UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

Medidas de similitud geométrica para la estratificación de territorios sobre el sistema de información geográfica QGis

Trabajo de diploma presentado para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Miguel Antonio Martínez Ochandarena

Tutores: Ing. Liset González Polanco

Ing. Yadian Guillermo Pérez Betancourt

Ing. Raniel Gé Vaillant

La Habana, 26 de junio de 2017

Luchar por la paz es el deber más sagrado de todos los seres humanos, cualesquiera que sean sus religiones o país de nacimiento, el color de su piel, su edad adulta o su juventud.

Fidel Castro Ruz

Declaración de autoría

Declaro ser autor de la presente tesis y se reconoce a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste se firma la presente a los 26 días del mes de Junio del año 2017.

Miguel Antonio Martínez Ochandarena

Ing. Liset González Polanco

Ing. Yadian Guillermo Pérez Betancourt

Ing. Raniel Gé Vaillant

AGRADECIMIENTOS

Agradezco a la Revolución Cubana por brindarme la posibilidad de formarme como Ingeniero.

A mi mamá, por ser la guía de toda mi vida; gracias por ser madre, hermana, amiga, mi todo.

A mi papá, por enseñarme de una manera u otra a ser el hombre trabajador, que soy hoy en día.

A mi familia, por todo su apoyo, en especial a mis tías primos y primas quienes siempre han estado pendiente a mi desarrollo escolar, apoyándome siempre en todo lo que se pudiese.

A mis hermanos, gracias por estar aquí.

A mis abuelos,por trasmitirme todas sus experiencias las buenas para aprender de ellas y las malas para no repetirlas, en especial a mi abuelo Juan Antonio el cual ya no esta presente, pero se que me acompaña todos los días.

A mis tutores, por su apoyo, confianza y todo el tiempo que me han dedicado.

Al tribunal por sus recomendaciones y sugerencias en este proceso.

A mi familia en la UCI, Norberto, Osbel, Ernesto los cuales son mis hermanos. A todos mi amigos que de una manera u otra también pusieron su granito de arena para que yo pudiera ser ingeniero en especial a mi hermano Yordanis.

A todos los profesores de la mejor facultad de la UCI.

A mis compañeros de aula y apartamento por soportarme en este tiempo.

A las tías del comedor, docente y apartamento.

DEDICATORIA

A mi mamá Barbara Inez Ochandarena Martínez.

RESUMEN

La estratificación de territorios basada en Sistemas de Información Geográfica es una herramienta muy utilizada en estudios salubristas por las facilidades para la visualización en mapas. Se realizó un diagnostico sobre las soluciones existentes para la estratificación, identificándose limitaciones en el contexto de los estudios de espacialidad en salud. Las medidas de similitud empleadas consideran las características con igual importancia y están enfocadas a los datos temáticos. Este tratamiento no permite describir relaciones espaciales sobre objetos y por tanto dificulta la incorporación del espacio en el proceso. Aunque se brinda la posibilidad de incluir datos de naturaleza espacial, lo hacen a partir de transformaciones, en la que luego son tratados como temáticos, esto contradice la primera ley de la geografía. En este trabajo se propone la incorporación de medidas de similitud geométricas entre polígonos sobre QGis que favorezcan la incorporación de la componente espacial en el proceso de estratificación de territorios. Se define la arquitectura y los principales patrones de diseño a utilizar. Se obtuvo un sistema que permite integrar datos de naturaleza espacial y temática para construir estratos; la solución se basa en las medidas de similitud estudiadas y contribuye a la identificación de riesgos de salud de los territorios cubanos y a la toma de decisiones en las entidades de salud. Se realizaron pruebas de software según la metodología de desarrollo seleccionada y se aplicó a un caso de estudio para valorar los resultados de la solución propuesta.

Palabras clave: componente espacial, estudios de espacialidad en salud, estratificación, medidas de similitud geométricas.

ÍNDICE GENERAL

AGRADECIMIENTOS II
DEDICATORIA III
SÍNTESIS IV
INTRODUCCIÓN 1
1 FUNDAMENTOS TEÓRICOS SOBRE LA ESTRATIFICACIÓN DE TERRITORIOS Y LAS MEDIDAS DE SIMILITUD GEOMÉTRICA 7
1.1 Marco teórico 7
1.1.1 Estratificación 7
1.1.2 Minería de datos 8
1.1.2.1 Minería de datos espaciales 8
1.1.3 Medidas de similitud geométrica 10
1.2 Análisis crítico de soluciones existentes 12
1.2.1 SIG que soportan el proceso de estratificación: 13
1.3 Herramientas, lenguajes, metodología y tecnologías a utilizar 17
1.3.1 Lenguaje de modelado 17
1.3.2 Herramienta de modelado 17
1.3.3 Lenguaje de programación 18
1.3.4 Entorno de desarrollo integrado 20
1.4 Metodología de desarrollo 21
1.4.1 Programación Extrema 21
1.5 Conclusiones parciales 23
2 MEDIDAS DE SIMILITUD GEOMÉTRICAS PARA LA ESTRATIFICACIÓN DE TERRITORIOS 24
2.1 Propuesta de medidas de similitud geométricas 24
2.1.1 Componentes del modelo 24

2.2 Diseño o implementación de las medidas de similitud geométricas. 26
2.2.1 Medida de similitud según el criterio de distancia 26
2.2.1.1 Algoritmo para el criterio de distancia 28
2.2.2 Medida de similitud según el criterio de densidad 29
2.2.2.1 Algoritmo para el criterio de densidad 30
2.2.3 Medida de similitud según el criterio de tamaño 31
2.2.3.1 Algoritmo para el criterio de tamaño 33
2.2.4 Medida de similitud según el criterio de conectividad 33
2.2.4.1 Algoritmo para el criterio de conectividad 34
2.3 Requisitos de software 35
2.3.1 Requisitos funcionales 36
2.3.2 Requisitos no funcionales 36
2.4 Fase de planificación 37
2.4.1 Historias de usuarios 37
2.4.2 Estimación de esfuerzos por historias de usuario 38
2.4.3 Plan de iteraciones 39
2.4.4 Plan de entrega 39
2.5 Fase de diseño 40
2.5.1 Tarjetas Clase-Responsabilidad-Colaboración 40
2.6 Arquitectura de software 41
2.6.1 Estilo arquitectónico a utilizar 41
2.7 Patrones de diseño 42
2.7.1 Patrones Generales de Software para la Asignación de Responsabilidades . 43
2.7.2 Patrones del Grupo de Cuatro 45
2.8 Diagrama Entidad-Relación 46
2.9 Conclusiones del capítulo 47
3 VERIFICACIÓN DE LA SOLUCIÓN PROPUESTA 48
3.1 Fase de implementación 48
3.1.1 Tarea de ingeniería 48

3.1.2 Estándares de codificación 49
3.2 Fase de pruebas 50
3.2.1 Pruebas de aceptación 51
3.2.2 Pruebas de caja blanca 52
3.2.3 Caso de estudio 56
3.3 Conclusiones del capítulo 60
CONCLUSIONES 61
RECOMENDACIONES 62
REFERENCIAS BIBLIOGRÁFICAS 63
GLOSARIO DE TÉRMINOS 69

ÍNDICE DE FIGURAS

1 Sistema de Información Geográfica para la planeación y el Ordenamiento Territorial Nacional 2
2 La UBPC Lázaro Romero de la EA Héctor Molina 3

1.1 Estudio realizado por John Snow, sobre el cólera asiático en el año de 1854 en la ciudad de Londres 10
1.2 Representación de las capas en un mapa 13

2.1 Modelo conceptual de la propuesta de solución 26
2.2 Ejemplo de polígono 32
2.3 Evidencia de la arquitectura del sistema 42
2.4 Evidencia del patrón Experto 43
2.5 Evidencia del patrón Creador 44
2.6 Evidencia del patrón Controlador 45
2.7 Diagrama Entidad-Relación. 47

3.1 Resultado de aplicar la prueba de aceptación 52
3.2 Código del método run() 53
3.3 Grafo de flujo del método 54
3.4 Resultado de aplicar la prueba de caja blanca 56
3.5 Mapa temático de la estratificación realizada utilizando una de las medidas de similitud propuesta (Distancia0). 58

ÍNDICE DE TABLAS

1.1 Tabla comparativa entre resultados 16

2.1 Historia de usuario: implementar criterio densidad 38
2.2 Estimación de esfuerzos por Historia de Usuario 38
2.3 Plan de duración de las iteraciones 39
2.4 Plan de duración de las entregas 39
2.5 Tarjeta CRC para la clase Territorio. 40
2.6 Tarjeta CRC para la clase similitud distancia. 41

3.1 Distribución de tareas de ingeniería 49
3.2 Tarea de ingeniería para obtener características de los polígonos. 49
3.3 Caso de prueba de aceptación para HU implementar densidad. 51
3.4 Caso de prueba para el camino básico # 1 55
3.5 Caso de prueba para el camino básico # 2 55
3.6 Resultados de la estratificación realizada utilizando una de las medidas de similitud propuesta (Distancia) 58
3.7 Comparación entre ambas estratificaciones. 59
3.8 Comparación entre ambas estratificaciones en cuanto al indice GAP. 60

INTRODUCCIÓN

El desarrollo alcanzado por los Sistemas de Información Geográfica (**SIG**) desde sus orígenes hasta la actualidad es significativo, fundamentado por la necesidad creciente de información geográfica. La gran popularidad de las tecnologías y los esfuerzos de desarrollo en diversas áreas como la programación, disponibilidad de la información, mayores prestaciones de las computadoras, ha contribuido a redefinir la disciplina, incorporando nuevas funcionalidades para darle soporte a estudios más específicos como la aplicación de los SIG para la evaluación de amenazas y riesgos, aplicacion práctica en los Estudios de Paisaje [1], pero dentro de los temas más favorecidos por este desarrollo se pueden destacar los siguientes:

Aplicaciones en el ámbito de la Administración Pública: El Sistema de Información Geográfica para la planeación y el Ordenamiento Territorial Nacional (**SIGOT**) se muestra en la figura 1, es una herramienta cuyo objetivo central es contribuir a una eficiente y oportuna toma de decisiones, apoyando a los actores–autoridades e instancias en el sistema de planeación a nivel nacional, regional y local, con información político-administrativa, socio-económica y ambiental georreferenciada que soporte la gestión del desarrollo.

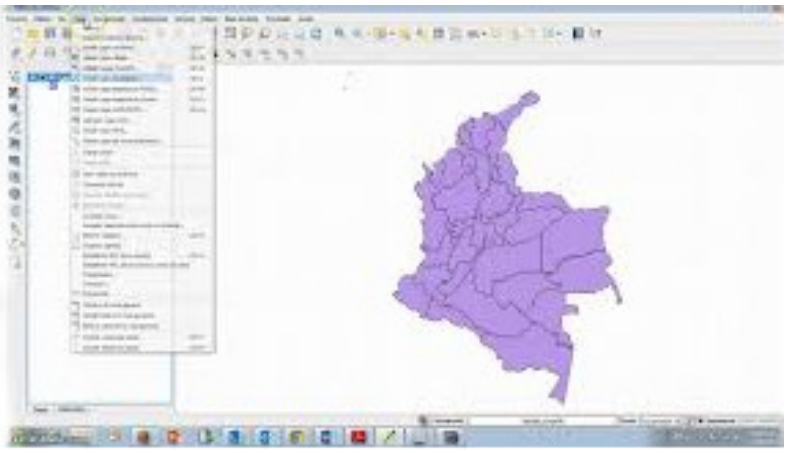

Figura 1: *Sistema de Información Geográfica para la planeación y el Ordenamiento Territorial Nacional*

Aplicaciones en el campo de las utilidades: Otro campo de aplicación con un fuerte desarrollo es el de las utilidades (traducción literal del inglés Utilities). Este suele incluir aquellos apartados referidos básicamente a redes de conducción de energéticos (gas, agua, electricidad entre otros). En muchos casos ha tenido un desarrollo paralelo al de la ingeniería por cada especialidad, dándose productos específicos e independientes de los SIG con propósito general [2].

En la agricultura este es de gran importancia ya que permite la evaluación de los suelos y su monitoreo como en el ejemplo que se muestra en la figura 2. El objetivo de este ejemplo es evaluar un problema ocasionado al suelo por la aplicación incorrecta de las tecnologías de riego, causantes de un fenómeno impactante para la agricultura: la salinización de los suelos.

Una de las regiones donde este fenómeno se presentó fue en las áreas del sur de la provincia Mayabeque. El sitio experimental se encuentra en la Unidad Básica de Producción Cooperativa (**UBPC**) Lázaro Romero de la Empresa Azucarera (**EA**) Héctor Molina, en el municipio San Nicolás de Bari. La UBPC está situada en los 22° 22´ de latitud norte y los 81° 56´ de longitud oeste; tiene una superficie de 1.272 ha, de las cuales 1320 ha están dedicadas a cultivar caña de azúcar y el resto a otros cultivos para el autoconsumo, como el arroz.

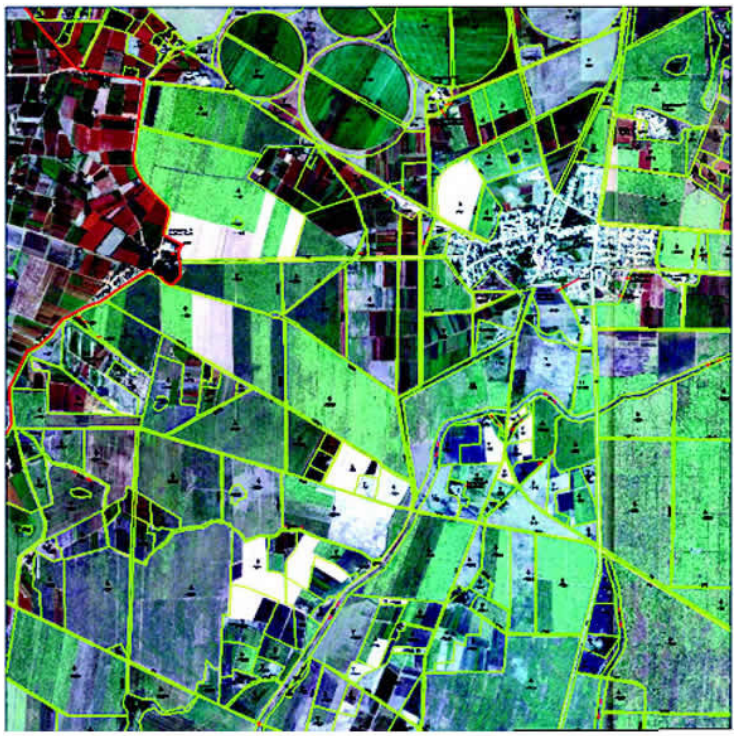

Figura 2: *La UBPC Lázaro Romero de la EA Héctor Molina*

Otra de las aplicaciones de gran importancia es en el análisis de la situación de la salud en los servicios y sus aportes para facilitar la toma de decisiones. Al priorizarse su desarrollo como un instrumento científico-metodológico en los espacios docentes, su aplicación constituye una necesaria contribución al desarrollo del Sistema Nacional de Salud. Actualmente, mediante las denominadas *transformaciones necesarias* del sistema, se promueve el desarrollo de herramientas y métodos para el análisis de la situación de salud, que permitan optimizar los servicios de salud. En la práctica un buen ejemplo de esto es la aplicación de los SIG para el análisis de la distribución espacial de problemas de salud o en los servicios a la población. Su empleo contribuye al fortalecimiento de la capacidad de análisis en materia de salud pública y epidemiológica, brindando información de utilidad para la toma de decisiones. Por otra parte, facilitan la identificación de las ubicaciones geográficas de establecimientos de salud y grupos de población que presentan mayor riesgo de enfermar o de morir prematuramente y por tanto requieran mayor atención preventiva, curativa o de promoción de salud.

