamInput, 2000, oTabla.Hallazgos): .Parameters.Append oParameter
   Set oParameter = .CreateParameter("@Incidentes", adVarChar, adParamInput, 2000, oTabla.Incidentes): .Parameters.Append oParameter
   Set oParameter = .CreateParameter("@Complicaciones", adVarChar, adParamInput, 2000, oTabla.Complicaciones): .Parameters.Append oParamet
   Set oParameter = .CreateParameter("@Materiales", adVarChar, adParamInput, 2000, oTabla.Materiales): .Parameters.Append oParameter
   Set oParameter = .CreateParameter("@indAnatomiaPat", adBoolean, adParamInput, 0, IIf(oTabla.IndAnatomiaPat = False, 0, 1)): .Parameters.Apper
   Set oParameter = .CreateParameter("@DescripcionAnatomiaPat", adVarChar, adParamInput, 4000, oTabla.DescripcionAnatomiaPat): .Parameters.A
   Set oParameter = .CreateParameter("@idRecetaAnatomiaPat", adInteger, adParamInput, 0, IIf(oTabla.idRecetaAnatomiaPat = 0, Null, oTabla.idRece
   Set oParameter = .CreateParameter("@idDestinoAtencion", adInteger, adParamInput, 0, IIf(oTabla.IdDestionAtencion = 0, Null, oTabla.IdDestionAte
   Set oParameter = .CreateParameter("@idMedicoPrincipal", adInteger, adParamInput, 0, IIf(oTabla.idMedicoPrincipal = 0, Null, oTabla.idMedicoPrinci
   Set oParameter = .CreateParameter("@idMedicoAyudante1", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante1 = 0, Null, oTabla.idMedicoA
   Set oParameter = .CreateParameter("@idMedicoAyudante2", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAyudante2 = 0, Null, oTabla.idMedicoA
   Set oParameter = .CreateParameter("@idMedicoAnestesiologo1", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAnestesiologo1 = 0, Null, oTabla.i
   Set oParameter = .CreateParameter("@idMedicoAnestesiologo2", adInteger, adParamInput, 0, IIf(oTabla.idMedicoAnestesiologo2 = 0, Null, oTabla.i
   Set oParameter = .CreateParameter("@idInstrumentista", adInteger, adParamInput, 0, IIf(oTabla.idInstrumentista = 0, Null, oTabla.idInstrumentista)): .
   Set oParameter = .CreateParameter("@idCirculante", adInteger, adParamInput, 0, IIf(oTabla.idCirculante = 0, Null, oTabla.idCirculante)): .Parameters.
   Set oParameter = .CreateParameter("@IdEstado", adInteger, adParamInput, 0, IIf(oTabla.IdEstado = 0, Null, oTabla.IdEstado)): .Parameters.Append
   Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParamete
.Execute
End With

Modificar = True
ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vbI
End Function
```

Metodo para Seleccionar un registro en la tabla ReporteOperatorio

```
Function SeleccionarPorId(ByVal oTabla As DoReporteOperatorio) As Boolean
On Error GoTo ManejadorDeError
Dim oRecordset As New ADODB.Recordset
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  SeleccionarPorId = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_select_ReporteOperatorioSeleccionarPorId_09042024"
      Set oParameter = .CreateParameter("@IdReporte", adInteger, adParamInput, 0, oTabla.idReporteOperatorio): .
    Set oRecordset = .Execute
  End With

  If Not (oRecordset.EOF And oRecordset.BOF) Then
     SeleccionarPorId = True
     oTabla.idReporteOperatorio = IIf(IsNull(oRecordset!idReporteOperatorio), 0, oRecordset!idReporteOperatorio
     oTabla.idAprobadoQx = IIf(IsNull(oRecordset!idAprobadoQx), 0, oRecordset!idAprobadoQx)
     oTabla.idCuentaAtencion = IIf(IsNull(oRecordset!idCuentaAtencion), 0, oRecordset!idCuentaAtencion)
     oTabla.IDPaciente = IIf(IsNull(oRecordset!IDPaciente), 0, oRecordset!IDPaciente)
     oTabla.FechaCirugia = IIf(IsNull(oRecordset!FechaCirugia), 0, oRecordset!FechaCirugia)
     oTabla.HoraInicioReal = IIf(IsNull(oRecordset!HoraInicioReal), "", oRecordset!HoraInicioReal)
     oTabla.HoraFinReal = IIf(IsNull(oRecordset!HoraFinReal), "", oRecordset!HoraFinReal)
      oTabla.Procedimiento = IIf(IsNull(oRecordset!Procedimiento), "", oRecordset!Procedimiento)
      oTabla.Hallazgos = IIf(IsNull(oRecordset!Hallazgos), "", oRecordset!Hallazgos)
      oTabla.Incidentes = IIf(IsNull(oRecordset!Incidentes), "", oRecordset!Incidentes)
      oTabla.Complicaciones = IIf(IsNull(oRecordset!Complicaciones), "", oRecordset!Complicaciones)
```

```
        oTabla.idMedicoAnestesiologo1 = IIf(IsNull(oRecordset!idMedicoAnestesiologo1), 0, oRecordset!idMedicoAnestesiologo1)
        oTabla.idMedicoAnestesiologo2 = IIf(IsNull(oRecordset!idMedicoAnestesiologo2), 0, oRecordset!idMedicoAnestesiologo2)
        oTabla.idCirculante = IIf(IsNull(oRecordset!idCirculante), 0, oRecordset!idCirculante)
        oTabla.idInstrumentista = IIf(IsNull(oRecordset!idInstrumentista), 0, oRecordset!idInstrumentista)
        oTabla.IdEstado = IIf(IsNull(oRecordset!IdEstado), 1, oRecordset!IdEstado)
    Else
       SeleccionarPorId = False
    End If

    oRecordset.Close
    ms_MensajeError = ""

Exit Function
ManejadorDeError:
    ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de sop
End Function
```

Para el desarrollo de la interface se estableció el siguiente formulario para el registro de Reporte Operatorio como se observa en la Fig. 55.

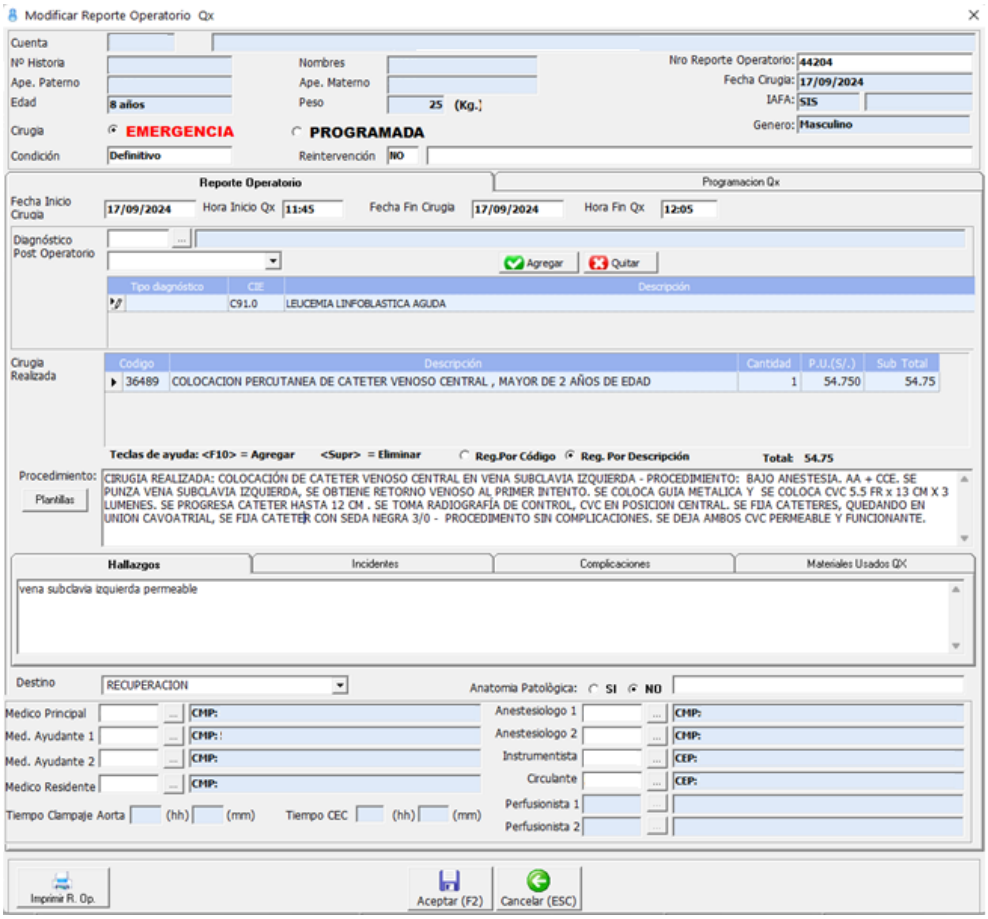


Fig. 55: Formulario de registro de reporte operatorio

A continuación, en la Tabla XXXIII se presenta la codificación del formulario para el registro de reporte operatorio.

Tabla XXXIII: Codificación de formulario de reporte operatorio

Variables globales del formulario

```
'-----------------------------------------
'Formulario Reporte Operatorio
'-----------------------------------------
Option Explicit
Dim mo_Teclado As New sighentidades.Teclado
Dim mo_Formulario As New sighentidades.formulario
Dim oReporteOperatorio As New sighdatos.ReporteOperatorio
Dim odoReporteOperatorio As New SIGHComun.DOReporteOpe
Dim ml_IdUsuario As Long
Dim ms_MensajeError As String
Dim mi_Opcion As sghOpciones
Dim ml_Id As Long
Dim lcBuscaParametro As New sighdatos.Parametros
Dim mo_AdminComun As New ReglasComunes
Dim mo_reglasQx As New SIGHNegocios.ReglasQx
Dim do_paciente As New SIGHComun.doPaciente
Dim mo_ReglasFacturacion As New SIGHNegocios.ReglasFactu
Dim mo_Apariencia As New sighentidades.GridInfragistic
Dim mo_lnIdTablaLISTBARITEMS As Long
Dim mo_lcNombrePc As String
Dim mo_Diagnosticos As New Collection
Dim mo_CPTs As New Collection
Dim mo_cmbTurno As New ListaDesplegable
Dim mo_cmbTipoAnestesia As New ListaDesplegable
Dim mo_cmbTipoRiesgoQuirurgico As New ListaDesplegable
Dim mo_cmbIdServicioOpera As New ListaDesplegable
Dim mo_cmbIdDestinoAtencion As New ListaDesplegable
Dim mo_cmbIdSala As New ListaDesplegable
```

Propiedades del Formulario

```
Property Let lcNombrePc(lValue As String)
  mo_lcNombrePc = lValue
End Property
Property Let lnIdTablaLISTBARITEMS(lValue As Long)
  mo_lnIdTablaLISTBARITEMS = lValue
End Property
Property Let Opcion(iValue As sghOpciones)
  mi_Opcion = iValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Let IdUsuario(lValue As Long)
  ml_IdUsuario = lValue
End Property
Property Get IdUsuario() As Long
  IdUsuario = ml_IdUsuario
End Property
Property Let Id(lValue As Long)
  ml_Id = lValue
End Property
Property Get Id() As Long
  Id = ml_Id
End Property
```

Metodo para cargar los datos de base a datos a los controles del formulario

```
Sub CargarDatosALosControles()
  Dim rsop As New Recordset
  Set rsop = mo_reglasQx.ReporteOperatorioCompletoPorId(ml_idReporteOperatorio)

 If rsop.RecordCount > 0 Then
    txtNroReporte.Text = rsop.Fields!NroReporte
    txtHoraInicioReal.Text = rsop.Fields!HoraInicioReal
    txtHoraFinReal.Text = rsop.Fields!HoraFinReal
    txtFechaCirugia.Text = rsop.Fields!fechacirugia
    txtFechaFinCirugia.Text = IIf(IsNull(rsop.Fields!FechaFinCirugia), IIf(IsNull(rsop.Fields!fechacirugia), "", rsop.Field
    txtFechaAProbado.Text = rsop.Fields!FechaProgramada
    Me.txtProcedimiento = rsop.Fields!Procedimiento
    Me.txtHallazgos.Text = rsop.Fields!Hallazgos
    Me.txtIncidentes.Text = rsop.Fields!Incidentes
    txtComplicaciones.Text = rsop.Fields!Complicaciones
    Me.txtMateriales.Text = rsop.Fields!Materiales
    Me.txtDescripcionAnatomiaPat.Text = IIf(IsNull(rsop.Fields!DescripcionAnatomiaPat), "", rsop.Fields!DescripcionA
    mo_cmbIdDestinoAtencion.BoundText = IIf(IsNull(rsop.Fields!idDestinoAtencion), 0, rsop.Fields!idDestinoAtencic

    txtDNIMedicoPrincipal.Tag = IIf(IsNull(rsop.Fields!idMedicoPrincipal), 0, rsop.Fields!idMedicoPrincipal)
    txtDNIMedicoPrincipal.Text = "CMP: " & IIf(IsNull(rsop.Fields!cmpmep), "", rsop.Fields!cmpmep) & " " & IIf(IsNull(r
    txtMedicoPrincipal.Text = IIf(IsNull(rsop.Fields!cmpmep), "", rsop.Fields!cmpmep)

    txtDNIMedicoAyudante1.Tag = IIf(IsNull(rsop.Fields!idMedicoAyudante1), 0, rsop.Fields!idMedicoAyudante1)
    txtDNIMedicoAyudante1.Text = "CMP: " & IIf(IsNull(rsop.Fields!cmpAyu1), "", rsop.Fields!cmpAyu1) & " " & IIf(IsN
    txtMedicoAyudante1.Text = IIf(IsNull(rsop.Fields!cmpAyu1), "", rsop.Fields!cmpAyu1)
    txtDNIMedicoAyudante2.Tag = IIf(IsNull(rsop.Fields!idMedicoAyudante2), 0, rsop.Fields!idMedicoAyudante2)
    txtDNIMedicoAyudante2.Text = "CMP: " & IIf(IsNull(rsop.Fields!cmpAyu2), "", rsop.Fields!cmpAyu2) & " " & IIf(IsNull(rsop.Fields!M
    txtMedicoAyudante2.Text = IIf(IsNull(rsop.Fields!cmpAyu2), "", rsop.Fields!cmpAyu2)

    txtDNIMedicoAnestesiologo1.Tag = IIf(IsNull(rsop.Fields!idMedicoAnestesiologo1), 0, rsop.Fields!idMedicoAnestesiologo1)
    txtDNIMedicoAnestesiologo1.Text = "CMP:" & IIf(IsNull(rsop.Fields!cmpane1), "", rsop.Fields!cmpane1) & " " & IIf(IsNull(rsop.Fiel
    txtMedicoAnestesiolo1.Text = IIf(IsNull(rsop.Fields!cmpane1), "", rsop.Fields!cmpane1)

    txtDNIMedicoAnestesiologo2.Tag = IIf(IsNull(rsop.Fields!idMedicoAnestesiologo2), 0, rsop.Fields!idMedicoAnestesiologo2)
    txtDNIMedicoAnestesiologo2.Text = "CMP: " & IIf(IsNull(rsop.Fields!cmpane2), "", rsop.Fields!cmpane2) & " " & IIf(IsNull(rsop.Fiel
    txtMedicoAnestesiolo2.Text = IIf(IsNull(rsop.Fields!cmpane2), "", rsop.Fields!cmpane2)

    txtDNIInstrumentista.Tag = IIf(IsNull(rsop.Fields!idInstrumentista), 0, rsop.Fields!idInstrumentista)
    txtDNIInstrumentista.Text = "CEP: " & IIf(IsNull(rsop.Fields!cmpinst), "", rsop.Fields!cmpinst) & " " & IIf(IsNull(rsop.Fields!Instrume
    txtInstrumentista.Text = IIf(IsNull(rsop.Fields!cmpinst), "", rsop.Fields!cmpinst)
    txtDNICirculante.Tag = IIf(IsNull(rsop.Fields!idCirculante), 0, rsop.Fields!idCirculante)
    txtDNICirculante.Text = "CEP: " & IIf(IsNull(rsop.Fields!cmpcir), "", rsop.Fields!cmpcir) & " " & IIf(IsNull(rsop.Fields!Circulante), "", r
    txtCirculante.Text = IIf(IsNull(rsop.Fields!cmpcir), "", rsop.Fields!cmpcir)


  End If

End Sub
```

Metodo para cargar los datos de los controles del formulario al objeto para enviar a base datos

```
Sub CargaDatosAlObjetosDeDatos()

With odoReporteOperatorio
   .idCuentaAtencion = ml_idCuentaAtencion
   .IdPaciente = ml_IdPaciente
   .NroReporte = val(txtNroReporte.Text)
   .fechacirugia = txtFechaCirugia.Text
   .FechaFinCirugia = txtFechaFinCirugia.Text
   .idAprobadoQx = ml_Id
   .idReporteOperatorio = ml_idReporteOperatorio
   .HoraInicioReal = txtHoraInicioReal.Text
   .HoraFinReal = txtHoraFinReal.Text
   .Procedimiento = txtProcedimiento
   .Hallazgos = txtHallazgos.Text
   .Incidentes = txtIncidentes.Text
   .Complicaciones = txtComplicaciones.Text
   .Materiales = txtMateriales.Text
   .DescripcionAnatomiaPat = txtDescripcionAnatomiaPat.Text
   .IdDestionAtencion = val(mo_cmbIdDestinoAtencion.BoundText)
   .idMedicoPrincipal = val(txtDNIMedicoPrincipal.Tag)
   .idMedicoAyudante1 = val(txtDNIMedicoAyudante1.Tag)
   .idMedicoAyudante2 = val(txtDNIMedicoAyudante2.Tag)
   .idMedicoResidente = val(txtDNIMedicoResidente.Tag)
   .TiempoClampajeAorta = val(Me.txtHorClampajeAorta.Text) * 60 + val(Me.txtMinClampajeAorta.Text)
   .TiempoCEC = val(Me.txtHorCEC.Text) * 60 + val(Me.txtminCEC.Text)
   .idMedicoAnestesiologo1 = val(txtDNIMedicoAnestesiologo1.Tag)
   .idMedicoAnestesiologo2 = val(txtDNIMedicoAnestesiologo2.Tag)
    .idCirculante = val(txtDNICirculante.Tag)
    .idInstrumentista = val(txtDNIInstrumentista.Tag)
    .IdUsuarioAuditoria = ml_IdUsuario
    .idEstado = 1
  End With

With mo_DOFactOrdenServicio
   .FechaDespacho = txtFechaCirugia.Text
   .FechaHoraRealizaCpt = lcBuscaParametro.RetornaFechaHoraServidorSQL
   .fechaCreacion = lcBuscaParametro.RetornaFechaHoraServidorSQL
   .idCuentaAtencion = ml_idCuentaAtencion
   .IdPaciente = ml_IdPaciente
   .IdFuenteFinanciamiento = ml_idFuenteFinanciamiento
   .IdTipoFinanciamiento = ml_IdTipoFinanciamiento
   .idPuntoCarga = 1
   .IdUsuarioAuditoria = ml_IdUsuario
   .IdUsuarioDespacho = ml_IdUsuario
   .IdUsuario = ml_IdUsuario
   .IdServicioPaciente = ml_IdServicioPaciente
   If mi_Opcion = sghAgregar Then
   .Idestadofacturacion = 1
   End If
End With

Me.ucDiagnosticoReporteOperQx1.idAtencion = ml_IdAtencion
Set mo_Diagnosticos = Nothing
ucDiagnosticoReporteOperQx1.CargarDiagnosticosAlObjetoDatos mo_Diagnosticos
```

Metodo para agregar un registro a la base datos

```
Function AgregarDatos() As Boolean
    Dim oConexion As New ADODB.Connection
    Dim oReporteDiagnostico As New ReporteOperatorioDiagnosticos

    oConexion.Open sighentidades.CadenaConexion
    oConexion.CursorLocation = adUseClient
    CargaDatosAlObjetosDeDatos

  Set oReporteOperatorio.Conexion = oConexion
  AgregarDatos = oReporteOperatorio.Insertar(odoReporteOperatorio)

  If odoReporteOperatorio.idReporteOperatorio > 0 Then
      ml_idReporteOperatorio = odoReporteOperatorio.idReporteOperatorio

      Set oReporteDiagnostico.Conexion = oConexion
      If Not oReporteDiagnostico.ActualizarDiagnosticosReporteOperatorio(mo_Diagnosticos, _
      odoReporteOperatorio.idReporteOperatorio) Then
          ms_MensajeError = "Dx: " & oReporteDiagnostico.MensajeError: GoTo ErrorManager
      End If
      'Inicio facturacion
      Call mo_ReglasFacturacion.FactOrdenServicioAgregar(mo_DOFactOrdenServicio, _
      Me.ucFacturacionItems1.FacturacionProductos, mo_lnIdTablaLISTBARITEMS, _
      mo_lcNombrePc, txtApellidoPaterno.Text + " " + txtApellidoMaterno + ", " + txtNombres, _
      ml_IdServicioPaciente, mo_Diagnosticos)
      mo_ReglasFacturacion.FacturacionCuentasAtencionPtosActualizar mo_DOFactOrdenServicio.idCuentaAtencion, False, 0
      ml_IdOrden = mo_DOFactOrdenServicio.Idorden
      Call mo_reglasQx.enlazarReporteOperatorioFacturacion(mo_DOFactOrdenServicio.Idorden, _
      odoReporteOperatorio.idReporteOperatorio, oConexion)
      'Fin Facturacion

  Else
      mo_AdminComun.MensajeError = "Error, no se pudo registrar en la tabla."
      AgregarDatos = False
  End If

  oConexion.Close
  Set oConexion = Nothing

ErrorManager:

End Function
```

Metodo para modificar un registro a la base datos

```
Function ModificarDatos() As Boolean
Dim oConexion As New ADODB.Connection
Dim oReporteDiagnostico As New ReporteOperatorioDiagnosticos

   oConexion.Open sighentidades.CadenaConexion
   oConexion.CursorLocation = adUseClient
   Set oReporteOperatorio.Conexion = oConexion

  CargaDatosAlObjetosDeDatos

   Set oReporteDiagnostico.Conexion = oConexion
   If Not oReporteDiagnostico.ActualizarDiagnosticosReporteOperatorio(mo_Diagnosticos, odoReporteOperatorio.idReporteOperatorio) The
      ms_MensajeError = "Dx: " & oReporteDiagnostico.MensajeError: GoTo ErrorManager
   End If

  ModificarDatos = oReporteOperatorio.Modificar(odoReporteOperatorio)

   'Actualiza Facturacion
  Call mo_ReglasFacturacion.FactOrdenServicioModificar(mo_DOFactOrdenServicio, Me.ucFacturacionItems1.FacturacionProductos, mo_lr
  mo_ReglasFacturacion.FacturacionCuentasAtencionPtosActualizar mo_DOFactOrdenServicio.idCuentaAtencion, False, 0
  'Fin Actualiza Facturacion

 oConexion.Close
  Set oConexion = Nothing

ErrorManager:
End Function
```

Metodo para eliminar un registro a la base datos

```
Function EliminarDatos() As Boolean
Dim oConexion As New ADODB.Connection

   oConexion.Open sighentidades.CadenaConexion
   oConexion.CursorLocation = adUseClient
   Set oReporteOperatorio.Conexion = oConexion

  CargaDatosAlObjetosDeDatos
  EliminarDatos = oReporteOperatorio.Eliminar(odoReporteOperatorio)

  'Inicio facturacion
 Call mo_ReglasFacturacion.FactOrdenServicioEliminar(mo_DOFactOrdenServicio, mo_lnIdTablaLISTBARITEMS, mo_lcNombrePc, txtApe
 mo_ReglasFacturacion.FacturacionCuentasAtencionPtosActualizar mo_DOFactOrdenServicio.idCuentaAtencion, False, 0
'fin facturacion

 oConexion.Close
  Set oConexion = Nothing
Exit Function
ErrorManager:

End Function
```

3.1.4.6 Implementación CU06 Registro de solicitud de sala de operaciones

Para la suspensión de sala de operación se estableció crear una tabla en base de datos SuspensionesQx, para guardar todas las suspensiones de sala de operaciones.

En el desarrollo se implementaron dos clases DOSuspencionQx y SuspensionQx, con el objetivo de realizar un registro eficiente.

A continuación, se muestra la clase DOSuspensionQx donde se visualiza las propiedades que representan los campos de la Tabla XXXIV SuspensionQx.

Tabla XXXIV: Clase Data Object SuspensionQx

Variables internas

```
'-------------------------------------------------------------
'Suspensiones de Sala de Operaciones
'-------------------------------------------------------------
Option Explicit

Dim ml_Auditoria As Long
Dim ml_idSuspension As Long
Dim ml_idAprobadoQx As Long
Dim ml_NroSuspension  As Long
Dim ml_idCuentaAtencion As Long
Dim ml_idPaciente As Long
Dim mda_FechaSuspende As Date
Dim ms_HoraSuspende As String
Dim ms_Observacion As String
Dim ml_IdMotivo As Long
Dim ml_idMedicoSuspende As Long
Dim ml_idEstado As Integer
Property Let IdUsuarioAuditoria(lValue As Long)
  ml_Auditoria = lValue
End Property
Property Get IdUsuarioAuditoria() As Long
  IdUsuarioAuditoria = ml_Auditoria
End Property
Property Let idSuspension(lValue As Long)
  ml_idSuspension = lValue
End Property
```

Propiedades de clase

```
Property Get idSuspension() As Long
  idSuspension = ml_idSuspension
End Property
Property Let FechaSuspende(lValue As Date)
  mda_FechaSuspende = lValue
End Property
Property Get FechaSuspende() As Date
  FechaSuspende = mda_FechaSuspende
End Property
Property Let idAprobadoQx(lValue As Long)
  ml_idAprobadoQx = lValue
End Property
Property Get idAprobadoQx() As Long
  idAprobadoQx = ml_idAprobadoQx
End Property
Property Let NroSuspension(lValue As Long)
  ml_NroSuspension = lValue
End Property
Property Get NroSuspension() As Long
  NroSuspension = ml_NroSuspension
End Property
Property Let idCuentaAtencion(lValue As Long)
  ml_idCuentaAtencion = lValue
End Property
Property Get idCuentaAtencion() As Long
  idCuentaAtencion = ml_idCuentaAtencion
End Property
```

Propiedades de clase

