



Centro de Investigación y de Estudios Avanzados  
del Instituto Politécnico Nacional  
Unidad Guadalajara

# **Nuevo Marco Conceptual de Sistemas de Autoingeniería y una Arquitectura para la Gestión de su Ciclo de Vida**

Tesis que presenta:  
**Giovana Patricia Pérez Carrillo**

para obtener el grado de:  
**Maestra en Ciencias**

en la especialidad de:  
**Ingeniería Eléctrica**

Directores de Tesis  
**Dr. Mario Angel Siller González Pico**  
**Dr. Luis Alberto Alonso Pastor**

# **Nuevo Marco Conceptual de Sistemas de Autoingeniería y una Arquitectura para la Gestión de su Ciclo de Vida**

## **Tesis de Maestría en Ciencias Ingeniería Eléctrica**

Por:

**Giovana Patricia Pérez Carrillo**

Ingeniera en Diseño Electrónico y Sistemas Inteligentes  
Centro de Enseñanza Técnica Industrial 2018-2022

Becaria de CONAHCYT, expediente no. 1229663

Directores de Tesis:

**Dr. Mario Angel Siller González Pico**

**Dr. Luis Alberto Alonso Pastor**

# Resumen

En los últimos años, los sistemas de autoingeniería (“Self-engineering Systems” - SES) se han estudiado en la literatura y definido desde una perspectiva de detección y corrección de fallos en tiempo de ejecución, dejando conceptualmente fuera aspectos y actividades fundamentales asociados a los procesos de ingeniería de sistemas, todo dentro del mismo ciclo de vida del sistema. En este sentido, no existe un concepto general en la literatura que defina a los sistemas que aborden dichos aspectos y actividades fundamentales y cuya ingeniería se realice sin intervención humana.

Este trabajo propone un marco conceptual para redefinir el término de autoingeniería, además de una arquitectura para la gestión del ciclo de vida de los SES, diversos parámetros y condiciones para evaluar la evolución de estos sistemas, y un caso de estudio con fines de validación tanto del marco conceptual como del la arquitectura propuesta. Como parte del marco conceptual destaca el término *metamorfosis*, que describe el paso evolutivo de un sistema original a la creación de uno nuevo cuando este es desecharido. Esto último deriva de un punto de quiebre en la elasticidad del sistema durante su ciclo de vida y es causado por los cambios de requerimientos del sistema.

# Abstract

In recent years, the concept of self-engineering systems (SES) has been the subject of research and analysis within the literature. Nevertheless, a consensus on a definition encompassing the full range of activities and processes associated with systems engineering has yet to be reached. In this regard, there is no overarching concept in the extant literature that defines systems that address such foundational aspects and activities and whose engineering is performed without human intervention.

This work proposes a conceptual framework for re-defining the term *self-engineering* and an architectural model for SES life cycle management. Furthermore, it presents a set of parameters and conditions for evaluating the evolution of these systems, as well as a case study to validate both the conceptual framework and the proposed architectural model. As part of the conceptual framework, the term *metamorphosis* is of particular interest. This concept describes the evolutionary step from an original system to the creation of a new one when the original system is discarded. This latter state arises from a break point in the elasticity of the system during its life cycle, which is caused by changes in system requirements.

# Dedicatoria

*A mi madre, por ser un ejemplo de fortaleza. Agradezco infinitamente tu amor y apoyo incondicional; eres y siempre serás la mejor.*

*A Javier, por tu amor sincero y por siempre estar ahí para mí. Por ser esa persona que me impulsa a seguir mis sueños.*

# Agradecimientos

A mi madre y a mi hermano, por toda su paciencia, fortaleza y amor brindados durante este trayecto.

A mi padre, porque gracias a su apoyo he podido llegar hasta aquí.

A Javier, por su amor, sus consejos y su compañía incondicional.

A mis amigos incondicionales Fernando, Sofía y Alejandra, gracias por su apoyo durante esta etapa, incluso a la distancia.

A mis asesores de tesis, el Dr. Mario Siller Ángel Siller González Pico y el Dr. Luis Alberto Alonso Pastor, por su apoyo invaluable durante el desarrollo de esta tesis.

Al grupo de redes, gracias por su apoyo y amistad. Especialmente a Liliana y Diego, por su apoyo, paciencia y por compartir sus conocimiento.

A mis compañeros y amigos del laboratorio de computación por sus consejos, amistad y compañía.

A Conahcyt por el apoyo económico brindado, con el cual se logró estudiar esta maestría.

# Índice de Contenido

<b>1</b>	<b>Introducción</b>	<b>11</b>
1.1	Problema . . . . .	11
1.2	Motivación . . . . .	12
1.3	Objetivo General . . . . .	13
1.4	Objetivos Específicos . . . . .	13
1.5	Hipótesis . . . . .	14
1.6	Estructura del documento del tesis . . . . .	14
<b>2</b>	<b>Marco Teórico</b>	<b>15</b>
2.1	Ingeniería de sistemas . . . . .	15
2.1.1	Ingeniería de software . . . . .	15
2.1.2	Ingeniería de requerimientos . . . . .	17
2.2	Ciclo de vida de un sistema . . . . .	18
2.3	Operación y mantenimiento de un sistema de software . . . . .	18
2.4	Evolución de un sistema de software . . . . .	20
2.4.1	Leyes de evolución del Software . . . . .	20
2.4.2	Gestión de cambio . . . . .	21
2.5	Definiciones de Metamorfosis . . . . .	22
2.6	Adaptación y Autoadaptación . . . . .	22
2.7	Cómputo Autonómico . . . . .	22
<b>3</b>	<b>Estado del Arte</b>	<b>24</b>
3.1	Conceptualización de sistemas de autoingeniería . . . . .	24
3.2	Arquitecturas de referencia . . . . .	25
3.3	Modelado basado en agentes para ingeniería de software . . . . .	27
3.4	Comparación de trabajos . . . . .	27

<b>4 Propuesta</b>	<b>29</b>
4.1 Metodología . . . . .	29
4.1.1 Revisión de la literatura . . . . .	29
4.1.2 Definición y conceptualización . . . . .	35
4.1.3 Análisis y validación . . . . .	36
4.2 Propuesta marco conceptual y arquitectura: Sistemas de autoingeniería.	38
4.2.1 Definición conceptual y formal . . . . .	38
4.2.2 Propuesta arquitectónica . . . . .	48
4.2.3 Resumen conceptual . . . . .	56
<b>5 Experimentación y Análisis de Resultados</b>	<b>60</b>
5.1 Análisis de sistemas para validación del marco conceptual . . . . .	60
5.1.1 Análisis del sistema: autoingeniería . . . . .	60
5.1.2 Análisis de cambios en un sistema: autoadaptación y metamorfosis . . . . .	62
5.2 Experimentación . . . . .	66
5.2.1 Caso de estudio: ciclo de vida de un sistema SES con aplicación a un sistema ciberfísico de exploración, coordinación y rescate	66
5.2.2 Modelo sistema original . . . . .	66
5.2.3 Experimento: autoadaptación . . . . .	71
5.2.4 Experimento: metamorfosis . . . . .	73
5.2.5 Análisis de resultados . . . . .	75
<b>6 Conclusiones y Trabajo Futuro</b>	<b>81</b>
6.1 Conclusiones . . . . .	81
6.2 Objetivos logrados . . . . .	82
6.3 Trabajo futuro . . . . .	82

# List of Figures

2.1	Modelo de cascada . . . . .	16
2.2	Proceso de ingeniería de sistemas a lo largo del ciclo de vida del sistema. . . . .	17
3.1	Comparativa de definiciones previas ubicadas en el ciclo de vida de un sistema (a: [62], b: [13].) . . . . .	25
3.2	Bucle de control MAPE-K [46] . . . . .	26
4.1	Metodología . . . . .	30
4.2	Proceso de realización de revisión sistemática . . . . .	31
4.3	Proceso de selección de artículos . . . . .	36
4.4	Representación temporal de la selección de artículos . . . . .	37
4.5	Representación de selección de artículos acorde a la palabra clave . . . . .	37
4.6	Ciclo de vida de un sistema de autoingeniería . . . . .	38
4.7	Comparativa de sistemas con similitudes a los sistemas de autoingeniería ubicados en el ciclo de vida de un sistema (a: [41], b:[65], c: [63], d: [31], e: [17], f: [23], g: [33], h: [4], i: [66], j: [47], k: [68]). . . . .	40
4.8	Componentes de un requerimiento funcional. . . . .	41
4.9	Proceso de gestión de cambio. . . . .	42
4.10	Metamorfosis por transformación directa. . . . .	43
4.11	Metamorfosis por integración . . . . .	43
4.12	Diagrama de flujo de los sistemas de autoingeniería . . . . .	44
4.13	Diagrama de actividad para el proceso de gestión de cambio . . . . .	45
4.14	Técnicas propuestas para la implementación de las etapas de la ingeniería de sistemas (a:[28], b: [2], c: [27], d: [61], e: [63], f:[47], g:[25], h:[56], i:[18], j:[3], k:[50]) . . . . .	46
4.15	Cambio de arquitectura de un <i>sistema A</i> a un <i>sistema B</i> . . . . .	49
4.16	¿Qué tanto hay en la arquitectura del nuevo sistema del sistema anterior? . . . . .	49
4.17	Éter de sistemas de autoingeniería . . . . .	50

# Nuevo Marco Conceptual de Sistemas de Autoingeniería y una Arquitectura para la Gestión de su Ciclo de Vida

---

4.18	Arquitectura de un sistema de autoingeniería como el éter. . . . .	50
4.19	Arquitectura SES como el éter de los sistemas de autoingeniería en su ciclo de vida. . . . .	51
4.20	Éter de la arquitectura en el proceso de ingeniería de sistemas . . . . .	51
4.21	Analogía marco MAPE-K y arquitectura SES . . . . .	52
4.22	Vista de despliegue de la arquitectura SES . . . . .	55
4.23	Analogía marco MAPE-K y arquitectura SES con agentes . . . . .	56
4.24	Vista de procesos de la arquitectura SES con agentes . . . . .	56
4.25	Diagrama de actividad de la arquitectura de los SES con agentes . . .	57
4.26	Vista lógica de la arquitectura de sistemas SES . . . . .	57
4.27	Diagrama de estados abstractos de la arquitectura de los SES con agentes . . . . .	58
4.28	Mapa conceptual de los sistemas de autoingeniería . . . . .	59
5.1	Diagrama de entidad relación de los agentes del sistema . . . . .	67
5.2	Arquitectura sistema A . . . . .	68
5.3	Procesos de los agentes en la arquitectura de sistemas SES . . . . .	69
5.4	Procesos de los agentes en la arquitectura de sistemas SES . . . . .	70
5.5	Implementación Sistema A . . . . .	71
5.6	Simulación de la implementación del sistema A evolucionado . . . . .	73
5.7	Arquitectura con los cambios implementados . . . . .	75
5.8	Simulación de la implementación del sistema A con cambios . . . . .	77
5.9	Evolución del sistema . . . . .	78
5.10	Ánalisis de cumplimiento de requerimientos en la fase de autoadaptación de un SES . . . . .	78
5.11	Ánalisis de cumplimiento de requerimientos en la fase de metamorfosis de un SES . . . . .	80
5.12	Ánalisis resultados . . . . .	80

# List of Tables

3.1	Análisis comparativo de definiciones previas como referencia para propuesta . . . . .	28
3.2	Análisis comparativo de trabajos seleccionados como referencia para propuesta . . . . .	28
4.1	Requerimientos funcionales para un sistema de autoingeniería . . . . .	53
4.2	Requerimientos no funcionales de un sistema de autoingeniería . . . . .	54
5.2	Funcionalidad y configuración del RF3 . . . . .	63
5.1	Requerimientos sistema cliente-servidor . . . . .	64
5.4	Requerimientos añadidos al sistema cliente-servidor . . . . .	65
5.3	Funcionalidad y configuración de requerimientos de metamorfosis . . . . .	65
5.5	Requerimientos funcionales sistema A . . . . .	72
5.6	Requerimientos funcionales Sistema A-V2 . . . . .	74
5.7	Requerimientos sistema B . . . . .	76

# Capítulo 1

## Introducción

Los sistemas de autoingeniería (“Self-engineering Systems” - SES) han cobrado relevancia en los últimos años, caracterizándose por su capacidad de detección de fallos y de preservar el funcionamiento del sistema mediante técnicas como autosanación, autoreparación, autoadaptación, autoconfiguración, entre otras. Además, estos sistemas se distinguen por tener como objetivo principal el prolongar su vida útil y mejorar su resiliencia [13]. Una de las definiciones conceptuales más recientes atribuidas a los SES se puede encontrar en [62], donde se definen como sistemas que registran una pérdida en su funcionamiento u operación y responden de manera automática realizando una o más acciones encaminadas para restaurar su funcionalidad.

En este trabajo se revisa la teoría existente en la literatura sobre los SES con el fin de identificar brechas conceptuales y de investigación que puedan ser abordadas a partir de la definición de un nuevo marco conceptual, en línea con las nuevas tendencias tecnológicas y de investigación dentro de la ingeniería de sistemas, la ingeniería de software y la inteligencia artificial. En este sentido, este trabajo aborda un *nuevo paradigma en el que la ingeniería se realiza con baja o nula intervención humana*. El marco conceptual propuesto se plantea en el contexto general de la teoría de sistemas, pero su aplicación se enfoca en la ingeniería de sistemas de software, IoT y ciberfísicos.

### 1.1 Problema

A lo largo del ciclo de vida de un sistema, posterior al despliegue de dicho sistema, este es susceptible a presentar cambios en sus requerimientos, una etapa conocida como *evolución del sistema* [64]. Tradicionalmente, durante dicha etapa pueden solicitarse múltiples cambios en los requerimientos cuya gestión podría dar lugar a

dos escenarios:

- (i) Es posible acoger el cambio, el sistema se modifica y continúa su ciclo de vida.
- (ii) No es posible acoger el cambio, el sistema se desecha y se desarrolla uno nuevo con dichos requerimientos, en cuyo caso, puede implicar o no el reuso parcial del sistema desecharido, y su desarrollo requiere la intervención humana.

A partir de lo anterior se plantean los siguientes cuestionamientos:

- *Ante el segundo escenario planteado, ¿es posible llevar a cabo la ingeniería del nuevo sistema sin intervención humana? En caso afirmativo, ¿cómo se realizaría dicha ingeniería, es decir, desarrollar un nuevo sistema (sistema B) a partir del sistema original (sistema A)?*
- *¿Es posible gestionar la ingeniería de un sistema sin intervención humana y que esta trascienda más allá de su propio ciclo de vida?*
- *¿Es posible llevar a cabo la gestión del cambio de los requerimientos que surgen durante la etapa de OEM y evolución sin intervención humana?*

En este contexto de ingeniería sin intervención humana (“self-engineering”), la literatura actual se limita a tratar las fallas y a restaurar el funcionamiento del sistema en tiempo de ejecución. Esta limitación deja fuera a otras actividades y aspectos fundamentales del desarrollo de la ingeniería de sistemas. Por ello, lo anterior constituye un tema abierto de investigación que se abordará en el presente trabajo. Asimismo, términos estrechamente relacionados con la autoingeniería, como *evolución* y *autoadaptación*, se han utilizado para describir la capacidad de los sistemas para gestionar y afrontar cambios. Sin embargo, la implementación de estas capacidades en el sistema se puede realizar sin considerar necesariamente las diversas actividades del proceso de ingeniería de sistemas y se restringe exclusivamente a un mismo ciclo de vida de los sistemas, es decir, no se aborda más allá del desecho de un sistema ni un nuevo inicio de ciclo de vida.

## 1.2 Motivación

El progreso en el desarrollo de sistemas tecnológicos ha experimentado un crecimiento notable en los últimos años. La humanidad ha avanzado considerablemente en el ámbito de la tecnología, pasando de diseñar simples máquinas mecánicas destinadas

a facilitar tareas humanas a desarrollar inteligencia artificial capaz de crear y generar soluciones que hasta hace poco parecían imposibles. Los sistemas IoT son un claro ejemplo de este progreso, ya que han experimentado un crecimiento exponencial en el número de dispositivos interconectados, contribuyendo a la creación de ciudades inteligentes y mejorando la gestión de recursos urbanos, el tráfico, la energía y otras áreas. Sin embargo, la gestión y el mantenimiento de estos sistemas, incluidos los sistemas de software, aún dependen en gran medida de las personas, lo que resulta costoso y requiere una considerable inversión de tiempo. Aunque se han propuesto sistemas autónomos capaces de autogestionarse sin intervención humana como solución prometedora para reducir los costes asociados al desarrollo y mantenimiento de los sistemas, aún no se ha abordado la ingeniería de sistemas sin dicha intervención para dar respuesta a cuando surgen cambios en los requerimientos y que ello pueda implicar ir más allá de su propio ciclo de vida.

La motivación central de este trabajo radica en la necesidad de desarrollar sistemas capaces de realizar, sin intervención humana, la ingeniería derivada de los cambios en los requerimientos cuando estos dan origen al desecho del sistema y al desarrollo de uno nuevo. Este problema se aborda mediante la propuesta de un marco conceptual denominado *sistemas de autoingeniería*.

### **1.3 Objetivo General**

Proponer un nuevo marco conceptual y arquitectura para los sistemas de autoingeniería para la gestión de su ciclo de vida, a partir de los cuales sea posible dar respuesta a cambios de requerimientos cuando estos dan origen al desecho del sistema y al desarrollo de uno nuevo, sin intervención humana.

### **1.4 Objetivos Específicos**

- Realizar una revisión sistemática de la literatura sobre la teoría existente de sistemas de autoingeniería con el propósito de evaluar si dicha teoría aborda la gestión de cambios de requerimientos que involucren las realización de actividades fundamentales de ingeniería de sistemas, tales como la ingeniería de requerimientos, diseño, implementación, pruebas, y evolución (operación y mantenimiento), sin intervención humana y que vayan más allá del propio ciclo de vida del sistema.
- Estudiar e investigar en el contexto de la ingeniería de sistemas la situación en

la que un sistema no pueda seguir evolucionando dentro de su ciclo de vida y en la que fuese necesario desecharlo y desarrollar un nuevo sistema a partir de este.

- Redefinir la autoingeniería (“self-engineering”) de sistemas y, con ello, crear un nuevo marco conceptual en el que se incluya la descripción de sus propiedades y funciones asociadas.
- Desarrollar una arquitectura para la gestión del ciclo de vida de sistemas de autoingeniería.
- Validar el marco conceptual y arquitectura propuestos mediante casos de estudio, utilizando planteamientos y análisis tanto teóricos como experimentales a través de simulaciones.

## 1.5 Hipótesis

En este trabajo se plantea que, a partir de un nuevo marco conceptual y una arquitectura de sistemas de autoingeniería, es posible desarrollar sistemas que den respuesta a cambios en los requerimientos que puedan implicar ir más allá de su propio ciclo de vida, así como gestionar la ingeniería correspondiente sin intervención humana.

## 1.6 Estructura del documento del tesis

La estructura del presente trabajo se organiza de la siguiente manera: en el capítulo 2, se presenta el marco teórico de esta investigación en donde se desarrolla el contenido teórico relevante existente en la literatura. Posteriormente, en el capítulo 3, se presenta una revisión del estado del arte, en donde se identifican los trabajos previamente propuestos que aborden temas relacionados con la definición de sistemas de autoingeniería (SES), la utilización de arquitecturas de sistemas autoadaptativos, las funcionalidades del cómputo autonómico, los cambios en los requerimientos de los sistemas, la autoadaptación y la evolución. Después, en el capítulo 4, se describe la propuesta en la que se presenta la metodología implementada para el desarrollo de la investigación, así como un marco conceptual de los sistemas de autoingeniería (SES), su modelo formal, y el diseño de su arquitectura. En el capítulo 5 se presentan la experimentación así como análisis teóricos basados en casos de estudio. Finalmente, en el capítulo 6, se incluyen las conclusiones de este trabajo y el trabajo futuro.

# Capítulo 2

## Marco Teórico

### 2.1 Ingeniería de sistemas

Un sistema se entiende como una estructura o conjunto compuesto por diversos elementos que, al interactuar, generan resultados que no podrían lograrse de manera individual [43, 24]. En la actualidad, existen diversos tipos de sistemas, entre ellos los mecánicos, eléctricos, de control y tecnológicos, como los sistemas ciberfísicos, de software y de Internet de las Cosas (IoT). En general, los sistemas se han diseñado para asistir en la realización de tareas específicas humanas.

Para el desarrollo de un sistema se siguen procesos metódicos establecidos que consisten en la aplicación de los principios de la ingeniería a las diversas actividades que se realizan para tal efecto, incluyendo entre otras, la conceptualización, el diseño, implementación, pruebas, despliegue y certificación del funcionamiento del sistema [26].

En un mundo tan interconectado y revolucionado por la tecnología, el avance y el desarrollo de sistemas es inminente, por lo que la ingeniería de sistemas desempeña un papel crucial para asegurar que los sistemas puedan desempeñar sus funciones y satisfacer las necesidades de los usuarios, que son el fin y propósito de su desarrollo.

#### 2.1.1 Ingeniería de software

El término *software* se refiere a los programas y documentación asociada a un sistema computacional cuyas funciones son realizadas o soportadas por *hardware* [64]. En consecuencia, un sistema de software carece de componentes físicos, lo que implica que estos no están restringidos por cuestiones físicas, materiales o de componentes tangibles. Por otro lado, esta falta de limitación en sus propiedades hace que un

sistema de software pueda llegar a ser complejo y costoso de implementar.

El término de *ingeniería de software* surgió en el año 1968 [57], cuando se afirmó que un enfoque individual para el desarrollo de cada sistema de software era complejo y costoso. Por ello, se sostuvo la necesidad de desarrollar nuevas técnicas y métodos para la gestión del desarrollo de sistemas software. Por esta razón, es necesario seguir un procedimiento metódico y bien definido para desarrollar cualquier sistema de software, aunque cabe destacar que no todos se implementan de la misma manera.

Así pues, para desarrollar un sistema de software es necesario llevar a cabo las tareas de ingeniería de requerimientos, diseño, codificación, pruebas y mantenimiento del software en cuestión [11], lo que deriva en una *ingeniería de software*, una disciplina de la ingeniería que aborda todos los aspectos de la producción de software e implementa todas las actividades implicadas en el desarrollo del sistema, incluyendo la etapa de *requerimientos, diseño, implementación, pruebas y mantenimiento* [64].