El análisis estratificado es una de las herramientas con mayor potencial en este sector, muestra de esto lo constituye el trabajo desarrollado en la Facultad 3 de la Universidad de las Ciencias Informáticas [3]. Este, constituye un aporte para resolver algunas limitaciones existentes, entre ellas: el acceso limitado a los SIG por los costos que ellos implican, poco conocimiento de las herramientas, el elevado tiempo de formación en el área de los SIG, elementos que limitaban el trabajo de estas herramientas solo a llevar información a la cartografía, sin explotar la componente espacial de los datos en su totalidad.

Si bien este trabajo permitió integrar el análisis estratificado en los SIG, aún presenta limitaciones, pues las medidas de similitud empleadas consideran las características con igual importancia y están enfocadas a los datos temáticos. Este tratamiento no permite describir relaciones espaciales sobre objetos y por tanto dificulta la incorporación del espacio en el proceso. Aunque brinda la posibilidad de incluir datos de naturaleza espacial, como objetos puntuales o líneas, lo hace a partir de una transformación, en la que luego son tratados como temáticos, esto contradice la primera ley de la geografía que plantea: *todos los objetos en el espacio están relacionados, pero objetos más cercanos están más relacionados entre sí que objetos más distantes* [4].

A partir de la situación problemática identificada, se formula el siguiente **problema a resolver**: las medidas de similitud empleadas en el proceso de estratificación de territorios limitan la incorporación de la componente espacial en estudios salubristas.

El **objeto de estudio** para la investigación lo constituye la minería de datos espaciales y como **idea a defender** se plantea: si se implementan medidas de similitud geométrica entre polígonos sobre Qgis, se favorecerá la incorporación de la componente espacial en el proceso de estratificación de territorios.

Para dar solución al problema planteado se trazó como **objetivo general**: implementar medidas de similitud geométrica entre polígonos sobre QGis que favorezcan la incorporación de la componente espacial en el proceso de estratificación de territorios. El **campo de acción** estará enmarcado en las medidas de similitud.

El objetivo general se desglosó en los siguientes **objetivos específicos**:

1. Construir el marco teórico referencial de la investigación, relacionado con la minería de

datos espaciales.

2. Diseñar medidas de similitud geométricas sobre polígonos.
3. Implementar un componente que dé soporte a las medidas de similitud geométricas para su utilización en la estratificación de territorios.
4. Verificar la solución informática propuesta aplicando diferentes pruebas y métricas.

En la consecución de los objetivos trazados se utilizaron los siguientes **métodos**:

- Análisis-síntesis: para identificar las medidas de similitud geométrica entre polígonos existentes profundizando en el estudio de cada una de ellas y luego sintetizarlas en la solución propuesta.
- Histórico-lógico:para analizar la evolución de los enfoques empleados en las técnicas de medida de la similitud geométrica entre polígonos existentes.
- Modelado: se emplea para mostrar los diversos diagramas que se construyen como resultado del proceso de ingeniería de software.

Estructura del trabajo

El trabajo está estructurado en introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y glosario de términos. A continuación se muestra una breve descripción de cada capítulo.

Capítulo 1: Fundamentos teóricos sobre la estratificación de territorios y las medidas de similitud geométrica.

En este capítulo se presentan elementos teóricos relacionados con la minería de datos espaciales para lograr un mejor entendimiento del trabajo a desarrollar. Se hace una valoración comparativa sobre las medidas de similitud geométricas en la estratificación de territorios. Además, se realiza un estudio de la metodología, herramientas, tecnologías y lenguajes a utilizar en el desarrollo de la solución.

Capítulo 2: Diseño e implementación de la propuesta de solución.

En este capítulo se realiza una descripción general de la solución propuesta, especificándose las fórmulas utilizadas para la misma, se trazan los requisitos funcionales que se tendrán en cuenta para su implementación, y se detallan aspectos relacionados con el diseño y la arquitectura.

Se especifican los patrones del diseño aplicados, los artefactos derivados de la metodología de desarrollo de software seleccionada y se describe la etapa de implementación.

Capítulo 3: Validación de la solución propuesta.

Se abordan todos los elementos relacionados con la implementación, se elaboran y documentan las pruebas para verificar el funcionamiento adecuado de la solución implementada. Finalmente se presentan los resultados obtenidos a partir de la aplicación a un caso de estudio.

1. FUNDAMENTOS TEÓRICOS SOBRE LA ESTRATIFICACIÓN DE TERRITORIOS Y LAS MEDIDAS DE SIMILITUD GEOMÉTRICA

En este capítulo se presentan un conjunto de elementos que conforman los fundamentos teóricos relacionados con el objeto de estudio de la investigación. Se realiza un estudio sobre las medidas de similitud geométrica empleadas en los procesos de estratificación territorial. Por último se analizan las tecnologías, herramientas informáticas y metodologías a utilizar en el proceso de desarrollo del software.

1.1. Marco teórico

Con el objetivo de facilitar la comprensión del alcance de la investigación, en el presente capítulo se exponen conceptos fundamentales asociados al dominio del problema planteado. Además, se realiza un análisis detallado del estado del arte que precede a la realización de este trabajo y que contribuye a esclarecer su objeto de estudio.

1.1.1. Estratificación

Según el diccionario de la Real Academia de la Lengua Española, la estratificación constituye la acción o efecto de disponer algo en un conjunto de elementos con determinados caracteres comunes, que se ha integrado con otros conjuntos previos o posteriores para la formación de una entidad o productos históricos de una lengua. Sin embargo en el ámbito científico esta se define como un conjunto de analogías que dan lugar a subconjuntos de unidades agregadas, denominadas estratos. Un estrato por tanto, es un conjunto de unidades que presentan uno o varios parámetros, que los hacen similares entre sí y a la vez se diferencia de unidades correspondientes a otros estratos. Es decir, que en cada estrato existe una igualdad interna con

diferencias o desigualdades externas [5]. Pudieran ser abordadas otras definiciones, pero todas concuerdan en que la estratificación es la separación de datos en categorías o clases que permite aislar la causa de un problema, identificando el grado de influencia de ciertos factores en el resultado de un proceso.

- **Estratificación territorial:** La estratificación territorial es un proceso que permite dimensionar espacialmente los eventos a través de un proceso de agregación y des-agregación de los territorios a evaluar. A partir de variables seleccionadas para dichos territorios que permitan agregaciones (por homologías de las características) o des-agregaciones (por heterogeneidades de estas) [6]. Estas variables utilizadas son seleccionadas de los datos dispuestos en el territorio a evaluar, evidenciándose la real importancia que tienen los datos y lo vital su correcto uso.

1.1.2. Minería de datos

La minería de datos está definida como "Un proceso no trivial de identificación válida, novedosa, potencialmente útil y atendible de patrones comprensibles que se encuentran ocultos en los datos"[7]. La minería de datos en esencia realiza tareas como reunir las ventajas de varias áreas como la Estadística, la Inteligencia Artificial, la Computación Gráfica, las Bases de Datos y el Procesamiento Masivo, principalmente usando como materia prima las bases de datos [8].

1.1.2.1. Minería de datos espaciales

La minería de datos espaciales se puede definir como el proceso automático o semiautomático de seleccionar, explorar, modificar, visualizar y valorar grandes volúmenes de datos espaciales con el objetivo de descubrir conocimientos [9]. Ella permite encontrar a través de diferentes técnicas y herramientas [10], patrones potencialmente útiles en bases de datos espaciales, este tipo de bases de datos soportan operaciones eficientes para la realización de tareas comunes como búsquedas por vecindad y uniones espaciales. Sin embargo, no almacenan explícitamente patrones o reglas que determinan las relaciones espaciales entre los objetos y algunas características no espaciales.

Inicialmente se podría pensar que la minería de datos espacial comparte las mismas técnicas

utilizadas en la minería de datos tradicional. Pero debido a la complejidad de los datos espaciales, que comprenden objetos como puntos, líneas, polígonos y las relaciones inherentes a su naturaleza. Es necesario pensar en técnicas que vayan en completa concordancia con el problema a tratar, pues la aproximación tradicional difiere a la aproximación espacial, por factores como [11]:

- El hecho que la primera asume características como la independencia existente en la distribución de los datos, que viola la primera ley de la geografía (todo se encuentra relacionado con todo lo demás, pero los objetos cercanos se encuentran mayormente relacionados que los objetos distantes).
- Los tipos de datos complejos.
- La existencia de correlación entre características espaciales.

El objetivo de la minería de datos espacial es permitir el hallazgo automatizado de patrones (hipótesis) inesperados, que serán validados por expertos del área estudiada. Ejemplos como el hallazgo de la relación entre un suministro de agua y las muertes ocasionadas por el cólera asiático en el año de 1854 en Londres. La relación existente entre la composición con cierto grado de fluoruro del agua suministrada y la buena salud dental de los habitantes de Colorado Spring, permiten visualizar el gran potencial de la minería de datos espaciales [12].
En el estudio realizado por John Snow, sobre el cólera asiático en el año de 1854 en la ciudad de Londres 1.1 se puede observar la relación de las muertes causadas por la epidemia y un suministro de agua específico.

Figura 1.1: *Estudio realizado por John Snow, sobre el cólera asiático en el año de 1854 en la ciudad de Londres*

Como antes se menciona una de las funciones en la minería de datos es la creación de grupos, o *clustering* en ingles, basándose en algunas reglas de similitud previamente definidas. En la presente investigación estas reglas están conformadas por las medidas de similitud geométricas para favorecer la incorporación de la componente espacial en el proceso de estratificación.

1.1.3. Medidas de similitud geométrica

En lenguaje cotidiano, cuando se habla de semejanza, casi siempre se hace referencia al concepto más general de parecido : color “parecido”, tamaño “parecido”, forma “parecida”, entre otros. En cambio, en matemática el concepto de semejanza está muy ligado al concepto de proporcionalidad, por ello se dice que dos objetos son semejantes si tienen una proporción entre

ellos.

La semejanza geométrica implica de modo estricto que se cumpla que la relación entre dimensiones homólogas de un modelo y un prototipo sean iguales. Un modelo y un prototipo son geométricamente similares si todas las dimensiones del cuerpo en cada una de las direcciones de los ejes coordenados, se relacionan mediante la misma escala de longitudes [13].

En los SIG la similitud es particularmente importante debido a la dificultad para obtener representaciones satisfactorias de los fenómenos geográficos y a la variedad de formalizaciones que existen de las propiedades espaciales tales como su forma, localización y relaciones espaciales [14].

En correspondencia con el tipo de información que se represente existen distintas definiciones de similitud. Teniendo en cuenta que un caso viene descrito por un conjunto de atributos, la aproximación más básica para el cálculo de la similitud consiste en contabilizar los valores iguales en los atributos comunes de los casos a comparar [15].

La similitud semántica: es fundamental para el procesamiento semántico de datos geoespaciales. Establece el grado de interoperabilidad semántica entre los datos [16].

Medida de semejanza entre objetos 3d con teoría de invariantes y con Algoritmo Húngaro1 [17].

Algoritmo Húngaro: Este algoritmo se usa para resolver problemas de minimización, ya que es más eficaz que el empleado para resolver el problema del transporte por el alto grado de degeneración que pueden presentar los problemas de asignación.

Modelo de datos: también juegan un papel importante ya que estos definen la manera en que se van a representar las características espaciales en un SIG. Los tipos de modelos son: modelo vector y modelo raster. Mientras que los sistemas raster están predominantemente orientados al análisis espacial, los vectoriales son eficientes en el almacenamiento de mapas, ya que solo distinguen entre límites de características, y no lo que existe en el interior de las mismas [18].

Las medidas de similitud a utilizar dependen del análisis posterior llevado a cabo con los datos, estos deben corresponder con lo necesitado por las medidas para su correcta evaluación. Por lo cual se pudiera tener en cuenta diferentes criterios para realizar dicha investigación.

Como la investigación está centrada en la necesidad de estratificar territorios según su

componente espacial, se plantea que las medidas de similitud pueden ser geométricas. Las de similitud geométricas se pueden definir como un conjunto de parámetros , atributos geográficos, o “criterios“ (distancia, área, perímetro, densidad) ligados a objetos enmarcados en un mapa, que posibilita su comparación. De esta forma se puede establecer cierta similitud entre estos, en dependencia de la relación arrojada por estos atributos.

La correcta selección de los criterios es de suma importancia, dado a que estos pueden variar en la misma cantidad que varían las operaciones matemáticas sobre los polígonos. Pero no todos tributan a una correcta solución, pues estas medidas en específico han de contribuir a la mejora de los estratos resultantes.

Como en esta investigación se aborda la dinámica espacial desde estudios salubristas, se decidió abordar los criterios de:

- distancia
- densidad
- tamaño
- conectividad

1.2. Análisis crítico de soluciones existentes

En este epígrafe se realiza un análisis crítico se soluciones existentes, así como los principales conceptos asociados a los sistemas de información geográfica. En especial el modelo de datos que utiliza y el tipo de informacion geográfica manejada.

Las herramientas de desarrollo de SIG se ajustan a una de las dos filosofías o enfoques fundamentales que existen a la hora de ver la relación entre información alfanumérica y geográfica. La filosofía seguida por la herramienta condiciona algunas decisiones de diseño en el modelo de datos usado, así como la gestión y almacenamiento de los mismos, introduciendo con ello ciertas restricciones y limitaciones que han de ser tomadas en cuenta a la hora de elegir la herramienta a utilizar [19].

La mayoría de las aplicaciones SIG requieren algún **modelo de datos** de entrada para poder utilizarse estos modelos están determinados por las necesidades que se tengan a la hora de utilizar los mismos. Existen fundamentalmente dos modelos de datos *raster* y *vectorial* [20].

los cuales se visualizan en la figura 1.2.

El modelo vectorial focaliza su interés en las entidades, en su posicionamiento sobre el espacio. Para modelar las entidades del mundo real se utilizan tres tipos de objetos espaciales: puntos, líneas y polígonos (áreas); por ejemplo, un lago puede representarse por medio de un polígono. Así, los objetos no son más que representaciones digitales de las entidades, las cuales no siempre son elementos visibles; por ejemplo, las divisiones administrativas o censales [20].

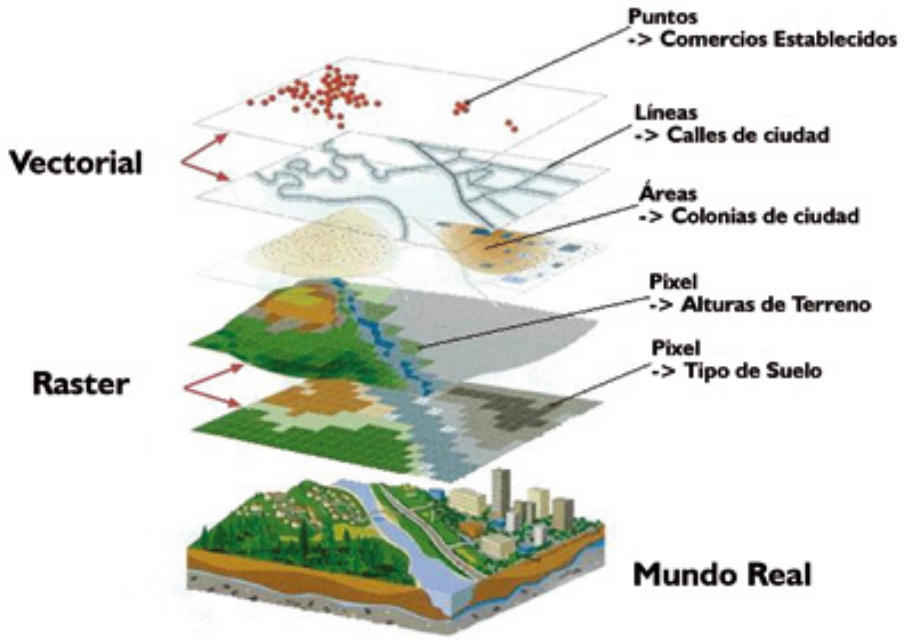

Figura 1.2: *Representación de las capas en un mapa*

1.2.1. SIG que soportan el proceso de estratificación:

Para el estudio de las principales herramientas existentes que soportan la realización de procesos de estratificación. Se consideró para el análisis,la forma de representación de los resultados, los modelos de datos que analizan y el tipo de información geográfica que manejan la cual puede clasificarse en dos categorías, dependiendo de si esta, representa características del espacio geográfico (atributos referentes al espacio geográfico) o por el contrario representa propiedades de los objetos gestionados (atributos referentes a los objetos) [19].

