[1]

Akour, M., & Falah, B. (2016). Application domain and programming language readability yardsticks. In 2016 7th International Conference on Computer Science and Information Technology (CSIT) (pp. 1–6). IEEE. <https://doi.org/10.1109/CSIT.2016.7549476>

**PARA ANEXAR A DOCUMENTO:**

**TAGS**

DOMINIO; LENGUAJE DE PROGRAMACION; METRICAS; FACTORES DE LEGIBILIDAD VARIABLES; EVOLUCION;

**\*\* INICIO \*\*\***

**RESPONDIENDO A PREGUNTA DE INVESTIGACIÓN 1:**

Akour et al (2016) [1], indicaron que los modelos de medición de la legibilidad del software deben ser especializados, teniendo en cuenta el dominio de la aplicación y el lenguaje de programación en el cual está desarrollado [1]. Los autores explicaron que las características evaluadas para determinar la legibilidad del código fuente no pueden medirse de la misma forma en una aplicación A y en una aplicación B de un dominio diferente, porque hay unos lenguajes de programación que facilitan más la legibilidad que otros, debido a la naturaleza misma de su sintaxis o de acuerdo a su paradigma, que puede ser procedimental, funcional, imperativo, orientado a objetos, lógico o simbólico [1]. De acuerdo a lo anterior, las métricas propuestas por los diferentes autores deberían tener la posibilidad de tener factores de legibilidad variables, que se adapten a un código fuente teniendo en cuenta su dominio y su lenguaje de programación [1].

**FACTORES:**

**DOMINIOS DE APLICACIÓN Y LENGUAJES DE PROGRAMACIÓN**

Akour et al (2016) [1], argumentaron que cuando se va a desarrollar un software y se define el dominio al que corresponde, eso condiciona la elección del lenguaje de programación que se usará, teniendo en cuenta que algunos lenguajes se especializan o facilitan el desarrollo de algunas operaciones. A continuación, se presenta una recopilación del autor, producto de su investigación, con los lenguajes de programación relacionados al dominio en el cual se suelen utilizar [1].

|  |  |  |
| --- | --- | --- |
| **Dominio** | **Lenguaje de programación** | **Referencia** |
| Negocios | Lenguaje COBOL. Usado en el 70% de las transacciones de negocios a nivel mundial. PL/B también es un lenguaje moderno popular en el dominio de negocios. | [1] |
| e-commerce | PHP, Ruby. | [1] |
| Herramientas de gestión de proyectos | PHP y Java son los lenguajes más populares. Otros son, Python, ASP.NET, Ruby, C, y SQLite. | [1] |
| CMS (sistemas de gestión de contenido) | la mayoría con PHP (11) y 1 con Ruby. | [1] |
| Inteligencia artificial | Lenguajes Lisp, Prolog (es mas conciso para éste y programación no-numérica). | [1] |
| Desarrollo de videojuegos | C, Lua, and C++. Para el acceso rápido al hardware de la pantalla. | [1] |
| Sistemas embebidos | C, C++, and java, porque ofrecen varias funcionalidades como expresioness, flujo de control, funciones recursivas, excepciones, recolector de basura. | [1] |
| Computación en la nube | El Mapreduce es un modelo computacional utilizado para procesar grandes conjuntos de datos de manera masivamente paralela. Las bibliotecas de este modelo computacional han sido escritas en C ++ y en java | [1] |

