Modelación de sistemas informáticos por representaciones sucesivas

Modeling computer systems by successive representations

Israel Sandoval Grajeda
Instituto de Investigaciones en Matemáticas Aplicadas y en sistemas (IIMAS), UNAM, Ciudad de México, México
israel.sandoval@iimas.unam.mx
ORCID: 0000-0003-0770-5006

https://doi.org/10.36825/RITI.13.31.003

Recibido: Junio 22, 2025
Aceptado: Septiembre 30, 2025

Resumen: El propósito de este documento es proponer una metodología para el desarrollo de software a partir de una serie de abstracciones cuyo resultado sea un sistema informático obtenido a partir de un modelo dado. Esta propuesta surge debido a ciertas problemáticas que surgieron al participar en un proyecto multidisciplinario con el rol de desarrollador de software, donde la tarea principal es analizar el comportamiento de un fenómeno social con una herramienta informática que refleje el comportamiento de un modelo construido para este fin. El trabajo incluye hallazgos, producto de la experiencia de ocho años colaborando en proyectos de desarrollo, cinco de ellos en el proyecto mencionado, así como definiciones de algunos conceptos relacionados con los procesos de modelación y desarrollo, comentadas a lo largo del documento.

Palabras clave: *Modelación de Software, Niveles de Abstracción, Proceso de Solución de Problemas, Modelos y Representaciones*.

Abstract: The purpose of this text is to present a methodology to software development based on abstractions to build an informatic system based on a given model. This idea came to be as a result of certain challenges I faced while working as a software developer on a multidisciplinary project, where the main purpose is to analyze the behavior of a social phenomenon using an informatic tool that could mirror its behavior. This work reunites products of my eight years of experience as a collaborator in projects of software development, five of those in the aforementioned project, this includes my own definitions of concepts related to modeling and development process.

Keywords: *Software Modeling, Abstraction Levels, Problem Solve Process, Models and Representations.*

1. Introducción

Hace aproximadamente 5 años, el proyecto Genealogía de los espacios académicos de la UNAM [1] en el Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS) de la Universidad Nacional Autónoma de México (UNAM), al que se hace referencia simplemente como el proyecto*, entró en una nueva etapa donde uno de los principales objetivos fue generar un producto de software, por lo que se requirió la incorporación de un desarrollador*. La integración de este nuevo rol comenzó con una inducción acerca los conceptos sociológicos

asociados al análisis de la estructura de los espacios académicos de la UNAM a través del tiempo - el objeto de estudio - así como del modelo asociado. El objetivo del proyecto es identificar patrones en los procesos de conformación de nuevos espacios académicos mediante acontecimientos históricos documentados.

En la primera etapa, el equipo solo estaba integrado por dos personas: la titular del proyecto y el desarrollador, por lo que fue un gran reto transformar dicho modelo en un sistema informático, que, desde la perspectiva del proyecto y para propósitos de este trabajo, defino como un producto de un desarrollo de código con entradas y salidas establecidas, que puede ser utilizado por expertos en el objeto de estudio pero no cuenta manuales de uso o documentación del desarrollo, es decir, no tiene la madurez para ser un producto de software.

Algunos de los retos enfrentados por un desarrollador sin ninguna formación en el campo de la sociología fueron participar en las definiciones y cambios en el modelo, pero al mismo tiempo desarrollar funcionalidades detectadas; comprender lo que se esperaba del código y convencer a la titular del proyecto de que las soluciones que serían implementadas en forma de código correspondían a los comportamientos del modelo y cambiar el proceso de construcción de software que había adoptado en proyectos anteriores, ya que hubo desarrollos que se volvieron obsoletos rápidamente.

Los siguientes apartados pretenden reflejar como fue conformada la presente propuesta, rescatando experiencias en el proyecto, enriquecidas con definiciones clave como son los modelos y el método de solución de problemas utilizado en desarrollos científicos. A continuación, se plantea la idea central, que es la modelación enfocada en la construcción de software, la cual articula la experiencia con los conceptos mencionados, complementada con la vigilancia epistemológica como una herramienta de reflexión, pero también de aseguramiento de calidad.

El resultado presentado pretende ser método para la construcción de software partiendo de un objeto de estudio y utilizando como principal herramienta el concepto de abstracción. El potencial de esta propuesta radica en que siempre será posible construir un sistema informático que sea funcional y refleje el comportamiento del modelo en el que está basado, ya que durante la aplicación de la metodología se detectan posibles errores o mejoras.

Dado que la presente propuesta no se ha aplicado de manera rigurosa a otros desarrollos, aún no se han podido definir métricas que permitan evidenciar su alcance o aplicabilidad, sin embargo, durante la ejecución del proyecto se han obtenido algunos resultados y lecciones aprendidas que permitirán establecerlas. Por ejemplo, se ha detectado una relación directa entre el tiempo de desarrollo y la definición del modelo.

2. Metodología

Para generar la presente propuesta, el primer paso fue buscar una forma de trabajo interdisciplinario que permitiera la comprensión de conceptos, evitando ambigüedades en la medida de lo posible; posteriormente, se buscó un marco conceptual para sistematizar el trabajo en el marco de la ingeniería de software, sin embargo, dada la dinámica del proyecto - una investigación incremental y cíclica - no fue posible adaptar ninguna metodología para desarrollo de software, así que se adaptó una estrategia utilizada en proyectos de investigación.

