Revista de investigación de sistemas e informática
14(2): 49-66 (2021)
DOI: https://doi.org/10.15381/risi.v14i2.23148

ISSN: 1815-0268 (Impreso) / ISSN: 1816-3823 (Electrónico)
Facultad de Ingeniería de Sistemas e Informática - UNMSM

Framework para la automatización del maquetado de páginas web aplicando procesamiento de lenguaje natural

Framework for automating the layout of web pages applying natural language processing

Julio Prudencio-Nieves
https://orcid.org/0000-0002-2416-9121
prudencio11@hotmail.com

Universidad Nacional Mayor de San Marcos,
Facultad de Ingeniería de Sistemas e
Informática. Lima, Perú

Recibido: 08/01/2022 - Aceptado: 15/02/2022 - Publicado: 28/02/2022

RESUMEN

El maquetado web está en constante evolución y cada vez se incorporan nuevos profesionales en la materia. No obstante, las herramientas y los entornos de desarrollo parecen haberse quedado rezagados a lo tradicional: simple uso del teclado para codificar y poca motivación e innovación en el uso de otros medios de codificación, como la voz, que permitan mejorar y ampliar su uso. Por ello, se ha desarrollado un framework web moderno e innovador que aplica la teoría de procesamiento de lenguaje natural para automatizar el maquetado web mediante el uso de las tecnologías CSS 3 y reconocimiento de voz. El framework ha sido validado mediante la aplicación de un cuestionario conformado por 6 dimensiones y 31 indicadores valorizados del 1 al 5 en función a la escala de Likert y para la obtención de los resultados se ha utilizado la técnica estadística denominada medidas de correlación. Los resultados demuestran que el framework mejora en un: 38.59% el maquetado web en general, 41.46% el maquetado de las animaciones, 39.22% el maquetado de los botones, 32.33% el maquetado del contenido, 39.94% el maquetado del fondo, 41.04% el maquetado de las imágenes y 39.28% el maquetado de la tipografía.

Palabras clave: Diseño de página web; Interfaces de usuario; Procesamiento de lenguaje natural; Reconocimiento automático del habla; Interacción humano computador.

ABSTRACT

Web layout is constantly evolving and new professionals in the field are joining each time. However, the development tools and environments seem to have lagged behind the traditional: simple use of the keyboard to code and little motivation and innovation in the use of other means of coding, such as voice, to improve and expand its use. For this reason, a modern and innovative web framework has been developed that applies natural language processing theory to automate web layout through the use of CSS 3 and speech recognition technologies. The framework has been validated through the application of a questionnaire made up of 6 dimensions and 31 indicators valued from 1 to 5 according to the Likert scale and to obtain the results, the statistical technique called correlation measures has been used. The results show that the framework improves by: 38.59% the web layout in general, 41.46% the layout of the animations, 39.22% the layout of the buttons, 32.33% the layout of the content, 39.94% the layout of the background, 41.04% the layout of the images and 39.28% the layout of the typograph.

Keywords: Web page design; User interfaces; Natural language processing; Automatic speech recognition; Human computer interaction.

I. INTRODUCCIÓN

Las páginas web permiten mostrar contenido estático, contenido que no es modificado por el usuario final, pero muestra información detallada sobre algún tópico en particular. Por otro lado, los sistemas web sí permiten la interacción entre el usuario final y el sistema con fines de presentación o generación de nueva información sobre algún tema en particular: académico, social, familiar, salud, negocio, gobierno, entre otros (Nagilla, 2012). Una de las características importantes de las páginas o sistemas web es el diseño estético, también llamado diseño o maquetado web, el cual es la expresión artística que dota de atractivo a la web, causando en los usuarios finales emociones diversas y consolidando el propósito de su construcción (Jarrar, 2002).

Existen muchos entornos de programación que permiten realizar el maquetado web, todos ellos consolidados y con muchos años en el mercado (Pilamunga, 2012). Sin embargo, todos ellos requieren del uso de un teclado de computadora para desarrollar las líneas de código necesarias para su funcionamiento (Mihalcin, 2007). Son tradicionales y no motivan a los profesionales o futuros profesionales, practicantes o curiosos a utilizar y expandir más sus conocimientos en diseño web. Pese a que el diseño web está en constante evolución y cada vez más se adhieren nuevos profesionales en la materia, las herramientas y entornos de desarrollo parecen haberse quedado rezagadas a lo tradicional: simple uso del teclado para codificar y poca motivación e innovación en el uso de otros medios de codificación, como la voz, que permitan mejorar y expandir su uso, enseñanza y aprendizaje (Delgado, Sandoval, & Arteaga, 2018).

Por tal motivo, el framework web propuesto aplica la teoría de procesamiento de lenguaje natural sobre el maquetado web, para su automatización, mediante el uso de las tecnologías CSS 3 y reconocimiento de voz. CSS 3 es la tecnología que nos permite maquetar páginas o sistemas desplegados en internet de forma profesional en un muy corto período de tiempo, sin la necesidad de escribir extensas líneas de código. La sintaxis básica se compone de un selector seguido de una o más declaraciones. Sin embargo, esta tecnología también admite el uso de reglas-AT lineales y anidadas, así como el uso de descriptores y comentarios contenidos y organizados dentro de una hoja de estilo (MDN Web Docs, 2021).

Una de las aplicaciones del procesamiento de lenguaje natural es la tecnología de reconocimiento de voz que se encarga de procesar y convertir una expresión hablada, captada como audio, a formato texto. Para ello se ha usado la Web Speech API (Roig, 2019). Esta API es gestionada mediante el lenguaje de programación JavaScript a través de las interfaces de control: SpeechRecognition y SpeechRecognitionEvent, con las propiedades: continuos, lang, interimResults, maxAlternatives y los métodos: onresult, onerror y onend los cuales permiten implementar el reconocimiento de voz en entornos web (MDN Web Dcos, 2021).

La presente investigación tiene como objetivo diseñar un framework para la automatización del maquetado de páginas web aplicando procesamiento de lenguaje natural con la finalidad de permitir el uso de comandos de voz CSS 3 dentro de un entorno web intuitivo, fácil, moderno e innovador que permite gestionar hojas de estilo mediante el uso del mouse y el teclado, así como de la voz, lo que representa una innovación del método tradicional de diseño web.

Este artículo se organiza de la siguiente manera: en la sección 1 se expone el problema de estudio, la sección 2 detalla la revisión de la literatura, la sección 3 expone la metodología de investigación, la sección 4 presente la Interface MFML: Modelo físico - Modelo lógico el cual es la base para la construcción del framework propuesto y se detallan los procesos y actividades de su construcción, la sección 5 detalla el procedimiento seguido para validar el framework y se muestran los resultados obtenidos, la sección 6 hace una discusión sobre los resultados, en la sección 7 se sugieren trabajos futuros por los que se puede extender la presente investigación y finalmente en la sección 8 se presenta las conclusiones.

II. REVISIÓN DE LA LITERATURA

En el trabajo de investigación sobre un framework web visual para HTML, CSS y JavaScript de apoyo a la docencia (Jaimez & Vargas, 2017) se demostró que el framework desarrollado posibilita la construcción de páginas web de manera gráfica y genera código HTML, CSS y JavaScript de forma automática, para cada uno de los lenguajes mencionados, separándolos en tres archivos diferentes. Además, mediante la selección de elementos HTML, aplicación de estilos CSS y validaciones JavaScript, se facilitó al estudiante la concentración en la estructura y contenido de un sitio web y su diseño gráfico, los cuales son cubiertos en los procesos de enseñanza – aprendizaje.

La investigación bibliográfica y de campo cuyo objetivo fue la determinación del nivel de incidencia en

la optimización de un sitio web al realizar el maquetado de sus páginas a base de scripts y hojas de estilo en cascada (CSS) (Pilamunga, 2012) obtuvo resultados indicando que CSS era la herramienta menos utilizada con un 9% de uso frente a un 78% de Dreamweaver y un 99% de Flash. No obstante, se demostró con un 100% que el uso de estas herramientas incrementa innecesariamente la cantidad de imágenes del sitio y hace que el tamaño total del sitio sea demasiado grande. Además, se demostró que la principal razón para no utilizar CSS fue que el total de los participantes tenía conocimientos insuficientes o nulos sobre dicha tecnología.

Por otro lado, el trabajo de investigación cuantitativo, cuyo objetivo principal fue el diseño e implementación de un Agente animado interactivo para informar sobre el calendario académico de la Universidad de Córdoba mediante reconocimiento de voz basado en un entorno web (Muñoz & Rodriguez, 2015), menciona en sus conclusiones que el aporte principal del proyecto ha sido el diseño y desarrollo de un agente animado capacitado para brindar información a partir de una aplicación multimedia con la funcionalidad de reconocimiento y respuesta de voz, permitiendo al usuario una mayor flexibilidad en la interacción hombre - máquina, facilitando así a los estudiantes estar informados sobre los cambios realizados de una manera rápida y confiable.