**EVOLUCIÓN DE LOS MODELOS Y LAS MÉTRICAS DE LEGIBILIDAD**

|  |  |
| --- | --- |
| **Autores** | **Descripción** |
| **Raymond P.L. et al [4] (Buse y Weimer 2010)** | especificó algunas características generales de legibilidad que deben tenerse en cuenta al medir la legibilidad de cualquier código dado. Los autores declararon que la métrica debe tener en cuenta lo siguiente:  **línea de longitud, identificadores, longitud del identificador, sangría, números de palabras clave, comentarios, puntos, comas, espacios, paréntesis, líneas en blanco y ocurrencias de cualquier carácter individual.** |
| Dorn. [5] (2012, Virginia) | mostró que decidir la legibilidad del código depende en gran medida de la estructura visual y espacial del programa. Cualquier métrica de legibilidad fuerte debe incorporar algún conocimiento sobre la estructura del idioma, las reglas impuestas y la organización. Su modelo de legibilidad propuesto capta que la legibilidad tiene dos aspectos: un aspecto universal y un aspecto particular. El aspecto particular de las características se debe al lenguaje de programación elegido del experimento realizado. Mientras que el aspecto universal, según el autor, está ligado a la longitud de la línea. |
| Collar y Valeri [6](Collar .Ir E, Valerdi 2006) | Propusieron que la legibilidad debería tener en cuenta la estructura del código. |

**\*\*\* FIN \*\*\***

**NOTAS**

De acuerdo a los modelos existentes, no todas las métricas propuestas tienen en cuenta la posibilidad de tener factores de legibilidad variables.

Este documento trata de investigar hasta qué punto los factores de legibilidad tienen en cuenta diferentes dominios de aplicación. Dado que la legibilidad es muy subjetiva, será interesante ver si es útil contar con modelos especializados de legibilidad que midan este atributo. Esto también implica analizar el impacto de estos modelos de legibilidad en la medición de la calidad del software.

En el dominio de la legibilidad, la investigación no explica cómo la diversidad de los factores de legibilidad afectaría los resultados cuando se realicen métricas de legibilidad en diferentes aplicaciones de dominio de aplicación. Por lo tanto, el principal problema que debe abordarse aquí es la falta de modelos especializados en el contexto de la legibilidad del software.

TEMAS CLAVE:

* Relación entre lenguajes de programación y la legibilidad.
* Variación de las métricas de legibilidad de acuerdo al dominio de la aplicación ( y al lenguaje de programación).

LEGIBILIDAD

Los desarrolladores pasan la mayor parte del tiempo tratando de leer y entender el código durante la fase de mantenimiento. Es decir, que menos tiempo de lectura en el código afecta directamente la cantidad de tiempo para mantener el software. La legibilidad del código es altamente subjetiva. Está directamente relacionado con el nivel de facilidad por el cual una persona lee un programa. Los niveles de los lectores dependen principalmente de su capacidad cognitiva para procesar información textual. La rapidez también contribuye a esta subjetividad, ya que cuando una persona trata de leer un programa, el código debe escribirse de tal manera que le tome menos tiempo leer y entender las líneas. Por lo tanto, el texto no debe ralentizar esta operación.

FACTORES

Rudolf Flesch (1948) indicó que la legibilidad de un texto dado está influenciada por tres factores principales: la longitud promedio de los sentencias, la longitud promedio de las palabras en sílabas, el porcentaje promedio de palabras personales y las sentencias. Estos factores se utilizaron para calcular las pruebas propuestas “Flesch Reading Ease” y “Flesch-Kincaid Grade”. Algunos editores de texto populares han integrado esas dos pruebas, como la verificación gramatical de MS.

Se deben tener en cuenta diferentes factores cuando se mide la legibilidad de un producto de software.

La legibilidad del código es una medida importante para medir la complejidad de un sistema dado. La complejidad de un sistema de software varía según el dominio de su aplicación. Por lo tanto, la forma de medir la legibilidad se ve directamente afectada por los dominios de la aplicación. Desafortunadamente, no todas las métricas propuestas tienen en cuenta la posibilidad de tener factores de legibilidad variables. De hecho, hay una falta en cómo esta variabilidad afectaría los resultados encontrados al realizar métricas de legibilidad en el software. Dicho esto, el principal problema que debe abordarse aquí es la falta de modelos especializados en el contexto de la legibilidad de diferentes dominios de aplicación.

Para desarrollar una aplicación específica en un dominio específico, **la elección de un lenguaje de programación juega un papel importante en la legibilidad del código.** Algunos idiomas tienden a ser más legibles que otros, debido a que su sintaxis tiene la naturaleza inherente de no hacer ruido. Los lenguajes son descritos como verbosos (mas palabras de las necesarias, con detalles adicionales) o concisos. Un lenguaje detallado como Java tiene un vocabulario muy rico y símbolos para una mejor expresividad que lo hacen redundante [3]. Otros lenguajes de programación son breves por naturaleza, de modo que pueden transmitir la solución de una manera muy concisa. Josh Bloch [3] ha expresado que "un poco de redundancia en un lenguaje es algo bueno. Es importante para la legibilidad". Por supuesto, una menor cantidad de palabras no significa necesariamente que el código sea más legible, ya que podría hacerlo ambiguo.

