MEDESOFT: METODOLOGÍA DE DESARROLLO DE SOFTWARE EN ENTIDADES DE EDUCACIÓN SUPERIOR

Luis García-Paucar

Piura, julio de 2009

FACULTAD DE INGENIERÍA

Maestría en Dirección Estratégica en Tecnologías de la Información

García, L. (2009). *MEDESOFT: Metodología de Desarrollo de Software en entidades de educación superior*. Tesis de maestría en Dirección Estratégica en Tecnologías de la Información. Universidad de Piura. Facultad de Ingeniería. Piura, Perú.

UNIVERSIDAD DE PIURA
FACULTAD DE INGENIERÍA

MEDESOFT: Metodología de Desarrollo de Software en Entidades de Educación Superior

Tesis para optar el grado de Master en Dirección Estratégica en Tecnologías de la Información

Por Luis Hernán García Paucar

Tutor: Ms.Sc. Pedro Chávez

Piura, Julio 2009

AGRADECIMIENTOS

Deseo expresar mi más sincero agradecimiento a:

A la Universidad de Piura y a la Fundación Universitaria Iberoamericana por haberme brindado la oportunidad de realizar esta maestría.

A mi tutor Pedro Chávez, por la confianza que siempre depositó en mí y por su constante apoyo y orientación durante el estudio.

A la Universidad Peruana de Ciencias Aplicadas.

Especial agradecimiento por la ayuda brindada durante el proceso de elaboración de la presente Tesis para:

Mis amigos: Ricardo Saavedra, Miriam Morote, Carlos Zambrano, Gustavo Morales, Julio Pinglo y Julio Cárdenas.

A todos ellos, y a tantos otros que harían interminable la relación, llegue mi más sincero agradecimiento.

DEDICATORIA

A Dios, por ser quien le da sentido a mi vida.

A mi esposa, Norka Mailin, por ser mi dulce inspiración, compañera inseparable y por su paciencia e invalorable apoyo en el logro de mis objetivos.

A mis padres, Blanca Luz y Hernán. Muchas gracias a ambos por sus consejos y buen ejemplo que me permiten hoy poder ser un hombre de bien.

A mis maestros de los diferentes niveles de enseñanza, por haber contribuido a mi formación profesional.

Síntesis

En el Perú, existe un gran potencial de crecimiento en la industria de desarrollo de software, la cual gestionada adecuadamente y utilizando estándares de calidad reconocidos internacionalmente podría ingresar con ventajas competitivas a otros mercados en corto tiempo, convirtiéndose en una nueva fuente de riqueza, debido a que la inversión que requiere el sector, es mínima en comparación con el beneficio que se puede obtener.

Dentro de esta industria, predomina en nuestro país la pequeña y mediana empresa (PYME), siendo aún su proceso de desarrollo de software poco formal. Los modelos que se utilizan habitualmente a nivel internacional como por ejemplo CMMI® [1] y/o las normas de Calidad ISO [2] resultan complejas en su implementación, difíciles de cumplir y de alto costo para las pequeñas y medianas empresas.

Esta realidad, alcanza a las entidades académicas de nuestro país: dentro de las carreras superiores de Computación e Informática, Ingeniería de Sistemas, y similares, generalmente se desarrollan proyectos de desarrollo de software como parte de uno o más cursos. Estos proyectos, muchas veces son relegados al campo meramente académico o si intentan resolver una situación problemática real, carecen de métodos, técnicas y herramientas adecuadas para ser desarrollados e implementados exitosamente.

Conscientes de esta situación y de la importancia de una adecuada articulación entre las entidades de educación superior, las capacidades de sus egresados y las necesidades de desarrollo nacional, se realizó un trabajo de investigación que generó como resultado una metodología propia de desarrollo de software para entidades académicas, basada en estándares reconocidos internacionalmente, a la cual se denominó MEDESOFT. Este trabajo se desarrolló en el Instituto Superior Tecnológico CIBERTEC, unidad tecnológica de la Universidad Peruana de Ciencias Aplicadas - UPC.

---

[1] CMMI: Capability Maturity Model Integration. Ha sido desarrollado por el Software Engineering Institute (SEI) de la Universidad Carnegie Mellon. CMMI es una marca registrada en US Patent and Trademark Office por la universidad Carnegie Mellon.

[2] ISO: Internacional Organization for Standardization. Es la organización no gubernamental más grande en el mundo encargada del desarrollo y publicación de estándares internacionales.

MEDESOFT, implicó inicialmente la revisión de diferentes normas y modelos de procesos tales como CMMI®, ISO/IEC 12207 (en sus enmiendas 1 y 2), MPS.BR de Brasil y MOPROSOFT de México. Se seleccionó como modelo de referencia a MOPROSOFT debido a su compatibilidad con las mejores prácticas internacionales y su facilidad de aprendizaje.

La metodología implementada, adapta el proceso de desarrollo de software del modelo de referencia a una realidad académica en la cual los principales actores son los alumnos. Dentro de este contexto, se definen un conjunto de pasos metodológicos, indicadores, técnicas y herramientas de ingeniería de software que integran transversalmente cursos específicos y en donde alumnos, equipo docente y la empresa cliente asumen roles activos a través de la implementación de un proyecto real de desarrollo de software. La metodología desarrollada se encuentra actualmente en su primera versión y es parte de un proceso de mejora continua, el cual garantiza la evolución constante de sus procesos individuales y de la metodología en su conjunto.

Inicialmente, ha permitido evaluar el proceso actual de desarrollo de software por alumnos de la carrera de Computación e Informática de CIBERTEC, evidenciando necesidades de mejora, así como el progreso de sus alumnos en el desarrollo de capacidades relacionadas con la Ingeniería de Software que deben ser adquiridas a lo largo de la carrera.

Se proyecta utilizar esta metodología en sus futuras versiones, primero dentro de las carreras de las escuelas de Tecnología y de Gestión de Negocios de CIBERTEC, para luego hacerlo con diferentes organizaciones académicas pertenecientes a la red LAUREATE INTERNATIONAL UNIVERSITIES[3].

[3] Laureate International Universities: Red privada internacional de entidades académicas a la cual pertenecen en el Perú CIBERTEC, UPC y la Universidad Privada del Norte.

Índice

CONTENIDOS **PÁG.**

Índice de Figuras

Índice de Cuadros

Glosario

Proceso: Conjunto de prácticas relacionadas entre sí, llevadas a cabo a través de roles y por elementos automatizados, que utilizando recursos y a partir de insumos producen un satisfactor de negocio para el cliente.

Objetivo: Fin a que se dirige o encamina una acción u operación.

Indicador: Mecanismo que sirve para mostrar o significar una cosa con evidencias y hechos.

Rol: Es responsable por un conjunto de actividades de uno o más procesos. Un rol puede ser asumido por una o más personas de tiempo parcial o completo.

Producto: Cualquier elemento que se genera en un proceso.

Producto Software: Es el producto que se genera en el proceso de desarrollo de Software. Los productos de software se clasifican de manera general como Especificación de Requerimientos, Análisis y Diseño, Software, Prueba, Registro de Rastreo y Manual. Esta clasificación puede ser especializada según las necesidades, por ejemplo, Prueba puede significar Plan de Pruebas o Reporte de Pruebas, Manual puede ser especializado en Manual de Usuario, Manual de Operación o Manual de Mantenimiento, mientras que el Software puede ser un Componente, un sistema de Componentes o un Sistema compuesto de sistemas.

Configuración de Software: Es un conjunto consistente de productos software.

Actividad: Conjunto de tareas específicas asignadas para su realización a uno o más roles.

Verificación: Actividad para confirmar que el producto refleja propiamente los requerimientos especificados para él.

Validación: Actividad para confirmar que el producto resultante es capaz de satisfacer los requerimientos para su aplicación especificada o uso previsto.

Flujo de Trabajo: Esquema que expresa las relaciones entre las actividades de un proceso. Una relación puede ser secuencial, paralela, cíclica, de selección o anidada.

Guía de Ajuste: Modificación a las prácticas, entradas y salidas de un proceso, siempre y cuando no afecte al cumplimiento de sus objetivos.

Gestión: Hacer diligencias conducentes al logro de un negocio.

Organización: Empresa o área interna de una organización dedicada al desarrollo y/o mantenimiento de software.

Infraestructura: Conjunto de elementos o servicios que se consideran necesarios para la creación y funcionamiento de una organización.

Base de Conocimiento: Es un repositorio de todos los productos tales como productos de software, planes, reportes, registros, lecciones aprendidas y otros documentos.

Situación Excepcional: Circunstancia que impide el desarrollo de una actividad.

Lección Aprendida: Experiencia positiva o negativa obtenida durante la realización de alguna actividad.

Plan: Programa detallado de las actividades, responsables por realizarlas y calendario.

Reporte: Informe del resultado de las actividades realizadas.

Registro: Evidencia de actividades realizadas.

Introducción

En general, las entidades académicas de Educación Superior en el Perú, carecen aún dentro de sus centros de investigación y carreras afines a la industria del software, de una metodología adecuada para gestionar e implementar proyectos de desarrollo de software, desarrollados por alumnos, dentro de sus cursos regulares, y producir software de calidad como parte de un proceso definible, repetible y susceptible de ser medido.

El Proyecto de Investigación presenta la solución a la situación problemática anteriormente mencionada y consta de cinco capítulos, resultados, conclusiones, recomendaciones, bibliografía y anexos.

En el Capítulo 1 se comenta la situación problemática existente, con especial énfasis en las carreras de Computación e Informática, Ingeniería de Sistemas y similares para luego detallar la justificación y el alcance del proyecto de investigación.

El Capítulo 2 define el marco conceptual necesario para comprender la metodología implementada en el presente trabajo. Se destacan conceptos de Calidad, Calidad de Software, Procesos de Software y Ciclo de Vida del Software.

El Capítulo 3 realiza un análisis de las principales normas y modelos de procesos de software existentes actualmente en el mercado. Hecho el análisis y en base a criterios de evaluación predefinidos, se selecciona el modelo que servirá de modelo de referencia para implementar la metodología académica de gestión y desarrollo de software.

El Capítulo 4 describe el proceso de implementación de la Metodología Académica de Desarrollo de Software MEDESOFT. Se detalla el planeamiento estratégico producto del cual se genera la metodología. Se describe también el plan táctico, identificando las fases de implementación de MEDESOFT. Finalmente, se define el proceso de Desarrollo de Software por Alumnos, identificando sus objetivos, actividades e indicadores, así como técnicas y herramientas específicas que contribuirán a alcanzar sus objetivos.

El Capítulo 5 presenta un ejemplo de aplicación de la metodología MEDESOFT a un proyecto de desarrollo de software desarrollado por los alumnos de la carrera de Computación e Informática de CIBERTEC.

Finalmente, en el capítulo 6, se presentan las conclusiones y recomendaciones, donde se manifiestan los principales logros alcanzados y consideraciones generales sobre la metodología implementada, resaltando los aspectos más sobresalientes.

En los anexos se observan formatos, estándares y plantillas de documentación, con el propósito de que contribuyan a una mejor comprensión de la metodología.

Confío en que el presente trabajo sirva de inspiración para posteriores investigaciones sobre modelos y metodologías orientadas al desarrollo de software dentro de entidades académicas de educación superior y a través de sus estudiantes se difunda principalmente dentro de la pequeña y mediana empresa de la industria del software.

Capítulo 1
Planteamiento del Problema

1.1 ANTECEDENTES

1.1.1 DESCRIPCIÓN DEL PROBLEMA

La educación superior debe cumplir un papel central en el desarrollo socio-económico del país, sin embargo, existe una débil articulación entre los centros de educación superior y el desarrollo nacional. En opinión de destacados estudiosos, tales como Fidel Tubino Arias Schereiber, profesor de la Pontificia Universidad Católica del Perú e integrante del Foro Educativo, esta situación problemática se explica como un conjunto de desfases, es decir, la educación superior estaría desfasada con el fenómeno de globalización que exige la vigencia de un sistema de acreditación internacional y también con la no consideración de las exigencias y oportunidades del mercado laboral[4]. Es el desfase con la empresa y la comunidad el mayor problema a superar en la educación superior de modo que pueda jugar un rol importante en el desarrollo de nuestro país.

Exceptuando a muy pocas instituciones de este nivel de enseñanza, hay una escasa cultura de trabajo conjunto, de investigación, de desarrollo de proyectos que satisfagan necesidades reales de la sociedad peruana, integrando equipos multidisciplinarios y aplicando buenas prácticas y/o estándares reconocidos internacionalmente. Estas deficiencias se ven reflejadas en diversos indicadores: alto porcentaje de proyectos no exitosos, alto nivel de desvinculación entre los planes de desarrollo formativo y los planes de crecimiento empresarial, educación con excesiva cantidad de horas asignadas a la formación académica pero con poca aplicabilidad relacionada con nuestra realidad nacional[5].

Las carreras superiores de Computación e Informática, Ingeniería de Sistemas, Informática y similares, no son ajenas a esta realidad. Dentro de la mayoría de ellas, generalmente se desarrollan proyectos de desarrollo de software como parte de uno o más cursos de la carrera. Estos proyectos, muchas veces son relegados al campo meramente académico o si intentan resolver una situación problemática real, carecen de métodos, técnicas y herramientas adecuadas para ser desarrollados exitosamente, peor aún, no se tienen definidos los procesos necesarios para la producción del producto software.

[4] Tubino Arias Schereiber, Fidel: en "La universidad que el Perú necesita" Foro Educativo y Consorcio de Universidades, Lima 1999.

[5] Diagnóstico de la Universidad Peruana: Razones para una nueva reforma universitaria Comisión Nacional por la Segunda Reforma Universitaria, 2002.

1.1.2 FORMULACIÓN DEL PROBLEMA

En general, las entidades académicas superiores en el Perú, a través de sus carreras de Computación e Informática, Ingeniería de Sistemas, Informática y afines, carecen de una metodología adecuada para gestionar e implementar proyectos de desarrollo de software, desarrollados por alumnos, dentro de sus cursos regulares, y producir software de calidad como parte de un proceso definible, repetible y susceptible de ser medido.

Universidades tales como la Universidad Nacional de Ingeniería, la Universidad de Lima, la Universidad de San Martín de Porres, la Universidad Católica, entre otras, cuentan con importantes centros de Investigación en los cuales se promueve la investigación científica y aplicada a través de la implementación de diferentes tipos proyectos en los que participan docentes y alumnos, sin embargo éstos no son parte de los cursos regulares de las carreras de Ingeniería o afines.

Con base en lo anterior y para efectos de estudio, se propone en este proyecto de investigación la elaboración de una metodología académica de desarrollo de software, sobre la base de modelos reconocidos y acreditados internacionales, aplicable dentro de los cursos regulares de una o más carreras de una entidad de educación superior y ejecutada principalmente por alumnos.

1.1.3 ESTADO DEL ARTE

El Project Management Institute (PMI) fue fundado en el año 1,969 y es considerado actualmente, la asociación profesional para la gestión de proyectos sin fines de lucro más grande del mundo. En el año 1,990, fue publicada la primera edición de la guía del Project Management Body of Knowledge (PMBOK), texto base para la enseñanza de gestión de proyectos y que contiene una descripción general de los fundamentos de la gestión de proyectos reconocidos como buenas prácticas. Actualmente se encuentra en su tercera edición y es el único estándar ANSI para la gestión de proyectos. En el Perú, el PMI se encuentra representado oficialmente a través del Capítulo Perú del PMI.

En el año 1995, se emitió la primera publicación de la norma ISO/IEC 12207 Information Technology / Software Life Cycle Processes, la cual es un estándar ampliamente difundido para los procesos de ciclo de vida del software. ISO/IEC 12207 incluye procesos y actividades que se aplican desde la definición de requisitos, pasando por la adquisición y configuración de los servicios del sistema, hasta la finalización de su uso. En el Perú, se adaptó la norma creándose la norma técnica peruana "NTP-ISO/IEC 12207:2006 TECNOLOGÍAS DE LA

INFORMACIÓN". Esta norma es de uso obligatorio en las entidades del sistema nacional de Informática.

El Ministerio de Administraciones Públicas de España desarrolló la metodología de Planificación, Desarrollo y Mantenimiento de sistemas de información: MÉTRICA versión 3. Esta metodología toma como referencia el modelo del ciclo de vida del software propuesto en la norma ISO/IEC 12207 y define como importantes interfaces de soporte la gestión de proyectos, el aseguramiento de calidad y la seguridad y gestión de proyectos.

En el año 2002, el Instituto de Ingeniería del Software de la universidad Carnegie Mellon (SEI), desarrolló el modelo para la mejora y evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software (CMMI®). CMMI® ayuda a mejorar diferentes áreas de proceso dentro del desarrollo de software e ingeniería de sistemas. CMMI® se ha convertido en un Standard de certificación internacional a través de sus niveles de madurez, siendo la Gestión de Proyectos una categoría importante dentro del modelo.