```
Property Let idPaciente(lValue As Long)
  ml_idPaciente = lValue
End Property
Property Get idPaciente() As Long
  idPaciente = ml_idPaciente
End Property
Property Let HoraSuspende(lValue As String)
  ms_HoraSuspende = lValue
End Property
Property Get HoraSuspende() As String
  HoraSuspende = ms_HoraSuspende
End Property
Property Let IdMotivo(lValue As Long)
  ml_IdMotivo = lValue
End Property
Property Get IdMotivo() As Long
  IdMotivo = ml_IdMotivo
End Property
Property Let Observacion(lValue As String)
  ms_Observacion = lValue
End Property
Property Get Observacion() As String
  Observacion = ms_Observacion
End Property
Property Let idMedicoSuspende(lValue As Double)
  ml_idMedicoSuspende = lValue
End Property

Property Get idMedicoSuspende() As Double
  idMedicoSuspende = ml_idMedicoSuspende
End Property
Property Let idEstado(lValue As Integer)
  ml_idEstado = lValue
End Property
Property Get idEstado() As Integer
  idEstado = ml_idEstado
End Property
```

En la Tabla XXXV se muestra la implementación de la clase SuspensionQx la cual contiene los métodos de insertar, modificar, eliminar un registro en base de datos.

Tabla XXXV: Clase ADO SuspensiónQx

Declaracion de variables y propiedades de la clase

```
'-------------------------------------------------------------
'      Programa:  Clase ADO Suspension Qx
'-------------------------------------------------------------
Option Explicit
Dim mo_Conexion As ADODB.Connection
Dim ms_MensajeError As String
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Set Conexion(oValue As ADODB.Connection)
  Set mo_Conexion = oValue
End Property
Property Get Conexion() As ADODB.Connection
  Set Conexion = mo_Conexion
End Property
```

Metodo para insertar un registro en la tabla SuspensionQx

```
Function Insertar(ByVal oTabla As DoSuspensionQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Insertar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
     .CommandText = "usp_insert_SuspensionQxAgregar_28022024"
     Set oParameter = .CreateParameter("@idSuspension", adInteger, adParamOutput): .Parameters.Append oParameter
     Set oParameter = .CreateParameter("@IdAprobadoQx", adInteger, adParamInput, 0, IIf(oTabla.idAprobadoQx = 0, Null, oTabla.idAprobadoQx)): .
```

```
    Set oParameter = .CreateParameter("@NroSuspension", adInteger, adParamInput, 0, IIf(oTabla.NroSuspension = 0, Null, oTabla.NroSuspension))
    Set oParameter = .CreateParameter("@idCuentaAtencion", adInteger, adParamInput, 0, IIf(oTabla.idCuentaAtencion = 0, Null, oTabla.idCuentaAte
    Set oParameter = .CreateParameter("@idPaciente", adInteger, adParamInput, 0, IIf(oTabla.IDPaciente = 0, Null, oTabla.IDPaciente)): .Parameters
    Set oParameter = .CreateParameter("@FechaSuspende", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaSuspende = 0, Null, oTabla.FechaS
    Set oParameter = .CreateParameter("@HoraSuspende", adVarChar, adParamInput, 5, oTabla.HoraSuspende): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idMotivo", adInteger, adParamInput, 0, IIf(oTabla.IdMotivo = 0, Null, oTabla.IdMotivo)): .Parameters.Appen
    Set oParameter = .CreateParameter("@Observacion", adVarChar, adParamInput, 4000, oTabla.observacion): .Parameters.Append oParameter
    Set oParameter = .CreateParameter("@idMedicoSuspende", adInteger, adParamInput, 0, IIf(oTabla.idMedicoSuspende = 0, Null, oTabla.idMedicc
    Set oParameter = .CreateParameter("@IdEstado", adInteger, adParamInput, 0, IIf(oTabla.IdEstado = 0, Null, oTabla.IdEstado)): .Parameters.Appe
    Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParam

   .Execute
     oTabla.idSuspension = .Parameters("@idSuspension")
 End With

 Insertar = True
 ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico", vb
Exit Function
End Function
```

Metodo para modificar un registro en la tabla SuspensionQx

```
Function Modificar(ByVal oTabla As DoSuspensionQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Modificar = False
  With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
     .CommandText = "[usp_update_SuspensionQxModificar_03092024]"
      Set oParameter = .CreateParameter("@IdSuspension", adInteger, adParamInput, 0, oTabla.idSuspension): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@IdAprobadoQx", adInteger, adParamInput, 0, IIf(oTabla.idAprobadoQx = 0, Null, oTabla.idAprobadoQx)): .
      Set oParameter = .CreateParameter("@NroSuspension", adInteger, adParamInput, 0, IIf(oTabla.NroSuspension = 0, Null, oTabla.NroSuspension))
      Set oParameter = .CreateParameter("@idCuentaAtencion", adInteger, adParamInput, 0, IIf(oTabla.idCuentaAtencion = 0, Null, oTabla.idCuentaAte
      Set oParameter = .CreateParameter("@idPaciente", adInteger, adParamInput, 0, IIf(oTabla.IDPaciente = 0, Null, oTabla.IDPaciente)): .Parameters
      Set oParameter = .CreateParameter("@FechaSuspende", adDBTimeStamp, adParamInput, , IIf(oTabla.FechaSuspende = 0, Null, oTabla.FechaS
      Set oParameter = .CreateParameter("@HoraSuspende", adVarChar, adParamInput, 5, oTabla.HoraSuspende): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@idMotivo", adInteger, adParamInput, 0, IIf(oTabla.IdMotivo = 0, Null, oTabla.IdMotivo)): .Parameters.Appen
      Set oParameter = .CreateParameter("@Observacion", adVarChar, adParamInput, 4000, oTabla.observacion): .Parameters.Append oParameter
      Set oParameter = .CreateParameter("@idMedicoSuspende", adInteger, adParamInput, 0, IIf(oTabla.idMedicoSuspende = 0, Null, oTabla.idMedicc
      Set oParameter = .CreateParameter("@IdEstado", adInteger, adParamInput, 0, IIf(oTabla.IdEstado = 0, Null, oTabla.IdEstado)): .Parameters.Appe
      Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oParam

     .Execute
  End With
  Modificar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) _
  + "Por favor contacte al personal de soporte técnico", vbInformation, "Error en la interface de acceso a datos"
End Function
```

Metodo para eliminar un registro en la tabla SuspensionQx

```
Function Eliminar(ByVal oTabla As DoSuspensionQx) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Eliminar = False
  With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
     .CommandText = "usp_delete_SuspensionQxEliminar_28022024"
        Set oParameter = .CreateParameter("@IdSuspension", adInteger, adParamInput, 0, IIf(oTabla.idSuspension = 0, Null, oTabla.idSuspension)):
     Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria)
     .Parameters.Append oParameter
     .Execute
  End With

  Eliminar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description: MsgBox ms_MensajeError + Chr(13) + "Por favor contacte al personal de soporte técnico",
Exit Function
End Function
```

Metodo para Seleccionar un registro en la tabla SuspensionQx

```
Function SeleccionarPorId(ByVal oTabla As DoSuspensionQx) As Boolean
On Error GoTo ManejadorDeError
Dim oRecordset As New ADODB.Recordset
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  SeleccionarPorId = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_select_SuspensionQxSeleccionarPorId_03092024"
      Set oParameter = .CreateParameter("@IdSuspension", adInteger, adParamInput, 0, oTabla.idSuspension): .Parameters.Append oParameter
    Set oRecordset = .Execute
  End With

  If Not (oRecordset.EOF And oRecordset.BOF) Then
      SeleccionarPorId = True
      oTabla.idSuspension = IIf(IsNull(oRecordset!idSuspension), 0, oRecordset!idSuspension)
      oTabla.idAprobadoQx = IIf(IsNull(oRecordset!idAprobadoQx), 0, oRecordset!idAprobadoQx)
      oTabla.idCuentaAtencion = IIf(IsNull(oRecordset!idCuentaAtencion), 0, oRecordset!idCuentaAtencion)
      oTabla.IDPaciente = IIf(IsNull(oRecordset!IDPaciente), 0, oRecordset!IDPaciente)
      oTabla.FechaSuspende = IIf(IsNull(oRecordset!FechaSuspende), 0, oRecordset!FechaSuspende)
      oTabla.HoraSuspende = IIf(IsNull(oRecordset!HoraSuspende), "", oRecordset!HoraSuspende)
     oTabla.observacion = IIf(IsNull(oRecordset!observacion), "", oRecordset!observacion)
      oTabla.IdMotivo = IIf(IsNull(oRecordset!IdMotivo), 0, oRecordset!IdMotivo)
      oTabla.idMedicoSuspende = IIf(IsNull(oRecordset!idMedicoSuspende), 0, oRecordset!idMedicoSuspende)
    oTabla.IdEstado = IIf(IsNull(oRecordset!IdEstado), 1, oRecordset!IdEstado)
```

Para el desarrollo de la interface en el registro se Suspensión de sala de operaciones se estableció el siguiente formulario, el cual permite obtener información como: responsable, tipo y motivo de suspensión de cirugía, como se muestra en la Fig. 56.

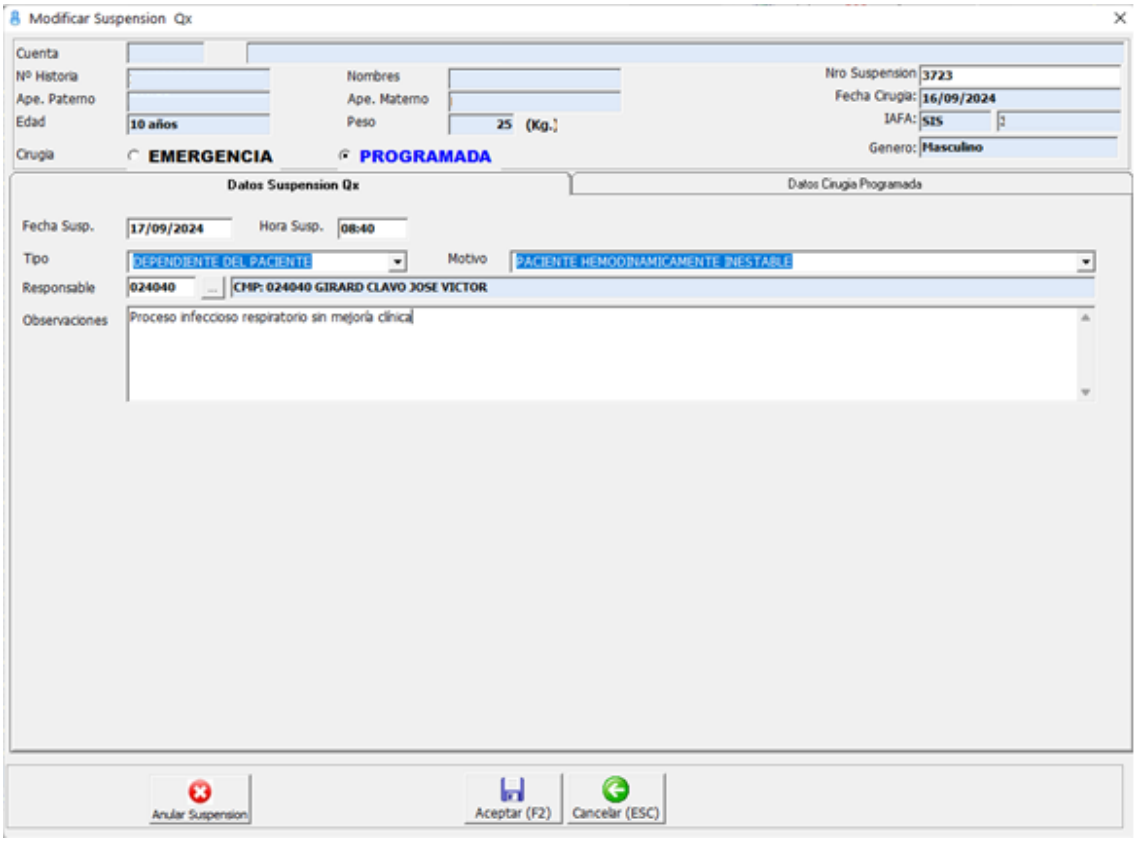


Fig. 56: Formulario de registro de suspensión de sala de operaciones

3.1.4.7 Implementación CU07 Consultar solicitudes de cirugía

Así mismo para la consulta del formulario se estableció una bandeja de solicitudes separadas por tipos de intervención, donde se muestran datos principales de las solicitudes de sala de operaciones, el cual se puede mostrar en las Fig. 57 y Fig. 58.

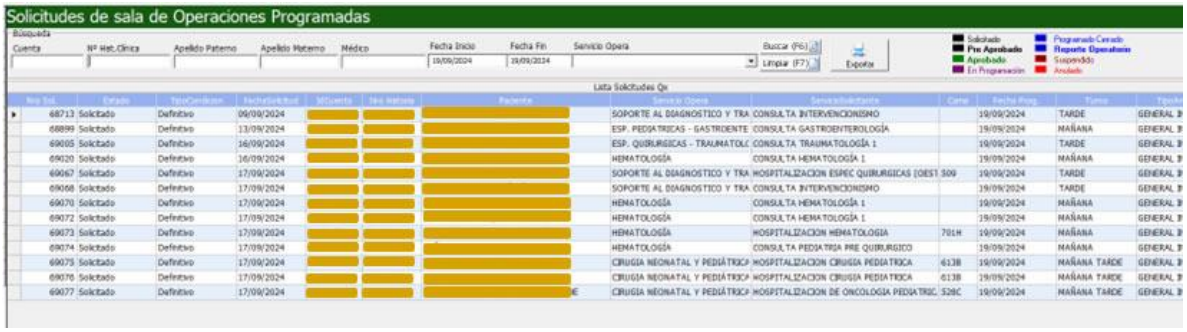


Fig. 57: Bandeja de solicitudes de sala de operaciones programadas

Fig. 58: Bandeja de solicitudes de sala de operaciones de emergencia

A continuación, en la Fig. 59 se muestra la codificación de la búsqueda de solicitudes de sala de operaciones, el evento del botón **buscar** y su método de **búsqueda**.

```
Private Sub btnBuscar_Click()
   Screen.MousePointer = vbHourglass
   RealizarBusqueda False
   Screen.MousePointer = vbDefault
End Sub
Public Sub RealizarBusqueda(lbSoloPacientesSINaltaMedica As Boolean)
Dim oDopaciente As New doPaciente
Dim oDOAtencion As New DOAtencion
Dim lbSigue As Boolean
Dim lnListIndex As Integer
Dim rsRespuesta As New Recordset
Dim lbServicioVacio As Boolean
      grdAdmision.Caption = "Lista Solicitudes Qx"
      If sighentidades.EsFecha(txtFI.Text, "DD/MM/AAAA") And sighentidades.EsFecha(txtFF.Text, "DD/MM/AAAA") Then
         Set rsRespuesta = mo_reglasQx.SolicitudesQxPorFecha(CDate(txtFI.Text + " 00:00:00"), CDate(txtFF.Text + " 23:59:59"), _
         val(mo_cmbIdServicioOpera.BoundText), txtCuenta.Text, txtNroHistoria.Text, txtApellidoPaterno.Text, txtApellidoMaterno.Text, txtMedico.Text)
         rsRespuesta.Filter = " idTipoIntervencion= " & CStr(ml_idTipoIntervencion)
         Set grdAdmision.DataSource = rsRespuesta
         mo_Apariencia.ConfigurarFilasBiColores grdAdmision, sighentidades.GrillaConFilasBicolor

      Else
         MsgBox "Favor de Ingrese una fecha valida"
      End If

End Sub
```

Fig. 59: Consulta de solicitudes de sala de operaciones

3.1.4.8 Implementación CU08 Registrar datos adicionales de cirugía

Para este caso se estableció una tabla en base de datos AprobadosDatosAdicionales, la cual almacena información adicional por cada cirugía realizada, como los son: Hora de llamado al paciente, hora de ingreso a Centro Quirúrgico, hora de ingreso a cirugía, hora de salida de cirugía, hora de ingreso anestesiología, hora de salida de anestesiología, hora salida de Centro Quirúrgico.

Para la implementación se usaron la clase DOAprobadosQxAdicional encargada de mapear la base datos y la clase AprobadosQxAdicional**,** encargada de establecer conexión para enviar y recibir información con la base de datos.

En la Tabla XXXVI visualiza la estructura de la clase **DOAprobadosQxAdicional**.

Tabla XXXVI: Clase Data Object DOAprobadosQxAdicional

Variables privadas	Propiedades de clase	Propiedades de clase

```
'----------------------------------------------------------
'Clase DOAprobadosQxAdicional
'----------------------------------------------------------

Private ml_IdUsuarioAuditoria As Long
Private ml_idAprobadosQxAdicional As Long
Private ml_idAprobadosQx As Long
Private ml_CheckList As Long
Private ms_HoraLlamado As String
Private ms_HoraIngreso As String
Private ms_HoraIngresoSala As String
Private ms_HoraIngresoAnestesio As String
Private ms_HoraIngresoCirugia As String
Private ms_HoraSalidaCirugia As String
Private ms_HoraSalidaAnestesio As String
Private ms_HoraSalida As String
Private ml_idServicioDestino As Long
Private ms_Reintervenciones As String
Private ml_idMotivoRetraso As Long
Private ml_indFallecido As Long
Private ms_HoraIngresoURPA As String
Private ms_HoraAltaAnestesica As String
Private ms_HoraSalidaURPA As String
Private ml_idDestinoURPA As Long
Private ml_idEnfermera As Long
Private ml_idMedicoAnestesiologo As Long
```

```
Property Let IdUsuarioAuditoria(lValue As Long)
    ml_IdUsuarioAuditoria = lValue
End Property
Property Get IdUsuarioAuditoria() As Long
    IdUsuarioAuditoria = ml_IdUsuarioAuditoria
End Property
Property Let idAprobadosQxAdicional(lValue As
    ml_idAprobadosQxAdicional = lValue
End Property
Property Get idAprobadosQxAdicional() As Long
    idAprobadosQxAdicional = ml_idAprobadosQ
End Property
Property Let idAprobadosQx(lValue As Long)
    ml_idAprobadosQx = lValue
End Property
Property Get idAprobadosQx() As Long
    idAprobadosQx = ml_idAprobadosQx
End Property
Property Let CheckList(lValue As Long)
    ml_CheckList = lValue
End Property
Property Get CheckList() As Long
    CheckList = ml_CheckList
End Property
Property Let HoraLlamado(sValue As String)
    ms_HoraLlamado = sValue
End Property

Property Let HoraIngreso(sValue As String)
    ms_HoraIngreso = sValue
End Property
Property Get HoraIngreso() As String
    HoraIngreso = ms_HoraIngreso
End Property
Property Let HoraIngresoSala(sValue As String)
    ms_HoraIngresoSala = sValue
End Property
Property Get HoraIngresoSala() As String
    HoraIngresoSala = ms_HoraIngresoSala
End Property
Property Let HoraIngresoAnestesio(sValue As String)
    ms_HoraIngresoAnestesio = sValue
End Property
Property Get HoraIngresoAnestesio() As String
    HoraIngresoAnestesio = ms_HoraIngresoAnestesio
End Property
Property Let HoraIngresoCirugia(sValue As String)
    ms_HoraIngresoCirugia = sValue
End Property
Property Get HoraIngresoCirugia() As String
    HoraIngresoCirugia = ms_HoraIngresoCirugia
End Property
Property Let HoraSalidaCirugia(sValue As String)
    ms_HoraSalidaCirugia = sValue
End Property
```

```
Property Let HoraSalidaAnestesio(sValue As String)
    ms_HoraSalidaAnestesio = sValue
End Property
Property Get HoraSalidaAnestesio() As String
    HoraSalidaAnestesio = ms_HoraSalidaAnestesio
End Property
Property Let HoraSalida(sValue As String)
    ms_HoraSalida = sValue
End Property
Property Get HoraSalida() As String
    HoraSalida = ms_HoraSalida
End Property
Property Let idServicioDestino(lValue As Long)
    ml_idServicioDestino = lValue
End Property
Property Get idServicioDestino() As Long
    idServicioDestino = ml_idServicioDestino
End Property
Property Let Reintervenciones(sValue As String)
    ms_Reintervenciones = sValue
End Property
Property Get Reintervenciones() As String
    Reintervenciones = ms_Reintervenciones
End Property
Property Let idMotivoRetraso(lValue As Long)
    ml_idMotivoRetraso = lValue
End Property

Property Let indFallecido(lValue As Long)
    ml_indFallecido = lValue
End Property
Property Get indFallecido() As Long
    indFallecido = ml_indFallecido
End Property
Property Let HoraIngresoURPA(sValue As String)
    ms_HoraIngresoURPA = sValue
End Property
Property Get HoraIngresoURPA() As String
    HoraIngresoURPA = ms_HoraIngresoURPA
End Property
Property Let HoraAltaAnestesica(sValue As String)
    ms_HoraAltaAnestesica = sValue
End Property
Property Get HoraAltaAnestesica() As String
    HoraAltaAnestesica = ms_HoraAltaAnestesica
End Property
Property Let HoraSalidaURPA(sValue As String)
    ms_HoraSalidaURPA = sValue
End Property
Property Get HoraSalidaURPA() As String
    HoraSalidaURPA = ms_HoraSalidaURPA
End Property
Property Let idDestinoURPA(lValue As Long)
    ml_idDestinoURPA = lValue
End Property
```

Continuando con la implementación en la Tabla XXXVII se muestra la clase ADO.

Tabla XXXVII: Clase ADO AprobadosQxAdicional

Declaracion de variables y propiedades de la clase

```
'---------------------------------------------------------
'Clase Ado AprobadosQxAdicional
'---------------------------------------------------------

Option Explicit
Dim mo_Conexion As ADODB.Connection
Dim ms_MensajeError As String
Property Let MensajeError(sValue As String)
  ms_MensajeError = sValue
End Property
Property Get MensajeError() As String
  MensajeError = ms_MensajeError
End Property
Property Set Conexion(oValue As ADODB.Connection)
  Set mo_Conexion = oValue
End Property
Property Get Conexion() As ADODB.Connection
  Set Conexion = mo_Conexion
End Property
```

Metodo para insertar un registro en la tabla AprobadosQxAdicional

```
Function Insertar(ByVal oTabla As doAprobadosQXAdicional) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

Insertar = False
With oCommand
.CommandType = adCmdStoredProc
Set .ActiveConnection = mo_Conexion
.CommandText = "usp_insert_AprobadosQXAdicionalAgregar_07062024"
  Set oParameter = .CreateParameter("@idAprobadosQx", adInteger, adParamInput, 0, IIf(oTabla.idAprobadosQx = 0, Null, oTabla.idAprobadc
  Set oParameter = .CreateParameter("@checkList", adInteger, adParamInput, 0, IIf(oTabla.CheckList = 0, Null, oTabla.CheckList)): .Parameter
  Set oParameter = .CreateParameter("@horaLlamado", adVarChar, adParamInput, 5, IIf(oTabla.HoraLlamado = "", Null, oTabla.HoraLlamado))
  Set oParameter = .CreateParameter("@horaIngreso", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngreso = "", Null, oTabla.HoraIngreso)): .P
  Set oParameter = .CreateParameter("@horaIngresoSala", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngresoSala = "", Null, oTabla.HoraIng
  Set oParameter = .CreateParameter("@horaIngresoAnestesio", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngresoAnestesio = "", Null, oTal
  Set oParameter = .CreateParameter("@horaIngresoCirugia", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngresoCirugia = "", Null, oTabla.Ho
  Set oParameter = .CreateParameter("@horaSalidaCirugia", adVarChar, adParamInput, 5, IIf(oTabla.HoraSalidaCirugia = "", Null, oTabla.HoraS
  Set oParameter = .CreateParameter("@horaSalidaAnestesio", adVarChar, adParamInput, 5, IIf(oTabla.HoraSalidaAnestesio = "", Null, oTabla.
  Set oParameter = .CreateParameter("@horaSalida", adVarChar, adParamInput, 5, IIf(oTabla.HoraSalida = "", Null, oTabla.HoraSalida)): .Param
  Set oParameter = .CreateParameter("@idServicioDestino", adInteger, adParamInput, 0, IIf(oTabla.IdServicioDestino = 0, Null, oTabla.IdServi
  Set oParameter = .CreateParameter("@reintervenciones", adVarChar, adParamInput, 50, IIf(oTabla.Reintervenciones = "", Null, oTabla.Reinte
  Set oParameter = .CreateParameter("@idMotivoRetraso", adInteger, adParamInput, 0, IIf(oTabla.idMotivoRetraso = 0, Null, oTabla.idMotivoR
  Set oParameter = .CreateParameter("@indFallecido", adInteger, adParamInput, 0, IIf(oTabla.indFallecido = 0, Null, oTabla.indFallecido)): .Par
  Set oParameter = .CreateParameter("@idAprobadosQxAdicional", adInteger, adParamOutput, 0): .Parameters.Append oParameter
  Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Append oP
  .Execute
   oTabla.idAprobadosQxAdicional = .Parameters("@idAprobadosQxAdicional")
End With

  Insertar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description
Exit Function
End Function
```

Metodo para modificar un registro en la tabla AprobadosQxAdicional

```
Function Modificar(ByVal oTabla As doAprobadosQXAdicional) As Boolean
On Error GoTo ManejadorDeError
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter

