Análisis del desarrollo de software en no desarrolladores

Juan Pablo Ángel Parra Arévalo

Fundación Universitaria San Mateo

Facultad de Ingeniería y Afines

Ingeniería de Sistemas

Bogotá, Colombia

Diciembre 2020

Análisis del desarrollo de software en no desarrolladores

Juan Pablo Ángel Parra Arévalo

Trabajo de investigación

Ing. William Mendoza Rodríguez

Director de investigación

Fundación Universitaria San Mateo

Facultad de Ingeniería y Afines

Ingeniería de Sistemas

Bogotá, Colombia

Diciembre 2020

Nota de aceptación

_____

_____

_____

_____

_____

_____

_____

_____

Firma Jurado

_____

_____

Firma Jurado

_____

_____

Firma jurado

_____

_____

Fecha

Índice

Lista de tablas

Dedicatoria

Este proyecto está dedicado a Dios por haber puesto a mi disposición la salud, sabiduría, la disposición y demás cuestiones necesarias para poder llevar a cabo este proyecto.

A mi familia por brindarme su apoyo, su compañía y por siempre darme consejos cuando los necesito.

A mis amigos por acompañarme durante este proceso.

A todas las personas a las que les será útil la información presente en un futuro.

Agradecimientos

Agradezco a Dios por brindarme todo lo necesario para poder llevar a cabo mis proyectos.

A mi familia por estar conmigo y siempre estar dispuestos a ayudarme.

A la Fundación Universitaria San Mateo por recibirme y brindarme los conocimientos necesarios en los temas que me gustan.

Al profesor Jhon Alexander López Fajardo, por ayudarme a forjar mi idea.

Al profesor William Mendoza Rodríguez, por acompañarme durante este proceso y ayudarme a materializar mi idea.

A los demás profesores que me han aportado conocimiento de gran ayuda para este proyecto.

A mis amigos, por brindarme apoyo durante este proceso.

Resumen

En el siguiente documento se realiza una investigación sobre la programación de software para “no desarrolladores”. Para ello se investiga sobre las soluciones existentes y opiniones sobre el tema.

También se realiza un acercamiento general a las soluciones existentes para resolver las posibles interrogantes que pueden surgir sobre las tecnologías Low-code/No-code, tales como *¿Qué impacto han tenido las tecnologías Low-code/No-code?¿Qué características deben cumplir?/¿Qué se debe buscar al momento de elegir?¿Qué alcance tienen?¿En qué situaciones conviene utiliza tecnologías Low-code/No-code y cuándo no es recomendable?*

Además, se hace un análisis sobre las posibles problemáticas que pueden retrasar o incluso anular el proceso del desarrollo de una aplicación y de qué manera las tecnologías Low-code/No-code son capaces de ofrecer una solución a este problema.

A lo largo del proceso de investigación se han encontrado diferentes recursos (reportes, noticias, opiniones, servicios, etc) que facilitaron el entendimiento de los conceptos necesarios para realizar este proyecto. Del mismo modo, los nuevos conceptos exigían el aprendizaje de más conceptos, lo que llevó a realizar un análisis detallado sobre los diferentes temas que tienen que ver con las tecnologías Low-code/ No-code para poder llegar de ese modo a una conclusión en la que se resuelven las preguntas anteriores.

Palabras Clave	
Término	**Significado**
Low-Code	Low-Code hace referencia a las tecnologías de programación de software en las que quien esté desarrollando una aplicación tendrá que escribir una parte del programa a través de código y para la otra parte no tendrá que hacerlo.
No-Code	No-Code hace referencia a las tecnologías de programación de software en las que quien esté desarrollando una aplicación no tendrá que escribir código.
Desarrollador ciudadano	El término “desarrollador ciudadano” hace referencia a los usuarios que utilizan tecnologías No-code/Low-code. Generalmente, no saben programar.
Automatización	La automatización de software hace referencia al proceso por el cual el software realiza tareas con la menor intervención humana posible.
Software	El software es un conjunto de programas, instrucciones y reglas informáticas que permiten ejecutar distintas tareas en un equipo informático.
Programación de Software	La programación o desarrollo de software es el proceso por el cual se crean las órdenes o instrucciones que realizará un determinado sistema.

Tabla 1: Palabras Clave

Abstract

In the following document, research is done on "non-developer" software programming. For this purpose, it is researched on the existing solutions and opinions on the subject.