A nivel regional, en Octubre de 2002 el Gobierno Federal de México, a través de la Secretaría de Economía, presentó el Programa para el Desarrollo De La Industria del Software (PROSOFT, 2002), en el cual establece las estrategias y líneas de acción para convertir a la industria mexicana del software en una industria líder en Latinoamérica. Así surgió Moprosoft en el año 2005, modelo de procesos de Software para la industria del software mexicano, que fue desarrollado a solicitud de la Secretaría de Economía de México en convenio con la facultad de ciencias de la Universidad Nacional Autónoma de México (UNAM). Este modelo fomenta la estandarización de las empresas del sector a través de la incorporación de las mejores prácticas en gestión e ingeniería de software y permite además alcanzar niveles internacionales de competitividad. Esfuerzos similares se llevaron a cabo en Brasil, lanzándose en Diciembre de 2003, el programa para la Mejora del Proceso del Software Brasileño: MPS.Br, este programa es compatible con el modelo CMMI® y las normas internacionales ISO/IEC 12207 e ISO/IEC 15504. Es coordinado por la Asociación para Promoción de la Excelencia del Software Brasileño (SOFTEX) y cuenta con el apoyo del Ministerio de Ciencia y Tecnología (MCT), de la Financiera de Estudios y Proyectos (FINEP) y del Banco Interamericano de Desarrollo (BID).

En el Perú, empresas tales como IBM y el Banco de crédito del Perú (BCP) cuentan con metodologías de gestión y desarrollo de software adecuadas y certificadas (El Banco de crédito del Perú cuenta con una certificación nivel 3 de CMMI®), sin embargo, estas experiencias de gestión aún no se han dado dentro del entorno académico.

1.2 JUSTIFICACIÓN

1.2.1 JUSTIFICACIÓN E IMPORTANCIA DEL ESTUDIO

1.2.1.1 CONVENIENCIA

General

Esta metodología definirá procesos, establecerá relaciones entre procesos, definirá objetivos, actividades e indicadores por proceso, así como técnicas y herramientas específicas que permitan sentar las bases necesarias para poder realizar una producción de software con calidad y de manera estandarizada, es decir, se constituye en punto de inicio para la creación en un futuro próximo, de una fábrica de software académica.

La metodología implementada permitirá evidenciar y medir logros de aprendizaje dentro de la entidad académica en que se implemente.

La implementación de esta metodología significará la optimización de tiempo, recursos, aseguramiento de calidad y por lo tanto incremento de las probabilidades de éxito en la producción de software dentro de la entidad académica.

Los profesionales egresados de la entidad académica en que se implemente la metodología contarán con conocimientos teóricos y prácticos de gestión de proyectos e ingeniería de software, dominarán procesos y buenas prácticas basadas en estándares internacionales, constituyéndose en agentes positivos de cambio dentro del entorno en el que se desarrollen.

La difusión de la metodología a través de los estudiantes y egresados en la sociedad, permitirá que la industria de desarrollo de software, en especial la de las pequeñas y medianas empresas (PYMES), se haga más competitiva: la mejora de la calidad del proceso de desarrollo de software, generará como consecuencia, la mejora de la calidad del producto software.

Específica

Actualmente la carrera de Computación e Informática de CIBERTEC participa de un proceso de evaluación internacional: ABET[6]. La metodología desarrollada

[6] ABET, o Accreditation Board for Engineering and Technology, Inc., acredita en los Estados Unidos programas de estudio de universidades y colleges en: ciencias aplicadas, computación, ingeniería y tecnología. Tratándose de instituciones de educación superior extranjeras, ABET evalúa sus programas y puede declararlos "sustancialmente equivalentes" a los acreditados en los Estados Unidos.

permitirá evidenciar las capacidades adquiridas por sus alumnos, contribuyendo con los requisitos solicitados para superar exitosamente el proceso en mención.

1.2.1.2 RELEVANCIA SOCIAL

Los beneficiados serán las entidades académicas, docentes, alumnos, empresas y la sociedad en general. Los alumnos, agrupados en equipos de trabajo y con la asesoría de los docentes involucrados en el proceso, asumen roles tales como el de jefe de proyecto, analista, desarrollador, responsable de aseguramiento de calidad, entre otros, beneficiándose de una experiencia real de desarrollo de software para una organización. De manera análoga las organizaciones involucradas participan y "aprenden" desde su rol de clientes y usuarios, a ser parte del proyecto. Finalmente, las entidades académicas reciben como retroalimentación las necesidades concretas de las empresas del entorno, lo que sirve de indicador de validez de los contenidos impartidos a la comunidad educativa, integrándose de esta manera la educación y formación profesional a la realidad nacional.

1.2.1.3 VALOR METODOLÓGICO

El valor metodológico de la presente investigación descansa en la escasa labor de investigativa desarrollada en nuestro país en el campo de la gestión e implementación de proyectos de desarrollo de software, realizados principalmente por alumnos, dentro de los cursos regulares de una o más carreras de una institución académica. Es por esta razón que la metodología que se implementará, así como los instrumentos que se diseñen para ella, serán un aporte en el conocimiento para las instituciones en que se aplique esta metodología.

1.2.2 DELIMITACIÓN DEL PROBLEMA

1.2.2.1 ESPACIAL

El presente trabajo implementará un proyecto piloto que aplique la metodología en el Instituto Superior Tecnológico Privado CIBERTEC, específicamente con los alumnos del sexto ciclo de la carrera de Computación e Informática.

1.2.2.2 TEMPORAL

Para realizar el presente trabajo, se tomó como referencia el ciclo académico 2008-2

1.3 OBJETIVOS GENERALES Y OBJETIVOS ESPECÍFICOS

Sobre la base del problema formulado se plantean los siguientes objetivos:

1.3.1 OBJETIVOS GENERALES

- Desarrollar un conjunto de pasos metodológicos, técnicas y herramientas para la gestión e implementación de proyectos de desarrollo de software, realizados por alumnos dentro de los cursos regulares de una o más carreras de una entidad académica de educación superior.

- Sentar las bases para que a través de la aplicación de la metodología desarrollada, se pueda implementar en un futuro, una Fábrica de Software Académica.

1.3.2 OBJETIVOS ESPECÍFICOS

Para alcanzar los objetivos generales planteados dentro del presente proyecto de investigación es necesario concretar los siguientes objetivos específicos:

- Analizar los modelos de procesos y normas de mayor prestigio y difusión internacional, relacionadas con el ciclo de vida del software. Se identificarán las principales características y aportes, de modo que sirvan de guía para la implementación de la metodología académica.

- Seleccionar un modelo de referencia, el cual proporcionará la estructura base sobre la que se desarrollará la metodología académica.

- Sobre el modelo de referencia seleccionado, definir procesos y sus relaciones, objetivos, indicadores y métricas por proceso, así como técnicas y herramientas a ser utilizadas por los procesos. Este conjunto de artefactos, constituirán la metodología académica de gestión y desarrollo de software.

- Haciendo uso de la metodología desarrollada, implementar un proyecto piloto de desarrollo de software en los cursos:

    - Desarrollo de Aplicaciones Web II y Certificación de Calidad por los alumnos del sexto ciclo de la carrera de Computación e Informática en el Instituto CIBERTEC.

  Es importante señalar que los cursos de Diseño de Proyectos, Calidad de Software (cursos prerrequisitos de Certificación de Calidad) y

Certificación de Calidad son proveedores de métodos, técnicas y herramientas que a manera de servicios son brindados a los cursos de programación: Desarrollo de Aplicaciones Web I y Desarrollo de Aplicaciones Web II.

Con el propósito de alcanzar los objetivos específicos, el proyecto de investigación ha sido estructurado tal como se muestra en el siguiente cuadro:

Objetivo Específico	Capítulo	Ideas Principales	Aporte
Analizar los modelos de procesos y normas de mayor prestigio y difusión internacional, relacionadas con el ciclo de vida del software.	**Capítulo 3**	Norma ISO / IEC 12207 Modelo de madurez CMMI Modelos regionales: • MOPROSOFT (México) • MPS.BR (Brasil)	Conocer las normas y modelos más Importantes dentro Del contexto del ciclo de vida del Software.
Seleccionar un modelo de referencia a partir del cual se implementará la metodología académica de desarrollo de software.	**Capítulo 3**	-Criterios de Evaluación -Aplicación de Evaluación y Análisis de Resultados -Selección del Modelo de Referencia	Identificar la Estructura base de la Metodología a Implementar
Implementar la metodología académica de desarrollo de software MEDESOFT, tomando como base el modelo de referencia seleccionado y como parte de un Enfoque Estratégico.	**Capítulo 4**	Plan Estratégico: Análisis FODA, misión, visión objetivos, indicadores y métricas de la organización. Plan Táctico: Estrategia de Implementación Y componentes MEDESOFT	MEDESOFT: Metodología Académica de Desarrollo de Software Implementada
Implementar un proyecto piloto de Desarrollo de software haciendo uso de la metodología MEDESOFT	**Capítulo 5**	Proyecto de Desarrollo De Software	Capacidades reales de alumnos en actividades de Inge-niería de Software

Cuadro 1. 1 Relación de Objetivos específicos y Capítulos del Proyecto de Investigación

La lectura de los capítulos sigue el esquema de la figura 1, la cual se agrega para facilitar la lectura al lector.

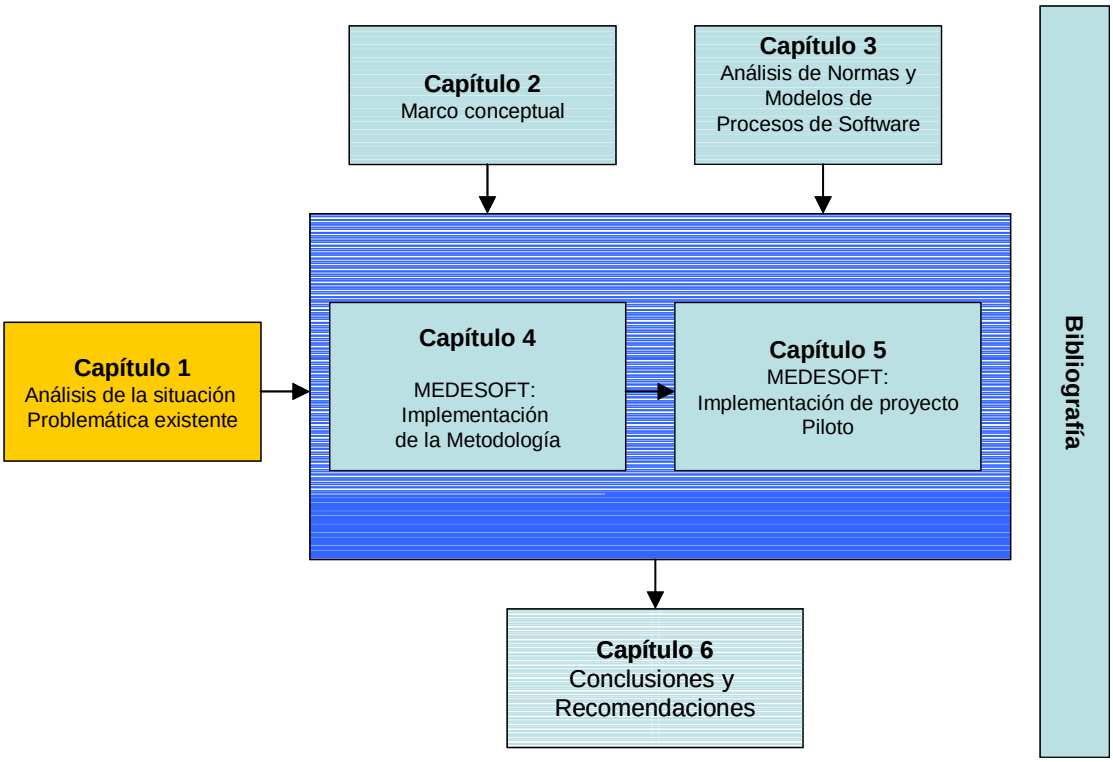


Figura 1.1 Como leer este documento

1.4 POBLACIÓN DE ESTUDIO

Instituto Superior Tecnológico CIBERTEC, cuyos alumnos implementan proyectos de desarrollo de software dentro de sus programas académicos regulares.

1.5 INSTRUMENTOS DE RECOLECCIÓN DE DATOS

Con el objetivo de validar el éxito de la aplicación de la metodología propuesta, en relación a un esquema de desarrollo carente de la misma, se definirán indicadores, que son expresiones cuantitativas de los objetivos del presente trabajo.

Para ello se seguirán los siguientes pasos:

- Selección inicial de indicadores
- Validación y aprobación del conjunto de indicadores
- Aplicación de los indicadores al proyecto

De esta manera, se obtendrá el medio adecuado para establecer que condiciones son las que señalan el logro de los objetivos de este trabajo de investigación, reduciéndose la ambigüedad y subjetividad en torno al grado de éxito alcanzado.

Capítulo 2
Marco Conceptual

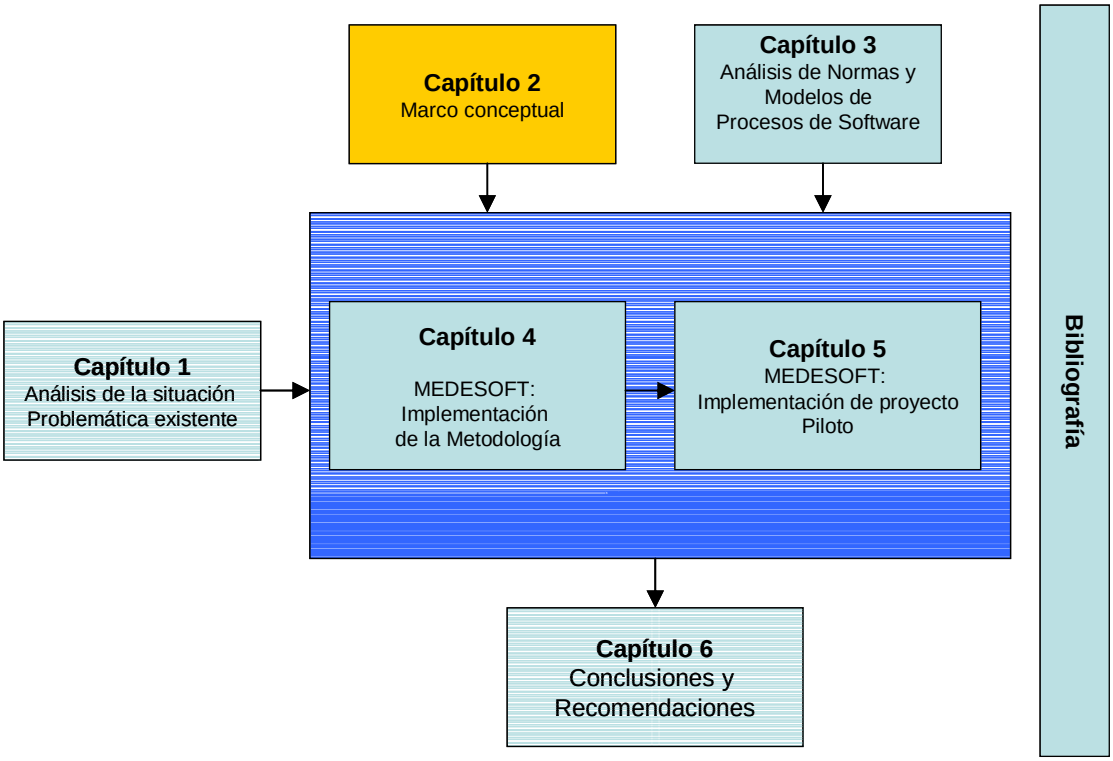


FIGURA 2.1 UBICACIÓN EN LA LECTURA DEL DOCUMENTO

Para el desarrollo de este proyecto de investigación es necesario tener plenamente definidos conceptos básicos de calidad, calidad de software, procesos de software y modelos de procesos de software.

2.1 CALIDAD DEL SOFTWARE

2.1.1 INTRODUCCIÓN

El desarrollo de software hoy en día ha tomado gran importancia en el mundo, siendo éste cada vez más creciente.

Así como esta importancia tiende a aumentar, surgen también problemas típicos tales como retrasos en la programación, inconsistencia en su funcionamiento, entre

otros; sin embargo, el problema más importante es la falta de calidad, aspecto de vital interés para el logro de eficiencia y productividad de las soluciones de software.

Ante esta situación se hizo necesario impulsar líneas de acción que permitan mejorar el software producido. Dentro de estas líneas de acción está la relacionada con el proceso mismo de desarrollo de software, manifestándose como necesidad primordial, el realizar una investigación que permita conocer de primera mano el estado en que se encuentra su proceso de desarrollo.

Existen actualmente organizaciones internacionales que definen estándares y modelos de ingeniería de software que permiten mejorar la calidad del producto software y de su respectivo proceso de desarrollo.

2.1.2 CALIDAD

ISO (International Standard Organization) define la calidad como la ausencia de deficiencias: "Es la totalidad de aspectos y características de un producto o servicio que se refieren a su capacidad para satisfacer necesidades dadas en la adecuación de sus objetivos".

El instituto de ingeniería de Software (SEI) en su modelo CMM define la calidad como:

- El grado en el cual un sistema, componente o proceso cumple con los requisitos especificados.
- El grado en el cual el sistema, componente o proceso cumple con las expectativas del cliente o usuario.

De ambas definiciones, podemos deducir que la calidad consiste en todos aquellos aspectos del producto que satisfacen las necesidades del cliente.

Estas necesidades son especificadas en un contrato y requieren ser identificadas y definidas, en términos de seguridad, utilización, disponibilidad, adaptabilidad con otros productos, confiabilidad, mantenimiento, costo, entre otros.