Otro aspecto relevante para desarrollar una propuesta propia fue el trabajo de modelación desde la perspectiva sociológica y la concepción de modelo como una abstracción de un fenómeno que podemos percibir con nuestros sentidos [2].

Finalmente, para articular la propuesta y darle valor para su aplicación se utilizó la vigilancia epistemológica [3] como enfoque teórico, que establece una postura crítica acerca de la validez del método, así como un ejercicio repetitivo e incremental, que ayuda a la clarificación de los conceptos utilizados y la creación de mecanismos para comprobar los experimentos en función de sus entradas y salidas. Esta actividad es importante porque hace entrar a los participantes del proceso en una reflexión constante, permitiendo encontrar posibles errores conceptuales, que son difíciles de detectar y corregir.

2.1. Trabajo interdisciplinario

Desde los primeros trabajos en el proyecto, el primer desafío a solucionar fue la comunicación entre colegas, sobre todo porque el lenguaje utilizado por los sociólogos es muy distinto al de las ciencias de la computación. Lograr tener la mínima comprensión necesaria para construir un sistema informático implicó varias reuniones de trabajo, explorar literatura relacionada con el objeto de estudio y el enfoque teórico de la investigación, pero sobre todo, corroborar la comprensión de los conceptos, debido a que se utilizaban palabras similares con diferente significado, ya sea porque se asocian con un campo semántico distinto o porque se interpretan desde diferentes perspectivas: un

ejemplo muy representativo es la palabra *entidad*, ya que en el modelo definido en el proyecto, representa un espacio académico de la UNAM, como la Facultad de Filosofía y Letras (FFyL) o el IIMAS, pero desde el punto de vista de bases de datos, se puede interpretar como un objeto que puede ser identificado de forma distintiva [4] o una tabla de base de datos.

Una vez comprendidos los conceptos teóricos del proyecto y la metodología de investigación desde la perspectiva sociológica, la siguiente labor fue empezar a obtener requerimientos de software, participando activamente en las definiciones y cambios en el modelo; a la par, se comenzó con el desarrollo de las funcionalidades detectadas para mostrarlas al equipo y corroborar su pertinencia teórica, mostrando los resultados que arrojaba el código y comprobar que el sistema informático reflejaba el comportamiento del modelo. Debido a la dinámica de trabajo del equipo, el proceso de construcción de software sufrió modificaciones ya que hubo desarrollos que se volvieron obsoletos rápidamente debido a cambios en la definición del modelo.

Al combinar las tareas asociadas con la comprensión y definición del modelo con la construcción del sistema informático se utilizaron conceptos de las ciencias de la computación que permitieron transformar las definiciones del modelo en algoritmos, pero de forma desordenada. En 2006, Jeanette Wing define este conjunto de habilidades como pensamiento computacional [5].

Con el propósito de dar orden al trabajo y sustentarlo teóricamente, se utilizó como base una definición de modelo establecida en los albores de la inteligencia artificial y para establecer un método de desarrollo, se adoptó un método de solución de problemas utilizado para desarrollos científicos, ambos especificados a continuación.

2.2. Los modelos como representaciones de objetos

La definición de modelo que será utilizada para este trabajo es la que proporciona Minsky [6], debido a que establece la relación entre los procesos mentales y las herramientas conceptuales que guían el proceso de solución de problemas, específicamente en el campo de la inteligencia artificial, pero es lo suficientemente general para aplicarlo en otras áreas de estudio. La Figura 1 ilustra dicho concepto, donde A es un objeto que existe en el mundo (puede ser percibido con los sentidos) y A^1 es un modelo de A, tal que un observador B puede contestar preguntas acerca de A.

Figura 1. Concepto de modelo según Minsky.

Esta percepción del modelo implica que un observador C no podría interpretar el modelo si no posee el conjunto de conocimientos W^1, es decir, los conceptos y significados relacionados con el modelo.

El modelo se construye a través del planteamiento de experimentos, es decir, preguntas que se buscan responder acerca del objeto de estudio, por lo que se dice que A^1 es un buen modelo de A si B puede observar las mismas entradas y salidas tanto en A como en A^1.

2.3. El proceso de solución de problemas

Pancake y Bergmark [7], plantean un proceso de solución de problemas a partir de 4 pasos (Figura 2) y los definen como subsistemas que operan a distinto nivel de abstracción, lo que implica que los objetos que los conforman y las operaciones son distintos, es decir, tienen un dominio propio. Dicho de otra forma, existe un conjunto de conocimientos a partir del cual se construye cada subsistema o paso.

El primer nivel de abstracción, llamado conceptual, es la definición del dominio del problema (es decir, un modelo conceptual del objeto de estudio), el segundo está definido por objetos que describen la forma en que los datos serán organizados, así como las operaciones de entrada y salida en forma de algoritmos.

