Universidad de las Ciencias Informáticas
Vertex, Interactive 3D Environments, Facultad 3

Plantilla LaTeX 2_ε para el informe de tesis de Grado Científico en Cuba.

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Wendy Díaz Ortiz

Tutores: Dr.C. Hassán Lombera Rodríguez

Lic. Sandy Díaz Ramos

La Habana, 2020

Dedicatoria

Agradecimientos

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Wendy Díaz Ortiz
Autor

Dr.C. Hassán Lombera Rodríguez
Tutor

Lic. Sandy Díaz Ramos
Tutor

Resumen

En la actualidad existe un numeroso grupo de herramientas informáticas que permiten la edición de informes de tesis. Estas herramientas se pueden clasificar según su costo, si el código es abierto no, si son multplataforma o no, etc. En el caso particular de los procesadores de texto el más utilizado en Cuba es Microsoft Word. No obstante en el caso de los informes para las tesis de grado científico, existen normas muy específicas, dictadas por La Comisión Nacional de Grado Científico. Cumplir estas normas puede agregar complejidad al proceso de elaboración de estos informes, puesto que el doctorando debe además de preocuparse por el contenido en sí también hacerlo por el formato. Desde la década de los 60 existe un sistema de composición de textos (LaTeX) que no es más que un conjunto de macros del lenguaje TeX, ideado por Donald Knuth. Este sistema se clasifica como software libre y de código abierto, además permite gestionar de forma separada el contenido del formato. Unido a que dentro de sus funcionalidades se encuentra la utilización de plantillas mediante archivos de clases (.cls), hacen de LaTeX una excelente elección como aplicación donde desarrollar una plantila que permita a los doctorandos cubanos abstraerse del formato y por tanto concentrarse en el contenido, en el proceso de elaboración de sus informes de tesis. La correcta utilización de esta plantilla traerá como valor agregado que al minimizarse los errores de formato en estos documentos, los revisores de los mismos se podrán concentrar también en el contenido de estos informes.

Palabras clave: Informe de tesis, plantilla, LaTeX.

Índice general

Índice de figuras

Índice de tablas

Introducción

El 11 de julio de 2019, el Consejo de Estado de la República de Cuba adoptó el acuerdo número 8625, en el que se establece que el Ministerio de Educación Superior debe definir los componentes que integran el modelo de formación continua de la educación superior cubana. Estos tres componentes son: La Formación de Pregrado en carreras de Perfil Amplio, La preparación para el Empleo y La Educación de Posgrado. En el caso de la última plantea que posibilita entre otras cuetiones la especialización, la reorientación y la actualiación permanente de los graduados universitarios. (Educación Superior, 2019a)

Por otra parte, la Resolución No.140-2019 del Reglamento de Posgrado en su capítulo II, artículos 4 y 10, plantea que la educación de posgrado es uno de los componentes del modelo de formación continua de la educación superior y complementa al pregrado en carreras de perfil amplio y a la preparación para el empleo. (Educación Superior, 2019b)

La superación profesional tiene como objetivo contribuir a la educación permanente y la actualización sistemática de los graduados universitarios, el perfeccionamiento del desempeño de sus actividades profesionales y académicas, así como el enriquecimiento de su acervo cultural. (ibíd.)

La investigación, es la actividad humana, orientada a la obtención de nuevos conocimientos a partir de interrogantes científicas, así como su aplicación para la solución de problemas de carácter tecnológico, económico, social u otro tipo. El trabajo de investigación, es el que se realiza en el marco de una investigación para contribuir al logro de sus objetivos. Puede incluir todas las etapas del proceso o solo algunas de ellas. A partir de las definiciones anteriores, las tesis y otros similares pueden constituir trabajos de investigación, aunque ello no se cumple en el sentido inverso (no todos los trabajos de investigación son tesis).

La tesis tiene como propósito demostrar que el graduado puede aplicar el conocimiento que caracteriza a su profesión o disciplina, así como los métodos de estudio propios de la misma, para solucionar problemas relacionados directamente con las actividades que caracterizan al perfil del egresado. Se busca que el estudiante demuestre que puede desarrollar y comprobar soluciones creativas a problemas profesionales. La tesis constituye el resultado de una investigación en un campo disciplinario o multidisciplinario, que se caracteriza por analizar críticamente diferentes puntos de vista teóricos y prácticos, y argumentar a partir de ello, la posición del investigador. Implica, plantearse interrogantes, fundamentarlas y responderlas por medio de la investigación.

Los resultados de las tesis doctorales constituyen un aporte original al conocimiento en la disciplina, demandando del doctorando un amplio dominio de la teoría y práctica relacionada con los métodos de investigación, y un uso pertinente del enfoque multidisciplinario. Estas características se ponen de manifiesto

al derivarse de las tesis uno o varios artículos especializados, que son publicados aprobados para su publicación en revistas especializadas indexadas. (San Martín de Porres (USMP), 2016)

La CNGC, organismo rector del Sistema Nacional de Grados Científicos, se adscribe al Ministerio de Educación Superior y está presidida por su Ministro, quien la representa ante las entidades nacionales y extranjeras, así como trasmite a la comisión las orientaciones sobre la política de grados científicos del Estado y el Gobierno. El Presidente de la CNGC establece su estructura, composición y funcionamiento. La función principal de esta entidad es elaborar los procedimientos para la realización de los procesos de obtención de grados científicos y controlar su cumplimiento.

En los artículos 5.2 del Decreto ley No. 372 del Sistema Nacional de Grados Científicos (Grados Científicos, 2005) y 14.4 y 22 de la Resolución No. 139/19 del Ministerio de Educación Superior (Educación Superior, 2019a) queda establecido que para la obtención del grado científico de Doctor en Ciencias de cualquier especialidad se debe realizar una tesis, cuyo contenido esencial deberá ser entregado en forma escrita para su defensa. En este documento el doctorando debe demostrar su grado de madurez científica, su capacidad de enfrentar y resolver problemas complejos de manera independiente y un profundo dominio teórico y práctico del área del conocimiento del programa cursado, a través de la exposición del resultado alcanzado, basado en la solución novedosa de un problema científico teórico o práctico. (Justicia, 2019)

Tomando en consideración lo anteriormente planteado, la CNGC adoptó una norma desde el año 2005 sobre cómo debe realizarse la redacción, formato y presentación de estas tesis. Estas normativas le imprimen al formato de los documentos de tesis un necesario rigor milimétrico; pero al mismo tiempo hacen de la escritura de estos documentos un proceso tedioso y obstaculizador del esencial proceso creativo e investigativo.

Por lo general es utilizado para la edición de estos documentos el software de ofimática Word (Microsoft, 2020). Este procesador de texto es muy popular incluso dentro de la comunidad científica. En Cuba el uso de Word, a pesar de su arraigo, va en contra de principios como la soberanía tecnológica y técnicamente hablando presenta algunas deficiencias al usarlo en la confección de textos como los informes para las tesis de grado científico. Realizando una revisión a 30 tesis doctorales, en formato doc, docx y pdf, defendidas de las especialidades de Ciencias de la Educación Médica, Ciencias de la Educación y Ciencias Pedagógicas se detectó al menos una incongruencia con el formato establecido por la CNGC en todas ellas. Por ejemplo, una regularidad detectada es el hecho de crear las páginas que no se paginan separadas de las paginadas, dando esto un indicio del hecho de constituir una práctica extendida la realización de índices manualmente. Resulta significativo observar que también las referencias bibliográficas se manejan en muchos casos de esta forma. Otras de las deficiencias detectadas son:

- Cambios de formato al utilizar varias versiones de Word para editar el documento.
- Múltiples espacios: De manera muy sencilla se introducen, consciente e inconscientemente múltiples espacios entre palabras, párrafos etc. Estos espacios suelen aparecer por querer alinear textos con la barra espaciadora o establecer sangrías con más espacios de los normados o simplemente por descuido.

- Cambios de fuentes y tamaños de letra: Es muy fácil realizar este tipo de cambios en páginas diferentes del documento sin percibirlo.
- Uso indiscriminado de la tecla Enter: Es usual establecer los saltos de página o sección, por ejemplo, utilizando esta vía y esto causa que las configuraciones establecidas para este tipo de saltos no se hagan efectivas. Esto también repercute en la generación de índices.
- Cambios de márgenes: Es muy fácil, incoscientemente, hacer estos cambios utilizando solo el mouse. Pueden afectar a todo el documento o algunas de sus secciones.
- Pies de figuras y tablas que no quedan en la misma página que estas.
- Dificultades asociadas al hecho de ser privativo: No se pueden agregar nuevas funcionalidades ni personalizaciones, escasa documentación y ayuda.
- Escritura de documentos extensos: Se hace tedioso y en muchos casos se introducen errores en partes del documento que ya están acabadas.

Es por todo ello que se formula el siguiente **problema a resolver**: ¿Cómo mejorar el proceso de elaboración del informe de Tesis de Grado Científico en Cuba en correspondencia con lo normado por la Comisión Nacional de Grado Científico?

En concordancia con lo anterior, se toma como **objeto de estudio** el proceso de elaboración de informes de tesis centrando el **campo de acción** en la elaboración de plantillas de informes de tesis de grado científico en Cuba.

En respuesta al problema, se define como **objetivo general** desarrollar una plantilla LaTeX para la elaboración de los informes de Tesis de Grado Científico en Cuba que mejore el proceso de elaboración de estos informes, garantizando de forma automática el cumplimiento de lo normado por la Comisión Nacional de Grado Científico. Lo anterior deriva en los siguientes **objetivos específicos**:

1. Elaborar el estado del arte de las distintas aplicaciones que permiten desarrollar plantillas para informes de Tesis Doctoral.
2. Seleccionar las características de los sistemas y aplicaciones anteriormente mencionados que podrían ser de utilidad para la resolución del problema planteado.
3. Implementar una plantilla (clase) LaTeX para la edición de los informes de Tesis Doctorales en Cuba, a partir de cumplir con las normativas planteadas por la Comisión Nacional de Grado Científico.
4. Generar la ayuda de la clase desarrollada, según los estándares de publicación de www.ctan.org y ejemplos de uso donde se evidencie la facilidad de generación de tablas y otros objetos complejos.
5. Realizar pruebas unitarias para la detección de errores en etapas tempranas del desarrollo, a partir del análisis de pequeñas unidades de código.

Para dar cumplimiento a estos objetivos, se deberán desarrollar las siguientes **tareas de investigación**:

1. Elaborar el perfil del trabajo de diploma.
2. Realizar un estado del arte de las regulaciones y normas vigentes para la escritura de informes de Tesis de Grado Científico en Cuba.

3. Seleccionar los paquetes de LaTeX necesarios para desarrollar la clase o plantilla, objetivo de la tesis.
4. Estudiar, a partir de los paquetes seleccionados, cómo se debe realizar la compilación de estos informes en LaTeX.
5. Estudiar, como elaborar ayudas de clases de LaTeX desde el mismo desarrollo de la clase.
6. Desarrollar la clase tesisdoccuba.cls. Teniendo en cuenta los elementos aprendidos anteriormente.
7. Depuración de errores mediante pruebas unitarias y consulta a expertos.

Además, para asegurar el correcto desarrollo de la investigación, se deberá hacer uso de **métodos científicos**. Entre ellos, de corte teórico:

- **Análisis histórico-lógico**: para el estudio de la evolución y situación actual del proceso de elaboración de informes de Tesis de Grado Científico a nivel mundial, y en Cuba de forma particular. Haciendo énfasis en la elboración de plantillas automatizadas que permitan al usuario abstraerse del formato requerido para estos documentos.
- **Analítico-sintético**: para un estudio crítico de las soluciones informáticas existentes a nivel nacional e internacional. Con ello se pretende identificar elementos relevantes, tendencias y tecnologías más comunes en el desarrollo de sistemas informáticos para la elaboración de informes de Tesis de Grado Científico, lo cual servirá como punto de partida para la solución propuesta.
- **Modelación**: para un mejor entendimiento de los procesos a informatizar. Provee una abstracción de la realidad que facilita la comunicación con el cliente, la mejor visualización del sistema y su posterior implementación.