La calidad es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad.

2.1.3 CALIDAD DEL SOFTWARE

"La calidad del software se define como la concordancia de los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares y procesos de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente"[7].

La definición anterior, nos permite destacar tres aspectos importantes:

a) Los requisitos de software son la base de las medidas de la calidad. La falta de concordancia con los requisitos es una falta de calidad.
b) Los estándares especificados definen un conjunto de criterios de desarrollo que guían la forma en que se aplica la ingeniería del software. En caso de no seguirse esos criterios, casi siempre habrá falta de calidad.
c) Existe un conjunto de requisitos implícitos que a menudo no se mencionan (por ejemplo la necesidad de una interfaz intuitiva). Si el software se ajusta a sus requisitos explícitos pero falla en alcanzar los requisitos implícitos, la calidad del software se debilita.

Queda claro, a partir de la definición de calidad del software, que ésta es siempre relativa a los requisitos o necesidades que se desean satisfacer. Es por ello que la evaluación de la calidad del software siempre va a implicar la comparación entre los requisitos y el producto generado.

La calidad del software es medible y varía de un sistema a otro o de un programa a otro. Por ejemplo, un software elaborado para el control de naves espaciales debe ser confiable al nivel de "cero fallas". Un software hecho para ser explotado durante un largo periodo de tiempo, necesita ser confiable, mantenible y flexible para disminuir los costos de mantenimiento y perfeccionamiento durante su tiempo de explotación.

Podríamos medir la calidad del software después de elaborado el producto, sin embargo, esto puede resultar muy costoso si se detectan problemas derivados de imperfecciones en el diseño, es por ello imprescindible tener en cuenta tanto la obtención de la calidad, como su control durante todo el proceso de desarrollo de software.

[7] Roger S. Pressman. Doctor en Ciencias Físicas en Ingeniería por la Universidad de Connecticut. Autoridad reconocida a nivel internacional en el mejoramiento del proceso de software y en las tecnologías de ingeniería de software.

2.2 PROCESO DE SOFTWARE

2.2.1 INTRODUCCIÓN

Definido por el SEI como "Conjunto de actividades, métodos, prácticas y transformaciones que la gente usa para desarrollar y mantener software y los productos de trabajo asociados (planes de proyecto, diseño de documentos, código, pruebas y manuales de usuario)"

La idea de proceso deriva del campo de la fabricación, donde los procesos (pasos de fabricación) están definidos y se controlan de manera continua. Este enfoque se puede aplicar al mundo del software para la gestión del proceso a nivel de cada proyecto y para mejorar las capacidades de los grupos de desarrollo.

El proceso que fabrica el software se denomina proceso software. Así, podemos establecer que proceso software es la secuencia de pasos necesaria para desarrollar o mantener software.

De manera más específica, el proceso software establece el marco de trabajo tanto técnico como de gestión para poder aplicar métodos, herramientas y personas a la tarea de desarrollo de software. La definición del proceso identifica los roles y las tareas específicas. Además establece medidas y proporciona criterios de entrada y salida para cada paso.

2.2.2 DEFINICIÓN DEL PROCESO DE SOFTWARE

Se denomina definición del proceso de software a la descripción del proceso para elaborar el software. Cuando el proceso está diseñado y documentado de manera apropiada, es la definición del proceso la que guía a los miembros del equipo de proyecto sobre la manera de trabajar. Un equipo que siga diferentes procesos para realizar un mismo producto, o en general, que no utilice un proceso definido, es como un equipo de fútbol que aunque pueda tener a los mejores jugadores, su rendimiento será bastante pobre. Por el contrario, un equipo que siga unas definiciones de proceso consistentes puede coordinar mejor el trabajo de cada miembro y por ende, se puede rastrear su progreso de una manera más precisa y efectiva.

Una definición diseñada de forma adecuada sirve para asegurar que cada elemento de trabajo está asignado de modo apropiado y que se conoce su estado en cada momento. También proporciona un mecanismo ordenado para su aprendizaje. A medida que se encuentran nuevos métodos, se incorporan en las definiciones oficiales del proceso de la organización. Un proceso definido permite que cada

nuevo proyecto sea construido en base a la propia experiencia y a la de sus predecesores.
Los problemas que surjan a medida que se utiliza el proceso definido permiten identificar sus causas. Entonces se pueden realizar correcciones sobre el proceso para eliminar las causas identificadas. Tanto el proceso, como su definición e infraestructura de soporte evolucionarán con la experiencia que se gana al utilizar el proceso.

Es por ello que las organizaciones que utilizan de manera rutinaria la definición de procesos, evolucionan, son más competitivas y eficientes que sus pares sin ella.

2.2.3 CICLO DE VIDA DEL SOFTWARE

El ciclo de vida de software es el periodo que comienza cuando un producto software es concebido y termina cuando deja de estar disponible.

Un modelo de ciclo de vida contiene los procesos, actividades y tareas involucradas en el desarrollo, operación y mantenimiento de un producto software y abarca toda la vida del sistema desde la definición de sus requerimientos hasta el final de su uso. El desarrollo de software es sólo una parte de todo el modelo de ciclo de vida.

A continuación se muestra el alcance del ciclo de vida:

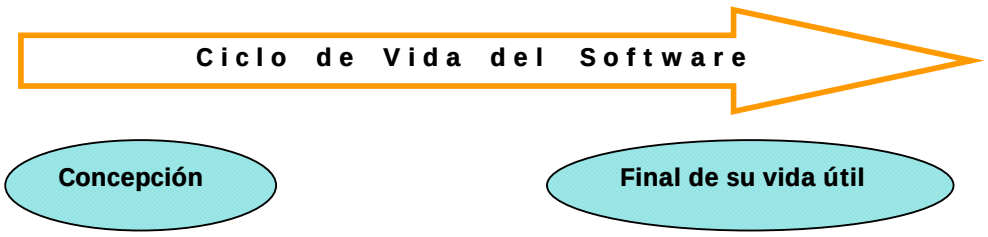


FIGURA 2.2 ALCANCE DEL CICLO DE VIDA DEL SOFTWARE

Las funciones principales de un ciclo de vida de software son:

- Determinar el orden de las fases y procesos involucrados en el desarrollo de software y su evolución incluyendo la explotación y mantenimiento del mismo.

- Establecer los criterios de transición para pasar de una fase a la siguiente. Todo ello incluye los criterios para verificar la terminación de la fase actual y los criterios para seleccionar e iniciar la fase siguiente.

Es esencial definir el ciclo de vida del software que debe seguir cada proyecto debido a que permite clasificar y controlar las distintas actividades para el desarrollo y mantenimiento del producto software.

2.2.4 MODELOS DE PROCESOS DE SOFTWARE

Producto del estudio de los diferentes procesos que intervienen en el desarrollo de software se crearon a partir del año 1980, diferentes modelos de procesos. Éstos, definen un conjunto estructurado de elementos que describen las características de procesos efectivos y de calidad, indicando que actividades se deben realizar.

Existen también modelos que permiten a partir de una evaluación, determinar la madurez y capacidades de los procesos dentro de una organización. A partir de esta evaluación se establecen las prioridades para mejorar los procesos.

Actualmente destacan como modelos de procesos por su difusión en el ámbito mundial:

- La norma ISO/IEC 12207. Estándar que define los procesos del ciclo de vida del software.

- La norma ISO/IEC 15504. Estándar para evaluar capacidades de los procesos del ciclo de vida del software.

- CMMI (Integración de Modelos de madurez de capacidades), desarrollado por el Instituto de Ingeniería de Software (SEI). Proporciona un marco de referencia y recoge entre otras, las mejoras prácticas de las normas ISO / IEC 12207 e ISO / IEC 15504.

A nivel latinoamericano destacan también:

- MOPROSOFT. Modelo de procesos de software desarrollado para la industria del software mexicano.

- MPS. BR. Programa de mejora del software brasileño. Incluye entre otros componentes, un modelo de referencia (MR-MPS) y un método de evaluación (MA-MPS).

2.3 RESUMEN DEL CAPÍTULO

En este capítulo se han descrito los principales conceptos en torno a los cuales gira el presente proyecto de tesis. Entender las definiciones de calidad, calidad de software, proceso de software y modelos de procesos de software permitirá comprender fácilmente el análisis comparativo realizado en el siguiente capítulo, de los diferentes modelos y normas relacionadas con los procesos en la industria del software. No es posible asegurar en el tiempo la calidad de un producto software si no se cuenta con un proceso de software plenamente establecido, necesidad que se pretende satisfacer dentro del mundo académico con la implementación de la metodología académica de desarrollo de Software: MEDESOFT.

Capítulo 3
Análisis de Normas y Modelos de Procesos de Software

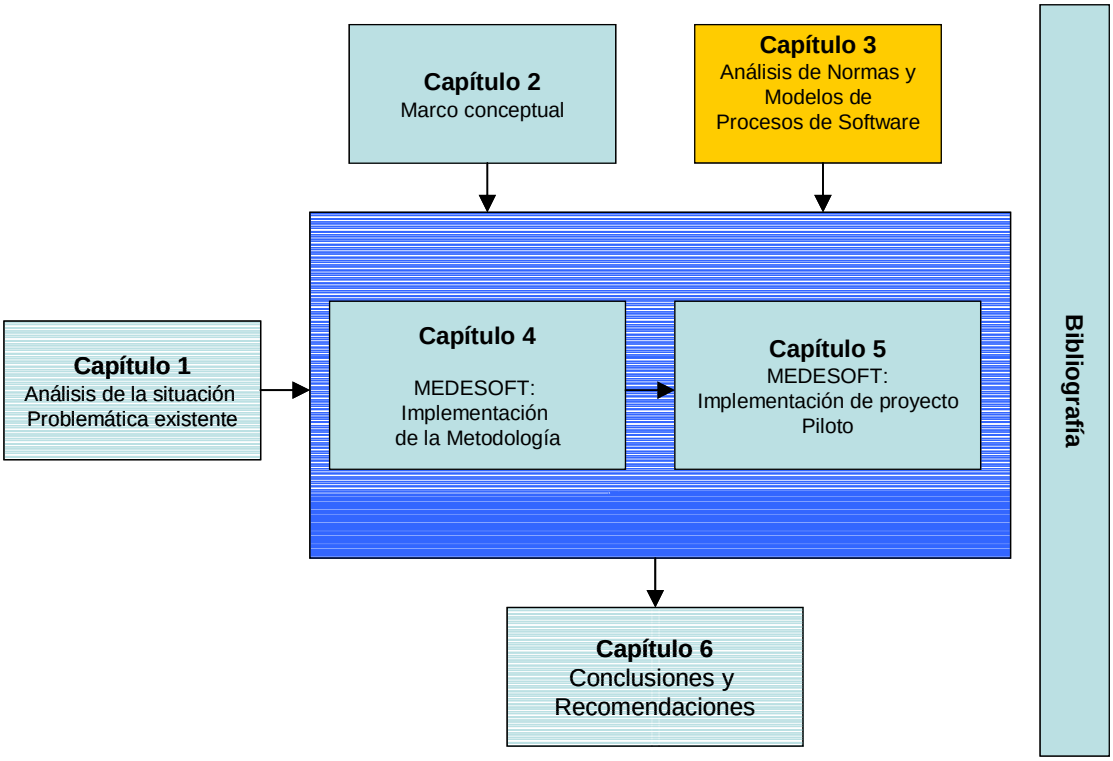


FIGURA 3.1 UBICACIÓN EN LA LECTURA DEL DOCUMENTO

En este capítulo se realiza un análisis comparativo de las principales normas y modelos de procesos de software vigentes a nivel mundial y latinoamericano, identificando sus principales características, ventajas y desventajas.

Este análisis permitió seleccionar el modelo de referencia que servirá de base para la implementación de la metodología académica objeto del presente trabajo de investigación.

Para realizar el análisis comparativo se definió una secuencia ordenada de pasos que permitieron seleccionar objetivamente y en base a criterios preestablecidos el modelo de referencia de la metodología académica a implementar. En la figura 3.2 se muestra la secuencia de actividades realizadas para implementar este proceso.

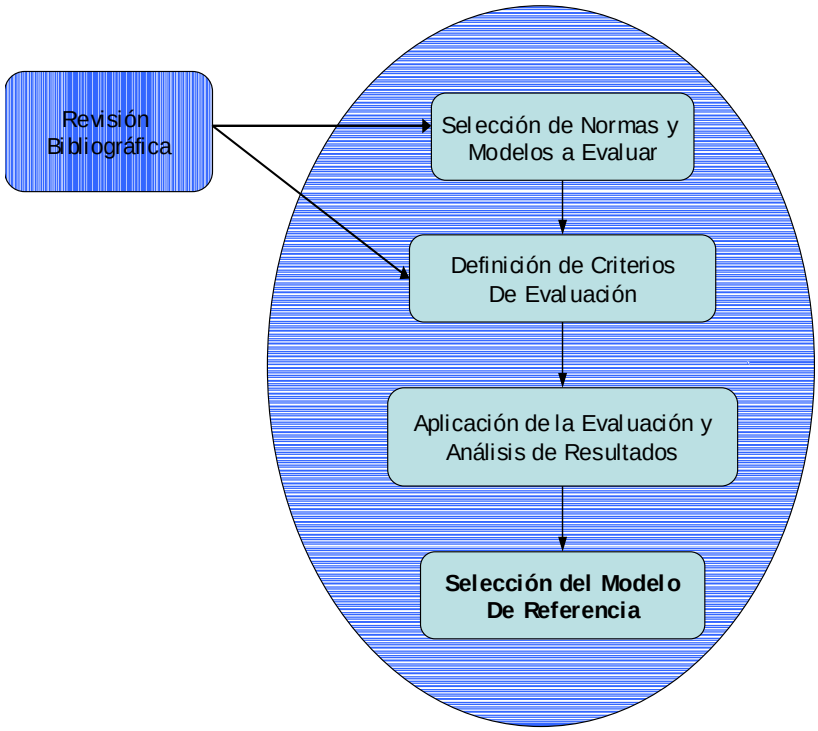


FIGURA 3.2 ANÁLISIS DE NORMAS Y MODELOS – SECUENCIA DE ACTIVIDADES

La revisión bibliográfica se planteó durante todo el desarrollo del proyecto de investigación. Aspectos relevantes de gestión de Proyectos, estándares y modelos de calidad de software, entre otros, fueron sujetos de revisión.

Específicamente en este capítulo, facilitó el desarrollo de las actividades de selección de normas y modelos internacionales a evaluar así como la definición de los criterios de evaluación empleados. Se detallan a continuación cada una de las actividades realizadas:

3.1 SELECCIÓN DE NORMAS Y MODELOS A EVALUAR

Se seleccionaron las siguientes normas y modelos:

- ISO / IEC 12207
- CMMI
- MPS.BR (Brasil)
- MOPROSOFT (México)

La selección se realizó teniendo en cuenta la importancia y vigencia actual de los modelos y normas dentro de la industria del Software. Así por ejemplo, la norma internacional ISO / IEC 12207 es también norma peruana a través de su equivalente nacional NTP-ISO /IEC 12207:2006. El modelo de madurez CMMI se ha convertido en un estándar de facto a nivel mundial y empresas de países líderes en la industria de software tales como Estados Unidos y la India ostentan altos niveles de madurez basados en este modelo. En Latinoamérica, se incrementa también el número de empresas que sigue esta tendencia. Finalmente, a nivel regional, se han realizado importantes esfuerzos para desarrollar modelos que se adapten a la realidad de las empresas latinoamericanas dedicadas a la industria del software: constituidas en su mayoría por micro y pequeñas empresas. Los esfuerzos más importantes han sido realizados en México y Brasil a través de sus modelos MOPROSOFT y MPS.BR respectivamente.

3.1.1 NORMA ISO / IEC 12207

3.1.1.1 RESEÑA HISTÓRICA

La norma ISO/IEC[8] 12207 es el estándar para los procesos de ciclo de vida del software de las organizaciones ISO e IEC.

En el año 1988, fue propuesto el desarrollo de la norma y, en agosto de 1995, fue publicada como norma internacional.

En los años 2002 y 2004, se hicieron actualizaciones en la norma ISO/IEC 12207, llamadas enmiendas 1 y 2 respectivamente, donde fueron incluidas algunas mejoras. Esas mejoras crearon nuevos procesos, o expandieron el alcance de otros, incluyeron también para cada proceso su propósito y resultados y para los nuevos procesos definieron sus actividades y tareas.

El 28 de Julio del año 2006, se oficializa como norma técnica peruana: NTP-ISO /IEC 12207:2006. Tecnología de la información. Procesos del ciclo de vida del

[8] IEC. Comisión Electrotécnica Internacional (IEC, por sus siglas del idioma inglés International Electrotechnical Commission). Es una organización de normalización en los campos eléctrico, electrónico y tecnologías relacionadas. Numerosas normas se desarrollan conjuntamente con la ISO: normas ISO/IEC.

software. 2da Edición. Esta norma peruana, es una adopción de la norma ISO/IEC 12207 y de sus dos enmiendas.

3.1.1.2 DESCRIPCIÓN DE LA NORMA

La norma ISO/IEC 12207 establece un marco de referencia común para los procesos del ciclo de vida del software. Contiene procesos, actividades y tareas para aplicar durante la adquisición de un sistema que contiene software, un producto software puro o un servicio software, y durante el suministro, desarrollo, operación y mantenimiento de productos software. El software incluye la parte software del firmware.

