

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

**“RESOLUCIÓN DEL PROBLEMA DE ASIGNACIÓN
DE TURNOS DE ENFERMERAS Y PARAMÉDICOS
DEL HOSPITAL CARLOS VAN BUREN”**

VIVIANA ALVARADO AGUILERA

INFORME FINAL DE PROYECTO
PARA OPTAR AL TÍTULO PROFESIONAL DE
INGENIERO DE EJECUCIÓN EN INFORMÁTICA

OCTUBRE, 2016

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

**“RESOLUCIÓN DEL PROBLEMA DE ASIGNACIÓN
DE TURNOS DE ENFERMERAS Y PARAMÉDICOS
DEL HOSPITAL CARLOS VAN BUREN”**

VIVIANA ALVARADO AGUILERA

Profesor Guía: **Ricardo Soto De Giorgis**

Carrera: **Ingeniería Ejecución Informática**

OCTUBRE, 2016

*A Dios por guiarme y darme fuerzas.
A mi familia y pololo, por su apoyo incondicional,
su ayuda en los momentos difíciles, comprensión
y amor. Gracias por creer en mí.
“Pon en manos del Señor todas tus obras,
Y tus proyectos se cumplirán” **Proverbios 16:3***

Índice

Resumen	iii
Abstract.....	iii
Lista de tablas	iv
Lista de Figuras	v
1 Introducción	1
2 Objetivos	2
2.1 Objetivo general.....	2
2.2 Objetivos específicos	2
3 Estado del Arte.....	3
4 Descripción de la Metodología Usada	5
5 Problema de Asignación de Enfermeras y Paramédicos.....	6
5.1 Descripción del problema	6
6 Caso real de estudio.....	8
6.1 Descripción del caso real de estudio.....	8
6.1.1 Hospital Carlos Van Buren.....	8
6.2 Modelado del Caso de Estudio	10
6.3 Turnos	11
6.3.1 Cuarto turno.....	11
6.4 Formulación del Modelo Matemático.....	11
6.4.1 Variables:.....	11
6.4.2 Constantes:.....	11
6.4.3 Restricciones:.....	12
7 Implementación.....	13
7.1 JaCop Solver	13
7.2 Modelo del Prototipo	13
7.3 Prototipo.....	14
7.3.1 Implementación de Código.....	14
8 Interfaz Gráfica	16
8.1 Acceder al sitio principal	16
8.2 Registrar usuario	17
8.3 Usuario incorrecto.....	18
8.4 Registro trabajador.....	18

8.5	Asignación	19
8.6	Modo de guardar información	20
9	Resultados	21
9.1	Notas	21
9.2	¿Es recomendable?.....	21
9.3	Grafico Recomendación del software	22
10	Conclusiones.....	23
	Referencias Bibliográficas.....	24
	Anexos	26
	A Encuesta realizada a enfermeros del Hospital.....	26

Resumen

La programación con restricciones es una tecnología de software que provee un desarrollo rápido y un mantenimiento económico. Los problemas que se pueden resolver abarcan muchas áreas de la vida real. Un problema que ha sido resuelto por esta técnica es el *Nurse Rostering Problem* (NRP), este es conocido por su dificultad y consiste en asignar turnos de los funcionarios (enfermeras y paramédicos) en un centro de salud. En este proyecto se modelará un caso real, basado en el Hospital Carlos Van Buren de Valparaíso.

Abstract

Constraint Programming is a software technology that provides us a faster and low cost development. We can use these methods to solve many real life problems. One of the problems that has been solved with this technique is the “Nurse Rostering Problem (NRP)”. This problem is known for his difficulty and it consists in assigning shifts for paramedics and nurses in a given health center. In this project we will model a real life case, based on the Hospital Carlos Van Buren of Valparaíso

Lista de tablas

Tabla 6.1 Cantidad de profesionales por área, y por turno	10
Tabla 6.2 Ejemplo de asignación de turnos al mes.....	10
Tabla 9.1 Resultado notas.....	21
Tabla 9.2 Recomendación de Software	21
Tabla A.1 Preguntas encuesta.....	26
Tabla A.2 Puntaje de cada pregunta	27

Lista de Figuras

Figura 7.1 Extracto de código ‘restricciones’	14
Figura 7.2 Extracto de código ‘restricciones’	15
Figura 7.3 Extracto de código ‘enfermeras’	15
Figura 8.1 Captura de pantalla ‘Acceder sitio principal’	16
Figura 8.2 Captura de pantalla ‘Registrar usuario’	17
Figura 8.3 Captura de pantalla ‘Usuario incorrecto’	18
Figura 8.4 Captura de pantalla ‘Registro trabajador’	18
Figura 8.5 Captura de pantalla ‘Asignación’	19
Figura 8.6 Captura de pantalla ‘XML usado’	20
Figura 9.1 Grafico Recomendación del software	22

1 Introducción

A lo largo del país existen diversos hospitales, uno de los 10 hospitales más grandes es el Carlos Van Buren, ya que pertenece a la Red integrada del Servicio de Salud de Valparaíso, San Antonio y es referente para la IV y V región en oncología, para la III región en neurocirugía e imagenología compleja, la población que abarca es de 481.916 personas en la Cuidad de Valparaíso. Este centro entrega un servicio de ayuda a la comunidad a través de un gran equipo de funcionarios entre los cuales se encuentran, enfermeras, paramédicos, auxiliares de servicio, entre otros.

Para este hospital la tarea de asignar turnos no es fácil, ya que tiene una relación directa con el número de camas y funcionarios (a mayor número de camas, mayor número de funcionarios), por lo tanto se decidió tomar este problema y realizar una resolución. Para los investigadores este problema es reconocido y estos definen como una solución la programación con restricciones.

El objetivo de este proyecto es modelar y resolver dicho problema mediante la programación con restricciones, esta se define como un estudio de modelos computacionales y sistemas basados en restricciones, es un paradigma de programación que permite resolver problemas combinatorios de una forma eficiente. Con el pasar de los años, se han acrecentado los campos de aplicación, y hoy en día son diversos, como en campos basados en escenarios reales.

Muchos de los problemas que se generan en las aplicaciones antes mencionadas son modelados como problemas de satisfacción de restricciones (CSP), estos se representan mediante variables, dominios y restricciones.