Como método empírico, la **entrevista** juega un papel esencial en el desarrollo del presente Trabajo de Diploma, específicamente la entrevista no estructurada, dada su flexibilidad y capacidad de obtener conocimiento de un tema directamente de los especialistas.

Como **resultado esperado** se aspira a una clase LaTeX que permita a los Doctorandos en Cuba concentrarse en el contenido y gestionar todo el formato a través de esta plantilla, automatizadamente.

Capítulo 1. Fundamentación teórica: describe los principales conceptos asociados al dominio del problema, así como las relaciones que se establecen entre ellos. Expone un análisis de las soluciones existentes en el mercado, resaltando las tendencias y tecnologías más comunes en su desarrollo. Finalmente, se argumentan las bases del desarrollo de la herramienta en cuestión.

Capítulo 2. Propuesta de solución: refleja cada uno de los pilares del desarrollo de la clase prpuesta, la especificación de sus funcionalidades así como el uso de patrones y estándares de codificación.

Capítulo 3. Validación de la propuesta: refiere la estrategia de pruebas definida a partir de la metodología de desarrollo, con el objetivo de garantizar la validación de la solución propuesta. Se analizan los niveles de pruebas ejecutados y los resultados alcanzados. Finalmente, se presentan evidencias de cómo la solución propuesta puede contribuir al proceso de elaboración de informes de Tesis de Grado Científico en Cuba.

CAPÍTULO 1

FUNDAMENTACIÓN TEÓRICA

El presente capítulo recoge los elementos teóricos que fundamentan el desarrollo de una clase o plantilla para la elaboración de informes de tesis de Grado Científico en Cuba. Se realiza un análisis de la situación actual y perspectivas del proceso de elaboración de este tipo de informes en Cuba y el mundo, así como la importancia de una eficiente automatización del formato normado para estos documentos. A continuación, se reflejan las características esenciales de las diferentes soluciones informáticas existentes en el mercado; dejando claro, de manera general, las tendencias actuales en el desarrollo de este tipo de sistemas y las tecnologías más comúnmente utilizadas. Finalmente, se exponen los elementos más relevantes de la metodología, lenguajes y herramientas definidas para el desarrollo; reflejándose el uso de pruebas para garantizar la efectividad del proceso de desarrollo y calidad del producto final.

1.1. DESCRIPCIÓN GENERAL DEL OBJETO DE ESTUDIO

Según Stanley Morrison, la tipografía es el "arte de disponer correctamente el material de imprimir, de acuerdo con un propósito específico: el de colocar las letras, repartir el espacio y organizar los tipos con vistas a prestar al lector la máxima ayuda para la comprensión del texto escrito verbalmente"(Morrison, 1929)

Desde antes de nuestra era los romanos tenían sellos que imprimían hojas completas de inscripciones sobre objetos de arcilla, sin embargo, no es hasta el siglo XV que surge la imprenta de manos de Johannes Gutenberg (Schwanitz, Dietrich, «1: Historia de Europa». La cultura: Todo lo que hay que saber, 2006).

Existe una gran acumulación de saberes avalados por investigaciones científicas sobre la comprensión humana, canalizadas en cómo diseñar, como dar un formato a los disímiles documentos donde la humanidad guarda su gran tesoro: el conocimiento. Ha sido tal la comprensión del hombre acerca de estudiar estos elementos que ha organizado ciencias que entre otras tienen la misión de investigar y perfeccionar el diseño de documentos, creando así normas, estándares de aceptación demostrada. En tal caso se encuentran las ciencias del diseño y las de la Información, entre otras.

Antes de la llegada a la era digital el proceso de edición de documentos era muy tedioso. Debían intervenir en el proceso de conformación de un documento disímiles actores, más allá del generador de ese conocimiento en sí. Un ejemplo claro, que además sentó un hito en el diseño y normalización de documentos fue el siguiente:

Donald Ervin Knuth, profesor en la Universidad de Stanford, recibió el 30 de marzo de 1977 las galeradas o pruebas de imprenta de la segunda edición del segundo volumen de su famosa obra The Art of Computer Programming, bien conocida por todos los informáticos y muchos matemáticos. La impresión que dichas pruebas causó al autor fue nefasta; él mismo las calificó de tipográficamente horribles y tan importantes le parecieron los problemas a los que se enfrentaba que decidió resolverlos por sí mismo, iniciando así un periodo de interés y estudio en torno al mundo de la tipografía, diseño de tipos y otros aspectos de la edición y composición profesionales de documentos. El 12 de julio siguiente había concluido el segundo de dos informes, de uso personal, conteniendo, en germen, la globalidad de TeX y METAFONT. Las dos metas fundamentales de Knuth fueron: la calidad, pues no se contentó con hacer algo bueno, sino que quiso lo mejor, y la "intemporalidad" o independencia, en el mayor grado posible, de los cambios tecnológicos en la impresión. (Cascales et al., 2000)

Es entonces, gracias al desarrollo de las TIC que se logra un acercamiento del proceso de maquetación de los documentos a los autores de los textos. Reduciendo drásticamente el tiempo para dejar listo un documento para ser distribuido y consultado por la sociedad.

El proceso de edición digital hoy en día es algo común y que una inmensa mayoría realiza utilizando herramientas de ofimática, principalmente el procesador de texto Word de Microsoft. En tal sentido se puede apreciar en la Figura 1.1 la tendencia que ofrece Google para las búsquedas asociadas al término Microsoft Word en el último año.

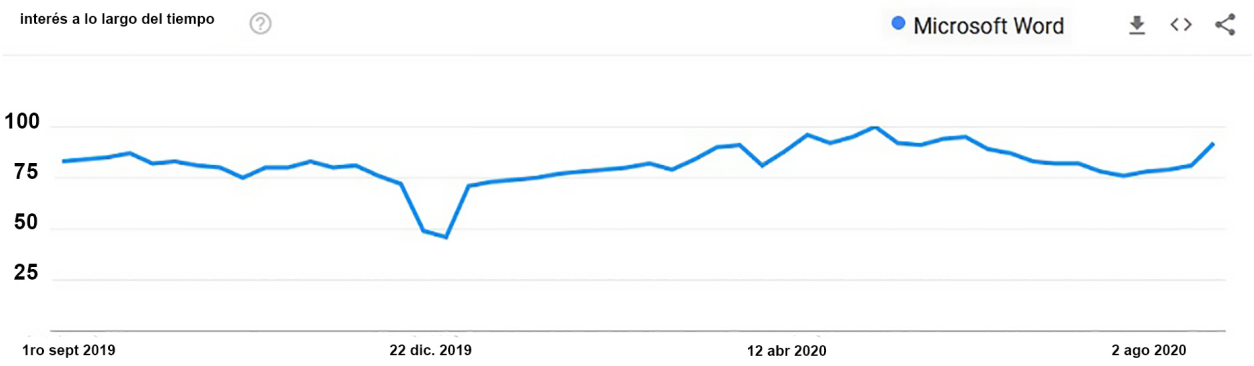

Figura 1.1. Tendencia de búsquedas en Google del término Microsoft Word en el último año.

También en la Figura 1.2 se puede apreciar una comparación entre las búsquedas hechas para los términos Microsoft Word, Writer Libre Office y LaTeX, respectivamente en el último año. Aquí se puede apreciar como las búsquedas hechas para Microsoft Word sobrepasan en todo momento a las realizadas para las otras herramientas, estando en segundo lugar LaTeX. Esto entre otras causas se debe a que:

- Word tiene una comunidad más amplia mientras los usuarios de LaTeX se concentran más en la comunidad científica.
- Existe vasta información en las redes sobre el uso de los tres softwares sin embargo en el caso de LaTeX existen páginas como www.ctan.org donde practicamente toda la información y paquetería se puede consultar y de forma muy organizada. Según datos que se pueden encontrar en la propia web existe un nodo principal y 75 espejos de este sitio en el mundo y solo en el nodo principal se registra un tráfico mensual de 6 TB.

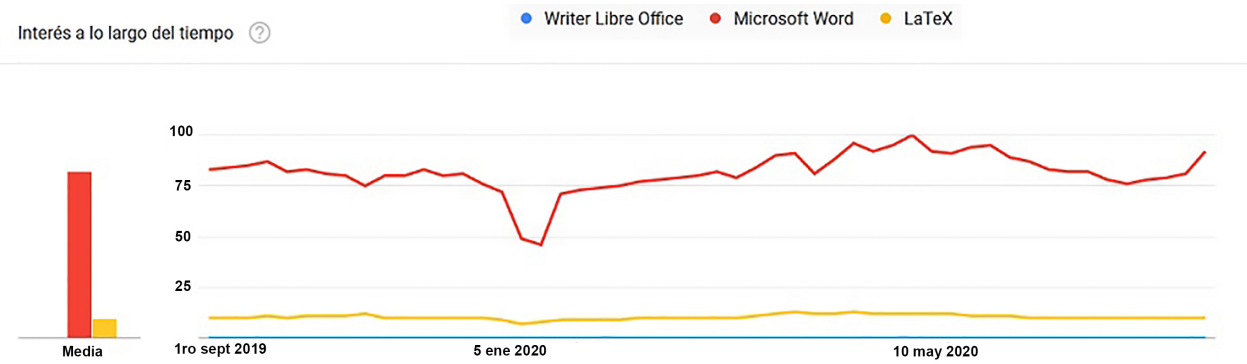

Figura 1.2. Comparación de las tendencias de búsqueda en Google de los términos Microsoft Word, Writer Libre Office y LaTeX , respectivamente en el último año.

Con la misma herramienta de tendencias de búsqueda de Google, GoogleTrends (Google, 2020) se puede obtener un ranking de las regiones del planeta donde se realizan más búsquedas de cada término en la Figura 1.3 y la Figura 1.4 se muestran los ranking para los términos Microsoft Word y LaTeX respectivamente. Un detalle que no se puede pasar por alto es como existe más interés por LaTeX en países desarrollados, al contrario de Microsoft Word.

Figura 1.3. Países que hicieron más búsquedas del término Microsoft Word en el último año.

Figura 1.4. Países que hicieron más búsquedas del término LaTeX en el último año.

Por otra parte, en el proceso de edición con Word se presentan algunas dificultades, esencialmente atribuidas al hecho de ser un procesador de textos del tipo lo que ves es lo que tienes (WYSIWYG, por sus siglas en inglés). Aquí es importante resaltar que esta característica (WYSIWYG) está indisolublemente ligada al hecho de que en el proceso de confección del documento están totalmente mezcladas las tareas relacionadas con el formato y el contenido en sí del documento. En la introducción de este informe de tesis, en la página 2 se presentaron algunas de estas dificultades.

Esta mezcla en la gestión del formato y del contenido hacen que sea muy fácil cambiar elementos del formato por personas poco experimenadas en el uso de Word y en ocasiones hasta incoscientemente se pueden introducir errores. Nótese que solo con tocar la barra espaciadora puede quedar un espacio indeseable en el documento, se puede correr una figura o una tabla, por solo citar ejemplos sencillos.

Una de las herramientas informáticas que permiten separar el contenido del formato es LaTeX. Como se mencionó anteriormente este sistema informático se emplea para la edición de documentos. Textos como "LaTeX, una imprenta en tus manos", de autores españoles, dan una idea bastante concreta de las facilidades de LaTeX en el diseño y edición de documentos y presentan a LaTeX más allá que como un sistema informático como un sistema de composición de textos, un procesador de texto. La robustez de este sistema viene heredada del lenguaje de composición tipográfica TeX, herramienta creada por Donald Knuth en los años 70 del pasado siglo.