No significa que los lenguajes verbales sean más o menos legibles. A menudo depende de la familiaridad del lector para trabajar con verbosos o concisos. Pero algunos idiomas, gracias a su naturaleza verbosa, permiten más expresividad y, por lo tanto, más opciones para poder utilizar la sintaxis simple. Efectivamente, la sintaxis simple eventualmente resultaría en la familiaridad del lector con el código más rápido que si el lector estuviera expuesto a una nueva sintaxis compleja.

investigación para ver los efectos que los lenguajes de programación pueden tener para orientar la investigación sobre la legibilidad del software hacia el enfoque en modelos orientados a dominios de aplicación. Además, exploramos el panorama de la literatura para encontrar una **relación existente entre la legibilidad del software y el uso de un lenguaje de programación específico**.

TRABAJOS RELACIONADOS

Divididos en 2 areas: modelos y conjuntos de métricas de legibilidad. y 2) elección de un lenguaje de programación como factor de legibilidad.

1. MODELOS Y CONJUNTOS DE MÉTRICAS

La primera área está relacionada con el trabajo que se realizó en la construcción de un conjunto de métricas y modelos de legibilidad para garantizar la calidad del software.

Raymond P.L. et al [4] (Buse y Weimer 2010) especificó algunas características generales de legibilidad que deben tenerse en cuenta al medir la legibilidad de cualquier código dado. Los autores declararon que la métrica debe tener en cuenta lo siguiente:

**línea de longitud, identificadores, longitud del identificador, sangría, números de palabras clave, comentarios, puntos, comas, espacios, paréntesis, líneas en blanco y ocurrencias de cualquier carácter individual.**

Es posible notar que el problema detrás de esta métrica es la estandarización de los factores elegidos que se aplican a cualquier idioma utilizado en un código de fragmento de código. De hecho, no tuvieron en cuenta el hecho de que el código se puede escribir utilizando diferentes lenguajes de programación que se encuentran bajo diferentes paradigmas: procedimental, funcional, imperativo, orientado a objetos, lógico y simbólico. Esos diferentes lenguajes de programación imponen algunos de los factores mencionados anteriormente, mientras que otros no lo hacen. Por lo tanto, es necesario ponderar esos factores según el lenguaje de programación utilizado al medir la legibilidad

Dorn. [5] (2012, Virginia) mostró que decidir la legibilidad del código depende en gran medida de la estructura visual y espacial del programa. En realidad, había señalado un punto muy importante relacionado con el hecho de que, después de todo, se están leyendo los códigos, lo que significa que cualquier métrica de legibilidad fuerte debe incorporar algún conocimiento sobre la estructura del idioma, las reglas impuestas y la organización. Su modelo de legibilidad propuesto capta que la legibilidad tiene dos aspectos: un aspecto universal y un aspecto particular. El aspecto particular de las características se debe al lenguaje de programación elegido del experimento realizado. Mientras que el aspecto universal, según el autor, está ligado a la longitud de la línea. Independientemente de su afirmación, las características, que afectan la legibilidad, varían de un lenguaje de programación a otro. Dorn. J no probó el modelo en lenguajes que tienen un alto grado de variedad, como el lenguaje de programación de consulta SQL, el cual podría no ser necesariamente afectado por la longitud de la línea.

Collar y Valeri [6](Collar .Ir E, Valerdi 2006) propusieron que la legibilidad debería tener en cuenta la estructura del código. Sin embargo, no tuvieron en cuenta el efecto real de los lenguajes de programación y sus dominios de aplicación relacionados en la estructura mencionada. Además de eso, han inferido que los lenguajes de programación podrían tener un papel que desempeñar durante el proceso. No han respondido a la larga lista de preguntas relacionadas con el hecho de que los idiomas están muy ajustados a los dominios de aplicación.