The nurse rostering problem (NRP), o también llamado el problema de asignación de turnos de enfermeras, como una breve descripción, este es un sistema que puede asignar turnos a un grupo determinado de profesionales de la salud. El problema es muy común en los hospitales y centros de salud y es conocido por su complejidad, ya que se debe tener en cuenta distintos requerimientos y un conjunto de restricciones, entre ellas blandas y duras. Para la resolución se ocuparon restricciones propias del hospital y de los servicios. También se tomaron en cuenta los turnos y sus estructuras, que no son las mismas de todos los centros públicos, que en definitiva serán modeladas como restricciones.

Este hospital cuenta con una gran cantidad de enfermeras y paramédicos, en cada servicio el enfermero jefe debe realizar la asignación para cada profesional. Hoy en día este listado es hecho en forma manual, en consecuencia resulta costoso, tanto en términos de tiempo, como en esfuerzo y en general se es más propenso a cometer errores.

En este documento se definieron los objetivos, tanto generales como específicos del proyecto, continuando con la descripción del problema, en la sección 4 se explicó detalladamente la formulación del modelo matemático y por último se presentará un prototipo del problema, el cual se resolvió con la librería de código abierto dedicada al modelado y resolución de problemas con restricciones, llamada JaCop (*Java Constraint Programming*).

2 Objetivos

En esta sección, se describe el objetivo general, que entrega a grandes rasgos la meta que se pretendió lograr y/o cumplir en el desarrollo de este proyecto, además se incluyen los objetivos específicos, estos definen más en detalle lo que se quiere lograr.

2.1 Objetivo general

Modelar un sistema que pueda facilitar la asignación de turnos en el hospital Carlos Van Buren utilizando programación con restricciones.

2.2 Objetivos específicos

- Resolver el problema de asignación de turnos utilizando el *solver* JaCop.
- Analizar las diferentes necesidades reales que poseen estos profesionales.
- Realizar una interfaz gráfica acorde al nivel del usuario.

3 Estado del Arte

El Nurse Rostering Problem, de ahora en adelante NRP, es un problema constante en el día a día de los hospitales, ya que es necesario producir repetidamente listas de asignación de turnos ya sea para enfermeras y/o paramédicos, una buena asignación tiene un alto impacto en la calidad de cuidados frente a pacientes y/o enfermos, en la contratación del personal y otras funciones de enfermería. El NRP es particularmente difícil debido a las diferentes necesidades de cada profesional es por esto que ha sido objeto de muchos estudios, investigadores de todo el mundo se han dedicado a este problema aportando nuevos modelos y mejorando los actuales.

Este problema ha sido tema de investigación desde hace décadas [18]. El término *The Nurse Rostering*” ha sido utilizado durante años para cubrir varios tipos de problemas de programación de personal, en una visión general [23] distingue 3 áreas principales en la decisión de realización de la investigación: la dotación de personal, programación y reasignación del personal de enfermería, y cinco criterios para la programación: cobertura, calidad, estabilidad, flexibilidad y precio. Warner estaba convencido de que la combinación de las tres etapas conducía a una mejor política de planificación.

Desde la década de 1970 muchos documentos han sido publicados, en varios aspectos de NRP. La mayoría de los enfoques de programación matemáticos hacen uso de una función objetivo que se optimiza sujeto a ciertas restricciones, uno de los primeros trabajos [23][25]. Como los enfermeros jefes de cada hospital desean generar una lista de asignación rápida y de alta calidad que satisfaga todas las restricciones (duras y un alto número de blandas), es que se han desarrollado sistemas de programación sobre la base de modelos heurísticos [26], se pensaba que los métodos heurísticos eran más apropiados y más flexibles que los métodos matemáticos [21].

Un caso real es el enfoque de la inteligencia artificial basada en restricciones mediante la descripción de una implementación prototipo desarrollado con el lenguaje Charme (un lenguaje de programación basada en restricciones) llamado Horoplan, este utiliza algunas heurísticas, y el objetivo es generar la programación del personal de enfermería. El resultado de los 100 horarios (asignaciones) generadas, en 88 casos Horoplan dio una solución completa, en los demás casos no se encontró una solución global en el límite de 5 minutos. La construcción de una programación completa de enfermeras con Horoplan incluyendo la dotación de personal, los deseos y las modificaciones manuales requieren aproximadamente media hora. Por lo tanto este prototipo ahorra un promedio de tres horas y media por cada mes y el sistema francés se ahorraría 105.000 horas anuales de trabajo. [21]

Otro ejemplo de un caso desarrollado [23] el autor implementó un algoritmo para la dirección de hospitales de Hong Kong (llamado SRS el cual comenzó a principios de 1997), utilizando la programación con restricciones, y utilizando la clase C++ y bibliotecas de Rogue Wave, el sistema informático desarrollado mejora la prestación de servicios de atención médica, aumenta la productividad. Para este proyecto se estudiaron los hospitales más grandes de Hong Kong, después de medio año de pruebas y refinamiento, la versión fue oficialmente desplegada a mediados de 1998. Así es como a principios de 1999 SRS fue utilizado en los siete hospitales más grandes de Hong Kong.

Al ser expuestos estos dos ejemplos se puede apreciar que se ha podido resolver NRP con éxito, en distintos lenguajes de programación junto a distintas bibliotecas.

4 Descripción de la Metodología Usada

La programación con restricciones es un paradigma de programación que permite resolver problemas combinatorios de una forma eficiente, se basa principalmente en buscar un estado en el cual una gran cantidad de restricciones sean satisfechas simultáneamente.

En este capítulo se describirá el CSP (problemas de satisfacción de restricciones). Los conceptos clave en esta metodología corresponden a los aspectos de:

- La modelización del problema, que permite representar un problema mediante un conjunto finito de variables, un dominio de valores finito para cada variable y un conjunto de restricciones que acotan las combinaciones válidas de valores que las variables pueden tomar.
- Se realiza la búsqueda de soluciones, puesto que de antemano no se sabe si existe o no una solución., apoyadas generalmente por criterios heurísticos. El objetivo consiste en encontrar valores para cada una de las variables, de modo que se satisfagan todas las restricciones

El primer paso en la resolución de un CSP es su modelización, es decir, su representación en términos de variables, dominios y restricciones. Al igual que ocurre con el lenguaje natural, la modelización de un problema se puede realizar de muchas maneras diferentes. Respecto a la modelización de un problema CSP hay dos aspectos básicos:

- La potencia expresiva de las restricciones, es decir, la capacidad de modelar las restricciones realmente existentes en el problema real.
- La eficiencia de la representación, ya que dependiendo de la modelización CSP, el problema se resolverá con más o menos eficiencia.

