VOL 1 Nº 1 Marzo - Agosto 2020 ISSN Nº 2708-0935

Innovación y Software

Facultad de Ingeniería
Universidad La Salle, Arequipa, Perú
facin.innosoft@ulasalle.edu.pe
https://revistas.ulasalle.edu.pe/innosoft

Vol. 1 N°. 1 2020 Marzo - Agosto

ISSN N°: 2708-0935

DOI: 10.48468/innosoft.s1

ARK: ark:/42411/s1

PURL: 42411/s1

Depósito Legal:2023-08884

Periodicidad: Semestral

Publicado: 30/03/2020

Editado por:

Universidad La Salle
RUC: 20456344004
Av. Alfonso Ugarte N° 517, Cercado, Arequipa

COMITÉ EDITORIAL

EDITORIAL

ARTÍCULOS CORTOS

ARTÍCULOS ORIGINALES

La Revista Innovación y Software (InnoSoft) de la Facultad de Ingeniería, en la Universidad La Salle, tiene el privilegio de divulgar el conocimiento que se genera y enseña en nuestra Universidad y otras Instituciones de Educación Superior en el ámbito de la comunidad peruana y Latinoamericana. La revista, desde sus líneas de trabajo ofrece a investigadores, empresarios, estudiantes y público en general investigaciones sobre: ingeniería, gestión y calidad de software; técnicas de programación; tecnologías de bases de datos; inteligencia artificial; procesamiento de imágenes; redes y seguridad informática; tecnologías de la información y las comunicaciones; entre otras áreas de la Ciencia de la Computación y la Informática.

InnoSoft se planifica, en el menor tiempo posible, ingresar a bases de datos internacionales, lo que le dará una amplia visibilidad y al mismo tiempo un reconocimiento a su relevancia científica; esto también constituye un reto para paulatinamente incrementar su indexación y el índice de impacto. Esta primera edición, persigue no solo la publicación de distintos y variados artículos sobre los escenarios con que en la actualidad se mueve el mundo, además ofrece ser una ventana por donde se vislumbren las voces, así como las venideras contribuciones sobre determinados temas, cuyo sentido está siempre atento ante las demandas de aquellos que intentan desplegar sus valiosos y siempre oportunos aportes al conocimiento en este campo. Para facilitar la lectura de esta edición, es oportuno presentar las cinco contribuciones que la conforman:

Primer artículo: “Optimización de los procesos de producción en la industria de la confección de prendas de vestir utilizando simulación de eventos discretos”. Este es un Artículo Corto que presenta un estudio de simulación del proceso de confección de prendas de vestir de una pequeña empresa textil con el fin de proponer una mejor distribución de los recursos a través de experimentar con un modelo de simulación del proceso. Constituye un aporte para la toma de decisiones basada en información precisa y confiable del proceso productivo. Los resultados obtenidos muestran que realizando cambios en asignar tareas y recursos se reduce el tiempo de producción de un lote en una hora y el tiempo promedio que una unidad permanece en el sistema.

Segundo artículo: “QuantityEr: An extensible and simple solution to obtain the amount of results of complex queries to GitHub”. Este es un Artículo Original que describe una solución extensible y orientada a objetos para obtener la cantidad de resultados de búsqueda de consultas complejas a GitHub utilizando el principio de inclusión-exclusión. GitHub es una plataforma que proporciona alojamiento para el control de versiones de desarrollo de software y cuenta con una enorme cantidad de información que es útil para la comunidad de código abierto. Sin embargo, el motor de búsqueda posee restricciones que hacen imposible emitir consultas complejas a la plataforma. El artículo presenta el modelo matemático, el diseño de la aplicación, las herramientas utilizadas y los resultados de ejemplos de ejecución.

Tercer artículo: “Aplicación de métodos numéricos utilizando MATLAB al modelado del fármaco AZT y la supervivencia con SIDA”. Este es un Artículo Original que muestra un estudio para la resolución de ecuaciones diferenciales aplicado en el modelado de un caso concreto en el sector

Biofarmacéutico. El problema se basa en el impacto de la cidovudina (AZT) sobre la supervivencia al Síndrome de Inmunodeficiencia Adquirida. Para la solución de este modelo se cuenta con una ecuación diferencial ordinaria de primer orden con valores iniciales sobre la cual se aplica el método de separación de variables para obtener la solución real de forma analítica. Se aplican tres métodos numéricos (Euler, Euler Mejorado y Runge Kuta 4) usando el asistente matemático MATLAB para calcular las soluciones aproximadas y se muestran los resultados obtenidos.

Cuarto artículo: "Simulación del proceso de desarrollo de software: una aproximación con Dinámica de Sistemas y el Método de Larman". Este es un Artículo Original, en el cual se presenta el modelado y simulación de un proceso de desarrollo de software por medio del enfoque de la Dinámica de Sistemas, que permite representar gráficamente los elementos intervinientes en el proceso e incorporar la cantidad relevante de parámetros involucrados. Se tomaron como referencia el modelo de estimación de costes COCOMO y método de Larman. El modelo de simulación presentado permite hacer estimaciones iniciales del comportamiento del proceso software, y de los elementos que lo conforman, durante el transcurso de un tiempo de simulación configurable. El modelo deriva en una herramienta de soporte a los equipos de gestión, y a las empresas que hacen de la Gestión de Proyectos Tecnológicos su negocio principal.

Quinto artículo: "Componente de indexación de huellas dactilares basado en características globales". Este es un Artículo Original que presenta algoritmos basados en el campo de orientación de la huella dactilar para hacer una búsqueda en un sistema de identificación. Se emplean grafos relacionales de atributos y máscaras dinámicas que permiten extraer un vector de características robusto e invariante a la traslación y rotación. Además, se utilizó un algoritmo genético de nuevo tipo y una estrategia dinámica de búsqueda en árbol para efectuar la comparación inexacta entre dos grafos relacionales de atributos dados. Para la abstracción de las máscaras dinámicas se empleó un algoritmo de triangularización de polígonos que permitió una correcta aproximación de los vértices de la máscara en un tiempo eficiente. Para corroborar los resultados obtenidos fueron empleados los bancos de datos aportados por Fingerprint Verification Competition.

En este momento inicial hacemos un reconocimiento a los Directivos de la Universidad La Salle que hacen posible con su apoyo y financiamiento el proceso de edición. Señalamos también, que la revista es y debe seguir siendo parte de la comunidad académica, y es a esa comunidad a quién le corresponde, proponer nuevas líneas editoriales que ayuden a conservar la trayectoria de InnoSoft. Finalmente, desde el Comité Editorial de la revista InnoSoft exhortar a los autores para que contribuyan a este empeño, con el envío de sus contribuciones, así como a los lectores para que hagan llegar al Comité Editorial sus valiosas sugerencias, en aras del fortalecimiento de la investigación y el conocimiento científico y dinámico.

Comité Editorial

Optimización de los procesos de producción en la industria textil utilizando simulación de eventos discretos

Optimization of production processes in the textile industry using simulation of discrete events

Luis Alfredo Aragón Guía
Universidad Tecnológica del Perú
aqpmjinnova@gmail.com
https://orcid.org/0000-0002-5915-8174

Yordi Jesús Díaz Callo
Universidad Tecnológica del Perú
ydiaz0402a@gmail.com
https://orcid.org/0000-0002-2503-8297

Marilyn Fabiola Juarez Flores
Universidad Tecnológica del Perú
e19100531@postgradoutp.edu.pe
https://orcid.org/0000-0002-3149-0263

RECIBIDO 15/11/2019 • ACEPTADO 20/12/2019 • PUBLICADO 30/03/2020

RESUMEN

Es importante para pequeñas empresas textiles que están creciendo tomar decisiones clave basadas en información precisa y confiable del proceso productivo presente y cómo los cambios futuros podrían afectarlas. Utilizando métodos matemáticos tradicionales no se logra reflejar de manera precisa y confiable el comportamiento real estocástico de procesos y actividades realizadas en esta industria. Sin embargo, existen tecnologías de información como la técnica de simulación por ordenador que permite obtener dicha información tomando en cuenta este comportamiento. La investigación realizó un estudio de simulación del proceso de confección de prendas de vestir de una pequeña empresa textil con el fin de proponer una mejor distribución de los recursos a través de experimentar con un modelo de simulación del proceso, además se presenta parte de la información cuantitativa que se obtiene con esta técnica. Los resultados obtenidos muestran que realizando cambios en asignar tareas y recursos se reduce el tiempo de producción de un lote de 120 unidades en una hora y se reduce aproximadamente 30 minutos el tiempo promedio que una unidad permanece en el sistema. Toda la información se obtuvo mediante un modelo de simulación verificado y validado sin la necesidad de experimentar dichos cambios con el sistema real.

Palabras claves: Simulación, textil, eventos discretos.

ABSTRACT

It's important for small textiles companies to take key decisions based on accurate and confidence information from the productive present process and future changes could they affect it. Using traditional maths methods it doesn't reflect the real process behavior of activities from that industry. Although there are information technologies as simulation techniques by computers which permit obtain this information taking into account this behavior. This research performed a study of clothing process simulation from a small textile company in order to propose a better distribution of resources through experiencing with a process simulation model also it presents a part of quantitative information which is obtain with that technique. The results from that tests

obtained show changes in assignment tasks and resources that reduce production time of a block of 120 units in an hour and reduce approximately 30 minutes the average time that unit remains in the system. All this information was obtained through a simulation model verified and validated without experiment changes with the real system.

Keywords: *Textile, simulation, discrete events.*

INTRODUCCIÓN

Una de las principales fuentes de empleo en el Perú es el sector textil y confección que representa el 8.9 % del total de la población económicamente activa (PEA). Gran parte de este empleo es generado por micro, pequeñas y medianas empresas (MIPYMEs) que contratan cerca de 302 mil trabajadores solamente en el rubro confecciones [1].

En [2] presenta cómo la contribución al valor agregado manufacturero es mayor en sectores productivos donde el índice de adopción-uso de tecnologías de información y comunicación (TICs) es alto. El sector textil y confección poseen índices de 0.69 y 0.52. Incrementar este índice generará que la productividad de las MIPYMES peruanas aumente.

Las TICs realizan la tarea de procesar información de manera rápida y confiable para la toma óptima de decisiones [3]. Dentro de las TICs se encuentra a la Simulación de Eventos Discretos (SED). Esta técnica es reconocida en [4] como la herramienta de investigación de operaciones más frecuentemente utilizada en industrias de manufactura, finanzas, salud, entre otras. Una encuesta realizada muestra en [5] que el modelado de simulación es usado en todos los niveles gerenciales de las 500 corporaciones más grandes en Estados Unidos. Además, donde la simulación se utilizó, se ahorró entre 5 % y 10 % del costo de capital.

La SED representa cuantitativamente sistemas del mundo real, simula sus dinámicas sobre una base de evento por evento y genera información detallada sobre su desempeño. Utilizándola se busca mejorar la productividad a través de rendimientos más altos, leads times más cortos, trabajos en proceso bajos y alta utilización de recursos. La SED evalúa, sin experimentar con el sistema real, el comportamiento de un proceso de manufactura bajo diferentes grupos de condiciones; analiza escenarios para identificar mejores condiciones físicas y políticas operacionales [5].

A partir de esta introducción, el artículo se basa en el estudio de SED desarrollado en la línea de confección de prendas de una pequeña empresa textil de Arequipa-Perú. Este trabajo tiene 4 secciones. La siguiente sección presenta el resumen de la metodología utilizada. La tercera sección presenta los resultados y discusión del estudio. Finalmente, la cuarta sección presenta las conclusiones.

MATERIALES Y MÉTODOS O METODOLOGÍA COMPUTACIONAL

El estudio de SED se desarrolló bajo la metodología propuesta en [6] que consta de 8 pasos presentados en la Figura 1.

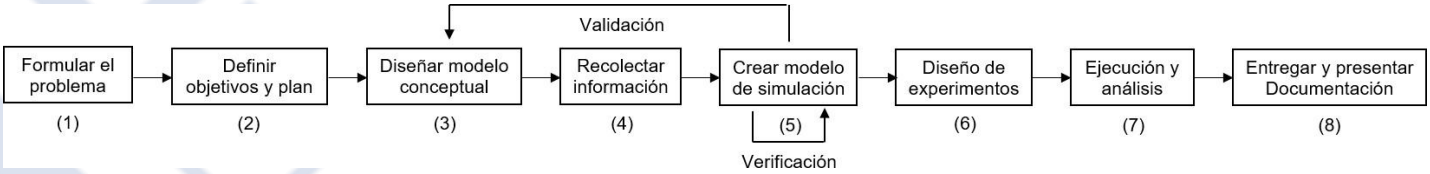

Figura 1. Pasos en un estudio de simulación

Mediante entrevistas al gerente, supervisor y visitas al proceso se evidenció el bajo nivel de utilización de recursos (operarios), cuellos de botella, tiempos de ciclo y tiempos totales de producción fuera del límite teórico establecido. Estos factores ocasionan la baja productividad en la línea y retrasos en la entrega de pedidos. El problema formulado nació por la necesidad del gerente de saber: ¿Cuánto va afectar a los factores de productividad el realizar cambios en la asignación de tareas y distribución física del personal? El objetivo es medir el impacto que tendrán los cambios propuestos en la productividad de la línea.

El modelo conceptual de la Figura 2 se diseñó recopilando información de la lógica y secuencia de operaciones del proceso para confeccionar un pedido de 120 polos cuello camisa. Este modelo establece que la línea trabaja con 13 operarios (cada letra de la Figura 2 representa un operario) distribuidos en 17 estaciones, algunos operarios trabajan en más de una estación. Se recolectó la información mediante un estudio de tiempos don- de se tomaron 100 muestras de tiempos por cada operación para generar con el analizador de datos Input Analyser distribuciones de probabilidad por cada operación. Cada distribución pasó la prueba Chi-cuadrado y Kolmogorov-Smirnov (K-S) con un p-value mayor a 0.05 (5 %) que indica un buen ajuste de los datos con la distribución [7]. La Tabla 2 muestra las distribuciones de probabilidad por cada operación.

El modelo de simulación se programó en el Software Arena versión libre. La verificación busca que el modelo se comporte como se planeó y la validación es asegurar que se comporta de la misma forma que el sistema real [8]. Ejecutando una réplica donde una sola entidad fluye por el modelo se comprobó que la lógica y datos son correctos. Para Validar el modelo se utilizó una prueba de hipótesis que comparó datos reales contra datos generados por el modelo [9]. La variable de comparación fue el número de polos cuello camisa producidos en un turno (8 horas). Se ejecutaron 10 réplicas del modelo y se comparó con los datos reales. Hechas las pruebas se concluyó que el modelo es válido. Con un modelo verificado y validado se planteó experimentar el escenario de mejora de la Figura 2 que mantiene la misma cantidad de recursos (13 operarios) pero se modifica el ordenamiento físico y la asignación de ciertas operaciones.

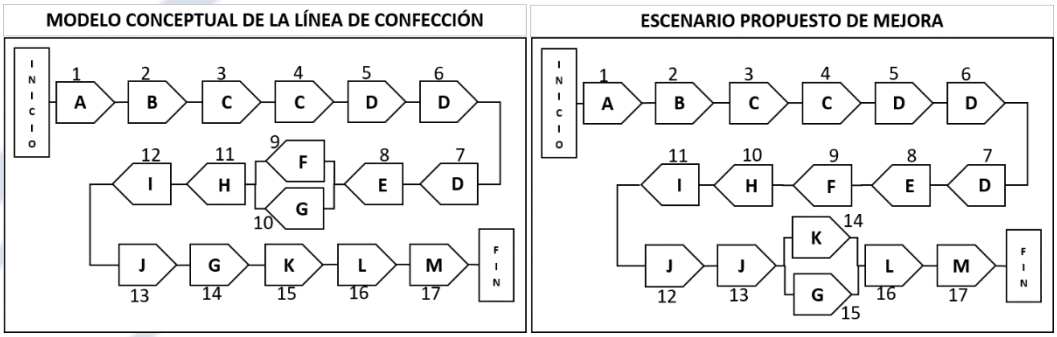

Figura 2. Distribución y flujo en la línea de confección

RESULTADOS Y DISCUSIÓN

Se realizó 35 réplicas independientes del modelo de simulación del sistema actual y modificado (escenario de mejora) para evaluar el desempeño de la línea en producir un pedido de 120 polos cuello camisa. Variables como tiempo de ciclo promedio, tiempo en el sistema promedio y tiempo total de producción promedio y sus intervalos de confianza (IC) al 95 % se muestran en la Tabla 1, además del valor mínimo, máximo y % de reducción entre escenarios. El Escenario propuesto de mejora reduce en 12 %, 19 % y 10 % el tiempo promedio de ciclo, en el sistema y total de producción respectivamente.

Tabla 1. Comparativo de las variables de estudio entre escenarios (tiempo en minutos)

	Escenario Actual				Escenario de mejora				% de reducción
	Promedio	95 % IC	Mínimo	Máximo	Promedio	95 % IC	Mínimo	Máximo	
Tiempo de ciclo	4.43	(4.42,4.44)	1.78	53.23	3.91	(3.90,3.92)	1.62	53.63	12 %
Tiempo en el sistema	172.67	(171.23,174.11)	48.86	304.12	139.74	(138.28,141.20)	49.92	242.23	19 %
Tiempo total de producción	580	(578.93,581.07)	573	587	520	(518.54,521.46)	512	531	10 %

CONCLUSIONES

A través del modelo de simulación generado esta pequeña empresa textil podrá experimentar otras configuraciones de trabajo, medir el impacto productivo de adquirir nuevos activos (maquinaria), de contratar nuevo personal o de experimentar con la fabricación de nuevos productos, entre otras situaciones con el fin de evaluar que escenarios son los más adecuados para incrementar la productividad.

Tabla 2. Distribuciones de probabilidad por cada operación

Código	Operaciones	Distribución (seg.)
A	Pegar pechera SET ON listado+Revisión	NORM(145, 25.8)
B	Acomodar prenda y pechera +Marcar para embolsar	40 + 102 * BETA(1.03, 1.33)
B	Embolsar pechera superior SET ON base + Revisión	NORM(85.7, 23.5)
C	Igualar hombros con tijera	TRIA(13.5, 27, 40.5)
C	Recortar abertura de pechera + Piquetear + Recortar pechera superior SET ON + Voltear prenda	35.5 + WEIB(22.3, 2.49)
C	Embolsar puntas de cuello + Revisión	27.5 + WEIB(18.5, 2.64)
C	Unir hombros listado con tira	NORM(24.2, 1.07)
D	Pespuntar hombros con cadeneta	NORM(16.8, 2.26)
D	Pespuntar hombros con cadeneta especial	10.5 + WEIB(6.51, 2.56)
D	Corte de pespunte de cadeneta	2.5+11* BETA(7.19, 6.91)
D	Marcar centro de cuello	4.5 + WEIB(4.3, 2.76)
D	Pegar cuello box con cinta	54.5 + 80 *BETA(0.901, 0.945)
E	Recortar cinta de cuello + Marcar para pegado de etiqueta	NORM(40, 3.74)
E	Asentar cinta cuello box con 2 etiquetas	NORM(119, 15.5)
F	Preparar + Asentar + Pespuntar pechera SET ON	NORM(213, 29.1)
G	Preparar + Asentar + Pespuntar pechera SET ON	155 + 103 * BETA(1.14, 1.12)
H	Atracar pechera listado interior + Recortar	25.5 + WEIB(27.3, 2.51)
H	Atracar pechera rectangular SET ON	NORM(120, 25.2)
I	Preparacion y marcado de recubre	39.5 + 34 * BETA(1.07, 0.858)
I	Recubrir basta faldon listado fino long tail	49.5 + 64 * BETA(0.918, 0.837)
I	Recortar basta faldon	14.5 + 10 * BETA(0.819, 1.14)
J	Preparar el pegado de mangas	NORM(24, 7.53)
J	Pegar manga corta listado fino	NORM(98.5, 8.98)
G	Recubrir sisa manga corta	47.5 + 15 * BETA(1.02, 1.17)
K	Cerrar costados manga corta listado fino long tail	TRIA(211, 248, 275)
L	Fijar costados + Piquetear costados	(NORM(13.3, 2.75))+ (NORM(13.3, 2.75))
L	Formar pinzas	(19.5 + 28 * BETA(4.78, 5.15))+ (19.5 + 28 * BETA(4.78, 5.15))

REFERENCIAS

[1] J. M. G. Carpio, "Estudio de investigación del sector textil y confecciones," 2015-12.

[2] P. PRODUCE, "Boletín de producción manufacturera, marzo 2017," 2017.

[3] P. E. de la Situación Actual, "de las empresas peruanas [en línea] ministerio de la producción: Perú. recuperado de (2015)."

[4] B. W. Hollocks, "Forty years of discrete-event simulation a personal reflection," Journal of the Operational Research Society, vol. 57, no. 12, pp. 1383–1399, 2006.

[5] E. Ing, E. Babulak, and M. Wang, "Trends in discrete event simulation."

[6] P. Sharma, "Simulación de evento discreto," Revista internacional de investigación científica y tecnológica, vol. 4, no. 4, pp. 136–140, 2015.

[7] P. J. T. Vega, Simulación de Sistemas con el software Arena. Fondo editorial Universidad de Lima, 2017.

[8] W. D. Kelton, Simulation with ARENA. McGraw-hill, 2002.

[9] A. Goti, Discrete Event Simulations. BoD–Books on Demand, 2010.

QuantityEr: An extensible and simple solution to obtain the amount of results of complex queries to GitHub

QuantityEr: Una solución simple y extensible para obtener la cantidad de resultados de consultas complejas a GitHub

Ernesto Soto Gómez
Universidad de las Ciencias Informáticas
esoto@uci.cu
https://orcid.org/0000-0001-6521-2221

RECIBIDO 03/12/2019 • ACEPTADO 20/12/2019 • PUBLICADO 30/03/2020

ABSTRACT

GitHub is a platform that provides hosting for software development version control using Git. It features an application programming interface to allow the software to interact with the platform. The enormous quantity of information Hosted in GitHub may be useful to make studies about the current presence of development tools in the open-source software development community. However, the search engine has restrictions that make it impossible to issue complex queries to the platform. In this report, it is described as an object-oriented and extensible solution, named QuantityEr, to obtain the number of search results of complex queries to GitHub by using the inclusion-exclusion principle. The mathematical definitions, as well as related concepts, are presented. The mathematical model is discussed. The application of general design and used development tools are presented. Also, the results of the execution examples are showed. It is concluded that the treated problem has been solved although more work may be done to improve the solution.

Keywords: search results amount, GitHub, inclusion-exclusion principle, object-oriented programming, Python.

RESUMEN

GitHub es una plataforma que proporciona alojamiento para el control de versiones de desarrollo de software utilizando Git. Cuenta con una interfaz de programación de aplicaciones para permitir que el software interactúe con la plataforma. La enorme cantidad de información alojada en GitHub puede ser útil para realizar estudios sobre la presencia actual de herramientas de desarrollo en la comunidad de desarrollo de software de código abierto. Sin embargo, el motor de búsqueda posee restricciones que hacen imposible emitir consultas complejas a la plataforma. En este informe, se describe una solución extensible y orientada a objetos, llamada QuantityEr, para obtener la cantidad de resultados de búsqueda de consultas complejas a GitHub utilizando el principio de inclusión-exclusión. Se presentan las definiciones matemáticas y los conceptos relacionados. Se discute el modelo matemático. Se presentan el diseño general de la aplicación y las herramientas de desarrollo utilizadas. Además, son mostrados resultados de ejemplos de ejecución. Se concluye que el problema tratado ha sido resuelto, aunque se puede trabajar para mejorar la solución.

Palabras claves: *cantidad de resultados de búsqueda, GitHub, principio de inclusión-exclusión, programación orientada a objetos, Python.*

INTRODUCTION

GitHub[1] is a platform that provides hosting for software development version control using Git[2]. It provides several collaboration features such as bug tracking, feature requests, task management, and wikis for every project. It also features an application programming interface (API) to allow software to interact with the platform[3] [1]. Through this API a search engine can be accessed. The search engine allows users to find almost every single aspect across several projects, source codes and other areas and features of the platform[4] [2]. A web page that serves as an interface to the search API is also available[5].