El hecho de poder separar la gestión del formato de la gestión del contenido permite que sea más viable desarrollar plantillas que no puedan ser facilmente editas por el usuario y mucho menos incoscientemente.

Algunas ventajas del uso de LaTeX son las siguientes. Es importante aclarar que la inmensa mayoría son características con las que no cuenta Word:

- La alta calidad tipográfica con que se obtienen los documentos. Esto principalmente como una característica heredada de TeX. Por ejemplo, en la Figura 1.5 aparecen textos similares generados en Word 2016 y LaTeX 2_ε.

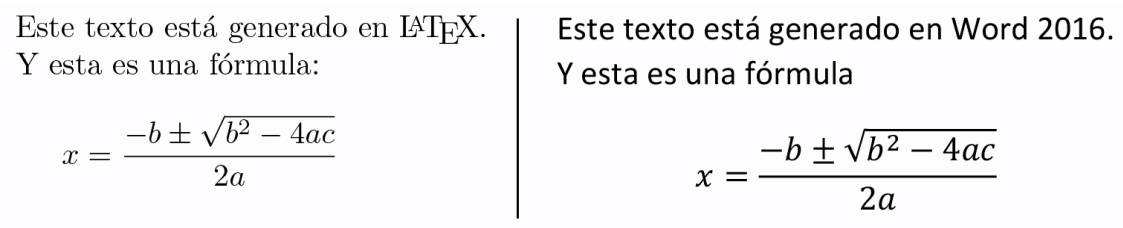

Figura 1.5. Comparación de la limpieza y calidad tipográfica de LaTeX y Word

- La facilidad y calidad para editar fórmulas matemáticas.
- Separa el contenido del formato, como método de trabajo. Esto permite que el usuario final se concentre en el contenido más que en el formato.
- Permite manejar fácilmente estructuras complejas como índices, glosarios, notas al pie, bibliografía etc.
- Es muy sencillo evitar la definición múltiple de acrónimos.
- Por defecto, y como filosofía de trabajo, para dejar más de un espacio es preciso utilizar un comando. Esto implica que solo se obtengan múltiples espacios si el usuario conscientemente los desea. Dicho de otra forma, es difícil que un usuario se equivoque en este sentido. Lo mismo ocurre con el tipo de letra.
- Permite crear proyectos donde es posible usar un archivo para cada parte del documento. Esto le aporta organización y evita la contaminación de errores en aquellas partes ya acabadas.
- Existen paquetes que permiten gestionar citas textuales según la cantidad de palabras de modo automático.
- También existen paquetes que permiten gestionar eficientemente referencias cruzadas y referencias bibliográficas. En el caso de estas últimas se utilizan ideas similares a las empleadas en la gestión de una base de datos.
- Es posible programar límites de hojas y palabras de tal forma que el usuario sepa cuando ha llegado a la cantidad límite en este sentido. En la Figura 1.6, se muestra la implementación de un comando dedicado a este tipo de avisos.
- Es posible crear nuevos paquetes que permita personalizar total o parcialmente plantillas y documentos.
- Permite generar plantillas dinámicas que facilitan considerablemente el proceso de revisión del documento.
- Permite la generación automática de tablas, a partir de la entrada de algunos parámetros básicos.
- Es software libre y existen millones de paquetes a disposición de todos sus usuarios para abordar los más disímiles formatos de documentos.
- Hereda de TeX su filosofía concerniente con que un documento escrito hoy al compilarlo dentro de 20 años se debe obtener el mismo resultado.

```
\newcommand{\set@pagelimit}[1]
{
    \setcounter{pagecount}{0}%
    \gdef\maxpages{#1}%
    \ifx\latex@outputpage\@undefined\relax%
        \global\let\latex@outputpage\@outputpage%
    \fi%
    \gdef\@outputpage{%
        \stepcounter{pagecount}
        \ifnum\value{pagecount}>\maxpages \relax%
             % Comment lines below until the \else
statement to silently remove all additional pages
         \@sc@wm@stampfalse % for avoiding writing the
draftwatermark package text on top
\AddEverypageHook{\@pagestamp}
          \latex@outputpage%
         \else%
            \latex@outputpage%
        \fi%
    }%
}
```

Figura 1.6. Fragmento de código utilizado para establecer una señal automática en cuanto a la cantidad de páginas del documento respecto a la cantidad normada por la CNGC

- Todos los archivos de paquetes, estilos, clases e incluso los propios documentos editados con LaTeX están constituidos por archivos en texto plano lo que permite que sea natural el uso de controladores de versiones como git y que su almacenaje y envío no sea engorroso a pesar de tratarse de grandes documentos.

Es por todo esto que se utilizará en el desarrollo de la herramienta propuesta como objetivo de la presente Tesis el sistema de composición de textos LaTeX así como el lenguaje TeX para aquellos elementos hasta donde LaTeX como sistema de macros de TeX no ha podido llegar.

1.2. METODOLOGÍA, LENGUAJES Y HERRAMIENTAS DE DESARROLLO

El análisis minucioso de los elementos abordados en el epígrafe anterior, arrojó los fundamentos del conjunto de decisiones relacionadas con el desarrollo del sistema en cuestión, entre ellas, la elección de la metodología, lenguajes y herramientas de desarrollo.

1.2.1. METODOLOGÍA DE DESARROLLO

El proceso de desarrollo de software, definido por Jacobson y otros (Jacobson, I., G. Booch y J. Rumbaugh, 2000) como el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software, tiene como finalidad la obtención de un producto que cumpla con las expectativas del cliente.

En el logro de ese objetivo, cobra vital importancia la elección de la metodología de desarrollo apropiada. La misma, guía el proceso de desarrollo para alcanzar la satisfacción del cliente y del equipo. La elección de la metodología no es un proceso trivial, en ocasiones se torna una tarea bien difícil de llevar a cabo, debe elegirse aquella que mejor se ajuste a las características del equipo de desarrollo y las exigencias de los usuarios finales.

Las metodologías de desarrollo se pueden dividir en dos grandes grupos:

Metodologías pesadas: se centran en la definición minuciosa de los procesos y tareas que se realizarán durante el ciclo de vida del software, generando una extensa documentación asociada a cada elemento del proceso de desarrollo. (Charvat, 2003)

El **Proceso Unificado de Desarrollo** (*Rational Unified Process, RUP*) resulta el exponente por excelencia entre las metodologías pesadas. Es un marco de trabajo genérico que puede especializarse para una gran cantidad de sistemas de software, para diferentes áreas de aplicación y diferentes tipos de organizaciones; se basa en la construcción de componentes de software interconectados a través de interfaces bien definidas. (Jacobson, I., G. Booch y J. Rumbaugh, 2000)

El Proceso Unificado se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo concluye con una versión del producto para los clientes, estos constan de cuatro fases: inicio, elaboración, construcción y transición. Cada fase se divide en iteraciones. Cada ciclo produce una nueva versión del sistema y cada versión es un producto preparado para su entrega. El producto final incluye los requisitos, casos de uso, especificaciones no funcionales, casos de pruebas, el modelo de arquitectura y el modelo visual. (Booch, G., J. Rumbaugh e I. Jacobson, 1999)

Metodologías ágiles: esbozan los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y responder a los cambios que puedan surgir a lo largo del proyecto, estas ofrecen una alternativa a los procesos de desarrollo de software tradicionales. (Letelier, P y M. C. Penadé, 2003)

La **Programación Extrema** (*eXtreme Programming, XP*) tal vez sea la más conocida y más extensamente usada de las metodologías ágiles, en sus inicios desafió numerosos dogmas convencionales asociados al desarrollo de software (Beck, K. y C. Andres, 2004). En XP todos los requisitos son expresados como escenarios (historias de usuarios), los cuales son implementados directamente como una serie de tareas. Esta metodología resulta apropiada para proyectos con requisitos imprecisos y cambiantes, en los que existe un alto riesgo técnico. Los programadores trabajan en parejas y se desarrollan pruebas para cada una de las tareas. Todas las pruebas deben ser ejecutadas satisfactoriamente

cuando el nuevo código es integrado al sistema (Sommerville, 2007).
Según (Sommerville, 2005) los métodos ágiles universalmente dependen de un enfoque iterativo para la especificación, desarrollo y entrega del software, y principalmente fueron diseñados para apoyar al desarrollo de aplicaciones de negocio donde los requerimientos del sistema normalmente cambian rápidamente durante el proceso de desarrollo. Están pensados para entregar software funcional de forma rápida a los clientes, quienes pueden entonces proponer que se incluyan en iteraciones posteriores del sistema nuevos requerimientos o cambios en los mismos. En la tabla 1.1 se observan los principios por los que se rigen estos métodos.

Principio	Descripción
Participación del cliente	Los clientes deben estar fuertemente implicados en todo el proceso de desarrollo. Su papel es proporcionar y prorizar nuevos requerimientos del sistema y evaluar las iteraciones del sistema.
Entrega incremental	El software se desarrolla en incrementos, donde el cliente especifica los requerimientos a incluir en cada incremento.
Personas, no procesos	Se deben reconocer y explotar las habilidades del equipo de desarrollo. Se les debe dejar desarrollar sus propias formas de trabajar, sin procesos formales, a los miembros del equipo.
Aceptar el cambio	Se debe contar con que los requerimientos del sistema cambian, por lo que el sistema se diseña para dar cabida a estos cambios.
Mantener la simplicidad	Se debe centrar en la simplicidad tanto en el software a desarrollar como en el proceso de desarrollo. Donde sea posible, se trabaja activamente para eliminar la complejidad del sistema.

Tabla 1.1. Principios de los métodos ágiles

1. Si bien la idea de la participación del cliente en el proceso de desarrollo es activa, su éxito depende de tener un cliente que esté dispuesto y pueda pasar tiempo con el equipo de desarrollo y que pueda representar a todos los stakeholders del sistema. Frecuentemente, los representantes de los clientes están sometidos a otras presiones y no pueden participar plenamente en el desarrollo del software.
2. Los miembros individuales del equipo pueden no tener la personalidad apropiada para la participación intensa que es típica de los métodos ágiles. Por lo tanto, es posible que no se relacionen adecuadamente con los otros miembros del equipo.
3. Priorizar los cambios puede ser extremadamente difícil, especialmente en sistemas en los que existen muchos stakeholders. Por lo general, cada stakeholders proporciona prioridades distintas a diferentes cambios.
4. Mantener la simplicidad requiere un trabajo extra. Bajo presión por las agendas de entregas, los miembros del equipo pueden no tener tiempo de llevar a cabo las simplificaciones deseables del sistema.

Una de las características distintivas de la Metodología XP es su indudable pertinencia bajo la premisa de utilizar el paradigma de programación orientada a objetos. En este caso no se utilizará ese paradigma de programación dado por las peculiares características de TEX y LATEX. El primero que puede ser clasificado como un lenguaje de "bajo nivel" y el segundo que está formado por un conjunto de macros de TEX con la intensión de facilitar el uso de este último. Es por ello que se vuelve la mirada a otras metodologías ágiles entre las que también se destaca Scrum.

Scrum es un marco de trabajo para desarrollo ágil de software que se caracteriza por :

- Seguir una estrategia de desarrollo incremental, en lugar de la planificación y por tanto ejecución completa del producto. (Malpica, 2014)
- Basar la calidad del resultado en el conocimiento de personas auto organizadas en equipos muy pequeños. (Green, 2016)
- No necesariamente seguir un ciclo secuencial o en cascada para las fases del desarrollo.
- Gestionar regularmente las expectativas del cliente, potenciando la flexibilidad y adaptación del equipo a sus demadas progresivas.
- Realizar regularmente una reunión de Scrum, que es una reunión de avance que de corta duración con el objetivo de obtener realimentación sobre las tareas del equipo y los obstáculos que se presentan. (Kniberg, 2015)
- Su flujo de trabajo se basa en la realización de sprints. Estos son los períodos en que se lleva a cabo el trabajo.
- Toma como una de las herramienta o artefacto las Historias de Usuarios, que son definidas a partir de los requisitos que sean necesarios para el desarrollo. (Mckenna, 2016)

La evaluación de las metodologías y la necesidad de una variante flexible a los cambios, preparada para ser ejecutada por un equipo pequeño y que genere los artefactos mínimos para la comunicación con el cliente, permitió identificar a Scrum como la alternativa más acertada.