5 Problema de Asignación de Enfermeras y Paramédicos

5.1 Descripción del problema

En todo el mundo se realiza una prestación de atención de salud continuada durante 24 horas, los 365 días del año, por esto es necesario que exista un sistema de rotación de enfermeras y paramédicos en el que un mismo paciente pueda ser atendido por varios profesionales de la salud en el tiempo que esté hospitalizado, esta es la razón principal de porque este problema es tan conocido.

Cada hospital necesita generar reiteradamente una lista apropiada del personal de la salud y estas son generadas manualmente por el enfermero (jefe del servicio), sin embargo, a lo largo de los años esto se ha transformado en un problema, ya que es una tarea difícil para el que lo ejecuta y se utiliza una gran cantidad de tiempo desarrollando estas listas, especialmente cuando hay mucho requerimiento de personal, como por ejemplo en la épocas de invierno o en las épocas en que muchos profesionales salen de vacaciones, también en algunos hospitales dan la posibilidad de que algunos enfermeros y/o paramédicos escojan sus turnos.

Debido a toda la dificultad y complejidad de este problema es que el *Nurse Rostering Problem* (NRP) ha llamado la atención de los investigadores.

El NRP consiste en producir una lista de asignaciones de turnos (mensual) para el personal de enfermería en un centro de salud, sujeto a una variedad de restricciones que pueden ser específicas de cada hospital, como también a nivel general. Dentro de estas restricciones encontramos:

- Restricciones duras: Son aquellas que deben satisfacerse, por lo tanto son obligatorias y deben cumplirse siempre, por ejemplo las capacidades o disponibilidades de los recursos.
 - Un ejemplo de estas, aparte de seguir la forma en que funcionan los turnos, es la cantidad de horas que pueda hacer un profesional, debido a que son turnos con una gran cantidad de horas los profesionales terminan agotados.
- Restricciones Blandas: Denotan la preferencia del usuario, son políticas flexibles. Puede que estas restricciones debiesen ser cumplidas pero no se exige que se cumplan siempre.
 - Un ejemplo de estas, son las preferencias de los enfermeros, al querer un turno en específico (todos los 1 de cada mes tener día libre).

El objetivo siempre es encontrar una solución que satisfaga las restricciones duras y que optimice las restricciones blandas. Cuando se encuentran soluciones que sólo satisfagan las restricciones duras, se podrá obtener una lista de asignación de turnos correctos, sin embargo es probable que se obtenga una solución difícil de implementar en la realidad, por lo tanto igual se debe tomar en cuenta las restricciones blandas.

Aunque de forma natural, incluso disponiendo de mucho tiempo, una persona sólo suele encontrar soluciones negativas o ninguna en absoluto

En la siguiente sección se expondrá el caso de estudio real realizado en el Hospital Carlos Van Buren, mostrando el modelo preliminar y su modelo matemático.

6 Caso real de estudio

En esta sección, se muestra la información recopilada en el caso de estudio, también, el modelo propuesto que representa a modo general la situación que ocurre en el Hospital Carlos Van Buren.

6.1 Descripción del caso real de estudio

Se expone una breve descripción de la historia del centro de salud y se explica la forma en que asignan los turnos de los profesionales.

6.1.1 Hospital Carlos Van Buren

Este hospital pertenece al servicio de salud Valparaíso- San Antonio y su dirección es Calle San Ignacio n° 725 en Valparaíso.

El 18 de noviembre se fundó el hospital el cual se llamó “San Juan de Dios”. El hospital recibió el nombre debido a que por muchos años y en sus inicios, estuvo a cargo de la congregación de hermanos hospitalarios de San Juan de Dios. En 1907 Carlos Van Buren, asume la sub- administración del Hospital San Juan de Dios, que estaba parcialmente destruido por el terremoto que afectó la ciudad en 1906. En 1912 Van Buren asume oficialmente la administración del recinto y con ello el hospital tomaría un cambio total, la renovación de sus nuevas dependencias, dejando al nuevo hospital a la altura de los requerimientos que la ciudad necesitaba. Luego de un mes de la muerte de Van Buren, una junta de beneficencia pidió al ministro de bienestar social, cambiar el nombre del entonces hospital “San Juan de Dios” al de “Carlos Van Buren”, realizándose por decreto n° 974 del 24 de mayo de 1929. Esto conforme a un sentimiento de enorme gratitud, desvelo, sacrificio y ayuda económica durante largos años de trabajo hacía muchas instituciones honró ampliamente esta determinación final.

Hoy el hospital cuenta con distintas especialidades y con más de 20 servicios de atención abierta y cerrada, entre ellos se destacan los que tienen más personal a cargo de pacientes y estos son:

- UCIM
- UCI
- Traumatología mujer y hombre
- Traumatología infantil
- Pediatría
- Medicina mediana complejidad medica
- Medicina baja complejidad medica
- Urgencias adulto

En este hospital trabajan con distintos tipos de turnos, pero el que se estudió para este proyecto es:

- Cuarto turno: Este consiste en un turno que cubre las 24 horas del día, y se divide en un turno de “larga” que cumple con el horario de 8:00 hasta las 20:30 horas, mientras que los turnos de noche van desde las 20:00 hasta las 8:30. La explicación por la que se tiene media hora más, o sea (el turno dura 12 horas y 30 minutos) es para que el personal pueda cambiarse de ropa y estar en condiciones de iniciar turno de manera inmediata cuando el turno de la enfermera o paramédico anterior termine.

En el hospital ocurre la particularidad de que cuando un enfermero o paramédico no puede asistir a un turno previamente asignado, ya sea por vacaciones, día administrativo o licencia, el enfermero(a) jefe debe asignar el turno faltante a algún otro profesional de la salud para poder suplir la demanda, esto se hace con mismo personal que cuenta el servicio, por lo tanto debido a la poca dotación de personal, se hace imposible mantener un “cuarto turno” de manera rígida.

Uno de los problemas más importante para el enfermero jefe de servicio, es asignar los turnos, debido a la cantidad de horas que tienen que cumplir por mes los profesionales. En este Hospital existe una cantidad de horas que se divide en dos grupos llamados: horas diurnas (220 horas) y horas festivas (60 horas). Si algún trabajador antes mencionado llegase a pasar la cantidad de las horas diurnas, se da la opción para seguir cumpliendo horas, pero estas ya serían “horas extras festivas”, si llegase a pasar la cantidad de horas festivas antes mencionadas, el profesional ya no puede seguir cumpliendo más horas. Si sobrepasa estas horas, éstas no serán pagadas adicionalmente al sueldo.