Existen distintos modelos del proceso de desarrollo de software, tales como el modelo de cascada (figura 2.1), modelo en espiral de Boehm, desarrollo incremental, orientado a la reutilización, desarrollo ágil, desarrollo scrum, modelo 'V', modelo RAD, modelo de prototipo, etc. Estos modelos corresponden a abstracciones del proceso utilizado para explicar los distintos enfoques y especificar las actividades asociadas al desarrollo del software.

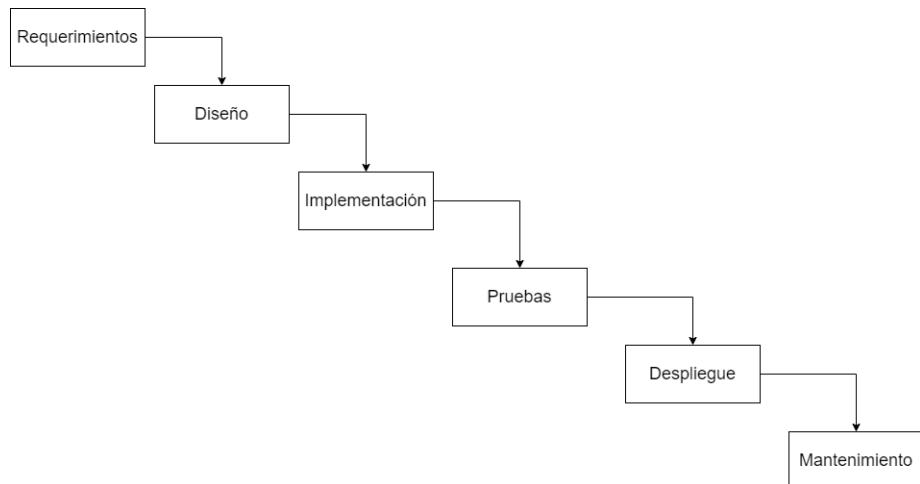


Figure 2.1: Modelo de cascada

### 2.1.2 Ingeniería de requerimientos

Un requerimiento corresponde a una sentencia en donde se expresa una característica o restricción operativa, funcional o de diseño del sistema que se va a desarrollar, y cuyo cumplimiento es necesario para su aceptabilidad [42]. En consecuencia, la etapa de requerimientos constituye la base del desarrollo de cualquier sistema, ya que en ella se define el problema que se va a atacar a través de una solución. Asimismo, en esta etapa de especificación de requerimientos, se identifica lo que el sistema debe realizar, su alcance o límites y qué necesidades de los usuarios, clientes o desarrolladores se deben cumplir.

Una clasificación comúnmente utilizada para los tipos de requerimientos del sistema corresponden a *los funcionales* y *los no funcionales*. Siendo los *funcionales* aquellos que describen las prestaciones del sistema, expresado mediante servicios, tareas o funciones que el sistema debe de realizar, así como también las condiciones bajo las cuales ciertas entradas deben convertirse en ciertas salidas [54]. Los requerimientos funcionales también podrían especificar para ciertos tipos de sistemas y condiciones lo que el sistema no debiera hacer. Por otro lado, los requerimientos *no funcionales* corresponden a los aspectos relacionados a las propiedades del sistema, su alcance y limitaciones bajo las cuales este tendrá que operar [35].

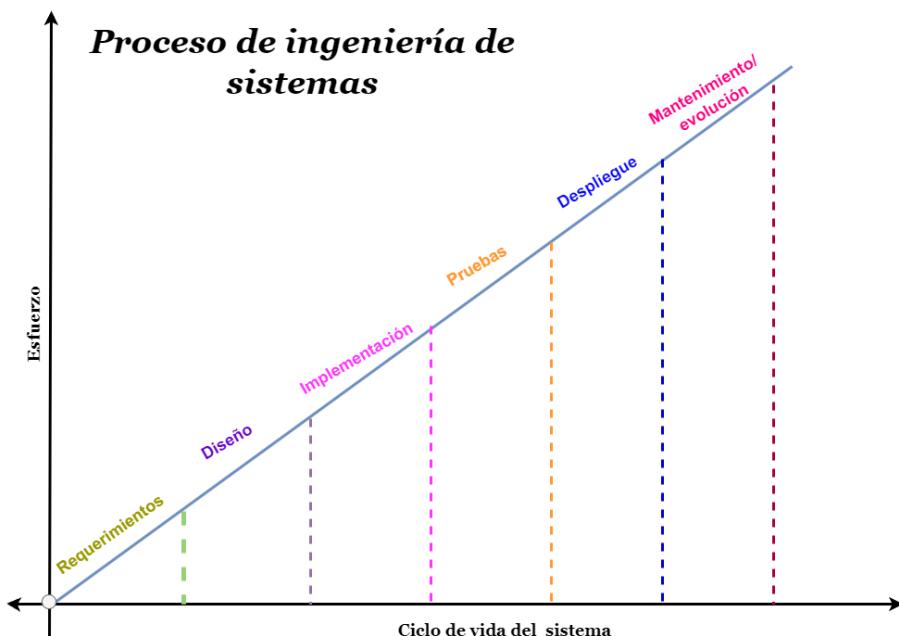


Figure 2.2: Proceso de ingeniería de sistemas a lo largo del ciclo de vida del sistema.

## 2.2 Ciclo de vida de un sistema

El ciclo de vida de los sistemas de software comprende desde su planeación hasta su desecho. Para describir y establecer un estándar entre las etapas del ciclo de vida de los sistemas de software, se ofrece un marco común para su proceso [44], denominado *Ciclo de Vida de Desarrollo de Software* (SDLC, por sus siglas en inglés). Básicamente, este consiste en un proceso organizado en etapas relacionadas entre sí que describen el proceso que sigue un sistema de software durante su vida útil.

Las etapas del SDLC son las siguientes [60]:

- *Inicialización*: en esta etapa se plantea la necesidad de desarrollar el sistema, así como su propósito y sus requerimientos. Dichos requerimientos son evaluados mediante la negociación entre las partes interesadas y los desarrolladores para llegar a acuerdos mutuos.
- *Adquisición/Desarrollo*: en primer lugar, se procesa, almacena y transmite la información necesaria para el desarrollo del sistema, según las especificaciones establecidas. Posteriormente, el sistema se diseña, programa y desarrolla.
- *Implementación*: en esta fase se revisa en detalle el diseño del sistema propuesto y se llevan a cabo pruebas exhaustivas al sistema antes de su implementación. Una vez realizadas las pruebas de las funcionalidades del sistema para asegurar su correcto funcionamiento, así como la validación del cumplimiento satisfactorio de los requerimientos, se implementa el sistema en cuestión en campo.
- *Operación&mantenimiento*: en esta etapa el sistema se encuentra en funcionamiento brindando los servicios para los cuales fue diseñado. Cualquier ajuste o modificación necesaria en el software o hardware se implementa según sea necesario para optimizar su rendimiento, así como al atención a sus posibles cambios en requerimientos.
- *Desecho*: esta fase implica realizar la evolución o la transición del sistema. Así bien, si se decide dar por finalizado al sistema, este puede desecharse, migrar o reutilizarse.

## 2.3 Operación y mantenimiento de un sistema de software

Se ha mencionado en las secciones anteriores el proceso de ingeniería de sistemas, el cual detalla las distintas etapas que intervienen el desarrollo de un sistema, in-

cluyendo *ingeniería de requerimientos*, para posteriormente generar un *diseño* del sistema (ingeniería de diseño), mediante una arquitectura, lo solicitado y acordado en los requerimientos, para así pasar a la etapa en donde se *implementa* el sistema. Posteriormente, el sistema se somete a una serie de *pruebas* para su verificación y validación. Finalmente, una vez realizado todo esto, el sistema es *desplegado*. Cuando el sistema ha sido desplegado, se dice que este entra en operación, iniciando entonces su fase de *operación y mantenimiento (O&M)* del ciclo de vida, en la que pueden ocurrir cambios derivados del ambiente en el que opera el sistema o modificaciones en las necesidades del usuario. Además de posibles cambios asociados al mantenimiento preventivo y correctivo del sistema.

El mantenimiento que puede ser requerido en el sistema varía entre las siguientes cuatro clases [53]:

- *Mantenimiento correctivo*: este tipo de mantenimiento se centra en la reparación de posibles fallos que puedan surgir y en atender a errores del sistema durante su tiempo de ejecución.
- *Mantenimiento adaptativo*: Se centra en la atención de cambios del tipo adaptativo, que corresponden a cambios en los que el sistema deba ajustarse a nuevos ambientes.
- *Mantenimiento perfeccionador*: este mantenimiento tiene como objetivo actualizar el sistema según los cambios en los requerimientos solicitados por el usuario final.
- *Mantenimiento preventivo*: este tipo de mantenimiento requiere una actualización de la documentación del sistema, así como los pertinentes ajustes para que sea más mantible y resiliente ante posibles fallos.

Algunos de los retos que se han planteado en relación a la etapa de mantenimiento son [52]: el papel que desarrollan los usuarios, la gestión de los cambios y las herramientas necesarias para realizarlos. Existen diferentes modelos de procesos asociados a la etapa de mantenimiento. Por ejemplo, el modelo *Quick Fix* [37] el cual tiene como objetivo identificar el problema actual y solucionarlo de la manera más rápida posible. Este modelo trabaja y es ejecutado de una manera rápida con bajo costo. Una de sus desventajas es que la confiabilidad y mantenibilidad del sistema puede verse deteriorada, por lo cual debe evaluarse en qué tipo de sistemas es viable aplicarlo [49]. Por otro lado, en el modelo de reutilización total [5], se realiza un análisis de los requerimientos del nuevo sistema, reutilizando todo lo posible del sistema anterior. Así bien, se construye un nuevo sistema utilizando documentación y componentes de los sistemas disponibles para el reuso.

## 2.4 Evolución de un sistema de software

La evolución de un sistema de software consiste en una secuencia de estados y las transiciones entre ellos del software desde su creación hasta su eventual desecho, comprendiendo así las actividades y procesos de mantenimiento de software que generan una nueva versión del sistema [20]. Así bien la evolución comprende entonces la etapa posterior al desarrollo y mantenimiento de la primera versión del sistema (despliegue del sistema) [30]. En esta etapa se atienden los cambios en las necesidades de los usuarios y problemáticas que esta implican (requerimientos).

Con el paso de los años y el incremento de la demanda de nuevos sistemas, la evolución del software ha desempeñado un papel cada vez más importante en el desarrollo de los sistemas. La evolución y el mantenimiento de un sistema son las etapas que generan más tiempo y coste de desarrollo. Los retos en el proceso de evolución han servido como referencia para demostrar la necesidad de ampliar la investigación en el campo de la evolución de los sistemas de software mediante el uso de tecnologías que consideren todo el ciclo de vida y los potenciales problemas del mantenimiento [8]. El ciclo de vida de un sistema no termina con su despliegue, sino que este sigue vigente hasta su mantenimiento y evolución, en donde se atienden los cambios que le puedan ser requeridos al sistema posterior a dicho despliegue.

### 2.4.1 Leyes de evolución del Software

Las leyes de evolución del software [51] dedicadas a la evolución de un programa de software, la cual deriva en nuevas versiones o entregas de un software que ya existe [29] surgen como resultado de estudios empíricos aplicados a distintos sistemas de software establecidos, evaluándolos a lo largo de su ciclo de vida, en donde los requerimientos se modificaron para reflejar las necesidades de los usuarios y partes interesadas en el proyecto. A continuación se presentan dichas leyes de evolución del software:

1. *Cambio constante*: los sistemas deben adaptarse continuamente, de lo contrario se vuelven menos satisfactorios.
2. *Incremento de la complejidad*: a medida que un sistema evoluciona, su complejidad aumenta, a menos que se tomen medidas para reducirla.
3. *Autoregulación*: el proceso de evolución se autorregula con distribuciones y patrones estadísticos predecibles.

4. *Conservación organizacional*: la tasa efectiva de la evolución corresponde a la estabilidad invariante a lo largo de la vida útil del producto.
5. *Conservación de la familiaridad*: a medida que el sistema evoluciona, todo lo asociado a él debe mantener el dominio de su contenido y comportamiento para lograr una evolución satisfactoria.
6. *Crecimiento continuo*: el contenido funcional de un sistema a lo largo de su vida útil debe aumentar continuamente para mantener la satisfacción del usuario.
7. *Calidad decreciente*: la calidad de los sistemas parecerá decreciente a menos que se les de un mantenimiento riguroso y se adapten a los cambios del entorno operativo.
8. *Retroalimentación del sistema*: los procesos de evolución constituyen sistemas de retroalimentación de usuarios y desarrolladores.

#### 2.4.2 Gestión de cambio

El cambio en un sistema de software consiste en el proceso de la introducción de nuevos requerimientos, la modificación de los ya previamente planteados e implementados, o en el cambio de ambiente de operación del sistema [8]. Mientras surjan cambios en los requerimientos así como en el entorno del sistema, la evolución se vuelve un proceso inevitable. Así pues, una manera de manejar los cambios en sistema es mediante la implementación de un proceso de gestión de cambio cual consiste en la identificación del cambio solicitado, el análisis del impacto del cambio, y la estimación del costo y el tiempo para la implementación de dicho cambio [45]. Es así que a partir de la solicitud de cambio en requerimientos, es esencial la administración estos, en donde las tareas desarrolladas en este proceso consisten en [64]:

- *Análisis del nuevo problema y especificaciones del cambio*. En esta etapa se evalúa la propuesta de cambio, así como el problema actual que se necesita abordar. Se analiza si es viable para la modificación de sus requerimientos y posterior implementación.
- *Análisis del cambio requerido*. Se realiza un análisis para tomar la decisión de si se procede o no al cambio de requerimientos.
- *Implementación del cambio*. Se presentan los cambios realizados en la implementación del sistema y en su documentación correspondiente.

## 2.5 Definiciones de Metamorfosis

Para este trabajo es importante analizar la definición que se le ha atribuido al término de *metamorfosis* para establecer la teoría que antecede o inspira al concepto que se propone más adelante. Etimológicamente procede del latín y a su vez del griego en donde *meta* significa *más allá* o *después de*, *morph* significa *forma o estructura*, y *-osis* indica *acciones, condiciones o estados*. El concepto surge en un contexto biológico, en donde se describe como el desarrollo de organismos a través de una serie de estados diferenciados entre sí, los cuales están asociados a funciones específicas [38]. De igual manera, la metamorfosis implica que la forma y estructura básicas de un organismo surgen como resultado de una secuencia de ajustes en el desarrollo [10].

## 2.6 Adaptación y Autoadaptación

Se denomina adaptación a la capacidad de un sistema de ajustar automáticamente su estructura o funcionalidad para satisfacer las diversas necesidades de individuos o grupos de usuarios [9].

Por otro lado, la autoadaptación consiste en la capacidad de un sistema de ajustar el comportamiento durante su ejecución en respuesta a la percepción tanto del entorno como del propio sistema en donde la principal causa de adaptación es el *cambio*, en donde es considerado su contexto, ambiente y actores que se involucran en él [22]. Del mismo modo, para que se produzca un proceso de autoadaptación se considera la fuente, tipo, frecuencia y anticipación del cambio.

No obstante, es posible establecer una comparación entre la adaptación y la autoadaptación, en donde la adaptación implica un ajuste *automático* para satisfacer distintas necesidades, y por su parte la autoadaptación es implementada mediante un nivel de *autonomía*, el cual señala el grado de intervención externa. Esta autonomía va en un rango desde autónoma hasta asistida, en donde en la autónoma no hay una referencia externa de cómo un sistema se adapta, mientras que la asistida contiene una participación de un sistema externo (software/humano) [22].

## 2.7 Cómputo Autonómico

El cómputo autonómico surge inspirado en el sistema nervioso autónomo humano, que regula funciones automáticas como la respiración, el control pupilar, el ritmo cardíaco y la presión arterial, entre otras. Este campo del cómputo autonómico fue

inicialmente explorado en [41], los sistemas de cómputo autonómico (ACS) son capaces de operar de manera independiente, adaptándose a diversas condiciones según sus necesidades previstas. Asimismo, se sostiene que para que un sistema se pueda considerar autonómico debe poseer ocho al menos ocho elementos claves o características:

1. *Conocimiento de sí mismo.*
2. *Capacidad de configurarse y reconfigurarse en condiciones variables e impredecibles.*
3. *Búsqueda de optimización de su funcionamiento.*
4. *Capacidad de recuperación frente a eventos que puedan causar fallos.*
5. *Funcionalidad de autoprotección.*
6. *Conocimiento del entorno y capacidad de actuar acorde al contexto.*
7. *No podrá existir en un ambiente hermético.*
8. *Anticipar y gestionar de forma eficiente los recursos necesarios, sin mostrar su complejidad.*

Posteriormente, se establece en [46] que un sistema autonómico debe de contar con cuatro propiedades básicas:

1. ***Autoconfiguración*** (“self-configuration”). El sistema mantiene una configuración automatizada de sus componentes.
2. ***Autosanación*** (“self-healing”). El sistema es capaz de diagnosticar y reparar fallas que se pudiesen presentar tanto a nivel software como hardware.
3. ***Autooptimización*** (“self-optimization”). Se busca continuamente oportunidades para mejorar el desempeño y la eficiencia del sistema y sus componentes.
4. ***Autoprotección*** (“self-protection”). El sistema es capaz de defenderse contra ataques maliciosos.

# Capítulo 3

## Estado del Arte

En esta sección se presentan y analizan las definiciones previas en la literatura sobre los sistemas de autoingeniería. Además, se analizan trabajos previos que abordan la gestión del cambio en los sistemas, así como trabajos que utilicen el modelado basado en agentes para abordar diversas preguntas de investigación asociadas al manejo de cambios durante la evolución de software.

### 3.1 Conceptualización de sistemas de autoingeniería

En la literatura existente se han identificado distintas definiciones de los sistemas de autoingeniería:

- En el año 2020, se plantea en [62] que “*Un sistema es de autoingeniería cuando registra y responde a una pérdida de funcionalidad o capacidad operativa, y automáticamente toma medidas para regresar la funcionalidad*”. Algunos de los métodos clave de los sistemas de autoingeniería son: autosanación, autoreparación, autoadaptación, autoreconfiguración, entre otros.
- Posteriormente en el año 2021, en [13] se definió la autoingeniería (“self-engineering”) como: “*Capacidad diseñada e integrada a un sistema para identificar de forma independiente cualquier pérdida o posible pérdida de funcionalidad, y luego restaurar automáticamente la funcionalidad total o parcial para mantener su disponibilidad y mejorar la resiliencia del sistema*.”.

En adición a las definiciones presentadas previamente los autores en [16, 12, 14, 15] enfatizan aspectos relevantes que deben observar los sistemas de autoingeniería,

tales como, la identificación, preservación y la recuperación (en caso de pérdida) de la funcionalidad; todo ello, sin intervención humana.

Las definiciones que se le han atribuido a los sistemas de autoingeniería, si bien tienen un mismo enfoque y la diferencia entre estas han sido simplemente variaciones en especificaciones, éstas se ven limitadas en relación a las actividades fundamentales de un proceso de ingeniería de sistemas. Es decir, las definiciones planteadas se centran únicamente en las actividades o etapas de operación, mantenimiento y evolución del sistema, además de no ir más allá del propio ciclo de vida del sistema. Esto se ilustra gráficamente en la figura 3.1.

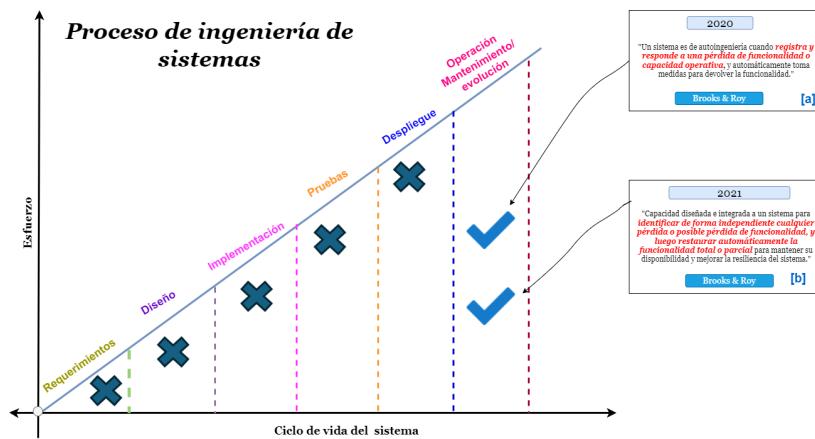


Figure 3.1: Comparativa de definiciones previas ubicadas en el ciclo de vida de un sistema (a: [62], b: [13].)

## 3.2 Arquitecturas de referencia

En esta sección se muestran las arquitecturas de referencia revisadas con un enfoque en la autogestión, ya que en el marco actual de los SES se establece que el sistema identifique una falla operativa y la aborde.

En [66] se propone un proceso de ingeniería ágil y centrado en el sistema para el desarrollo de sistemas de software autónomos mediante una arquitectura para los sistemas autogestionables centrándose en la ingeniería de software. Dicha arquitectura utiliza medidores que permiten el interactuar con el sistema en ejecución y recopilar datos para traducirlos en métricas adecuadas para el ajuste del rendimiento del sistema y/o la adaptación. El diseño arquitectónico toma como guía los principios del modelo del sistema viable así como el uso de cómputo autonómico. El

*modelo del sistema viable* [7] se centra en la idea de que todas las organizaciones son sistemas y comparten principios y características, las cuales determinan su capacidad para sobrevivir y adaptarse en entornos cambiantes. Dicho modelo propone cinco subsistemas *implementación, coordinación, control, inteligencia y política*. Si bien este trabajo ilustra como los modelos planteados facilitan la ingeniería de sistemas autónomos considerando la autogestión, no hay un desarrollo de las actividades fundamentales del proceso de ingeniería, centrando su gestión únicamente en la operación del sistema en donde se busca responder y adaptarse continuamente a estímulos inesperados.