1.2.2. LENGUAJE DE PROGRAMACIÓN

De acuerdo con las valoraciones de Trejos Buritica (Trejos Buritica, 1999) un lenguaje de programación es un conjunto de instrucciones entendibles y ejecutables por una computadora. Al profundizar sobre este aspecto, Mitchell (Mitchell, 2002) considera que el lenguaje de programación deberá facilitar a los desarrolladores la confección de programas de manera concisa y rápida. En su evaluación, se debe tener en cuenta el soporte a cada parte del ciclo de vida del software y debido a la necesidad de mejora y mantenimiento en el período de explotación, debe ayudar a otros programadores a comprender el funcionamiento de la solución.

La elección del lenguaje de programación está determinada por la necesidad de una herramienta que se adapte a las heterogeneas condiciones de los doctorandos cubanos en sus estaciones de trabajo para la realización de sus informes de tesis, regidos por las normas establecidas por la CNGC.

Según las consideraciones de (Knuth, 1986), (Lamport, 1994), (Cascales et al., 2000) y (Alexander Borbón y Walter Mora, 2016) acerca de las potencialidades de TEX y sobre todo de LATEX, y las bondades relacionadas con el hecho de no ser softwares privativos y si ser sistemas de composición tipográficos y de textos que permiten manejar separadamente el contenido del formato, hacen que estos softwares sean los escogidos como lenguajes de programación para el desarrollo que se propone en esta tesis. Existe mucha información sobre los disímiles paquetes que se pueden utilizar para incorporarar funcionalidades a estas aplicaciones y está muy bien organizada por las distintas comunidades ((CervanTEX, 2020), (CTAN, 2020), entre otras) que mejoran y amplian sus códigos. además esta información es de acceso fácil y gratuito.

1.2.3. ENTORNO DE DESARROLLO INTEGRADO (IDE)

Las observaciones de (Matellán, 2004) y (Rouse, 2007) señalan que un entorno de desarrollo integrado (IDE, por sus siglas en inglés) es un programa compuesto por un conjunto de herramientas que proveen facilidades a los programadores para agilizar el proceso de desarrollo de software. Consta de un editor de código, un compilador, un depurador y en algunas ocasiones un constructor de interfaz gráfica.

TeXstudio es un IDE de LATEX que proporciona un soporte moderno de escritura, como la corrección ortográfica interactiva, plegado de código y resaltado de sintaxis. Es multiplataforma, conservando su apariencia y funcionalidades en todos los sistemas operativos en donde se puede ejecutar. (TEX studio, 2020)

A pesar de que TeXstudio no contiene un compilador propio detecta automaticamente un grupo de compiladores entre los más utilizados MikTEX y TEXLive. Otras funcionalidades que unido con las anteriores hacen que se prefiera el uso de TeXstudio para el desarrrollo de esta tesis son:

- Autocompletado de comandos LATEX.
- Coloreado de sintaxis y resaltado de paréntesis.
- Acceso directo a muchas etiquetas de LATEX.
- Vista estructural.
- Resaltado de sintaxis TEX avanzada.
- Visualización clara de los errores y advertencias de LATEX (en editor y como lista).
- Visor incorporado en pdf.

1.2.4. CONTROL DE VERSIONES

Según (Somasundaram, 2013) y (Tsitoara, 2020), un sistema de control de versiones es capaz de registrar los cambios realizados en un archivo o un conjunto de archivos durante un período de tiempo de tal manera que nos permita retroceder en el tiempo desde el futuro para recuperar una versión específica de ese archivo, se llama control de versión sistema. Para darle una explicación más formal, un sistema de control de versiones es un paquete de software que, cuando se inicia, monitoreará sus archivos en busca de cambios y le permitirá etiquetar los cambios en diferentes niveles para que pueda volver a visitar esas etapas etiquetadas cuando sea necesario. Además un elemeto muy importante en la actualidad es la posibilidad de interactuar

varias personas con los archivos de un mismo desarrollo de software al mismo tiempo, sin que se solape o pierda información. Esta posibilidad permite que la interacción del equipo de desarrollo con el cliente sea más fluida y no sea necesario concertar reuniones presenciales.

Por su amplio y reiterado uso y demostrado desempeño para acompañar y controlar las versiones del presente desarrollo se utilizará GIT en su vesión 2.22.0. (Ahmad y Belanger, 2019) (Santacroce, 2017)

1.3. ESTRATEGIA DE PRUEBAS

La gestión de la calidad es una actividad que se aplica a lo largo del proceso de software. El cumplimiento de la especificación del programa y la satisfacción de las expectativas del cliente deben estar en la mira del equipo de desarrollo cada momento, a lo que usualmente suele llamarse verificación y validación.

Las pruebas de software constituyen el último bastión en el aseguramiento de la calidad, son un conjunto de actividades planeadas con anticipación y realizadas de forma sistemática, con el fin de descubrir resultados distintos al esperado; de ahí la necesidad de contar con una estrategia de pruebas desde los primeros momentos del proceso de software.

Reconocidos autores (Pressman, 2005); (Sommerville, 2007) tienden a coincidir en la definición de los siguientes niveles de pruebas, adaptables a las condiciones del desarrollo en cuestión:

Pruebas unitarias: se concentran en el diseño y comportamiento de cada componente del software, desde la perspectiva de su implementación.

Pruebas de integración: prueban la correcta relación entre los componentes del software.

Pruebas de aceptación: evalúan el cumplimiento de los requisitos pactados con el cliente.

Pruebas de sistema: se ejecutan en un ambiente similar al ambiente operacional real, probando el software como un todo de conjunto con el resto de los elementos del sistema.

En la metodología SCRUM las pruebas son tan importantes como la programación, fortalecen la confianza del cliente en el equipo de desarrollo y dentro de este. Los máximos exponentes de esta metodología (Beck, K. y C. Andres, 2004), proponen dos principios para aumentar la rentabilidad de las pruebas: el chequeo doble y el incremento del costo de los defectos.

El chequeo doble consiste en la aplicación de dos conjuntos de pruebas: uno escrito desde la perspectiva de los programadores y orientado a probar exhaustivamente cada uno de los componentes del sistema, y el otro escrito desde la perspectiva de los clientes o usuarios finales, para probar el funcionamiento del sistema como un todo. Lo cual es coherente con los niveles de pruebas de unidad y aceptación respectivamente.

Por otro lado, el costo de los defectos se incrementa con el tiempo, de ahí la máxima de corregirlos tan pronto como sean descubiertos, lo cual implica ejecutar pruebas a cada uno de los componentes una vez que están listos y luego con relativa frecuencia. De lo anterior se infiere que las mismas personas que cometen los errores (los programadores), deben escribir las pruebas de unidad, que a su vez deben ser automatizadas y siempre que sea posible, escritas antes de la implementación, evitando así que se vean influenciadas por

la misma. Para la validación del cumplimiento de las expectativas del cliente, así como el descubrimiento de errores que solo los usuarios finales son capaces de detectar, resultan de vital importancia las pruebas de aceptación, usualmente pruebas alfa y beta.

Las pruebas alfa las realiza el usuario en un entorno controlado por el equipo de desarrollo, mientras los desarrolladores registran los errores y problemas detectados (Pressman, 2005).

Las pruebas beta se aplican en el lugar y condiciones reales en que será ejecutado el software, en ausencia de los desarrolladores. El usuario final registra los problemas detectados (reales o imaginarios) y los informa regularmente a los desarrolladores (ibíd.).

1.4. CONCLUSIONES PARCIALES

El estudio de la bibliografía y análisis de las soluciones informáticas existentes en el área del proceso de elaboración de informes de Tesis basados en plantillas predefinidas, suscitó una elevada comprensión del estado del arte en el área del conocimiento en cuestión, aportando los elementos teóricos que fundamentan el presente Trabajo de Diploma. Entre ellos:

- Para la obtención del grado científico de Doctor en determinada ciencia es indispensable elaborar un informe de Tesis donde queden recogidos todos los elementos escenciales de la investigación.
- En Cuba existe la CNGC, que norma y rige todo el proceso doctoral. En particular la elaboración de estos informes.
- El proceso de elaboración de estos informes es sumamente tedioso, toda vez que no se cuenta con una plantilla y un software que permita dedicar menos tiempo a cuestiones de formato y más tiempo a elaborar el contenido en sí.
- La concepción de una plantilla y la utilización del LaTeX, es resultado de una minuciosa revisión bibliográfica, el análisis de soluciones existentes y esencialmente, el estudio de las particularidades del proceso de elaboración de informes de tesis de grado doctoral.

Resultando:

Metodología de desarrollo: SCRUM
Lenguaje de programación: LaTeX y TeX.
Entorno de desarrollo integrado: TeXstudio
Compilador: TeXLive
Control de versiones: GIT

CAPÍTULO 2

PROPUESTA DE SOLUCIÓN

El presente capítulo recoge las principales características de la propuesta de solución. Se analizan las técnicas de captura de requisitos y presentan las historias de usuario que se derivan de estos requisitos. Además se realiza una descripción detallada de la solución propuesta con las herramientas y tecnologías a utilizadas.

2.1. DESCRIPCIÓN DE LA SOLUCIÓN

Para dar solución a la problemática planteada se decidió la implementación de una plantilla LaTeX para dar formato a los informes de tesis de grado científico en Cuba. Esta solución implica el uso de LaTeX para la elaboración de dichos informes. En tal sentido es recomendable que se utilice el editor TeXstudio para la redacción de estos documentos formateados por la plantilla LaTeX desarrollada.

Una plantilla como la que se pretende elaborar, para seguir los estándares actuales, se construye apartir de dos ficheros. Uno con extensión **ins** y otro con extensión **dtx**. Estos ficheros continen texto plano, comandos de LaTeX y TeX.

En el fichero con extensión **dtx** está empaquetado todo lo necesario para generar otros ficheros como el fichero de clase (**cls**), ficheros de imágenes para logos etc (**eps**), un README (**txt**) con algunas informaciones generales de la plantilla, un fichero de documentación de la plantilla (**pdf**) y ficheros de estilo (**sty**) y ejemplos (**tex**) si son necesarios.

Por otra parte, en el fichero con extensión **ins** aparecen todas las ordenes para desempaquetar del fichero con extensión **dtx** los archivos anteriormente descritos y así quede instalada la clase o plantilla en el ordenador.

Ahora para utilizar la plantilla una vez instalada se debe pasar como parámetro al comando \documentclass{ }, el nombre del archivo cls. En este caso quedaría \documentclass{drcngc}. Hecho esto, y para compilar el informe de tesis escrito con esta plantilla (obtener el archivo pdf) se deben seguir los siguientes pasos:

Compilar con la extensión PDFLaTeX permite generar la versión pdf del documento pero no gestiona ni

las referencias bibliográficas ni los glosarios. (F6[1])

Compilar con el programa Biber se encarga de forma general de gestionar todo lo referente con la bibliografía: formato de la lista de referencias, las citas etc. (F8)

Compilar con el programa makeglossaries para incorporar todos los glosarios: de términos, acrónimos etc. (F9)

Compilar con la extensión PDFLaTeX esta segunda compilación con PDFLaTeX permite finalmente obtener la versión más reciente del documento con todos los enlaces, índices, referencias bibliográficas, glosarios etc. (F6 o F5 para que además de compilar muestre el pdf)

Es importante aclarar que todas estas compilaiones se realizan desde el IDE.