El tercer paso es claro para los profesionales de la computación dado que la interpretación de algoritmos y la codificación están muy relacionadas. El resultado del tercer paso es un conjunto de códigos en uno o más lenguajes de programación ejecutados en un ambiente diseñado específicamente para el desarrollo, así como otros objetos para facilitar la entrada, salida y almacenamiento de información. En este paso se toman decisiones que determinan la estructura del código, los conocimientos que deben tener los programadores e influyen en el cuarto paso del

proceso, que es llegar a un (objeto) ejecutable, por ejemplo, un archivo que puede ser leído directamente en un equipo de cómputo o un conjunto de objetos que están preparados interactuar en un ambiente específico.

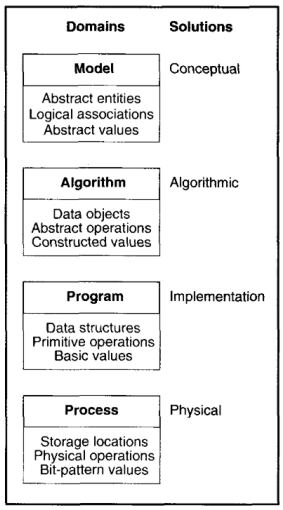


Figura 2. Proceso de solución de problemas según Pancake y Bergmark [7].

2.3. Abstracciones y representaciones

De acuerdo con la definición de modelo expuesta anteriormente, un modelo A^1 representa a un objeto A cuando A^1 conserva las entradas, salidas y experimentos de A o de forma equivalente, que A^1 es una representación de A y dado que, A^1 es un modelo de A, también podemos decir que A^1 es una abstracción de A, como lo define Bailer-Jones [2], es decir, bajo estos contextos es posible establecer una equivalencia entre una representación y una abstracción. Esta afirmación también implica que una representación de A^1 es también un modelo.

Por lo tanto, cada paso en el proceso de solución de problemas es una representación distinta del paso anterior, construida a partir de los elementos de un conjunto W^i, donde i es cada uno de los niveles de abstracción.

2.4. La modelación

Dado que el objeto de estudio del proyecto fue analizado desde una perspectiva sociohistórica, se utilizó el concepto de modelo que el sociólogo Pierre Bourdieu [3] define como un diseño formal de relaciones, cuyas operaciones corresponden a las que podemos percibir con nuestros sentidos al observar un fenómeno u objeto y la modelación es la construcción teórica del objeto empírico.

Sin embargo, cuando se utiliza el proceso solución de problemas, la construcción de las representaciones algorítmica, de implementación y física, son un problema de la ingeniería de software, es decir, cambia el campo de estudio desde donde se realiza la modelación y el concepto adquiere otro significado. Por lo tanto, cuando se realiza la actividad de construir representaciones, el término modelación será entendido como la acción y efecto de construir (o conformar) un objeto basado en otro, ajustándose a ese objeto. La modelación por representaciones simplemente puntualiza que los objetos de los que hablamos son representaciones como las definidas anteriormente.

En general, el proceso inicia con un modelo definido por las personas expertas en el campo de estudio desde el cual se analizará el objeto de estudio. Es decir, no es responsabilidad del encargado de construir el sistema informático construir el modelo conceptual (A^1).

Es importante plantear un par de supuestos con respecto a la representación A^1: el primero es que un modelo podría no estar completamente construido cuando se inicie el proceso, de hecho, se aconseja que sea de esa forma, cómo se justificará a continuación; el segundo es que no se puede asegurar que el modelo sea correcto, es decir, que represente al objeto.

Para lograr representaciones que nos lleven a un sistema informático es necesario establecer un marco conceptual que permita delimitar los alcances de cada representación, buscar que cada conjunto W^i contenga elementos adecuados para la construcción de cada representación, tal que cada una sea más cercana al objeto final deseado y al mismo tiempo, alejarse de los conceptos relacionados con el objeto de estudio.

2.5. La vigilancia epistemológica

Para la construcción de esta propuesta se fueron incorporando los pasos mencionados en las actividades del proyecto y probándolos. En la etapa inicial se desarrolló el sistema informático utilizando solamente los conceptos sociológicos aprendidos; Posteriormente, se incorporó el concepto de modelo y el método de solución de problemas para ir construyendo representaciones parciales del modelo.

Definir el mecanismo de construcción de cada representación, es decir, la modelación, fue una tarea que requirió de una discusión profunda con los colegas sociólogos dado que el concepto desde su enfoque teórico tiene un significado distinto al utilizado en la presente propuesta, como se discutió en el apartado anterior.

Esta reflexión acerca del concepto de modelación es un buen ejemplo del papel de la vigilancia epistemológica como mecanismo de verificación de transferencia de conocimientos, que es uno de los puntos relevantes en la modelación por representaciones, dado que hay una especie de transferencia o mapeo entre conjuntos W^i para lograr que la representación resultante sea capaz de comportarse igual que su predecesora bajo los subsistemas definidos en el método de solución de problemas. La vigilancia epistemológica es una actividad transversal en todo el proceso, ya que es parte de todos los pasos de transferencia, articulación y comprobación en cada representación.

3. Resultado: Modelación por representaciones

Para facilitar la representación esquemática de la metodología, se retomó la notación del modelo de Minsky para nombrar al modelo como A^1, que corresponde al paso 1 del proceso de solución de problema, se etiquetaron los pasos restantes tomando en cuenta que son un modelo que viene del paso anterior, por lo que A^2 corresponde a la representación algorítmica, A^3 a la implementación y A^4 a la física y se asigna un conjunto W^i para cada representación, donde i es cada paso.

