REQUERIMIENTOS SOFTWARE PARA EL ANÁLISIS ESTILOMÉTRICO DE OBRAS LÍRICAS MEDIANTE LOS ALGORITMOS GENÉTICOS

SOFTWARE REQUERIMENTS FOR STYLOMETRIC ANALYSIS OF LYRIC WORKS THROUGH GENETIC ALGORITHMS

Andres Felipe Chauta Velandia[1]

Resumen

Este trabajo tiene como objetivo determinar cuáles son los requerimientos funcionales y no funcionales de software de un paquete programado en *Python* para el análisis estilométrico de obras líricas mediante su estudio prosódico-métrico y los algoritmos genéticos que cumplen, en mayor medida los criterios establecidos en el estándar IEEE Std. 830-1998. Concretamente, este trabajo se centrará en los requisitos de software pertinentes en el modelado de los *corpora* para la implementación del algoritmo genético en el análisis textual. Dicho lo anterior, la metodología del estudio tendrá un enfoque mixto y diseño correlacional que tendrá dos fases una cualitativa y una cuantitativa. Adicionalmente, cabe resaltar que la población de estudio serán tres propuestas de requerimientos de software desarrollados mediante el método Scrum. También es pertinente resaltar que la tasa de cumplimiento de los criterios de validación establecidos en el estándar IEEE Std. 830-1998 de cada propuesta se medirá con el coeficiente de correlación de Pearson. En el estudio, se determinó que los datos de la propuesta n° 3 tienen un coeficiente de correlación con los datos de la propuesta hipotética igual a 1, los de la propuesta n° 2 tuvieron un coeficiente de correlación de 0,964 y los de la propuesta n°1 tuvieron un coeficiente de correlación de 0,784. Así, es posible concluir que de la propuesta n° 3 cumple, en mayor medida, los criterios de validación del estándar IEEE Std. 830-1998, comparado con las otras dos propuestas.

Palabras clave: Análisis métrico, *corpus*, estilometría, requerimientos de software funcionales, requerimientos de software no funcionales.

Abstract

This work has as objective to determine which are the functional and non-functional software requirements for a package, programmed in Python for the stylometric analysis of lyric works through prosodic-metric study and genetic algorithms, that satisfy criteria of validation of the standard IEEE Std. 830-1998 to a greater extent. In fact, this research focuses on the software requirements that are pertinent in modelling of *corpora* for the implementation of genetic algorithms in textual analysis. Therefore, the methodology of the research will have a mixed approach and a correlational design that will have a qualitative phase and a quantitative one. Additionally, it is to point out that the population of the study is three proposals of software requirements that were developed by the method Scrum. Also, it is to say that the rate of fulfilment of the criteria of validation that were presented in the standard IEEE Std. 830-1998 of each proposal will be measured with the correlation coefficient of Pearson. In this study, it was determined that the data of the proposal n° 3 have a correlation coefficient equal to 1, the data of

Recibido: 10 de marzo de 2025 /Evaluación: 15 de abril de 2025 / Aprobado: 10 de mayo de 2025

[1] Magister en filosofía. Universidad Nacional de Colombia. Docente investigador en la Universidad ECCI- Bogotá. Email: achautav@ecci.edu.co. ORCID: https://orcid.org/0000-0003-0482-1407

the proposal n° 3 have a correlation coefficient equal to 0,964 and the data of the proposal n° 1 have a correlation coefficient equal to 0,784. So, it is possible to conclude that the proposal n° 3 satisfy criteria of validation of the standard IEEE Std. 830-1998 to a greater extent, in comparison with the other two proposals.

Keywords: metric analysis, *Corpus*, stylometry, functional software requirement, non-functional software requirements.

Introducción

La implementación de herramientas informáticas en la crítica textual y la ecdótica se remonta a la década de 1940. En efecto, como señala Morrás (2003), en esta época se llevaron a cabo los primeros análisis informáticos de concordancias durante la edición de la obra de Santo Tomás de Aquino realizada por Roberto Busa.

No obstante, el uso filológico y ecdótico de software ha sido objeto de debate. Ciertamente, como reconoce Bernabé (2010), históricamente los académicos han discutido principalmente en torno a la capacidad de los softwares para llevar a cabo análisis filogenéticos de la tradición manuscrita de las obras, la validez filológica de la automatización de los procesos de evaluación de materiales en la *recensio* mediante procesos estemáticos o no estemáticos[2] y la utilidad de las herramientas digitales para la configuración ecdótica de las ediciones críticas[3].

Además, es necesario resaltar que, a pesar del desarrollo del trabajo académico en torno a la implementación de herramientas digitales que ha tenido lugar en las últimas décadas, la exploración al respecto se ha limitado a la *recensio* y no ha trascendido a otras fases de la crítica textual como la *constitutio textus* o la *emendatio*. De esta manera, estos aspectos de la edición de obras se han mantenido, hasta cierto punto, ajena a su automatización, pues los filólogos han priorizado, junto con autores como el mismo Bernabé (2010), en estas fases métodos no sistemáticos para la elección de variantes y la postulación de correcciones textuales en las ediciones sobre los métodos estadísticos.

La resiliencia de los críticos textuales frente a la adopción de métodos estadísticos en la *constitutio textus* y la *emendatio* se pueden fundamentar en los postulados que se encuentran en la tradición teórica de la crítica textual misma y las condiciones de posibilidad respectivamente. En el caso de la *constitutio textus*, se prioriza el uso de métodos no estadísticos, en tanto que el criterio estadístico que se ha contemplado en gran parte de los análisis computacionales es la frecuencia de la aparición de ciertas variantes textuales en la transmisión manuscrita y este no garantiza la validez de la elección de variantes. Mientras tanto, la *emendatio* no puede tener un fundamento estadístico, entendido exclusivamente como análisis de frecuencia, pues supone la adopción de una variante textual que no se encuentra en la transmisión manuscrita de un texto.