Esta norma incluye también un proceso que puede emplearse para definir, controlar y mejorar los procesos del ciclo de vida del software.

3.1.1.3 CAMPO DE APLICACIÓN

Esta norma es aplicable a la adquisición de sistemas, productos y servicios software, al suministro, desarrollo, operación y mantenimiento de productos software y a la parte software del firmware, independientemente de que sea hecha interna o externamente a una organización. Incluye también aquellos aspectos de la definición de sistemas necesarios para proporcionar el contexto de los productos y servicios software.

Esta norma está orientada para ser usada en situaciones en las que haya dos partes, incluido el caso en que estas dos partes pertenezcan a la misma organización. La situación puede ir desde un acuerdo informal, hasta un contrato con responsabilidades legales. Es posible también sea utilizada por una sola parte como una autoimposición.

3.1.1.4 ORGANIZACIÓN

Esta norma agrupa las actividades que pueden llevarse a cabo durante el ciclo de vida del software en cinco procesos principales, ocho procesos de apoyo y cuatro procesos organizativos. Se muestran en la Figura 2.3 los procesos agrupados por categorías:

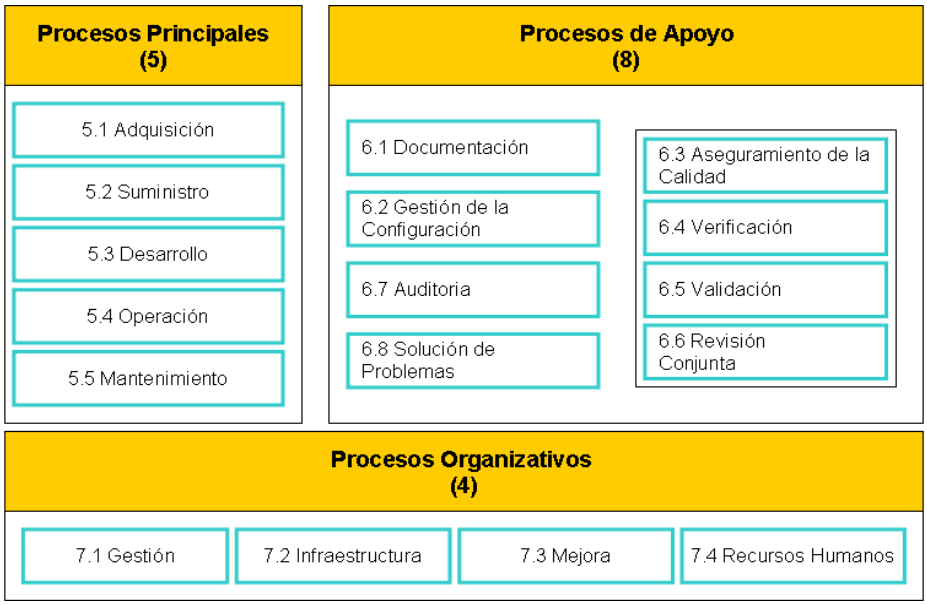


Figura 3.3 Procesos del ciclo de Vida del Software

Cada proceso del ciclo de vida está dividido en un conjunto de actividades; cada actividad se sub-divide a su vez en un conjunto de tareas. En la Figura 2.4 se puede apreciar esta clasificación:

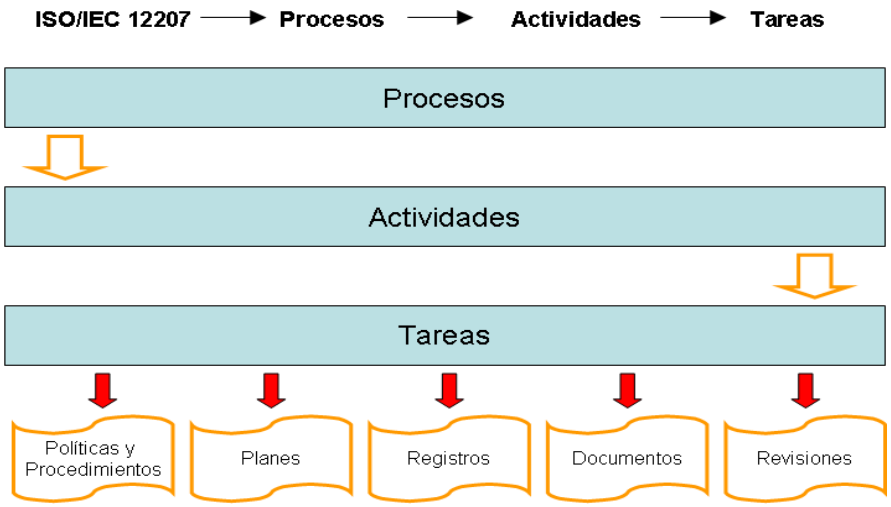


Figura 3.4 Procesos, Actividades y Tareas

Para todos los casos, la norma describe los procesos, actividades y tareas, pero no especifica como implementarlos:

"What to do" action, not a "how to do" action.

Son las organizaciones las encargadas de seleccionar y aplicar los procedimientos, planes, o documentos que estimen convenientes para llevar a cabo las actividades y tareas.

Para la implementación de la norma se deberá elegir cuales procesos, actividades y tareas son requeridas para el proyecto específico, no se exige que se empleen todos los procesos y/o actividades, sino solo aquellos que se crean convenientes.

Procesos Principales del Ciclo de Vida.

Los procesos principales del ciclo de vida son cinco, y dan servicio a las partes principales durante el ciclo de vida del software. Una parte principal es aquella que inicia o lleva a cabo el desarrollo, operación, o mantenimiento de los productos software. Estas partes principales son el adquiriente, el proveedor, el desarrollador, el operador y el responsable de mantenimiento de productos software. Los procesos principales son:

Proceso de adquisición. Define las actividades del adquiriente, la organización que adquiere un sistema, producto software o servicio software.

Proceso de suministro. Define las actividades del proveedor, organización que proporciona un sistema, producto software o servicio software al adquiriente.

Proceso de desarrollo. Define las actividades del desarrollador, organización que define y desarrolla el producto software. Contiene las actividades para el análisis de los requerimientos, diseño, codificación, integración, pruebas e instalación y aceptación relacionadas con los productos software.

Este proceso consta de las siguientes actividades:

a) Implementación del proceso
b) Análisis de los requerimientos del sistema
c) Diseño de la arquitectura del sistema
d) Análisis de los requerimientos software
e) Diseño de la arquitectura del software
f) Diseño detallado del software
g) Codificación y pruebas del software
h) Integración del software
i) Pruebas de calificación del software
j) Integración del Sistema

k) Pruebas de calificación del sistema
l) Instalación del Software
m) Apoyo a la aceptación del software

Proceso de operación. Define las actividades del operador, organización que proporciona el servicio de operar un sistema informático en su entorno real, para sus usuarios.

Proceso de Mantenimiento. Define las actividades del responsable de mantenimiento, organización que proporciona el servicio de mantenimiento del producto software; esto es, la gestión de las modificaciones al producto software para mantenerlo actualizado y operativo. Este proceso incluye la migración y retirada del producto software.

Procesos de Apoyo del Ciclo de Vida.

Son ocho los procesos de apoyo del ciclo de vida. Un proceso de apoyo es el que apoya a otro proceso como parte esencial del mismo, con un propósito bien definido, y contribuye al éxito y calidad del proyecto software. Un proceso de apoyo se emplea y ejecuta por otro proceso, según sus necesidades. Los procesos de apoyo son:

Proceso de Documentación. Define las actividades para el registro de la información producida por un proceso de ciclo de vida.

Proceso de Gestión de la Configuración. Define las actividades de la gestión de la configuración.

Proceso de Gestión de Aseguramiento de la Calidad. Define las actividades para asegurar, de una manera objetiva, que los productos software y los procesos son conformes a sus requisitos especificados y se ajustan a los planes establecidos. Los procesos de revisión conjunta, auditoría, verificación y validación pueden ser utilizados como técnicas de aseguramiento de calidad.

Proceso de Verificación. Define las actividades (para el adquiriente, proveedor o una parte independiente) para verificar hasta un nivel de detalle dependiente del proyecto de software, los productos software.

Proceso de Validación. Define las actividades (para el adquiriente, proveedor o una parte independiente) para validar los productos software del proyecto software.

Proceso de revisión conjunta. Define las actividades para evaluar el estado y productos de una actividad. Este proceso puede ser empleado por cualquiera de las

dos partes, donde una de las partes (la revisora) revisa a la otra parte (la parte revisada), de una manera conjunta.

Proceso de auditoría. Define las actividades para determinar la conformidad con los requisitos, planes y contrato. Este proceso puede ser empleado por dos partes cualesquiera, donde una parte (la auditora), audita los productos software o actividades de la otra parte (la auditada).

Proceso de solución de problemas. Define un proceso para analizar y eliminar problemas (incluyen las no conformidades) que sean descubiertas durante la ejecución del proceso de desarrollo, operación, mantenimiento u otros procesos, cualesquiera que sea su naturaleza o causa.

Procesos Organizativos del Ciclo de Vida

Los procesos organizativos del ciclo de vida son cuatro. Se emplean por una organización para establecer e implementar una infraestructura constituida por procesos y personal asociado al ciclo de vida y para mejorar continuamente esa infraestructura. Se usan habitualmente fuera del ámbito de proyectos y contratos específicos; sin embargo, la experiencia adquirida mediante dichos proyectos y contratos contribuye a la mejora de la organización. Los procesos organizativos son:

Proceso de Gestión. Define las actividades básicas de gestión, incluyendo la gestión de proyectos, durante un proceso de ciclo de vida.

Proceso de Infraestructura. Define las actividades básicas para establecer la infraestructura de un proceso de ciclo de vida.

Proceso de Mejora. Define las actividades básicas que una organización (adquiriente, proveedor, desarrollador, operador, responsable de mantenimiento o gestor de otro proceso) lleva a cabo para establecer, medir, controlar y mejorar su proceso del ciclo de vida.

Proceso de Recursos Humanos. Define las actividades básicas para conseguir personal adecuadamente formado.

Proceso de Adaptación

Define las actividades básicas necesarias para llevar a cabo adaptaciones de esta norma. Proporciona una breve guía sobre cómo adaptar las directrices de esta norma; enumera los factores claves sobre los que se pueden basar las decisiones de adaptación.

3.1.1.5 FORTALEZAS Y DEBILIDADES

Fortalezas

- La norma ISO / IEC 12207 establece un lenguaje común entre las partes involucradas en la adquisición o suministro de productos software, define por cada proceso propósitos y resultados que servirán de indicadores para evaluar si una organización ha logrado establecer un proceso específico.

- A través de la definición de actividades y tareas para cada proceso, establece buenas prácticas generales de Gestión, Ingeniería, Formación de Personal entre otras, que una organización debería optar por implementar.

Debilidades

- Define una arquitectura de procesos de ciclo de vida de software pero no especifica como implementarla.

- No define nombres, formatos o contenido explícito de la documentación que se genere en cada uno de los procesos.

- Dentro del contexto del trabajo de investigación, es bastante general, involucra procesos que van más allá de la gestión y desarrollo de software, haciendo más difícil su comprensión.

- No es una norma prescriptiva, es decir, define que actividades y tareas se deben de hacer pero no indica cómo hacerlas.

3.1.2 CAPABILITY MATURITY MODEL INTEGRATION: CMMI®

3.1.2.1 RESEÑA HISTÓRICA

CMMI (Capability Maturity Model Integration) es la evolución natural de un conjunto de modelos de madurez, agrupados bajo la denominación genérica CMM (Capability Maturity Model).

CMM fue desarrollado por el SEI (Software Engineering Institute) de la universidad Carnegie-Mellon, a solicitud del Departamento de Defensa del gobierno federal de los Estados Unidos de América. Se creó una primera definición de modelo de madurez de procesos en el desarrollo de software, la cual sería publicada en septiembre de 1987. Este trabajo evolucionó al modelo CMM o SW-CMM (CMM for Software), publicándose su última versión (v1.1) en febrero de 1993.

CMM creció popularmente, el SEI y la comunidad de usuarios CMM, comenzaron a ver el potencial de aplicar el modelo a un amplio grupo de actividades de TI. Es así que el SEI desarrolló formas extensibles de CMM, a partir de la versión base: SW-CMM. Por ejemplo, otra versión fue orientada para los procesos y actividades de la ingeniería de sistemas, conocida como SE-CMM. Una nueva versión fue orientada a manejar las necesidades de productos estratégicos y servicios de adquisición y finalmente una última versión fue desarrollada para apoyar la integración de Productos y el Desarrollo de Procesos.

Cada uno de estos modelos contó con la aceptación y el éxito dentro de las organizaciones que las manejaban. Sin embargo, todos estos modelos tenían muchos elementos comunes. Después de algunos años, muchas de las organizaciones que utilizaron los cuatro modelos evidenciaron la necesidad de contar con un nuevo modelo que capitalice los elementos comunes y a la vez permita que sea configurable y flexible.

CMMI® sería la iniciativa que agruparía estas varias versiones en un único modelo configurable. La Integración de Modelos de Madurez de Capacidades representa el resultado final. Se publicó la primera versión de CMMI en el año 2002, evolucionando hasta la versión actual (CMMI v1.2), la cual se publicó en Agosto de 2006.

Actualmente, el SEI se encuentra en la etapa de planificación de la versión 1.3 de la suite de productos CMMI. Esta versión incluirá actualizaciones a los modelos en las áreas de de Desarrollo, Adquisición y Servicios. Tan pronto se culmine con esta etapa, estará disponible mayor información sobre esta versión en el sitio Web del SEI, proyectándose su lanzamiento para el año 2010.

3.1.2.2 DESCRIPCIÓN DEL MODELO

CMMI es un modelo para la mejora de procesos que proporciona a las organizaciones los elementos esenciales para contar con procesos eficaces. CMMI trabaja bajo el postulado de que toda organización que pretenda ser exitosa, se demuestra a sí misma que lo es a través del tiempo, madurando sus procesos. Las organizaciones exitosas diseñan procesos, los implementan, los estudian, y luego los refinan a lo largo del tiempo.

En las palabras del reconocido estudioso William Deming (1960), la definición precisa sería Planificar, Hacer, Verificar y Actuar. (**P**lan, **D**o, **C**heck, **A**ct).

Las mejores prácticas CMMI se publican en documentos llamados modelos. La versión actual de CMMI es la versión 1.2 y son dos los modelos disponibles:

- CMMI para el Desarrollo (DEV-CMMI), versión 1.2. En este modelo se tratan procesos de desarrollo de productos y servicios. Es especialmente útil para organizaciones tecnológicas que deben desarrollar productos, procesos o ambos para sus clientes.

- CMMI para la adquisición (ACQ-CMMI), versión 1.2. En éste modelo se tratan la gestión de la cadena de suministro, adquisición y contratación externa en los procesos del gobierno y la industria.

Independientemente del modelo que utilice una organización, las prácticas CMMI deben adaptarse a cada organización en función de sus objetivos de negocio. Luego, la organización es evaluada (por ejemplo, usando un método de evaluación como SCAMPI[9]) y recibe una calificación de nivel entre 1 y 5.

3.1.2.3 COMPONENTES DEL MODELO

Área de Proceso

Un área de proceso es un conjunto de prácticas en un área específica que cuando son ejecutadas en forma colectiva, satisfacen un conjunto de objetivos considerados importantes para lograr una mejora significativa en esa área en particular.

CMMI® está estructurado como un conjunto de Áreas de Proceso (PA por sus siglas en inglés). Podemos pensar en cada área de proceso como una colección de mejores prácticas que ayudan a una organización a gestionar sus actividades y controlar su calidad. El modelo completo contiene 22 áreas de procesos, las cuales

[9] El Standard CMMI Appraisal Method for Process Improvement (SCAMPI) es el método oficial SEI para proveer puntos de referencia de sistemas de calificación en relación con los modelos CMMI. SCAMPI se usan para identificar fortalezas y debilidades de los procesos, revelar riesgos de desarrollo/adquisición, y determinar niveles de capacidad y madurez.

pueden ser organizadas en categorías y son mostradas a continuación en el cuadro 2.1:

Categoría	Áreas de Proceso
Gestión de Proyectos	Planificación de Proyectos Monitorización y Control Gestión y Acuerdo con Proveedores Gestión Integrada de Proyectos Gestión de Riesgos Gestión Cuantificada de Proyectos
Soporte	Gestión de la Configuración Gestión de la Calidad de Procesos y Productos Medición y Análisis Análisis y Resolución de Decisiones Análisis y Resolución de Problemas|
Ingeniería	Desarrollo de Requisitos Gestión de Requisitos Soluciones Técnicas Integración de Producto Verificación Validación
Gestión de Procesos	Definición de los procesos de la Organización Procesos orientados a la Organización Formación Rendimiento de los procesos de la Organización Innovación y Desarrollo

Cuadro 3.1 Categorías y Áreas de Proceso CMMI

Cada área de proceso dentro de CMMI® establece uno o más objetivos que deben ser alcanzados por una organización para que ésta pueda declarar que cumple efectivamente con el programa. Existen dos clases de objetivos en el modelo: objetivos específicos y objetivos generales.