6.2 Modelado del Caso de Estudio

Las siguientes tablas resumen los datos de entrada en los cuales se pueden desarrollar los modelos de los distintos turnos de manera simplificada.

Tabla 6.1 Cantidad de profesionales por área, y por turno

	Enfermeras			Paramédicos		
	Total	Larga/día	Noche	Total	Larga/día	Noche
UCIM	16	4	4	24	6	6
UCI	16	4	4	16	4	4
Traumatología mujer/hombre	4	1	1	32	8	8
Traumatología infantil	4	1	1	24	6	6
Pediatría	6	2	1	28	7	7
Medicina mediana complejidad	8	2	2	32	8	8
Medicina baja complejidad	6	2	1	32	8	8
Urgencias adulto	20	5	5	48	12	12

En esta tabla se puede apreciar un ejemplo de asignación de turnos, dado que la duración de los meses es irregular, se definirá el mes con una duración de “n días”.

Tabla 6.2 Ejemplo de asignación de turnos al mes

Días	1	2	3	4	5	6	7	8	9	10	11	12	13	14	n-días
Enfermeras															
1° turno	L	N			L	N			L	N			L	N	...
2° turno			L	N			L	N			L	N			...
3° turno		L	N			L	N			L	N			L	...
4° turno	N			L	N			L	N			L	N		...

Como se aprecia en la tabla 6.2 se han asignado L y N Para representar todos los turnos se dan valores entre 0 y 2. Por lo tanto, para el caso de las enfermeras: dado la enfermera i y el día j se tiene una variable E_{ij} la cual puede tomar 3 valores:

- $E_{ij} = 0$; Cuando la enfermera i tiene asignado un turno libre en el día j .
- $E_{ij} = 1$; Cuando la enfermera i tiene asignado un turno de larga en el día j .
- $E_{ij} = 2$; Cuando la enfermera i tiene asignado un turno de noche en el día j .

Mientras que en el caso de los paramédicos: dado el paramédico i y el día j se tiene una variable $P_{i,j}$ la cual puede tomar cinco valores:

- $P_{i,j} = 0$; Cuando el paramédico i tiene asignado un turno libre en el día j .
- $P_{i,j} = 1$; Cuando el paramédico i tiene asignado un turno de larga en el día j .
- $P_{i,j} = 2$; Cuando el paramédico i tiene asignado un turno de noche en el día j .

6.3 Turnos

6.3.1 Cuarto turno

Este tipo de turno es muy común y el más usado en los centros de salud. Para determinar el modelo correspondiente es necesario entender cómo funciona la rotación. Primero se define como una secuencia de cuatro días con turnos de forma “Larga – Noche – Libre – Libre” tal como muestra la tabla 2 en la página 10 (anterior).

Para todos estos turnos se definen algunas constantes:

- *Totalenf*: Representa el total de enfermeras según el servicio del hospital
- *Totalparam*: Representa el total de los paramédicos, al igual que el anterior el total de profesionales de la salud es variado, al tener distintos servicios, según la especialidad.
- *Totaldias*: Representa el total de días, dado que la duración de los meses cambia constantemente.

Se definen dos vectores que permiten incluir todos los turnos $[0,2]$, de la siguiente manera:

- $[E_{1,1}, E_{1,2}, \dots, E_{1,j}] \in [0,2]$, tal que $i \in [1, totalenf], j \in [1, totaldias]$
- $[P_{1,1}, P_{1,2}, \dots, P_{1,j}] \in [0,2]$, tal que $i \in [1, totalparam], j \in [1, totaldias]$

6.4 Formulación del Modelo Matemático

Teniendo identificadas las variables y restricciones, la formalización del modelo es la etapa a seguir:

6.4.1 Variables:

- $[E_{1,1}, E_{1,2}, \dots, E_{1,j}] \in [0,2]$, tal que $i \in [1, totalparam], j \in [1, totaldias]$
- $[P_{1,1}, P_{1,2}, \dots, P_{1,j}] \in [0,2]$, tal que $i \in [1, totalparam], j \in [1, totaldias]$

6.4.2 Constantes:

- *Totalenf*
- *Totalparam*
- *Totaldias*

6.4.3 Restricciones:

Para $i \in [1, totalenf] \wedge j \in [1, totaldias-2]$ se tiene:

- $[(E_{i,j} = 0 \wedge E_{i,j+1} = 0) \Rightarrow (E_{i,j+2} = 1)]$

Si en un día j se tiene un turno “libre” y al día siguiente ($j + 1$) también, entonces el día subsiguiente ($j + 2$) debe ser obligatoriamente un día con turno de larga.

- $[(E_{i,j} = 2) \Rightarrow (E_{i,j+1} = 0 \wedge E_{i,j+2} = 0)]$

Si en un día j se tiene un turno de “noche”, entonces obligatoriamente para los siguientes dos días (j y $j+1$) se debe tener turnos “libres”.

Para $i \in [1, totalenf] \wedge j \in [1, totaldias-1]$ se tiene:

- $[(E_{i,j} = 1) \Rightarrow (E_{i,j+1} = 2)]$

Si en un día j se ha asignado un turno de “larga”, entonces al día siguiente ($j + 1$) se debe asignar un turno de “noche”.

Estas restricciones definen el patrón que se repite a lo largo del mes para las enfermeras. Para los paramédicos se tendría la misma secuencia que en el caso de las enfermeras.

Para $i \in [1, totalparam] \wedge j \in [1, totaldias-2]$ se tiene:

- $[(P_{i,j} = 0 \wedge P_{i,j+1} = 0) \Rightarrow (P_{i,j+2} = 1)]$

- $[(P_{i,j} = 2) \Rightarrow (P_{i,j+1} = 0 \wedge P_{i,j+2} = 0)]$

Para $i \in [1, totalparam] \wedge j \in [1, totaldias-1]$ se tiene:

- $[(P_{i,j} = 1) \Rightarrow (P_{i,j+1} = 2)]$

7 Implementación

Este capítulo expone una breve reseña de JaCop, que es el *solver* que se utilizó. Se muestra y/o explica cómo resuelve el problema, basado en el sistema de asignación del tipo de turno expuesto en el capítulo 6 en la sección 3, por ejemplo los métodos a utilizar que incluye la biblioteca.