En [47] plantea una visión de los sistemas de autogestión a nivel arquitectónico en donde sus componentes configuran automáticamente su interacción para que sean compatibles con especificaciones arquitectónicas y objetivos del sistema, minimizando así el grado de gestión. Asimismo para abordar los desafíos de la autogestión de sistemas de software se define un modelo de referencia de tres capas: control de componentes, gestión de cambios y gestión de objetivos. Se tiene un enfoque basado en los objetivos del sistema dejando de lado una etapa de requerimientos o consideración de cambio en los requerimientos.

Por otro lado, en [46] se presenta la arquitectura MAPE-K ("Monitoring, Analyzing, Planning, Execution y Knowledge"), siendo una representación conceptual de los componentes de un elemento autónomo. En donde el componente monitor observa y recopila los datos del sistema, así como del entorno. Por su parte, el componente análisis estudia los datos para la detección de situaciones que requieran una adaptación. El componente plan desarrolla estrategias de adaptación para responder a los cambios suscitados. Ejecución, realiza las estrategias de adaptación planificadas. Finalmente, componente conocimiento almacena el estado del sistema, así como el conocimiento compartido por los otros componentes. En esta arquitectura se consideran los parámetros del ambiente y del sistema, no obstante no gestiona su ciclo de vida ni un proceso de ingeniería de sistemas.

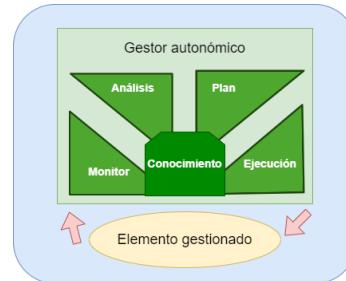


Figure 3.2: Bucle de control MAPE-K [46]

### **3.3 Modelado basado en agentes para ingeniería de software**

El estudio de la evolución de sistemas en la ingeniería de software se ha abordado utilizando el enfoque del modelado basado en agentes.

El Modelado Basado en Agentes (ABM), es una herramienta empleada para simular y estudiar fenómenos complejos emergentes.

En el trabajo de [39] se plantea un modelado basado en agentes para la evolución del software el cual permite el describir el comportamiento de los desarrolladores en detalle, pareciendo natural entonces modelar los procesos de software desde el punto de vista del comportamiento humano. De igual manera, en este trabajo se consideran los cambios realizados en el software, así como las causas de ello y el impacto generado. Asimismo, se hace la propuesta de distintos modelos y algoritmos, los cuales atienden la gestión del sistema considerando su complejidad, requerimientos y el esfuerzo realizado durante su evolución. Por ejemplo, ofrece el modelo en donde se lleva a cabo la simulación de varios aspectos de la evolución del software considerando las dependencias entre entidades de software como redes. No obstante, este trabajo no aborda todas actividades del proceso de desarrollo en ingeniería de software, solamente se enfoca en la etapa de evolución. Además, no se ofrece una arquitectura en donde se realice una gestión del ciclo de vida del sistema.

En [40] se presenta un modelo de simulación basado en agentes para la evolución del software a través de aprendizaje estadístico y la minería de datos para los parámetros de la simulación. En su modelo se considera el crecimiento del sistema, la densidad de errores y la actividad de los desarrolladores. En [1] se define un modelo de simulación para evaluar cuantitativamente aspectos del software durante las fases de mantenimiento para reflejar el proceso de evolución del software en el mundo real.

Actualmente las definiciones de los SES dejan fuera aspectos de ingeniería de sistemas y se concentran en la operación y mantenimiento, es entonces que agregar a los SES un enfoque sistemático como el que brinda la ingeniería de software habilitaría sistemas que aborden la evolución con mayor autonomía.

### **3.4 Comparación de trabajos**

Una nueva visión para los SES enfocada en la evolución y gestión de su ciclo de vida usando el enfoque de la ingeniería requiere comparar posibles características que se puedan presentar en estos sistemas.

	Brooks&Roy (2020) [62]	Brooks&Roy (2021) [13]
Autoingeniería	X	X
Sin intervención humana	X	X
Ingeniería de sistemas		
Ciclo de vida		
Cambio en requerimientos		

Table 3.1: Análisis comparativo de definiciones previas como referencia para propuesta

	Kephart (2001) [46]	Taleb (2005) [66])	Kramer (2007) [47]	Honsel (2015) [40]	Ali (2018) [1]	Honsel (2019) [39]
Autoingeniería						
Sin intervención humana	X	X	X			
Propiedades autonómicas	X	X	X			
Ingeniería de software				X	X	X
Evolución del software				X	X	X
Cambio en requerimientos		X	X			
Gestión del cambio		X	X			
Modelo arquitectónico	X	X	X			
ABM				X	X	X

Table 3.2: Análisis comparativo de trabajos seleccionados como referencia para propuesta

Se realiza un análisis con las distintas definiciones teóricas sobre los sistemas de autoingeniería tabla ilustrado en la tabla 3.1, se señala cuales son los aspectos en los que las definiciones actuales se enfocan, tales como el concepto de autoingeniería, si es realizada sin intervención humana, las actividades de la ingeniería de sistemas, así como la consideración del ciclo de vida del sistema al igual que el cambio en sus requerimientos.

De igual manera, se realiza un análisis en donde se consideran los trabajos ya mencionados que funcionan como referencia para el desarrollo de nuestra propuesta para la parte arquitectónica para la gestión del ciclo de vida, en la tabla 3.2 se señala qué aspectos sí se abordan en su investigación, tales como un enfoque de autoingeniería así como su funcionalidad sin intervención humana y propiedades autonómicas. Se evalúa si se realiza una ingeniería de software, su evolución, el cambio de requerimientos y un proceso de gestión de cambios. También se analiza si se emplea un modelo arquitectónico así como un modelado basado en agentes.

# Capítulo 4

## Propuesta

### 4.1 Metodología

El proceso metodológico de este trabajo se estructura en tres bloques fundamentales (figura [4.1]); siendo el primero una revisión de la literatura, en el que se ha realizado una selección de la literatura más relevante sobre terminología de sistemas de autoingeniería mediante una revisión sistemática basada en la metodología PRISMA[59], con la cual se tiene como objetivo el extraer los conceptos clave para el desarrollo de un nuevo marco conceptual. En el segundo bloque se plantean la definición y conceptualización del marco conceptual de los sistemas de autoingeniería, con el objetivo de definir e iterar en la teoría propuesta. Por último, en el bloque de análisis y validación se involucra una prueba de concepto mediante la experimentación con un caso estudio específico así como el análisis en escenarios teóricos con el objetivo de validar la teoría propuesta.

#### 4.1.1 Revisión de la literatura

El núcleo de la metodología empleada en este trabajo es una *revisión sistemática de la literatura*, en la que se establece un proceso estructurado para llevar a cabo una revisión exhaustiva de la literatura con el objetivo de reconocer la teoría existente sobre los sistemas de autoingeniería y poder establecer un nuevo marco conceptual a partir de lo ya planteado en la literatura así como de aquello que fue abordado. Es importante destacar que previo a la realización de la revisión sistemática fue importante la obtención de conocimiento en el campo de los sistemas de autoingeniería y tópicos relacionados para poder comprender y delimitar el alcance de la revisión. Las actividades realizadas para el proceso de la revisión sistemática están representadas

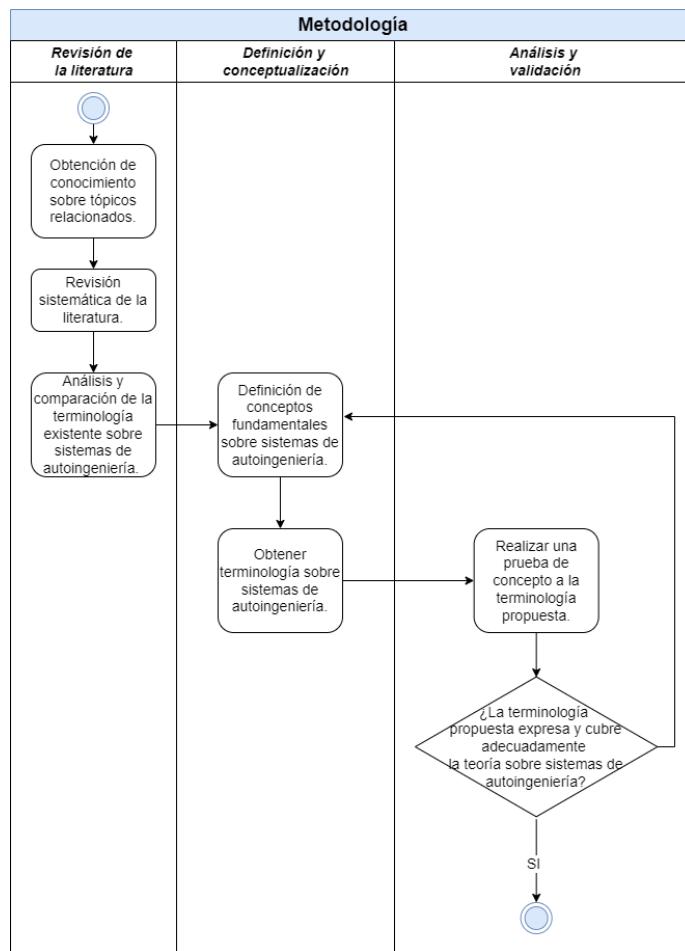


Figure 4.1: Metodología

en la figura 4.2, de igual manera, los detalles del proceso se encuentran especificados a continuación.

- 1. Planificación:** El propósito de esta etapa es diseñar un protocolo de revisión sistemática. En dicho protocolo se establece el objetivo de la revisión, la pregunta de investigación, los criterios de elegibilidad, la selección de la base de datos y la metodología de la revisión. A continuación, se muestra el contenido de dicho protocolo para nuestra revisión sistemática de la literatura sobre sistemas de autoingeniería.

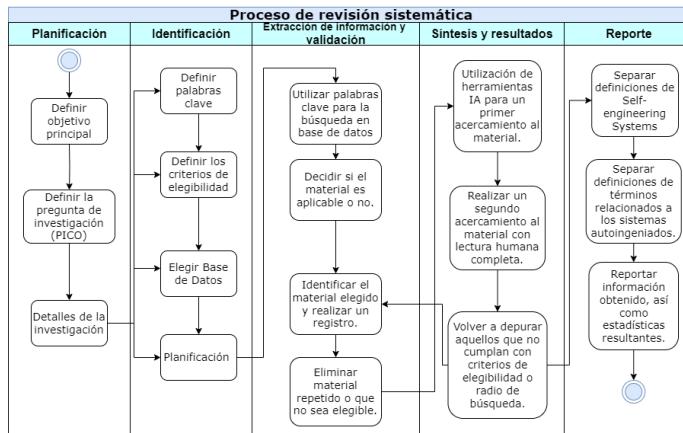


Figure 4.2: Proceso de realización de revisión sistemática

- **Objetivo.** Para determinar el alcance y el enfoque de la investigación bibliográfica sistemática, es fundamental establecer el objetivo principal de la misma. De este modo, podremos analizar los puntos fuertes y débiles de la investigación y los resultados esperados.

**Objetivo: Realizar una búsqueda exhaustiva sobre el contenido existente en la literatura sobre sistemas de autoingeniería, su concepto y el contexto al que se aplican, y validar cómo se ha abordado respecto a la ingeniería de sistemas.**

- **Pregunta de investigación.** Con el fin de validar la necesidad de hacer una revisión sistemática de la literatura es necesario responder a una pregunta de investigación.

El método utilizado para definir la pregunta de investigación adecuada es la técnica PICO [58], una guía que describe los pasos para determinar las principales variables que deben ser analizarse. Este método se utiliza habitualmente en el campo de la medicina; sin embargo, en este trabajo se adapta para su uso y aplicación en los campos de la informática y la ingeniería. Las variables de la técnica se expresan como **P** para paciente, problema de investigación o población, el cual es el objeto de la investigación; **I** para intervención o innovación, el cual es el enfoque o técnica que se va a implementar en la investigación; **C** para comparación o contexto, siendo el ámbito y alcance que se va a diseñar en la investi-

gación; y por último, **O** para los resultados esperados que debe arrojar la investigación.

- **P**: *Sistemas de Autoingeniería*
- **I**: *Realizar un marco conceptual apropiado para los sistemas de autoingeniería.*
- **C**: *El contexto de la ingeniería de sistemas, así como las ciencias de la computación.*
- **O**: *Terminologías e investigaciones previas propuestas por otros autores.*

Tras haber identificado las variables previas, disponemos de todos los elementos necesarios para formular nuestra pregunta de investigación y establecer un propósito claro que guiará todo el proceso:

**¿Qué información hay disponible en la literatura actual sobre la terminología y el concepto de sistemas de autoingeniería?**

- *Palabras clave.* El primer paso con en el que se inició la investigación fue profundizar en el tema de la autoingeniería. Esto nos permitió descubrir cuáles son los principales términos y definiciones, los autores principales, los trabajos e investigaciones previas realizados sobre el tema, así como conocimiento relacionado, entre otros parámetros. Este conocimiento relacionado, definió las palabras clave de la investigación y, por tanto, la búsqueda de la revisión bibliográfica sistemática.

**Cómputo autónomo, punto de quiebre, sistemas complejos, métricas de evolución, capacidad de evolución, resiliencia, ciclo de vida del software, autoadaptación, autoingeniería, evolución del software, mantenimiento del software.**

- *Criterios de inclusión.* Consisten en características que debe reunir el material seleccionado de la bibliografía consultada. Si un trabajo no reúne todas esas características, pero al menos coincide con una, entonces ese trabajo es apto para ahondar en su revisión.

- Artículos donde se conceptualice la autoingeniería.
- Artículos donde se describan los términos auto-\*.
- Artículos que aborden el cómputo autonómico y la autoadaptación en los sistemas.
- Artículos en los que se plantean métricas de evolución de sistemas.
- Artículos que abordan el ciclo de vida de los sistemas.
- Artículos que incluyen al menos una palabra clave propuesta y que exista relación con la ingeniería de sistemas.
- Artículos que muestren casos de estudio referentes a la autoingeniería.
- *\*\*\*Todos estos artículos deben estar en el ámbito de ciencias de la computación e ingeniería. No obstante, podría ser evaluado el valor de la información en algún artículo si que no perteneciera a las ámbitos ya definidos.*

- *Criterios de exclusión.* Todas aquellas características que no nos interesa tener en el material elegido para la revisión sistemática; esto puede deberse a que esa característica no añade ningún valor al enfoque de nuestra investigación o simplemente a que esa característica está fuera de nuestro alcance.
  - Artículos en donde no aborde un enfoque hacia la Ingeniería.
  - Artículos que no expongan información referente a la autoingeniería.
  - *\*\*\*Si un artículo coincide con uno de estos criterios, pero contiene información relevante para la investigación en otros campos o contextos, debe ser evaluado y decidir si se excluye o no.*
- *Base de datos:* una base de datos es una herramienta esencial en donde se encuentran registrados virtualmente distintos archivos bibliográficos. Para cubrir la mayor parte de estos archivos es necesario elegir bases de datos confiables y de amplio contenido. Según [67], la base de datos con mejores resultados en búsquedas es Scopus con ACM Digital Library, para un mejor resultado se recomienda el mezclar el uso de estas dos bases de datos. Por otra parte, basado en [19], donde se analizan diferentes bases

de datos como Google Scholar, Scopus, Web of Science, DBLP, y INSPEC; Se concluye que Scopus y DBLP tienen la mejor cobertura de artículos de informática. Teniendo en cuenta que la ingeniería y la computación son los campos donde se realiza la investigación, la decisión será el utilizar distintas bases de datos para complementarlas entre sí.

**Scopus, Google Scholar, IEEE Xplore, Science Direct, y ACM Digital Library.**

- *Área de enfoque:*

**Ingeniería de sistemas y las ciencias de la computación.** De igual manera, podría incluirse otro estudio que agregue información valiosa a la investigación, y sería evaluado con los criterios establecidos y los objetivos antes mencionados.

- *Tipo de artículo:*

**Los tipos de artículos que se incluyen en la investigación son artículos relevantes de revistas y de conferencias.**

- *Fecha:*

**La fecha no estará delimitada, puesto que será el parámetro indicador para rastrear los inicios de la investigación sobre sistemas de autoingeniería.**

**Identificación:** en esta etapa se realiza la búsqueda bibliográfica mediante las bases de datos seleccionadas y se comparan con los criterios de elegibilidad, de igual manera se comienzan a seleccionar artículos. A continuación, con los parámetros establecidos, se establece una metodología para la búsqueda y análisis del material:

1. Se realiza una primera búsqueda con las palabras clave en la base de datos decidida, y se ajustan los filtros a los parámetros establecidos.
2. Se compararon los artículos resultantes con los criterios de inclusión y exclusión.
3. Los artículos aprobados se añaden a nuestra colección de material para investigación y los que no coinciden con los criterios establecidos se descartan.

4. Los artículos que se incluyeron en la colección fueron evaluados con un primer filtro con el uso de herramientas tecnológicas, en donde se realizan análisis exploratorios para decidir si el artículo aporta algo como referencia para nuestra investigación. Los artículos resultantes se clasifican según el orden de importancia e impacto.
5. Después de establecer el primer filtro ya mencionado, se realiza un segundo acercamiento a partir de la clasificación anterior, los artículos se analizan completamente sin uso de herramientas externas, y de acuerdo con los criterios elegibles, la pregunta de investigación y los objetivos, se realiza una segunda depuración.
6. Posteriormente, los artículos resultantes a partir de estos dos filtros se clasifican de acuerdo con el orden de importancia e impacto.
7. Finalmente, con estos dos filtros, se tiene como resultado una selección final del material. En esta selección se buscan definiciones de autoingeniería y términos relacionados para analizar su relación y desarrollo en el tema.

**Evaluación:** se cotejan los artículos seleccionados con el objetivo, pregunta de investigación y criterios de elegibilidad. En esta etapa de realiza la aceptación y depuración de artículos acorde a los criterios de exclusión e inclusión, para finalmente obtener como resultante 55 artículos relevantes con posibles aportaciones a nuestra revisión sistemática. El proceso de depuración se muestra en la figura 4.3.

**Síntesis:** se realiza una presentación de resultados y conclusiones de los artículos seleccionados. Se expone de manera cuantitativa los artículos considerados como parte de la revisión sistemática, así como un análisis temporal (fig. 4.4), de igual manera se genera un análisis de las palabras clave con mayor resultados de búsqueda de los artículos seleccionados (fig. 4.5).

**Difusión y publicación:** el procedimiento de la revisión sistemática desde su protocolo hasta su ejecución se representa en un informe estructurado el cual es publicado como parte de los resultados.

#### 4.1.2 Definición y conceptualización

El bloque anterior correspondiente a la revisión de la literatura, se abordó la realización de una revisión sistemática la cual funcionó como referencia de la teoría existente sobre sistemas de autoingeniería.

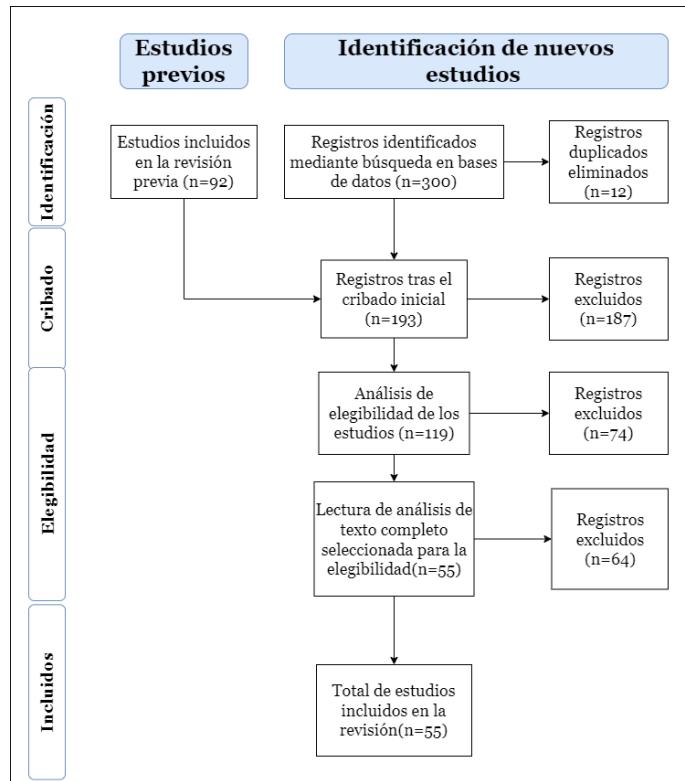


Figure 4.3: Proceso de selección de artículos

Este siguiente bloque corresponde a la definición y propuesta teórica sobre los sistemas de autoingeniería, con el objetivo de cubrir lo que la literatura no ha cubierto y así como analizar lo que otras definiciones proponen. La propuesta teórica realizada se presenta en la sección (sección:4.2) .

#### 4.1.3 Análisis y validación

Una vez realizados los primeros dos bloques de la metodología, se implementa el tercer bloque el cual consta de una validación y experimentación de la teoría propuesta en casos de estudio específicos los cuales logren expresar el cumplimiento de la hipótesis planteada en esta investigación. En este caso de estudio se presentan análisis teóricos, así como un caso práctico el cual consta de un modelado basado en agentes en donde se presenta y valida la teoría propuesta en el bloque de definición y conceptualización. El objetivo principal de este bloque es evaluar la precisión y alcance de la teoría propuesta a través de la experimentación, es decir, si en la validación se encontrarán



Figure 4.4: Representación temporal de la selección de artículos

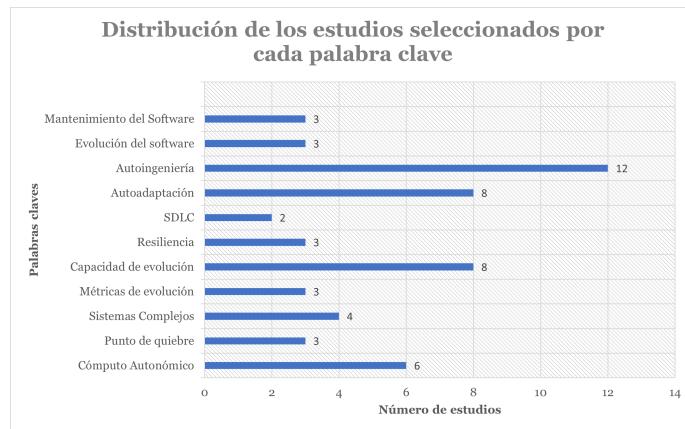


Figure 4.5: Representación de selección de artículos acorde a la palabra clave

deficiencias y/o inconsistencia entonces se ha de regresar al segundo bloque a pulir lo planteado para continuar con el flujo de trabajo y validar lo nuevo que se ha propuesto. La experimentación y validación es presentada en la sección: 5.1.1.2.