Con esto dicho, es posible contraargumentar estas posturas: aunque estos argumentos son válidos, no son suficientes para rechazar o relegar a un segundo plano los métodos estadísticos ni en la *constitutio textus* ni en la *emendatio*. La razón de esto es que, particularmente dentro de la

[2] El análisis filogenético de manuscritos ha tenido gran acogida en los últimos años. En este sentido, ha tenido un fuerte desarrollo académico. Al respecto, véase el trabajo de Geer y Racine (2013), Hyytiänen (2023), Finney (2018), Hyytiänen (2021), Lin (2016), McCollum (2019) y Wasserman y Gurry (2017).

[3] Resultado del interés en esta cuestión es el desarrollo del proyecto DHARMA, el cual ha desarrollado un modelado estandarizado para la anotación de ediciones críticas digitales de diversos tipos, tanto de ediciones críticas como de transcripciones diplomáticas y la anotación prosódica de *corpora*. Véase Balogh y y Janiak (2023), Griffiths y Janiak (2023), Balogh y Griffiths (2020) y Balogh y Griffiths (2020a).

bioinformática, existen técnicas estadísticas, *e.g.*, el análisis mediante algoritmos genéticos[4], que no se reducen a análisis de frecuencia y permiten prever error que este tipo de estudio implica, mientras que también posibilitan la estimación de datos que no se encuentran inicialmente en la población investigada.

Concretamente, la implementación de algoritmos genéticos en el análisis textual ha demostrado ser válida para la parametrización estilométrica de obras líricas griegas, en tanto permite reconocer cuantitativamente las diferencias existentes entre la estructura métrica de textos pertenecientes a diferentes géneros. Esto tiene repercusiones en la crítica textual aplicada a este tipo de textos: aunque no se ha explorado totalmente, es posible sostener

que mediante la técnica de los algoritmos genéticos es posible, al menos en el caso de los textos líricos griegos, definir criterios estadísticos para la elección de variantes textuales, en tanto esta es válida para la parametrización métrica de las obras.

En este contexto, se plantea el presente trabajo. Este tiene como objetivo determinar cuáles son los requerimientos funcionales y no funcionales de software[5] de un paquete programado en *Python* para el análisis estilométrico de obras líricas mediante su estudio prosódico-métrico y los algoritmos genéticos que cumplen, en mayor medida los criterios establecidos en el estándar IEEE Std. 830-1998. Además, cabe resaltar que, en el presente trabajo solo se plantearán los requisitos de software pertinentes en el modelado de los *corpora* para la implementación del algoritmo genético en el análisis textual.

Metodología

El presente trabajo se plantea con base en un diseño de investigación correlacional.[6] Ciertamente, el tratamiento de los datos se centrará fundamentalmente en la determinación del grado de asociación entre las características que son propias de los requerimientos de software, según el estándar IEEE Std. 830-1998, y las características propias de cada una de las propuestas de requerimientos de software para un paquete programado en *Python* para el análisis estilométrico de obras líricas mediante su estudio prosódico-métrico y los algoritmos genéticos que constituyen la población de estudio de este trabajo.

Con esto en mente, cabe resaltar que el enfoque de investigación mixto. La razón de ello es que los datos recolectados tendrán en primera instancia un carácter cualitativo: la validación de los requerimientos de software se llevará a cabo con una lista de chequeo[7] como instrumento de recolección de datos. Sin embargo, la evaluación de los mismos se llevará a cabo de manera cuantitativa, ya que la asociación de los requerimientos de software se determinará mediante el cálculo del coeficiente de correlación de Pearson entre los datos, mediante una matriz de correlaciones.

Además, cabe resaltar que la lista de chequeo que se implementará como instrumento de recolección de datos en la fase cualitativa de esta investigación se diseñó con base en los criterios para la validación de requerimientos de software establecidos por el estándar IEEE Std. 830-1998, a saber: consistencia, verificabilidad, precisión, univocidad, modificabilidad, rastreabilidad y orden por necesidad. Así pues, cada uno de estos elementos serán las categorías de análisis en el presente estudio.

[4] Véase Koch (2014).
[5] Respecto de estos términos, véase Sethi (2023), Cross (2021), el estándar IEEE Std. 610.12-1990, Bjørner (2006) y Aurum y Wohlin (2005).
[6] Acerca del diseño de investigación cuasi-experimental, véase Monje Álvarez (2011).
[7] Sobre la implementación de listas de chequeo en el desarrollo de software, véase Chemuturi (2018)

Ahora bien, para el tratamiento cuantitativo de los datos se llevará a cabo una codificación en números reales, la cual se puede observar en la tabla 1. Dicho esto, se asignará un código numérico al cumplimiento de cada uno de los criterios de validación contemplados en la lista de chequeo y se tendrá como referencia de asociación una propuesta hipotética que cumple con todos los criterios de validación de requerimientos de software establecidos por el estándar IEEE Std. 830-1998.

A la luz de lo anterior, es pertinente resaltar que los coeficientes de correlación de Pearson que se calcularán en el presente trabajo demostrarán el grado de asociación entre el cumplimiento de los criterios de validación de la propuesta hipotética que los cumple todos y cada una de las propuestas que conforman la población de este estudio. En este sentido, si el coeficiente de correlación de alguna de las propuestas y el caso hipotético es igual a "1" o a "-1", se podrá concluir que tal propuesta cumple con todos los criterios de validación. Mientras tanto, un coeficiente de correlación distinto a estos valores supondrá que una propuesta no se puede validar, en tanto no cumple con todos los criterios del estándar IEEE Std. 830-1998.

