

UNIVERSIDAD PÚBLICA DE EL ALTO

CARRERA INGENIERÍA DE SISTEMAS



TESIS DE GRADO

**“MODELO DE PREDICCIÓN APLICADO AL INDICE
CRECIMIENTO DE CANCER DE CUELLO UTERINO EN LA
CIUDAD DE EL ALTO EN BASE A ALGORITMOS GENÉTICOS”
CASO: (SEDES EL ALTO)**

Para optar al título de Licenciatura en Ingeniería de Sistemas

MENCIÓN: INFORMÁTICA Y COMUNICACIONES

**Postulante : Jaime Marcial Hidalgo Mamani
Tutor Metodológico : M. Sc. Ing. Enrique Flores Baltazar
Tutor Especialista : Ing. Ramiro Kantuta Limachi
Tutor Revisor : Ing. Elías Carlos Hidalgo Mamani
EL ALTO – BOLIVIA
2020**

DEDICATORIA

A Eva Julia mi amada esposa, por todo el apoyo y comprensión incondicional, a mis hijos Israel, Gabriela y Ángel por su paciencia, atención y fortaleza durante este tiempo.

A mis padres Tomás y Teresa (+) por darme la vida e inculcar en mí los valores del respeto, esfuerzo y valentía, de no temer las adversidades porque Dios está conmigo siempre.

Jaime Marcial Hidalgo Mamani

AGRADECIMIENTOS

Le agradezco a Dios por haberme acompañado y guiado a lo largo de mi carrera, por ser mi fortaleza en los momentos de debilidad y por brindarme una vida llena de aprendizajes, experiencias y sobre todo felicidad.

A mi familia por ser parte importante de mi vida y representar la unidad familiar.

Al Ingeniero Enrique Flores Baltazar por su apoyo, paciencia, comprensión y colaboración en la planeación, desarrollo y aplicación del presente trabajo de investigación.

Al Ingeniero Ramiro Kantuta por su apoyo y predisposición en la puntualización de conceptos importantes.

Al Ingeniero Elías Carlos Hidalgo por su apoyo, comprensión y corrección del presente trabajo de investigación.

A mis amigos por confiar y creer en mí y haber hecho de mi etapa universitaria un trayecto de vivencias que nunca olvidaré. A todos ellos por llenar mi vida de grandes momentos que hemos compartido.

A la Universidad Pública de El Alto, por acogerme y brindarme una nueva familia dentro de sus aulas.

A los docentes por todo el apoyo brindado a lo largo de la carrera, por su tiempo, amistad y por los conocimientos que me transmitieron.

.

RESUMEN

La presente tesis de grado plantea un modelo de predicción del índice de crecimiento de cáncer de cuello uterino en base a algoritmos genéticos que permite pronosticar el crecimiento de esta enfermedad en cinco años, además este instrumento es de gran beneficio para los especialistas y mujeres. Se considera como objeto de estudio, el índice de crecimiento de cáncer de cuello uterino en base a datos estadísticos facilitados por SEDES El Alto. El cáncer de cuello uterino es una de las causas principales de muerte en mujeres de nuestro país y el mundo, que puede afectar a todas las edades, donde el riesgo de sufrir se incrementa con la edad.

El modelo de predicción muestra la cantidad de mujeres con cáncer de cuello uterino, clasificados en grupos etarios y gestiones del 2015 al 2020, asimismo se visibiliza los índices históricos, selección, cruce, mutación e índices pronosticados en números binarios, que se interpretan por medio de porcentajes. Por otro lado, la población inicial y los índices de crecimiento visibiliza gráficamente el comportamiento del número de pacientes con cáncer de cuello uterino en relación a los años.

Posteriormente se realiza la simulación del modelo de predicción a través de un prototipo desarrollado en Java con base a algoritmos genéticos. Se considera los datos históricos recolectados como muestras de comparación.

Los datos históricos obtenidos evalúan el modelo de predicción, llegando a concluir: que los diagnósticos obtenidos tienen un grado de confiabilidad de un 95 % respecto a los resultados reales.

Palabras claves: Índice de crecimiento, modelo de predicción, algoritmos genéticos, cáncer de cuello uterino.

ABSTRACT

This degree thesis proposes a model for predicting the growth rate of cervical cancer based on genetic algorithms that allows predicting the growth of this disease in five years, and this instrument is of great benefit to specialists and women. The growth rate of cervical cancer is considered as an object of study, based on statistical data provided by SEDES El Alto. Cervical cancer is one of the leading causes of death in women in our country and the world, which can affect all ages, where the risk of suffering increases with age.

The prediction model shows the number of women with cervical cancer, classified into age groups and administrations from 2015 to 2020, as well as the historical indices, selection, crossing, mutation and predicted indices in binary numbers, which are interpreted by percentages. On the other hand, the initial population and the growth rates graphically show the behavior of the number of patients with cervical cancer in relation to the years.

Subsequently, the simulation of the prediction model is carried out through a prototype developed in Java based on genetic algorithms. Historical data collected is considered as comparison samples.

The historical data obtained evaluate the prediction model, concluding: that the diagnoses obtained have a degree of reliability of 95% with respect to the real results.

Key words: Growth rate, prediction model, genetic algorithms, cervical cancer.

INDICE GENERAL

CAPITULO I

1.	MARCO PRELIMINAR	1
1.1.	Introducción	1
1.2.	Antecedentes.....	3
1.3.	Planteamiento del Problema	5
1.4.	Objetivos.....	6
1.5.	Hipótesis.....	7
1.6.	Justificaciones	9
1.7.	Metodología (Metodologías Ágiles).....	11
1.8.	Herramientas	18
1.9.	Límites y Alcances.....	20
1.10.	Aportes	21

CAPITULO II

2.	MARCO TEÓRICO.....	22
2.1.	Introducción	22
2.2.	El Cáncer.....	22
2.3.	Inteligencia Artificial	27
2.4.	Algoritmos Genéticos.....	31
2.5.	Método Científico (Bunge)	43
2.6.	Modelo XP	47
2.7.	Métrica de calidad ISO 9126.....	58
2.8.	Estudio Costo Beneficio COCOMO II.....	70
2.9.	JAVA	75
2.10.	PHP	78
2.11.	MATLAB	80
2.12.	Base de Datos MySQL	83

CAPITULO II

3.	MARCO APLICATIVO	85
-----------	-------------------------------	-----------

3.1.	Introducción	85
3.2.	Análisis de Requerimiento	91
3.3.	Modelado en Base a la Metodología XP	93
3.4.	Implementación del Modelo	116
3.5.	Calidad Del Software (COCOMO II).....	126
3.6.	Costos (COCOMO II).....	131

CAPITULO IV

4.	PRUEBAS Y RESULTADOS	137
4.1.	Prueba al Modelo de Predicción	137
4.2.	Predicción Según el Modelo Implementado	137
4.3.	Pruebas	139

CAPITULO V

5.	CONCLUSIONES Y RECOMENDACIONES	144
5.1.	Introducción	144
5.2.	Estado de los Objetivos	144
5.3.	Estado de la Hipótesis	146
5.4.	Conclusiones	147
5.5.	Recomendaciones	147

BIBLIOGRAFIA	
---------------------------	--

ANEXOS	
---------------------	--

ANEXO A. Arbol de problemas

ANEXO B. Arbol de objetivos

ANEXO C. Manual de usuario

INDICE ESPECÍFICO

CAPITULO I

1.	MARCO PRELIMINAR	1
1.1.	Introducción	1
1.2.	Antecedentes.....	3
1.2.1.	Antecedentes Internacionales.	3
1.2.2.	Antecedentes Nacionales.....	4
1.2.3.	Antecedentes Locales	5
1.3.	Planteamiento del Problema	5
1.3.1.	Problema General	6
1.3.2.	Problemas Específicos	6
1.4.	Objetivos.....	6
1.4.1.	Objetivo General	6
1.4.2.	Objetivos Específicos	6
1.5.	Hipótesis.....	7
1.5.1.	Identificación De Variables	7
1.5.2.	Operaciones De Variables.....	8
1.5.3.	Conceptualización De Variables.....	8
1.5.3.1.	Índice De Cáncer Uterino	8
1.5.3.2.	Modelo De Predicción	9
1.5.3.3.	Algoritmos Genéticos	9
1.6.	Justificaciones	9
1.6.1.	Justificación Técnica	9
1.6.2.	Justificación Económica	10
1.6.3.	Justificación Social	10
1.6.4.	Justificación Científica.....	11
1.7.	Metodología (Metodologías Ágiles).....	11
1.7.1.	Método Científico	11
1.7.2.	Método De Ingeniería.....	13

1.7.2.1. Metodología XP (Extreme Programming).....	13
1.7.2.2. ISO 9126 (Métrica de calidad).....	16
1.7.2.3. Costos (COCOMO II).....	17
1.8. Herramientas	18
1.8.1. Lenguajes de Programación.....	18
1.8.2. Base de Datos.....	19
1.9. Límites y Alcances.....	20
1.9.1. Límites.....	20
1.9.2. Alcances	20
1.10. Aportes	21
CAPITULO II	
2. MARCO TEÓRICO.....	22
2.1. Introducción	22
2.2. El Cáncer.....	22
2.2.1. Fases del Cáncer	24
2.2.2. Cáncer de Cuello Uterino (VPH).....	25
2.2.3. Estructura del VRION.....	26
2.3. Inteligencia Artificial	27
2.4. Algoritmos Genéticos.....	31
2.4.1. Fundamentos los Algoritmos Genéticos	32
2.4.2. Características de los Algoritmos Genéticos	33
2.4.2.1. Codificación de la Variable.....	35
2.4.2.2. Proceso de Selección	35
2.4.2.3. Cruce y Mutación	36
2.4.2.4. Modificaciones al Algoritmo Genético Simple.....	38
2.4.3. Programación Evolutiva	39
2.4.4. Estrategias de Evolución.....	40
2.4.5. Programación Genética.....	42
2.5. Método Científico (Bunge)	43
2.5.1. Características del método científico	43

2.5.2.	La Pauta de la Investigación Científica Diseño de la prueba	44
2.5.3.	Objetivos del método científico	46
2.6.	Modelo XP	47
2.6.1.	Fase de Exploración.....	49
2.6.2.	Fase de Planificación	49
2.6.3.	Fase de Iteraciones.....	49
2.6.4.	Reglas y Practicas.....	50
2.6.4.1.	Planificación.....	50
2.6.4.2.	Diseño	52
2.6.4.3.	Desarrollo del Código	54
2.6.4.4.	Pruebas	57
2.7.	Métrica de calidad ISO 9126.....	58
2.8.	Estudio Costo Beneficio COCOMO II.....	70
2.8.1.	Modelo básico.....	72
2.9.	JAVA	75
2.9.1.	Conceptos Previos de Poo	76
2.9.2.	Concepto de Clase y Objeto.....	76
2.9.3.	Concepto de Campo, Método y Constructor.....	77
2.9.4.	Concepto de Herencia y Polimorfismo.....	77
2.9.5.	Modificadores de Acceso	77
2.9.6.	Clases Abstractas e Interfaces	78
2.10.	PHP	78
2.11.	MATLAB	80
2.11.1.	Aplicación de Matlab	81
2.12.	Base de Datos MySQL	83
2.12.1.	Características de MySQL.....	83

CAPITULO II

3.	MARCO APLICATIVO	85
3.1.	Introducción	85
3.1.1.	Población	86

3.1.2.	Selección	89
3.1.3.	Cruce y Mutación	90
3.1.4.	Nueva Población	91
3.2.	Análisis de Requerimiento	91
3.2.1.	Requerimientos Funcionales	92
3.2.2.	Requerimientos no Funcionales	93
3.3.	Modelado en Base a la Metodología XP	93
3.3.1.	Fase I: Exploración.....	94
3.3.2.	Fase II: Planificación	97
3.3.2.1.	Estimación de Esfuerzo	98
3.3.2.2.	Planificación de Iteraciones	99
3.3.3.	Fase III: Iteraciones.....	100
3.3.3.1.	Primera Iteración.....	101
3.3.3.1.1.	Historia 1: Acceso al Sistema.....	101
3.3.3.1.2.	Historia 2: Recolección de Datos	103
3.3.3.1.3.	Historia 3: Entrenamiento de Algoritmos	104
3.3.3.2.	Pruebas de Aceptación	107
3.3.3.2.1.	Historia 1: Acceso al Sistema.....	107
3.3.3.2.2.	Historia 2: Recolección de Datos	108
3.3.3.2.3.	Historia 3: Entrenamiento de Algoritmos	110
3.3.3.3.	Segunda Iteración.....	111
3.3.3.3.1.	Historia 1: Predicción del Sistema.....	111
3.3.3.3.2.	Historia 2: Generación de Predicción	113
3.3.3.4.	Pruebas de Aceptación	114
3.3.3.4.1.	Historia 1: Predicción del Sistema.....	114
3.3.3.4.2.	Historia 2: Generación de Predicción	115
3.4.	Implementación del Modelo	116
3.4.1.	Datos.....	116
3.4.1.2.	Datos Estandarizados	118
3.4.2.	Etapas de Funcionamiento del Modelo.....	119
3.4.3.	Implementación de Algoritmos	120

3.4.4.	Publicación, Despliegue	124
3.4.5.	Resultados	125
3.5.	Calidad Del Software (COCOMO II).....	126
3.5.1.	Funcionalidad.....	126
3.5.2.	Confiabilidad	126
3.5.3.	Usabilidad	127
3.5.4.	Eficiencia.....	128
3.5.5.	Mantenibilidad	129
3.5.6.	Portabilidad	130
3.5.7.	Resultados	130
3.6.	Costos (COCOMO II).....	131
3.6.1.	Puntos de función.....	131
3.6.2.	Aplicación de COCOMO II.....	134
3.6.3.	Costo de elaboración del Software	136
3.6.4.	Costo total.....	136

CAPITULO IV

4.	PRUEBAS Y RESULTADOS	137
4.1.	Prueba al Modelo de Predicción	137
4.2.	Predicción Según el Modelo Implementado	137
4.3.	Pruebas	139
4.3.1.	Prueba de la Hipótesis	139

CAPITULO V

5.	CONCLUSIONES Y RECOMENDACIONES	144
5.1.	Introducción	144
5.2.	Estado de los Objetivos	144
5.3.	Estado de la Hipótesis	146
5.4.	Conclusiones	147
5.5.	Recomendaciones	147

BIBLIOGRAFIA

ANEXOS

ANEXO A. Arbol de Problemas

ANEXO B. Arbol de Objetivos

ANEXO C. Manual de Usuario

INDICE DE FIGURAS

1.	MARCO PRELIMINAR	1
	Figura 1.1 Ciclo de vida de XP	15
	Figura 1.2 Factores ISO 9126.....	17
2.	MARCO TEÓRICO.....	22
	Figura 2.1 Etapas típicas del desarrollo del Cáncer.....	25
	Figura 2.2 Estructura del Virus del Papiloma Humano	26
	Figura 2.3 Tópicos relacionados con la Inteligencia Artificial	30
	Figura 2.4 Espacios Genotípicos y Fenotípicos en la Evolución	32
	Figura 2.5 Pseudocódigo de un Algoritmo Genético Simple	34
	Figura 2.6 Ruleta de Selección del más Apto	36
	Figura 2.7 Operaciones de cruce y mutación.....	37
	Figura 2.8 Pseudocódigo de Algoritmo para Programación Evolutiva	40
	Figura 2.9 Pseudocódigo de Procedimiento de Estrategias de Evolución.....	41
	Figura 2.10 Pseudocódigo para Procesamiento de Programación Genética ...	42
	Figura 2.11 Ciclos de desarrollo en cascada y tradicional.....	48
	Figura 2.12 Norma de Evaluación ISO/IEC 9126	59
	Figura 2.13 Evaluación Interno, externa y Calidad de uso ISO/IEC 9126	60
	Figura 2.14 Característica de funcionalidad	61
	Figura 2.15 Característica de Confiabilidad	62
	Figura 2.16 Característica de Usabilidad	63
	Figura 2.17 Característica de Eficiencia.....	65
	Figura 2.18 Característica de Mantenimiento.....	66
	Figura 2.19 Característica de portabilidad	67
	Figura 2.20 Característica Calidad de uso	69
	Figura 2.21 Resumen de la ISO/IEC 9126.....	70
3.	MARCO APLICATIVO	85
	Figura 3.1 Modelo de predicción CACUGEN	86
	Figura 3.2 Crecimiento de la población en la Ciudad de El Alto	87

Figura 3.3 Municipio de El Alto: Progresiones de Población por sexo, 2019.....	87
Figura 3.4 Ruleta de Selección del más Apto	90
Figura 3.5 punto de cruce para mutación e hijos	91
Figura 3.6 Cruce, mutación y nueva población	95
Figura 3.7 Diseño rápido del sistema.....	100
Figura 3.8 Gráficos que mostrara el sistema.....	100
Figura 3.9 Ingreso al Modelo de Predicción CACUGEN	102
Figura 3.10 Caso de Uso Selecciona.....	103
Figura 3.11 Relación Edad, Gestión y Fase.....	105
Figura 3.12 Población de cruce generada	106
Figura 3.13 Entrenamiento en MATLAB	107
Figura 3.14 Ventana principal	108
Figura 3.15 Base de datos.....	109
Figura 3.16 Algoritmo MultiplayerPerceptron	111
Figura 3.17 Predicción del sistema	112
Figura 3.18 Predicción del cáncer de cuello uterino.....	113
Figura 3.19 Funcionamiento del Modelo	114
Figura 3.20 Visibilizarían de la población final	115
Figura 3.21 Servicios atendidos en la gestión 2019 Sedes El Alto.....	117
Figura 3.22 Esquema del funcionamiento del modelo de predicción.....	119
Figura 3.23 Resultado de la predicción.....	125
4. PRUEBAS Y RESULTADOS	137
Figura 4.1 Modelo de predicción de Cáncer de Cuello Uterino	138
Figura 4.2 Campana de Gauss representando T Student.....	142
5. CONCLUSIONES Y RECOMENDACIONES	144

BIBLIOGRAFIA

ANEXOS

INDICE DE TABLAS

Tabla 1.1 Operación de Variables.....	8
Tabla 2.1 Población inicial, cromosomas y aptitud.....	35
Tabla 2.2 Equivalencia de Código - Gray.....	38
Tabla 2.3 Estimación de esfuerzo.....	72
Tabla 2.4 Valores de los factores de escala.....	74
Tabla 3.1 Tabla de la población en Grupos Etarios y por gestión.....	88
Tabla 3.2 Clasificación de Fases del Cáncer.....	89
Tabla 3.3 Población inicial, cromosomas y aptitud.....	89
Tabla 3.4 Requerimiento funcionales de CACUGEN.....	92
Tabla 3.5 Rendimiento no Funcional.....	93
Tabla 3.6 Historia 1.....	94
Tabla 3.7 Conexión del usuario.....	95
Tabla 3.8 Historia 2.....	95
Tabla 3.9 Interfaz datos.....	96
Tabla 3.10 Historia 3.....	96
Tabla 3.11 Algoritmos.....	97
Tabla 3.12 Estimación de Esfuerzo.....	98
Tabla 3.13 Planificación de iteraciones.....	99
Tabla 3.14 Tarjeta CRC, Acceso al Modelo.....	101
Tabla 3.15 Tarjeta CRC: Recolección de Datos.....	103
Tabla 3.16 Tarjeta CRC: Entrenamiento de algoritmos.....	104
Tabla 3.17 Caso de prueba: Acceso al sistema.....	107
Tabla 3.18 Caso de prueba: Recolección de datos.....	108
Tabla 3.19 Caso de prueba: Entrenamiento de algoritmos.....	110
Tabla 3.20 Resultados de entrenamiento de algoritmos.....	110
Tabla 3.21 Tarjeta CRC: Predicción del sistema.....	111
Tabla 3.22 Tarjeta CRC: generación de predicción.....	113

Tabla3.23 Caso de prueba: Predicción del Sistema	114
Tabla 3.24 Datos normalizados de casos de CACU	118
Tabla 3.25 Ponderación de la funcionalidad	126
Tabla 3.26 Ponderación de métricas internas usabilidad	127
Tabla 3.27 Totales de métricas internas usabilidad	128
Tabla 3.28 Evaluación de desempeño	128
Tabla 3.29 Análisis global de calidad	130
Tabla 3.30 Estimación de interfaces a ser desarrolladas	131
Tabla 3.31 Puntos de función del sistema	132
Tabla 3.32 Cálculo de ajuste de complejidad.....	133
Tabla 3.33 Factor LCD/PF de lenguaje de programación	134
Tabla 3.34 Tipo de proyecto de software	134
Tabla 3.35 Costo de elaboración de Prototipo	136
Tabla 3.36 Costo total del prototipo.	136
Tabla 4.1 T – Student para el punto crítico	141

CAPITULO I

MARCO PRELIMINAR

MARCO PRELIMINAR

En este capítulo se hace referencia a los antecedentes internacionales, nacionales y locales. Por otro lado, se describe el problema general, los problemas específicos, la delimitación de la hipótesis, variables, justificaciones (económica, social y científica) así como la demarcación del método de investigación que se utiliza en la presente investigación. Así mismo detalla la metodología y herramientas de ingeniería utilizados como: XP, ISO 9126, COCOMO, JAVA, Bases de datos MySql. Finalmente se conocemos los alcances, límites y aportes del presente trabajo de investigación.

1.1. Introducción

El avance de la tecnología y por sobre todo el mundo de la informática, dentro de la Inteligencia Artificial (IA) que fue aportando en los casos de predicción, mediante un modelo lógico en la forma en que los seres humanos identifican estructuras, infieren resultados de problemas difíciles. (Yujra Challco, 2017)

Por esta razón los sistemas expertos como los algoritmos genéticos, es un área que se encarga de estudiar sistemas basados en datos estadísticos y aplicados a números campos como son la medicina, estrategias militares, ingeniería, desastres naturales, derecho y otros, también se puede decir que tratan de imitar el razonamiento de un experto para resolver un problema de un tópico definido. Su comportamiento se basa en recopilar datos estadísticos mediante los cuales,

son capaces de predecir resultados a mediano, largo y corto plazo. (García Serrano, 2012)

Considerando que la idea básica de estos programas es capturar en una computadora la experiencia y el conocimiento de una persona experta en un área determinada del saber, del modo tal que otra persona llamada usuario pueda utilizarla y aprovechar esa información.

Estos sistemas son desarrollados con diferentes objetivos y muy variado propósito, entre algunas tareas que realiza tenemos el diagnóstico, predicciones, monitoreo, selección, planificación, control y para la toma de decisiones o como herramientas de entrenamiento y enseñanza, según la situación que se planteen. (García Serrano, 2012) p.7

Para el caso de estudio es necesario puntualizar que el cáncer es una de las causas principales de muerte puede afectar a todas las edades, inclusive a fetos, pero el riesgo de sufrir los más comunes se incrementa con la edad, el cáncer causa cerca del 13% de todas las muertes. (Programas Nacionales de Control del Cáncer. OMS 2004)

De acuerdo con la Sociedad Americana del Cáncer, 7.6 millones de personas murieron de cáncer en el mundo durante 2007, el cáncer es un conjunto de enfermedades en las cuales el organismo produce un exceso de células malignas (conocidas como cancerígenas o cancerosas), con crecimiento y división más allá de los límites normales.

En la actualidad en Bolivia, la incidencia de cáncer es de 381 por cada 100 mil mujeres. “Es decir que 381 mujeres por cada 100 mil desarrollarían cáncer de forma anual”, dijo. En caso de los varones, la incidencia es de 185 por cada 100 mil hombres. Bolivia: casos de cáncer de cuello uterino se dan desde los 25 años. (4 de febrero de 2019). Los Tiempos, p. 2

Los tipos de cáncer que más afectan a las mujeres en Bolivia son: el de cuello uterino, el de mama, el de vesícula biliar, el de estómago y el de ovario. En los varones, los tipos de cáncer que más los afectan son: el de próstata, estómago, pulmón, vesícula biliar y los linfomas. Particularmente el cáncer de cuello uterino, hace unos cinco años atrás se presentaba a partir de los 35 o 40 años, pero ahora se ve la enfermedad en mujeres más jóvenes. Bolivia: casos de cáncer de cuello uterino se dan desde los 25 años. (4 de febrero de 2019). Los Tiempos, p. 2

Por lo tanto, la Tesis de investigación plantea predecir el incide de crecimiento de la enfermedad del cáncer de cuello uterino utilizando como herramienta el modelo de predicción en base a algoritmos genéticos, capaz de hacer la predicción en 5 años es decir al 2025.

1.2. Antecedentes

El Cáncer, es un crecimiento tisular patológico originado por una proliferación continua de células anormales que produce una enfermedad por su capacidad para elaborar sustancias con actividad biológica nociva, por su capacidad de expansión local o por su potencial de invasión y destrucción de los tejidos adyacentes o a distancia. El cáncer, que puede originarse a partir de cualquier tipo de célula de los diferentes tejidos del organismo, no es una enfermedad única sino un conjunto de enfermedades con manifestaciones y curso clínico muy variables en función de la localización y de la célula de origen

1.2.1. Antecedentes Internacionales.

(Olga Viñas Aparicio, 2012) “**Red neuronal artificial como modelo predictivo en una unidad de patología cervical**”. Ampliar el conocimiento teórico en relación con la utilización de las redes neuronales y los algoritmos genéticos como modelos de proyección y optimización, con el fin de evaluar su aplicación práctica en la estimación del número de pacientes con cáncer hospitalizados en

la CCSS y en la estimación de la estancia hospitalaria esperada para este tipo de pacientes. Universidad de Valladolid, España.

(Vilches Aguirre, 2017) “**Desarrollo de un modelo predictivo de supervivencia en pacientes con cáncer avanzado**”. Determinar cuáles son los factores predictivos de supervivencia en una cohorte de pacientes con cáncer avanzado ingresados en una Unidad de Cuidados Paliativos de un Hospital de tercer nivel y construir un modelo pronóstico. Universidad Autónoma de Madrid, España.

1.2.2. Antecedentes Nacionales

(Yujra Challco, 2017) “**Sistema Experto para el Diagnóstico de Cáncer de Próstata con Lógica Difusa**”. El objeto de estudio es el Cáncer de próstata es verificar las causas del crecimiento anormal de la glándula prostática que también tiene la capacidad de invadir a otros órganos.

Para ello se creará un Sistema Experto el cual tendrá como finalidad determinar según una serie de síntomas el tipo de Cáncer de Próstata que el paciente pudiese llegar a presentar y el tratamiento que éste debería seguir. Universidad Mayor de San Andrés (UMSA), Bolivia.

(Lic. Sandro Saúl Camacho Centellas, 2016). “**Método Heurístico para el Diagnóstico de Cáncer de Mama Basado en Minería de Datos**”. Elaborar un método heurístico que permita la selección de patrones obtenidos a través de la minería de datos para facilitar el diagnóstico oportuno del cáncer de mama.

El enfoque que se dará es cuantitativo porque se trabajará con datos estadísticos (patrones) obtenidos mediante la aplicación de la minería de datos y se buscará probar una hipótesis que plantea que mediante un método heurístico orientado a la selección de patrones obtenidos de mayor interés se podrá detectar y diagnosticar el cáncer de mama con mayor rapidez y precisión. Universidad Mayor de San Andrés (UMSA), Bolivia.

1.2.3. Antecedentes Locales

(Quisberth Ibañez, 2012)“**Modelo de Algoritmos Genéticos para Maximizar Nutrientes y Minimizar Costos**”, Desarrollar un Modelo Basado en Algoritmos Genéticos, que sea Capaz de Disminuir las Deficiencias Nutricionales, Generando Dietas que Maximicen la Cantidad de Nutrientes y Minimicen los Costos Económicos, así Contribuir a la Salud Integral de la Población la Ciudad de El Alto”. Para poder aplicar la ingeniería de software se debe tomar en cuenta: Metodología de desarrollo de software, un ciclo de desarrollo de software y la métrica de calidad de software. Universidad Mayor de San Andrés (UMSA), Bolivia.

1.3. Planteamiento del Problema

Cáncer. La palabra todavía evoca temores profundos acerca de un asesino silencioso que se arrastra hacia nosotros sin ser advertido. El cáncer despierta tanta desesperación que se ha convertido en una metáfora de la aflicción y el dolor, un flagelo que consume nuestros recursos intelectuales y emocionales. Las estadísticas alcanzan una magnitud tal que cada uno de nosotros será afectado, ya sea como paciente, miembro de la familia o amigo. Hay más de 20 millones de personas que padecen cáncer en la actualidad y la mayoría vive en el mundo en desarrollo. No obstante, se puede hacer mucho en cada país para prevenir, curar y aliviar este sufrimiento. Con los conocimientos existentes es posible prevenir al menos un tercio de los 10 millones de casos de cáncer que ocurren anualmente en todo el mundo. Cuando se dispone de recursos suficiente, los conocimientos actuales también permiten la detección temprana y el tratamiento eficaz de otro tercio de esos casos. El alivio del dolor y los cuidados paliativos pueden mejorar la calidad de vida de los pacientes con cáncer y sus familias, aun en los entornos de escasos recursos, gracias a los enfoques eficaces y de bajo costo.

1.3.1. Problema General

La principal problemática que presenta la Ciudad de El Alto, es el no contar con un modelo de predicción del índice de crecimiento de Cáncer de Cuello Uterino, para que la sociedad en general pueda detectar en su fase inicial y prevenir esta mortal enfermedad. El Cáncer que no toma en cuenta la edad, situación social, económica o grado de formación.

1.3.2. Problemas Específicos

Entre los problemas específicos tenemos:

- ✓ Falta de una aplicación de agentes inteligentes en el área de predicción.
- ✓ Falta de una base de datos de datos estandarizados sobre la enfermedad del cáncer de cuello uterino.
- ✓ No se tiene el índice de crecimiento del cáncer de cuello uterino de la Ciudad de El Alto.

¿Cómo predecir el Índice de Crecimiento del Cáncer de Cuello Uterino en la Ciudad de El Alto?

1.4. Objetivos

1.4.1. Objetivo General

Desarrollar un Modelo de predicción en base a Algoritmos Genéticos aplicado al Índice de Crecimiento del Cáncer de Cuello Uterino en la Ciudad de El Alto.

1.4.2. Objetivos Específicos

- ✓ Realizar en análisis y estudio de la situación actual.
- ✓ Recopilar información relevante con respecto al cáncer de cuello uterino.

- ✓ Obtener las estadísticas de pacientes atendidos con cáncer de cuello uterino en la Ciudad de El Alto.
- ✓ Sistematizar una base de datos estandarizados de la enfermedad del cáncer de cuello uterino.
- ✓ Obtener el conocimiento del experto humano mediante entrevista con el profesional del área.
- ✓ Buscar y aplicar algoritmos genéticos para la predicción.
- ✓ Programar una aplicación de agentes inteligentes en el área de predicción.
- ✓ Sistematizar el cuadro de índice de crecimiento del cáncer de cuello uterino de la Ciudad de El Alto.
- ✓ Implementar el prototipo del modelo de predicción del índice de crecimiento del cáncer de cuello uterino.
- ✓ Describir e interpretar los datos obtenidos.

1.5. Hipótesis

La construcción del modelo de predicción del índice de crecimiento del cáncer uterino apoyado en técnicas de desarrollo, metodologías y algoritmos genéticos tendrá una certeza del 95%.

1.5.1. Identificación De Variables

Variable Independiente

- ✓ Modelo de predicción.

Variable Dependiente

- ✓ Índice de Crecimiento de Cáncer de Cuello Uterino

Variable Interviniente (Término Relacionador)

- ✓ Algoritmos genéticos

1.5.2. Operaciones De Variables

Tabla 1.1 Operación de Variables

VARIABLE	TIPO DE VARIABLE	DIMENSION	INDICADORES
Modelo de predicción.	VARIABLE INDEPENDIENTE	Interface de usuario. Medio para la administración del conocimiento.	Usabilidad de interfaz. Porcentaje de eficacia,
Índice de crecimiento del cáncer uterino.	VARIABLE DEPENDIENTE	Edad Casos de pacientes atendidos con cáncer uterino	Porcentaje de pacientes por grupos etarios. Porcentaje de pacientes con cáncer uterino
Algoritmos genéticos	VARIABLE INTERVINIENTE	Mutación Cruzamiento	Número de mutaciones. Número de cruzamientos.

Fuente: Elaboración propia

1.5.3. Conceptualización De Variables

1.5.3.1. Índice De Cáncer Uterino

El índice de cáncer uterino, indica que de cuatro a cinco mujeres muere al día por cáncer de cuello uterino (CACU) en Bolivia, una de las más altas tasas de incidencia y mortalidad a nivel mundial y la más alta de América Latina, según la Organización Mundial de la Salud (OMS).

1.5.3.2. Modelo De Predicción

El modelo de predicción, infiere probables consecuencias de situaciones dadas. Algunas veces usan modelos de simulación para generar situaciones que puedan ocurrir. Ejemplos: predecir daños a cosechas por algún tipo de insecto.

1.5.3.3. Algoritmos Genéticos

Los Algoritmos Genéticos son métodos adaptativos, generalmente usados en problemas de búsqueda y optimización de parámetros, basados en la reproducción sexual y en el principio supervivencia del más apto. (Gestal, 2000)

Más formalmente, y siguiendo la definición dada por Goldberg, “Los Algoritmos Genéticos son algoritmos de búsqueda basados en la mecánica de selección natural y de la genética natural. Combinan la supervivencia del más apto entre estructuras de secuencias con un intercambio de información estructurado, aunque aleatorizado, para constituir así un algoritmo de búsqueda que tenga algo de las genialidades de las búsquedas humanas” (Goldberg, 1989).

Para alcanzar la solución a un problema se parte de un conjunto inicial de individuos, llamado población, generado de manera aleatoria. Cada uno de estos individuos representa una posible solución al problema. Estos individuos evolucionarán tomando como base los esquemas propuestos por Darwin (Darwin, 1859) sobre la selección natural, y se adaptarán en mayor medida tras el paso de cada generación a la solución requerida. (Gestal, 2000)

1.6. Justificaciones

1.6.1. Justificación Técnica

El presente trabajo se justifica técnicamente porque se facilita una herramienta tecnológica de apoyo a profesionales y personas, el cual proporciona información adecuada en base a los historiales médicos atendidos, así mismo emplea

técnicas de sistemas expertos en el área de inteligencia artificial aprendidas en la carrera de Ingeniería de Sistemas.