7.1 JaCop Solver

El *Java Constraint Programming Solver* abreviado JaCop, es una librería Java de código abierto, fue creado en el 2001 por krzysztof Kuchcinski y Radoslaw Szymanek , provee una tecnología dedicada al modelado y resolución de problemas con restricciones. El objetivo principal para JaCop es facilitar la investigación, existe un número significativo de publicaciones para la realización de experimentos, también existe un número determinado de investigadores de variados países tales como; Suecia, Francia, España, Irlanda, entre otros, que están usando JaCop con propósitos de investigación.

El desarrollo de esta biblioteca fue conducido con tres prioridades, estas son: facilidad de uso, facilidad del mantenimiento, y la eficacia. JaCop proporciona un conjunto de restricciones primitivas, que incluyen, operaciones aritméticas básicas, relaciones básicas (=, <, >, entre otras). También permite la utilización de restricciones a partir de predicados lógicos. Además contiene un número de restricciones globales.

7.2 Modelo del Prototipo

El modelo de prototipo está conformado por enfermeras y paramédicos que se ajustan a distintos tipos de turnos, entre ellos cuarto turno. El mes está representado por una variable, ya que los meses del año son irregulares, ósea que no tienen la misma cantidad de días.

Para ocupar esta biblioteca se requiere especificar declaraciones de importación para todas las clases, por ejemplo para importar todo el sub-paquete de JaCop se muestra a continuación:

```
Import org.jacop.core.*;
import org.jacop.constraints.*;
import org.jacop.search.*;
```

Las variables y restricciones creadas se almacenarán en un *Store* este debe ser creado antes de definir las variables y restricciones, un ejemplo de declaración:

```
Store store = new Store ();
```

Para definir las variables, JaCop ofrece un método para el dominio de las variables. Este permite que pueda ser representada por un solo intervalo con un valor mínimo y un valor máximo, llamado *BoundDomain*. Este método sirve para definir las variables en cuanto a turnos, ya que estas podrán tomar un valor mínimo de 0 y un valor máximo de 2.

Para que se pueda entender mejor, se presenta un ejemplo de código, que crea una variable *v* con un dominio.

```
IntVar v = new IntVar (s, "v", new BoundDomain (1,10));
```

Las restricciones que se ocuparán en el proyecto son del tipo primitivas por lo tanto van a ser declaradas con un método llamado “*PrimitiveConstraint*” ya que sólo se ocupan las relaciones básicas (=, ≠, ≤, <, ≥, >). Si bien estas restricciones están definidas con *PrimitiveConstraint* se deben imponer primero, para definir las como tal y ser ingresadas en el “store”.

Para imponer dichas restricciones se ocupa el método de *Store* “impose()”. Estas restricciones se definen mediante el método *XeqY(x₁,x₂)* la notación será de la forma:

```
PrimitiveConstraint c = new XeqY (x1,x2);
```

```
c.impose (store);
```

Pero como tengo restricciones tipo:

- $[(E_{i,j} = 2) \Rightarrow (E_{i,j+1} = 0 \wedge E_{i,j+2} = 0)]$

Se utilizara un método llamado “*IfThen(X,Y)*” para imponerlas.

```
Store.impose (new IfThen (x, new And (y,z)));
```

7.3 Prototipo

El prototipo fue desarrollado en el lenguaje Java, En eclipse versión Mars.1 Release (4.5.1) utilizando la librería de código abierto JaCop [1] versión 4.3.0. Este consta de distintas clases, Siendo la principal el Main.

7.3.1 Implementación de Código

Para la secuencia descrita por la restricción $[(E_{i,j} = 2) \Rightarrow (E_{i,j+1} = 0 \wedge E_{i,j+2} = 0)]$ del modelo matemático, se definen tres restricciones del tipo *PrimitiveConstraint*. Estas restricciones se definen mediante el método *XeqC (X, Const)*.

```
1 for(j=0; j<total_d-2; j++){
2     asig3= new XeqC (enfermeras.get(i).dias[j], 2);
3     asig4= new XeqC (enfermeras.get(i).dias[j+1], 0);
4     asig5= new XeqC (enfermeras.get(i).dias[j+2], 0);
5     store.impose(new IfThen(seq3, new And (asig4, asig5)));
6 }
```

Figura 7.1 Extracto de código ‘restricciones’

De manera análoga a la restricción $[(E_{i,j} = 1) \Rightarrow (E_{i,j+1} = 2)]$, se definen las restricciones seq3, seq4 y seq5 , las cuales definen que para los días j, j+1, j+2, se tengan valores 2,0,0 respectivamente.

En este caso, el argumento del método “*impose*” es distinto dado que lo que queremos representar es:

$(asig3 = 2) \rightarrow (asig4 = 0 \wedge asig5 = 0)$

Para representar lo anterior, se cambió el segundo parámetro del IfThen (X,Y), el cual se reemplaza por el método And (X,Y), donde X e Y son restricciones del tipo *PrimitiveConstraint*. De esta manera, la restricción para la secuencia 2->0->0 que representa la restricción $[(E_{i,j} = 2) \Rightarrow (E_{i,j+1} = 0 \wedge E_{i,j+2} = 0)]$ queda impuesta.

Al igual que en la secuencia anterior, se asegura la secuencia $[(E_{i,j} = 0 \wedge E_{i,j+1} = 0) \Rightarrow (E_{i,j+2} = 1)]$ de la misma manera que en la secuencia $[(E_{i,j} = 2) \Rightarrow (E_{i,j+1} = 0 \wedge E_{i,j+2} = 0)]$ como se ve en la figura 6.2.

```

1   for(j=0; j<total_d-2; j++){
2       asig6= new XeqC (enfermeras.get(i).dias[j], 0);
3       asig7= new XeqC (enfermeras.get(i).dias[j+1], 0);
4       asig8= new XeqC (enfermeras.get(i).dias[j+2], 1);
5       store.impose(new IfThen (new And (asig6, asig7),asig8));
6   }
```

Figura 7.2 Extracto de código ‘restricciones’

Cada profesional tendrá un vector el cual guarda y representa los días del mes, junto a su planificación, la cual está determinada.