  Modificar = False
  With oCommand
    .CommandType = adCmdStoredProc
    Set .ActiveConnection = mo_Conexion
    .CommandText = "usp_update_AprobadosQXAdicionalModificar_15072024"
      Set oParameter = .CreateParameter("@idAprobadosQxAdicional", adInteger, adParamInput, 0, IIf(oTabla.idAprobadosQxAdicional = 0, N
      Set oParameter = .CreateParameter("@idAprobadosQx", adInteger, adParamInput, 0, IIf(oTabla.idAprobadosQx = 0, Null, oTabla.idAprol
      Set oParameter = .CreateParameter("@checkList", adInteger, adParamInput, 0, IIf(oTabla.CheckList = 0, Null, oTabla.CheckList)): .Param
      Set oParameter = .CreateParameter("@horaLlamado", adVarChar, adParamInput, 5, IIf(oTabla.HoraLlamado = "", Null, oTabla.HoraLlama
      Set oParameter = .CreateParameter("@horaIngreso", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngreso = "", Null, oTabla.HoraIngreso
      Set oParameter = .CreateParameter("@horaIngresoSala", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngresoSala = "", Null, oTabla.Hor
      Set oParameter = .CreateParameter("@horaIngresoAnestesio", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngresoAnestesio = "", Null,
      Set oParameter = .CreateParameter("@horaIngresoCirugia", adVarChar, adParamInput, 5, IIf(oTabla.HoraIngresoCirugia = "", Null, oTabl
      Set oParameter = .CreateParameter("@horaSalidaCirugia", adVarChar, adParamInput, 5, IIf(oTabla.HoraSalidaCirugia = "", Null, oTabla.H
      Set oParameter = .CreateParameter("@horaSalidaAnestesio", adVarChar, adParamInput, 5, IIf(oTabla.HoraSalidaAnestesio = "", Null, oT
      Set oParameter = .CreateParameter("@horaSalida", adVarChar, adParamInput, 5, IIf(oTabla.HoraSalida = "", Null, oTabla.HoraSalida)): .P
      Set oParameter = .CreateParameter("@idServicioDestino", adInteger, adParamInput, 0, IIf(oTabla.IdServicioDestino = 0, Null, oTabla.IdS
      Set oParameter = .CreateParameter("@reintervenciones", adVarChar, adParamInput, 50, IIf(oTabla.Reintervenciones = "", Null, oTabla.R
      Set oParameter = .CreateParameter("@idMotivoRetraso", adInteger, adParamInput, 0, IIf(oTabla.idMotivoRetraso = 0, Null, oTabla.idMoti
      Set oParameter = .CreateParameter("@indFallecido", adInteger, adParamInput, 0, IIf(oTabla.indFallecido = 0, Null, oTabla.indFallecido)):
       Set oParameter = .CreateParameter("@IdUsuarioAuditoria", adInteger, adParamInput, 0, oTabla.IdUsuarioAuditoria): .Parameters.Appen
      .Execute
```

```
  End With

  Modificar = True
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description
Exit Function
End Function
```

Metodo para Seleccionar un registro en la tabla AprobadosQxAdicional

```
Function SeleccionarPorId(ByVal oTabla As doAprobadosQXAdicional) As Boolean
On Error GoTo ManejadorDeError
Dim oRecordset As New ADODB.Recordset
Dim oCommand As New ADODB.Command
Dim oParameter As ADODB.Parameter
  SeleccionarPorId = False
  With oCommand
     .CommandType = adCmdStoredProc
     Set .ActiveConnection = mo_Conexion
     .CommandText = "usp_AprobadosQXAdicionalSeleccionarPorId_07062024"
     Set oParameter = .CreateParameter("@idAprobadosQxAdicional", adInteger, adParamInput, 0, oTabla.idAprobadosQxAdicional): .Paramet
     Set oRecordset = .Execute
  End With
  If Not (oRecordset.EOF And oRecordset.BOF) Then
     SeleccionarPorId = True
     CargaCampos oTabla, oRecordset
  Else
     SeleccionarPorId = False
  End If
  oRecordset.Close
  ms_MensajeError = ""

Exit Function
ManejadorDeError:
  ms_MensajeError = Err.Number & " " + Err.Description
End Function
```

Para las interfaces de usuario se estableció el siguiente formulario para realizar el registro de datos adicionales de las cirugías realizadas en la Fig. 60.

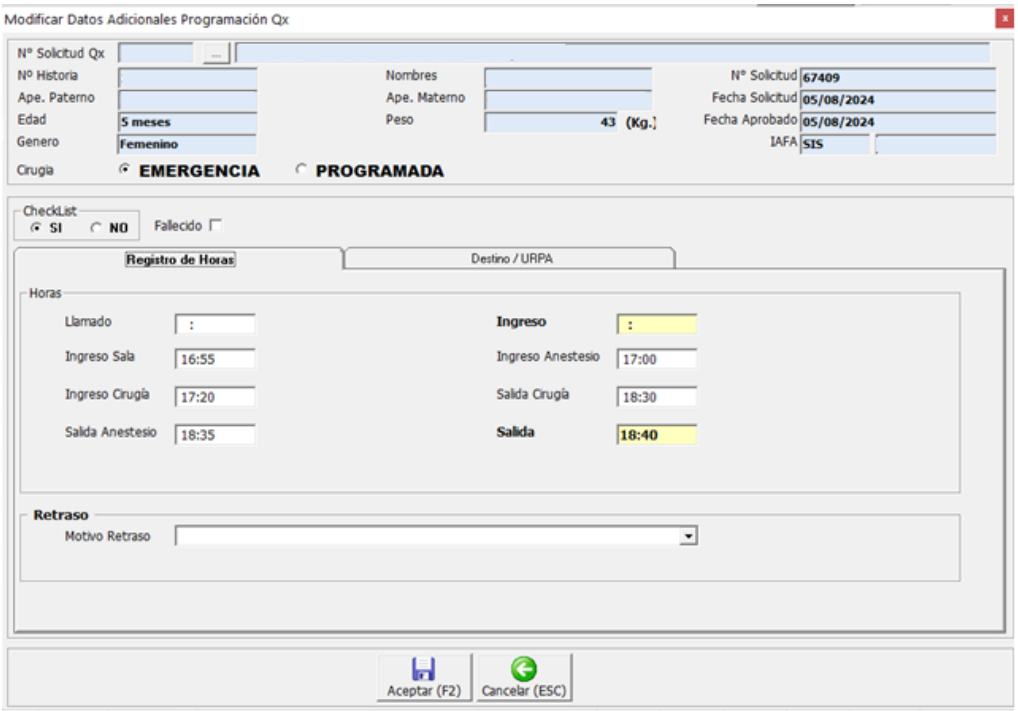


Fig. 60: Formulario de registrar datos adicionales de cirugía

En la Tabla XXXVIII se muestra la codificación del formulario para el registro de datos adicionales de cirugía.

Tabla XXXVIII: Codificación de formulario registro adicional de cirugía

Método para cargar datos de base de datos e imprimir en los controles

```
Sub CargarDatosALosControles()
   Dim oConexion As New ADODB.Connection
   Dim rsap As New Recordset

   oConexion.Open sighentidades.CadenaConexion
   oConexion.CursorLocation = adUseClient
   Set oAprobadoQx.Conexion = oConexion

   Set oAprobadoQxAdicional.Conexion = oConexion
   odoAprobadoQxAdicional.idAprobadosQx = ml_Id
   If oAprobadoQxAdicional.SeleccionarPorIdAprobado(odoAprobadoQxAdicional) Then
      With odoAprobadoQxAdicional
         txtHoraLlamado.Text = .horallamado
         txtHoraIngreso.Text = .horaIngreso
         txtHoraIngresoSala.Text = .HoraIngresoSala
         txtHoraIngresoAnestesio.Text = .HoraIngresoAnestesio
         txtHoraIngresoCirugia.Text = .HoraIngresoCirugia
         txtHoraSalidaCirugia.Text = .HoraSalidaCirugia
         txtHoraSalidaAnestesio.Text = .HoraSalidaAnestesio
         txtHoraSalida.Text = .HoraSalida
      End With
   End If
   Set rsap = mo_reglasQx.AprobadoQxCompletoPorId(ml_Id)
   If rsap.RecordCount > 0 Then
      txtNroSolicitud = rsap.Fields!NroSolicitud
      txtNSolicitud.Text = rsap.Fields!NroSolicitud
      txtFechaAProbado.Text = rsap.Fields!FechaAprobado
      txtPeso.Text = rsap.Fields!Peso
      If rsap.Fields!idTipoIntervencion = 1 Then
         rbProgramada.Value = True
      Else
         rbEmergencia.Value = True
      End If
   End If

End Sub
```

Método para Validar reglas antes de Agregar o Modificar un registro

```
Function ValidarReglas() As Boolean
   ValidarReglas = False

   Dim sMensaje As String
   sMensaje = ""

   If Not HoraValida(txtHoraAltaAnestesica.Text) Then
      sMensaje = sMensaje & "La hora de Alta Anestesica Ingresada no es válida" & Chr(13)
   End If
   If Not HoraValida(txtHoraIngreso.Text) Then
      sMensaje = sMensaje & "La hora de Ingreso no es válida" & Chr(13)
   End If
   If Not HoraValida(txtHoraLlamado.Text) Then
      sMensaje = sMensaje & "La hora de Llamado no es válida" & Chr(13)
   End If
   If Not HoraValida(txtHoraIngresoCirugia.Text) Then
      sMensaje = sMensaje & "La hora de Ingreso de Cirugia no es válida" & Chr(13)
   End If
   If Not HoraValida(txtHoraSalidaCirugia.Text) Then
      sMensaje = sMensaje & "La hora de Salida de Cirugia no es válida" & Chr(13)
   End If
   If Not HoraValida(txtHoraIngresoSala.Text) Then
      sMensaje = sMensaje & "La hora de Ingreso a Sala no es válida" & Chr(13)
   End If
   If Not HoraValida(txtHoraSalida.Text) Then
      sMensaje = sMensaje & "La hora de Ingreso de Salida no es válida" & Chr(13)
   End If
```

```
If Not HoraValida(txtHoraIngresoAnestesio.Text) Then
   sMensaje = sMensaje & "La hora de Ingreso Anestesio no es válida" & Chr(13)
End If
If Not HoraValida(txtHoraSalidaAnestesio.Text) Then
   sMensaje = sMensaje & "La hora de Salida de Anestesio no es válida" & Chr(13)
End If
If Not HoraValida(txtHoraIngresoURPA.Text) Then
   sMensaje = sMensaje & "La hora de Ingreso a URPA no es válida" & Chr(13)
End If
If Not HoraValida(txtHoraSalidaURPA.Text) Then
   sMensaje = sMensaje & "La hora de Salida de URPA no es válida" & Chr(13)
End If
If txtHoraIngresoCirugia.Text <> "__:__" And txtHoraSalidaCirugia.Text <> "__:__" Then
   If CDate(txtHoraIngresoCirugia.Text) > CDate(txtHoraSalidaCirugia.Text) Then
      sMensaje = sMensaje & "El ingreso cirugia debe ser menor que la salida cirugia " & Chr(13)
   End If
End If
If txtHoraSalidaCirugia.Text <> "__:__" And txtHoraSalidaAnestesio.Text <> "__:__" Then
   If CDate(txtHoraSalidaCirugia.Text) > CDate(txtHoraSalidaAnestesio.Text) Then
      sMensaje = sMensaje & "La salida cirugia debe ser menor que la salida anestesia " & Chr(13)
   End If
End If

If sMensaje <> "" Then
   MsgBox sMensaje, vbInformation, Me.Caption
   Exit Function
End If
```

Método para cargar los datos de los controles hacia el objeto para enviar a la base de datos (agregar o modificar)

```
Sub CargaDatosAlObjetosDeDatos()
  With odoAprobadoQxAdicional
     If rbChecklistSi.Value = True Then
        .CheckList = 1
     Else
        .CheckList = 0
     End If
     If chkFallecido.Value Then
        .indFallecido = True
     Else
        .indFallecido = False
     End If
     .horallamado = txtHoraLlamado.Text
     .horaIngreso = txtHoraIngreso.Text
     .HoraIngresoSala = txtHoraIngresoSala.Text
     .HoraIngresoAnestesio = txtHoraIngresoAnestesio.Text
     .HoraIngresoCirugia = txtHoraIngresoCirugia.Text
     .HoraSalidaCirugia = txtHoraSalidaCirugia.Text
     .HoraSalidaAnestesio = txtHoraSalidaAnestesio.Text
     .HoraSalida = txtHoraSalida.Text
     .idServicioDestino = val(txtIdServicioDestino.Tag)
     .idDestinoURPA = val(txtIdServicioDestinoURPA.Tag)
     .Reintervenciones = txtReintervenciones.Text
     .IdUsuarioAuditoria = ml_IdUsuario
  End With

End Sub
```

Evento click del botón aceptar del formulario, ejecutara la opción correspondiente

```
Private Sub btnAceptar_Click()
  Select Case mi_Opcion
  Case sghModificar
     If ValidarDatosObligatorios() Then
        If ValidarReglas() Then
           If ModificarDatos() Then
              MsgBox "Los datos se modificaron correctamente", vbInformation, Me.Caption
              Me.Visible = False
           Else
              MsgBox "No se pudo modificar los datos" + Chr(13) + _
              mo_reglasQx.MensajeError, vbExclamation, Me.Caption
           End If
        End If
     End If
  Case sghEliminar
        If ValidarReglas() Then
           If EliminarDatos() Then
              MsgBox "Los datos se eliminaron correctamente", vbInformation, Me.Caption
              Me.Visible = False
           Else
              MsgBox "No se pudo eliminar los datos" + Chr(13) + _
              mo_reglasQx.MensajeError, vbExclamation, Me.Caption
           End If
        End If
  End Select

End Sub
```

Metodos ModificarDatos para modificar un registro y EliminarDatos para eliminar un registro

```
Function ModificarDatos() As Boolean
  Dim oConexion As New ADODB.Connection

    oConexion.Open sighentidades.CadenaConexion
    oConexion.CursorLocation = adUseClient
    Set oAprobadoQxAdicional.Conexion = oConexion

   CargaDatosAlObjetosDeDatos
   ModificarDatos = oAprobadoQxAdicional.Modificar(odoAprobadoQxAdicional)

oConexion.Close
Set oConexion = Nothing
End Function

Function EliminarDatos() As Boolean
  Dim oConexion As New ADODB.Connection

    oConexion.Open sighentidades.CadenaConexion
    oConexion.CursorLocation = adUseClient
    Set oAprobadoQxAdicional.Conexion = oConexion

   CargaDatosAlObjetosDeDatos
   odoAprobadoQxAdicional.idEstado = 0
   EliminarDatos = oAprobadoQxAdicional.Eliminar(odoAprobadoQxAdicional)