La investigación aplicada, perteneciente al área de investigación de la Ingeniería de Software que sigue la línea de investigación en Ingeniería en Tecnología Web, describe el diseño de un asistente de voz basado en Web para personas con visión subnormal (Collado & Aparicio, 2017) mediante el uso de Tecnologías de Información y Comunicación que permite incluirlos al mundo de la tecnología orientada en la búsqueda de información por medio de la web. Los resultados demuestran un tiempo de carga de 26.3 s con una disponibilidad del 99% alojado en servidores de GoDaddy, de tal manera que el 63.3% de encuestados indican que se sienten satisfechos con la interacción que tienen con la aplicación, el 100% indicó que era necesaria una capacitación previa en el uso de la aplicación y el 66.7% que resultó fácil interactuar con la aplicación.

La investigación exploratoria llevada a cabo con 10 participantes para evaluar el entorno de programación guiado por voz (Delgado, Sandoval, & Arteaga, 2018), demuestra el uso del editor Blockly elegido por ser de código fuente libre y de uso gratuito haciendo propicia su adaptación para cumplir con los criterios siguientes: el reconocimiento de voz, el análisis sintáctico, la generación de código y la integración con el IDE propuesto. Las conclusiones demuestran que el estudio fue realizado con participantes con diferente experiencia en programación y se evaluó la factibilidad técnica de la herramienta para la elaboración de pequeños programas.

En el trabajo de investigación cuyo objetivo fue crear una aplicación multimedia para el entrenamiento en la certificación TOEFL mediante reconocimiento de voz (Hernadez & Lemuz, 2016), se describe la generación de voz artificial a partir de un texto haciendo uso de JSAPI, que es el conjunto de clases e interfaces Java para brindar al programador la facilidad de escribir aplicaciones de voz. La aplicación cuenta con las secciones: Reading, Listening, Writing y Speaking para los que tiene un apartado de learn y otro de practice.

Finalmente, en el trabajo de investigación cuyo objetivo fue crear una silla de ruedas controlada mediante reconocimiento de comandos de voz en español (Gil, Castillo, & Flórez, 2016), los cuales se encontraron contenidos dentro de un vocabulario reducido compuesto por palabras reservadas, se usó la tecnología de Interfaz de Programación de Aplicaciones de Voz (SAPI) de Microsoft, con System.Speech.Recognition como espacio de nombres, para reconocer las palabras reservadas del vocabulario, las cuales estuvieron relacionadas con las funcionalidades ofrecidas en cuanto a desplazamiento de izquierda a derecha o de adelante y atrás, domótica y salud en cuanto a la medición de presión, temperatura, pulso y oxígeno.

III. MATERIAL Y MÉTODOS

En el desarrollo del presente estudio se aplicó la metodología de investigación siguiente.

3.1 Tipo y diseño de investigación

La investigación realizada fue de tipo correlacional debido al estudio de la influencia del framework basado en procesamiento de lenguaje natural (variable independiente) sobre la automatización del maquetado de páginas web (variable dependiente). El diseño es de carácter pre-experimental (Murillo, 2013) pues se realizó una medición de la variable dependiente (automatización del maquetado de páginas web) antes (pre-test) y después (post-test) del diseño e implementación del framework web CSS 3 con reconocimiento de voz. Se consideró como unidad de análisis a toda aquella persona dedicada al diseño o maquetado de páginas web.

3.2 Población y muestra

La población de estudio estuvo conformada por todas aquellas instituciones públicas o privadas dedicadas al diseño de páginas web en general, sin distinción de su ubicación, tamaño o finanza. La muestra, en este caso, estuvo conformada por una institución pública peruana, la cual fue la Universidad Nacional Santiago Antúnez de Mayolo (UNASAM). Dentro de esta institución se eligió particularmente a los alumnos del noveno ciclo de la Escuela de Ing. de Sistemas e Informática, los cuales fueron un grupo de quince estudiantes dedicados al diseño web.

La muestra en esta investigación fue no probabilística y se utilizó la técnica de muestreo por conveniencia (Hernandez & Carpio, 2019). Esta decisión resultó realmente obvia al considerar el gran tamaño de la población que abarcó instituciones que se encuentran fuera del alcance del investigador y por ende no fueron posibles de evaluar. Considerar a la UNASAM como institución de muestra, por ser una universidad conocida por el investigador y convenientemente disponible para este trabajo de investigación, facilitó ampliamente su desarrollo.

3.3 Técnica de recolección de datos

La técnica de recolección de datos aplicada ha sido un cuestionario, el cual representó el instrumento de recolección de datos, conformado por 6 dimensiones y 31 indicadores de maquetado web, mostrado en el Apéndice A. Los 31 indicadores de maquetado web del cuestionario se presentan en forma de pregunta cerrada con 5 alternativas categorizadas del 1 al 5 según la escala de Likert (ver Tabla 1), siendo posible la elección de sólo una de ellas por indicador.

Tabla 1

Escala de Likert utilizada en el cuestionario.

Escala	Respuesta
1	Muy Deficiente
2	Deficiente
3	Regular
4	Eficiente
5	Muy Eficiente

Fuente: Elaboración propia

Los indicadores de maquetado web sirven como directrices altamente recomendadas durante el proceso de maquetado para asegurar su aceptación por parte de los usuarios finales (Szigeti, 2012). La obtención de los indicadores que componen el cuestionario inició con la búsqueda de investigaciones referente a ellos. Durante la actividad de búsqueda encontramos dos investigaciones resaltantes. El primer trabajo se titula: *Web Design Guidelines for WSDM* (Jarrar, 2002), que propone la aplicación de un conjunto de indicadores como aporte a la *fase de implementación del diseño*, la cual forma parte de la metodología de diseño: WSDM (De Troyer, 2001). El segundo trabajo de investigación es titulado: *Research-Based Web Design & Usability Guidelines* (U.S. Department of Health and Human Services, 2006), que establece una *metodología completa de diseño web* abarcando las fases de análisis, maquetado propiamente y pruebas de usabilidad.

Una vez obtenidos los indicadores que forman parte del cuestionario, proseguimos con su evaluación mediante el análisis de validez por juicio de expertos en diseño web, a quienes se les envió el instrumento mencionado. Estos expertos en diseño web fueron elegidos mediante un muestreo no probabilístico usando la técnica de muestreo por conveniencia (Hernandez & Carpio, 2019) y fueron contactados mediante correo electrónico. Luego de que cada experto nos hiciera llegar sus respuestas, procedimos a aplicar la técnica estadística V de Aiken (Aiken, 1985) y, el resultado del procesamiento estadístico fue un coeficiente V de 0.8 de concordancia entre jueces (ver Tabla 2), un nivel en el cual se puede concluir que los expertos en diseño web coinciden en la validez y consistencia del instrumento de recolección de datos (Escurra, 1988).

Tabla 2

Análisis de validez por juicio de expertos.

Jueces	Acuerdo	IA	PB	V	P
5	4	0.8	0.156	0.8	0.032

Fuente: Elaboración propia

IV. FRAMEWORK PROPUESTO

Para hacer viable la integración entre el maquetado web usando la tecnología CSS 3 y el reconocimiento de voz a través de la Web Speech API, que nos permita maquetar páginas o sistemas web haciendo uso de la voz humana mediante comandos de voz en español, programados y reconocidos por el framework para gestionar gráficamente las hojas de estilo y sus elementos, se ha diseñado la Interface MFML: Modelo físico – Modelo lógico. Esta interface desarrolla un modelo matemático para gestionar la inserción, actualización y eliminación de los

elementos CSS y una teoría de nomenclaturas que permite gestionar las clases e IDs de los elementos fila, columna y textarea de una tabla html la cual representa a dichos elementos CSS.

En la construcción del framework web CSS 3 con reconocimiento de voz se siguieron las actividades descritas por la metodología ágil Scrum la cual nos ha permitido realizar un desarrollo iterativo e incremental generando valor y alta funcionalidad a través de entregas progresivas y revisiones técnicas formales obviando la documentación excesiva e innecesaria (Palacio, 2020). Con la finalidad de comprender mejor las funciones, actividades, componentes y relaciones entre los elementos del framework a construir se ha elaborado un diagrama de casos de uso. La arquitectura general del framework consta de un diagrama de paquetes, un diagrama de componentes y un diagrama de despliegue.

La codificación del framework web CSS 3 con reconocimiento de voz se realizó usando las tecnologías de programación web: CSS 3, HTML 5 y JavaScript que son lenguajes de programación ampliamente difundidos y conocidos internacionalmente; el API Web Speech ha sido gestionado a través de JavaScript. Finalizada la etapa de codificación y pruebas se subió el editor a un servidor en la nube que tiene instalado: Apache como servidor web y PHP como lenguaje de programación del lado del servidor, permitiendo el acceso del usuario final en cualquier momento y desde cualquier ubicación y dispositivo.

4.1 Interface MFML: Modelo físico – Modelo lógico

Al analizar la estructura de las hojas de estilo y su proceso de construcción normal notamos que este proceso se rige por las operaciones de inserción, eliminación y actualización de los elementos: selectores, declaraciones, reglas-AT, descriptores y comentarios teniendo en cuenta la estructura siguiente: las declaraciones son contenidas por los selectores y los descriptores son contenidos por las reglas-AT, además, una regla-AT puede ser de tipo lineal o anidada.