1.6.2. Justificación Económica

El presente trabajo se justifica económicamente porque al informar a las personas sobre el índice de crecimiento, se reducen los gastos de tiempo y costos, ya que cuando el cáncer es detectado en las últimas etapas los tratamientos son costosos y dolorosos. Sin embargo, cuando los pacientes asisten de manera oportuna a los análisis, el cáncer es detectado en sus etapas iniciales inclusive se pueden curar.

Así mismo el costo del modelo de predicción aplicado al índice de crecimiento del cáncer uterino, no tiene costo ya que se utilizará el software libre, dejando de lado el software con licencias costosas.

1.6.3. Justificación Social

El presente trabajo se justifica socialmente porque ayuda de manera sustancial a la población femenil, ya que apoya en el conocimiento de las predicciones en los próximos 5 años y de esta manera acudir de manera oportuna a las consultas, la asistencia y la detección oportuna del cáncer, en sus etapas iniciales para evitar costos innecesarios y tratamientos dolorosos hasta la pérdida de vidas y la destrucción de las familias.

Así mismo de manera indirecta se ven favorecidas las familias, ya que esta enfermedad involucra colateralmente a madres, padres hijos y nietos con el dolor y los gastos.

Por otro lado, son beneficiados los expertos en salud y en especial, los del área de oncología. Al conocer el índice de crecimiento, estos podrán difundir y socializar con los pacientes y toda la comunidad teniendo un instrumento tecnológico a su alcance.

1.6.4. Justificación Científica

Los algoritmos genéticos (AG) funcionan entre el conjunto de soluciones de un problema llamado fenotipo, y el conjunto de individuos de una población natural, codificando la información de cada solución en una cadena. El modelo de predicción en base a los algoritmos genéticos predice oportunamente el índice de crecimiento del cáncer de cuello uterino en la Ciudad de El Alto, que favorece a la población femenina en la oportuna atención y detección de la misma.

1.7. Metodología (Metodologías Ágiles)

1.7.1. Método Científico

El método científico es una serie de mecanismos que usa el ser humano para llegar a comprender un hecho de la naturaleza. Para resolver un problema, se sigue una serie de pasos que generalmente no son lineales ni fáciles. (Bunge, 1960)

Toda investigación supone un proceso, pero un proceso planeado, no errático, en el que se sabe lo que se busca, utilizando métodos, técnicas y procedimientos adecuados a ese fin.

Según Mario Bunge, en su libro Ciencia: su método y filosofía, podemos afirmar que el método científico debe seguir las siguientes etapas o pasos:

Planteamiento del problema

En primer lugar, hay que partir del reconocimiento de los hechos, es decir, aquellos hechos específicos relevantes para lo que se quiere investigar o conocer.

En seguida, se pasa al descubrimiento del problema. En este punto, la curiosidad es clave, no importa cuánto sepamos sobre un hecho, si no tenemos curiosidad en descubrir alguna dificultad o problema.

Después, viene la pregunta que nos va a permitir buscar la solución a aquella dificultad que queremos resolver. A esto se le conoce como formulación del problema.

Construcción de un modelo teórico

Para responder la pregunta, debemos hacer una selección de los factores pertinentes. Tenemos que suponer las respuestas más adecuadas y posibles, de acuerdo a lo que se conoce.

La invención de las hipótesis no es más que proponer suposiciones que puedan explicar los hechos del problema a resolver.

Deducción de consecuencias particulares

Esta etapa consiste en la búsqueda de los soportes teóricos y empíricos ya realizados que nos permitan deducir los mecanismos para probar las hipótesis.

Prueba de las hipótesis

Para probar las hipótesis, hay que diseñar y ejecutar los experimentos, mediciones, recolección de datos y otras operaciones necesarias. Es de gran importancia el diseño y la descripción detallada de los experimentos de manera que otros investigadores puedan repetir y validar los resultados.

Una vez recolectados los datos, estos deben ser analizados, clasificados y evaluados. En esta etapa, toma fuerza la estadística.

Luego de la elaboración de los datos, se realiza la interpretación según el modelo teórico. Este es el proceso de inferencia de la conclusión.

Introducción de las conclusiones en la teoría

Al interpretar los datos, hay que comparar las conclusiones con las predicciones y considerar si el modelo teórico es aceptado o rechazado.

Si el modelo teórico es rechazado, se debe reajustar el modelo, corregirlo o reemplazarlo.

El método científico no acaba cuando se confirma el modelo teórico; hay una continua búsqueda de problemas, consecuencias o errores en la teoría o los procedimientos empíricos. Para eso siempre serán bienvenidas las sugerencias acerca del trabajo anterior.

Comunicación de los hallazgos

Una parte importante del trabajo científico es la comunicación y la transmisión de la información. Esto se hace a través de artículos científicos, tesis, libros, o conferencias en institutos educativos y de investigación.

1.7.2. Método De Ingeniería

1.7.2.1. Metodología XP (Extreme Programming)

Extreme Programming (XP), surge como una nueva manera de encarar proyectos de software, proponiendo una metodología basada esencialmente en la simplicidad y agilidad. Las metodologías de desarrollo de software tradicionales (ciclo de vida en cascada, evolutivo, en espiral, iterativo, etc.) aparecen, comparados con los nuevos métodos propuestos en XP, como pesados y poco eficientes. La crítica más frecuente a estas metodologías “clásicas” es que son demasiado burocráticas. Hay tanto que hacer para seguir la metodología que, a veces, el ritmo entero del desarrollo se retarda. Como respuesta a esto, se ha visto en los últimos tiempos el surgimiento de “Metodologías Ágiles”. Estos nuevos métodos buscan un punto medio entre la ausencia de procesos y el abuso de los mismos, proponiendo un proceso cuyo esfuerzo valga la pena.

Los métodos ágiles cambian significativamente algunos de los énfasis de las metodologías “clásicas” (Fowler, 2005).

Los métodos ágiles son adaptables en lugar de predictivos. Los métodos “clásicos” tienden a intentar planear una gran parte del proceso del software en gran detalle para un plazo largo de tiempo.

Esto funciona bien hasta que las cosas cambian. Así que su naturaleza es resistirse al cambio. Para los métodos ágiles, no obstante, el cambio es bienvenido. Intentan ser procesos que se adaptan y crecen en el cambio.

Los métodos ágiles son orientados a la gente y no orientados al proceso. El objetivo de los métodos “clásicos” es definir un proceso que funcionará bien independientemente de quien lo utilice.

Los métodos ágiles afirman que ningún proceso podrá nunca maquillar las habilidades del equipo de desarrollo, de modo que el papel del proceso es apoyar al equipo de desarrollo en su trabajo. XP es una de las llamadas metodologías ágiles de desarrollo de software más exitosas de los tiempos recientes. La metodología propuesta en XP está diseñada para entregar el software que los clientes necesitan en el momento en que lo necesitan. XP alienta a los desarrolladores a responder a los requerimientos cambiantes de los clientes, aún en fases tardías del ciclo de vida del desarrollo. (JoKowicz, 2008)

La metodología también enfatiza el trabajo en equipo. Tanto gerentes como clientes y desarrolladores son partes del mismo equipo dedicado a entregar software de calidad. XP fue introducida como metodología ágil de desarrollo de software sobre finales de los 1990s. Uno de los conocidos “caso de éxito” fue publicado a fines de 1998, cuando Kent Beck, introdujo la nueva metodología en el proyecto de desarrollo denominado C3 (Chrysler Comprehensive Compensation) para la firma Chrysler.

Al igual que otras metodologías de gestión de proyectos, tanto Ágiles como tradicionales, el ciclo XP incluye:

- ✓ Entender lo que el cliente necesita > Fase de Exploración
- ✓ Estimar el esfuerzo > Fase de Planificación
- ✓ Crear la solución > Fase de Iteraciones
- ✓ Entregar el producto final al cliente > Fase de puesta en producción

Lo que caracteriza a XP, al igual que al resto de métodos Ágiles es un ciclo de vida dinámico. Ver figura1.1.

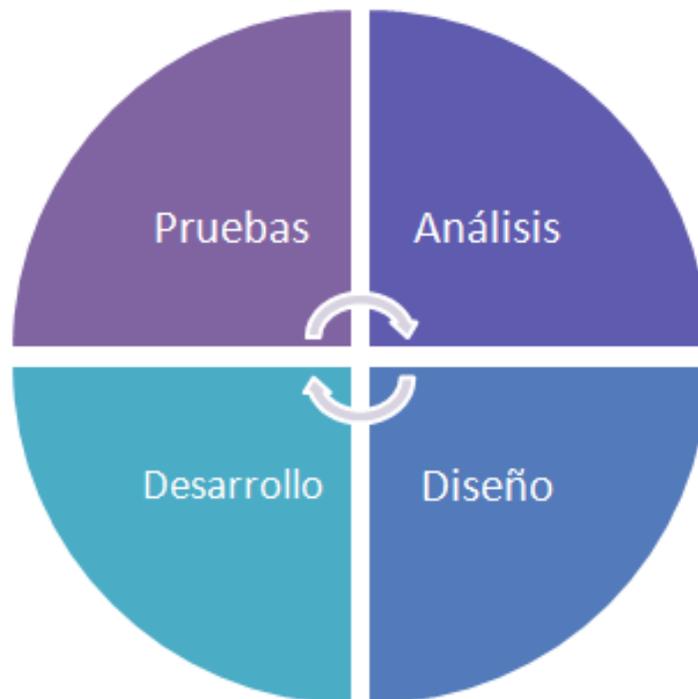


Figura 1.1 Ciclo de vida de XP

Fuente: Bahit, 2012

Estos ciclos de desarrollo cortos (llamados iteraciones), al fin de los cuales se generan unos entregables funcionales.

En cada iteración se realiza un ciclo completo de análisis, diseño, desarrollo y pruebas, pero utilizando un conjunto de reglas y prácticas específicas de XP. Un proyecto con XP, implica de entre a 10 a 15 iteraciones habitualmente.

1.7.2.2. ISO 9126 (Métrica de calidad)

El estándar ISO 9126 ha sido desarrollado para identificar los atributos clave de calidad para el software. El estándar identifica 6 atributos clave de calidad: (Pressman, 2010)

Funcionalidad: El grado en que el software satisface las necesidades indicadas por los siguientes sub características: idoneidad, corrección, interoperabilidad, conformidad y seguridad.

Confiabilidad: Cantidad de tiempo que el software está disponible para su uso. Está referido por los siguientes sub características: madurez, tolerancia a fallos y facilidad de recuperación.

Usabilidad: Grado en que el software es fácil de usar. Viene reflejado por los siguientes sub características: facilidad de comprensión, facilidad de aprendizaje y operatividad.

Eficiencia: Grado en que el software hace óptimo el uso de los recursos del sistema. Está indicado por los siguientes sub características: tiempo de uso y recursos utilizados.

Facilidad de mantenimiento: la facilidad con que una modificación puede ser realizada. Está indicada por los siguientes sub características: facilidad de análisis, facilidad de cambio, estabilidad y facilidad de prueba.

Portabilidad: La facilidad con que el software puede ser llevado de un entorno a otro. Está referido por los siguientes sub características: facilidad de instalación, facilidad de ajuste, facilidad de adaptación al cambio.



Figura 1.2 Factores ISO 9126

Fuente: Banda Hugo 2014

1.7.2.3. Costos (COCOMO II)

Como se conoce, una de las tareas de mayor importancia en la planificación de proyectos de software es la estimación, la cual consiste en determinar, con cierto grado de certeza, los recursos de hardware y software, costo, tiempo y esfuerzo necesarios para el desarrollo de los mismos.

COCOMO II. Este modelo permite realizar estimaciones en función del tamaño del software, y de un conjunto de factores de costo y de escala. Los factores de costo describen aspectos relacionados con la naturaleza del producto, hardware utilizado, personal involucrado, y características propias del proyecto. El conjunto de factores de escala explica las economías y deseconomías de escala producidas a medida que un proyecto de software incrementa su tamaño.

COCOMO II. Posee tres modelos denominados, Composición de Aplicación, Diseño Temprano y Post-Arquitectura. Cada uno de ellos orientados

a sectores específicos del mercado de desarrollo del software y a las distintas etapas del software. (Gomez, C.López, Migani, & Otazú, 2010)

1.8. Herramientas

Las herramientas que nos permitirán el desarrollo del presente trabajo de investigación, de acuerdo a los requerimientos son:

1.8.1. Lenguajes de Programación

JAVA. Un lenguaje de programación de propósito general, orientado a objetos, cuyo espíritu se resume en las siglas WORA: Written Once, Run Anywhere, es decir: Escrito una vez, funciona en cualquier parte. La idea era diseñar un lenguaje universal empleando sintaxis derivada de los lenguajes C y C++, pero empleando menos utilidades de bajo nivel que cualquiera de ambos.

PHP. Es “transparente”, invisible. Por esta razón, es difícil explicar qué es y cuál su funcionamiento. PHP es una sigla, un acrónimo de “PHP: Hypertext Preprocessor”, o sea, “Pre - procesador de Hipertexto marca PHP”. El hecho de que sea un “pre” procesador es lo que marca la diferencia entre el proceso que sufren las páginas Web programadas en PHP del de aquellas páginas Web comunes, escritas sólo en lenguaje HTML. Para llegar a entender qué es un pre - procesador, examinaremos primero cuál es la diferencia entre el proceso de una página Web normal (HTML) y el “pre” proceso de una página escrita en lenguaje PHP. (Beáti, 2012)p.2

En resumen, el pre - proceso de páginas PHP consiste en esta serie de pasos, destinadas al software intérprete de PHP (órdenes que casi siempre consisten en que el software de PHP obtenga cierta información,luego, colocamos otras órdenes para que el software intérprete de PHP “realice algo” con esa información, típicamente, que la escriba dentro del código fuente de la página HTML que se enviará al navegador del usuario. (Beáti, 2012) p.7

MATLAB

Es una de las muchas sofisticadas herramientas de computación disponibles en el comercio para resolver problemas de matemáticas, tales como Maple, Mathematica y MathCad. A pesar de lo que afirman sus defensores, ninguna de ellas es “la mejor”. Todas tienen fortalezas y debilidades. Cada una permitirá efectuar cálculos matemáticos básicos, pero difieren en el modo como manejan los cálculos simbólicos y procesos matemáticos más complicados, como la manipulación de matrices. Por ejemplo, MATLAB es superior en los cálculos que involucran matrices, mientras que Maple lo supera en los cálculos simbólicos. El nombre mismo de MATLAB es una abreviatura de Matrix Laboratory, laboratorio matricial.

En un nivel fundamental, se puede pensar que estos programas son sofisticadas calculadoras con base en una computadora. Son capaces de realizar las mismas funciones que una calculadora científica, y muchas más. Si usted tiene una computadora en su escritorio, descubrirá que usará MATLAB en lugar de su calculadora incluso para la más simple de sus aplicaciones matemáticas, por ejemplo, para el balance de su chequera. En muchas clases de ingeniería, la realización de cálculos con un programa de computación matemático como MATLAB sustituye la programación de computadoras más tradicional. Esto no significa que el lector no deba aprender un lenguaje de alto nivel como C++ o FORTRAN, sino que los programas como MATLAB se han convertido en una herramienta estándar para ingenieros y científicos.

1.8.2. Base de Datos

Base de datos MySQL, es un sistema para la gestión de base de datos con más de seis millones de instalaciones en el mundo, y que por lo tanto es el más extendido dentro de las aplicaciones relacionadas. MySQL funciona como software libre dentro del esquema de licencias GNU GPL.

MySQL es muy utilizado en aplicaciones web como WordPress, PhpBB, MediaWikio Drupal, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura, lo que hace a MySQL ideal para este tipo de aplicaciones.

Todos los planes de hosting de Hostinet disponen de MySQL para poder crear y gestionar el contenido en bases de datos, como Mat Lab, Magic Draw y UWE.

1.9. Límites y Alcances

1.9.1. Límites

El presente trabajo abarca el desarrollo de un modelo de predicción del índice de crecimiento del cáncer de cuello uterino en la Ciudad de El Alto, en base a algoritmos genéticos y de acuerdo a los reportes de pacientes atendidos en los centros de salud tanto privados y estatales, que son registrados en SEDES El Alto.

El modelo de predicción no cuenta con una interface a la base de datos de los centros de salud, hospitales y SEDES El Alto.

El modelo de predicción no cuenta con datos generales de todo es Estado Plurinacional de Bolivia.

1.9.2. Alcances

El modelo de predicción tendrá una recopilación de datos sobre los casos de cáncer de cuello uterino, aprendizaje continuo, procesamiento de la información e interpretación de los resultados. Que plantea las siguientes etapas para su aplicación, como inserción de datos, generación de la población, selección de los mejores genes, mutación, cruce y la creación de la nueva población del índice de crecimiento de cáncer de cuello uterino.

El periodo establecido del modelo de predicción está fijado para 5 años es decir del 2020 al 2025, siendo que después de este tiempo se deben actualizar los datos y los nuevos alcances.

El área de conocimiento, está dirigido a la Inteligencia Artificial, redes neuronales, algoritmos genéticos y sistemas expertos.

1.10. Aportes

Loa aportes de la presente investigación se resumen en los siguientes puntos:

En cuanto al trabajo de investigación un aporte principal es de desarrollar el modelo de predicción al índice de crecimiento del cáncer de cuello uterino, que será útil para los especialistas, así como para la población en general.

El modelo de predicción es un modelo de sistema de aprendizaje para la predicción de otras enfermedades cancerígenas en la Ciudad de El Alto.

Se proporciona una nueva herramienta tecnológica al sector de salud para poder predecir el índice de crecimiento del cáncer uterino en otros departamentos del Estado Plurinacional de Bolivia.

CAPITULO II

MARCO TEÓRICO

MARCO TEÓRICO

En este capítulo se define los conceptos de Cáncer, Cáncer de cuello uterino, el VRION, el genoma del VPH, Inteligencia Artificial, algoritmos genéticos, cruce, mutación y generación. Se describe el método científico según Mario Bunge, etapas del método científico, la metodología XP Xtreme Programming (Fases y Reglas), la métrica ISO 9126 y COCOMO II con las etapas correspondiente como métricas de calidad, el estudio e importancia de JAVA y Matlab para el desarrollo del presente trabajo de investigación.

2.1. Introducción

En este acápite se definirán los conceptos básicos que se requiere para el desarrollo e interpretación del modelo de predicción en base a algoritmos genéticos, tales como: Cáncer, Cáncer de cuello uterino, algoritmos genéticos y el método científico. De la misma manera, dentro de la metodología de Software se describirán el modelo XP, la métrica de calidad ISO 9126, COCOMO II así como lenguajes de programación: JAVA, MATLAB, MySQL como herramientas del presente modelo de predicción CACUGEN.

2.2. El Cáncer

Cáncer es el término genérico para designar a un grupo de enfermedades que pueden afectar a cualquier parte del cuerpo. Los tumores malignos del cerebro,

del pulmón, de la mama, la próstata, la piel y el colon son algunas de las enfermedades conocidas como cáncer. Otros ejemplos de cáncer incluyen la leucemia, los sarcomas, la enfermedad de Hodgkin y el linfoma no Hodgkin. Ciertas características morfológicas comunes diferencian todas las formas de cáncer de otros tipos de estados patológicos, incluidas otras enfermedades no transmisibles y trastornos causados por agentes tóxicos. No obstante, hay conexiones entre el cáncer y estas otras enfermedades. En particular, muchas enfermedades no transmisibles comparten con el cáncer factores causales, como el consumo de tabaco, el régimen alimentario insalubre, la obesidad y la falta de ejercicio físico. Por consiguiente, los enfoques de la prevención a menudo son idénticos. (OMS, 2011).

Los distintos tipos de cáncer tienen varias características biológicas comunes. Una característica definitoria es la proliferación de células anormales. El proceso de reproducción y reemplazo de células normalmente está bien controlado a lo largo de la vida por mecanismos biológicos básicos. Sin embargo, en el caso del cáncer fracasan los mecanismos de control. Las células en la parte afectada del cuerpo crecen más allá de sus límites usuales, invaden los tejidos colindantes y pueden propagarse a otros órganos o tejidos más distantes como metástasis.

Los adelantos en la biología molecular han incrementado los conocimientos de los investigadores acerca del mecanismo de crecimiento de las células normales, lo cual permite investigar la proliferación de células aberrantes y el fracaso de la muerte de células (apoptosis) que constituyen el cáncer. El crecimiento celular es controlado por una clase de genes conocidos como protooncogenes o genes supresores. Si se produce una mutación o translocación de genes dentro de un cromosoma, un protooncogén puede perder su capacidad de regular la replicación de células y convertirse en un oncogén. Esos cambios, desencadenados por una variedad de factores, podrían ser la vía común final en el mecanismo biológico del cáncer. (OMS, 2011)

Un tumor maligno se origina a partir de una célula alterada y prolifera inicialmente en el sitio primario. Después suele propagarse por diversas vías, como infiltración local en la vecindad del órgano de origen, el sistema linfático (los ganglios linfáticos) o los canales vasculares, y lleva a la metástasis, principal causa de muerte por enfermedades malignas.

2.2.1. Fases del Cáncer

El desarrollo del cáncer tiene varias fases, según el tipo de tejido afectado. En general estas fases son: la displasia, el cáncer in situ, el cáncer invasivo localizado, el compromiso de ganglios linfáticos regionales y las metástasis distantes (véase la figura 2.1). La primera indicación de anormalidad es un cambio en las características de las células, conocido como displasia. La lesión puede retroceder espontáneamente en este estadio y a veces aun en el siguiente, en el del carcinoma in situ (como indican las flechas en ambas direcciones). El término “carcinoma in situ” se usa cuando el examen microscópico revela células con ciertas características del cáncer, esto es, cambios en los núcleos celulares, pero sin penetración de la membrana subyacente (o base) que los sostiene en el tejido de origen.

El término carcinoma in situ se reserva generalmente para los cambios que afectan al espesor total del epitelio. Cuando el crecimiento celular anormal llega a áreas que están por debajo del tejido de origen, se considera que el cáncer es invasivo. Al proseguir el crecimiento, se produce una invasión cada vez mayor y la destrucción de los tejidos adyacentes.

A menudo el cáncer se extiende a los ganglios linfáticos regionales que drenan el área. Las células cancerosas también pueden propagarse a través de la sangre o del sistema linfático y afectar a otros órganos (metástasis distantes).

Por ejemplo, el cáncer de colon puede propagarse al hígado o los pulmones.

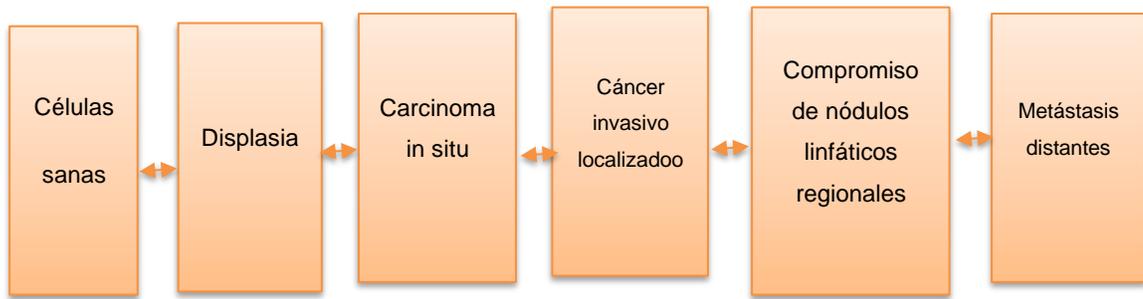


Figura 2.1 Etapas típicas del desarrollo del Cáncer.

Fuente: Concha, 2007.

Cuando se llega a una multiplicación suficiente de las células anormales, el cáncer se hace patente para el individuo o el médico. Adopta comúnmente la forma de una masa abultada que puede verse o sentirse en el órgano afectado, por ejemplo, la piel, la mama o la próstata. A veces, aun antes de la detección, el cáncer se habrá propagado a los ganglios linfáticos o, si es rápidamente progresivo, habrá causado metástasis distantes detectables. El crecimiento del cáncer puede afectar los vasos sanguíneos y provocar hemorragias que serán evidentes si el tumor llega a una parte de un órgano en contacto directo o indirecto con el exterior. Por ejemplo, puede haber sangre en el esputo resultante del cáncer del pulmón, sangre en las heces provocada por el cáncer intestinal o sangre en la orina a causa del cáncer de vejiga. El crecimiento del tumor maligno también puede dar origen a trastornos funcionales; por ejemplo, el cáncer de cerebro puede provocar síntomas y signos neurológicos. En el cáncer avanzado, uno de los síntomas más graves es el dolor inducido por la presión sobre los nervios.

2.2.2. Cáncer de Cuello Uterino (VPH)

Desde la década del 70 se ha considerado directamente al VPH como el agente causal del Cáncer de Cuello Uterino considerado en la actualidad el factor de mayor riesgo para el desarrollo de lesiones preinvasivas o invasivas de cérvix y otros en región genital femenina y masculina, así como también en región anal,

debido al conocimiento de su potencial oncogénico y la asociación con tumores humanos. Este virus despierta una creciente importancia en Salud Pública y en la vigilancia epidemiológica, por cuanto ha aumentado su frecuencia, sobre todo en la población sexualmente activa con inicio temprano de relaciones sexuales.

Taxonómicamente el VPH pertenece a la familia Papovaviridae, incluida en el género Papilomavirus, se caracteriza por ser causante de lesiones mucosas y cutáneas en humanos y primates, infecta a la mayoría de los mamíferos y aves, existiendo una posible excepción con ratones de laboratorio.

Los VPH que tienen afinidad por las células de la mucosa anogenital, son agentes de transmisión sexual más comunes en hombres y mujeres a nivel mundial. Su persistencia es considerada como el principal factor de riesgo para el desarrollo del Cáncer de Cuello Uterino. No existen datos precisos de la prevalencia o incidencia debido a la existencia de individuos infectados que no presentan signos ni síntomas de la infección por VPH. (Concha, 2007)

2.2.3. Estructura del VRION

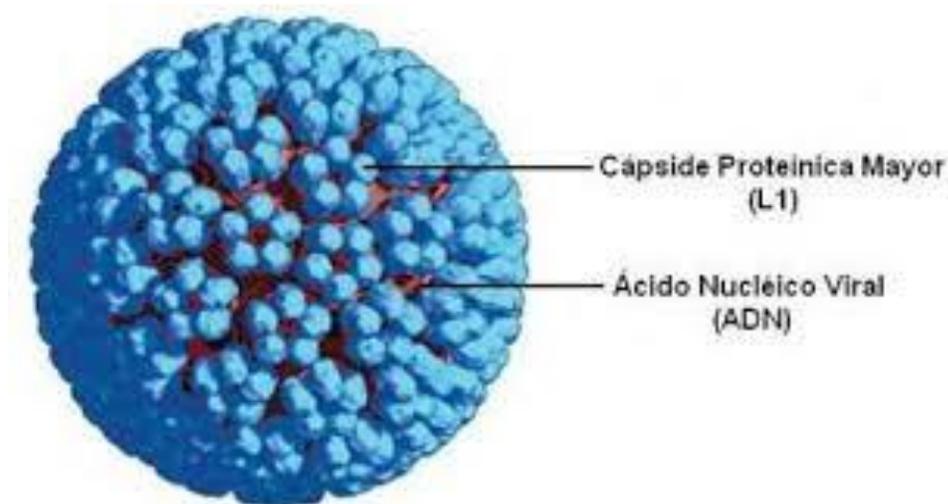


Figura 2.2 Estructura del Virus del Papiloma Humano

Fuente: Concha, 2007.

Son virus pequeños y sin envoltura, con un diámetro de 52-55 nm la cápside viral es icosaédrica, organizada por 72 capsómeros, (60 hexámeros y 12 pentámeros). Cada uno de estos capsómeros está constituido por dos proteínas estructurales codificadas por el virus, que se encuentran unidas por puentes de disulfuro, generando estabilidad de la cápside, La proteína mayor L1, responsable de la adherencia específica del virus a la célula, representa el 80% del total de la Cápside y la proteína menor L2, en menor proporción que L1 y es contra estas proteínas que se dirige la respuesta inmune del huésped. (Dermatología, 2004)

2.3. Inteligencia Artificial

La inteligencia artificial (IA) es una disciplina muy amplia que comprende tópicos que van desde la filosofía hasta la ciencia y la tecnología. (Banda Gamboa, 2014)

Después de transcurrido más de medio siglo desde la Conferencia de Dartmouth en 1956, fecha mayoritariamente reconocida como el inicio de la inteligencia artificial, ésta se ha constituido en una disciplina sumamente amplia. Contiene muchos y diversos sub campos y tiene fuertes vínculos con áreas como la ciencia cognitiva y la filosofía de la mente.

La investigación de inteligencia artificial abarca una amplia gama de temas. Incluye la mejora de computadoras en tareas que son ampliamente percibidas como inteligentes, como probar teoremas matemáticos. También investiga procesos que recientemente se han reconocido tienen dificultad importante, como el reconocimiento de objetos en imágenes. Investiga problemas que no se habían considerado antes de la invención de las computadoras, tales como procesar grandes cantidades de datos y encontrar información en la World Wide Web.

El campo de la Inteligencia Artificial no sólo se preocupa por replicar la inteligencia humana. Sistemas de inteligencia artificial, tales como programas de

juego de ajedrez o que incluyen procesos de decisión, no se comportan necesariamente como lo harían los humanos, sin embargo, han demostrado habilidad para resolver problemas complejos. Hay todo un campo independiente, la ciencia cognitiva, que se dedica a comprender cómo los seres humanos (y otros animales) piensan. Hay muchos vínculos fértiles entre los dos campos. En muchos casos una mejor comprensión de la cognición humana puede llevar a avances en Inteligencia Artificial. Al mismo tiempo encontrar una forma para que un computador pueda realizar una tarea, puede arrojar luz sobre la forma en la que los seres humanos podrían pensar.

Por otra parte, no es sorprendente que la Inteligencia artificial, sea la inspiración para mucha ficción especulativa. A menudo se trata de historias en las que computadoras o robots se comportan como versiones especialmente inteligentes y físicamente más fuertes que los seres humanos. Hay una serie de problemas prácticos y filosóficos detrás de estas ideas. Todavía no es totalmente claro si la naturaleza de la investigación de la Inteligencia Artificial podrá producir este tipo de robot inteligente y malévolo.

Los campos de la Inteligencia Artificial y las Ciencias Cognitivas son importantes componentes de las Ciencias de la Computación y la Psicología. El desafío de la Inteligencia Artificial incluye desarrollar en las máquinas la capacidad de percibir, aprender, almacenar información, razonar acerca de lo que conocen, comunicarse utilizando lenguaje humano e interactuar con el entorno físico.

Todas estas habilidades todavía están siendo estudiadas por los investigadores y ninguna puede considerarse resuelta. Esto hace que la Inteligencia Artificial y las Ciencias Cognitivas sean importantes campos de estudio.

La inteligencia Artificial es una parte central de uno de los más profundos avances científicos e intelectuales de los últimos 60 años: el estudio de la información y del conocimiento; y, cómo estos pueden ser adquiridos, almacenados, entendidos, manipulados, usados y transmitidos. La investigación ha permitido

desarrollar una nueva ciencia para el estudio de los principios por los cuales: el conocimiento es adquirido y utilizado, metas son generadas y logradas, información es comunicada, colaboración es alcanzada, conceptos formados, y el lenguaje es utilizado. A ésta se la puede llamar la ciencia del conocimiento o la ciencia de la inteligencia. Igualmente, la creciente necesidad de desarrollar aplicaciones capaces de resolver problemas complejos, ejecutando procesos que se pueden llamar inteligentes, ha dado lugar al apareamiento de la ingeniería de sistemas inteligentes.

Acorde con estas consideraciones, en términos simples, se puede decir que:

La Inteligencia Artificial comprende la investigación científica y tecnológica de los sistemas inteligentes.

Se denomina sistema inteligente a una entidad capaz de percibir, razonar, aprender, adaptarse, tomar decisiones y actuar racionalmente para satisfacer sus metas, en un determinado entorno.

Las entidades pueden ser: máquinas, humanos u otros animales

Dada su complejidad, el desarrollo de la investigación científica y tecnológica de los sistemas inteligentes, requiere del apoyo de otras áreas del conocimiento, tales como: Filosofía, Psicología, Lingüística, Ciencias de Computación, Biología, Neurociencias, Matemática, Física, Química, Cibernética, Electrónica y Comunicaciones.

Desde la perspectiva filosófica, muchos teóricos de IA creen que ciertos procesos computacionales son necesarios y suficientes para la inteligencia. John Searle llama esta creencia Inteligencia Artificial Fuerte y sostiene que es fundamentalmente equivocada. La inteligencia artificial fuerte plantea que los procesos realizados por una computadora son idénticos a los que realiza el cerebro, y por lo tanto se puede deducir que, si el cerebro genera conciencia, también las computadoras deben ser conscientes. En su opinión, las aplicaciones

de IA tratan únicamente con la sintaxis, no con la semántica, por lo que un computador no puede explicar la intencionalidad o significado de lo que está procesando. La IA fuerte actualmente cae dentro de la ciencia ficción y la especulación futurista. El campo de la robótica es uno de los que más ha impulsado el estudio de la inteligencia general (Inteligencia Artificial Fuerte). Si bien se han logrado importantes avances, también se han evidenciado las dificultades de tratar de generar modelos de procesos mentales que todavía los seres humanos no han logrado entender a un nivel suficiente como para poder describirlos, representarlos y modelarlos.

Desde la perspectiva tecnológica, en cambio los sistemas inteligentes no necesariamente deben emular los procesos característicos de los seres inteligentes. A esta premisa se la denomina Inteligencia Artificial Débil.

En la siguiente figura se presenta un mapa mental de los principales tópicos que, desde esta perspectiva, trata actualmente la inteligencia artificial: Ver Figura 2.3.