También se definen cuatro roles, dado que cada uno cuenta con un conjunto distinto de conocimientos Wi que le permite construir cada representación y, sobre todo, corroborar que representa al objeto de estudio. La persona P es el encargado de modelar A^1, es decir, es el especialista en el objeto de estudio y posee el conjunto de conocimiento W^1; P^A es el encargado de generar la representación algorítmica (A^2) y posee el conjunto W^2; P^C es el desarrollador que construirá A^3 y que posee a W^3 y finalmente, P^D es la persona que generará el objeto ejecutable y posee a W^4.

Para la construcción de representaciones se utilizará un proceso de aproximaciones sucesivas muy similar a la filosofía del método en espiral [8], con algunas variaciones, esquematizado en la Figura 3, el cual comienza con el modelo A^1 y se van construyendo las representaciones hasta llegar a la A^3, siguiendo las líneas curvas oscuras continuas; sin embargo, antes de construir A^4 se comprueba si A^3 representa al objeto de estudio (representada por las líneas discontinuas), labor que realiza P^D junto con P. En caso afirmativo, se procede a construir A^4 y en caso negativo, no se construye A^4, en su lugar se procede a modificar el modelo (representado por las flechas grises), debido a que, cuando A^3 no representa al objeto de estudio, quiere decir que existe un error en el proceso, sin embargo, dado que las representaciones A^2 y A^3 también representan al modelo (A^1), se concluye que viene la definición de A^1.

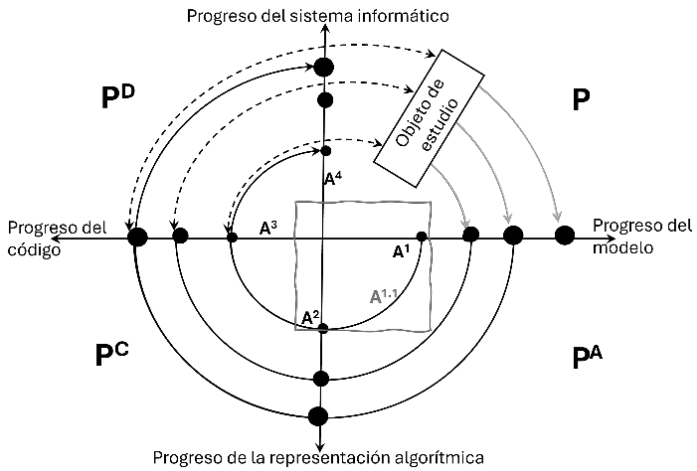


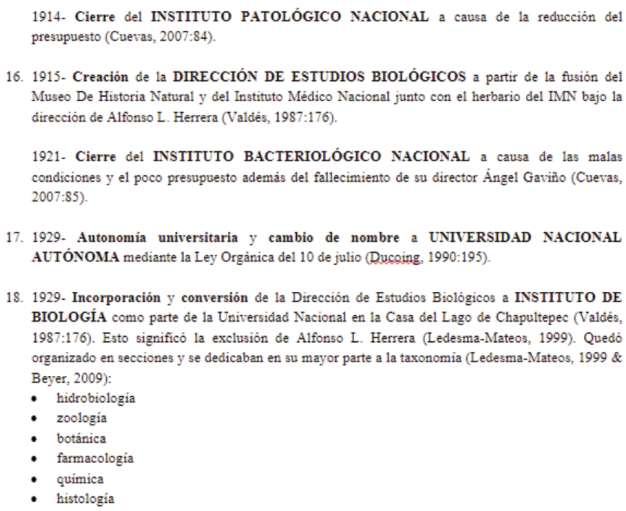
Figura 3. Proceso de modelación por representaciones sucesivas.

La construcción del sistema informático siempre comenzará con alguna modificación en el modelo, pero pueden suceder varias etapas antes de construir la siguiente representación A^4, sin embargo, el método garantiza que cada versión del sistema informático representa al objeto de estudio.

Las flechas rectas que parten del centro representan la trayectoria de cada representación, como una especie de plano cartesiano. Para evitar confusiones en la notación, no se colocaron las siguientes versiones de cada representación, pero se debe asumir que cada vez que las curvas oscuras continuas tocan una flecha, genera una versión distinta de la representación etiquetada, aunque no necesariamente implica que se agregó alguna funcionalidad o elementos, sino que es un refinamiento de la anterior, es decir, si seguimos la flecha Progreso del modelo, cada versión de A^1 representa de manera más cercana al objeto de estudio.

Los cuadrantes representan el área de acción de cada persona en la generación de las representaciones, tal como se mencionó al inicio de este apartado: P^A construirá A^2 a partir de A^1 y P^D será el encargado de construir A^4, es decir, la nueva versión del sistema informático. Las curvas oscuras continuas representan las actividades que son realizadas por profesionales de las ciencias de la computación y las grises, la modelación realizada por P, el experto en el objeto de estudio.