Tabla 1

Matriz de codificación

Evento	Código
Cumple con el requerimiento de precisión	1
Cumple con el requerimiento de univocidad	2
Cumple con el requerimiento de completitud	3
Cumple con el requerimiento de consistencia	4
Cumple con el requerimiento de verificabilidad	5
Cumple con el requerimiento de modificabilidad	6
Cumple con el requerimiento de rastreabilidad	7
Cumple con el requerimiento de orden por necesidad	8
No cumple con el requerimiento.	0

Respecto de la población de estudio, esta se conforma por tres propuestas de requerimientos de software que se desarrollaron, mediante el método Scrum.[8] Estos se presentan en pseudocódigos que están codificados en lógica proposicional de segundo grado con extensiones multimodales, temporales y de ordenación de variables que permiten determinar su necesidad en cada momento de la implementación del algoritmo y definir una indexación de los elementos que están presentes en los *corpora*. Con este fin, se siguió el lenguaje proposicional presentado por Roggenbach et al. (2022). Así pues, las propuestas pueden observarse en las figuras 1, 2 y 3.

Finalmente, el método de validación de las propuestas que se implementó fue la deducción natural. Esta, tal como señalan Roggenbach et al. (2022), es un método formal[9] que permite determinar la validez de una formula proposicional dentro de un modelo lógico y puede ser aplicado para la validación de requerimientos de un software, en tanto que una *formula* es válida,

[8] Acerca del método Scrum aplicado al desarrollo de software, véase Dooley (2017).
[9] Acerca de los métodos formales en la ingeniería de software, véase Shaoying et al. (2009).

si esta derivable de un conjunto de axiomas que definen un sistema lógico, mediante las reglas formales de la deducción natural.[10] Concretamente, en el presente trabajo los axiomas expresarán los requerimientos no funcionales del software, tanto sus constricciones como los parámetros de calidad, y la fórmula que debe ser derivable se plantea como los requerimientos funcionales que determinan los objetivos de cada uno de los algoritmos y funciones del software.

Figura 1

Propuesta de requerimientos n° 1

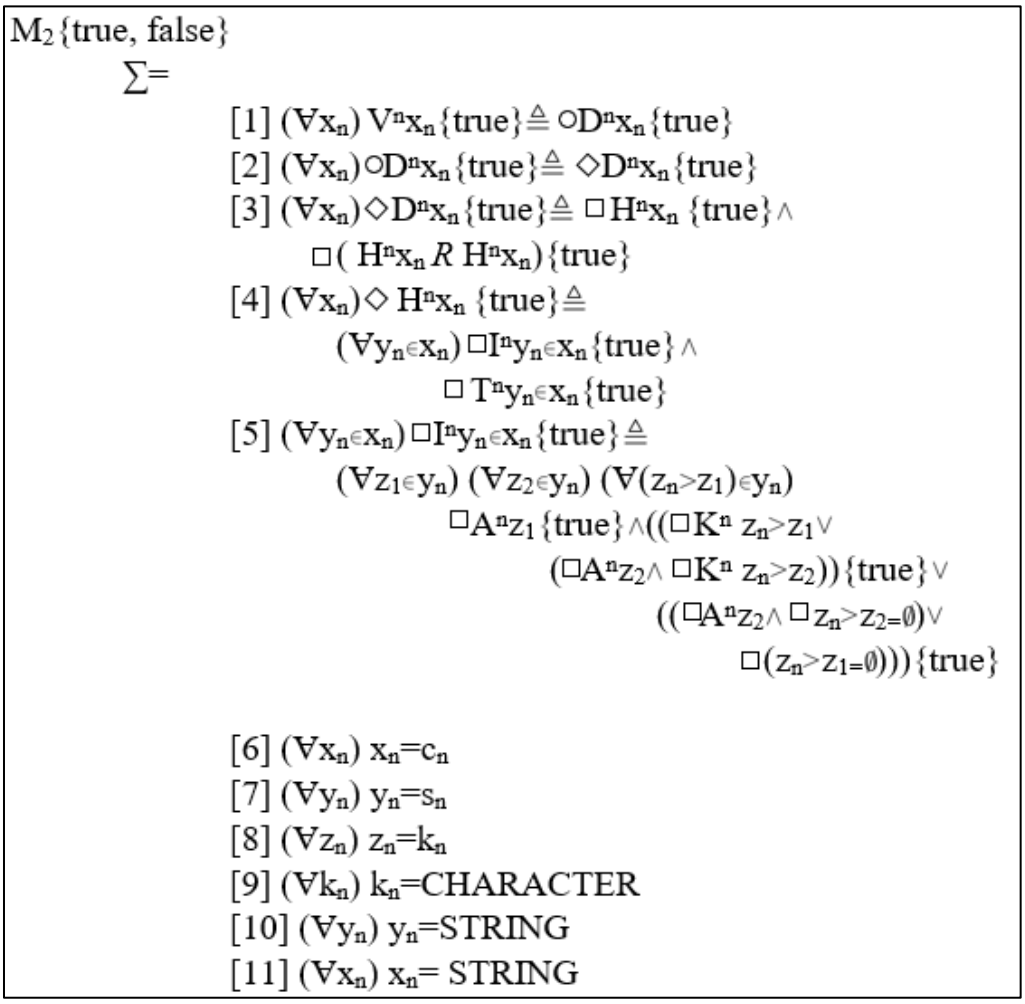
$$
\begin{aligned}
&M_1\{true, false\}\\
&\quad \sum=\\
&\qquad [1]\ (\forall x_n)\, V^n x_n\{true\} \triangleq \circ D^n x_n\{true\}\\
&\qquad [2]\ (\forall x_n) \circ D^n x_n\{true\} \triangleq \Diamond D^n x_n\{true\}\\
&\qquad [3]\ (\forall x_n) \Diamond D^n x_n\{true\} \triangleq \Box H^n x_n\{true\} \wedge\\
&\qquad\quad \Box(H^n x_n\, R\, H^n x_n)\{true\}\\
&\qquad [4]\ (\forall x_n) \Diamond H^n x_n\{true\} \triangleq\\
&\qquad\quad (\forall y_n \in x_n) \Box I^n y_n \in x_n\{true\} \wedge\\
&\qquad\quad\quad \Box T^n y_n \in x_n\{true\}\\
&\qquad [5]\ (\forall y_n \in x_n) \Box I^n y_n \in x_n\{true\} \triangleq\\
&\qquad\quad (\forall z_1 \in y_n)\ (\forall z_2 \in y_n)\ (\forall (z_n > z_1) \in y_n)\\
&\qquad\quad\quad \Box A^n z_1\{true\} \wedge ((\Box K^n\, z_n > z_1 \vee\\
&\qquad\quad\quad\quad (\Box A^n z_2 \wedge \Box K^n\, z_n > z_2))\{true\} \vee\\
&\qquad\quad\quad\quad\quad ((\Box A^n z_2 \wedge \Box z_n > z_2 = \emptyset) \vee\\
&\qquad\quad\quad\quad\quad\quad \Box(z_n > z_1 = \emptyset)))\{true\}\\
&\qquad [6]\ (\forall y_n \in x_n)\ (\forall z_n \in y_n) \Box T^n y_n \in x_n\{true\} \triangleq\\
&\qquad\quad (\forall z_1 \in y_n)\ (\forall z_2 \in y_n)\ (\forall (z_n > z_1) \in y_n)\\
&\qquad\quad\quad (\Box A^n z_1\{true\} \wedge ((\Box K^n\, z_n > z_1 \vee\\
&\qquad\quad\quad\quad (\Box A^n z_2 \wedge \Box K^n\, z_n > z_2))\{true\} \vee\\
&\qquad\quad\quad\quad\quad ((\Box A^n z_2 \wedge \Box z_n > z_2 = \emptyset) \vee\\
&\qquad\quad\quad\quad\quad\quad \Box(z_n > z_1 = \emptyset)))\{true\} \wedge\\
&\qquad\quad\quad\quad\quad\quad\quad \neg \Diamond A^n z_1\{true\})\\
&\qquad [7]\ (\forall x_n)\ x_n = c_n\\
&\qquad [8]\ (\forall y_n)\ y_n = s_n\\
&\qquad [9]\ (\forall z_n)\ z_n = k_n\\
&\qquad [10]\ (\forall k_n)\ k_n = \text{CHARACTER}\\
&\qquad [11]\ (\forall y_n)\ y_n = \text{STRING}\\
&\qquad [12]\ (\forall x_n)\ x_n = \text{STRING}
\end{aligned}
$$