A general approach to existing solutions is also made to solve possible questions that may arise about Low-code/No-Code technologies, such as What impact have Low-code/No-Code technologies had? What characteristics should they fulfill? What scope do they have? In what situations is it convenient to use Low-code/No-Code technologies and when is it not advisable?

In addition, an analysis is made of the possible problems that can delay or even cancel out the development process of an application and how Low-code/No-code technologies are able to offer a solution to this problem.

Throughout the research process we have found different resources (reports, news, opinions, services, etc) that facilitated the understanding of the concepts needed to carry out this project. Similarly, the new concepts required the learning of more concepts, which led to a detailed analysis of the different issues that have to do with Low-code/No-Code technologies in order to reach a conclusion that resolves the previous questions.

Keywords	
Term	**Meaning**
Low-Code	Low-Code refers to the software programming technologies in which the person who is developing an application will have to write a part of the program through code and for the other part he will not have to do it.
No-Code	No-Code refers to the software programming technologies in which the person who is developing an application will not have to write code.
Citizen developer	The term "citizen developer" refers to users who use No-code/Low-code technologies. Generally, they do not know how to program.
Software automation	Software automation refers to the process by which software performs tasks with the least possible human intervention.
Software	Software is a set of programs, instructions and computer rules that allow the execution of different tasks in a computer.
Software programming	Software programming or development is the process by which the orders or instructions that a certain system will perform are created.

Tabla 2: Keywords

Introducción

El presente proyecto plantea una investigación sobre la programación de software en personas sin conocimientos de programación, mediante soluciones “Low-code”.

A lo largo del presente documento, se evidencia la investigación realizada en temas como:

- ¿Qué es software?
- ¿Cómo se crea software?
- ¿Qué es el software para empresas?
- ¿Cómo se crea el software para empresas?
- ¿Qué son las tecnologías Low-code?
- ¿Qué son las tecnologías No-code?
- ¿Cuál es la diferencia entre Low-code y No-code?
- ¿Qué impacto han tenido las tecnologías Low-code/No-code?
- ¿Qué características deben cumplir estas tecnologías?
- ¿Qué se debe buscar al momento de elegir una solución Low-code/No-code?
- ¿Qué alcance tienen?
- ¿En qué situaciones conviene utiliza tecnologías Low-code/No-code y cuándo no es recomendable?

Para encontrar una solución a las preguntas anteriores se investigarán en diferentes fuentes de información. Luego se analizará la información obtenida y se llegará a una conclusión sobre las tecnologías Low-code/No-code.

Problema de la investigación

Planteamiento del problema de la investigación

Cuando un proyecto, sea grande, mediano, o pequeño, quiere hacerse presente frente a un público amplio, una de las mejores alternativas es a través de software (páginas web, aplicaciones móviles o aplicaciones de escritorio). Un número considerable de empresas (generalmente, las empresas más grandes) tiene los recursos (hardware, personal, capital) necesarios para desarrollar sus propios aplicativos, por lo que no necesitan ayuda de terceros, ya sea para desarrollar dichas aplicaciones o para hacer uso de hardware como servidores.

Por otra parte, existen empresas, organizaciones o fundaciones pequeñas que no cuentan con los recursos económicos, el personal o el hardware necesario para poder realizar ellos mismos sus aplicativos. Solo las organizaciones de esta segunda categoría con mayor presupuesto pueden solicitar a una empresa externa. por ejemplo, un centro de datos, para alojar información o una casa de software para poder cumplir el objetivo de tener una aplicación.

Pero, ni las empresas o fundaciones más pequeñas, que no cuentan con los recursos necesarios pueden elegir alguna de las opciones anteriores. En este punto, una de las únicas soluciones posibles para este problema consiste en que ellos desarrollen sus propios aplicativos.

A primera vista, esta opción parece que no es viable del modo “tradicional”. Por modo “tradicional”, se hace referencia a la manera de desarrollar software en el que varias personas que han estudiado temas involucrados en el desarrollo de software están trabajando para escribir todo el código necesario, alojar sus productos en servidores y realizar los diseños necesarios para que su aplicación o su página web sea agradable a la vista.

Pero, tal como se explicó anteriormente, esta opción exige recursos que la organización pequeña general mente no tiene.

Con la propuesta anterior, se piensa que cualquier persona, sepa o no programar, pueda desarrollar aplicaciones sin tener que escribir la totalidad del código (en algunos casos, no se tendrá que escribir nada de código).

No todas las personas saben programar y eso es una limitación para las personas que quieren hacer software, puesto que requiere de mucho estudio y práctica para dominar solo un lenguaje de programación.