La Figura 4 representa en detalle del paso de A^2 hacia A^3, para lo cual también es necesario el conjunto de conocimientos W^2, a partir de los cuales se comienzan a identificar, por un lado, los elementos que serán transmitidos a W^3 y por otro, los experimentos, entradas y salidas; a continuación, se realiza la articulación de todos los experimentos identificados, generando A^3 en términos de W^3. Como último paso, se comprueba que A^3 es una representación correcta y completa de A^2. Esta labor no es sencilla, debido a que las representaciones operan en niveles de abstracción distintos, por lo que se propone establecer un proceso de vigilancia epistemológica para llevar a cabo de forma congruente estas tareas.



1914- **Cierre** del **INSTITUTO PATOLÓGICO NACIONAL** a causa de la reducción del presupuesto (Cuevas, 2007:84).

16. 1915- **Creación** de la **DIRECCIÓN DE ESTUDIOS BIOLÓGICOS** a partir de la fusión del Museo De Historia Natural y del Instituto Médico Nacional junto con el herbario del IMN bajo la dirección de Alfonso L. Herrera (Valdés, 1987:176).

1921- **Cierre** del **INSTITUTO BACTERIOLÓGICO NACIONAL** a causa de las malas condiciones y el poco presupuesto además del fallecimiento de su director Ángel Gaviño (Cuevas, 2007:85).

17. 1929- **Autonomía universitaria** y **cambio de nombre** a **UNIVERSIDAD NACIONAL AUTÓNOMA** mediante la Ley Orgánica del 10 de julio (Ducoing, 1990:195).

18. 1929- **Incorporación** y **conversión** de la Dirección de Estudios Biológicos a **INSTITUTO DE BIOLOGÍA** como parte de la Universidad Nacional en la Casa del Lago de Chapultepec (Valdés, 1987:176). Esto significó la exclusión de Alfonso L. Herrera (Ledesma-Mateos, 1999). Quedó organizado en secciones y se dedicaban en su mayor parte a la taxonomía (Ledesma-Mateos, 1999 & Beyer, 2009):
- hidrobiología
- zoología
- botánica
- farmacología
- química
- histología

Figura 4. Ficha extensa.

Aunque esta propuesta es teórica, algunos de los elementos mencionadas han sido puestos a prueba y se han obtenido algunos hallazgos, expuestos a continuación, analizando la construcción de cada representación desde A^1 hasta A^4, es decir, los pasos del método de solución de problemas.

El paso más complejo en el proceso de modelación es construir A^2 a partir de A^1, debido a que el modelo inicial está definido en términos de otra área de estudio, lo que implica que los conjuntos W^1 y W^2 son muy diferentes y, por lo tanto, la tarea de transferencia llevará más tiempo. Otro factor que influye es el tamaño del modelo, ya que existen más experimentos por identificar, en conclusión, ente más simple sea el modelo será más fácil construir la primera versión del sistema informático.

Es importante tomar en cuenta que A^2 es una representación de tipo algorítmica, es decir, el conjunto W^2 contiene la especificación de cómo se construyen los algoritmos, la forma en que se tratarán los datos (por ejemplo, la notación del modelo entidad-relación, o bien una especificación para el uso de archivos) y todos los conceptos necesarios para construir A^2. En la práctica, esta tarea no consiste en realizar analogías o sustituciones de objetos de una representación a otra, sino una serie de discusiones para homologar conceptos, aclarar posibles homonimias y en general identificar los conocimientos necesarios para lograr la tarea de articulación en términos de W^2.

Una estrategia que se podría adoptar para simplificar la construcción de una representación es pensar en un paso intermedio, como una representación $A^{1.1}$, que puede derivar directamente del modelo A^1 sin llegar a ser A^2, como se puede ver la Figura 3, donde el conjunto $W^{1.1}$ podría contener una matriz de requerimientos o historias de usuario, así como un conjunto de reglas de conversión a nivel de lenguaje natural, logrando una representación intermedia que simplifique la construcción de una representación, aunque aún no se ha determinado si es posible llegar a este tipo de representaciones de forma simple en todos los casos, dado que el conjunto W^1 podría contener objetos como imágenes o ecuaciones matemáticas que no siempre son fáciles de transformar en lenguaje natural.

En el paso de A^2 hacia A^3, el conjunto W^3 está determinado por las tecnologías elegidas para realizar el desarrollo, inclusive los mapeos entre algoritmos u otros conceptos de A^2, por ejemplo, el manejador de bases de datos o el tipo de archivos utilizados para el almacenamiento. No obstante, la identificación de experimentos y la transferencia de conocimientos es más sencilla para P^C, aún si no está totalmente familiarizado con W^2 y la articulación de A^3 también lo es, por lo que P^C y P^A podrían ser la misma persona.

El paso de A^3 hacia A^4 es inmediato cuando se realiza el desarrollo en un lenguaje de programación que genera un archivo ejecutable al compilar y se almacenan los datos en archivos locales; sin embargo, para la mayoría de las aplicaciones se debe preparar un ambiente de producción, debido a que se utilizan otras tecnologías para almacenamiento, como los motores de bases de datos, para monitoreo de las solicitudes, como los servidores web, entre otras. En este caso, el conjunto W^4 que debe poseer P^D puede contener conocimientos relacionados con herramientas de despliegue automático, entornos de nube, instalación y puesta en marcha de servidores que interpretan código, entre otros. La vigilancia epistemológica es una actividad transversal en todo el proceso, ya que es parte de todos los pasos de transferencia, articulación y comprobación en cada representación, así como la comprobación de A^3 con el objeto de estudio.