## 4.2 Propuesta marco conceptual y arquitectura: Sistemas de autoingeniería.

### 4.2.1 Definición conceptual y formal

Como parte de la propuesta de este trabajo, se presenta un marco conceptual que define la terminología asociada a los sistemas de autoingeniería. Este marco incluye términos clave como la definición de sistemas de autoingeniería, sus etapas de autoadaptación y metamorfosis, especificaciones para la gestión de cambio, su evolución y una definición del punto de quiebre. De igual manera, se propone un modelado formal de los sistemas de autoingeniería.

#### 4.2.1.1 Definición

**Sistemas de Autoingeniería (“Self-engineering systems - SES”):** “*Un sistema de autoingeniería es aquel que es capaz de gestionar y llevar a cabo el proceso de ingeniería de sistemas (requerimientos, diseño, implementación, pruebas, despliegue, operación, mantenimiento, evolución y desecho) sin intervención humana más allá de su ciclo de vida, permitiendo la transformación del sistema original en un nuevo sistema a partir de su desecho implicando o no su reuso.*”

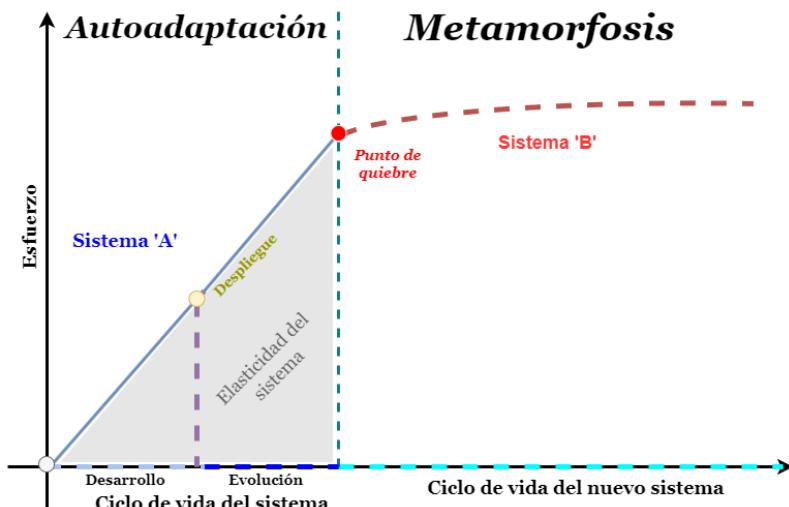


Figure 4.6: Ciclo de vida de un sistema de autoingeniería

#### 4.2.1.2 Caracterización

Para que un sistema pueda ser considerado de autoingeniería debe de cumplir con los siguientes características:

- Realizar el proceso de ingeniería de sistemas (requerimientos, diseño, implementación, pruebas, despliegue, mantenimiento/evolución y disposición) sin intervención humana.
- Gestionar los cambios en los requerimientos del sistema.
- Su ejecución deberá constar de dos fases: autoadaptación y metamorfosis del sistema.
- Todas las funcionalidades deberán realizarse sin intervención humana.

A partir de esto, es posible realizar una comparativa de sistemas con aquellos que pudiesen parecer similares, por su funcionalidad, con la definición de autoingeniería redefinida.

Por ejemplo, los sistemas de cómputo autonómico constan de cuatro propiedades: autoconfiguración, autosanación, autooptimización y autoprotección. Estos sistemas se caracterizan por operar sin intervención externa. El diseño de estos sistemas les permite adaptarse a distintas necesidades existiendo un aprendizaje de por medio. No obstante, se dejan de lado las demás actividades o etapas del proceso de ingeniería de software.

La IA generativa es un tipo de inteligencia artificial que puede crear contenido nuevo, incluso se plantea la aplicación en la generación de software [63]. Si bien esta tecnología es capaz de generar contenido de software, no considera las actividades del proceso de ingeniería de software, pudiendo generar imprecisiones en el desarrollo del software.

Por otra parte, se encuentran los sistemas multifuncionales, los cuales son sistemas que ofrecen una variedad de funciones para distintos fines y casos de uso [17]. El cambio de funcionalidad para un caso de uso determinado no implica un nuevo sistema con un nuevo ciclo de vida, sino un mismo sistema ejecutando una funcionalidad específica distinta.

Asimismo, los sistemas autoadaptativos consideran el *cambio* como la principal causa de la adaptación, y su manera de gestionarlo es ajustando su comportamiento durante su ejecución en respuesta a la percepción tanto del entorno como del propio sistema [21]. La autoadaptación se enfoca principalmente en cambios de parámetros, entorno del sistema y necesidades del usuario.

De igual manera, se consideran los sistemas autogestionables, los cuales se enfocan a la gestión de cambios basada en objetivos. Sus componentes configuran automáticamente su interacción para que sean compatibles con especificaciones arquitectónicas y objetivos del sistema, por otro lado la autoingeniería se realiza en función de los requerimientos.

En la figura 4.7 se representa de manera gráfica el análisis de esta comparativa.

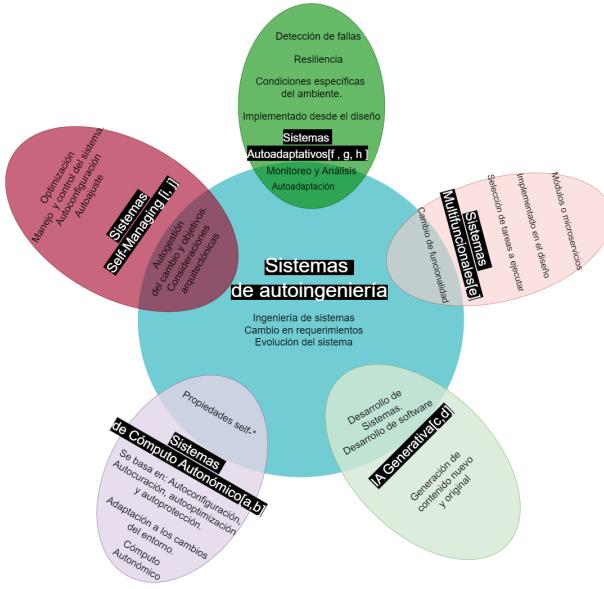


Figure 4.7: Comparativa de sistemas con similitudes a los sistemas de autoingeniería ubicados en el ciclo de vida de un sistema (a: [41], b:[65], c: [63], d: [31], e: [17], f: [23], g: [33], h: [4], i: [66], j: [47], k: [68]).

#### 4.2.1.3 Especificaciones teóricas

- **Requerimientos funcionales.** Los requerimientos funcionales de un sistema corresponden a una sentencia en donde se define lo que el sistema debe realizar. Las especificaciones contenidas en estos requerimientos, acorde a la propuesta de este marco conceptual, corresponden a los componentes: *funcionalidad* y *configuración*. En donde la parte funcional define la acción que deberá realizar el sistema, y la configuración define los parámetros de operación para la ejecución de la funcionalidad (ver figura 4.11).
- **Gestión de cambio.** El proceso de gestión de cambio en los sistemas de

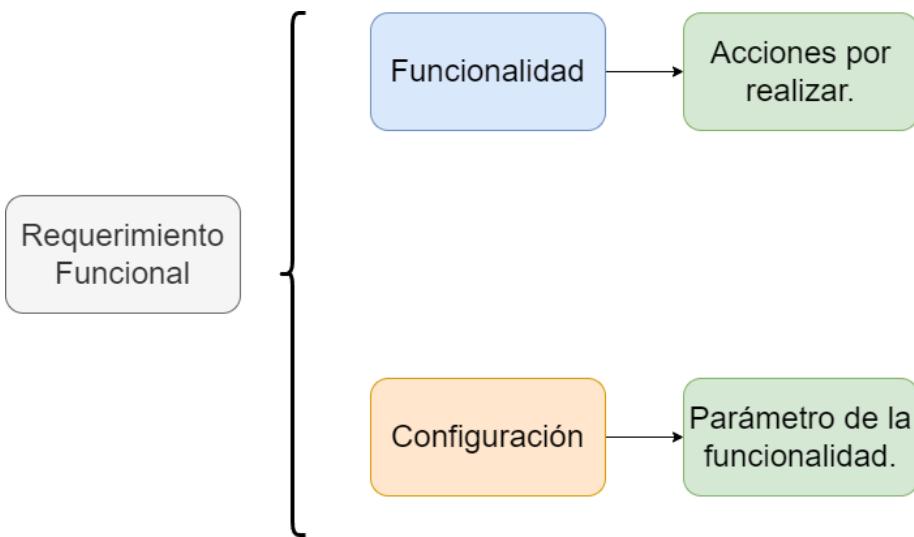


Figure 4.8: Componentes de un requerimiento funcional.

autoingeniería, consiste en las actividades realizadas para atender los cambios en los requerimientos funcionales del sistema durante su etapa de evolución.

Un sistema es desarrollado a partir de la necesidad de solucionar un determinado problema. Los cambios en requerimientos, los cuales normalmente vienen de las partes interesadas en el desarrollo del sistema (*“stakeholders”*) o de los usuarios del sistema, surgen a partir de la necesidad de abordar una problemática distinta.

El proceso de gestión de cambio es el encargado de atender estos cambios mediante las siguientes actividades (ver figura 4.9):

1. *Análisis del problema y especificaciones del cambio.* Surge a partir del cambio de la problemática.
2. *Análisis del cambio.* Una vez que el problema cambió, por ende los requerimientos deben cambiar, en este paso se realiza un análisis para evaluar el cambio.
3. *Implementación del cambio.* Este paso depende directamente del análisis del cambio, en donde se determinará la viabilidad del cambio, así como el impacto de la implementación el cuál detonará en una evolución del sistema o bien en un nuevo sistema.

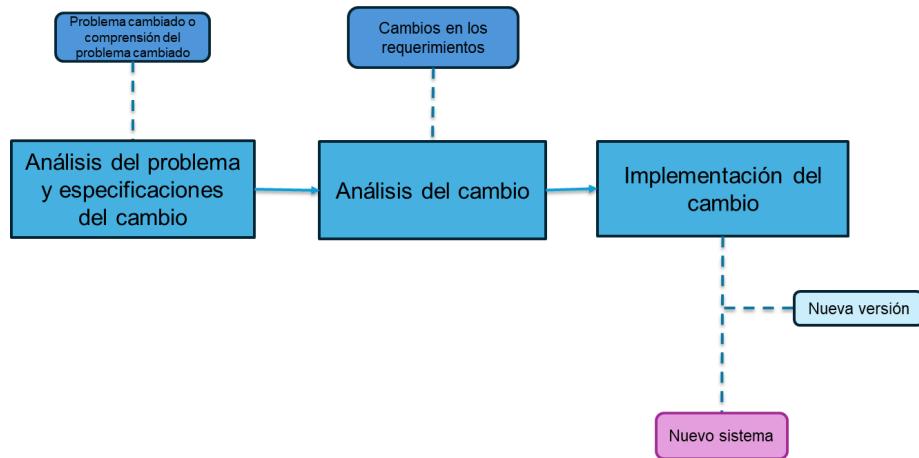


Figure 4.9: Proceso de gestión de cambio.

#### 4.2.1.4 Ciclo de vida

Los sistemas de autoingeniería se enfocan en la etapa de evolución de un sistema basándose en identificar cambios como requerimientos, siguiendo un proceso de ingeniería de sistemas.

La figura 4.6 corresponde a una representación gráfica del ciclo de vida de un sistema de autoingeniería. A continuación se describen cada uno de estos conceptos ilustrados:

- **Autoadaptación:** corresponde a la etapa posterior al despliegue del sistema encargada de lidiar con los **cambios en la configuración de los requerimientos**, los cuales provocarán una **evolución** en el sistema sin llegar al punto de quiebre.
- **Punto de quiebre:** punto del sistema en donde los **cambios en requerimientos** no pueden ser satisfechos por el sistema actual y necesita convertirse en uno nuevo.
- **Elasticidad del sistema:** capacidad del sistema para soportar e implementar cambios de requerimientos sin tener que convertirse en un nuevo sistema.
- **Metamorfosis:** transición de un **sistema original a uno nuevo** a través de un nuevo proceso de ingeniería de sistemas. La metamorfosis está determinada por el **punto de quiebre** del sistema original, a partir de los **cambios en la funcionalidad** y la evaluación de la elasticidad del sistema.

La metamorfosis puede ser de dos tipos:

- **Metamorfosis por transformación directa:** en donde un sistema A (original) pasa a ser un sistema B (nuevo), en donde su funcionamiento que satisface a los requerimientos del sistema ha cambiado.



Figure 4.10: Metamorfosis por transformación directa.

- **Metamorfosis por integración:** el funcionamiento que satisface a requerimientos del sistema se convierte en una parte de la funcionalidad total del sistema B, es decir, se implementan funcionalidades en el sistema provenientes de dos o más sistemas para formar un nuevo sistema.

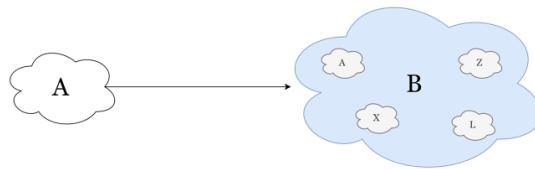


Figure 4.11: Metamorfosis por integración

- **Condiciones de metamorfosis:** evalúan la evolución del sistema a partir de su elasticidad y el punto de quiebre para determinar la necesidad de llevar acabo la metamorfosis en el sistema.
  - En un sistema SES se detona metamorfosis cuando hay cambios en las funcionalidades de los requerimientos las cuales no pueden ser satisfechas por el sistema original.
  - La metamorfosis se detona cuando los nuevos requerimientos no puedan ser satisfechos por la arquitectura actual y se necesite el diseño de una nueva.

#### 4.2.1.5 Funcionamiento de los sistemas de autoingeniería

- El funcionamiento de un sistema de autoingeniería enfocado a su implementación se ve ilustrado en la figura 4.12. En esta representación se puede observar que los SES se encuentran bajo un monitoreo constante para la posible ocurrencia de cambios en requerimientos, en donde se analiza cuál será la consecuencia. Es decir, se determinará si habrá una autoadaptación o metamorfosis.

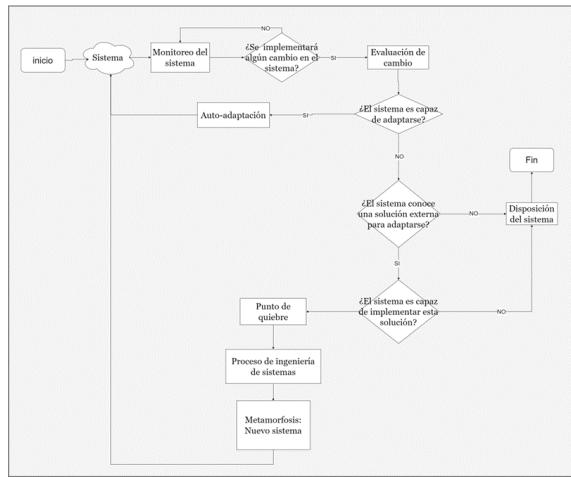


Figure 4.12: Diagrama de flujo de los sistemas de autoingeniería

- De igual manera, el proceso de gestión de cambio (mencionado previamente en las especificaciones teóricas), implementado por los sistemas de autoingeniería se ilustra en la figura 4.13.
- Cuando el sistema de autoingeniería ha llegado a la etapa de metamorfosis, este se convierte en un nuevo sistema a partir de la realización del proceso de ingeniería de sistemas, sin intervención humana. La implementación de las actividades fundamentales de la ingeniería de sistemas queda fuera del alcance de este trabajo, sin embargo se proponen distintas tecnologías y técnicas que se podrían implementar, esta propuesta se ve ilustrada en la figura 4.14.

#### 4.2.1.6 Modelado formal

Este modelado formal se propone a partir de la teoría LTL (Lógica lineal temporal) y teoría de conjuntos para describir el comportamiento de los sistemas de autoingeniería a lo largo del tiempo, o en este caso a lo largo de su ciclo de vida.

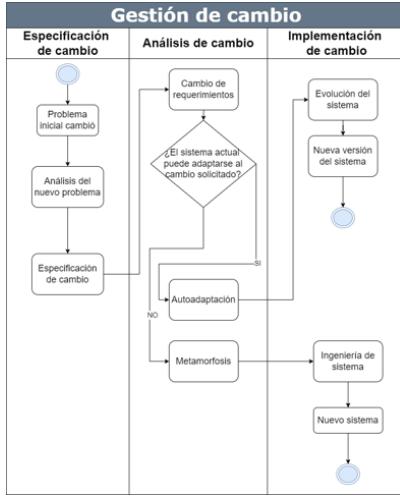


Figure 4.13: Diagrama de actividad para el proceso de gestión de cambio

Para este modelado se identifican los términos del marco conceptual como: sistemas de autoingeniería *SES*, las etapas de autoingeniería de auto adaptación *AD* y metamorfosis *M*, evolución *E* así como el punto de quiebre *Q*.

La ejecución de un sistema de autoingeniería consta de dos fases: *autoadaptación y metamorfosis*.

$$(AD \vee M) \subseteq SES \quad (4.1)$$

Asimismo, el núcleo temático de los sistemas autoingeniería es el evento de *cambio*, el cual siempre provoca que en el sistema se produzca una evolución; una vez que se aplica el cambio el sistema evoluciona.

$$\sigma_1 = \Diamond(cambio) \quad (4.2)$$

$$\sigma_2 = \Box(cambio \Rightarrow E) \quad (4.3)$$

Entonces, partiendo de este evento, es posible que el cambio implementado corresponda a un conjunto de cambios *Ci* implementados al sistema:

$$C_i = \{i \in \mathbb{N}\} \quad (4.4)$$

Los cambios que los sistemas de autoingeniería abordan son los cambios en requerimientos. Estos requerimientos se encuentran bajo un dominio de aplicación,

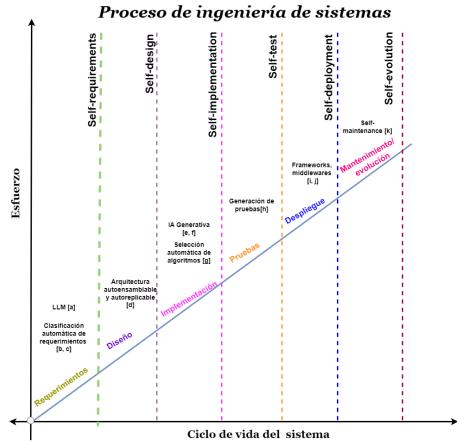


Figure 4.14: Técnicas propuestas para la implementación de las etapas de la ingeniería de sistemas (a:[28], b: [2], c: [27], d: [61], e: [63], f:[47], g:[25], h:[56], i:[18], j:[3], k:[50])

siendo éste el segmento de la realidad para el cual se desarrolla un sistema [70] y define un lenguaje específico sobre un área de interés [6]. Consecuentemente, el sistema original cuenta con  $R_j$  requerimientos determinados para el proceso de ingeniería de sistemas, estos mismos requerimientos, así como el funcionamiento del sistema, se asignan a un dominio de aplicación  $D$ .

$$R_j = \{j \in \mathbb{N} \mid R_{jn} \in D\} \quad (4.5)$$

Así como ya fue planteado en las especificaciones teóricas, los requerimientos cuentan: funcionalidad y configuración.

Entonces, se puede deducir que el cambio  $C_i$  determinado puede ser de tipo funcionalidad o configuración en el requerimiento.

$$(Funcionalidad \wedge Configuración) \subseteq R_j \quad (4.6)$$

$$(Funcionalidad_j \vee Configuración_j) \subseteq C_i \quad (4.7)$$

Como resultado del planteamiento anterior, se establece que la etapa encargada de los SES para afrontar el cambio de requerimientos en el aspecto de su configuración sea la autoadaptación  $AD$  y para el cambio en funcionalidad sea la metamorfosis  $M$ .

$$\sigma_3 = \square(C_i : Configuración_j \Rightarrow AD) \quad (4.8)$$

$$\sigma_4 = \square(C_i : Funcionalidad_j \Rightarrow M) \quad (4.9)$$

Todo el conjunto de  $i$  cambios puede implicar la etapa de evolución o bien, alguno o varios de los cambios suscitados puede provocar la metamorfosis del sistema.

$$\sigma_5 = \square \diamond (\bigwedge_{1 \leq i \leq a} (AD_i) \vee \bigvee_{1 \leq i \leq m} (M_i)) \quad (4.10)$$

*Donde a corresponde a los cambios posibles en el sistema que impliquen la autoadaptación, mientras m corresponde a los cambios que impliquen metamorfosis*

Entonces, podemos afirmar que eventualmente después de aplicar cambios en el sistema, los cuales detonan la evolución en el sistema en algún momento el sistema no soportará estos cambios y se tendrá la etapa de metamorfosis detonando en un nuevo sistema.

$$\sigma_6 = \diamond(E \Rightarrow M) \quad (4.11)$$

**Etapa de autoadaptación.** Eventualmente el sistema deberá de ejecutar la etapa de autoadaptación a partir de la aplicación de distintos cambios en los requerimientos, lo cuál resultará en el sistema original pero evolucionado con los cambios aplicados.

$$\sigma_7 = \diamond(Sistema_A \Rightarrow AD_i) \Rightarrow \bigcirc(Sistema_A \cup C_i) \quad (4.12)$$

**Etapa de Metamorfosis:** Cuando a un sistema original le sea requerido aplicar cambios los cuales este no sea capaz de soportar entonces se tendrá un nuevo sistema. La metamorfosis puede darse de dos maneras:

**Metamorfosis por transformación directa.** Se pasa de tener un sistema A a un sistema B a través de los cambios aplicados.

$$\sigma_8 = \square[((Sistema_A \cup C_i) \Rightarrow M_i)) \Rightarrow \bigcirc Sistema_B] \quad (4.13)$$

**Metamorfosis por integración (absorción).** Se pasa a tener un sistema A, el cual contenía ciertas funcionalidades, para a partir de los cambios requeridos pasar a un sistema B el cual contiene la funcionalidad de  $s$  sistemas.

$$\sigma_9 = \square \left( \bigcup_{1 \leq s \leq n} ((Sistema_A \cup C_i) \rightarrow M_i) \right) \Rightarrow \bigcirc Sistema_B \quad (4.14)$$

*Donde  $s$  corresponde a los cambios posibles en el sistema que impliquen la autoadaptación, mientras  $m$  corresponde a los cambios que impliquen metamorfosis*

**Punto de quiebre:** El punto de quiebre se manifiesta a partir de una evolución en el sistema. Es decir, eventualmente se dará el punto de quiebre a partir de la evolución.

$$\sigma_{10} = \square(E \Rightarrow \diamond Q) \quad (4.15)$$

Siempre que el punto de quiebre haya surgido, entonces se tendrá la fase de metamorfosis del sistema.

$$\sigma_{11} = \square(Q \Rightarrow \bigcirc M) \quad (4.16)$$

## 4.2.2 Propuesta arquitectónica

Se dice que la esencia de todo sistema de software es su arquitectura, la cual consiste en ser el conjunto de decisiones de diseño principales sobre un sistema. Estas decisiones consisten en aspectos estructurales como sus componentes y configuraciones, el despliegue del sistema, sus propiedades no funcionales, así como los patrones de evolución mismo, incluida su adaptación en tiempo de ejecución [55]. Asimismo, una arquitectura de software consiste en una abstracción de un sistema la cual realiza una tarea concreta mediante un grupo de componentes arquitectónicos [34].