2.2. INGENIERÍA DE REQUISITOS

La ingeniería de requisitos constituye un elemento fundamental en todo proceso de desarrollo de software que tenga como finalidad la obtención de sistemas informáticos que se ajusten a las necesidades reales de los clientes.

Es un proceso centrado en el cliente y sus necesidades, con el objetivo de establecer los servicios que el sistema deberá proveer y las restricciones bajo las cuales deberá operar y ser desarrollado (Báez, G. y S. B. Brunner, 2001).

2.2.1. OBTENCIÓN DE REQUISITOS

La captura de requisitos es la etapa de mayor interacción con el cliente, en ella el equipo de desarrollo busca comprender las necesidades que debe cubrir el sistema, apoyándose en toda fuente de información disponible. (ibíd.)

El propósito de la captura de requisitos es ganar conocimientos relevantes del problema con el objetivo de producir una especificación formal del software. La ingeniería de requisitos ha trabajado por años en el desarrollo de técnicas que permitan realizar este proceso de forma eficiente y precisa. Como parte del análisis previo al diseño de la propuesta de solución, se emplearon diferentes técnicas de captura de requisitos.

La **entrevista** es una técnica muy aceptada dentro de la ingeniería de requisitos. Su aplicación permite obtener una amplia visión de las necesidades del usuario, proporcionando una mejor comprensión de los objetivos de la solución buscada (Escalona y Koch, 2002).

Se realizaron entrevistas a Doctores en Ciencias de diferentes especialidades y con experiencia en el rol de tutor de tesis de grado científico, así como a especialistas en la realización de plantillas utilizando las herramientas propuestas en este informe. Inicialmente estas entrevistas fueron no estructuradas, para un conocimiento general del dominio del problema. Seguidamente, se hizo uso de una combinación de entrevistas estructuradas y no estructuradas con el fin de profundizar en los conocimientos adquiridos (Consultar Apéndice A).

[1] Estas teclas de atajo están asociadas al uso de TeXstudio

Las entrevistas realizadas, arrojaron información valiosa acerca de las necesidades y perspectivas de utilización de aplicaciones que no sean WYSIWYG, que permitan gestionar por separado el contenido del formato (como el IDE escogido), resaltando la importancia de desarrollar una plantilla encargada de la gestión automatizada del formato que sea sencilla, amigable y fácil de utilizar. Se puso en evidencia el uso de varias normas bibliográficas por los diferentes tribunales de tesis de grado científico y por lo tanto la necesidad de establecer opcionalmente el uso de cualquiera de ellas. También se concretó la necesidad del uso de herramientas multiplataforma dado el arraigo del uso de Windows como sistema operativo y el llamado institucional a contribuir a la soberanía tecnológica utilizando software libre. Además se identificaron algunas funcionalidades que permiten viabilizar el proceso de revisión por parte de tutores y oponentes.

Por otro lado, la **tormenta de ideas** es un proceso interactivo no estructurado de grupo, donde las opiniones individuales se analizan y estudian en colectivo. Es útil en la generación de una amplia variedad de puntos de vista sobre el problema, así como su formulación de formas diferentes, superando muchas de las barreras de comunicación en la captura de requisitos. (Pytel, P. et al., 2011)

En varias sesiones, se desarrollaron tormentas de ideas con la participación del equipo de desarrollo y usuarios potenciales de la plantilla. Como resultado se lograron consolidar y aclarar las funcionalidades identificadas en las entrevistas.

Estas técnicas se complementaron con el **análisis de tendencias y tecnologías** más comunes en el desarrollo de este tipo de sistemas, así como el **estudio de la documentación referente al tema**.

2.2.2. REQUISITOS FUNCIONALES

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, la forma en que debe reaccionar ante ciertas entradas y cómo se debe comportar en situaciones particulares. (Sommerville, 2007)

En el análisis realizado, se identificaron los siguientes requisitos funcionales:

RF-01 Establecer la estructura del documento, en cuanto a las secciones que lo componen.
RF-02 Generar la portada del documento.
RF-03 Generar los índices.
RF-04 Generar formato de los títulos de las secciones (capítulos y todos sus acápites).
RF-05 Establecer el tamaño de letra.
RF-06 Establecer el espaciado.
RF-07 Establecer los márgenes.
RF-08 Delimitar la cantidad de páginas.
RF-09 Subrayar los términos que aparezcan en otro idioma.
RF-10 Definir la forma de paginación.
RF-11 Definir el uso de la sangría.
RF-12 Establecer como se imprimen los números.
RF-13 Generar formato de las referencias bibliográficas.

RF-14 Generar formato de tablas.
RF-15 Generar formato de figuras.
RF-16 Establecer el tipo y tamaño de papel.

2.2.3. REQUISITOS NO FUNCIONALES

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Se aplican al sistema en su totalidad y no se refieren directamente a funciones específicas, sino a sus propiedades emergentes; la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento son algunas de ellas. Pueden estar dirigidos incluso al proceso de desarrollo, especificando herramientas o estándares de calidad a emplear en función de las necesidades del usuario. (Sommerville, 2007)

A continuación se enuncian los requisitos no funcionales identificados en la propuesta de solución:

Requisitos de hardware:

RNF-1 Recursos tecnológicos.

Teniendo en cuenta la base tecnológica que como generalidad está en manos de los doctorandos cubanos y las demandas de TEXLive y TEXstudio como compilador y editor respectivamente para la generación de los informes de tesis de grado científico en Cuba las prestaciones mínimas de hardware garantizadas por el equipo de desarrollo son las siguientes:

- Procesador: 1.6 GHz. o superior.
- Memoria RAM: 512 MB o superior.
- Disco duro: 40 GB o superior.

Requisitos de software:

RNF-2 Entorno de ejecución.

Para la ejecución de la aplicación, con independencia del sistema operativo, constituyen premisa indispensable las siguientes instalaciones: :

- Editor: TEXstudio, versión 2.10.4 o superior.
- Compilador: TEXLive, versión 2015 o superior.

Además, los siguientes paquetes del repositorio de LATEX son estrictamente requeridos para el correcto funcionamieto de la plantilla. Para algunos paquetes la versión mínima está indicada entre corchetes. Si el repositorio de paquetes de TEXLive está actualizado quizás no sea necesario actualizar manualmente estos paquetes.

- algorithm
- algpseudocode
- amsmath
- amssymb
- amsthm
- appendix
- aurical
- babel [2014/09/25]
- biber [2]
- biblatex [2014/06/25]
- bigstrut
- bookmark
- calligra
- caption [2013/05/02]
- csquotes
- datetime
- draftwatermark [2012/01/10]
- enumitem
- environ
- epstopdf
- etoolbox
- fancyhdr
- fncychap
- fontenc
- fp
- geometry
- glossaries [2015/03/16]
- graphicx
- hhline
- hyperref
- ifdraft
- ifthen
- inputenc
- kbordermatrix
- lipsum
- listings
- longtable
- mdframed [2013/07/01]
- multirow
- pdflscape
- pdfpages
- rotating
- setspace
- soul
- subcaption
- textpos
- tocloft
- todonotes [2012/07/25]
- txfonts
- upquote
- xcolor
- xkeyval

2.2.4. ESPECIFICACIÓN DE REQUISITOS FUNCIONALES

La especificación de requisitos establece la base para el acuerdo entre usuarios y desarrolladores de software, quedando definido el comportamiento deseado del producto. (IEEE, 2004)

En el entorno de SCRUM, las historias de usuario constituyen el artefacto utilizado para describir las funcionalidades del sistema. Contienen una breve descripción del comportamiento del sistema desde la perspectiva del usuario y representan un medio de comunicación entre el mismo y el equipo de desarrollo. A continuación, se presentan las historias de usuario correspondientes a tres de los requisitos funcionales identificados (Tablas 2.1, 2.2 y 2.3).

Tabla 2.1. Historia de usuario # 1

Historia de usuario	
Número: 1	**Nombre:** Generar los índices
Usuario: Todos	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Wendy Díaz Ortiz	

Continúa en la próxima página

[2] Biber no es un paquete, pero para usuarios de BibTEX es necesario que lo reemplacen por BibLATEX.

Tabla 2.1. Continuación de la página anterior

Descripción: Permite al usuario que se generen automatizadamente los índices, con todos los requerimientos exigidos por la CNGC. Los índices que se podrán generar de esta forma son: Indice general, Indice de Tablas, Indie de Figuras, Indice de Algoritmos e Indice de Códigos.
Observaciones:

Tabla 2.2. Historia de usuario # 2

Historia de usuario	
Número: 2	**Nombre:** Establecer los márgenes
Usuario: Todos	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 5	**Iteración asignada:** 2
Programador responsable: Wendy Díaz Ortiz	
Descripción: Permite al usuario que se generen automatizadamente los márgenes estipulados por la CNGC.	
Observaciones:	

Tabla 2.3. Historia de usuario # 3

Historia de usuario	
Número: 3	**Nombre:** Delimitar la cantidad de páginas
Usuario: Todos	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Bajo
Puntos estimados: 8	**Iteración asignada:** 3
Programador responsable: Wendy Díaz Ortiz	
Descripción: Permite generar un aviso automático cuando la cantidad de páginas exceda lo estipulado por la CNGC.	
Observaciones:	

2.2.5. PILA DEL PRODUCTO, FASES Y OTROS ARTEFACTOS SEGÚN SCRUM

Para entender todo el proceso de desarrollo utilizando Scrum es necesario conocer que de manera general esta metodología propone la planificación de tres fases o **reuniones**. Estas reuniones también forman parte de los artefactos de la metodología.

La columna vertebral de estas reuniones y fases lo constituye el Backlog o Pila del producto. Este es un documento en el que se reflejarán los requisitos del sistema por prioridades. Cómo no se pueden abordar todos los requisitos al mismo tiempo, incluso porque puede haber dependencias entre estos, existen las iteraciones o Sprint. Estas iteraciones serán incrementales, o sea, en cada una se asumen los pendientes de

las anteriores, así como los cambios propuestos a última hora por el cliente y otro grupo de requisitos. Esto hace que desde un inicio no se puedan proponer las funcionalidades a desarrollar en cada Sprint.

Después de mencionados estos elementos medulares de la metodología se pueden definir las fases como sigue:

1. **Planificación del Backlog o Pila del Producto:** en esta fase además de generar la Pila del Producto se define la planificación de la **iteración 0**, decidiéndose cuál será su objetivo y las tareas a realizar. Es así como se obtiene una Pila de la iteración.
2. **Seguimiento de la iteración:** En esta fase se realizan reuniones diarias para evaluar el avance de las tareas de la iteración. Para esto se pueden utilizar las siguientes preguntas:
   - ¿Qué trabajo se realizó desde la reunión anterior?
   - ¿Qué trabajo se hará hasta una nueva reunión?
   - Inconvenientes que han surgido y que hay que solucionar para poder continuar.
3. **Revisión de la iteración:** Se realiza una revisión del incremento que se ha generado y se presenta una nueva versión al cliente.

Es importante comprender que estas tres fases se repiten hasta la conclusión del proyecto. Con la diferencia de que en la primera se genra la Pila del Producto solo al inicio del desarrollo. Esto no significa que sobre la marcha esta Pila pueda ser modificada, todo lo contrario, cualquier miembro del equipo puede hacerlo en cualquier momento. A continuación se presenta la Pila del presente desarrollo de software:

Tabla 2.4. Pila del Producto

ID	Nombre	Importancia [3]	Tiempo estimado [4]	Cómo probarlo	Notas
1	Establecer la estructura del documento, en cuanto a las secciones que lo componen.	100	10	Generar las cabeceras de capítulos y otras secciones y compilar con la extensión PDFLaTeX.	
2	Generar la portada del documento.	90	9	Entrar los campos necesarios para completar una portada y compilar con la extensión PDFLaTeX	
3	Generar los índices.	70	7	Generar las cabeceras de capítulos y otras secciones y compilar con la extensión PDFLaTeX.	