2) LENGUAJE DE PROGRAMACIÓN COMO FACTOR DE LEGIBILIDAD

La elección del lenguaje de programación como un posible factor de legibilidad.

Python, obliga al programador a usar la identación. La identación mejora la legibilidad, facilitando la lectura (los bloques se representan gráficamente). En otros lenguajes los corchetes son útiles para definir inicio y final de los bloques de código, pero la naturaleza de estos lenguajes no impone el uso de espacios en blanco para un código bien estructurado.

SQL no es sensible a las mayúsculas, pero algunos profesionales adoptan la práctica de escribir en mayúscula las palabras propias del lenguaje, y en minúscula los nombres de los identificadores.

Lenguaje verboso: como MapReduce (usa Java).

Lenguaje conciso, como Pig.

Pig tiende a ser mas legible que MapReduce, porque Pig usa un conjunto de palabras clave y sintaxis más intuitiva y comprimida lo cual incrementa la familiaridad con el desarrollador, mientras que con MapReduce el mismo programa requiere un mayor número de líneas de código (más complejo y menos legible).

El desafío sería aplicar modelos de legibilidad anteriores a estos dos ejemplos, que están escritos en dos idiomas diferentes, pero producen la misma funcionalidad. Los resultados se usarían para evaluar la validez de los modelos de legibilidad propuestos, junto con proponer una idea a los desarrolladores dentro del dominio de la aplicación de computación en la nube, para usar un lenguaje en lugar de otro para algunas funcionalidades específicas con el fin de mejorar las posibilidades de su Código para ser legible.

APLICACIONES DE DOMINIO Y LENGUAJES DE PROGRAMACIÓN

Dominio de negocios: Lenguaje COBOL. Usado en el 70% de las transacciones de negocios a nivel mundial. PL/B también es un lenguaje moderno popular en el dominio de negocios.

Dominio e-commerce: PHP, Ruby.

Dominio de Herramientas de gestión de proyectos: PHP y Java son los lenguajes más populares. Otros son, Python, ASP.NET, Ruby, C, y SQLite.

CMS (sistemas de gestión de contenido): la mayoría con PHP (11) y 1 con Ruby.

Dominio de inteligencia artificial: Lenguajes Lisp, Prolog (es mas conciso para éste y programación no-numérica)

.

Dominio de desarrollo de video juego: C, Lua, and C++. Acceso rapido al hardware de la pantalla.

Sistemas embebidos: C, C++, and java, porque ofrecen varias funcionalidades como expresioness, flujo de control, funciones recursivas, excepciones, recolector de basura.

Computación en la nube: El Mapreduce es un modelo computacional utilizado para procesar grandes conjuntos de datos de manera masivamente paralela. Las bibliotecas de este modelo computacional han sido escritas en C ++ y en java.

PROPUESTA:

modelo especializado que enfatice los factores de legibilidad más relevantes de los diferentes lenguajes de programación que caen bajo ese dominio de aplicación.

Para mejorar la calidad del software, Ming [13] (Ming-Chang 2014). investigó los factores de calidad del software y las métricas de calidad del software. Su trabajo muestra cómo la legibilidad del software desempeña un papel crucial en el aumento de la calidad del software. Ermira et al. [14] (Daka E, 2015) muestra cómo las pruebas ilegibles son más difíciles de mantener y pierden algo de su valor para los desarrolladores. Proponen un modelo de dominio específico de la capacidad de prueba de la unidad basado en juicios humanos, y utilizan este modelo para aumentar la generación de pruebas de unidades automatizadas.

Hemos demostrado que existe una fuerte relación entre la legibilidad del software y el uso de un lenguaje de programación específico, debido a una serie de factores, como los paradigmas de programación y las restricciones estructurales. También hemos explorado los dominios de aplicación existentes que se basan en lenguajes de programación específicos (o paradigma de programación). Por lo tanto, afirmamos que debería haber modelos especializados para probar la legibilidad de los programas que beneficiarían a las compañías altamente especializadas en su ciclo de mantenimiento de software.