Herramienta para la estratificación de municipios en zonas de riesgo para la salud.

La herramienta es parte del programa atención integral de los servicios de salud de Hidalgo; la misma permite realizar estructuraciones de los municipios del estado de Hidalgo de forma automatizada. Brinda al usuario la opción de obtener estratificaciones libres o restringidas a un cierto número de grupos. Se basa en técnicas del Reconocimiento Lógico Combinatorio de Patrones (RLCP), permitiendo manejar diversas formas en la presentación de resultados, en forma gráfica y tabular, así como brindar información acerca de qué indicadores influyen más en la obtención del promedio de riesgo total de cada uno de los grupos formados [21].

Desventajas

- La herramienta es privativa.
- Solo permite estratificar los 84 municipios que conforman el estado Hidalgo.
- Analiza solo indicadores estadísticos por lo que no tiene en cuenta la componente espacial.
- Solo utiliza el modelo de datos raster.

Estratificador del Instituto Nacional de Estadística y Geografía de Méjico INEGI.

La herramienta permite construir agrupaciones o estratificaciones de áreas geográficas en base a información estadística. El sistema brinda al usuario la opción de seleccionar las variables que muestran mayor afinidad con el tema de su interés, y elegir uno o más procedimientos de estratificación; de este modo será posible disponer de dos o más estratificaciones alternativas. Incluye, una serie de ayudas gráficas que permiten al analista realizar comparaciones y decidir cuál de todas las combinaciones de datos y métodos satisface de la mejor manera sus objetivos. Utiliza para la clasificación de los territorios tres algoritmos de agrupamiento, K-medias, Mulvar y MClust, permitiendo representar los resultados mediante diferentes gráficos que pueden ser: mapas temáticos, burbujas, y centroides [22].

Desventajas

- Analiza solo indicadores estadísticos por lo que no incorpora la componete espacial.
- Solo utiliza el modelo de datos raster.
- No brinda información sobre el riesgo de salud de los territorios evaluados.
- Tiene definido un conjunto estático de datos, lo que restringe el campo de análisis.

- No permite almacenar las estratificaciones que se realizan.

SIG para la secretaría de planeación del municipio de Guadalajara de Buga.

La herramienta tiene como propósito la organización de la información cartográfica y aprovechar las herramientas que suministran los SIG, para lograr detectar las diversas alteraciones dentro de lo concerniente a usos del suelo, organización vial y riesgos. Este sistema cuenta con un módulo para representar la diferencia social en la distribución de los bienes de acuerdo con los lineamientos del DANE1, para ello se aplica una metodología de estratificación logrando la separación de los resultados en estratos y representándolos en mapas temáticos [23].

Desventajas

- Analiza solo indicadores estadísticos por lo que no posibilita la incorporación de la componete espacial.
- Solo analiza el modelo de datos raster.
- Tiene definido un conjunto estático de datos, lo que restringe el campo de análisis.
- No permite almacenar las estratificaciones que se realizan.
- Presenta un conjunto estático de estratos.
- Los datos se agrupan de forma manual.

Estratificación del Riesgo en Euskadi.

El objetivo que este modelo busca es la identificación de personas con riesgo de requerir un gran consumo de recursos sanitarios en el futuro, lo que posibilita plantear intervenciones proactivas que se ajusten a sus necesidades de cuidados sanitarios futuras.La utilización de información procedente de diagnósticos y prescripciones consigue paliar algunas limitaciones atribuibles a los sistemas de información en uso y obtiene mejores resultados que otros sistemas.Utiliza modelos estadísticos basados en: el uso previo de recursos sanitarios, variables demográficas, socioeconómicas y clínicas.

Desventajas

- Analiza solo indicadores estadísticos por lo que no posibilita la incorporación de la componete espacial.
- Contempla un conjunto muy reducido de indicadores estadísticos lo que limita el análisis.

- No está orientado a la estratificación territorial.

Software QGIS

Quantum GIS (QGIS, por sus siglas en inglés) es una aplicación SIG con grandes potencialidades para la edición de mapas, multiplataforma y desarrollado utilizando Qt Toolkit1 y C++. Ofrece muchas características SIG, entre las que se encuentran.

- Permite crear, editar, administrar y exportar mapas vectoriales en varios formatos.
- Permite realizar análisis de datos espaciales de PostgreSQL/PostGIS2 usando el complemento de Python fTools3.
- Incorpora a través de las herramientas de procesado, decenas de comandos de GRASS y SAGA para realizar análisis espacial tanto con datos vectoriales como ráster.
- Permite la integración de plugins desarrollados en Python a través del módulo PyQGis.
- Presenta una interfaz amigable.

Tabla 1.1: *Tabla comparativa entre resultados*

Herramientas / Factores de comparación	Herramienta para la estratificación de municipios en zonas de riesgo para la salud	Estratificador INEGI	SIG para la secretaría de planeación del municipio de Guadalajara de Buga	Estratificación del Riesgo en Euskadi	QGIS
Algoritmos de agrupamiento	MClust, K-Means	*K-Means, Mulvar, MClust*	Los datos se agrupan de forma manual	Modelo de regresión	K-Means y Matriz de similitud
Modelo de de datos	raster	raster	raster	raster	raster
Tipo de información geográfica	Estadístico	Estadístico	Estadístico	Estadístico	Estadístico
Tipo de licencia	Privativo	Libre	Libre	Libre	Libre

Como se puede apreciar al finalizar el estudio de la herramientas existentes todas estas tienen algo en común, no incluyen en el análisis de sus datos al modelo vectorial y por tanto tampoco a la componente espacial, que puede ser de mucha utilidad en el proceso de estratificación, ya que influye de forma directa en la propagación de las enfermendades, basandose en la primera ley de la geografía, la cual plantea que *todos los objetos en el espacio estan relacionados pero*

a mayor cercanía mayor relación y a menor cercanía menor relación.

1.3. Herramientas, lenguajes, metodología y tecnologías a utilizar

En todo proceso investigativo es necesario la utilización de sistemas que permitan dar soporte, organizar, facilitar, agilizar y automatizar las tareas generadas durante el transcurso de la investigación. Las herramientas, lenguajes y tecnologías empleadas que se describen a continuación son de vital importancia para su correcta realización.

1.3.1. Lenguaje de modelado

UML es el acrónimo de Lenguaje Unificado de Modelado, este es el lenguaje estándar para visualizar, especificar, construir y documentar los artefactos de un sistema, utilizándose para el modelado del negocio y sistemas de software [24].

Se selecciona UML porque ofrece un estándar para describir los modelos, incluyendo aspectos conceptuales como procesos de negocio, funciones del sistema, expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. Lo que le da ventaja por sobre otros lenguajes de modelado como:

- Cadena de proceso guiada por eventos o (**EPC**) por sus siglas en inglés (*Event-driven Process Chain*).
- Notación para la Modelación de Procesos de Negocio por sus siglas en inglés (*BPMN*).

1.3.2. Herramienta *Computer Aided Software Engineering* (CASE)

CASE es el acrónimo de *Computer Aided Software Engineering*, las herramientas CASE son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software [25].

Visual Paradigm

Es una herramienta de diseño UML y CASE UML diseñada para ayudar al desarrollo de software. Ofrece un paquete completo necesario para la captura de requisitos, la planificación del software y de pruebas, así como el modelado de clases y de los datos [26]. Las principales características de la herramienta son:

- Soporta las últimas versiones del UML.
- Posee un poderoso generador de documentación y reportes en formato PDF, HTTP y JPG.
- Proporciona soporte para varios lenguajes en la generación de código e ingeniería inversa como: Java, C++, CORBA IDL, PHP, Ada y Python.
- Disponibilidad en múltiples plataformas (Windows, Linux).
- Capacidades de ingeniería directa e inversa.

Se selecciona Visual Paradigm for UML en su versión 8.0 como herramienta para el modelado UML, pues permite trabajar de forma colaborativa, hacer un trabajo organizado y ágil. Posibilita la realización de los diagramas necesarios para el desarrollo y mejor entendimiento de la aplicación. Permite realizar ingeniería inversa a partir del código fuente. Al ser seleccionado el lenguaje de modelado UML, es conveniente tener en cuenta su vinculación con Visual Paradigm, resaltando que este último presenta abundante documentación y demostraciones interactivas.

1.3.3. Lenguaje de programación

Los lenguajes de programación son un conjunto de símbolos junto a un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Constan de un léxico, una sintaxis y una semántica [27].

Partiendo de las características de la aplicación, se hace necesaria la selección de un lenguaje mediante el cual se pueda cumplir con los requisitos que se propongan. Actualmente existen muchos lenguajes para el desarrollo de aplicaciones, surgidos a partir de las tendencias y necesidades de los escenarios. El análisis se centró fundamentalmente en el lenguaje Python.

Python

Se trata de un lenguaje interpretado o de *script*, con tipado dinámico, multiplataforma y orientado a objetos, que permite la programación imperativa, funcional y orientada a aspectos [28].

Se seleccionó Python en su versión 2.7.5 pues al seleccionar Qgis como el software que soportará la integración de la solución, el lenguaje de programación más eficiente y conveniente para utilizar es el anteriormente mencionado; este SIG a partir de su versión 0.9 trae soporte del lenguaje Python que junto con el módulo PyQT4 entrega una solución óptima al desarrollo de *plugins* e interfaces gráficas de usuario.

Es importante tener en cuenta que su sintaxis es simple, clara y sencilla logrando de esta manera que los programas elaborados en este lenguaje parezcan pseudocódigo. Además el tipado dinámico, el gestor de memoria, la gran cantidad de librerías disponibles y la potencia del lenguaje, entre otros, hacen que desarrollar una aplicación en Python sea sencillo y rápido.

PyQt

PyQt es un conjunto de enlaces Python para la biblioteca gráfica Qt. El módulo está desarrollado por la firma británica Riverbank Computing y se encuentra disponible para Windows, GNU/Linux y Mac OS bajo diferentes licencias. PyQt se distingue por su sencillez, posee un número importantes de herramientas que gestionen su manipulación y por su posibilidad de adecuarse a las distintas plataformas de software [29].

Se selecciona PyQt en su versión 4.0 en el desarrollo de la herramienta informática ya que se escogió Python como lenguaje de programación , y también permite la creación de una interfaz visual sencilla, posee los componentes visuales necesarios para su desarrollo, así como una abundante documentación y ejemplos.

Qt Designer

Qt Designer es una herramienta que permite acelerar el desarrollo de interfaces multilenguaje debido a que genera un archivo XML cuyo contenido es el formato de dicha interfaz, pudiéndolo

convertir con los programas pertinentes a cada lenguaje. Esta herramienta provee características muy poderosas como la previa visualización de la interfaz, soporte para widgets y un editor de propiedades con gran variedad de opciones.

En correspondencia con la elección anterior de PyQt, se ha decidido emplear Qt Designer en su versión 4.7.4 como elemento que soportará el diseño de las interfaces. Su utilización permite la creación de las interfaces visuales de la aplicación de forma sencilla, además de la fácil manipulación de las variables de configuración de cada una de ellas.

1.3.4. Entorno de desarrollo integrado

Un entorno de desarrollo integrado (IDE, por sus siglas en inglés) es una herramienta que permite a los desarrolladores de software escribir sus programas en uno o más lenguajes. Consiste básicamente en una plataforma en la que se integran un editor de código, un compilador[1], un depurador[2] y una interfaz gráfica de usuario [30].

Pycharm

Pycharm es un editor de código inteligente que proporciona soporte de primera clase para los lenguajes de programación: Python, JavaScript, CoffeeScript, TypeScript, HTML/CSS, Cython, lenguajes de plantilla, AngularJS y Node.js, y otros menos utilizados. Pycharm funciona en las plataformas Windows, Mac OS y Linux con una única clave de licencia, también ofrece un espacio de trabajo con colores personalizables y atajos de teclado.

La decisión de seleccionar como IDE, Pycharm en su versión 3.4, está dada a que ofrece auto-completación inteligente de código, comprobación de errores sobre la marcha, soluciones rápidas y fácil navegación en el proyecto. Pycharm mantiene la calidad del código bajo control con chequeos, asistencia a pruebas, refactorizaciones inteligentes, y una serie de inspecciones, lo que ayuda a escribir un código limpio y fácil de mantener [31].

[1]Compilador: es programa informático que traduce un programa escrito en un lenguaje de programación a otro lenguaje de programación

[2]Depurador : es programa usado para probar y eliminar los errores de otros programas.

1.4. Metodología de desarrollo

El desarrollo de un software es de gran importancia por ello, se debe contar con un proceso bien detallado, para esto se necesita aplicar una metodología que sea capaz de llevar a cabo el control total del producto. Las metodologías de desarrollo de software surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un producto de software. Dichas metodologías pretenden guiar a los desarrolladores, sin embargo los requisitos de un software son muy variados y cambiantes, y se ha dado lugar a que exista una gran variedad de ellas [32].

Las metodologías se dividen en dos grupos, tradicionales (pesadas) y ágiles (ligeras). Las tradicionales, se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requiere una extensa documentación, pretendiendo prever todo de antemano, además dependen de un equipo de desarrollo bastante grande. En las ágiles es más importante lograr que un producto de software se desarrolle con la calidad requerida, que realizar una buena documentación. En este tipo de metodología el cliente está presente en todo momento y colabora con el proyecto, que posee un equipo de desarrollo pequeño [32].

1.4.1. Programación Extrema

Programación extrema (**XP**, por sus siglas en inglés) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Además se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico [33].

Características de la metodología XP :

- XP es una metodología “liviana” que no tiene en cuenta la utilización de casos de uso y la generación de una extensa documentación.

- XP tiene asociado un ciclo de vida y es considerado a su vez un proceso.
- La tendencia de entregar software en espacios de tiempo cada vez más pequeños con exigencias de costos reducidos y altos estándares de calidad.
- XP define Historias de Usuario (**HU**) como base del software a desarrollar, estas historias las escribe el cliente y describen escenarios sobre el funcionamiento del programa. A partir de las HU y de la arquitectura perseguida se crea un plan de liberaciones entre el equipo de desarrollo y el cliente [34].

Fases de la metodología XP:

- Planificación: durante esta etapa se lleva a cabo el proceso de identificación y confección de las HU.
- Diseño: durante esta etapa se crea un diseño evolutivo que va mejorando incrementalmente y que permite hacer entregas pequeñas y frecuentes de valor para el cliente, basado principalmente en el desarrollo de las tarjetas Clase-Responsabilidad-Colaboración (**CRC**).
- Desarrollo: en esta fase se realiza la implementación de las HU que fueron seleccionadas por cada iteración. Al inicio se lleva a cabo un chequeo del plan de iteraciones por si es necesario realizar modificaciones. Como parte de este plan se crean tareas de ingeniería para ayudar a organizar la implementación exitosa de las HU.
- Pruebas: esta fase permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones. XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñadas por los programadores, y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñada por el cliente final.