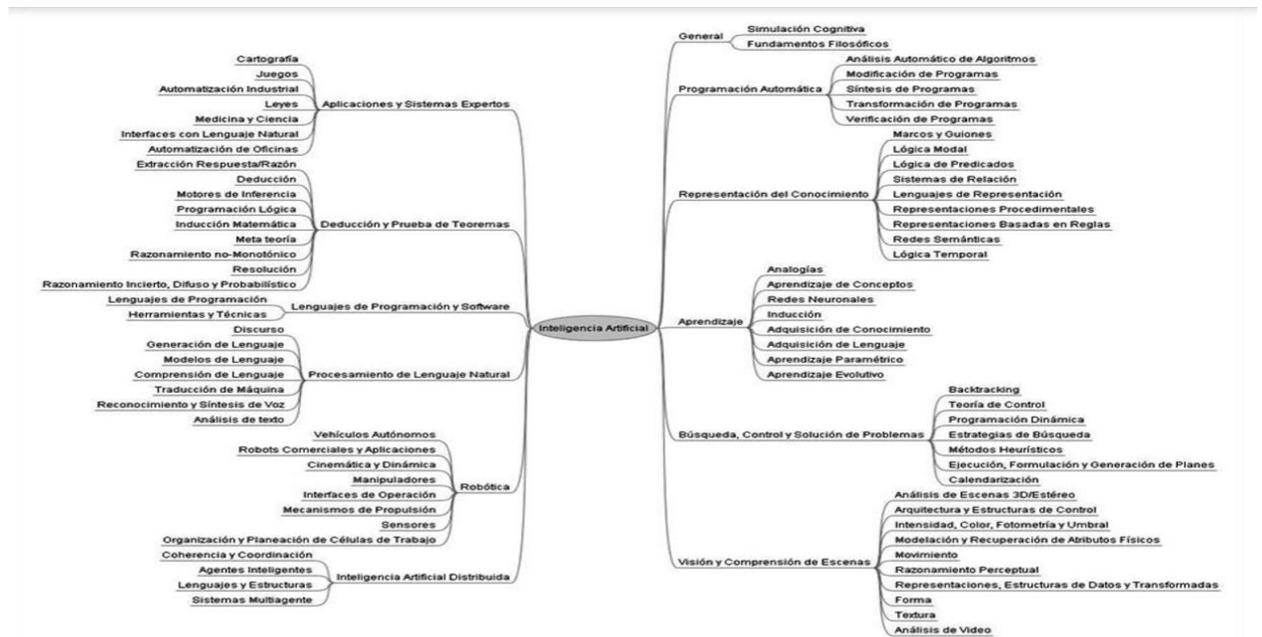


Figura 2.3 Tópicos relacionados con la Inteligencia Artificial

Fuente: Banda (2014)

La actual proliferación y desarrollo de la infraestructura de comunicación basada en las redes como la World Wide Web (WWW), ha creado un escenario sin precedentes y está potenciando la probabilidad de éxito de los sistemas inteligentes para la recopilación masiva de información y su uso en la automatización de diversas funciones empresariales.

2.4. Algoritmos Genéticos

Los algoritmos genéticos están basados en los sistemas biológicos que son el resultado de procesos evolutivos. La robustez y adaptabilidad de los sistemas biológicos son una fuente de inspiración para el desarrollo de sistemas de HW y SW con características comparables.

La evolución natural no tiene una meta predeterminada y es en esencia un proceso de adaptación abierto; sin embargo, la versión artificial es un proceso de optimización que trata de encontrar soluciones a problemas predefinidos.

La teoría de la evolución natural descansa sobre 4 pilares:

- ✓ Población. - La premisa de la evolución es la existencia de una población.
- ✓ Diversidad. - Implica que las características de los individuos varían de una población a otra.
- ✓ Heredad. - Indica que ciertas características individuales de los padres, pueden transmitirse a los descendientes a través de la reproducción.
- ✓ Selección. - Establece que sólo una parte de la población es capaz de reproducirse y transmitir sus características a generaciones futuras.

El avance reciente de la genética y de la genómica funcional, ha aportado claves a los mecanismos moleculares y procesos que intervienen en la heredad y la diversidad.

Las especies se crean, evolucionan y desaparecen si no se adaptan de forma que solo los que mejor se adapten al medio sobreviven para perpetuar sus

aptitudes. De acuerdo con esta visión de la evolución, la computación ve en este marco un claro proceso de optimización: se toman los individuos mejor adaptados, se cruzan, generando nuevos individuos que contendrán parte del código genético de sus antecesores, y el promedio de adaptación de toda la población se mejora.

La computación evolutiva se ha convertido en un método clave para resolución de problemas difíciles de optimización, debido, entre otros factores, a la flexibilidad y adaptabilidad en la resolución, combinados con la robustez y las ventajas de la búsqueda global.

2.4.1. Fundamentos los Algoritmos Genéticos

La unidad fundamental de información en los seres vivos es el gen. El gen es parte de una estructura denominada cromosoma. El material genético de un individuo se conoce como el genotipo. Su manifestación como organismo, que determina o afecta la propiedad visible se denomina fenotipo. Los organismos vivos pueden ser visualizados como un dual de su genotipo (la codificación genética) y de su fenotipo (la propiedad visible). Ver figura 2.4 (Martin & Lopez, 2012)

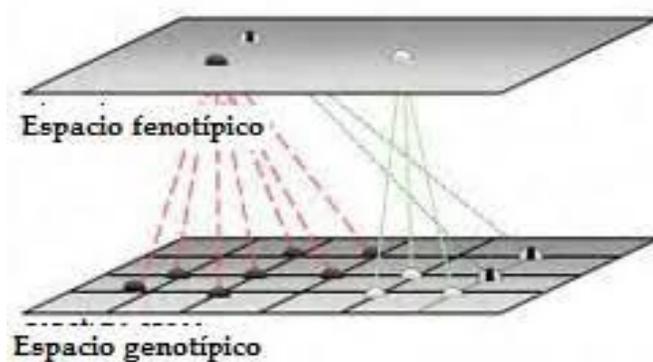


Figura 2.4 Espacios Genotípicos y Fenotípicos en la Evolución

Fuente: Martin & López (2012)

Un problema de optimización requiere hallar un conjunto de parámetros de forma que se cumpla un cierto criterio de calidad que se quiere optimizar, es decir, maximizando o minimizando una cierta función de evaluación $f(x)$ dada.

La computación evolutiva es especialmente útil cuando se enfrenta problemas difíciles o complejos, como lo son aquellos caracterizados por una alta dimensionalidad, multimodalidad, fuerte no linealidad, no diferenciabilidad, presencia de ruido y cuando se trata con funciones dependientes del tiempo.

La computación evolutiva comprende aplicaciones basadas en los mecanismos de la evolución natural tales como la genética biológica y la selección natural. El campo de la computación evolutiva, en general incluye cuatro paradigmas:

- ✓ Algoritmos genéticos.
- ✓ Programación evolutiva.
- ✓ Estrategias de evolución.
- ✓ Programación genética.

Los paradigmas de la computación evolutiva difieren de los métodos tradicionales, en cuatro aspectos fundamentales:

- ✓ Trabajan con un código del conjunto de parámetros. Este código es una cadena de caracteres de valores binarios o variables de valor real.
- ✓ Realizan la búsqueda de la potencial solución en una población de puntos, no en un sólo punto.
- ✓ Utilizan la información de una función objetivo, no derivadas u otro conocimiento auxiliar.
- ✓ Usan reglas probabilísticas de transición, en lugar de reglas determinísticas.

2.4.2. Características de los Algoritmos Genéticos

Los Algoritmos Genéticos son métodos adaptativos que se derivan de la simulación de los procesos genéticos naturales. Comprenden los mecanismos

de selección (la supervivencia del más apto), cruce (intercambio genético) y mutación. Se utilizan para resolver problemas de búsqueda y optimización.

Los Algoritmos Genéticos trabajan en el espacio del genotipo del código de la información:

- ✓ Se selecciona una población de individuos, cada uno de los cuales representa una solución factible a un problema dado.
- ✓ Cada individuo es codificado como una cadena de caracteres binarios (cromosomas), y se le asigna un valor de aptitud relacionado con la bondad de la solución que representa.
- ✓ Dependiendo de la aptitud de un individuo, la probabilidad de que sea seleccionado para la reproducción y por tanto de que su material genético se propague en sucesivas generaciones, puede ser mayor o menor. Cuanto mayor sea la aptitud de un individuo, mayor será la probabilidad de que el mismo sea seleccionado para reproducirse intercambiando aleatoriamente su material genético con otro individuo seleccionado de igual forma.
- ✓ Este cruce producirá nuevos individuos, descendientes de los anteriores, los cuales comparten algunas de las características de sus ancestros.
- ✓ Ocasionalmente, una variante nueva es probada, cambiando aleatoriamente un gen de algún cromosoma descendiente, como una medida de posible mejora (mutación). Ver figura 2.5

```
BEGIN /* Algoritmo Genetico Simple */
  Generar una poblacion inicial.
  Computar la funcion de evaluacion de cada individuo.
  WHILE NOT Terminado DO
    BEGIN /* Producir nueva generacion */
      FOR Tamaño poblacion/2 DO
        BEGIN /*Ciclo Reproductivo */
          Seleccionar dos individuos de la anterior generacion,
          para el cruce (probabilidad de seleccion proporcional
          a la funcion de evaluacion del individuo).
          Cruzar con cierta probabilidad los dos
          individuos obteniendo dos descendientes.
          Mutar los dos descendientes con cierta probabilidad.
          Computar la funcion de evaluacion de los dos
          descendientes mutados.
          Insertar los dos descendientes mutados en la nueva generacion.
        END
      IF la poblacion ha convergido THEN
        Terminado := TRUE
      END
    END
  END
```

Figura 2.5 Pseudocódigo de un Algoritmo Genético Simple

Fuente: Martin & López (2012)

2.4.2.1. Codificación de la Variable

Para codificar la variable que representa la serie de números entre 0 y 255, se puede utilizar notación binaria, con 8 bits. Esta cadena de 1s y 0s que representa cada número constituye un cromosoma. A cada cromosoma se le asigna un valor de aptitud. En este caso, como el objetivo es localizar el máximo número, se puede utilizar como función de optimización el cuadrado del número decimal ya que, mientras más cerca esté del 255, mayor será su aptitud para constituirse en solución al objetivo esperado.

2.4.2.2. Proceso de Selección

Para iniciar el proceso de selección, se calcula la aptitud relativa de cada ejemplar, en relación a la aptitud total de la población. Con estos valores, se puede construir una ruleta en la que las áreas de los sectores circulares son proporcionales a la aptitud relativa de cada ejemplar.

A continuación, se procede a hacer girar la ruleta dos veces, para seleccionar los dos ejemplares más aptos de la población inicial. Nótese que los ejemplares más aptos ocupan las mayores áreas, por lo que, al jugar la ruleta, estos son los que más probabilidad tienen de ser seleccionados. Estos detalles se muestran en la tabla 2.1 y figura 2.6

Tabla 2.1 Población inicial, cromosomas y aptitud

ITEM	DECIMAL	BINARIO	APTITUD	APTITUD RELAITVA
1	127	01111111	16129	11%
2	82	01010010	6724	5%
3	104	01101000	10816	7%
4	69	01000101	4761	3%
5	55	00110111	3025	2%
6	221	11011101	48841	34%
7	16	00010000	256	0%

8	211	11010011	44521	31%
9	27	00011011	729	1%
10	98	01100010	9604	7%
			14546	100%

Fuente: Martin & López (2012)

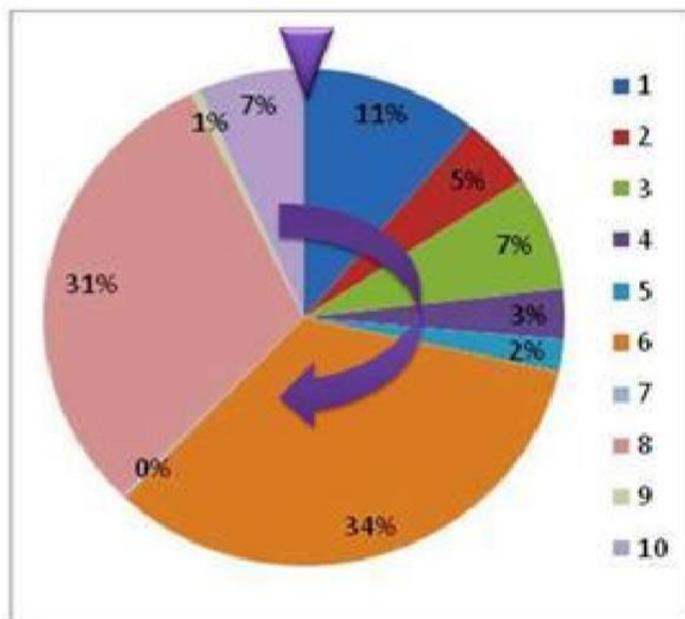


Figura 2.6 Ruleta de Selección del más Apto

Fuente: Martin & López (2012)

2.4.2.3. Cruce y Mutación

Supóngase que la ruleta selecciona el ítem 8 (211) y el ítem 6 (221). Aleatoriamente se determina un punto de cruce y se procede a intercambiar el material genético de los dos padres, para configurar sus dos hijos. Hecho esto, a continuación, se introduce una mutación. En este caso, se complementa el valor del gen, escogido aleatoriamente, y se obtienen los dos ejemplares que

ingresan a la nueva población. En este ejemplo, aparece el número 251 [11111101] y se mantiene el 211 [11010011]. Ver figura 2.7

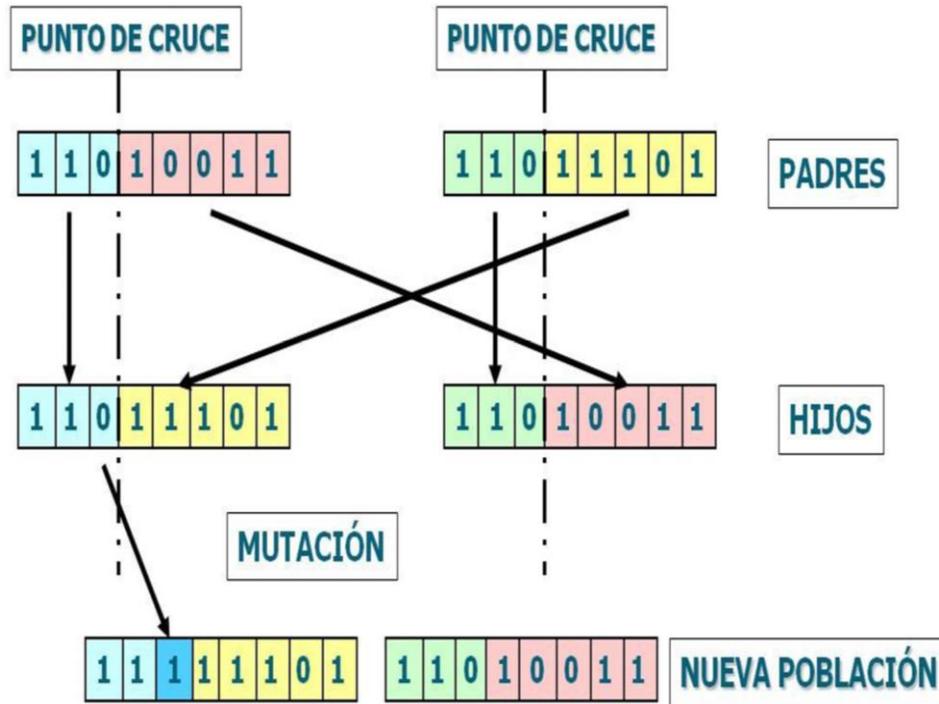


Figura 2.7 Operaciones de cruce y mutación

Fuente: Banda (2014)

Este proceso se realiza hasta emparejar todos los ejemplares de la población inicial, esto es 4 veces más. Hasta que la mayoría de ejemplares se aproximen al número 255, se produce una nueva generación. Como se puede apreciar, a través de estas operaciones, van apareciendo números de la serie que inicialmente no estuvieron presentes en la población.

Además, generación tras generación, los nuevos ejemplares irán aproximándose al 255, hasta que eventualmente este número domine la población o se agote el material genético y se estanque la evolución, en cuyo caso se tendrá que iniciar un nuevo proceso ya sea con los mismos ejemplares iniciales o generando aleatoriamente una nueva población inicial.

2.4.2.4. Modificaciones al Algoritmo Genético Simple

Entre las posibles modificaciones al algoritmo genético simple, se pueden considerar las siguientes:

- ✓ Utilizar una población variable. - En este caso, regularmente se introduce nuevo material genético para refrescar la población.
- ✓ Representar los cromosomas utilizando el Código Gray. En ciertos problemas puede reducirse las variaciones entre cromosomas de ejemplares sucesivos. El código Gray se caracteriza porque el código binario de dos números sucesivos cambia solo en un bit. Ver tabla 2.2

Tabla 2.2 Equivalencia de Código - Gray

Decimal	Binario	Código Gray	Decimal	Binario	Código Gray
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

Fuente: Banda (2014)

- ✓ Métodos de Selección. - A más de la ruleta, se puede utilizar una selección aleatoria o una selección ranqueada.

- ✓ Operaciones Genéticas. - Para problemas en los que las variables se codifican en cromosomas largos, puede ser de ayuda utilizar múltiples puntos de cruce. Otras opciones son: cruce uniforme, máscaras aleatorias o reordenamiento.

2.4.3. Programación Evolutiva

Fue utilizada por primera vez por Lawrence J. Fogel en los EE.UU. en los 1960s con el fin de utilizar la evolución simulada como un proceso de aprendizaje para generar inteligencia artificial. Actualmente, la programación evolutiva es un dialecto de la amplia gama de la computación evolutiva. (Banda Gamboa, 2014)

La programación evolutiva se deriva de la simulación del comportamiento adaptativo propio de un proceso evolutivo. Se caracteriza por su énfasis en el desarrollo de modelos de comportamiento. Trabaja en el espacio del fenotipo, es decir, la evolución es en el comportamiento observable.

La programación evolutiva se utiliza usualmente para resolver problemas de predicción. Emplea máquinas de estados finitos para la predicción y la evolución.

- ✓ El entorno es descrito como una secuencia de símbolos tomados de un alfabeto finito.
- ✓ El algoritmo evolutivo opera sobre la secuencia de símbolos observados, de tal forma que produce un símbolo de salida que maximiza el resultado, tomando en cuenta el siguiente símbolo que aparecerá (predicción) y una función de costo definida que guía el proceso.

Su operador principal es la mutación, los miembros de la población son vistos como parte de una especie específica en lugar de miembros de la misma especie. Ver figura 2.8.

```

INICIO/* Programación Evolutiva: Procedimiento*/
    Generar una población inicial.
    MIENTRAS NO terminado HACER
        INICIO /* Buscar Solución*/
            Exponer la población a su entorno.
            Calcular la aptitud de cada individuo.
            Aleatoriamente mutar cada miembro de la población y genera
            descendiente.
            Calcular la aptitud de los descendientes.
            Seleccionar los miembros más aptos para la nueva generación
        SI el objetivo se ha logrado ENTONCES
            Terminado = VERDADERO
        FIN/* Buscar Solución*/
    FIN/* programación Evolutiva: Procedimiento*/

```

Figura 2.8 Pseudocódigo de Algoritmo para Programación Evolutiva

Fuente: Banda (2014)

2.4.4. Estrategias de Evolución

Las estrategias de evolución están basadas en el concepto de meta evolución y, al igual que la programación evolutiva, enfatizan el aspecto fenotípico del proceso evolutivo.

A pesar de utilizar las operaciones de mutación y cruce, que en este caso se las denomina operaciones de recombinación, las perspectivas de ellas son diferentes a las utilizadas por los algoritmos genéticos o la programación evolutiva:

- ✓ Operaciones de mutación o recombinación que caen fuera de una ventana de evolución predefinida por la función de optimización, no son de utilidad.
- ✓ El ajuste dinámico del tamaño de la mutación a una ventana de evolución también dinámica, favorece la meta evolución.

A diferencia de los algoritmos genéticos, las estrategias de evolución manipulan los valores de las variables, durante la recombinación:

- ✓ Forma un descendiente utilizando los datos de dos padres seleccionados aleatoriamente.
- ✓ El valor de la variable del descendiente, es igual al valor intermedio de los valores de sus padres.

La estrategia de evolución opera con una mayor cantidad de descendientes que los otros paradigmas de computación evolutiva:

El número de descendientes (N) es mayor que el número de padres (M), en una relación típica N/M igual a 2.4. Ver figura 2.9

```
INICIO/* Programación de Evolución: Procedimiento*/
  Generar una población inicial.
  MIENTRAS NO terminado HACER
    INICIO /* Buscar Solución*/
      Realiza la recombinación de pares de los M padres
      Para generar N descendientes.
      Mutar a todos los descendientes.
      Calcular la aptitud de los N descendientes.
      De entre los N descendientes, seleccionar los M
      Miembros más aptos para la nueva generación.
      SI el objetivo se ha logrado ENTONCES
        Terminado = VERDADERO
    FIN/* Buscar Solución*/
  FIN/* programación Evolutiva: Procedimiento*/
```

Figura 2.9 Pseudocódigo de Procedimiento de Estrategias de Evolución

Fuente: Banda (2014)

2.4.5. Programación Genética

La programación genética está diseñada para evolucionar programas de computadora, genéticamente. A diferencia de las tres implementaciones anteriores, que generalmente emplean cromosomas en forma de cadena de caracteres como miembros individuales de la población, la programación genética utiliza árboles para representar a los programas sujetos a evolución:

- ✓ Las funciones definidas para el problema aparecen en los nodos (conjunto interno). Ver figura 2.10
- ✓ Las variables de estado y las constantes están localizadas en las hojas del árbol (conjunto terminal).

```
INICIO/* Programación Evolutiva: Procedimiento*/  
    Inicializar la población de programas de computadora.  
    MIENTRAS NO terminado HACER  
        INICIO /* Buscar Solución*/  
            Determinar la aptitud de cada programa individual.  
            Aplicar la reproducción de acuerdo con la aptitud y la  
            probabilidad de reproducción  
            Realizar el cruce de las subexpresiones.  
            SI el objetivo se ha logrado ENTONCES  
                Terminado = VERDADERO  
        FIN/* Buscar Solución*/  
    FIN/* programación Evolutiva: Procedimiento*/
```

Figura 2.10 Pseudocódigo para Procesamiento de Programación Genética

Fuente: Banda (2014)

Previo a la ejecución de un proceso de programación genética, se deben cumplir los siguientes pasos:

- ✓ Especificar el conjunto de variables de estado y constantes (conjunto terminal).
- ✓ Especificar el conjunto de funciones.
- ✓ Especificar la medida de aptitud.
- ✓ Seleccionar los parámetros del sistema de control.
- ✓ Especificar las condiciones de finalización del procedimiento.

2.5. Método Científico (Bunge)

El método científico es el conjunto de procedimientos por los cuales se plantean los problemas científicos y se ponen a prueba las hipótesis científicas. (Bunge, 1960)

Un método es un procedimiento para tratar un conjunto de problemas. En ciencia, el método científico es la estrategia para la investigación y la exploración de lo desconocido.

El método científico y el conocimiento objetivo del mundo constituyen la principal distinción entre la ciencia y la pseudociencia.

2.5.1. Características del método científico

Las características del método científico se plantean en: (Bunge, 1960)

Es objetivo: se fundamenta en los hechos y no en las experiencias subjetivas de los investigadores.

Es sistemático: sigue un orden con propósito.

Es un proceso circular: se inicia con las observaciones y los hechos, se levantan problemas o dudas, se plantean hipótesis o explicaciones tentativas, se diseñan experimentos y se generan nuevas observaciones y hechos.

Es corregible: a medida que se van obteniendo resultados se puede corregir los errores que van apareciendo.

Es verificable: puede ser sometido al examen independiente y externo.

No es rígido: el método científico no es un método rígido e infalible y se puede perfeccionar.

2.5.2. La Pauta de la Investigación Científica Diseño de la prueba

La variedad de habilidades y de información que exige el tratamiento científico de los problemas ayuda a explicar la extremada división del trabajo prevaleciente en la ciencia contemporánea, en la que encuentra lugar toda capacidad natural y toda habilidad adquirida. Es posible apreciar esta variedad exponiendo la pauta general de la investigación científica. Para ello se deben seguir los siguientes pasos planteados:

Planteo del Problema

Reconocimiento de los hechos: examen del grupo de hechos, clasificación preliminar y selección de los que probablemente sean relevantes en algún respecto.

Descubrimiento del problema: hallazgo de la laguna o de la incoherencia en el cuerpo del saber.

Formulación del problema: planteo de una pregunta que tiene probabilidad de ser la correcta; esto es, reducción del problema a su núcleo significativo, probablemente soluble y probablemente fructífero, con ayuda de conocimiento disponible.

Construcción de un Modelo Teórico

Selección de los factores pertinentes: invención de suposiciones plausibles relativas a las variables que probablemente son pertinentes.

Invención de las hipótesis centrales y de las suposiciones auxiliares: propuesta de un conjunto de suposiciones concernientes a los nexos entre las variables pertinentes; p. ej. formulación de enunciados de ley que se espera puedan amoldarse a los hechos observados.

Traducción matemática: cuando sea posible, traducción de las hipótesis, o de parte de ellas, a alguno de los lenguajes matemáticos.

Deducción de Consecuencias Particulares

Búsqueda de soportes racionales: deducción de consecuencias particulares que pueden haber sido verificadas en el mismo campo o en campos contiguos.

Búsqueda de soportes empíricos: elaboración de predicciones sobre la base de modelo teórico y de datos empíricos, teniendo en vista técnicas de verificación disponibles o concebibles.

Prueba de las Hipótesis: planeamiento de los medios para poner a prueba las predicciones; diseño de observaciones, mediciones, experimentos y demás operaciones instrumentales.

Ejecución de la prueba: realización de las operaciones y recolección de datos.

Elaboración de los datos: clasificación, análisis, evaluación, reducción, etc., de los datos empíricos.

Inferencia de la conclusión: interpretación de los datos elaborados a la luz del modelo teórico.

Introducción de las Conclusiones en la Teoría Comparación de las conclusiones con las predicciones: contraste de los resultados de la prueba con

las consecuencias del modelo teórico, precisando en qué medida éste puede considerarse confirmado o disconforme (inferencia probable).

Reajuste del modelo: eventual corrección o aun reemplazo del modelo.

Sugerencias acerca de trabajo ulterior: búsqueda de lagunas o errores en la teoría y/o los procedimientos empíricos, si el modelo ha sido disconforme; si ha sido confirmado, examen de posibles extensiones y de posibles consecuencias en otros departamentos del saber.

Comunicación de los hallazgos

Una parte importante del trabajo científico es la comunicación y la transmisión de la información. Esto se hace a través de artículos científicos, tesis, libros, o conferencias en institutos educativos y de investigación.

2.5.3. Objetivos del método científico

¿Para qué fines se emplea el método científico? Podríamos decir que el método científico tiene dos objetivos generales: uno intrínseco y otro extrínseco.

Objetivo intrínseco o cognitivo

El objetivo intrínseco o cognitivo es incrementar nuestro conocimiento. Esto es, investigar algo por el mero gusto de saber.

Objetivo extrínseco o utilitario

El método científico nos ayuda a aumentar nuestro bienestar y nuestro poder (objetivos extrínsecos o utilitarios). Mucho del conocimiento básico adquirido es luego empleado para fines útiles.

Por ejemplo, en la actualidad se pueden diagnosticar enfermedades genéticas gracias a una enzima (la Taq polimerasa) encontrada en la bacteria *Thermus aquaticus*.

2.6. Modelo XP

La **Ingeniería de Software** es la rama de la ingeniería que estudia todo lo relacionado con la informática o sistemas de computación, con una orientación metódica, ordenada y cuantificable al incremento, ejecución y conservación del software. En este caso utilizaremos XP para desarrollar el trabajo de investigación.

La metodología XP define cuatro variables para cualquier proyecto de software: costo, tiempo, calidad y alcance. Además, se especifica que, de estas cuatro variables, sólo tres de ellas podrán ser fijadas arbitrariamente por actores externos al grupo de desarrolladores (clientes y jefes de proyecto). El valor de la variable restante podrá ser establecido por el equipo de desarrollo, en función de los valores de las otras tres. Este mecanismo indica que, por ejemplo, si el cliente establece el alcance y la calidad, y el jefe de proyecto el precio, el grupo de desarrollo tendrá libertad para determinar el tiempo que durará el proyecto. Este modelo es analizado por Kent Beck, en donde propone las ventajas de un contrato con alcances opcionales. (Kent Beck, 1999).

Considerando aportes de otros investigadores que aclaran, los ciclos de vida “tradicionales” proponen una clara distinción entre las etapas del proyecto de software, y tienen un plan bien preestablecido acerca del proceso de desarrollo. Asimismo, en todos ellos se parte de especificaciones claras, si no del total del proyecto, por lo menos de una buena parte inicial. (JoKowicz, 2008)

El ciclo de vida de un proyecto XP incluye, al igual que las otras metodologías, entender lo que el cliente necesita, estimar el esfuerzo, crear la solución y entregar el producto final al cliente. Sin embargo, XP propone un ciclo de vida dinámico, donde se admite expresamente que, en muchos casos, los clientes no son capaces de especificar sus requerimientos al comienzo de un proyecto.

Por esto, se trata de realizar ciclos de desarrollo cortos (llamados iteraciones), con entregables funcionales al finalizar cada ciclo. En cada iteración se realiza un ciclo completo de análisis, diseño, desarrollo y pruebas, pero utilizando un conjunto de reglas y prácticas que caracterizan a XP (y que serán detalladas más adelante).

Típicamente un proyecto con XP lleva 10 a 15 ciclos o iteraciones. La siguiente figura 2.11 esquematiza los ciclos de desarrollo en cascada e iterativos tradicionales (por ejemplo, incremental o espiral), comparados con el de XP. (JoKowicz, 2008)

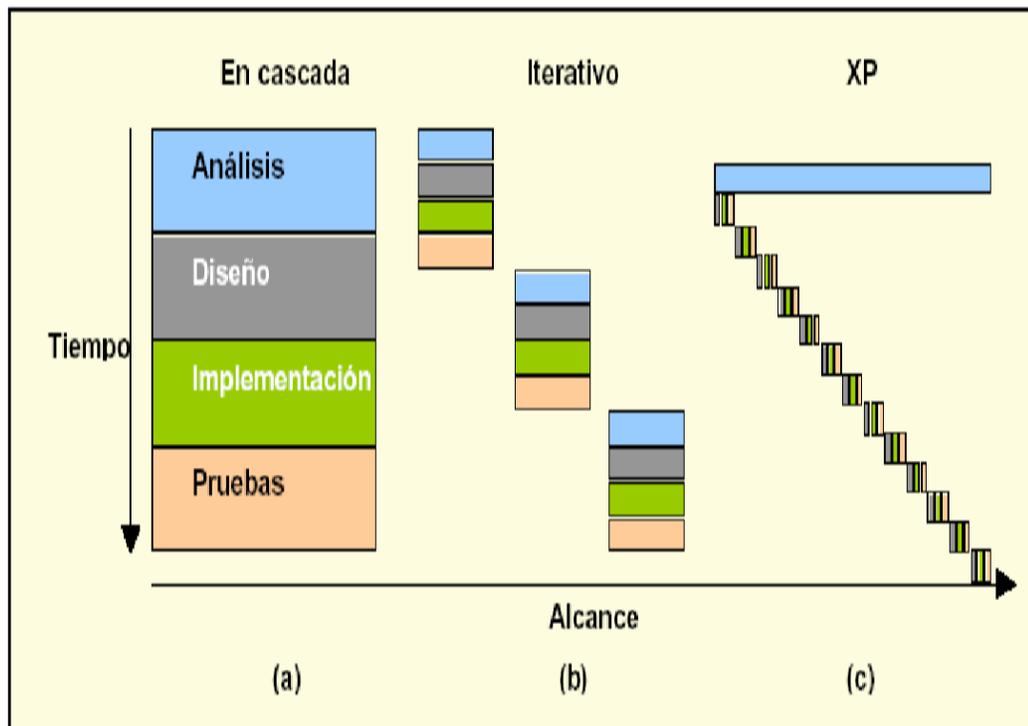


Figura 2.11 Ciclos de desarrollo en cascada y tradicional.

Fuente: JoKowicz, 2008, p.8

Si bien el ciclo de vida de un proyecto XP es muy dinámico, se puede separar en fases. Varios de los detalles acerca de las tareas de estas fases se detallan más adelante, en la sección “Reglas y Practicas”:

2.6.1. Fase de Exploración

Es la fase en la que se define el alcance general del proyecto. En esta fase, el cliente define lo que necesita mediante la redacción de sencillas “historias de usuarios”. Los programadores estiman los tiempos de desarrollo en base a esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias (ya que estarán basadas en datos de muy alto nivel, y podrían variar cuando se analicen más en detalle en cada iteración). Esta fase dura típicamente un par de semanas, y el resultado es una visión general del sistema, y un plazo total estimado.

2.6.2. Fase de Planificación

La planificación es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario y asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas, o “Release Plan”, como se detallará en la sección “Reglas y Practicas”.

2.6.3. Fase de Iteraciones

Esta es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas en esta fase, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración. Como las historias de usuario no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se realizan las tareas necesarias de análisis, recabando con el cliente todos los datos que sean necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo. Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.

Fase de puesta en producción Si bien al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción hasta tanto no se tenga la funcionalidad completa. En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste (“fine tuning”).

2.6.4. Reglas y Practicas

La metodología XP tiene un conjunto importante de reglas y prácticas. En forma genérica, se pueden agrupar en:

- Reglas y prácticas para la Planificación
- Reglas y prácticas para el Diseño
- Reglas y prácticas para el Desarrollo
- Reglas y prácticas para las Pruebas

2.6.4.1. Planificación

La metodología XP plantea la planificación como un dialogo continuo entre las partes involucradas en el proyecto, incluyendo al cliente, a los programadores y a los coordinadores o gerentes. El proyecto comienza recopilando “Historias de usuarios”, las que sustituyen a los tradicionales “casos de uso”. Una vez obtenidas las “historias de usuarios”, los programadores evalúan rápidamente el tiempo de desarrollo de cada una. Si alguna de ellas tiene “riesgos” que no permiten establecer con certeza la complejidad del desarrollo, se realizan pequeños programas de prueba (“spikes”), para reducir estos riesgos. Una vez realizadas estas estimaciones, se organiza una reunión de planificación, con los diversos actores del proyecto (cliente, desarrolladores, gerentes), a los efectos de establecer un plan o cronograma de entregas (“Release Plan”) en los que todos estén de acuerdo. Una vez acordado este cronograma, comienza una fase

de iteraciones, en dónde en cada una de ellas se desarrolla, prueba e instala unas pocas “historias de usuarios”.