Se define un vector días (días [i]), este es un vector que guarda variables enteras, las cuales pueden tomar valores enteros entre “0” y “2”, para los cuales el valor 0 representa a la enfermera tiene un día libre, un valor 1 el cual representa a la enfermera tiene un turno de larga. Un valor 2 cuando la enfermera tiene un turno de noche.

```

1   public enfermeras() {
2       for (int i=0; i<dias_mes; i++)
3           dias [i] = new IntVar (store, "Dia"+i, new BoundDomain (0, 3));
4   }
```

Figura 7.3 Extracto de código ‘enfermeras’

En la línea 3 de la figura 6.3, cada variable que se agrega al vector se guarda automáticamente asignándole un *store*, un nombre, y un límite inferior (0) y un límite superior (2).

La variable “store” de tipo *Store* utilizada como parámetro, es la responsable de almacenar todas las variables y las restricciones para las cuales se busca una solución.

8 Interfaz Gráfica

En esta sección se presentará la interfaz, el principal objetivo es que el usuario pueda manipular con facilidad y permita una comunicación y/o interacción sencilla entre el usuario y el *software*.

Generalmente la interfaz gráfica es creada a un usuario en particular, en este caso el usuario será un(a) enfermero(a) con nivel de conocimientos básicos de funcionamiento y manipulación en computación

En el contexto del proceso de interacción persona-computador, la interfaz gráfica es el sistema interactivo que brinda la posibilidad, a través del uso, simplificar tareas que desempeñan las personas; en este caso enfermeros, se le permitirá tener un acceso rápido y simple para la asignación de turnos.

8.1 Acceder al sitio principal

En esta ventana se muestra el acceso al sitio principal, en donde pide un usuario y una contraseña, estos están guardados en un archivo XML.



Figura 8.1 Captura de pantalla 'Acceder sitio principal'

8.2 Registrar usuario

Esta ventana muestra el registro de un usuario, si no está registrado tiene la opción de crear un nuevo usuario.



The image shows a screenshot of a web application window titled "Regístrate". The window has a light blue background and a purple header bar. Below the header, the word "Registro:" is displayed in bold blue text. To the right of this text are two small icons: a document with a list and a circular arrow. Below the header, there are four input fields, each preceded by a label: "Nombre:", "Apellidos:", "Usuario:", and "Contraseña:". Each label and its corresponding input field are aligned to the left. The input fields are empty text boxes.

Figura 8.2 Captura de pantalla 'Registrar usuario'

8.3 Usuario incorrecto

En esta ventana se muestra el acceso al sitio principal, pero con un usuario incorrecto. Se mostrara un mensaje de error y se volverá a pedir usuario y contraseña.



Figura 8.3 Captura de pantalla 'Usuario incorrecto'

8.4 Registro trabajador

En esta ventana se muestra el acceso a registro trabajador, esta función muestra distintas funciones, el "registrar..." da la posibilidad de ingresar nuevos profesionales. La opción buscar, busca un profesional, ya sea paramédico o enfermera. La opción eliminar (elimina el trabajador buscado anteriormente). La opción limpiar, (limpia todos los campos).

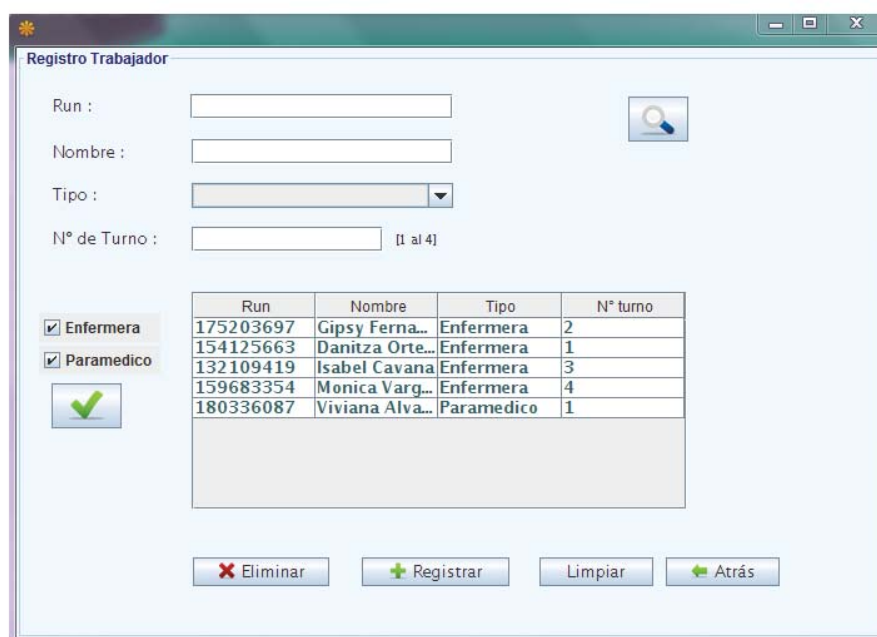


Figura 8.4 Captura de pantalla 'Registro trabajador'

8.5 Asignación

En esta ventana se muestra la asignación en sí, con los nombres de cada enfermera los cuales están guardados en un archivo XML, una de las funcionalidades más importante es la opción que ofrece para elegir el mes, ya que durante el año la cantidad de días para cada mes es diferente, ejemplo el mes de Enero 31, Febrero 29, Abril 30, entre otros.

The screenshot shows a software window titled 'Archivo' with a 'Configuración' section. It includes checkboxes for 'Enfermeras' (checked) and 'Paramédicos' (unchecked), a dropdown menu for 'Mes para asignar' set to 'Marzo', and a text input for 'Cantidad de profesionales' set to '4'. A table below shows the assignment schedule for four professionals: Gipsy Fernandez, Danitza Ortega, Isabel Cavana, and Monica Vargas, across the days of the month.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Gipsy Fernandez	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	
Danitza Ortega	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N
Isabel Cavana	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--
Monica Vargas	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--	L	N	--	--

Figura 8.5 Captura de pantalla 'Asignación'

8.6 Modo de guardar información

Los archivos XML son un subconjunto de SGML (*Estándar Generalised Mark-up Language*), simplificado y adaptado a Internet. Este tipo de archivo es un estándar internacionalmente conocido y no pertenece a ninguna compañía. La función principal del XML es que representa la información estructurada en la web, de modo que esta información pueda ser almacenada, transmitida, procesada, visualizada e impresa, por diversos tipos de aplicaciones. Una de las principales ventajas es su fácil procesamiento y lectura ya que separa radicalmente el contenido y el formato de presentación.