As of August 2019, GitHub reports having over 40 million users and more than 100 million repositories[6]. This enormous quantity of information may be useful, among other things, to obtain the number of projects, source codes, issues, etc, that mention a set of technologies, tools, development libraries, etc, in order to make studies about the current presence of these tools in the open source software development community. Other kind of quantitative studies may be done as well [3]. Examples of those kinds of research are [4–7].

However, the search engine has some restrictions[4] that make impossible to issue complex queries to the platform. According to the GitHub Developer Guide[4], the restrictions are the following:

- The Search API does not support queries that:
  - are longer than 256 characters (not including operators or qualifiers).
  - have more than five AND, OR, or NOT operators.
- For authenticated requests can be made up to 30 requests per minute. For unauthenticated requests, the rate limit allows making up to 10 requests per minute.

Furthermore, if the search is over source code files, especial restrictions apply[7].

[1] https://github.com/
[2] https://git-scm.com/
[3] https://developer.github.com/v3
[4] https://developer.github.com/v3/search/
[5] https://github.com/search
[6] https://github.com/about
[7] https://developer.github.com/v3/search/#search-code

A system named GHTorrent have been already developed to ease the interaction with the large quantity of information hosted in GitHub[8] [8]. This solution is mainly conceived to mirror the data hosted in GitHub in order to facilitate parallel access and studies on snapshots of the data, but does not provide an alternative to making complex queries to GitHub. In fact, this system has its own restrictions on the quantity of data that can be accessed at any time[9][10]. Also, the system only provides snapshots for a reduced set of projects[11][12]. Moreover, its design is centered only on the interaction with the repositories of GitHub. This means, for example, that search on source code is not allowed. Furthermore, the objective of the system is to interact with GitHub, which means that a future interaction with other platforms is not currently conceived.

A different kind of alternative is GH Archive[13] which records events form GitHub[14]. The recorded data can be accessed through BigQuery[15] which allows any kind of SQL-like queries. GH Archive, although a powerful and flexible solution, does not constitute an alternative to explore the data stored in GitHub but a tool to explore the data that represents the interaction with GitHub. This means that, for example, searching inside public source code cannot be done with GH Archive.

Moreover, both of these systems are server like development tools and not client applications ready to use for making queries.

In the context of this article, complex queries are those that have many logical connectives and sub-expressions –for example: A OR (C AND (D OR E))– especially those that exceed the allowed number of logical operators. By getting the results number of queries of this kind, analysis of the current presence of technologies might be done. Although many reporting tools has been developed none of them are capable of getting the results number of complex queries directly to GitHub. Some of these tools are listed in https://www.gharchive.org/. Another example not listed in previous URL is https://www.programcreek.com/. In that case the reports are just for statically-selected libraries from statically-selected languages.

In this report, it is described a simple solution, named QuantityEr[16], to obtain the search results number of complex queries directly to GitHub. The proposed design was conceived with the aim of extension in mind, in such a way that it would be possible to incorporate the ability to interact with other similar platforms besides GitHub as well as other queries languages and algorithms for obtaining the amount of search results.

[8] http://ghtorrent.org/
[9] http://ghtorrent.org/raw.html
[10] http://ghtorrent.org/mysql.html
[11] http://ghtorrent.org/mongo.html
[12] http://ghtorrent.org/relational.html
[13] https://www.gharchive.org/
[14] https://developer.github.com/v3/activity/events/types/
[15] https://developers.google.com/apps-script/advanced/bigquery
[16] Source code accessible from https://github.com/EStog/QuantityEr/tree/v0.1

The current document is structured in the following manner. Section exposes some mathematical definitions and concepts necessary to understand the proposed solution. Section describes the proposed solution as well as some usage examples. Section makes the final remarks and conclude.

MATHEMATICAL BACKGROUND

In order to understand the proposed solution, some mathematical background is necessary. To archive a self- contained report, in this section is mentioned the principal mathematical concepts used in the design of the solution. The following definitions (or equivalent ones) as well of other complementary concepts and profs can be found in the cited references [9–17].

The following notations will be used in this report.

- $\wp(\mathrm{A})$ denotes the power set of a set A, that is the set of all subsets of A.
- $|\mathrm{A}|$ denotes the cardinality of a set A, that is the number of elements in A.
- $\emptyset$ denotes the empty set.

Boolean algebras

The first essential concept important to the design of the proposed solution is that of Boolean algebra.

Definition 1. A Boolean algebra is a tuple $(S, +, \cdot, I, \perp, T)$ where S is a set containing distinct elements $\perp$ and T, $+$ and $\cdot$ are binary operators on S and I is a unary operator on S. Every Boolean algebra satisfies the following laws for all $x, y, z \in S$.

Commutative laws:	$x + y = y + x$	$x \cdot y = y \cdot x$
Distributive laws:	$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$	$x + (y \cdot z) = (x + y) \cdot (x + z)$
Identity laws:	$x + \perp = x$	$x \cdot T = x$
Complement laws:	$x + x^I = T$	$x \cdot x^I = \perp$

Associative and idempotent laws, as well as other laws can be also considered since they follow from the definition laws. Furthermore, other useful operators can be derived from the previous ones [12, 14, 16].

Fact 1. In a Boolean algebra $(S, +, \cdot, I, \perp, T)$ the following laws are satisfied for all $x, y, z \in S$:

Associative laws: $x + (y + z) = (x + y) + z$ $\quad x \cdot (y \cdot z) = (x \cdot y) \cdot z$

Idempotent laws: $x + x = x$ $\quad x \cdot x = x$

Boolean algebras are used to model operations over the elements of a set that relates two elements with the maximum ($+$ operation) or the minimum ($\cdot$ operation) of both elements in a partial order where the minimum and the maximum are $\perp$ and T, respectively. In other words, a partial order $\leq$ can be defined over S where

$$\forall\, a, b \in S\, (a \leq b \Leftrightarrow a + b = b)$$

or equivalently

$$\forall\, a, b \in S\, (a \leq b \Leftrightarrow a \cdot b = a)$$

and

$$\forall\, a \in S\, (\perp \leq a \leq T)$$

[14].

Also, intuitively speaking, all the elements have an associated complement counterpart that together form the maximum but apart from the minimum as stated in the complement laws.

Fact 2. The tuple $(\{0, 1\}, \vee, \wedge, \neg, 0, 1)$ is a Boolean algebra with the operations of disjunction $(\vee)$, conjunction $(\wedge)$ and negation $(\neg)$ defined as follow.

$\vee$	0	1
0	0	1
1	1	1

$\wedge$	0	1
0	0	0
1	0	1

x	$\neg x$
0	1
1	0

This is the most elemental Boolean algebra and is the one found in classical binary logic that has applications in several areas of computer sciences [10, 12–14].

Fact 3. The tuple $(\wp(U), \cup, \cap, C, \emptyset, U)$ is a Boolean algebra with the operation of union $(\cup)$, intersection $(\cap)$ and complement (C) defined as follows for all $X, Y \in \wp(U)$.

$X \cup Y = \{x \mid x \in X \vee x \in Y$ $X \cap Y = \{x \mid x \in X \wedge x \in Y\}$ $X^C = \{x \mid x \notin X\}$

This specific Boolean algebra is of great interest in science since mathematics in general are founded in set theory [11–14].

In this specific work, the last two described Boolean algebras are crucial because the current problem is to find the number of objects that makes true a logical sentence. In this context, the logical sentence is the query to be issue to the platform. The proposed solution takes advantage of the equivalences between classical logic and set theory in the context of Boolean algebras to solve this problem.

Boolean functions

In some contexts, the combination of operations in the set $\{0,1\}$ are called Boolean functions. The following definitions relate to this subject.

Definition 2. A Boolean function of degree n is a function $f: \{0,1\}^n \rightarrow \{0,1\}$ where f is an atom (a single variable or value) or a composition of the operations ∧, ∨ and ¬ of the Boolean algebra $(\{0,1\},\vee,\wedge,\neg,0,1)$. This composition is called a Boolean expression, and the variables of the Boolean expression are called Boolean variables.

This concept has wide application in logic gates circuits design. In this topic one of the main problems is the simplification of Boolean expressions [9, 12, 14, 16].

In the case of this work, these are of great importance because, as we will see, each query has an associated Boolean expression. The objective is to simplify it in order to obtain an expression that involves less computation.

The simplification of a Boolean expressions may be done symbolically by applying the laws of a Boolean algebra (definition 1) but also by applying specific methods that simplify an equivalent form of the expression.

Definition 3. Two Boolean expressions $A(x_1, x_2, \ldots, x_n)$ and $B(x_1, x_2, \ldots, x_n)$ are equivalent if $\forall x_1, x_2, \ldots, x_n \in \{0,1\}\ (A(x_1, x_2, \ldots, x_n) = B(x_1, x_2, \ldots, x_n))$

Definition 4. A normal form of a Boolean expression $f(x_1, x_2, \ldots, x_n)$ is an equivalent Boolean expression in the form $g(x_1, x_2, \ldots, x_n) = \mathrm{t}_1 * \mathrm{t}_2 *, \ldots, * \mathrm{t}_m$ where each $\mathrm{t}_1 \leq i \leq m$ is in the form $\mathrm{y}_1 * \mathrm{y}_2 *, \ldots, * \mathrm{y}_{k \leq n}$ and each $\mathrm{y}_1 \leq j \leq k$ is in the form x_k or $\neg x_k$ where $1 \leq k \leq n$.

When $*$ is $\wedge$ and $\star$ is $\vee$ the normal form is called conjunctive (CNF). Similarly, when $*$ is $\vee$ and $\star$ is $\wedge$ the normal form is called disjunctive (DNF). Additionally, when the normal form is conjunctive each $t_1 \leq i \leq m$ is called a maxterm. Similarly, when the normal form is disjunctive each $t_1 \leq i \leq m$ is called a minterm.

The Quine-McCluskey algorithm is one of such methods that uses the normal form of a Boolean expression, specifically DNF, to obtain an equivalent minimal expression. The algorithm, in essence, test combinations of the minterms in order to find those that are essential to represent the value of the expression. It is known that it does not performance well when the size of the input, in this case the expression to simplify, is big. In fact, the problem of simplification of Boolean expressions is considered NP-hard [12, 14, 16].

However, the simplification of a Boolean expression is steel of great importance to this work, because small queries are preferable to big ones.

Definition 5. Let $X_1, X_2, \ldots, X_n$ be given sets. A predicate is a function $P : X_1 \times X_2 \times, \ldots, X_n \rightarrow \{0,1\}$ [10, 13].

It obvious that a predicate has an associated Boolean expression if each atom is replaced by a Boolean variable.

Definition 6. The expression $S = \{x \mid P(x)\}$ is equivalent to $x \in S \Leftrightarrow P(x)$ [11].

The following theorem will be useful in the modeling of the solution.

Theorem 1. The following relations are satisfied for any $A = \{x \mid P(x)\}$ and $B = \{x \mid Q(x)\}$:

(a) $A \cup B = \{x \mid P(x) \vee Q(x)\}$

(b) $A \cap B = \{x \mid P(x) \wedge Q(x)\}$

(c) $A^C = \{x \mid \neg P(x)\}$

Demonstration. Proof follows directly from fact 3 and definition 6.

This relations may be easily understood, since if A contains all the elements x such that $P(x) = 1$ and B is all the elements x such that $Q(x) = 1$ then it follows – from the definition 6 and the definition of union in the fact 3 – that $A \cup B$ will have the elements x such that $P(x) \vee Q(x) = 1$. The same analysis can be done for the intersection and complement cases.

Inclusion-exclusion principle

First let consider the cardinality of the power set. This will be useful later in the description of the proposed solution.

Fact 4. The cardinality of the power set of U is

$$\wp(U) = 2^{|U|}$$

[13].

The inclusion-exclusion principle (IEP) is a mathematical formula that can be used to obtain the cardinality of the union of finite sets taking into account the cardinality of all possible intersections of the given sets.

Fact 5 (Inclusion-exclusion principle). The cardinality of the union of sets $S_{i \in \{1,2,\ldots,n\}}$ is

$$\left|\bigcup_{i=1}^{n} S_i\right| = \sum_{\emptyset \neq J \in \{1,2,\ldots n\}} (-1)^{|J|+1} \left|\bigcap_{j \in J} A_j\right|$$

The number of every possible intersection of n sets is the same that the number of subsets of a set of n elements without counting the empty set. This leads to the following fact taking into account fact 4.

Fact 6. There are

$$2^n - 1$$

terms in the inclusion-exclusion principle formula for n sets.

This means that an algorithm that calculates the cardinality of the union of n sets by directly using the IEP have an exponential complexity [15, 17].

In the proposed solution the IEP is used to decompose a given query in many smaller sub-queries that will be issued to the platform search API. In the next section, will be shown how to manage the problem of the exponential complexity when using this method.

RESULTS AND DISCUSSION

The problem to solve is: How to get the results number of complex queries to GitHub?

The proposed solution follows a divide and conquer approach as follows:

1. Simplify and decompose complex queries into smaller simple sub-queries.

2. Issue the sub-queries to the server and obtain the results amount of each one.

3. Sum up the results of the sub-queries into one that constitutes the results amount of the initial complex query.

In the next subsection a mathematical model and formalization of the solution is given.

Mathematical model

Mathematically speaking, the problem to solve is as follows.

Let O be the set of all the objects in the platform (projects, source codes, etc). Let $Q : O \rightarrow \{0,1\}$ be a predicate that represents the query to issue. Then, the set r_Q of all objects that match the query Q is

$$r_Q = \{o \mid Q(o)\}$$

The problem to solve is finding r_Q when the associated Boolean expression given by Q has many compositions and logical connectives.

The first step of the proposed solution is to simplify the Boolean expression associated to the query. This may be done by symbolic transformations applying the laws that a Boolean algebra satisfies or also by using the Quine-McCluskey algorithm. It is known that this solution is not effective when the size of the input is too big. For this reason, the resultant expression (simplified or not) must be decomposed into various sub-expressions. For this purpose, the DNF expression is used. By applying theorem 1 it is known that if $Q(o) = Q_1(o) \vee Q_2(o) \vee \ldots \vee Q_n(o)$ is the DNF then

$$r_Q = \{o \mid Q_1(o) \vee Q_2(o) \vee \ldots \vee Q_n(o)\} = r_{Q_1} \cup r_{Q_2} \cup \ldots \cup r_{Q_n}$$

where

$$r_{Q_1} = \{o \mid Q_i(o)\}$$

for each $1 \leq i \leq n$.

Each $Q_i(o)$ is in the form $Q_{i_1}(o) \wedge Q_{i_2}(o) \wedge \ldots \wedge Q_{i_m}(o)$. This kind of query can be issued directly to GitHub because it does not have composition and only have conjunctive connectives. The conjunctives connectives (AND in the query language of GitHub) can be stripped of the sub-query since GitHub automatically interprets a tuple of atoms as a conjunction. In this case the is no use

of conjunctive or disjunctive connectives in the query. Nevertheless, the case of the negation is a problem that, for now, cannot be avoided. So, in this case, a query must be designed with care in order not to exceed the restriction that GitHub Search API imposes in the number of operators.

After the sub-queries have been sent, the next step is to find the results amount of the main query by applying IEP (fact 5). The problem with this approach is that the number of terms –according to fact 6– in IEP formula with n sets is $2^n - 1$, which is the number of sub-queries to be issued to the server.

However, each term in IEP is of the form of an intersection. Moreover, the terms in the expression associated to the DNF are also in the form of intersection. Then, by applying fact 1, that it is possible to reduce each term of the IEP formula so that some terms might be repeated afterwards. For this reason, it is proposed to use a cache for storing already issued queries as well as its respective results quantities in order to reduce the number of issued queries. However, work still need to be done to accelerate the computations of the terms in the IEP formula.

Solution design

QuantityEr is designed by using the object-oriented paradigm. Care on extension has been taken from the beginning by assigning a class to each sub-process in the solution. In Figure 1 is outlined the class diagram of the most important classes. The classes are given as abstract base classes, so they must be extended for a particular problem. Currently, the extensions for solving the problem in the specific case of GitHub are implemented. Next, it is briefly described each class.

Main: Coordinate the interaction between the Input, Engine and Output classes objects. That is, the main algorithm is implemented inside this class.

Input: Currently, the queries can be presented to QuantityEr from two sources: the command line and files. Several queries can be presented to the application in one single execution. The responsibility of this class is to present these sources as a stream to the Parser. Since the logic of the input is encapsulated in one class, other kind of inputs may be added in the future like, for example, inputs from the network.

Parser: Translate the queries presented as input to a standard language that can be managed by the other entities. Since the logic of parsing is encapsulated in one class the syntax of the language used in the input queries do not need to be like the one expected by GitHub. This may ease the input allowing a cleaner syntax.

MiddleCode: Represents the intermediate language that the other classes understand. All the queries inside the application are in this format.

Engine: Coordinate the interaction between the Decomposer, Cache, Translator and QueryIssuer classes objects. That is, the algorithm that give the solution to the problem is implemented inside this class.

Decomposer: Decompose a complex query into several smaller simple queries. Currently, the extension using IEP is implemented.

Cache: Store the results amounts of already issued queries. Currently, an in-memory cache is available as well as a file-based one.

Translator: Translate a given simple sub-query to an issuable one. Currently, only GitHub is supported but more platforms may be added in the future.

QueryIssuer: Emit a simple sub-query to the platform and obtain the results amount or inform of an error if it was the case.

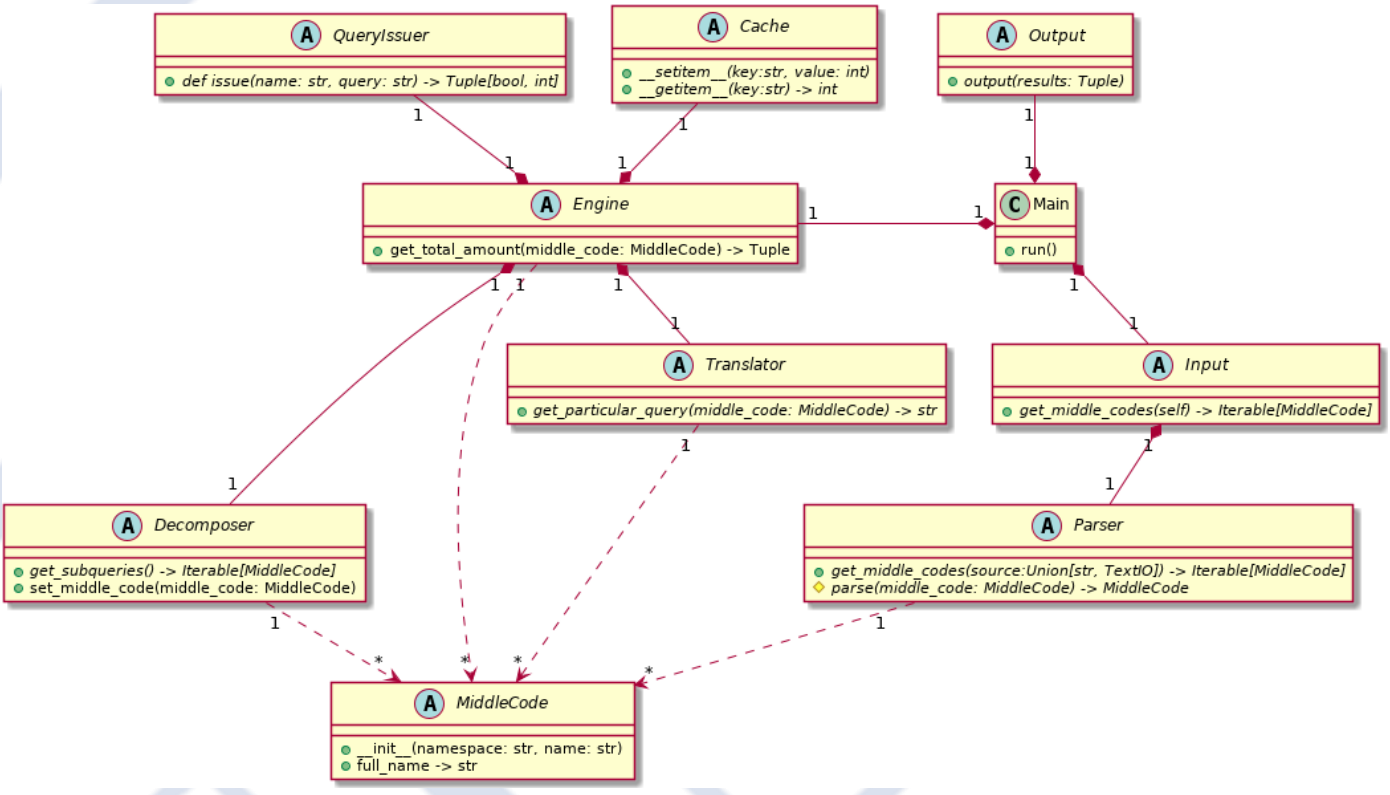

Figure 1. Class diagram of main classes of QuantityEr

Execution example results

In this section we consider a usage example result in order to study the behavior of the application with complex queries.

In this case, the queries ask for the amount of source codes that use the classical synchronization mechanisms defined in the asyncio, multiprocessing and threading Python libraries.

The results are summarized in table 1 and figure 2.

The command lines options to the program, the actual output, the presented queries as well as other execution example can be found in attached document examples.html[17].

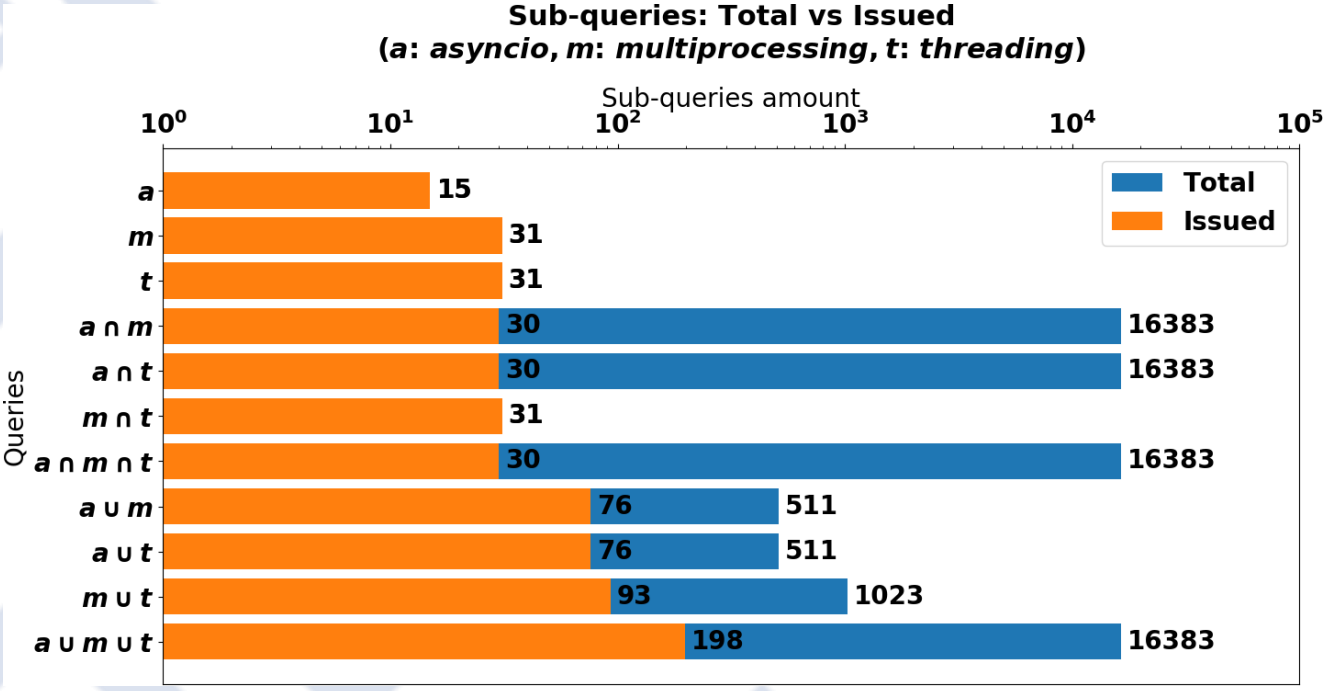

Figure 2. Sub-queries amount. Total vs Issued

In table 1 and figure 2 can be seen that the number of sub-queries depend on the ability of the Python's[18] [19] [18] Sympy[20] [21] [19] library to simplify the given expression. Also, in this case, the

[17] Downloadable from https://github.com/EStog/QuantityEr/blob/v0.1/running/jupyterlab/examples.html
[18] https://www.python.org/
[19] https://docs.python.org/3.7/
[20] https://www.sympy.org/en/
[21] https://docs.sympy.org/latest/index.html

presence of the cache effects a great reduction on the number of issued queries, especially when the number of sub-queries is big.