oConexion.Close
Set oConexion = Nothing
End Function
```

Registros adicionales de traslados a centro quirúrgico como horas de llamado, hora de ingreso, hora de salida a quirófano, tiempo de recuperación en la Fig. 61.

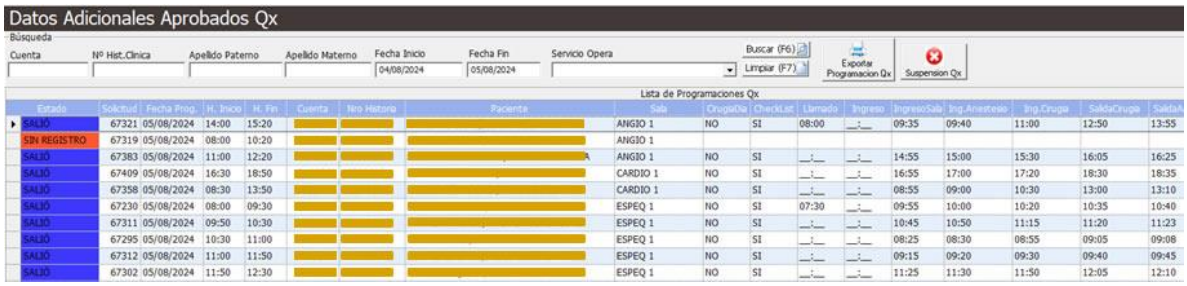


Fig. 61: Bandeja de registro de datos adicionales de cirugías

3.1.5 Fase: Pruebas y despliegue

Pruebas

En esta fase se realizaron las pruebas al módulo de centro quirúrgico, la validación de los formularios se rige en las interfaces con los métodos de **ValidarDatosObligatorios()** y **ValidarReglas(),** aplicados en cada formulario correspondiente a los casos identificados en la fase de diseño.

3.1.5.1 Pruebas CU01 Registrar solicitud de sala de operaciones

Para el formulario de Registro de Solicitud de sala de operaciones, se realizaron las pruebas mostradas en la Tabla XXXIX.

Tabla XXXIX: Pruebas realizadas para solicitud de sala de operaciones

CU01 Registrar Solicitud de sala de operaciones				
Caso de prueba	**Datos de prueba**	**Resultado actual**	**Resultado esperado**	**Estado**
Registro de solicitud con datos completos	Datos del paciente, diagnósticos, procedimientos, servicio que opera, especialidad, sala, turno, fecha, hora	La solicitud se registra correctamente	Como se esperaba	Cumple
Registro de solicitud con campos obligatorios vacíos	Falta registrar diagnósticos	Se muestra un mensaje de error indicando que falta registrar diagnósticos	Como se esperaba	Cumple
Registro de solicitud con datos incorrectos	Escribe fechas anteriores, fecha incorrecta	El sistema muestra un mensaje de validación	Como se esperaba	Cumple

A continuación, en las Fig. 62 y Fig. 63 se evidencias las pruebas realizadas al formulario de registro de solicitud de sala de operaciones

Fig. 62: Validación datos obligatorios en solicitud de sala de operaciones

Fig. 63: Validación de datos incorrectos en solicitud de sala de operaciones

3.1.5.2 Pruebas CU02 Aprobar solicitud de sala de operaciones

Para las pruebas de aprobación de solicitud de sala de operaciones se realizaron tanto para usuario con permiso asignado para aprobación por jefatura y usuario con permiso asignado para aprobación por dirección, resultando información en la Tabla XL.

Tabla XL: Pruebas realizadas para aprobación de sala de operaciones

CU02 Aprobar solicitud de sala de operaciones				
Caso de prueba	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**
Pre Aprobar solicitud válida	Pre Aprobar solicitud válida	Pre Aprobar solicitud válida	Pre Aprobar solicitud válida	Pre Aprobar solicitud válida
Aprobar solicitud válida	Aprobar solicitud válida	Aprobar solicitud válida	Aprobar solicitud válida	Aprobar solicitud válida
Intentar aprobar solicitud sin permiso	Intentar aprobar solicitud sin permiso	Intentar aprobar solicitud sin permiso	Intentar aprobar solicitud sin permiso	Intentar aprobar solicitud sin permiso

3.1.5.3 Pruebas CU03 Programar sala de operaciones

Para las pruebas en la programación de sala de operaciones, se tomaron en cuenta la validación de datos obligatorios y aplicación de reglas solicitadas por el usuario. Datos obligatorios: sala, fecha cirugía, hora inicio y fin de cirugía, anestesiólogo, y como validación de reglas, no se puede programar con fechas anteriores, no se puede programar si el día ya fue cerrado, como se observa en la Tabla XLI.

Tabla XLI: Pruebas realizadas para programación de sala de operaciones

CU03 Programar sala de operaciones				
Caso de prueba	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**
Registro de programación exitosa	Registro de programación exitosa	Registro de programación exitosa	Registro de programación exitosa	Registro de programación exitosa
Programación de cirugía de paciente duplicado	Programación de cirugía de paciente duplicado	Programación de cirugía de paciente duplicado	Programación de cirugía de paciente duplicado	Programación de cirugía de paciente duplicado
Registro de programación con datos incompletos	Registro de programación con datos incompletos	Registro de programación con datos incompletos	Registro de programación con datos incompletos	Registro de programación con datos incompletos

3.1.5.4 Pruebas CU04 Cerrar día programación de sala de operaciones

Para las pruebas cierre de día programación de sala de operaciones, se tomaron en cuenta la asignación de permiso correspondiente, así mismo se tomó en cuenta la cantidad de cirugías programadas en el día, como se observa en la Tabla XLII.

Tabla XLII: Pruebas de cierre de día de programación de sala de operaciones

CU04 Cerrar día programación de cirugía				
Caso de prueba	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**
Cerrar programación con éxito	Lista de Programación confirmada	El día se cierra satisfactoriamente	Como se esperaba	Cumple
Intento de cerrar sin datos	No existe programaciones de sala de día	El sistema muestra un mensaje de error	Como se esperaba	Cumple

3.1.5.5 Pruebas CU05 Registrar reporte operatorio

Para las pruebas cierre de día de programación de sala de operaciones, se tomaron en cuenta la asignación de permiso asignado en la programación correspondiente como cirujano principal, se validaron los datos obligatorios para el registro del reporte.

Todos los campos son obligatorios a excepción de los médicos ayudantes, segundo anestesiólogo, incidencias, complicaciones. En la Tabla XLIII se muestra las pruebas realizas al registrar el reporte operatorio.

Tabla XLIII: Pruebas de registro de reporte operatorio

CU05 Registrar reporte operatorio				
Caso de prueba	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**
Registro de reporte operatorio exitoso	Datos del procedimiento quirúrgico completo, fechas y horas, diagnósticos, equipo quirúrgico	Reporte operatorio registrado correctamente	Como se esperaba	Cumple
Registro con datos faltantes	Falta circulante asignado	El sistema solicita completar los datos	Como se esperaba	Cumple
Registro con datos incorrectos	Registro de horas incorrectas	El sistema valida con un mensaje de datos incorrectos	Como se esperaba	Cumple
Registro de reporte con fuera de fecha	Registro excedió el día siguiente hábil para registrar	El sistema valida con mensaje que se encuentra fuera de fecha	Como se esperaba	Cumple

3.1.5.6 Pruebas CU06 Registrar suspensión de sala de operaciones

Para validar las reglas de suspensión de sala de operaciones, se consideró validación de datos obligatorios como motivo de suspensión, tipo de suspensión, responsable de suspensión, y como aplicación de reglas se consideró que no se pueda suspender si ya cuenta con reporte operatorio, como se observa en la Tabla XLIV.

Tabla XLIV: Pruebas de suspensión de sala de operaciones

CU06 Registrar suspensión de sala de operaciones				
Caso de prueba	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**
Suspensión de cirugía con motivo válido	Motivo justificado registrado	Cirugía suspendida correctamente	Como se esperaba	Cumple
Intentar suspender sin motivo	No se registra motivo de suspensión	El sistema muestra un error y no permite la acción	Como se esperaba	Cumple
Registro con datos incorrectos	Registro de hora incorrecta	El sistema valida con un mensaje de datos incorrectos	Como se esperaba	Cumple

3.1.5.7 Pruebas CU07 Registrar solicitud de sala de operaciones

Par validar las reglas de consulta se solicitud de sala de operaciones, se consideró el formato de fechas y el rango correcto, validando que la fecha de inicio sea menor que la fecha final como filtro de la consulta, tal como se muestra en la Tabla XLV.

Tabla XLV: Pruebas en consultar solicitud de sala de operaciones

CU07 Consultar solicitudes de cirugía				
Caso de prueba	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**
Consulta exitosa	Solicitudes existentes en el sistema	Muestra la lista de solicitudes con sus estados	Como se esperaba	Cumple
Búsqueda con filtro incorrecto	Fecha fuera del rango	No muestra resultados, indica mensaje de validación	Como se esperaba	Cumple

3.1.5.8 Pruebas CU08 Registrar solicitud de sala de operaciones

Para validación de reglas del registro de datos adicionales, se tomó en cuenta el formato de las horas, ya que la mayoría de datos son los tiempos que pasa el paciente dentro del centro quirúrgico, como datos obligatorios se consignaron la hora de ingreso a centro quirúrgico y hora de salida de centro quirúrgico, como validación de reglas se considera que la hora de salida debe ser mayor a la hora de ingreso, a menos que la hora de salida sea del día siguiente, se muestra en la Tabla XLVI con las pruebas realizadas.

Tabla XLVI: Pruebas en registro de datos adicionales de cirugía

CU08 Consultar solicitudes de cirugía				
Caso de prueba	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**	**Caso de prueba**
Registro de Datos adicionales de cirugía con datos mínimos obligatorios	Registro de horas de ingreso, hora de salida de centro quirúrgico	Se realiza el registro con éxito	Como se esperaba	Cumple
Registro con datos faltantes	Sin registro de hora de ingreso a Cirugía	Muestra mensaje de validación de datos obligatorio	Como se esperaba	Cumple
Registro de datos incorrectos	Registro de horas incorrectas	Muestra mensaje de error de datos incorrectos	Como se esperaba	Cumple

Las pruebas realizadas en el módulo de centro quirúrgico garantizan el correcto funcionamiento, asegurando que cumpla con los requerimientos definidos. Se realizaron las pruebas de validación y de reglas en cada uno de los casos de uso identificados, obteniendo resultados esperados.

Despliegue

La publicación del módulo de centro quirúrgico se dio dentro del sistema SisGalenPlus como se observa en la Fig. 63, para esto se aplicaron las siguientes acciones:

- Se asignó el rol al personal administrativo para el registro de sala de operaciones, así como también el permiso para cerrar y abrir día de programación de sala de operaciones.

- Se modificó el rol de personal médico, para agregar la opción de solicitud de sala de operaciones (programadas y emergencia), registro de reporte operatorio.
- Se agregó el permiso a las jefaturas de especialidades para que puedan preaprobar las solicitudes de sala de operaciones
- Se asignó el permiso a dirección para que puedan aprobar las solicitudes aprobadas por las jefaturas.
- Se asignó al personal administrativo de centro quirúrgico la opción para que puedan registrar datos adicionales de cirugía.

Despliegue

La Fig. 64 muestra el despliegue, se realizó el plan piloto con apoyo del servicio de quemados, en esta etapa se realizó retroalimentación del módulo en los diferentes casos de uso identificados. Se realizaron las capacitaciones del uso adecuado del módulo de centro quirúrgico al personal médico de consulta externa, emergencia y hospitalización. Luego del plan piloto, el módulo de centro quirúrgico sale a producción pasando por control de calidad y aprobación de la coordinación de desarrollo de la unidad de tecnologías de la información del Instituto Nacional de Salud del Niño San Borja.

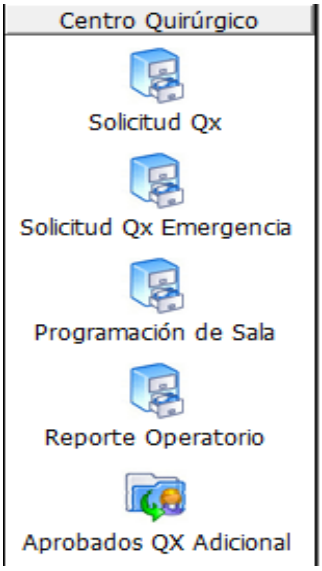


Fig. 64: Despliegue del módulo de centro quirúrgico en SisGalenPlus

Luego del despliegue de módulo de centro quirúrgico, en las figuras de la 65 a la 68 se describen los nuevos procesos.

Para el registro de la solicitud de Sala de operaciones se redujo el tiempo de registro ya que el medico cuenta con información completa del paciente, tal como se muestra en la Fig. 65.

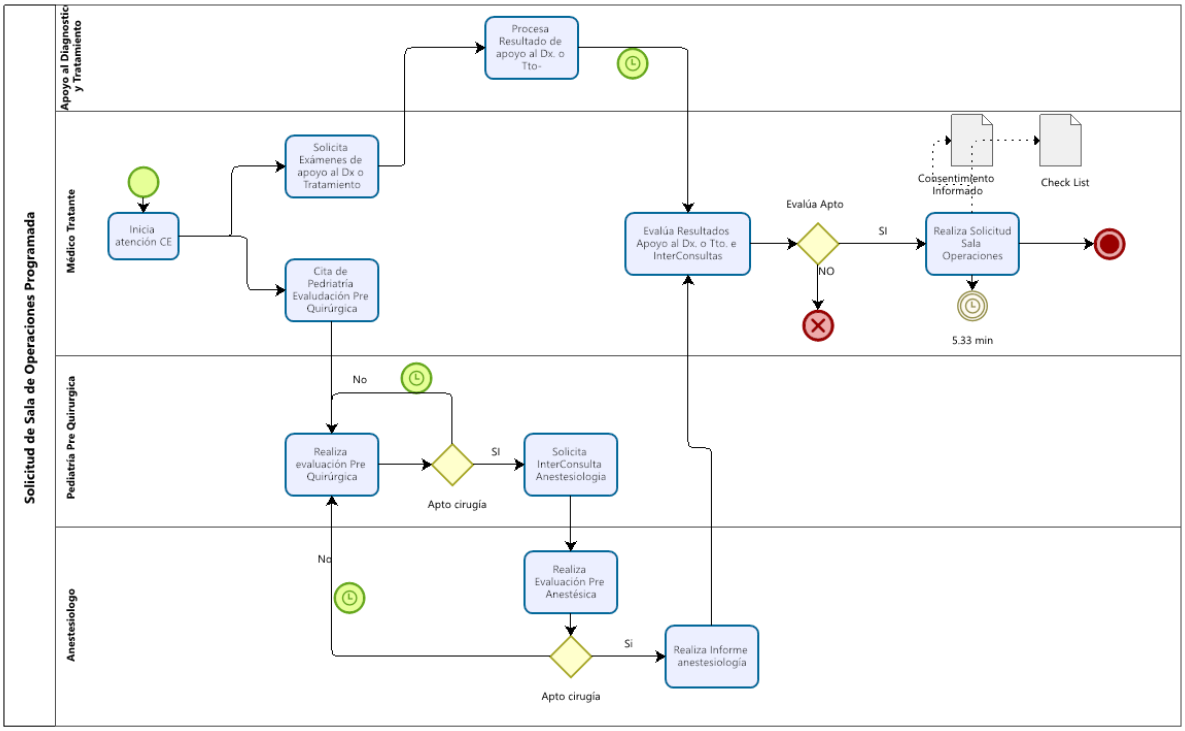


Fig. 65 Nuevo Proceso de Solicitud de Sala de Operaciones

Para el proceso de Aprobación de solicitud de Sala de Operaciones, se redujo el tiempo en la aprobación por Jefatura, ya que ahora es necesario ingresar al sistema y actualizar el estado de aprobado o rechazado de acuerdo a la Fig. 66.

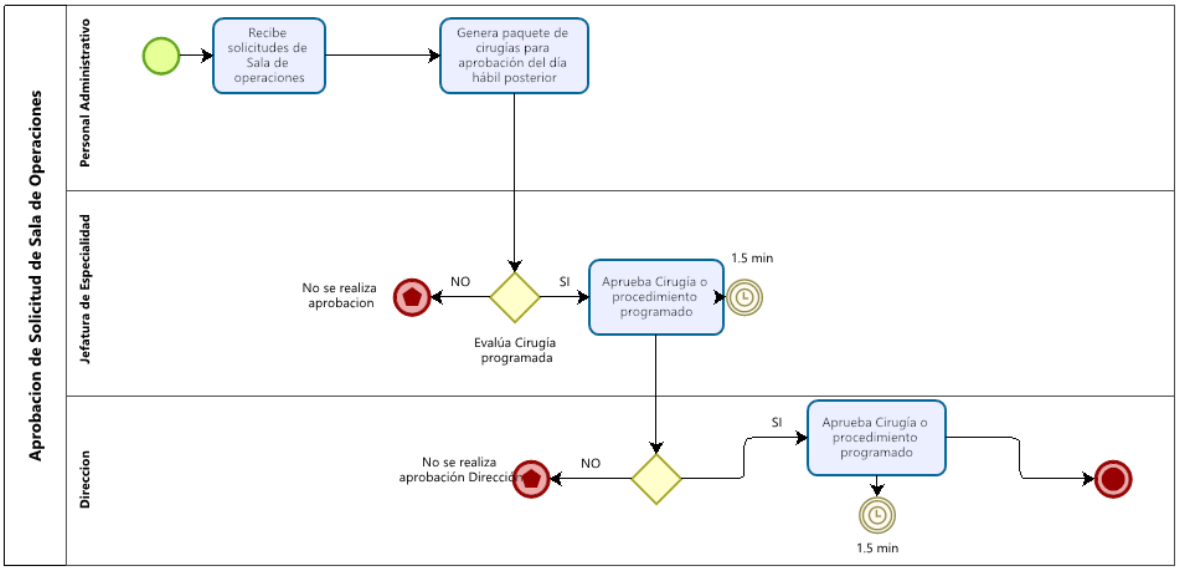


Fig. 66 Nuevo Proceso de Aprobación de Solicitud de Sala de Operaciones

Para el proceso de Programación de Sala de operaciones se redujo el tiempo a 7 minutos debido a que retira el tiempo de transcripción de las solicitudes a una hoja de cálculo Excel, teniendo alta probabilidad de equivocarse y de realizar una revisión posterior a la generación de la lista de pacientes programados, como se muestra en la Fig. 67.

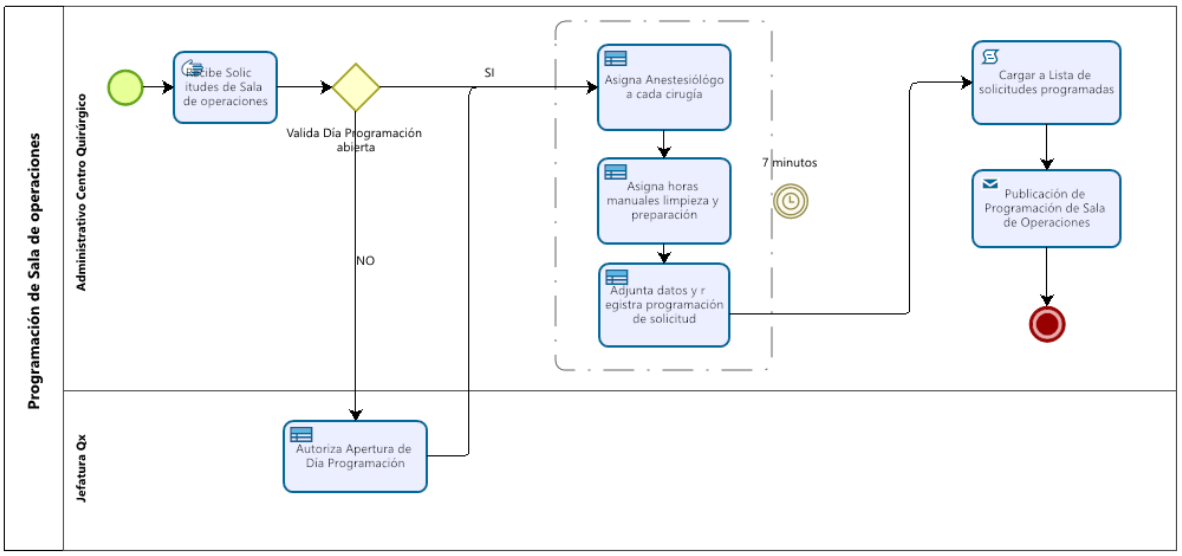


Fig. 67 Nuevo Proceso en Programación de Sala de Operaciones

Finalmente, para el proceso de realización de cirugía se evidencia una reducción de tiempo, ya que el módulo ofrece datos pre llenados para el registro del reporte operatorio, tal como datos del paciente, cirugía programada, diagnostico, evitando y disminuyendo la probabilidad de equivocación de paciente, procedimiento o diagnóstico, como se muestra en la Fig. 68

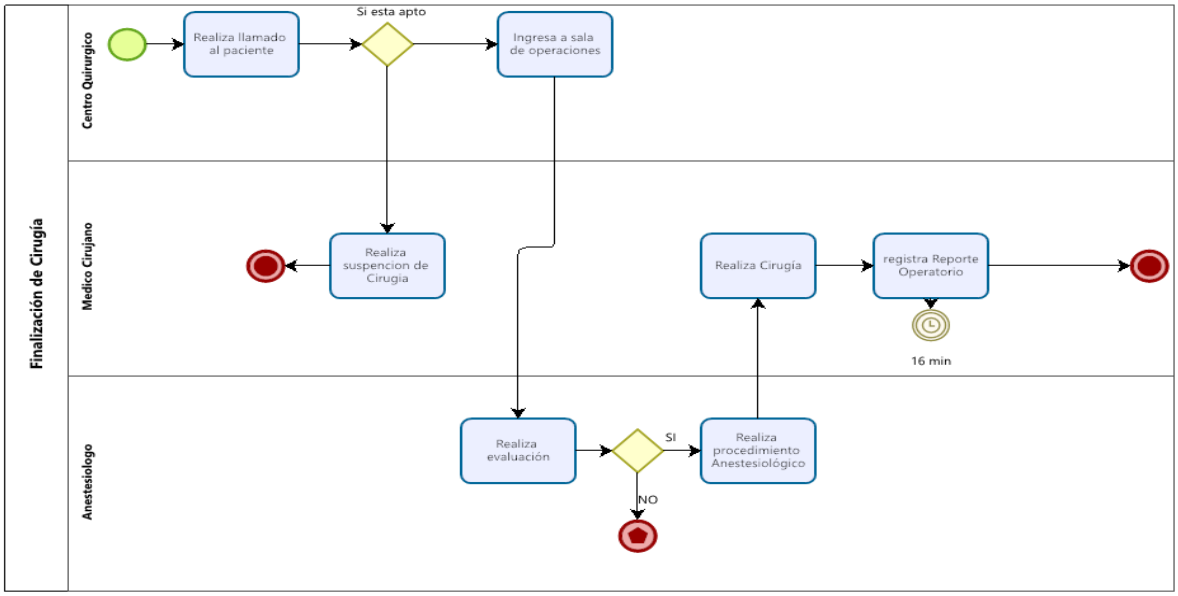


Fig. 68 Nuevo Proceso de Realización de Cirugía

Para el despliegue se implementaron nuevos roles y permisos del sistema para asegurar el correcto funcionamiento de acuerdo a los diferentes actores del sistema, como se muestra en la figura 69 y 70.

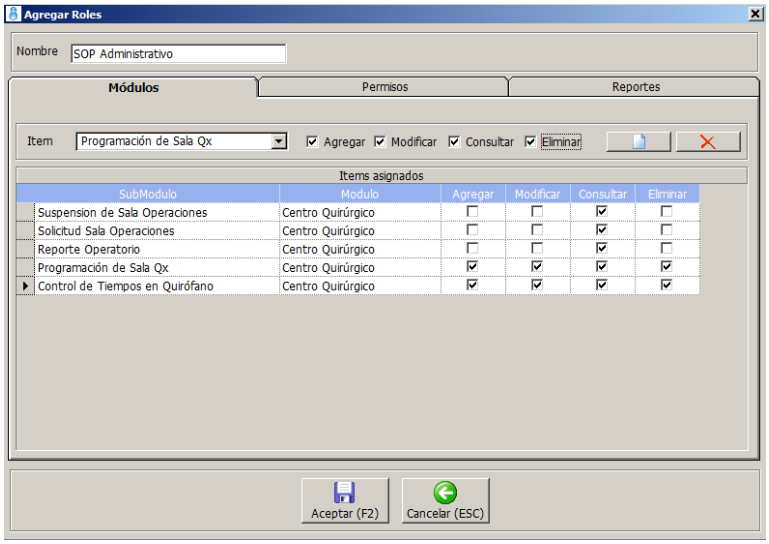


Fig. 69 Rol para Personal Administrativo Centro Quirúrgico

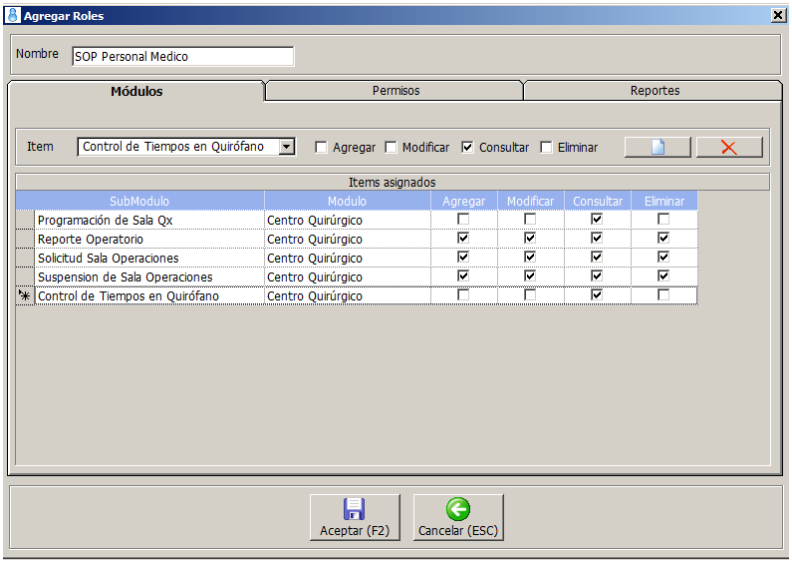


Fig. 70 Rol para Personal Médico Asistencial en Centro Quirúrgico

3.1.6 Fase: Mantenimiento

En esta fase se realiza el seguimiento al módulo de Centro Quirúrgico, donde se realizaron las siguientes acciones:

- Validaciones adicionales por parte del usuario médico de consulta externa como por ejemplo acceso directo de la atención de consulta externa, el histórico de cirugías y solicitudes de sala de operaciones para la toma de decisiones, esto mismo se agregó en hospitalización y emergencia.
- Mejorar la búsqueda de procedimientos históricos por diagnostico registrado los más usados en la parte superior para ayudar en la generación de solicitud de sala de operaciones.
- Validar que no se repita el médico principal con el médico ayudante, caso que no se había considerado.
- Se recibió la retroalimentación de expandir el tamaño de caracteres en el procedimiento del reporte operatorio.
- Tener la opción de procedimientos históricos por médico cirujano para ayuda en la tipificación de diagnósticos iguales en pacientes anteriores.
- Se realizaban capacitaciones por teléfono del uso para el registro de reporte operatorio y solicitud de sala de operaciones por sistema, ya que esto era obligatorio.

3.2 Tratamiento, análisis de datos y presentación de resultados

3.2.1 Tratamiento y análisis de datos

Para el tratamiento de los datos obtenidos, se aplicaron los instrumentos propuestos en dos momentos: pre test y post test.

Cuestionario de la variable independiente

Este instrumento estuvo orientado a obtener información sobre la implementación del módulo de centro quirúrgico en el sistema SisGalenPlus, a través de preguntas estructuradas en escala de Likert. Se evaluaron dimensiones como funcionalidad, facilidad de uso, acceso a la información y cumplimiento de los objetivos del sistema. Para ello, se recolectaron datos antes y después de la propuesta de implementación del módulo de centro quirúrgico.

Cuestionario de la variable dependiente

Para la obtención de datos de la variable dependiente, se aplicó un cuestionario diseñado para medir el impacto de la implementación del módulo de centro quirúrgico en el proceso asistencial, específicamente en términos de tiempo, calidad y satisfacción del personal asistencial y administrativo que utiliza directamente el módulo. Se empleó una escala de Likert de cinco niveles, aplicada en dos momentos: pre test y post test.

Ficha de observación

Utilizada para el registro directo de los tiempos de ejecución en los subprocesos de solicitud de sala de operaciones (dirigido a médicos asistenciales que solicitan una sala), programación de sala de operaciones (dirigido al personal administrativo encargado de programar y publicar la disponibilidad), y registro del reporte operatorio (dirigido a los médicos cirujanos que realizan la intervención). Esta ficha permitió captar datos reales del comportamiento operativo en el uso del módulo, antes y después de su implementación.

Ficha de cotejo

Este instrumento permitió verificar el cumplimiento de los pasos establecidos para los procesos quirúrgicos. La ficha se aplicó en dos momentos (pre y post) con el objetivo de identificar errores, omisiones o mejoras en la ejecución operativa.

En consecuencia, la presente investigación fue diseñada como un estudio de tipo aplicado y de diseño preexperimental, como se muestra en la Tabla XLVII, implicando la ejecución de tres fases fundamentales: pre test, implementación de la propuesta y post test. En la etapa de pre test se midieron los procesos existentes, mientras que en la fase de post test se evaluaron esos mismos procesos tras la incorporación de la solución tecnológica desarrollada. Esta comparación permitió medir el impacto de la variable independiente (el módulo de centro quirúrgico) sobre la variable dependiente (mejora del proceso de programación de sala de operaciones).

Tabla XLVII: Diseño de investigación pre test – post test

Datos obtenidos (Pre test)	Aplicación de Propuesta	Datos obtenidos (Post test)
(VD) O_1	(VI) X	(VD) O_2

Donde:

- O1: Pre test (datos obtenidos antes de implementación del módulo)
- X: Propuesta (módulo de centro quirúrgico)
- O2: Post test (datos obtenidos después de implementación del módulo)

En ese sentido aplicando el pre test y post test dirigidos a los procesos: registro de solicitudes de sala de operaciones, programación de sala de operaciones y registro del reporte operatorio, contenidos en el módulo de centro quirúrgico del sistema SisGalenPlus, se utilizó una aplicación móvil que permitió digitalizar el uso de la ficha de observación y registrar los tiempos y eventos relevantes en campo.

Los datos obtenidos fueron procesados en hoja de cálculo de Google para su codificación y análisis, aplicando medidas estadísticas descriptivas como promedios, desviación estándar y porcentaje, donde se eligió la prueba Z para comparar los promedios de las muestras obtenidas del pre y post test, considerando una distribución normal aproximada y un tamaño de muestra superior a 30 casos por muestra, permitiendo verificar si existía una diferencia estadísticamente significativa entre las dimensiones evaluadas antes y después de la implementación del módulo de centro quirúrgico.

Validación y confiabilidad de los instrumentos

Para determinar los instrumentos utilizados en la recolección de datos se debe cumplir con requisitos de validez y confiabilidad para asegurar la calidad de los resultados obtenidos. En esta investigación, los instrumentos aplicados fueron revisados por expertos en el área de salud, tecnología y gestión hospitalaria, quienes evaluaron la claridad, coherencia y pertinencia de los ítems incluidos. Esta validación por juicio de expertos se evidencia en los Anexos correspondientes.

Asimismo, se aplicó el coeficiente Alfa de Cronbach para evaluar la confiabilidad interna de los cuestionarios estructurados en escala tipo Likert. Este análisis se realizó utilizando el software estadístico IBM SPSS en versión prueba de 30 días, obteniendo valores superiores a 0.7, lo que indica un nivel de confiabilidad alto. El coeficiente α de Cronbach toma valores entre 0 y 1, donde 0 representa una confiabilidad nula y 1 una confiabilidad perfecta. Estos resultados respaldan la consistencia interna de los instrumentos utilizados para la recolección de datos.

Para el análisis de todos los datos obtenidos en los procesos que involucra el módulo de centro quirúrgico fueron agrupados en una matriz, ordenados y codificados por dimensión, se utilizó la prueba Z, ya que el tamaño de muestra utilizado ($n \geq 30$), esto permite asumir una distribución normal estándar bajo la hipótesis nula. En tal sentido, esta prueba evalúa la media de una población normalmente distribuida con varianza conocida, siendo adecuada para comparar dos muestras relacionadas con suficientes observaciones.

Una prueba Z es una prueba de hipótesis, el cual sigue la distribución normal estándar bajo la hipótesis nula, de modo que evalúa si existe una diferencia significativa entre las medias de dos grupos de muestras, dando los resultados obtenidos antes (pre test) y después (post test) de la implementación del módulo quirúrgico. En esta investigación se aplicó la prueba Z para diferenciar dos medias relacionadas, utilizadas para medir los mismos procesos evaluados en dos momentos distintos. La fórmula aplicada fue:

$$z = \frac{\overline{\mathrm{d}}}{(\sigma \mathrm{d}/\sqrt{\mathrm{N}})}$$

Donde:

- **z:** valor estadístico obtenido del análisis
- $\bar{d}$**:** media aritmética de las diferencias entre los resultados antes y después
- σd**:** desviación estándar de las diferencias entre ambos momentos
- **N:** tamaño de la muestra

Las hipótesis estadísticas planteadas fueron:

Hipótesis nula (H_0): $\mu_1 = \mu_2$ (no hay diferencia entre las medias del pre test y post test)

Hipótesis alternativa (Ha): $\mu_1 < \mu_2$ (la media del post test es mayor que la del pre test)

Durante esta etapa, se realizaron observaciones in situ a los procesos involucrados del módulo de centro quirúrgico (registro de solicitud, programación de sala, reporte operatorio), tanto en su ejecución manual (pre test) como con el uso del módulo implementado (post test), complementando así el análisis estadístico con evidencia empírica recolectada en campo.