En respuesta a la necesidad de gestionar el proceso de construcción y la estructura de cualquier hoja de estilo CSS se ha desarrollado un modelo matemático para automatizar estas operaciones de construcción y una teoría de nomenclaturas para administrar la estructura de la hoja de estilo, las cuales son desarrolladas a continuación.

4.1.1 Modelo matemático: el modelo matemático desarrollado permite automatizar las operaciones de manipulación de los elementos CSS dentro de una hoja de estilos: inserción, eliminación y actualización. Estas operaciones posibles de realizar sobre los elementos CSS modifican la estructura de las hojas de estilo y siguen un patrón de desarrollo incremental en cuanto al número de elementos. Esto es, cada elemento contenido en dichas hojas se define por dos características: cantidad y nivel de anidamiento.

Resulta apropiado entonces representar la estructura de la hoja de estilo CSS 3 en el plano cartesiano asumiendo al eje X como la cantidad de reglas-AT anidadas padre o la cantidad de selectores o comentarios y, al eje Y, como el nivel de anidamiento o número de descendientes de una regla-AT anidada padre. La unidad (01) se interpretará como el nivel inicial de anidamiento de las reglas-AT anidadas a partir del cual se empezarán a anidar los elementos contenidos y como el nivel único de profundidad de los elementos que no contienen descendientes en el caso de reglas-AT lineales, selectores o comentarios.

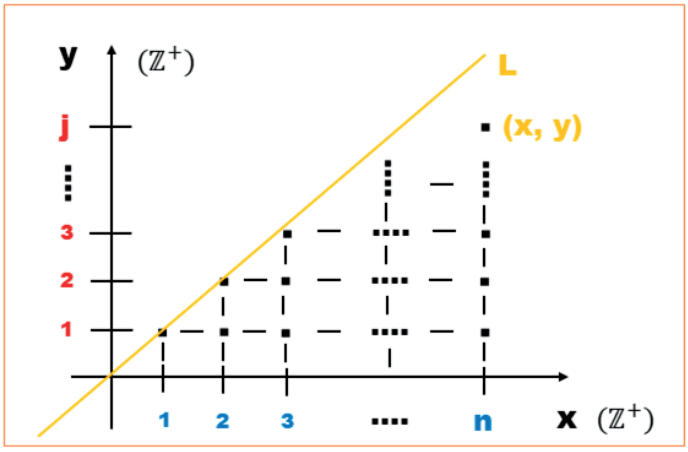
La recta diagonal “L” que pasa por el punto (0, 0) del plano representa la cantidad infinita de elementos descendientes soportados por el modelo matemático, de tal manera que para localizar y administrar un elemento del modelo sólo será necesario especificar el valor de la variable “n” que representa el n-ésimo elemento dentro de la hoja de estilo y especificar el valor de la variable “j” que representa el j-ésimo nivel de anidamiento de dicho elemento. Es decir, indicar las coordenadas (x, y) (ver Figura 1).

La **operación insertar u operación de inserción** permite insertar una nueva regla-AT: $f(ins\ AT)$, en el mismo nivel j-ésimo de la regla-AT desde donde fue invocada la operación, incrementando el número de reglas-AT en dicho nivel, o insertar la nueva regla-AT en el siguiente nivel j-ésimo de la regla-AT desde donde fue invocada, provocando un incremento del nivel de anidamiento. Estos AT reciben el nombre de AT hermano: $AT\binom{j}{(n+1)}$ o AT hijo: $AT\binom{(j+1)}{n=1}$, respectivamente.

La inserción de un nuevo descriptor: $f(ins\ Ds)$, involucra la adición de su propiedad (Ps) en el mismo nivel j-ésimo de la n-ésima regla-AT desde donde fue invocada la operación, provocando el incremento del número de descriptores: $AT\binom{j}{n}\ Ps(h+1)$ e involucra el mismo procedimiento para su valor (Vs): $AT\binom{j}{n}\ Vs(h+1)$.

Figura 1

Interface MFML – Modelo matemático.



Donde:

X: Número de reglas-AT anidadas padre, es decir, reglas-AT de nivel 1

y: Nivel de anidamiento, es decir, el número de descendientes de una regla-AT anidada padre

L: Recta definida por puntos (X, Y) que representan la n-ésima regla-AT de nivel j-ésimo

$\mathbb{Z}^+$: Los valores n-ésimos y j-ésimos pertenecen a enteros positivos

Fuente: Elaboración propia

La inserción de un nuevo selector: $f(ins\ S)$, inserta el selector en el mismo nivel j-ésimo de la n-ésima regla-AT desde donde fue invocada la operación, incrementando el número de selectores anidados en dicho nivel: $AT\binom{j}{n}\ S(m+1)$, o inserta el selector en el primer nivel provocando un incremento del número de selectores de primer nivel: $S(m+1)$, si fue invocada desde una declaración.

La inserción de una nueva declaración: $f(ins\ D)$, perteneciente a un selector anidado, involucra la adición de su propiedad (P) dentro del m-ésimo selector anidado perteneciente a la n-ésima regla-AT de nivel j-ésimo desde donde fue invocada la operación, incrementando el número de declaraciones: $AT\binom{j}{n}\ S(m+1)\ P(k+1)$, e involucra el mismo procedimiento para su valor (V): $AT\binom{j}{n}\ S(m+1)\ V(k+1)$.

A su vez, la inserción de una nueva declaración: $f(ins\ D)$, perteneciente a un selector no anidado, involucra la adición de su propiedad (P) dentro del m-ésimo selector desde donde fue invocada la operación, incrementando el número de declaraciones: $S(m+1)\ P(k+1)$, e involucra el mismo procedimiento para su valor (V): $S(m+1)\ V(k+1)$.

La **operación eliminar u operación de eliminación** permite quitar una regla-AT: $f(del\ AT)$, lo que implica el borrado descendente de todas las reglas-AT hijas y sus respectivos descriptores, selectores y declaraciones, a partir del j-ésimo nivel de la regla-AT padre eliminada desde donde fue invocada la operación, disminuyendo también la numeración de las reglas-AT hermanas y sus respectivas reglas-AT hijas junto a sus propios elementos descriptores, selectores y declaraciones en forma descendente, a partir de la regla-AT padre eliminada.

La eliminación de un descriptor: $f(del\ Ds)$, implica el borrado del descriptor desde donde fue invocada la operación e implica disminuir la numeración de todos los descriptores hermanos que se encuentran en el mismo nivel j-ésimo a partir del descriptor eliminado.

La eliminación de un selector: $f(del\ S)$, implica el borrado descendente de todas las declaraciones pertenecientes al selector eliminado desde donde fue invocada la operación, e implica disminuir la numeración de los selectores hermanos que se encuentran en el mismo nivel j-ésimo, a partir del

selector eliminado, y de sus respectivas declaraciones en forma descendente.

La eliminación de una declaración: $f(del\ D)$, implica el borrado de la declaración desde donde fue invocada la operación, e implica disminuir la numeración de todas las declaraciones hermanas a partir de la declaración eliminada.

La **operación actualizar u operación de actualización** se ejecuta de forma automática inmediatamente después de finalizada la operación de eliminación. Esta operación permite actualizar una regla-AT: $f(upd\ AT)$, conllevando a realizar tres acciones: disminuir la numeración de reglas-AT hermanas a partir de la n-ésima regla-AT eliminada desde donde fue invocada la operación, reduciendo el número de reglas-AT en dicho nivel: $_{AT}\binom{j}{n'}$; disminuir la numeración de los descriptores hijos de cada una de las reglas-AT actualizadas en la acción anterior: $_{AT}\binom{j}{n'}(Ds(h))$; disminuir la numeración de los selectores hijos de cada una de las reglas-AT actualizadas en la primera acción: $_{AT}\binom{j}{n'}(S(m))$.

La actualización de un descriptor: $f(upd\ Ds)$, implica disminuir la numeración de cada propiedad (Ps) perteneciente a cada descriptor hermano a partir del descriptor eliminado desde donde fue invocada la operación, reduciendo el número de propiedades: $Ps(h')$, e involucra el mismo procedimiento para su valor (Vs): Vs(h').

La actualización de un selector: $f(upd\ S)$, conlleva a disminuir la numeración de los selectores en el mismo nivel j-ésimo a partir del selector eliminado desde donde fue invocada la operación, reduciendo el número de selectores en dicho nivel: $S(m')$ y conlleva a actualizar el número de cada una de las declaraciones hijas de los selectores actualizados: $S(m')\ (D(k))$.

La actualización de una declaración: $f(upd\ D)$, conlleva a disminuir la numeración de cada propiedad (P) perteneciente a cada declaracion hermana, a partir de la declaracion eliminada desde donde fue invocada la operación, reduciendo el número de propiedades: $P(k')$, e involucra el mismo procedimiento para su valor (V): $V\ (k')$.