Los objetivos generales representan los elementos comunes de CMMI®. Los objetivos específicos están orientados a cada área de proceso.

Adicionalmente, cada objetivo es soportado por prácticas recomendadas. Los objetivos son expresados en términos de alto nivel, mientras que las prácticas son expresadas en términos y acciones más concretas. La asunción con la estructura práctica-objetivo es que si implementamos las prácticas descritas para cada objetivo, alcanzaremos la intención del objetivo en sí mismo.

Hay prácticas específicas para los objetivos específicos del modelo y hay prácticas genéricas para los objetivos genéricos del modelo. A continuación se muestra en la figura 2.5 un resumen de los componentes del modelo CMMI:

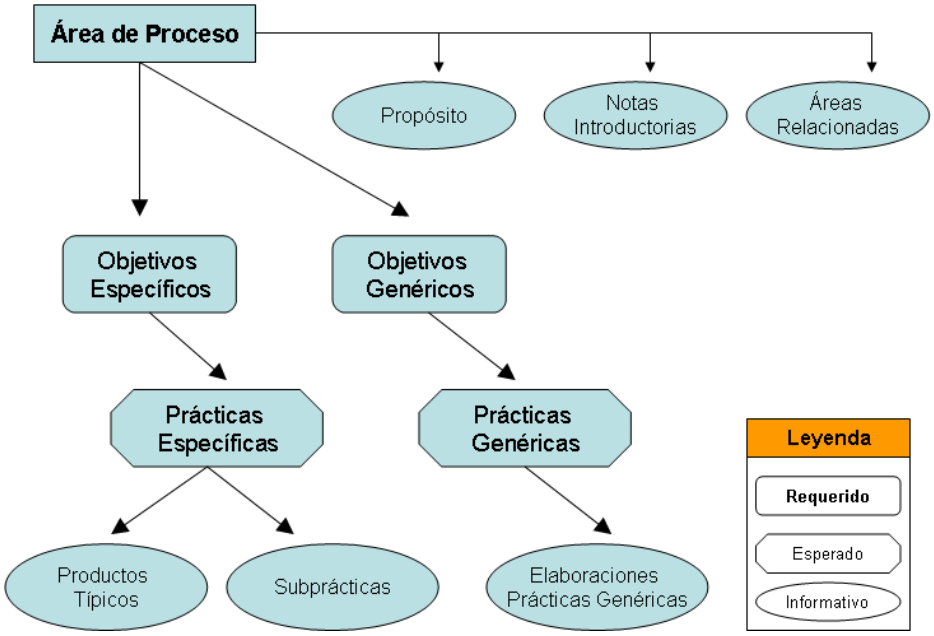


Figura 3.5 Componentes del Modelo CMMI

CMMI define componentes requeridos, esperados e informativos. Describimos cada uno de estos componentes a continuación:

Componentes Requeridos

Son las componentes que obligatoriamente deben ser satisfechas y visiblemente implementadas para poder cumplir con un área de proceso. Un componente requerido es usado en las evaluaciones para la determinación de los niveles de madurez dentro de la organización. Los componentes requeridos dentro del modelo CMMI son los siguientes:

Objetivos Genéricos. Cada nivel de capacidad contiene un único objetivo genérico que describe el grado de institucionalización que la organización debe lograr para ese nivel de capacidad en particular. Éstos aparecen en todas las Áreas de proceso. Los objetivos genéricos y sus prácticas aplican a múltiples áreas de proceso.

El logro de un objetivo genérico en un área implica haber mejorado el control en la planificación e implementación de los procesos asociados con esa área en

particular, y adicionalmente implica que esos procesos serán probablemente efectivos, repetibles y duraderos.

Objetivos Específicos. Son objetivos que se aplican a las áreas de proceso y se orientan únicamente a las características que describen lo que debe ser implementado para satisfacer el área de proceso. Como componentes requeridos, en ellos se basan las evaluaciones para la determinación de si un área en particular es satisfecha por un determinado proceso (o un conjunto de procesos) o no.

Componentes Esperados

Son componentes que pueden ser utilizados para alcanzar un componente requerido, es decir se podrían implementar estos componentes o modificaciones válidas de ellos con el objetivo de alcanzar los objetivos genéricos o específicos. Los componentes esperados pueden ser utilizados como guías de mejora y de evaluación de procesos. Son componentes esperados dentro del modelo los siguientes:

Prácticas Genéricas. Las prácticas genéricas proveen institucionalización para asegurarse de que los procesos serán efectivos, repetibles y duraderos.

Existen dos tipos de dependencia de las prácticas genéricas para con las áreas de proceso:

- Algunas prácticas genéricas "se basan" en las áreas de proceso.
- Algunas prácticas genéricas no pueden ser ejecutadas sin la salida previa de un área de proceso determinado.

Prácticas Específicas. Las prácticas son actividades que se consideran importantes para el logro de los objetivos específicos asociados. Cada práctica específica está asociada a un nivel de capacidad. Las prácticas específicas pueden ser de dos tipos:

- *Prácticas Base.* En la representación continua, todas las prácticas específicas asociadas a un nivel de capacidad 1, son denominadas "Prácticas Base".

- *Prácticas Avanzadas.* En la representación continua, todas las prácticas específicas asociadas a un nivel de capacidad 2 o mayor, son denominadas "Prácticas Avanzadas".

Componentes Informativos

Son componentes que deben ser entendidos sólo como ayudas propuestas por el modelo para entender mejor los componentes requeridos y esperados. A continuación se describen los principales componentes informativos del modelo:

Elaboración de las Prácticas Genéricas. Son componentes del modelo que aparecen en cada área de proceso para proveer una guía de cómo cada práctica genérica debe ser aplicada para cada área de proceso.
Productos de Evaluación. Los productos son una denominación para las salidas de una práctica específica o genérica.

Sub-prácticas. Son especificaciones más detalladas de las prácticas genéricas o específicas. Proveen una guía de cómo implementar las prácticas. Su nivel de detalle podría interpretarse como prescriptito, pero sin embargo, son componentes "Informativos" para ejemplarizar.

3.1.2.4 REPRESENTACIONES DE CMMI

CMMI® cuenta con dos representaciones, la representación continua y la escalonada. Ambas representaciones son equivalentes y cada organización puede optar por utilizar la que se adapte a sus características y prioridades de mejora.

La representación continua se basa en niveles de capacidad para medir la mejora de los procesos, mientras que la representación escalonada se basa en niveles de madurez. A continuación se detallan ambos conceptos dentro del contexto de CMMI®:

- *Madurez***.** La madurez es un atributo de la organización: Indica cuan previsible y repetible es la calidad de sus resultados, y su capacidad para aprender de la experiencia.

- *Capacidad.* La capacidad es un atributo de los procesos: Indica su eficacia y eficiencia para obtener el resultado.

Representación escalonada

Esta representación se aplica a toda la organización. Establece 5 niveles de madurez para clasificar a las organizaciones, en función de qué áreas de procesos consiguen sus objetivos y se gestionan con principios de ingeniería.

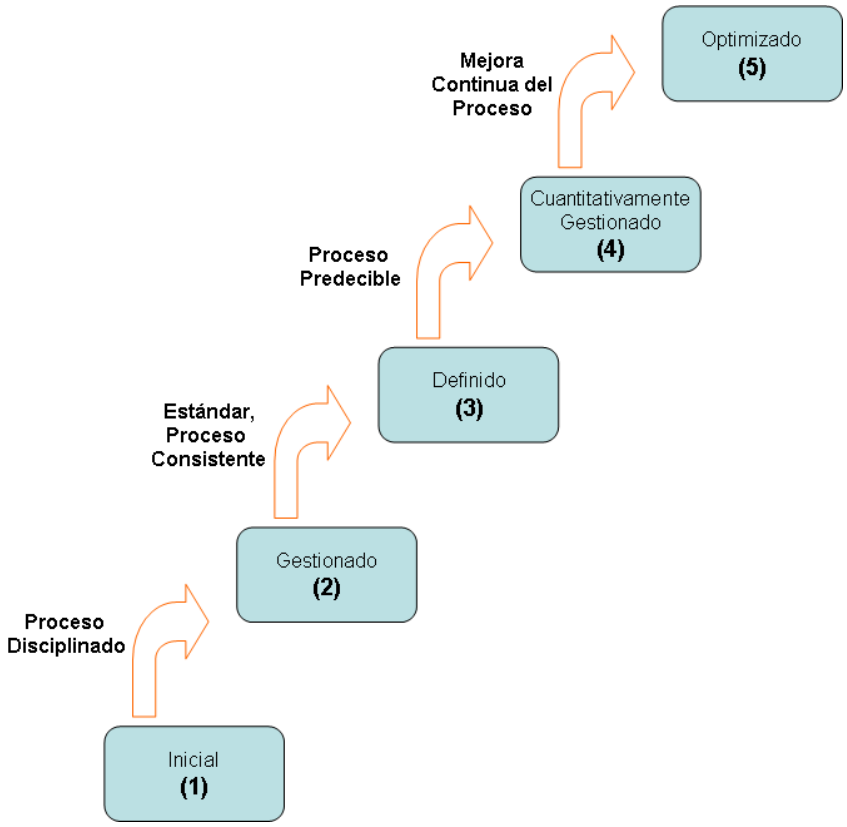


Figura 3.6 Niveles de madurez CMMI

Cada nivel de madurez comprende un conjunto de áreas de proceso; representa una capa en la base de la mejora continua de procesos, usando una secuencia de mejoras probadas, que comienzan con prácticas base de gestión y progresan a través de un camino predefinido de niveles exitosos certificables. Así, en el nivel 1, los procesos son reactivos, impredecibles y pobremente controlados, mientras que en el nivel 5 los procesos ya son medidos y controlados, evidenciándose una mejora continua de los mismos.

Representación continua

Esta representación se aplica a un área de proceso específica y permite que una organización seleccione un área de proceso y mejore con relación a esta área en particular.

La representación continua usa niveles de capacidad para caracterizar la mejora relacionada a un área de proceso en particular. Existen seis niveles de capacidad (numerados del cero al cinco) los cuales apreciamos en su relación con los niveles de madurez en el siguiente gráfico:

	Representación Continua	**Representación Escalonada**
	Nivel de Capacidad	Nivel de Madurez
Nivel 0	Incompleto	-
Nivel 1	Realizado	Inicial
Nivel 2	Gestionado	Gestionado
Nivel 3	Definido	Definido
Nivel 4	Cuantitativamente Gestionado	Cuantitativamente Gestionado
Nivel 5	Optimizado	Optimizado

Figura 3.7 Niveles de Representación Continua y Escalonada CMMI

Los niveles son acumulativos: los niveles más altos de capacidad incluyen los atributos de los niveles más bajos. A continuación se describen cada uno estos niveles:

Nivel 0: Incompleto. El proceso no se realiza, o no se consiguen sus objetivos.

Nivel 1: Ejecutado. El proceso se ejecuta y se logra su objetivo.

Nivel 2: Gestionado. Además de ejecutarse, el proceso se planifica, se revisa y se evalúa para comprobar que cumple los requisitos.

Nivel 3: Definido. Además de ser un proceso gestionado se ajusta a la política de procesos que existe en la organización, alineada con las directivas de la empresa.

Nivel 4: Cuantitativamente Gestionado. Además de ser un proceso definido se controla utilizando técnicas cuantitativas.

Nivel 5: Optimizado. Además de ser un proceso cuantitativamente gestionado, de forma sistemática se revisa y modifica o cambia para adaptarlo a los objetivos del negocio, es decir, implica una mejora continua.

3.1.2.5 FORTALEZAS Y DEBILIDADES

Fortalezas

- El modelo CMMI proporciona una visión estructurada de la mejora de procesos dentro de una organización.
- Permite establecer objetivos y prioridades de mejora de procesos.
- Proporciona una guía para tener procesos de calidad.
- Proporciona un mecanismo para evaluar las prácticas actuales dentro de una organización.
- Es un modelo probado y exitoso, que a través de la mejora de procesos, logra beneficios tangibles para una organización: mejora de tiempos de desarrollo, mejora productividad, mejora de la calidad (medida en número de defectos encontrados), mejora la satisfacción del cliente [10]

Debilidades

- No es fácil de entender, dado el nivel de detalle que presenta en la definición de cada una de sus áreas de proceso.
- No es fácil de aplicar, en general, ha sido desarrollado para ser implementado por grandes organizaciones.
- Puede ser excesivamente detallado para algunas organizaciones, en especial por el nivel de documentación que requiere, esto limita su aplicación dentro de la pequeña y mediana empresa.
- Los costos de capacitación y evaluación son altos.

[10] http://www.sei.cmu.edu/cmmi/results.html

3.1.3 MPS.BR PROGRAMA DE MEJORA DEL PROCESO DE SOFTWARE BRASILEÑO

3.1.3.1 RESEÑA HISTÓRICA

En el año 2003, datos de la secretaria de Política de Informática y Tecnología del Ministerio de Ciencia y Tecnología (MCT/SEITEC) de Brasil, mostraban que solo 30 empresas en Brasil poseían evaluación CMM-SW. De éstas, 24 habían alcanzado el nivel de madurez 2; 5 se encontraban en el nivel 3; 1 en el nivel 4; y ninguna en el nivel 5.

El análisis previo, permitió evidenciar que dentro de la industria de software brasileña, se podían identificar dos grandes grupos de empresas:

- Un primer grupo reducido, constituido por las empresas exportadoras de software y otras grandes empresas que deseaban alcanzar niveles más altos de madurez (4 ó 5) de CMM, en un esfuerzo que puede llevar de 4 a 10 años.

- Un segundo grupo integrado por la gran masa de pequeñas y medianas empresas de software, con pocos recursos y con la necesidad de obtener mejoras significativas en sus procesos de software en 1 ó 2 años como máximo.

Es dentro de este contexto que surge en Diciembre de 2003, el programa de Mejora del Proceso de Software Brasileño: MPS.BR. Este programa se orienta principalmente, aunque no de manera exclusiva, a mejorar la calidad del proceso software de la pequeña y mediana empresa utilizando un modelo de procesos propio, basado en estándares y modelos internacionales tales como CMMI e ISO/IEC 12207.

El programa MPS.BR es coordinado por la Asociación para la Promoción de la Excelencia del Software Brasileño (SOFTEX), institución equivalente de la Asociación peruana de Software (APESOFT), y cuenta con el apoyo del Ministerio de Ciencia y Tecnología de Brasil (MCT), el Banco Interamericano de Desarrollo (BID) entre otras importantes instituciones.

La coordinación del programa, cuenta adicionalmente con dos estructuras de apoyo para el desarrollo de sus actividades: el Foro de Acreditación y Control (FCC) y el Equipo Técnico del Modelo (ETM).

A través de estas estructuras, el MPS.BR obtiene la participación de representantes de Universidades, instituciones Gubernamentales, Centros de Investigación y de organizaciones privadas, las cuales contribuyen con sus visiones complementarias y agregan calidad al emprendimiento.

Actualmente son más de 2800 personas las que han sido capacitadas en diferentes cursos relacionados con el modelo de procesos del programa MPS.BR, contando esta iniciativa, con más de 30 instructores entrenados y certificados para dictar dichos cursos. Finalmente, son más de 72 empresas las que ya han alcanzado diferentes niveles de madurez dentro del modelo, siendo el objetivo para el periodo 2008 – 2010 superar las 300 empresas acreditadas[11].

3.1.3.2 DESCRIPCIÓN DEL PROGRAMA

El programa MPS.BR tiene como una de sus principales metas, definir y perfeccionar un modelo de mejora y evaluación de procesos de software. Este modelo debe considerar de una manera especial a las pequeñas y medianas empresas, de modo que se atiendan sus necesidades de negocio.

MPS.BR establece también un proceso y un método de evaluación, el cual da sustento y asegura que el programa está siendo empleado de modo coherente con sus definiciones. Se pretende que el modelo sea reconocido nacional e internacionalmente como un modelo aplicable a la industria de software.

La base técnica para la construcción y perfeccionamiento del modelo está compuesta por las normas ISO/IEC 12207, ISO/IEC 15504 y el modelo CMMI. Es dentro de este contexto, que MPS.BR define tres componentes principales: un modelo de Referencia (MR-MPS), un método de Evaluación (MA-MPS) y un modelo de Negocio (MN-MPS).

Cada de uno estos componentes y el programa en general, es descrito por medio de documentos en formato de guías. A continuación se describen las guías más importantes:

- Guía General. Contiene la descripción general del programa y detalla el Modelo de Referencia (MR-MPS), sus componentes y las definiciones comunes necesarias para su entendimiento y aplicación.

- Guía de Adquisición. Describe un proceso de adquisición de software y servicios correlativos. Está escrito de modo que apoye a las instituciones que quieran adquirir productos de software y servicios correlativos apoyándose en el MR-MPS.

- Guía de Evaluación. Describe el proceso y el método de evaluación MA-MPS, los requisitos para evaluadores líderes, evaluadores adjuntos e Instituciones Evaluadoras (IA).