Nota: Creación propia.

[10] En torno a este aspecto, véase Franco y Martin (2021).

Figura 2

Propuesta de requerimientos n° 2



Nota: Creación propia.

Figura 3

Propuesta de requerimientos n° 3

$M_3\{true, false\}$

$\sum=$

[1] $(\forall x_n)\ \Box V^n x_n\{true\} \triangleq \bigcirc C^n x_n\{true\}$

[2] $(\forall x_n) \bigcirc C^n x_n\{true\} \triangleq \Box D^n x_n\{true\}$

[3] $(\forall x_n)\ \Box D^n x_n\{true\} \triangleq \Box H^n x_n\{true\}$

[4] $(\forall x_n)\ \Box H^n x_n\{true\} \triangleq$

$(\forall y_n \in x_n)\ \Box I^n y_n \in x_n\{true\} \wedge$

$\Box T^n y_n \in x_n\{true\}$

[5] $(\forall y_n \in x_n)\ \Box I^n y_n \in x_n\{true\} \triangleq$

$(\forall z_1 \in y_n)\ (\forall z_2 \in y_n)\ (\forall (z_n > z_1) \in y_n)$

$(((\Box AC^n z_1 \vee \Box AL^n z_1)\{true\} \wedge \Box K^n\ z_n > z_2) \vee$

$(((\Box AC^n z_2 \vee \Box AL^n z_2) \wedge \Box K^n\ z_n > z_2)\{true\} \vee$

$((\Box AC^n z_2 \vee \Box AL^n z_2) \wedge \Box\ z_n > z_2 = \emptyset) \vee$

$\Box(z_n > z_1 = \emptyset))\{true\}$

[6] $(\forall y_n \in x_n)\ \Box T^n y_n \in x_n\{true\} \triangleq (\Diamond SLy_n \in x_n \vee$

$\Diamond SCy_n \in x_n)\{true\}$

[7] $(\forall y_n \in x_n)\ \Diamond SLy_n \in x_n\{true\} \triangleq \neg \Diamond SCy_n \in x_n)\{true\}$

[8] $(\forall y_n \in x_n) \Diamond SCy_n \in x_n\{true\} \triangleq$

$(\forall k_n \in y_n)(\Box AC^n k_1 \wedge \Box(k_n > k_1 = \emptyset))\{true\} \vee$

$((\Box AC^n k_1 \wedge \Box(\neg \Diamond KD^n k_2 \wedge \Box(K^n k_n > k_2 = \emptyset)))\{true\} \wedge$

$\neg \Diamond DP\ k_1 k_2)\{true\}$

[9] $(\forall x_n)\ x_n\{true\} = c_n\{true\}$

[10] $(\forall y_n)\ y_n\{true\} = s_n\{true\}$

[11] $(\forall z_n)\ z_n\{true\} = k_n\{true\}$

[12] $(\forall k_n)\ k_n\{true\} = CHARACTER\{true\}$

[13] $(\forall y_n)\ y_n\{true\} = STRING\{true\}$

[14] $(\forall x_n)\ x_n\{true\} = STRING\{true\}$

[15] $(\forall k_n)\ ACk_n\{true\} \triangleq k_n = \bigoplus(\alpha, \varepsilon, \iota, \upsilon, o)\{true\}$