4.1.2 Teoría de nomenclaturas: permite administrar el orden de inserción, el número y la jerarquía de cualquier elemento dentro de una hoja de estilo CSS. A diferencia del modelo matemático que automatiza las operaciones de manipulación de los elementos CSS dentro de una hoja de estilos, la teoría de nomenclaturas administra estas operaciones estableciendo: un mecanismo de identificación, un orden de inserción, un número correlativo según tipo y una jerarquía o nivel de anidamiento a cada elemento dentro de la hoja de estilo. Para ello considera el uso de una tabla html tradicional la cual por definición se compone de filas dentro de las cuales se encuentran las columnas y, estas a su vez, contienen elementos llamados textarea los cuales son elementos de entrada, es decir, elementos donde el usuario puede ingresar contenido el cual puede tener un formato de: texto, número, carácter o una combinación de estos en caso se requiera. Una fila contendrá como máximo dos columnas y una columna contendrá un solo elemento textarea. El uso de una tabla html como representación de la estructura de una hoja de estilo resulta conveniente pues resulta práctico administrar sus elementos y gestionar sus operaciones.

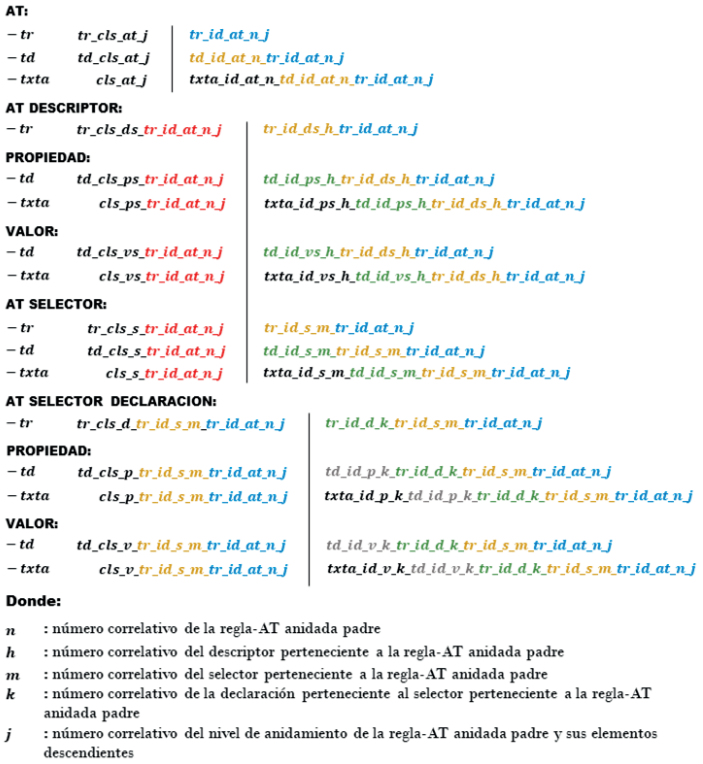
El mecanismo de identificación está compuesto de ocho (08) abreviaciones que representan a los elementos CSS: “at” para reglas-AT, “ds” para descriptores, “ps” para propiedades de descriptores, “vs” para valores de descriptores, “s” para selectores, “d” para declaraciones, “p” para propiedades de declaraciones y “v” para valores de declaraciones. Además, está compuesto de tres abreviaciones (03) para los elementos de la tabla html: “tr” para filas, “td” para columnas y “txta” para elementos textarea. Finalmente se establecen cuatro (04) variables globales numéricas correlativas: “n” para reglas-AT, “h” para descriptores, “m” para selectores y “k” para declaraciones.

La **teoría de nomenclaturas para reglas-AT lineales** establece la formación y asignación de clases e IDs CSS para cada fila, columna y elemento textarea que representa un determinado elemento dentro de la hoja de estilo, teniendo a la unidad (01) como único nivel de anidamiento; es decir, las reglas-AT no tienen reglas-AT descendientes, pero sí contienen descriptores, selectores o declaraciones y sus respectivas propiedades y valores. Es sumamente importante tener presente el orden entre filas, columnas y elementos textarea, las que son contenidas unas dentro de otras, respectivamente, pues son considerados al momento de formar y asignar dichas clases e ID.

La **teoría de nomenclaturas para reglas-AT anidadas** es una generalización de la teoría anterior. Esta teoría pone especial énfasis en la gestión del nivel de anidamiento y en la longitud de los caracteres de los IDs, formados en función de las clases asignadas a cada fila, columna y textarea. Distinguiéndose entre reglas-AT anidadas padre e hijas, donde una regla-AT puede cumplir el doble rol de ser padre e hija simultáneamente (ver Figura 2).

Figura 2

Nomenclatura regla-AT anidada (j > 1).



Fuente: Elaboración propia

La variable global de anidamiento "j" cumple el rol principal de representar el nivel correlativo de anidamiento en el que se encuentra un determinado elemento CSS dentro de una regla-AT anidada padre. La longitud de los caracteres que componen un ID es variable, pero se determina según la cantidad de números a agregar al final de la abreviación general del ID de la fila de cualquier regla-AT: "tr_at_id". Esta cantidad de números se determina por la función: f (longitud ID) = [(nuevo nivel j-ésimo) x (2)], unidos por guiones bajos (_), donde el primer número representa la cantidad correlativa de reglas-AT, el segundo número representa el nivel j-ésimo en la que se encuentra y, la cantidad de números restantes, representan los números del elemento AT padre del cual desciende.

Como se explicó anteriormente, al eliminar una regla-AT se eliminan también todos sus elementos descendientes sin excepción. Luego se ejecuta automáticamente la operación de actualización de las reglas-AT hermanas, a partir de la regla-AT eliminada, y de cada uno de sus elementos descendientes. Para ubicar el carácter numérico que representa el correlativo a actualizar se aplica el concepto de tupla. Una tupla es la agrupación por pares de los caracteres numéricos que conforman el ID de un elemento. Luego de formar la tupla, se procede a identificar el nivel del elemento a actualizar y el nivel de la regla-AT eliminada para luego aplicar la función: f (tupla actualizar) = [(nivel elemento hijo + 1) – (nivel padre eliminado)], cuyo resultado es el número de tupla de la cual el primer carácter numérico será actualizado (se disminuirá en uno el valor que tenga al momento de ser calculado), pues representa el correlativo de la regla-AT padre del cual desciende.

La **teoría de nomenclaturas para selectores lineales** rige cuando un selector no desciende de ninguna regla-AT, pero sigue los criterios de formación y asignación de clases e IDs para los elementos fila, columna y textarea de una tabla html, al igual que el de las teorías anteriores. El ID de la columna contienen el ID de la fila que la contiene y los elementos textarea contienen, a su vez, el ID de la columna que los contiene. Así, por ejemplo, el ID de un elemento textarea que representa a la propiedad o valor de una declaración se forma uniendo el ID de la columna de dicha propiedad o valor, el ID de la fila de la declaración al cual pertenece la propiedad o valor y el ID de la fila del selector del cual desciende

la declaración, resultando el ID: “txta_id_propiedad_n_td_id_propiedad_n_tr_id_declaracion_n_tr_id_selector_n”. La variable “n” indica el número correlativo de selectores lineales y es calculada sumando uno (01) a la cantidad de filas de aquellos selectores que tengan asignada la clase: “tr_cls_selector”.

La **teoría de nomenclaturas para comentarios** sigue los criterios de formación y asignación de clases e IDs a los elementos fila, columna y textarea de una tabla html, al igual que el de las teorías anteriores. La característica clave es que los comentarios son considerados como elementos flotantes que no forman parte ni contienen ningún otro elemento. El ID del textarea se forma uniendo el ID de la columna que lo contiene y el ID de la fila que contiene a su vez a la columna: “txta_id_comentario_n_td_id_comentario_n_tr_id_comentario_n”. La variable “n” representan el correlativo de comentarios y se calcula sumando uno (01) a la cantidad de filas que tengan asignada la clase: “tr_cls_comentario”.

4.1.3 Interface MFML: permite usar la voz para desarrollar código CSS 3 y automatizar la generación de hojas de estilo, entendiéndose como tal al hecho de vincular y ejecutar comandos de voz, como mecanismo de programación, que permiten construir hojas de estilo dentro de un entorno web usando los elementos fila, columna y textarea de una tabla html. Por otro lado, la interface MFML se concibe también con el propósito de: permitir la transformación de la tabla html en un archivo .css apto para ser incluido directamente en la carpeta del proyecto de maquetado web y, a su vez, permitir la conversión de dicho archivo en la misma tabla html donde fue construido (ver Figura 3).

El **modelo físico** permite el uso de los comandos de voz los cuales son expresiones habladas mediante el cual se interactúa con los elementos de la tabla html dentro del framework web CSS 3. Haciendo uso de la Web Speech API, estas expresiones son enviadas a un servicio web de reconocimiento de voz y son devueltas en formato texto.

Figura. 3

Interface MFML: Modelo físico – Modelo lógico.