El ciclo de desarrollo consiste en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.

2. El programador estima el esfuerzo necesario para su implementación.

3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.

4. El programador construye ese valor de negocio.

5. Vuelve al paso 1 [33].

A partir del estudio de XP, se concluye que responde a las necesidades principales de tiempo, entorno y cantidad de programadores, e incluye al cliente como parte fundamental del equipo de desarrollo. Además, se preocupa más en el avance exitoso del producto que en generar una documentación detallada del mismo, siendo capaz de adaptarse a los cambios de requisitos en cualquier punto del ciclo de vida del proyecto. Estos elementos demuestran que es una metodología factible para guiar el proceso de desarrollo de la solución, por lo que se decide incluir en la propuesta.

1.5. Conclusiones parciales

Con el desarrollo de este capítulo se obtuvo un mejor entendimiento del problema existente a partir del análisis de los principales conceptos asociados a la solución. El estudio de los *plugins* existentes hoy en día para los SIG permitió identificar las fortalezas y debilidades que conlleva el trabajo con Qgis y la necesidad de mejorar una de sus funcionalidades con el fin de permitir la incorporación de la componente espacial en el proceso de estratificación del territorio. Para la implementación de la solución se analizaron un grupo de herramientas y metodologías de desarrollo que tributan a un producto de software basadas en licencias de software libre, encaminado a obtener un producto de alta independencia tecnológica y utilizable en diferentes plataformas. Finalmente se escogió la metodología XP para guiar el proceso de desarrollo de la solución.

2. Medidas de similitud geométricas para la estratificación de territorios

En este capítulo se describen los elementos relacionados con el desarrollo de un sistema informático para la estratificación de territorios y las medidas de similitud geométrica. Se especifican los requisitos de software. Se obtienen los artefactos correspondientes a las fases de planificación y diseño de la metodología seleccionada. Se define la arquitectura y los principales patrones de diseño utilizados en el desarrollo de la solución. Se detallan las tareas de ingenierías que conforman cada HU definida en la fase de planificación. Se establece el estándar de codificación que se utilizará en el desarrollo de la solución.

2.1. Propuesta de medidas de similitud geométricas

En la presente investigación para realizar el proceso de estratificación se propone la utilización de indicadores estadísticos definidos por el usuario y emplear la naturaleza geométrica de los datos a través de los criterios cartográficos siguientes:

1. Distancia
2. Composición
3. Tamaño
4. Conectividad

2.1.1. Componentes del modelo

Extracción: el primer paso asegura la necesidad de seleccionar, extraer, construir características de cada entidad a comparar. Dicho enfoque requiere conocer a priori, lo significativo de la entidad en cuestión, lo restante, por lo tanto, no es información necesaria. La justificación

es la minimización del costo computacional de la función de similitud, al considerar menos información de cada entidad para el proceso de comparación.

Esta extracción se le aplica a los mapas que se vayan a analizar. Para esto el QGIS carga el mapa y lo divide en capas o layers separando los datos vectoriales de los raster. En esta etapa se obtienen los datos geométricos de todos los componentes del mapa. En otras palabras esta parte del modelo se limita solo a la obtención de los datos. Para lo que se utiliza una clase llamada qgsGeometry la cual mediante uno de sus métodos provee a la etapa siguiente las características geométricas de cada uno de los elementos de la capa que se analice en ese momento.

Similitud: el criterio de comparación, es el proceso o acto de medir, existe gran relación entre las entidades a comparar y el criterio de comparación. Según importantes autores , medir es "el acto o proceso de asignar números al fenómeno de acuerdo a alguna regla", todo aquel fenómeno que no pueda ser medido, no puede ser comparado [35]. Estos indican la dependencia del criterio de comparación con el conjunto de entidades a comparar.

En este problema en específico la similitud viene enlazada con las reglas que la determinan, las cuales tributan o fueron creadas para optimizar las situaciones posibles en función de la salud. Para llevar a cabo este proceso, primeramente se elige el criterio geográfico (distancia, perímetro, tamaño, densidad) por el cual se realizará la comparación y así según su elección se gestiona el cómputo de la medida de similitud. Todo esto basándose en los datos de la capa superior, obteniéndose el grado de similitud entre dos polígonos, lo cual se le provee a la capa posterior.

Integración: la integración no es más que el proceso en cual se aplican los resultados obtenidos (similitud entre dos polígonos) a los mapas del QGIS acoplando así los resultados matemáticos con los de la aplicación, donde según los valores obtenido les asigna un estrato a los polígonos en cuestión. finalmente se obtiene un mapa con la representación geográfica de los territorios a estratificar pero no solo con los elementos semánticos, sino también con los geométricos.

En la figura 2.1 se presenta el modelo conceptual que guiará la propuesta de solución.

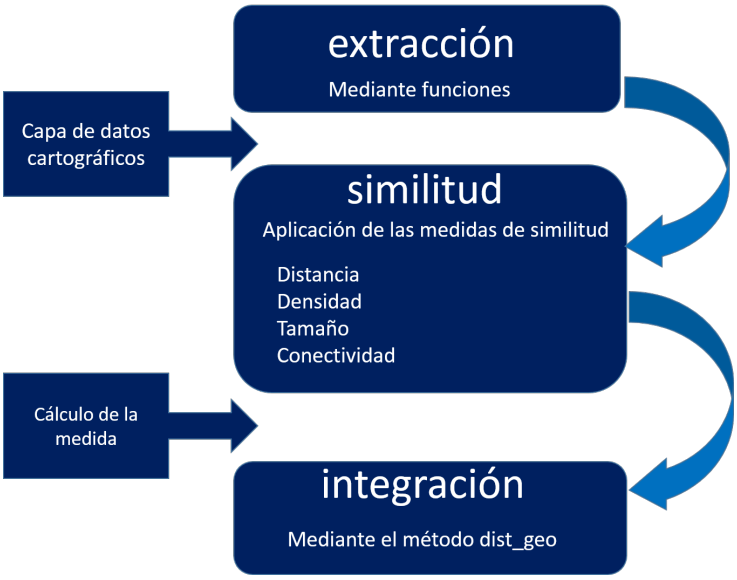

Figura 2.1: *Modelo conceptual de la propuesta de solución.*

2.2. Diseño o implementación de las medidas de similitud geométricas.

A continuación se describen las medidas de similitud geométricas atendiendo a varios criterios.

2.2.1. Medida de similitud según el criterio de distancia

Muchas medidas de distancia no son medidas porque no cumplen los requisitos. Por ejemplo, las pseudométricas no cumplen las condiciones: distancias no negativas ni la identidad de indiscernibles. Las quasimétricas no cumplen la condición de simetría, y por último las semimétricas no cumplen la desigualdad triangular.

Definición Formal de Distancia

Una definición formal de distancia $D(X_1, X_2)$

No negativo:

$$D(X_1, X_2) \geq 0 \rightarrow \forall X_1, X_2 \in X \tag{2.1}$$

Simetría:

$$D(X_1, X_2) = D(X_2, X_1) \rightarrow \forall X_1, X_2 \in X \tag{2.2}$$

Desigualdad triangular:

$$D(X_1, X_2) \leq D(X_1, X_3) + D(X_3, X_2) \rightarrow \forall X_1, X_2, X_3 \in X \tag{2.3}$$

Axioma de coincidencia o identidad de indiscernibles:

$$D(X_1, X_2)) = 0 \Leftrightarrow X_1 = X_2 \tag{2.4}$$

A continuación se muestras las siguientes consideraciones especiales para las medidas de similitud según el criterio de distancia:

1. Varias entidades pueden estar igual de cerca de otra entidad. Cuando se da esta situación, una de las entidades igual de próximas se selecciona de forma aleatoria como la entidad más cercana.
2. Cuando una entidad contiene a otra o está contenida en ella, la distancia entre ambas es cero.
3. Esto significa que cuando una entidad está dentro de un polígono, la distancia entre la entidad y el polígono que la rodea es cero.
4. La distancia entre dos entidades es cero siempre que haya al menos una coordenada x,y compartida por las dos.
5. Esto significa que cuando dos entidades se interceptan, se superponen, se cruzan o se tocan, la distancia entre ellas es cero.
6. La distancia siempre se calcula hasta el límite de una entidad poligonal, no hasta el centro o el centroide del polígono.
7. Como ya se ha indicado, si una entidad está totalmente contenida en un polígono, la distancia entre la entidad y el polígono que la rodea es cero.
8. La distancia entre dos entidades (de cualquier tipo) es siempre la misma con independencia de cuál de ellas se elija como inicio o fin de la medición.

A partir de las coordenadas de cada localización en un espacio absoluto pueden ser calculadas diferentes medidas de distancia, denominadas métricas.

La distancia en línea recta o Distancia Euclidiana como se muestra en la ecuación 2.5, la cual surge ante la consideración de un espacio ideal a partir del cual no existen limitaciones para transitar en cualquier sentido se obtiene mediante la aplicación de la fórmula siguiente:

Distancia Euclidiana:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2.5}$$

Los resultados se obtienen a partir de la consideración de coordenadas absolutas sobre el espacio geográfico este es la resolución pitagórica del cálculo de la hipotenusa de un triángulo.

Con la finalidad de generar posibilidades de cálculo más flexibles que tiendan a superar la métrica de distancia Euclidiana la cual no funciona a cavalidad en polígonos multipartes, se plantea calcular el área de los mismos por su *centroide*, el cual sería el punto para calcular la distancia entre los polígonos.

En Matemáticas, los centroides de una figura bidimensional se refieren al punto en el cual todas las líneas de la figura correspondiente se interceptan unas con otras de tal manera que dividen la figura en dos partes iguales en los momentos equivalentes [36].

$$x = \frac{\int_a^b x(x_2 - x_1)dx}{\int_a^b x(y_2 - y_1)dx}, \; y = \frac{\int_c^d y(x_2 - x_1)dx}{\int_c^d (x_2 - x_1)dx} \tag{2.6}$$

2.2.1.1. Algoritmo para el criterio de distancia

Algoritmo cálculo de distancia

Algoritmo 1 $SimilitudDistancia$

Entrada: Identificadores de los objetos a comparar (id_1, id_2) y una capa vectorial (v_capa) que contiene los objetos geoespaciales a comparar

Salida: Distancia entre los objetos

1: $territorios = capadeterritorios.getFeatures() \rightarrow$ obtengo las características de todos lo territorios del mapa
2: **Para todo** $feature \in territorios$ **hacer**
3: **Si** $feature.id = id_1$ **entonces**
4: $terrx = feature \rightarrow$ le asigno el valor de feature cando encuentre el id pasado por parámetro
5: **Fin Si**
6: **Si** $feature.id = id_2$ **entonces**
7: $terry = feature \rightarrow$ le asigno el valor de feature cuando encuentre el id pasado por parámetro
8: **Fin Si**
9: **Fin Para**
10: **Si** $multipart(terrx)$ || $multipart(terry)$ **entonces**
11: $geomx = terrx.centroid() \rightarrow$ le asigno el valor del centroide con el método centroid de la clase qgsGeometry
12: $geomy = terry.centroid() \rightarrow$ le asigno el valor del centroide con el método centroid de la clase qgsGeometry
13: $dis = d.measureLine(geomx, geomy) \rightarrow$ calculo la distancia entre los dos puntos que se le pasan por parámetro
14: **Sino Si** $part(terrx)$ || $part(terry)$ **entonces**
15: $geomx = terrx.geometry() \rightarrow$ se obtienen las geometrías del conjunto de características
16: $geomy = terry.geometry() \rightarrow$se obtienen las geometrías del conjunto de características
17: $punx = nearestPoint(geomx, geomy) \rightarrow$ obtengo los puntos más cercanos entre las geometrías.
18: $dis = d.measureLine(punx, puny) \rightarrow$ calculo la distancia entre estos puntos
19: **Fin Si**
20: **Retornar** dis

2.2.2. Medida de similitud según el criterio de densidad

La densidad de una figura 3d está dada por su relación masa volumen. Pero de las figuras en el plano o 2d no tiene esta misma forma para determinar dicha relación pues estas carecen de estas dos medidas (masa, volumen) por tanto esta se determina mediante área y cantidad [37]. Por lo que se define que la densidad de una entidad geográfica en este caso las provincias, las cuales están representadas como polígonos: estará dada por la parte que represente la suma de todos los elementos de un mismo tipo con respecto a el área del polígono. Después se procedería a comparar el resultado de las densidades de los polígonos considerándose como el mejor, a la menor relación de división entre las densidades. 2.7

$$den(P_x, P_y) = \frac{D(P_x)}{D(P_y)} \leftrightarrow D(P_x) < D(P_y) \tag{2.7}$$

den()- densidad

Px- polígono x

D()- densidad

A(x)<A(y)- siempre que la densidad de x sea menor que la de y.

$D(P_x) = A(P_x)/Cant(Y)$

D()- densidad

Px- polígono x

A()- área

Cant()- cantidad

Y- tipo de elemento geográfico (puntos, polígonos, líneas)

2.2.2.1. Algoritmo para el criterio de densidad

Algoritmo 2 $SimilitudDensidad$

Entrada: Identificadores de los objetos a comparar (id_1, id_2) y una capa vectorial (v_capa) que contiene los objetos geoespaciales a comparar

Salida: Densidad entre los objetos

$D_1 = getcan(id1) \rightarrow$ obtengo la densidad del polígono con el id especificado

$D_2 = getcan(id2) \rightarrow$ obtengo la densidad del polígono con el id especificado

Si $D_1 > D_2$ **entonces**

    $result = D1/D2$

Fin Si

Si $D_2 >= D_1$ **entonces**

    $result = D1/D2$

Fin Si

Retornar $result$

Algoritmo getcan(id*)

Algoritmo 3 $Algoritmogetcan(id*)$

Entrada: Identificador del objeto a comparar (id_1) y una capa vectorial (v_capa) que contiene los objetos geoespaciales a comparar

Salida: Densidad entre los objetos en cuanto a líneas puntos y polígonos

1: $territorios = capadeterritorios.getFeatures() \rightarrow$
2: **Para todo** $feature \in territorios$ **hacer**
3: **Si** $feature.id = id_1$ **entonces**
4: $featuresAux = feature \rightarrow$ le asigno el valor de feature cando encuentre el id pasado por parámetro
5: **Fin Si**
6: **Fin Para**
7: **Si** $featuresAux! = vacio$ **entonces**
8: **Si** $featuresAux.ismultipart()$ **entonces**
9: $cart = featuresAux.geometry().asGeometryCollection()$ creo un arreglo con todas los tipos de geometrías que este tiene adentro
10: $long = cart.length \rightarrow$ obtengo la longitud del arreglo
11: **Fin Si**
12: $area = featuresAux.area() \rightarrow$obtengo el área
13: $result = area/long \rightarrow$ obtengo la densidad del polígono
14: **Fin Si**
15: **Retornar** $result$

2.2.3. Medida de similitud según el criterio de tamaño

En este momento se puede denotar entonces que otro de los criterios importantes para demostrar la similitud entre polígonos es su semejanza en cuanto al área o tamaño. Para ello se plantea que la similitud de un polígono está dada en la proximidad a 1 del resultado de la división de el más grande por el más pequeño, ya que una división indica la cantidad de veces que cabe el divisor dentro del dividendo. Para esto primero se determina una forma de calcular el área en polígonos irregulares, para después establecer la relación planteada entre sus áreas. Se muestra la ecuación del criterio de tamaño 2.8 a continuación:

$$t(P_x, P_y) = \frac{A(x)}{A(y)} <-> A(x) < A(y) \tag{2.8}$$

Donde P es el polígono seleccionado, x el índice del polígono, $A(x)$ el área del polígono x, $A(y)$ el área del polígono y, $A(x) < A(y)$ siempre que él polígono x sea menor que él polígono y.