[16] $(\forall k_n)\ ACk_n\{true\} \triangleq k_n = \bigoplus(\bar{\alpha}, \eta, \bar{\iota}, \bar{\upsilon}, \omega)\ \{true\}$

[17] $(\forall k_n)\ ACk_n\{true\} \triangleq k_n =$

$\bigoplus((o \wedge \upsilon), (\alpha \wedge \iota), (\varepsilon \wedge \iota), (\eta \wedge \iota), (\alpha \wedge \iota), (o \wedge \iota), (\omega \wedge \iota), (\varepsilon \wedge \upsilon), (\varepsilon \wedge \omega))\{true\}$

[18] $(\forall k_n)\ Kk_n\{true\} \triangleq k_n = \bigoplus(\beta, \delta, \gamma, \theta, \kappa, \lambda, \mu, \nu, \pi, \rho, \sigma, \tau, \chi, \varphi)\{true\}$

[19] $(\forall k_n)\ KDk_n\{true\} \triangleq k_n = \bigoplus(\zeta, \psi, \xi)\ \{true\}$

Nota: Creación propia.

Análisis y discusión de resultados

Como se planteó en la metodología del trabajo, la recolección de datos llevada a cabo en el presente trabajo tuvo dos fases: una fase cualitativa y una cuantitativa. La primera se centró en determinación de cumplimiento de cada uno de los criterios de validación de software presentados en el estándar IEEE Std. 830-1998 y la segunda se enfocó en el análisis correlacional cuantitativo de los resultados obtenidos en la primera fase respecto de cada propuesta de requerimientos de software, en contraste con el caso hipotético que cumple con todos los parámetros de validación. Esta misma estructuración determinará la presentación de los resultados en este apartado del documento.

En la primera fase del estudio, se evidenció que las propuestas 1 y 2 no cumplen con los requerimientos de precisión y univocidad. Ciertamente, estas contienen la formula "$\Box\ H^n x_n\ \{true\} \wedge \Box(\ H^n x_n\ R\ H^n x_n)\ \{true\}$" y esta no representa un requerimiento necesario para deducir los requerimientos funcionales. En consecuencia, puede sostenerse que este no es preciso, a la luz de los planteamientos del estándar IEEE Std. 830-1998: en este documento, se sostiene que los requerimientos de software son correctos, si y solo si son necesarios en el software.

Adicionalmente, es posible sostener que la propuesta 1 no cumple con el requerimiento de precisión, en tanto contiene una definición, *i.e.*, un requerimiento, innecesaria. Concretamente, esta definición es la número 6: esta es una extensión de la definición número 5. La definición 6 se puede observar en la figura 4.

Figura 4

Definición 6 de la propuesta n° 1

$$[6]\ (\forall y_n \in x_n)\ (\forall z_n \in y_n)\ \Box T^n y_n \in x_n \{true\} \triangleq$$
$$(\forall z_1 \in y_n)\ (\forall z_2 \in y_n)\ (\forall (z_n > z_1) \in y_n)$$
$$(\Box A^n z_1 \{true\} \wedge ((\Box K^n\ z_n > z_1 \vee$$
$$(\Box A^n z_2 \wedge \Box K^n\ z_n > z_2))\{true\} \vee$$
$$((\Box A^n z_2 \wedge \Box\ z_n > z_{2=\emptyset}) \vee$$
$$\Box(z_n > z_{1=\emptyset}))\{true\} \wedge$$
$$\neg \Diamond A^n z_1 \{true\})$$

Nota: Creación propia.

También se determinó en esta fase que las propuestas 1 y 2 no cumplen con el requerimiento de univocidad. El fundamento de ello es que existen *formulae* en estas propuestas, e.g. "$A^n z_n$", cuyas condiciones de verdad no están definidas, de modo que estas propuestas no cumplen con este criterio, tal como es presentado en el estándar IEEE Std. 830-1998.

Del mismo modo, se determinó que estas propuestas 1 y 2 no cumplen con el requerimiento completitud estipulado por el estándar IEEE Std. 830-1998. Ciertamente, estas propuestas no contienen todos los requerimientos respecto del performance, constricciones de diseño y su

funcionalidad:[11] concretamente, en estas propuestas no aparecen las constricciones de diseño determinadas por el carácter mismo de los *corpora*.

Ahora bien, cabe resaltar que, en la fase cualitativa del estudio, se determinó que las propuestas 1 y 2 sí cumplen con los criterios de consistencia, verificabilidad, rastreabilidad y orden por necesidad. El fundamento de esta afirmación es que aspectos son garantizados por el método formal y axiomático que se implementó para su formalización y construcción.

La construcción de las propuestas en función de modelos proposicionales garantiza la consistencia interna, ya que las definiciones son *formulae* tautológicas que implican que no es posible la derivación de elementos incompatibles con esos mismos axiomas.[12] Dicho de otra manera, la consistencia interna de las propuestas, formuladas en modelos proposicionales tautológicos, es necesaria lógicamente, puesto que, al ser modelos tautológicos, toda *formula* derivada en ese modelo será solamente una exploración tautológica de las definiciones iniciales.

La misma formalización de las propuestas en modelos proposicionales basados en axiomas que representan los requerimientos funcionales y no funcionales del software también garantiza su verificabilidad. Ciertamente, los modelos construidos de tal manera permiten que cada una de sus axiomas, los requerimientos funcionales y no funcionales del software, son derivables, en función de los otros,[13] ya que están basados en exploraciones tautológicas del mismo modelo.