### 4.2.2.1 Éter de un sistema de autoingeniería

Se ha hablado de qué es lo que detona la metamorfosis en un sistema y cuál es su efecto, es decir, sabemos que la metamorfosis se presenta cuando hubiese cambios en la funcionalidad de los requerimientos, pero, ¿cuál es la afectación a nivel arquitectura al tener esta transición?

Así bien, cuando se ha detonado metamorfosis en un sistema se dice que se ha realizado una transición a un nuevo sistema (sistema B) implicando nuevas funcionalidades a partir de nuevos cambios en los requerimientos, dichas funcionalidades no pueden ser soportadas por la arquitectura anterior del *sistema A*, por lo tanto se debe diseñar una nueva arquitectura para este *sistema B* (figura 4.15).

Consecuentemente, existe la interrogativa de qué tanto hay en la arquitectura del sistema B de la arquitectura del sistema A, independientemente de si hubiese implicado reuso en la arquitectura, es un hecho que estas arquitecturas comparten algo entre sí (figura 4.16).

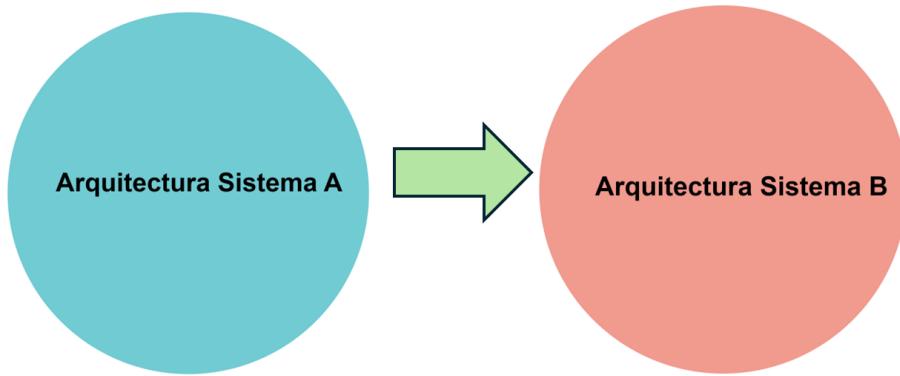


Figure 4.15: Cambio de arquitectura de un *sistema A* a un *sistema B*

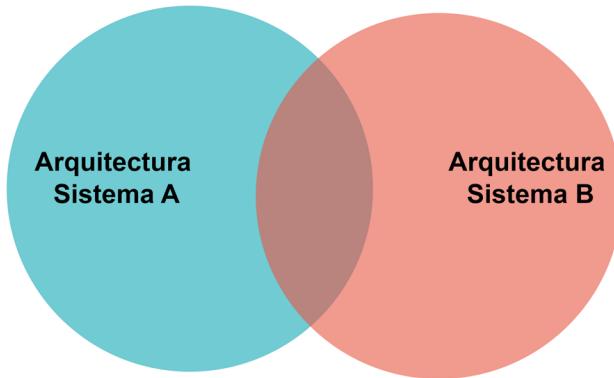


Figure 4.16: ¿Qué tanto hay en la arquitectura del nuevo sistema del sistema anterior?

Ese *algo* que comparten las arquitecturas del sistema original y el sistema nuevo se le denomina **éter**.

La palabra éter tiene un origen desde la filosofía clásica griega, si bien se ha utilizado para cuestiones existenciales y fenómenos naturales, ésta palabra hace referencia a lo más puro o esencial de algo; una sustancia pura, perfecta e inmutable, algo que corre siempre y eternamente. El éter también es conocido como la quintaesencia.

En el contexto de los sistemas de autoingeniería le llamamos éter a ese *algo* o esa *esencia* que tienen en común la arquitectura del *sistema A* y el *sistema B* (figura 4.17).

Una vez planteado que los elementos que mantienen en común la arquitectura del

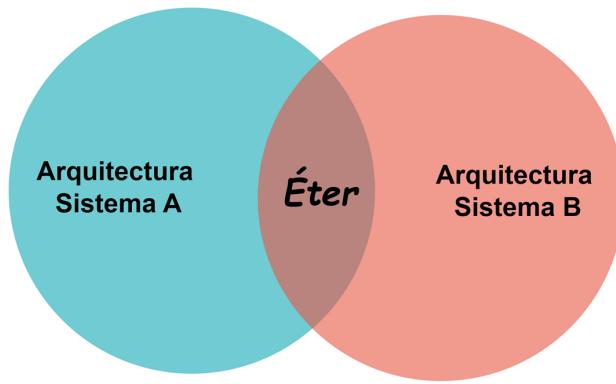


Figure 4.17: Éter de sistemas de autoingeniería

sistema A y la arquitectura del sistema B corresponden al éter, existe la interrogativa de en sí qué es el éter como tal de un sistema de autoingeniería.

Dicho esto, el éter de un sistema de autoingeniería corresponde a la arquitectura propia de un sistema de autoingeniería para la gestión de su ciclo de vida (figura 4.18). La arquitectura propia de un sistema de autoingeniería no va a cambiar conforme el paso de tiempo de ejecución del sistema, independientemente de si está en desarrollo el *sistema A* o se ha hecho metamorfosis a un *sistema B*.

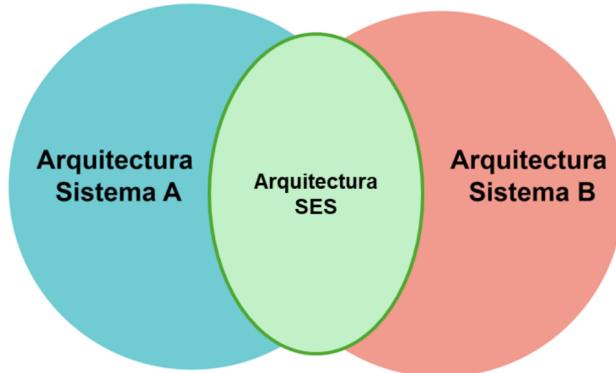


Figure 4.18: Arquitectura de un sistema de autoingeniería como el éter.

La arquitectura de un sistema de autoingeniería está presente a lo largo de todo su ciclo de vida (figura 4.19), mientras que las arquitecturas de los sistemas A y B se encuentran definidas por la etapa de diseño del proceso de ingeniería de sistemas (figura 4.20).

## Nuevo Marco Conceptual de Sistemas de Autoingeniería y una Arquitectura para la Gestión de su Ciclo de Vida

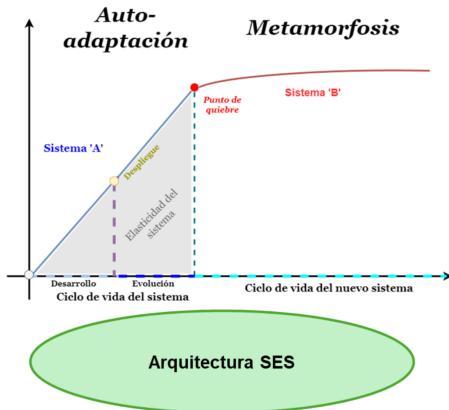


Figure 4.19: Arquitectura SES como el éter de los sistemas de autoingeniería en su ciclo de vida.

Por ende, el éter de los sistemas de ingeniería no podrá ser alterado en ninguna de las etapas de su ciclo de vida.

La arquitectura de los sistemas de autoingeniería se encuentra planteada en la sección 4.2.2.2.

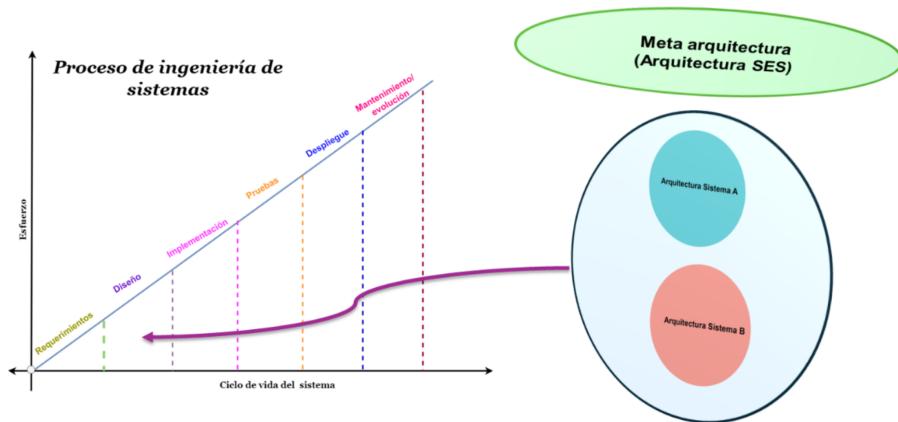


Figure 4.20: Éter de la arquitectura en el proceso de ingeniería de sistemas

#### 4.2.2.2 Propuesta de la arquitectura de los sistemas de autoingeniería

La arquitectura propuesta a continuación considera los componentes, propiedades, relaciones y funcionalidad que un sistema de autoingeniería denota para la gestión de su ciclo de vida.

Para plasmar la arquitectura de los sistemas de autoingeniería primero deben establecerse los requerimientos funcionales (tabla: 4.1) en donde se describa cuáles son los aspectos de diseño que la arquitectura debe poseer, así como los no funcionales (tabla: 4.2) los cuales consideran los aspectos que midan el comportamiento del sistema tales como el rendimiento.

Con los requerimientos planteados ahora se puede proponer la arquitectura correspondientes a los sistemas de auto ingeniería.

Considerando que se gestionará una ingeniería de sistemas implementada por el sistema mismo, entonces se toma como inspiración el bucle MAPE-K que se presentó como parte de la teoría del cómputo autónomo 2.7. En la figura 4.21 se identifica la relación de cada subsistema con lo propuesto para satisfacer los requerimientos funcionales previamente mencionados.

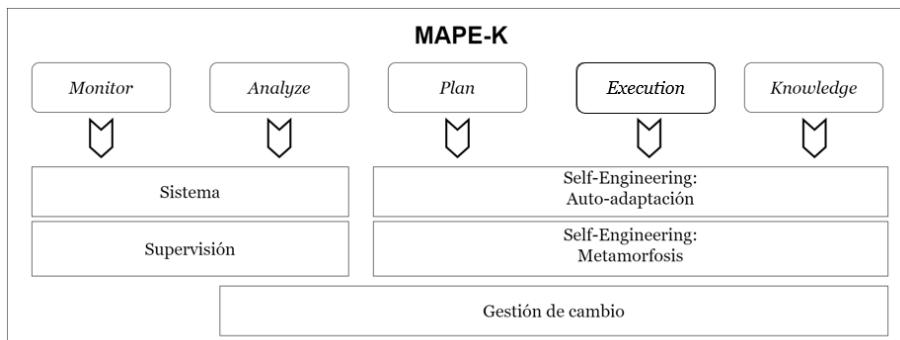


Figure 4.21: Analogía marco MAPE-K y arquitectura SES

Teniendo ya establecidos los requerimientos funcionales y no funcionales sobre los sistemas de autoingeniería y la relación con el bucle de control MAPE-K, entonces se propone la siguiente arquitectura de los sistemas de autoingeniería (figura 4.22) en donde se observan los módulos de:

- **Supervisión.** Este módulo es el encargado de recibir las solicitudes de cambio en requerimientos.
- **Gestión del cambio.** En este módulo se realiza el proceso de gestión de cambios, en donde se analizan los cambios en requerimientos a implementar.

Requerimientos funcionales	
	<b>Requerimiento funcional (RF1)</b>
<i>Nombre</i>	Supervisión de cambios
<i>Descripción</i>	El sistema deberá monitorear los cambios en los requerimientos que le sean solicitados.
<i>Entrada</i>	Cambios en requerimientos
<i>Salida</i>	Nuevo conjunto de requerimientos
	<b>Requerimiento funcional (RF2)</b>
<i>Nombre</i>	Gestión de cambio
<i>Descripción</i>	El sistema deberá de emplear un método para lidiar con los cambios solicitados en los requerimientos.
<i>Entrada</i>	Conjunto de cambios en requerimientos.
<i>Salida</i>	Método de gestión del cambio.
	<b>Requerimiento funcional (RF3)</b>
<i>Nombre</i>	Autoadaptación
<i>Descripción</i>	El sistema deberá monitorear los cambios en los requerimientos que le sean solicitados.
<i>Entrada</i>	Resultado de la evaluación del cambio.
<i>Salida</i>	Estrategia de adaptación.
	<b>Requerimiento funcional (RF4)</b>
<i>Nombre</i>	Metamorfosis
<i>Descripción</i>	El sistema deberá de ser capaz de convertirse en nuevo sistema para satisfacer los cambios solicitados.
<i>Entrada</i>	Resultado de la evaluación del cambio.
<i>Salida</i>	Proceso de ingeniería de sistemas para la transición a un nuevo sistema.

Table 4.1: Requerimientos funcionales para un sistema de autoingeniería

Requerimientos no funcionales	
	Requerimiento no funcional (RNF1)
Nombre	Robustez [13]
Descripción	El sistema no deberá presentar alteraciones en su función normal, debe de ser capaz de lidiar con posibles fallas.
	Requerimiento no funcional (RNF2)
Nombre	Redundancia [13]
Descripción	El sistema deberá poseer métodos a nivel componente o de funcionalidad como respaldo en caso de fallos

Table 4.2: Requerimientos no funcionales de un sistema de autoingeniería

- **Autoingeniería.** A partir del análisis realizado en el módulo pasado, en esta etapa entonces se pueden realizar dos acciones:
  - *Autoadaptación:* será la técnica empleada a partir de que en el análisis de cambio se haya determinado que fuese del tipo configuración en los requerimientos. Esta etapa provoca la evolución del sistema y eventualmente nuevas versiones del sistema.
  - *Metamorfosis:* será la solución a implementar a partir del resultado del análisis del cambio, en donde se haya determinado que el cambio es del tipo funcionalidad del sistema, derivando así en un nuevo sistema.

#### 4.2.2.3 Arquitectura de los sistemas de autoingeniería basada en agentes

A partir de la arquitectura original se propone una arquitectura basada en agentes, el análisis correspondiente a la relación entre cada uno de los módulos propuestos se muestra en la figura 4.23, finalmente la arquitectura propuesta se muestra en la figura 4.24, en donde los agentes involucrados en la arquitectura son:

- **Monitor:** envía los requerimientos actuales al agente gestor de conocimiento, para posteriormente recibir los cambios solicitados en los requerimientos traducirlos y comunicarlos al agente analizador.
- **Analizador:** consulta los requerimientos actuales al agente gestor de conocimiento. A partir de los nuevos requerimientos dados se realiza una gestión del cambio, se crea un nuevo conjunto de requerimientos y se analiza la fase conveniente para el sistema a partir de los cambios solicitados, es decir, se analizarán si los cambios corresponden a autoadaptación o metamorfosis.

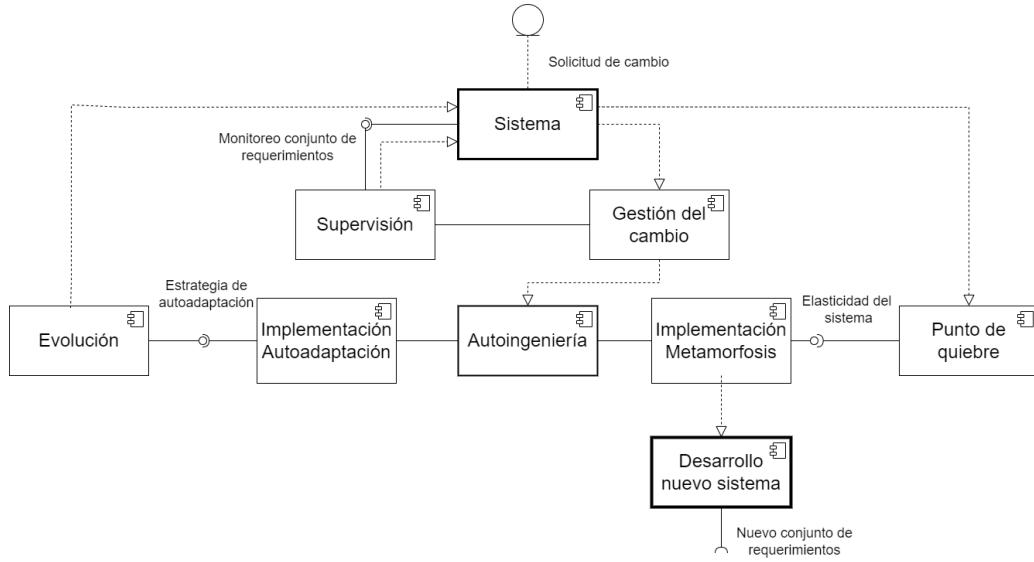


Figure 4.22: Vista de despliegue de la arquitectura SES

- **Planificador:** si en el análisis se determinó autoadaptación, entonces el agente planificador elabora una estrategia de adaptación. Por otro lado, si se determinó que es necesaria una metamorfosis, entonces el agente planificará una nueva ingeniería del sistema.
- **Ejecutor:** este agente aplicará los cambios correspondientes al sistema a partir de lo que el planificador le haya indicado.
- **Gestor de conocimiento:** almacena y entrega los requerimientos del sistema a los agentes que lo soliciten.

Asimismo, se muestra mediante un diagrama de actividad 4.25 y una máquina de estados abstractos 4.27 el funcionamiento que desempeñan cada uno de los agentes de la arquitectura propuesta.

#### 4.2.2.4 Modelo de vistas arquitectónicas

El modelo de vistas 4+1 de Krutchen [48] es utilizado para describir la arquitectura de un sistema software basado en el uso de múltiples puntos de vista. Para la representación de la arquitectura de la gestión del ciclo de vida de los sistemas de autoingeniería fueron consideradas la vista de despliegue, en donde se muestra como esta dividido el sistema SES en componentes (figura 4.22). De igual manera también

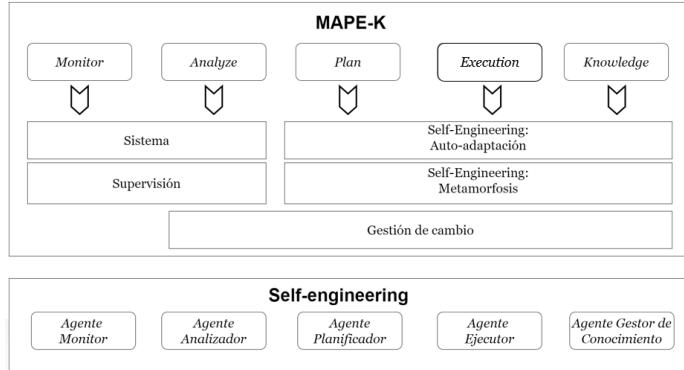


Figure 4.23: Analogía marco MAPE-K y arquitectura SES con agentes

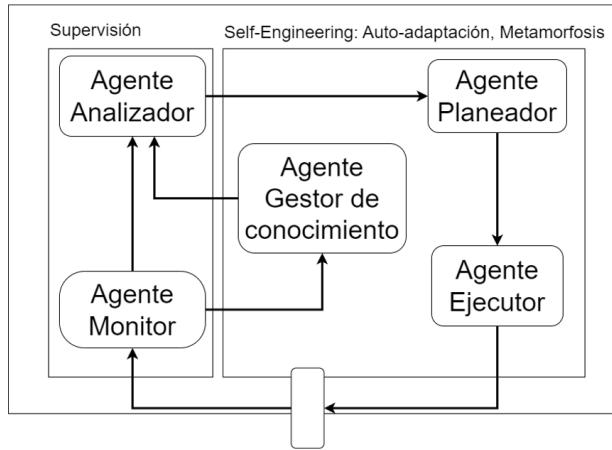


Figure 4.24: Vista de procesos de la arquitectura SES con agentes

se considera la vista de procesos en la que se muestran los agentes que hay en el sistema SES y la forma en la que estos se comunican (figuras 4.24 y 4.25). Por su parte, también se considera la vista lógica la cual representa la funcionalidad que el sistema proporcionará (figura 4.26).

