Análisis de las tesis de grado orientadas a los lenguajes de programación de las universidades del Ecuador

1. **Introducción**

La informática es la ciencia que abarca el estudio y aplicación del tratamiento automático de la información; tales como la programación para el desarrollo de software, las redes de datos, la arquitectura de las computadoras y la inteligencia artificial. La informática ha sido aplicada en diferentes campos de investigación, tales como la medicina, la gestión de negocios, la física, las telecomunicaciones, la educación y otros campos, lo que ha permitido conseguir más y mejores resultados en estos campos. Esto conlleva al estudio de las teorías y los artefactos que posibilitan el desarrollo de **prácticas comunicativas.**

En Ecuador existe la Red de Repositorios de Acceso Abierto (RRAAE), cuya función es recolectar y tener a disposición todos los trabajos de investigación, ya sean tesis de grado, tesis de maestrías, tesis doctorales o artículos científicos publicados por las universidades ecuatorianas. Esto facilita el estudio de cada una de ellas para la recolección de datos e información requeridos para realizar este trabajo de investigación.

Este trabajo analiza las tesis de grados del área de informática que han desarrollado algún tipo de tecnologías de comunicación entre los años 2015 y 2020, y determina:

1. ¿Cuántas tesis de grado en las universidades ecuatorianas han desarrollado algún tipo de software en el periodo comprendido entre los años 2015 y 2020?
2. ¿En qué ámbito se han implementado el software desarrollado en las tesis de grado?
3. ¿En qué áreas del conocimiento han sido aplicados los softwares desarrollados en las tesis de grado?
4. ¿Cuáles son los lenguajes de programación más utilizados en los softwares desarrollados en las tesis de grado?
5. ¿Cuál es el lenguaje de programación más utilizado en los softwares desarrollados en las tesis de grado?
6. ¿Cómo ha evolucionado en uso de los lenguajes de programación en las tesis de grado?
7. ¿Qué paradigmas de programación se han utilizado en los softwares desarrollados en las tesis de grado?
8. ¿Cuáles son las universidades que aplican en sus tesis a los lenguajes de programación más utilizados?
9. ¿Cuál es la universidad que más ha desarrollado software en las tesis de grado?
10. ¿Cuál es la universidad con más tesis orientadas al desarrollado software?

**Plan del estudio.** El resto del documento está organizado como sigue. En la Sección 2 se definen los conceptos básicos de las tecnologías de comunicación y otros términos relacionados con el estudio. La Sección 3 se detalla el proceso del mapeo sistemático. La Sección 4 muestra los resultados del estudio. Finalmente, en la Sección 5 se presentan las conclusiones y algunas recomendaciones.

1. **Marco teórico**

En esta sección se definen los conceptos necesarios para el resto del documento.

**Lenguajes de programación**

De manera general y con base en la definición de varios autores, un lenguaje de programación es un lenguaje formal que, mediante una serie de instrucciones, le permite a un programador escribir un conjunto de órdenes, acciones consecutivas, datos y algoritmos para, de esa forma, crear programas que controlen el comportamiento físico y lógico de una máquina.

Mediante este lenguaje se comunican el programador y la máquina, permitiendo especificar, de forma precisa, aspectos como:

* cuáles datos debe operar un software específico;
* cómo deben ser almacenados o transmitidos esos datos;
* las acciones que debe tomar el software dependiendo de las circunstancias variables.

Para explicarlo mejor (en otras y con menos palabras), el lenguaje de programación es un sistema estructurado de comunicación, el cual está conformado por conjuntos de símbolos, palabras claves, reglas semánticas y sintácticas que permiten el entendimiento entre un programador y una máquina.

Es importante recalcar que existe el error común de usar como sinónimos el lenguaje de programación y el lenguaje informático, pero ¿por qué no se debe confundirlos?

Pues, es debido a que el lenguaje de programación obedece a un conjunto de reglas que permiten expresar las instrucciones que serán interpretadas por el programador. Y el lenguaje informático comprende otros lenguajes que dan formato a un texto, pero no son programación en sí mismos.

Entonces, no todos los lenguajes informáticos son de programación, pero todos los lenguajes de programación son a la vez informáticos.