Formulación problema de investigación

¿Cómo desarrollar aplicaciones cuando no se tienen los conocimientos en programación?

Objetivos

Objetivo general

Hacer un análisis de la programación de aplicaciones con servicios denominados “Low-code”.

Objetivos específicos

Realizar una investigación referente a las tecnologías sin código.

Realizar una investigación referente a las soluciones existentes.

Realizar una investigación referente al cómo se desarrollaría un software para tal fin.

Determinar si es posible desarrollar una aplicación sin programar y en cuáles casos no es conveniente.

Presupuesto

Presupuesto	
Descripción	Costo
Equipos electrónicos	$1.200.000,00
Internet	$291.000,00
Tiempo	$1.755.606,00
Total	$3.246.606,00

Tabla 3: Presupuesto de la investigación

Cronograma

Cronograma																
	Agosto				Septiembre				Octubre				Noviembre			
Estado del arte																
Objetivo General y Objetivos Específicos																
Problema de la investigación																
Justificación																
Antecedentes de la investigación																
Bases teóricas o fundamentos conceptuales																
Diseño metodológico																
Resultados de la investigación																
Conclusión																

Tabla 4: Cronograma

Propósito de la investigación

La investigación se realiza con el fin de aclarar temas y responder preguntas que giran en torno a las tecnologías Low-code/No-code, así como el impacto que han tenido entre organizaciones y su beneficio para personas o empresas sin recursos no solo económicos sino de conocimiento en desarrollo de software.

Justificación

El proceso de desarrollo de software en la actualidad es muy importante y ha generado un impacto considerable en la vida de muchas personas, pero resulta imposible el poder desarrollar aplicaciones por parte de las personas que carecen de los conocimientos necesarios como la lógica, el manejo de un lenguaje, la metodología, entre otros.

El presente proyecto dará respuestas a preguntas como: *¿Es posible desarrollar aplicaciones funcionales sin saber programar?¿Existen servicios que permitan desarrollar aplicaciones sin saber programar y?, en caso de que existan, ¿Cómo se debería elegir uno?¿Se pueden crear aplicativos por medio de este software, sin que el usuario tenga la necesidad de escribir código?¿En qué casos se puede desarrollar una aplicación sin ingresar código directamente?¿Cuál es el alcance de estas aplicaciones?*

Para resolver las anteriores preguntas, se realizará una investigación de publicaciones anteriores en las que se traten temas como: Aplicaciones sin código, opiniones de expertos sobre el tema, qué soluciones existen para prestar este tipo de servicios y el impacto que han tenido este tipo de soluciones, además del alcance que ofrecen. También se realizará una investigación sobre los conocimientos y habilidades necesarias para poder realizar un aplicativo de este tipo.

Antecedentes de la investigación

Las tecnologías Low-code/No-code presentan un nuevo camino a la hora de programar aplicaciones, y es gracias al impacto que han tenido las diferentes soluciones que ofrecen estos servicios la popularidad que han adquirido, llegando así a ser una opción que vale la pena considerar a la hora de desarrollar aplicaciones.

Según un reporte realizado por (Forrester, 2020), en el que se estudia el impacto que ha tenido Mendix, una plataforma para desarrollar aplicaciones sin código, se encontró que en tres años se ha obtenido una ganancia de más de 20 millones de dólares.

Las tecnologías de poco o nada de código aceleran el proceso de desarrollar software, mientras que reducen las herramientas necesarias, así como el costo de utilizarlas, sin la ayuda de tecnologías de poco código, las empresas necesitarían personal especializado y diferentes herramientas de trabajo, lo que se traduce en costos elevados. Pero con una solución de poco o nada de código, se reduce el tiempo y costos necesarios.

Según este reporte, varias de las razones por las que este tipo de servicios son usados son:

Facilidad de responder a las necesidades de un cliente, habilidad de una compañía de manejar sus propias mejoras, facilidad de acceso, facilidad de adaptabilidad y cambio, entre otras (Trejo Trejo, 2020).

También se reportó que un 45% de los participantes del estudio aceptaron que sus empresas tuvieron o tienen planes de utilizar una herramienta de poco o nada de código (Low-code/No-code).

Otro reporte realizado por (kintone, 2017) aclara que un 74% de las empresas participantes tienen debilidades en la velocidad de entrega de sus aplicaciones. Lo que se traduce en aumento de costos por el hecho de tener que mantener el proceso de desarrollo por más tiempo y reducción de ganancias, al no contar con el impulso que tendrían si se contara con su aplicativo.