#### 4.2.3 Resumen conceptual

Presentada la propuesta, ahora se muestra un mapa conceptual el cual resume lo que se ha propuesto y lo que define de manera general a los sistemas de auto ingeniería. En donde se plasman su definición, sus etapas, la arquitectura correspondiente, sus atributos y sus métodos 4.28

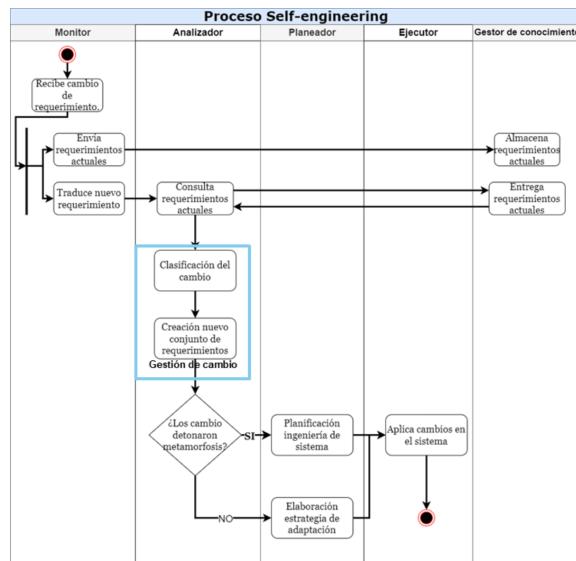


Figure 4.25: Diagrama de actividad de la arquitectura de los SES con agentes

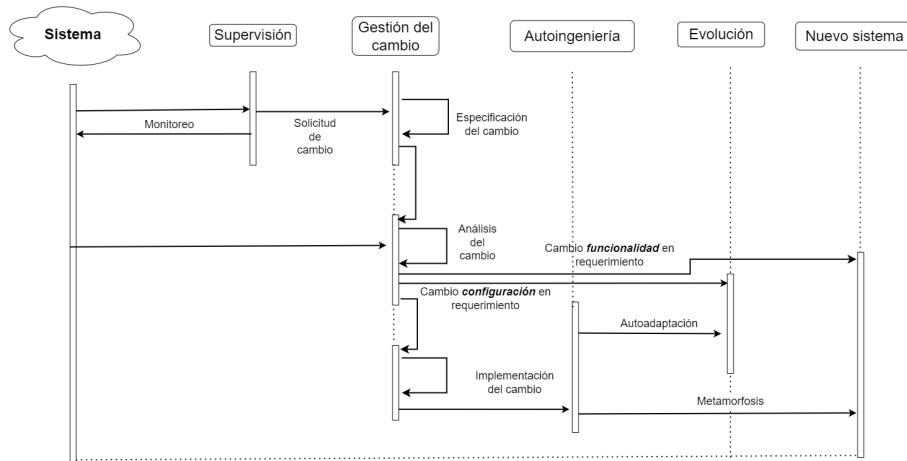


Figure 4.26: Vista lógica de la arquitectura de sistemas SES

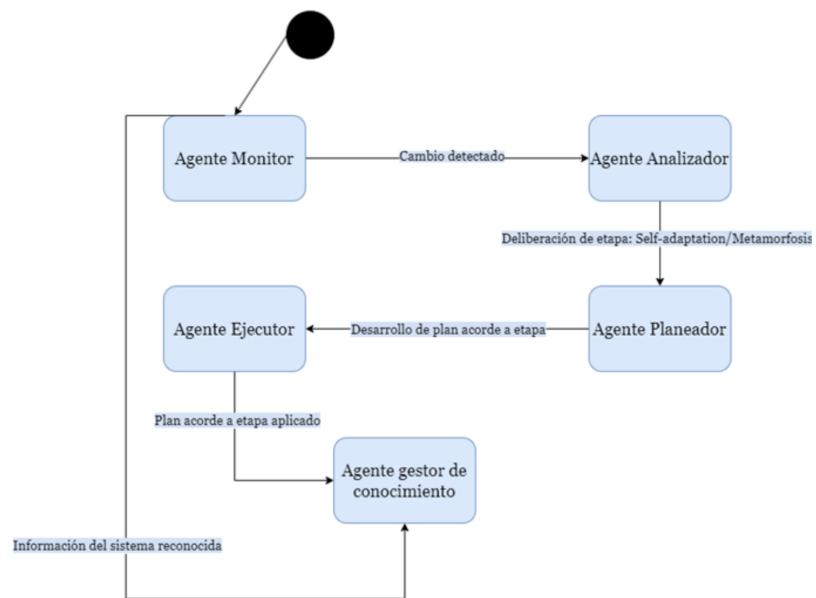


Figure 4.27: Diagrama de estados abstractos de la arquitectura de los SES con agentes

# Nuevo Marco Conceptual de Sistemas de Autoingeniería y una Arquitectura para la Gestión de su Ciclo de Vida

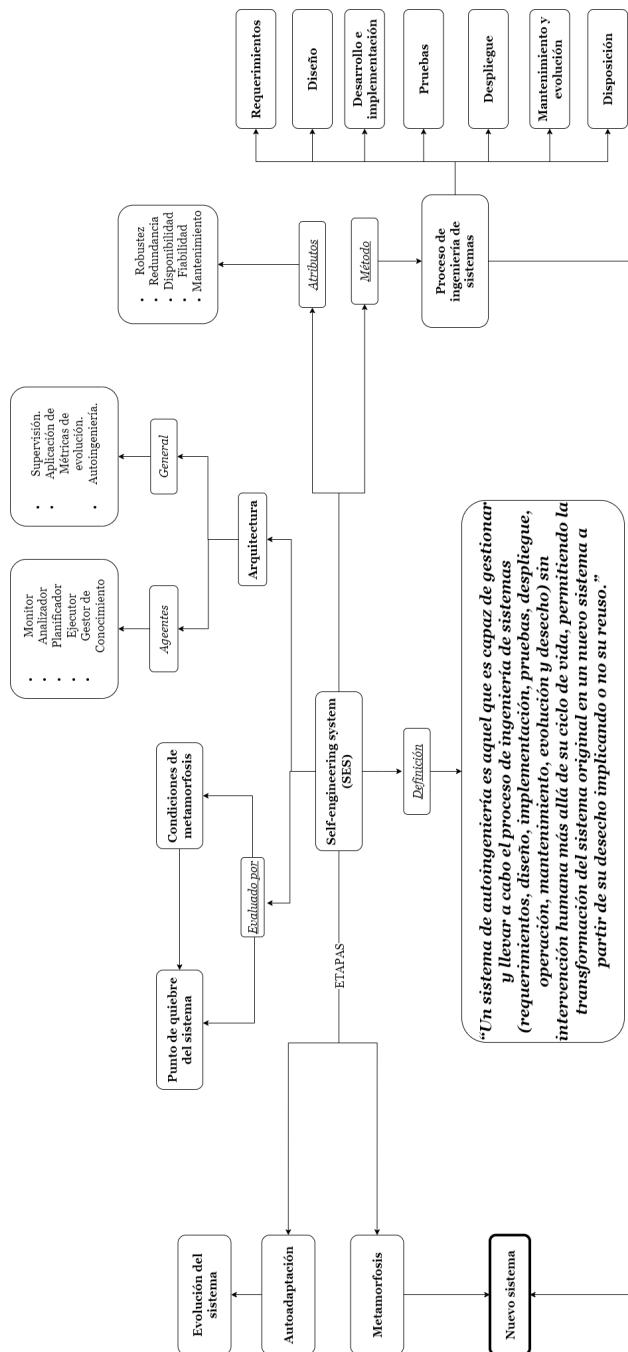


Figure 4.28: Mapa conceptual de los sistemas de autoingeniería

# **Capítulo 5**

## **Experimentación y Análisis de Resultados**

En esta sección se validará la propuesta del marco conceptual mediante el análisis y comparación con sistemas planteados previamente en la literatura, exemplificando casos de autoingeniería, fases y conceptos. Además, se presenta una prueba de concepto de la teoría y arquitectura, mediante una simulación basada en agentes para el caso de estudio de un sistema de autoingeniería con aplicación en un sistema ciberfísico.

### **5.1 Análisis de sistemas para validación del marco conceptual**

En esta sección se realizará un análisis teórico con sistemas existentes de la literatura, para plasmar con ejemplos de distintos sistemas la teoría anterior propuesta para su validación.

#### **5.1.1 Análisis del sistema: autoingeniería**

El propósito de este análisis será el evaluar un sistema en particular para determinar si sus principios de funcionamiento corresponden al de un sistema de autoingeniería.

##### **5.1.1.1 Planteamiento y descripción**

En [36] se plantea un sistema de estructuras robóticas autoensamblables, las cuales podrían ser utilizadas en futuras misiones espaciales, compuestas por armazones

cubo-octaédricos, en donde un robot camina por la superficie con sus extremidades llevando uno de estos armazones, lo coloca, y posteriormente un robot encargado de la fijación aprieta los puntos de fijación. Estos robots pueden adaptarse a realizar su trabajo en órbita, en la superficie lunar e incluso en otros planetas. El diseño de sistema este sistema permite el montaje y la reconfiguración automatizados de grandes estructuras. Este sistema demuestra el potencial de los metamateriales autónomos autoreconfigurables para diversas aplicaciones.

### 5.1.1.2 Análisis teórico

Es aquí en donde surge el cuestionamiento, ¿este sistema puede ser considerado como un sistema de autoingeniería? Analicemos sus propiedades, funcionamiento y métodos:

- *¿Sin intervención humana?* Si, el sistema realiza sus funciones y su adaptación a los cambios en distintas arquitecturas sin intervención humana.
- *¿Adaptación/autoadaptación?* Si, el sistema utiliza técnicas de autoadaptación para implementar las distintas estructuras que le son requeridas a través de las capacidades de los materiales utilizados.
- *¿Ingeniería de sistemas?* No, este sistema realiza sus ajustes a nivel lógico, funcional y de sus materiales, es decir el proceso para ajustar sus parámetros y objetivos de construcción no lleva las actividades involucradas en el proceso de ingeniería de sistemas, tales como requerimientos, diseño, despliegue, etc.
- *¿Ciclo de vida del sistema?* No, el ciclo de vida del sistema no es considerado.
- *Gestión del cambio.* Para el cambio en este sistema se realiza un análisis a nivel lógico y algorítmico para el cambio de estructuras, por ende si hay una gestión de cambio, pero sin un enfoque en cambio de requerimientos como la planteada en este marco conceptual.
- *¿Metamorfosis?* No hay una metamorfosis como tal, el sistema sigue funcionando para lo mismo, sigue realizando la construcción solo que distintos tipos de construcciones.
- *Deliberación:* El sistema no considera el ciclo de vida del sistema ni el proceso de ingeniería de sistemas, por lo tanto **no es un sistema de autoingeniería.**

### **5.1.2 Análisis de cambios en un sistema: autoadaptación y metamorfosis**

El propósito de este análisis será el evaluar en un sistema en particular las fases de autoadaptación y metamorfosis de la autoingeniería, a través de un análisis de la estructura de los requerimientos propuesta: *funcionalidad y configuración*.

#### **5.1.2.1 Planteamiento y descripción**

En [23] se aborda a la autoadaptación mediante un enfoque de tres etapas: supervisión, modelado y control del objetivo del sistema. En dicho trabajo, se reconoce las principales condiciones para la autoadaptación: *errores en el sistema, cambios en el ambiente en el que se desarrolla el sistema, y cambios en las preferencias del usuario (cambio en requerimientos)*. Además, se propone un marco arquitectónico que utiliza arquitecturas de software, y por medio de infraestructura reutilizable apoya la autoadaptación en estos sistemas. Se realiza una aplicación de su marco arquitectónico en un caso práctico de un sistema cliente-servidor con un enfoque en el rendimiento, el cual consiste en varios clientes conectados a un clúster de servidores. A través de la aplicación de este sistema se comprobó que la autoadaptación mejoró significativamente el rendimiento del sistema. En este caso de aplicación, se considera como principal factor para su estrategia de autoadaptación la latencia en el tiempo de respuesta del sistema. La estrategia de autoadaptación emplea una condicional, la latencia es monitoreada y cuando este valor excede dos segundos, se realiza un proceso en el cual el sistema se adapta y vuelve al umbral de latencia aceptado manteniendo su nivel de desempeño.

#### **5.1.2.2 Análisis teórico: autoadaptación en sistema de autoingeniería**

La estrategia de autoadaptación que fue diseñada para el sistema, a través del modelo arquitectónico propuesto, es capaz de reconocer un umbral para el valor de la latencia. Esta es una técnica para sobrellevar el error basado en los objetivos del sistema, sin embargo al no expresarse como requerimiento si el objetivo cambia el sistema no puede adaptarse. Así bien, recordemos que en los sistemas de autoingeniería su acción va enfocada hacia los cambios de requerimientos, ya sea en su configuración o funcionalidad, en donde su reacción para lidiar con ello no está predefinida o ya diseñada, sino que el sistema es capaz por sí solo de implementar la mejor solución para atender estos cambios solicitados. Entonces, ¿qué pasaría si a este mismo sistema planteado se le solicitaran cambios en sus requerimientos?

Consideremos el caso anterior del umbral de la latencia y su estrategia de autoadaptación para lidiar con ello, pero no desde el punto de vista de existencia de fallas y reparación de las mismas, sino como un cambio en sus requerimientos. Es decir, considerar que el parámetro de la latencia estaba especificado como uno de los requerimientos funcionales del sistema, ¿qué pasa cuando este *parámetro* es modificado?

Para esto, entonces planteamos los requerimientos para el sistema cliente-servidor, los cuales son mostrados en la tabla 5.1.

Podemos observar que el requerimiento RF3 se compone de las partes *funcionalidad*, en donde se especifica la acción por realizar en el sistema; y *configuración*, en donde se indica el parámetro de la acción a realizar (ver tabla 5.2).

Requerimiento funcional (RF3)	
Funcionalidad	Configuración
Respuesta a solicitudes	Max: 2 segundos.

Table 5.2: Funcionalidad y configuración del RF3

Entonces, podemos concluir que si el parámetro de este requerimiento o de cualquier otro cambiase podemos asegurar que el sistema seguirá siendo el mismo, no habrá como tal una nueva función y por ende no habrá que implementar un nuevo sistema.

### 5.1.2.3 Análisis teórico: metamorfosis en un sistema de autoingeniería

Por otro lado, surge el cuestionamiento ¿que es lo que llevaría este sistema un punto de quiebre derivando en una metamorfosis? La solución a esta pregunta se ha ido planteado a lo largo de la propuesta: *cambios en la funcionalidad*. Es decir, debe de existir un cambio en la funcionalidad de un conjunto de requerimientos de un sistema los cuales no puedan ser satisfechos por la arquitectura original del sistema y sea necesario entonces el desarrollo de un nuevo sistema.

Por ejemplo, añadirle funcionalidades al sistema del que se ha seleccionado, el servidor web, en donde el usuario final ha decidido cambiar, agregar o quitar ciertas funcionalidades pese a que ha cambiado el problema o necesidad al cual hay que satisfacer mediante el desarrollo del sistema, lo cual podría derivar en la necesidad de desechar el sistema actual y desarrollar un nuevo sistema implicando o no según sea el caso su reuso.

Veamos los nuevos requerimientos solicitados al sistema en donde se le está indicando las nuevas funcionalidades que tendrá que desempeñar, tabla 5.4.

Sistema cliente-servidor	
Requerimientos funcionales	
	<b>Requerimiento funcional (RF1)</b>
<i>Nombre</i>	Recepción de solicitudes
<i>Descripción</i>	El sistema se encarga de manejar las solicitudes de clientes (usuarios).
<i>Entrada</i>	Solicitudes de clientes
<i>Salida</i>	Aceptación de solicitud
	<b>Requerimiento funcional (RF2)</b>
<i>Nombre</i>	Procesamiento de solicitudes
<i>Descripción</i>	El servidor deberá de procesar las solicitudes de los usuarios.
<i>Entrada</i>	Aceptación de solicitud
<i>Salida</i>	Procesamiento de la solicitud
	<b>Requerimiento funcional (RF3)</b>
<i>Nombre</i>	Respuesta del servidor
<i>Descripción</i>	El servidor deberá de responder ante las solicitudes de los usuarios en un tiempo máximo de dos segundos.
<i>Entrada</i>	Respuesta del servidor
<i>Salida</i>	Tiempo de respuesta
	<b>Requerimiento funcional (RF4)</b>
<i>Nombre</i>	Gestión de conexiones (usuarios)
<i>Descripción</i>	A los usuarios se les tendrá que registrar, así como validarlos y autenticarlos como método de control de acceso.
<i>Entrada</i>	Conexiones, usuarios.
<i>Salida</i>	Control y monitoreo de los usuarios y sus respectivas conexiones.
Requerimientos no funcionales	
	<b>Requerimiento no funcional (RNF1)</b>
<i>Nombre</i>	Escalabilidad
<i>Descripción</i>	el servidor debe ser capaz de crecer, y soportar más solicitudes.
	<b>Requerimiento no funcional (RNF2)</b>
<i>Nombre</i>	Seguridad
<i>Descripción</i>	El servidor debe ser capaz de implementar estrategias para protección contra posibles ataques.

Table 5.1: Requerimientos sistema cliente-servidor

Nuevo Marco Conceptual de Sistemas de Autoingeniería y una Arquitectura para la Gestión de su Ciclo de Vida

---

Sistema cliente-servidor	
Requerimientos funcionales	
<b>Requerimiento funcional (RF5)</b>	
<i>Nombre</i>	Almacenamiento
<i>Descripción</i>	El sistema deberá ser capaz de almacenar archivos a gran escala.
<i>Entrada</i>	Archivos.
<i>Salida</i>	Almacenamiento de los archivos
<b>Requerimiento funcional (RF6)</b>	
<i>Nombre</i>	Procesamiento de datos
<i>Descripción</i>	El sistema deberá manejar, transformar y analizar los datos necesarios en respuesta a las solicitudes de los clientes.
<i>Entrada</i>	Datos de las conexiones
<i>Salida</i>	Control y monitoreo de los usuarios y sus respectivas conexiones.
<b>Requerimiento funcional (RF7)</b>	
<i>Nombre</i>	Análisis para el balanceo de carga
<i>Descripción</i>	El sistema deberá ser capaz de distribuir el tráfico de las solicitudes entre múltiples servidores.
<i>Entrada</i>	Múltiples solicitudes
<i>Salida</i>	Planificación y organización de las solicitudes.

Table 5.4: Requerimientos añadidos al sistema cliente-servidor

En donde podemos observar que los cambios se dan en añadir los requerimientos RF5, RF6, RF7.

Estos cambios se ven afectados en la funcionalidad del sistema (ver tabla 5.3), por lo tanto para que el sistema sea capaz de implementar estos cambios tendrá que convertirse en un nuevos sistema por un **metamorfosis**, empleando las actividades de un proceso de ingeniería de sistemas derivada.

Requerimiento funcional (RF5)	
Funcionalidad	Configuración
Almacenar archivos	tamaño (GB)
Requerimiento funcional (RF6)	
Funcionalidad	Configuración
Procesar datos	velocidad indicada
Requerimiento funcional (RF7)	
Funcionalidad	Configuración
Balancear carga	cantidad de tráfico y servidores designados

Table 5.3: Funcionalidad y configuración de requerimientos de metamorfosis

## 5.2 Experimentación

La experimentación llevada a cabo consiste en un caso de estudio de un sistema de autoingeniería con aplicación en un sistema ciberfísico mediante una simulación basada en agentes realizada en la plataforma GAMA, la cual es un entorno de modelado y simulación de código abierto para la creación de simulaciones basadas en agentes [32].

El objetivo de esta simulación es mostrar los conceptos de los sistemas de autoingeniería implementando su arquitectura, mostrando como resultado las condiciones de las fases de autoadaptación y metamorfosis a través de la representación del cambio de requerimientos.

Es importante aclarar que la simulación realizada **no** representa el funcionamiento de un sistema de autoingeniería, así como tampoco una implementación de sistemas de autoingeniería, sino la prueba de los conceptos planteados previamente en la propuesta.

### 5.2.1 Caso de estudio: ciclo de vida de un sistema SES con aplicación a un sistema ciberfísico de exploración, coordinación y rescate

El caso de estudio seleccionado surge a partir de la inspiración del caso planteado en [69] en donde se plantea un *sistema ciberfísico de coordinación para el rescate posterior a un terremoto*, en donde se abordan las interacciones entre robots, bomberos, paramédicos y víctimas desarrollándose en un escenario de un edificio posterior a un terremoto. Dicho trabajo plantea una perspectiva de la ingeniería de software con el enfoque de la autoadaptación para hacer frente a los cambios en tiempo de ejecución. Este caso de estudio fue elegido para la representación de sistemas de autoingeniería por la posibilidad de plasmar estos comportamientos en la simulación con agentes, además de la demostración la autoadaptación y metamorfosis de un SES con un enfoque en el cambio de requerimientos llevando más allá de su ciclo de vida al sistema.

### 5.2.2 Modelo sistema original

**Sistema A: Sistema ciberfísico de exploración y coordinación de rescate de personas en un edificio posterior a un sismo.**

### 5.2.2.1 Propósito

Modelar el ciclo de vida de un sistema de autoingeniería.

### 5.2.2.2 Entidades

- *Agente robot*: Tiene capacidades físicas para explorar y detectar posibles víctimas humanas. De igual manera, posee un software encargado de la parte lógica y de procesamiento que analiza y toma decisiones para la coordinación de los agentes bombero para el procedimiento de atención a los agentes objetivo.

Dado que el software del robot corresponde a un sistema SES, se involucran los agentes correspondientes a la arquitectura SES: monitor, analizador, planeador, ejecutor y gestor de conocimiento.

- *Agente bombero*: se encuentran en comunicación con los robots, recibiendo indicaciones sobre las acciones para proceder a la búsqueda de agentes objetivo.
- *Agente objetivo*: se encuentra localizado en alguna zona específica esperando a ser detectado por los agentes robot y atendido por los agentes bombero.

La interacción entre estos agentes, así como los procesos que realizan se ilustran como un diagrama de entidad relación en la figura 5.1.

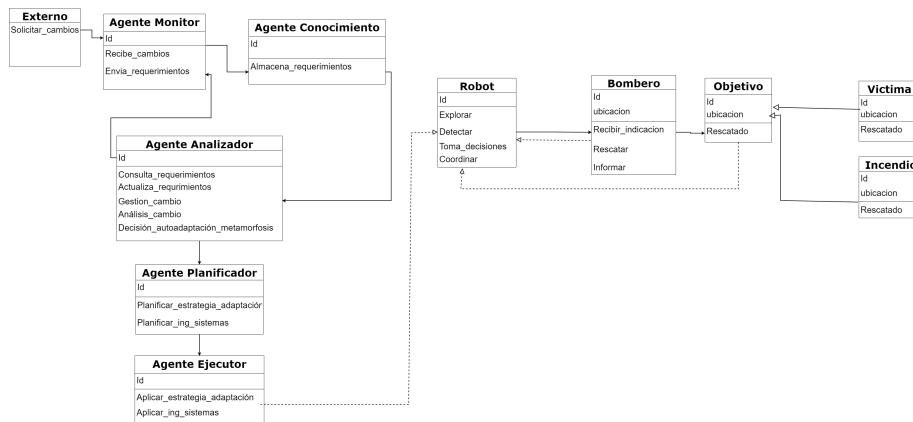


Figure 5.1: Diagrama de entidad relación de los agentes del sistema

### 5.2.2.3 Submodelos

El modelo del sistema original se compone de dos submodelos: arquitectura SES y sistema ciberfísico.