El lenguaje de programación es la base para construir todas las aplicaciones digitales que se utilizan en el día a día y se clasifican en dos tipos principales: lenguaje de bajo nivel y de alto nivel.

* **Lenguaje de bajo nivel:**

Son lenguajes totalmente orientados a la máquina.

Este lenguaje sirve de interfaz y crea un vínculo inseparable entre el hardware y el software.

Además, ejerce un control directo sobre el equipo y su estructura física. Para aplicarlo adecuadamente es necesario que el programador conozca sólidamente el hardware. Éste se subdivide en dos tipos:

* **Lenguaje máquina**

Es el más primitivo de los lenguajes y es una colección de dígitos binarios o bits (0 y 1) que la computadora lee e interpreta y son los únicos idiomas que las computadoras entienden.

Ejemplo: **10110000 01100001**

* **Lenguaje ensamblador**

El lenguaje ensamblador es el primer intento de sustitución del lenguaje de máquina por uno más cercano al utilizado por los humanos.

Un programa escrito en este lenguaje es almacenado como texto (tal como programas de alto nivel) y consiste en una serie de instrucciones que corresponden al flujo de órdenes ejecutables por un microprocesador.

Sin embargo, dichas máquinas no comprenden el lenguaje ensamblador, por lo que se debe convertir a lenguaje máquina mediante un programa llamado Ensamblador.

Este genera códigos compactos, rápidos y eficientes creados por el programador que tiene el control total de la máquina.

Ejemplo: **MOV AL, 61h** (asigna el valor hexadecimal 61 al registro «AL»)

* **Lenguaje de programación de alto nivel:**

Tienen como objetivo facilitar el trabajo del programador, ya que utilizan unas instrucciones más fáciles de entender.

Además, el lenguaje de alto nivel permite escribir códigos mediante idiomas que conocemos (español, inglés, etc.) y luego, para ser ejecutados, se traduce al lenguaje de máquina mediante traductores o compiladores.

* **Traductor**

Traducen programas escritos en un lenguaje de programación al lenguaje máquina de la computadora y a medida que va siendo traducida, se ejecuta.

* **Compilador**

Permite traducir todo un programa de una sola vez, haciendo una ejecución más rápida y puede almacenarse para usarse luego sin volver a hacer la traducción.

**tecnología de equipo:** engloba los procedimientos tecnológicos que se aprovechan en distintas industrias de fabricación, investigación o desarrollo.

**tecnología de producto:** que combina los conocimientos y las herramientas que unidas pueden permitir la creación de un producto específico.

En general un lenguaje de programación sirve para programar. Sin embargo, cada uno tiene un alcance y forma de comunicación diferente.

En resumidas cuentas, el lenguaje de bajo nivel permite la comunicación interna de la máquina, cada instrucción tiene su código único de operación.

Y el lenguaje de alto nivel facilita la captación de instrucciones que el programador le da a la máquina, mientras que éste introduce datos en el idioma conocido la máquina lo va absorbiendo en lenguaje de máquinas mediante traductores o compiladores, permitiendo así:

* reducir el tiempo de programación;
* entender más fácilmente la tarea a realizar;
* permitir al programador desvincularse del funcionamiento interno de la máquina, entre otros.

En otras palabras, el lenguaje de bajo nivel es cercano a los idiomas de las máquinas mientras que el lenguaje de alto nivel está más cerca del entendimiento e idioma humano.

**Softwares de programación**

Por software de programación entendemos el conjunto de todas las herramientas que le permiten al programador, crear, escribir códigos, depurar, mantener y empaquetar los proyectos.

Algunos de los distintos programas por los que pasará el proyecto para gestionarlo son:

• **Editores de código o texto**

Al escribir los códigos se auto-completan marcando los errores sintácticos y la refactorización.

• **Compiladores**

Estos traducen el código ingresado a lenguaje de máquina generando un código binario ejecutable.

• **Depuradores**

Sirven para optimizar el tiempo de desarrollo mediante el monitoreo de la ejecución de un programa, el seguimiento a los valores de ciertas variables, las referencias a objetos en memoria y, por ende, nos ayuda a corregir errores.