MODELO FÍSICO (*tabla html*)	INTERFACE MFML (*Clase, ID, Valor, Nivel*)	MODELO LÓGICO (*archivos .css /.xml*)
(1° selector)	S1	S1 {
(2° declaración)	P1: V1	P1: V1;
		}
(3° regla-AT anidada nivel 1)	RA-1_1	RA-1_1 {
(4° regla-AT anidada nivel j)	RA-1_j	RA-1_j {
(5° selector regla-AT)	S_1_1_j	S_1_1_j {
(6° declaración selector regla-AT)	P_1_1_j: V_1_1_j	P_1_1_j: V_1_1_j;
(7° declaración selector regla-AT)	P_2_1_j: V_2_1_j	P_2_1_j: V_2_1_j;
		}
		}
		}
(8° selector regla-AT anidada nivel 1)	S_1_1	S_1_1 {
(9° declaración selector regla-AT)	P_1_1_1: V_1_1_1	P_1_1_1: V_1_1_1;
(10° declaración selector regla-AT)	P_2_1_1: V_2_1_1	P_2_1_1: V_2_1_1;
		}
(11° regla-AT anidada nivel 1)	RA-2_1	RA-2_1 {
(12° declaración descriptor regla-AT)	P_1_2_1: V_1_2_1	P_1_2_1: V_1_2_1
(13° declaración descriptor regla-AT)	P_2_2_1: V_2_2_1	P_2_2_1: V_2_2_1
		}
(14° comentario)	C1	/* C1 */
(15° comentario)	C2	/* C2 */

Donde:

# *	: número de fila de la tabla html
Clase	: clase CSS asignada a cada elemento html de la tabla (fila, columna y textarea)
ID	: identificador asignado a cada elemento html de la tabla (fila, columna y textarea)
Valor	: contenido del elemento html textarea representando el valor de cada elemento CSS
Nivel	: nivel de cada elemento CSS dentro de la tabla html

Fuente: Elaboración propia

Una vez recibido el texto este es evaluado internamente mediante el uso de expresiones regulares para determinar si es un comando reconocido y proceder a llevar a cabo la instrucción hablada, correspondiente a dicho comando de voz.

El **modelo lógico** está compuesto de la hoja de estilo CSS 3 y del archivo XML el cual contiene los datos siguientes: número de fila, nombre del elemento, clase e ID de la fila, clase e ID de la columna, clase e ID del elemento textarea, contenido o valor del elemento textarea y el nivel de anidamiento. Es gracias al archivo xml, que la interface MFML, puede reconstruir la tabla html para continuar desarrollando código CSS 3 dentro del framework propuesto.

El archivo .css generado y exportado cumple con todas las normas sintácticas, semánticas y estructurales establecidas por la tecnología CSS 3 (MDN Web Docs, 2021). Esto incluye: añadir el carácter dos puntos entre las propiedades y los valores de los descriptores y declaraciones, añadir el carácter punto y coma después del valor de un descriptor o declaración, añadir las llaves de inicio y fin que delimitan los bloques de código CSS definiendo reglas-AT o selectores, añadir los caracteres barra diagonal y asterisco para el inicio y cierre de comentarios, añadir las sangrías entre los elementos padre e hijo según el nivel de anidamiento.

El archivo .xml generado y exportado es la metadata del archivo .css. El nombre que recibe es el mismo que el de la hoja de estilo. Este archivo xml define la etiqueta global: “<modeloLogicoCSSXML>”, que encapsula todo el contenido del documento. Dentro de la etiqueta anterior se encuentra la etiqueta: “<arrayMFML>” y dentro de ésta se encontrarán las etiquetas: “<elementoMFML>” cuya cantidad será igual a la cantidad de elementos CSS que contenga la tabla html. Dentro de la etiqueta anterior se encontrarán las etiquetas: “<fila>”, “<elemento>”, “<tr_clase>”, “<tr_id>”, “<td_clase>”, “<td_id>”, “<txta_clase>”, “<txta_id>”, “<contenido>” y “<nivel>”, las cuales representan el número de fila, elemento, clase, ID, valor y nivel de cada elemento CSS contenido dentro de la tabla html, los cuales hacen posible su reconstrucción posterior dentro del framework.

4.2 Comandos de voz

La interface MFML, haciendo uso de la Web Speech API, hace posible el uso de la voz para generar código CSS. Para que esto sea posible se emplean los comandos de voz, los cuales son frases pronunciadas para provocar una respuesta. Cada comando de voz tiene dos (02) formas de ser pronunciado. La primera es llamada la forma completa y consiste en pronunciar la palabra “comando” seguido de la “acción a realizar”, la segunda es llamada la forma abreviada y consiste en pronunciar la palabra “comando” seguido de su abreviación representada por el “número de orden” en que se encuentra el comando ordenado alfabéticamente de forma ascendente.

Al momento de diseñar cada comando de voz se puso especial énfasis en la facilidad de pronunciación e identificación por parte del servicio de reconocimiento de voz. Resulta evidente entonces que: la gestión de la tecnología de reconocimiento de voz, es realizada por las siete (07) categorías en las que se agrupan los comandos de voz.

4.2.1 Categorías de los comandos de voz: las siete categorías (07) de los comandos de voz son: comandos propiamente y modifican la estructura del modelo físico y gestionan el contenido de sus elementos; comandos auxiliares que permiten insertar letras del abecedario, vocales y caracteres; comandos de texto que permiten insertar cualquier número, palabra o frase literalmente; comandos de selector que permiten insertar selectores de tipo, de pseudoclase, de pseudoelemento, de clase, de ID, de reglas-AT page, de reglas-AT keyframes, de reglas-AT font-feature-values; comandos de declaración que permiten insertar propiedades y sus valores respectivos; comandos de reglas-AT que permiten insertar reglas-AT lineales y anidadas; comandos de descriptores que permiten insertar propiedades y sus valores respectivos, pertenecientes a reglas-AT anidadas.

Los **comandos propiamente** modifican la estructura del modelo físico y gestionan el contenido de sus elementos. La modificación de la estructura se refiere a añadir o eliminar elementos CSS como son reglas-AT, descriptores, selectores, declaraciones y comentarios. La gestión del contenido se refiere a las operaciones de desplazamiento y edición de contenido dentro del modelo físico (tabla html). Estos comandos son: insertar selector, insertar declaración, insertar at&t, insertar at&t hermano, insertar comentario, eliminar “elemento”, abajo, arriba, atrás, avanzar, borrar, comentario, copiar, cortar, espacio, final, inicio, pegar, regla at&t “número”, seleccionar, seleccionar todo, selector “número”.

Los **comandos auxiliares** permiten insertar letras del abecedario, vocales y caracteres. Las letras del abecedario posibles de ser insertadas son: s, d, m, n, t. Esto debido a que por facilidad de pronunciación e interpretación del servicio de reconocimiento de voz

de la Web Speech API no es necesario generar comandos para el resto de las letras del abecedario, tan sólo se usará la categoría: comando texto. Por otro lado, los caracteres posibles de ser insertados son: acento circunflejo, ampersand, asterisco, coma, comilla simple, comillas, corchete fin, corchete inicio, diagonal, diagonal invertida, dólar, dos puntos, guion, guion bajo, igual, llave fin, llave inicio, mas, mayor que, menor que, numeral, palote, parentesis fin, parentesis inicio, porcentaje, punto, punto y coma, virgulilla.

Los **comandos de texto** permiten insertar cualquier número, palabra o frase, libre y literalmente. Es decir, este comando es usado en casos donde se requiera insertar contenido alternativo, útil en el caso de comentarios, u otros motivos que se consideren apropiados.

Los **comandos de selector** permiten insertar ciento once (111) elementos html pertenecientes al selector de tipo, cincuenta y cinco (55) selectores de pseudoclase pronunciando la palabra “clase” seguido del selector, once (11) selectores de pseudoelemento pronunciando la palabra “elemento” seguido del selector, selectores de clase pronunciando la palabra “class” seguido del selector, selectores de ID pronunciando la palabra “id” seguido del selector, diecisiete (17) selectores de reglas-AT page pronunciando la palabra “pagina” seguido del nombre del selector, selectores de reglas-AT keyframes pronunciando la palabra “marco principal” seguido de la palabra “from”, “to” o “valores porcentuales”, seis (06) selectores de reglas-AT font-feature-values pronunciando la palabra “valor letra” seguido del nombre del selector.

Los **comandos de declaración** permiten insertar propiedades pronunciando la palabra “propiedad” seguido del nombre de la propiedad y valores pronunciando la palabra “valor” seguido del nombre del valor.

Los **comandos de reglas-AT** permiten insertar reglas-AT lineales y anidadas. Algunas reglas-AT requieren de condiciones lógicas, tipos de media queries, características de media queries, pseudoclases o identificadores asignados a criterio por el usuario seguidos de los nombres de dichas reglas-AT. Entonces para insertar las reglas-AT lineales: charset, import y namespace pronunciamos la palabra “at&t lineal” seguido del nombre de la regla-AT. Para insertar las reglas-AT anidadas: media, document, page, font-face, viewport, counter-style, keyframes, supports y font-feature-values pronunciamos la palabra “at&t anidada” seguido del nombre de la regla-AT.