#### 1. Submodelo arquitectura.

- Supuestos:
  - La evolución es derivada de cambios en el ambiente y sigue el ciclo de vida de los SES.
  - Para gestionar el cambio derivado en una metamorfosis se implementan las actividades fundamentales de una ingeniería de sistemas: requerimientos, diseño, implementación, pruebas, despliegue, mantenimiento y evolución.
  - Todas las etapas de la ingeniería de sistemas realizadas tienen un producto de salida que alimenta a la etapa siguiente; dado que el propósito es simular el ciclo de vida se asume que estas salidas fueron dadas por una herramienta externa.
  - El sistema tiene como requerimientos establecidos los ilustrados en la tabla 5.5.
  - La arquitectura del sistema está basada en la mostrada en la figura 5.2.
- Proceso: las actividades realizadas por cada uno de los agentes de la arquitectura SES se muestra en la figura 5.3.

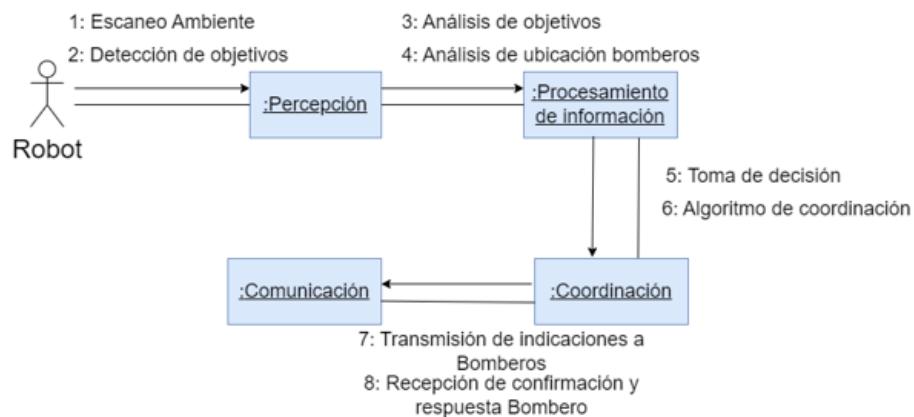


Figure 5.2: Arquitectura sistema A

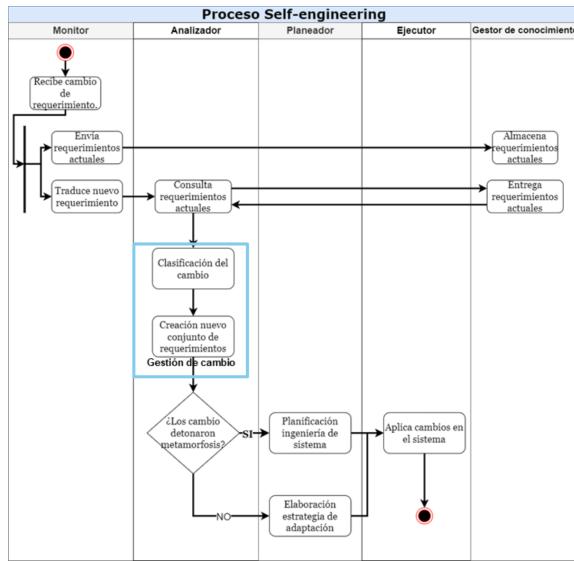


Figure 5.3: Procesos de los agentes en la arquitectura de sistemas SES

## 2. Submodelo sistema ciberfísico

- Supuestos:
  - Los robots son capaces de detectar en un radio de diez unidades a los agentes objetivo en un edificio posterior a un sismo.
  - El robot indica cuando ha encontrado agentes objetivo y el bombero indica cuando ha atendido estos objetivos.
  - El bombero se mantiene alerta para recibir notificaciones sobre posibles objetivos.
  - El bombero llega a la ubicación que el robot le para atender al objetivo.
  - Un cierto número de objetivos están ubicados en un área del edificio.

Cuando el agente objetivo cambia a ser un incendio:

- Los robots coordinan a los bomberos y generan escuadrones de bomberos designando a uno como líder.
- El robot calcula la gravedad del incendio y comparte estos datos con el líder del escuadrón.
- Los bomberos líderes reciben la información de los agentes robot.

- Los escuadrones de bomberos combaten los incendios en orden de importancia.
- Los incendios aparecen en el área forestal.
- Los incendios tienen nivel del uno al cuatro según su gravedad.
- Proceso: las actividades generales realizadas por cada uno de los agentes del sistema ciberfísico se muestra en la figura 5.4.

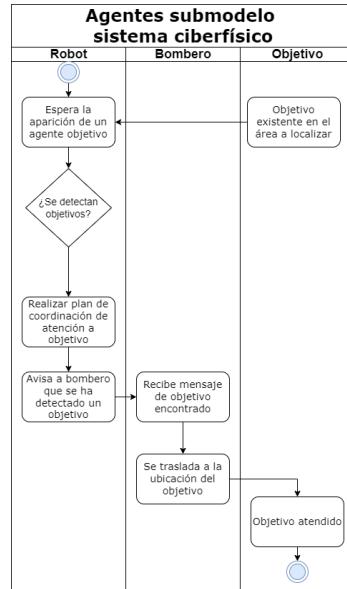


Figure 5.4: Procesos de los agentes en la arquitectura de sistemas SES

#### 5.2.2.4 Datos de entrada

Indican los parámetros para el modelo de simulación.

- Requerimientos: serán dados al sistema en formato *funcionalidad y configuración*.
- Etapa: el sistema modelado se puede encontrar en la etapa de despliegue para el sistema original, en evolución para el experimento de autoadaptación o en el punto de quiebre para el experimento de metamorfosis.
- Agente Objetivo: puede ser una víctima humana para el modelo original o un incendio para el experimento de adaptación y metamorfosis.

- Ambientes: para el modelo original se realiza en un edificio posterior a un sismo y en el experimento de autoadaptación y metamorfosis se da en un área forestal.

#### 5.2.2.5 Inicialización modelo original

- Requerimientos: los requerimientos señalados en la tabla 5.6 son dados en el formato *funcionalidad y configuración*.
- Etapa: despliegue.
- Agente Objetivo: víctima humana.
- Ambientes: edificio posterior a un sismo La inicialización en la simulación puede ser observada en la figura 5.5.

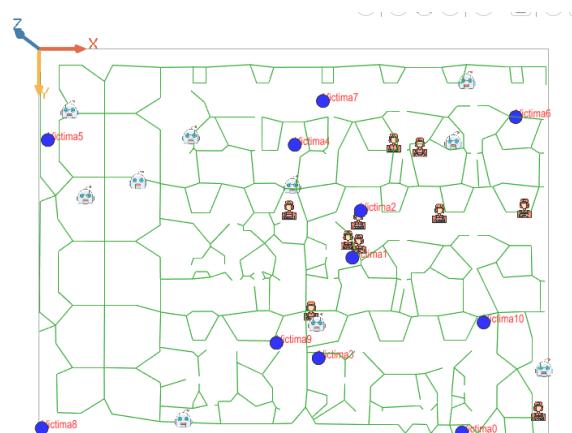


Figure 5.5: Implementación Sistema A

### 5.2.3 Experimento: autoadaptación

#### 5.2.3.1 Supuestos

#### 5.2.3.2 Configuración del experimento

- Requerimientos: los requerimientos señalados en la tabla 5.6 son dados en el formato *funcionalidad y configuración*.

Sistema A	
Sistema ciberfísico de exploración y coordinación de rescate en un edificio posterior a un sismo.	
Requerimientos funcionales	
	<b>Requerimiento funcional (RF1)</b>
<i>Nombre</i>	Detección de víctimas humanas
<i>Descripción</i>	Los robots deben escanear el entorno haciendo uso de sensores integrados para detectar la presencia de las víctimas así como su localización.
<i>Entrada</i>	Búsqueda por sensado de víctimas.
<i>Salida</i>	Localización de las víctimas.
	<b>Requerimiento funcional (RF2)</b>
<i>Nombre</i>	Uso de sensores
<i>Descripción</i>	Los robots realizan la detección de las víctimas mediante sensores determinados.
<i>Entrada</i>	Parámetros del ambiente
<i>Salida</i>	Mediciones de los sensores
	<b>Requerimiento funcional (RF3)</b>
<i>Nombre</i>	Comunicación y transmisión de datos.
<i>Descripción</i>	Los robots deben transmitir la localización de las víctimas a los bomberos, así los bomberos pueden dirigirse hacia la víctima.
<i>Entrada</i>	Dato de la localización de las víctimas.
<i>Salida</i>	Transmisión del dato.
	<b>Requerimiento funcional (RF4)</b>
<i>Nombre</i>	Coordinación de rescate
<i>Descripción</i>	Los robots detectan la presencia de víctimas y coordinan con el bombero mejor ubicado para llegar a la víctima.
<i>Entrada</i>	Localización del robot, víctima y bombero.
<i>Salida</i>	Decisión de rescate.
	<b>Requerimiento funcional (RF5)</b>
<i>Nombre</i>	Monitoreo del estado de las víctimas
<i>Descripción</i>	Los robots utilizarán sus sensores para la detección del estado general de las víctimas para que los bomberos se encarguen de atenderles.
<i>Entrada</i>	Estado de la víctima.
<i>Salida</i>	Atención a la víctima.

Table 5.5: Requerimientos funcionales sistema A



Figure 5.6: Simulación de la implementación del sistema A evolucionado

- Etapa: evolución
- Agente Objetivo: incendios.
- Ambientes: zona forestal. La configuración del experimento en la simulación puede ser observada en la figura 5.6.

### 5.2.3.3 Propósito del experimento

Validar la autoadaptación del sistema mediante el cambio de configuración en requerimientos.

### 5.2.4 Experimento: metamorfosis

#### 5.2.4.1 Configuración del experimento

- Requerimientos: los requerimientos señalados en la tabla 5.7 son dados en el formato *funcionalidad y configuración*.
- Etapa: punto de quiebre.
- Agente Objetivo: incendios.
- Ambientes: zona forestal con evaluación del suelo. La configuración del experimento en la simulación puede ser observada en la figura 5.8, de igual manera la arquitectura a ser implementada como parte en la metamorfosis se muestra en la figura 5.7 .

Sistema A-V2	
Sistema ciberfísico de coordinación de detección y atención a incendios	
Requerimientos funcionales	
	<b>Requerimiento funcional (RF1)</b>
<i>Nombre</i>	Detección de incendios
<i>Descripción</i>	Los robots deben escanear el entorno haciendo uso de sensores integrados para detectar la presencia de las incendios así como su localización.
<i>Entrada</i>	Búsqueda por sensado de incendios.
<i>Salida</i>	Localización de los incendios.
	<b>Requerimiento funcional (RF2)</b>
<i>Nombre</i>	Uso de sensores
<i>Descripción</i>	Los robots realizan la detección de los incendios mediante sensores
<i>Entrada</i>	Parámetros del ambiente
<i>Salida</i>	Mediciones de los sensores
	<b>Requerimiento funcional (RF3)</b>
<i>Nombre</i>	Comunicación y transmisión de datos.
<i>Descripción</i>	Los robots deben transmitir la localización de los incendios a los bomberos, así los bomberos pueden dirigirse hacia el incendio.
<i>Entrada</i>	Dato de la localización de los incendios.
<i>Salida</i>	Transmisión del dato.
	<b>Requerimiento funcional (RF4)</b>
<i>Nombre</i>	Coordinación de atención a incendios
<i>Descripción</i>	Los robots detectan la presencia de víctimas y coordinan con el bombero mejor ubicado para llegar a la víctima
<i>Entrada</i>	Localización del robot, incendio y bombero.
<i>Salida</i>	Decisión de la atención al incendio.
	<b>Requerimiento funcional (RF5)</b>
<i>Nombre</i>	Monitoreo del entorno
<i>Descripción</i>	Los robots utilizarán sus sensores para el registro el nivel y condiciones del incendio para que los bomberos se encarguen de atenderles.
<i>Entrada</i>	Nivel del incendio
<i>Salida</i>	Estrategia para defensa para el incendio.

Table 5.6: Requerimientos funcionales Sistema A-V2

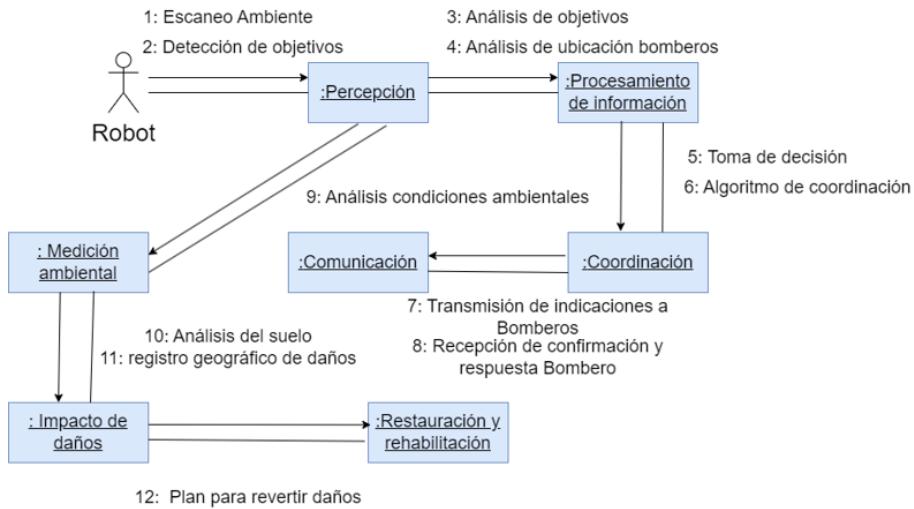


Figure 5.7: Arquitectura con los cambios implementados

#### 5.2.4.2 Propósito

Validar la metamorfosis del sistema a partir del cambio en la funcionalidad de requerimientos en la etapa del punto de quiebre del sistema.

### 5.2.5 Análisis de resultados

El propósito de la experimentación era la demostración del ciclo de vida de los sistemas de autoingeniería, es así que como parte de nuestros resultados podemos identificar que se ha validado la teoría correspondiente al marco conceptual propuesto, así como de la arquitectura propuesta para la gestión del ciclo de vida de un sistema de autoingeniería.

#### 5.2.5.1 Análisis de resultados: experimento autoadaptación

A partir de lo planteado en la propuesta se realiza un análisis con los resultados obtenidos en la experimentación correspondiente al experimento de autoadaptación.

- En un sistema de autoingeniería los cambios en sus requerimientos provocarán una evolución en el sistema

$$\sigma_2 = \square(cambio \Rightarrow E) \quad (5.1)$$

Sistema B	
<b>Sistema ciber físico de coordinación de detección de incendios forestales con valoración de erosión de suelo y estado de vegetación para restauración y rehabilitación de la zona</b>	
Requerimientos funcionales	
	<b>Requerimiento funcional (RF1)</b>
<i>Nombre</i>	Detección de incendios
<i>Descripción</i>	Los robots deben escanear el entorno haciendo uso de sensores integrados para detectar la presencia de los incendios así como su localización.
<i>Entrada</i>	Búsqueda por sensado de incendios.
<i>Salida</i>	Localización de los incendios.
	<b>Requerimiento funcional (RF2)</b>
<i>Nombre</i>	Uso de sensores
<i>Descripción</i>	Los robots realizan la detección de los incendios mediante sensores
<i>Entrada</i>	Parámetros del ambiente
<i>Salida</i>	Mediciones de los sensores
	<b>Requerimiento funcional (RF3)</b>
<i>Nombre</i>	Comunicación y transmisión de datos.
<i>Descripción</i>	Los robots deben transmitir la localización de los incendios a los bomberos, así los bomberos pueden dirigirse hacia los incendios.
<i>Entrada</i>	Dato de la localización de los incendios.
<i>Salida</i>	Transmisión del dato.
	<b>Requerimiento funcional (RF4)</b>
<i>Nombre</i>	Coordinación de atención a incendios
<i>Descripción</i>	Los robots detectan la presencia de incendios y coordinan con el bombero mejor ubicado para llegar a la víctima
<i>Entrada</i>	Localización del robot, incendio y bombero.
<i>Salida</i>	Decisión de la atención al incendio.
	<b>Requerimiento funcional (RF5)</b>
<i>Nombre</i>	Monitoreo del entorno
<i>Descripción</i>	Los robots utilizarán sus sensores para el registro del estado general del incendio para que los bomberos se encarguen de atenderlos
<i>Entrada</i>	Nivel del incendio
<i>Salida</i>	Estrategia para defensa para el incendio.
	<b>Requerimiento funcional (RF6)</b>
<i>Nombre</i>	Ánálisis de daños de incendio
<i>Descripción</i>	El sistema hará mediciones pertinentes a la erosión del suelo y condiciones de la vegetación
<i>Entrada</i>	Condiciones del ambiente
<i>Salida</i>	Daños generados
	<b>Requerimiento funcional (RF7)</b>
<i>Nombre</i>	Planificación de estrategias
<i>Descripción</i>	El sistema definirá el nivel de daño presente, así como estrategias para revertir o tratar el área
<i>Entrada</i>	Análisis del daño
<i>Salida</i>	Planificación de restauración

Table 5.7: Requerimientos sistema B

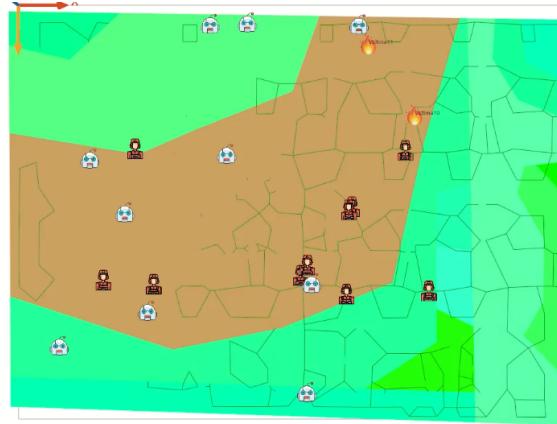


Figure 5.8: Simulación de la implementación del sistema A con cambios

- A partir de los cambios en la **configuración** de los requerimientos se da una autoadaptación.

$$\sigma_3 = \square(C_i : Configuración_j \Rightarrow AD) \quad (5.2)$$

- La autoadaptación de un sistema original resultará en un sistema evolucionado.

$$\sigma_7 = \diamond(Sistema_A \Rightarrow AD_i) \Rightarrow \bigcirc(Sistema_A \cup C_i) \quad (5.3)$$

- En la experimentación realizada para la observación del ciclo de vida del sistema, se tiene un sistema evolucionado a partir del cambio en la configuración de un requerimiento (el tipo de agentes objetivo), en donde no se llega a un punto de quiebre (figura 5.9). De igual manera podemos observar que en un análisis de cumplimiento de requerimientos (figura 5.10) cuando se presenta un cambio en la configuración no se ve afectado este cumplimiento, además que esto no implicaría la transformación a un nuevo sistema.

### 5.2.5.2 Experimento: metamorfosis

A partir de lo planteado en la propuesta se realiza un análisis con los resultados obtenidos en la experimentación correspondiente al experimento de metamorfosis.

- En un sistema de autoingeniería los cambios en sus requerimientos provocarán una evolución en el sistema

$$\sigma_2 = \square(cambio \Rightarrow E) \quad (5.4)$$

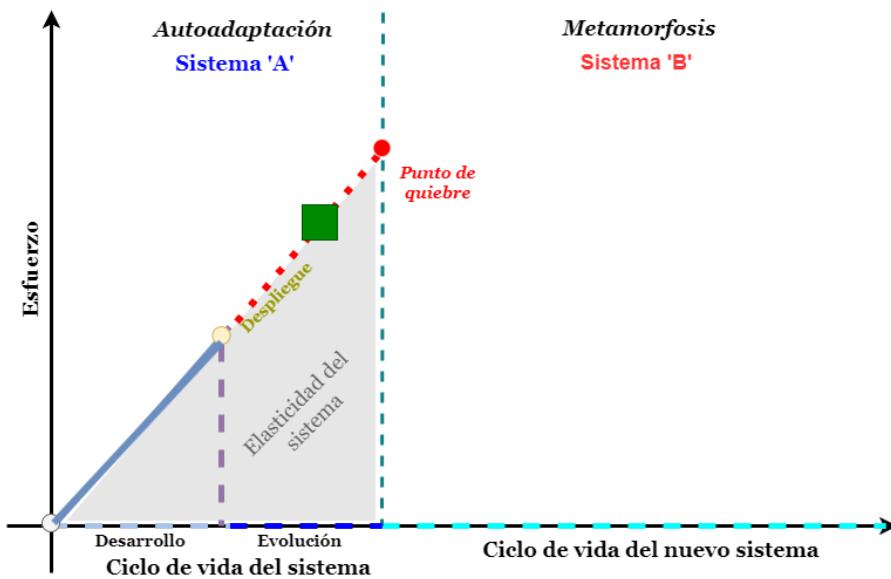


Figure 5.9: Evolución del sistema

#### Verificación del funcionamiento del sistema acorde a los requerimientos planteados

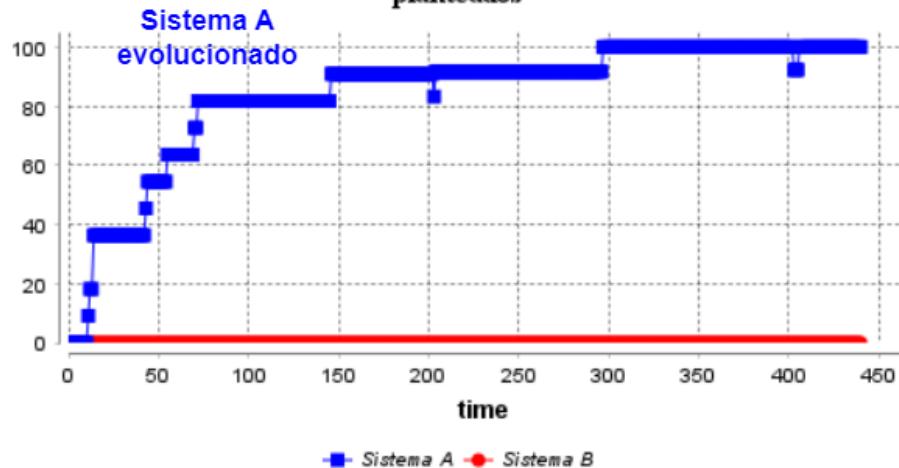


Figure 5.10: Análisis de cumplimiento de requerimientos en la fase de autoadaptación de un SES

- A partir de los cambios en la **funcionalidad** de los requerimientos se da la **metamorfosis**.

$$\sigma_4 = \square(C_i : Funcionalidad_j \Rightarrow M) \quad (5.5)$$

- La metamorfosis se da a partir de un cambio en el conjunto de requerimientos.

$$\sigma_5 = \square\Diamond(\bigwedge_{1 \leq i \leq a} (AD_i) \vee \bigvee_{1 \leq i \leq m} (M_i)) \quad (5.6)$$

- La evolución de un sistema eventualmente conllevará a una metamorfosis.

$$\sigma_6 = \Diamond(E \Rightarrow M) \quad (5.7)$$

- La metamorfosis en un sistema conllevará un nuevo sistema.

$$\sigma_8 = \square[((Sistema_A \cup C_i) \Rightarrow M_i)) \Rightarrow \bigcirc Sistema_B] \quad (5.8)$$

- En la experimentación realizada para la observación del ciclo de vida del sistema, el sistema llega a un punto de quiebre a partir del cambio en la funcionalidad de los requerimientos, para así pasar a la fase de metamorfosis del sistema y convertirse en un nuevo sistema (figura 5.11). De igual manera podemos observar que en un análisis de cumplimiento de requerimientos cuando se presenta un cambio en la funcionalidad de los requerimientos estos dejan de ser satisfechos por el sistema original (sistema A) y es entonces que necesita de un nuevo sistema (sistema B) para satisfacer estos nuevos requerimientos (figura 5.12).