El 76% de los entrevistados anuncian que una o varias de sus aplicaciones fueron desarrolladas por fuera de la empresa. Lo que da una impresión acertada de la situación de las empresas que no cuentan con personal especializado y por eso tienen que ajustarse a las tarifas de empresas externas para el desarrollo de sus aplicaciones.

Esto significa que una empresa debe invertir más si se desarrolla su aplicación por medio de un tercero.

Los “Desarrolladores ciudadanos” crean sus aplicaciones más rápido que departamentos grandes de IT. Los retos que enfrentan son la seguridad y la adquisición de habilidades de programación para el manejo de datos.

Este es uno de los aspectos más importantes de las tecnologías Low-code/No-code, ya que se definen no solo los principales beneficios, sino que también se exponen los desafíos a los que los usuarios de las tecnologías Low-code/No-code se deben enfrentar: La seguridad en sus aplicaciones y la capacidad de darles un uso adecuado par evitar problemas con el manejo de datos.

La razón por la que los desarrolladores ciudadanos trabajan de esta manera es porque sienten que los departamentos de IT son muy lentos.

Del mismo modo que el punto anterior, esto representa uno de los principales beneficios de las Tecnologías Low-Code/No-code: la reducción de tiempos. Esta reducción de tiempos se traduce en la reducción de costos.

Un tercio de las empresas apoyan a los desarrolladores ciudadanos. Aunque un cuarto de los ejecutivos no tienen habilidades de programación.

Esta es una de las limitaciones que se han impuesto a las Tecnologías Low-code/No-code: A la hora de desarrollar una aplicación estos servicios no se tienen en cuenta y, en muchos casos, se desaconsejan (probablemente debido al desconocimiento del tema).

Bases teóricas

Con base a los reportes estudiados, se espera que el consumo de tecnologías Low-code aumente exponencialmente en los próximos años. Este consumo no será solo por parte de los usuarios conocidos como “desarrolladores ciudadanos”, sino también por parte de pequeñas y medianas empresas, que se verán beneficiadas por las características y facilidades que las tecnologías Low-code/No-code ofrecen, entre las cuales, las dos más importantes son:

Reducción de costos: No es necesario pagar por personal ni hardware especializado.

Facilidad de uso: No hay que tener conceptos mayores de programación, por lo que, en teoría, cualquier persona con acceso a tecnologías Low-code es capaz de crear su propia aplicación.

Estos beneficios han creado la popularidad que las tecnologías sin código tienen entre los desarrolladores ciudadanos, fundaciones y empresas.

Esta es una forma de pensar que, según Nicolás Verdejo, escritor de (WWWhat’snew, 2020), ha pasado de ser en una herramienta a convertirse en una filosofía, en un movimiento que acerca cada vez a más personas a la creación de software, sin los requisitos de tener conocimientos extensos sobre los diferentes temas que conlleva la creación de software.

Los únicos requisitos que deberían cumplir el usuario, según Nicolás, son: tener una buena idea y contar con un plan para ponerla en práctica. De esta manera, las tecnologías Low-code/No-code pueden garantizar que los usuarios no tengan que

pensar en temas complejos a la hora de llevar a cabo la creación de software, con solo saber qué quieren hacer debe ser suficiente para poder materializar su idea.

Esta es la idea principal de la programación de software mediante tecnologías Low-code/No-code, la facilidad de uso. Esta facilidad ha animado a todo tipo de clientes (usuarios comunes, negocios pequeños, fundaciones, y empresas) a utilizar estas tecnologías, aumentando de este modo la popularidad.

El éxito de las tecnologías Low-code/No-code ha generado un impacto tan grande que, para el 2023, se espera que un 50% de empresas de todo tipo estén desarrollando sus aplicaciones por medio de tecnologías Low-code/No-code, según un reporte de (Newgen, 2020). De esta manera, se espera que el uso de tecnologías Low-code/No-code aumente exponencialmente, obteniendo así un papel fundamental en el mercado del desarrollo de software y entre las empresas que hacen uso de ellas.

Pero así como las tecnologías Low-code/No-code solucionan muchas de las problemáticas de quienes hacen uso de ellas, también han tenido que afrontar desafíos que, en el caso de solucionarse, podrán conseguir más usuarios y, de este modo, obtener una mejor reputación.

Estos desafíos definen el porqué una organización acudiría a uno de estos servicios.