[11] Información extraída de la documentación de MPS.BR. Mejora del proceso de Software Brasileño. [En línea]. <http://www.softex.br/portal/softexweb/uploadDocuments/_mpsbr/Apresentação%2024ABR2008%20MPS.BR%20Forum%20TIC%20Governo%202008.pdf >. [Consulta: Julio de 2007]

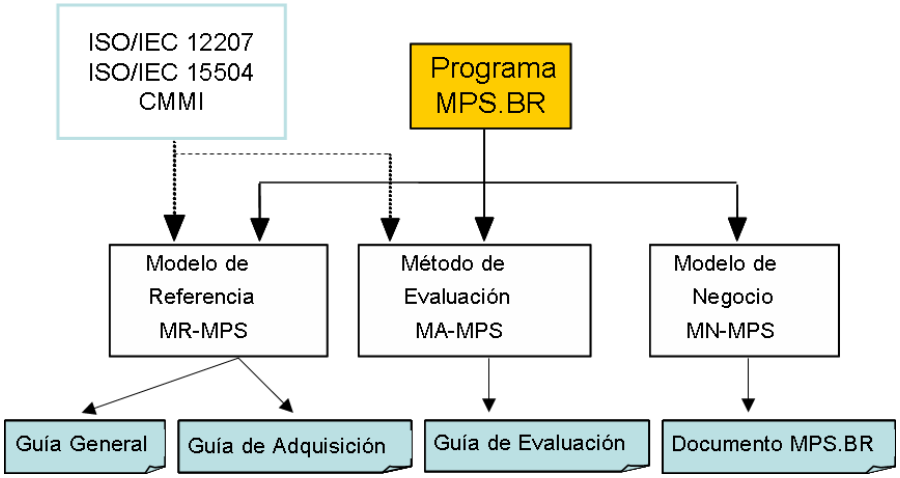


Figura 3.8 Componentes de MPS.BR

3.1.3.3 DESCRIPCIÓN DEL MODELO DE REFERENCIA: MR-MPS.

El Modelo de Referencia MR-MPS define niveles de madurez que son una combinación entre procesos y su capacidad.

Procesos

La definición de los procesos sigue la forma presentada en la enmienda 1 de la norma ISO/IEC 12207, es decir, se declaran el propósito y los resultados esperados de su ejecución:

- El propósito describe el objetivo general que debe ser logrado durante la ejecución del proceso.

- Los resultados esperados del proceso establecen los resultados que deben ser obtenidos con la efectiva implementación del proceso.

Adicionalmente por cada proceso se cuenta con una sección denominada informaciones adicionales. Las informaciones adicionales son referencias que pueden ayudar en la definición del proceso por la organización. Normalmente, se cita el proceso o subproceso de la norma ISO/IEC 12207 o el área de proceso correspondiente de CMMI.

Definir el propósito y los resultados esperados de cada proceso, permite evaluar y atribuir grados de efectividad en la ejecución de los mismos. Es importante destacar que el modelo no define las actividades y tareas necesarias para cumplir el propósito y los resultados esperados de cada proceso: su definición es responsabilidad de la organización usuaria del modelo.

Niveles de Madurez

Los niveles de madurez establecen etapas de evolución de procesos, identificando escalones de mejora de la implementación de los procesos en la organización.

MR-MPS define siete niveles de madurez:

A: En Optimización
B: Gestionado Cuantitativamente
C: Definido
D: Ampliamente Definido
E: Parcialmente Definido
F: Gestionado y
G: Parcialmente Gestionado

La escala de madurez se inicia en el nivel G y progresa hasta el nivel A. Para cada uno de estos niveles, se atribuye un perfil de procesos que indica adonde la organización debe realizar el esfuerzo de mejora.

El progreso y el logro de un determinado nivel de madurez se obtiene cuando se cumplen los propósitos y todos los resultados esperados de los procesos correspondientes al nivel de madurez, así como también los atributos de proceso establecidos para aquel nivel.

Capacidad de Proceso

La capacidad del proceso está representada por un conjunto de atributos de proceso (AP) descritos en términos de sus resultados esperados (RAP). La capacidad del proceso expresa el grado de refinamiento e institucionalización con que el proceso es ejecutado en la organización. En el modelo, a medida que la organización evoluciona en los niveles de madurez, debe lograr un mayor nivel de capacidad para llevar a cabo el proceso.

MR-MPS posee cinco AP identificados como: AP 1.1, AP 2.1, AP 2.2, AP 3.1 y AP 3.2. Cada AP está detallado en términos de los resultados esperados del atributo de proceso (RAP). A continuación se detallan los atributos de proceso AP 1.1 y AP 2.1:

AP 1.1: El proceso es ejecutado. Este atributo es una medida de la extensión en la cual el proceso logra su propósito.

Resultado Esperado:

RAP 1. El proceso logra sus resultados definidos

AP 2.1: El proceso es gestionado. Este atributo es una medida de la extensión en la cual la ejecución del proceso es gestionada.

Resultados Esperados:

RAP 2. Existe una política de la organización establecida y mantenida para el proceso.

RAP 3. La ejecución del proceso es planificada.

RAP 4. (para el Nivel G). La ejecución del proceso es supervisada y ajustes son realizados para cumplir los planes.

RAP 4. (a partir del Nivel F). Las medidas son planificadas y recolectadas para la supervisión de la ejecución del proceso.

RAP 5. Los recursos necesarios para la ejecución del proceso son identificados y puestos a disposición.

RAP 6. Las personas que llevan a cabo el proceso son competentes en términos de formación, entrenamiento y experiencia.

RAP 7. La comunicación entre las partes interesadas en el proceso es gestionada de modo que se asegure su participación en el proyecto.

RAP 8. El estado, actividades y los resultados del proceso son revisados con los niveles apropiados de Gerencia (incluyendo la gerencia de Alto Nivel) y los problemas pertinentes son tratados.

A continuación se muestra en el cuadro 2.2 un resumen de los niveles de madurez del MR-MPS, los procesos y los atributos de proceso asociados a cada nivel.

7 Niveles	21 Procesos	5 Atributos de Proceso (Capacidad)
A – En Optimización (más alto)	Implantación de Innovaciones en la Organización – IIO Análisis de Causas y Resolución - ARC	AP 1.1, AP 2.1, AP 2.2, AP 3.1 y AP 3.2
B – Gestionado Cuantitativamente	Desempeño del Proceso Organizacional - DEP Gestión Cuantitativa del Proyecto – GQP	AP 1.1, AP 2.1, AP 2.2, AP 3.1 y AP 3.2
C – Definido	Gestión de Riesgos - GRI Análisis de Decisión y Resolución – ADR	AP 1.1, AP 2.1, AP 2.2, AP 3.1 y AP 3.2
D – Ampliamente Definido	Desarrollo de Requisitos - DRE Solución Técnica - STE Validación - VAL Verificación - VER Integración del Producto – ITP	AP 1.1, AP 2.1, AP 2.2, AP 3.1 y AP 3.2
E – Parcialmente Definido	Entrenamiento - TRE Definición del Proceso Organizacional – DFP Evaluación y Mejora del Proceso Organizacional – AMP Adaptación del Proceso para Gestión de Proyecto – APG	AP 1.1, AP 2.1, AP 2.2, AP 3.1 (proceso estándar es definido) y AP 3.2 (proceso estándar está implementado posibilitando demostrar lo apropiado y la eficacia del proceso, y evaluar adonde puede hacerse la mejora continua del proceso)
F – Gestionado	Gestión de Configuración - GCO Aseguramiento de la Calidad – GQA Medición – MED Adquisición - AQU	AP 1.1, AP 2.1 y AP 2.2 (productos de trabajo del proceso son gestionados)
G – Parcialmente Gestionado (más bajo)	Gestión de Proyecto - GPR Gestión de Requisitos – GRE	AP 1.1 (proceso es ejecutado) y AP 2.1 (proceso es gestionado)

Cuadro 3.2 Niveles de Madurez del modelo de Referencia MR-MPS

3.1.3.4 FORTALEZAS Y DEBILIDADES

Fortalezas

- Está orientado a la pequeña y mediana empresa. Por ejemplo, aunque se base en los niveles de madurez de CMMI, tiene una gradación diferente. La posibilidad de realizar evaluaciones considerando más niveles permite una visibilidad de los resultados de mejora de procesos en un plazo más corto, aspecto crítico para las PYMES al contar con un retorno de inversión más rápido.
- Hereda de la norma ISO / IEC 12207 la definición de propósitos y resultados por cada proceso. Ambos componentes sirven de indicadores

para evaluar si una organización ha logrado establecer un proceso específico.
- Es menos costoso que CMMI
- Es compatible con CMMI. Sirve de base a las empresas que implementen el modelo, para futuras evaluaciones CMMI

Debilidades

- No cuenta con el nivel de detalle adecuado en la definición de actividades y tareas para un proceso específico. De manera similar a la norma ISO / IEC 12207, indique que actividades deben hacerse pero no el cómo implementarlas.
- No existe una relación explícita entre la mejora del proceso de software y los objetivos de negocio de la organización.

3.1.4 MOPROSOFT MODELO DE PROCESOS DE SOFTWARE MEXICANO

3.1.4.1 RESEÑA HISTÓRICA

El Plan Nacional de Desarrollo de México para el periodo 2001 - 2006 planteaba el objetivo de elevar y extender la competitividad del país, mediante la estrategia de promover el uso y el aprovechamiento de la tecnología y la información.

En base a lo anterior, la secretaría de Economía de México (SE) define el Programa para el Desarrollo de la industria del Software (PROSOFT), como uno de los medios para concretar el objetivo.

El programa PROSOFT

Desde su creación, el objetivo de PROSOFT fue impulsar a la industria de software y extender el mercado de tecnologías de información en México. Para alcanzar el objetivo trazado, la SE, en consenso con la industria y con los organismos gubernamentales relacionados con el sector, acordaron desarrollar siete estrategias, las cuales son enumeradas a continuación:

1.- Promover las exportaciones y la atracción de inversiones

2.- Educación y formación de personal competente en el desarrollo de software, en cantidad y calidad convenientes.

3.- Contar con un marco legal promotor de la industria.

4.- Desarrollar el mercado interno.

5.- Fortalecer a la industria local.

6.- Alcanzar niveles internacionales en capacidad de procesos.

7.- Promover la construcción de infraestructura básica y de telecomunicaciones

Como parte de la implementación de la estrategia 6: “Alcanzar niveles internacionales en capacidad de procesos, se evaluó dentro del programa, la adopción de los siguientes modelos:

- ISO 9000
- ISO 15504
- SW-CMM

El resultado de la evaluación fue que ninguno de los estándares o modelos cumplía con los requisitos expresados por la industria de software mexicana, por lo tanto, se decidió elaborar un modelo propio basado en los modelos evaluados.

La SE estableció un convenio con la Facultad de Ciencias de la Universidad Nacional Autónoma de México (UNAM), encargándole a ésta y a la Asociación Mexicana para la Calidad en Ingeniería del Software (AMCIS) la elaboración del Modelo de Procesos para la Industria del Software: MOPROSOFT.

El modelo MOPROSOFT

La primera versión del modelo se terminó de elaborar en Diciembre de 2002, iniciándose su proceso de normalización dentro del NYCE[12] en el año 2003.

Este proceso culminaría el 15 de agosto de 2005 cuando se publica en el diario oficial de la federación mexicana la declaratoria de vigencia de la norma NMX-059-NYCE-2005, bajo el nombre de Tecnología de la Información-Software-Modelos de Procesos y Evaluación para Desarrollo y Mantenimiento de Software.

MOPROSOFT es actualmente el modelo de referencia del proyecto COMPETISOFT. Este proyecto se orienta a la mejora de procesos para fomentar la competitividad de la pequeña y mediana industria del software de Iberoamérica. Participan en el proyecto de investigación universidades, empresas, centros públicos y organismos de estandarización de trece países de la región, entre los que se encuentran: Argentina, Brasil, Chile, Colombia, Costa Rica, Cuba, Ecuador, España, México, Perú, Portugal, Uruguay y Venezuela.

3.1.4.2 DESCRIPCIÓN DEL MODELO

MOPROSOFT define un modelo de procesos de software, a través del cual proporciona a las empresas o áreas internas dedicadas al desarrollo y/o mantenimiento de software, un conjunto integrado de buenas prácticas de gestión e ingeniería de software basadas en modelos y estándares reconocidos internacionalmente.

Las organizaciones, que no cuenten con procesos establecidos, pueden usar el modelo ajustándolo de acuerdo a sus necesidades. Por otro lado, las organizaciones que ya tienen procesos establecidos, pueden usarlo como punto de referencia para identificar los elementos que les hace falta cubrir.

[12] La asociación denominada "Normalización y Certificación Electrónica" (NYCE), actúa en México como Organismo Nacional de Normalización, facultado para elaborar, coordinar y emitir Normas mexicanas de Electrónica, Telecomunicaciones e Informática.

El modelo permite apoyar a las empresas a establecer sus procesos o sirve de punto de referencia para mejorar su madurez a niveles de calidad internacionales.

Características

- El modelo define pocos procesos que abarcan todos los niveles de una organización: directivo, gerencial y operativo.
- Se enfoca en los procesos.
- Define explícitamente una Base de Conocimiento Organizacional en la cual se guardan todos los productos generados, se administran y se consultan de acuerdo con mecanismos predefinidos.
- Define explícitamente guías de ajuste que sugieren la adaptación de los procesos a las necesidades de las organizaciones, siempre y cuando no se pierda de vista el cumplimiento de los objetivos de los procesos.
- Los procesos del modelo pueden ser reajustados en base al contexto de la organización.
- Los procesos de todas las categorías se encuentran integrados.
- Define explícitamente actividades de verificación, validación y pruebas para fomentar la calidad de los productos.
- Define explícitamente roles responsables por las actividades de cada proceso y la capacitación requerida.
- Definición explícita del propósito, objetivos específicos, indicadores, metas cuantitativas y mediciones para cada proceso.

Enfoque basado en Procesos

El desarrollo y mantenimiento de software se lleva a cabo a través de una serie de actividades realizadas por equipos de trabajo.

La Ingeniería de Software se ha dedicado a identificar las mejores prácticas para realizar estas actividades recopilando las experiencias exitosas de la industria de software a nivel mundial. Estas prácticas se han organizado por áreas de aplicación, y se han dado a conocer como áreas clave de procesos, en el caso del modelo CMM, o como procesos de software en el caso de ISO/IEC 122207.

El modelo MOPROSOFT está enfocado en procesos y considera los tres niveles básicos de la estructura de una organización que son: la Alta Dirección, Gestión y Operación.

A través de la definición e integración de procesos dentro de estos tres niveles, apoya a las organizaciones en la estandarización de sus prácticas, en la evaluación de su efectividad y en la integración de la mejora continua.

Criterios empleados para la elaboración del Modelo

Para la elaboración de MOPROSOFT, se tomaron en cuenta los siguientes criterios:

1. Generar una estructura de procesos acorde con la estructura de las organizaciones de la industria de software: Alta Dirección, Gestión y Operación.

2. Destacar el papel de la alta dirección en la planificación estratégica, su revisión y mejora continua como el promotor del buen funcionamiento de la organización.

3. Considerar a la gestión como proveedora de recursos, procesos y proyectos, así como responsable de vigilar el cumplimiento de los objetivos estratégicos de la organización.

4. Considerar a la operación como ejecutora de los proyectos de desarrollo y mantenimiento de software.

5. Integrar de manera clara y consistente los elementos indispensables para la definición de procesos y relaciones entre ellos.

6. Integrar los elementos para la administración de proyectos en un solo proceso.

7. Integrar los elementos para la ingeniería de productos de software en un solo marco que incluya los procesos de soporte (verificación, validación, documentación y control de configuración).

8. Destacar la importancia de la gestión de recursos, en particular los que componen la base de conocimiento de la organización tales como: productos generados por proyectos, datos de los proyectos, incluyendo las mediciones, documentación de procesos y los datos recaudados a partir de su uso y lecciones aprendidas.

9. Basar el modelo de procesos en la norma ISO 9001:2000 y en los niveles 2 y 3 de SW-CMM. Usar como marco general la norma ISO/IEC 15504 e incorporar las mejores prácticas de otros modelos de referencia tales como PMBOK.

3.1.4.3 ESTRUCTURA DEL MODELO DE PROCESOS

MOPROSOFT divide los procesos de una organización en tres categorías de procesos:

- Alta Dirección (DIR)
- Gestión (GER)
- Operación (OPE)

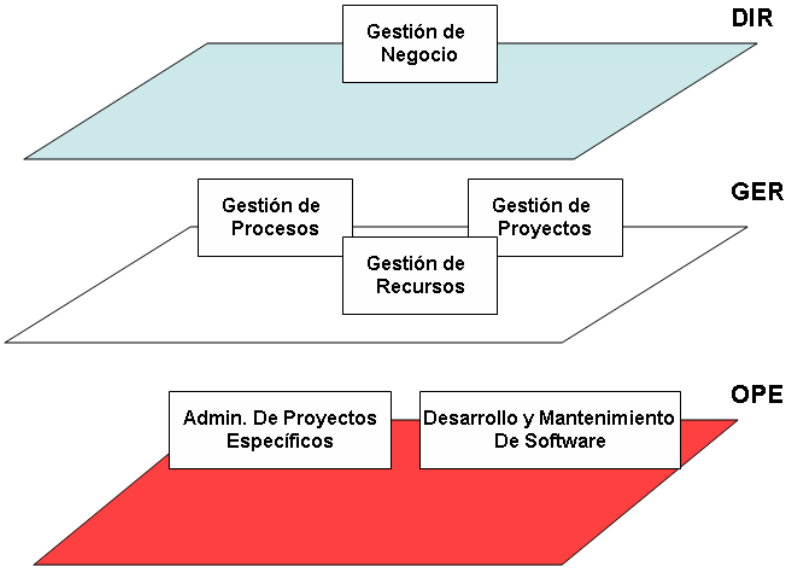


Figura 3.9 Categorías de Procesos MOPROSOFT

Las categorías reflejan la estructura de una organización y dentro de cada categoría existen procesos. Todos los procesos son definidos en términos de un propósito y de sus objetivos. Se describen a continuación los principales procesos por categoría.