Dimensión calidad de software (Cuestionario variable independiente)

El análisis de la dimensión de calidad del software refleja que el personal de salud y administrativo percibe positivamente la implementación del módulo quirúrgico del sistema SisGalenPlus. Los puntajes obtenidos en las escalas de Likert (entre 3.76 y 4.94) reflejan una alta aceptación del sistema en términos de funcionalidad, usabilidad y eficiencia, validado estadísticamente mediante la prueba Z ($p < 0.05$).

Completitud funcional

Grado en que el sistema proporciona todas las funcionalidades necesarias para cubrir los requisitos del usuario.

Tabla XLVIII: Cuestionario de la variable independiente, completitud funcional

Ítem	Pregunta	Promedio (escala de Likert)
P01	¿El módulo cubre todas las necesidades funcionales del proceso de programación?	4.00
P13	¿Cumple con las expectativas de la jefatura en verificación y aprobación?	4.94
Promedio general		**4.47**

En la Tabla XLVIII se observa que el personal percibe que el módulo quirúrgico ahora cubre de manera completa las funciones necesarias en todo el flujo de programación, incluyendo verificación, asignación y validación superior.

Corrección funcional

Grado en que el sistema proporciona resultados correctos, sin errores y confiables. En la Tabla XLIX se observa que la percepción sobre la exactitud y seguridad de los datos quirúrgicos mejoró notablemente, esto refuerza la confianza del personal en el uso clínico del sistema.

Tabla XLIX: Cuestionario de la variable independiente, corrección funcional

Ítem	Pregunta	Promedio (escala de Likert)
P02	¿El módulo garantiza que los datos registrados sean correctos y sin errores?	4.35
P06	¿Garantiza la protección y fiabilidad de los datos personales y quirúrgicos?	4.18
Promedio general		**4.47**

Pertinencia funcional

Grado en que las funciones del sistema están alineadas con las necesidades reales del usuario.

Tabla L: Cuestionario de la variable independiente, pertinencia funcional

Ítem	Pregunta	Promedio (escala de Likert)
P03	¿El módulo está alineado con las tareas reales del centro quirúrgico?	3.76
P11	¿Satisface las necesidades de los médicos en programación quirúrgica?	4.71
P12	¿Facilita el trabajo del personal administrativo?	4.29
Promedio general		**4.25**

En la Tabla XL se observa que el sistema es visto como más útil y alineado con las tareas operativas tanto para médicos como para personal administrativo, lo que facilita su integración en la práctica diaria.

Capacidad de entender fácilmente

Grado en que los usuarios pueden comprender el módulo desde el primer contacto.

Tabla LI: Cuestionario de la variable independiente, capacidad de entender fácilmente

Ítem	Pregunta	Promedio (escala de Likert)
P04	¿Es fácil comprender las funciones del módulo desde su primera interacción?	3.94
Promedio general		**3.94**

En la Tabla LI se observa que el módulo se ha vuelto más claro y comprensible para nuevos usuarios facilitando su adopción sin requerir entrenamiento intensivo.

Capacidad de aprender fácilmente

Grado en que los usuarios pueden aprender a utilizar el sistema con rapidez y sin asistencia extensa.

Tabla LII: Cuestionario de la variable independiente, capacidad de aprender fácilmente

Ítem	Pregunta	Promedio (escala de Likert)
P05	¿Permite operar sin asistencia adicional?	4.35
Promedio general		**4.35**

En la Tabla LII se observa que el sistema es percibido como intuitivo, reduciendo la necesidad de soporte técnico o capacitación continua.

Capacidad de operar fácilmente

Grado en que los usuarios pueden usar el sistema de forma eficiente y sin dificultad.

Tabla LIII: Cuestionario de la variable independiente, capacidad de operar fácilmente

Ítem	Pregunta	Promedio (escala de Likert)
P07	¿Permite doble verificación de programaciones quirúrgicas?	4.47
P08	¿El tiempo de registro de solicitudes es adecuado?	4.16
P09	¿Permite generar reportes de forma eficiente?	4.53
P10	¿El tiempo para aprobar solicitudes es apropiado?	4.41
Promedio general		**4.39**

En la Tabla LIII se observa la operatividad del sistema se percibe como mucho más ágil y funcional, especialmente en tareas críticas como registrar solicitudes, generar reportes y validar cirugías.

El análisis muestra una percepción del personal sobre la efectividad y utilidad del módulo quirúrgico. Esta mejora ha sido estadísticamente significativa, validada mediante la prueba Z ($p < 0.05$). Se observaron en los ítems relacionados con la completitud funcional y la corrección funcional, lo cual refleja que el sistema satisface de forma más integral las funciones requeridas y mejora la calidad del registro. Asimismo, el módulo es percibido como más intuitivo, fácil de operar y de aprender, lo cual favorece su adopción por parte del personal. Por otro lado, en los ítems operativos (P08 a P13) muestran como el sistema contribuye a disminuir el tiempo de registro, generar reportes, facilitar la programación y responder a las necesidades del personal médico y administrativo, alineándose con criterios de eficiencia y pertinencia funcional.

Dimensión tiempo - Variable dependiente - (Ficha de observación)

La dimensión de tiempo fue evaluada mediante la ficha de observación, aplicada a los subprocesos principales del módulo de centro quirúrgico. Se realizaron tres ciclos de observación tanto antes (pre test) como después (post test) de la implementación del módulo, registrando la duración en minutos para cada proceso. A continuación, de la Tabla LIV a la Tabla LVII se detallan los resultados obtenidos.

Tabla LIV: Datos para Tiempo de registro de solicitudes de sala de operaciones

Tiempo de registro de solicitudes de sala de operaciones			
Ciclo	**Pre test (min)**	**Post test (min)**	**% Reducción**
C1	12	7	41.67%
C2	10	6	40.00%
C3	14	7	50.00%
Promedio	**12.00**	**6.67**	**44.44%**

Tabla LV: Datos para Tiempo de aprobación de solicitudes de sala de operaciones

Tiempo de aprobación de solicitudes de sala de operaciones			
Ciclo	**Pre test (min)**	**Post test (min)**	**% Reducción**
C1	3	2	33.33%
C2	4	3	25.00%
C3	5	4	20.00%
Promedio	**4**	**3**	25.00%

Tabla LVI: Datos para Tiempo de generación de programación diaria

Tiempo de generación de programación diaria			
Ciclo	**Pre test (min)**	**Post test (min)**	**% Reducción**
C1	12	6	50.00%
C2	13	8	38.46%
C3	12	7	41.67%
Promedio	**12.33**	**7**	43.24%

Tabla LVII: Datos para Tiempo de registro de reporte operatorio

Tiempo de registro de reporte operatorio			
Ciclo	**Pre test (min)**	**Post test (min)**	**% Reducción**
C1	25	15	40.00%
C2	24	16	33.33%
C3	22	17	22.73%
Promedio	**23.67**	**16.00**	32.39%

Los resultados muestran una reducción significativa en el tiempo de ejecución de los procesos quirúrgicos observados. En el proceso de registro de solicitud quirúrgica, se redujo un 44.44%. En la aprobación de solicitudes quirúrgicas, la reducción fue del 25.00%. En la programación de sala, se redujo un 43.24%, y en el reporte operatorio, un 32.39%.

Estos resultados reflejan no solo mejoras objetivas en los tiempos reales observados, sino también una mejora percibida significativa por parte del personal en cuanto a la eficiencia temporal del sistema.

Dimensión de satisfacción – Variable dependiente (Cuestionario Variable Dependiente)

La dimensión de satisfacción fue evaluada mediante un cuestionario estructurado en nueve ítems, aplicado al personal asistencial y administrativo del centro quirúrgico. Esta dimensión agrupó tres dimensiones, cada ítem fue valorado en una escala Likert del 1 al 5.

Satisfacción al proceso en la solicitud de sala de operaciones y generación del reporte operatorio.

Esta categoría de agrupación mide el grado de satisfacción del personal asistencial en el proceso de registro de solicitud de sala de operación y el reporte operatorio:

Tabla LVIII: Datos de proceso en la solicitud de sala de operaciones y generación del reporte operatorio

Satisfacción al proceso en la solicitud de sala de operaciones y generación del reporte operatorio				
Ítem	**Preguntas del tiempo de atención**	**Pre test (escala Likert)**	**Post test (escala Likert)**	**% Mejora**
P05	¿Qué tan satisfecho estás con el tiempo requerido para registrar una solicitud de programación de sala quirúrgica?	2.29	4.26	86.03%
P06	¿Qué tan satisfecho estás con la rapidez en la generación de reportes quirúrgicos desde la programación de salas?	2.29	4.21	83.84%
P07	¿Qué tan satisfecho estás con el soporte del módulo en la solicitud y programación de salas quirúrgicas para los médicos?	2.47	4.33	75.30%
Promedio general		2.35	4.27	81.70%

En la Tabla LVIII se observa los resultados mostrando una mejora consistente; asimismo, el promedio de las respuestas en el pre test fue de 2.35, mientras que en el post test se elevó a 4.27, reflejando un incremento del 81.70%. Esta mejora evidencia que el módulo de centro quirúrgico satisface de forma positiva los procesos de generación de solicitud de sala de operaciones, así como el reporte operatorio por parte de los médicos.

Satisfacción en el proceso de la programación y disponibilidad diaria de sala de operaciones.

Esta categoría incluyó preguntas enfocadas en evaluar la percepción de satisfacción en el proceso de la programación quirúrgica.

Tabla LIX: Datos en el proceso de la programación y disponibilidad diaria de sala de operaciones

Satisfacción en el proceso de la programación y disponibilidad diaria de sala de operaciones				
Ítem	**Preguntas del tiempo de atención**	**Pre test (escala Likert)**	**Post test (escala Likert)**	**% Mejora**
P01	¿Qué tan satisfecho estás con la precisión de los horarios asignados en la programación de salas de operaciones?	1.88	4.05	115.43%
P02	¿Qué tan satisfecho estás con la fiabilidad de los datos registrados sobre los pacientes en la programación?	2	4.1	105.00%
P04	¿Qué tan satisfecho estás con el tiempo que toma generar una programación diaria de salas?	2	4.13	106.50%
P08	¿Qué tan satisfecho estás con la facilidad del módulo para gestionar la programación diaria de salas quirúrgicas?	2.24	4.26	90.18%
Promedio general		**2.03**	**4.14**	**103.94%**

En la Tabla LIX se observa los resultados indican un crecimiento de 2.03 a 4.14 puntos promedio, lo que representa una mejora del 103.94%. Este cambio evidencia una mejora en el proceso de registro de la programación de sala de operaciones por el personal de centro quirúrgico, en la disminución de tiempo y la fiabilidad de los datos, ya que estos provienen de un correcto registro de las solicitudes, evitando revisiones continuas para la publicación de la programación de sala de operaciones.

Satisfacción en el proceso de verificación y aprobación de cirugías y procedimientos programados.

La última agrupación del cuestionario midió directamente la percepción de satisfacción, la doble verificación y la aprobación de las solicitudes de sala de operaciones para la programación quirúrgica.

Tabla LX: Datos para el proceso Satisfacción en el proceso de verificación y aprobación de cirugías y procedimientos programados

Satisfacción en el proceso de verificación y aprobación de cirugías y procedimientos programados				
Ítem	**Preguntas del tiempo de atención**	**Pre test (escala Likert)**	**Post test (escala Likert)**	**% Mejora**
P03	¿Qué tan satisfecho estás con el proceso de doble verificación de las programaciones antes de su aprobación?	2.29	4	74.67%
P09	¿Qué tan satisfecho estás con el soporte del módulo para la aprobación de cirugías programadas?	2.76	4.33	56.88%
Promedio general		2.5	4.2	**68.00%**

Esta dimensión alcanzó una mejora del 68.00% como se muestra en la tabla LX, al pasar de un promedio de 2.5 en el pre test a 4.2 en el post test. En esta parte se evidencia una buena aceptación por parte del personal, quienes valoran positivamente la funcionalidad del módulo, su facilidad de uso y su utilidad en las aprobaciones de las solicitudes del centro quirúrgico.

El análisis global de los nueve ítems de la dimensión satisfacción arroja un promedio total de 2.29 en el pre test y 4.20 en el post test, lo que representa una mejora del 83.40%

Dimensión calidad – Variable Dependiente (ficha de cotejo)

La ficha de cotejo fue diseñada para evaluar el cumplimiento efectivo de los pasos definidos en los procedimientos quirúrgicos. Este instrumento fue aplicado antes y después de la implementación del módulo del Centro Quirúrgico del sistema SisGalenPlus.

Cada proceso fue descompuesto en pasos críticos que debían cumplirse según protocolos establecidos. A través del análisis documental, se contabilizó cuántos pasos eran correctamente ejecutados.

Tabla LXI: Datos obtenidos para dimensión de calidad

Datos obtenidos para la dimensión calidad						
Indicador	**Total, esperado**	**Cumplidos (Pre test)**		**Cumplidos (Post test)**		**% mejora**
Nivel de Fiabilidad en la disponibilidad de horarios en salas de operaciones	3	1.33	44.43%	3.00	100%	125%
Nivel de fiabilidad de datos personales correctos de los pacientes	3	1.67	55.67%	3.00	100%	80%
Nivel de fiabilidad de doble revisión y aprobación de cirugías solicitadas	3	1.33	44.33%	3.00	100%	125%

En la Tabla LXI se muestra que después de la implementación del módulo, no se reportan conflictos ni retrasos en los horarios quirúrgicos. El sistema muestra mejor control y sincronización. Así mismo se evidencia un cambio en la calidad de los datos registrados, especialmente en exactitud y reducción de errores, lo que fortalece la fiabilidad de los datos, por otro lado, el módulo asegura un flujo de validación formal y documentado, involucrando a la jefatura y reduciendo errores previos de aprobación y verificación a la cirugía.

La ficha de cotejo fue diseñada de acuerdo con los indicadores definidos en la tabla de operacionalización de variables. Los resultados obtenidos muestran mejoras significativas en las tres dimensiones observadas. En cuanto a la disponibilidad de horarios, se pasó de un cumplimiento parcial del 44.43% a un cumplimiento total, lo cual demuestra mayor sincronización y actualización de programación en el sistema. La fiabilidad de los datos personales alcanzó una mejora del 80%, destacando la precisión y coherencia en la documentación de pacientes. Finalmente, la dimensión de doble revisión y aprobación mostró una mejora del 125%, reflejando que el módulo garantiza validaciones consistentes por parte de la jefatura, minimizando omisiones y errores en la programación quirúrgica.

Resumen comparativo general

A continuación, se presenta un resumen global de los resultados obtenidos tras la aplicación de los cuatro instrumentos utilizados en la investigación.

Tabla LXII: Resumen comparativo general de los instrumentos

Instrumento	Indicadores evaluados	Mejora observada
Cuestionario Variable Dependiente	Satisfacción, calidad, percepción del tiempo	Mejora en +83.40% respecto de la percepción de Satisfacción
Ficha de observación	Tiempo real en 4 procesos quirúrgicos	Reducción del tiempo en 37.18%
Ficha de cotejo	Cumplimiento de pasos en procesos quirúrgicos	Ha habido un 125% de mejora

La Tabla LXII muestra el resumen consolidando los resultados y demuestra que la implementación del módulo quirúrgico del sistema SisGalenPlus produjo mejoras sustanciales tanto en términos de eficiencia, calidad, percepción de usuarios en los procesos, con resultados validados estadísticamente

3.2.1.1 Contrastación de hipótesis

Normalidad de datos

Para determinar la prueba de hipótesis se realizó la prueba de normalidad a los datos de la variable dependiente; con un nivel de confianza del 95% con un margen de error del 5%; considerando las hipótesis siguientes:

- **H_0:** Los datos de la variable programación de salas de operaciones en el centro quirúrgico es normal presentan una distribución normal
- **H_a:** Los datos de la variable programación de salas de operaciones en el centro quirúrgico es normal no presentan una distribución normal

Criterio de decisión

- Se rechaza H_0 si sig < 0.05, caso contrario aceptar H_0
- Si sig. > 0.05, entonces aceptar H_0

Los datos obtenidos fueron procesados en IBM SPSS (software estadístico de versión de prueba por 30 días) y se tiene en cuenta la cantidad de datos que se tiene; para este caso se toma en cuenta el nivel de significancia de la prueba Shapiro – Wilk ($n \leq 50$) y nuestra muestra es de 38 profesionales de la salud.

Tabla LXIII: Prueba de normalidad del pre test

Pruebas de normalidad						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
TOTAL_PRE_TEST_VD	,128	38	,118	,962	38	,220
a. Corrección de significación de Lilliefors						

Tabla LXIV: Prueba de normalidad del post test

Pruebas de normalidad						
	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
TOTAL_POST_TEST_VD	,122	38	,165	,952	38	,101
a. Corrección de significación de Lilliefors						

En la Tabla LXIII y Tabla LXIV se observa la prueba de normalidad siendo p-valor del conjunto de datos es mayor a 0.05 para el pre test (0.220 > 0.05), post test 0.101 > 0.05 se acepta la hipótesis H_0; por lo tanto, la distribución de los datos de la variable programación de salas de operaciones en el centro quirúrgico presenta normalidad de datos; asimismo, se opta por usar una prueba paramétrica para muestras relacionadas como la distribución Z (N>30).

Formulación de la hipótesis general

H_0: La implementación del módulo de centro quirúrgico no impacta de manera positiva en la mejora de la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima.

No existen diferencias significativas entre los resultados obtenidos antes y después de la implementación del módulo quirúrgico.

H_a: La implementación del módulo de centro quirúrgico impacta de manera positiva en la mejora de la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima.

Existen diferencias significativas entre los resultados obtenidos antes y después de la implementación del módulo quirúrgico

Nivel de significancia

$$\alpha = 0.05$$

Valor estadístico

Se realiza la prueba estadística Z para la comparación de medias pre y post test de los datos obtenidos.

Tabla LXV: Contrastación de hipótesis

Instrumento	Dimensiones Evaluadas	Valor Z calculado	Nivel de significancia	Resultado del contraste
Ficha de observación	Dimensión de Tiempo	> Zα	0.05	Se rechaza H_0
Cuestionario variable dependiente	Dimensión de Satisfacción	> Zα	0.05	Se rechaza H_0
Ficha de cotejo	Dimensión de calidad	> Zα	0.05	Se rechaza H_0

En la Tabla LXV se observa que en todos los casos el valor de Z calculado fue mayor al valor $Z\alpha = 1.96$, y el p-valor ($0.02500 < 0.05$) se rechaza la H_0; concluyendo, que existen diferencias estadísticamente significativas entre los resultados obtenidos antes y después de la implementación del módulo quirúrgico del sistema SisGalenPlus, confirmando así la validez de la hipótesis alternativa planteada en esta investigación.

Para determinar si la implementación del módulo quirúrgico generó mejoras en los procesos evaluados, se aplicó la prueba estadística Z para comparar los promedios de las muestras pre test y post test.

Dimensión de tiempo

La ficha de observación midió los tiempos reales (en minutos) empleados en cada subproceso quirúrgico antes y después de la implementación. Se proceso la prueba Z para muestras pareadas obteniendo $Z\alpha = 3.466$, y el p-valor ($0.0002701 < 0.05$) se rechaza la H_0; por lo tanto, la implementación del módulo de centro quirúrgico impacta de manera positiva en la dimensión

tiempo de las actividades de la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima; además, hay diferencias significativas entre los resultados obtenidos antes y después de la implementación. Asimismo, se obtuvo el porcentaje de mejoras en: registro de solicitud con una reducción de -44.44%, aprobación de solicitud con -25%, programación de sala con -43.24%, y registro de reporte operatorio con -32.39% como se detalla en la Tabla LXVI.

Tabla LXVI: Prueba Z para la dimensión tiempo

Dimensión tiempo			
Indicadores	**Pre test**	**Post test**	**% Mejora**
Tiempo promedio usado para registrar la solicitud quirúrgica	12.00	6.67	44.44%
Tiempo promedio usado para aprobar la solicitud quirúrgica	4.00	3.00	25.00%
Tiempo promedio usado para programar la sala de operaciones	12.33	7	43.24%
Tiempo promedio usado para registrar el reporte operatorio	23.7	16.00	32.39%
Media	13.00	8.17	**37.18%**
Varianza	65.64	30.55	
Desviación estándar	8.10	5.52	

Dimensión de satisfacción

Este cuestionario evaluó la calidad, tiempo y satisfacción del proceso de programación quirúrgica. Se procesó la prueba Z para determinar diferencias en las medias pre y post test de la dimensión satisfacción; obteniendo $Z\alpha = 2.330$, y el p-valor ($0.00990 < 0.05$) se rechaza la H_0; por lo tanto, la implementación del módulo de centro quirúrgico impacta de manera positiva en la dimensión satisfacción en los procesos de la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima; además, hay diferencias significativas entre los resultados obtenidos antes y después de la implementación. Asimismo, se obtuvo el porcentaje de mejoras en: percepción de satisfacción al proceso de solicitud de sala de operaciones y reporte operatorio con +81.70%, percepción de satisfacción en el proceso de programación diaria en sala de operaciones con +103.94%, y finalmente una percepción de satisfacción en el proceso de verificación y aprobación de sala quirúrgica con +68.00% como se detalla en la Tabla LXVII.