Un ejemplo parcial de este proceso se puede ver en [9] donde se detalla uno de los principales elementos del modelo, que son los indicios, que, a grandes rasgos, son acontecimientos históricos que registran la fecha, el lugar, la acción, la fuente documental y el lugar donde ocurre. Al listado de indicios que ocurren en el mismo lugar se le llama ficha extensa, como se puede ver en la Figura 5. El documento también narra el progreso del indicio como parte del modelo, sin embargo, este ejemplo sólo reflejará la versión inicial.

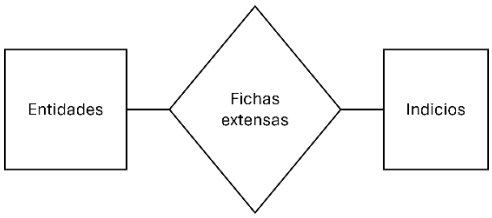


Figura 5. Modelo entidad relación para la ficha extensa.

Estrictamente, podemos tomar el párrafo anterior como una definición simplificada de A^1. El conjunto W^1 está conformado por las teorías y conceptos que conforman el modelo, por ejemplo el concepto de campo de Bourdieu y el enfoque bibliográfico de Bertaux. los cuales se citan en [9].

Una vez comprendidos los conceptos anteriores y el enfoque del proyecto, se generó A^2, definiendo un modelo entidad relación para almacenar en una base de datos los acontecimientos (Indicios), los lugares donde suceden (Entidades) y las fichas extensas se obtienen cuando se consultan los indicios de una entidad (Figura 6). Este enunciado trae consigo una operación que se transformó en algoritmos, específicamente, en una consulta de base de datos. En este paso, W^2 está conformado principalmente por la teoría asociada al modelo relacional y la especificación de cómo se construirán los algoritmos. En el proyecto, el sistema debe reflejar el comportamiento del modelo, pero también debe ser útil para seguirlo construyendo, por lo que se agregaron otros requerimientos como las operaciones de alta, baja y modificación de indicios y entidades.

La construcción de A^3 ya no se especifica en la referencia mencionada, pero al inicio del proyecto se tomó la decisión de realizar el desarrollo en PHP, por lo que W^3 está constituido por las referencias a dicho lenguaje y al motor de base de datos MYSQL 8.0. En este ejemplo, A^4 está compuesta por un conjunto de archivos en PHP y un archivo con la definición de la base de datos, donde se incluyen datos del servidor para realizar la conexión, los cuales son particulares de la configuración del servidor que interpretará el código. Esto quiere decir que se debe configurar el ambiente de producción antes de finalizar la codificación, generando una dependencia entre el código y el ambiente de producción.

Finalmente, W^4 está compuesto por la documentación acerca del ambiente de producción, como la configuración del servidor web, así como la forma en que se deben ingresar los parámetros en el código para acceder a la base datos una vez que se reciban peticiones y el código sea interpretado.

4. Discusión

Los modelos son útiles para comprender los fenómenos que suceden en el mundo, ya que permiten conocer algún aspecto en particular a partir de una representación con un propósito definido, enfocada en algunos elementos, estructuras y/o comportamientos que son interesantes y dejando fuera otros, reduciendo su complejidad para facilitar su análisis y así lograr identificar acciones que suceden ante ciertas entradas y que arrojan salidas similares al objeto de estudio. Esta simplificación inherente a los modelos implica que no se comportarán exactamente como el objeto original, por lo que es importante establecer diferencias y limitantes con respecto al objeto modelado.

Dado que las representaciones son construidas como modelos de otros más complejos, es importante notar que conforme las representaciones se van acercando a la A^4 (el sistema informático) el nivel de complejidad va disminuyendo, debido a que los conocimientos en cada conjunto W^i son cada vez más más específicos conforme i aumenta, pero también significa que cada vez es más fácil definir la representación siguiente.

La disminución en la complejidad de cada representación se debe también al lenguaje utilizado para su construcción: un modelo conceptual se define con lenguaje natural, ecuaciones o inclusive esquemas, sin ninguna regla establecida; la representación algorítmica contiene reglas que rigen la sintaxis de los algoritmos y otras que permiten aumentar su expresividad, por ejemplo, operaciones aritméticas o de bases de datos; la implementación está totalmente regida por las reglas estrictas de un lenguaje de programación y finalmente la representación física en muchas ocasiones es producto de un proceso de compilación o reinterpretación del código, por lo que las restricciones para su generación son muy estrictas.

Por otro lado, al aumentar la complejidad en la definición del modelo A^1, también se deben utilizar una mayor cantidad de conceptos para crear las siguientes representaciones, es decir, crece el conjunto W inclusive, parte de los elementos del conjunto son a su vez modelos, es decir, un modelo explicando algún aspecto de otro. Ludewig [10] establece una discusión muy útil acerca de lo que se puede considerar un modelo y algunos términos que suelen usarse cómo sinónimos, pero no lo son: Por ejemplo, un virus informático es una metáfora de los virus biológicos, pero no son un modelo, mientras que las teorías son modelos de fenómenos observables.