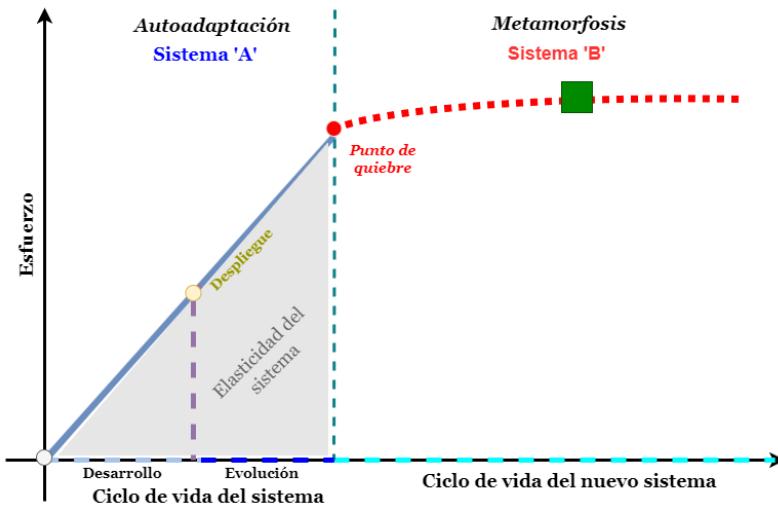


Figure 5.11: Análisis de cumplimiento de requerimientos en la fase de metamorfosis de un SES

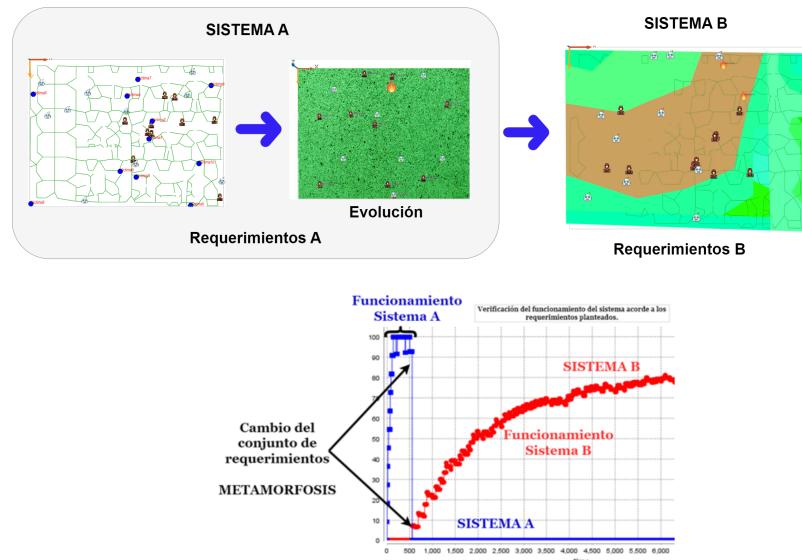


Figure 5.12: Análisis resultados

# Capítulo 6

## Conclusiones y Trabajo Futuro

### 6.1 Conclusiones

En este trabajo se ha propuesto un nuevo marco conceptual sobre los sistemas de autoingeniería desde una perspectiva distinta a la que se les había atribuido en trabajos previos, centrándose principalmente en la realización de las actividades de la ingeniería de sistemas sin intervención humana, dejando de lado la detección y atención de fallos como funcionamiento principal de estos sistemas.

Además, se presentaron los conceptos fundamentales que componen este marco teórico y se definieron formalmente, entre los que destacan: definición de los sistemas de autoingeniería, fase de autoadaptación, metamorfosis, evolución y punto de quiebre. Asimismo, se presentó una propuesta de arquitectura para la gestión del ciclo de vida de los sistemas de autoingeniería.

Se evaluaron las condiciones de metamorfosis para el análisis de sistemas. En donde se validó la clasificación propuesta del tipo de cambios en requerimientos: *configuración y funcionalidad*.

Para el análisis teórico, se propusieron dos casos de estudio: en el primero se analizó un sistema establecido para determinar si puede considerarse un sistema de autoingeniería, con base en las definiciones dadas; en el segundo, se consideró un sistema autoadaptativo para detallar la teoría de sistemas de autoingeniería.

Adicionalmente, se realizó la validación de la teoría en un tercer caso de estudio mediante la experimentación con un modelo basado en agentes, donde se manifestó

la importancia de los sistemas que pueden gestionarse y adaptarse a través de su evolución y eventual metamorfosis sin necesidad de intervención humana.

Como resultado, se concluyó que el punto de quiebre de un sistema se produce cuando la funcionalidad en el conjunto de requerimientos es distinta a la original, derivando en una metamorfosis que transforma al sistema en uno nuevo a nivel arquitectónico y funcional a través de la realización de una ingeniería de sistemas.

Se puede afirmar que el trabajo de investigación realizado supuso un aporte fundamental para el futuro desarrollo de los sistemas de autoingeniería. Para que el desarrollo de los sistemas de autoingeniería se convierta en una realidad, es esencial tener primero una base conceptual, la cual ha sido propuesta en este trabajo.

## 6.2 Objetivos logrados

- Se realizó una revisión sistemática sobre la teoría de sistemas de autoingeniería, lo que permitió identificar la falta de un proceso de ingeniería de sistemas sin intervención humana que llevara al sistema más allá de su ciclo de vida.
- Se investigó el límite de la evolución de un sistema dentro de su ciclo de vida en el contexto de la ingeniería de sistemas.
- Se logró redefinir la teoría de autoingeniería (self-engineering) de sistemas mediante un nuevo marco conceptual.
- Se desarrolló una arquitectura para la gestión del ciclo de vida de los sistemas de autoingeniería.
- Se validó el marco conceptual y arquitectura propuestos mediante casos de estudio.

## 6.3 Trabajo futuro

Este trabajo de tesis se limitó al planteamiento conceptual de los sistemas de autoingeniería, por lo que se pueden identificar distintas áreas de investigación como trabajo futuro para los sistemas de autoingeniería.

## Nuevo Marco Conceptual de Sistemas de Autoingeniería y una Arquitectura para la Gestión de su Ciclo de Vida

---

En primer lugar, debe considerarse la aplicación del marco conceptual desarrollado y la experimentación a casos de estudio con otros sistemas de software.

También se deberá ampliar el horizonte de aplicación más allá de los sistemas de software propuestos en este trabajo. Esto servirá para validar la compatibilidad con dichos sistemas y explorar adecuadamente las fortalezas y debilidades de este enfoque.

Por último, se deberá profundizar en las herramientas y tecnologías a utilizar para cada una de las etapas de la ingeniería de sistemas en un SES, debido a que este proceso es realizado sin intervención humana.

# Bibliografía

- [1] Shallaw Mohammed Ali et al. “Developing an agent-based simulation model of software evolution”. In: *Information and Software Technology* 96 (2018), pp. 126–140.
- [2] Hala Alrumaih, Abdulrahman Mirza, and Hessah Alsalamah. “Toward automated software requirements classification”. In: *2018 21st Saudi computer society national computer conference (NCC)*. IEEE. 2018, pp. 1–6.
- [3] Charles Anderson. “Docker [software engineering]”. In: *Ieee Software* 32.3 (2015), pp. 102–c3.
- [4] Konstantinos Angelopoulos et al. “Engineering self-adaptive software systems: From requirements to model predictive control”. In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 13.1 (2018), pp. 1–27.
- [5] Victor R. Basili. “Viewing maintenance as reuse-oriented software development”. In: *IEEE software* 7.1 (1990), pp. 19–25.
- [6] Alessandro Bassi et al. *Enabling Things to Talk: Designing IoT Solutions with TheIoT Architectural Reference Model*. Springer Nature, 2013.
- [7] Stafford Beer. “The viable system model: Its provenance, development, methodology and pathology”. In: *Journal of the operational research society* 35.1 (1984), pp. 7–25.
- [8] Keith H Bennett and Václav T Rajlich. “Software maintenance and evolution: a roadmap”. In: *Proceedings of the Conference on the Future of Software Engineering*. 2000, pp. 73–87.
- [9] D. Benyon and Dianne M. Murray. “Adaptive systems: from intelligent tutoring to autonomous agents”. In: *Knowl. Based Syst.* 6 (1993), pp. 179–219. DOI: [10.1016/0950-7051\(93\)90012-I](https://doi.org/10.1016/0950-7051(93)90012-I).
- [10] Cory Bishop et al. “What is metamorphosis?” In: *Integrative and comparative biology* 46 (Dec. 2006), pp. 655–61. DOI: [10.1093/icb/icl004](https://doi.org/10.1093/icb/icl004).

- [11] Barry W. Boehm. “Software Engineering”. In: *IEEE Transactions on Computers* (1976). DOI: 10.1109/TC.1976.1674590.
- [12] Sam Brooks and Rajkumar Roy. “A Complexity Framework for Self-Engineering Systems”. In: *Smart and Sustainable Manufacturing Systems* 4 (Mar. 2020), p. 20200059. DOI: 10.1520/SSMS20200059.
- [13] Sam Brooks and Rajkumar Roy. “An Overview of Self-engineering Systems”. In: *Journal of Engineering Design* 32 (Apr. 2021). DOI: 10.1080/09544828.2021.1914323.
- [14] Sam Brooks and Rajkumar Roy. “Complexity of self-engineering systems across the life cycle – Biological and engineering systems”. In: *Procedia CIRP* 98 (Jan. 2021), pp. 121–126. DOI: 10.1016/j.procir.2021.01.016.
- [15] Sam Brooks and Rajkumar Roy. “Design and complexity evaluation of a self-cleaning heat exchanger”. In: *International Journal of Heat and Mass Transfer* 191 (Aug. 2022), p. 122725. DOI: 10.1016/j.ijheatmasstransfer.2022.122725.
- [16] Sam Brooks et al. “A systematic study of biological SE systems from complexity and design perspectives”. In: *Journal of Engineering Design* 34 (Oct. 2023), pp. 1–25. DOI: 10.1080/09544828.2023.2266864.
- [17] Manfred Broy. “Multifunctional software systems: Structured modeling and specification of functional requirements”. In: *Science of Computer Programming* 75.12 (2010), pp. 1193–1214.
- [18] Andrew T Campbell, Geoff Coulson, and Michael E Kounavis. “Managing complexity: Middleware explained”. In: *IT professional* 1.5 (1999), pp. 22–28.
- [19] Antonio Cavacini. “What is the best database for computer science journal articles?” In: *Scientometrics* 102 (Mar. 2014), pp. 2059–2071. DOI: 10.1007/s11192-014-1506-1.
- [20] Ned Chapin et al. “Types of software evolution and software maintenance”. In: *Journal of software maintenance and evolution: Research and Practice* 13.1 (2001), pp. 3–30.
- [21] Betty Cheng et al. “08031 – Software Engineering for Self-Adaptive Systems: A Research Road Map”. In: () .
- [22] Betty H. C. Cheng et al. *Software Engineering for Self-Adaptive Systems*. Springer Science Business Media, June 2009.

- [23] Shang-Wen Cheng, David Garlan, and Bradley Schmerl. “Making Self-Adaptation an Engineering Reality”. In: Jan. 2005, pp. 158–173. ISBN: 978-3-540-26009-7. DOI: 10.1007/11428589\_11.
- [24] D. Dori et al. “System Definition, System Worldviews, and Systemness Characteristics”. In: *IEEE Systems Journal* 14 (2020), pp. 1538–1548. DOI: 10.1109/JSYST.2019.2904116.
- [25] Liliana Durán-Polanco and Mario Siller. “A taxonomy for decision making in IoT systems”. In: *Internet of Things* 24 (2023), p. 100904.
- [26] Jon Elphick. “What is systems engineering?” In: *IEEE Aerospace and Electronic Systems Magazine* 15 (2000), pp. 9–10. DOI: 10.1109/62.879392.
- [27] Muhammad Faisal et al. “How automate requirements engineering system effects and support requirement engineering”. In: *2022 International Conference on Business Analytics for Technology and Security (ICBATS)*. IEEE. 2022, pp. 1–3.
- [28] Angela Fan et al. “Large language models for software engineering: Survey and open problems”. In: *2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE)*. IEEE. 2023, pp. 31–53.
- [29] Juan Fernandez-Ramil and Meir Lehman. “Metrics of software evolution as effort predictors - a case study”. In: Feb. 2000, pp. 163–172. ISBN: 0-7695-0753-0. DOI: 10.1109/ICSM.2000.883036.
- [30] Juan Fernández-Ramil and Meir M. Lehman. “Metrics of software evolution as effort predictors - a case study”. In: *Proceedings 2000 International Conference on Software Maintenance* (2000), pp. 163–172. URL: <https://api.semanticscholar.org/CorpusID:6548377>.
- [31] Stefan Feuerriegel et al. “Generative ai”. In: *Business & Information Systems Engineering* 66.1 (2024), pp. 111–126.
- [32] *GAMA Platform — GAMA Platform*. URL: <http://gama-platform.org/>.
- [33] David Garlan, Bradley Schmerl, and Shang-Wen Cheng. “Software Architecture-Based Self-Adaptation”. In: Apr. 2009, pp. 31–55. ISBN: 978-0-387-89827-8. DOI: 10.1007/978-0-387-89828-5\_2.
- [34] M. Gillani, Hafiz Adnan Niaz, and A. Ullah. “Integration of Software Architecture in Requirements Elicitation for Rapid Software Development”. In: *IEEE Access* PP (2022), pp. 1–1. DOI: 10.1109/ACCESS.2022.3177659.

- [35] Martin Glinz. “On Non-Functional Requirements”. In: *15th IEEE International Requirements Engineering Conference (RE 2007)*. 2007, pp. 21–26. DOI: 10.1109/RE.2007.45.
- [36] Christine E Gregg et al. “Ultralight, strong, and self-reprogrammable mechanical metamaterials”. In: *Science robotics* 9.86 (2024), eadi2746.
- [37] Penny Grubb and Armstrong A Takang. *Software Maintenance: Concepts and Practice (Second Edition)*. World Scientific, July 2003.
- [38] Martin Hall and Daniel Martín-Vega. “Visualization of insect metamorphosis”. In: *Philosophical transactions of the Royal Society of London. Series B, Biological sciences* 374 (Aug. 2019), p. 20190071. DOI: 10.1098/rstb.2019.0071.
- [39] D. Honsel, S. Waack, and J. Grabowski. *Development of Agent-based Simulation Models for Software Evolution*. Georg-August-Universität Göttingen, 2019.
- [40] Verena Honsel. “Statistical learning and software mining for agent based simulation of software evolution”. In: *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*. Vol. 2. IEEE. 2015, pp. 863–866.
- [41] Petr Jan Horn. “Autonomic Computing: IBM’s Perspective on the State of Information Technology”. In: 2001. URL: <https://api.semanticscholar.org/CorpusID:59693806>.
- [42] “IEEE Standard for Application and Management of the Systems Engineering Process”. In: *IEEE Std 1220-1998* (1999), pp. 1–84. DOI: 10.1109/IEEESTD.1999.88825.
- [43] *International Council on Systems Engineering Website*. URL: <https://www.incose.org/>.
- [44] ISO and IEC. “International Standard ISO/IEC 12207 : Information Technology - Software Life Cycle Processes / ISO, IEC.” In: *Systems and software engineering — Software life cycle processes*.
- [45] Shalinka Jayatilleke and Richard Lai. “A systematic review of requirements change management”. In: *Information and Software Technology* 93 (2018), pp. 163–185.
- [46] Jeffrey Kephart and D.M. Chess. “The Vision Of Autonomic Computing”. In: *Computer* 36 (Feb. 2003), pp. 41–50. DOI: 10.1109/MC.2003.1160055.
- [47] Jeff Kramer and Jeff Magee. “Self-managed systems: an architectural challenge”. In: *Future of Software Engineering (FOSE’07)*. IEEE. 2007, pp. 259–268.

- [48] Phillippe Kruchten. “Architecture blueprints—the “4+ 1” view model of software architecture”. In: *Tutorial proceedings on TRI-Ada’91: Ada’s role in global markets: solutions for a changing complex world*. 1995, pp. 540–555.
- [49] F. Lanubile and G. Visaggio. “Iterative reengineering to compensate for quick-fix maintenance”. In: *Proceedings of International Conference on Software Maintenance*. 1995, pp. 140–146. DOI: 10.1109/ICSM.1995.526536.
- [50] Jay Lee, Mahsa Ghaffari, and Shaza Elmeligy. “Self-maintenance and engineering immune systems: Towards smarter machines and manufacturing systems”. In: *Annual Reviews in Control* 35 (Apr. 2011), pp. 111–122. DOI: 10.1016/j.arcontrol.2011.03.007.
- [51] Meir Lehman et al. “Metrics and Laws of Software Evolution - The Nineties View.” In: Jan. 1997, pp. 20–. DOI: 10.1109/METRIC.1997.637156.
- [52] Bennet P Lientz. “Issues in software maintenance”. In: *ACM Computing Surveys (CSUR)* 15.3 (1983), pp. 271–278.
- [53] Bennett P. Lientz and E. Burton Swanson. *Software Maintenance Management*. USA: Addison-Wesley Longman Publishing Co., Inc., 1980. ISBN: 0201042053.
- [54] Pericles Loucopoulos and Vassilios Karakostas. *System requirements engineering*. McGraw-Hill, Inc., 1995.
- [55] N. Medvidović and R. Taylor. “Software architecture: foundations, theory, and practice”. In: *2010 ACM/IEEE 32nd International Conference on Software Engineering* 2 (2009), pp. 471–472. DOI: 10.1145/1810295.1810435.
- [56] Abha Moitra et al. “Automating requirements analysis and test case generation”. In: *Requirements Engineering* 24 (2019), pp. 341–364.
- [57] Peter Naur and Brian Randell. *Software Engineering: Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct. 1968, Brussels, Scientific Affairs Division, NATO*. 1969.
- [58] Matthew Page et al. “The PRISMA 2020 statement: an updated guideline for reporting systematic reviews”. In: (Sept. 2020). DOI: 10.31222/osf.io/v7gm2.
- [59] PRISMA <http://www.prisma-statement.org/>.
- [60] Shirley Radack. *The system development life cycle (sdlc)*. Tech. rep. National Institute of Standards and Technology, 2009.
- [61] David Ramírez. *Arquitectura Autonómica, Autoensamblable y Autorreplicables para Sistemas IoT*. CINVESTAV Guadalajara, 2020.

- [62] Rajkumar Roy and Sam Brooks. “Self-engineering – Technological Challenges”. In: May 2020, pp. 16–30. ISBN: 978-3-030-46816-3. DOI: 10.1007/978-3-030-46817-0\_2.
- [63] Jaakko Sauvola et al. “Future of software development with generative AI”. In: *Automated Software Engineering* 31.1 (2024), p. 26.
- [64] Ian Sommerville and Sergio Fuenlabrada Velázquez. *Ingeniería de software*. Jan. 2011.
- [65] Roy Sterritt and David Bustard. “Towards an autonomic computing environment”. In: Feb. 2003, pp. 694–698. ISBN: 0-7695-1993-8. DOI: 10.1109/DEXA.2003.1232103.
- [66] A. Taleb-Bendiab et al. “Model-Based Self-Managing Systems Engineering.” In: Jan. 2005, pp. 155–159. DOI: 10.1109/DEXA.2005.137.
- [67] Aline Valente et al. “Analysis of Academic Databases for Literature Review in the Computer Science Education Field”. In: Oct. 2022, pp. 1–7. DOI: 10.1109/FIE56618.2022.9962393.
- [68] Xinyuan Wang, Zhiqiang Meng, and Chang Qing Chen. “Robotic materials transformable between elasticity and plasticity”. In: *Advanced Science* 10.13 (2023), p. 2206637.
- [69] Danny Weyns et al. “Six Software Engineering Principles for Smarter Cyber-Physical Systems”. In: Sept. 2021, pp. 198–203. DOI: 10.1109/ACSOS-C52956.2021.00058.
- [70] Heinz Züllighoven. *Object-oriented construction handbook: Developing application-oriented software with the tools & materials approach*. Elsevier, 2004.