Uno de los principales proveedores de servicios Low-code, (Appian, s.f.), establece las características con las que todas las plataformas de desarrollo Low-code/No-code deberían contar:

Modelado visual: Los procesos deberían ser representados por herramientas gráficas, para que a los desarrolladores ciudadanos les resulte más sencillo entender.

Interfaces de “arrastrar y soltar”: Escribir fragmentos enteros de código no solo es complejo, sino que también requiere de más tiempo. Con interfaces en las que los usuarios solo tengan que arrastrar y soltar se ahorra tiempo y dinero, a la vez que es más sencillo de entender y realizar.

Movilidad instantánea: Esta característica es muy representativa del lenguaje de programación Java, *Write once, run anywhere (WORA) (s.a., 2002)*, que especifica que se debería programar una aplicación una sola vez para poder ejecutarse en cualquier sistema operativo. Este principio también se debería poder aplicar a las tecnologías Low-code/No-code, un usuario debería diseñar su aplicación una sola vez y debería poder ejecutar esas aplicaciones de manera nativa en los diferentes sistemas operativos y dispositivos.

Herramientas de declaración: A través de modelos visuales o reglas de negocios se elimina la necesidad de escribir código específico. De esta manera, también se reducen los costos y el tiempo requerido par completar una aplicación.

Seguridad: La seguridad de la información es un tema que no solo se reduce a esto servicios, sino que debería estar presente en todos los servicios de nuestro entorno: La seguridad y la integridad de la información.

Este es el motivo principal por el que un gran número de usuarios deciden si van a utilizar un determinado servicio o no, ya que en la mayoría de los casos las empresas

estarían confiando información sensible de sus usuarios que debe ser protegida para que no se vean comprometidos los datos de los usuarios.

Si en algún caso la empresa que utiliza tecnologías Low-code/No-code no resguarda información de sus clientes, la información que se debe proteger es la de la empresa que utiliza estos servicios.

En cualquiera de los dos casos, uno de los escenarios más probables es que se tenga guardada información sensible como medios de pago, por lo que es primordial proteger esta información.

Escalabilidad: Al igual que la mayoría de proyectos, e incluso el mismo Low-code/No-code, el software suele comenzar como un proyecto pequeño, con un alcance reducido y con un impacto pequeño. Del mismo modo, los proyectos de las empresas actuales generalmente inician como un proyecto pequeño pero, conforme avanza el tiempo, el software debe ser más grande y atender cada vez más peticiones, lo que implica que tanto el software de las empresas, como las mismas tecnologías Low-code/No-code, estén en constante actualización y crecimiento.

A estas características, se pueden agregar la noción de los desafíos que enfrentan las tecnologías Low-code/No-code, desafíos que pueden ser cruciales a la hora de elegir un servicio.

El primer desafío que enfrentan las tecnologías Low-code/No-code es el costo de utilizar estos servicios. En muchos casos las fundaciones, empresas pequeñas y medianas y todo tipo de organizaciones acuden a las tecnologías Low-code/No-code porque no tienen los recursos necesarios para producir ellos mismos su software.

De este modo, las tecnologías Low-code/No-code y quienes las ofrecen deben tener en cuenta que sus usuarios no siempre contarán con los recursos necesarios, por lo que lo ideal sería contar con diferentes planes para que los posibles clientes puedan costear estos servicios.

El segundo desafío que enfrentan las tecnologías Low-code/No-code es la facilidad de uso.

Uno de los casos más comunes por el que los usuarios utilizan estas tecnologías es por que ellos no cuentan con los conocimientos necesarios para desarrollar por sí mismos sus aplicaciones.

Es por eso que las tecnologías Low-code/No-code deben contar con herramientas o mecanismos que faciliten la creación de software por parte de usuarios sin conocimientos de programación. Estos mecanismos pueden ser de diversos tipos, tales como:

Manuales de usuario: Recursos escritos o audiovisuales en los que se den soluciones a las preguntas más frecuentes entre los usuarios.

Atención al cliente: En caso de que un usuario presente un problema que no cuente con una solución en los manuales de usuario, se debe poder contar con un equipo que atienda las peticiones de los usuarios, para dar así una solución pronta a los problemas presentados.

Con todos los desafíos que las tecnologías Low-code/No-code deben enfrentar se están aumentando exponencialmente las posibilidades de que un servicio sea elegido por sobre la competencia.