El mismo caso se da respecto del criterio de rastreabilidad y orden por necesidad. En efecto, la formalización de las propuestas en modelos proposicionales permite dar cuenta de una indexación de cada *formula* que permite ubicar cada requerimiento en la documentación del software. [14] Mientras tanto, la formalización de las propuestas en lenguaje lógico que contiene elementos multimodales permite codificar las *formulae* en términos de necesidad: en los modelos que representan las propuestas de requerimientos de software en este documento, esta codificación multimodal de los requerimientos se basa en el uso de los símbolos "□" y "◇".[15]

Finalmente, es necesario señalar que la propuesta 1 no cumple con el criterio de modificabilidad, mientras que la propuesta 2 sí lo hace. Ciertamente, como se señaló anteriormente, la definición 6 del modelo que representa la propuesta 1 es una extensión de la definición 5, de modo que es redundante y el criterio de modificabilidad, tal como es presentado en el estándar IEEE Std. 830-1998, no admite redundancias. Por su parte, la propuesta 2 cumple con este criterio, en tanto no contiene elementos redundantes.

Respecto de la propuesta 3, en la fase cualitativa de esta investigación se corroboró que este cumple con todos los criterios de validación de requerimientos de software establecidos en el estándar IEEE Std. 830-1998. Ciertamente, al igual que en el caso de las propuestas 1 y 2 la formalización de estos en un modelo proposicional axiomático, tautológico y multimodal garantiza que este cumple con los criterios de consistencia, verificabilidad, rastreabilidad y orden por necesidad.

Adicionalmente, se comprobó que este no contiene elementos redundantes. Mientras tanto, también es posible afirmar que es completo y univoco, en tanto contiene las definiciones pertinentes para la definición de las constricciones del software implicadas por las características

[11] De manera similar es definido el criterio de completitud en el estándar IEEE Std. 830-1998.
[12] De manera similar es definido el criterio de consistencia en el estándar IEEE Std. 830-1998.
[13] De manera similar es definido el criterio de verificabilidad en el estándar IEEE Std. 830-1998.
[14] La rastreabilidad se define en el estándar IEEE Std. 830-1998, en función de la posibilidad de ubicar cada uno de los requerimientos dentro de la documentación.
[15] Acerca de los lenguajes proposicionales multimodales, véase Roggenbach et al. (2022).

propias de los *corpora* y contiene todas las definiciones necesarias para determinar el contenido semántico de todas las *formulae* que lo conforman.

Con todo lo anterior, es posible afirmar que las propuestas 1 y 2 solo cumplen parcialmente con los criterios de validación de requerimientos de software establecidos en el estándar IEEE Std. 830-1998. Por otra parte, se puede aseverar, con lo dicho hasta este punto, que la propuesta 3 cumple con tales criterios. Esto se puede evidenciar en la tabla 2.

Tabla 2

Cumplimiento de los criterios validación por parte de cada propuesta de requerimientos de software

Criterio	Propuesta de requerimientos de software		
	1	2	3
Precisión	No cumple	No cumple	Cumple con el requerimiento de precisión
Univocidad	No cumple	No cumple	Cumple con el requerimiento de univocidad
Consistencia	Cumple con el requerimiento de completitud	Cumple con el requerimiento de completitud	Cumple con el requerimiento de completitud
Completitud	Cumple con el requerimiento de consistencia	Cumple con el requerimiento de consistencia	Cumple con el requerimiento de consistencia
Verificabilidad	Cumple con el requerimiento de verificabilidad	Cumple con el requerimiento de verificabilidad	Cumple con el requerimiento de verificabilidad
Modificabilidad	No cumple	Cumple con el requerimiento de modificabilidad	Cumple con el requerimiento de modificabilidad
Rastreabilidad	Cumple con el requerimiento de rastreabilidad	Cumple con el requerimiento de rastreabilidad	Cumple con el requerimiento de rastreabilidad
Orden por necesidad	Cumple con el requerimiento de orden por necesidad	Cumple con el requerimiento de orden por necesidad	Cumple con el requerimiento de orden por necesidad

Continuando con los resultados de la fase cuantitativa, lo primero que es necesario señalar es que, luego de la codificación de los datos, se construyó una matriz de correlaciones. Esta matriz permite presentar los datos obtenidos tras la validación de cada una de las propuestas de requisitos de software, a la luz del paralelo con el caso hipotético de la propuesta que cumple con todos los criterios de validación de requerimientos de software establecidos en el estándar IEEE Std. 830-1998. Esta matriz de correlaciones se puede observar en la tabla 3.

Tabla 3

Matriz de correlaciones que refleja el cumplimiento de los criterios de validación de cada propuesta

Propuesta	Criterio de validación IEEE Std. 180-1998							
	Exact.	**Compl.**	**Univo.**	**Consis.**	**Verif.**	**Modif.**	**Rastr.**	**Orden por necesidad**
Hipotética	1	2	3	4	5	6	7	8
1	0	0	0	4	5	0	7	8
2	0	0	0	4	5	6	7	8
3	1	2	3	4	5	6	7	8

Dicho lo anterior, el cálculo del coeficiente de correlación de los parámetros de cumplimiento de los criterios en el caso de la propuesta 1 en contraste con los que tiene el caso de la propuesta hipotética permitió determinar que hay esta los cumple parcialmente. El coeficiente de correlación entre los datos de estos dos casos es de 0,784, de modo que hay una asociación considerable entre ellos. Esto se puede evidenciar en la figura 5.