Los conceptos básicos de esta planificación son los siguientes:

Historias de usuarios. Las “Historias de usuarios” (“User stories”) sustituyen a los documentos de especificación funcional, y a los “casos de uso”. Estas “historias” son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. La diferencia más importante entre estas historias y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido. Las historias de usuario deben tener el detalle mínimo como para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo. Cuando llegue el momento de la implementación, los desarrolladores dialogarán directamente con el cliente para obtener todos los detalles necesarios. Las historias de usuarios deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia. (JoKowicz, 2008)

Plan de entregas (“Release Plan”) El cronograma de entregas establece qué historias de usuario serán agrupadas para conformar una entrega, y el orden de las mismas. Este cronograma será el resultado de una reunión entre todos los actores del proyecto (cliente, desarrolladores, gerentes, etc.). XP denomina a esta reunión “Juego de planeamiento” (“Planning game”), pero puede denominarse de la manera que sea más apropiada al tipo de empresa y cliente. Típicamente el cliente ordenará y agrupará según sus prioridades las historias de usuario. El cronograma de entregas se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los desarrolladores. Luego de algunas iteraciones es recomendable realizar nuevamente una reunión con los actores

del proyecto, para evaluar nuevamente el plan de entregas y ajustarlo si es necesario.

Plan de iteraciones (“Iteration Plan”) Las historias de usuarios seleccionadas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo al orden preestablecido. Al comienzo de cada ciclo, se realiza una reunión de planificación de la iteración. Cada historia de usuario se traduce en tareas específicas de programación. Asimismo, para cada historia de usuario se establecen las pruebas de aceptación. Estas pruebas se realizan al final del ciclo en el que se desarrollan, pero también al final de cada uno de los ciclos siguientes, para verificar que subsiguientes iteraciones no han afectado a las anteriores. Las pruebas de aceptación que hayan fallado en el ciclo anterior son analizadas para evaluar su corrección, así como para prever que no vuelvan a ocurrir.

Reuniones diarias de seguimiento (“Stand-up meeting”) El objetivo de tener reuniones diarias es mantener la comunicación entre el equipo, y compartir problemas y soluciones. En la mayoría de estas reuniones, gran parte de los participantes simplemente escuchan, sin tener mucho que aportar. Para no quitar tiempo innecesario del equipo, se sugiere realizar estas reuniones en círculo y de pie.

2.6.4.2. Diseño

La metodología XP hace especial énfasis en los diseños simples y claros. Los conceptos más importantes de diseño en esta metodología son los siguientes:

Simplicidad. Un diseño simple se implementa más rápidamente que uno complejo. Por ello XP propone implementar el diseño más simple posible que funcione. Se sugiere nunca adelantar la implementación de funcionalidades que no correspondan a la iteración en la que se esté trabajando.

Soluciones “spike”. Cuando aparecen problemas técnicos, o cuando es difícil de estimar el tiempo para implementar una historia de usuario, pueden utilizarse pequeños programas de prueba (llamados “spike”¹), para explorar diferentes soluciones. Estos programas son únicamente para probar o evaluar una solución, y suelen ser desechados luego de su evaluación.

Recodificación. La recodificación (“refactoring”) consiste en escribir nuevamente parte del código de un programa, sin cambiar su funcionalidad, a los efectos de hacerlo más simple, conciso y/o entendible. Muchas veces, al terminar de escribir un código de programa, pensamos que, si lo comenzáramos de nuevo, lo hubiéramos hecho en forma diferente, más clara y eficientemente. Sin embargo, como ya está pronto y “funciona”, rara vez es reescrito. Las metodologías de XP sugieren recodificar cada vez que sea necesario. Si bien, puede parecer una pérdida de tiempo innecesaria en el plazo inmediato, los resultados de esta práctica tienen sus frutos en las siguientes iteraciones, cuando sea necesario ampliar o cambiar la funcionalidad. La filosofía que se persigue es, como ya se mencionó, tratar de mantener el código más simple posible que implemente la funcionalidad deseada.

Metáforas Una “metáfora” es algo que todos entienden, sin necesidad de mayores explicaciones. La metodología XP sugiere utilizar este concepto como una manera sencilla de explicar el propósito del proyecto, y guiar la estructura y arquitectura del mismo. Por ejemplo, puede ser una guía para la nomenclatura de los métodos y las clases utilizadas en el diseño del código. Tener nombres claros, que no requieran de mayores explicaciones, redundará en un ahorro de tiempo. Es muy importante que el cliente y el grupo de desarrolladores estén de acuerdo y compartan esta “metáfora”, para que puedan dialogar en un “mismo idioma”. Una buena metáfora debe ser fácil de comprender para el cliente y a su vez debe tener suficiente contenido como para que sirva de guía a la arquitectura del proyecto. Sin embargo, ésta práctica resulta, muchas veces, difícil de realizar.

En un trabajo realizado en el School of Computer Science del Carnegie Mellon, se cuestiona la utilidad de su uso (JoKowicz, 2008)

2.6.4.3. Desarrollo del Código

Disponibilidad del cliente Uno de los requerimientos de XP es tener al cliente disponible durante todo el proyecto. No solamente como apoyo a los desarrolladores, sino formando parte del grupo. El involucramiento del cliente es fundamental para que pueda desarrollarse un proyecto con la metodología XP. Al comienzo del proyecto, el cliente debe proporcionar las historias de usuarios. Pero, dado que estas historias son expresamente cortas y de “alto nivel”, no contienen los detalles necesarios para realizar el desarrollo del código. Estos detalles deben ser proporcionados por el cliente, y discutidos con los desarrolladores, durante la etapa de desarrollo. No se requieren de largos documentos de especificaciones, sino que los detalles son proporcionados por el cliente, en el momento adecuado, “cara a cara” a los desarrolladores.

Si bien esto parece demandar del cliente recursos por un tiempo prolongado, debe tenerse en cuenta que en otras metodologías este tiempo es insumido por el cliente en realizar los documentos detallados de especificación. Adicionalmente, al estar el cliente en todo el proceso, puede prevenir a tiempo de situaciones no deseables, o de funcionamientos que no eran los que en realidad se deseaban. En otras metodologías, estas situaciones son detectadas en forma muy tardía del ciclo de desarrollo, y su corrección puede llegar a ser muy complicada.

Uso de estándares Si bien esto no es una idea nueva, XP promueve la programación basada en estándares, de manera que sea fácilmente entendible por todo el equipo, y que facilite la recodificación.

Programación dirigida por las pruebas (“Test-driven programming”) En las metodologías tradicionales, la fase de pruebas, incluyendo la definición de los

tests, es usualmente realizada sobre el final del proyecto, o sobre el final del desarrollo de cada módulo. La metodología XP propone un modelo inverso, en el que, lo primero que se escribe son los test que el sistema debe pasar. Luego, el desarrollo debe ser el mínimo necesario para pasar las pruebas previamente definidas. Las pruebas a las que se refiere esta práctica, son las pruebas unitarias, realizados por los desarrolladores. La definición de estos test al comienzo, condiciona o “dirige” el desarrollo. Un ejemplo de cómo realizar esto de manera práctica puede verse en.

Programación en pares XP propone que se desarrolle en pares de programadores, ambos trabajando juntos en un mismo ordenador. Si bien parece que ésta práctica duplica el tiempo asignado al proyecto, al trabajar en pares se minimizan los errores y se logran mejores diseños, compensando la inversión en horas. El producto obtenido es por lo general de mejor calidad que cuando el desarrollo se realiza por programadores individuales. En un estudio realizado por Cockburn y Williams se concluye que la programación en pares tiene un sobre costo aproximado de 15%, y no de un 100% como se puede pensar a priori. Este sobre costo es rápidamente pagado por la mejor calidad obtenida en el producto final. Adicionalmente, la programación en pares tiene las siguientes ventajas:

- ✓ La mayoría de los errores se descubren en el momento en que se codifican, ya que el código es permanentemente revisado por dos personas.
- ✓ La cantidad de defectos encontrados en las pruebas es estadísticamente menor.
- ✓ Los diseños son mejores y el código más corto.
- ✓ El equipo resuelve problemas en forma más rápida.
- ✓ Las personas aprenden significativamente más, acerca del sistema y acerca de desarrollo de software.
- ✓ El proyecto termina con más personas que conocen los detalles de cada parte del código.

- ✓ Las personas aprenden a trabajar juntas, generando mejor dinámica de grupo y haciendo que la información fluya rápidamente.
- ✓ Las personas disfrutan más de su trabajo.
- ✓ Integraciones permanentes

Todos los desarrolladores necesitan trabajar siempre con la “última versión”. Realizar cambios o mejoras sobre versiones antiguas causan graves problemas, y retrasan al proyecto. Es por eso que XP promueve publicar lo antes posible las nuevas versiones, aunque no sean las últimas, siempre que estén libres de errores. Idealmente, todos los días deben existir nuevas versiones publicadas. Para evitar errores, solo una pareja de desarrolladores puede integrar su código a la vez. Propiedad colectiva del código En un proyecto XP, todo el equipo puede contribuir con nuevas ideas que apliquen a cualquier parte del proyecto. Asimismo, cualquier pareja de programadores puede cambiar el código que sea necesario para corregir problemas, agregar funciones o recodificar. En otras metodologías, este concepto puede parecer extraño. Muchas veces se asume que, si hay algo de propiedad colectiva, la responsabilidad también es colectiva. Y que “todos sean responsables”, muchas veces significa que “nadie es responsable”. Ward Cunningham explica en una entrevista con Bill Veners, que este razonamiento no es correcto cuando se trabaja con la metodología de XP. En este caso, quienes encuentran un problema, o necesitan desarrollar una nueva función, pueden resolverlo directamente, sin necesidad de “negociar” con el “dueño” o autor del módulo (ya que, de hecho, este concepto no existe en XP). Muchas veces, explica Cunningham, una solución pasa por la recodificación de varios módulos, que atraviesan de forma horizontal una determinada jerarquía vertical. Si es necesario dialogar y convencer al encargado de cada módulo, posiblemente la solución no se pueda implementar, por lo menos en tiempos razonables. En XP, se promueve la recodificación, en aras de generar códigos más simples y adaptados a las realidades cambiantes. Cualquier pareja de programadores puede tomar la responsabilidad de este cambio. Los testeos

permanentes deberían de asegurarse que los cambios realizados cumplen con lo requerido y, además, no afectan al resto de las funcionalidades.

Ritmo sostenido La metodología XP indica que debe llevarse un ritmo sostenido de trabajo. Anteriormente, ésta práctica se denominaba “Semana de 40 horas”. Sin embargo, lo importante no es si se trabajan, 35, 40 o 42 horas por semana. El concepto que se desea establecer con esta práctica es el de planificar el trabajo de manera de mantener un ritmo constante y razonable, sin sobrecargar al equipo. Cuando un proyecto se retrasa, trabajar tiempo extra puede ser más perjudicial que beneficioso. El trabajo extra desmotiva inmediatamente al grupo e impacta en la calidad del producto. En la medida de lo posible, se debería renegociar el plan de entregas (“Release Plan”), realizando una nueva reunión de planificación con el cliente, los desarrolladores y los gerentes. Adicionalmente, agregar más desarrolladores en proyectos ya avanzados no siempre resuelve el problema.

2.6.4.4. Pruebas

Pruebas unitarias. Las pruebas unitarias son una de las piedras angulares de XP. Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados. Por otra parte, como se mencionó anteriormente, las pruebas deben ser definidas antes de realizar el código (“Test-driven programming”). Que todo código liberado pase correctamente las pruebas unitarias es lo que habilita que funcione la propiedad colectiva del código. En este sentido, el sistema y el conjunto de pruebas debe ser guardado junto con el código, para que pueda ser utilizado por otros desarrolladores, en caso de tener que corregir, cambiar o recodificar parte del mismo.

Detección y corrección de errores. Cuando se encuentra un error (“bug”), éste debe ser corregido inmediatamente, y se deben tener precauciones para que errores similares no vuelvan a ocurrir. Asimismo, se generan nuevas pruebas para verificar que el error haya sido resuelto.

Pruebas de aceptación. Las pruebas de aceptación son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada.

Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información.

2.7. Métrica de calidad ISO 9126

En el sentido, de poder visualizar claramente que la calidad, es un elemento que contiene muchas aristas, y no solamente la determina un factor bivalente (1 o 0), es decir, tiene calidad o no tiene. (Valdés Souto, 2013)

La norma ISO/IEC 9126 permite especificar y evaluar la calidad del software desde diferentes criterios asociados con adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad y auditoría de software. Los modelos de calidad para el software se describen así:

Calidad interna y externa: Especifica 6 características para calidad interna y externa, las cuales, están subdivididas. Estas divisiones se manifiestan externamente cuando el software es usado como parte de un sistema Informático, y son el resultado de atributos internos de software.

Calidad en uso: Calidad en uso es el efecto combinado para el usuario final de las 6 características de la calidad interna y externa del software. Especifica 4 características para la calidad en uso.

Al unir la calidad interna y externa con la calidad en uso se define un modelo de evaluación más completo, se puede pensar que la usabilidad del modelo de calidad externa e interna pueda ser igual al modelo de calidad en uso, pero no, la usabilidad es la forma como los profesionales interpretan o asimilan la funcionabilidad del software y la calidad en uso se puede asumir como la forma que lo asimila o maneja el usuario final. Si se unen los dos modelos, se puede definir que los seis indicadores del primer modelo tienen sus atributos y el modelo de calidad en uso sus 4 indicadores pasarían hacer sus atributos, mirándolo gráficamente quedaría así:



Figura 2.12 Norma de Evaluación ISO/IEC 9126

Se establecen categorías para las cualidades de la calidad externa e interna y calidad en uso del software, teniendo en cuenta estos 7 indicadores (funcionalidad, confiabilidad, utilidad, eficiencia, capacidad de mantenimiento, portabilidad y calidad en uso), que se subdividen a su vez en varios indicadores; estas se pueden medir por métrica interna o externa.



Figura 2.13 Evaluación Interno, externa y Calidad de uso ISO/IEC 9126

Las definiciones se dan para cada característica y sub característica de calidad del software que influye en la calidad. Para cada característica y sub característica, la capacidad del software es determinada por un conjunto de atributos internos que pueden ser medidos. Las características y sub características se pueden medir externamente por la capacidad del sistema que contiene el software.

Funcionalidad

Funcionalidad es la capacidad del software de cumplir y proveer las funciones para satisfacer las necesidades explícitas e implícitas cuando es utilizado en condiciones específicas. A continuación, se muestra la característica de Funcionalidad y las sub características que cubre:



Figura 2.14 Característica de funcionalidad

La funcionalidad se divide en 5 criterios:

Adecuación: La capacidad del software para proveer un adecuado conjunto de funciones que cumplan las tareas y objetivos especificados por el usuario.

Exactitud: La capacidad del software para hacer procesos y entregar los resultados solicitados con precisión o de forma esperada.

Interoperabilidad: La capacidad del software de interactuar con uno o más sistemas específicos.

Seguridad: La capacidad del software para proteger la información y los datos de manera que los usuarios o los sistemas no autorizados no puedan acceder a ellos para realizar operaciones, y la capacidad de aceptar el acceso a los datos de los usuarios o sistemas autorizados

Conformidad de la funcionalidad: La capacidad del software de cumplir los estándares referentes a la funcionalidad.

Confiabilidad

La confiabilidad es la capacidad del software para asegurar un nivel de funcionamiento adecuado cuando es utilizado en condiciones específicas. En este caso la confiabilidad se amplía a sostener un nivel especificado de funcionamiento y no una función requerida.



Figura 2.151 Característica de Confiabilidad

La confiabilidad se divide en 4 criterios:

Madurez: La capacidad que tiene el software para evitar fallas cuando encuentra errores. Ejemplo, la forma como el software advierte al usuario cuando realiza operaciones en la unidad de diskett vacía, o cuando no encuentra espacio suficiente en el disco duro donde está almacenando los datos.

Tolerancia a errores: La capacidad que tiene el software para mantener un nivel de funcionamiento en caso de errores.

Recuperabilidad: La capacidad que tiene el software para restablecer su funcionamiento adecuado y recuperar los datos afectados en el caso de una falla.

Conformidad de la fiabilidad: La capacidad del software de cumplir a los estándares o normas relacionadas a la fiabilidad.

Usabilidad

La usabilidad es la capacidad del software de ser entendido, aprendido, y usado en forma fácil y atractiva. Algunos criterios de funcionalidad, fiabilidad y eficiencia afectan la usabilidad, pero para los propósitos de la ISO/IEC 9126 ellos no clasifican como usabilidad. La usabilidad está determinada por los usuarios finales y los usuarios indirectos del software, dirigidos a todos los ambientes, a la preparación del uso y el resultado obtenido.



Figura 2.16 Característica de Usabilidad

La usabilidad se divide en 5 criterios:

Entendimiento: La capacidad que tiene el software para permitir al usuario entender si es adecuado, y de una manera fácil como ser utilizado para las tareas y las condiciones particulares de la aplicación. En este criterio se debe tener en cuenta la documentación y de las ayudas que el software entrega.

Aprendizaje: La forma como el software permite al usuario aprender su uso. También es importante considerar la documentación.

Operabilidad: La manera como el software permite al usuario operarlo y controlarlo.

Atracción: La presentación del software debe ser atractiva al usuario. Esto se refiere a las cualidades del software para hacer más agradable al usuario, ejemplo, el diseño gráfico.

Conformidad de uso: La capacidad del software de cumplir los estándares o normas relacionadas a su usabilidad.

Eficiencia

La eficiencia del software es la forma del desempeño adecuado, de acuerdo a al número recursos utilizados según las condiciones planteadas. Se debe tener en cuenta otros aspectos como la configuración de hardware, el sistema operativo, entre otros.



Figura 2.17 Característica de Eficiencia

La eficiencia se divide en 3 criterios:

Comportamiento de tiempos: Los tiempos adecuados de respuesta y procesamiento, el rendimiento cuando realiza su función en condiciones específicas. Ejemplo, ejecutar el procedimiento más complejo del software y esperar su tiempo de respuesta, realizar la misma función, pero con más cantidad de registros.

Utilización de recursos: La capacidad del software para utilizar cantidades y tipos adecuados de recursos cuando este funciona bajo requerimientos o condiciones establecidas. Ejemplo, los recursos humanos, el hardware, dispositivos externos.

Conformidad de eficiencia: La capacidad que tiene el software para cumplir con los estándares o convenciones relacionados a la eficiencia.

Capacidad de Mantenimiento

La capacidad de mantenimiento es la cualidad que tiene el software para ser modificado. Incluyendo correcciones o mejoras del software, a cambios en el entorno, y especificaciones de requerimientos funcionales.



Figura 2.18 Característica de Mantenimiento

El mantenimiento se divide en 5 criterios:

Capacidad de ser analizado: La forma como el software permite diagnósticos de deficiencias o causas de fallas, o la identificación de partes modificadas.

Cambiabilidad: La capacidad del software para que la implementación de una modificación se pueda realizar, incluye también codificación, diseño y documentación de cambios.

Estabilidad: La forma como el software evita efectos inesperados para modificaciones del mismo.

Facilidad de prueba: La forma como el software permite realizar pruebas a las modificaciones sin poner el riesgo los datos.

Conformidad de facilidad de mantenimiento: La capacidad que tiene el software para cumplir con los estándares de facilidad de mantenimiento.

Portabilidad

La capacidad que tiene el software para ser trasladado de un entorno a otro.



Figura 2.19 Característica de portabilidad

La usabilidad se divide en 5 criterios:

Adaptabilidad: Es como el software se adapta a diferentes entornos especificados (hardware o sistemas operativos) sin que implique reacciones negativas ante el cambio. Incluye la escalabilidad de capacidad interna (Ejemplo: Campos en pantalla, tablas, volúmenes de transacciones, formatos de reporte, etc.).

Facilidad de instalación: La facilidad del software para ser instalado en un entorno específico o por el usuario final.

Coexistencia: La capacidad que tiene el software para coexistir con otro o varios softwares, la forma de compartir recursos comunes con otro software o dispositivo.

Reemplazabilidad: La capacidad que tiene el software para ser reemplazado por otro software del mismo tipo, y para el mismo objetivo. Ejemplo, la reemplazabilidad de una nueva versión es importante para el usuario, la propiedad de poder migrar los datos a otro software de diferente proveedor.

Conformidad de portabilidad: La capacidad que tiene el software para cumplir con los estándares relacionados a la portabilidad.

Calidad de Uso

Calidad en uso es la calidad del software que el usuario final refleja, la forma como el usuario final logra realizar los procesos con satisfacción, eficiencia y exactitud. La calidad en uso debe asegurar la prueba o revisión de todas las opciones que el usuario trabaja diariamente y los procesos que realiza esporádicamente relacionados con el mismo software.



Figura 2.20 Característica Calidad de uso

La calidad de uso se divide en 4 criterios:

Eficacia: La capacidad del software para permitir a los usuarios finales realizar los procesos con exactitud e integridad.

Productividad: La forma como el software permite a los usuarios emplear cantidades apropiadas de recursos, en relación a la eficacia lograda en un contexto específico de uso. Para una empresa es muy importante que el software no afecte a la productividad del empleado

Seguridad: Se refiere al que el Software no tenga niveles de riesgo para causar daño a las personas, instituciones, software, propiedad intelectual o entorno. Los riesgos son normalmente el resultado de deficiencias en la funcionalidad (Incluyendo seguridad), fiabilidad, usabilidad o facilidad de mantenimiento.

Satisfacción: La satisfacción es la respuesta del usuario a la interacción con el software, e incluye las actitudes hacia el uso del mismo.

A continuación, se describe un cuadro donde podemos resumir las características y cada uno de sus atributos, este cuadro le ayudara a visualizar el proceso de evaluación.

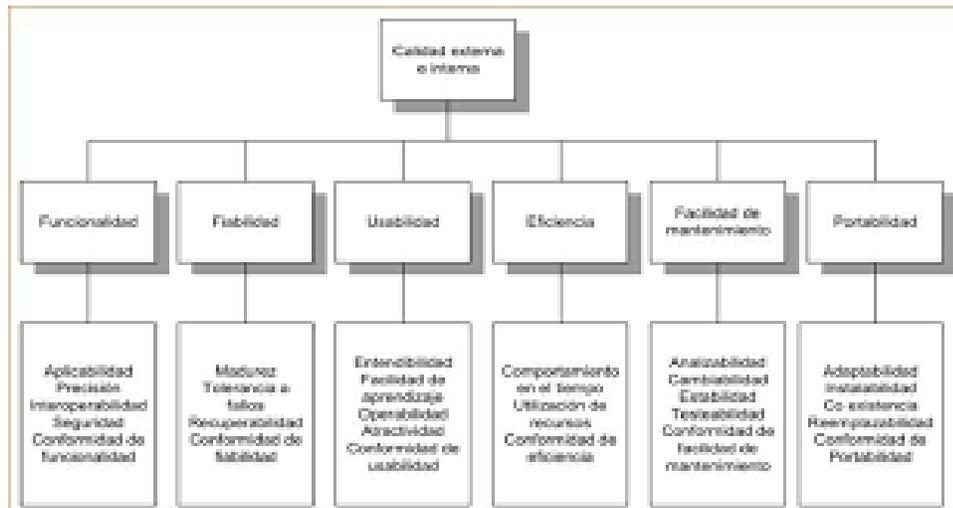


Figura 2.212 Resumen de la ISO/IEC 9126

2.8. Estudio Costo Beneficio COCOMO II

El modelo original COCOMO (Constructive Cost Model) se publicó por primera vez en 1981 por Barry Boehm y reflejaba las prácticas en desarrollo de software de aquel momento. En la década y media siguiente las técnicas de desarrollo software cambiaron drásticamente. Estos cambios incluyen el gasto de tanto esfuerzo en diseñar y gestionar el proceso de desarrollo software como en la creación del producto software, un giro total desde los main frame que trabajan con procesos batch nocturnos hacia los sistemas en tiempo real y un énfasis creciente en la reutilización de software ya existente y en la construcción de nuevos sistemas que utilizan componentes software a medida. (Boehm, 1981)

Estos y otros cambios hicieron que la aplicación del modelo COCOMO original empezara a resultar problemática. La solución al problema era reinventar el modelo para aplicarlo a los 90. Después de muchos años de esfuerzo

combinado entre USC-CSE1, IRUS y UC Irvine22 y las Organizaciones Afiliadas al Proyecto COCOMO II, el resultado es COCOMO II, un modelo de estimación de coste que refleja los cambios en la práctica de desarrollo de software profesional que ha surgido a partir de los años70. Este nuevo y mejorado COCOMO resultará de gran ayuda para los estimadores profesionales de coste software.

Por tanto, COCOMO II es un modelo que permite estimar el coste, esfuerzo y tiempo cuando se planifica una nueva actividad de desarrollo software.

El principal cálculo en el modelo COCOMO es el uso de la ecuación del esfuerzo para estimar el número de personas o de meses necesarios para desarrollar el proyecto. El resto de resultados del modelo se derivan de esta medida.

Por un lado, COCOMO define tres modos de desarrollo o tipos de proyectos:

- ✓ **Orgánico:** proyectos relativamente sencillos, menores de 50 KDLC líneas de código, en los cuales se tiene experiencia de proyectos similares y se encuentran en entornos estables.
- ✓ **Semi - acoplado:** proyectos intermedios en complejidad y tamaño (menores de 300 KDLC), donde la experiencia en este tipo de proyectos es variable, y las restricciones intermedias.
- ✓ **Empotrado:** proyectos bastante complejos, en los que apenas se tiene experiencia y se engloban en un entorno de gran innovación técnica. Además, se trabaja con unos requisitos muy restrictivos y de gran volatilidad.

Y por otro lado existen diferentes modelos que define COCOMO:

- ✓ **Modelo básico:** Se basa exclusivamente en el tamaño expresado en LDC.
- ✓ **Modelo intermedio:** Además del tamaño del programa incluye un conjunto de medidas subjetivas llamadas conductores de costes.

- ✓ **Modelo avanzado:** Incluye todo lo del modelo intermedio además del impacto de cada conductor de coste en las distintas fases de desarrollo.

La función básica que utilizan los tres modelos es:

$$E = a(KI)^b * m(X)$$

Dónde:

- ✓ a y b son constantes con valores definidos en cada submodelo.
- ✓ KI es la cantidad de líneas de código, en miles.
- ✓ m(X) Es un multiplicador que depende de 15 atributos.

El resultado se da en unidades salario/mes y horas-hombre.

2.8.1. Modelo básico.

Se utiliza para obtener una primera aproximación rápida del esfuerzo, y hace uso de la siguiente tabla de constantes para calcular distintos aspectos de costes:

Tabla 2.3 Estimación de esfuerzo

MODO	A	B	C	D
Orgánico	2.40	1.05	2.50	0.38
Semi libre	3.00	1.12	2.50	0.35
Rígido	3.60	1.20	2.50	0.32

Fuente: Grupo de Investigación de costos (Beltrán, 2008)

Estos valores son para las fórmulas:

- ✓ Personas necesarias por mes para llevar adelante el proyecto (MM) = a*(KI^b)
- ✓ Tiempo de desarrollo del proyecto (TDEV) = c*(MM^d)

- ✓ Personas necesarias para realizar el proyecto ($\text{CosteH} = \text{MM/TDEV}$)
- ✓ Costo total del proyecto ($\text{CosteM} = \text{CosteH} * \text{Salario medio}$ entre los programadores y analistas).

Se puede observar que a medida que aumenta la complejidad del proyecto (modo), las constantes aumentan de 2.4 a 3.6, que corresponde a un incremento del esfuerzo del personal. Hay que utilizar con mucho cuidado el modelo básico puesto que se obvian muchas características del entorno.

Atributos. Cada atributo se cuantifica para un entorno de proyecto. La escala es muy baja -bajo -nominal -alto -muy alto -extremadamente alto. Dependiendo de la calificación de cada atributo, se asigna un valor para usar de multiplicador en la fórmula (por ejemplo, si para un proyecto el atributo DATA es calificado como muy alto, el resultado de la fórmula debe ser multiplicado por 1000).

El significado de los atributos es el siguiente, según su tipo: Software

- ✓ RELY: garantía de funcionamiento requerida al software. Indica las posibles consecuencias para el usuario en el caso que existan defectos en el producto. Va desde la sola inconveniencia de corregir un fallo (muy bajo) hasta la posible pérdida de vidas humanas (extremadamente alto, software de alta criticidad).
- ✓ DATA: tamaño de la base de datos en relación con el tamaño del programa. El valor del modificador se define por la relación: D / K , donde D corresponde al tamaño de la base de datos en bytes y K es el tamaño del programa en cantidad de líneas de código.
- ✓ CPLX: representa la complejidad del producto.

Hardware

- ✓ TIME: limitaciones en el porcentaje del uso de la CPU.
- ✓ STOR: limitaciones en el porcentaje del uso de la memoria.
- ✓ VIRT: volatilidad de la máquina virtual.

- ✓ TURN: tiempo de respuesta requerido.

Personal

- ✓ ACAP: calificación de los analistas.
- ✓ AEXP: experiencia del personal en aplicaciones similares.
- ✓ PCAP: calificación de los programadores.
- ✓ VEXP: experiencia del personal en la máquina virtual.
- ✓ LEXP: experiencia en el lenguaje de programación a usar.

Proyecto

- ✓ MODP: uso de prácticas modernas de programación.
- ✓ TOOL: uso de herramientas de desarrollo de software.
- ✓ SCED: limitaciones en el cumplimiento de la planificación.

El valor de cada atributo, de acuerdo a su calificación, se muestra en la siguiente tabla:

Tabla 2.4 Valores de los factores de escala

Atributos	Valor					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extra alto
Atributos de software						
Fiabilidad	0.75	0.88	1.00	1.15	1.40	
Tamaños de Base de datos		0.94	1.00	1.08	1.16	
Complejidad	0.70	0.85	1.00	1.15	1.30	1.65
Atributos de hardware						
Restricciones de tiempo de ejecución			1.00	1.11	1.30	1.66

Restricciones de memoria virtual			1.00	1.06	1.21	1.56
Volatilidad de la máquina		0.87	1.00	1.15	1.30	
Tiempo de respuesta		0.87	1.00	1.07	1.15	
Atributos de persona						
Capacidad de análisis	1.46	1.19	1.00	0.86	0.71	
Experiencia en la aplicación	1.29	1.13	1.00	0.91	0.82	
Calidad de los programadores	1.42	1.17	1.00	0.86	0.70	
Experiencia en la máquina	1.21	1.10	1.00	0.90		
Experiencia en el lenguaje	1.14	1.07	1.00	0.95		
Atributos del proyecto						
Técnicas actualizadas de programación	1.24	1.10	1.00	0.91	0.82	
Utilización de herramientas de software	1.24	1.10	100	0.91	0.83	
Restricciones de tiempo de desarrollo	1.23	1.08	1.00	1.04	1.10	

Fuente: Modelos de estimación (Beltrán, 2008)

2.9. JAVA

Java es un lenguaje de programación creado por Sun Microsystems, (empresa que posteriormente fue comprada por Oracle) para poder funcionar en distintos tipos de procesadores. Su sintaxis es muy parecida a la de C o C++, e incorpora como propias algunas características que en otros lenguajes son extensiones: gestión de hilos, ejecución remota, etc.

El código Java, una vez compilado, puede llevarse sin modificación alguna sobre cualquier máquina, y ejecutarlo. Esto se debe a que el código se ejecuta sobre una máquina hipotética o virtual, la Java Virtual Machine, que se encarga de interpretar el código (ficheros compilados .class) y convertirlo a código particular de la CPU que se esté utilizando (siempre que se soporte dicha máquina virtual).

2.9.1. Conceptos Previos de Poo

Java es un lenguaje orientado a objetos (OO), por lo que, antes de empezar a ver qué elementos componen los programas Java, conviene tener claros algunos conceptos de la programación orientada a objetos (POO).

2.9.2. Concepto de Clase y Objeto

El elemento fundamental a la hora de hablar de programación orientada a objetos es el concepto de objeto en sí, así como el concepto abstracto de clase. Un objeto es un conjunto de variables junto con los métodos relacionados con éstas. Contiene la información (las variables) y la forma de manipular la información (los métodos). Una clase es el prototipo que define las variables y métodos que va a emplear un determinado tipo de objeto, es la definición abstracta de lo que luego supone un objeto en memoria.

Poniendo un símil fuera del mundo de la informática, la clase podría ser el concepto de coche, donde nos vienen a la memoria los parámetros que definen un coche (dimensiones, cilindrada, maletero, etc), y las operaciones que podemos hacer con un coche (acelerar, frenar, adelantar, estacionar). La idea abstracta de coche que tenemos es lo que equivaldría a la clase, y la representación concreta de coches concretos (por ejemplo, Peugeot 307, Renault Megane, Volkswagen Polo...) serían los objetos de tipo coche.

2.9.3. Concepto de Campo, Método y Constructor

Toda clase u objeto se compone internamente de constructores, campos y/o métodos. Veamos qué representa cada uno de estos conceptos: un campo es un elemento que contiene información relativa a la clase, y un método es un elemento que permite manipular la información de los campos. Por otra parte, un constructor es un elemento que permite reservar memoria para almacenar los campos y métodos de la clase, a la hora de crear un objeto de la misma.

2.9.4. Concepto de Herencia y Polimorfismo

Con la herencia podemos definir una clase a partir de otra que ya exista, de forma que la nueva clase tendrá todas las variables y métodos de la clase a partir de la que se crea, más las variables y métodos nuevos que necesite. A la clase base a partir de la cual se crea la nueva clase se le llama superclase.

Por ejemplo, podríamos tener una clase genérica Animal, y heredamos de ella para formar clases más específicas, como Pato, Elefante, o León. Estas clases tendrían todo lo de la clase padre Animal, y además cada una podría tener sus propios elementos adicionales.