Son estas características las que definen el porqué una organización acudiría a uno de estos servicios. Ya que el principal objetivo de la programación sin código es la facilidad. Según (Laurence hart, 2019), las tecnologías Low-code/No-code se encargan de un 80% del trabajo necesario, dejando una parte pequeña al usuario. De lo anterior podemos deducir que la idea de que las tecnologías low-code/no-code realizan todo el trabajo es incorrecta. Esto se debe a que, independientemente de los servicios que sea capaz de ofrecer, el usuario tendrá que diseñar y materializar su idea, es decir dar órdenes al software. Por esta razón se llama PROGRAMACIÓN sin código, ya que el usuario está dando órdenes y pasos a seguir a un software, lo que, según la (RAE, s.f) se define como programar.

Con respecto a las tecnologías Low-code/No-code pueden surgir diferentes preguntas:

¿Cómo elegir un servicio Low-code/No-code?

Una de las ventajas de las tecnologías Low-code/No-code, es la cantidad de servicios que hay para elegir pero, visto de otro modo, la cantidad de opciones puede abrumar a los usuarios. Así que, para elegir un servicio, se debe primero consultar cuáles son las posibilidades.

Existen páginas como noncoders (Pawan Kumar, s.f.), que contienen todo tipo de recursos sobre tecnologías Low-code/No-code, estos recursos se encuentran en constante actualización y organizados en las siguientes categorías: “Freelancers”, “Entrepreneurs”, “Designers”, “Product Managers”, “Employees”, “Sales”, “Marketers”,

“Remote Teams”, “Podcasters”, “Bloggers”, “Web App”, “Mobile App”, “Online Store”, “Fund Your Work”

Una vez, elegida la categoría, la página mostrará los resultados organizados, al seleccionar uno de ellos, se redirecciona a la página principal del servicio, en la cuál se podrá encontrar información como precios, tipo de servicios ofrecidos, alcance y política de privacidad. Analizando la información obtenida, se puede elegir el servicio que mejor se ajuste, tanto a las necesidades del usuario, como a su presupuesto.

También existen servicios que ofrecen versiones de prueba gratuitas, para que los usuarios comprueben la calidad del servicio y así les resulte más sencillo decidirse por un servicio u otro.

¿Cuáles son las diferencias entre programación “tradicional” y programación sin código?

Es posible encontrar una solución a esta pregunta volviendo al ejemplo de la empresa que decidió utilizar Low-code/No-code para su software.

Si la empresa hubiese optado por desarrollar una aplicación, por ejemplo, con el lenguaje de programación Java, significa que cuenta con el personal y el presupuesto necesarios.

Para empezar, se necesita realizar un levantamiento de información, para saber cuáles son las necesidades y qué es lo que quiere el cliente. Luego, se debe realizar un análisis de requerimientos para determinar los requisitos funcionales y no funcionales.

Para la etapa de diseño, se realizan los modelos necesarios para documentar cómo sera el software a realizar.

Al momento de empezar a programar se debe tener claro qué tipo de metodología de desarrollo se va a llevar a cabo (Cascada, espiral, SCRUM…).

Para realizar, por ejemplo, una interfaz gráfica en Java, se debe escribir la totalidad del código para el aspecto visual y el comportamiento.

El software puede contener errores, por lo que es necesaria una etapa de Testing o pruebas de software que se lleve a cabo constantemente para garantizar que se puede avanzar o en caso contrario, volver para corregir errores.

Este es un proceso largo, que requiere de personal, mucho tiempo y dinero solo para desarrollar una aplicación. Si se desea una página web o una base de datos, el proceso es cada vez más largo. Pero si la empresa opta por una plataforma Low-code/ No-code, el proceso es más sencillo, pues se puede evadir la realización de varios de los pasos descritos anteriormente, por ejemplo, para desarrollar una interfaz gráfica, no es necesario escribir código, pues se cuentan con herramientas gráficas denominadas “drag-and-drop” (arrastrar y soltar) que, como su nombre lo indica, consiste en “acomodar” los controles que se desean, evitando de esta manera, la tarea de programar.

¿Cuál es la diferencia entre Low-code y No-code?

El escritor de Outsystems, (Chris Souther, 2019), aclara que, aunque para muchas personas, son bastante similares, pero tienen diferencias:

Por un lado, Low-code es realmente similar a un IDE (Integrated Development Enviroment), en el que se cuentan con herramientas para facilitar el trabajo de los programadores. Se trata de una herramienta en la que el usuario arrastra y suelta bloques de código ya existentes.

Según Chris, las principales ventajas del Low-code son:

Velocidad: se pueden escribir varias aplicaciones simultáneamente y mostrar ejemplos en días o, incluso horas.