Para insertar la definición “url” de la regla-AT lineal import pronunciamos la palabra “importar” seguido de la palabra “url”, para insertar los tipos: all, aural, braille, handheld, print, projection, screen, tty, tv, embossed y speech de la regla-AT anidada media, pronunciamos la palabra “tipo” seguido del nombre del tipo, para insertar las cuarenta y tres (43) características de la regla-AT anidada media pronunciamos la palabra “caracteristica” seguido del nombre de la caracteristica, para insertar las condiciones: or, not y only de la regla-AT anidada media pronunciamos la palabra “condicion” seguido del nombre de la condición, para insertar la definición: “url”, de la regla-AT lineal namespace pronunciamos la palabra “espacio nombre” seguido de la palabra “url”, para insertar las definiciones: url, url-prefix, domain y regex de la regla-AT anidada document pronunciamos la palabra “documento” seguido del nombre de la definición, para insertar la definición: blank, first, left y right de la regla-AT anidada page pronunciamos la palabra “pagina” seguido del nombre de la definición.

Los **comandos de descriptores** permiten insertar las propiedades: margin, padding, border y background de la regla-AT anidada page pronunciando la palabra “pagina” seguido del nombre de la propiedad, para insertar las propiedades de la regla-AT anidada font-face pronunciamos la palabra “letra” seguida del nombre de la propiedad, para insertar las propiedades de la regla-AT anidada viewport pronunciamos la palabra “ventana” seguida del nombre de la propiedad, para insertar las propiedades de la regla-AT anidada counter-style pronunciamos la palabra “contador estilo” seguida del nombre de la propiedad.

Habiendo expuesto las bases teóricas representadas por la Interface MFML y los comandos de voz se procederá a describir el proceso seguido para el análisis, diseño y construcción del framework web CSS 3 con reconocimiento de voz.

4.3 Metodología de desarrollo

Definiremos los roles establecidos por la metodología de desarrollo ágil Scrum, posteriormente se detallará el artefacto: estimación de la pila del producto.

***4.3.1 Roles de scrum*:** el rol propietario del producto (product owner), desarrollador (developer) y scrum master han sido cumplidos por el autor de esta investigación.

***4.3.2 Estimación de la pila del producto*:** se usó el método planing pocker para asignar valores a cada

historia de usuario clasificándolos como: XtraSmall (XS) con un valor de 01 como punto de historia y 1/2 día como tiempo de trabajo; Small (S) con un valor de 02 como punto de historia y 01 día como tiempo de trabajo; Medium (M) con un valor de 03 como punto de historia y 02 días como tiempo de trabajo; Large (L) con un valor de 05 como punto de historia y 03 días como tiempo de trabajo; XtraLarge (XL) con un valor de 08 como punto de historia y 05 días como tiempo de trabajo [22]. La estimación de la pila del producto se ha realizado siguiendo un orden de prioridad de implementación (ver Tabla 3).

***4.3.3 Planificación del primer sprint*:** el sprint es el núcleo fundamental de Scrum, en torno al que se organizan todas las demás actividades. A veces llamado iteración, representa a cada etapa de trabajo con un objetivo concreto dentro del proyecto y tiene una duración fija y constante (Palacio, 2020).

En el presente trabajo de investigación, el primer sprint engloba las siguientes cuatro historias de usuario: (HU-01) Cargar archivo html con un tiempo de duración de 02 días, (HU-02) Importar archivo xml con una duración de 02 días, (HU-03) Crear archivo css con un tiempo de duración de 01 día y (HU-04) Pronunciar comandos de voz con una duración de 05 días. El esfuerzo estimado para completar el primer sprint es de 16 puntos de historia con una duración de desarrollo de 10 días.

***4.3.4 Planificación del segundo sprint*:** el sprint es el núcleo fundamental de Scrum, en torno al que se organizan todas las demás actividades. A veces llamado iteración, representa a cada etapa de trabajo con un objetivo concreto dentro del proyecto y tiene una duración fija y constante (Palacio, 2020).

En el presente trabajo de investigación, el segundo sprint engloba las siguientes cuatro historias de usuario: (HU-05) Ocultar editor con un tiempo de duración de 03 días, (HU-06) Mostrar editor con una duración de 01 día, (HU-07) Exportar archivos .xml y .css con un tiempo de duración de 03 días y (HU-08) Descargar código fuente con una duración de 03 días. El esfuerzo estimado para completar el primer sprint es de 17 puntos de historia con una duración de desarrollo de 10 días.

***4.3.5 Arquitectura general del framework web CSS3*:** está compuesta por el diagrama de paquetes que es el diagrama estructural del UML y representa la forma estática de los componentes del sistema (No Magic, Inc., 2010). Así, el editor ha sido desarrollado siguiendo el patrón de arquitectura de software MVC donde el **controlador** está conformado por los archivos **js**; el **modelo** está conformado por los archivos **json** que contienen los comandos de voz, el archivo **php** que exporta y descarga los archivos .css / .xml generados los cuales también forman parte del modelo; la **vista** está conformada por los archivos **html** y **css**.

El diagrama de componentes describe componentes software y sus dependencias donde un componente es una parte del software encapsulada, reutilizable, reemplazable, tiene mayor responsabilidad que una clase e interactúa con otros a través de interfaces requeridas y proveídas (No Magic, Inc., 2010). El diagrama de componentes del editor consta de 05 componentes: **inicialización** con 10 operaciones proveídas a través de la interface “cargaArchivo”, **pronunciación** con 15 operaciones proveídas a través de la interface “comandoVoz”, **descarga** con 01 operación proveída a través de la interface “descargaArchivo”, **operación** con 08

Tabla 3
Scrum – Estimación de la pila del producto.

Sprint	HU	Descripción	Tamaño	Puntos de historia	Tiempo (días)
Primer sprint	HU-01	Cargar archivo html	M	3	2
	HU-02	Importar archivo xml	M	3	2
	HU-03	Crear archivo css	S	2	1
	HU-04	Pronunciar comandos de voz	XL	8	5
Segundo sprint	HU-05	Ocultar editor	L	5	3
	HU-06	Mostrar editor	S	2	1
	HU-07	Exportar archivos .xml y .css	L	5	3
	HU-08	Descargar código fuente	L	5	3
Puntos de historia / Tiempo estimado				**33**	20

Fuente: Elaboración propia

operaciones proveídas a través de las interfaces "previsualización" y "exportaMFML", **índex** con 01 operación proveída a través de la interface "verificaDispositivo" y ésta además requiere de todas las interfaces proveídas por los componentes antes mencionados.

El diagrama de despliegue representa la arquitectura en tiempo de ejecución de procesadores, dispositivos (nodos) y de los artefactos software que se ejecutan en esa arquitectura (No Magic, Inc., 2010). El diagrama de despliegue del editor consta de 01 dispositivo llamado **cliente** y de 02 procesadores llamados **servidor web** y **servidor de aplicaciones** donde están instalados: Apache Web Server v.2.4.48 y PHP Application Server v.7.2.34, respectivamente. Además, en el servidor web se encuentran desplegados: 25 artefactos de los cuales 10 son archivos **js**, 09 son archivos **html**, 05 son archivos **json** y 01 archivo **comprimido**. En el servidor de aplicaciones se encuentra desplegado 01 archivo **php**.

La funcionalidad global del editor y su interacción con el diseñador se muestra en el diagrama de casos de uso (ver Figura 4) donde se constatan 14 casos de uso, de los cuales 06 de ellos son casos de uso incluidos. Sus nombres son claramente representativos e identifican a todas las funciones realizadas por el editor, siendo estos: Ocultar editor, Mostrar editor, Exportar archivos xml - css, Descargar código fuente, Cargar html, Importar xml, Crear css, Pronunciar comandos de voz, Gestionar edición texto, Gestionar comentario css, Gestionar selector css, Gestionar declaración css, Gestionar reglaAT css, Gestionar descriptor css.

V. RESULTADOS

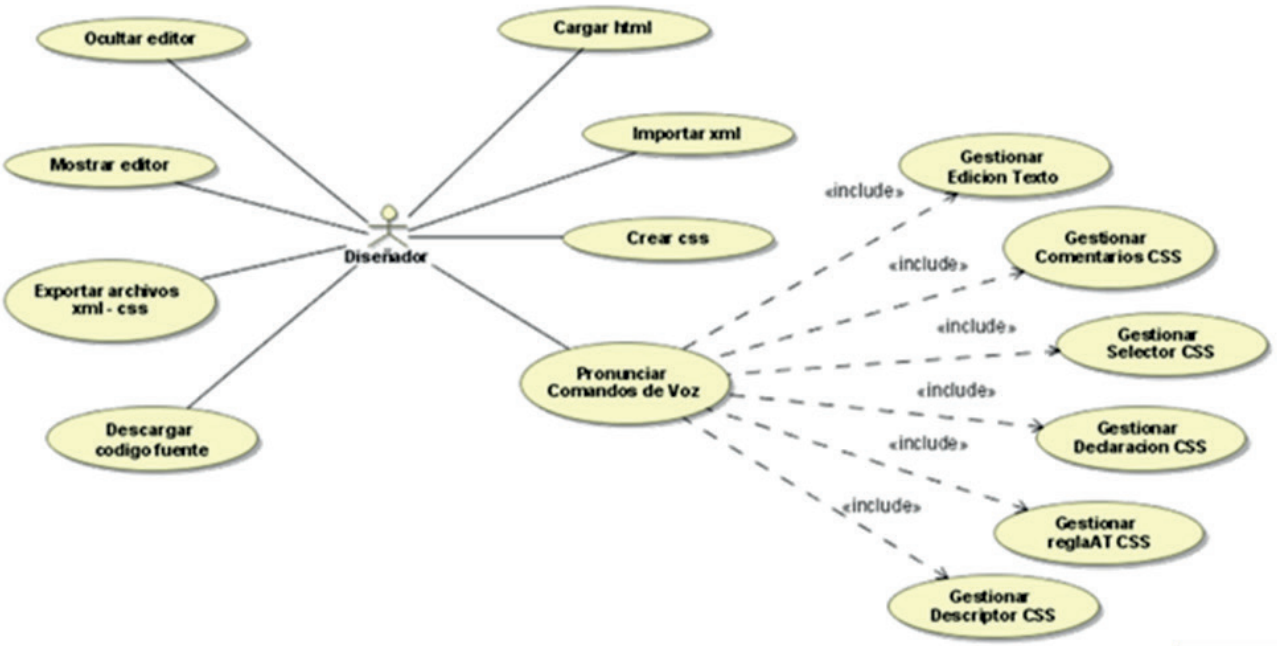
La validación del framework propuesto, así como los resultados obtenidos se detallan a continuación.