Continúa en la próxima página

[3] El rango de importancia definido fue de 10 a 100

[4] Se mide en dias/hombre. Este tiempo se irá ajustando a lo largo del desarrollo

Tabla 2.4. Continuación de la página anterior

Concluye **SPRINT 1**			16		
4	Generar formato de los títulos de las secciones (capítulos y todos sus acápites).	50	5	Generar las cabeceras de capítulos y otras secciones y compilar con la extensión PDFLaTeX.	
5	Establecer el tamaño de letra.	20	2	Utilizar el paquete Lipsum para generar texto (\lipsum[4-57] por ejemplo) y compilar con la extensión PDFLaTeX.	
6	Establecer el espaciado.	20	2	Utilizar el paquete Lipsum para generar texto (\lipsum[23] por ejemplo) y compilar con la extensión PDFLaTeX.	
7	Establecer como se imprimen los números.	30	3	Escribir números del 1 al 9 con símbolos indo-arábigos en un renglón y en otro renglón la secuencia 2, tres, 10, 9 y en otro renglón el número 12324.5 y la fecha 2 de mayo de 2000 y al compilar con la extensión PDFLaTeX los números del primer renglón deben aparecer con palabras, la secencia del segundo con símbolos y en el tercer renglón debe mostrarse lo siguiente: 12 324,5 y 2 de mayo de 2000	
8	Establecer los márgenes.	80	8	Utilizar el paquete Lipsum para generar texto (\lipsum[4-57] por ejemplo) y compilar con la extensión PDFLaTeX.	
Concluye **SPRINT 2**			20		
9	Delimitar la cantidad de páginas.	100	10	Utilizar el paquete Lipsum para generar texto (\lipsum[1-1000] tres veces seguidas), compilar con la extensión PDFLaTeXy observar que la marca de agua de borrador cambia de color a rojo y muestra el texto "Máximo de pág sobrepasado".	
10	Subrayar los términos que aparezcan en otro idioma.	50	15	Escribir palabras en otros idomas y compilar con la extensión PDFLaTeX.	
Concluye **SPRINT 3**			25		
11	Definir la forma de paginación.	100	15	Utilizar el paquete Lipsum para generar texto (\lipsum[1-1000]) en los diferentes niveles de secciones y compilar con la extensión PDFLaTeX	

Continúa en la próxima página

Tabla 2.4. Continuación de la página anterior

12	Definir el uso de la sangría.	20	2	Utilizar el paquete Lipsum para generar texto (\lipsum[23] por ejemplo) y compilar con la extensión PDFLaTeX.	
Concluye **SPRINT 4**			17		
13	Generar formato de las referencias bibliográficas.	100	20	Compilar con el programa Biber	
Concluye **SPRINT 5**			20		
14	Generar formato de tablas.	30	3	Insertar una tabla y compilar con la extensión PDFLaTeX.	
15	Generar formato de figuras.	30	3	Insertar una figura y compilar con la extensión PDFLaTeX.	
16	Establecer el tipo y tamaño de papel.	50	5	Compilar con la extensión PDFLaTeX.	
Concluye **SPRINT 6**			11		

2.2.6. ESTÁNDARES DE CODIFICACIÓN

En función de la homogeneidad en la implementación, se definió un conjunto de pautas a seguir durante la codificación.

Delimitación del código necesario para abordar cada requisito: En el momento en que se comience a abordar cada requisito utilizar un comentario para especificar cual es.

```
6
7   %Tamaño de letra e interlineado
8
9   \renewcommand{\normalsize}{\fontsize{12pt}{24pt}\selectfont}
10
```

Figura 2.1. Estándar de codificación para delimitar el código necesario para abordar un requisito.

Llamada a un paquete: Cuando se requeriera la utilización de algún paquete, escribir un comentario en la misma línea de código especificando brevemente para qué se utiliza el paquete.

```
15  \RequirePackage{ifthen} %para poder usar el comando \ifthenelse{}{}{}
```

Figura 2.2. Estándar de codificación para comentar la llamada a un nuevo paquete.

Indentación: Los lenguajes LaTeXy TeXno obligan al usuario a identar el código y el IDE escogido tampoco. Sin embargo la correcta indentación hace que el código sea más legible.

```
94          \if@mainmatter
95            \DOTI{#1}%
96          \else%
97            \DOTIS{#1}%
98            \vspace*{1cm}%
99          \fi
```

Figura 2.3. Estándar de codificación para indentar el código.

2.3. CONCLUSIONES PARCIALES

Con este capítulo, quedaron definidos los elementos esenciales para la construcción de la propuesta de solución. Se especificaron y validaron los requisitos del sistema, realizando una cuidadosa planificación de su implementación en función del esfuerzo necesario y su prioridad para el negocio. Finalmente, se definieron los estándares de codificación que servirán de guía para alcanzar un producto no solo funcional, sino de código legible y escalable.

CAPÍTULO 3

VALORACIÓN DE LA VIABILIDAD DE LA PROPUESTA DE SOLUCIÓN

Una vez detallada la propuesta de solución se procedió a su ejecución. El presente capítulo precisa cada uno de los pasos que desde ese momento tuvo lugar en el afán de continuar y reforzar el aseguramiento de la calidad, iniciado desde la misma planeación del proceso de software. Se ofrece un análisis de cada uno de los niveles de pruebas ejecutados, los términos en que se realizaron las pruebas y los resultados alcanzados.

3.1. TÉCNICAS DE VALIDACIÓN DE REQUISITOS

Con el objetivo de ratificar que los requisitos del software obtenidos definen el sistema que el cliente desea se llevó a cabo un proceso de validación de los mismos, para el cual se emplearon las siguientes técnicas (Sommerville, 2007):

Revisiones de los requisitos: Se realizaron revisiones a cada uno de los requisitos por parte del equipo de desarrollo y por el cliente en la reparación de cada Pila de Sprint. En el primer Sprint las revisiones internas generaron no conformidades de tipo técnicas, las cuales fueron corregidas satisfactoriamente en el segundo. En el Sprint 2 las revisiones internas no generadon no conformidades. Con el cliente se desarrolló 1 revisión por cada Sprint. Después de estas, el mismo quedó satisfecho con el trabajo efectuado por el equipo de desarrollo por lo que se aprobaron finalmente los requisitos.

3.2. MÉTRICAS APLICADAS A LOS REQUISITOS

Con el objetivo de medir la calidad de la especificación de los requisitos se aplicó la métrica Calidad de especificación (CE). Para obtener cuán entendibles y precisos son los requisitos, primeramente, se calcula el total de requisitos de la especificación como se muestra a continuación:

N_r**:** El total de requisitos de especificación.
N_f**:** Cantidad de requisitos funcionales.

N_{nf}**:** Cantidad de requisitos no funcionales.

Como resultado de la sustitución de los valores, para el sistema se obtiene:

$$\begin{aligned} N_r &= N_f + N_{nf} \\ N_r &= 16 + 2 \\ N_r &= 18 \end{aligned}$$

Para determinar, finalmente, la Especificidad de los Requisitos (*ER*) o ausencia de ambigüedad en los mismos se realiza la siguiente operación:

$$ER = \frac{N_{ui}}{N_r}$$

Donde N_{ui} es el número de requisitos para los cuales todos los revisores tuvieron interpretaciones idénticas. Mientras más cerca de 1 esté el valor de *ER*, menor será la ambigüedad. Para el caso de los requisitos obtenidos para la plantilla propuesta solo uno produjo contradicción en las interpretaciones (el requisito funcional 4). Sustituyendo las variables se obtiene:

$$ER = \frac{17}{18} \approx 0.94$$

Arrojando un resultado satisfactorio. Indicando que el grado de ambigüedad de los requisitos es sumamente bajo (6 %) ya que el 94 % es entendible. El requisito ambiguo fue modificado y validado para garantizar su correcta comprensión.

3.3. PRUEBAS DE ACEPTACIÓN

Las pruebas de aceptación tienen como objetivo validar que una funcionalidad ha sido completada satisfactoriamente. En consecuencia, y a diferencia de las pruebas de unidad, en Scrum las pruebas de aceptación no tienen por qué ser 100 % efectivas luego de cada iteración, las correcciones o mejoras en las historias de usuario asociadas a pruebas de aceptación no superadas, forman parte de la planeación de la Pila del siguiente Sprint.(Kniberg, 2015)

Para las pruebas de aceptación, se ejecutaron pruebas alfa luego de cada iteración de desarrollo. Estas pruebas se realizaron en un entorno controlado por el equipo de desarrollo, comprobando la correspondencia entre las funcionalidades del sistema y el comportamiento esperado. Algunos de los resultados obtenidos en estas pruebas se pueden observar en el Anexo B Finalizada cada iteración de desarrollo, se realizaron pruebas a las funcionalidades implementadas, validando unas e identificando no conformidades en otras. En las sucesivas iteraciones, además de las pruebas a las nuevas funcionalidades, se repitieron las pruebas anteriores aumentando su criticidad e incorporando nuevos casos.

En la tabla 3.1 se presenta el diseño del caso de prueba y los juegos de datos empleados en la validación de la Historia de Usuario 7 después de la primera iteración.

Tabla 3.1. **Prueba de aceptación**

CASO DE PRUEBA DE ACEPTACIÓN			
Código: HU_7		**Historia de usuario:** #7	
Nombre: Establecer como se imprimen los números			
Descripción: Permite formatear, según lo dispuesto por la CNGC los números que aparecen en el texto de forma homogénea			
Condiciones de ejecución: • El usuario utiliza la aplicación TEXstudio y carga los archivos que conforma su informe de tesis. • Selecciona para editar un archivo tex que constituye una de las secciones de su tesis.			
Escenarios de prueba		**Flujo del escenario**	**Resultados esperados**
EP1	Codificar los números 1, 3 y 7 con sus respectivos símbolos indo-arábigos.	Se compila con la extensión PDFLATEX	Al generar el pdf se muestran los números escritos con su numeral porque en la serie no hay ningún número mayor que 9.
EP2	Codificar los números 10, 3 y 7 con sus respectivos símbolos indo-arábigos.	Se compila con la extensión PDFLATEX	Al generar el pdf se muestran los números escritos con sus símbolos indo-arábigos porque en la serie hay un número mayor que 9.
EP3	Codificar el número 12,3456 utilizando la coma para separar los dígitos decimales.	Se compila con la extensión PDFLATEX	Al generar el pdf se muestra el número tal y como se codificó, con la coma como separador.
EP4	Codificar el número 12.3456 utilizando el punto para separar los dígitos decimales.	Se compila con la extensión PDFLATEX	Al generar el pdf se muestra el número con la coma como separador de los dígitos decimales.
EP5	Codificar el número 12 3456 utilizando un espacio en blanco para separar las unidades de mil.	Se compila con la extensión PDFLATEX	Al generar el pdf se muestra el número tal y como se codificó, con un espacio separando las unidades de mil.
EP6	Codificar el número 123456 sin utilizar un espacio en blanco para separar las unidades de mil.	Se compila con la extensión PDFLATEX	Al generar el pdf se muestra el número con un espacio en blanco para separar las unidades de mil.