Table 1. Execution example results summary. # means quantity. % means percent.

			Sub-queries				
				Cached		Issued	
No.	**Queries libraries**	**Amount**	**Total**	**#**	**%**	**#**	**%**
01	asyncio	69 053	15	0	0	15	100
02	multiprocessing	159 515	31	0	0	31	100
03	threading	1 451 344	31	0	0	31	100
04	asyncio ∩ multiprocessing	3 095	16 383	16 353	99.82	30	0.18
05	asyncio ∩ threading	29 658	16 383	16 353	99.82	30	0.18
06	multiprocessing ∩ threading	124 327	31	0	0	31	100
07	asyncio ∩ multiprocessing ∩ threading	1 947	16 383	16 353	99.82	30	0.18
08	asyncio ∪ multiprocessing	228 130	511	435	85.13	76	14.87
09	asyncio ∪ threading	1 494 420	511	435	85.13	76	14.87
10	multiprocessing ∪ threading	1 489 850	1 023	930	90.91	93	9.09
11	asyncio ∪ multiprocessing ∪ threading	1 528 155	16 383	16 185	98.79	198	1.21

CONCLUSIONS

In this report a tool, named QuantityEr, to obtain the results number of complex queries to GitHub search API has been described. The application uses the inclusion-exclusion principle and other mathematical abstractions to decompose the query in several simple sub-queries. The application uses a cache in order to reduce the number of sub-queries issued to the server. Even though it is considered that the use of the cache improves the solution and makes it viable, more work may to be done in order to accelerate the computations of the IEP formula terms. Moreover, the application may be extended to resolve other restrictions problems in GitHub and other platforms.

REFERENCES

[1] C. Dawson and B. Straub, Building Tools with GitHub: Customize Your Workflow, 1st ed. O'Reilly Media, Inc., 2016.

[2] ——, "Python and the Search API," in Building Tools with GitHub: Customize Your Workflow, 1st ed. O'Reilly Media, Inc., 2016, pp. 53–80.

[3] S. Amann, S. Beyer, K. Kevic, and H. Gall, "Software Mining Studies: Goals, Approaches, Artifacts, and Replicability," in Software Engineering: International Summer Schools, LASER 2013-2014, Elba, Italy, Revised Tutorial Lectures, ser. Lecture Notes in Computer Science, B. Meyer and M. Nordio, Eds. Cham: Springer International Publishing, 2015, pp. 121–158. [Online]. Available: https://doi.org/10.1007/978-3-319-28406-4_5

[4] M. Beller, R. Bholanath, S. McIntosh, and A. Zaidman, "Analyzing the State of Static Analysis: A Large- Scale Evaluation in Open Source Software," in 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), vol. 1, Mar. 2016, pp. 470–481.

[5] Y. Zhang, G. Yin, Y. Yu, and H. Wang, "Investigating Social Media in GitHub's Pull-requests: A Case Study on Ruby on Rails," in Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies, ser. CrowdSoft 2014. New York, NY, USA: ACM, 2014, pp. 37–41, event-place: Hong Kong, China. [Online]. Available: http://doi.acm.org/10.1145/2666539.2666572

[6] ——, "A Exploratory Study of @-Mention in GitHub's Pull-Requests," in 2014 21st Asia-Pacific Software Engineering Conference, vol. 1, Dec. 2014, pp. 343–350.

[7] A. A. Sawant and A. Bacchelli, "fine-GRAPE: fine-grained APi usage extractor – an approach and dataset to investigate API usage," Empirical Software Engineering, vol. 22, no. 3, pp. 1348–1371, Jun. 2017. [Online]. Available: https://doi.org/10.1007/s10664-016-9444-6

[8] G. Gousios, B. Vasilescu, A. Serebrenik, and A. Zaidman, "Lean GHTorrent: GitHub Data on Demand," in Proceedings of the 11th Working Conference on Mining Software Repositories, ser. MSR 2014. New York, NY, USA: ACM, 2014, pp. 384–387, event-place: Hyderabad, India.

[9] J. W. Grossman, "Functions," in Handbook of Discrete and Combinatorial Mathematics, 2nd ed., K. H. Rosen, Ed. Chapman & Hall/CRC, 2018, pp. 32–42.

[10] ——, "Propositional and Predicate Logic," in Handbook of Discrete and Combinatorial Mathematics, 2nd ed., K. H. Rosen, Ed. Chapman & Hall/CRC, 2018, pp. 12–22.

[11] ——, "Set Theory," in Handbook of Discrete and Combinatorial Mathematics, 2nd ed., K. H. Rosen, Ed. Chapman & Hall/CRC, 2018, pp. 22–32.

[12] R. Johnsonbaugh, "Boolean Algebras and Combinatorial Circuits," in Discrete Mathematics, 8th ed. New York, NY: Pearson, 2017, pp. 532–567.

[13] ——, "Sets and logic," in Discrete Mathematics, 8th ed. New York, NY: Pearson, 2017, pp. 1–61.

[14] J. G. Michaels, "Boolean Algebras," in Handbook of Discrete and Combinatorial Mathematics, 2nd ed., K. H. Rosen, Ed. Chapman & Hall/CRC, 2018, pp. 269–379.

[15] R. G. Rieper, "Inclusion/Exclusion," in Handbook of Discrete and Combinatorial Mathematics, 2nd ed., K. H. Rosen, Ed. Chapman & Hall/CRC, 2018, pp. 110–116.

[16] K. H. Rosen, "Boolean Algebra," in Discrete Mathematics and Its Applications, 8th ed. New York, NY: McGraw-Hill, 2019, pp. 847–883.

[17] ——, "Inclusion–Exclusion," in Discrete Mathematics and Its Applications, 8th ed. New York, NY: McGraw-Hill, 2019, pp. 579–585.

[18] S. Kapil, Clean Python: Elegant Coding in Python. Apress, 2019.

[19] J. M. Stewart, "SymPy: A Computer Algebra System," in Python for Scientists, 2nd ed. New York, NY: Cambridge University Press, 2017, pp. 128–149.

Aplicación de métodos numéricos utilizando MATLAB al modelado del fármaco AZT y la supervivencia con SIDA

Application of numerical methods with MATLAB for modelling the AZT drug and the survival by AIDS

Ihosvany Rodríguez González
Centro Nacional de Biopreparados
ihosvany@biocen.cu
https://orcid.org/0000-0003-0212-9556

Anié Bermudez Peña
Universidad de las Ciencias Informáticas
abp@uci.cu
https://orcid.org/0000-0002-1387-7472

RECIBIDO 02/12/2019 • ACEPTADO 27/12/2019 • PUBLICADO 30/03/2020

RESUMEN

En la presente investigación se realiza un estudio para aplicar métodos numéricos a la ecuación diferencial modelada en un caso concreto del sector Biofarmacéutico, que fue necesario en su momento realizarlo por la importancia de la aplicación de este medicamento en una enfermedad que se convirtió en una pandemia a nivel mundial. En este problema se describe el impacto de la cidovudina (acidotimidina o AZT) sobre la supervivencia de quienes desarrollan el Síndrome de Inmunodeficiencia Adquirida por infección con el Virus de la Inmunodeficiencia Humana. Para la solución de este modelo se cuenta con una ecuación diferencial ordinaria de primer orden con valores iniciales sobre la cual se aplica el método de separación de variables para obtener la solución real de forma analítica. Se aplican tres métodos numéricos (Euler, Euler Mejorado y Runge Kuta 4) usando el asistente matemático MATLAB para calcular las soluciones aproximadas. Finalmente se muestran los resultados de los métodos, los errores absolutos y relativos de cada uno y la comparación con la solución analítica, con sus respectivas tablas y gráficas.

Palabras claves: fármaco AZT, MATLAB, métodos numéricos, SIDA.

ABSTRACT

In our work we conducted a study to apply numeric methods to the differential equation modeled in a concrete case of the sector of the biopharmaceutical sector, which was necessary at the time realize the importance of the application of this drug in a disease that became a pandemic Worldwide. In this problem the impact of zidovudine (azidothymidine or AZT) described on the survival of those who develop Acquired Immunodeficiency Syndrome infection with the Human Immunodeficiency Virus. For the solution of this model it is has an ordinary differential equation of first order with initial values on which the separation of variables is applied to obtain the real solution analytically and apply numerical methods 3 (Euler, Improved Euler and Runge Kuta 4) using MATLAB mathematical wizard to calculate approximate solutions. Finally, we show the results of the methods, the absolute and relative errors of each and a comparison with the analytical solution and their respective tables and graphs.

Keywords: *AIDS, AZT drug, MATLAB, numerical methods.*

INTRODUCIÓN

El Virus de la Inmunodeficiencia Humana (VIH), como los demás virus, no es una célula y no tiene metabolismo ni se puede reproducir fuera de una célula viva. Su información genética se encuentra en dos cadenas idénticas de ARN (ácido ribonucleico). Para reproducirse, debe emplear el aparato reproductor de la célula que invade a fin de producir copias exactas ARN. Lo que hace el VIH es transcribir su ARN pasándolo a ADN (ácido desoxirribonucleico) con una enzima, la transcriptasa inversa, que está presente en el virus. El ADN viral, de doble cadena, emigra al núcleo de la célula invadida y se intercala en el genoma de ésta, con ayuda de otra enzima viral, la integrasa. Quedan así integrados el ADN vira1 y el ADN celular. Cuando la célula invadida recibe un estímulo para reproducirse, el ADN proviral se transcribe y forma ARN viral y se sintetizan nuevas partículas virales [1].

En [2] se describe el impacto de la cidovudina (acidotimidina o AZT, del inglés azidothymidine) sobre la supervivencia de quienes desarrollan el síndrome de Inmunodeficiencia Adquirida (SIDA) por infección con el VIH. Puesto que la cidovudina inhibe a la transcriptasa inversa del virus e interrumpe la síntesis de la cadena de ADN en el laboratorio, se esperaba que sirviera para desacelerar o detener el avance de la infección con VIH en los humanos.

La causa de que el VIH sea tan peligroso es que, además de ser un virus rápidamente mutante, ataca en forma selectiva a los linfocitos ayudantes T (vitales en el sistema inmunológico del anfitrión) porque se enlaza a la molécula CD4 de la superficie celular. Los linfocitos T (células CD4 T o células T4) son fundamentales en la organización de una defensa contra cualquier infección [3]. Aunque los parámetros inmunológicos del sistema inmunitario en un anfitrión infectado con VIH cambian cuasi estáticamente tras la etapa aguda de la infección, miles de millones de linfocitos T4 y VIH son destruidos y reemplazados cada día durante un periodo de incubación que puede durar dos décadas o más. La densidad de linfocitos T4 es un marcador muy común para evaluar el avance de la enfermedad porque su disminución es paralela al deterioro del sistema inmunitario infectado por VIH.

La presente investigación no pretende tener un aporte directo al tratamiento del VIH y la utilización del AZT, sino aplicar métodos numéricos para resolver la ecuación diferencial modelada [4]. Como continuidad de la investigación, estos métodos pudieran ser significativos en los estudios de otros fármacos en el sector de BIOCUBAFARMA. Tiene como objetivos:

1. Aplicar los métodos numéricos que se ajustan a este tipo de problema de valor inicial donde interviene una ecuación diferencial ordinaria de primer orden y valores iniciales, para calcular las soluciones aproximadas.

2. Aplicar un asistente matemático para facilitar el cálculo de las operaciones en cada método que se aplique.

3. A partir de los resultados hacer un análisis comparativo de los métodos y la solución analítica que se muestra.

MATERIALES Y MÉTODOS

Métodos numéricos

Los métodos numéricos diseñados para ecuaciones diferenciales de Problemas de Valor Inicial tienen como objetivo determinar el punto siguiente usando la información de los puntos anteriores y la derivada.

En la presente investigación se analizaron varias familias de métodos [5], [6], entre los que se encuentran los Métodos de un Paso y Explícitos que solo utilizan la información del punto anterior para calcular el siguiente.

1. Método de Euler: es un procedimiento de integración numérica para resolver Ecuaciones Diferenciales Ordinarias (EDO) a partir de un valor inicial dado. El método de Euler es el más simple de los métodos numéricos para resolver un problema de valor inicial, y el más simple de los Métodos de Runge Kutta (RK1). La fórmula iterativa que se aplica es la Ecuación 1:

$$y_{n+1} = y_n + hf(x_n, y_n); donde\ x_n = x_0 + h * n \quad (1)$$

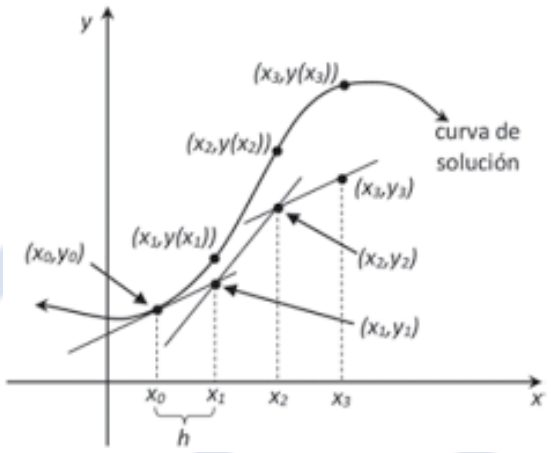

Figura 1. Curva de la solución para el método de Euler.

2. Método de Euler Mejorado (Método Heun): este método se basa en la misma idea del método anterior, pero hace un refinamiento en la aproximación, tomando un promedio entre ciertas pendientes. La fórmula que se aplica de forma iterativa es la Ecuación 2:

$$y_{n+1} = y_n + h\left[\frac{f(x_n,y_n)+f(x_{n+1},y^*_{n+1})}{2}\right]; donde\ y^*_{n+1} = y_n + h * f(x_n, y_n) \quad (2)$$

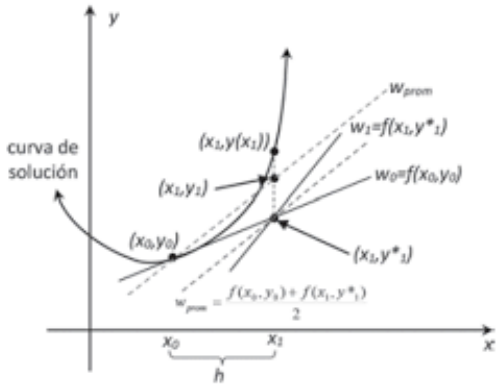

Figura 2. Curva de la solución para el método de Euler Mejorado. La pendiente de la recta punteada es el promedio de w_0 y w_1.

3. Métodos de Runge Kutta [7], [8]: esta es una familia de métodos muy utilizada debido a su gran exactitud. Existen muchas versiones y modificaciones, pero todas trabajan bajo el mismo principio básico, tomando una ponderación entre ciertas pendientes. En este caso se muestra el método de Runge Kuta de orden 4 (RK4) que es el que se va a utilizar. El método RK4 está dado por la Ecuación 3:

$$y_{i+1} = y_i + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4); donde \begin{cases} k_1 = f(x_i, y_i) \\ k_2 = f\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1 h\right) \\ k_3 = f\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2 h\right) \\ k_4 = f(x_i + h, y_i + k_3 h) \end{cases} \quad (3)$$

k_1: es la pendiente al principio del intervalo.

k_2: es la pendiente en el punto medio del intervalo, usando k_1: para determinar el valor de y en el punto $x_n + \frac{h}{2}$ usando el método de Euler.

k_3: es otra vez la pendiente del punto medio, pero ahora usando k_2 para determinar el valor de y.

k_4: es la pendiente al final del intervalo, con el valor de y determinado por k_3.

Así, el siguiente valor (yn+1) es determinado por el presente valor (yn) más el producto del tamaño del intervalo (h) por una pendiente estimada. La pendiente es un promedio ponderado de pendientes. Promediando las cuatro pendientes, se les asigna mayor peso a las pendientes en el punto medio, Ecuación 4:

$$pendiente = \frac{k_1+2k_2+2k_3+k_4}{6} \quad (4)$$

Asistente matemático MATLAB

Existen asistentes matemáticos para la resolución de problemas de este tipo como lo son MATLAB [9], [10], MATHCAD [7] y MATHEMATICA [6], [11], los cuales ayudan a obtener soluciones de una manera muy rápida y eficiente a muchos problemas de la vida real que a veces no tienen soluciones con los métodos analíticos.

Para el desarrollo de la investigación se utiliza el entorno de base matemática MATLAB (Matrix Laboratory) el cual es uno de los entornos de desarrollo matemático más extendido en aplicaciones de ingeniería para analizar y desarrollar prototipos de algoritmos. MATLAB es un sistema de cómputo numérico que ofrece un entorno de desarrollo integrado con un lenguaje de programación propio (lenguaje M) y disponible para las plataformas Unix, Windows [12], Mac OS X y GNU/Linux.

Desarrollo del modelo y justificación de los métodos

A continuación, una breve introducción del modelo y los resultados presentados por [2], donde se describe el impacto de la AZT en los años 80 y principios de los 90 sobre la supervivencia de quienes desarrollaban el SIDA por infección con el VIH.

La fracción de sobrevivientes $S(t)$ para este modelo es una solución de la Ecuación Diferencial Ordinaria de primer orden, Ecuación 5:

$$\frac{dS(t)}{dt} = -k(S(t) - S_i) \tag{5}$$

Donde:

t: representa el tiempo transcurrido hasta la aparición del SIDA clínico en un grupo de personas infectadas.

S_i: fracción inmortal.

k: probabilidad de morir, por unidad de tiempo en el momento t, se supondrá constante.

$S(t)$: la fracción del grupo que sigue viva en el momento t.

t -> Variable independiente.

S -> Variable dependiente.

En el modelo se aplica la técnica de separación de variables como método analítico, dando como resultado que la solución para la fracción sobreviviente es, Ecuación 6:

$$S(t) = S_i + (1 - S_i)e^{-kt} \quad (6)$$

Se define $T = k^{-1} \ln 2$ y se puede escribir la Ecuación 6 en su forma equivalente, Ecuación 7:

$$S(t) = S_i + (1 - S_i)2^{(\frac{t}{-T})} \quad (7)$$

Se aplicó el programa de mínimos cuadrados como método estadístico para ajustar la función de fracción de supervivencia en la Ecuación 7 y se apreciaron los siguientes resultados:

Para 159 habitantes que desarrollaron el SIDA en 1985, se obtuvo:

1. Un valor de la fracción inmortal Si = 0.0665 es un límite superior del valor real.
2. Un valor de periodo medio de supervivencia de 0.666 años.
3. El 10 % de estas personas sobrevivieron tres años con SIDA clínico.

Para 1415 personas infectadas por VIH y tratadas con AZT, se obtuvo:

1. Los supervivientes a más largo plazo viven cuando la densidad de sus linfocitos T4 es mayor de 10 por milímetro cúbico.
2. Se redefine el tiempo $t = 0$ como el momento en que la densidad de linfocitos T4 en una persona infectada con VIH baja de 10 por milímetro cúbico.
3. La supervivencia, $S(t)$, de las personas estudiadas fue 0.4700, 0.3160 y 0.1780, luego de pasado 1 año, 1.5 años y 2 años, respectivamente.

El ajuste de mínimos cuadrados de la fracción de supervivencia dio como resultado que, a los datos de individuos con densidades entre 0 y 10 linfocitos T4 por milímetro cúbico, produce un valor de la fracción inmortal $S_i = 0$.

Propuesta de solución utilizando los métodos numéricos Euler, Euler Mejorado

Dada la EDO de 1er Orden mostrada en la Ecuación 5, para $S_{(0)} = 1$ -> $t_0 = 0$, $0 \leq Si \leq 0.0665$, $0 \leq K \leq 1$.

Se plantean las ecuaciones de dos de los métodos en el problema en concreto de nuestra EDO:

$$\begin{aligned} S^*_{n+1} &= S_n + hf(t_n, S_n) \\ f(t_n, S_n) &= -k(S_n - S_i) \\ S^*_{n+1} &= S_n - hk(S_n - S_i) \end{aligned} \quad (8)$$

$$S_{n+1}= S_n+h\left(f(t_n,S_n)-f(t_n+1,S^*_{n+1})\right)/2$$
$$S_{n+1}= S_n+h(-k(S_n-S_i)+k(S^*_{n+1}-S_i))/2$$
$$S_{n+1}= S_n+h(kS^*_{n+1}-kS_n)/2$$
$$S_{n+1}= S_n+hk(S^*_{n+1}-S_n)/2 \quad (9)$$

Con el propósito de que la investigación pueda ser reproducida utilizando estos métodos numéricos para otros fármacos, a continuación, se presenta el código en MATLAB donde se realizaron las corridas que aparecen en las tablas y gráficos mostrados en la siguiente sesión de Resultados y Discusión.

```
clear all
a=0;
b=1;
K=0.80; SI=0.0300;
f=@(t,S)[-K*(S-SI)];
K1=@(t,S,h)[h*f(t,S)];
K2=@(t,S,h)[h*f(t+h/2,S+K1(t,S,h)/2)];
K3=@(t,S,h)[h*f(t+h/2,S+K2(t,S,h)/2)];
K4=@(t,S,h)[h*f(t+h,S+K3(t,S,h))];
h=0.0001;
N=(b-a)/h;
t=zeros(N+1,1); VR=zeros(N+1,1);E=zeros(N+1,1);A1=zeros(N+1,1);R1=zeros(N+1,1);
EM=zeros(N+1,1);A2=zeros(N+1,1);
R2=zeros(N+1,1);RK=zeros(N+1,1);A3=zeros(N+1,1);R3=zeros(N+1,1);
t(1)=a; VR(1)=1;E(1)=1;EM(1)=1;RK(1)=1;
for k=1:N
t(k+1)=t(k)+h;
VR(k+1)=SI+(1-SI)*exp(-K*t(k+1));
E(k+1)=E(k)+f(t(k),E(k))*h;
A1(k+1)=abs(VR(k+1)-E(k+1));
R1(k+1)=abs(100*(VR(k+1)-E(k+1))/VR(k+1));
EM(k+1)=EM(k)+h/2*(f(t(k),EM(k))+f(t(k+1),EM(k)+h*f(t(k),EM(k))));
A2(k+1)=abs(VR(k+1)-EM(k+1));
R2(k+1)=abs(100*(VR(k+1)-EM(k+1))/VR(k+1));
RK(k+1)=RK(k)+(K1(t(k),RK(k),h)+2*K2(t(k),RK(k),h)+2*K3(t(k),RK(k),h)+K4(t(k),RK(k),h))/6;
A3(k+1)=abs(VR(k+1)-RK(k+1));
R3(k+1)=abs(100*(VR(k+1)-RK(k+1))/VR(k+1));
end
hold on
plot(t,E,'b')%solución en azul para el método de Euler
plot(t,EM,'r')%solución en rojo para euler mejorado
plot(t,VR,'g')%solución en verde para el valor real
plot(t,RK,'k')%solución en negro para Runge-Kutta
xlabel('Tiempo')
ylabel('la fracción del grupo que sigue viva en el momento t')
title('Método de Euler, Euler Modificado y Runge Kuta de orden 4')
hold off
```

Figura 3. Código Matlab donde se realizaron las corridas que aparecen en las tablas y gráficos.

RESULTADOS Y DISCUSIÓN

Según los comandos en MATLAB mostrados en la Figura 3 se aplicaron diferentes métodos para comparar los, entre ellos el Error Absoluto (Ecuación 10) y Error Relativo (Ecuación 11).

$$Error\ Absoluto = VR - VA \quad (10)$$

$$Error\ Relativo = 100 * \frac{VR-VA}{VR} \quad (11)$$

Donde Valor Real (VR) es el resultado de la Ecuación 6 y el valor aproximado (VA) es el resultado de las Ecuaciones 1, 2 y 3 para cada método numérico y cada iteración. En las Tabla 1 y 2 se muestra la comparación de los métodos numéricos en distintos experimentos [7], [13].