Una característica derivada de la herencia es que, por ejemplo, si tenemos un método dibuja (Animal a), que se encarga de hacer un dibujo del animal que se le pasa como parámetro, podremos pasarle a este método como parámetro tanto un Animal como un Pato, Elefante, o cualquier otro subtipo directo o indirecto de Animal. Esto se conoce como polimorfismo.

2.9.5. Modificadores de Acceso

Tanto las clases como sus elementos (constructores, campos y métodos) pueden verse modificados por lo que se suelen llamar modificadores de acceso, que indican hasta dónde es accesible el elemento que modifican. Tenemos tres tipos de modificadores: • privado: el elemento es accesible únicamente dentro de la

clase en la que se encuentra. • protegido: el elemento es accesible desde la clase en la que se encuentra, y además desde las subclases que hereden de dicha clase. • público: el elemento es accesible desde cualquier clase.

2.9.6. Clases Abstractas e Interfaces

Mediante las clases abstractas y los interfaces podemos definir el esqueleto de una familia de clases, de forma que los subtipos de la clase abstracta o la interfaz implementen ese esqueleto para dicho subtipo concreto. Por ejemplo, volviendo con el ejemplo anterior, podemos definir en la clase Animal el método dibuja () y el método imprime (), y que Animal sea una clase abstracta o un interfaz.

Vemos la diferencia entre clase, clase abstracta e interfaz con este supuesto: • En una clase, al definir Animal tendríamos que implementar el código de los métodos dibuja () e imprime (). Las subclases que hereden de Animal no tendrían por qué implementar los métodos, a no ser que quieran redefinirlos para adaptarlos a sus propias necesidades.

- ✓ En una clase abstracta podríamos implementar los métodos que nos interese, dejando sin implementar los demás (dejándolos como métodos abstractos). Dichos métodos tendrían que implementarse en las clases hijas.
- ✓ En un interfaz no podemos implementar ningún método en la clase padre, y cada clase hija tiene que hacer sus propias implementaciones de los métodos. Además, las clases hijas podrían implementar otros interfaces.

2.10. PHP

PHP es un lenguaje interpretado del lado del servidor que surge dentro de la corriente denominada código abierto (Open Source). Se caracteriza por su potencia, versatilidad, robustez y modularidad. Al igual que ocurre con tecnologías similares, los programas son integrados directamente dentro del código HTML. En este libro se explicará en detalle la sintaxis y el funcionamiento

de este lenguaje, de momento se realiza a continuación una breve comparativa con las otras tecnologías del lado del servidor descritas previamente.

Comparado con ASP, la principal ventaja de PHP es su carácter multiplataforma. Por otro lado, los programas en ASP resultan más lentos y pesados, y también menos estables. En los entornos Microsoft la ventaja de ASP es que los servidores web de Microsoft soportan directamente ASP sin necesidad de ninguna instalación adicional

Señalar también la existencia de herramientas que permiten convertir programas desarrollados en ASP al lenguaje PHP, una de las más conocidas es asp2php.

Comparando el lenguaje PHP con el lenguaje Perl, utilizado habitualmente en la programación CGI, puede decirse que PHP fue diseñado para desarrollo de scripts orientados a web, mientras que Perl fue diseñado para hacer muchas más cosas y debido a esto, se hace muy complicado. La sintaxis de PHP es menos confusa y más estricta, pero sin perder la flexibilidad.

En comparación con ColdFusion, PHP es más rápido y eficiente para tareas complejas de programación, además PHP resulta más estable y usa una menor cantidad de recursos. Por el contrario, ColdFusion posee un mejor gestor de errores, un buen motor de búsquedas, abstracciones de bases de datos y un gran número de funcionalidades para el procesamiento de fechas. Finalmente, ColdFusion no está disponible para todas las plataformas.

En definitiva, PHP es uno de los lenguajes más utilizados actualmente en el desarrollo de aplicaciones web y viene experimentando un constante crecimiento en su nivel de utilización en Internet. Este libro trata de humildemente contribuir a continuar con el proceso de difusión de esta tecnología.

2.11. MATLAB

MATLAB es una de las muchas sofisticadas herramientas de computación disponibles en el comercio para resolver problemas de matemáticas, tales como Maple, Mathematica y MathCad. A pesar de lo que afirman sus defensores, ninguna de ellas es “la mejor”. Todas tienen fortalezas y debilidades. Cada una permitirá efectuar cálculos matemáticos básicos, pero difieren en el modo como manejan los cálculos simbólicos y procesos matemáticos más complicados, como la manipulación de matrices. Por ejemplo, MATLAB es superior en los cálculos que involucran matrices, mientras que Maple lo supera en los cálculos simbólicos. El nombre mismo de MATLAB es una abreviatura de Matrix Laboratory, laboratorio matricial. En un nivel fundamental, se puede pensar que estos programas son sofisticadas calculadoras con base en una computadora. Son capaces de realizar las mismas funciones que una calculadora científica, y muchas más. Si usted tiene una computadora en su escritorio, descubrirá que usará MATLAB en lugar de su calculadora incluso para la más simple de sus aplicaciones matemáticas, por ejemplo, para el balance de su chequera. En muchas clases de ingeniería, la realización de cálculos con un programa de computación matemático como MATLAB sustituye la programación de computadoras más tradicional. Esto no significa que el lector no deba aprender un lenguaje de alto nivel como C++ o FORTRAN, sino que los programas como MATLAB se han convertido en una herramienta estándar para ingenieros y científicos.

Dado que MATLAB es tan fácil de usar, muchas tareas de programación se llevan a cabo con él. Sin embargo, MATLAB no siempre es la mejor herramienta para usar en una tarea de programación. El programa destaca en cálculos numéricos, especialmente en los relacionados con matrices y gráficas, pero usted no querrá escribir un programa de procesamiento de palabras en MATLAB. C++ y FORTRAN son programas de propósito general y serían los programas de elección para aplicaciones grandes como los sistemas operativos o el software

de diseño. (De hecho, MATLAB, que es un programa grande de aplicación, se escribió originalmente en FORTRAN y después se rescribió en C, precursor de C++.) Por lo general, los programas de alto nivel no ofrecen acceso fácil a la Graficación, que es una aplicación en la que destaca MATLAB. El área principal de interferencia entre MATLAB y los programas de alto nivel es el “procesamiento de números”: programas que requieren cálculos repetitivos o el procesamiento de grandes cantidades de datos. Tanto MATLAB como los programas de alto nivel son buenos en el procesamiento de números. Por lo general, es más fácil escribir un programa que “procese números” en MATLAB, pero usualmente se ejecutará más rápido en C++ o FORTRAN. La única excepción a esta regla son los cálculos que involucran matrices: puesto que MATLAB es óptimo para matrices, si un problema se puede formular con una solución matricial, MATLAB lo ejecuta sustancialmente más rápido que un programa similar en un lenguaje de alto nivel.

MATLAB está disponible en versiones tanto profesional como estudiantil. Es probable que en el laboratorio de cómputo de su colegio o universidad esté instalada la versión profesional, pero disfrutará tener la versión estudiantil en casa. MATLAB se actualiza de manera regular; este texto se basa en MATLAB 7. Si utiliza MATLAB 6 podrá observar algunas diferencias menores entre éste y MATLAB 7. En versiones anteriores a MATLAB 5.5 existen diferencias sustanciales.

2.11.1. Aplicación de Matlab

En las disciplinas de ingeniería, ciencias y programación de computadoras, es importante tener un enfoque consistente para resolver los problemas técnicos. El enfoque que se plantea a continuación es útil en cursos tan distintos como química, física, termodinámica y diseño de ingeniería. También se aplica a las ciencias sociales, como economía y sociología. Otros autores quizá formulen sus

esquemas de resolución de problemas de forma ligeramente diferente, pero todos tienen el mismo formato básico:

- ✓ Plantear el problema. m En esta etapa con frecuencia es útil hacer un dibujo. m Si no tiene una comprensión clara del problema, es improbable que pueda resolverlo.
- ✓ Describir los valores de entrada (conocidos) y las salidas (incógnitas) que se requieren. m Tenga cuidado de incluir las unidades conforme describe los valores de entrada y salida. El manejo descuidado de las unidades con frecuencia lleva a respuestas incorrectas. m Identifique las constantes que tal vez requiera en el cálculo, como la constante de los gases ideales y la aceleración de la gravedad. m Si es apropiado, en un dibujo escriba los valores que haya identificado o agrúpelos en una tabla.
- ✓ Desarrollar un algoritmo para resolver el problema. En aplicaciones de cómputo, es frecuente que esto se logre con una prueba de escritorio. Para ello necesitará. m Identificar cualesquiera ecuaciones que relacionen los valores conocidos con las incógnitas. m Trabajar con una versión simplificada del problema, a mano o con calculadora.
- ✓ Resolver el problema. En este libro, esta etapa involucra la creación de una solución con MATLAB.
- ✓ Probar la solución. m ¿Sus resultados tienen sentido físico? m ¿Coinciden con los cálculos de la muestra? m ¿La respuesta es la que se pedía en realidad? m Las gráficas con frecuencia son formas útiles de verificar que los cálculos son razonables.

Si utiliza en forma consistente un enfoque estructurado de resolución de problemas, como el que se acaba de describir, descubrirá que los problemas tipo “narración” son mucho más fáciles de resolver. El ejemplo 1.1 ilustra esta estrategia de resolución de problemas.

2.12. Base de Datos MySQL

Base de datos MySQL, es un sistema para la gestión de base de datos con más de seis millones de instalaciones en el mundo, y que por lo tanto es el más extendido dentro de las aplicaciones relacionadas. MySQL funciona como software libre dentro del esquema de licencias GNU GPL.

MySQL es muy utilizado en aplicaciones web como WordPress, PhpBB, MediaWikio Drupal, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura, lo que hace a MySQL ideal para este tipo de aplicaciones.

Todos los planes de hosting de Hostinet disponen de MySQL para poder crear y gestionar el contenido en bases de datos, como Mat Lab, Magic Draw y UWE.

2.12.1. Características de MySQL

Estas no son las únicas características de MySQL:

Arquitectura Cliente y Servidor: MySQL basa su funcionamiento en un modelo cliente y servidor. Es decir, clientes y servidores se comunican entre sí de manera diferenciada para un mejor rendimiento. Cada cliente puede hacer consultas a través del sistema de registro para obtener datos, modificarlos, guardar estos cambios o establecer nuevas tablas de registros, por ejemplo.

Compatibilidad con SQL: SQL es un lenguaje generalizado dentro de la industria. Al ser un estándar MySQL ofrece plena compatibilidad por lo que si has trabajado en otro motor de bases de datos no tendrás problemas en migrar a MySQL.

Vistas: Desde la versión 5.0 de MySQL se ofrece compatibilidad para poder configurar vistas personalizadas del mismo modo que podemos hacerlo en otras bases de datos SQL. En bases de datos de gran tamaño las vistas se hacen un recurso imprescindible.

Procedimientos almacenados. MySQL posee la característica de no procesar las tablas directamente, sino que a través de procedimientos almacenados es posible incrementar la eficacia de nuestra implementación.

Desencadenantes. MySQL permite además poder automatizar ciertas tareas dentro de nuestra base de datos. En el momento que se produce un evento otro es lanzado para actualizar registros o optimizar su funcionalidad.

Vistas: Desde la versión 5.0 de MySQL se ofrece compatibilidad para poder configurar vistas personalizadas del mismo modo que podemos hacerlo en otras bases de datos SQL. En bases de datos de gran tamaño las vistas se hacen un recurso imprescindible.

Procedimientos almacenados. MySQL posee la característica de no procesar las tablas directamente, sino que a través de procedimientos almacenados es posible incrementar la eficacia de nuestra implementación.

Desencadenantes. MySQL permite además poder automatizar ciertas tareas dentro de nuestra base de datos. En el momento que se produce un evento otro es lanzado para actualizar registros o optimizar su funcionalidad.

Transacciones. Una transacción representa la actuación de diversas operaciones en la base de datos como un dispositivo. El sistema de base de registros avala que todos los procedimientos se establezcan correctamente o ninguna de ellas. En caso por ejemplo de una falla de energía, cuando el monitor falla u ocurre algún otro inconveniente, el sistema opta por preservar la integridad de la base de datos resguardando la información.

CAPITULO III

MARCO APLICATIVO

MARCO APLICATIVO

En este capítulo se dispuso la práctica de las especificaciones mencionadas en los anteriores capítulos, respecto al desarrollo del modelo de predicción del crecimiento del índice de cáncer de cuello uterino mediante algoritmos genéticos, para los cual se delimita, estandariza y procesa la población de estudio, posteriormente se modela el software para hacer más amigable para al usuario con XP, así mismo de califica la investigación con métricas de calidad como ISO 9126 y COCOMO II. Por otro lado, se muestra la utilidad del modelo de predicción, para el caso específico de la Ciudad de El Alto.

3.1. Introducción

El modelo de predicción es un procedimiento de inteligencia artificial, por el cual se podrá visibilizar el índice de crecimiento del Cáncer de Cuello Uterino (CACU) en la Ciudad de El Alto. Existen varios modelos de predicción, así como las herramientas que se utilizan dentro de lo que refiere a Redes Neuronales e Inteligencia Artificial. Es así que dentro del presente modelo se entrenó mediante Redes Neuronales la modelación del algoritmo genético.

En consecuencia, un paso metodológico para la construcción de un modelo predictivo entrenado, requiere de datos históricos, que fueron en esta ocasión facilitados por SEDES El Alto, desde su página en la web. Los cuales fueron

normalizados de acuerdo a los requerimientos para la predicción del presente trabajo.

A continuación, se muestra en la Figura el esquema del modelo de predicción con algoritmos genéticos. Ver figura 3.1.

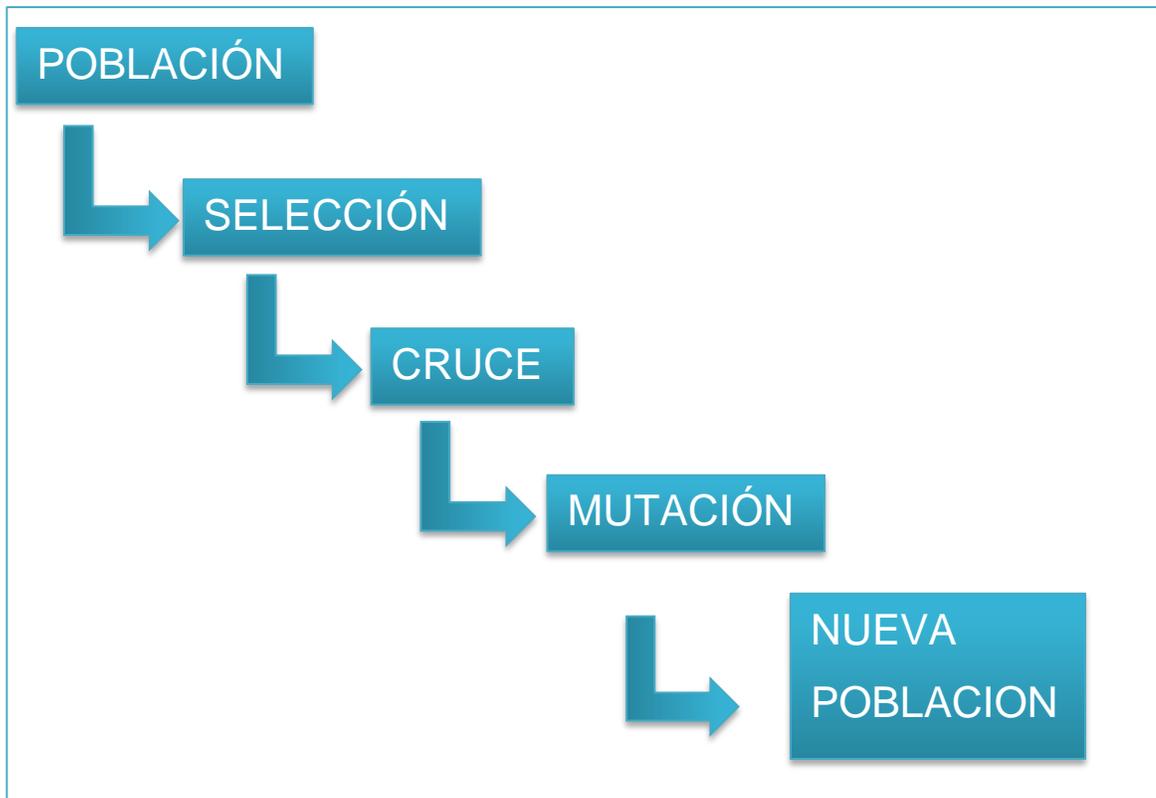


Figura 3.1 Modelo de predicción CACUGEN

Fuente: Elaboración propia

3.1.1. Población

Según los datos del INE las proyecciones poblacionales al 2020 señalan que el municipio de El Alto tiene 943.558 habitantes, mostrando un crecimiento de 31.352 personas respecto al 2017. De las cuales la población masculina alcanza a 458.447 habitantes respecto a la femenina, que llega a 485.111 personas, vale decir que el número de mujeres supera al de hombres. En este sentido se

muestra el índice de crecimiento pronosticado al 2020 en el cuadro 3.1, así mismo podemos observar en el cuadro 3.2 la relación entre la población masculina y femenina. Véase el cuadro 3.1 y 3.2 respectivamente.

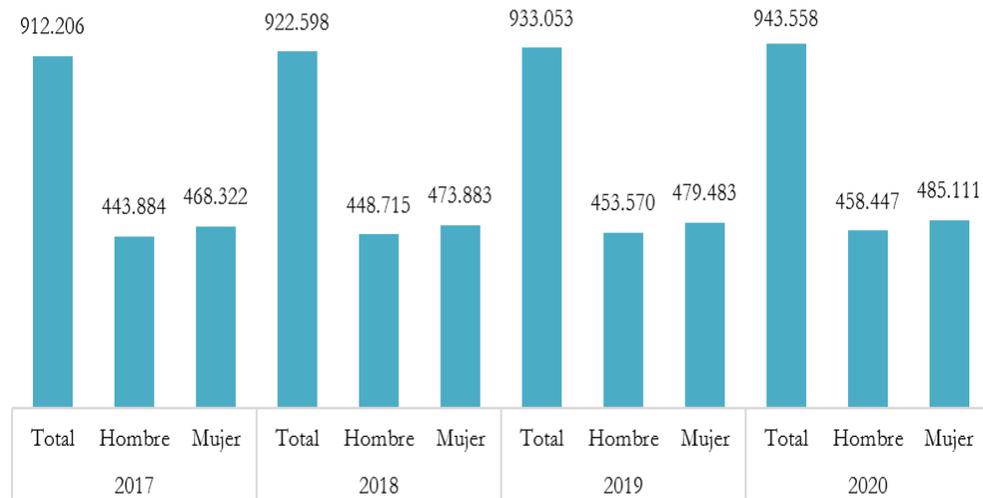


Figura 3.2 Crecimiento de la población en la Ciudad de El Alto

Fuente: Instituto Nacional de Estadística – Revisión 2014

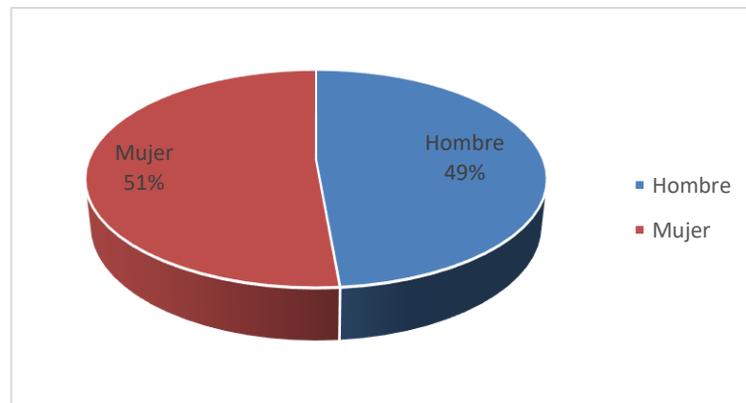


Figura 3.3 Municipio de El Alto: Progresiones de Población por sexo, 2019

Fuente: Instituto Nacional de Estadística – Revisión 2014

De este universo de 479.483 habitantes mujeres el 2019, se tomó como población de a 1.082 mujeres, vale decir que se consideró a un total de población entre las gestiones 2015 al 2019 estudio a 8.141 mujeres. Siendo el total de mujeres de la Ciudad de El Alto que realizaron la prueba del PAP e IVAA según los datos del SEDES El Alto. (Página oficial del SNIS - VE)

Para su mejor interpretación se clasificaron por grupos etarios que son:

- ✓ Grupo 1 que corresponde a mujeres entre 15 y 19 años.
- ✓ Grupo 2 que corresponde a mujeres entre 20 y 34 años.
- ✓ Grupo 3 que corresponde a mujeres entre 35 y 49 años.
- ✓ Grupo 4 que corresponde a mujeres mayores e igual a 50 años.

Tabla 3.1 Tabla de la población en Grupos Etarios y por gestión

	2015	2016	2017	2018	2019	TOTAL
GRUPO 1	85	41	86	91	35	338
GRUPO 2	912	682	924	982	520	4020
GRUPO 3	602	603	609	648	399	2861
GRUPO 4	194	195	196	209	128	922
TOTAL	1793	1521	1815	1930	1082	8141

Fuente: Elaboración Propia.

Por otro, lado se consideró como una variable independiente la fase en la que se presentaron los índices de cáncer de cuello uterino (CACU), que son Displasia, Carcinoma in situ, Cáncer Invasivo localizado, Nódulos linfáticos y Metástasis.

Tabla 3.2 Clasificación de Fases del Cáncer

FASE	DESCRIPCION
Fase 1	La displasia, indica la anormalidad es un cambio en las características de las células.
Fase 2	El “carcinoma in situ”, se usa cuando el examen microscópico revela células con ciertas características del cáncer, esto es, cambios en los núcleos celulares.
Fase 3	El Cáncer invasivo, es cuando el crecimiento celular anormal llega a áreas que están por debajo del tejido de origen.
Fase 4	Los nódulos linfáticos, se produce una invasión cada vez mayor y la destrucción de los tejidos adyacentes. A menudo el cáncer se extiende a los ganglios linfáticos regionales que drenan el área.
Fase 5	La metástasis, se produce cuando las células cancerosas se propagan a través de la sangre o del sistema linfático y afectar a otros órganos.

Fuente: Elaboración propia

3.1.2. Selección

La selección de la población se tomará, en relación a la aptitud total de la población. Con estos valores, se puede construir una ruleta en la que las áreas de los sectores circulares son proporcionales a la aptitud relativa de cada ejemplar. Ver Tabla 3.3 y figura 3.4.

Tabla 3.3 Población inicial, cromosomas y aptitud

ITEM	DECIMAL	BINARIO	APTITUD	APTITUD RELATIVA
1	1973	11110110101	3892729	30
2	1521	10111110001	2313441	17
3	1815	11100010111	3294225	25
4	1930	11110001010	3724900	28
TOTAL	8141		13225295	100

Fuente: Elaboración propia

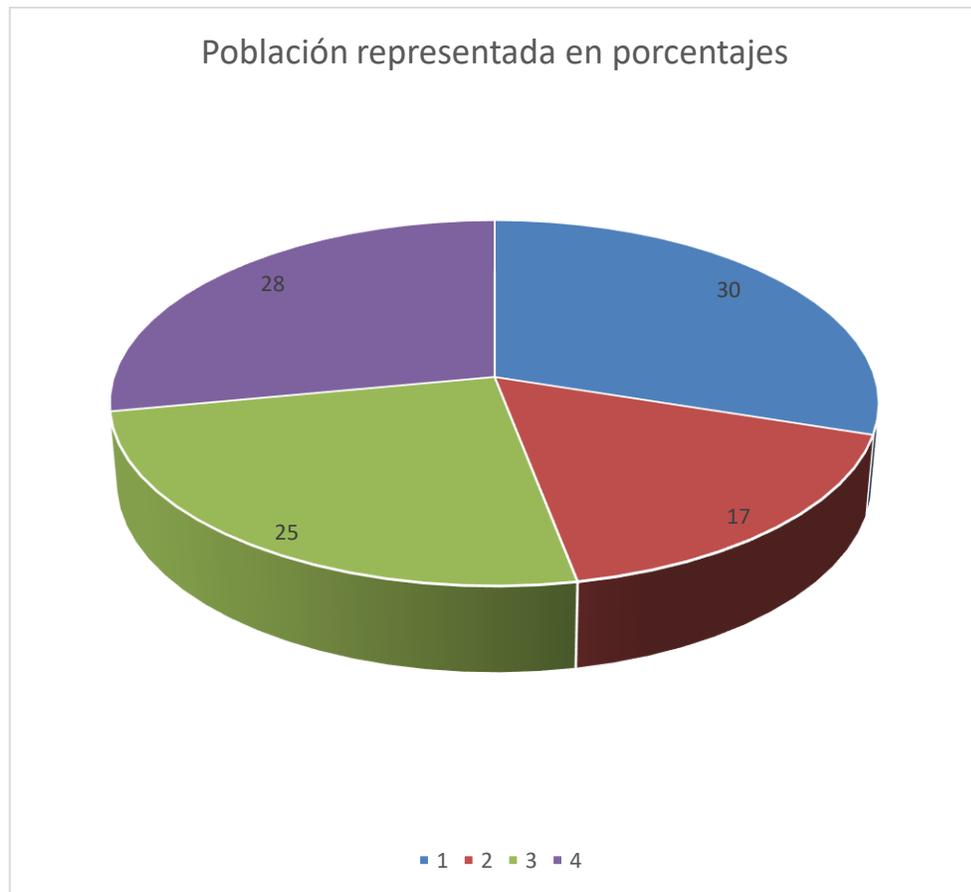


Figura 3.4 Ruleta de Selección del más Apto

Fuente: Elaboración propia

3.1.3. Cruce y Mutación

Considerando que la ruleta seleccione el ítem 1 (1973) y el ítem 4 (1930). Aleatoriamente se determina un punto de cruce y se procede a intercambiar el material genético de los dos padres, para configurar sus dos hijos. Hecho esto, a continuación, se introduce una mutación. En este caso, se complementa el valor del gen, escogido aleatoriamente, y se obtienen los dos ejemplares que ingresan a la nueva población. En nuestro caso, aparece el número 1973 [11110110101] y se mantiene el 1930 [11110001010]. El cual se mostrará de en el siguiente cuadro. Donde se puede la selección de la mejor población, tomando

en cuenta el punto de cruce, así como los cruces y mutaciones que se mostraran para obtener la nueva población. Véase figura (3.5).

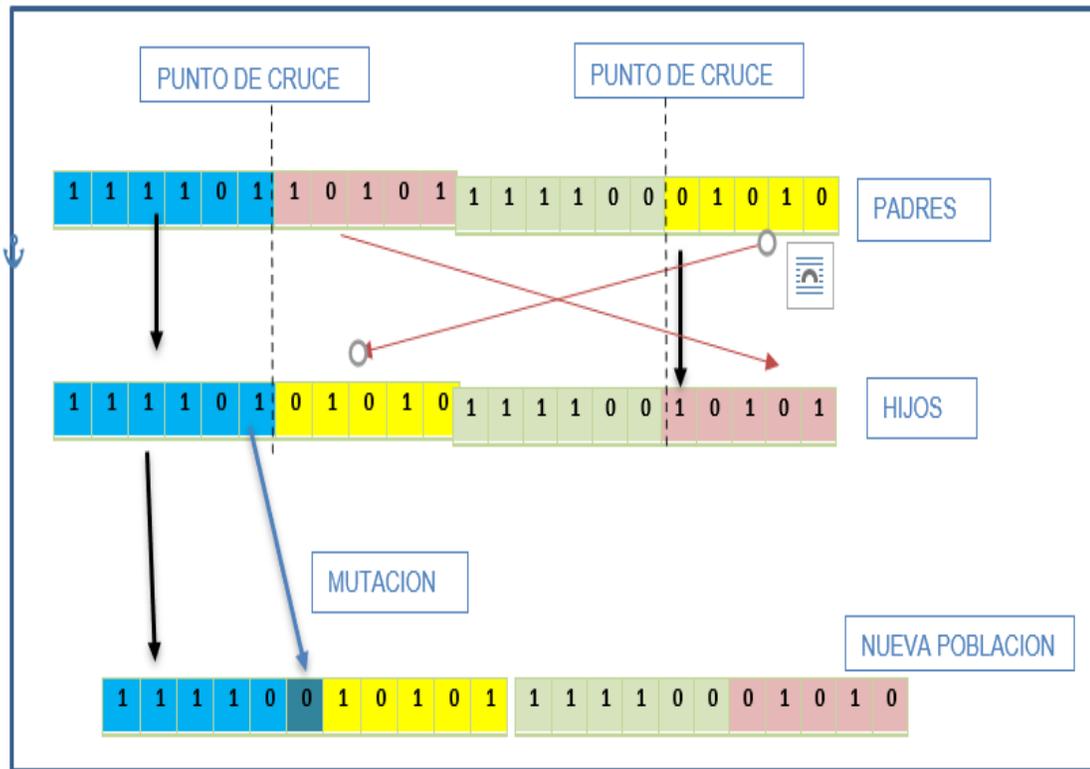


Figura 3.5 punto de cruce para mutación e hijos

Fuente: Elaboración propia

3.1.4. Nueva Población

Durante cada iteración se crea una población completa, por lo que la nueva población reemplaza directamente a la anterior.

3.2. Análisis de Requerimiento

Al momento de iniciar la tesis no se cuenta con un modelo de predicción del índice de crecimiento del Cáncer de Cuello Uterino en la Ciudad de El Alto, por lo que se inició con los siguientes puntos tomando en cuenta la metodología XP que

son: Planeación, Plan de iteraciones, Plan de entregas. El cual mostrara los índices de crecimiento de CACU durante los próximos 5 años, vale decir del 2020 al 2025.

Se trata de una aplicación que muestra los índices de crecimiento del Cáncer de Cuello Uterino (CACU), en un formato de consulta tomado de una población de 8.141 mujeres que fueron diagnosticadas positivas con el padecimiento del CACU, de la Ciudad de El Alto, datos que son clasificados en grupos etarios y por gestión.

3.2.1. Requerimientos Funcionales

En base a la entrevista con los especialistas y pacientes con Cáncer de Cuello Uterino se tiene los siguientes requerimientos:

Tabla 3.4 Requerimiento funcionales de CACUGEN

REQUERIMIENTOS	DESCRIPCIÓN
1	El sistema debe ser diseñado con un entorno amigable y de fácil uso
2	Los usuarios no deben tener la necesidad de logearse
3	Los datos deben ser cargados por el usuario.
4	El usuario debe seleccionar los datos para su consulta.
5	El usuario puede visibilizar los pronósticos tanto en porcentajes como gráficos
6	El sistema debe correr sobre la plataforma Windows 10 o posterior
7	La base de datos debe estar sobre la plataforma MySQL
8	Disponibilidad de 24 /7

Fuente: Elaboración propia

3.2.2. Requerimientos no Funcionales

De acuerdo a lo descrito en los requerimientos funcionales y la de los usuarios se tiene los siguientes requerimientos no funcionales.

Tabla 3.5 Rendimiento no Funcional

REQUERIMIENTOS	DESCRIPCIÓN
1	El sistema debe ser capaz de procesar N datos por segundo.
2	El sistema debe ser capaz de estandarizar los datos.
3	El sistema debe entrenar con diferentes algoritmos para optimizar la consulta.
4	El sistema debe seleccionar los datos y predecir los nuevos casos de cáncer de cuello uterino.
5	La base de datos se puede respaldar de acuerdo a requerimientos.
6	El sistema debe contar con manuales de usuario estructurados adecuadamente.
7	El sistema debe formar con interfaces porcentuales y gráficas bien formadas.

Fuente: Elaboración propia

3.3. Modelado en Base a la Metodología XP

A continuación se detallará las actividades planificadas para la implementación de la metodología XP, en sus fases de exploración, planificación e iteración, así como las reglas en cada iteración.

3.3.1. Fase I: Exploración

De acuerdo a la metodología en esta etapa se plantean las historias de usuario, de la misma manera se realiza una planificación sobre las historias de usuario para priorizar en la primera entrega.

De este modo se denota los avances alcanzados en la primera iteración, contando con las historias recogidas y las tareas respectivas, que se obtuvieron de entrevistas con los especialistas y programadores.

Tabla 3.6 Historia 1

NUMERO HISTORIA 1	
GRUPO HISTORIA	USUARIO: MEDICOS
FECHA	6 DE MARZO DE 2020
CANTIDAD DE USUARIOS	4
TIEMPO ESTIMADO	2 HORAS
DESCRIPCION	
<ul style="list-style-type: none">✚ Todo usuario podrá ingresar a sistema de predicción sin ninguna restricción.✚ Podrá seleccionar el algoritmo y la gestión de predicción.✚ El usuario podrá visualizar en la pantalla los datos de predicción, así como en un gráfico.	

Fuente: Elaboración propia

Tabla 3.7 Conexión del usuario

Tarea	
Número de tarea: 1.1	Número de historia: 1
Nombre de tarea: Interfaz de acceso	
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha de inicio:	Fecha fin:
Responsable: Jaime Marcial Hidalgo	
Descripción: Desarrollar interfaz para el usuario donde pueda ingresar al sistema. Mostrar el avance del prototipo.	

Fuente: Elaboración propia

Tabla 3.8 Historia 2

NUMERO HISTORIA 2	
GRUPO HISTORIA	USUARIO: PROGRAMADOR
FECHA	16 DE MARZO DE 2020
CANTIDAD DE USUARIO	1
TIEMPO ESTIMADO	4 HORAS
DESCRIPCION	
<p>El administrador debe conseguir los datos sobre de cáncer de cuello uterino de la Ciudad de El Alto y seleccionarlos.</p> <p>Debe migrar los datos obtenidos en una base de datos.</p> <p>Se mostrará los datos obtenidos.</p>	

Fuente: Elaboración propia

Tabla 3.9 Interfaz datos

Tarea	
Número de tarea: 2.1	Número de historia: 2
Nombre de tarea: Selección de Datos	
Tipo de tarea: Base de datos	Puntos estimados:
Fecha de inicio:	Fecha fin:
Responsable: Jaime Marcial Hidalgo	
Descripción: Seleccionar los datos obtenidos de las instituciones para conocer la situación actual. Diseñar y modelar la base de datos con la información obtenida. Mostrar el avance del prototipo.	