Continúa en la próxima página

Tabla 3.1. Continuación de la página anterior

EP7	Codificar la fecha 30 de julio de 2020 sin utilizar un espacio en blanco para separar las unidades de mil del número que representa el año.	Se compila con la extensión PDFLaTeX	Al generar el pdf se muestra la fecha tal y cómo se codificó, sin utilizar un espacio en blanco para separar las unidades de mil del año.
EP8	Codificar la fecha 30 de julio de 2 020 utilizando un espacio en blanco para separar las unidades de mil del número que representa el año.	Se compila con la extensión PDFLaTeX	Al generar el pdf se muestra la fecha sin utilizar un espacio en blanco para separar las unidades de mil del año.
EP9	Codificar la fecha 30/julio/2020 sin utilizar un espacio en blanco para separar las unidades de mil del número que representa el año.	Se compila con la extensión PDFLaTeX	Al generar el pdf se muestra la fecha tal y cómo se codificó, sin utilizar un espacio en blanco para separar las unidades de mil del año.
EP10	Codificar la fecha 30/julio/2 020 utilizando un espacio en blanco para separar las unidades de mil del número que representa el año.	Se compila con la extensión PDFLaTeX	Al generar el pdf se muestra la fecha sin utilizar un espacio en blanco para separar las unidades de mil del año.

La gráfica de la Figura 3.1, ilustra los resultados de las pruebas de aceptación realizadas al finalizar cada una de las iteraciones en que quedó dividido el proceso de desarrollo.

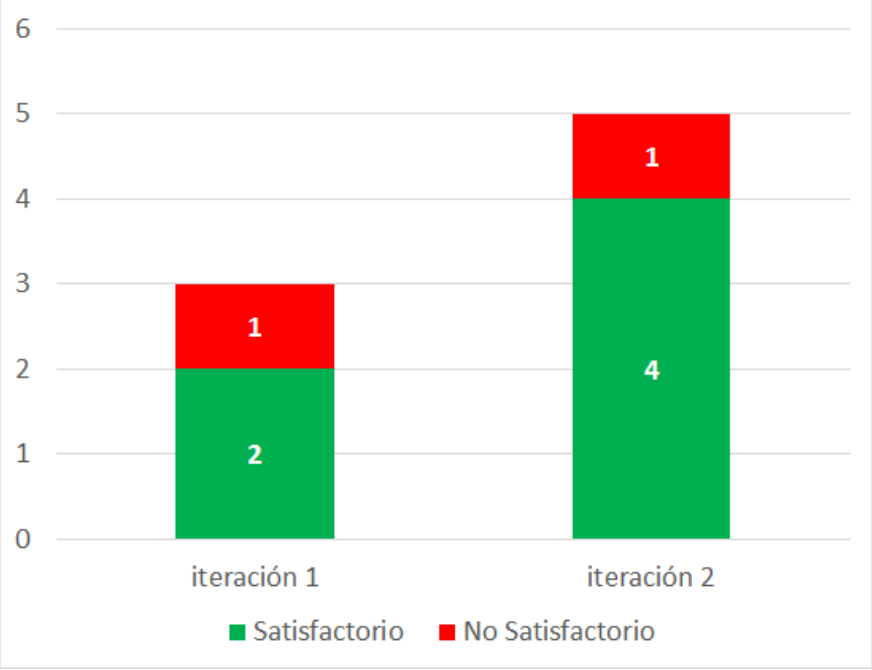

Figura 3.1. Resltados de pruebas de aceptación

En general, se evidencia un ascenso en el número de casos de prueba escritos y ejecutados en cada iteración con respecto a la anterior y una tendencia creciente en la efectividad de las pruebas. En la primera iteración se implementaron y probaron 3 Historias de Usuario, de las cuales 2 resultaron satisfactorias (66 % de efectividad). En la segunda iteración se implementaron las funcionalidades descritas en otras 5 Historias de Usuario y se realizaron correcciones a la que no superó la primera vuelta de pruebas. Se ejecutaron un total de 8 pruebas de aceptación (incluidas las 3 iniciales), el resultado fue de 6 pruebas satisfactorias y 2 no satisfactorias, para un 75 % de efectividad. Todas las no conformidades de la primera iteración fueron subsanadas.

3.4. PRUEBAS DE UNIDAD

Para las pruebas de unidad se utilizó el mismo IDE de desarrollo a partir de las diferentes formas de compilación que posibilita y tomando como guía la Pila del Producto expuesta anteriormente en la Tabla 2.4.

Se realizaron pruebas para los primeros 8 requsitos, que permitieron corregir los errores. Esto fue posible, entre otras a las funcionalidades del IDE de desarrollo relacionadas con la detección y descripción de los errores de codificación. Así se logra tener una versión entregable de la segunda iteración.

3.5. CONCLUSIONES PARCIALES

La aplicación de técnicas de validación de requisitos permitió ratificar que los 16 requisitos obtenidos están en correspondencia con las solicitudes del cliente. Además el uso de la métrica CE proporcionó una medida cuantitativa de la calidad de la especificación de estos, proveyendo las ambigüedades existentes para su corrección. Se ejecutó un total de 8 casos de prueba a medida que se fueron entregando las funcionalidades acordadas al cierre de cada iteración, corrigiendo las no conformidades y repitiendo las pruebas para su validación según la metodología SCRUM. Luego de cada cambio significativo en el sistema, se realizaron pruebas de unidad a través de las diferentes opciones de compilación que ofrece el IDE TEXstudio, exigiéndose un 100 % de efectividad para cada una.

Conclusiones

Con el desarrollo del presente trabajo de diploma se realizaron 2 iteraciones del desarrollo de la "Plantilla LaTeX 2_ε para el informe de tesis de Grado Científico en Cuba", para lo cual:

- El estudio de la bibliografía y análisis de las tecnologías de la información y las comunicaciones aplicadas al procesamiento de informes de tesis de Grado Científico en Cuba, demostró la necesidad de utilizar un procesador donde se pudiera separar la gestión del formato y el contenido. En este caso y por las ventajas expuestas se escogió LaTeX.
- La selección de SCRUM como metodología para guiar el desarrollo de la propuesta de solución, permitió la generación de los artefactos ingenieriles: historias de usuario, pila del producto y reuniones de culminación de iteraciones, utilizados estos como referencias para la implementación.
- La utilización de las herramientas y tecnologías: LaTeX 2_ε y TeX $_{3.14159265}$ como lenguajes de programación, TeXstudio en su versión 2.10.4 como IDE, TeXLive 2017 como compilador, Biber 2.1 como backend del procesador bibliográfico BibLaTeX y GIT 2.22.0 como sistema de control de versiones; permitió establecer la infraestructura tecnológica para el desarrollo de la propuesta de solución.
- La implementación de las funcionalidades definidas para las dos primeras iteraciones permite que los doctorandos cubanos se concentren en el contenido de sus tesis. Gestionando de manera automatizada y sin riesgo de cambios, el formato de sus informes de tesis de grado científico en Cuba.
- Las pruebas realizadas y la aplicación de la técnicas de validación propiciaron determinar la aceptación por parte del cliente de las funcionalidades implementadas, siendo de gran valor para la gestión con calidad y apego a las normas dictadas por la CNGC de los informes de tesis de grado científico en Cuba.

Acrónimos

CE Calidad de especificación. 27, 31

CNGC Comisión Nacional de Grado Científico. 2, 10, 13, 16, 22, 29, 32, 39

IDE entorno de desarrollo integrado. 14, 18, 19, 25, 31, 32

WYSIWYG lo que ves es lo que tienes. 8, 19

Referencias bibliográficas

Ahmad, Jawwad y Chris Belanger (2019). *Mastering Git. Understanding Git Internals and Commands*. raywenderlich Tutorial Team (vid. pág. 15).

Alexander Borbón y Walter Mora (2016). *Edición de Textos Científicos con LaTeX*. 2.ª ed. Instituto Tecnológico de Costa Rica. URL: http://tecdigital.tec.ac.cr/revistamatematica/ (visitado 09-05-2019) (vid. pág. 14).

Báez, G. y S. B. Brunner (2001). *Metodología DoRCU para la Ingeniería de Requerimientos*. Ed. por Instituto Superior Politécnico Jose Antonio Echeverría (vid. pág. 18).

Beck, K. y C. Andres (2004). *Extreme Programming Explained: Embrace Change*. 2.ª ed. Addison Wesley Professional. ISBN: 0-321-27865-8 (vid. págs. 11, 15).

Booch, G., J. Rumbaugh e I. Jacobson (1999). *El Lenguaje Unificado de Modelado/The Unified Software Development Process*. Pearson Education. ISBN: O-201-57169-2 (vid. pág. 11).

Cascales, Bernardo et al., (2000). *Latex, una imprenta en tus manos*. Murcia, España: Aula documental de Investigación (vid. págs. 6, 14).

CervanTEX (2020). *CervanTEX*. URL: http://www.cervantex.es/ (visitado 15-01-2020) (vid. pág. 14).

Charvat, J. (2003). *Project Management Methodologies: Selecting, Implementing, and Supporting Methodologies and Processes for Projects.* New Jersey: John Wiley & Sons, Inc, ISBN: 0471221783 (vid. pág. 11).

CTAN (2020). *Comprehensive TEXArchive Network*. URL: http://www.ctan.org/ (visitado 15-01-2020) (vid. pág. 14).

Educación Superior, Ministerio de (2019a). *Resolución No.138/19*. URL: www.mes.gob.cu/es/resoluciones (vid. págs. 1, 2).

— (2019b). *Resolución No.140/19*. URL: www.mes.gob.cu/es/resoluciones (vid. pág. 1).

Escalona, M. J. y N. Koch (2002). *Ingeniería de Requisitos en Aplicaciones para la Web. Un estudio comparativo.* Universidad de Sevilla (vid. pág. 18).

Google (2020). *GoogleTrends*. URL: www.googletrends.com (visitado 01-04-2020) (vid. pág. 7).

Grados Científicos, Comisión Nacional de (2005). «NORMAS Y RESOLUCIONES VIGENTES PARA EL DESARROLLO DE LOS GRADOS CIENTIFICOS EN LA REPUBLICA DE CUBA» (vid. pág. 2).

Green, David (2016). *Scrum, novice to ninja*. USA: SitePoint. ISBN: 978-0-9943469-1-9 (vid. pág. 13).

IEEE (2004). *Guide to the Software Engineering Body of Knowledge (SWEBOK)* (vid. pág. 21).

Jacobson, I., G. Booch y J. Rumbaugh (2000). *El Proceso Unificado de Desarrollo de Software*. Madrid: Pearson Educación (vid. pág. 11).

Justicia, Ministerio de (2019). «Gaceta Oficial de la República de Cuba». En: *Gaceta Oficial de la República de Cuba* (vid. pág. 2).

Kniberg, Henrik (2015). *Scrum and XP from the trenches. How We Do Scrum*. 2.ª ed. USA: C4Media (vid. págs. 13, 28).

Knuth, Donald (1986). *The TEXbook*. Addison-Wesley (vid. pág. 14).

Lamport, Leslie (1994). *LATEX, A Document Preparation System. User's Guide and Reference Manual*. 2.ª ed. Addison-Wesley (vid. pág. 14).

Letelier, P y M. C. Penadé (2003). *Métodologías Ágiles en el Desarrollo de Software: eXtreme Programming (XP)*. Universidad Politécnica de Valencia (vid. pág. 11).

Malpica, Carlos J. (2014). «APLICACIÓN DE LA METODOLOGÍA SCRUM PARA INCREMENTAR LA PRODUCTIVIDAD DEL PROCESO DE DESARROLLO DE SOFTWARE EN LA EMPRESA CCJ S.A.C. LIMA». Tesis de grado. Facultad de Ingeniería de Sistemas, Universidad Nacional del centro del Perú (vid. pág. 13).

Matellán, V. (2004). *Compilación de ensayos sobre software libre*. Madrid: Dykinson. ISBN: 849772402X (vid. pág. 14).

Mckenna, Dave (2016). *The art of Scrum. How Scrum Masters Bind Dev Teams and Unleash Agility*. CA Technologies. ISBN: 978-1-4842-2277-5. DOI: 10.1007/978-1-4842-2277-5 (vid. pág. 13).

Microsoft (2020). URL: https://products.office.com/word (visitado 05-01-2020) (vid. pág. 2).