Tabla 1. Comparación de los métodos numéricos con h=0.1.

tn	VR	**Euler. VA**	E. Abs	E.relativo %	**Euler Mejorado. VA**	E. Abs	E.relativo %	**RK4. VA**	E. Abs	E.relativo %
						1.0e-03 *			**1.0e-07 ***	**1.0e-04 ***
0.00	1	1	0	0	1	0	0	1	0	0
0.10	0.9292	0.9265	0.0027	0.2894	0.9293	0.0676	0.0073	0.9292	0.1914	0.0206
0.20	0.8635	0.8585	0.005	0.5769	0.8636	0.1255	0.0145	0.8635	0.3551	0.0411
0.30	0.8025	0.7956	0.0069	0.8625	0.8027	0.1747	0.0218	0.8025	0.4942	0.0616
0.40	0.7460	0.7375	0.0085	1.1461	0.7462	0.2160	0.0290	0.7460	0.6114	0.082
0.50	0.6935	0.6836	0.0099	1.4275	0.6938	0.2506	0.0361	0.6935	0.7090	0.1022
0.60	0.6449	0.6339	0.011	1.7066	0.6452	0.279	0.0433	0.6449	0.7893	0.1224
0.70	0.5997	0.5878	0.0119	1.9833	0.600	0.3019	0.0503	0.5997	0.8543	0.1424
0.80	0.5578	0.5452	0.0126	2.2574	0.5582	0.3202	0.0574	0.5578	0.9058	0.1624
0.90	0.5190	0.5058	0.0131	2.5288	0.5193	0.3342	0.0644	0.5190	0.9454	0.1822
1.00	**0.4829**	**0.4694**	0.0135	**2.7973**	**0.4833**	0.3445	**0.0713**	**0.4829**	0.9745	0.2018

Tabla 2. Comparación de los métodos numéricos con h=0.05.

tn	VR	**Euler. VA**	E. Abs	E.relativo %	**Euler Mejorado. VA**	E. Abs	E.relativo %	**RK4. VA**	E. Abs	E.relativo %
						1.0e-04 *			**1.0e-08 ***	**1.0e-05 ***
0	1	1	0	0	1	0	0	1	0	0
0.05	0.9639	0.9633	0.0007	0.0706	0.9639	0.0853	0.0009	0.9639	0.0602	0.0062
0.10	0.9292	0.9279	0.0013	0.141	0.9292	0.1644	0.0018	0.9292	0.1159	0.0125
0.15	0.8957	0.8938	0.0019	0.2113	0.8957	0.2375	0.0027	0.8957	0.1675	0.0187
0.20	0.8635	0.8611	0.0024	0.2814	0.8635	0.3050	0.0035	0.8635	0.2151	0.0249
0.25	0.8324	0.8295	0.0029	0.3513	0.8325	0.3672	0.0044	0.8324	0.259	0.0311
0.30	0.8025	0.7992	0.0034	0.421	0.8026	0.4245	0.0053	0.8025	0.2994	0.0373
0.35	0.7737	0.7699	0.0038	0.4906	0.7738	0.477	0.0062	0.7737	0.3364	0.0435

0.40	0.746	0.7418	0.0042	0.5599	0.7461	0.5251	0.0070	0.7460	0.3703	0.0496
0.45	0.7193	0.7148	0.0045	0.629	0.7193	0.5690	0.0079	0.7193	0.4013	0.0558
0.50	0.6935	0.6887	0.0048	0.6979	0.6936	0.6089	0.0088	0.6935	0.4295	0.0619
0.55	0.6688	0.6636	0.0051	0.7666	0.6688	0.6451	0.0096	0.6688	0.455	0.0680
0.60	0.6449	0.6395	0.0054	0.835	0.6449	0.6779	0.0105	0.6449	0.4781	0.0741
0.65	0.6219	0.6163	0.0056	0.9032	0.6219	0.7074	0.0114	0.6219	0.4989	0.0802
0.70	0.5997	0.5939	0.0058	0.9711	0.5998	0.7337	0.0122	0.5997	0.5175	0.0863
0.75	0.5784	0.5724	0.006	1.0388	0.5785	0.7572	0.0131	0.5784	0.5340	0.0923
0.80	0.5578	0.5517	0.0062	1.1062	0.5579	0.778	0.0139	0.5578	0.5487	0.0984
0.85	0.5380	0.5317	0.0063	1.1733	0.5381	0.7962	0.0148	0.5380	0.5615	0.1044
0.90	0.519	0.5125	0.0064	1.2401	0.5191	0.8120	0.0156	0.519	0.5727	0.1103
0.95	0.5006	0.4941	0.0065	1.3066	0.5007	0.8256	0.0165	0.5006	0.5822	0.1163
1.00	**0.4829**	**0.4763**	0.0066	1.3728	**0.4830**	0.8370	0.0173	**0.4829**	0.5903	0.1222

En las Figuras 4, 5, 6 y 7 se muestran los gráficos con la comparación de los métodos numéricos en distintos experimentos.

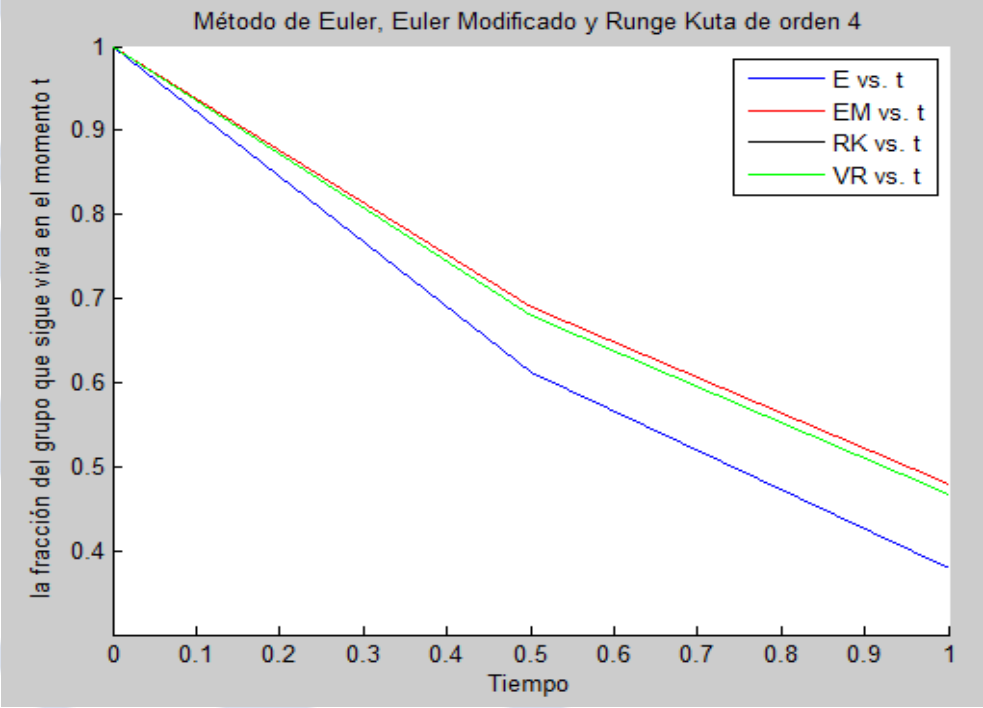

Figura 4. Gráfico comparativo entre los tres métodos para h= 0.5.

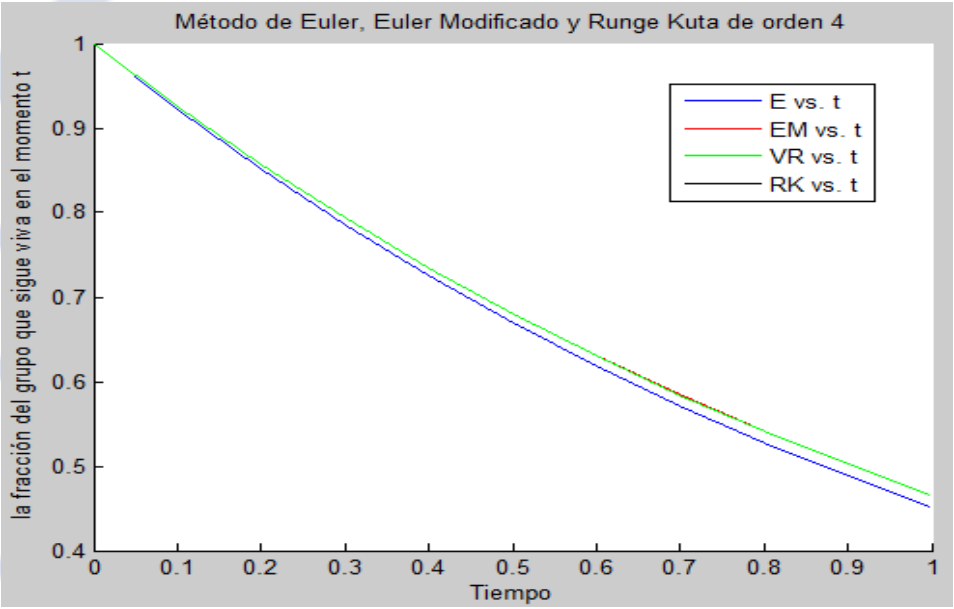

Figura 5. Gráfico comparativo entre los tres métodos para h= 0.1.

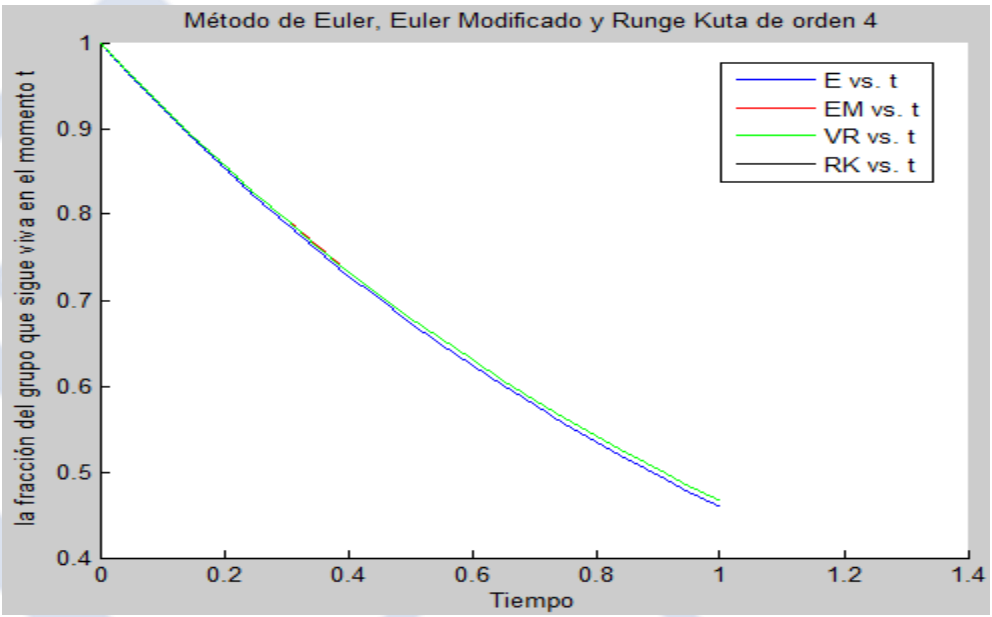

Figura 6. Gráfico comparativo entre los tres métodos para h= 0.05.

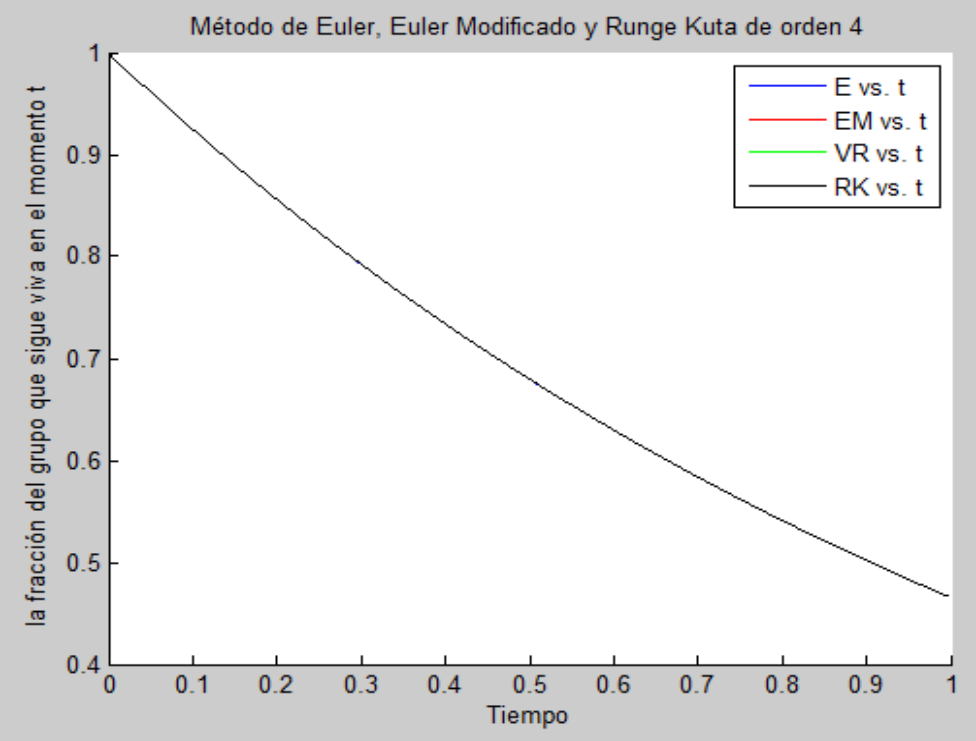

Figura 7. Gráfico comparativo entre los tres métodos para h= 0.0001.

Como se aprecia en los gráficos el método de Euler necesita de un paso muy fino para proporcionar una solución cercana a la real, mientras los otros dos son capaces de llegar hasta ella sin necesidad de pasos tan pequeños. Realmente esto en los ejemplos que he expuesto no tiene mucha importancia ya que los programas no necesitan de gran cantidad de operaciones para ejecutarse. Donde la eficiencia se pone de manifiesto es en aquellos modelos que usan gran cantidad de operaciones o necesitan de mucha precisión o trabajan con datos muy grandes (con matrices de 500×500 o mayor).

En la última figura no solo se muestra la solución que proporciona Runge-Kutta sino que se diagraman todas, lo que sucede es que coinciden y la última se sobrepone a las demás. Estos son métodos numéricos [14], de resultados con aproximaciones.

Con esta aplicación de métodos matemáticos utilizando MATLAB se corrobora lo planteado por el autor del problema [2]. Estos resultados demuestran con claridad que la AZT no fue eficaz para suspender la replicación en todas las cepas de HIV, porque quienes la recibieron terminaron por fallecer casi tan pronto como quienes no la tomaron. Por consiguiente, la capacidad inicial de la AZT para prolongar la supervivencia con VIH desaparece en último término y la infección retorna su avance. Se estima que la farmacoterapia con AZT extiende la supervivencia de un paciente infectado con VIH cinco o seis meses en promedio. El porcentaje de personas para las que el SIDA no era fatal es menor que 6.65 % y podría ser 0.

CONCLUSIONES

Con la presente investigación se mostraron los resultados de los tres métodos numéricos para el problema de valor inicial y su comparación con la solución exacta o valor real. Para la validación de la propuesta se realizaron los cálculos de los errores absolutos y relativos. Se aprecia que RK4 desde las primeras aproximaciones está muy próximo al valor real y, por tanto, el mejor método como refiere la literatura. En segundo lugar, Euler mejorado y por último Euler que necesita de un paso muy pequeño y es el más sencillo de los 3. Además, se corrobora mediante métodos numéricos lo planteado por el autor del problema. Estos resultados demuestran con claridad que solamente el fármaco AZT no era eficaz para suspender la replicación en todas las cepas de VIH. En estos momentos el AZT se está estudiando y aplicando unido a otros fármacos para el tratamiento VIH-SIDA, como es el caso del ABACAVIR [15]. El país se prepara en estas condiciones para la producción de nuevos genéricos que, al ser incluidos en su arsenal terapéutico, garantizarán que los pacientes cubanos dispongan de nuevas combinaciones de ARV para el control de la infección por VIH.

REFERENCIAS

[1] I. Kramer, "Is AIDS an invariably fatal disease?: A model analysis of AIDS survival curves," Mathematical and Computer Modelling, vol. 15, no. 9, pp. 1–19, Jan. 1991, doi: 10.1016/0895-7177(91)90001-N.

[2] I. Kramer, "The impact of zidovudine (AZT) therapy on the survivability of those with the progressive HIV infection," Mathematical and Computer Modelling, vol. 23, no. 3, pp. 1–14, Feb. 1996, doi: 10.1016/0895-7177(95)00229-4.

[3] P. Easterbrook, J. Emami, G. Moyle, and B. Gazzard, "Progressive CD4 cell depletion and death in zidovudine-treated patients," Journal of acquired immune deficiency syndromes, vol. 6, no. 8, pp. 927–929, Aug. 1993.

[4] D. G. Zill, A. E. G. Hernández, and E. F. López, Ecuaciones diferenciales con aplicaciones de modelado. Thomson Learning México, 2002.

[5] D. G. Zill, A first course in differential equations with modeling applications. Cengage Learning, 2012.

[6] J. G. S. León, Mathematica beyond mathematics: The Wolfram language in the real world. Chapman and Hall/CRC, 2017.

[7] C. M. Rodríguez, "Análisis comparativo de los métodos de Euler y Runge-Kutta en la solución numérica de ecuaciones diferenciales de primer orden mediante programación en mathcad," Revista Ingeniería, Matemáticas y Ciencias de la Información, vol. 3, no. 5, 2016.

[8] A. Cánovas Pérez, "Estudio de algoritmos matemáticos implementados en librerías de código abierto. Ecuaciones diferenciales: métodos de Ruge-Kutta," 2015.

[9] A. Quarteroni and F. Saleri, Introduzione al calcolo scientifico: esercizi e problemi risolti con MATLAB. Springer Science & Business Media, 2007.

[10] A. Gilat, MATLAB: An introduction with Applications. John Wiley & Sons, 2009.

[11] J. R. Segarra Escandón, "Resolución numérica de ecuaciones diferenciales en Wolfram Mathematica," 2018.

[12] M. Golubitsky and M. Dellnitz, Linear algebra and differential equations using MATLAB. Brooks/Cole Publishing Co., 1999.

[13] S. Blanes Zamora, D. Ginestar Peiro, and M. D. Roselló Ferragud, Introducción a los métodos numéricos para ecuaciones diferenciales. Editorial Universitat Politècnica de València, 2014.

[14] M. G. Caligaris, G. Rodríguez, A. Favierib, and L. Laugeroa, "Desarrollo de habilidades matemáticas durante la resolución numérica de problemas de valor inicial usando recursos tecnológicos," Revista Educación en Ingeniería, vol. 14, no. 27, pp. 30–40, 2019.

[15] C. L. Rabeiro Martínez, A. Martínez Rodríguez, R. Gravier Hernández, Y. Bermudez Alfonso, and L. Gil del Valle, "Abacavir: una revisión actualizada sobre sus propiedades y aplicaciones," Rev Cubana Farm, vol. 49, no. 4, pp. 751–764, Dec. 2015.

Simulación del proceso de desarrollo de software: una aproximación con Dinámica de Sistemas y el Método de Larman

Simulation of the software development process: an approximation using System Dynamics and the Larman Method

German Lenin Dugarte Peña
Universidad Carlos III de Madrid
gdugarte@inf.uc3m.es
https://orcid.org/0000-0001-9760-7084

Maria Isabel Sanchez Segura
Universidad Carlos III de Madrid
misanche@inf.uc3m.es
https://orcid.org/0000-0002-2339-7851

Fuensanta Medina Domínguez
Universidad Carlos III de Madrid
fmedina@inf.uc3m.es
https://orcid.org/0000-0002-3249-2834

Antonio de Amescua Seco
Universidad Carlos III de Madrid
amescua@inf.uc3m.es
https://orcid.org/0000-0003-4355-6896

RECIBIDO 02/12/2019 • ACEPTADO 07/03/2020 • PUBLICADO 30/03/2020

RESUMEN

Poner en marcha cualquier proyecto de software involucra el consumo de recursos críticos. El ingeniero de software no puede experimentar con procesos de desarrollo sin ponerlos en marcha en proyectos reales, debido al tiempo que ello conlleva y a los elementos implicados, de modo que es importante contar con herramientas para previsualizar el resultado de la ejecución del proceso y cómo las variables de entorno le afectan, buscando anticipar en qué condiciones se va a desplegar el proceso. Este artículo presenta el modelado y simulación de un proceso de desarrollo de software por medio del enfoque de la Dinámica de Sistemas (DS), que permite representar gráficamente los elementos intervinientes en el proceso e incorporar la cantidad relevante de parámetros involucrados. Se tomó como referencia el modelo de estimación de costes COCOMO, que cuenta con una fundamentación teórico-práctica que avala su fiabilidad. Para la construcción del modelo, la referencia de sistema real fue el proceso software de Craig Larman (Método de Larman). El modelo de simulación presentado permite hacer estimaciones iniciales del comportamiento del proceso software, y de los elementos que lo conforman, durante el transcurso de un tiempo de simulación configurable. Se analizan variables de estado del sistema, que permiten concluir sobre efectos de los parámetros en el comportamiento del sistema en general. El modelo deriva en una herramienta de soporte a los equipos de gestión, y a las empresas que hacen de la Gestión de Proyectos Tecnológicos su negocio principal.

Palabras claves: Dinámica de Sistemas, Software Engineering Economics, Gestión del Proceso Software, Método de Craig Larman, Modelado y Simulación de Procesos Software.

ABSTRACT

The implementation of any software development process involves the consumption of critical resources. Software engineers cannot experiment with different development processes before starting them in real projects, due to the time that would entail and the number of elements that are involved, so it is vital to have tools that allow the pre-visualization of the results of executing the software development process and how the environmental variables affect it, thus being able to anticipate under what conditions the software development process will be deployed. This paper presents the modelling and simulation of a software development process using System Dynamics (SD), which allows the graphical representation of the elements intervening in the software process, and the incorporation of as many relevant elements as possible. As a software costs estimation reference, the COCOMO estimation model was used; which beyond being reliable has a theoretical-practical foundation. As an ideal, and real, software process system, the Craig Larman Software Process model was chosen, also known as the Larman Method. The simulation model developed here, allows one to make some initial estimation of the software process and its elements' behavior in the course of the simulation time. This is possible thanks to the observation and study of the system's state variables, empowering one to discern about the effect of changes in the parameters on the general process. This model becomes a tool for supporting Software Project Management teams and enterprises whose business care on Technological Projects Management.

Keywords: *Systems Dynamics, Software Engineering Economics, Software Process Management, Craig Larman's Method, Software Process Simulation Modelling.*

INTRODUCIÓN

En la gestión de procesos de ingeniería de software, cada vez se hace más evidente la necesidad de contar con herramientas que permitan una mayor supervisión y monitorización del proceso en su totalidad, con la idea de prever con antelación cual puede ser el mejor proceso a aplicar a un proyecto concreto según sus características, o qué decisiones se deben tomar en un momento determinado. El tiempo de desarrollo de los proyectos y el consumo de recursos necesarios para el proceso de ingeniería del software se han convertido en los factores clave que son sujeto de observación a la luz de distintas perspectivas o estilos que rigen los procesos de ingeniería de software; es así que se ha comenzado a hablar de técnicas ágiles para el desarrollo de software: XP, Scrum, etc. En este sentido, es evidente lo que Yu [1] y Smith [2] defienden sobre el hecho de que la necesidad de innovación en los equipos de software no está tan enfatizada en el producto software en sí, sino en la gestión de proyectos software, para lo que el modelado y la simulación, así como la gamificación [3], no solo son útiles sino necesarios.