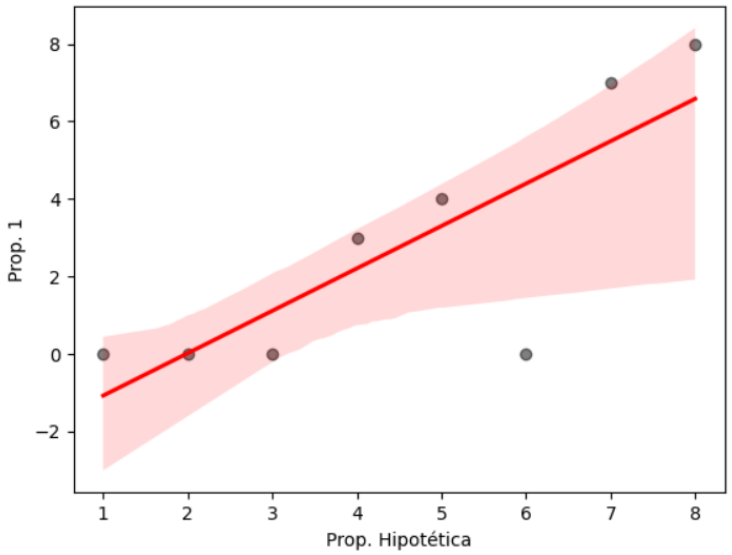
Por su parte, se determinó, mediante el cálculo de la correlación entre los datos de la propuesta 2 y los de la propuesta hipotética, que la propuesta 2 cumple en gran medida con los criterios de validación de software establecidos en el estándar IEEE Std. 830-1998. En efecto, el coeficiente de correlación que existe entre los datos del caso de la propuesta hipotética y la propuesta 2 demuestra una asociación alta entre estas dos: el coeficiente de correlación entre estos dos casos es de 0,964. Esto se puede evidenciar en la figura 6.

Finalmente, se pudo confirmar, en función del cálculo del coeficiente de correlación entre los datos de la propuesta 3 y los que se plantean en el caso de la propuesta hipotética, que la propuesta 3 cumple en su totalidad con los parámetros que definen la validación de requerimientos de software establecidos por el estándar IEEE Std. 830-1998. Ciertamente, el coeficiente de correlación entre estas dos variables es de 1. Esto se evidencia en la figura 7.

En conclusión, es posible afirmar que la propuesta 3 cumple, en mayor medida con los criterios de validación de requerimientos de software presentados en el estándar IEEE Std. 830-1998. Ciertamente, esta propuesta tuvo un coeficiente de correlación de 1 con los parámetros del caso ideal que cumple con todos esos criterios, mientras que, en las mismas condiciones, la propuesta 2 tuvo un coeficiente de correlación de 0,964 y la propuesta 3 tuvo un coeficiente de correlación de 0,784.

Figura 5

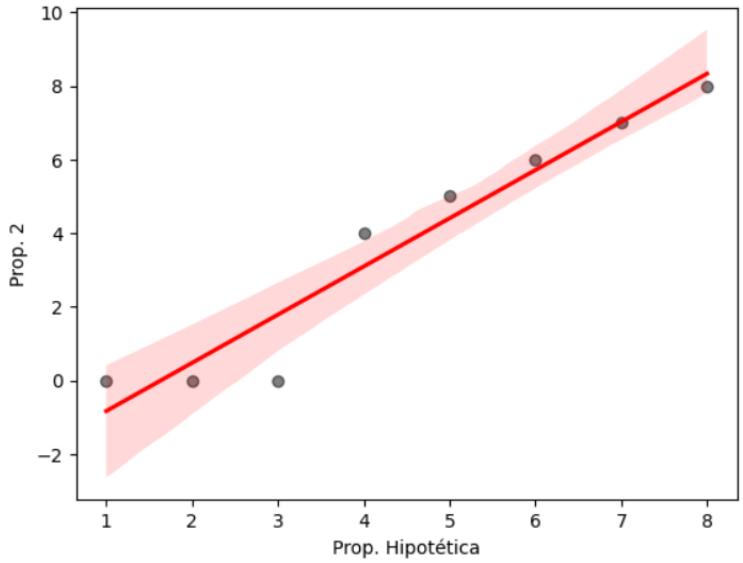
Gráfico de dispersión con recta de correlación entre los datos de la propuesta 1 y la propuesta hipotética



Nota: Creación propia.

Figura 6

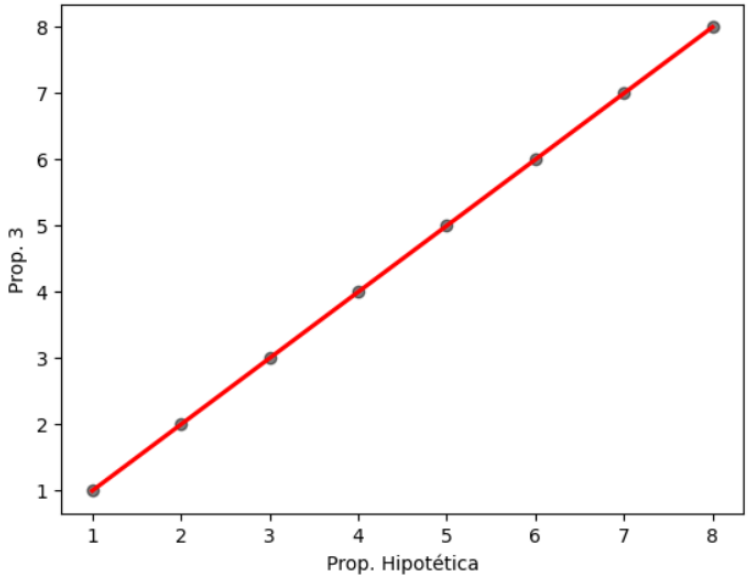
Gráfico de dispersión con recta de correlación entre los datos de la propuesta 2 y la propuesta hipotética



Nota: Creación propia.

Figura 7

Gráfico de dispersión con recta de correlación entre los datos de la propuesta 3 y la propuesta hipotética



Nota: Creación propia.