Más recursos: no es necesario esperar a que los desarrolladores terminen sus otros proyectos, disminuyendo el tiempo y, de esta forma, los gastos.

Riesgo bajo: todas las tareas de seguridad ya están integradas y listas para configurar e implementar, por lo que el riesgo es menor.

Despliegue rápido: las aplicaciones desarrolladas con Low-code, se realizan en menos tiempo, funcionan como se espera y, si se presenta un comportamiento inesperado, se puede volver con un clic.

Por la otra parte, No-code está orientado a desarrolladores ciudadanos que, generalmente, no conocen alguno de los diferentes lenguajes de programación existentes, y no es necesario, ya que las herramientas No-code traen integradas todas las funcionalidades que un desarrollador ciudadano puede necesitar. Ejemplos de No-

code son las plataformas de Blog o de comercio electrónico, que permiten a los usuarios desarrollar sus aplicaciones en poco tiempo.

¿Cuáles son las ventajas y desventajas de las tecnologías Low-code/No-code?

Entre las ventajas se encuentran, según (Joe Stangarone, 2019):

Creación de aplicaciones dentro de una empresa: Existen ocasiones en las que las aplicaciones de las empresas son desarrolladas por fuera de dicha empresa. Con tecnologías Low-code/No-code es más sencillo desarrollar aplicaciones dentro de una empresa.

Tiempos de entrega: Con tecnologías Low-code/No-code las aplicaciones se desarrollan más rápido, ya que eliminan la etapa de código en la mayoría de ocasiones y reducen las pruebas necesarias.

Entre las desventajas, se encuentran;

Personalización: La capacidad de personalizar una aplicación con Low-code/No-code varía de plataforma en plataforma, en algunos casos las opciones de personalización son más reducidas en algunas plataformas que en otras.

Bloqueo de los vendedores: Existen proveedores que generan código que es fácil de mantener por fuera de la plataforma e incluso podría ser mantenido en una plataforma externa. Por otra parte, también hay proveedores que limitan el mantenimiento de una aplicación a una plataforma, lo que dificulta el proceso de desarrollar una aplicación.

Método

Hasta ahora, se han desglosado las principales temáticas relacionadas a las problemáticas de una empresa y las principales temáticas relacionadas a las tecnologías Low-code/No-code.

Con toda la información obtenida se pueden establecer las causas y los efectos de el impacto que han tenido las tecnologías Low-code/No-code. Este tipo de método es, según (Ricardo Cannan, s.f.), un método analítico, en el que se extrae la información desde lo abstracto hacia lo concreto, para poder así, encontrar causas y consecuencias.

Entre las causas de las problemáticas de las empresas se encuentran:

La carencia de recursos para desarrollar sus aplicaciones. Esta carencia es, a su vez, ocasionada por diversos factores que varían con cada empresa.

El tamaño de la empresa, es muy poco probable que las empresas más grandes sufran las mismas problemáticas que las empresas pequeñas.

El impacto de la empresa, uno de los principales factores que definen el crecimiento de la empresa tiene que ver con los servicios que prestan. Por ejemplo, con las empresas que prestan servicios relacionados a la salud es poco probable ver que dejen de ser utilizadas un día.

Si ponemos el ejemplo de una empresa que se ve afectada por las problemáticas anteriores, se encuentra con un candidato prefecto a uno de los usuarios más comunes de las tecnologías Low-code/No-code: Empresas pequeñas que no tienen suficiente presupuesto para realizar sus propias aplicaciones.

Por parte de las tecnologías Low-code/No-code encontramos los servicios que ofrecen, así como las características que deben cumplir y los desafíos que enfrentan.

Entre más empresas cumplan con las características anteriores, mayor será el impacto ocasionado por las tecnologías Low-code/No-code que, a su vez, beneficiará más a las empresas que hagan uso de ellas. De este modo, se hace cada vez más notorio el crecimiento de ambas partes.

Resultados

Con todo lo anterior en mente podemos deducir que las tecnologías Low-code/No-code se enfrentan a un aumento exponencial por lo que, los proveedores de servicios de poco o nada de código deben estar preparados para la llegada en masa de nuevos clientes.

Estos resultados han sido posibles gracias a las empresas que invirtieron en estas tecnologías. Estas empresas son de todo tipo y de todos los tamaños, fundaciones, empresas pequeñas, medianas e incluso grandes empresas han contribuido a que los proveedores de servicios Low-code/No-code tengan el impacto que han tenido en los últimos años.