El consumo de recursos en la aplicación de los procesos de ingeniería del software conlleva gastos que, dependiendo de cada proyecto, pueden ser de significativa importancia, por lo que estrategias que sirvan para reducir, prevenir o controlar el consumo excesivo de recursos, son

muy valoradas y tienden a tener una gran receptividad en las organizaciones. Según Kellner et al. [4] el modelado y simulación de procesos de desarrollo de software ha ido ganando el interés de los investigadores y desarrolladores dado que se convierte en una herramienta para analizar la complejidad del negocio del proceso software como un todo y para buscar respuesta a todas las interrogantes que se presentan en este proceso. Además, hay al menos dos poderosas razones para trabajar con modelos de procesos software: conseguir mejores formas de definir y guiar el desarrollo, mantenimiento y evolución del proceso; y, conseguir mejores formas de mejorar los procesos al nivel de actividades y del proceso como un todo [5].

El enfoque del modelado y la simulación ha estado orientado hacia procesos industriales, químicos, tangibles, medibles, cuyos parámetros están bajo el control absoluto de los implicados en el proceso. Sin embargo, en los últimos años se ha dado cierta evolución en el espectro de enfoques, motivada por la necesidad de acceso al control de aspectos como la gestión estratégica [6], la búsqueda de mejoras de los procesos, o el entrenamiento en cuanto a gestión de proyectos de software, desde una perspectiva cercana a los datos reales y con garantía de replicabilidad práctica [7].

Un posible impacto importante, que motiva a perseguir avances en el modelado y simulación de procesos software, es la reducción de la brecha que hay entre el equipo de desarrollo y las personas interesadas o involucradas como beneficiarios del producto a desarrollar. Una herramienta de simulación del proceso software puede representar un lenguaje común que permite el acercamiento entre desarrolladores y stakeholders, capaz de habilitar un entorno de diálogo en torno al proceso [8]. Esto es de interés para los desarrolladores de software, que cuentan con representaciones técnicas, como UML (entre otras muchas [9]), para abstraer y representar su sistema a desarrollar, las cuales pueden resultar confusas para los stakeholders, que suelen dominar un lenguaje mucho menos técnico y más cercano al mundo de la empresa, pudiendo perder interacción y comprensión suficiente del dominio del problema, si éste es abordado en conjunto con el equipo de desarrollo [10], [11].

Los avances tecnológicos surgen y se desarrollan a un ritmo bastante acelerado y en paralelo a la disponibilidad de equipos potentes de cómputo surgen exigencias cada vez mayores de rendimiento por parte de los equipos humanos de desarrollo de software, así como de contar con estrategias y herramientas que permitan estimar el desempeño de los equipos de producción y los procesos que ellos implican. Debido a esto es menester contar con sistemas simulación y predicción de procesos software que apoyen y respalden la toma de decisiones en torno a todo el proceso de desarrollo.

Para ahondar en torno a esto, se puede partir de dos bases de conocimiento concretas que ya cuentan con numerosos trabajos de investigación y proyectos y desarrollos concretos, lo que se detallará a continuación.

Por un lado, existe un marco conceptual claramente estructurado para el modelado y simulación de sistemas en general (Simulation Modelling), con múltiples aplicaciones y usos en áreas como la industrial, ambiental, química, electromecánica, etc. [6]; y por otro lado, se cuenta con una base conceptual en torno a la comprensión, análisis, diseño y monitorización de procesos de desarrollo de software; lo que permite explorar diferentes modelos de proceso, desde los más clásicos como el modelo estructurado en cascada (Waterfall), hasta modelos orgánicos y menos rígidos como el Rational Unified Process (RUP) [12], o el que se explica más adelante en este trabajo, el Método de Craig Larman [13].

A partir de las dos bases de conocimiento mencionadas anteriormente, surge un interés de investigación en torno al modelado y simulación de procesos software, campo hasta el momento poco explorado y en el que la diversidad y dinamismo de los factores intervinientes aporta un grado de complejidad que afecta notablemente el comportamiento de los equipos de desarrollo y por ende todo el comportamiento organizativo. Se presenta aquí un trabajo en el que se tomó el Método de Larman y se diseñó un modelo de simulación para representar el funcionamiento de dicho método como proceso organizativo de desarrollo de software, dando continuidad al trabajo de Dugarte-Peña [14][8].

El creciente auge en torno al desarrollo de software, las exigencias cada vez más complejas sobre las capacidades de los productos software y el deseo, cada vez mayor, por parte de la industria y de la academia, de comprender y acercar los productos software a la realidad a la que sirven; hacen que sea una necesidad importante contar con herramientas para comprender estos sistemas y su complejidad. El modelado y simulación de los procesos de desarrollo de software representa una oportunidad importante para ambos colectivos en el sentido de que con un mínimo consumo de recursos permite hacer exploraciones sobre el sistema emulado como si se tratase del sistema real, apoyando la toma de decisiones y proyectando escenarios que en la mayoría de los casos son inexplorables o inaccesibles por los altos costes que suponen.

El Método de Larman, ilustrado en la Figura 1, al ser iterativo, incremental y dirigido por casos de uso, ha sido escogido en este trabajo como modelo ideal del proceso a ser simulado. Para hacer el modelado y simulación de este sistema "real" [1], se ha decidido trabajar con el enfoque de la Dinámica de Sistemas por la amplia documentación existente sobre su uso y aprovechamiento y por el valor agregado que le otorga el carácter sistémico que inspiró su nacimiento y las herramientas existentes para su uso. Además, se justifica su uso por la aún vigente importancia de los estudios *in silico* en el ámbito de la ingeniería del software [15].

La Dinámica de Sistemas proporciona todo el marco conceptual para la construcción del modelo.

[1] Se usa la palabra "real" para diferenciar del sistema simulado, siguiendo la metodología de Modelado y Simulación de Sistemas. EL sistema "real" hace referencia a la situación compleja de interés, que en este caso es el proceso complejo de desarrollo, teniendo en cuenta todos sus factores, interrelaciones y perspectivas. El sistema simulado, en cambio, será una abstracción restringida de esta "realidad compleja".

Se ha decidido trabajar con VENSIM [16] por ser una herramienta con amplia documentación a disposición de la comunidad, con una gran colección de experiencias de uso que argumentan su fiabilidad y por ofrecer a la comunidad académica una versión gratuita para fines académicos, como es el caso de este trabajo. Además de permitir la representación del sistema real, VENSIM es fácilmente manipulable y modificable para experimentar con variaciones de los parámetros usados en el modelo, permitiendo así modificaciones del estado del sistema que se pueden observar, medir y repetir, resultando idóneo para modelado y la simulación de sistemas, no sólo en el campo de la Ingeniería del Software sino en general [17].

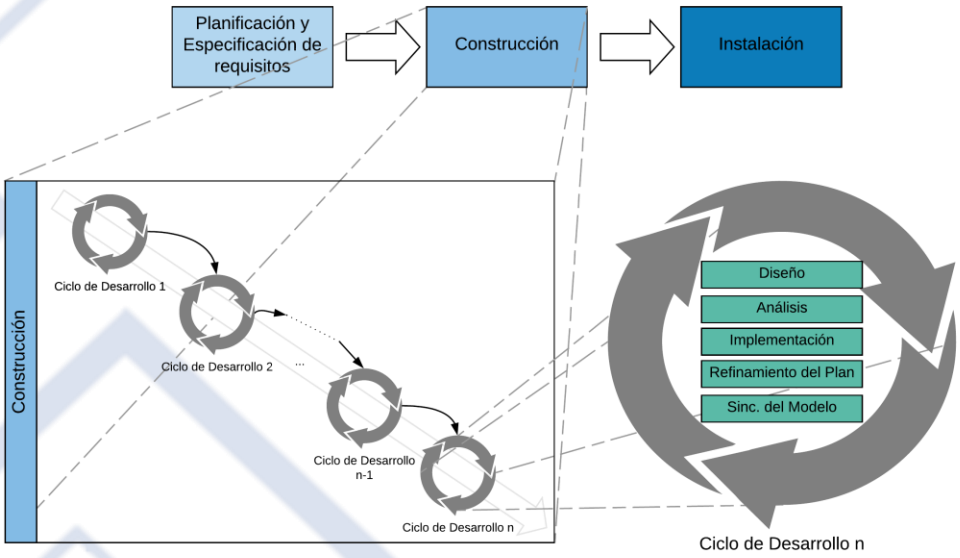

Figura 1. Bosquejo conceptual del método de Larman.

Selección de Parámetros del Proceso Software

En este modelo de simulación del proceso software se hará uso de los parámetros del proceso software que el modelo COCOMO ofrece. COCOMO® II (Constructive Cost Model II), es una versión más completa y mejorada de la herramienta COCOMO® (Constructive Cost Model), consistente de la implementación de un modelo de estimación implementado por Barry Boehm a finales de los años setenta, en el seno de la University of Southern California [18].

COCOMO® II permite la estimación de coste, esfuerzo y programación de tareas en el proceso de planificación de un nuevo desarrollo de software. Además, permite la manipulación de tres submodelos: el de Composición de Aplicaciones (Applications Composition), el de Diseño Temprano (Early Design), y el de Modelos Post-arquitectura (Post-architecture models). Con respecto a los aspectos de un proyecto de desarrollo de software, Cocomo II se fija en cuatro tipos de atributos de un proceso software para hacer la estimación. Una descripción detallada de estos atributos, las funciones de estimación que usa y como se interrelacionan los atributos entre

sí, se encuentra disponible en [19]. Debido a su importancia para este trabajo, el listado de atributos por tipo se presenta en la Tabla 1. Se muestra también una escala numeral que aporta los valores referenciales de entrada a los modelos y fórmulas matemáticas de las que se hace uso para estimar el coste y esfuerzo de un proyecto.

Tabla 1. Parámetros-Atributos del modelo de Estimación de costes COCOMO II.

COCOMO II: Parámetros y atributos		Valor					
Nombre	**Atributo**	**Muy bajo**	**Bajo**	**Nominal**	**Alto**	**Muy alto**	**Extra alto**
Atributos de Software							
RELY	Fiabilidad	0.75	0.88	1.00	1.15	1.40	__
DATA	Tamaño de Base de Datos	__	0.94	1.00	1.08	1.16	__
CPLX	Complejidad	0.70	0.85	1.00	1.15	1.30	1.65
Atributos de Hardware							
TIME	Restricciones de tiempo de ejecución	__	__	1.00	1.11	1.30	1.66
STOR	Restricciones de memoria virtual	__	__	1.00	1.06	1.21	1.56
VIRT	Volatilidad de la máquina virtual	__	0.87	1.00	1.15	1.30	__
TURN	Tiempo de respuesta	__	0.87	1.00	1.07	1.15	__
Atributos de Personal							
ACAP	Capacidad de análisis	1.46	1.19	1.00	0.86	0.71	__
AEXP	Experiencia en la aplicación	1.29	1.13	1.00	0.91	0.82	__
PCAP	Calidad de los programadores	1.42	1.17	1.00	0.86	0.70	__
VEXP	Experiencia en la máquina virtual	1.21	1.10	1.00	0.90	__	__
LEXP	Experiencia en el lenguaje	1.14	1.07	1.00	0.95	__	__
Atributos del Proyecto							
MODP	Técnicas actualizadas de programación	1.24	1.10	1.00	0.91	0.82	__
TOOL	Utilización de herramientas de software	1.24	1.10	1.00	0.91	0.83	__
SCED	Restricciones de tiempos de desarrollo	1.22	1.08	1.00	1.04	1.10	__

Modelado y Simulación del Proceso Software

Teniendo como referencia la metodología de [20] para modelado con Dinámica de Sistemas, y la utilidad que tiene para ser explotada en el ámbito de la ingeniería del software[21]–[23], primero se hizo una descripción del sistema a modelar, después una representación de las causalidades existentes entre los elementos del sistema, seguido por la identificación y representación de los elementos propios de la Dinámica de Sistemas, y finalmente la configuración del modelo para su posterior uso, evaluación, y análisis de funcionalidad.

El sistema que se va a modelar está compuesto por un conjunto de elementos intervinientes en el proceso de desarrollo de software como lo son: personas (analistas, programadores, directores, etc.), equipos informáticos, requisitos, activos de proceso software (código, especificaciones, prototipos, documentación), y elementos propios del entorno del sistema como las restricciones o fronteras (tiempos, capacidades máximas, etc.). Todos estos elementos están interrelacionados

de alguna manera en alguna o en todas las fases que comprende el proceso de desarrollo de software, sin embargo, el funcionamiento de este conjunto de elementos sobrepasa esa mera descripción, resultando realmente en un comportamiento dinámico y lleno de causalidades complejas y variantes en el tiempo.

El Método de Craig Larman se compone de tres fases o macro etapas claramente diferenciadas una de otra, y en esencia comunes a la gran mayoría de modelos y métodos de desarrollo de software: Planificación y Especificación de requisitos, Construcción e Instalación.

Diagrama causal del modelo.

A continuación, como si se tratase de una especie de "zoom" o "acercamiento" hacia lo que realmente se quiere representar, se muestra primero un macro diagrama que muestra cómo se relacionan las tres fases del proceso de desarrollo ya explicadas: planificación y especificación de requisitos, construcción e instalación.

Seguidamente se muestra un primer grado de dinamismo e interactividad que se da en el proceso de software consecuencia de que en la fase de construcción están implicados no uno sino varios ciclos de desarrollo que son necesarios para alcanzar el grado de madurez aceptable del producto software que se desea construir.

Finalmente se presenta un diagrama causal detallado del subproceso de la fase de construcción, lo que es un precedente y requisito para construir el modelo de Dinámica de Sistemas, es decir, el modelo en términos de niveles y flujos dinámicos.

Macro diagrama del Proceso General de Desarrollo de Software de Craig Larman.

Las tres fases del proceso están interconectadas, y, como se observa en las Figuras 2 y 3, suelen tener una fuerte interacción durante todo el proceso de desarrollo. Aunque la fase de planificación y especificación de requisitos es la inicial, desde las fases de construcción y de instalación se hacen constantes referencias a esta fase para realizar ajustes que corresponden a actividades de planificación y especificación de requisitos. Lo mismo ocurre entre las fases de Construcción e Instalación, lo que significaría que durante la instalación pueden generarse solicitudes de "re-engineering" de actividades que corresponden a la fase de construcción, es decir, se hacen cambios en los requisitos. Sin embargo, el núcleo de todo el proceso, o la fase en que la carga de trabajo es significativamente mayor, es la fase de construcción, que implica fuerte consumo de recursos para lograr obtener el producto listo para implantación a partir de su especificación.

Macro diagrama de la fase "Construcción" de un proceso de desarrollo de software.

La fase de Construcción de un proceso de desarrollo de software tiene por sí misma cierta complejidad y comportamiento dinámico que merece ser estudiado y comprendido, tanto desde un punto de vista sistémico y sinérgico, como comprendiendo a su vez las distintas subetapas que contiene. Estas subetapas son los llamados ciclos de desarrollo que se llevan a cabo formalmente con posterioridad a la Planificación y Especificación de Requisitos y derivan en la presentación de un producto software que luego estará disponible para ser llevado a la fase de instalación.

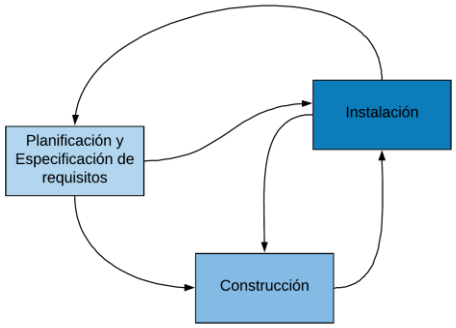

Figura 2. Diagrama general de fases del Método de Larman.

Como se puede ver en la Figura 3, los ciclos de desarrollo están relacionados entre sí. En cada ciclo de desarrollo se van atendiendo casos de uso de los que se definen en la planificación y especificación de requisitos. El orden en que estos casos de uso son atendidos responde a la priorización que se hace sobre ellos, y, además, un caso de uso puede ser parte de más de un ciclo de desarrollo si la complejidad del caso lo amerita.

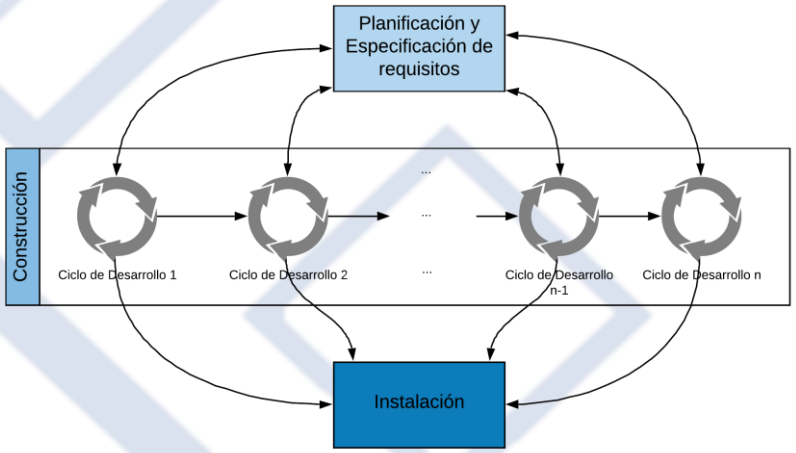

Figura 3. Primer desdoblamiento de la complejidad del proceso: Construcción.

La cantidad de ciclos de desarrollo no es exacta, sino que se determina para cada caso en particular según la complejidad del proyecto y de sus casos de uso. En la Figura 3, observamos un primer "acercamiento" o "zoom" sobre el proceso general de desarrollo, en el que se muestra que la fase de construcción implica a su vez varios ciclos de desarrollo y que estos no son independientes uno del otro sino que están relacionado y hay relaciones de causalidad entre ellos: Para el ejemplo de la Figura 3, el ciclo 1 produce efectos en ciclos 2 y 3, y el ciclo 2 que se ha visto influenciado por el ciclo 1, también tendrá relevancia sobre el ciclo 3.

Diagrama en detalle: Fase de construcción

A continuación, en la Figura 4, se presentará el diagrama causal del sistema real que se está modelando en este trabajo. En este diagrama, a forma de segundo "acercamiento" o "zoom" sobre el sistema real, se han incorporado todos los elementos que se considera tienen algún efecto en la fase de construcción del proceso software. Se ha agregado la mayor cantidad de elementos posibles con la finalidad de hacer la representación pictográfica más representativa.

Siguiendo el sentido de las flechas, se puede apreciar relaciones de causalidad entre los elementos a los extremos de las flechas, lo que nos permite apreciar la complejidad del proceso software al haber no sólo una cantidad considerable de flechas sino también un número elevado de realimentaciones, lo que según [20] justifica el uso de la Dinámica de Sistemas para modelar sistemas como éste. A continuación, se presentan los elementos clasificados en términos de la dinámica de sistemas (Detalles se han omitido en este artículo, pero las definiciones extensas se alinean con [13], [24]).

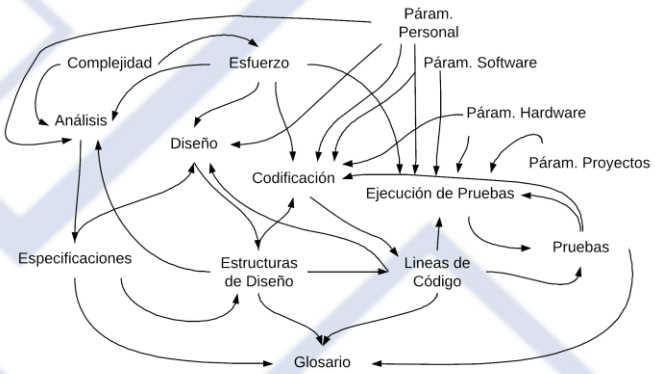

Figura 4. Diagrama causal de la fase "Construcción" del Método Larman de ingeniería del Software.

Identificación de Acumuladores, Niveles o Stocks

Para el proceso de Ingeniería del Software aquí presentado, los elementos que tienden a acumularse en el tiempo y que por ende representan "niveles del sistema" en el dominio de la

Dinámica de Sistemas son: Especificaciones, Estructuras de Diseño, Código, Pruebas realizadas y un nivel cuya funcionalidad es transversal pero no crítica para todo el proceso: Glosario.

Identificación de flujos o tasas de crecimiento y decrecimiento.

A continuación, se presentan los elementos del sistema que representan de alguna manera flujos de crecimiento y decrecimiento de los niveles anteriormente especificados: Análisis, Diseño, Implementación, Corrección y Pruebas.

MODELO COMPUTACIONAL

El modelo que se presenta, debido a su tamaño, comprende tres vistas de modelo, tomando ventaja de la propiedad de modelado paralelo del software Vensim. Inicialmente se pretendía construir un modelo general que englobara las tres fases del Método de Larman, sin embargo, durante el proceso de modelado fue necesario delimitar el proceso software de manera que las fases de "Planificación y Especificación de Requisitos" e "Instalación" sólo se llevan a cabo una vez, e implican tareas concretas, mientras que la fase de "Construcción" tiene implícita toda una dinámica (e iteratividad) en sí misma. La Figura 5 muestra el submodelo de la fase de Planificación y Especificación de Requisitos.

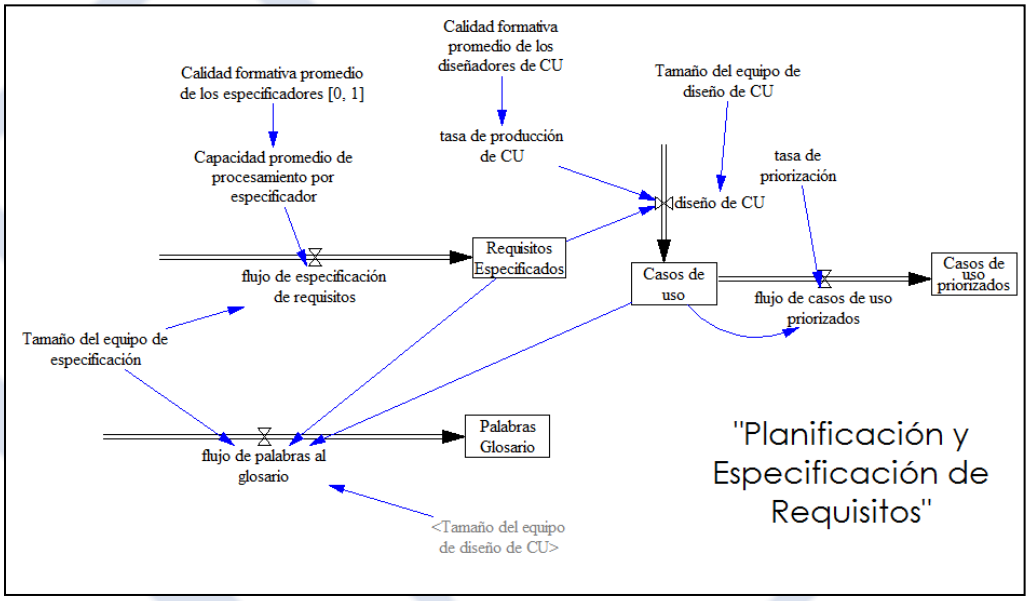

Figura 5. Submodelo de simulación para la fase "Planificación y Especificación de Requisitos".

Seguidamente se presenta el modelo de simulación para la fase de construcción (en tres vistas, o niveles de abstracción que forman parte del mismo modelo). La primera vista, presenta de forma esquemática los parámetros del proceso que afectan el proceso software que fueron

explicados anteriormente. Estos parámetros tienen efecto en distintas partes del modelo de simulación, sin embargo, se han conglomerado en esta primera vista para facilitar su manipulación y la fácil calibración del modelo al momento de hacer pruebas variando sus valores. En esta primera vista (Figura 6) también se ha incorporado la parte del modelo en la que se implementan las ecuaciones usadas por [19] para la estimación de esfuerzo.

La segunda vista del modelo en VENSIM (Figura 7) es la parte central del modelo, y en ella se han agregado los elementos principales representativos del proceso de desarrollo de software.