Los conjuntos W^i están formados por conceptos y artefactos de ingeniería de software, pero también pueden contener otros modelos, lo que aumenta la complejidad de la representación. En el proyecto, para definir la representación algorítmica A^2 se utilizó el modelo entidad – relación para representar la forma en que se guardarán los objetos dentro de la base de datos.

En este trabajo, el término complejidad y sus derivados se utiliza para comparar la construcción de representaciones. Por ejemplo, se puede decir que es menos complejo construir la representación A^3 que A^2, es decir, que puede llevar menos tiempo, esfuerzo, o inclusive una combinación de ambas.

Otro factor que influye en la complejidad es que la persona encargada de construir el modelo posea (o no) el conjunto de conocimientos W^i. Un problema asociado es que se desconozcan los elementos de dicho conjunto, por ejemplo, podría suceder que una persona P^A (analista) que posee los conocimientos de W^2 podría no estar familiarizado con los de W^1, lo que dificulta la transferencia de conocimientos del conjunto W^1 al conjunto W^2 si quisiera realizar dicha actividad por su cuenta.

Adicionalmente, los procesos de desarrollo de software en general se han vuelto más complejos a través del tiempo no solo por el producto a construir, sino por el surgimiento de tecnologías que permiten automatizar cada paso parcial o totalmente, por lo que también existe una especialización en los roles de las personas involucradas, dando como resultado que exista un especialista para cada uno de los pasos, aunque en proyectos de desarrollo pequeños o especializados estos roles los asume una persona, lo cual implica que la elección de tecnologías para el desarrollo y la implementación influyen en la complejidad de ciertas representaciones.

Además de la complejidad, también se debe asegurar que no exista pérdida de elementos en la construcción de una representación a partir de otra, por lo que siempre debe existir un mecanismo para corroborar que la representación sea completa (que no se hayan perdido elementos) y correcta (que los experimentos arrojen las mismas salidas ante las mismas entradas), además de un proceso efectivo de transmisión de conocimientos, es decir, usar la vigilancia epistemológica en la construcción de representaciones.

5. Conclusiones

Aunque la metodología propuesta está motivada por el trabajo de investigación, se puede aplicar a cualquier proyecto de desarrollo si se toma un proceso o alguna idea con potencial de comercialización - como los procesadores de palabras, las aplicaciones de *streaming* o las redes sociales - como un objeto de estudio. Particularmente, se puede aplicar en proyectos donde aún no se conoce como será la versión final y se va construyendo a través del tiempo.

Uno de los aspectos que se desprenden del planteamiento de la metodología es la delimitación de roles en el proceso, desde el modelado hasta la puesta en marcha del sistema informático. Es muy común es creer que la persona P^A (analista que realiza la representación algorítmica) y P^C (el desarrollador) poseen el mismo conjunto de conocimientos es decir, que puede ser la misma persona, pero este proceso plantea que esto no es necesariamente correcto, debido a que trabajan en un nivel de abstracción distinto, es decir, un analista tiene conocimiento de análisis de algoritmos, de bases de datos, modelos orientados a la construcción de software, entre otros, mientras que un programador conoce de bibliotecas, sintaxis de lenguaje, y todo lo relacionado a las tecnologías con las que se codifica.

Existe la expectativa de que un desarrollador puede analizar un problema y convertirlo en un algoritmo o es capaz optimizar una consulta de bases de datos y esto no es correcto. También existe la creencia de que un programador puede implementar el código compilado en un ambiente de producción, pero eso tampoco se cumple.

Una de las principales motivaciones para utilizar representaciones es aprovechar los conceptos y los objetos de ingeniería de software, ya que, de no existir, no sería posible concebir un método de esta naturaleza, debido a que los conjuntos W^i requieren contener objetos que puedan ayudar a construir las representaciones.

En la metodología propuesta, los experimentos no cambian durante la construcción de las representaciones de A^1 hasta A^4 y así debe ser debido a que cada paso representa al modelo en el que se basa el objeto de estudio, por lo que los cambios en las salidas, entradas y experimentos se deben realizar siempre en A1, pero también cualquier corrección, no importa en que representación se detecte algún error.

6. Trabajo futuro

Dado que el planteamiento metodológico proviene de un proyecto solamente, hace falta realizar otros desde cero, con modelos de distinta naturaleza y complejidad, así como algunas aplicaciones que puedan tener potencial de comercialización, convirtiendo el problema base en un modelo.

Es muy común que en los proyectos de investigación se transforme el modelo directamente en código, al menos de forma parcial ya que sí se analizan los algunos algoritmos que son importantes para la investigación, pero podrían existir elementos no representados y, por tanto, no se podría garantizar que la representación creada represente al modelo, aunque tampoco se puede aseverar que ésta forma de trabajo realmente excluye elementos, por lo es importante llevar a cabo un estudio de casos en proyectos donde se utilicen modelos como herramienta metodológica.

Otro tema por discutir es el de los niveles de abstracción aplicados en la construcción de las representaciones, ya que incide directamente en la definición de los conjuntos W^i. El primer paso sería definir el significado del concepto nivel de abstracción, sus alcances y cómo se delimita cada uno en el proceso de solución de problemas, para llegar también a una definición de complejidad que sea útil para medir el esfuerzo de construcción de una representación.