* **Enlazadores**

Este programa toma objetos generados en los primeros pasos del proceso de compilación y los recursos necesarios de la biblioteca, quita aquellos procesos y datos que no necesita, y enlaza el código con dicha biblioteca para así aumentar su tamaño y extensión.

* **Interpretadores o traductores**

El traductor (o intérprete) carga el código ingresado y traduce las instrucciones para que el programa pueda ser ejecutado.

* **IDE**

El IDE (Integrated Development Environment) o **Entorno de Desarrollo Integrado**, es una aplicación informática que proporciona una serie de servicios que facilitan la programación de software, tales como:

* funciones de autocompletado;
* un editor de código fuente;
* gestión de conexiones a bases de datos;
* integración con sistemas de control de versiones;
* simuladores de dispositivos;
* un depurador para agilizar el proceso de desarrollo de software, entre otros.

En resumen, sin el lenguaje de programación, programar sería imposible, debido a que no existirían reglas (tanto semánticas como sintácticas), expresiones (como la estructura y el significado de todos los elementos que los componen) ni una forma establecida sobre cómo deben “hablar” el programador y la máquina.

Además, algunas de las funciones que le permiten a un programador crear este lenguaje son: crear una web y hacerla funcionar o desarrollar aplicaciones para los sistemas operativos, entre muchas otras.

En la actualidad, el conocimiento y uso del mundo digital y de la informática son dos de las principales armas de cualquier empresa.

**Paradigmas**

Los paradigmas de programación son un estilo de desarrollo de programas. Es decir, un modelo para resolver problemas computacionales. Los lenguajes de programación, necesariamente, se encuadran en uno o varios paradigmas a la vez a partir del tipo de órdenes que permiten implementar, algo que tiene una relación directa con su sintaxis.

Existen un conjunto de paradigmas de programación que son utilizados al momento de desarrollar software, entre los cuales tenemos:

* **Imperativo.**

Los programas se componen de un conjunto de sentencias que cambian su estado. Son secuencias de comandos que ordenan acciones a la computadora.

Los programas consisten en una sucesión de instrucciones o conjunto de sentencias, como si el programador diera órdenes concretas. El desarrollador describe en el código paso por paso todo lo que hará su programa.

Algunos lenguajes: Pascal, COBOL, FORTRAN, C, C++, etc.

Otros enfoques subordinados al paradigma de programación imperativa son:

* + **Programación estructurada:** La programación estructurada es un tipo de programación imperativa donde el flujo de control se define mediante bucles anidados, condicionales y subrutinas, en lugar de a través de GOTO.
  + **Programación procedimental:** Este paradigma de programación consiste en basarse en un número muy bajo de expresiones repetidas, englobarlas todas en un procedimiento o función y llamarlo cada vez que tenga que ejecutarse.
  + **Programación modular:** consiste en dividir un programa en módulos o subprogramas con el fin de hacerlo más manejable y legible. Se trata de una evolución de la programación estructurada para resolver problemas de programación más complejos.
* **Declarativo.**

Opuesto al imperativo. Los programas describen los resultados esperados sin listar explícitamente los pasos a llevar a cabo para alcanzarlos.

Este paradigma no necesita definir algoritmos puesto que describe el problema en lugar de encontrar una solución al mismo. Este paradigma utiliza el principio del razonamiento lógico para responder a las preguntas o cuestiones consultadas.

Este paradigma a su vez se divide en dos:

**Programación Lógica:** El problema se modela con enunciados de lógica de primer orden. Prolog

**Programación funcional:** Los programas se componen de funciones, es decir, implementaciones de comportamiento que reciben un conjunto de datos de entrada y devuelven un valor de salida. Lisp, Scala, Java, Kotlin

* **Orientado a objetos.**

El comportamiento del programa es llevado a cabo por objetos, entidades que representan elementos del problema a resolver y tienen atributos y comportamiento.

En este modelo de paradigma se construyen modelos de objetos que representan elementos (objetos) del problema a resolver, que tienen características y funciones. Permite separar los diferentes componentes de un programa, simplificando así su creación, depuración y posteriores mejoras. La programación orientada a objetos disminuye los errores y promociona la reutilización del código. Es una manera especial de programar, que se acerca de alguna manera a cómo expresaríamos las cosas en la vida real.