5.1 Validación

El proceso de validación del framework web CSS 3 con reconocimiento de voz propuesto constó de dos actividades desarrolladas en un primer escenario: pre-test, y de otras dos actividades más desarrolladas en un segundo escenario: post-test (ver Figura 5). En el primer escenario se capacitó a los participantes sobre el uso del framework y evaluamos sus capacidades sobre maquetado web mediante la aplicación del cuestionario explicado en la sección 3.3 Técnicas de recolección de datos, considerando las dimensiones e indicadores, así como las escalas numéricas para obtener los datos del pre-test.

Figura 4

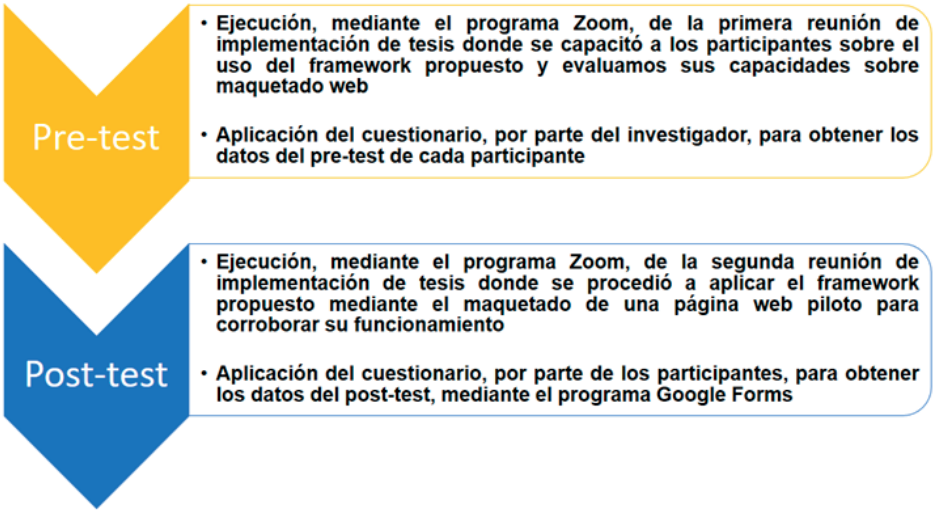
UML – Diagrama general de casos de uso.



Fuente: Elaboración propia

Figura 5

Procedimiento de validación del framework propuesto.



Fuente: Elaboración propia

En el segundo escenario, se procedió a aplicar el framework propuesto mediante el maquetado de una página web piloto para que, luego de finalizada la demostración del diseño de dicha página, se entregue a los participantes el cuestionario del post-test el cual fue llenado según su percepción sobre el grado de cumplimiento de cada indicador.

***5.1.1 Escenario 1: obtención de datos del pre-test*:** este primer escenario de validación inició con la capacitación a los participantes sobre el uso del framework propuesto indicando los conceptos básicos sobre los que se fundamenta su construcción, las actividades necesarias para su uso como son: el paso uno para cargar el archivo html a maquetar, el paso dos para importar una hoja de estilo anteriormente generada o para crear una nueva hoja desde cero, el paso tres donde se explicaron los comandos de voz reconocidos por el framework.

Luego de finalizada la capacitación iniciamos la evaluación del grado de conocimiento de los participantes sobre las dimensiones e indicadores de maquetado web expuestas en el cuestionario mostrado en el Apéndice A. Este proceso consistió en la evaluación, por parte del investigador, de los conocimientos y habilidades de cada participante sobre los indicadores de maquetado web establecidos y valorizados, según la escala de Likert, en el cuestionario mencionado lo cual nos sirvió para obtener los datos del pre-test.

***5.1.2 Escenario 2: obtención de datos del post-test*:** en este segundo escenario finalizamos la capacitación sobre el funcionamiento del framework web CSS 3 con reconocimiento de voz y se procedió a aplicarlo mediante la maquetación de una página web piloto, como medio para corroborar su correcto funcionamiento y cumplimiento con las dimensiones e indicadores del cuestionario. Finalizado el maquetado de la página web piloto solicitamos a los participantes el llenado del cuestionario, según su percepción sobre el grado de cumplimiento de los indicadores con lo cual obtuvimos los datos del post-test.

VI. RESULTADOS

Luego de obtener los datos del pre-test y del post-test se procedió a aplicar la prueba estadística T de Student (ver Tablas 4, 5, 6, 7, 8, 9 y 10).

Se observa que la media de los puntajes con respecto al maquetado web, para el pre-test fue de 77.53, mientras que en el post-test aumentó a 126.27, con una mejora del 38.59%.

Tabla 4

Puntajes del maquetado web.

Test	N	Media	Desviación estándar	Media de error estándar
Pre- test	15	77.53	8.262	2.133
Post-test	15	126.27	23.426	6.049

Fuente: Elaboración propia

Se observa que la media de los puntajes con respecto a la dimensión animación, para el pre-test fue de 9.60, mientras que en el post-test aumentó a 16.40, con una mejora del 41.46%.

Tabla 5
Puntajes de la dimensión animación.

Test	N	Media	Desviación estándar	Media de error estándar
Pre- test	15	9.60	1.454	0.375
Post-test	15	16.40	3.180	0.821

Fuente: Elaboración propia

Se observa que la media de los puntajes con respecto a la dimensión botones, para el pre-test fue de 12.40, mientras que en el post-test aumentó a 20.40, con una mejora del 39.22%.

Tabla 6
Puntajes de la dimensión botones.

Test	N	Media	Desviación estándar	Media de error estándar
Pre- test	15	12.40	2.384	0.616
Post-test	15	20.40	3.961	1.023

Fuente: Elaboración propia

Se observa que la media de los puntajes con respecto a la dimensión contenido, para el pre-test fue de 16.60, mientras que en el post-test aumentó a 24.53, con una mejora del 32.33%.

Tabla 7
Puntajes de la dimensión contenido.

Test	N	Media	Desviación estándar	Media de error estándar
Pre- test	15	16.60	3.291	0.850
Post-test	15	24.53	4.533	1.171

Fuente: Elaboración propia

Se observa que la media de los puntajes con respecto a la dimensión fondos, para el pre-test fue de 12.33, mientras que en el post-test aumentó a 20.53, con una mejora del 39.94%.

Tabla 8
Puntajes de la dimensión fondos.

Test	N	Media	Desviación estándar	Media de error estándar
Pre- test	15	12.33	2.059	0.532
Post-test	15	20.53	3.623	0.935

Fuente: Elaboración propia

Se observa que la media de los puntajes con respecto a la dimensión imágenes, para el pre-test fue de 12.07, mientras que en el post-test aumentó a 20.47, con una mejora del 41.04%.

Tabla 9
Puntajes de la dimensión imágenes.

Test	N	Media	Desviación estándar	Media de error estándar
Pre- test	15	12.07	2.840	0.733
Post-test	15	20.47	4.121	1.064

Fuente: Elaboración propia

Se observa que la media de los puntajes con respecto a la dimensión tipografías, para el pre-test fue de 14.53, mientras que en el post-test aumentó a 23.93, con una mejora del 39.28%.

Tabla 10
Puntajes de la dimensión tipografías.

Test	N	Media	Desviación estándar	Media de error estándar
Pre- test	15	14.53	2.532	0.654
Post-test	15	23.93	4.698	1.213

Fuente: Elaboración propia

La prueba T de Student de igualdad de medias de los puntajes, para cada una de las dimensiones analizadas, muestra que dichas dimensiones tienen valores menores al nivel de significancia 0.05 (ver Tabla 11). Por tanto, se concluye, que el framework web CSS 3 con reconocimiento de voz desarrollado influye positivamente en la automatización del maquetado de páginas web, dado que los valores obtenidos en el post-test son mayores que los del pre-test.

Tabla 11
Prueba t de igualdad de medias de los puntajes.