Tabla LXVII: Prueba Z para la dimensión para la dimensión de satisfacción

Dimensión de satisfacción			
Indicadores	**Pre test**	**Post test**	**% Mejora**
Satisfacción al proceso en la solicitud de sala de operaciones y generación del reporte operatorio	2.35	4.27	81.70%
Satisfacción en el proceso de la programación y disponibilidad diaria de sala de operaciones	2.03	4.14	103.94%
Satisfacción en el proceso de verificación y aprobación de cirugías y procedimientos solicitados.	2.5	4.2	68.00%
Media	13.0075	8.1675	
Varianza	65.646	30.552	
Desviación estándar	8.10224	5.52741	

Dimensión de calidad

La ficha de cotejo evaluó el cumplimiento de pasos críticos del proceso quirúrgico (cumple / no cumple). Se procesó la prueba Z para determinar diferencias en las medias pre y post test de la dimensión satisfacción; obteniendo $Z\alpha = 1.85$, y el p-valor ($0.03216 < 0.05$) se rechaza la H_0; por lo tanto, la implementación del módulo de centro quirúrgico impacta de manera positiva en la dimensión calidad respecto al nivel de fiabilidad; además, hay diferencias significativas entre los resultados obtenidos antes y después de la implementación. Asimismo, se obtuvo el porcentaje de mejoras en: nivel de fiabilidad de horarios con más de 125%, nivel de fiabilidad de datos personales con más de 80%, y finalmente la validación por doble revisión con más 125% como se observa en la Tabla LXVIII.

Tabla LXVIII: Prueba Z para la dimensión calidad

Dimensión calidad					
	Pre test		**Post test**		**% mejora**
Nivel de fiabilidad en la disponibilidad de horarios en salas de operaciones	1.333	44.43%	3	100%	125%
Nivel de fiabilidad de datos personales correctos de los pacientes	1.667	56%	3	100%	80%
Nivel de fiabilidad de doble revisión y aprobación de cirugías solicitadas	1.333	44%	3	100%	125%
Media	1.4433		3.000		
Varianza	0.039		0.000		
Desviación estándar	0.19630		0.00000		

3.2.2 Presentación de resultados

En el presente capítulo se muestran los resultados obtenidos a partir de la aplicación de los instrumentos de recolección de datos establecidos en el diseño metodológico. Dichos instrumentos fueron aplicados a una muestra conformada por personal asistencial y administrativo del centro quirúrgico antes y después de la implementación del módulo del sistema SisGalenPlus.

La presentación de resultados se estructura de acuerdo a los instrumentos utilizados: cuestionario de la variable independiente, cuestionario de la variable dependiente, ficha de observación y ficha de cotejo. Para la variable dependiente se detallan los resultados obtenidos en la fase pre test y post test, con sus respectivos análisis comparativos, gráficos y tablas de apoyo.

Variable independiente

Resultados para calidad de software

Este instrumento evaluó la percepción del personal sobre la implementación del módulo quirúrgico, considerando sub características como completitud funcional, corrección funcional, pertinencia funcional, y aspectos de usabilidad como la facilidad de comprensión, aprendizaje y operación. Se aplicó una escala de Likert de 1 a 5.

Completitud funcional

Respecto a la completitud funcional de acuerdo a la Fig. 71 del módulo de centro quirúrgico desarrollado, se tiene que en promedio un 53% de los usuarios están “totalmente de acuerdo” indicando que las funciones cubren todas las tareas para la programación de sala de operaciones, además que permite lograr los objetivos de los procesos involucrados, 31% de “acuerdo” menciona que las datos resultante del sistema les ayuda con tareas relacionadas al proceso de programación de sala de operaciones; sin embargo, el 10% esta “indeciso” porque mencionan que se tienen que registrar datos adicionales que antes no se registraba, dado que existían personal de digitación que se encargaban de recolectar la información de la historia clínica.

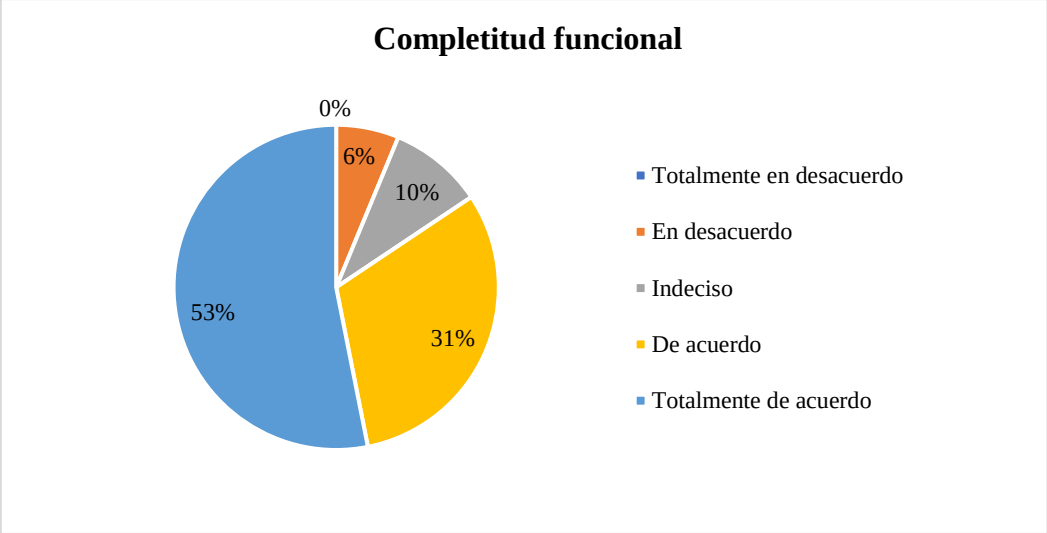


Fig. 71: Resultados gráficos de la completitud funcional

Corrección funcional

Respecto a la Corrección funcional de acuerdo a la Fig. 72 al utilizar las interfaces del módulo de centro quirúrgico, se tiene que en promedio un 50% de los usuarios están "totalmente de acuerdo" mencionando que las correcciones cubren todas las tareas para la programación de sala de operaciones, el proceso de registro de solicitud de sala de operaciones como también la programación de sala dentro del módulo, además que permite realizar modificaciones del acto médico validado por permisos, perfil y roles, 34% de "acuerdo" menciona que los roles y permisos para permitir modificar o corregir los actos médicos sea más flexible en el sistema; sin embargo, el 10% esta "indeciso" debido al no integrar el sistema a un módulo web.

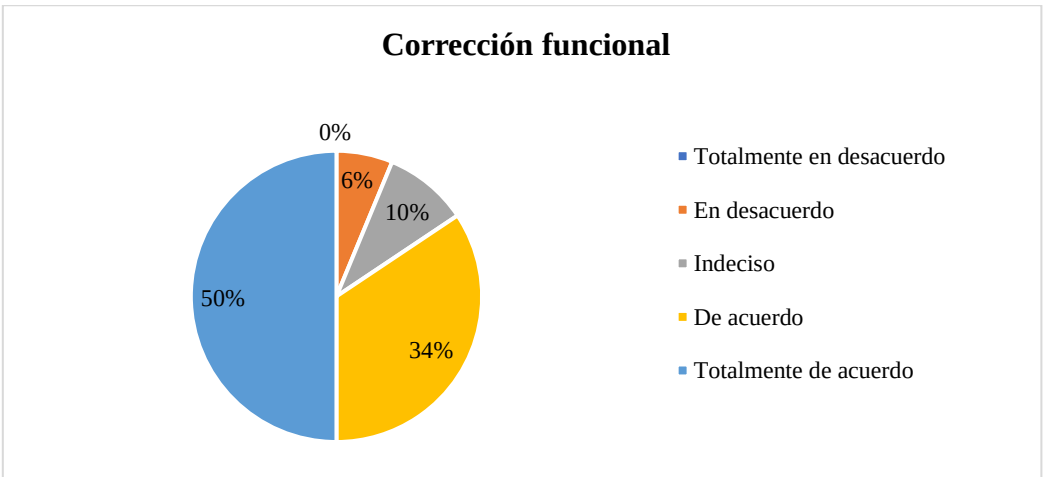


Fig. 72: Resultados gráficos de la corrección funcional

Capacidad de entender fácilmente

Respecto a la capacidad de entender fácilmente las nuevas interfaces que se necesitan para realizar los procesos del módulo de centro quirúrgico de acuerdo a la Fig. 73 al utilizar las interfaces del módulo de centro quirúrgico, se tiene que en promedio un 69% de los usuarios están "totalmente de acuerdo" mencionando que las interfaces para realizar todas las tareas como la programación de sala de operaciones, el proceso de registro de solicitud de sala de operaciones y la programación de sala dentro del módulo son intuitivas para ubicar y registro de datos; 6% de "acuerdo" menciona que las interfaces deben mostrarse como acceso directo desde los diferentes procesos del sistema; sin embargo, el 13% está "en desacuerdo" debido a que debería registrarse menos información, dejando de lado la captura de indicadores importantes para el trabajo estadístico.

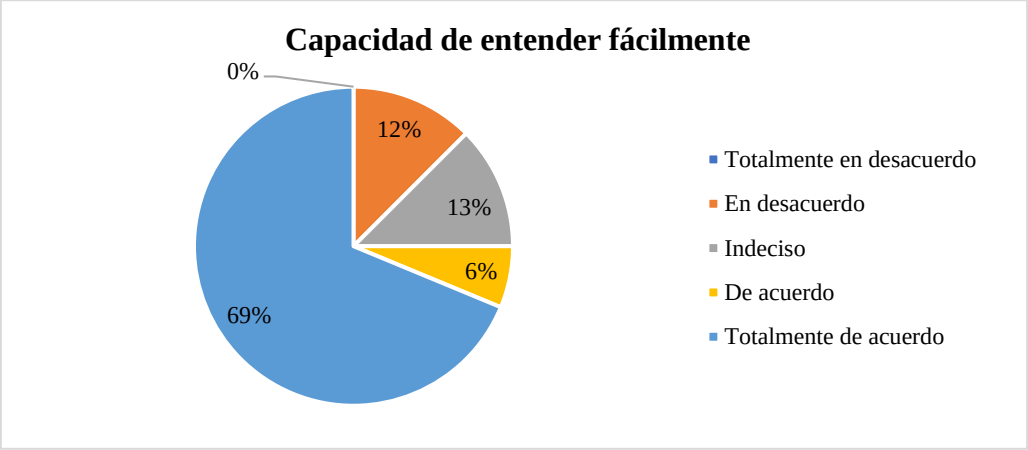


Fig. 73: Resultados gráficos de la capacidad de entender fácilmente

Variable dependiente

Resultados para dimensión satisfacción

Para medir la dimensión satisfacción de la variable dependiente se utilizó un cuestionario empleando un diseño de "antes y después", para medir la satisfacción en tres procesos como: la satisfacción al proceso en la solicitud de sala de operaciones y generación del reporte operatorio, donde mostró un incremento en la percepción de satisfacción del 71.56%; también se midió, la satisfacción en el proceso de la programación y disponibilidad diaria de sala de operaciones, observando un aumento de 103.69%; y por último, la satisfacción en el proceso de verificación y aprobación de cirugías programados con un incremento del 64.95%, esto

debido a la implementación del módulo de centro quirúrgico. Los resultados se detallan en la Fig. 74.

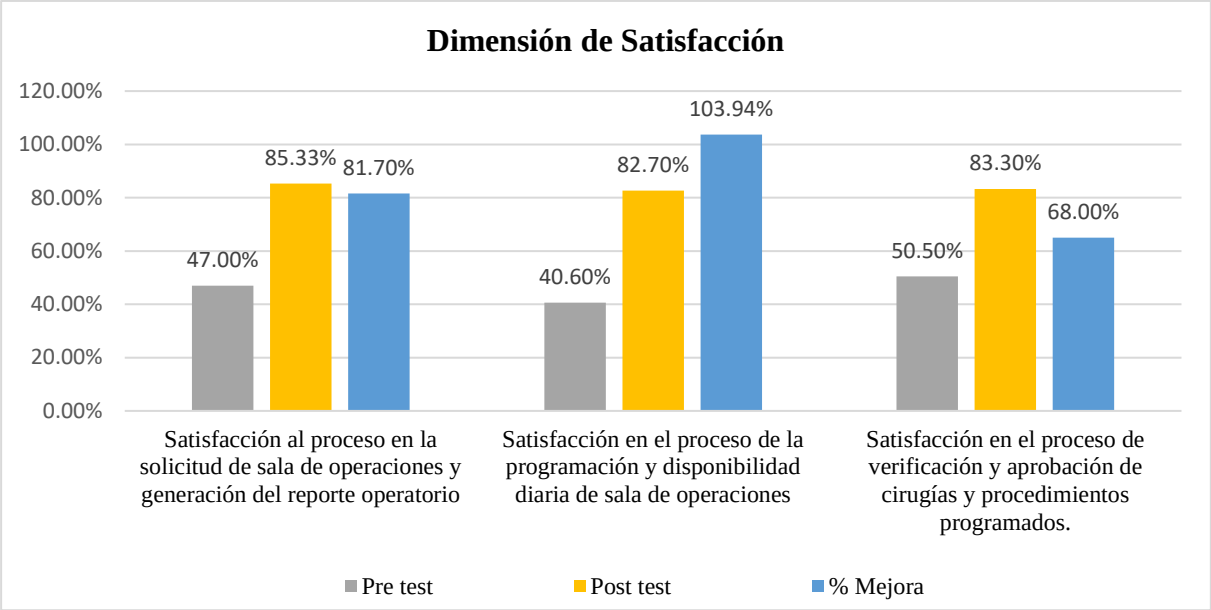


Fig. 74: Resultados de la dimensión de satisfacción

En la Fig. 72 se observa que la dimensión satisfacción alcanzó un incremento del 83.40% en promedio, por lo tanto, se observa mejoras significativas por el personal en su rutina operativa.

Resultados para dimensión del tiempo

La ficha de observación fue aplicada para medir el tiempo real que tomaban los procesos quirúrgicos en cuatro subprocesos clave: registro de solicitud, aprobación, programación y reporte operatorio. Los resultados mostraron reducciones de tiempo en todos los casos, con un promedio general del 37.18%. En este proceso, se observa una reducción significativa en el tiempo requerido para registrar una solicitud quirúrgica. Antes de la implementación del módulo, el tiempo promedio entre los tres ciclos fue de 12.00 minutos, luego de la implementación se redujo a 6.67 minutos, esto representa una disminución del tiempo en 5.33 minutos equivalente a un 44%. Esto se atribuye a la digitalización de los campos obligatorios, la validación automática de datos, y la simplificación del flujo de registro en el módulo del sistema SisGalenPlus como se detalla en la Fig. 75.

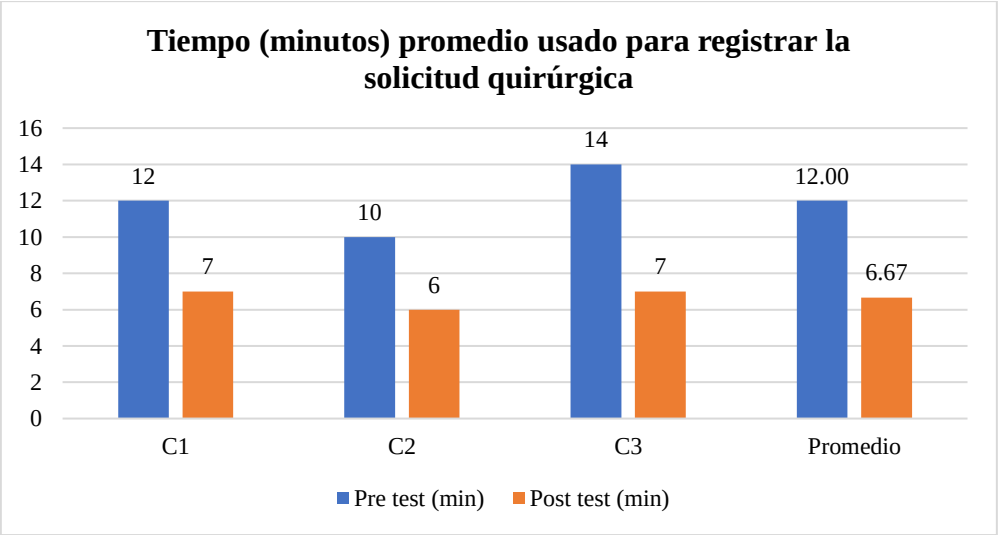


Fig. 75: Resultados Tiempo de registro de solicitudes de sala de operaciones por los médicos.

En la Fig. 76 se muestra el tiempo necesario para la aprobación de solicitudes por parte de jefaturas o responsables también muestra una mejora destacada. En el pre test, el promedio era de 4 minutos, mientras que en el post test disminuyó a 3 minutos, lo que representa una mejora del 25.00%. Esta reducción se debe a la disponibilidad inmediata de la solicitud digital en el sistema y a la visibilidad jerárquica que permite revisar y aprobar con firma digital integrada.

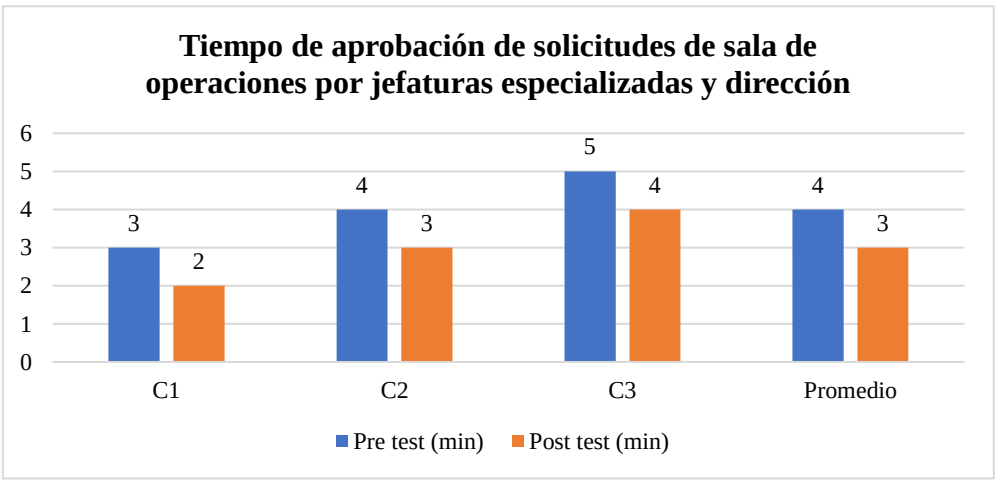


Fig. 76: Resultados Tiempo de aprobación de solicitudes de sala de operaciones por jefaturas especializadas y dirección

La generación de programación diaria de sala de operación por el personal administrativo de centro quirúrgico mejoró notablemente logrando una disminución de 5.33 minutos, con un promedio en el pre test de 12.33 minutos para registrar la programación de una solicitud, mientras que en el post test, los datos de las programaciones se agregan automáticamente en su mayoría, representando en el post test de 7 minutos en promedio para asignar el turno hora de

inicio y fin, disponibilidad de sala y disponibilidad de médico anestesiólogo. Los datos expresados en minutos en cada ciclo se muestran en la Fig. 77.

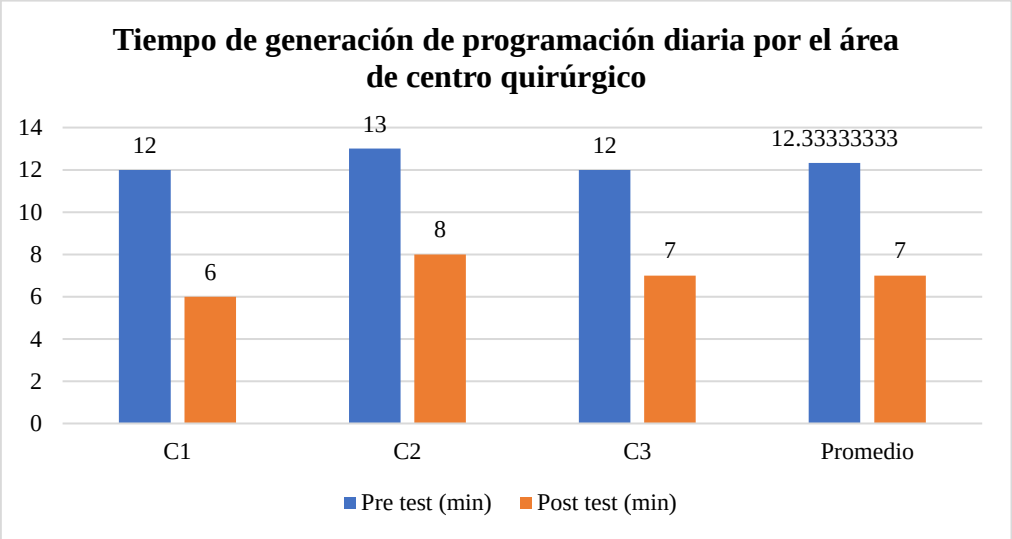


Fig. 77: Resultados Tiempo de aprobación de solicitudes de sala de operaciones por jefaturas especializadas y dirección

El registro del reporte operatorio mostró también una mejora tras la implementación del módulo. En el pre test, el promedio fue de 23.67 minutos, mientras que en el post test fue de 16 minutos, representando una reducción del 32.39%. Esta mejora se atribuye a la integración de campos estandarizados, digitación del equipo quirúrgico pre registrado y la automatización parcial de datos clínicos dentro del módulo quirúrgico como el registro del procedimiento (CPMS), diagnóstico (CIE10), lo cual facilita y acelera el registro del reporte operatorio en sala, como se observa en la Fig. 78.

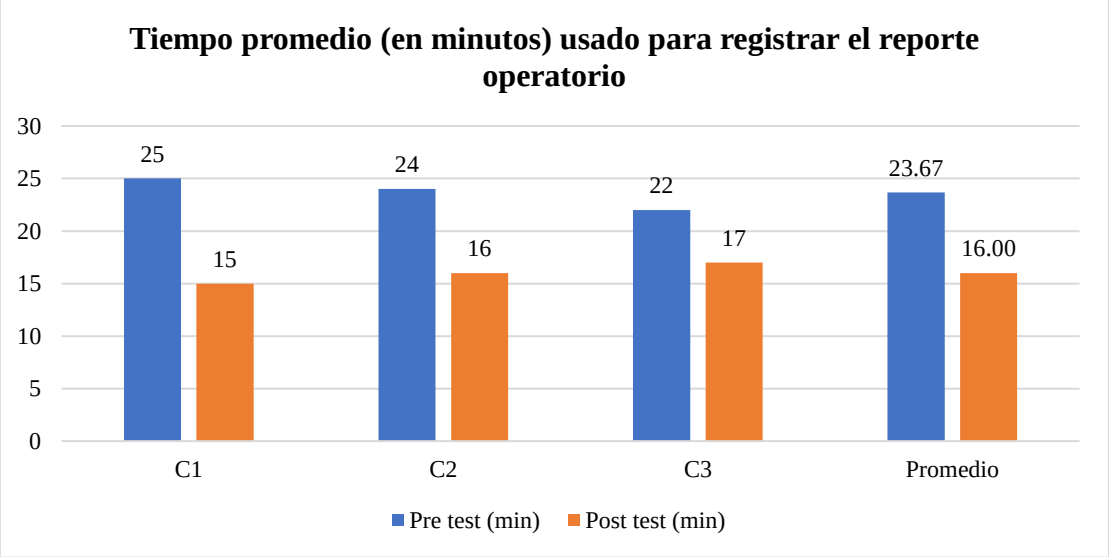


Fig. 78: Resultados Tiempo de registro de reporte operatorio de cirugía o procedimiento por los médicos.

En esta vista consolidada se comparan los promedios generales de los cuatro subprocesos medidos: aprobación de solicitudes, programación de sala, registro de solicitud y registro del reporte operatorio. Todos muestran reducciones consistentes entre el pre test y post test.

En la Fig. 79 se muestra la reducción de tiempo más destacada que se da en el proceso de registro de solicitud de sala quirúrgica, con una reducción de 5.33 minutos representando una reducción de tiempo del 44.44%. Le siguen la programación de salas con disminución del tiempo del 43.24%, luego el registro de reporte operatorio que ha disminuido en un 32.39%. En conjunto, estos resultados evidencian una mejora general de más del 36% en tiempo del proceso quirúrgico gracias al módulo implementado.

Estas mejoras operativas evidencian que el módulo no solo mejora la percepción del sistema, sino que tiene un impacto tangible en la eficiencia del flujo quirúrgico.

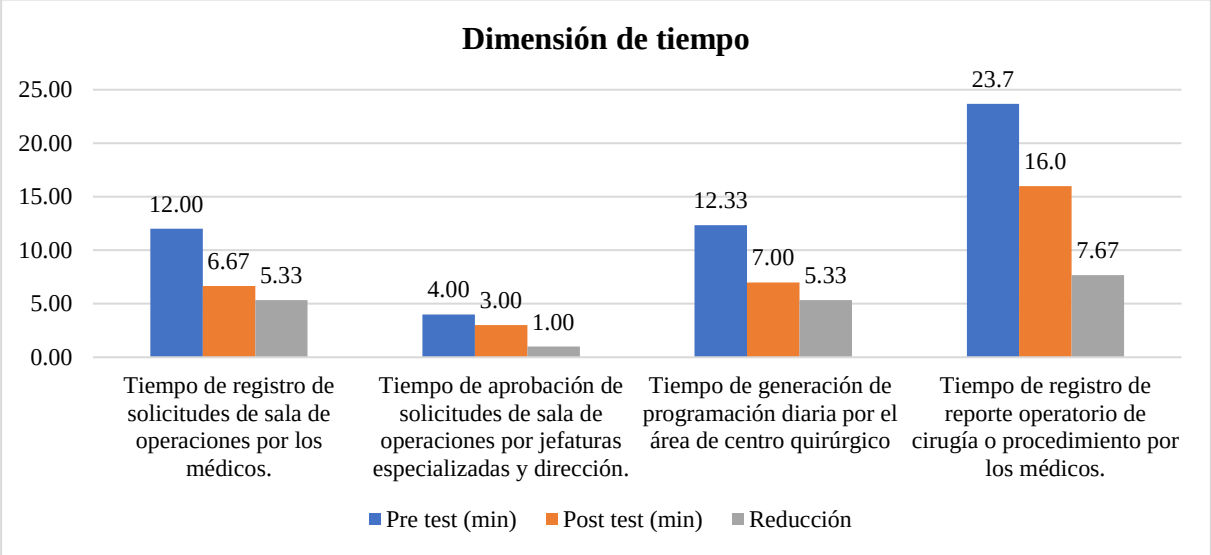


Fig. 79: Resultados gráficos de la dimensión tiempo, pre test – post test

Resultados dimensión de calidad

La ficha de cotejo evaluó el cumplimiento de pasos definidos en los procesos observados. Se estructuró según los indicadores definidos en la operacionalización de variables: fiabilidad de horarios, fiabilidad de datos personales y validación por doble revisión. En los tres casos, se pasó de un cumplimiento parcial a un cumplimiento total (100%) en el post test.

En el primer indicador se evaluó si los horarios quirúrgicos programados estaban correctamente asignados, sin generar conflictos ni sobreposición de cirugías. En el pre test, se evidenció un cumplimiento parcial, con rangos inferiores o iguales al 50% en algunos ciclos; sin embargo,

tras la implementación del módulo, el cumplimiento fue del 100% en todos los ciclos observados. Esto representa una mejora operativa significativa, atribuida a la automatización en la validación de horarios y a la visibilidad compartida entre áreas médicas, lo que permitió evitar errores comunes como la doble reserva o salas ocupadas, como se muestra en la Fig. 80.

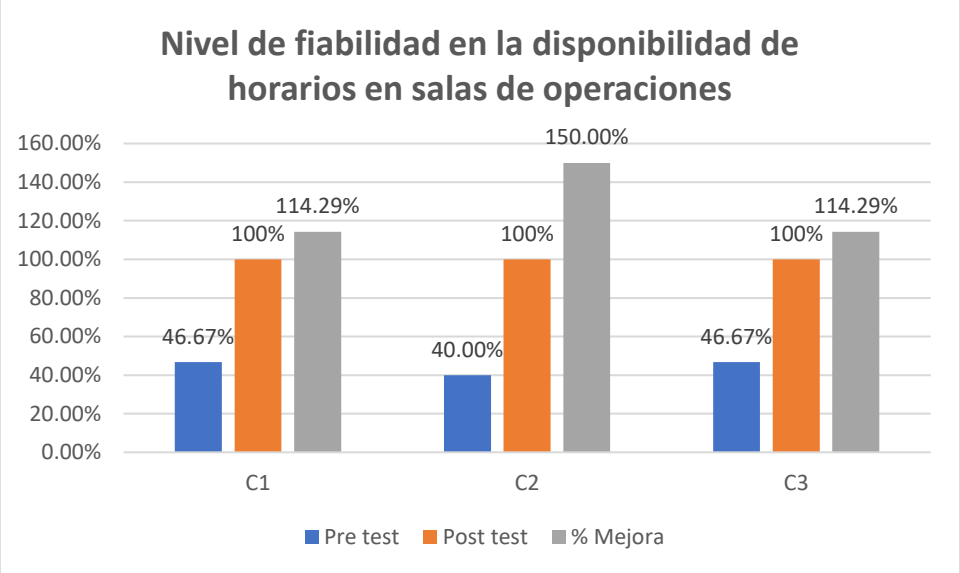


Fig. 80: Resultados de fiabilidad en la disponibilidad de horarios en salas de operaciones

El siguiente indicador valoró si los datos del paciente (nombre, historia clínica, edad) estaban completos y correctamente registrados en el sistema. El pre test evidenció brechas importantes, con cumplimiento parcial en la mayoría de ciclos, principalmente por registros incompletos o transcripciones erróneas desde documentos físicos. Con el uso del módulo quirúrgico, el cumplimiento llegó al 100%, logrando una mejora de hasta el 100% en algunos ciclos. Esto se debe a los campos obligatorios, validación automática de formatos, y trazabilidad del origen de la información en el sistema SisGalenPlus, como se observa en la Fig. 81.

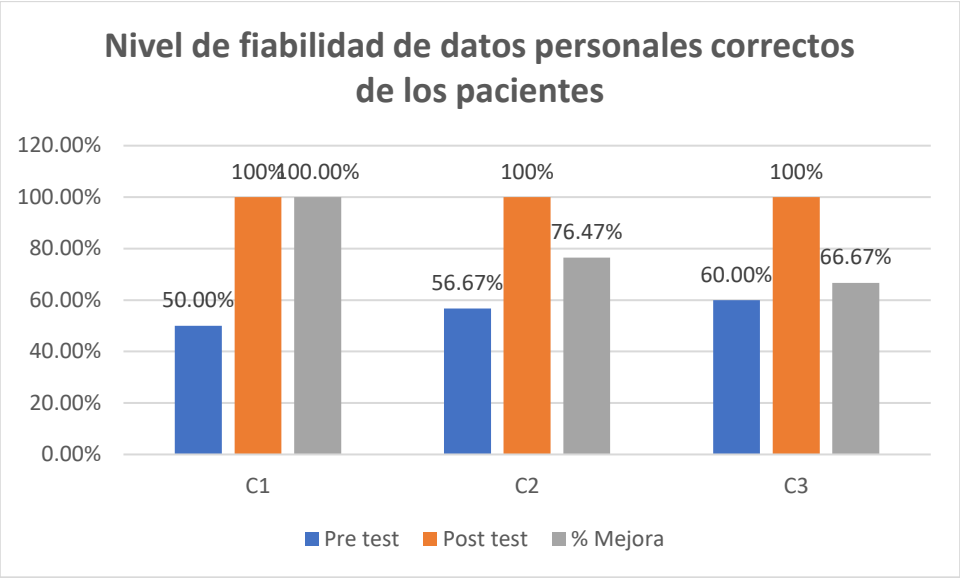


Fig. 81: Resultados de fiabilidad de datos personales correctos de los pacientes.

El tercer indicador evaluó si el proceso de programación incluía una validación doble por parte del sistema y revisión por la jefatura. En el pre test, el cumplimiento era parcial, con cifras entre el 36% y 50%. Tras la implementación del módulo, se logró un cumplimiento total del 100%, con mejoras porcentuales que superan el 170% en algunos ciclos. Esta mejora se debe a la incorporación de controles de validación digital por roles y la visibilidad del historial de autorizaciones dentro del sistema SisGalenPlus, lo cual fortalece la trazabilidad y responsabilidad en la programación quirúrgica, como se observa en la Fig. 82.

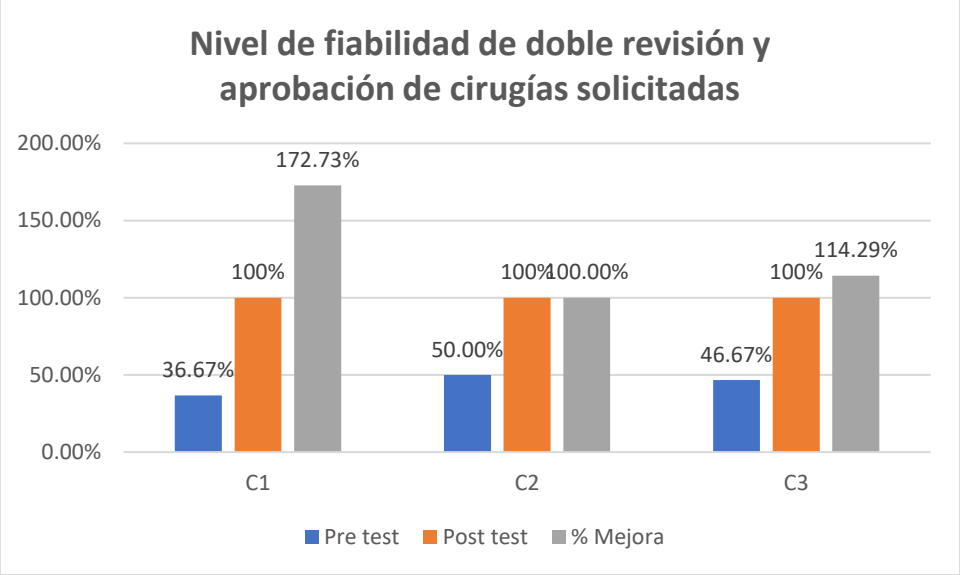


Fig. 82: Resultados fiabilidad de doble revisión y aprobación de cirugías solicitadas.

La Fig. 83 presenta una visión consolidada de los tres indicadores de la dimensión calidad evaluados a través de la ficha de cotejo: disponibilidad de horarios, fiabilidad de datos personales y doble revisión. Se observa un incremento significativo en todos ellos, alcanzando el 100% de cumplimiento en el post test, frente a un cumplimiento parcial registrado inicialmente. Este resultado confirma una mejora integral en la estandarización del proceso quirúrgico, reducción de omisiones y fortalecimiento del cumplimiento de protocolos institucionales.

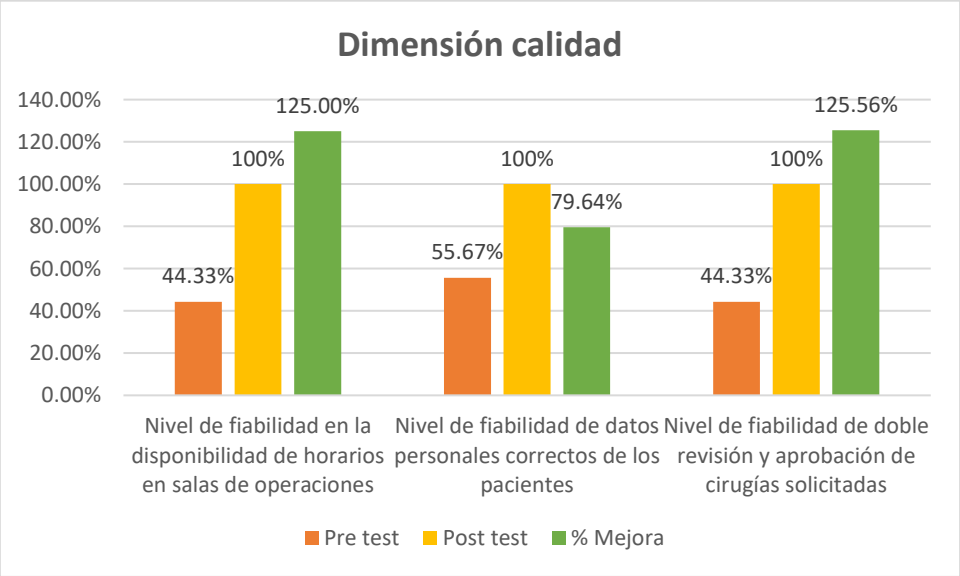


Fig. 83: Resultados Dimensión de Calidad

CAPÍTULO IV. ANÁLISIS Y DISCUSIÓN DE RESULTADOS

Los resultados evidencian que la implementación del módulo de centro quirúrgico generó mejoras en todos los procesos evaluados, tanto desde el punto de vista o percepción del personal como en los tiempos reales de ejecución.

Desde la percepción de expertos conocedores de software (cuestionario variable independiente) se registró una mejora global del 91.12%, destacando los subdimensiones de completitud funcional, corrección funcional y comprensión del sistema. Estos resultados están alineados con lo planteado por ISO/IEC 25010 en cuanto a la calidad del producto software, validando que el sistema responde adecuadamente a las necesidades funcionales del proceso quirúrgico.

Desde la experiencia operativa (variable dependiente): se describen los resultados atendiendo a los objetivos, es así que la presente investigación tuvo como objetivo general determinar el impacto de la implementación del módulo de Centro Quirúrgico en la programación de sala de operaciones en el Instituto Nacional de Salud del Niño San Borja, Lima. Para ello, se abordaron tres dimensiones clave asociadas a la variable dependiente: satisfacción del personal, tiempo de ejecución de actividades y calidad del proceso quirúrgico. Los resultados obtenidos tras la aplicación de instrumentos antes y después de la intervención evidencian mejoras en todos los aspectos evaluados.

En relación con el primer objetivo específico, conocer el impacto del módulo en la satisfacción del personal médico y administrativo, se observó un incremento en los niveles de satisfacción percibidos. En una escala de Likert del 1 al 5, el puntaje promedio general pasó de 2.29 (pretest) a 4.20 (postest), lo que representa una mejora del 83.4%. Este aumento resalta que los usuarios valoraron positivamente los cambios introducidos por el sistema, especialmente en cuanto a la claridad de los procesos, la reducción de errores y la facilidad de acceso a la información quirúrgica. La mayor mejora se observa en la programación y disponibilidad diaria de salas de operaciones, con un incremento del 103.94%, lo que indica una muy buena gestión de recursos quirúrgicos.

En cuanto al segundo objetivo específico, que consistió en analizar el impacto del módulo en el tiempo de ejecución de las actividades de programación quirúrgica, los hallazgos revelaron reducciones en todos los subprocesos evaluados. El tiempo promedio necesario para registrar solicitudes pasó de 12.00 a 6.67 minutos (reducción del 44.44%), mientras que el tiempo de

aprobación disminuyó de 4 a 3 minutos (reducción del 25.00%). La generación de la programación diaria mostró una disminución del 43.24% y el tiempo para registrar reportes operatorios se redujo en un 32.39%. En conjunto, el tiempo total del proceso disminuyó en un 37.18%, evidenciando una mejora operativa sustancial. Estos datos demuestran que la propuesta del módulo no solo eliminó tareas redundantes, sino que también permitió agilizar la toma de decisiones clínicas y administrativas. Esta evidencia se alinea con investigaciones de los antecedentes.

Respecto al tercer objetivo, centrado en determinar el impacto del módulo en la calidad del proceso quirúrgico, los indicadores evaluados mediante ficha de cotejo revelaron un cumplimiento pleno de los estándares esperados tras la implementación del sistema. Se pasó de un cumplimiento promedio del 48% a un 100%, representando una mejora del 108%. Destacan los aumentos en la disponibilidad de horarios en quirófanos (125%), la precisión en los datos personales de los pacientes (80%) y la garantía de doble revisión de las solicitudes quirúrgicas (125%). Esta mejora integral en la calidad del proceso es relevante, ya que fortalece la trazabilidad, la seguridad del paciente y la toma de decisiones clínicas fundamentadas. Asimismo, garantiza la reducción de errores administrativos, una problemática ampliamente documentada en instituciones con procesos manuales, tal como se reporta en [8].

Los hallazgos obtenidos en la presente investigación coinciden en gran medida con lo reportado en estudios previos tanto a nivel internacional como nacional. En primer lugar, los resultados referidos a la mejora en los tiempos del proceso de programación quirúrgica encuentran correspondencia directa con lo señalado por Villar [12], quien en su investigación sobre un sistema web para programación quirúrgica reportó reducciones significativas de tiempo en procesos similares: registro de pacientes, verificación y seguimiento. Villar identificó una eficiencia del 63.04% en el registro, mejora de 24.39 minutos en la verificación, y un 91.01% en el seguimiento de pacientes. En el presente estudio, se alcanzó una reducción promedio del 37.18% en los tiempos operativos, lo cual está cerca del rango de mejora, lo que valida la efectividad de los módulos tecnológicos en la automatización de tareas críticas en entornos hospitalarios.

Asimismo, los resultados sobre la satisfacción del personal tras la implementación del módulo también coinciden con lo hallado por Ahmed y Ali [10], quienes concluyeron que la consideración de preferencias y la participación del paciente en la elección de cirujano impactaba directamente en la satisfacción y en una menor tasa de reingresos. Aunque el enfoque

de estos autores fue sobre la satisfacción del paciente, ambos estudios coinciden en que la percepción de control sobre el proceso y la disponibilidad de información aumentan notablemente los niveles de satisfacción. En el caso del Instituto Nacional de Salud del Niño San Borja, la satisfacción del personal médico y administrativo mejoró en un 83.4% en promedio, lo cual puede atribuirse a la accesibilidad a la información, la disminución del trabajo manual y la mayor confiabilidad del proceso.

Por otra parte, los resultados que muestran una mejora sustancial en la calidad del proceso quirúrgico, con un cumplimiento del 100% en todos los indicadores tras la implementación del módulo, guardan estrecha relación con los hallazgos de Cantera et al. [11]. En su estudio, los autores implementaron un módulo web para el proceso quirúrgico y concluyeron que este tipo de herramienta digital facilita la trazabilidad, la generación de informes operatorios completos y la integración con otras áreas. De forma similar, el módulo implementado en esta investigación permitió mejorar la fiabilidad de datos personales, la disponibilidad de horarios y el control de revisiones, reflejando una mejora promedio del 108% en calidad, lo que sugiere que los módulos quirúrgicos bien diseñados no solo automatizan tareas, sino que también aseguran el cumplimiento de protocolos clínicos establecidos.