Podemos definir un objeto como una estructura abstracta que, de manera más fiable, describe un posible objeto del mundo real y su relación con el resto del mundo que lo rodea a través de interfaces. Ejemplos de lenguajes de programación orientados a objetos serían Java, Python o C#.

La programación orientada a objetos se sirve de diferentes conceptos como:

* Abstracción de datos
* Encapsulación
* Eventos
* Modularidad
* Herencia
* Polimorfismo

Otros son de aparición relativamente reciente y no forman parte del grupo principal:

* **Dirigido por eventos.** El flujo del programa está determinado por sucesos externos (por ejemplo, una acción del usuario).
* **Orientado a aspectos.** Apunta a dividir el programa en módulos independientes, cada uno con un comportamiento bien definido.

1. **Método de investigación**

En esta investigación se realizó un mapeo sistemático que analiza las tesis de grado orientadas al desarrollo de software utilizando los diferentes lenguajes y paradigmas de programación en las universidades ecuatorianas. Un mapeo sistemático es un estudio que categoriza y resume todas las investigaciones relevantes sobre un tema de interés y hace una representación visual.

La Table 1 muestra los resultados obtenidos con el buscador de la RRAAE, en donde se accedió a 242 tesis de grado orientadas al desarrollo de software, cabe resaltar que la última búsqueda se hizo a corte de enero 30 del 2021. Después de aplicar el proceso de selección

**Table 1.** Resultados de búsqueda y fuente primaria de estudio

| **Fuente** | **Fecha de búsqueda** | | **Resultados de búsqueda** | **Fuente primaria de estudio** |
| --- | --- | --- | --- | --- |
| RRAAE | 30/01/2021 | 242 | | RRAAE |

**Preguntas de investigación**

El objetivo principal de este mapeo sistemático es clasificar, cuantificar y cualificar las tecnologías de comunicación que se ha desarrollado en las tesis de grado de las universidades ecuatorianas. Por lo que se requiere determinar características de las tecnologías, tales como objetivos principales cubiertos, aplicación, tendencia de interés y la forma cómo se han desarrollado estas tecnologías. Las siguientes preguntas de investigación se plantean de acuerdo con este objetivo:

1. ¿Cuántas tesis de grado en las universidades ecuatorianas han desarrollado algún tipo de software en el periodo comprendido entre los años 2015 y 2020?
2. ¿En qué ámbito se han implementado el software desarrollado en las tesis de grado?
3. ¿En qué áreas del conocimiento han sido aplicados los softwares desarrollados en las tesis de grado?
4. ¿Cuáles son los lenguajes de programación más utilizados en los softwares desarrollados en las tesis de grado?
5. ¿Cuál es el lenguaje de programación más utilizado en los softwares desarrollados en las tesis de grado?
6. ¿Cómo ha evolucionado en uso de los lenguajes de programación en las tesis de grado?
7. ¿Qué paradigmas de programación se han utilizado en los softwares desarrollados en las tesis de grado?
8. ¿Cuáles son las universidades que aplican en sus tesis a los lenguajes de programación más utilizados?
9. ¿Cuál es la universidad que más ha desarrollado software en las tesis de grado?
10. ¿Cuál es la universidad con más tesis orientadas al desarrollado software?

**Estrategia de búsqueda**

La cadena de búsqueda utilizada está definida en la Tabla 2. Esta cadena está formada por una combinación de términos unidos por operadores lógicos “AND” y “OR” que permite acotar la búsqueda al objetivo deseado.