Un procedimiento muy útil para hallar el área de cualquier polígono irregular es a través

del determinante de Gauss. Supone dibujar la figura sobre un plano cartesiano, fijando las coordenadas de cada uno de los vértices del polígono, como se muestra en la figura 2.2.

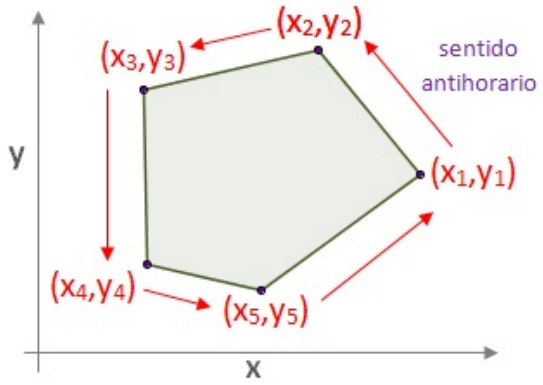

Figura 2.2: *Ejemplo de polígono.*

Se elige al azar cualquiera de ellos y se colocan los pares en la fórmula 2.9. Se ha de recorrer el polígono en el sentido contrario al de las agujas del reloj, teniendo en cuenta que el primer par de coordenadas corresponden al vértice elegido y, después de recorrer en sentido antihorario todos los vértices, el último par debe volver a ser el par inicial. Sean los vértices del polígono: $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$. La fórmula es la siguiente:

$$A = \frac{1}{2}\left(\begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} + \begin{vmatrix} x_2 & x_3 \\ y_2 & y_3 \end{vmatrix} + \ldots + \begin{vmatrix} x_N & x_1 \\ y_N & y_1 \end{vmatrix} \right) \tag{2.9}$$

2.2.3.1. Algoritmo para el criterio de tamaño

Algoritmo 4 $SimilitudTamao$

Entrada: Identificadores de los objetos a comparar (id_1, id_2) y una capa vectorial (v_capa) que contiene los objetos geoespaciales a comparar

Salida: relación de division entre los objetos

1: $territorios = capadeterritorios.getFeatures() \rightarrow$ obtengo las características de todos lo territorios del mapa
2: **Para todo** $feature \in territorios$ **hacer**
3: **Si** $feature.id = id_1$ **entonces**
4: $areax = feature.geometry().area() \rightarrow$ calculo el área que tiene el polígono que me pasan por parámetro
5: **Fin Si**
6: **Si** $feature.id = id_2$ **entonces**
7: $areay = feature.geometry().area() \rightarrow$ calculo el área que tiene el polígono que me pasan por parámetro
8: **Fin Si**
9: **Fin Para**
10: **Si** $areax < areay$ **entonces**
11: $result = areax/areay \rightarrow$ calculo el valor del área
12: **Sino Si** $areax >= areay$ **entonces**
13: $result = areay/areax \rightarrow$ calculo el valor del área
14: **Fin Si**
15: **Retornar** dis

2.2.4. Medida de similitud según el criterio de conectividad

Para resolver el problema de saber que tan similares son dos polígonos en cuanto a su conectividad habría que sumar el total de las longitudes de los lados comunes de los polígonos. Partiendo de que estos están conectados si tienen al menos un lado en común considerándose de esta manera polígonos vecinos, por ello una variable a tener en cuenta sería los vecinos del polígono en cuestión. Analizándose posteriormente la magnitud del tamaño de estos lados en común para cada uno de su vecinos, para después analizar la similaridad en esta se basaría en la comparación de estas distancias resultantes. Teniendo como función de comparación que mientras mayor sea la suma de los lados comunes del polígonos más similares se puede decir que estos son.

El borde del polígono está dado por una geometría que conforma al mismo,la cual no tiene más que coordenadas de tipo (x, y) estas, que son proporcionadas por el QGIS en su capa de datos vectoriales, pueden ser almacenadas para posteriormente realizar con ellas varias operaciones básicas ya definidas por el sistema como la intersección la cual devuelve un booleano lo que

permite obtener la vecindad del polígono en cuestión. Posibilitando el cálculo del perímetro de este lado en común con cada uno de sus vecinos. A continuación la fórmula 2.10 muestra con se conectividad entre los punto i y j.

$$V(P_i, P_j) = \sum long(P_{ia}, P_{ja}) \tag{2.10}$$

V(Pi,Pj)- valor de la sumatoria "polígono i / polígono j"

P- polígono

i- polígono en cuestión

a- lado común

long – longitud

2.2.4.1. Algoritmo para el criterio de conectividad

Algoritmo 5 $SimilitudConectividad$

Entrada: Identificadores de los objetos a comparar (id_1, id_2) y una capa vectorial (v_capa) que contiene los objetos geoespaciales a comparar

Salida: Tamaño de las fronteras comunes entre dos polígonos

1: $territorios = capadeterritorios.getFeatures() \rightarrow$ obtengo las características de todos lo territorios del mapa
2: **Para todo** $feature \in territorios$ **hacer**
3: **Si** $feature.id = id_1$ **entonces**
4: $terrx = feature \rightarrow$ le asigno el valor de feature cuando encuentre el id pasado por parámetro
5: **Fin Si**
6: **Si** $feature.id = id_2$ **entonces**
7: $terry = feature \rightarrow$ le asigno el valor de feature cuando encuentre el id pasado por parámetro
8: **Fin Si**
9: **Fin Para**
10: **Si** $intersects(terrx.geometry(), terry.geometry())$ **entonces**
11: $geom = interception(terry, terrx) \rightarrow$ obtengo la intersepción entre los dos territorios
12: $perimetrox = d.measurePerimeter(terrx.geometry()) \rightarrow$ calculo el perímetro de terrx
13: $perimetroy = d.measurePerimeter(terry.geometry()) \rightarrow$ calculo el perímetro de terry
14: $x = d.measurePerimeter(geom) \rightarrow$ calculo el perímetro de la parte en común
15: **Si** $perimetrox < perimetroy$ **entonces**
16: $distancia = ((perimetrox - x)/(perimetroy - x)) \rightarrow$ obtengo el tamaño de su lado en común
17: **Sino Si** $perimetrox > perimetroy$ **entonces**
18: $distancia = ((perimetroy - x)/(perimetrox - x)) \rightarrow$obtengo el tamaño de su lado en común
19: **Sino Si** $perimetrox == perimetroy$ **entonces**
20: $distancia = 1 \rightarrow$le asigno 1 si son iguales
21: **Fin Si**
22: **Fin Si**
23: **Retornar** $distancia$

2.3. Requisitos de software

"Un requisito es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste" [38]. La calidad con que se realiza la captura de los requisitos incide en todo el proceso de desarrollo del software repercutiendo en el resto de las fases de su desarrollo. Además contribuye a tomar mejores decisiones de diseño y arquitectura.

Existen muchas técnicas para la obtención de requerimientos como: entrevistas, cuestionarios, lluvia de ideas, observación, estudio de documentación, entre otras. En esta investigación se decide utilizar Desarrollo Conjunto de Aplicaciones o Joint Application Development (JAD) en inglés, es una técnica que consiste en realizar sesiones conjuntas entre los analistas de sistemas y los expertos del dominio [39]. Esta tiene como principales ventajas las siguientes pautas:

- Con esta técnica se obtienen sistemas más enfocados a la realidad, muchas metodologías nuevas se fundamentan en esta premisa.
- Porque las entrevistas son lentas, difíciles de hacer y complicadas de obtener datos.
- Al ser muchos revisores del proyecto es más fácil detectar errores.
- El JAD propugna una participación más profunda de los usuarios en el proyecto.

En la reunión JAD llevada a cabo se tratarán los siguientes puntos [39]:

1. Conducir la orientación. Introducción.
2. Refinar y limitar los requisitos de alto nivel identificados en la fase de plan JAD.
3. Desarrollar un flujo de trabajo o workflow en inglés.
4. Desarrollar la descripción de dicho workflow.
5. Diseñar la interfaz de usuario.
6. Especificar requisitos de procesamiento.
7. Definir interfaces.
8. Identificar grupos de datos
9. Documentar las decisiones consensuadas.
10. Conclusión.

2.3.1. Requisitos funcionales

Los requisitos funcionales (**RF**) indican características y restricciones sobre la funcionalidad del software o sus componentes, además son la condición necesaria de un atributo para que cumpla una función determinada. Los requerimientos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone que un sistema debe cumplir [40]. A continuación se muestran los RF identificados:

- **RF 1:** Calcular medida de similitud geométrica
  - **RF 1.1:** Calcular medida por criterio de distancia
  - **RF 1.2:** Calcular medida por criterio de densidad
  - **RF 1.3:** Calcular medida por criterio de tamaño
  - **RF 1.4:** Calcular medida por criterio de conectividad
- **RF 2:** Integrar medidas de similitud

2.3.2. Requisitos no funcionales

Los requisitos no funcionales (**RNF**) son propiedades o cualidades que el sistema debe tener. Estas propiedades o cualidades se refieren a las características que hacen al sistema estable, usable, rápido, confiable y escalable [41]. A continuación se muestran los RNF identificados:
Requisitos de Software

- **RNF 1:** Se debe tener instalada la herramienta QGIS en su versión 2.6 o superior.
- **RNF 2:** Se debe tener instalado el módulo Postgis en su versión 2.1.5 o superior.

Requisitos de Hardware

- **RNF 4:** La estación de trabajo debe contar con al menos 1,0 GB de *Memoria de Acceso Aleatorio* (**RAM**, por sus siglas en inglés).
- **RNF 5:** La capacidad mínima de espacio en disco debe ser 2.0 GB.

Requisitos de Usabilidad

- **RNF 6:** Debe tener una interfaz gráfica, visualmente atractiva para el usuario. La aplicación podrá ser usada por cualquier usuario con conocimientos básicos sobre geografía, informática y medicina. Debe mostrar mensajes al usuario que le ayuden a llevar a cabo la tarea que realiza.

Requisitos de Interfaz

- **RNF 7:** Debe tener una interfaz amigable y con apariencia profesional.
- **RNF 8:** La interfaz debe tener un diseño sencillo y ser de fácil comprensión para el usuario.

Restricciones de diseño e implementación

- **RNF 9:** Se hace uso de la herramienta Qgis en su versión 2.6 e IDE Pycharm versión 3.4.
- **RNF 10:** El lenguaje de programación usado para la implementación es Python.

2.4. Fase de planificación

La metodología XP define como fase inicial la planificación. Durante esta etapa se lleva a cabo el proceso de identificación y confección de las historias de usuario, así como la familiarización del equipo de trabajo con las tecnologías y herramientas seleccionadas para el desarrollo del software. El cliente especifica la prioridad en que se deben implementar las historias de usuario, estimándose además el esfuerzo para la realización de cada una de estas por el equipo de desarrollo. El resultado de la fase es un plan de entregas donde se realiza una estimación de las versiones que tendrá el producto en su realización, de manera tal que guíe su desarrollo [34].

2.4.1. Historias de usuarios

Las Historias de usuarios (HU) sustituyen a los documentos de especificación funcional, Estas "historias" son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. El tratamiento de las HU es muy dinámico y flexible, en cualquier momento pueden reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas [42].

Tabla 2.1: *Historia de usuario: implementar criterio densidad*

Historia de Usuario "Implementar criterio densidad "	
Número: 1.2	Nombre Historia de Usuario: Implementar criterio densidad
Usuario: Experto	
Prioridad en Negocio: Alto	Riesgo en Desarrollo: Alto
Puntos Estimados: 21	Iteración Asignada: 1
Programador responsable: Miguel Antonio Martínez Ochandarena	
Descripción: La aplicación debe ser capaz de calcular la similitud en cuanto a densidad entre polígonos a partir de: 1-) Los datos geográficos de cada polígono.	
Observaciones:	

2.4.2. Estimación de esfuerzos por historias de usuario

No se debe pasar por alto que las HU 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. Estas estimaciones deben ser lo más fieles posible a la realidad ya que estas dan una medida de la velocidad del proyecto y ofrecen una guía a la cual ajustarse [42]. Los resultados se muestran el la tabla 2.2.

Tabla 2.2: *Estimación de esfuerzos por Historia de Usuario.*

Historia de usuario	Puntos de Estimación (semanas)
HU 1.1: Implementar criterio Distancia.	3
HU 1.2: implementar criterio Densidad.	2
HU 1.3: implementar criterio Tamaño.	1
HU 1.4: implementar criterio Conectividad	3
HU 2 : integrar las nuevas medidas de similitud	3

2.4.3. Plan de iteraciones

Una vez finalizadas las HU se debe crear un plan de iteraciones, indicando cuáles se desarrollarán en cada iteración del programa.

Tabla 2.3: *Plan de duración de las iteraciones.*

Iteraciones	Orden de las historias de usuario a Implementar	Duración de las iteraciones (sem)
Iteración 1	implementar criterio densidad	3
Iteración 2	implementar criterio distancia	3
Iteración 3	implementar criterio tamaño	1
Iteración 4	implementar criterio conectividad	3
Iteración 5	integrar las nuevas medidas de similitud	3
	Total	**12**

2.4.4. Plan de entrega

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 y está conformado por el tiempo estimado de desarrollo según los desarrolladores [42]. Tiene como objetivo definir el número de liberaciones que se realizarán en el transcurso del proyecto y las iteraciones que se requieren para desarrollar cada una. Trazándose así el plan de entrega en función de estos dos parámetros: tiempo de desarrollo ideal y grado de importancia para el cliente.

Tabla 2.4: *Plan de duración de las entregas.*

	Final de la 1ra Iteración	Final de la 2da Iteración	Final de la 3ra Iteración	Final de la 4ta Iteración	Final de la 5ta Iteración
Módulos	1ra semana de marzo	4ta semana de marzo	1ra semana de abril	4ta semana de abril	3ra semana de mayo
Medidas de similitud geométrica	v1.0	v1.1	v1.2	v1.3	v1.4

2.5. Fase de diseño

La metodología de desarrollo XP plantea prácticas especializadas que accionan directamente en la realización del diseño para lograr un sistema robusto y reutilizable. Se trata en todo momento de conservar su simplicidad, es decir, crear un diseño evolutivo que va mejorando incrementalmente y que permite hacer entregas pequeñas y frecuentes de valor para el cliente, basado principalmente en el desarrollo de las tarjetas Clase-Responsabilidad-Colaboración (CRC).

2.5.1. Tarjetas Clase-Responsabilidad-Colaboración

Las tarjetas CRC son utilizadas para representar las responsabilidades de las clases y sus interacciones. Estas tarjetas permiten trabajar con una metodología basada en objetos, permitiendo que el equipo de desarrollo completo contribuya en la tarea del diseño. En cada tarjeta CRC el nombre de la clase se coloca a modo de título, las responsabilidades se colocan a la izquierda y las clases que se implican en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente.

Una clase es cualquier persona, evento, concepto, pantalla o reporte. Las responsabilidades de una clase son las cosas que conoce y las que realizan, sus atributos y métodos. Los colaboradores de una clase son las demás clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades [43].

En las tablas 2.5 y 2.6 se muestran las tarjetas CRC correspondientes a las clases Territorio y similitud Perímetro.