Fuente: Elaboración propia

Tabla 3.10 Historia 3

NUMERO HISTORIA 3	
GRUPO HISTORIA	USUARIO: ALGORITMOS
FECHA	16 DE MARZO DE 2020
CANTIDAD DE USUARIO	1
TIEMPO ESTIMADO	4 HORAS
DESCRIPCION El programador debe entrenar los algoritmos genéticos con población generada y seleccionar el mejor algoritmo... Se mostrará los datos obtenidos.	

Fuente: Elaboración propia

Tabla 3.11 Algoritmos

Tarea	
Número de tarea: 3.1	Número de historia: 3
Nombre de tarea: Entrenamiento de algoritmos	
Tipo de tarea: Seleccionar algoritmo	Puntos estimados:
Fecha de inicio:	Fecha fin:
Responsable: Jaime Marcial Hidalgo	
Descripción: Entrenar los algoritmos en Matlab, para seleccionar la mejor predicción y nueva generación. Se mostrará los datos obtenidos.	

Fuente: Elaboración propia

3.3.2. Fase II: Planificación

Considerando que se agruparon las historias en especialistas y usuarios, donde estos últimos se agruparon por grupos etarios. Así mismo se tomó en cuenta que el nivel de conocimiento y experiencias vividas entre grupos varia, por lo que se consideró dos iteraciones con el fin de abstraer la información necesaria sin perder el horizonte del trabajo y tomando en cuenta la hipótesis.

3.3.2.1. Estimación de Esfuerzo

Tabla 3.12 Estimación de Esfuerzo

N°	Usuario	Historia de usuario	Pts.
1	Acceso al sistema	Interfaz de acceso	2
		Conexión al sistema	2
2	Cargado de datos	Diseño carga visibilizarían de datos	2
3	Generación de predicción	Diseño de módulo de predicción	2
4	Generación de gráficos	Conexión del módulo de gráficos	2

Fuente: Elaboración propia

3.3.2.2. Planificación de Iteraciones

Tabla 3.13 Planificación de iteraciones

N°	Iteraciones	Historias	Días	Observación
1	Primera	Interfaz de acceso	2	
		Conexión al sistema	1	
		Diseño cargado de datos	2	
		Diseño del visor	2	
		Conexión con el módulo de gráficos	2	
2	Segunda	Diseño de interfaz	5	
		Generación de predicción	3	
		Generación de gráficos	3	
		Generación de reportes	5	

Fuente: Elaboración propia

3.3.3. Fase III: Iteraciones

Para que el cliente tenga una visión de lo que quiere como producto terminado, relatado en las historias de usuario, se da una idea del cómo se verá el sistema a través de diseños de ventana a la que se quiere llegar.

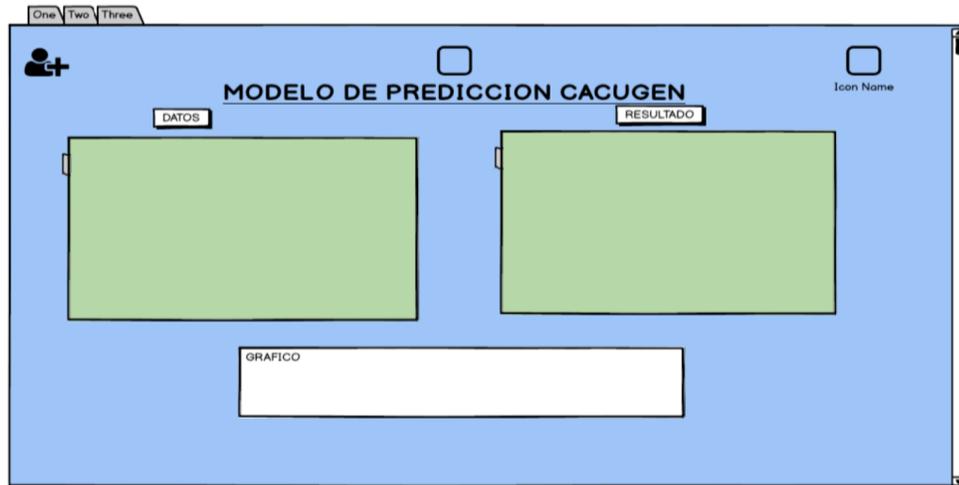


Figura 3.7 Diseño rápido del sistema

Fuente: Elaboración propia

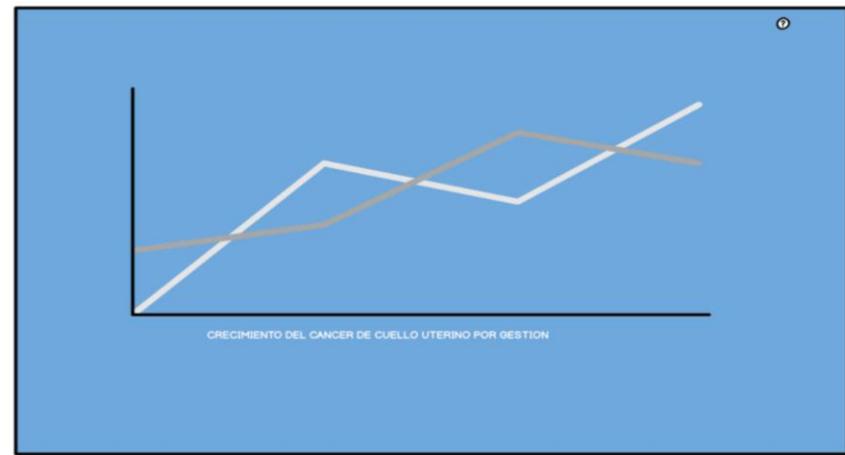


Figura 3.8 Gráficos que mostrara el sistema

Fuente: Elaboración propia

De acuerdo a la planificación de iteraciones propuesto para el proyecto, seleccionamos las historias de usuario que se implementarán, para cada iteración se debe realizar las siguientes tareas.

- **Especificación de tarjetas CRC**, Son las clases que interactúan en el sistema, además poseen responsabilidades y colaboraciones con otras clases.
- **Especificación de tarjeta de ingeniería**, Son tareas que se van a realizar por cada clase identificada.
- **Diagrama de datos**, Modelado de la base de datos.
- **Diagrama de clase**, Atributos y métodos que tiene la clase.

3.3.3.1. Primera Iteración

3.3.3.1.1. Historia 1: Acceso al Sistema

Tarjetas CRC

Tabla 3.14 Tarjera CRC, Acceso al Modelo

Acceso al Modelo	
Responsabilidades	Colaboradores
Ingreso libre Carga libre Calcular libre	

Fuente: Elaboración propia

Tarjeta de Ingeniería

Tarea de ingeniería	
Tarea número: 1	Historia de usuario: Acceso al sistema
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha inicio:	Fecha final:
Programador o responsable: Jaime Hidalgo	
Descripción: Desarrolla códigos de entrada al Modelo de Predicción CACUGEN	

Fuente: Elaboración propia

Caso de Uso Acceso al Modelo de Predicción CACUGEN

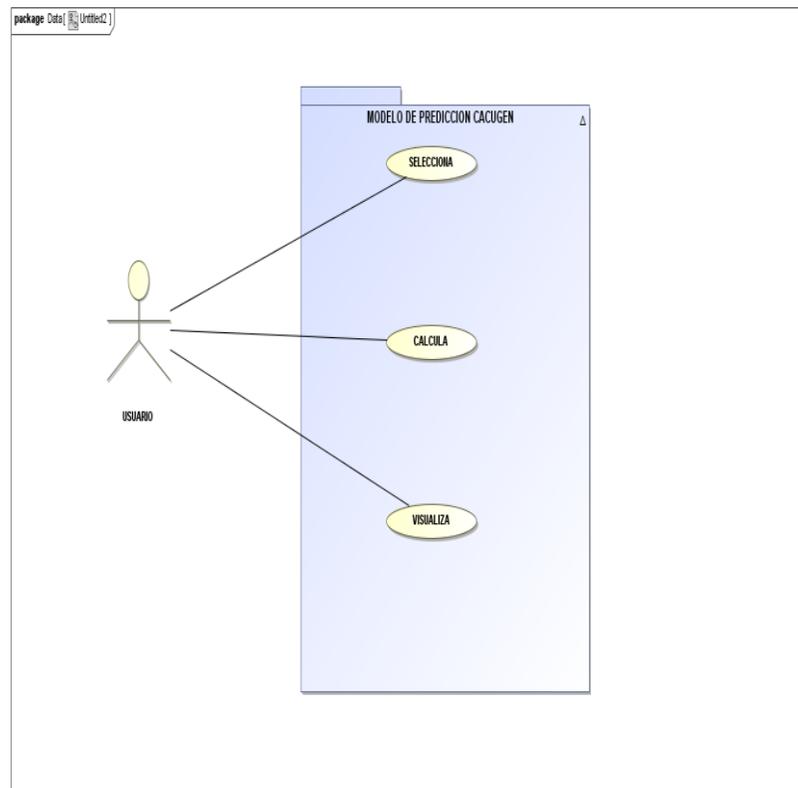


Figura 3.9 Ingreso al Modelo de Predicción CACUGEN

Fuente: Elaboración propia

Caso de uso Selección

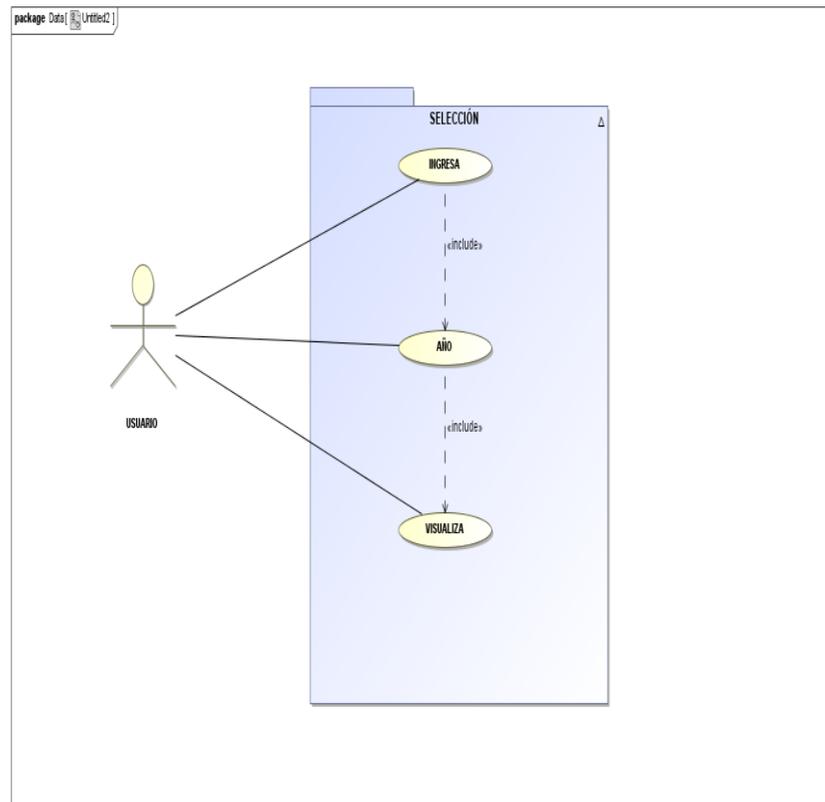


Figura 3.10 Caso de Uso Selecciona

Fuente: Elaboración propia

3.3.3.1.2. Historia 2: Recolección de Datos

Tarjetas CRC

Tabla 3.15 Tarjeta CRC: Recolección de Datos

Recolección de Datos	
Responsabilidades	Colaboradores
Recolectar datos Historia Estandarizar datos	Excel Xampp MySQL

Fuente: Elaboración propia

Tarjeta de Ingeniería

Tarea de ingeniería	
Tarea número: 1	Historia de usuario: Recolección de datos
Tipo de tarea: Recolección	Puntos estimados:
Fecha inicio:	Fecha final:
Programador o responsable: Jaime Hidalgo	
Descripción: Recolectar datos de las instituciones y estandarizar los mismos para el desarrollo del <u>software</u>	

Fuente: Elaboración propia

3.3.3.1.3. Historia 3: Entrenamiento de Algoritmos

Tarjetas CRC

Tabla 3.16 Tarjeta CRC: Entrenamiento de algoritmos

Entrenamiento de algoritmos	
Responsabilidades	Colaboradores
Entrenamiento de algoritmos genéticos	Matlab Weka
Entrenamiento de algoritmos mediante árboles.	

Fuente: Elaboración propia

Tarjeta de Ingeniería

Tarea de ingeniería	
Tarea número: 1	Historia de usuario: Entrenamiento de algoritmos
Tipo de tarea: Entrenamiento	Puntos estimados:
Fecha inicio:	Fecha final:
Programador o responsable: Jaime Hidalgo	
Descripción: Entrenamiento de algoritmos genéticos mediante Matlab. Entrenamiento de algoritmos de algoritmos por clasificación con la ayuda de Weka.	

Fuente: Elaboración propia

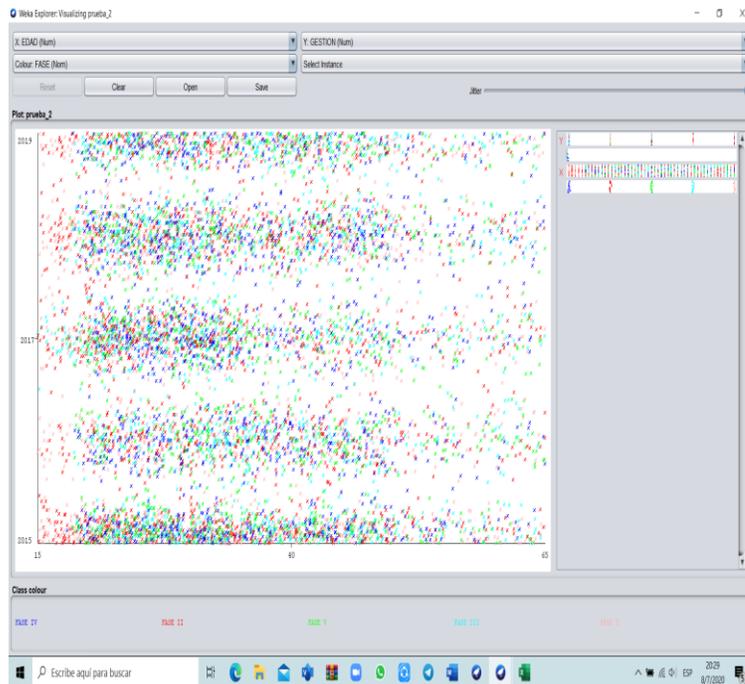


Figura 3.11 Relación Edad, Gestión y Fase

Fuente: Elaboración propia

Variables - pobCruz

A x pobMut x pobInt x pobCruz x

205x10 double

	1	2	3	4	5	6	7	8	9	10	11	12
1	1	0	1	0	1	0	0	1	0	1		
2	1	1	0	0	0	0	1	0	1	1		
3	1	1	0	1	1	1	1	1	0	1		
4	1	1	0	1	1	1	1	0	0	1		
5	1	1	1	1	0	1	0	0	1	0		
6	1	1	1	1	0	1	0	0	0	1		
7	1	1	0	1	0	0	1	0	1	0		
8	1	1	1	1	1	1	1	0	1	0		
9	1	1	0	1	1	1	1	0	1	1		
10	1	1	1	1	0	1	1	1	1	0		
11	1	1	0	1	1	0	1	0	1	1		
12	1	0	1	1	0	1	1	1	0	0		
13	1	1	0	1	0	0	1	1	0	1		
14	1	1	0	1	1	0	0	1	0	0		
15	1	1	1	1	1	0	1	0	1	0		
16	1	1	1	0	1	1	1	0	1	0		
17	1	1	1	1	0	1	0	1	0	0		
18	1	1	1	1	1	0	0	0	1	1		
19	1	1	0	1	1	0	0	0	1	1		
20	1	0	1	0	1	1	1	1	1	0		
21	1	1	0	0	0	0	1	1	1	1		
22	1	1	0	1	0	1	0	1	1	0		
23	1	1	0	1	0	1	0	0	1	0		
24	1	0	1	1	0	0	1	1	0	0		
25	1	1	0	1	1	1	1	1	1	0		
26	1	1	1	1	1	0	1	1	0	1		
27	1	1	0	0	1	1	0	0	1	1		

Figura 3.12 Población de cruce generada

Fuente: Elaboración propia

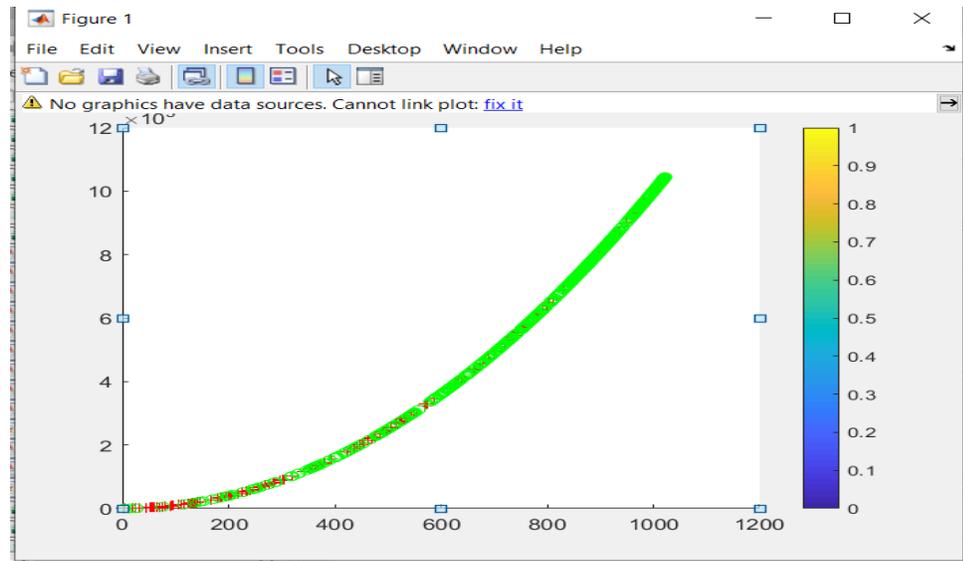


Figura 3.13 Entrenamiento en MATLAB

Fuente: Elaboración propia

3.3.3.2. Pruebas de Aceptación

3.3.3.2.1. Historia 1: Acceso al Sistema

Tabla 3.17 Caso de prueba: Acceso al sistema

CASO DE PRUEBA	
Código: 1	N° Historia de Usuario: 1
Historia de Usuario: Acceso al sistema	
Condiciones de ejecución: Los usuarios pueden ingresar sin restricción	
Entrada/Pasos de ejecución: Abrir la ventana del Módulo de predicción CACUGEN y cargar los datos.	
Resultado esperado: Acceder a la presentación de modelo, poder cargar datos de las anteriores gestiones y poder calcular la predicción de las diferentes gestiones.	
Evaluación de la prueba: La prueba se concluyó satisfactoriamente.	

Fuente: Elaboración propia



Figura 3.14 Ventana principal

Fuente: Elaboración propia

3.3.3.2.2. Historia 2: Recolección de Datos

Tabla 3.18 Caso de prueba: Recolección de datos

CASO DE PRUEBA	
Código: 2	N° Historia de Usuario: 2
Historia de Usuario: Recolección de Datos	
Condiciones de ejecución: El usuario cuenta con datos estandarizados para el sistema.	
Entrada/Pasos de ejecución: Cargar los datos en un base de datos en MySQL.	
Resultado esperado: Realizar consultas sobre la base de tos estandarizados.	
Evaluación de la prueba: La prueba se concluyó satisfactoriamente.	

Fuente: Elaboración propia

The screenshot shows the phpMyAdmin interface for a MySQL database named 'cacugen'. The selected table is 'cancer'. The table structure is as follows:

GESTION	SEXO	EDAD	FASE
2015	FEMENINO	27	FASE IV
2015	FEMENINO	17	FASE II
2015	FEMENINO	20	FASE II
2015	FEMENINO	24	FASE V
2015	FEMENINO	29	FASE III
2015	FEMENINO	51	FASE III
2015	FEMENINO	26	FASE III
2015	FEMENINO	33	FASE IV
2015	FEMENINO	25	FASE III
2015	FEMENINO	31	FASE III
2015	FEMENINO	27	FASE IV
2015	FEMENINO	19	FASE I
2015	FEMENINO	23	FASE V
2015	FEMENINO	62	FASE II
2015	FEMENINO	40	FASE IV
2015	FEMENINO	30	FASE II
2015	FEMENINO	26	FASE V
2015	FEMENINO	47	FASE III
2015	FEMENINO	21	FASE II
2015	FEMENINO	31	FASE III
2015	FEMENINO	25	FASE III
2015	FEMENINO	48	FASE I
2015	FEMENINO	17	FASE II
2015	FEMENINO	21	FASE V
2015	FEMENINO	37	FASE V

The interface includes a sidebar with a tree view of databases and tables, a top navigation bar with various actions (Examinar, Estructura, SQL, etc.), and a bottom status bar showing the current page (1 of 25 rows).

Figura 3.15 Base de datos

Fuente: Elaboración propia

3.3.3.2.3. Historia 3: Entrenamiento de Algoritmos

Tabla 3.19 Caso de prueba: Entrenamiento de algoritmos.

CASO DE PRUEBA	
Código: 3	N° Historia de Usuario: 3
Historia de Usuario: Entrenamiento de algoritmos	
Condiciones de ejecución: Entrenamiento de algoritmos genéticos en Matlab. Entrenamiento de algoritmos de Clasificación en Weka	
Entrada/Pasos de ejecución: Mostrar resultados de los entrenamientos en Matlab y Weka	
Resultado esperado: Definir mejor algoritmo de predicción para el modelo de predicción CACUGEN	
Evaluación de la prueba: La prueba se concluyó satisfactoriamente.	

Fuente: Elaboración propia

Tabla 3.20 Resultados de entrenamiento de algoritmos

Algoritmo	Mean absolute error	Root mean squared error	Relative absolute error	Root relative squared error
RandomForest	0.0003	0.006	0.0625%	1.1968%
DecisionStump	0.1441	0.2686	28.8217%	53.716%
MultilayerPerceptron	0.0011	0.0014	0.2256%	0.2748%
PART	0.0012	0.0334	0.2473%	6.6715%

Fuente: Elaboración propia

Por lo que se muestra la figura de una red neuronal que se entrenó con MultilayerPerceptron

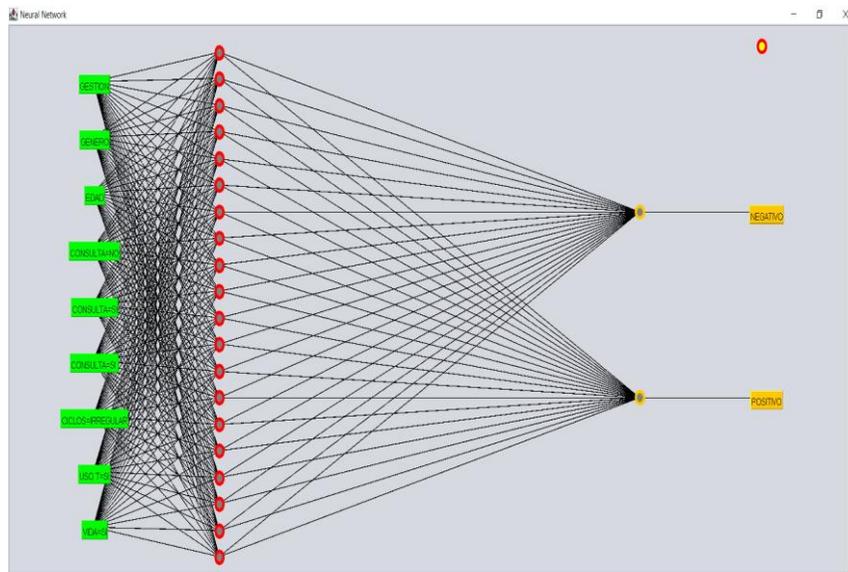


Figura 3.16
Algoritmo

MultilayerPerceptron

Fuente: Elaboración propia

3.3.3.3. Segunda Iteración

3.3.3.3.1. Historia 1: Predicción del Sistema

Tarjetas CRC

Tabla 3.21 Tarjeta CRC: Predicción del sistema

Predicción del Sistema	
Responsabilidades	Colaboradores
Generar la predicción del índice de crecimientos de CACU. Determinar los valores probables de crecimiento	Modelo de predicción CACUGEN

Fuente: Elaboración propia

Tarjeta de Ingeniería

Tarea de ingeniería	
Tarea número: 1	Historia de usuario: Predicción del sistema
Tipo de tarea: Modelado	Puntos estimados:
Fecha inicio:	Fecha final:
Programador o responsable: Jaime Hidalgo	
Descripción: Hacer correr el modelo de predicción CACUGEN, y determinar los índices de crecimiento del cáncer de cuello uterino.	

Fuente: Elaboración propia

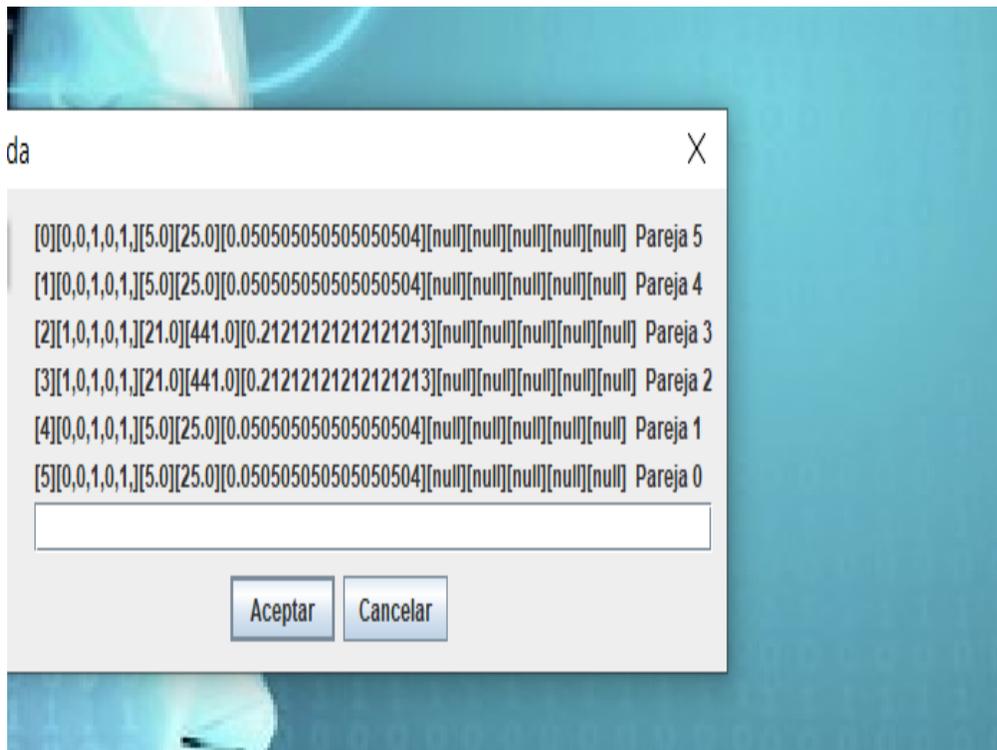


Figura 3.17 Predicción del sistema

Fuente: Elaboración propia

3.3.3.3.2. Historia 2: Generación de Predicción

Tarjetas CRC

Tabla 3.22 Tarjeta CRC: generación de predicción

Generación de predicción	
Responsabilidades	Colaboradores
Mostrar la predicción del índice de cáncer de cuello uterino de barras.	Modelo de predicción CACUGEN

Fuente: Elaboración propia

Tarjeta de Ingeniería

Tarea de ingeniería	
Tarea número: 1	Historia de usuario: Generación de predicción en gráficos
Tipo de tarea: Modelado	Puntos estimados:
Fecha inicio:	Fecha final:
Programador o responsable: Jaime Hidalgo	
Descripción: Se muestra el gráfico en barrar del índice de crecimiento del cáncer de cuello uterino.	

Fuente: Elaboración propia

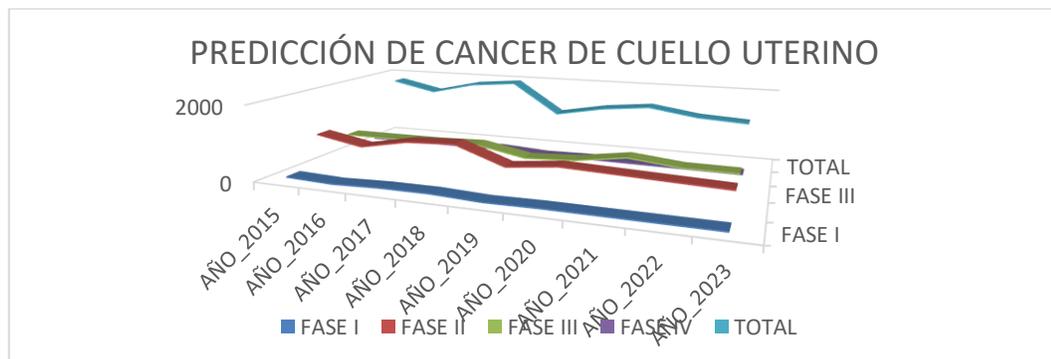


Figura 3.18 Predicción del cáncer de cuello uterino

Fuente: Elaboración propia

3.3.3.4. Pruebas de Aceptación

3.3.3.4.1. Historia 1: Predicción del Sistema

Tabla 3.23 Caso de prueba: Predicción del Sistema

CASO DE PRUEBA	
Código: 4	N° Historia de Usuario: 4
Historia de Usuario: Predicción del Sistema	
Condiciones de ejecución: El sistema	
Entrada/Pasos de ejecución: Seleccionar el algoritmo y la gestión para predecir	
Resultado esperado: El sistema predice los nuevos casos de cáncer de cuello uterino	
Evaluación de la prueba: La prueba se concluyó satisfactoriamente.	

Fuente: Elaboración propia



Figura 3.19 Funcionamiento del Modelo

Fuente: Elaboración propia

3.3.3.4.2. Historia 2: Generación de Predicción

CASO DE PRUEBA	
Código: 5	N° Historia de Usuario: 5
Historia de Usuario: Generación de predicción gráficamente.	
Condiciones de ejecución: El sistema genera nuevas predicciones en gráficos	
Entrada/Pasos de ejecución: En la pantalla muestra la predicción de cáncer y sus porcentajes	
Resultado esperado: Visualiza la predicción en porcentajes.	
Evaluación de la prueba: La prueba se concluyó satisfactoriamente.	

Fuente: Elaboración propia

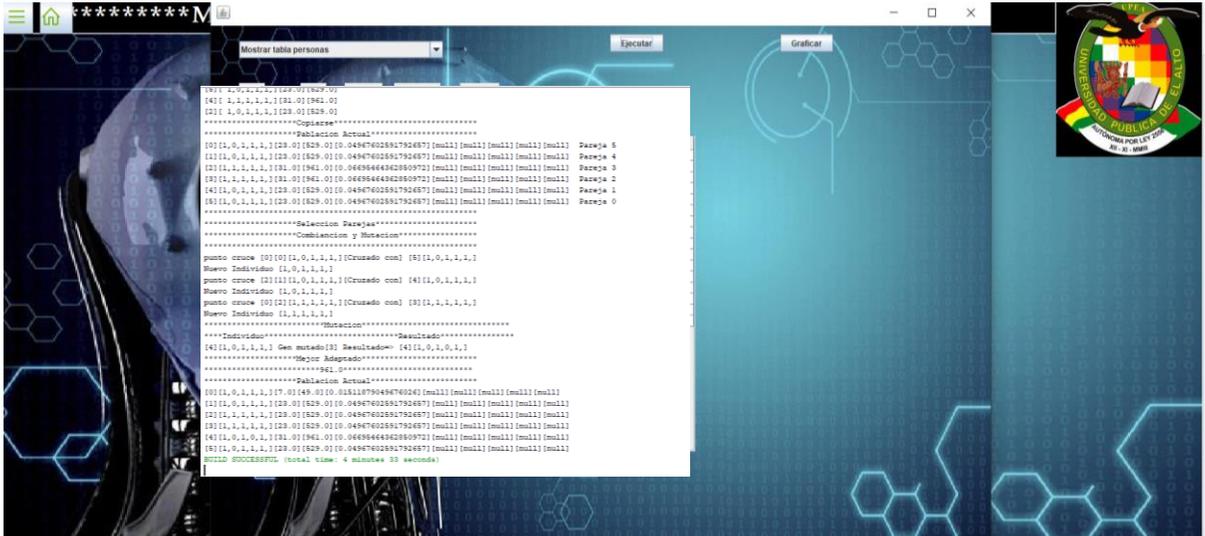


Figura 3.20 Visibilizarían de la población final

Fuente: Elaboración propia

3.4. Implementación del Modelo

En la implementación del modelo predictivo de Cáncer de Cuello Uterino se toman los datos históricos, los cuales son estandarizados, considerando los grupos etario y las gestiones, estos datos son visibles. Análisis de las etapas de funcionamiento del modelo con respecto a la base datos y los usuarios. Desarrollo del algoritmo genético con las funciones de selección u optimización, función cruce, función mutación y función generación de la nueva población o índice de crecimiento de las siguientes gestiones. Así como la compilación de los mismos y finalmente observar los resultados que se espera para poder dar una respuesta a la hipótesis planteada en la presente tesis.

3.4.1. Datos

Los datos con que cuenta el SEDES El Alto, se encuentran en la plataforma virtual, el cual contiene varias columnas dentro de las tablas, que muestran todos los servicios que prestados los centros de salud, hospitales y redes de salud en la Ciudad de El Alto. De la misma manera muestra que dentro de la Ciudad de El Alto sólo 3 redes atienden a mujeres con cáncer de cuello uterino para una población aproximadamente de 45.771 mujeres en la Ciudad de El Ato. Se verifica que de esta población un porcentaje menor realiza su control de PAP e IVAA , así mismos con la problemáticas del país este número de vio reducido en las últimas gestiones Véase la Figura (3.9).