**Tabla 2.** Cadena de búsqueda definida para el mapeo sistemático

| **Cadena de búsqueda** |
| --- |
| Title= (“lenguajes programacion”) OR Subject= (“lenguajes programcion”) AND Match=(“any term”) AND Language=(“spa”) AND Format=(barchelortThesis) AND Year between (2015 and 2020) |

**Selección de estudios**

Para el proceso de selección de trabajos se establecieron los criterios de inclusión y exclusión. Los criterios de inclusión son los siguientes:

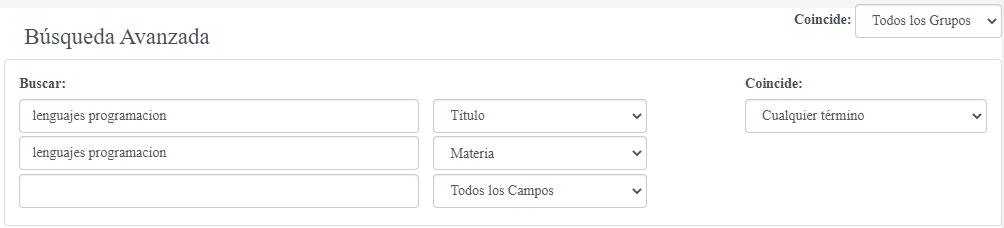
* Tesis de grado orientadas al desarrollo de software de desarrollo utilizando lenguajes y paradigmas de programación.
* La fecha de publicación en la RRAAE esté comprendida entre el año 2015 y 2020.

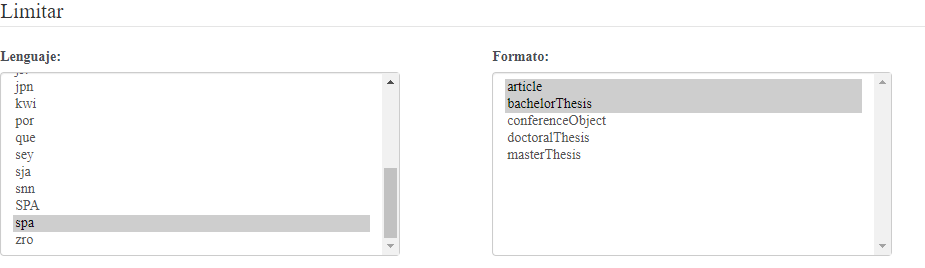
Los criterios de exclusión son los siguientes:

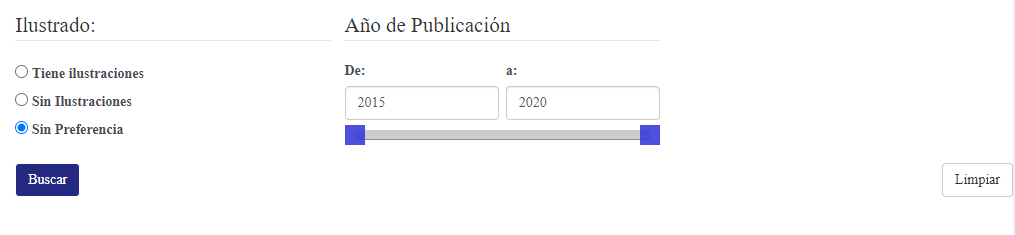
* Tesis que no contengan los términos de la búsqueda presentada en la Tabla 2.
* Tesis duplicadas del mismo trabajo en diferentes fuentes.
* Tecnologías de comunicación que no tenga autoría propia.
* Tecnologías de comunicación que no evidencie que está terminado o implementación.