Tabla 2.5: *Tarjeta CRC para la clase Territorio.*

Clase: Territorio	
Responsabilidad	Colaboración
Calcular el aporte de riesgo de cada territorio. Crear instancias de la clase Indicador.	Estrato, Indicador, Algoritmo

Tabla 2.6: *Tarjeta CRC para la clase similitud distancia.*

Clase: similitud_Distancia	
Responsabilidad	Colaboración
Calcular la similitud en cuanto al criterio de distancia	ControladorEstratificador

2.6. Arquitectura de software

La arquitectura de software es la definición y estructuración de una solución que cumple con los requisitos técnicos y operativos. Optimiza atributos que implican una serie de decisiones, tales como la seguridad, el rendimiento y la capacidad de administración. Estas decisiones en última instancia, afectan la calidad de la aplicación, el mantenimiento, el rendimiento y el éxito global [44].

2.6.1. Estilo arquitectónico a utilizar

Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software. Estos permiten expresar un esquema de organización estructural esencial para un sistema de software [44].

Modelo Vista Controlador:

Modelo Vista Controlador(*Model View Controller*, **MVC** por sus siglas en inglés) es un patrón de diseño que considera dividir una aplicación en tres módulos claramente identificables y con funcionalidades bien definidas: el Modelo, las Vistas y el Controlador. Este se ve frecuentemente en aplicaciones web, donde la vista es la interfaz de usuario y el código es el que provee de datos dinámicos a la página; el modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio; y el controlador es el responsable de recibir los eventos de entrada desde la vista. Este tiene como ventajas que facilita la evolución por separado de ambos aspectos e incrementa la reutilización y la flexibilidad [45].

El uso de esta arquitertura está condicionado por la necesidad de integrarse con un *plugin* que sigue este mismo estilo arquitectónico.

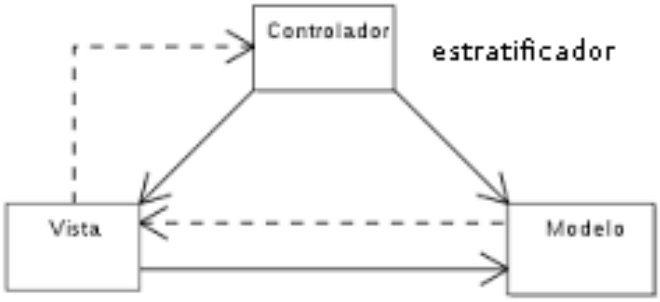

Figura 2.3: *Evidencia de la arquitectura del sistema.*

Modelo: Encapsula los datos y las funcionalidades. El modelo es independiente de cualquier representación de salida y/o comportamiento del controlador o la vista.
Vista: Muestra la información a través de una interfaz de usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador con el que interactúa.
Controlador: Reciben las entradas de las vistas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, etc. Los eventos son traducidos a solicitudes de servicio(*service requests*, en inglés) para el modelo o la vista.

2.7. Patrones de diseño

Un patrón de diseño describe una estructura de diseño que resuelve un problema particular del diseño dentro de un contexto específico que afectan la manera en la que se aplica y en la que se utiliza dicho patrón. El objetivo de cada patrón de diseño es proporcionar una descripción que permita a un diseñador determinar **1)** si el patrón es aplicable al trabajo en cuestión, **2)** si puede volverse a usar (con lo que se ahorra tiempo de diseño) y **3)** si sirve como guía para desarrollar un patrón distinto en funciones o estructura.
Los patrones de diseño constituyen la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber

comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

2.7.1. Patrones de diseño GRASP

Los Patrones Generales de Software para la Asignación de Responsabilidades (**GRASP**, por sus siglas en inglés) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. El nombre se eligió para indicar la importancia de captar estos principios, si desea diseñar eficazmente el software orientado a objetos [40].

Experto: Define cómo asignar de forma adecuada las responsabilidades en un modelo de clases. Este expone que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Dicho patrón se evidencia en la solución planteada en la clase silmilitudPerimetro, como esta contiene todos los elementos para calcular el perímetro dicha responsabilidad recae sobre ella. En la figura 2.4 se muestra una imagen de dicha clase.

Figura 2.4: *Evidencia del patrón Experto.*

Creador: Es el responsable de la creación o instanciación de nuevos objetos o clases. Este patrón nos dice que la nueva instancia podrá ser creada por una clase si:

1. Tiene la información necesaria para realizar la creación del objeto.
2. Usa directamente las instancias creadas del objeto.
3. Almacena o maneja varias instancias de la clase.
4. Contiene o agrega la clase.

En la aplicación informática se pone de manifiesto dicho patrón en la clase Estrato, a esta se le asigna la responsabilidad de crear instancias de la clase Territorio. En la figura 2.5 se presenta la evidencia de dicho patrón.

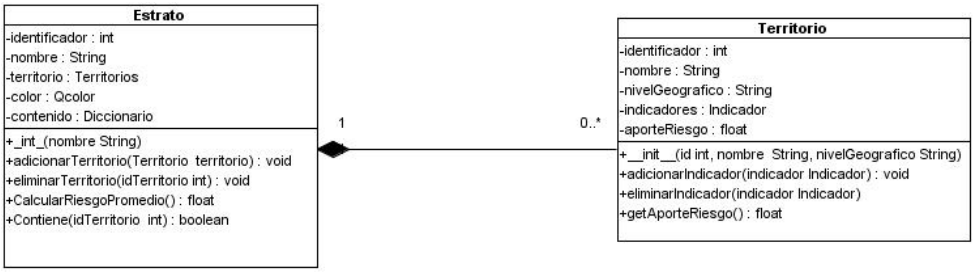

Figura 2.5: *Evidencia del patrón Creador.*

Controlador: permite asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas, facilitando la centralización de actividades. El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Un error muy común es asignarle demasiada responsabilidad y alto nivel de acoplamiento con el resto de los componentes del sistema. Este patrón se evidencia en la aplicación informática en la clase ControladoraEstratificacion, a esta se le asignó la responsabilidad de manejar los eventos del sistema generados por el usuario. En la figura 2.6 se muestra una imagen de dicha clase.

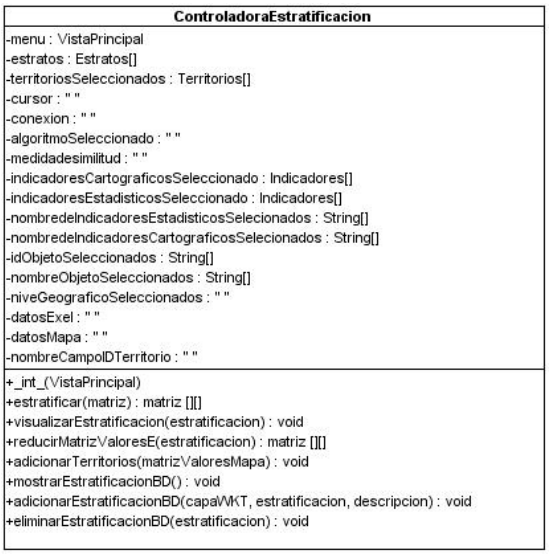

Figura 2.6: *Evidencia del patrón Controlador.*

Bajo acoplamiento: Se basa en tener la menor dependencias entre las clases. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, la conoce y recurre a ellas. En la implementación del Plugin es aplicado este patrón en la mayoría de las clases, propiciando que los componentes sean fáciles de entender por separado y de reutilizar.

Alta cohesión: La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo. No conviene este tipo de clases pues son difíciles de comprender, reutilizar y conservar.

2.7.2. Patrones de diseño GoF

Los Patrones del Grupo de Cuatro (**GoF**, por sus siglas en inglés) resuelven problemas específicos de diseño de software [1]. Estos patrones se agrupan en las siguientes categorías: creacionales, estructurales y de comportamiento.

- **Comportamiento:** Contribuyen a definir la comunicación e interacción entre los objetos de un sistema [46].

    - **Método plantilla:** es un patrón de comportamiento que define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar estructura. Este patrón se evidencia en las clases AlgoritmoKMedias y AlgoritmoMatrizSimilitud, estas heredan todas las funcionalidades de la clase.

- **Creacionales:** Permiten abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados [46].

    - **El patrón Singleton:** forma parte de los patrones creacionales. Se trata de uno de los patrones más usados y conocidos por los desarrolladores, y también es uno de los patrones más controvertidos. El patrón Singleton se encarga de controlar que únicamente se crea una instancia de una clase en toda la aplicación mediante el uso de un único punto de acceso.Este patrón se evidencia en la clase controladora en donde se crea una intancia única a la que todas las clases hacen referencia para comunicarse con esta.

- **Estructurales:** Describen cómo las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades [46].

    - **El patrón Facade:** simplifica el acceso a un conjunto de clases proporcionando una única clase que todos utilizan para comunicarse con dicho conjunto de clases acceso. Este patrón se evidencia en la clase Vistaprincipal desde donde se aceden a todas las clases.

2.8. Diagrama Entidad-Relación

Los Diagramas Entidad-Relación **DER** definen todos los datos que se introducen, se almacenan, se transforman y se producen dentro de una aplicación [44]. Estos modelos representan a la realidad a través de un esquema gráfico empleando la terminología de Entidades, que son objetos

que existen y son los elementos principales que se identifican en el problema a resolver.

En la figura 2.7 se muestra el diagrama Entidad-Relación utilizado para la consulta y almasenamiento de los datos.

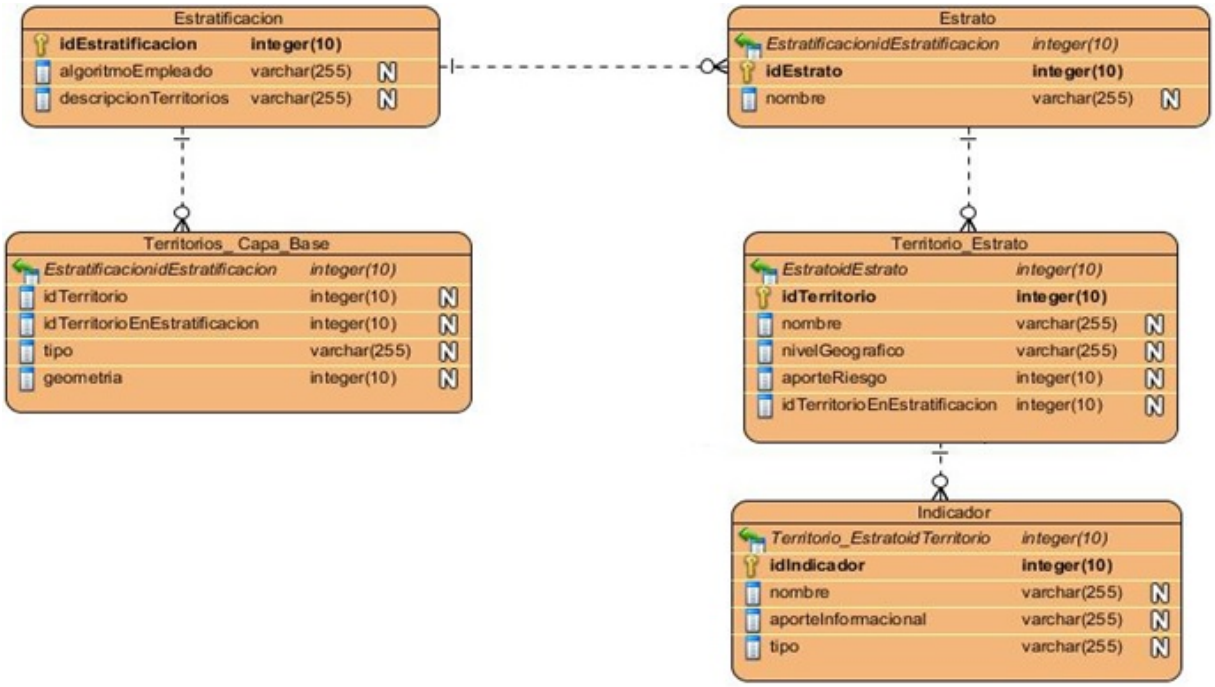

Figura 2.7: *Diagrama Entidad-Relación.*

2.9. Conclusiones del capítulo

La propuesta de solución definida posibilitó la evolución del proceso de estratificación de territorios en Sistema de información geográfica QGIS al incluirle las medidas de similitud geométricas. Mediante la descripción de las HU separadas por iteraciones y la planificación del esfuerzo dirigido al desarrollo de cada una de ellas, se logró una mejor organización del trabajo y el establecimiento de fechas de culminación para cada iteración. También se evidencia la gran utilidad del estilo arquitectónico MVC y de los patrones de diseño GRASP y GoF los que permitieron una mejor estructuración de la aplicación.

3. Verificación de la solución propuesta

En el siguiente capítulo se detallan las tareas de ingenierías que conforman cada HU definida en la fase de planificación. Se define el estándar de codificación que se estará utilizando en el desarrollo de la solución. Además se realizan las pruebas definidas por la metodología seleccionada, unitarias y de aceptación. Se realiza un caso de estudio para verificar la validez de los resultados de la solución propuesta.

3.1. Fase de implementación

Luego de haber definido los elementos necesarios en la etapa de planificación y diseño se pasa a la de codificación o implementación de la aplicación, donde se da cumplimento al plan de iteraciones. En esta fase se realiza la implementación de las HU que fueron seleccionadas por cada iteración, además se crean las tareas de ingeniería para ayudar a organizar la implementación exitosa de las HU. Al finalizar esta fase el cliente estará listo para comenzar a realizar las pruebas.

3.1.1. Tarea de ingeniería

Cada HU está compuesta por una o varias tareas de ingeniería, estas se realizan para especificar las acciones llevadas a cabo por los programadores. En la tabla 3.1 se detallan para la iteración número uno, las tareas a desarrollar por cada HU y en la tabla 3.2 se describe una tarea de ingeniería que responde a una HU arquitectónicamente significativa, el resto se encuentran especificadas en anexos.

Tabla 3.1: *Distribución de tareas de ingeniería por HU (iteración 1).*

HU	Tareas de ingeniería por HU
implementar criterio densidad	Obtener características de los polígonos. calcular la densidad del polígono

Tabla 3.2: *Tarea de ingeniería para obtener características de los polígonos.*

Tarea de ingeniería	
Número Tarea:1	Número Historia de Usuario: HU # 1.2
Nombre Tarea: Obtener características de los polígonos.	
Tipo Tarea: Desarrollo	Puntos Estimados: 1
Fecha Inicio: 17/02/2017	Fecha Fin: 23/02/2017
Programador Responsable:Miguel Antonio Martínez Ochandarena	
Descripción: Esta tarea permite obtener las caracteristicas geométricas dígase (líneas , puntos, subpolígonos) que contiene el polígono a evaluar .	

3.1.2. Estándares de codificación

La metodología XP resalta que la comunicación de los programadores es a través del código, por lo que es necesario que sigan ciertos estándares de programación para lograr un entendimiento entre los programadores, de manera que cualquier persona del equipo de desarrollo pueda modificar el código. Además, se hace preciso que el código sea entendible para que posteriormente otros programadores puedan apoyarse en ese trabajo y desarrollen otras soluciones.

En el caso de el plugin que se desarrolla, el estándar que se utiliza es:

Máxima longitud de las líneas

- Todas las líneas se limitan a un máximo de 79 caracteres.
    - territorios = self.menu.capaTerritorios.getFeatures()

Importaciones

- Las importaciones se encuentran en líneas separadas.

- from qgis.core import *
- from qgis.gui import *
- from PyQt4.QtCore import *

Comentarios

- Se utilizan comentarios de una línea para hacer más entendible el código.

  Comentarios de una línea: comentario pequeño que solo abarca una línea y describe el código que le sigue.