COD_FORM	PROVINCIA	RED_DE_SALUD	CODMUN	ESTABLECIMIENTO	AMBITO	NIVEL	SUBSECTOR	TIPO	INSTITUCION	MES	SEMANA	COD_GRUP	GESTION	MES	SEM	COD_DEPTO	SEDES	PROVINCIA	COD_AF
TOTAL_GRAL										GRUPO_ETAREO									
SEDES	MUNICIPIO	VARIABLE								< 15	15 - 19	20 - 34	35 - 49	50 y +	Total General				
LA PAZ	EL ALTO	a. DIU: Usuarías nuevas										9	194	109	1	313			
		b. DIU: Usuarías continuas										3	103	74	1	181			
		c. Número de DIU insertados										9	197	110	1	317			
		d. Inyectable Trimestral: Usuarías Nuevas								2	265	2345	624	5	3241				
		e. Inyectable Trimestral: Usuarías Continuas								1	384	6306	2763	21	9475				
		f. Número de inyectables trimestrales suministrados								3	649	8651	3387	26	12716				
		g. Condón Masculino: Usuarías (os) Nuevas (os)										207	2057	945	64	3273			
		h. Condón Masculino: Usuarías (os) Continuas (os)										169	6141	4014	618	10942			
		i. Condón Femenino: Usuarías Nuevas										1	4	1	6				
		j. Condón Femenino: Usuarías Continuas											27	10	37				
		k. Número de condones entregados										3613	102804	63837	9618	179872			
		l. Píldora: Usuarías Nuevas										76	776	305	3	1160			
		ll. Píldora: Usuarías Continuas										23	523	265	1	812			
		m. Número de ciclos entregados										256	3481	1532	12	5281			
		n. Implante subdérmico: Usuarías Nuevas										204	1554	444	1	2203			
		ñ. Implante subdérmico: Usuarías Continuas										33	475	164		672			
		o. Número de implantes insertados										205	1683	470	1	2359			
		p. Métodos naturales: Usuarías Nuevas								10	388	2490	919	41	3848				
		q. Métodos naturales: Usuarías Continuas								2	68	628	304	4	1006				
		r. AQV femenino											207	116	323				
		s. AQV masculino											18	18	36				
		t. Número de mujeres con píldora anticonceptiva oral de emergencia										23	265	83	5	366			
		u. Número de personas que recibieron orientación en Anticoncepción								47	3415	19052	9024	2021	33559				
		v. Número de mujeres con muestras de PAP tomadas								17	1341	19748	14421	5049	40576				
		w. Número de mujeres con resultados de PAP positivos								7	22	397	264	92	782				
		x. Número de mujeres con resultados de PAP positivos tratadas										3	39	35	21	98			
		y. Número de mujeres con IVAA								4	79	2022	2377	713	5195				
		z. Número de mujeres con IVAA positivos										6	123	135	36	300			
		z1. Número de mujeres con IVAA positivos tratadas										3	27	29	16	75			
		z2. Número de mujeres con examen clínico de mama positivos (sospechoso de nódulo neoplásico)										1	17	5	2	25			
		z3. Número de mujeres con estudios mamaográficos positivos												1	1				
EL ALTO	Total									93	11455	182344	106785	18373	319050				

Figura 3.21 Servicios atendidos en la gestión 2019 Sedes El Alto

Fuente: SEDES-VID, 2020

3.4.1.2. Datos Estandarizados

De todos los datos que se pudieron recabar se empezó a discriminar de acuerdo a las necesidades del proyecto por lo que se consideró los datos referidos a la detección de Cáncer de Cuello Uterino entre las que contamos con los PAP positivos e IVAA positivos, así mismo se tomó en cuenta las gestiones, género y grupos etarios en los que se presenta mayor número de mujeres con CACU positivo. Los mismo que se estandarizaron colocando en código binario para su aplicación como se observa. Ver Tabla (3.10):

	A	B	C	D	E	F	G	H	I
1	GESTION	GENERO	EDAD	CONSULTA	CICLOS	USO T	VIDA	CANCER	
2	2015	F	31	SI	REGULAR	NO	NO	NEGATIVO	
3	2015	F	20	NO	REGULAR	SI	SI	POSITIVO	
4	2015	F	64	SI	IRREGULAR	NO	NO	NEGATIVO	
5	2015	F	42	NO	REGULAR	NO	SI	NEGATIVO	
6	2015	F	27	SI	REGULAR	NO	NO	NEGATIVO	
7	2015	F	29	NO	IRREGULAR	SI	SI	POSITIVO	
8	2015	F	28	NO	IRREGULAR	SI	SI	POSITIVO	
9	2015	F	29	NO	IRREGULAR	SI	SI	POSITIVO	
10	2015	F	34	SI	REGULAR	NO	NO	NEGATIVO	
11	2015	F	15	SI	IRREGULAR	NO	NO	NEGATIVO	
12	2015	F	35	NO	IRREGULAR	SI	SI	POSITIVO	
13	2015	F	28	SI	REGULAR	SI	NO	NEGATIVO	
14	2015	F	25	SI	REGULAR	SI	NO	NEGATIVO	
15	2015	F	30	SI	REGULAR	SI	NO	NEGATIVO	
16	2015	F	27	SI	REGULAR	SI	NO	NEGATIVO	
17	2015	F	28	NO	REGULAR	SI	SI	POSITIVO	
18	2015	F	22	SI	REGULAR	NO	NO	NEGATIVO	
19	2015	F	65	NO	REGULAR	NO	SI	NEGATIVO	
20	2015	F	27	NO	REGULAR	NO	SI	NEGATIVO	
21	2015	F	30	SI	IRREGULAR	SI	SI	POSITIVO	
22	2015	F	21	NO	REGULAR	SI	SI	POSITIVO	
23	2015	F	20	NO	IRREGULAR	SI	SI	POSITIVO	
24	2015	F	28	SI	REGULAR	SI	NO	NEGATIVO	
25	2015	F	40	SI	REGULAR	NO	NO	NEGATIVO	
26	2015	F	26	NO	IRREGULAR	SI	SI	POSITIVO	
27	2015	F	31	NO	REGULAR	SI	SI	POSITIVO	
28	2015	F	45	SI	IRREGULAR	NO	NO	NEGATIVO	
29	2015	F	16	SI	IRREGULAR	SI	SI	POSITIVO	
30	2015	F	41	NO	REGULAR	NO	SI	NEGATIVO	
31	2015	F	22	SI	REGULAR	NO	NO	NEGATIVO	
32	2015	F	23	SI	REGULAR	SI	NO	NEGATIVO	
33	2015	F	20	NO	REGULAR	SI	SI	POSITIVO	
34	2015	F	15	SI	REGULAR	NO	NO	NEGATIVO	
35	2015	F	23	NO	IRREGULAR	SI	SI	POSITIVO	
36	2015	F	34	SI	REGULAR	NO	NO	NEGATIVO	
37	2015	F	25	SI	REGULAR	NO	NO	NEGATIVO	
38	2015	F	20	SI	REGULAR	NO	NO	NEGATIVO	
39	2015	F	53	NO	REGULAR	NO	SI	NEGATIVO	
40	2015	F	40	SI	IRREGULAR	NO	NO	NEGATIVO	
41	2015	F	43	SI	IRREGULAR	SI	SI	POSITIVO	
42	2015	F	16	NO	REGULAR	NO	SI	NEGATIVO	
43	2015	F	23	NO	REGULAR	NO	SI	NEGATIVO	
44	2015	F	26	NO	REGULAR	SI	SI	POSITIVO	
45	2015	F	32	NO	IRREGULAR	SI	SI	POSITIVO	
46	2015	F	36	NO	IRREGULAR	SI	SI	POSITIVO	
47	2015	F	18	NO	IRREGULAR	NO	SI	POSITIVO	
48	2015	F	41	SI	IRREGULAR	NO	NO	NEGATIVO	

Tabla 3.24 Datos normalizados de casos de CACU

Fuente: Elaboración propia

Cabe aclarar que se cuentan con **8141** registros, que se recabaron entre las gestiones 2015 y 2019, así mismo se debe considerar que la gestión 2019 sólo se atendieron 10 meses y es por eso que se observa una notable reducción de casos con CACU en esta gestión.

3.4.2. Etapas de Funcionamiento del Modelo

El modelo de predicción se iniciará de manera directa sin ninguna prohibición, se insertarán los datos, para luego estandarizarlos para el sistema, esto se almacenará en su memoria para luego seleccionar el algoritmo que mejor funcione con los datos para luego mostrar de manera visual y gráfica los resultados. Finalmente el usuario podrá interpretar estos resultados para tomar las posteriores decisiones.

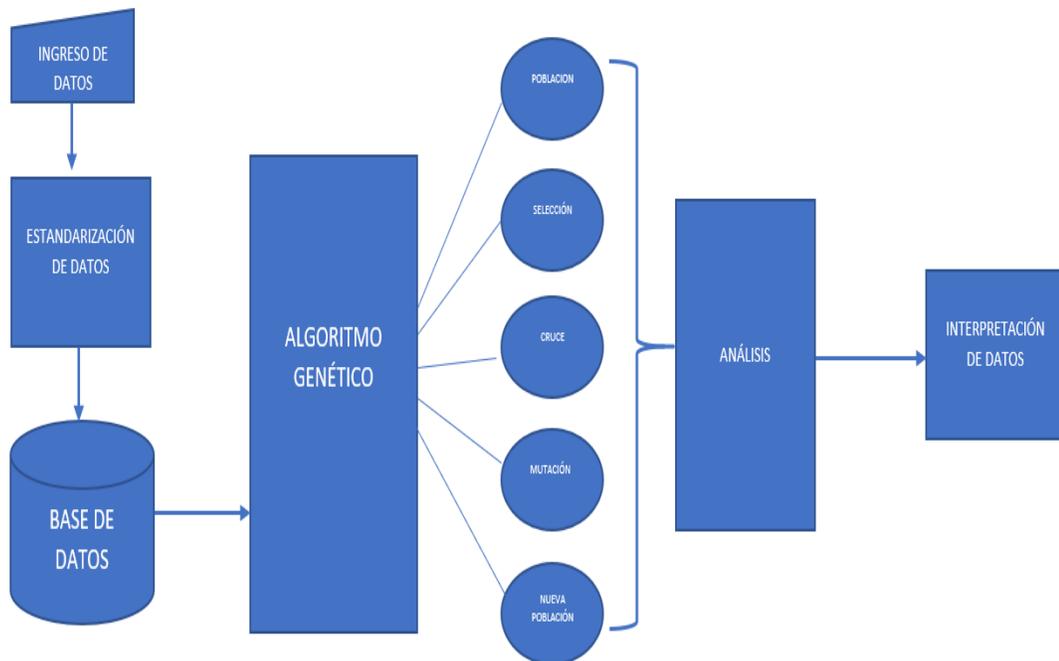


Figura 3.22 Esquema del funcionamiento del modelo de predicción

3.4.3. Implementación de Algoritmos

Para este acápite en primer lugar se entrenó con la mayoría de los algoritmos de selección en Weka, sin embargo, los que mejor rindieron para el objetivo del Modelo de predicción de Cáncer de cuello uterino fueron: Ridor, PART y NNge.

Código Principal, java

```
package formularios;

import java.util.Random;

import javax.swing.JOptionPane;

/**
 * @author Jaime Hidalgo
 *      *jaimehidalgo10@yahoo.com
 */

public class Main {

    public static void main(String[] args) {

        IniciarPoblacion(Poblacion);

        verPoblacion(Poblacion,false);

        double adaptados=0;

        while(adaptados<961){
```

```

        convertir_individuo(Poblacion);

        adaptados=Calidad_Individuo(Poblacion);

        adaptabilidad(Poblacion,sumaToria);

        verPoblacion(Poblacion,true);

        Seleccion_Parejas(Poblacion);

        Torneo(Poblacion);

        verGanadores(Ganadores);

        Copiarse(Poblacion,PoblacionTem);

        verPoblacion(PoblacionTem,true);

        Seleccion_Parejas(Poblacion);

        Combinacion_Mutacion(Poblacion,PoblacionTem);

    }

    adaptados=Calidad_Individuo(Poblacion);

    verPoblacion(Poblacion,false);

```

Código de mutación

```

public static void Combinacion_Mutacion(String [][] Poblacion,String [][] PoblacionTem ){

    System.out.println("*****Combiancion y Mutacion*****");

    System.out.println("*****");

    Random ri = new Random();//aleatorio para el punto de combinacion

    int puntocruce=0;

```

```

String [] IndividuoA;

String [] IndividuoB;

String ParejaA="";

//se hace sólo hasta la mitad porque cada uno tiene una pareja
for(int i=0;i<Parejas.length/2;i++){

    IndividuoA=Poblacion[i][1].split(",");

    ParejaA=Parejas[i]; //se obtiene la pareja del vector de parejas

    String cadAdn="";

    IndividuoB=Poblacion[Integer.parseInt(ParejaA)][1].split(",");

    puntocruce=ri.nextInt(4);//punto de cruce aleatorio

    System.out.println("punto cruce ["+puntocruce+"]["+Poblacion[i][0]+"]"

    +["+Poblacion[i][1]+"] [Cruzado con] ["+Poblacion[Integer.parseInt(ParejaA)][0]+"]"

    +["+Poblacion[Integer.parseInt(ParejaA)][1]+"]");

    //Genes del primer Individuo

    for(int t=0;t<puntocruce;t++){

        cadAdn+=IndividuoA[t]+",";

    }

    //Genes del segundo individuo

    for(int t=puntocruce;t<IndividuoA.length;t++){

        cadAdn+=IndividuoB[t]+",";

    }

    System.out.println("Nuevo Individuo ["+cadAdn+"]");

    PoblacionTem[i][0]=""+i;

    PoblacionTem[i][1]=cadAdn;

```

```

} //for de parejas

for(int i=0;i<Parejas.length;i++){

    Poblacion[i][0]=PoblacionTem[i][0];

    Poblacion[i][1]=PoblacionTem[i][1];

}

//se muta en gen despues de la combinacion

int mutado =(Parejas.length/2)+1;

IndividuoA=Poblacion[mutado][1].split(",");

System.out.println("*****Mutacion*****");

System.out.println("****Individuo*****Resultado*****");

//System.out.println(""+Poblacion[mutado][0]+" "+" "+Poblacion[mutado][1]+"");

int gen=ri.nextInt(4);//gen aleatorio

if(IndividuoA[gen].equals("0")){

    IndividuoA[gen]="1";

}else{

    IndividuoA[gen]="0";

}

//se arma la cadena de ADN para mutar el gen

String cadAdn="";

for(int t=0;t<IndividuoA.length;t++){

    cadAdn+=IndividuoA[t]+",";

}

```

3.4.4. Publicación, Despliegue

En la pantalla se puede visibilizar los siguientes resultados tomando como algoritmo genético.

```
*****Poblacion Actual*****
[0][1,0,1,1,1,],[23.0][529.0][0.018414731785428344] Pareja 5
[1][1,0,1,1,1,],[23.0][529.0][0.018414731785428344] Pareja 4
[2][1,1,1,1,1,],[31.0][961.0][0.024819855884707767] Pareja 3
[3][1,1,1,1,1,],[31.0][961.0][0.024819855884707767] Pareja 2
[4][1,0,1,1,1,],[23.0][529.0][0.018414731785428344] Pareja 1
[5][1,0,1,1,1,],[23.0][529.0][0.018414731785428344] Pareja 0

*****
*****Seleccion Parejas*****
*****Combiancion y Mutacion*****
*****
punto cruce [1][0][1,0,1,1,1,][Cruzado con] [5][1,0,1,1,1,]
Nuevo Individuo [1,0,1,1,1,]
punto cruce [2][1][1,0,1,1,1,][Cruzado con] [4][1,0,1,1,1,]
Nuevo Individuo [1,0,1,1,1,]
punto cruce [3][2][1,1,1,1,1,][Cruzado con] [3][1,1,1,1,1,]
Nuevo Individuo [1,1,1,1,1,]
*****Mutacion*****
***Individuo*****Resultado*****
[4][1,0,1,1,1,] Gen mutado[1] Resultado=> [4][1,1,1,1,1,]
*****Mejor Adaptado*****
```

*****961.0*****

*****Poblacion Actual*****

[0][1,0,1,1,1,],[23.0][529.0][0.018414731785428344]

[1][1,0,1,1,1,],[23.0][529.0][0.018414731785428344]

[2][1,1,1,1,1,],[23.0][529.0][0.018414731785428344]

[3][1,1,1,1,1,],[23.0][529.0][0.018414731785428344]

[4][1,1,1,1,1,],[31.0][961.0][0.024819855884707767]

[5][1,0,1,1,1,],[23.0][529.0][0.018414731785428344]

3.4.5. Resultados

Se puede observar que hay un crecimiento de cáncer de cuello uterino en el siguiente gráfico entregado por el modelo de predicción CACUGEN.

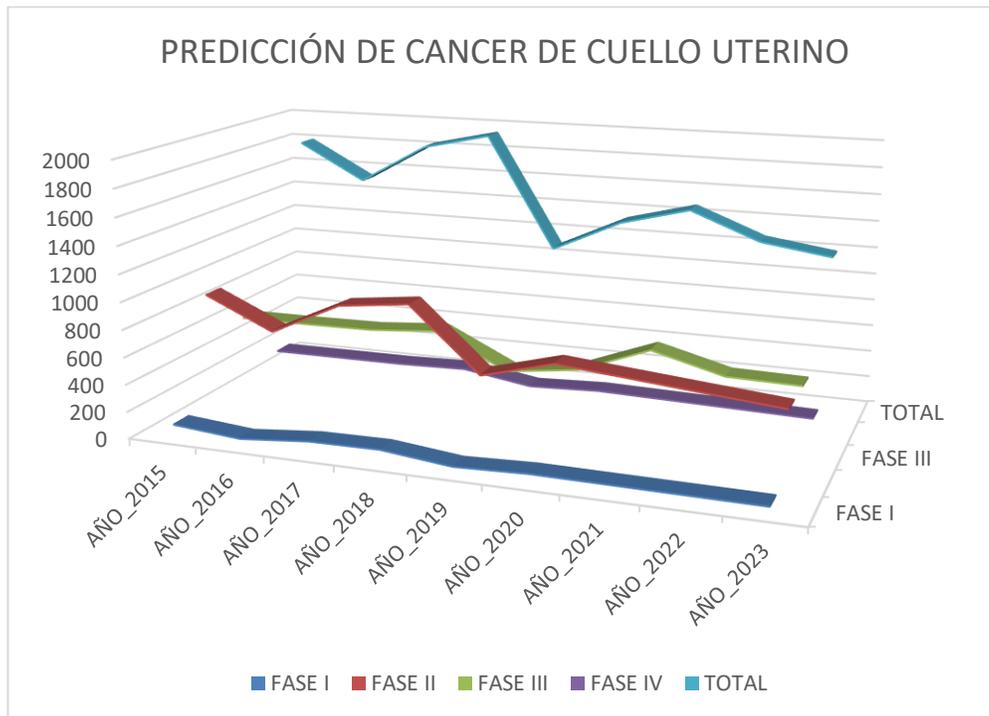


Figura 3.23 Resultado de la predicción

Fuente: Elaboración propia

3.5. Calidad Del Software (COCOMO II)

Para que el software sea un producto de calidad, se aplicó la métrica ISO/IEC 9126 de calidad de software, proporciona una clasificación sobre la calidad en un conjunto estructurado.

3.5.1. Funcionalidad

Las funcionalidades son aquellas que satisfacen las necesidades implícitas o explícitas. A continuación, se muestra la ponderación de las características funcionales.

Tabla 3.25 Ponderación de la funcionalidad

Característica	Ponderación
Adecuación	90%
Exactitud	95%
Conformidad	95%
Cumplimiento funcional	90%
Promedio	92.5%

Fuente: Elaboración propia

Por tanto, se deduce que el prototipo tiene una funcionalidad del 92.5%.

3.5.2. Confiabilidad

Los Atributos relacionados con la capacidad del software dado para mantener su usabilidad bajo condiciones establecidas en un tiempo establecido.

$$\text{Confiabilidad} = 1 - (5/1200) \times 100$$

$$\text{Confiabilidad} = 99 \%$$

El sistema tiene una confiabilidad del 99%.

3.5.3. Usabilidad

Los atributos y su conjunto están relacionados con el esfuerzo necesario para su uso, y en la valoración individual de tal uso.

Tabla 3.26 Ponderación de métricas internas usabilidad

Característica	Métrica interna	Puntaje
Interfaz de usuario amigable	I1: Interfaz de datos amigable	90
	I2: Interfaz de gráficos amigable	90
Comprensión	C1: Comprensión de datos	85
	C2: Comprensión de gráficos	90
Operatividad	O1: Correcta operacionalidad de la interfaz	95
	O2: Correcta operacionalidad de visualización de datos	95
	O3: Correcta operacionalidad de los gráficos	95
Atractividad	A1: Atractividad de la interfaz	90
	A2: Atractividad de los gráficos	90
	A3: Atractividad de la visualización de los datos	85

Fuente: Elaboración propia

Tabla 3.27 Totales de métricas internas usabilidad

Métrica	Puntaje promedio
Interfaz de usuario amigable (I)	95
Comprensión (C)	90
Operatividad (O)	95
Atractividad (A)	89

Fuente: Elaboración propia

Con los datos obtenidos en la tabla 3.28 se aplica en la fórmula:

$$\text{Usabilidad} = \sum(xi/n)$$

$$\text{Usabilidad} = 369/4$$

$$\text{Usabilidad} = 92.25\%$$

3.5.4. Eficiencia

Para poder obtener el cálculo de la eficiencia del sistema se consideró ponderar las características esenciales que el sistema desempeña.

Tabla 3.28 Evaluación de desempeño

Característica de desempeño	Ponderación
Rapidez de inicio	5
Rapidez de proceso	4
Proceso rápido de búsqueda	5
Fluidez	5
Disponibilidad	4
Total	23

Fuente: Elaboración propia

En base a los datos de la anterior tabla se podría llegar a tener una idea de la eficiencia, para ello se utilizó la siguiente formula:

$$\text{Eficiencia} = \Sigma x_i / n * 100/n$$

$$\text{Eficiencia} = 23/5 * 100/5$$

$$\text{Eficiencia} = 92\%$$

3.5.5. Mantenibilidad

Son el conjunto de atributos los cuales nos da la opción de poder corregir, aumentar o modificar los errores del software, dicho resultado se obtiene mediante la siguiente formula:

$$\text{Mantenibilidad} = \frac{(Mt - (Fc + Fa + Fd))}{Mt}$$

Donde:

Mt = número de módulos en la versión actual.

Fc = número de módulos en la versión actual que han cambiado.

Fa = número de módulos en la versión actual añadido.

Fd = número de módulos en la versión anterior que se ha borrado.

Entonces:

$$Mt = 1; Fc = 1; Fa = 0; Fd = 0$$

$$\text{Mantenibilidad} = \frac{(3 - (0 + 0 + 0))}{3}$$

$$\text{Mantenibilidad} = 1$$

$$\text{Mantenibilidad} = 100 \%$$

3.5.6. Portabilidad

Es la capacidad que tiene el software para ser trasladado de un entorno a otro.
Se lo calcula mediante la fórmula:

$$Portabilidad = 1 - \frac{ndpm}{ndim}$$

Donde:

ndpm = número de días para portar el modelo.

días. ndim = número de días para implementar el modelo

$$Portabilidad = 1 - \frac{1}{5}$$

$$Portabilidad = 0.8 * 100$$

$$Portabilidad = 80 \%$$

3.5.7. Resultados

Calculando de manera independiente cada uno de los factores en cuanto a las características de la norma ISO 9126, estos resultados nos sirven para poder realizar los cálculos.

Tabla 3.29 Análisis global de calidad

N°	Característica	Resultado
1	Funcionabilidad	92.5%
2	Confiabilidad	99%
3	Usabilidad	92.25%
4	Eficiencia	92%
5	Mantenibilidad	100%
6	Portabilidad	80%
Evaluación de la Calidad Global		92.625%

Fuente: Elaboración propia

Según Pressman dice que el resultado de la evaluación de una métrica o modelo si supera el 65% es aceptado. Por lo que el 92.625 % encontrado en la medición es aceptable para el modelo.

3.6. Costos (COCOMO II)

Con ayuda de COCOMO II, en este punto se estima el costo de producción del software desarrollado.

3.6.1. Puntos de función

La estimación por puntos de función está en la medida de la funcionalidad del sistema de información y un conjunto de factores individuales del sistema. Los puntos de función son estimadores que puede ser de utilidad en las etapas iniciales del modelo. La medida de puntos de función está cuantificada en base a diferentes funcionalidades. Entrada, salida, consulta y archivos.

Se estima la construcción de interfases. La tabla siguiente describe los componentes relacionados con su complejidad asignada a cada uno de los factores que se deben considerar para la estimación del modelo.

Tabla 3.30 Estimación de interfaces a ser desarrolladas

Tipo de Parámetros	Cantidad
Entrada	3
Salida	2
Archivos	2
Consultas	5
Interfases	3

Fuente: Elaboración propia

Según el estimado de interfaces de la tabla anterior, se procede a clasificarlos según su complejidad y luego multiplicar por los pesos establecidos de acuerdo a COCOMO II, para estimar los puntos función del sistema

Tabla 3.31 Puntos de función del sistema

Tipo de Cuenta parámetro	Cuenta	Factor ponderación	de Total
Entrada	3	5	15
Salida	2	6	12
Archivos	2	5	10
Consultas	5	6	30
Interfaces	3	5	15
Cuenta total			82

Fuente: Elaboración propia

En la siguiente tabla se muestra los factores de ajuste donde se pondera un puntaje que se encuentre entre 0 y 5

Tabla 3.32 Cálculo de ajuste de complejidad

Fi	Factor	Valor
1	Mecanismo de recuperación	2
2	Comunicación de datos	5
3	Rendimiento	5
4	Configuración usada rigurosamente	2
5	Entrada de datos en línea	2
6	Factibilidad operativa	4
7	Actualización en línea	1
8	Interfaces complejas	3
9	Proceso interno complejo	4
10	Reusabilidad de código	4
11	Fácil instalación	5
12	Instalaciones múltiples	2
13	Facilidad de cambios	3
14	Funciones de proceso distribuido	2
$\sum F_i$		44

Fuente: Elaboración propia

Con el promedio encontrado, se reemplaza los datos e la fórmula de punto de función ajustado.

$$PFA = \text{cuenta total} * (0.65 + 0.01 * \sum F_i)$$

$$PFA = 82 * (0.65 + 0.01 * 44)$$

$$PFA = 89.38$$

3.6.2. Aplicación de COCOMO II

Para poder calcular las líneas de código, utilizamos el valor del punto de función ajustado, de igual forma utilizaremos el valor de Factor de línea de código del lenguaje de programación utilizada para el desarrollo.

Tabla 3.33 Factor LCD/PF de lenguaje de programación

Lenguaje	Nivel	Factor LCD/PF
C	2.5	128
ANSI/basic	5	64
Java	6	53
PL/I	4	80
Visual Basic	7	46
ASP	9	36
PHP	11	29
Visual C++	9.5	34

Fuente: Pessman,2002

Reemplazamos los datos en la fórmula para calcular las líneas de código:

$$LDC = PFA * \text{Factor LDC/PF}$$

$$LDC = 89.38 * 53$$

$$LDC = 4738.14$$

$$KLDC = \frac{4738.14}{1000} = 4.73714$$

Tabla 3.34 Tipo de proyecto de software

MODO	a	b	c	d
Orgánico	2.4	1.05	2.5	0.38
Semiacoplado	3.0	1.12	2.5	0.35
Empotrado	3.6	1.2	2.5	0.32

Fuente: Elaboración propia en base a COCOMO

En base a la tabla 3.31 y el KLDC, se aplica a la fórmula para hallar el esfuerzo que se planteó en el capítulo anterior.

$$E = a * (KLDC)^b$$

$$E = 2.4 * (4.73714)^{1.05}$$

$$E = 10.24 = 11 \text{ personas /mes}$$

Ahora se calcula el tiempo de desarrollo

$$T = c * (E)^d$$

$$T = 2.5 * (10.24)^{0.38}$$

$$T = 6.05 = 6 \text{ meses}$$

Número de personas para el desarrollo

$$P = \frac{E}{T}$$

$$P = \frac{10.24}{6.05T}$$

$$P = 1.69 = 2 \text{ personas}$$

Calculando el costo del software se considera el sueldo aproximado de un Ingeniero de Sistemas Junior de 2.500 Bs. Mensuales.

Reemplazando en la fórmula tendremos:

$$CT = 2500 * (P * T)$$

$$CT = 2500 * (2 * 6)$$

$$CT = 30.000 \text{ Bs.}$$

3.6.3. Costo de elaboración del Software

En este punto se toman se consideran los otros costos que se muestran a continuación:

Tabla 3.35 Costo de elaboración de Prototipo

Detalle	Importe
Análisis y diseño del prototipo	700 Bs.
Material de Escritorio	120 Bs..
Conexión a internet	298 Bs.
Otros	50 Bs.
Total	1.168 Bs

Fuente: Elaboración propia

3.6.4. Costo total

Para el cálculo del costo total se tomó en cuenta el costo del software calculado anteriormente y el costo de elaboración.

Tabla 3.36 Costo total del prototipo.

Detalle	Importe
Costo del software	30.000 Bs.
Costo de elaboración	1.168 Bs.
Total	31.168 Bs.

Fuente: Elaboración propia.

Considerando la tabla anterior se concluye que el costo total del software es de 31.168 Bs.

CAPITULO IV

PRUEBAS Y

RESULTADOS

PRUEBAS Y RESULTADOS

El este capítulo se describen las pruebas realizadas al prototipo del modelo de predicción en base a algoritmos genéticos, utilizando los resultados obtenidos con los diferentes ensayos. Las pruebas de la hipótesis y del sistema.

4.1. Prueba al Modelo de Predicción

A continuación, se muestran los resultados al proceso de modelado de predicción en base a algoritmos genéticos, de acuerdo a los datos estandarizados con que se cuentan. Donde se visibilizan tanto en porcentajes como gráficamente el funcionamiento del modelo de predicción de cáncer de cuello uterino en la Ciudad de El Alto para las gestiones 2020 hasta la gestión 2025.

4.2. Predicción Según el Modelo Implementado

El modelo de predicción está en pleno funcionamiento, en la que se puede visibilizar las ventanas de entrada, de selección del tipo de algoritmo, el número de iteraciones que pueden ser predefinidas por el sistema o de manera manual para posteriormente, cargar los datos y mostrar los resultados de manera gráfica, como se ve en los gráficos.

Cabe puntualizar, que los datos fueron en primera instancia estandarizados y entrenados para que posteriormente fueran introducidos en em modelo de predicción y este puede realizar los procesos internos.

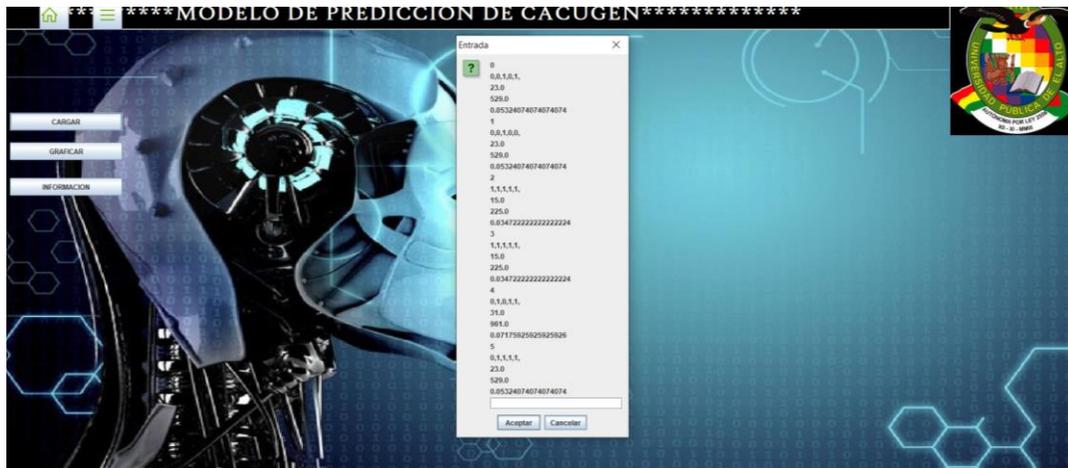


Figura 4.1 Modelo de predicción de Cáncer de Cuello Uterino

Fuente: Elaboración propia

En la figura 4.1 se puede visibilizar como se está corriendo algoritmo genético, para la predicción del índice de crecimiento de Cáncer de Cuello Uterino.

Así mismo se cargaron los datos seleccionados, que son 8141 personas que fueron registradas por el SEDES El Alto. De los cuales se tomaron algunos parámetros para predecir los índices de crecimiento de Cáncer de Cuello uterino.

Con los datos ya estandarizados de los índices de crecimiento histórico por gestiones y grupo etarios se puede observar los pronosticados para las siguientes gestiones.

Según los siguientes datos:

[0][1,1,0,0,1,][[27.0][729.0][0.005352031438935914][null]

[1][1,1,1,0,1,][[31.0][961.0][0.009371221281741234][null]

[2][1,1,1,1,1,][[29.0][841.0][0.007766626360338573][null]

[3][1,1,1,1,1,][[29.0][841.0][0.008766626360338573][null]

[4][1,1,0,0,1,][[19.0][361.0][0.005743651753325272][null]

[5][1,1,1,0,1,][[27.0][729.0][0.008162031438935914][null]

Observado los índices de crecimiento históricos que se muestran en la tabla 4.1 se puede observar los índices obtenidos manualmente. Con estos datos podemos también aseverar, que el modelo de predicción del índice de cáncer de cuello uterino en base a algoritmos genéticos, está funcionando de manera efectiva.