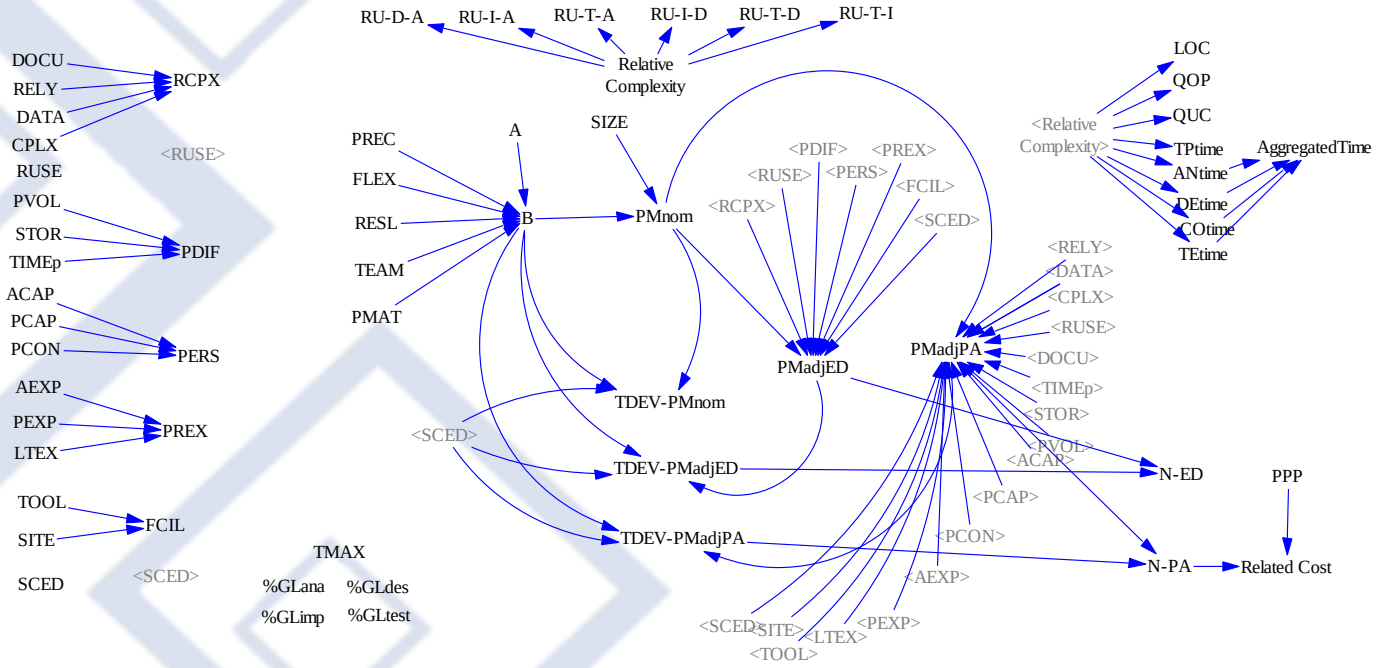

Figura 6. Vista 1: Submodelo de configuración de parámetros y estimación.

En la Figura 6 se pueden observar los elementos del modelo COCOMO II de estimación, y como éstos afectan la estimación tanto en el modelo "Early Design" (ED) como "Post Architecture" (PA). Se han respetado los acrónimos del modelo COCOMO tanto para los atributos del modelo (Variables auxiliares en esta vista) como para las variables producto de la estimación (Estimaciones ED y PA).

La Figura 7 muestra la vista del modelo que implementa las los elementos del proceso de desarrollo de software y cómo estas se van acumulando a medida que el tiempo de proyecto transcurre. Las fases que se "acumulan", es decir, que se van cumpliendo durante el proceso son: Analysis, Design, Code, y Test. Hay otro nivel que se acumula referente a la incorporación de términos al Glosario de un proyecto. Los flujos que nutren estos niveles determinan el ritmo con el que crecen o decrecen y son dependientes de distintos factores que afectan el proceso, que en este caso se tomaron del modelo COCOMO de estimación, como se ha explicado anteriormente.

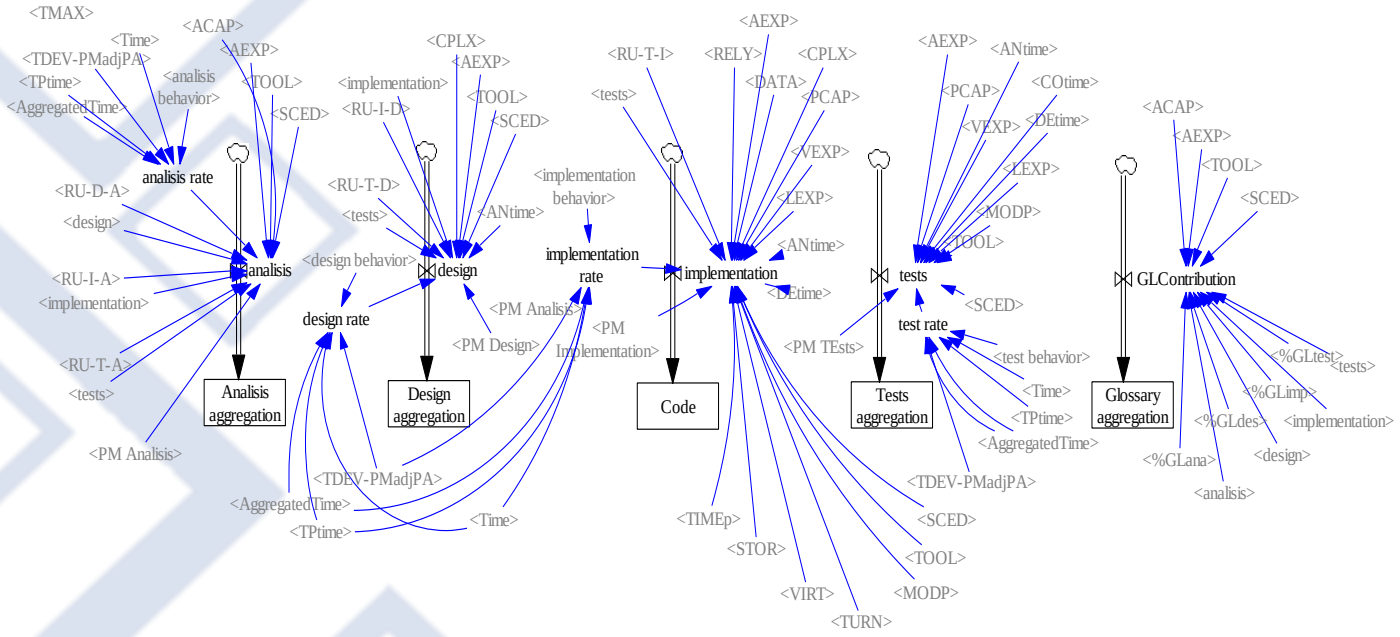

Figura 7. Vista 2: Submodelo de la fase de "Construcción" en un proceso de desarrollo.

La tercera vista (Figura 8) se ha agregado para representar el consumo paulatino del esfuerzo correspondiente a cada una de las etapas de la fase de construcción de software. Se puede observar que el esfuerzo distribuido está acumulado en los niveles PM Analysis, PM Design, PM Implementation, y PM Tests. Los flujos que salen de cada nivel definen el ritmo en que este esfuerzo es consumido, y como se puede observar, son afectados por distintas variables del proceso.

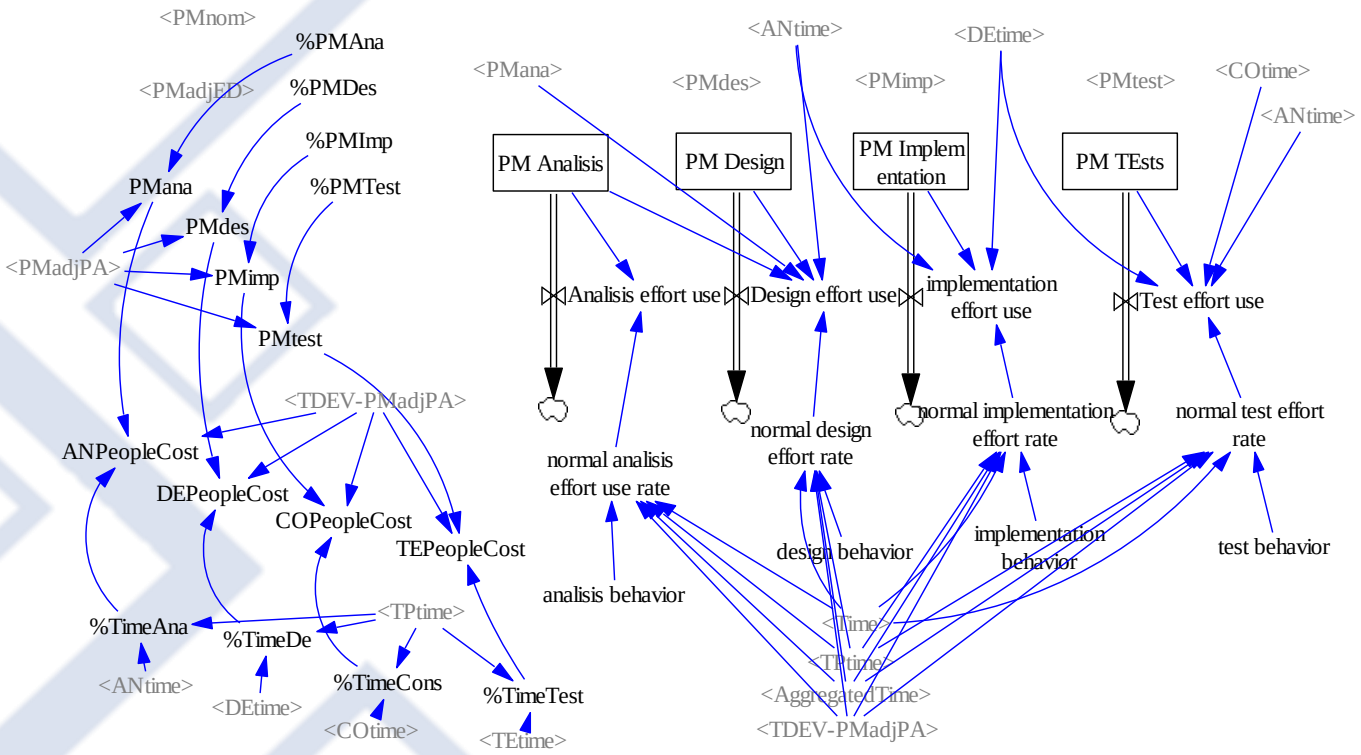

Figura 8. Vista 3: Submodelo de distribución del esfuerzo en el proceso de desarrollo de software

EJECUCIÓN Y DISCUSIÓN DE LA SIMULACIÓN DEL MODELO COMPUTACIONAL

Se ejecutó el modelo con los parámetros de COCOMO en sus valores "nominales", obteniéndose lo que se esperaría sea el comportamiento normal del sistema simulado. Se puede observar una gran cantidad de factores y su variación en el tiempo; sin embargo, se enfocará la atención en mostrar aquellos aspectos del proceso que son de interés en la gestión de proyectos software: **el comportamiento del sistema como un todo, el consumo de los recursos en el transcurso del tiempo, y los comportamientos de los acumuladores por tratarse de "variables de estado" que permiten crear panoramas generales del funcionamiento del sistema.**

En la Figura 8, se observa el comportamiento general del sistema con la figuración de parámetros nominales. Se puede observar como la realización de las actividades del proceso de construcción (representadas en el eje y como unidades atómicas de tareas por cada fase), aunque con ciertas simultaneidades, se concentra en su mayoría en distintas etapas (temporales) del proceso general. El análisis y el diseño tienden a concentrarse en la primera mitad del ciclo de vida del proceso, mientras que la implementación (y su correspondiente construcción de código fuente) se concentra en la mitad del proceso y las pruebas hacia la segunda mitad del ciclo de vida del proceso. Este comportamiento presenta cierta similitud con el proceso de software descrito en el

Rational Unified Process, lo que da una primera idea de la aproximación de este modelo de simulación a los modelos teóricos existentes en la literatura.

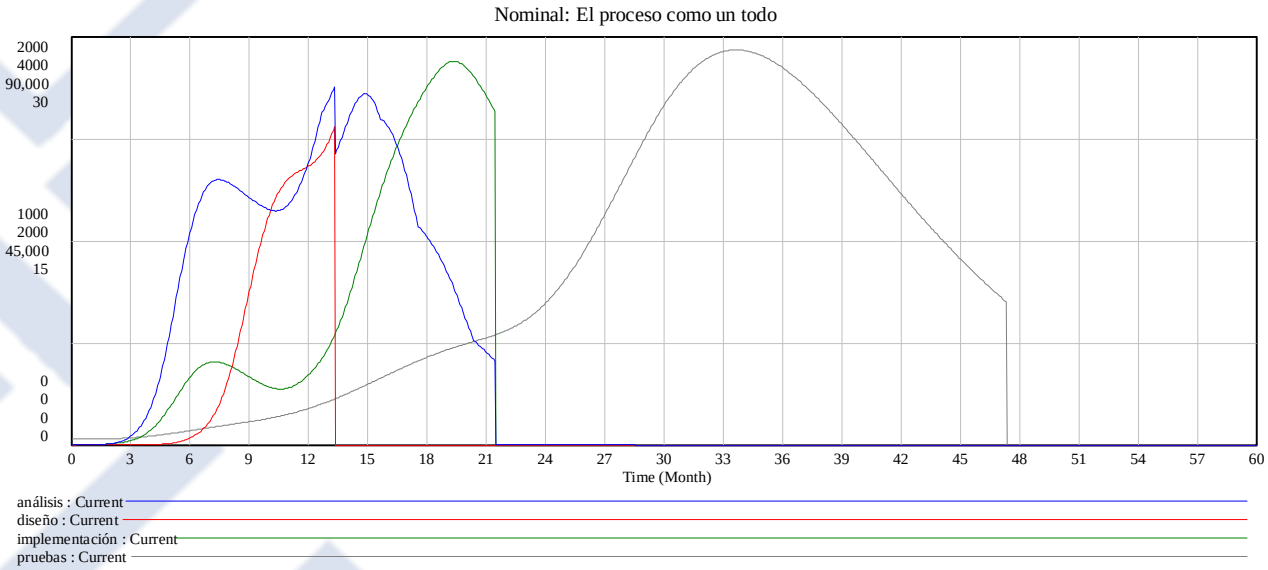

Figura 8. Las fases de la "construcción" del proceso software: modelo nominal.

Si se piensa en las actividades del proceso de construcción como metas que deben ser alcanzadas en el transcurso del tiempo, la Figura 9, es una ilustración del comportamiento de la maduración del alcance de estas metas. Se puede ver que los primeros conjuntos de actividades en completarse son los correspondientes a “Análisis” y “Diseño”, seguido por el correspondiente a la construcción del código, y finalmente las actividades de pruebas, que al principio del proceso eran muy pocas, pero que dominan el proceso hacia el final, cuando el producto se prepara para ser entregado.

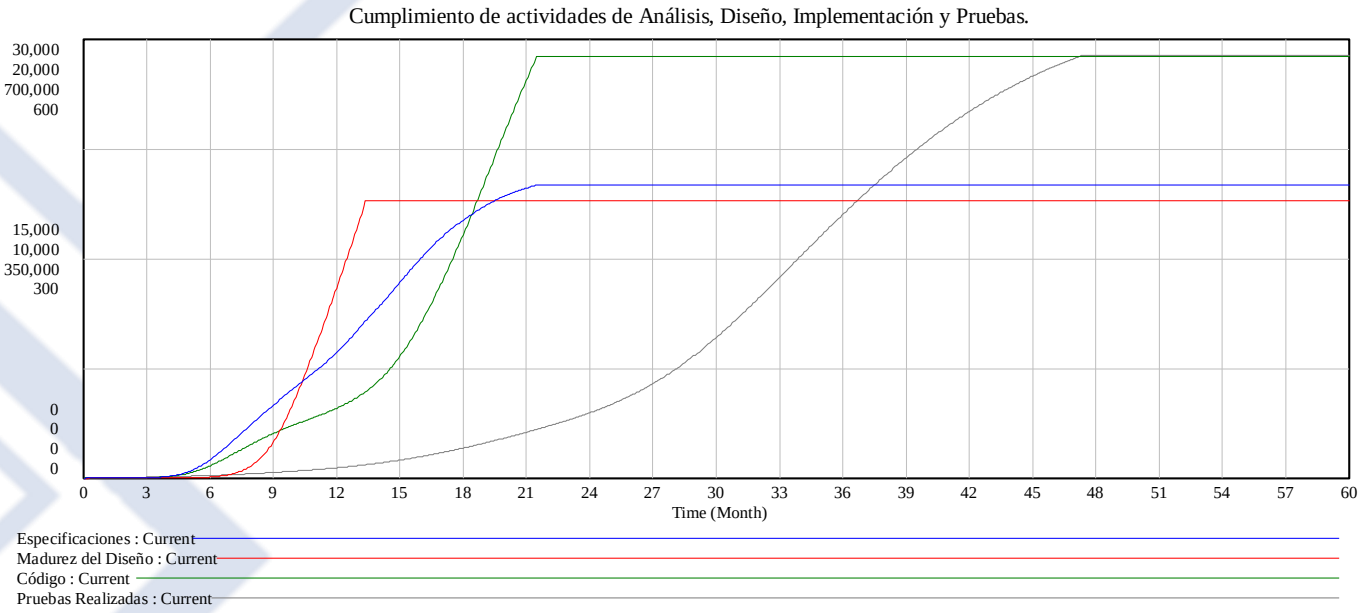

Figura 9. Maduración de las fases del proceso de construcción: modelo nominal.

Una forma también interesante de verificar el funcionamiento del proceso es observando el comportamiento del "esfuerzo", dado que este se distribuye durante todo el proceso en los distintos tipos de actividades que se llevan a cabo. En la Figura 10, (sin escalar) se ve que el esfuerzo consumido en el proyecto se distribuye entre análisis, diseño, codificación y pruebas respectivamente. En dicha imagen se observa en qué momentos de todo el proceso hay mayores flujos de esfuerzo en el proyecto, sin embargo, nótese que como es de esperar, las escalas no se corresponden; ya que, en cuanto a cantidades, el esfuerzo dedicado a algunas actividades de diseño, es mucho mayor que el dedicado a las pruebas, por ejemplo.

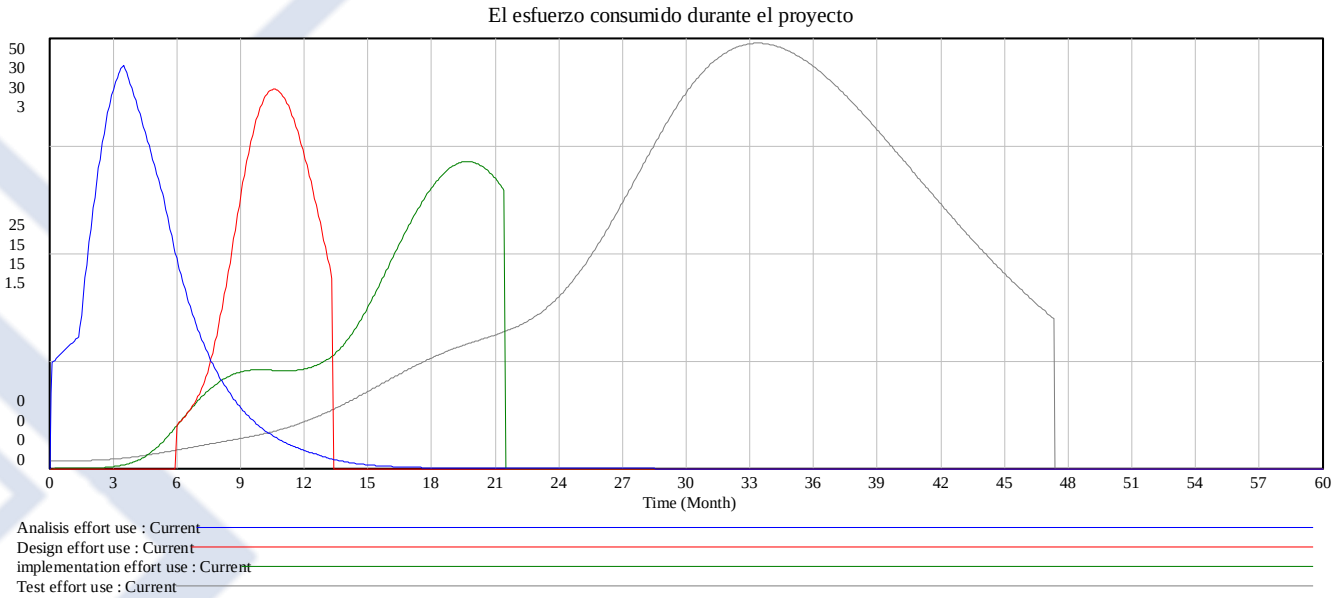

Figura 10. Esfuerzo consumido en el proceso de construcción, separado por fases: modelo nominal.

Adicionalmente, el modelo ha permitido a los autores crear escenarios con los que estudiar el comportamiento del sistema y cómo este se ve afectado al variar los elementos de estimación que afectan el proceso. Los resultados parciales se presentan en [8] y se refinan para un futuro trabajo enfocado en el efecto que tienen los parámetros de estimación del producto software en el desempeño del proceso.

CONCLUSIONES

En este trabajo se decidió trabajar con la Dinámica de Sistemas como enfoque para la construcción del modelo de proceso software, basados en el hecho de que el proceso software, lejos de ser metódico y procedimental resulta ser complejo, variante y en muchas ocasiones contraintuitivo y con muchos fenómenos emergentes e impredecibles.

Para la construcción del modelo, y atendiendo a la metodología tomada como referencia [20]; se hizo primero una profunda inmersión en el sistema a modelar, lo que se tradujo en una comprensión profunda y sistémica del proceso de ingeniería de software del Método de Larman, su funcionamiento, fases, características, y propiedades, constituyendo lo que es un primer nivel de abstracción del proceso. Parte de esta inmersión fue la exploración de enfoques que describieran los parámetros que afectan el proceso, para lo que resultó de gran utilidad y aprovechamiento hacer uso del modelo de costes COCOMO II, que tiene definidos una serie de parámetros comunes al proceso que tienen incidencia en el desarrollo y en el consumo de recursos relacionado. En el modelo que aquí se presentó se hizo uso de estos parámetros como una forma de incrementar la representatividad del modelo, lo que permitió incorporar los efectos de los

parámetros del proceso, sobre el funcionamiento del proceso como un todo, y de esta manera estimar posibles efectos y comportamientos esperados a partir de ciertas configuraciones de parámetros; representando esta propiedad un aporte significativo para quienes quieren estudiar el proceso software con mínimos costes y por medio de una manera práctica, manejable, utilizable por múltiples usuarios y con un importante potencial para dar soporte a la toma de decisiones en los equipos de gestión de procesos software.

Lo anterior fue el fundamento para la construcción de los diagramas causales representativos, inicialmente de todo el proceso, y después, a manera de un zoom dentro del sistema general, de la fase de construcción del proceso de ingeniería de software. El aporte principal en este aspecto vino al momento de modelizar dicho sistema (el subproceso de construcción) haciendo uso del enfoque de la Dinámica de Sistemas. En esta modelización, la identificación de variables, niveles y flujos que interactúan en el sistema, representan un segundo nivel de abstracción del sistema, de suma importancia y utilidad, ya que aquí se identifican relaciones entre los elementos del sistema que no siempre son evidentes a simple vista ni con una exploración superficial. Posteriormente, todas las relaciones, causalidades y ciclos de realimentación identificados fueron implementados en el diseño, configuración, validación y manipulación del modelo en sí, logrando un nivel de representatividad del sistema bastante amplio y conformando una de las principales fortalezas de este trabajo, **al ser un modelo representativo del proceso software, que toma en cuenta un gran número de elementos y parámetros de distinto tipo que inevitablemente tienen influencia sobre el sistema y que no se deben ignorar.**

También han surgido argumentos para sustentar la afirmación de que **es necesario ahondar en lograr contar con un enfoque metodológico estructurado que sirva de contexto para estudiar problemas similares al que aquí se enfrentó: simular procesos de ingeniería del software.** Se trataría de un enfoque en el que los criterios para escoger un enfoque u otro sean un activo a disposición del usuario o investigador, cosa que permitiría múltiples beneficios tanto para las ciencias en general como para los campos de ésta que se dedican a comprender los fenómenos organizativos, entre los que se cuentan los del proceso software.