Mitchell, J. C. (2002). *Concepts in Programming Languages*. Cambridge University Press. ISBN: 0-511-03492-X (vid. pág. 13).

Morrison, Stanley (1929). *Principos fundamentales de la tipografía* (vid. pág. 5).

Pressman, R. (2005). *Ingeniería de Software. Un enfoque prático.* Mc Graw Hill. ISBN: 9701054733 (vid. págs. 15, 16).

Pytel, P. et al., (2011). *Ingeniería de requisitos basada en técnicas de ingeniería del conocimiento* (vid. pág. 19).

Rouse, M. (2007). *What is integrate development environment (IDE). In Software Quality information, news and tips*. URL: http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment] (visitado 15-01-2020) (vid. pág. 14).

San Martín de Porres (USMP), Universidad de (2016). *Manual para la elaboración de las tesis y los trabajos de investigación*. Lima, Perú (vid. pág. 2).

Santacroce, Ferdinando (2017). *Git Essential*. 2.ª ed. Birmingham: Packt Publishing Ltd. ISBN: 978-1-78712-072-3 (vid. pág. 15).

Somasundaram, Ravishankar (2013). *Git: Version Control for Everyone Beginner's Guide* (vid. pág. 14).

Sommerville, Ian (2005). *Ingeniería del software*. 7.ª ed. Madrid: Pearson Educación. ISBN: 84-7829-074-5 (vid. pág. 12).

— (2007). *Software Engineering*. Pearson Education. ISBN: 7-111-19770-4 (vid. págs. 12, 15, 19, 20, 27).

TEX studio (2020). *TEXstudio*. URL: http://texstudio.sourceforge.net/ (visitado 10-02-2020) (vid. pág. 14).

Trejos Buritica, O.I. (1999). *La esencia de la Lógica de Programación*. ISBN: 958-33-1125-1 (vid. pág. 13).

Tsitoara, Mariot (2020). *Beginning Git and GitHub. A Comprehensive Guide to Version Control, Project Management, and Teamwork for the New Developer*. Madagascar: Springer Science. ISBN: 978-1-4842-5312-0 (vid. pág. 14).

Generado con LATEX: 21 de septiembre de 2020: 11:18pm

Apéndices

APÉNDICE A

ENTREVISTA PARA LA CAPTURA DE REQUISITOS

Objetivo: Identificar los principales requisitos funcionales y no funcionales que debería satisfacer la plantilla de informes de tesis de grado científico en Cuba.

Preguntas:

1. ¿Cuáles son las principales deficiencias del proceso de elaboración de los informes de tesis de grado científico en Cuba?
2. ¿Cuáles documentos del marco legal cubano establecen las normas para la escritura de los informes de tesis de grado científico?
3. ¿Qué ventajas representa la utilización de una herramienta informática para la confección del informe de tesis de grado científico que gestione por separado el contenido y el formato?
4. ¿Sería idóneo la realización de una plantilla que se encargara de gestionar el formato de la tesis automátizadamente? ¿Porqué?
5. ¿Qué funciones, además de las ya implícitas en las normas legales le gustaría que tuviese la plantilla de tesis?
6. ¿Qué funcionalidades relacionadas con el proceso de revisión de los informes de tesis por parte de tutores y oponentes deben ser implementadas?
7. ¿Cuáles normas bibliográficas son utilizadas por los diferentes tribunales de tesis de grado científico en Cuba?
8. ¿Cuáles son las características de la base tecnológica en manos de un doctorando cubano promedio?

APÉNDICE B

RESULTADOS VISUALES DE ALGUNAS PRUEBAS REALIZADAS

En la Figura B.1 se muestra el cumplimiento de algunas de las normativas de la CNGC referida al formato del índice. Estas son:

- Se encabezará con la palabra INDICE (o TABLA DE CONTENIDOS según la preferencia del autor) en mayúsculas sostenidas, debidamente centrada.
- Los títulos correspondientes a los capítulos del texto se escribirán con mayúsculas sostenidas.
- La indicación de la página correspondiente se colocará al margen derecho en forma de columna encabezada por la abreviatura "Pág."
- Los títulos correspondientes a los diferentes acápites en que se divide cada capítulo se escribirán en minúsculas, precedidos del número de orden correspondiente y a dos espacios. Se utilizará una sangría de manera que el numeral aparezca al mismo nivel que comienza el título del capítulo. A la derecha aparecerá indicada su ubicación en la tesis.

En la Figura B.2 se presenta el mensaje de sobrepaso de la cantidad de páginas definida para los informes de tesis de grado científico en Cuba. Dado que está definido en la normativa 6, inciso d de la CNGC que:

- El texto de la tesis tendrá no más de 100 páginas, sin incluir los gráficos, figuras, esquemas, apéndices y la bibliografía.
- Para las Ciencias Sociales y Humanísticas el contenido podrá tener hasta un 20 % más de lo señalado.

Como uno de los parámetros a modificar para la plantilla está `pagelimitmode`. Teniendo como posibles valores `csh` para las Ciencias Sociales y Humanísticas y `oc` para las otras Ciencias.

Para la generación de estas y otras pruebas se utilizó el paquete `blindtext`, en su versión $2 \cdot 0$. Este permite generar un informe con una texto aleatoreo donde se presntan todo tipo de estructuras, viñetas, fórmulas matemáticas, tablas, etc. Permitiendo así probar la plantilla desarrollada.

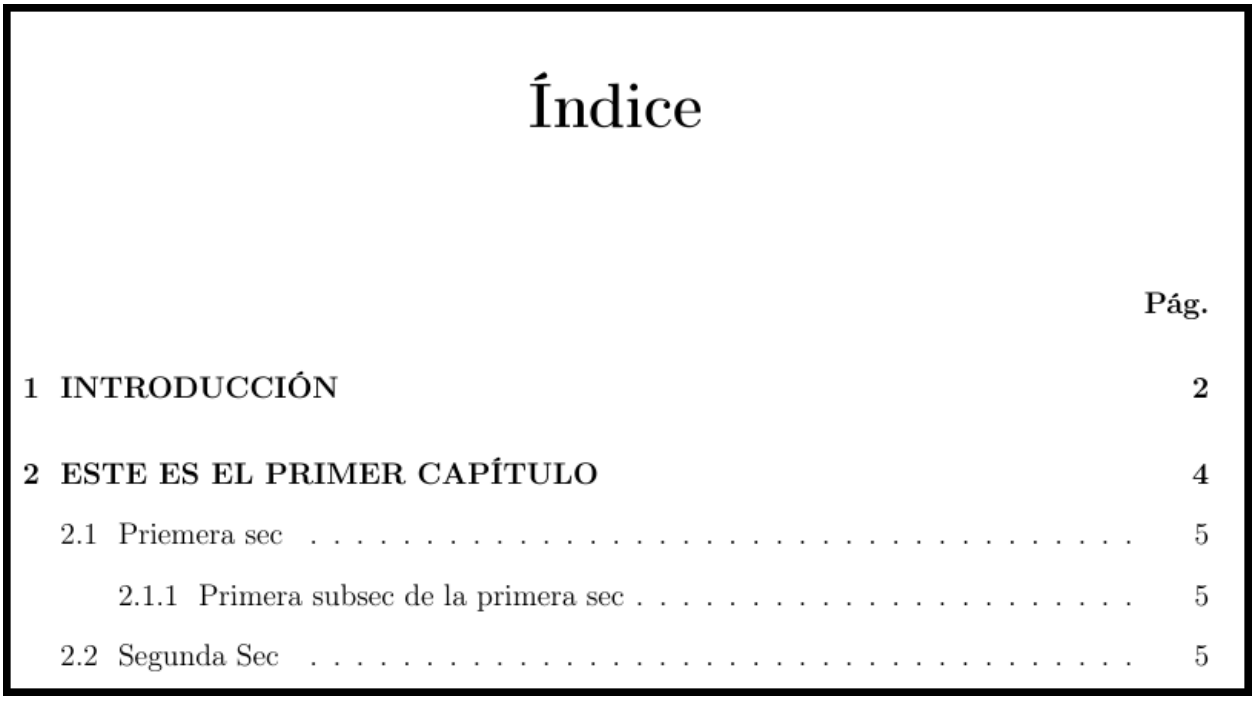

Figura B.1. Fragmento del índice de un informe de tesis generado con la plantilla desarrollada

Figura B.2. Página de un informe de tesis generado con la plantilla desarrollada, donde se observa el mensaje de sobrepaso de la cantidad de páginas definida para estos informes.

Figura B.3. Muestra del uso de la coma como separador decimal y el espacio para separar las unidades de mil.

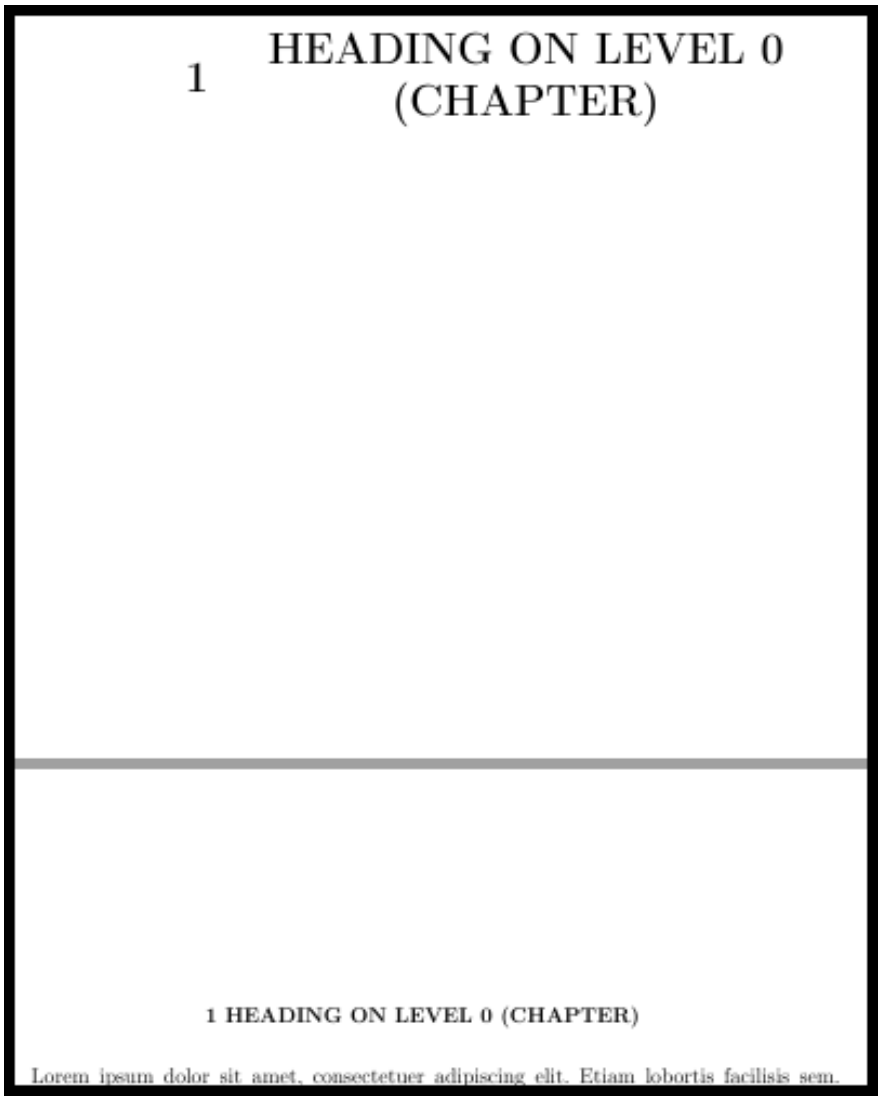

Figura B.4. Muestra del formato de los nombres de los capítulos con la página que lo antecede con el nombre del capiítulo centrado.