  - # Esto es un comentario de una línea

Estilo de los nombres

- **Clases e Interfaces:**
  - Se utilizará el estilo de codificación *Upper Camel Case* para nombrar las clases este establece que la primera letra sea en mayúscula, en caso de ser un nombre compuesto, la inicial de cada palabra se representa en mayúscula.
  - Se utilizan nombres simples y de alguna manera que describan el contenido, se usan palabras completas, a no ser que la abreviatura sea muy conocida.
- **Métodos y variables:**
  - Se utilizará el estilo de codificación *Lower Camel Case* el cual define que los nombres de cada métodos se representan en minúscula, en caso de ser un nombre compuesto, la inicial de la primera palabra se simboliza en minúscula, y la de las otras palabras que lo componen en mayúscula.
  - Los nombres de las variables son cortos pero con significados lógicos, capaces de permitir a un observador identificar su función.

3.2. Fase de pruebas

Cuando se desarrolla una solución informática se deben realizar una gran cantidad de pruebas para verificar que el código esté correcto. Estas pruebas normalmente tienen que ser ejecutadas

en varias ocasiones y se ven afectadas por los cambios que se introducen conforme se va construyendo la solución. XP divide las pruebas en dos grupos: pruebas de aceptación, o pruebas funcionales diseñadas por el cliente final, destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida y pruebas unitarias, encargadas de verificar el código y diseñadas por los programadores.

3.2.1. Pruebas de aceptación

Las pruebas de aceptación en XP son especificadas por el cliente, y se centran en las características y funcionalidades generales del sistema, que son visibles y revisables por parte del usuario. Estas pruebas se derivan de las HU que se han implementado como parte de la liberación del software [33].

Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Así mismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una HU 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 [33].

Casos de prueba

En la tabla 3.3 se muestran los casos de prueba de aceptación aplicados a las HU implementar criterio densidad.

Tabla 3.3: *Caso de prueba de aceptación para HU implementar densidad.*

Caso de prueba de aceptación	
Código: HU1_P2	Historia de Usuario: 1.2
Nombre: Implementar criterio densidad	
Descripción: Prueba para validar la funcionalidad implementar criterio densidad	
Condiciones de ejecución: El usuario debe escoger dentro de las opciones existentes la densidad El usuario debe seleccionar la opción Aceptar	
Resultados esperados: En caso que se cumplan las condiciones de ejecución, el sistema estratifica según la medida de similitud seleccionada. En caso contrario el sistema ejecuta la estratificación por el método predefinido	
Evaluación de la prueba: Prueba satisfactoria	

Análisis de los resultados

Para comprobar que el resultado obtenido por el sistema coincide con el resultado esperado por el cliente se crearon un total de 7 casos de prueba de aceptación en conjunto cliente-desarrolladores. De este total, 4 arrojaron el resultado esperado mientras que 3 pruebas resultaron fallidas, las funcionalidades que respondían a estas pruebas fueron tratadas en la siguiente iteración y al volver a aplicar las pruebas de funcionalidad mostraron un resultado de 6 exitoso y 1 fallida. Se le dio tratamiento a la funcionalidad defectuosa y finalmente al realizar una nueva iteración se obtuvieron un total de 7 pruebas satisfactorias con respecto a los 7 casos de prueba aplicados como se muestra en la figura 3.1.

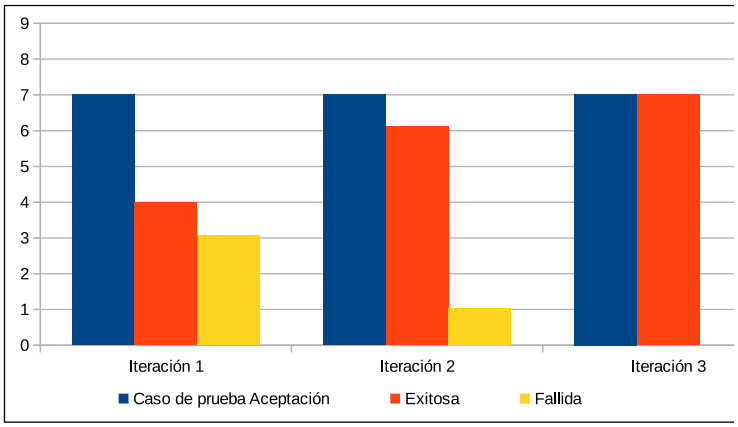

Figura 3.1: *Resultado de aplicar la prueba de aceptación.*

3.2.2. Pruebas de caja blanca

Las pruebas de caja blanca se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al *código fuente*. Se escogen distintos valores de entrada para examinar cada uno de los posibles *flujos de ejecución* del programa cerciorándose que se devuelvan los valores de salida adecuados [44]. Las pruebas de caja blanca intentan garantizar que:

- Se ejecutan al menos una vez todos los caminos independientes de cada módulo.
- Se utilizan las decisiones en su parte verdadera y en su parte falsa.
- Se ejecuten todos los bucles en sus límites.
- Se utilizan todas las estructuras de datos internas.

La técnica utilizada dentro de las pruebas de caja blanca fue camino básico. En la figura 3.2 se enumeran las sentencias de código del método principal para el criterio tamaño.

```
def run(self,territorioX, territorioY):

    territorios = self.menu.capaTerritorios.getFeatures()
    x = False
    y = False
    areaX=""
    areaY=""
    for feature in territorios and  x and y  :
        if territorioX == feature.id():
            # print feature.geometry()
            terrx = feature.geometry()
            areaX =terrx.area()
            x= True
        elif territorioY == feature.id():
            # print feature.geometry()
            terry = feature.geometry()
            areaY = terry.area()
            y = True

    if areaX!="" and areaY!="" :
        if areaX< areaY:

            simitama=areaX/areaY


        else:
            simitama=areaY/areaX
    else:
        simitama=0
    #print simitama
    return simitama
```

Figura 3.2: *Código del método run().*

Luego de haberse construido el grafo se realiza el cálculo de la complejidad ciclomática (es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa) mediante las tres fórmulas descritas a continuación, las cuales deben arrojar el mismo resultado para asegurar que el cálculo de la complejidad sea correcto.

1. La complejidad ciclomática coincide con el número de regiones del grafo de flujo.
2. La complejidad ciclomática, V(G), de un grafo de flujo G, se define como V(G) = Aristas - Nodos + 2.
3. La complejidad ciclomática, V(G), de un grafo de flujo G, también se define como V(G) = Nodos de predicado (es él que representa una condicional if o case, es decir, de él salen varios caminos.) 2 + 1.

Guiándose por el grafo de flujo del método run() mostrado en la figura 3.3, la complejidad ciclomática sería:

- Como el grafo tiene 6 regiones, $V(G) = 6$
- Como el grafo tiene 17 aristas y 13 nodos, $V(G) = 17 - 13 + 2 = 6$
- Como el grafo tiene 5 nodos de predicado, $V(G) = 5 + 1 = 6$

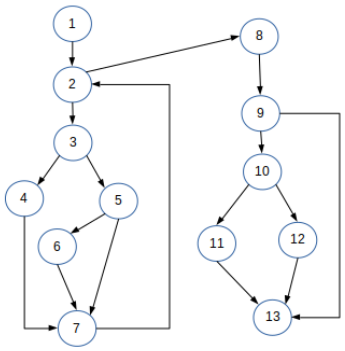

Figura 3.3: *Grafo de flujo del método*

Como el calculo de las fórmulas anteriores arrojó el mismo resultado se puede afirmar que la complejidad ciclomática del método es tanto. Esto da entender que existen 4 posibles caminos por donde el flujo circula. Este valor representa el número mínimo de casos de pruebas para el procedimiento tratado.

Caminos básicos identificados:

- Camino 1: 1-2-3-5-7-2-8-9-13
- Camino 2: 1-2-3-4-7-2-8-9-13
- Camino 3: 1-2-3-5-7-2-3-4-6-7-8-9-10-11-13
- Camino 4: 1-2-3-4-6-7-2-8-9-13
- Camino 5: 1-2-3-4-6-7-2-5-7-2-8-9-13
- Camino 6: 1-2-3-4-7-2-3-4-6-7-2-8-9-13

A continuación se realizan los casos de prueba para una muestra de los caminos básicos como se muestra en las tablas 3.4 y 3.5, que representan los caminos básicos 1 y 2 respectivamente.

Tabla 3.4: *Caso de prueba para el camino básico # 1.*

Caso de prueba para el camino básico #1 (1-2-3-5-7-2-8-9-13)	
Descripción	Prueba para comprobar los resultados de la función run() de la clase similitudTamaño en caso que la lista de territorios a evaluar sea vacía.
Condición de ejecución	longitud de territorios = 0
Entrada	territorios=[ ]
Resultado	semitama= 0
Resultado de la prueba	Prueba satisfactoria

Tabla 3.5: *Caso de prueba para el camino básico # 2.*

Caso de prueba para el camino básico #2 (1-2-3-4-5-2-8-9-13)	
Descripción	Prueba para validar los resultados del método run() de la clase simitudTamanno() cuando solo un id coincide.
Condición de ejecución	longitud de territorios > 0 id1=" " id2!= " "
Entrada	territorios=[ ]
Resultado	semitama=0
Resultado de la prueba	Prueba satisfactoria

Análisis de los resultados

Para la validación del código creado en el desarrollo de la herramienta se decidió seleccionar los métodos más importantes, a los que se le efectuaron las pruebas para poder evaluar si el funcionamiento de cada uno de ellos se comporta de la manera esperada. Se realizaron un total de 9 pruebas a las 4 funcionalidades seleccionadas como relevantes, de las cuales 7 resultaron satisfactorias en una primera iteración de pruebas y 2 no satisfactorias. Estas últimas fueron solucionadas en una segunda iteración para obtener en su totalidad todas las pruebas satisfactorias, comprobándose la estabilidad de la lógica aplicada en el código generado en el desarrollo de la funcionalidad.

Para tener un mejor entendimiento de lo anterior se puede observar la figura 3.4 que se muestra a continuación:

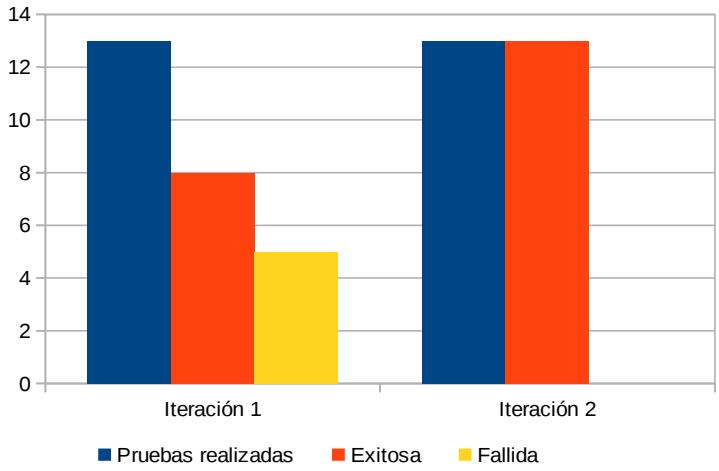

Figura 3.4: *Resultado de aplicar la prueba de caja blanca.*

3.2.3. Caso de estudio

Con el fin de valorar los resultados de la solución propuesta se decide aplicar a un caso de estudio, en correspondencia con el trabajo realizado por los ingenieros Yanislay Torres Vega y Rolando Morales Pérez, en su trabajo de diploma "Propuesta para la estratificación de territorios basada en indicadores de salud." donde se realiza un proceso de estratificación de las catorce provincias de Cuba definidas en la división política-administrativa de 1976. Se utilizó como fuente de información el Anuario Estadístico del año 2001 [47] y se seleccionaron los indicadores siguientes:

- Mortalidad infantil por cada 1000 nacidos vivos.
- Mortalidad infantil de los niños menores de cinco años por cada 1000 nacidos vivos.
- Mortalidad por enfermedades del corazón por cada 100 000 habitantes.
- Mortalidad por tumores malignos por cada 100 000 habitantes.
- Mortalidad por enfermedades cerebro vasculares por cada 100 000 habitantes.
- Mortalidad por influenza y neumonía por cada 100 000 habitantes.
- Mortalidad por accidentes por cada 100 000 habitantes.
- Mortalidad perinatal por cada 1000 nacidos vivos.
- Mortalidad por enfermedades infecciosas y parasitarias por cada 100 000 habitantes.

- Mortalidad materna por cada 100 000 nacidos vivos.
- Incidencia de tuberculosis por cada 100 000 habitantes
- Incidencia de hepatitis por cada 100 000 habitantes.
- Incidencia de diabetes por cada 100 000 habitantes.
- Incidencia de hipertensión por cada 100 000.
- Incidencia de asma por cada 100 000 habitantes.
- Incidencia de bajo peso al nacer.
- Consultas médicas por habitante.
- Ingresos por cada 100 habitantes.
- Camas de asistencia por cada 1000 habitantes.
- Consultas de puericultura por habitante.
- Consultas de pediatría por habitante.
- Densidad poblacional.
- Población mayor de 60 años.
- Población menor de 1 año.
- Población menor de 15 años.
- Natalidad por cada 1000 habitantes.

Aplicación al caso de estudio

Para realizar la clasificación de cada una de las provincias de Cuba utilizando la herramienta sobre la cual se trabajo, se empleó el algoritmo de agrupamiento k-medias y el número de estratos se fijó en 4 utilizando primeramente una de las medidas de similitud definidas en la propuesta de solución (distancia).
Los resultados se muestran en la figuras 3.5 y en la tabla 3.6.

Resultados de la aplicación del caso de estudio

A continuación se muestran los resultados de la estratificación de las provincias de Cuba a partir del proceso analítico-estadístico de las variables de salud escogidas. En la figura se presentan en forma de mapa y en la tabla de manera más detallada.

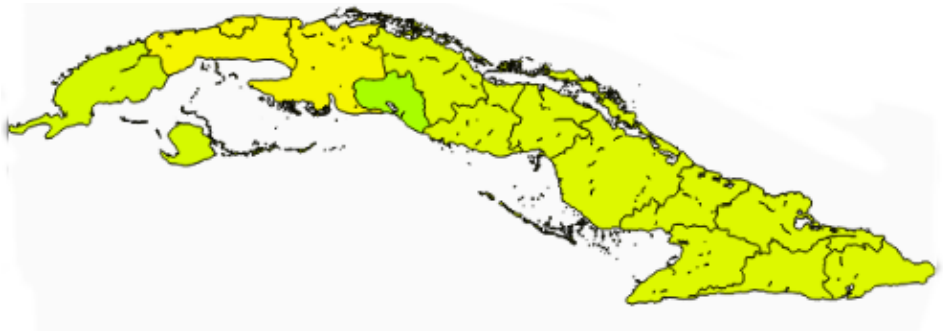

Figura 3.5: *Mapa temático de la estratificación realizada utilizando una de las medidas de similitud propuesta (Distancia).*

Tabla 3.6: *Resultados de la estratificación realizada utilizando una de las medidas de similitud propuesta (Distancia).*

Nombre del estrato	Provincias	Riesgo de salud
Estrato 1	Pinar del Rio, Isla de la Juventud	0.215822150119
Estrato 2	Cienfuegos	0.195472955699
Estrato 3	Villa Clara, Granma, Santiago de Cuba, Guantánamo, Las Tunas, Holguín, Camagüey, Sancti Spíritus, Ciego de Ávila	0.20093039793
Estrato 4	La Habana, Ciudad de la Habana, Matanzas	0.440393010707

Comparación de resultados

Para determinar la relación existente entre los resultados obtenidos por la herramienta con las medidas de similitud desarrolladas y las del proceso realizado en el trabajo de diploma [3], a partir del caso de estudio descrito se analiza la continuidad espacial de los polígonos pertenecientes a un mismo estrato, basandose en la primera ley de la geografía y en el análisis experimental sobre la aplicación de agrupación relizada en el artículo [48], en donde se pudo observar mediante la aplicacion de la fórmula de **Gap Statistic** [49] que con la adición de

los atributos espaciales dentro de la función de similitud se permite una comparación más precisa de los polígonos, y se posibilita la formación de clusters espacialmente contiguos y compactos. Pues segun el artículo [48], cuando el agrupamiento se basa únicamente en atributos no espaciales, se obtienen agrupaciones espacialmente disjuntas. Por lo tanto, los principios de la autocorrelación espacial y la heterogeneidad espacial no se cumplen, mientras que con la adición de atributos espaciales se producen clusters espacialmente contiguos y compactos. Considerándose un mejor resultado la estratificación que mejor cumpla con dichas características.