En cuanto a la problemática relacionada con la cancelación de cirugías, Desta et al. [2] encontró que un 31.6% de cirugías programadas fueron canceladas, de las cuales el 20.5% se debió a una inadecuada programación. Esta cifra es un fuerte indicador de los efectos negativos de procesos manuales y mal gestionados. Si bien la presente investigación no cuantificó directamente la tasa de cancelación, sí abordó las causas estructurales que suelen llevar a este tipo de eventos, logrando intervenirlas mediante el módulo quirúrgico. Por lo tanto, puede inferirse que la mejora en programación y calidad también contribuiría indirectamente a reducir cancelaciones, como lo sugiere la literatura.

Wang et al. [3], en Bélgica, introdujeron una clasificación de eficiencia de quirófanos diferenciando entre pacientes ambulatorios y hospitalizados, argumentando que los primeros presentan menores tasas de infección y cancelación, y mayor rentabilidad. Si bien el presente estudio no segmentó por tipo de paciente, los aportes de Wang respaldan la importancia de disponer de información oportuna para tomar decisiones operativas eficientes, algo que el módulo implementado en el Instituto Nacional de Salud del Niño permite mediante la digitalización de solicitudes y programación quirúrgica.

Finalmente, Arias [13] y Anchante [14], a nivel nacional, abordaron factores administrativos como causa de retrasos y suspensiones de cirugías. Arias identificó que los factores administrativos generaban un retardo excesivo en un 40.2% de los casos, mientras que Anchante encontró que el exceso de programación y la falta de materiales eran las principales causas de suspensión. Estos hallazgos son coherentes con la situación previa a la intervención en el presente estudio, donde la ausencia de un módulo digital generaba acumulación de tareas manuales, sobrecarga administrativa y errores en la disponibilidad de recursos. Tras la implementación, la reducción de tiempos y el aumento en la fiabilidad de los procesos sugiere que el módulo actuó directamente sobre estas debilidades, previniendo su recurrencia.

5 CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

A partir de los resultados obtenidos, es posible afirmar que se cumplieron satisfactoriamente todos los objetivos de esta investigación, y que la implementación del módulo de centro quirúrgico tuvo un impacto positivo en la programación de salas de operaciones en el Instituto Nacional de Salud del Niño San Borja. En relación con el primer objetivo específico, orientado a conocer el impacto del módulo en la satisfacción del personal médico y administrativo, los resultados muestran un cambio significativo en la percepción positiva del sistema. El puntaje promedio en la escala Likert se incrementó de 2.29 en la etapa pre test a 4.20 en la etapa post test, lo cual representa un aumento del 83.40%. Este resultado se desglosa en mejoras específicas del 81.70% en la satisfacción respecto al proceso de solicitud de sala y generación de reportes operatorios, 103.94% en la programación diaria de salas, y 68.00% en la verificación y aprobación de cirugías. Estas cifras reflejan una aceptación favorable del sistema por parte del personal involucrado, principalmente por su utilidad, facilidad de uso y precisión en la ejecución de tareas.

Respecto al segundo objetivo específico, centrado en analizar el tiempo de las actividades relacionadas con la programación quirúrgica, se evidenció una reducción sustancial en los tiempos de ejecución de todos los subprocesos observados. El tiempo promedio para registrar una solicitud quirúrgica disminuyó de 12.00 a 6.67 minutos (reducción del 44.44%), el tiempo de aprobación se redujo de 4.00 a 3.00 minutos (25.00%), el tiempo de generación de programación diaria bajó de 12.33 a 7.00 minutos (43.24%) y el tiempo de registro de reporte operatorio descendió de 23.67 a 16.00 minutos (32.39%). En promedio, la mejora general fue del 37.18%, lo que demuestra un impacto positivo del sistema en la eficiencia temporal de las actividades quirúrgicas.

Para el tercer objetivo específico, orientado a evaluar el impacto en la calidad de las actividades del proceso quirúrgico, los datos obtenidos mediante la ficha de cotejo mostraron mejoras evidentes en el cumplimiento de pasos críticos definidos en los protocolos. Se observó un incremento del 125% en el nivel de fiabilidad en la disponibilidad de horarios quirúrgicos, una mejora del 80% en la exactitud de los datos personales registrados, y un aumento del 125% en la validación por doble revisión de cirugías programadas. Estos resultados demuestran que el

módulo contribuye no solo a optimizar tiempos, sino también a fortalecer la calidad del proceso asistencial, reduciendo errores y aumentando la trazabilidad y fiabilidad de la información.

Como resultado de la presente investigación, se concluye que la implementación del módulo de Centro Quirúrgico en el sistema SisGalenPlus impacta de manera positiva en la mejora de la programación de salas de operaciones en el Instituto Nacional de Salud del Niño San Borja, Lima. Esta afirmación se sustenta en los hallazgos obtenidos mediante el análisis estadístico aplicado, particularmente con la prueba Z, donde se obtuvo un valor de $Z\alpha$ que superan el valor crítico de 1.96 y presentan niveles de significancia $p < 0.05$, lo que permite rechazar la hipótesis nula y aceptar la hipótesis alternativa, confirmando diferencias estadísticamente significativas entre los resultados del pre test y post test. De forma global, se observó una mejora del 37.18% en los tiempos de ejecución de los procesos quirúrgicos, un aumento del 83.40% en la percepción de satisfacción del personal y un incremento del 125% en el cumplimiento de los pasos críticos definidos, consolidando así el impacto favorable de la implementación tecnológica en la gestión de salas operatorias.

Adicionalmente, los hallazgos de este estudio tienen implicancias prácticas y teóricas relevantes. En el aspecto práctico, se demuestra que la implementación de un módulo quirúrgico en un sistema de información hospitalaria mejora significativamente la eficiencia, calidad y satisfacción en los procesos de programación de salas de operaciones. Este modelo puede ser replicado en otros hospitales públicos con necesidades similares. En el plano teórico, la investigación aporta evidencia sobre la utilidad del diseño preexperimental para evaluar intervenciones tecnológicas en salud, y refuerza el valor de las dimensiones de calidad del software como indicadores válidos para medir el impacto. Entre las principales limitaciones del estudio se identifica la dificultad para disponer del tiempo del personal de salud, lo que retrasó la recolección de datos y obligó a ajustar el cronograma previsto. Esta situación refleja la necesidad de diseñar estrategias más flexibles para la aplicación de instrumentos en contextos clínicos con alta carga operativa.

5.2 RECOMENDACIONES

A partir de los resultados mostrados en el punto anterior de esta investigación y reconociendo que existen áreas complementarias no abordadas directamente en el presente estudio, se proponen las siguientes recomendaciones para fortalecer la gestión del proceso quirúrgico en el contexto hospitalario:

Promover la integración progresiva del módulo quirúrgico con otros sistemas departamentales, como anestesiología, recuperación postoperatoria, farmacia y laboratorio, a fin de consolidar un flujo de información integral. Esta interoperabilidad permitiría evitar la duplicación de registros, mejorar la continuidad del proceso asistencial y facilitar una toma de decisiones más precisa y oportuna.

Implementar programas de capacitación continúa dirigidos al personal médico y administrativo, orientados no solo al uso funcional del sistema, sino también al fortalecimiento de competencias en calidad de datos, cumplimiento de protocolos clínico-administrativos y adopción de nuevas funcionalidades. Esto ayudaría a una adecuada apropiación del sistema y su sostenibilidad en el tiempo.

Establecer mecanismos de monitoreo periódico del uso del módulo quirúrgico, mediante la definición y seguimiento de indicadores de desempeño específicos, permitiendo identificar cuellos de botella, registrar retroalimentación de los usuarios y aplicar mejoras incrementales que optimicen la eficiencia operativa y la experiencia del usuario.

Explorar la viabilidad de replicar la experiencia de implementación del módulo quirúrgico en otras áreas estratégicas del hospital, como hospitalización, emergencia o unidades críticas, con el propósito de avanzar hacia una transformación digital integral, buscando estandarizar procesos, fortalecer la gestión clínica y elevar la calidad del servicio a nivel institucional.

Desarrollar e implementar protocolos de validación electrónica y uso de firma digital en los procesos quirúrgicos, a fin de reforzar la trazabilidad, la legalidad de los actos médicos y la seguridad de los datos clínicos, contribuyendo a modernizar la gestión documental quirúrgica y a cumplir con los estándares de interoperabilidad exigidos por los sistemas nacionales de salud.

Finalmente, se sugiere que, en futuras versiones del módulo de centro quirúrgico, se implemente más interfaces que mejoren el acceso a la información, además de desarrollar una interfaz adaptativa que permita personalizar la visualización y las funcionalidades del sistema en

función del rol del usuario. Esta mejora facilitaría la interacción del personal médico, de enfermería, administrativo y de dirección, mostrando únicamente las opciones pertinentes según su perfil, lo cual no solo optimizaría la usabilidad y reduciría la carga cognitiva, sino que también incrementaría la seguridad del sistema al restringir el acceso a funciones no autorizadas. Asimismo, esta adaptación contribuiría a una experiencia de usuario más eficiente, permitiendo que cada profesional enfoque su atención en las tareas propias de su competencia dentro del flujo quirúrgico.

REFERENCIAS BIBLIOGRÁFICAS

[1] Organizacion Mundial de la Salud, «La cirugía segura salva vidas,» Ginebra - Suiza, 2008.

[2] Melaku Desta, Addissu Manaye, Abiot Tefera, Atalay Worku, Alemitu Wale, Alemlanchi Mebrat y Negesso Gobena, «Incidence and causes of cancellations of elective operation on the intended day of surgery at a tertiary referral academic medical center in Ethiopia,» Debre Markos, Ethiopia, 2018.

[3] Lien Wang, Erik Demeulemeester, Nancy Vansteenkiste y Frank E. Rademakers, «Operating room planning and scheduling for outpatients and inpatients: A review and future research,» *Operations Research for Health Care*, vol. 31, p. 100323, 2021.

[4] Belén Navarro Carmona y Paz Pérez González, Artists, *Programación y planificación de quirófanos: Clasificación y análisis.* [Art]. Universidad de Sevilla, 2020.

[5] S. P. Badillo Díaz, C. P. Avellaneda Olarte, L. F. Avila Gonzalez, M. G. Cote Ante y H. A. Castillo Mosquera, «Propuesta para el uso de una herramienta tecnológica para gestionar, realizar seguimiento y controlar los sistemas de gestión de la sociedad de cirugía xyz,» Colombia, Bogotà, 2018.

[6] Milton A. Londoño, Cristiam A. Gil, Juan S. Mock kow y Juan P. Orejuela, «Programación multiobjetivo de quirófanos considerando el bienestar del cliente interno y externo,» *Información tecnológica*, vol. 33, nº 1, pp. 0718-0764, 2022.

[7] R.A.Abelda˜no y S.M.Coca, «Tasasycausasdesuspensióndecirugíasenunhospital públicoduranteela˜no2014,» Buenos Aires, Argentina, 2014.

[8] T. Y. Chora Chara , «Causas y costo de la suspensión de cirugías programadas en Centro Quirúrgico Del Hospital Regional Honorio Delgado De Arequipa – 2017,» Arequipa Peru, 2019.

[9] Astrid Díaz y Jorge Osada, «Tiempo de espera quirúrgica en un hospital de Chiclayo, Perú,» *Rev Peru Med Exp Salud Publica,* vol. 32, nº 1, p. 204, 2015.

[10] A. Abdulaziz y A. Haneen, «Modeling patient preference in an operating room scheduling problem,» *Operations Research for Health Care,* vol. 25, p. 100257, 2020.

[11] Migdeily Chiroles Cantera, Yunior Pacheco Correa, Raymari Reyes Chirino y Aldo Sisto Díaz, «Implementación de una aplicación web para el módulo servicio quirúrgico de la aplicación Behique,» *Revista de Ciencias Médicas de Pinar del Río,* vol. 21, nº 6, pp. 828-835, 2017.

[12] E. K. W. Villar Gonzales, «Implementación de un sistema para el proceso de registro de intervenciones quirúrgicas de la Clínica "Los Cocos E.I.R.L",» Piura Peru, 2020.

[13] Julia Guadalupe Arias Cuya, «Factores determinantes del tiempo de espera quirúrgico en un instituto especializado de salud de Lima, 2017,» Lima, 2018.

[14] F. M. Anchante Simborth, «Relación de los factores médicos y administrativos de la suspensión de cirugías con las características de los pacientes en un hospital de Lima, 2017 - 2018,» Lima Perú, 2018.

[15] E. A. Cáceres, «Análisis y Diseño de Sistemas de Información,» 2014.

[16] C. Brook, «What is a Health Information System?,» 2023.

[17] D. A. E. Morales, «Modularidad,» 2013. [En línea]. Available: https://sites.google.com/site/informatica1obh/iv-desarrollo-de-modularidad/unidad-iv-desarrollo-de-modularidad.

[18] A. a. d. cirugia, A. a. d. instrumentadoras, F. a. d. a. d. anestesia y S. a. d. infectología, «Centro Quirúrgico de Establecimientos con Internación,» Buenos Aires - Argentina, 2020.

[19] E. G. Maida y J. Pacienzia, «Metodologías de desarrollo de software,» Argentina, 2015.

[20] Ministerio de Salud del Peru, «MANUAL DE SISGALENPLUS,» 2014.

[21] Javier García de Jalón, José Ignacio Rodríguez y Alfonso Brazález, «Aprenda Visual Basic 6.0 como si estuviera en primero,» 1999.

[22] Geeks for Geeks, «SQL Tutorial,» 2023. [En línea]. Available: https://www.geeksforgeeks.org/sql-tutorial/?ref=lbp.

[23] M. I. S. S. Xavier Ferré Grau, «Desarrollo Orientado a Objetos con UML,» 2021-2022.

[24] J. J. I. &. B. G. Rumbaugh, «El Lenguaje Unificado de Modelado Manual de Referencia Segunda Edición,» 2006.

[25] H. Schuldt, «Multi-Tier Architecture.,» *Encyclopedia of Database Systems.,* pp. 1862-1865, 2009.

[26] R. S. Sandhu, E. J. Coyne, H. L. Feinstein y C. E. Youman, «Role-Based Access Control Models.,» *IEEE Computer*, vol. II, nº 29, pp. 38-47, 1996.

[27] R. Sandhu, D. F. Ferraiolo y D. R. Kuhn, «The NIST Model for Role-Based Access Control: Towards a Unified Standard.,» *ACM Workshop on RBAC 2000,* pp. 47-63, 2000.

[28] J. Britton, «What Is ISO 25010?,» Perforce, Londres, 2021.

[29] T. C. Hurtado, «Diagnóstico médico,» 2015.

[30] Ministerio de Salud del Peru MINSA, «Norma Técnica de Salud para la Gestión de la Historia Clínica,» Resolución Ministerial N°265-2018, Lima, 2018.

[31] M. d. S. d. Peru, «Resolucion Ministerial 280-2013/MINSA,» Lima Peru, 2013.

[32] Paul K. Mohabir y André V Coombs , «Cirugía,» The Manual's Editorial Staff, España, 2023.

[33] J. Hrdalo, J. Fiorentini, A. Schiaffi, B. Portillo Olivera, C. Santos, M. Serrano, G. Lardino, P. Baracco, N. Español y M. Oliva, «El Informe Quirúrgico,» 2020.

[34] P. Ruiz Lopez, J. Alcalde Escribano y J. I. Landa Garcia, Gestion Cìinica en Cirugìa, España: Asociacion Clinica de Cirujanos, 2006.

[35] E. G. Maida y J. Pacienzia, «Metodologías de desarrollo de software».

[36] F. Paredes-Olano, M. Malpica-Rodríguez, D. Pérez-Aguilar, J. Rodriguez-Alvarado, J. Pérez-Aguilar, A. Pérez-Aguilar y Llanos-Bardales, «Information system applied to the patient care process at a health facility in northern Peru(Conference Paper),» 2023.

[37] C. Institucional, «¿Qué es la investigación aplicada y cuáles son sus principales características?,» *IBERO TIJUANA*, 08 10 2020.

[38] A. Muguira, «¿Qué es la investigación descriptiva?,» 2005. [En línea]. Available: https://www.questionpro.com/blog/es/investigacion-descriptiva/.

[39] C. R. Galarza, «Diseños de investigación experimental,» *CienciAmérica*, vol. 10, nº 1, 2021.

ANEXOS

Anexo 1: Cuestionario de la variable independiente

Cuestionario de la Variable Independiente Modulo de Centro Quirúrgico

Encuestador: Ronald Enrique Chicoma Roca | **Fecha: ...../...../.....**

Objetivo: Evaluar las características del módulo de centro quirúrgico para programación de salas en base a la operacionalización de variables y la norma ISO/IEC 25010.

Instrucciones

Por favor, responda cada pregunta marcando el valor correspondiente de manera clara y precisa.

Valores de los ítems del cuestionario (en rango de preguntas cerradas):

1 = Totalmente en desacuerdo	**2** = En desacuerdo	**3** = Indeciso	**4** = De acuerdo	**5** = Totalmente de acuerdo

Datos Generales del Encuestado

- **Nombre del Experto**: .....
- **Cargo que desempeña**: .....

PREGUNTAS:

Dimensión	Pregunta	Valor Cuantitativo
Calidad		
Completitud funcional	¿El módulo cubre todas las necesidades funcionales del proceso de programación de salas quirúrgicas?	1 2 3 4 5
Corrección funcional	¿El módulo garantiza que los datos registrados sean correctos y sin errores?	1 2 3 4 5
Pertinencia funcional	¿El módulo está alineado con las tareas y procesos reales del centro quirúrgico?	1 2 3 4 5
Capacidad de entender fácilmente	¿Es fácil para los usuarios comprender las funciones del módulo desde su primera interacción?	1 2 3 4 5
Capacidad de operar fácilmente	¿El módulo permite realizar las operaciones requeridas sin necesidad de asistencia adicional?	1 2 3 4 5
Fiabilidad de los datos	¿El módulo garantiza que los datos personales y quirúrgicos estén protegidos y sean fiables?	1 2 3 4 5
Doble revisión de datos	¿El módulo permite realizar una doble verificación de las programaciones quirúrgicas antes de su ejecución?	1 2 3 4 5
Tiempo		
Tiempo de registro de solicitudes	¿El tiempo para registrar solicitudes en el módulo es adecuado para las necesidades del centro quirúrgico?	1 2 3 4 5
Tiempo de generación de reportes	¿El módulo permite generar reportes quirúrgicos de manera rápida y eficiente?	1 2 3 4 5
Tiempo de aprobación de solicitudes	¿El tiempo requerido para aprobar una solicitud en el módulo es apropiado?	1 2 3 4 5
Satisfacción		
Satisfacción de los médicos asistenciales	¿El módulo satisface las necesidades de los médicos en la programación y gestión de salas quirúrgicas?	1 2 3 4 5
Satisfacción del personal administrativo	¿El módulo facilita el trabajo del personal administrativo en los procesos de programación quirúrgica?	1 2 3 4 5
Satisfacción de la jefatura	¿El módulo cumple con las expectativas de la jefatura en la verificación y aprobación de solicitudes quirúrgicas?	1 2 3 4 5

Anexo 2: Cuestionario de la variable dependiente

Cuestionario de la Variable dependiente Programación de Sala de Operaciones

Encuestador: Ronald Enrique Chicoma Roca | **Fecha: ...../...../.....**

Objetivo: Evaluar la calidad, el tiempo y la satisfacción en la programación de salas de operaciones en el Instituto Nacional de Salud del Niño San Borja.

Instrucciones

Responda cada pregunta marcando el valor correspondiente de manera clara y precisa.

Valores de los ítems del cuestionario (en rango de preguntas cerradas):

1 = Muy Insatisfecho	**2** = Insatisfecho	**3** = Normal	**4** = Satisfecho	**5** = Muy Satisfecho

Datos Generales del Encuestado

- **Nombre**: .....
- **Cargo que desempeña**: .....

Preguntas con respecto a la programación de sala de operaciones

Indicador	Pregunta	Valor Cuantitativo
Nivel de fiabilidad en horarios de salas de operaciones	¿Qué tan satisfecho estás con la precisión de los horarios asignados en la programación de salas de operaciones?	1 2 3 4 5
Fiabilidad en datos personales de los pacientes	¿Qué tan satisfecho estás con la fiabilidad de los datos registrados sobre los pacientes en la programación?	1 2 3 4 5
Doble verificación y aprobación de cirugías programadas	¿Qué tan satisfecho estás con el proceso de doble verificación de las programaciones antes de su aprobación?	1 2 3 4 5
Tiempo de generación de programación diaria	¿Qué tan satisfecho estás con el tiempo que toma generar una programación diaria de salas?	1 2 3 4 5
Tiempo de registro de solicitudes	¿Qué tan satisfecho estás con el tiempo requerido para registrar una solicitud de programación de sala quirúrgica?	1 2 3 4 5
Tiempo de generación de reportes	¿Qué tan satisfecho estás con la rapidez en la generación de reportes quirúrgicos desde la programación de salas?	1 2 3 4 5
Satisfacción de médicos asistenciales	¿Qué tan satisfecho estás con el soporte del módulo en la solicitud y programación de salas quirúrgicas para los médicos?	1 2 3 4 5
Satisfacción del personal administrativo	¿Qué tan satisfecho estás con la facilidad del módulo para gestionar la programación diaria de salas quirúrgicas?	1 2 3 4 5
Satisfacción de la jefatura	¿Qué tan satisfecho estás con el soporte del módulo para la aprobación de cirugías programadas?	1 2 3 4 5

Anexo 3: Ficha de observación

Ficha de Observación sobre el Uso del Módulo de Centro Quirúrgico en la Programación de Salas				
Observador: Ronald Enrique Chicoma Roca				
Objetivo: Obtener información sobre el desempeño del módulo de centro quirúrgico en las actividades relacionadas con la programación de salas de operaciones, considerando el tiempo				
Instrucciones Cronometrar y registrar el tiempo en minutos de cada actividad realizada por el usuario en la programación de salas.				