Igualmente, es de importancia para investigaciones futuras, la ejecución de experimentos en los que bajo las mismas condiciones y con la misma fijación de parámetros se exploren simulaciones basadas en diferentes modelos de un mismo sistema software "real". Esto permitiría por un lado contrastar un modelo con otro con respecto a su grado de representación de la complejidad del sistema real, y por el otro permitiría una observación sistémica que permitiría estimar otros aspectos diferenciadores de los modelos y enfoques, como son la funcionalidad, grado de usabilidad y reproducibilidad de un software; asunto que se presenta como uno de los intereses para evolucionar el cuerpo actual de conocimiento de la Ingeniería del Software [25].

REFERENCIAS

[1] W. D. Yu, "A Modeling Approach to Software Cost Estimation," IEEE J. Sel. Areas Commun., vol. 8, pp. 309–314, 1990.

[2] C. Smith, "Improving Service While Controlling Costs," IEEE Softw., vol. March, pp. 95–96, 1991.

[3] C. Alejandro and M. Ruiz, "Coverage of ISO/IEC 12207 Software Lifecycle Process by a Simulation-Based Serious Game," in SPICE 2016: Software Process Improvement and Capability Determination, 2016, pp. 56–70, doi: https://doi.org/10.1007/978-3-319-38980-6_5.

[4] M. I. Kellner, R. J. Madachy, and D. M. Raffo, "Software process simulation modeling: Why? What? How?," J. Syst. Softw., vol. 46, no. 2–3, pp. 91–105, 1999, doi: 10.1016/S0164-1212(99)00003-5.

[5] S. T. Acuña and M. I. Sánchez-Segura, New Trends in Software Process Modeling. Singapore: World Scientific Publishing Co. Pte. Ltd., 2006.

[6] A. Greasley, Simulation modelling for business. Routledge, 2017.

[7] M. I. Lunesu, J. Münch, M. Marchesi, and M. Kuhrmann, "Using simulation for understanding and reproducing distributed software development processes in the cloud," Inf. Softw. Technol., vol. 103, no. July, pp. 226–238, 2018, doi: 10.1016/j.infsof.2018.07.004.

[8] G. L. Dugarte-Peña, "Modelado y Simulación de un Proceso de Desarrollo de Software dirigido por el Método de Craig Larman: Una aplicación de la dinámica de sistemas. (Modelling and Simulation of a Craig Larman Methods' Software Development Process: A System Dynamics Approach," Universidad Carlos III de Madrid, 2015.

[9] J. A. García-García, J. G. Enríquez, and F. J. Domínguez-Mayo, "Characterizing and evaluating the quality of software process modeling language: Comparison of ten representative model-based languages," Comput. Stand. Interfaces, vol. 63, no. October 2018, pp. 52–66, 2019, doi: 10.1016/j.csi.2018.11.008.

[10] R. Vicente, "Modelamiento semántico con Dinámica de Sistemas en el proceso de desarrollo de software," Iber. J. Inf. Syst. Technol., no. 10, pp. 19–33, Dec. 2012, doi: 10.4304/risti.10.19-34.

[11] S. Robertson, "Learning from Other Disciplines," 2005.

[12] I. Jacobson, G. Booch, and J. E. Rumbaugh, The unified software development process. Boston, MA: Addison-Wesley Longman Publishing Co., Inc., 1999.

[13] C. Larman, Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3rd ed. Prentice Hall, 2004.

[14] G.-L. Dugarte-Peña, "Software Engineering under the prism of System Dynamics," in XXIII Jornadas. Internacionales de Ingeniería de Sistemas. Universidad Católica de Santa María., 2016.

[15] G. H. Travassos and M. Barros, "Contributions of In Virtuo and In Silico Experimentes for the Future of Empirical Studies in Software Engineering," 2nd Work. Work. Ser. Empir. Softw. Eng. Futur. Empir. Stud. Softw. Eng., no. January, pp. 1–14, 2003.

[16] Ventana Systems, "VENSIM." Ventana Systems, 2011.

[17] J. M. García, Teoría y ejercicios prácticos de Dinámica de Sistemas. Independently Published, 2018.

[18] USC COCOMO II: User´s manual. Center for Software Engineering, USC, 2003.

[19] B. Boehm, "COCOMO II Model Definition Manual," vol. 87, no. 5 Suppl, p. i, 2012, doi: 10.4269/ajtmh.2012.875suppack.

[20] J. Sterman, Business Dynamics: Systems Thinking and Modeling for a Complex World. USA: McGraw-Hill, 2000.

[21] B. B. Nicolau De França and G. Horta Travassos, "Reporting guidelines for simulation-based studies in software engineering," IET Semin. Dig., vol. 2012, no. 1, pp. 156–160, 2012, doi: 10.1049/ic.2012.0019.

[22] H. Zhang, B. A. Kitchenham, and D. Pfahl, "Reflections on 10 Years of Software Process Simulation Modeling: A Systematic Review," in Making Globally Distributed Software Development a Success Story, Q. Wang, D. Pfahl, and D. M. Raffo, Eds. Springer, 2008, pp. 345–356.

[23] H. Zhang, B. Kitchenham, and D. Pfahl, "Software Process Simulation Modeling: Facts, Trends and Directions," 2008 15th Asia-Pacific Softw. Eng. Conf., pp. 59–66, 2008, doi: 10.1109/APSEC.2008.50.

[24] X. Ferré-Grau and M.-I. Sanchez-Segura, "Desarrollo Orientado a Objetos con UML," pp. 1–38, 2013.

[25] M.-I. Sanchez-Segura, A. Jordan, F. Medina-Dominguez, and G.-L. Dugarte-Peña, "Software Engineers must speak the Systemic Intangible Process Assets Language," 2016.

Componente de indexación de huellas dactilares basado en características globales

Fingerprints indexing component based on global features

Estela Odelsa Martín Coronel
Universidad de las Ciencias Informáticas
eomartin@uci.cu
https://orcid.org/0000-0001-5534-2479

Adrián Hernández Barrios
Universidad de las Ciencias Informáticas
adrian2monk@gmail.com

Allens Torres Ruíz
Universidad de las Ciencias Informáticas
allenstr90@gmail.com

RECIBIDO 04/12/2019 • ACEPTADO 07/03/2020 • PUBLICADO 30/03/2020

RESUMEN

El campo de orientación de la huella dactilar como característica global es fundamental para el desarrollo de una estrategia de indexación, aportando estabilidad y disminuyendo los tiempos de respuesta durante el proceso de búsqueda de un sistema de identificación basado en huellas dactilares. El empleo de grafos relacionales de atributos y máscaras dinámicas permite explotar las características brindadas por el esquema de particionado para la búsqueda. De cada una de las estrategias propuestas se determinó el índice de penetración de la base de datos, así como el error en que incurren los algoritmos de indexación implementados. Para corroborar los resultados obtenidos fueron empleados los bancos de datos aportados por la Competencia de verificación de huellas dactilares. La implementación computacional de las estrategias de búsquedas propuestas demuestra las virtudes que ofrece el campo de orientación para dirigir el proceso de búsqueda, logrando reducir el número de comparaciones en un 53.24 % como promedio, comportándose de manera estable ante imágenes de huellas dactilares de diferente calidad. Basado en los resultados del estudio se concluyó que la adopción de dichas estrategias de indexación permitirá reducir el tiempo de respuesta en el módulo de identificación de individuos basado en patrones de la huella dactilar propuesto por el Centro de Identificación y Seguridad Digital de la Universidad de las Ciencias Informáticas.

Palabras claves: campo de orientación, grafos relacionales de atributos, indexación de huellas dactilares, máscaras dinámicas.

ABSTRACT

The fingerprint orientation field is a widely used feature for developing indexing strategies. Such feature brings stability and decreases the response times during the identification process. The use of attribute relational graphs and dynamic masks can exploit the features provided by the partitioning scheme. The penetration index and the error rate for each of the proposed strategies were determined. To verify the results obtained were used the databases provided by the

Fingerprint Verification Competition. Both search schemes reveal the facilities offered by the orientation field for guiding the searching process; reducing the number of comparisons in 53.24%; proving its stability against different fingerprint image qualities. Based on the results, it was proven that the adoption of such indexing strategies will reduce the response time in the identification module proposed by the Identification and Digital Security Center at the University of Informatics Sciences.

Keywords: *attribute relational graphs, dynamic masks, fingerprints indexing, orientation field.*

INTRODUCIÓN

Los sistemas de reconocimiento de huellas dactilares se basan fundamentalmente en dos procesos: la verificación e identificación. Este último, requiere la búsqueda y comparación del patrón de entrada teniendo como campo de análisis la totalidad de la base de datos, cuyo tamaño actualmente se encuentra en el orden de los millones de impresiones, por lo cual la búsqueda lineal resulta impracticable cuando se requiere inmediatez en el proceso. Algunas estrategias para agilizar el proceso de búsqueda proponen dividir la base de datos en un conjunto de grupos, basados en alguna clasificación predefinida. La más ampliamente difundida de estas técnicas de clasificación se basa en los estudios de Francis Galton en 1892 y Edward Henry en 1900; este esquema reconoce como clases: arco, arco tendido, lazo izquierdo, lazo derecho y espiral. Con la perspectiva de solventar los problemas asociados al problema en cuestión, surge otra técnica definida como indexación de huellas dactilares, cuyo método emplea una clave para retornar una lista ordenada por relevancia de aquellos candidatos potenciales a coincidir con el patrón dactilar de entrada [1].

Las características globales, tales como patrón descrito por las crestas y las singularidades presentes en la huella dactilar, son obtenidas en las primeras etapas del preprocesado de su imagen. La independencia y robustez de las estrategias de indexación basadas en características globales de la huella dactilar, arrojan resultados que validan la calidad de la misma, como se puede constatar en [1] - [4].

El Centro de Identificación y Seguridad Digital (CISED), perteneciente a la Universidad de las Ciencias Informáticas (UCI), desarrolla soluciones de software empleadas en casi todas las esferas de la sociedad para el control y resguardo de sus recursos. El sistema de verificación de identidades propuesto por CISED se encuentra en constante actualización y requiere de un proceso continuo de investigación para continuar mejorando la eficiencia y efectividad de su servicio. La aplicación de las estrategias implementadas en CISED hasta hoy, por sí solas no solventan el problema de la búsqueda durante la identificación de individuos por medio de huellas dactilares.

Varias han sido las propuestas a lo largo de los años que utilizan características globales de las huellas para la indexación. En [5], se utilizan los resultados planteados en [6] para presentar un algoritmo invariante a la traslación y rotación que no requiere la detección de las singularidades, evolucionando el enfoque presentado en [7]. En [8] proponen la utilización de un vector de 156 componentes de orientación e información procedente del campo de orientación para la búsqueda. Un enfoque estadístico más sofisticado, es propuesto en [9]; los cuales utilizan los coeficientes del modelo FOMFE, como vector de características. En [10] presentan el uso de los momentos polares complejos para extraer una representación invariante a la rotación del campo de orientación de la huella; dicha representación es utilizada para generar un vector de características compacto y de longitud fija. Un estudio más reciente es [11], este utiliza redes neuronales para implementar un clasificador (de 4 clases) con altos niveles de efectividad.

La presente investigación tiene como objetivo el análisis y evaluación de los algoritmos propuestos en [5], [7] los cuales utilizan el campo de orientación de la huella dactilar para implementar una estrategia de búsqueda simplificada que empleando grafos relacionales de atributos y máscaras dinámicas a partir de la entrada permiten extraer un vector de características robusto e invariante a la traslación y rotación lo suficientemente discriminante en un tiempo adecuado para su implementación en un sistema de verificación que atienda peticiones en línea.

Lo novedoso de la investigación se ve reflejado en el empleo del algoritmo propuesto en [7] para la indexación de huellas dactilares, en lugar de la clasificación, principio fundamental para el que fue creado. Además, la utilización de un algoritmo genético de nuevo tipo y una estrategia dinámica de búsqueda en árbol, definida como TS, propuesto en [12] para efectuar la comparación inexacta entre dos grafos relacionales de atributos dados. Por otra parte, a la estrategia propuesta en [5] se le incorpora un procedimiento semiautomático que contribuye a la elaboración de las máscaras dinámicas utilizadas por dicha estrategia. Para la abstracción de las máscaras dinámicas es empleado un algoritmo de triangularización de polígonos para lograr la correcta aproximación de los vértices de la máscara en un tiempo eficiente, requiriendo un mínimo de ajuste en lugar de la creación manual de dichas máscaras. Las estrategias de indexación propuestas se implementaron empleando el lenguaje de programación C# y fueron insertadas en un framework que permitió comprobar la efectividad y el rendimiento de los mismos. Los bancos de datos utilizados para las pruebas del experimento son las publicadas en el sitio oficial de Competencia de Verificación de Huellas Dactilares (FVC - Fingerprint Verification Competition) FVC2000 y la FVC2004: DB1_B, DB2_B, DB3_B, DB4_B [13].

METODOLOGÍA COMPUTACIONAL

Utilización de grafos relacionales de atributos

El enfoque estructural propuesto por [7] versa sobre la aplicación de la estructura de grafo sobre el campo de orientación de la huella dactilar para generar una clasificación exclusiva empleada

en el proceso de identificación de los Sistemas Automáticos de Identificación de Huellas Dactilares (AFIS - Automatic Fingerprint Identification System). La comparación de grafos consiste en encontrar una correspondencia entre dos grafos dados, donde los vértices son rotulados de acuerdo al mayor ajuste encontrado. El esquema funcional de dicha propuesta se divide en cinco pasos fundamentales: 1) cálculo de la imagen direccional o campo de orientación, 2) particionado de la imagen, 3) construcción del grafo relacional, 4) comparación inexacta de grafos y 5) clasificación dactilar (ver Figura 1).

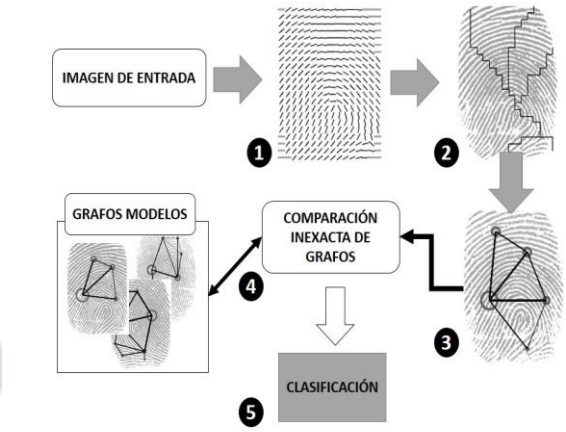

Figura 1. Pasos fundamentales de la propuesta descrita en [7]. Elaboración propia inspirada en la Figura 3 de [7].

Particionado del campo de orientación

Partiendo de los algoritmos propuestos en [14], [15] para el cálculo del campo de orientación, en la etapa de particionado se enfatizan las características topológicas y estructurales de la huella. Un algoritmo dinámico de agrupamiento propuesto en [7] es adoptado para dicho propósito de acuerdo a un criterio de ajuste óptimo, intentando minimizar la varianza entre los elementos direccionales pertenecientes a la misma región y simultáneamente conservar la regularidad de la forma.

Definición 1. Sea D, el campo de orientación de la huella, una partición R de D se define como $R = \{R_1, R_2, ..., R_n\}$ en D con R_i regiones conectadas.

Una región R_i está conectada si, para cada par de elementos (d_i,d_j) en R_i, existe un camino en R_i que comunique d_i y d_j.

La obtención de una partición óptima se encuentra subordinada a un modelo de costo propuesto en [7] donde se describe una función para cada región R_i como se muestra en la Ecuación 1:

$$C(R_i) = C_0 + w * C_{dir}(R_i) + (1 - w) * C_{sh}(R_i) \quad (1)$$

Donde $C_{dir}(R_i)$ denota la no homogeneidad entre los elementos direccionales pertenecientes a R_i; $C_{sh}(R_i)$ denota la irregularidad de la región R_i; $w \in [0,1]$ constituye un valor de peso, y C_0 es el valor umbral correspondiente al costo definido para una región a la cual pertenezca un solo elemento.

Las definiciones propuestas en [7] para $C_{dir}(R_i)$ y $C_{sh}(R_i)$ se muestran en las Ecuaciones 2 y 3:

$$C_{dir}(R_i) = p1 * Variance(R_i) * Area(R_i)^{p2} \quad (2)$$

$$C_{sh}(R_i) = \frac{Perimeter(R_i)}{\sqrt{4\pi * Area(R_i)}} \quad (3)$$

Donde *Variance*(R_i) está dada por la diferencia existente entre los elementos direccionales que pertenecen a R_i; *Area*(R_i) y *Perimeter*(R_i) de R_i representan el área y perímetro respectivos de dicha región.

Entre todas las posibles particiones de D, en [7] se define una partición R como óptima, si esta minimiza el costo total $C_{tot}(R)$ según se muestra en la Ecuación 4:

$$C_{tot}(R) = \sum_{i=1}^{n} C(R_i) \quad (4)$$

Construcción del grafo relacional de atributos y comparación inexacta

El análisis de los aspectos esquemáticos del modelo, evidencia la difícil obtención de una partición óptima de la imagen de la huella dactilar en sus unidades fundamentales, por lo que no se puede esperar un isomorfismo entre dos grafos de imágenes diferentes, aunque estas representen la misma huella dactilar. El isomorfismo denota la correspondencia biunívoca entre dos estructuras algebraicas que conserva las operaciones. Por ello es necesario aplicar técnicas de comparación inexacta de grafos, que incurren en la optimización de alguna función objetivo. En el caso de este estudio, dicha función objetivo está influenciada por el costo de la partición obtenida por la Ecuación 4. Para realizar este proceso en la fase posterior al particionado del campo de orientación, se construye un grafo relacional de atributos, donde cada región es representada como un nodo y cada par de regiones adyacentes constituye una arista del dicho grafo.

La función para medir el grado de ajuste existente entre los vértices y los nodos de ambos grafos, basada en los atributos especificados para representar cada uno de estos, así como la topología descrita, fue tomada de [12]. En la presente investigación se implementaron tres estrategias fundamentales de las propuestas por [12]; primero se desarrolló un enfoque estructural que describe una búsqueda en árbol para encontrar la mejor correspondencia posible entre dos grafos relacionales de atributos dados. Luego, una alternativa determinista del enfoque anterior y un

algoritmo genético son utilizados para comparar el rendimiento y exactitud de la propuesta analizada.

Empleo de máscaras dinámicas

El estudio realizado en [5] pretende solventar la mayoría de los problemas asociados a la estrategia descrita en el Epígrafe 2.1. En lugar del enfoque de particionado anterior, se expone una estrategia estructural de agrupamiento global que realiza un particionado "guiado" del campo de orientación y genera vectores de mayor exactitud para el espacio de búsqueda, reduciendo los grados de libertad del algoritmo dinámico propuesto en [7], otorgando así estabilidad al proceso de indexación propuesto. Las máscaras dinámicas representan una abstracción del particionado y resumen la información extraída de las particiones modelos creadas para cada clase de huella dactilar establecida. Dicha estructura conserva la invariante de traslación y rotación sin incurrir en la detección de puntos singulares. Un algoritmo genético con enfoque similar al referenciado en [5], es empleado utilizando los operadores de reproducción y mutación propuestos en [17] para la clusterización jerárquica de grafos; la definición de dicha estrategia metaheurística se extrajo de [18]. Con la implementación de dicha estrategia se propone otra clusterización de los bloques pertenecientes al campo de orientación. La idea básica propuesta en [5] consiste en:

1. Cálculo y realzado de la imagen direccional de la huella dactilar (campo de orientación).

2. Empleo del conjunto de máscaras dinámicas previamente calculadas para lograr el mejor ajuste posible. Cada una es aplicada de manera independiente al campo de orientación de entrada. Luego, para determinar el mejor ajuste se emplea una función de costo derivada de la propuesta en [7].

3. El vector de costo resultante constituye la base para la clasificación final.

Definición de las máscaras dinámicas

La aplicación de una máscara en particular a D puede ser vista como una segmentación "guiada" de la misma, donde la cantidad de regiones y una aproximación de la forma de estas se encuentran fijados a priori. Cada máscara está compuesta por un conjunto de vértices que definen los bordes de las regiones que delimitan la segmentación. La posición de algunos vértices puede ser modificada durante el proceso de ajuste con el objetivo de lograr la mejor correspondencia con las singularidades de la huella dactilar de entrada, las cuales tienden a ocupar diferentes posiciones incluso para huellas de la misma clasificación. A la definición dada en [5] para las máscaras dinámicas, se le agregaron algunos conceptos para lograr un procedimiento que permitiera la construcción semiautomática de las mismas. A continuación, se ofrece la definición formal de máscaras dinámicas utilizada en la presente investigación (ver Figura 2).

Definición 2. Una máscara dinámica se define como una 7-tupla, $M = (V,P,T,R_D,A,f_{win},f_{mov})$, donde:

$V = V_F \cup V_M \cup V_D$: conjunto de vértices p, cada uno de los cuales posee una posición inicial (p_x,p_y); V_F denota los vértices fijos; V_M denota los vértices móviles, cuya posición se puede ajustar de forma independiente durante la adaptación de la máscara; V_D denota los vértices dependientes, cuya posición se encuentra anclada a la de uno de los vértices móviles. Los tres subconjuntos de vértices son disjuntos: $V_F \cap V_M = \emptyset$, $V_F \cap V_D = \emptyset$, $V_D \cap V_M = \emptyset$.

$P = \{P_1,P_2,\ldots,P_n\}$: conjunto de regiones poligonales cuyos vértices están en V; cada región $P_i = \{p_a,p_b,p_c,\ldots\}$ está delimitada por el polígono definido por los vértices $p_a,p_b,p_c,\ldots$ tomado en un orden dado, por lo que P es un subconjunto del conjunto potencia de $V : P \subset \wp(V)$.

$T = \{T_1,T_2,\ldots,T_n\}$: conjunto de triángulos agrupados de acuerdo al polígono P_i que estos describen. Cada elemento, $T_i = \{t_a,t_b,t_c,\ldots\}$, donde $t_\vartheta = (p_l,p_m,p_n) || p_l,p_m,p_n \in P_i$ posee una propiedad que lo define como válido o no, dicha propiedad es empleada para el procedimiento de ajuste de la máscara dinámica.

$R_D \subseteq V_D \times V_M$: relación que codifica la correspondencia entre los vértices dependientes y móviles.

$A \subseteq P \times P \times \Delta_\vartheta$: codifica una relación entre los pares de regiones que se asocian a los ángulos que representan los valores "ideales" de su diferencia de fase. Δ_ϑ denota el dominio de la diferencia de fase. Para cada par $[P_i,P_j]$, cuya diferencia de orientación $\vartheta_{ij} \in \Delta_\vartheta$ es significativa, la tripleta $(P_i,P_j,\vartheta_{ij}) \in A$.

$f_{win} : V_M \rightarrow \Delta_{xmax} \times \Delta_{ymax}$: función que asocia cada vértice móvil a una ventana de permisividad en el movimiento de los vértices durante la adaptación de las máscaras. Δ_{xmax} y Δ_{ymax} representan el dominio del máximo desplazamiento a lo largo de los ejes x y y, respectivamente.

fmov : RD × Δx × Δy → Δx × Δy: función que indica, para cada par en RD, el movimiento del vértice dependiente sobre la base del vértice móvil correspondiente. Δx × Δy representan el dominio de desplazamiento a lo largo de los ejes x y y, respectivamente.

En la Figura 2 se muestran algunas de las máscaras dinámicas obtenidas después de corregir los resultados del procedimiento automático propuesto para su construcción. Los puntos móviles son representados con color negro, los cuales tienen asociado su respectiva ventana de movilidad; en gris, se definen aquellos puntos dependientes y con una saeta se delinea su correspondiente punto móvil; los puntos blancos son aquellos cuya posición y dependencia los ubica como fijos en la máscara. Las líneas oscuras describen los polígonos que delimitan cada región y las claras los triángulos válidos que la conforman. Dichas máscaras se corresponden con huellas de la FVC2000 DB4_B.