Conclusiones

El presente trabajo tuvo como objetivo determinar cuáles son los requerimientos funcionales y no funcionales de software de un paquete programado en Python para el análisis estilométrico de obras líricas mediante su estudio prosódico-métrico y los algoritmos genéticos que cumplen, en mayor medida los criterios establecidos en el estándar IEEE Std. 830-1998, enfocado

exclusivamente en los requerimientos pertinentes para el modelado de los *corpora*. Con este fin, se planteó una investigación de diseño correlacional con una fase cualitativa y una cuantitativa, mientras que su población de estudio se conformó por tres propuestas de requerimientos de software formuladas mediante el método Scrum.

En la fase cualitativa, se determinó que solo la propuesta de requerimientos de software 3 cumple con los criterios de validación establecidos en el estándar IEEE Std. 830-1998. En efecto, las propuestas 1 y 2 solo cumplen con una parte de tales criterios. Esto se comprobó, en función del diligenciamiento de una lista de chequeo que funcionó como instrumento de recolección de datos en esta fase.

Por su parte, en la fase cuantitativa, se concluyó que la propuesta de requerimientos de software 3 cumple, en mayor medida con los criterios de validación establecidos en el estándar IEEE Std. 830-1998, mediante el calculo del coeficiente de correlación entre los datos de cumplimiento de tales criterios propios de cada propuesta y los datos de cumplimiento de una propuesta hipotética que los cumple a cabalidad. Para ello, se codificaron tales datos y se concluyó que los datos de la propuesta 3 tienen un coeficiente de correlación con los datos de la propuesta hipotética igual a 1, mientras que, en las mismas circunstancias, los datos de la propuesta 2 tuvieron un coeficiente de correlación de 0,964 y los de la propuesta 1 tuvieron un coeficiente de correlación de 0,784.

Referencias bibliográficas

Aurum, A.; Wohlin, C. (2005). Requirements Engineering: Setting the Context. En A. Aurum; C. Wohlin (2005) (Ed.), *Engineering and Managing Software Requirements: With 54 Figures and 48 Tables* (pp. 1-12). Springer.

Balogh, B.; Janiak, A. (2023). *DHARMA Zotero Guide* [Archivo PDF]. https://github.com/erc-dharma/project-documentation/blob/master/docs/zotero/DHARMA_ZoteroGuide_v01.1.1.pdf

Balogh, B.; Griffiths, A. (2020). *DHARMA Encoding Guide for Diplomatic Editions*. HAL open science.

- (2020a). *DHARMA Transliteration Guide.* HAL open science.

Bernabé, A. (2010). *Manual de crítica textual y edición de textos griegos*. Akai.

Bjørner, D. (2006). *Software Engineering 3: Domains, Requirements, and Software Design with 100 Figures.* Springer.

Chemuturi, M. (2018). *Software Design: A Comprehensive Guide to Software Development Projects.* CRC Press.

Cross, N. (2021). *Engineering Desing Methods: Strategies for Product Design*.Wiley

Dooley, J. F. (2017). *Software Development, Design and Coding: with Patterns, Debugging, Unit Testing, and Refactoring.* Apress.

Finney, T. (2018). How to discover textual groups. *Digital Studies/ Le Champ Numérique*, 8 (1), 1-99.

Geer, T. C.; Racine, J. F. (2013). Analyzing and categorizing New Testament Greek manuscripts. En B. D. Ehreman; M. W. Holmes (2013) (Ed.), *The Text of the New Testament in Contemporary Research: Essays on the Status Questionis* (pp. 497-518). Brill.

Griffiths, A.; Janiak, A. (2023). *DHARMA Encoding Guide for Critical Editions*. HAL open science.

Hyytiänen (2023). A new method in establishing quantitative relationships between manuscripts of the New Testament. *Digital Scholarship in the Humanities*, 38(1), 151-166.

- (2021). The changing text of acts: a phylogenetic approach. *TC: A Journal of Biblical Textual Criticism*, 26, 1-28. https://jbtc.org/v26/TC-2021-Hyyti%C3%A4inen.pdf

IEEE (1998). *IEEE Std. 830-1998 Software Requirements Specification*. The Institute of Electrical and Electronics Engineers.

IEEE (1990). *IEEE Std. 610.12-1990 IEEE Standard Glossary of Software Engineering Terminology*. The Institute of Electrical and Electronics Engineers.

Koch, S. (2014). *Genetische Algorithmen für das Order Batching-Problem in manuellen Kommissioniersystemen*. Springer.

Franco, J.; Martin, J. (2021). A History of Satisfiability. En A. Biere; M. Heule; H. Van Maaren; T. Walsh (2021) (Ed.), *Handbook of Satisfiability* (2. ed., pp. 3-74.). IOS Press.

Lin, Y. J. (2016). *The Erotic Life of Manuscripts: New Testament Textual Criticism and the Biological Sciences*. Oxford University Press.

McCollum, J. (2019). Biclustering readings and manuscripts via non-negative matrix factorization, with application to the text of Jude. *AUSS*, 57 (1), 61-89.

Monje Álvarez, C. A. (2011). *Metodología de la investigación cuantitativa y cualitativa. Guía didáctica*. Universidad Surcolombiana.

Morrás, M. (2003). Informática y crítica textual: Realidades y deseos. En M. J. Vega (2003) (Ed.), *Literatura hipertextual y teoría literaria* (pp. 225-241). Mare Nostrum Comunicación.

Roggenbach, M.; Cerone, A.; Schlingloff, B. H.; Schneider, G.; Shaikh; S. A. (2022). *Formal methods for Software Engineering: Languages, Methods, Application Domains.* Springer.

Sethi (2023). *Software Engineering: Basic Principles and Best Practices*. Cambridge University Press.

Shaoying, L.; Hayashi, T.;Takahashi, K.; Nakayama, T. (2009). Teaching Formal Methods in the Context of Software Engineering. *SIGCSE Bulletin*, 41(2), 17-23.

Wasserman, T.; Gurry, P. (2017). *A new Approach to Textual Criticism: An Introduction to the Coherence- Based Genealogical Method*. Society of Biblical Literature.