Categoría de Alta Dirección (DIR)

Categoría de procesos que aborda las prácticas de Alta Dirección relacionadas con la gestión del negocio. Proporciona los lineamientos a los procesos de la categoría de Gerencia y se retroalimenta con la información generada por ellos.

Proceso de Gestión de Negocio

Propósito

Establecer la razón de ser de la organización, sus objetivos y las condiciones para lograrlos, para lo cual es necesario considerar las

necesidades de los clientes, así como evaluar los resultados para poder proponer cambios que permitan la mejora continua.

Adicionalmente habilita a la organización para responder a un ambiente de cambio y a sus miembros para trabajar en función de los objetivos establecidos

Objetivos

- O1 Lograr una planificación estratégica exitosa mediante el cumplimiento del Plan Estratégico.
- O2 Lograr que la organización trabaje en función del Plan Estratégico mediante la correcta comunicación e implantación del mismo.
- O3 Mejorar el Plan Estratégico mediante la implementación de la Propuesta de Mejoras.

Figura 3.10 Proceso de Gestión de Negocio

Categoría de Gerencia (GER)

Esta categoría aborda las prácticas de gestión de procesos, gestión de proyectos y gestión de recursos en función de los lineamientos establecidos en la categoría de Alta Dirección. Proporciona los elementos para el funcionamiento de los procesos de la categoría de Operación, recibe y evalúa la información generada por éstos y comunica los resultados a la categoría de Alta Dirección.

Proceso de Gestión de Procesos

Propósito

Establecer los procesos de la organización, en función de los Procesos Requeridos identificados en el Plan Estratégico. Así como definir, planear, e implantar las actividades de mejora en los mismos.

Objetivos

- O1 Planificar las actividades de definición, implantación y mejora de los procesos en función del Plan Estratégico.
- O2 Dar seguimiento a las actividades de definición, implantación y mejora de los procesos mediante el cumplimiento del Plan de Procesos.
- O3 Mejorar el desempeño de los procesos mediante el cumplimiento del Plan de Mejora.
- O4 Mantener informado a Gestión de Negocio sobre el desempeño de los procesos mediante el reporte Cuantitativo y Cualitativo.

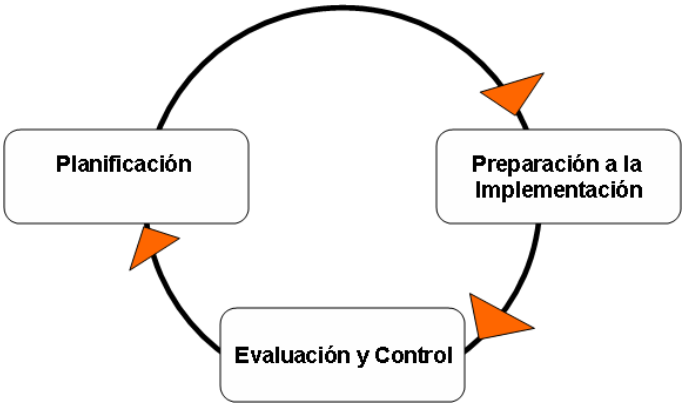


Figura 3.11 Proceso de Gestión de Procesos

Proceso de Gestión de Proyectos

Propósito

Asegurar que los proyectos contribuyan al cumplimiento de los objetivos y estrategias de la organización.

Objetivos

- O1 Cumplir con el Plan Estratégico de la organización mediante la generación e instrumentación de proyectos.
- O2 Mantener bajo control las actividades de Gestión de Proyectos mediante el cumplimiento del Plan de Gestión de Proyectos.
- O3 Proveer la información del desempeño de los proyectos a Gestión de Negocio mediante la generación del Reporte Cuantitativo y Cualitativo.
- O4 Atender los Comentarios y Quejas del Cliente mediante la definición y ejecución de Acciones Correctivas o Preventivas.

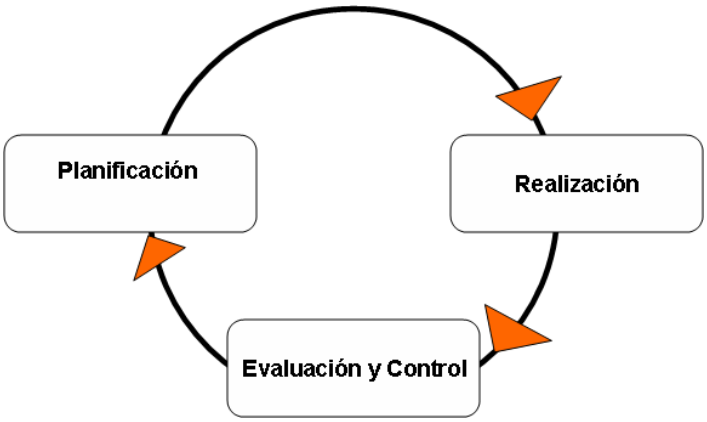


Figura 3.12 Proceso de Gestión de Proyectos

Proceso de Gestión de Recursos

Propósito

Conseguir y dotar a la organización de los recursos humanos, infraestructura, ambiente de trabajo y proveedores, así como crear y mantener la Base de Conocimiento de la organización. La finalidad es apoyar el cumplimiento de los objetivos del Plan Estratégico de la organización.

Objetivos

- O1 Lograr los objetivos del Plan Estratégico mediante la provisión de los recursos suficientes y calificados a la organización.

- O2 Proveer a los miembros de la organización de los medios y mecanismos adecuados para el uso y resguardo de la información mediante la Base de Conocimiento.
- O3 Mantener a la organización informada oportunamente sobre las tendencias tecnológicas mediante la elaboración de Propuestas Tecnológicas.

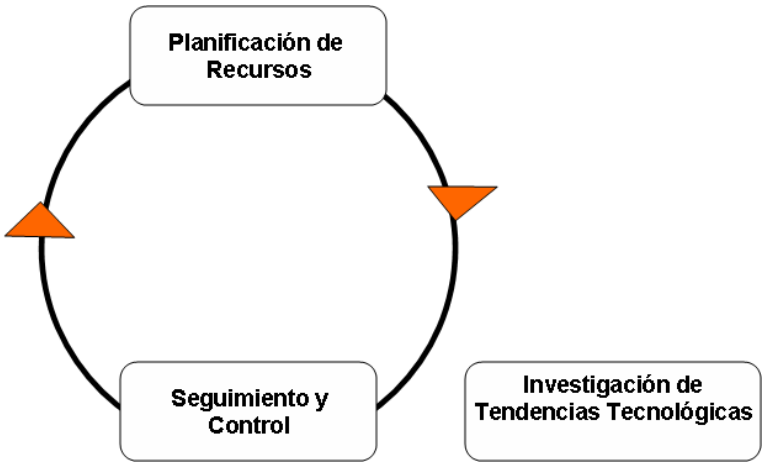


Figura 3.13 Proceso de Gestión de Recursos

Categoría de Operación (OPE)

Categoría de procesos que aborda las prácticas de los proyectos de desarrollo y mantenimiento de software. Esta categoría realiza las actividades de acuerdo a los elementos proporcionados por la categoría de Gerencia y entrega a ésta la información y productos generados.

Proceso de Administración de Proyectos Específicos

Propósito

Establecer y llevar a cabo sistemáticamente las actividades que permitan cumplir con los objetivos de un proyecto en tiempo y costo esperados.

Objetivos

- O1 Lograr los Objetivos del proyecto en tiempo y costo mediante la coordinación y el manejo de los recursos del mismo.
- O2 Mantener informado al Cliente mediante la realización de reuniones de avance del proyecto.

- O3 Atender las Solicitudes de Cambio del cliente mediante la recepción y análisis de las mismas.

Figura 3.14 Proceso de Administración de Proyectos Específicos

Proceso de Desarrollo y Mantenimiento de Software

Propósito

Es la realización sistemática de las actividades de análisis, diseño, construcción, integración y pruebas de productos de software nuevos o modificados cumpliendo con los requerimientos especificados.

Objetivos

- O1 Lograr que los productos de salida sean consistentes con los productos de entrada en cada fase de un ciclo de desarrollo mediante las actividades de verificación, validación o prueba.
- O2 Sustentar la realización de ciclos posteriores o proyectos de mantenimiento futuros mediante la integración de la Configuración de Software del ciclo actual.
- O3 Llevar a cabo las actividades de las fases de un ciclo mediante el cumplimiento del Plan de Desarrollo actual.

3.1.4.4 FORTALEZAS Y DEBILIDADES

Fortalezas

- Es específico para el desarrollo y mantenimiento de software.
- Es sencillo de entender y adoptar.
- Facilita el cumplimiento de los requisitos de otros modelos como ISO 9000 y CMMI.
- Es práctico en su aplicación, principalmente en organizaciones pequeñas, con bajos niveles de madurez.
- Además de ser un marco de referencia o certificación, está orientado a mejorar los procesos y así contribuir a los objetivos de negocio.
- Tiene un bajo costo, tanto para su adopción como para su evaluación.
- Cuenta con el nivel de detalle requerido en la definición de cada uno de sus procesos: Define Objetivos, Indicadores, Roles, Capacidades, Actividades, Tareas, Mecanismos de Verificación y Validación.
- Es la tendencia actual de desarrollo de software para la pequeña y mediana empresa a nivel hispanoamericano a través del proyecto COMPETISOFT.
- Es base de de la nueva norma ISO XYZ para desarrollo de Software dentro de la Pequeña y Mediana Empresa.

Debilidades

- No cuenta con un plan de difusión y capacitación explícito e institucionalizado tal como si existe con MPS.BR en Brasil.

3.2 DEFINICIÓN DE CRITERIOS DE EVALUACIÓN

Se tomaron como base los criterios de evaluación definidos por el programa PROSOFT para la industria del software de México [13] adaptándolos al contexto de la metodología académica de desarrollo de software a implementar. A continuación se describen los criterios resultantes:

- **A. ESPECÍFICO PARA LA GESTIÓN Y DESARROLLO DE SOFTWARE**

El modelo debe contar con un conjunto de procesos orientados específicamente a la gestión y desarrollo de software, objeto del presente trabajo de investigación. La metodología a ser implementada será ejecutada por alumnos de una o más carreras de una entidad académica, exclusivamente dentro de los cursos de gestión de proyectos e ingeniería de software en los que se encuentren matriculados.

La aplicación de la metodología contribuirá a evidenciar las buenas prácticas y/o necesidades de mejora en aspectos tales como pre requisitos en los cursos, necesidades de capacitación del equipo docente, reestructuración de la malla curricular, etc., relacionadas con la gestión y el desarrollo de software.

- **B. CONJUNTO DE CARACTERÍSTICAS MÍNIMAS POR PROCESO**

Cada proceso definido en el modelo debe contar como mínimo con las siguientes características:

- Objetivos, Indicadores y Metas específicas del proceso
- Documentos de entrada, salida y en proceso (de existir)
- Roles y Actividades por Roles dentro de cada proceso

Un modelo que cumpla con las características mínimas requeridas por proceso, facilitará la implementación de la metodología académica: se definirán tareas específicas para cada una de las actividades del modelo. La metodología definirá también productos específicos de entrada, salida y en proceso así como métricas de evaluación.

[13] http://www.tidap.gob.mx/Presentaciones/Material%20para%20Talleres/hannaoktabaMoProSoft.ppt

- **C. DEFINIDO COMO UN CONJUNTO DE PROCESOS**

El modelo debe indicar explícitamente las relaciones existentes entre cada uno de los procesos a nivel de productos de entrada, salida, roles y actividades. Se debe evidenciar también como los procesos contribuyen a que la organización alcance sus objetivos de negocio.

- **D. PRÁCTICO Y FÁCIL DE APLICAR**

Los documentos y actividades del modelo deben estar orientados a organizaciones con poca o nula experiencia en implementación de proyectos de desarrollo de software de manera estandarizada, de modo que su aprendizaje y aplicación sea fácil para equipos de trabajo integrados por estudiantes con conocimientos pero sin experiencia en actividades de gestión e ingeniería de software. La documentación tampoco debe ser excesiva, tomando en cuenta que los proyectos tendrán como duración promedio un ciclo de estudio: 17 semanas académicas en el caso de CIBERTEC.

- **E. ORIENTADO A MEJORAR LOS PROCESOS DEL MODELO**

El modelo debe incluir un proceso específico orientado a la mejora continua de cada uno de los procesos del modelo. Los objetivos de cada proceso deberán estar alineados a los objetivos de negocio de la organización que los implemente. El modelo en su conjunto debe contar con niveles de madurez basados en estándares internacionales que permitan evidenciar la evolución de la organización, llámese ésta entidad académica, en su camino hacia la institucionalización de las mejores prácticas de gestión e ingeniería de software.

- **F. DEBE DE TENER UN MECANISMO DE EVALUACIÓN**

Debe contar con un método de evaluación, basado en estándares internacionales, que permita determinar las fortalezas y debilidades de los procesos de gestión y desarrollo de software de una organización, así como el estado real de la misma para un periodo de vigencia específico.

3.2.1 FORMATO DE EVALUACIÓN

En base a los criterios identificados se elaboró un formato básico de evaluación. El esquema de valoración se basa en la presencia o ausencia del criterio evaluado. El modelo que cumpla con la mayor cantidad de criterios será seleccionado como modelo de referencia de la metodología a implementar.

3.3 APLICACIÓN DE LA EVALUACIÓN Y ANÁLISIS DE RESULTADOS

Se aplicó la evaluación a la norma ISO / IEC 12207 y los modelos CMMI, MPS.BR y MOPROSOFT. Los resultados muestran que el modelo mexicano MOPROSOFT cumple con todos los criterios de evaluación excepto el primero: MOPROSOFT es más amplio y soporta también actividades de mantenimiento de software.

Norma /Modelo / Criterio	ISO / IEC 12207	CMMI	MPR.BR	MOPROSOFT
Específico para la gestión y desarrollo De software	X	X	X	X
Características mínimas por Proceso	X	X	X	√
Definido como un conjunto de Procesos	√	√	√	√
Práctico y Fácil de Aplicar	X	X	√	√
Orientado a Mejorar los Procesos del Modelo	X	√	√	√
Debe tener un mecanismo de evaluación	X	√	√	√

Cuadro 3.3 Resultados de Evaluación de Modelos de Procesos de Software

La norma ISO / IEC 12207 y el modelo CMMI están definidos como un conjunto de procesos pero son bastante generales y difíciles de entender para una organización con poca o nula experiencia en el uso de estándares y buenas prácticas de gestión y desarrollo de software. Finalmente, el modelo MPS.BR, es práctico y fácil de aplicar, sin embargo, carece de las características mínimas requeridas dentro de sus procesos específicos.

3.4 SELECCIÓN DEL MODELO DE REFERENCIA

Del análisis previo, se puede concluir que el modelo que mejor cumple con los criterios de evaluación predefinidos es MOPROSOFT, por lo tanto, será el modelo de referencia que se utilizará para implementar la metodología académica de gestión y desarrollo de software.

Adicionalmente y como producto del análisis realizado sobre este modelo, se determinó que a nivel de definición general de proceso, cada uno de los procesos de este modelo cuenta con las siguientes características: Nombre de Proceso, Categoría, Propósito, Descripción, Objetivos, Indicadores, Metas cuantitativas, Subprocesos, Procesos relacionados, Entradas, Salidas, Productos Internos y Referencias bibliográficas. A nivel de prácticas, se identificó también que en cada proceso se definen como características: Roles involucrados y capacitación, Actividades, Verificaciones y Validaciones, Incorporación a la base de conocimiento, Recursos de Infraestructura, Mediciones, Situaciones Excepcionales y Lecciones Aprendidas.

Ambos aspectos, las características generales por proceso y las características a nivel de prácticas, servirán de estructura base en la definición de cada uno de los procesos de la metodología académica, pudiendo ser también modificados en atención al contexto específico de cada proceso.

3.5 RESUMEN DEL CAPÍTULO

En este capítulo se realiza inicialmente un análisis general de los más importantes modelos y normas de procesos de software vigentes actualmente a nivel mundial y regional. Se encuentran cosas muy interesantes como que existen aún muchas oportunidades de mejora: inicialmente los modelos no fueron concebidos para la pequeña y mediana empresa, menos aún para entidades académicas: surgen ante esta necesidad los modelos regionales MOPROSOFT en México y MPS.BR en Brasil. Realizado el proceso de análisis, se definen luego criterios de evaluación para seleccionar el modelo de referencia a utilizar en la implementación de la metodología académica objeto del presente trabajo de investigación. Como resultado de esta evaluación, se selecciona a MOPROSOFT como el mejor modelo de referencia, el cual, sin excluir las mejores prácticas de otros modelos, servirá de base, para implementar la metodología académica de desarrollo de Software: MEDESOFT.