Para guardar la información de este proyecto se crearon distintos archivos XML, entre estos se encuentra un archivo llamado data.xml, en este se encuentra la información de todos los usuarios registrados, los datos guardados se dividen en: usuario (este define el nombre del usuario, el que ocupara para iniciar sesión) y pass (el que define una contraseña, esta se encuentra encriptada para dar mayor seguridad).

```
<?xml version="1.0" encoding="UTF-8"?>
- <usuarios>
  - <user>
    <usuario>Administrador</usuario>
    <pass>8d969eef6ecad3c29a3a629280e686cf0c3f5d5a86aff3ca12020c923adc6c91</pass>
  </user>
  - <user>
    <usuario>Vivi</usuario>
    <pass>8d969eef6ecad3c29a3a629280e686cf0c3f5d5a86aff3ca12020c923adc6c92</pass>
  </user>
  - <user>
    <usuario>EnfermeroJefe</usuario>
    <pass>20d2995509ff4357692e8c421f3f81b876b0b2391dc84108fd34d86edc33201a</pass>
  </user>
</usuarios>
```

Figura 8.6 Captura de pantalla 'XML usado'

9 Resultados

En este capítulo se exhiben los resultados obtenidos de la encuesta realizada a diferentes enfermeros del hospital Carlos Van Buren.

En esta encuesta se busca obtener información relacionada a la usabilidad de la plataforma.

9.1 Notas

El cuadro siguiente muestra las notas obtenidas en cada encuesta realizada.

Tabla 9.1 Resultado notas

Anónimo (Encuestas)/puntos	NOTA
1./ 21 Puntos	6,5
2../23 Puntos	6,7
3./ 18 puntos	6,3
4./ 15 puntos	5,8
5./ 20 puntos	6,5
6./ 15 puntos	5,8
Promedio	6,26

9.2 ¿Es recomendable?

La siguiente tabla muestra los resultados respecto a la pregunta número 7, la cual define si el software es recomendable

Tabla 9.2 Recomendación de Software

(Encuestas)	Respuesta
Anónimo 1	Si
Anónimo 2	Si
Anónimo 3	SI
Anónimo 4	No
Anónimo 5	Si
Anónimo 6	No
Porcentaje	Si 66,6% / no 33,3 %

9.3 Grafico Recomendación del software

Como podemos ver en el siguiente gráfico, un 67% de las personas encuestadas recomendaría el software, esto se debe principalmente por la facilidad de uso para los usuarios y las buenas herramientas que entrega.

Por otro lado el 33% de las personas encuestadas dijo no recomendar el software. Por sus respuestas esto se debe principalmente a la poca comprensión de la asignación de turnos para cada persona.

Cabe destacar que la encuesta se realizó a un grupo sin segmentación de edad y los resultados pueden variar dependiendo de la experiencia con software similares.

Recomendación del software

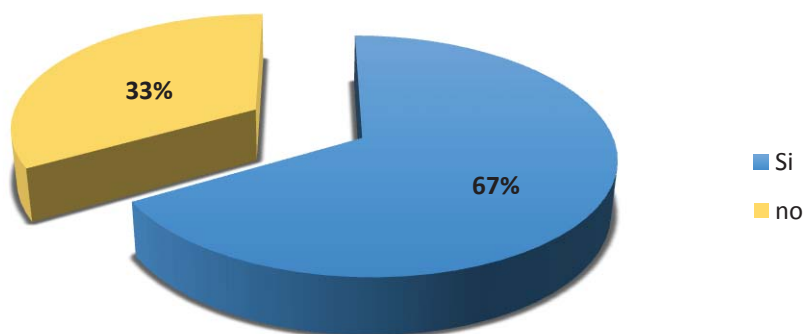


Figura 9.1 Grafico Recomendación del software

10 Conclusiones

Después de presentar y describir el problema de asignación de turnos de enfermería dirigido al Hospital público de Valparaíso “Carlos Van Buren”, y haber mostrado el tipo de asignación de los sistemas de turnos que ocupa este centro, que hoy por hoy se realizan manualmente por el jefe de servicio y/o área. Se ha realizado y/o aplicado el modelo matemático que satisface distintos tipos de restricciones enfocados al turno (cuarto turno) y en las horas de trabajo que pueden realizar los profesionales (con un máximo que definirá el tope de turnos que pueden realizar), incluyendo las variables y constantes que se ocuparan.

Se realizaron entrevistas a los enfermeros(as) jefes de los distintos servicios de dicho hospital y se solicitaron los datos necesarios para la creación de las restricciones, estos dejan en claro la responsabilidad que tienen al hacer que funcione a la perfección, la asignación de turnos, sin embargo al ser llevado a la realidad esto es algo muy difícil de cumplir, debido a que los profesionales constantemente están presentando licencias médicas, vacaciones, y días administrativos, es por esta razón en que los enfermeros jefes deciden tener enfermeros y paramédicos “suplentes”, estos cubren todos los vacíos que dejan los trabajadores contratados.

Para realizar este trabajo se debía tener un conocimiento básico sobre la Programación con restricciones, conceptos y definiciones de variables, dominios y restricciones.

Al tener definidas todas las restricciones a ocupar, se decidió implementar la solución al problema con el *solver* llamado *Java Constraint Programming* abreviado JaCop. Luego de tener el aplicativo listo se procedió a realizar distintas encuestas a enfermeros de dicho hospital, se obtuvieron buenos resultados al llevarlo a la realidad. Como trabajo futuro se pretende realizar funciones extras, de acuerdo a los resultados y sugerencias obtenidos de la encuesta realizada a los enfermeros, y que el aplicativo sea utilizado en el hospital.