Tabla 4.1 Índice de crecimiento histórico

	2015	2016	2017	2018	2019	2020
GRUPO 1	85	41	86	91	35	
GRUPO 2	912	682	924	982	520	
GRUPO 3	602	603	609	648	399	
GRUPO 4	194	195	196	209	128	
GRUPO 5	1793	1521	1815	1930	3700	
INDICE		0,003	0,0084	0,0054	0,0037	0,0051

Fuente: Elaboración propia

4.3. Pruebas

4.3.1. Prueba de la Hipótesis

Para la demostración de la hipótesis se empleó el método de T-Student de una cola independiente, a partir de los siguientes datos:

- ✚ Tamaño de la muestra
- ✚ Desviación estándar
- ✚ Media aritmética
- ✚ Hipótesis nula
- ✚ Valor crítico
- ✚ Valor probable

Tamaño de la muestra

En este caso se determinará el tamaño de la muestra a partir de la siguiente fórmula:

$$n = \frac{Z^2 * p * q * N}{e^2(N - 1) + Z^2 * p * q}$$

Donde:

Entonces tenemos:

n=muestra	n=?
Z=nivel de confianza	Z=95% = 1.96
p=probabilidad a favor	p=65% = 0.65
q=probabilidad en contra	q=35% =0.35
e=error de muestra	e=5% = 0.05
N= población	N= 8141

Reemplazando en la fórmula tenemos:

$$n = \frac{1.96^2 * 0.65 * 0.35 * 8141}{0.05^2(8141 - 1) + 1.96^2 * 0.65 * 0.35}$$

$$n = 335.$$

Plantear la hipótesis nula (H0) y la hipótesis alternativa (H1)

Ho: La construcción del modelo de predicción del índice de crecimiento del cáncer uterino apoyado en técnicas de desarrollo y algoritmos genéticos tendrá una eficacia del 95%.

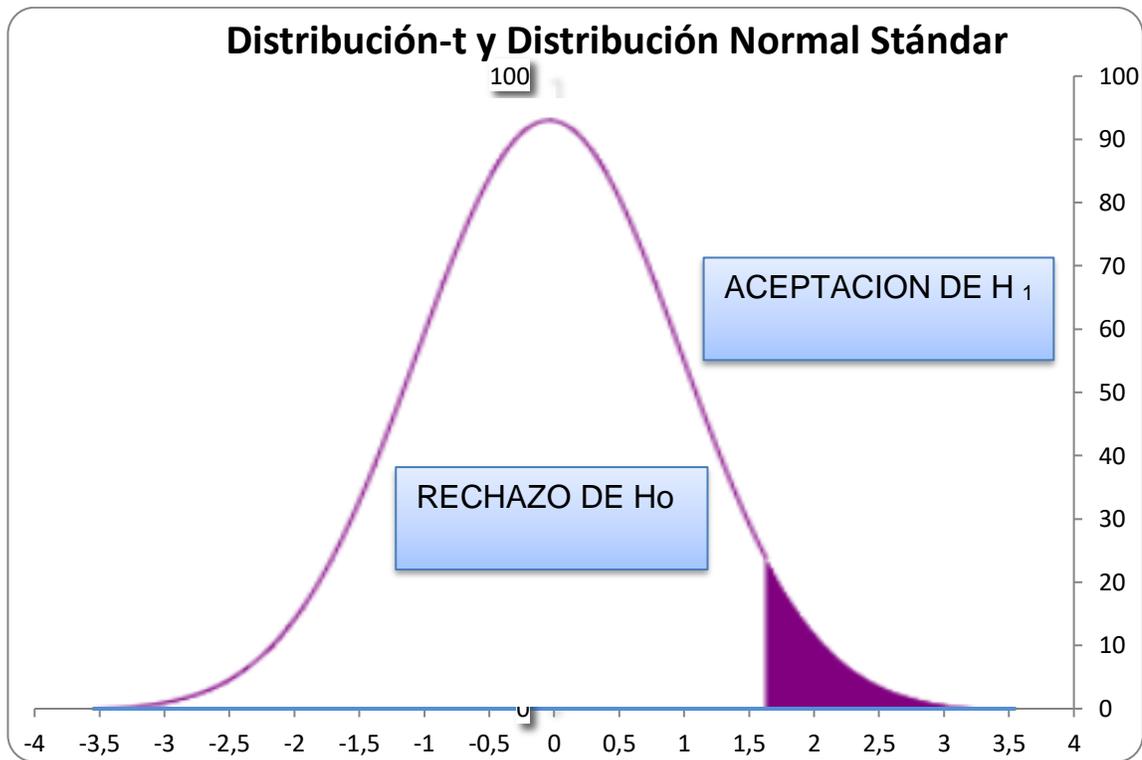


Figura 4.2 Campana de Gauss representando T Student

Fuente: Elaboración propia

Por lo que tiene el punto crítico de:

$$t_{\text{crítico}} = 1.64$$

Determinamos el punto de prueba con la siguiente fórmula de:

$$t_{\text{prueba}} = \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{n}}}$$

Donde:

$$\begin{aligned}t &=? \\ X &=95.5 \\ \mu &=95 \\ \sigma &=6.2 \\ n &=335\end{aligned}$$

Reemplazando los valores tenemos

$$t = \frac{95.32 - 95}{\frac{6.2}{\sqrt{335}}}$$

$$t = 0.94$$

Según la tabla anterior el valor crítico es: $t_{\text{critico}} = 1.64$

Para comparar con el valor t obtenido anteriormente: $t_{\text{prueba}} = 0.94$

Si $|t_{\text{critico}}| > |t_{\text{prueba}}|$ por lo que se rechaza H_0 que es la prueba nula.

Por tanto, la probabilidad de obtener los datos si H_1 es cierta según α es de 95%, es aceptada.

Mediante las fórmulas de T - Student de una cola, demostramos que la hipótesis se cumple en el modelo de predicción del índice de crecimiento del cáncer uterino apoyado en técnicas de desarrollo y algoritmos genéticos tendrá una eficacia del 95%

CAPITULO V

CONCLUSIONES Y

RECOMENDACIONES

CONCLUSIONES Y RECOMENDACIONES

El presente capítulo describe las conclusiones alcanzadas en el desarrollo de la propuesta de investigación, el estado de la hipótesis y las recomendaciones que se da al lector, sobre futuros temas, como una continuación al presente trabajo de investigación.

5.1. Introducción

El modelo de predicción de Cáncer de Cuello Uterino, desarrollado en este tema de investigación, está implementado en un entorno local, esto para que los especialistas, así como las personas de salud puedan utilizar como un medio de concientización a la población femenina en realizar los chequeos correspondientes especialmente a partir de los 35 años de edad.

5.2. Estado de los Objetivos

El objetivo general descrito en el capítulo 1 menciona: “Desarrollar el Modelo de predicción aplicado al Índice de crecimiento del Cáncer de Cuello Uterino en la Ciudad de El Alto, en Base a Algoritmos Genéticos”.

En el Capítulo 3, se presenta el proceso de desarrollo del modelo de predicción. Este proceso es realizado utilizando algoritmos genéticos y redes neuronales, mostrando la predicción del índice de crecimiento del cáncer de cuello uterino.

Por lo anterior descrito se logró alcanzar en su totalidad, ya que se construyó el modelo de predicción en base a algoritmos genéticos.

En cuanto a los objetivos específicos se justifica cada uno de ellos en los siguientes incisos:

- a) “Realizar en análisis y estudio de la situación actual”. Al obtener los datos del SEDES para la Ciudad de El Alto se pudo conocer la situación actual de los casos positivos de Cáncer de Cuello Uterino.
- b) “Recopilar información relevante con respecto al cáncer de cuello uterino”. Se pudo discriminar de todos los servicios que presta los centros de salud y hospitales a la Ciudad, los que son preponderantes para el análisis de los casos con cáncer de cuello uterino.
- c) “Obtener las estadísticas de pacientes atendidos con cáncer de cuello uterino en la Ciudad de El Alto”. Gracias a la colaboración de SEDES El Alto se pudo acceder a la página, donde se almacenan todos los servicios que prestan los centros de salud y hospitales tanto públicos como privados en la Ciudad de El Alto.
- d) “Sistematizar una base de datos estandarizar de la enfermedad del cáncer de cuello uterino”. Se logró estandarizar los datos que son relevantes para el estudio del cáncer de cuello uterino, así como la discriminación por grupos étnicos en el capítulo 3.
- e) “Obtener el conocimiento del experto humano mediante entrevistas con el profesional del área”. En las entrevistas con los médicos encargados de SEDES El Alto se logró tener una ficha de referencia que se presenta en el capítulo 3.
- f) “Desarrollar una aplicación de agentes inteligentes en el área de predicción”. Se desarrolló el modelo de predicción en todo el capítulo 3.

- g) “Buscar y aplicar algoritmos genéticos para la predicción”. Con la ayuda de redes neuronales, así como el WEKA se pudo entrenar y buscar la población mejor estandarizada, para luego utilizar algoritmos genéticos en el trabajo de investigación.
- h) “Sistematizar el cuadro de índice de crecimiento del cáncer de cuello uterino de la Ciudad de El Alto”. Se logró sistematizar en el presente trabajo de investigación gracias al modelo los índices de crecimiento de cáncer de cuello uterino.
- i) “Desarrollar un prototipo del modelo de predicción del índice de crecimiento del cáncer de cuello uterino”. En el capítulo 3 y de acuerdo a modelo XP, se pudo modelar el prototipo de predicción con los datos estandarizados.
- j) “Describir e interpretar los datos obtenidos”. Mediante el modelo de predicción se pudo visibilizar los índices de crecimiento pronosticados para las siguientes gestiones.

5.3. Estado de la Hipótesis

La hipótesis establecida en capítulo 1 es la siguiente: “La construcción del modelo de predicción del índice de crecimiento del cáncer uterino apoyado en técnicas de desarrollo y algoritmos genéticos tendrá una eficacia del 95%”. La planificación y estructura de la secuencia de los pasos a seguir a la hora de aplicar el modelo de predicción en base a algoritmos genéticos, y según la metodología ISOS/IEC se logró el 92.625% de eficiencia al momento de aplicar con los usuarios.

Por tanto, el modelo de predicción en base a logaritmos genéticos, asegura la calidad de los resultados del mismo, debido a que no sólo se documentó de manera ordenada y precisa, sino que también es útil para poder prevenir a las mujeres del crecimiento en el índice de casos de cáncer de cuello uterino en la Ciudad de El Alto.

5.4. Conclusiones

Una vez realizado el “Modelo de predicción aplicado al Índice de crecimiento del Cáncer de Cuello Uterino en la Ciudad de El Alto, en Base a Algoritmos Genéticos”, se puede llegar a las siguientes conclusiones.

- ✓ Con el modelo de predicción en base a algoritmos genéticos, se puede predecir el índice de crecimiento de cáncer de cuello uterino en la Ciudad de El Alto.
- ✓ Teniendo los datos estandarizados de mujeres con Cáncer de Cuello uterino, se pueden realizar predicciones futuras y alertar a las mujeres sobre los riesgos.
- ✓ Conociendo el índice de crecimiento de cáncer de cuello uterino de la Ciudad de El Alto, los especialistas pueden realizar políticas de prevención.
- ✓ En modelo XP, es muy dinámica y fácil de implementar para el desarrollo de software a medida de los usuarios y el programador.

5.5. Recomendaciones

Considerando que la información que se pudo recabar durante la investigación, se detalla las siguientes recomendaciones:

- ✓ Se recomienda realizar las actualizaciones de los datos periódicamente para que el modelo de predicción tenga utilidad.

- ✓ Se recomienda continuar la investigación georreferenciando los casos de cáncer de cuello uterino, ya que se cuenta con los datos de las redes de salud en la Ciudad de El Alto.

- ✓ Se recomienda a los especialistas seguir con la investigación, tomando en cuenta los datos con los que facilita SNIS - VE.

BIBLIOGRAFÍA

BIBLIOGRAFIA

- Banda Gamboa, H. (2014). *Inteligencia Artificial Principios y Aplicaciones*. Ecuador;Quito.
- Beáti, H. (2012). *El gran libro PHP*. Argentina: Marcombo.
- Boehm, B. W. (1981). *Software Engineering Economics*. Prentice-Hall.
- Bunge, M. (1960). *La ciencia, su método y su filosofía*. Mexico: 2000.
- Concha, M. (2007). *Diagnóstico y Terapia del Virus Papiploma Humana*. Santiago , Chile.
- Dermatología, S. A. (2004). *Consenso de Papiloma Virus Humano (HPV) y Herpes Simplex Virus (HSV). Genital. .* Argentina.
- Fowler, M. (2005). *The new methodoly*. New Methodology.
- Garcia Serrano, A. (2012). *Inteligencia Artificial, fundamentos, práctica y aplicaiones*. España: CR Libros.
- Gestal, P. M. (2000). *Introducción a los Algoritmos Genéticos*. Mexico: Siglo XXI Editores.
- Gomez, A., C.López, M., Migani, S., & Otazú, A. (2010). *Un modelo de estimación de Software*.
- Guerra, T., & Gonzales, G. (2000). *El virus del Papiloma humano*. Guerra T A, Gonzales G E. El virus del Papiloma humano. Repercusión en la patología dermatológica de las nuevas vacunas. Más Dermatol. 200.
- Hilera, Hilera , G., & Martinez, V. (1995). *Redes Neuronales, Fundamentos Modelos y aplicaciones*. Mexico: Addison-WesleyIberoamerica.
- JoKowicz. (2008). *Reglas y Prácticas en eXtreme Programming*. España.

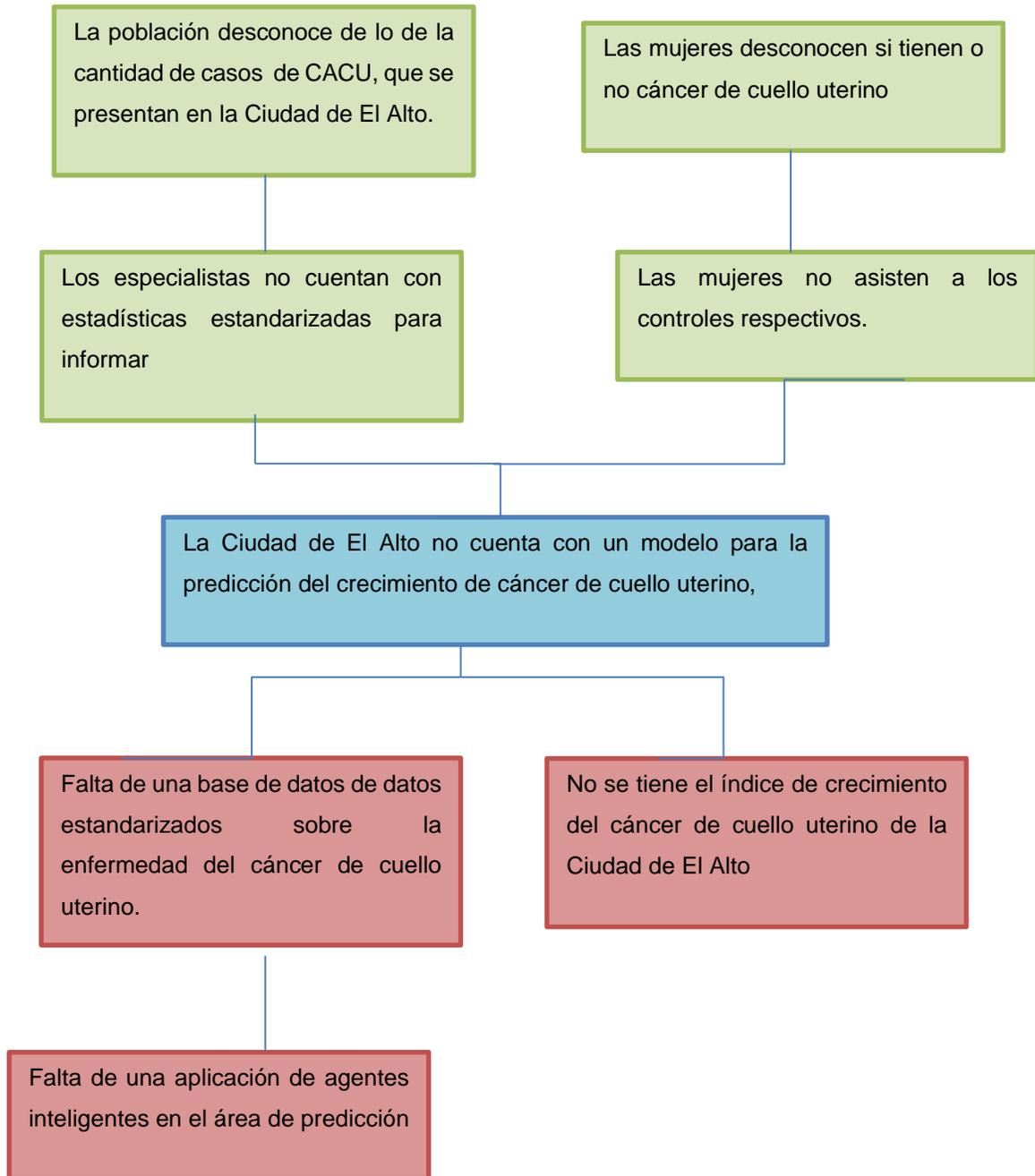
- Kent Beck, D. C. (1999). *Optional Scope Contracts* .
- Koch, N., & Kraus, A. (2002). *The Expressive Power of UML- base Web Engineering*. Alemania: Universidad de Munich.
- Leitzmann, M., Platz, E., Stampfer, M., Willett, W., & Giovannucci, E. (2004). *Ejaculation frequency and subsequent risk of prostate cancer*. Journal JAMA.
- Martin, J., & Lopez, P. (2012). *La evolución sin recortes*. Madrid, España: Editores de evolución.
- Mercado, S. (1998). *Como hacer una tesis*. Mexico: McGraw Hill.
- Miller, D., Hafez, K., & Stewart , A. (2003). *Prostate carcinoma presentation, diagnosis and staging*. National Cancer Data Base.
- Moore, K., & Dalley, A. (1999). *Clinically Oriented Anatomy*. Lippincott Williams & Wilkins.
- OMS. (2011). *Programa Nacional del Cancer*. Washington, D.C.:
- Pressman, R. (2010). *Ingeniería del Software un Enfoque Práctico*. Mexico: McGraw-Hill Companies.
- Quisberth Ibañez, G. W. (2012). *Modelo de Algoritmos Genéticos para Maximizar Nutrientes y Minimizar Costos*. La Paz.
- Ramos, G., & Páez, J. (2011). *ANALISIS DEL METODO PARA CALIFICACION DE SOFT-WARE QSOS* . Quito, Ecuador.
- Valdés Souto, F. (2013). *Midiendo la Calidad de Software*. Mexico.
- Vilches Aguirre, Y. (2017). *Desarrollo de un modelo predictivo de supervivencia en pacientes con cancer avanzado*. España: Universidad Autonoma de Madrid.

Viñas Aparicio, O. (2012). *Red Nueronal Artificial*. España: Universidad de Balladolid.

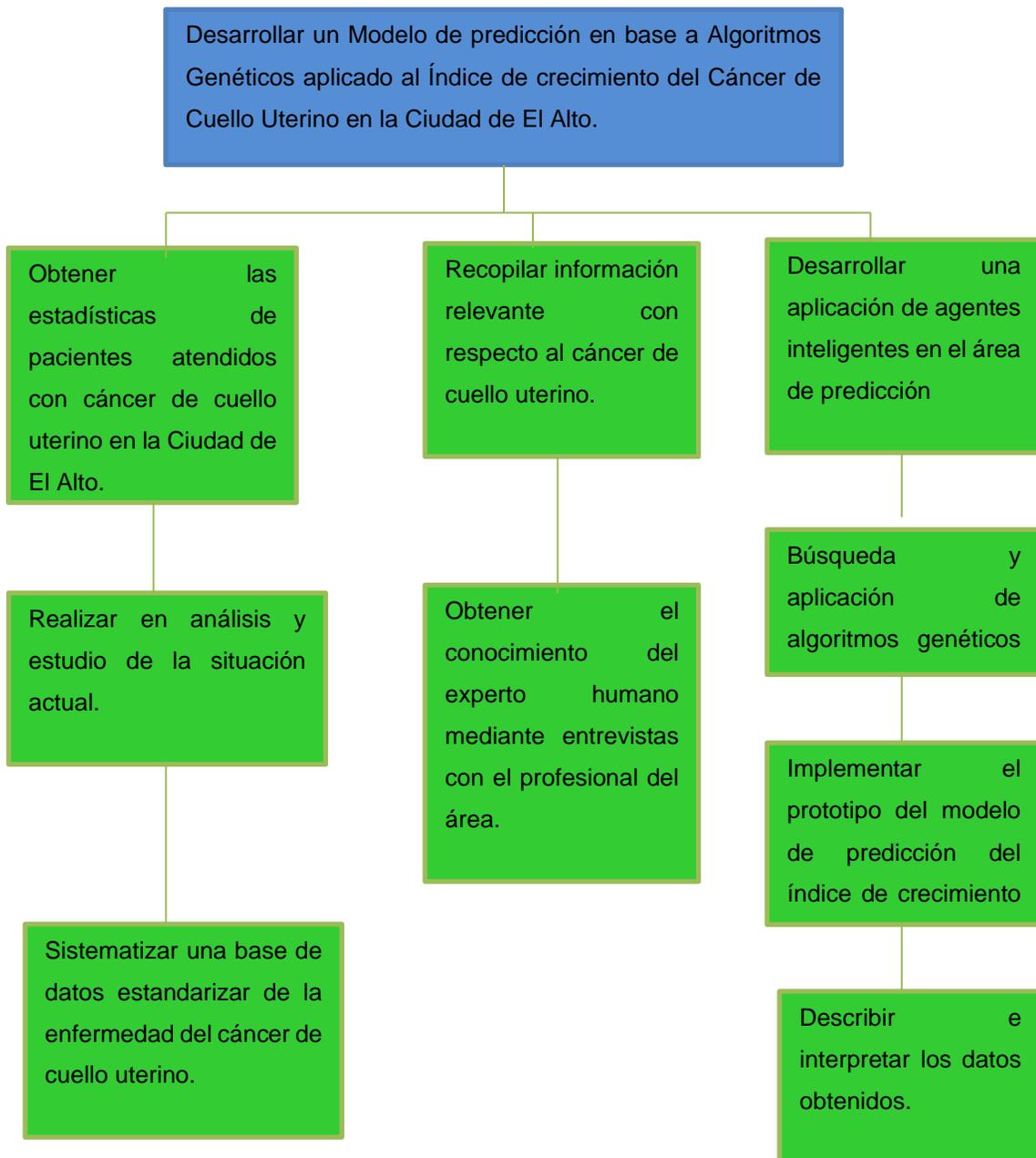
Yujra Chalco, M. A. (2017). *Sistema Experto para el diagnóstico de cáncer de próstata con lógica difusa*. Bolivia: UMSA.

ANEXOS

ANEXO A. ARBOL DE PROBLEMAS



ANEXO B. ARBOL DE OBJETIVOS



ANEXO C.

MANUAL DE USUARIO

MODELO DE PREDICCIÓN CACGUGEN

MANUAL DE USUARIO

El Modelo de Predicción CACUGEN, es una herramienta destinada a realizar la predicción del índice de crecimiento del Cáncer de Cuello Uterino en la Ciudad de El Alto, ya que esta tiene un crecimiento considerable.

1. **Objetivo**

Desarrollar las competencias requeridas para el uso adecuado y facilitar al usuario la manipulación del Modelo de Predicción de manera amigable. Ya que es un modelo abierto a los usuarios sin ninguna restricción.

2. **Requerimiento de Hardware**

- ✚ Pc con procesador equivalente a Intel Core i3 o superior
- ✚ Memoria Ram de 4 GB o superior
- ✚ Disco duro de memoria de 500 Megabyts o superior
- ✚ Dispositivos de entrada y salida

3. **Requerimiento de Software**

- ✚ Sistema operativo Windows
- ✚ Base de datos MySQL
- ✚ Servidor Apache
- ✚ Java Netbeans

4. **Desarrollo del manual de Usuario**

El usuario tiene la posibilidad de:

- ✚ Cargar la población desde la base de datos
- ✚ Determinar el número de generaciones
- ✚ Determinar el porcentaje de mutación
- ✚ Cargar el gráfico de la nueva población
- ✚ Iniciar el Modelo de Predicción
- ✚ Desplazar la ventana de los requisitos
- ✚ Finalizar el Modelo de Predicción

5. Ingreso al Modelo de Predicción

Para acceder al Modelo de Predicción, el usuario debe hacer correr el sistema.



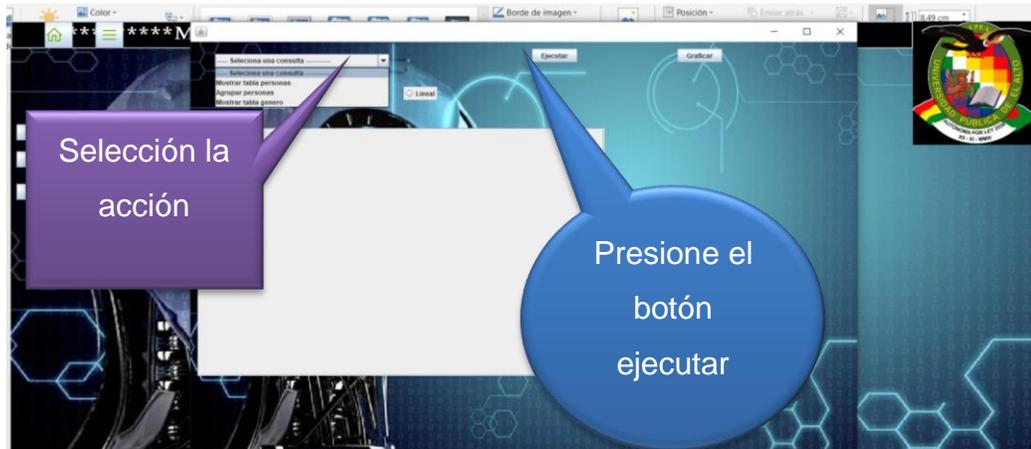
6. Desplazar a la ventana de las acciones

Una vez que el usuario inicie la sesión, el modelo se redireccionará al panel principal, donde tendrá la posibilidad de visibilizar el panel de graficas.



7. Cargar la población

Una vez redireccionado al panel de gráficas, el usuario tendrá la posibilidad de cargar la población inicial desde la base de datos, que se visibilizará en un cuadro de textos.



8. Selección

Luego de cargado los datos de la población inicial, el usuario puede elegir entre mostrar el comportamiento de la población inicial o generar la nueva población eligiendo el número de generaciones y el porcentaje de mutaciones.

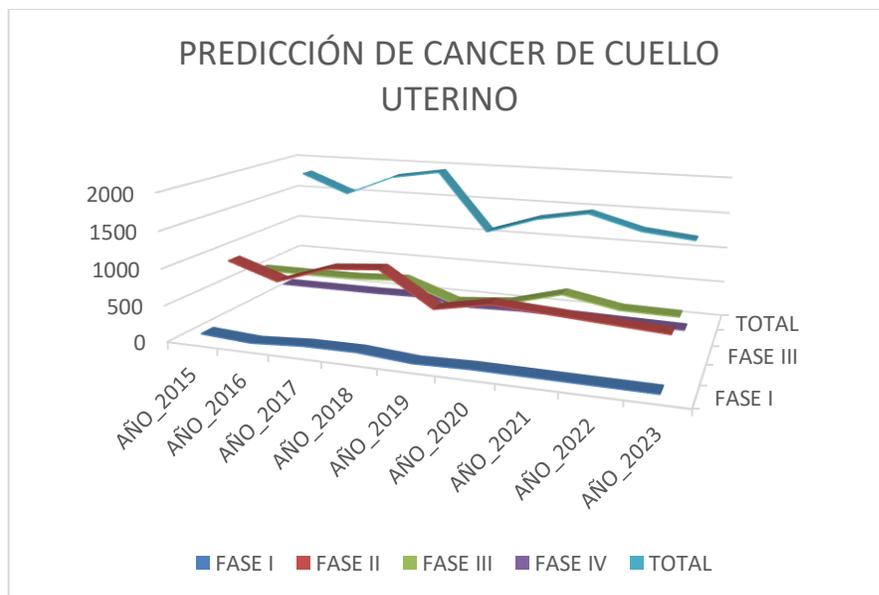
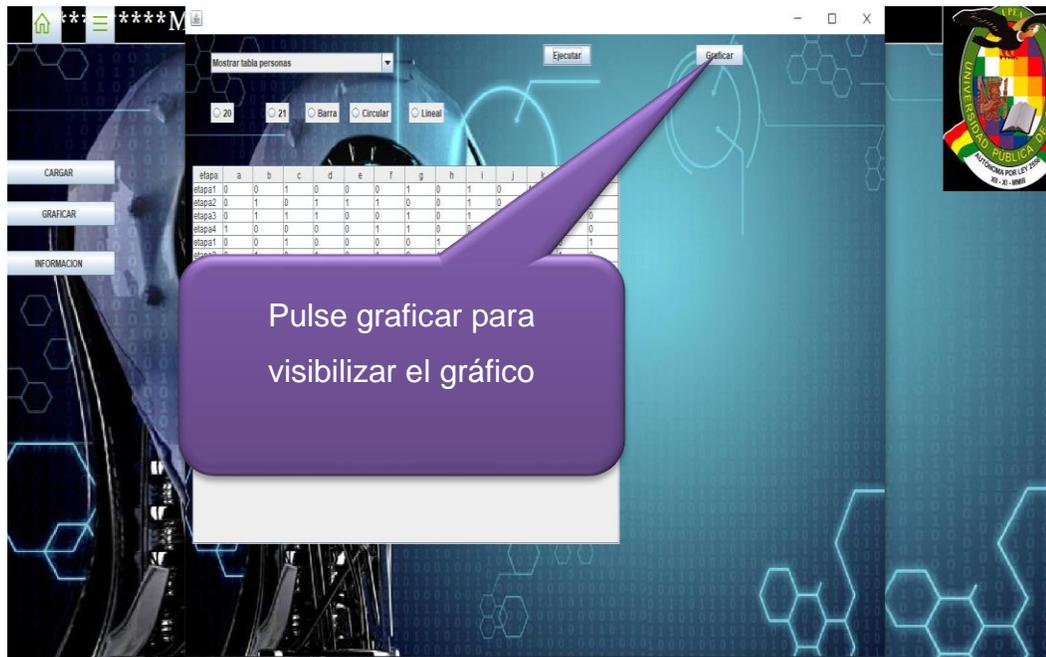
Seleccione la gestión

Seleccione generación y porcentaje de mutación

etapa	a	b	c	d	e	f	g	h	i	j	k	l	m
etapa1	0	0	1	0	0	0	1	0	0	1	0	0	0
etapa2	1	0	1	1	1	0	0	1	1	1	0	0	0
etapa3	0	1	1	0	0	0	1	0	1	1	0	0	0
etapa4	1	0	0	0	1	1	0	0	0	0	0	0	0
etapa1	0	1	0	0	0	0	0	1	0	1	0	0	0
etapa2	0	1	0	1	0	1	0	1	0	0	0	1	1
etapa3	0	1	1	1	0	0	1	0	1	1	0	1	1
etapa4	1	0	0	0	0	0	1	1	0	0	0	0	1
etapa1	0	1	0	0	0	0	0	1	0	0	0	1	0
etapa2	0	1	0	1	0	1	1	0	0	1	1	1	0
etapa3	0	1	1	1	0	1	0	1	1	0	1	1	1
etapa4	1	0	0	0	1	0	0	0	1	1	0	0	0
etapa1	0	1	0	0	0	0	0	1	0	0	0	1	1
etapa2	0	1	0	1	0	0	0	0	0	1	0	0	0
etapa3	0	1	1	0	1	1	0	0	0	1	1	1	1
etapa4	1	0	0	0	0	1	0	0	0	1	1	0	0

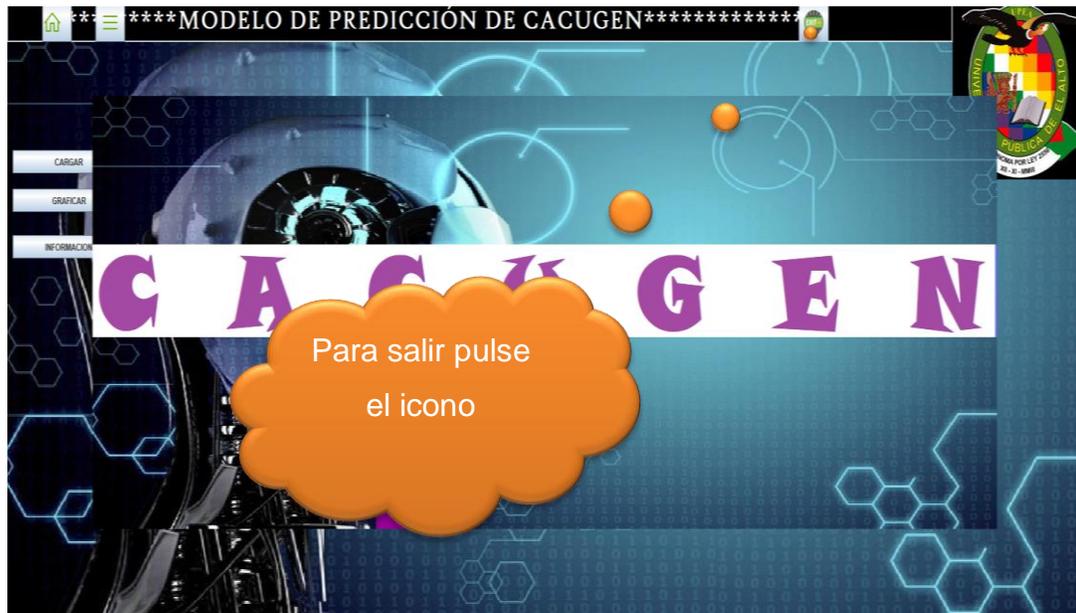
9. Calcular y Graficar la población

Posterior a la selección de la población el usuario puede visibilizar mediante el botón de graficar. Con esta acción se mostrará en el lado derecho de la pantalla las gráficas correspondientes de la población y su comportamiento respecto a la gestión.



10. Salir del Modelo de Predicción

Luego de consultar las opciones el usuario puede salir del Modelo de Predicción mediante el botón salir.



11. Conclusión.

El Modelo de Predicción CACUGEN, es un sistema muy amigable y vistoso para mostrar a la población como el cáncer de cuello uterino está presente en las familias de la Ciudad de El Alto.