Dimensiones	t	A
Maquetado web	-7.598	0.05
Animación	-7.531	0.05
Botones	-6.702	0.05
Contenido	-5.485	0.05
Fondo	-7.622	0.05
Imágenes	-6.500	0.05
Tipografías	-6.822	0.05

Fuente: Elaboración propia

VII. DISCUSIÓN

La implementación y gestión de todos los elementos CSS 3, como son: selectores, declaraciones, descriptores, propiedades, valores, reglas-AT y comentarios dentro del framework permite al diseñador web conocer y utilizar todos estos elementos en la maquetación profesional de cualquier elemento de una página web, permitiéndole conocer y disponer a voluntad de estos elementos dentro de un entorno gráfico, mejorando así su capacidad creativa, permitiéndole además el uso de la voz para realizar la maquetación web. Por otro lado, la gestión de los elementos, propiedades y valores a través de comandos de voz soportados por el framework, representa una innovación del método tradicional de diseño web.

Estos resultados concuerdan con los encontrados en (Pilamunga, 2012) donde se evidenció que el uso de hojas de estilo CSS y Scripts en el maquetado de cualquier sitio web hace posible la optimización de su tamaño y funcionalidad pues con tan sólo escribir unas cuantas líneas de código es posible aplicar efectos, estilos, color, forma y animación profesional a un determinado elemento dentro de una página web.

De igual forma los resultados concuerdan con lo encontrado en (Delgado, Sandoval, & Arteaga, 2018) donde se propuso incorporar la voz dentro de un framework como medio de innovación de la programación tradicional como ayuda al proceso de enseñanza – aprendizaje en estudiantes con discapacidad.

VIII. RECOMENDACIÓN

Como trabajos futuros que pueden derivar de esta investigación se sugiere incorporar comandos de voz que hagan posible la automatización del proceso de creación de páginas web mediante la tecnología HTML. Y, así también, se recomienda incorporar comandos de voz que hagan posible la automatización del proceso de validación de páginas web mediante la tecnología JavaScript. Finalmente, resulta altamente interesante extender la funcionalidad del framework para poder visualizar el desarrollo de la hoja de estilo dentro del propio framework y no sólo cuando se descargue el archivo CSS generado.

IX. CONCLUSIONES

Se concluyó que el diseño e implementación de un framework web CSS 3 basado en procesamiento de lenguaje natural influyó de manera significativa en la automatización del maquetado de páginas web y permitió el uso de comandos de voz CSS 3 dentro de un entorno web intuitivo, fácil, moderno e innovador mediante la realización de vistas previas del maquetado según el avance, generación y descarga automática de las hojas de estilo CSS gestionadas mediante el uso del mouse y el teclado, así como de la voz, lo que representó una innovación del método tradicional de diseño web.

X. REFERENCIAS

[11] Aiken, L. (1985). Three coefficients for Analyzing the Reliability and Validity of Ratings. *Educational and Psychological Measurement*, 131-142.

[12] Collado, K. Z., & Aparicio, E. A. (2017). *Diseño de un asistente de voz basado en web para personas con vision subnormal.* Arequipa, Peru.

[13] De Troyer, O. (2001). *Audience-driven web design.* Bruselas: IDEA GroupPublishing.

[14] Delgado, C., Sandoval, J., & Arteaga, W. (2018). Blockly Voice: un entorno de programación guiado por voz. *ACTA NOVA; Vol. 9, N° 1, marzo 2019*, 115-129.

[15] Escurra, L. (1988). *Cuantificación de la validez de contenido por criterio de jueces.* Lima.

[16] Gil, L., Castillo, L., & Flórez, R. (2016). *Reconocimiento de comandos de voz en español orientado al control de una silla de ruedas.* Manizales, Colombia.

[17] Hernadez, M., & Lemuz, R. (2016). Aplicación multimedia para el entrenamiento en la certificación TOEFL mediante reconocimiento de voz. *Research in Computing Science*, 67-75.

[18] Hernandez, C., & Carpio, N. (2019). Introducción a los tipos de muestreo. *ALERTA*, 76-79.

[19] Jaimez, C., & Vargas, R. (2017). Editor web visual para HTML, CSS y JavaScript de apoyo a la docencia. *Virtualidad, Educación y Ciencia*, 136-152.

[20] Jarrar, S. (2002). *Web Design Guidelines for WSDM.* Bruselas.

[21] MDN Web Dcos. (30 de Mayo de 2021). *Uso de la Web Speech API*. Obtenido de Uso de la Web Speech API: https://developer.mozilla.org/es/docs/Web/API/Web_Speech_API/Using_the_Web_Speech_API

[22] MDN Web Docs. (29 de Mayo de 2021). *CSS Tutoriales*. Obtenido de CSS Tutoriales: https://developer.mozilla.org/es/docs/Web/CSS

[23] Mihalcin, T. (2007). *Web Frameworks Comparison Concerning the Efficiency of Development.* Praga.

[24] Muñoz, M. A., & Rodriguez, S. P. (2015). *Diseño e Implementación de un agente animado interactivo para informar sobre el calendario académico de la Universidad de Córdoba mediante reconocimiento de voz basado en un entorno web.* Cordoba, España.

[25] Murillo, J. (2013). *Métodos de Investigación de Enfoque Experimental.* Lima, Peru.

[26] Nagilla, R. (2012). *Comparison of Web Development Technologies - ASP.NET & PHP.* Vasteras y Eskilstuna.

[27] No Magic, Inc. (2010). *Magicdraw Architecture Made Simple.* No Magic, Inc.

[28] Palacio, M. (2020). *Scrum Master - Temario Troncal 1. Versión 3.0.* Iubaris Info 4 Media SL.

[29] Pilamunga, E. M. (2012). *El maquetado a base de scripts y hojas de estilo en cascada (CSS) y su incidencia en la optimización de un sitio web.* Ambato, Ecuador.

[30] Roig, A. (2019). *Desarrollo de una aplicación para la web utilizando el Web Speech API.* Valencia.

[31] Szigeti, S. (2012). *The challenge of web design guidelines: Investigating issues of awarenes, interpretation, and efficacy.* Toronto.

[32] U.S. Department of Health and Human Services. (2006). *Research-Based Web Design and Usability Guidelines.* Washington, D.C: U.S. Government Printing Office.

Fuentes de financiamiento:

Propia.

Conflictos de interés:

El autor declara no tener conflictos de interés.

APÉNDICE

Apéndice A. Cuestionario - Instrumento de recolección de datos

N	Dimensiones	Muy deficiente	Deficiente	Regular	Eficiente	Muy eficiente
	Animación					
1	El maquetado mantiene un diseño de animación y movimiento estándar					
2	El maquetado evita el uso de gifs animados					
3	El maquetado usa animaciones creadas con CSS, evitando el uso de tecnologías desfasadas de animación (flash)					
4	El maquetado mantiene la coherencia entre las animaciones y el contenido para su mejor entendimiento					
	Botones					
5	El maquetado mantiene un diseño de botones e íconos estándar					
6	El maquetado utiliza colores para indicar la acción a realizar (verde para grabar, rojo para eliminar, entre otros)					
7	El maquetado utilizar íconos según la funcionalidad a realizar (basurero para eliminar, flecha izquierda para retroceder, entre otros)					
8	El maquetado evita el uso innecesario de íconos y gráficos					
9	El maquetado cambia de color o forma para enfatizar la interacción entre el elemento y el usuario					
	Contenido					
10	El maquetado mantiene un diseño de contenido estándar					
11	El maquetado permite el movimiento vertical dinámico del encabezado de las columnas de una tabla con más de dos filas					
12	El maquetado evita el uso innecesario de fotografías					
13	El maquetado evita o limita el uso de mapas de imágenes					
14	El maquetado brinda una descripción por escrito de cualquier información crítica contenida en los archivos de audio publicados					
15	El maquetado utiliza los mismos colores para agrupar elementos relacionados (menús de navegación, íconos, datos relacionados, entre otros)					
	Fondo					
16	El maquetado mantiene un diseño de fondo estándar, evitando tener fondos de colores diferentes para cada página					
17	El maquetado asegura diseños de fondos sencillos para mejorar la legibilidad del texto					
18	El maquetado asegura que las imágenes de fondo no hagan imposible la lectura y compresión de la página					
19	El maquetado cuenta con colores de fondo para proporcionar el máximo contraste, evitando sombras					
20	El maquetado cuenta con el color blanco o colores claros, para el fondo					
	Imágenes					
21	El maquetado mantiene un diseño de imágenes estándar					
22	El maquetado asegura que todas las imágenes sean nítidas y fáciles de visualizar					
23	El maquetado mantiene la cantidad de colores en sus imágenes al mínimo					
24	El maquetado evita las imágenes peige o marrones					
25	El maquetado utilizar imágenes entrelazadas o progresivas					
	Tipografía					
26	El maquetado mantiene un diseño de tipografía estándar					
27	El maquetado cuenta con textos fáciles de leer e interpretar por el usuario					
28	El maquetado evita texto de colores amarillo o blanco, ya que son difíciles de leer e imprimir, y evita la combinación de rojo y azul					
29	El maquetado evita el uso de letras mayúsculas en todo el texto					
30	El maquetado evita el texto parpadeante					
31	El maquetado utiliza sólo un número limitado de fuentes cuidadosamente seleccionadas					