Momento de la observación:	**Antes del uso del módulo ( ) Después del uso del módulo ( )**			
Fecha de Observación:	**FECHA: ……/……../…….**			
Duración de la observación:	**HORA INICIO: …… HORA FIN: ……**			
Datos Generales del trabajador • **Nombre:** …… • **Cargo:** ……				
Tiempo con respecto a las actividades de programación de salas				
Actividad	**Tiempo por Observación**			**Tiempo Promedio**
	O1	**O2**	**O3**	
Tiempo de registro de solicitudes de cirugía				
Tiempo de aprobación de solicitudes				
Tiempo de generación de reportes operatorios				
Tiempo de ajustes en programación diaria				

Anexo 4: Ficha de cotejo

Ficha de Cotejo para Evaluar la Fiabilidad de Datos en la Programación de Salas de Operaciones			
Responsable del Cotejo: Ronald Enrique Chicona Roca		**Fecha: ...../...../.....**	
Objetivo Evaluar la fiabilidad en la disponibilidad de horarios, precisión de datos personales de los pacientes y la revisión/aprobación de cirugías programadas en el módulo de programación de salas de operaciones.			
Instrucciones Para cada ítem, seleccione el nivel de cumplimiento según la siguiente escala: **0** = No cumple | **1**= Cumplimiento parcial (menos del 50%) | **2** = Cumplimiento aceptable (50% a 80%) | **3** = Cumplimiento total (más del 80%) Complete todas las secciones registrando observaciones que ayuden a contextualizar las respuestas.			
Datos Generales • **Nombre:** ..... • **Cargo:**			
Indicador	**Descripción.....**	**Cumple (Sí / No)**	**Observaciones**
Sección 1: Disponibilidad de Horarios en Salas de Operaciones			
Disponibilidad de horarios	¿Los horarios programados en el módulo están disponibles en el sistema sin conflictos con otras cirugías?	0 1 2 3	
Conflictos en la asignación de salas	¿Se detectaron conflictos de horarios (doble reserva, salas ocupadas)?	0 1 2 3	
Actualización oportuna	¿Los cambios en las programaciones se reflejan de manera inmediata en el sistema?	0 1 2 3	
Sección 1: Fiabilidad de Datos Personales de los Pacientes			
Exactitud de datos	¿Los datos personales del paciente (nombre, HC, edad, peso) son precisos y están completos en el sistema?	0 1 2 3	
Errores en el registro	¿Se detectaron errores o inconsistencias en los datos personales de los pacientes?	0 1 2 3	
Protección de datos	¿Se garantiza la confidencialidad de los datos personales en el sistema?	0 1 2 3	
Sección 3: Doble Revisión y Aprobación de Cirugías Programadas			
Proceso de doble revisión	¿El sistema permite realizar una doble revisión antes de aprobar una programación?	0 1 2 3	
Validación por parte de la jefatura	¿La jefatura tiene acceso y participa en la validación final de las programaciones quirúrgicas?	0 1 2 3	
Errores detectados en revisiones	¿Se identificaron errores durante las revisiones que llevaron a correcciones?	0 1 2 3	
Sección 4: Observaciones Generales			

Aspecto Evaluado	**Comentarios Adicionales**
Disponibilidad de horarios	
Fiabilidad de datos personales	
Doble revisión y aprobación	

Anexo 6: Confiabilidad del cuestionario para la variable independiente

	ITEMS													
ENCUESTADOS	1	2	3	4	5	6	7	8	9	10	11	12	13	SUMA
E1	4	4	2	3	2	4	4	3	2	4	3	4	4	43
E2	5	5	4	5	5	5	4	5	4	5	5	5	5	62
E3	4	5	5	3	5	4	3	4	5	3	4	3	5	53
E4	4	4	3	4	5	5	4	3	5	4	5	3	3	52
E5	5	5	4	5	5	4	5	3	3	5	4	3	5	56
E6	4	5	4	5	5	5	5	5	5	4	5	5	5	62
E7	3	3	2	2	2	2	3	4	3	3	4	3	2	36
E8	5	5	5	5	5	4	5	5	4	5	4	5	5	62
E9	4	2	2	3	4	4	2	2	4	2	4	2	4	39
E10	2	5	3	2	3	3	4	4	3	3	3	3	4	42
E11	3	5	4	3	3	3	5	5	5	4	5	3	5	53
E12	5	5	4	5	5	5	5	4	5	4	4	5	5	61
E13	5	5	5	4	5	4	5	5	5	5	5	5	5	63
E14	4	5	5	4	5	4	5	5	5	5	4	4	4	59
E15	5	5	4	5	5	4	5	4	5	5	5	4	5	61
E16	5	4	5	5	5	5	5	4	5	4	5	4	5	61
Promedio	4	4.35	3.8	3.9	4.35	4.2	4.5	4.3	4.5	4.4	4.7	4.3	4.941	
VARIANZA	0.777	0.750	1.152	1.184	1.215	0.684	0.840	0.809	0.938	0.809	0.465	0.902	0.746	
SUMATORIA DE VARIANZAS	11.270													
VARIANZA DE LA SUMA DE LOS ÍTEMS	79.309													

α:	Coeficiente de confiabilidad del cuestionario →	**0.93**
k:	Número de ítems del instrumento →	13
$\sum_{i=1}^{k} S_i^2$	:Sumatoria de las varianzas de los ítems. →	11.270
S_t^2:	Varianza total del instrumento. →	79.309

Anexo 7: Confiabilidad del cuestionario para la variable dependiente

	ITEMS									
ENCUESTADOS	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**SUMA**
E1	3	4	4	5	3	3	4	3	3	**32**
E2	3	4	3	5	4	3	4	4	4	**34**
E3	5	3	4	4	4	5	5	4	5	**39**
E4	3	5	4	3	3	5	3	4	3	**33**
E5	4	4	3	5	3	5	4	4	4	**36**
E6	5	5	5	4	4	4	5	5	5	**42**
E7	4	4	5	5	5	5	5	5	5	**43**
E8	5	5	5	4	5	5	5	5	5	**44**
E9	3	3	3	3	4	3	3	3	3	**28**
E10	4	4	4	4	5	3	3	4	3	**34**
E11	5	3	4	3	5	5	5	4	3	**37**
E12	3	4	3	4	5	3	5	5	4	**36**
E13	3	4	4	3	4	4	3	5	3	**33**
E14	3	4	5	5	3	4	5	4	5	**38**
E15	3	5	3	3	5	3	5	3	5	**35**
E16	5	5	4	5	5	5	5	5	5	**44**
E17	4	5	4	5	4	4	5	5	4	**40**
E18	5	5	5	4	4	4	4	3	3	**37**
E19	4	5	4	4	5	5	5	5	5	**42**
E20	5	4	5	5	5	5	4	5	4	**42**
E21	5	3	4	4	5	4	3	4	5	**37**
E22	4	3	4	3	3	4	3	3	5	**32**
E23	5	4	5	4	4	5	5	4	5	**41**
E24	5	5	5	5	5	5	5	4	4	**43**
E25	4	4	4	4	5	4	5	4	5	**39**
E26	3	3	3	3	3	3	3	5	4	**30**
E27	4	4	4	4	3	3	5	5	4	**36**
E28	5	5	4	4	5	5	5	4	4	**41**
E29	3	5	3	4	3	3	5	3	3	**32**
E30	5	5	4	5	5	5	5	5	5	**44**
E31	3	4	4	3	4	4	4	3	5	**34**
E32	4	3	4	4	4	3	3	4	4	**33**
E33	5	4	5	5	5	4	5	5	5	**43**
E34	5	3	3	5	4	4	3	4	5	**36**
E35	3	4	5	5	3	5	3	4	4	**36**
E36	5	5	4	3	5	4	4	4	3	**37**
E37	5	4	3	4	5	5	4	3	4	**37**
E38	5	5	4	5	5	5	5	5	5	**44**
PROM	4.05	4.10	4.00	4.13	4.26	4.21	4.33	4.26	4.33	
VARIANZA	**0.746**	**0.554**	**0.499**	**0.588**	**0.654**	**0.659**	**0.720**	**0.554**	**0.640**	
SUMATORIA DE VARIANZAS	**5.615**									
VARIANZA DE LA SUMA DE LOS ÍTEMS	**18.670**									

α:	Coeficiente de confiabilidad del cuestionario →		**0.79**
k:	Número de ítems del instrumento →		9
$\sum_{i=1}^{k} S_i^2$	Sumatoria de las varianzas de los ítems. →		5.615
S_t^2:	Varianza total del instrumento. →		18.6703601

Anexo 5: Ficha de validación de instrumentos

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: RONALD VARGAS ALVAREZ
1.2. Especialidad: INGENIERIA DE SISTEMAS
1.3. Cargo actual: COORDINADOR DE INFORMATICA
1.4. Grado académico: BACHILLER
1.5. Institución: INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE DEPENDIENTE
1.7. Lugar y fecha: LIMA 02/05/2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto

CIP: 215868

124

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: RONALD VARGAS ALVAREZ
1.2. Especialidad: INGENIERIA DE SISTEMAS
1.3. Cargo actual: COORDINADOR DE INFORMATICA
1.4. Grado académico: BACHILLER
1.5. Institución: INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE DEPENDIENTE
1.7. Lugar y fecha: LIMA 02/05/2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto

CIP: 215868

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: Ana María Valverde Bejar
1.2. Especialidad: Médico Cirujano
1.3. Cargo actual: Jefe de Equipo
1.4. Grado académico: Maestría
1.5. Institución: ESAN
1.6. Tipo de instrumento: Cuestionario de la Variable Dependiente
1.7. Lugar y fecha: San Borja, 15 de mayo 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100%

III. OBSERVACIONES Y/O RECOMENDACIONES

—

INSTITUTO NACIONAL DE SALUD DEL NIÑO - SAN BORJA
EQUIPO DE SEGUROS PÚBLICOS Y PRIVADOS
MC. ANA MARIA VALVERDE BEJAR
CMP N° 51512 / RNE
Jefe de Equipo de Seguros Públicos y Privados

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: RONALD URGO ALVAREZ
1.2. Especialidad: INGENIERO DE SISTEMAS
1.3. Cargo actual: COORDINADOR DE INFORMÁTICA
1.4. Grado académico: BACHILLER
1.5. Institución: INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: FICHA DE COTEJO
1.7. Lugar y fecha: LIMA 02/05/2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento		X				
7	Acorde al avance de la ciencia y tecnologia	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	45	4				

Coeficiente de valoración porcentual: c = 98%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto

CIP: 215968

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: Ana María Valverde Bejar
1.2. Especialidad: Médico Cirujano
1.3. Cargo actual: Jefe de Equipo
1.4. Grado académico: Maestría
1.5. Institución: ESAN.
1.6. Tipo de instrumento: Ficha de cotejo
1.7. Lugar y fecha: San Borja, 15 de mayo 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología		X				
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	45	4				

Coeficiente de valoración porcentual: c = 98%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

INSTITUTO NACIONAL DE SALUD DEL NIÑO - SAN BORJA
EQUIPO DE SEGUROS PÚBLICOS Y PRIVADOS
MC. ANA MARIA VALVERDE BEJAR
CMP N° 51513 - RNE 907
Jefe de Equipo de Seguros Públicos y Privados

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: RONALD VARGAS ALVAREZ
1.2. Especialidad: INGENIERIA DE SISTEMAS
1.3. Cargo actual: COORDINADOR DE INFORMATICA
1.4. Grado académico: BACHILLER
1.5. Institución: INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE INDEPENDIENTE
1.7. Lugar y fecha: LIMA 02/05/2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio		X				
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología		X				
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	40	8				

Coeficiente de valoración porcentual: c = 96%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto

CIP: 215868

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: RONNY VARGAS ALVAREZ
1.2. Especialidad: INGENIERIA DE SISTEMAS
1.3. Cargo actual: COORDINADOR DE INFORMATICA
1.4. Grado académico: BACHILLER
1.5. Institución: INSTITUTO NACIONAL DE SALUD DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: FICHA DE OBSERVACION
1.7. Lugar y fecha: LIMA 02/05/2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología		X				
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	45	4				

Coeficiente de valoración porcentual: c = 98%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto
CIP: 215868

124

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: Teodoro Zuasnábar Junes
1.2. Especialidad: Estadístico
1.3. Cargo actual: Jefe de Oficina
1.4. Grado académico: Licenciado
1.5. Institución: Hospital de Emergencias Pediátricas
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE DEPENDIENTE
1.7. Lugar y fecha: 14.05.25

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100 %

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

MINISTERIO DE SALUD
HOSPITAL DE EMERGENCIAS PEDIATRICAS
Lic. Est. TEODORO ZUASNABAR JUNES
COESPE N° 201
JEFE DE LA OFICINA DE ESTADISTICA E INFORMATICA

.....
Firma y sello del Experto

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: Teodoro Zuasnábar Junes
1.2. Especialidad: Estadístico
1.3. Cargo actual: Jefe de Oficina
1.4. Grado académico: Licenciado
1.5. Institución: Hospital de Emergencias Pediátricas
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE INDEPENDIENTE
1.7. Lugar y fecha: 14.05.25

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100 %

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

MINISTERIO DE SALUD
HOSPITAL DE EMERGENCIAS PEDIÁTRICAS
Lic. Est. TEODORO ZUASNABAR JUNES
COESPE N° 201
JEFE DE LA OFICINA DE ESTADÍSTICA E INFORMÁTICA

Firma y sello del Experto

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: Teodoro Zuasnábar Junes
1.2. Especialidad: Estadístico
1.3. Cargo actual: Jefe de Oficina
1.4. Grado académico: Licenciado
1.5. Institución: Hospital de Emergencias Pediátricas
1.6. Tipo de instrumento: FICHA DE OBSERVACION
1.7. Lugar y fecha: 14.05.25

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100 %

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

MINISTERIO DE SALUD
HOSPITAL DE EMERGENCIAS PEDIÁTRICAS
Lic. Est. TEODORO ZUASNABAR JUNES
COESPE N° 20
JEFE DE LA OFICINA DE ESTADÍSTICA E INFORMÁTICA

.....
Firma y sello del Experto

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: Teodoro Zuasnábar Junes
1.2. Especialidad: Estadístico
1.3. Cargo actual: Jefe de Oficina
1.4. Grado académico: Licenciado
1.5. Institución: Hospital de Emergencias Pediátricas
1.6. Tipo de instrumento: FICHA DE COTEJO
1.7. Lugar y fecha: 190525

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100 %

III. OBSERVACIONES Y/O RECOMENDACIONES

—

MINISTERIO DE SALUD
HOSPITAL DE EMERGENCIAS PEDIÁTRICAS
Lic. Est. TEODORO ZUASNABAR JUNES
COESPE N° 201
JEFE DE LA OFICINA DE ESTADÍSTICA E INFORMÁTICA

Firma y sello del Experto

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: ANNIA MELISSA LAURA RAMOS
1.2. Especialidad: INGENIERÍA DE SISTEMAS
1.3. Cargo actual: COORDINADORA DE PROYECTOS
1.4. Grado académico: INGENIERA
1.5. Institución: INSTITUTO NACIONAL DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE DEPENDIENTE
1.7. Lugar y fecha: LIMA, 02 DE MAYO DE 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100%

III. OBSERVACIONES Y/O RECOMENDACIONES

SE SUGIERE CAMBIAR LA PREGUNTA DE SATISFACCIÓN DE LA JEFATURA POR: "¿QUÉ TAN SATISFECHO ESTÁS CON LA FACILIDAD DEL USO DEL MODULO PARA LA APROBACION DE ARQUEOS PROGRAMADOS?"

Firma y sello del Experto
CIP: 180816

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: ANNIA MELISSA LAURA RAMOS
1.2. Especialidad: INGENIERIA DE SISTEMAS
1.3. Cargo actual: COORDINADORA DE PROYECTOS
1.4. Grado académico: INGENIERA
1.5. Institución: INSTITUTO NACIONAL DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE INDEPENDIENTE
1.7. Lugar y fecha: LIMA, 02 DE MAYO DE 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	x					
2	Formulado con lenguaje apropiado	x					
3	Adecuado para los sujetos en estudio	x					
4	Facilita la prueba de hipótesis	x					
5	Suficiencia para medir la variable	x					
6	Facilita la interpretación del instrumento		x				
7	Acorde al avance de la ciencia y tecnología		x				
8	Expresado en hechos perceptibles	x					
9	Tiene secuencia lógica	x					
10	Basado en aspectos teóricos	x					
	Total	40	8				

Coeficiente de valoración porcentual: c = 96%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

.....
Firma y sello del Experto
CIP: 180316

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: ANNIA MELISSA LAURA RAMOS
1.2. Especialidad: INGENIERIA DE SISTEMAS
1.3. Cargo actual: COORDINADORA DE PROYECTOS
1.4. Grado académico: INGENIERA
1.5. Institución: INSTITUTO NACIONAL DEL NIÑO SAN BORJA
1.6. Tipo de instrumento: FICHA DE COTEJO
1.7. Lugar y fecha: LIMA, 02 DE MAYO DE 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología		X				
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	45	4				

Coeficiente de valoración porcentual: c = 98%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto
CIP. 180316

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: VICENTE BARRIOS CARRANZA
1.2. Especialidad: ING SISTEMAS
1.3. Cargo actual: DIRECTOR EJECUTIVO TI
1.4. Grado académico: .....
1.5. Institución: INS NIÑO SAN BORJA
1.6. Tipo de instrumento: CUESTIONARIO DE LA VARIABLE DEPENDIENTE
1.7. Lugar y fecha: LIMA, 02 DE MAYO DE 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología	X					
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	50					

Coeficiente de valoración porcentual: c = 100%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto
DNI: 09780745
CIP: 132053

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: VICENTE BARRIOS CORONADO
1.2. Especialidad: ING. SISTEMAS
1.3. Cargo actual: DIRECTOR EJECUTIVO TI
1.4. Grado académico:
1.5. Institución: INS NIÑO SAN BORJA
1.6. Tipo de instrumento: FICHA DE COTEJO
1.7. Lugar y fecha: LIMA, 02 DE MAYO DE 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio	X					
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología		X				
8	Expresado en hechos perceptibles	X					
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	45	4				

Coeficiente de valoración porcentual: c = 98%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto
DNI: 09789025
CIP: 132053

FICHA PARA VALIDACIÓN DEL INSTRUMENTO

I. REFERENCIA

1.1. Experto: VICENTE BARRIOS CARRANZA
1.2. Especialidad: ING. SISTEMAS
1.3. Cargo actual: DIRECTOR EJECUTIVO - TI
1.4. Grado académico: INGENIERO
1.5. Institución: INSTITUTO NACIONAL DE SALUD NIÑO SAN BORJA
1.6. Tipo de instrumento: FICHA DE OBSERVACION
1.7. Lugar y fecha: LIMA, 02 DE MAYO DE 2025

II. TABLA DE VALORACIÓN POR EVIDENCIAS

N°	EVIDENCIAS	VALORACION					
		5	4	3	2	1	0
1	Pertinencia de indicadores	X					
2	Formulado con lenguaje apropiado	X					
3	Adecuado para los sujetos en estudio		X				
4	Facilita la prueba de hipótesis	X					
5	Suficiencia para medir la variable	X					
6	Facilita la interpretación del instrumento	X					
7	Acorde al avance de la ciencia y tecnología		X				
8	Expresado en hechos perceptibles		X				
9	Tiene secuencia lógica	X					
10	Basado en aspectos teóricos	X					
	Total	35	12				

Coeficiente de valoración porcentual: c = 94%

III. OBSERVACIONES Y/O RECOMENDACIONES

.....
.....
.....
.....
.....

Firma y sello del Experto
DNI: 09787025
CIP 132053

Anexo 8: Operacionalización de variables

Tabla LXIX: Operacionalización de variables

OPERACIONALIZACIÓN DE VARIABLES				
VARIABLE	**DESCRIPCIÓN**	**DIMENSIÓN**	**INDICADOR**	**TÉCNICA / INSTRUMENTOS**
Variable Independiente: Módulo de software de centro quirúrgico	Subsistema de un sistema de información mucho más grande llamado: sistema de información en salud (HIMS), el cual se encarga de la gestión asistencial de un hospital o centro médico, integrando a los procesos.	Calidad de sw	▪ Completitud funcional. ▪ Corrección funcional. ▪ Pertinencia funcional. ▪ Capacidad de entender fácilmente. ▪ Capacidad de aprender fácilmente. ▪ Capacidad de operar fácilmente.	Encuesta / Cuestionario
Variable Dependiente: Programación de salas de operaciones en el centro quirúrgico	La programación de las salas de operaciones es un proceso clave en la gestión del centro quirúrgico, ya que garantiza que los procedimientos se realicen de manera eficiente y dentro de los plazos establecidos.	Tiempo	▪ Tiempo de registro de solicitudes de sala de operaciones por los médicos. ▪ Tiempo de aprobación de solicitudes de sala de operaciones por jefaturas especializadas y dirección. ▪ Tiempo de generación de programación diaria por el área de centro quirúrgico ▪ Tiempo de registro de reporte operatorio de cirugía o procedimiento por los médicos.	Observación / Ficha de observación
		Satisfacción	▪ Satisfacción al proceso en la solicitud de sala de operaciones y generación del reporte operatorio. ▪ Satisfacción en el proceso de la programación y disponibilidad diaria de sala de operaciones ▪ Satisfacción en el proceso de verificación y aprobación de cirugías y procedimientos programados.	Encuesta / Cuestionario
		Calidad	▪ Nivel de fiabilidad en la disponibilidad de horarios en salas de operaciones. ▪ Nivel de fiabilidad de datos personales correctos de los pacientes. ▪ Nivel de fiabilidad de doble revisión y aprobación de cirugías solicitadas.	Análisis documental / Ficha de cotejo

Anexo 9: Matriz de consistencia

Tabla LXX: Matriz de consistencia

<table>
<tr><th colspan="5">“Uso de un sistema de monitoreo de información y su efecto en la gestión de tecnologías de información de una empresa minera del Perú”</th></tr>
<tr><th>PROBLEMA</th><th>OBJETIVOS</th><th>HIPÓTESIS</th><th>VARIABLES</th><th>INDICADORES</th></tr>
<tr><td rowspan="2">¿Cuál es el impacto de la implementación del módulo de Centro Quirúrgico en la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima?</td><td>Objetivo general
Establecer el impacto de la implementación del módulo de centro quirúrgico en la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima.</td><td rowspan="2">Hipótesis general
La implementación del módulo de Centro Quirúrgico impacta en la mejora de la programación de sala de operaciones en Instituto Nacional de Salud Del Niño San Borja, Lima</td><td>Variable Independiente:
Módulo de software de centro quirúrgico</td><td>▪ Completitud funcional.
▪ Corrección funcional.
▪ Pertinencia funcional.
▪ Capacidad de entender fácilmente.
▪ Capacidad de aprender fácilmente.
▪ Capacidad de operar fácilmente.</td></tr>
<tr><td>Objetivos específicos
▪ Estudiar el impacto en el tiempo de las actividades que realizan para cumplir con el proceso de programación de salas de operaciones.
▪ Conocer el nivel de satisfacción del personal médico y administrativo del centro quirúrgico en el proceso de la programación de salas de operación.
▪ Determinar el impacto del uso del módulo de centro quirúrgico en la calidad de las actividades del proceso de centro quirúrgico para la programación de sala de operaciones.</td><td>Variable Dependiente:
Programación de salas de operaciones en el centro quirúrgico</td><td>▪ Tiempo de registro de solicitudes de sala de operaciones por los médicos.
▪ Tiempo de aprobación de solicitudes de sala de operaciones por jefaturas especializadas y dirección.
▪ Tiempo de generación de programación diaria por el área de centro quirúrgico
▪ Tiempo de registro de reporte operatorio de cirugía o procedimiento por los médicos.
▪ Satisfacción al proceso en la solicitud de sala de operaciones y generación del reporte operatorio.
▪ Satisfacción en el proceso de la programación y disponibilidad diaria de sala de operaciones
▪ Satisfacción en el proceso de verificación y aprobación de cirugías y procedimientos programados.
▪ Nivel de fiabilidad en la disponibilidad de horarios en salas de operaciones.
▪ Nivel de fiabilidad de datos personales correctos de los pacientes.
▪ Nivel de fiabilidad de doble revisión y aprobación de cirugías solicitadas.</td></tr>
</table>