A continuación se presenta una **tabla comparativa** 3.7 entre los resultados del proceso de estratificación realizado por los dos modos planteados con anterioridad.

Tabla 3.7: *Comparación entre ambas estratificaciones.*

Resultados utilizando solamente medidas temáticas		Resultados obtenidos utilizando ambas medidas de similitud
estrato 1	Pinar del Río, Santiago de Cuba, Ciego de Ávila, Granma	Pinar del Río, Isla de la Juventud
estrato 2	Matanzas, Cienfuegos, La Habana, Camagüey, Guantánamo	Cienfuegos
estrato 3	Ciudad de la Habana	Villa Clara, Granma, Santiago de Cuba, Guantánamo, Las Tunas, Holguín, Camagüey,Sancti Spíritus, Ciego de Ávila
estrato 4	Holguín, Las Tunas, Villa Clara, Sancti Spíritus	La Habana, Ciudad de la Habana, Matanzas

Partiendo de los resultados obtenidos en ambos procesos de estratificación, y utilizando el metodo de comparación abordado en el artículo [48] anteriormente descrito, se aplica como criterio para comparar las estratificaciones la fórmula de **Gap Statistic** [49]. La cual plantea que mientras mayor sea el indice del resultado mejor es la estratificación.

Datos los cuales se presentan en la tabla 3.8.

Tabla 3.8: *Comparación entre ambas estratificaciones en cuanto al indice GAP.*

Estratificación utilizando solo medidas temáticas		Estratificación utilizando ambas medidas de similitud
Índice Gap	9.230654721	204.0688891

Demostrándose que existe una diferencia entre los resultados finales en ambos procesos. Esto posibilita verificar la veracidad de los estratos generados por la herramienta con las nuevas medidas de similitud desarrolladas, a partir del caso de estudio propuesto.

Los resultados mostrados demuestran la necesidad, utilidad y validez del empleo de las medidas de similitud geométrica en la estratificación de territorios, permitiendo obtener mejores respuestas en cuanto a la información estratificada de la situación salud-enfermedad del país, posibilitando implementar acciones aún más efectivas, y facilitando la mejora en la distribución de los recursos con un enfoque equitativo.

3.3. Conclusiones del capítulo

En el presente capítulo de cada HU se detallaron las tareas de ingeniería correspondientes, permitiendo que el trabajo se organizara en una secuencia lógica de pasos. Con la utilización de un correcto estándar de codificación se garantizó un buen entendimiento del código y una mejor organización del mismo. Al aplicar las pruebas de aceptación y caja blanca se pudo detectar, documentar y corregir las no conformidades existentes. La realización del caso de estudio evidenció la efectividad y veracidad de la solución presentada. El valor arrojado por la medida del error demostró que existe un grado de similitud aceptable entre los procesos descritos.

CONCLUSIONES

Como resultados de la presente investigación se obtuvo una propuesta de solución para las medidas de similitud geométrica en la estratificación de territorios sobre el QGis que contribuye al mejoramiento de la capacidad de gestión de dicho sistema y por tanto, de las entidades de salud que se ven beneficiados por las prestaciones de este. En función de los resultados obtenidos se arribó a las siguientes conclusiones:

- La definición del marco teórico referencial de la investigación relacionado con el proceso de estratificación de territorios, fundamentó la necesidad de implementar medidas de similitud geométricas entre polígonos para favorecer la incorporación de la componente espacial en el proceso de estratificación de territorios.
- La definición de criterios de similitud entre polígonos en función de las características de estudios de espacialidad en salud, facilitó el diseño de medidas que integran la componente espacial y pueden describir relaciones espaciales sobre objetos.
- La implementación de un componente que integra medidas de similitud geométricas en el proceso de estratificación de territorios basada en SIG facilita la incorporación del espacio en estudios salubristas y dota al SIG QGis de flexibilidad para integrar datos de variada naturaleza en estos estudios.
- Las pruebas aplicadas para la verificación de la solución informática y la valoración de los resultados a través de un caso de estudio demostró que el sistema cumple con los requisitos definidos, garantizando su correcto funcionamiento.

RECOMENDACIONES

Al concluir esta investigación, se recomienda para futuros trabajos asociados a esta área del conocimiento:

1. Definir otras estrategias para la medida de similitud basada en distancia cuando se está en presencia de multipolígonos y comparar los resultados con el método del centroide utilizado en este trabajo.
2. Diseñar otras medidas de similitud basadas en el criterio de la forma de polígonos para extender su uso en otros tipos de investigaciones.
3. Utilizar las medidas propuestas, en otros contextos asociados al análisis espacial y evaluar su comportamiento en comparación con otras medidas de la literatura.

REFERENCIAS BIBLIOGRÁFICAS

[1] Jiménez, J. M. R. Patrones pedagógicos en educación virtual. *Revista de Educación a Distancia*, 2009.

[2] Bravo, J. D. *Breve introducción a la cartografía ya los sistemas de información geográfica (SIG)*. Ciemat, 2000.

[3] Pérez Betancourt, Y.; Betancourt, Y. G. P.; Polanco, L. G.; Pérez, R. M. & Vega, Y. T. Estratificación de territorios basada en indicadores de salud sobre el Sistema de Información Geográfica QGIS. *Revista Cubana de Ciencias Informáticas*, 10(0):163–175, May 2016. Disponible en: http://rcci.uci.cu/?journal=rcci&page=article&op=view&path[]=1374.

[4] Harvey, D. & Rodrigo, G. L. *Teorías, leyes y modelos en geografía*. Alianza Editorial, 1983.

[5] Batista Moliner, R.; Coutin Marie, G.; Feal Cañizares, P.; González Cruz, R. & Rodríguez Milord, D. Determinación de estratos para priorizar intervenciones y evaluación en Salud Pública. *Revista Cubana de Higiene y Epidemiología*, 39(1):32–41, 2001.

[6] Batista Moliner, R.; Feal Cañizares, P.; Coutin Marie, G.; Rodríguez Milord, D. & González Cruz, R. Guía para la realización del proceso de estratificación epidemiológica. *La Habana: MINSAP*, 2001.

[7] Orallo, H.; RAMIREZ, J.; QUINTANA, C. R.; Orallo, M. J. H.; Quintana, M. J. R. & Ramírez, C. F. *Introducción a la Minería de Datos*. Pearson Prentice Hall,, 2004.

[8] Vegas Villalmanzo, I. Predicción de valores de bolsa mediante minería de datos para mercado de alta frecuencia. 2016.

[9] González Polanco, L. & Pérez Betancourt, G. La minería de datos espaciales y su aplicación en los estudios de salud y epidemiología. *Revista Cubana de Información en Ciencias de la Salud*, 24(4):482–489, 2013.

[10] Qian, Y. & Zhang, K. GraphZip: a fast and automatic compression method for spatial data clustering. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 571–575. ACM, 2004.

[11] Parra, C. A. H. Minería de Datos Espacial. 2006.

[12] Louden, K. C. *Lenguajes de programación: principios y práctica*. Cengage Learning Latin America, 2004.

[13] Antamba Yaselga, C. S. & Cabrera Gordón, A. G. Arrastre de sedimentos pluviales en meandros. B.S. thesis, Quito: UCE, 2015.

[14] Fonseca, F. T. *Ontology-driven geographic information systems*. PhD thesis, State University of New York, Buffalo, 2001.

[15] Bonillo, M. L. *Razonamiento basado en casos aplicado a problemas de clasificación*. PhD thesis, Tesis de grado. Universidad de Granada, España, 2003.

[16] Machado-García, N.; González-Ruiz, L. & Balmaseda-Espinosa, C. Recuperación de objetos geoespaciales utilizando medidas de similitud semántica. *Revista Cubana de Ciencias Informáticas*, 8(2):132–144, 2014.

[17] Sánchez, H. Optimización de una medida de Semejanza para Objetos Tridimensionales a Partir de Invariantes y Transformaciones. *Computación y Sistemas*, 3(004), 2000.

[18] Rapallo, R. Utilización de Sistema de Información Geográfica para la Seguridad Alimentaria sostenible en zonas marginadas de Honduras, Nicaragua y Guatemala (digital). FAO. Consultado el 6 de Octubre del 2008.

[19] Brisaboa, N. R.; Lema, J. A. C.; Fariña, A.; Luaces, M. R. & Viqueira, J. R. Sistemas de Información Geográfica: Revisión de su Estado Actual. *Ingeniería del Software en la Década*, pages 77–94, 2000.

[20] Carmona, A. & Monsalve, J. Sistemas de información geográficos. In *Congreso de Ingeniería de Sistemas en la Universidad San Buenaventura de Medellín Colombia*, 2004.

[21] López Caviedes, M. A. Herramienta para la estratificación de municipios en zonas de riesgo para la salud. 2004.

[22] Estratificador INEGI - Guia_EstratificadorV1_1.pdf. Disponible en: http://www3.inegi.org.mx/estratificador/assets/pdf/Guia_EstratificadorV1_1.pdf.

[23] Meneses Hernandez, J. M. & Cardenas Velasco, J. *Diseño e implementación de un sistema de información geográfico (SIG) sobre software libre para la Secretaría de Planeación del municipio de Guadalajara de Buga [recurso electrónico]*. PhD thesis, 2011.

[24] Schefer-Wenzl, S.; Sobernig, S. & Strembeck, M. Evaluating A Uml-Based Modeling Framework For Process-Related Security Properties: A Qualitative Multi-Method Study. In *ECIS*, page 134, 2013.

[25] Piattini, M.; Calvo-Manzano, J.; Cervera, J. & Fernández, L. Análisis y diseño de aplicaciones informáticas de gestión. *Alfa y Omega, Bogotá*, 2007.

[26] Génova, G.; Fuentes, J. & Valiente, M. Evaluación comparativa de herramientas CASE para UML desde el punto de vista notacional. *Novática*, 181:59–64, 2006.

[27] LOUDEN, K. C. *Lenguajes de programación: Principios y práctica*. Cengage Learning Latin America, 2004.

[28] Duque, R. G. Python para todos. 2011. Disponible en: mundogeek.net/tutorial-python/.

[29] Summerfield, M. *Rapid GUI programming with Python and Qt: the definitive guide to PyQt programming*. Pearson Education, 2007.

[30] Martínez, L. H. Intérprete y Entorno de Desarrollo para el Aprendizaje de Lenguajes de Programación Estructurada. In *JIISIC*, pages 189–196, 2008.

[31] Islam, Q. N. *Mastering PyCharm*. Packt Publishing Ltd, 2015.

[32] Letelier, P. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). Disponible en: www.cyta.com.ar/ta0502/b_v5n2a1.htm.

[33] Joskowicz, J. Reglas y prácticas en eXtreme Programming. *Universidad de Vigo*. Disponible en: http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf.

[34] Beck, K. *Extreme programming explained: embrace change*. Disponible en: https://books.google.es/books?hl=es&lr=&id=G8EL4H4vf7UC&oi=fnd&pg=PR13&dq=Extreme+programming+explained:+embrace+change+[online].+Addison-Wesley+Professional.+&ots=j9yIrujYxk&sig=8XY-aKnAt1OXxjtMi8nrIj0QwQ8#v=onepage&q=Extreme%20programming%20explained%3A%20embrace%20change%20[online].%20Addison-Wesley%20Professional.&f=false.

[35] Yreta, A. & Alberto, M. *Cómputo de la similitud entre figuras geométricas*. PhD thesis, Instituto Politécnico Nacional. Centro de Investigación en Computación, 2006.

[36] Ayres, F.; Mendelson, E. & Rapún, L. A. *Cálculo diferencial e integral*. McGraw-Hill, 1991.

[37] Godino, J. D.; del Carmen Batanero, M. & Roa, R. *Medida de magnitudes y su didáctica para maestros*. Universidad de Granada, Departamento de Didáctica de la Matemática, 2002.

[38] Pressman, R. S. Ingeniería del software. Ed, 2005.

[39] Cuadra, P. G. Técnicas para la obtención de Requerimientos. Disponible en: https://es.slideshare.net/PedroGutirrezCuadra/tcnicas-para-la-obtencin-de-requerimientos.

[40] Martín, J. M. F. Optimización del proceso de gestión de requisitos. *Byte España*, (138):82–86, 2007.

[41] Sommerville, I. & Galipienso, M. I. A. *Ingeniería del software*. Disponible en: https://books.google.es/books?hl=es&lr=&id=gQWd49zSut4C&oi=fnd&pg=PA1&dq=Ingenier%C3%ADa+del+software.+Pearson+Educaci%C3%B3n.&ots=s656ptwvtf&sig=rNlnqw86hZOeG7oXJPvvZJzJEo8#v=onepage&q=Ingenier%C3%ADa%20del%20software.%20Pearson%20Educaci%C3%B3n.&f=false.

[42] Joskowicz, J. Reglas y prácticas en eXtreme Programming. *Universidad de Vigo*, page 22, 2008.

[43] Casas, S. & Reinaga, H. Identificación y modelado de aspectos tempranos dirigido por tarjetas de responsabilidades y colaboraciones. In *XIV Congreso Argentino de Ciencias de la Computación*. Disponible en: http://sedici.unlp.edu.ar/handle/10915/21813.

[44] Pressman, R. Ingeniería del software. Un enfoque práctico. Sexta edición. Editoria l McGraw-Hill, 2005.

[45] Garrido, J. C. Arquitectura y diseño de sistemas Web modernos. *InforMAS, Revista de Ingeniería Informática del CIIRM*, (1), 2004.

[46] 06Patrones - 06Patrones.pdf. Disponible en: http://siul02.si.ehu.es/~alfredo/iso/06Patrones.pdf.

[47] Anuario Estadístico de Cuba | Biblioteca Virtual en Salud de Cuba. Disponible en: http://bvscuba.sld.cu/anuario-estadistico-de-cuba/.

[48] Joshi, D. Polygonal spatial clustering. 2011.

[49] Tibshirani, R.; Walther, G. & Hastie, T. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

GLOSARIO DE TÉRMINOS

GRASS Geographic Resources Analysis Support System

SAGA Sistema de Análisis Geocientífico Automatizado (del inglés System for Automated Geoscientific Analyses)

Qgis Quantum GIS

SIG Sistemas de Información Geográfica (del inglés Geographic Information Systems)

CASE Herramienta que brinda asistencia a los analistas (del inglés Computer Aided Software Engineering)

CRC Clase, Responsabilidad y Colaboración (del inglés Class, Responsability and Collaboration)

GoF Patrones del Grupo de Cuatro (del inglés Gang of Four)

GRASP Patrones Generales de Software para Asignación de Responsabilidades (del inglés General Responsibility Assignment Software Patterns)

HU Historia de Usuario (artefacto generado por la metodología Programación Extrema)

RF Requisito Funcional

RNF Requisito No Funcional

XP Metodología de desarrollo de software ágil (del inglés Extreme Programing)