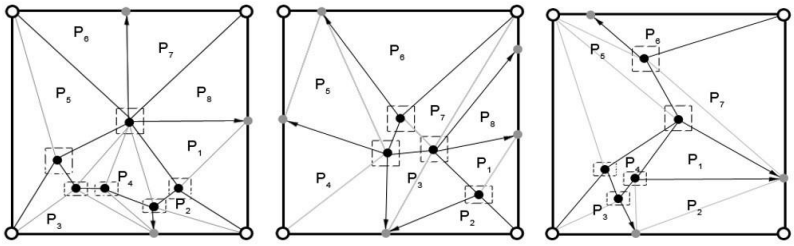

Figura 2. Máscaras dinámicas obtenidas del procedimiento automático propuesto.

La construcción de un conjunto de máscaras prototipos es realizada durante la etapa de configuración del algoritmo de indexación al banco de datos especificado, sin incurrir en un costo adicional para el AFIS durante cada consulta realizada.

Algoritmo para la estimación automática de máscaras dinámicas

En los estudios anteriores no se incurre en el tema de la generación automática de las máscaras dinámicas propuestas por [5], ni se tiene conocimiento de cualquier otro procedimiento desarrollado para dicho propósito. Por tal motivo, se desarrolla un algoritmo dinámico que construye de manera incremental las máscaras dinámicas. La abstracción de las máscaras y la elección de una estructura consistente que almacene la información para ser aplicada en las fases posteriores del procedimiento de indexación propuesto en [5], es una tarea ardua que no está ajena a errores. Ello podría ser una de las causas por la cual no se ha encontrado ningún otro estudio realizado sobre el tema, a pesar de la notada rapidez y efectividad que denota el método en contraste con la poca información de la huella dactilar que emplea. La base fundamental de este procedimiento es la construcción de una matriz de particiones de la imagen direccional de la huella, a partir del resultado obtenido por el algoritmo de particionado empleado.

Definición 3. Sea S una matriz de dimensiones igual a D, cuyos componentes representan números enteros, cada uno de los cuales establece una correspondencia para cada elemento direccional ubicado en la posición $[i,j]$ de D con el entero dispuesto en la posición $[i,j]$ de S. Cada uno de estos determina a qué región pertenece dicho bloque de orientación de la matriz D a través de S. El dominio de la matriz S se corresponde con la cantidad de regiones determinadas por el algoritmo de particionado escogido.

El procedimiento empleado para la generación automática de máscaras dinámicas se realiza siguiendo los pasos:

1. Aplicar un procedimiento lineal para adicionar los vértices fijos de la máscara a V_F, puesto que estos constituyen las aristas de la imagen.

2. Definir los vértices dependientes dispuestos en V_D: se realiza un recorrido por los bordes de la

matriz S que describe la partición de D obtenida para detectar cambios de región en los bordes de la imagen. Cada cambio de región detectado incurre en la aparición de un vértice dependiente de la máscara.

3. Detectar los puntos móviles agregados a V_M que de manera general se corresponden con las singularidades de la huella dactilar: se emplea una estrategia similar a la expuesta en [19], sin utilizar el análisis multiresolución sobre la matriz S para detectar los puntos móviles de la máscara.

4. Definir a priori las ventanas de movilidad para los elementos en V_M: se utiliza un procedimiento arbitrario que asigna a cada ventana perteneciente a un punto móvil valores entre 4, 2 y 6 a cada una de sus dimensiones, lo cual genera indistintamente ventanas cuadradas y rectangulares. Según el análisis realizado de la propuesta descrita en [5] se puede seguir un orden fijado a priori para obtener resultados similares a los descritos en ese estudio.

5. Utilizar el Algoritmo H.1 propuesto en [20] para la construcción de las regiones poligonales que describen la partición definida por la máscara dinámica a partir del conjunto V de vértices. Luego, un algoritmo de triangularización de polígonos propuesto en [21] es empleado con el objetivo de determinar un subconjunto del espacio de triangulaciones en las que puede ser dividido el polígono detectado. La solución propuesta es óptima y permite lidiar con la inclusión en la región de algún vértice que no le corresponda, debido a que el algoritmo dinámico empleado para asignar los vértices a regiones no es lo suficientemente robusto. La triangularización resultante del algoritmo propuesto en [21] requiere el mínimo de modificación humana para lograr una correcta descripción de las máscaras dinámicas propuestas por [5].

6. Definir R_D: se determina la relación existente entre los vértices del conjunto V basado en la matriz de partición S.

7. Establecer el conjunto A: se definen las relaciones de fase entre regiones de la máscara utilizando una estrategia heurística definida a priori por los autores en correspondencia con la diferencia de fase obtenida entre el promedio de las orientaciones de cada región. El dominio previamente se encuentra fijado en el intervalo máximo admisible $[0,\pi]$. Con la realización de un análisis de las diferencias de fase entre regiones se establece un rango r (ver Ecuación 5) el cual es tomado como dominio admisible para la máscara construida.

$$r = \Delta Phase_{min} + \frac{1}{2}(\Delta Phase_{max} - \Delta Phase_{min}) \quad (5)$$

Las máscaras dinámicas definidas son representadas en la aplicación resultante de dicho estudio como un XML que las describe formalmente. Dicho archivo en el formato especificado es cargado posteriormente por el procedimiento encargado de "guiar" el particionado durante el proceso de indexado propuesto. Para la codificación de las máscaras se emplea la clase nativa del .NET Framework 4.5, XmlSerializer.

RESULTADOS Y DISCUSIÓN

En orden de comparar el rendimiento de las dos propuestas de indexación descritas en el Epígrafe 2.1 y el Epígrafe 2.2, tomando como referencia la igual denominación propuesta en [5], se escogió una de las estrategias de recuperación definidas en [6]. Se simularon pruebas de acuerdo a la metodología BC, sobre los bancos de datos de la FVC2000 DB_ B, DB2_B, DB3_B y DB4_B. La estrategia escogida para la experimentación plantea un escenario incremental que permite ajustar el espacio de búsqueda sobre el banco de datos a partir de la distancia euclidiana entre el vector de entrada y el resto de los vectores almacenados en la base de datos.

Las bases de datos con las que se cuenta para la simulación contienen 80 impresiones de huellas dactilares cada una, las cuales han sido obtenidas con diferentes dispositivos de captura y la última de estas generada sintéticamente. La estructura de las imágenes presentes en los respectivos bancos de datos comprende 10 huellas, contando con 8 impresiones diferentes de cada una. En la Tabla 1 se ofrecen algunas consideraciones importantes de los conjuntos de pruebas empleados. Los parámetros empleados para los algoritmos propuestos son los siguientes: para la Ecuación 1 se utilizaron los valores C_0 = 1,5 y w = 0,2 y en la Ecuación 1, se emplean los valores p_1 = 0,001 y p_2 = 1,3. El rango de desplazamiento global de las máscaras utilizadas incurre en un intervalo de [−6,6] para ambos ejes x e y; el rango de rotación global alcanzado por las máscaras es de $[-\pi,\pi]$, utilizando pasos discretos de $\frac{\pi}{8}$.

Tabla 1. Distribución de huellas por clases de los bancos de datos utilizados.

Base de datos	A	T	L	R	W	D
DB1_B	-	-	2	4	4	-
DB2_B	-	-	2	4	4	-
DB3_B	2	1	4	2	-	1
DB4_B	1	-	3	3	1	2

En la Tabla 1 se utilizan las siguientes iniciales para representar las diferentes topologías de huellas propuestas para este estudio, A (arco), T (arco tendido), L (lazo izquierdo), R (lazo derecho), W (espiral) y D (doble lazo).

La descripción de los parámetros evolutivos empleados para cada uno de los experimentos es especificada en la Tabla 2. La Figura 3 muestra el comportamiento de las diferentes configuraciones genéticas utilizadas de acuerdo a su función de ajuste, así como el óptimo alcanzado por el algoritmo ávido propuesto en [7], el cual se tomó como referencia para el siguiente análisis.

Tabla 2. Configuraciones genéticas utilizadas.

Serie	Generaciones	Población	Cruzamiento	Mutación	Particiones
1	1000	100	1.0	0.13	10
2	1000	50	0.5	0.13	10
3	1000	100	0.5	0.13	10
4	1000	30	0.5	0.13	10

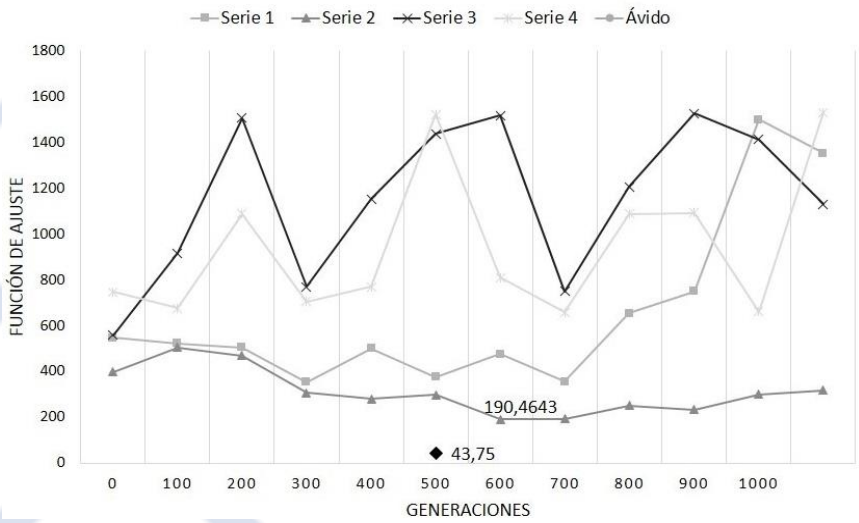

Figura 3. Comportamiento obtenido por las series expuestas en la Tabla 2. Los puntos etiquetados se corresponden con el mejor resultado ofrecido por las diferentes configuraciones genéticas y el óptimo logrado por el algoritmo ávido propuesto en [7].

El estudio realizado en [5] pretende solventar la mayoría de los problemas asociados a la estrategia descrita en el Epígrafe 2.1. En lugar del enfoque de particionado anterior, se expone una estrategia estructural de agrupamiento global que realiza un particionado "guiado" del campo de orientación y genera vectores de mayor exactitud para el espacio de búsqueda, reduciendo los grados de libertad del algoritmo dinámico propuesto en [7], otorgando así estabilidad al proceso de indexación propuesto. Las máscaras dinámicas representan una abstracción del particionado y resumen la información extraída de las particiones modelos creadas para cada clase de huella dactilar establecida. Dicha estructura conserva la invariante de traslación y rotación sin incurrir en la detección de puntos singulares. Un algoritmo genético con enfoque similar al referenciado en [5], es empleado utilizando los operadores de reproducción y mutación propuestos en [17] para la clusterización jerárquica de grafos; la definición de dicha estrategia metaheurística se extrajo de [18]. Con la implementación de dicha estrategia se propone otra clusterización de los bloques pertenecientes al campo de orientación. La idea básica propuesta en [5] consiste en:

1. Cálculo y realzado de la imagen direccional de la huella dactilar (campo de orientación).
2. Empleo del conjunto de máscaras dinámicas previamente calculadas para lograr el mejor

ajuste posible. Cada una es aplicada de manera independiente al campo de orientación de entrada. Luego, para determinar el mejor ajuste se emplea una función de costo derivada de la propuesta en [7].

3. El vector de costo resultante constituye la base para la clasificación final.

Definición de las máscaras dinámicas

A continuación, se muestran algunos análisis realizados producto de la ejecución de los diferentes algoritmos propuestos. En lo adelante, se denota como TS a la primera propuesta analizada en el Epígrafe 2.1; se define como CLMM a la propuesta descrita en el Epígrafe 2.2 utilizando como conjunto de máscaras prototipos, aquellas definidas en [5] como ideales para las cinco clasificaciones de huellas dactilares propuestas en dicho estudio; por último, se define como CLMM* a la aplicación del procedimiento anterior, empleando para ello el conjunto de máscaras prototipos generadas por el algoritmo descrito en el Epígrafe 2.2.2 para la construcción de las máscaras dinámicas. Según los resultados expuestos, los mejores ajustes son obtenidos por la estrategia dinámica especificada en [7], por lo cual se emplea esta, en lugar del algoritmo genético sugerido en [5].

El proceso de indexación de huellas dactilares se rige por una serie de parámetros que cuantifican la eficiencia y calidad de la indexación aportada por cada uno de los métodos evaluados. A continuación, se detallan los rasgos que permiten apreciar el tiempo de respuesta y la estabilidad de las estrategias de búsqueda propuestas en este estudio.

Índice de penetración. Dicha medida es comúnmente utilizada para cuantificar el rendimiento de los sistemas de clasificación y la misma influye directamente en el tiempo de respuesta del proceso. Se define como se muestra en la Ecuación 6:

$$PR = \frac{100n}{N}\% \tag{6}$$

sean n: número de huellas revisadas, y N: total de huellas presentes en el banco de datos. Este indicador se mide en una escala del [0,100]%.

Tiempo de respuesta. Se define como se muestra en la Ecuación 7:

$$\Delta t = t_e - t_0 \tag{7}$$

sean t_e: tiempo en el que la acción culmina, y t_0: instante inicial de la operación. Usualmente es medido en milisegundos (ms) o segundos (s).

Tasa de error. Representa la razón entre el número de huellas mal clasificadas y el número total de muestras empleadas para la prueba. Se define como se muestra en la Ecuación 8:

$$ER = \frac{100c_n}{N}\,\% \tag{8}$$

sean C_n: la cantidad de huellas mal clasificadas, y N: el total de impresiones del conjunto de prueba. Este indicador se mide en una escala del [0,100]%. Suele expresarse en función del índice de penetración.

Durante el análisis exploratorio para el algoritmo en cuestión se definen los siguientes indicadores de rendimiento:

- ER100 (el menor PR para un ER ≤ 1%).
- ER1000 (el menor PR para un ER ≤ 0,1%).
- Gráfica [ER × PR].
- IS (el promedio del PR para el escenario de búsqueda incremental). En tal escenario no existen errores de retorno, en el peor de los casos, la búsqueda es extendida a todo el banco de datos, y el PR promedio es reportado como un indicador de rendimiento simple.
- Tiempo de enrolamiento promedio (μ_e).
- Tiempo de indexación promedio (μ_x).

Teniendo en cuenta la distribución de huellas por clases dada en la Tabla 1, se utiliza el conjunto de máscaras dinámicas asociadas a las topologías de huellas encontradas para cada una de las bases de datos utilizadas. Los indicadores de IS, ER100 y ER1000, presentan iguales valores dado el pequeño número de huellas con las que cuentan las bases de datos empleadas en las pruebas, por tal motivo, estos se tratan como un todo en los resultados expuestos. Otros valores de interés, son el tiempo de enrolamiento e indexación promedio, definidos como μ_e y μ_x, respectivamente.

En la Tabla 3 se muestran algunas estadísticas calculadas a partir de las pruebas realizadas sobre el banco de datos de la FVC2000 perteneciente al conjunto B de la misma.

Tabla 3. Resultados recopilados sobre la FVC2000 conjunto B.

Base de datos FVC2000	**Estrategia**	*IS*	μ_x (ms)	μ_e (ms)
DB1_B	**TS**	**38.271**	686.579	**543.658**
	CLMM	65.432	**155.165**	61058.250
	CLMM*	54.320	255.903	50593.437
DB2_B	**TS**	**44.737**	1198.080	449.584
	CLMM	54.320	**182.343**	1961.548
	CLMM*	58.025	2050.548	**284.950**
DB3_B	**TS**	55.560	4636.166	**892.520**
	CLMM	**50.000**	**359.865**	130173.850

	CLMM*	61.250	465.698	101888.012
DB4_B	**TS**	85.185	351.295	**315.712**
	CLMM	**43.210**	**202.550**	47203.875
	CLMM*	**43.210**	282.297	66022.437

Según los resultados obtenidos durante las pruebas, el algoritmo TS ofrece mejor tasa de penetración (PR) que los restantes en dos de las cuatro pruebas realizadas, así como un tiempo de enrolamiento mínimo en el total de las pruebas ejecutadas. Por otro lado, el tiempo de indexación mostrado por el algoritmo CLMM constituye el mejor resultado de las pruebas, ello es debido a que este algoritmo emplea un conjunto de máscaras más pequeño que la propuesta CLMM*; sin embargo, cuenta con los peores tiempos de enrolamiento para la mayoría de las pruebas ejecutadas (3/4).

En la Figura 4 se describe el comportamiento de las tres propuestas antes mencionadas para las cuatro bases de datos analizadas. Las dos implementaciones probadas CLMM y CLMM*, manifiestan tasas de penetración ligeramente similares, siendo la primera de estas, la que en mayor cantidad de ocasiones coloca los mejores resultados (3/4), llegando a superar las tasas de penetración resultantes de la estrategia TS en dos de las cuatro ocasiones. Además, se constata el correcto funcionamiento de la estrategia CLMM* para la mayoría de las bases de datos de prueba (3/4), en las cuales esta ofrece un mejor comportamiento global con respecto a la propuesta CLMM.

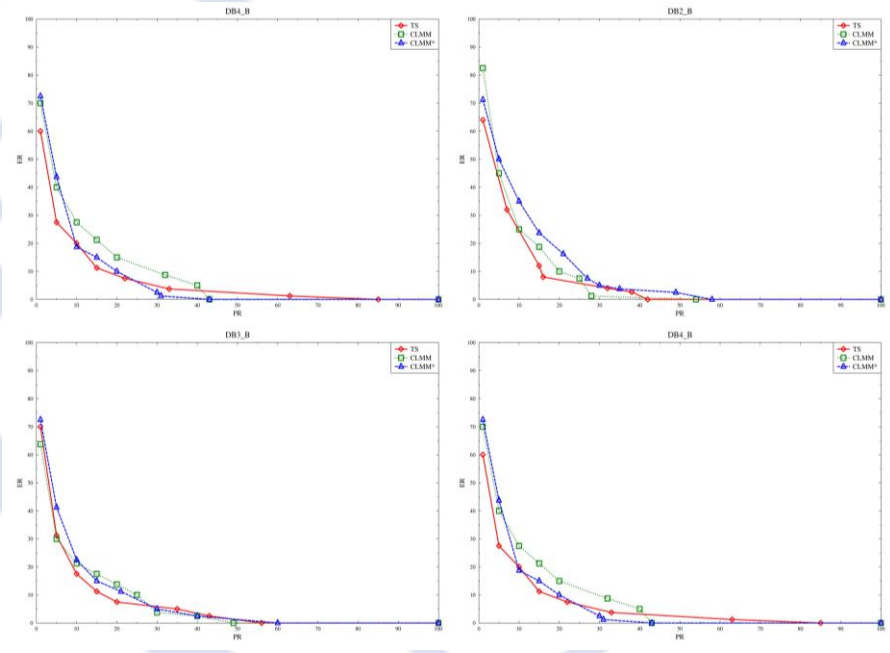

Figura 4. Resultados de la ejecución de los algoritmos TS, CLMM y CLMM* sobre el banco de datos de la FVC2000 DB1_B, DB2_B, DB3_B y DB4_B de izquierda a derecha y de arriba hacia abajo respectivamente.

CONCLUSIONES

De los resultados mostrados, su análisis y discusión, se pueden obtener las siguientes conclusiones sobre las estrategias de indexación propuestas. La utilización de características globales para la indexación de huellas dactilares garantiza reducir en al menos un 53.24 % el número de comparaciones realizadas durante el proceso de búsqueda de un AFIS. Los tiempos de ejecución del procedimiento de indexación utilizando máscaras dinámicas se mantienen aceptables, garantizando rapidez durante la búsqueda del AFIS. La estrategia de búsqueda en árbol para la comparación inexacta de grafos ofrece un balance aceptable entre rapidez y efectividad durante el proceso de indexación de huellas dactilares. Su comportamiento posee valores de similitud comparables con aquellos obtenidos por otras estrategias metaheurísticas utilizadas con propósitos similares. El procedimiento semiautomático para la generación de máscaras dinámicas, aporta fluidez al proceso de indexación y requiere poca intervención humana, garantizando tasas de error similares a aquellas aportadas por la generación manual de las mismas. La utilización de un mayor número de máscaras dinámicas no siempre garantiza la obtención de mejores resultados en el proceso de indexación abordado. El empleo de estrategias que utilicen el campo de orientación de la huella dactilar como única característica de entrada, no permite alcanzar tasas de penetración lo suficientemente adecuadas para garantizar la rapidez en el proceso de identificación de un AFIS, de acuerdo a los resultados actuales alcanzados a nivel internacional en dicha área.

REFERENCIAS

[1] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, Handbook of fingerprint recognition. Springer Science & Business Media, 2009.

[2]. M. H. Bhuyan, S. Saharia, and D. K. Bhattacharyya, "An effective method for fingerprint classification," arXiv preprint arXiv:1211.4658, 2012.

[3] S. U. Maheswari and E. Chandra, "A review study on fingerprint classification algorithm used for fingerprint identification and recognition," IJCST, vol. 3, no. 1, pp. 739–744, 2012.

[4] D. Parekh and R. Vig, "Review of Parameters of Fingerprint Classification Methods Based on Algorithmic Flow," presented at the International Conference on Advances in Computing and Information Technology, 2011, pp. 28–39.

[5] R. Cappelli, A. Lumini, D. Maio, and D. Maltoni, "Fingerprint classification by directional image partitioning," IEEE Transactions on pattern analysis and machine intelligence, vol. 21, no. 5, pp. 402–421, 1999.

[6] A. Lumini, D. Maio, and D. Maltoni, "Continuous versus exclusive classification for fingerprint retrieval," Pattern Recognition Letters, vol. 18, no. 10, pp. 1027–1034, 1997.

[7] D. Maio and D. Maltoni, "A structural approach to fingerprint classification," presented at the Proceedings of 13th International Conference on Pattern Recognition, 1996, vol. 3, pp. 578–585.

[8] M. Liu, X. Jiang, and A. C. Kot, "Efficient fingerprint search based on database clustering," Pattern Recognition, vol. 40, no. 6, pp. 1793–1803, 2007.

[9] Y. Wang, J. Hu, and D. Phillips, "A fingerprint orientation model based on 2D Fourier expansion (FOMFE) and its application to singular-point detection and fingerprint indexing," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 4, pp. 573–585, 2007.

[10] M. Liu and P.-T. Yap, "Invariant representation of orientation fields for fingerprint indexing," Pattern Recognition, vol. 45, no. 7, pp. 2532–2542, 2012.

[11] Ş. Parlakyıldız and F. Hardalaç, "A New and Effective Method in Fingerprint Classification," Life Science Journal, vol. 10, no. 12, pp. 584–588, 2013.

[12]. R. M. Cesar Jr, E. Bengoetxea, I. Bloch, and P. Larrañaga, "Inexact graph matching for model-based recognition: Evaluation and comparison of optimization algorithms," Pattern Recognition, vol. 38, no. 11, pp. 2099–2113, 2005.

[13] "FVC-onGoing." [Online]. Available: https://biolab.csr.unibo.it/FVCOnGoing/UI/Form/Home.aspx . [Accessed: 08-Ene-2020].

[14] N. K. Ratha, S. Chen, and A. K. Jain, "Adaptive flow orientation-based feature extraction in fingerprint images," Pattern Recognition, vol. 28, no. 11, pp. 1657–1672, 1995.

[15] B. G. Sherlock, D. M. Monro, and K. Millard, "Fingerprint enhancement by directional Fourier filtering," IEE Proceedings-Vision, Image and Signal Processing, vol. 141, no. 2, pp. 87–94, 1994.

[16] S. Rizzi, "Genetic operators for hierarchical graph clustering," Pattern Recognition Letters, vol. 19, no. 14, pp. 1293–1300, 1998.

[17] D. Maio, D. Maltoni, and S. Rizzi, "Topological clustering of maps using a genetic algorithm," Pattern Recognition Letters, vol. 16, no. 1, pp. 89–96, 1995.

[18] C.-Y. Huang, L. Liu, and D. D. Hung, "Fingerprint analysis and singular point detection," Pattern Recognition Letters, vol. 28, no. 15, pp. 1937–1945, 2007.

[19] A. Hernández and E. Martín, "Componente de indexación de huellas dactilares basada en características globales," Universidad de las Ciencias Informáticas, 2014.[1]

[20] M. Van Kreveld, O. Schwarzkopf, M. de Berg, and M. Overmars, Computational geometry algorithms and applications. Springer, 2000.


LaSalle
Universidad