Referencias Bibliográficas

- [1] K. Kuchcinski y R. Szymanek. *JaCop Library User's Guide*, Version 4.3, July 28 2015
- [2] F. Barber and M. A. Salido. Introducción a la programación con restricciones. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, vol. 7, num. 20, 13-30, 2003.
- [3] R. Soto, B. Crawford, E. Monfroy, W. Palma, y F. Paredes. *Nurse and Paramedic Rostering with Constraint Programming: A Case Study*. In *Romanian Journal of information science and technology*, vol. 16, num. 1, pp. 52-64, 2013.
- [4] R. Haralick & G. Elliott. *Increasing tree efficiency for constraint satisfaction problems*. *Artificial Intelligence*, vol. 14, pp. 263-313, 1980.
- [5] F. Barber and M. A. Salido. *A constraint ordering heuristic for scheduling problems*. In *Proceeding of the 1st Multidisciplinary international conference on Scheduling: Theory and Applications*, vol. 2, pp. 476-490, 2003.
- [6] K. Kuchcinski, *Constraints-Driven Scheduling and Resource Assignment In ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol 8, num. 3, pp. 355-383, 2003
- [7] S. Minton, A. Philips, M. D. Johnston y P. Laird, *Minimizing conflicts: A heuristic repair method for constraint-satisfaction and scheduling problems*, *Artificial Intelligence Research*, vol.58, pp. 161-205, 1993
- [8] R. Dechter and I. Meiri. *Experimental evaluation of preprocessing algorithms for constraint satisfaction problems*. *Artificial intelligence*, 68, pp. 211-241,1994
- [9] P. A. Geelen. *Dual viewpoint heuristics for binary constraint satisfaction problems*. In *Proceeding of the 10th European conference on Artificial Intelligence*, pp. 31-35, 1992
- [10] K. Kuchcinski y C. Wolinski, *Global approach to assignment and scheduling of complex behaviors based on HCDG and constraint programming*, *Journal of Systems Architecture*, vol. 49, num. 12-15, pp.489-503, 2003.
- [11] P. Shaw, *A Constraint for Bin Packing*, In *Proceedings of the 10th International Conference on Principles and Practice of Constraint Programming (CP 2004)*, pp. 648-662, 2004.
- [12] E. Burk, J. Li, R. Qu., *A hybrid model of integer programming and variable neighborhood search for highly-constrained nurse rostering problems*, *European Journal of Operational Research*, vol. 203, num. 8, pp.484-493, 2010.
- [13] J. C. Regin, *A filtering algorithm for constraint of difference in CSPs*. In *12th National Conference on Artificial Intelligence (AAAI-94)*, vol. 1, pp. 362-367, 1994.
- [14] LD. Smith. y A. Wiggins, *A computer- Based Nurse Scheduling System*, *Computers and Operation Research*, vol. 4, num. 3 pp. 195-212. 1977.

- [15] E. K. Burke, P. de Causmaecker, G. Vanden Berghe, y H. Van Landeghem, *The State of the Art of Nurse Rostering*, *Journal of Scheduling*, vol. 7, num. 6, pp. 441-499, 2004.
- [16] U. Aickelin, y K. Dowsland, *Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem*, *Journal of Scheduling*, vol. 3, pp. 139-153, 2000.
- [17] E. K. Burke, P. De Causmaecker, S. Petrovic, and G. Vanden Berghe, *Fitness evaluation for Nurse scheduling problems*, in *Proceedings of Congress on Evolutionary Computation, CEC2001*, IEEE Press, pp. 1139-1146, 2001.
- [18] J. G. Chen y T. Yeung, *Hybrid expert system for Nurse shift scheduling*, *Computers in nursing*, vol. 11, num. 4, pp. 183-190, 1993.
- [19] F. Easton, D. Rossin, and W. Borders, *Analysis of alternative scheduling policies for hospital Nurses*, *Production and Operations Management*, vol. 1, num 2, pp. 159-174 1992.
- [20] S. J. Darmoni, A. Fajner, N. Mahé, A. Leforestier, M. Vondracek, O. Stelian, M. Baldenweck, *Horoplan: computer-assisted nurse scheduling using constraint-based programming*, *Journal of the Society for Health Systems*, vol. 5, num. 1, pp. 41-54, 1994.
- [21] D. Sitompul y SU. Randhawa, *Nurse scheduling models: a state-of-the-art review*, *Journal of the Society for Health Systems*, vol. 2, num. 1, pp. 62-72, 1990.
- [22] M. Warner D. y J. Prawda, *A Mathematical Programming Model for Scheduling Nursing Personnel in a Hospital*, *Management Science*, vol. 19, num. 4, pp. 411-422 1972.
- [23] A. Hon Wai Chun, S. Ho Chuen Chan, G. Pui Shan Lam, F. Ming Fai Tsang, J. Wong and D. Wai Ming Yeung. *Nurse Rostering at the Hospital Authority of Hong Kong “ Innovative Applications of Artificial Intelligence, 2000”*.
- [24] M. Vandankumar Trivedi y M. Warner, *A Branch and Bound Algorithm for Optimum Allocation of Float Nurses*, *Management Science*, vol. 22, num. 9, pp. 972-981, 1976
- [25] MW Isken y WM Hancock., *A heuristic approach to nurse scheduling in hospital units with non-stationary, urgent demand, and a fixed staff size*, *Journal of the Society for Health Systems*, 2(2), pp. 24-41. 1991

Anexos

A Encuesta realizada a enfermeros del Hospital

Evaluación del software “Designación de turnos enfermeros”

Instrucciones:

Leer atentamente cada casilla correspondiente al número que indica el grado o categorías y asignar valores de 1 a 4 mediante una “X” a cada indicador.

Es de carácter Anónimo

Los puntajes de cada uno de los criterios a evaluar estarán indicados en cada uno de ellos.

Los criterios de evaluación son: **Excelente [4]; Bien [3]; Regular [2]; Malo [1]**

Tabla A.0.1 Preguntas encuesta

Indicadores	4	3	2	1
1.- Calidad de la interfaz. 4 puntos.				
2. Con respecto a la designación de turnos, (Fácil ingreso datos). 4 puntos.				
3.- Es entendible la lista de turnos que se le asignan a cada enfermero. 4 puntos.				
4.-Búsqueda y eliminación de enfermeros 4puntos.				
5.- Botones en general (utilización, compresión, ayuda) 4 puntos.				
6.-Rendimiento general del software 4puntos.				
7. Es recomendable el software	Si		No	

Tabla A.2 Puntaje de cada pregunta

Sugerencia:

Puntaje: _____

Nota: _____

CRITERIOS A EVALUAR	NOTA
24 puntos	7,0
23-22 puntos	6,7
21-20 puntos	6,5
19-18 puntos	6,3
17-16 puntos	6,0
15-14 puntos	5,8
13-12 puntos	5,5
11-10 puntos	5,2
9-8 puntos	4,8
7-6 puntos	4,5
5-4 puntos	4,0
4-3 puntos	3,5
2-1 puntos	3,0
0 puntos	2,0

Valores	Referente en nota
1. Malo	2.0
2. Regular	4.0
3. Bien	6.0
4. Excelente	7.0