También se sabe que, debido a la preferencia de las empresas por las tecnologías Low-code/No-code han creado una idea errónea en varias personas: Legará el momento en que la programación de software de la manera “tradicional” se quede obsoleta.

Nicolás Verdejo, sostiene que la programación y el código nunca desaparecerán, ya que son el corazón que hace que funcionen un gran número de dispositivos y operaciones, por lo que, aunque las tecnologías Low-code/No-code están presentando un crecimiento exponencial, sin el software, no sería posible que existiera.

Conclusión

A lo largo de este proceso de investigación, he encontrado bastantes recursos sobre este tema que parecía tan desconocido, pero que en realidad lleva mucho tiempo desarrollándose y tomando un papel primordial en el mercado y para muchas empresas.

También me di cuenta del impacto que ha tenido y de las posibilidades que ofrecen, no solo por empresas, sino para todo tipo de personas que quieran acercarse a la programación de software sin la necesidad de saber programar.

Al mismo tiempo aprendí que el éxito de las Tecnologías Low-code/No-code no excluye el éxito de la programación tradicional, por lo que no es necesario preocuparse por la idea de que los desarrolladores de software se quedarán sin trabajo.

Incluso, existirán ocasiones en las que, por diferentes factores que se pueden presentar, la carga de un desarrollador se vea reducida gracias a la programación sin código.

Referencias

Anthony Abdulla. (2019). Low-Code Platforms Bring Diversity to Applications. 2020, octubre, de CMS Wire Recuperado de https://www.cmswire.com/digital-workplace/low-code-platforms-bring-diversity-to-applications/

Appian. (s.f.). Low-code Development Platform Features | Low-code Basics.. 2020, Noviembre, de Appian Recuperado de https://www.appian.com/low-code-basics/features/

Chris Southern. (2019, Agosto). Low-Code vs. No-Code: What's the Real Difference?. 2020, Noviembre, de Out Systems Recuperado de https://www.outsystems.com/blog/posts/low-code-vs-no-code/

Forrester. (2020). The Total Economic Impact™ Of The Mendix Low-Code Application Development Platform". 2020, septiembre, de Mendix Recuperado de https://www.mendix.com/resources/forrester-total-economic-impact-of-the-mendix-platform/

Joe Stangarone. (2019, Marzo). Pros and cons of low-code development platforms. 2020, Noviembre, de MRC Productivity Recuperado de https://www.mrc-productivity.com/blog/2019/03/pros-and-cons-of-low-code-development-platforms/

Kintone. (2017). THE RISE OF THE EMPOWERED CITIZEN DEVELOPER. 2020, septiembre, de Kintone Recuperado de https://resources.kintone.com/citizen-developer-business-application-report-2017

Laurence Hart. (2019). Is Low-Code Technology Right For You?,. 2020, Nomviembre, de CMS Wire Recuperado de https://www.cmswire.com/information-management/is-low-code-technology-right-for-you/

Newgen. (2020). Magic Quadrant for Enterprise Low-Code Application Platforms. 2020, octubre, de Newgen Recuperado de https://newgensoft.com/resources/analyst-report-magic-quadrant-for-enterprise-low-code-application-platforms/

Nick Langley. (2002). Write once, run anywhere?. 2020, Noviembre, de Computer Weekly Recuperado de https://www.computerweekly.com/feature/Write-once-run-anywhere

Nicolás Verdejo. (2020, Octubre). ¿Qué es el No-Code y por qué debería interesarte? 2020, Noviembre, de WWWHat's new. Recuperado de https://wwwhatsnew.com/2020/10/13/que-es-el-no-code-y-por-que-deberia-interesarte/

Pawan Kumar. (s.f.). Non-coders, A Curated library of tools and resources for non-coders! Noviembre 2020, de Non-coders Recuperado de https://noncoders.club/

Real Academia Española. (s.f.). Programar | Definición. 2020, Noviembre, de Real Academia Española Recuperado de https://dle.rae.es/programar

Ricardo Cannan. (s.f.). Los 8 Tipos de Métodos de Investigación Más Habituales,. 2020, Noviembre, de Lifeder Recuperado de https://www.lifeder.com/tipos-metodos-de-investigacion/

Trejo Trejo, M. (2020). Arte, diseño y mercancía: Operaciones de la imagen en la sociedad industrial y posindustrial. Designio, 2(1), 9–23. https://doi.org/10.52948/ds.v2i1.102