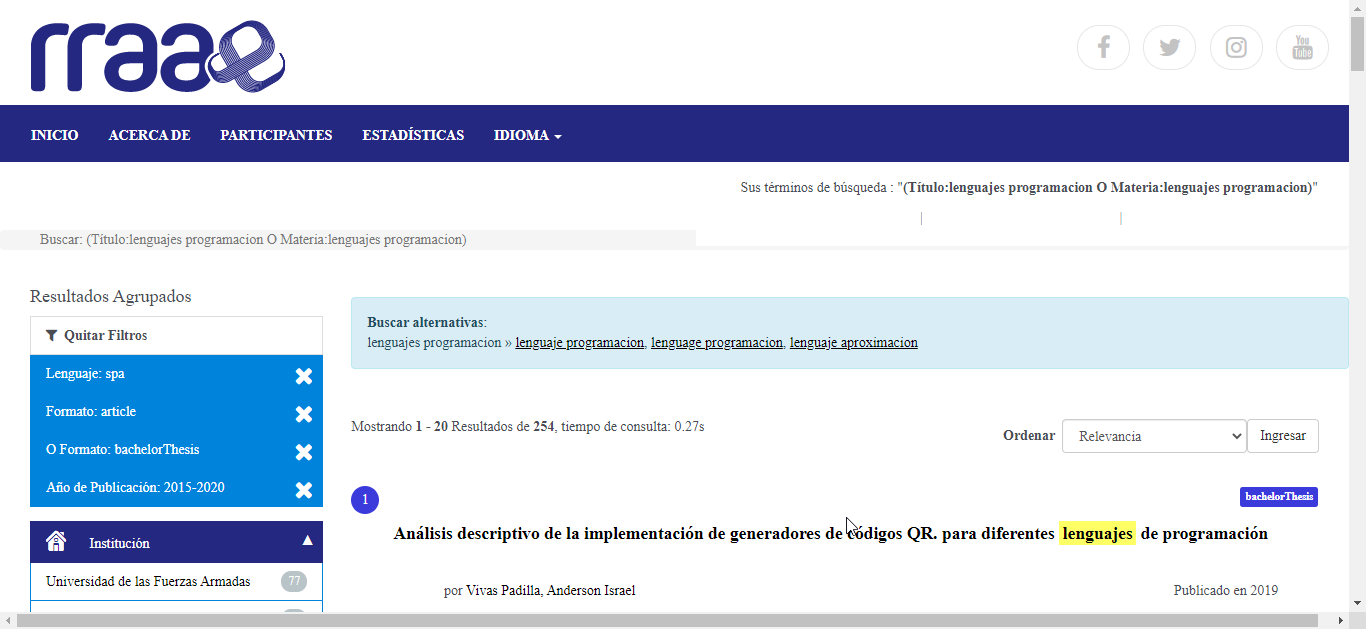
**Clasificación de estudios**

El proceso de clasificación garantiza que sólo se incluirán los trabajos relevantes sobre el desarrollo de software educativo. Para cumplir con este objetivo, este proceso define tres tareas a realizar: 1) Aplicar la estrategia de búsqueda. 2) Leer los títulos, resúmenes y palabras clavespara aplicar el criterio de inclusión. 3) Aplicar el criterio de exclusión mediante la lectura completa del trabajo que permita evidenciar que dicho software está terminado, ya sea mediante la visualización de certificados o capturas de pantalla. La Fig. 1 muestra el proceso de clasificación realizada en esta investigación.









**Estrategia de extracción de datos y síntesis**

La estrategia de extracción de datos se basa en una serie de posibles respuestas para cada pregunta de investigación. La Table 3 muestra la estrategia definida en esta investigación, en donde se define un identificador de la pregunta (primera columna) y los criterios utilizados para la delimitación de las preguntas (segunda columna).

**Tabla 3.** Estrategia de extracción de datos y síntesis utilizada

| **Id** | **Criterio** |
| --- | --- |
| 1 | C1. Año de publicación |
| C2. Universidades |
| 2 | C1. Título |
| 3 | C1. Materia |
| 4 | C1. Idioma |
| 5 | C1. Artículo |
| 6 | C1. Tesis de Grado |

**Referencias**

|  |  |
| --- | --- |
| [1] | Red Nacional de Investigación y Educación del Ecuador, «RRAAE,» [En línea]. Available: http://rraae.org.ec/. [Último acceso: 28 11 2020]. |
| [2] | R. R. Content, «¿Qué es un lenguaje de programación y qué tipos existen?,» 20 04 2019. [En línea]. Available: https://rockcontent.com/es/blog/que-es-un-lenguaje-de-programacion/. [Último acceso: 20 05 2021]. |
| [3] | 4rsoluciones, «4rsoluciones,» 22 05 2013. [En línea]. Available: https://www.4rsoluciones.com/blog/que-son-los-paradigmas-de-programacion-2/. [Último acceso: 20 06 2021]. |
| [4] | M. M. Canelo, «Profile,» 09 06 2020. [En línea]. Available: https://profile.es/blog/que-son-los-paradigmas-de-programacion/. [Último acceso: 20 06 2021]. |