También se buscará realizar un estudio de distintos modelos y artefactos utilizados en ingeniería de software como los diagramas de flujo; diagramas de casos de uso; patrones de software, entre otros, con la finalidad de clasificarlos en algún nivel de abstracción. Esta labor llevará tiempo debido a que existe una gran variedad de artefactos que se pueden utilizar, aunque el propósito no es generar conceptos nuevos, sino lograr establecer un uso para objetos existentes y agruparlos en los conjuntos W^i, permitiendo identificar a que nivel de abstracción pertenecen y al mismo tiempo, establecer un método para documentar las representaciones.

Establecer niveles de abstracción para las representaciones permitirá definir equivalencias, complementos y quizá hasta posibles mapeos entre objetos, de tal forma que el proceso sea confiable mediante la vigilancia epistemológica, resolviendo, por lo menos de forma parcial, algunos de los problemas que puedan surgir. Por ejemplo, no se puede afirmar que el proceso arroja siempre las mismas representaciones partiendo del mismo modelo, es decir, no se garantiza el determinismo.

Una situación frecuente durante en el proyecto fue encontrar errores durante la implementación (paso 3 del proceso de solución de problemas), los cuales, en su mayoría, venían de la definición del modelo, no obstante, las consecuencias fueron la corrección de códigos, cambios en la funcionalidad y hasta modificaciones sustanciales en el modelo de datos, provocando estructuras de datos que se desechan, códigos que se abandonan y varias correcciones sobre el producto terminado y validado , lo que incrementa el esfuerzo en la construcción del sistema informático, pero en compensación, existe la certeza de que cada versión generada representa al objeto de estudio. Este escenario no es nuevo en el campo de la ingeniería de software, debido a que existen muchos sistemas que van cambiando cuando sus resultados no son los esperados, pero las metodologías contemplan como mitigar o eliminar este tipo de inconvenientes. En la metodología aún no se tiene un mecanismo de mitigación o eliminación para este problema.

Hasta ahora, la vigilancia epistemológica una tarea teórica que implica un ejercicio de discusión y reflexión, sin embargo, en la metodología se ha integrado mediante actividades en la construcción de una representación. Se buscará establecer algún tipo de métrica que permita reflejar que el proceso se ha realizado correctamente, pero más importante, que ayude a comprobar que una representación sea correcta y completa.

Una posible aplicación de este estudio es la posibilidad de generar software de forma automática, ya que, generar representaciones con un nivel similar de abstracción, podría facilitar la construcción de herramientas que puedan llevar de una representación a otra con su debida verificación, garantizando que sean correctas y completas. Un ejemplo orientado en la misma dirección es [11], el cual utiliza herramientas de inteligencia artificial, pero establece su base en un metamodelo, que al final convierten en una herramienta que genera software y pruebas unitarias.

7. Agradecimientos

A Susana Inés García Salord por ser mi mentora en la práctica de la vigilancia epistemológica, por identificar mis hallazgos y brindarme el apoyo para su publicación y difusión. También para Ana María Medeles Hernández por sus comentarios y recomendaciones que ayudaron a mejorar el escrito.

8. Referencias

[1] Immas. (2025). *Genealogía de los espacios académicos de la UNAM*. https://mmss.iimas.unam.mx/proyectos/genealogia-de-la-unam/

[2] Bailer-Jones, D. M. (2009). *Scientific models in philosophy of science*. University of Pittsburgh Press.

[3] Bourdieu, P. (2008). *El oficio de sociólogo: presupuestos epistemológicos* (2ª Ed.). Siglo XXI.

[4] Chen Pin-Shan, P. (1976). The Entity-Relationship Model—toward a Unified View of Data. *ACM Transactions on Database Systems (TODS), 1* (1), 9–36. https://doi.org/10.1145/320434.320440

[5] Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49* (3), 33–35. https://doi.org/10.1145/1118178.1118215

[6] Minsky, M. (1965). *Matter, Mind and Models*. https://dspace.mit.edu/handle/1721.1/6119

[7] Pancake, C. M., Bergmark. D. (1990). Do Parallel Languages Respond to the Needs of Scientific Programmers? *Computer, 23* (12), 13–23. https://doi.org/10.1109/2.62090

[8] Boehm, B. W. (1988). A Spiral Model of Software Development and Enhancement. *Computer, 21* (5), 61-72. https://doi.org/10.1109/2.59

[9] García Salord, S., Sandoval Grajeda, I. (2024). *Los indicios: piezas clave en la modelación sociológica y computacional del patrón de reproducción de los espacios académicos mediante la indagación genealógica*. XVII Congreso Nacional de Investigación Educativa. Villahermosa, Tabasco. https://www.comie.org.mx/congreso/memoriaelectronica/v17/doc/1529.pdf

[10] Ludewig, J. (2003). Models in software engineering - an introduction. *Software and Systems Modeling, 2* (1), 5–14. https://doi.org/10.1007/S10270-003-0020-3

[11] Rajbhoj, A., Somase, A., Kulkarni, P., Kulkarni, V. (2024). *Accelerating Software Development Using Generative AI: ChatGPT Case Study*. 17th Innovations in Software Engineering Conference (ISEC). Bangalore, India. https://doi.org/10.1145/3641399.3641403