Capítulo 4
MEDESOFT: Implementación de la metodología

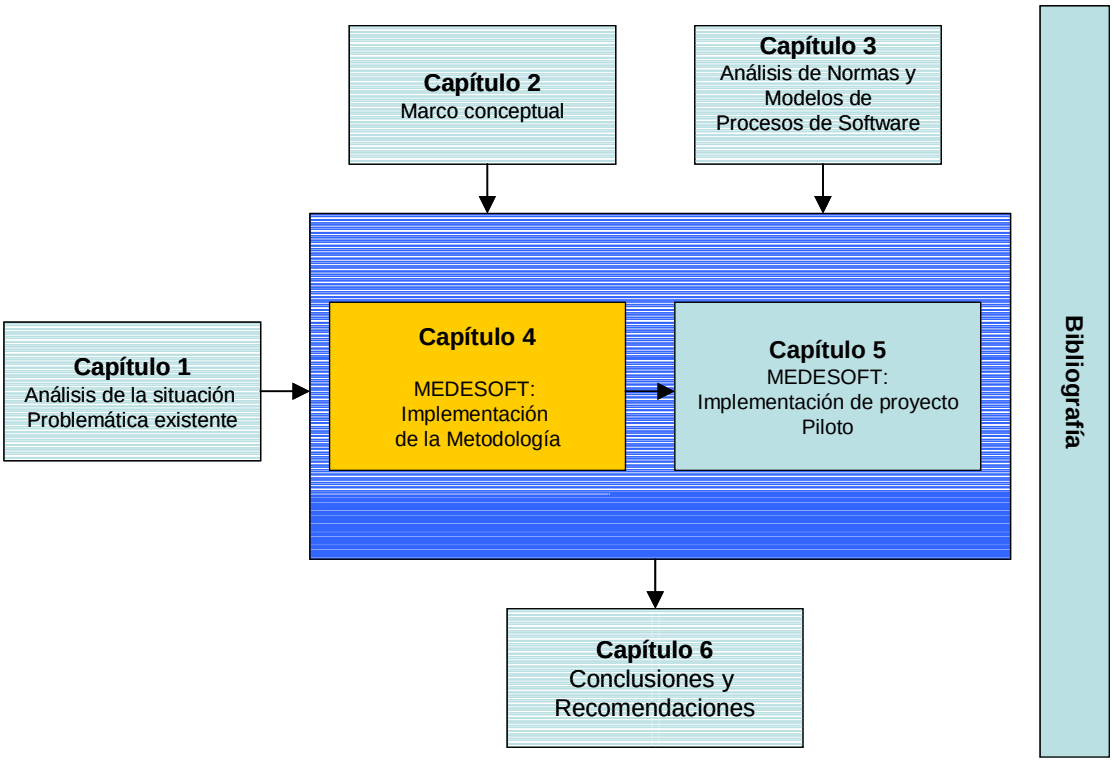


FIGURA 4.1 UBICACIÓN EN LA LECTURA DEL DOCUMENTO

En este capítulo se describe el proceso de implementación de la metodología académica de desarrollo de software MEDESOFT. La metodología se crea como resultado de un proceso de Planeamiento Estratégico realizado para la carrera de Computación e Informática del Instituto CIBERTEC, constituyéndose en una de sus líneas de acción.

Se definió la visión de la organización así como sus objetivos generales y específicos. Se realizó también un análisis FODA a partir del cual se generaron estrategias, líneas de acción y metas concretas. Finalmente, se detalla la estrategia

de implementación de la metodología en el plan táctico presentado como parte final de este capítulo.

Para Implementar la metodología MEDESOFT se siguieron una serie ordenada de pasos a través de los cuales se justifica su existencia y se evidencia su aporte a la organización en que se implemente. En la figura 4.2 se muestra la secuencia de actividades realizadas para implementar este proceso:

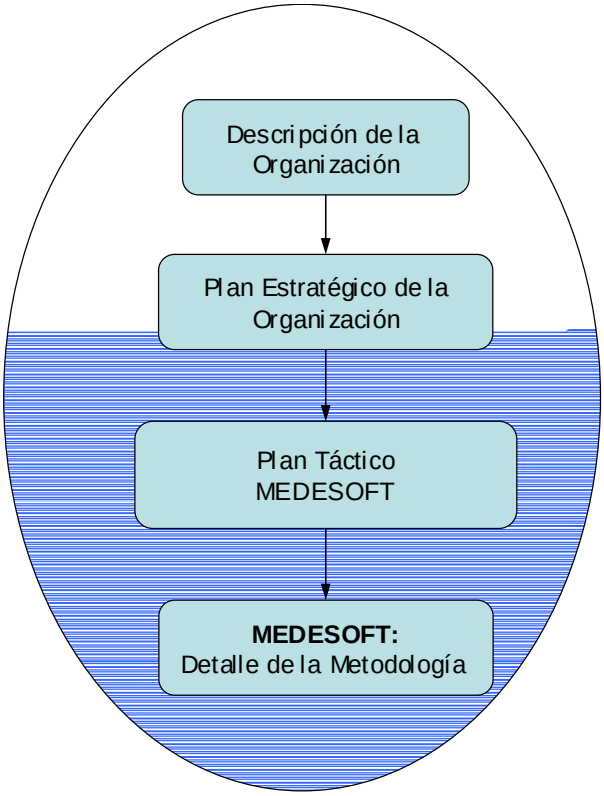


FIGURA 4.2 SECUENCIA DE ACTIVIDADES PARA IMPLEMENTAR LA METODOLOGÍA MEDESOFT

4.1 DESCRIPCIÓN DE LA ORGANIZACIÓN

4.1.1 INSTITUTO SUPERIOR TECNOLÓGICO CIBERTEC

4.1.1.1 RESEÑA HISTÓRICA

El instituto superior tecnológico CIBERTEC, en adelante CIBERTEC, es una institución educativa que cuenta con más de 25 años de experiencia en la Formación y Capacitación de profesionales en las diferentes áreas de las Tecnologías de la Información, Gestión y Negocios.

Se fundó en el año 1983 y, desde entonces, sus principales pilares de crecimiento son la permanente actualización e innovación de sus carreras profesionales, programas y cursos que van alineados con los avances tecnológicos que ocurren con el mundo; así como la calidad y el servicio de excelencia.

Actualmente es también centro autorizado de enseñanza en el Perú de importantes corporaciones de software, tales como Microsoft Corporation, Oracle y Cisco Systems. También ha realizado un convenio con el Ministerio de la Producción del Perú, por el cual se convierte en el primer centro de innovación de las nuevas tecnologías de la información, orientado a fomentar el uso de las mismas entre las PYMES.

CIBERTEC pertenece al Grupo Educativo UPC (Universidad Peruana de Ciencias Aplicadas), el cual desde septiembre del año 2004 es parte de Laureate Internacional Universities, red privada internacional de instituciones de educación superior.

En la actualidad CIBERTEC cuenta con dos Unidades de Negocios: Carreras Técnicas y Extensión Profesional. Dentro de la unidad de negocios de Carreras Técnicas existen actualmente dos escuelas con sus respectivas carreras profesionales:

• Escuela de Tecnologías de Información
  - ✓ Carrera de Computación e Informática
  - ✓ Carrera de Administración y Sistemas
  - ✓ Carrera de Diseño Gráfico
  - ✓ Carrera de Electrónica
  - ✓ Carrera de Redes y Comunicaciones

• Escuela de Gestión de Negocios
  - ✓ Carrera de Administración
  - ✓ Carrera de Contabilidad

- ✓ Carrera de Negocios Internacionales
- ✓ Carrera de Marketing

4.1.1.2 CARRERA DE COMPUTACIÓN E INFORMÁTICA - CCI

La carrera de Computación e Informática, a la cual denominaremos en adelante CCI, se orienta a formar profesionales técnicos capaces de diseñar e implementar soluciones de tecnologías de información. El perfil de desempeño de la carrera permitirá a sus egresados:

- Definir la estructura de los sistemas de información de una organización para explotar sus ventajas competitivas.
- Gestionar las etapas del desarrollo de los sistemas (análisis y diseño, construcción, pruebas, implementación y mantenimiento).
- Diseñar y desarrollar sistemas de información para los diversos niveles de la organización.
- Crear aplicaciones en diferentes plataformas para reducir los tiempos de operación.

La CCI tiene una duración de tres años, y como resultado del planeamiento estratégico de la misma, se detectó la necesidad de implementar una metodología académica de desarrollo de Software que contribuya a alcanzar los Objetivos Educacionales de la carrera, evidenciando las capacidades de sus alumnos y egresados así como integrando vertical y transversalmente sus cursos, a través de la implementación de proyectos de software.

Se detalla a continuación el proceso de planeamiento estratégico realizado sobre la CCI.

4.2 PLANEAMIENTO ESTRATÉGICO DE LA CCI

El proceso de planeamiento estratégico de la CCI, involucró el desarrollo de las fases mostradas a continuación en la figura 4.3:

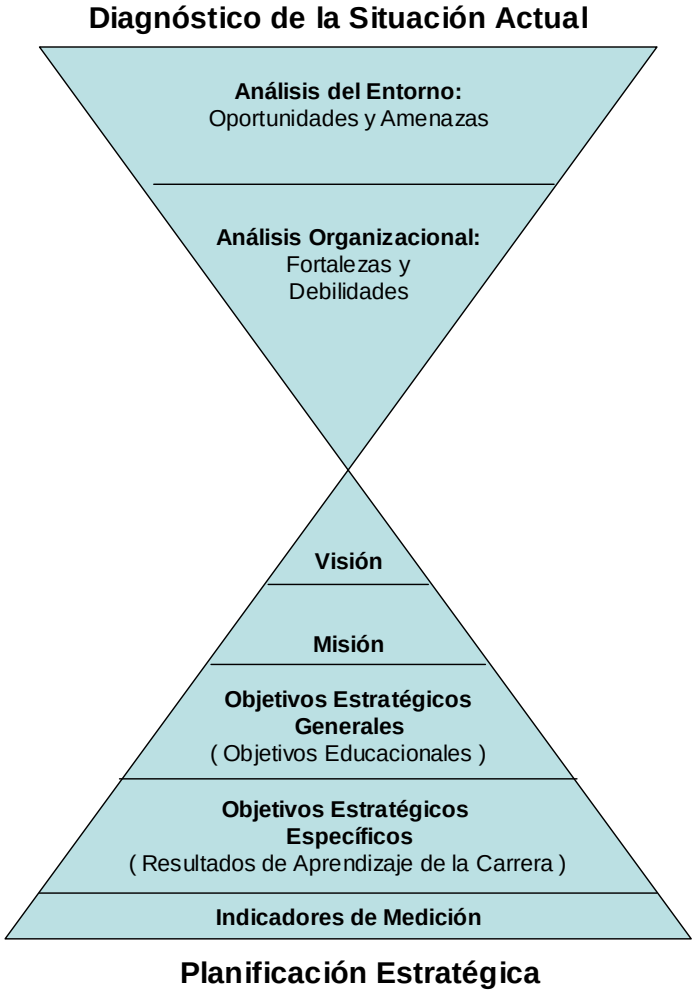


FIGURA 4.3 PROCESO DE PLANEAMIENTO ESTRATÉGICO

4.2.1 DIAGNÓSTICO DE LA SITUACIÓN ACTUAL

4.2.1.1 ANÁLISIS EXTERNO

Entre los principales factores del entorno general de la CCI tenemos los siguientes:

Factores Económicos

1. Estabilidad de los indicadores económicos del país durante los últimos años.

2. Mejora del poder adquisitivo de la población.

3. Adecuado nivel de capitales internacionales al contar con un factor riesgo-país bajo (Los capitales son selectivos en su selección de riesgo). Esto repercute positivamente en la industria nacional, generándose mayores puestos de trabajo y capacidad para invertir en educación.

Factores Geográficos

1. Servicios de telecomunicaciones e infraestructura inician un proceso de descentralización, sin embargo, aún existe una gran concentración en las grandes ciudades. Las zonas rurales no están adecuadamente atendidas.

2. Geografía nacional de difícil acceso por lo que la mayoría de servicios están aún centralizados en las principales ciudades del país, especialmente en la costa peruana.

Factores Demográficos

1. Población peruana principalmente joven.

2. Distribución bastante irregular concentrada principalmente en la capital y ciudades de la costa con excepción de la ciudad de Arequipa, ubicada en la parte Sur y Central del Perú.[14]

Factores Político – Legales

1. Marco legal del sector de Educación en pleno proceso de reestructuración.

[14] Fuente: http://www.unfpa.org.pe/infosd/poblacion/poblacion_02.htm

Factores Tecnológicos

1. Nuevos servicios y tecnologías, con altas tasas de crecimiento y con la creación de nuevos mercados.

2. Rapidez de los cambios tecnológicos.

3. Tendencia a la convergencia en tecnologías, servicios y empresas.

Dentro del entorno específico de la CCI se tiene:

¿Quiénes son sus clientes?

1. Jóvenes recién egresados del colegio.

2. Alumnos universitarios o de otros institutos que se trasladan a CIBERTEC.

3. Personas adultas que desean formalizar sus conocimientos en TI.

¿Qué otras instituciones prestan servicios similares?

1. Universidades con carreras de Ingeniería de Sistemas o Similares.

2. Otros Institutos Superiores Tecnológicos con carreras de Computación e Informática o similares.

Las oportunidades y amenazas identificadas son:

Principales Oportunidades para aprovechar

O1. El actual contexto mundial fomenta el uso intensivo de las nuevas Tecnologías de Información y Comunicación (TIC) lo que implica también mayores necesidades de capacitación e investigación en tecnologías de información.

O2. Escaso número de entidades académicas con carreras de tecnología y con un prestigio ganado como el de CIBERTEC y la CCI.

O3. Mercados potenciales en diferentes zonas urbanas de la capital y en provincias, inclusive se puede replicar la experiencia exitosa de CIBERTEC en otros países.

O4. Posibilidad de brindar nuevos y variados servicios a los estudiantes como alcanzar certificaciones internacionales dentro de la CCI.

O5. Ser parte de la red internacional de educación Laureate Internacional Universities.

Principales Amenazas que neutralizar

A1. Posible inestabilidad económica del país producto de la actual recesión mundial.

A2. Perder cuota de mercado por insatisfacción de los estudiantes de la CCI de no contar con un programa de actualización curricular ágil y acorde con los cambios y evolución de las Tecnologías de Información.

A3. Interés de entidades académicas en ingresar al mercado con carreras similares a la CCI: nuevos competidores en el sector.

A4. Fuga de información: procedimientos, esquema de trabajo utilizado dentro de la CCI.

A5. Competencia capte recursos humanos calificados de la CCI.

A6. Dados los bajos costos de algunas universidades, los estudiantes pueden preferir una carrera universitaria de Ingeniería a una técnica de Computación e Informática.

4.2.1.2 ANÁLISIS INTERNO

Las fortalezas y debilidades identificadas son las siguientes:

Principales Fortalezas a utilizar

F1. Disponibilidad de Material Didáctico y Bibliográfico actualizado para la CCI.

F2. Disponibilidad de una excelente infraestructura de soporte para el dictado de los cursos de la CCI (aulas, laboratorios, hardware, software).

F3. Experiencia en implementación de proyectos por alumnos para empresas reales dentro de los cursos de la CCI (Últimos ciclos). De esta manera se evidencian los logros de los cursos y sus prerrequisitos.

F4. Staff de docentes altamente calificado y de comprobada experiencia profesional.

F5. Alto nivel de especialización y actualización en los cursos de la CCI, en especial en los siguientes: Análisis y Diseño de Sistemas, Diseño de Proyectos, Calidad de Software, Desarrollo de Aplicaciones Web 1 y 2, Dispositivos Móviles 1 y 2, Lenguaje de Programación 1 y 2).

F6. Prestigio de la CCI y de sus egresados dentro del mercado peruano.

F7. Identificación de los estudiantes y docentes con la institución y la CCI.

Principales Debilidades a superar

D1. Insuficiente integración vertical y transversal entre los cursos de la CCI a nivel de prerrequisitos y proyectos respectivamente.

D2. Necesidad de mayores evidencias, indicadores y métricas para verificar los logros individuales de los cursos y los objetivos de aprendizaje de la CCI.

D3. No existe un procedimiento formal para el desarrollo de proyectos de software por alumnos dentro de la CCI.

D4. Los profesores de CCI carecen en general, de tiempo para dedicarse a actividades de investigación.

D5. No se cuenta con un apoyo formalizado de la organización para tareas de investigación del equipo docente y estudiantes de la CCI.

4.2.1.3 MATRIX FODA

La matriz FODA es el instrumento de planificación estratégica que nos permite obtener una primera aproximación a una estrategia organizacional. Nos muestra cuatro tipos de estrategias alternativas derivadas de la interacción de los cuatro grupos de variables constituidos por las Fortalezas y Debilidades (factores Internos a la organización), así como las Oportunidades y Amenazas (factores externos a la organización).

Después realizar el análisis interno y externo de la CCI, se procedió a elaborar la matriz FODA, cuyo formato se muestra a continuación en el cuadro 4.1

Factores Internos / Factores Externos	**Relación de Fortalezas**	**Relación de Debilidades**
	F1. **F2.** **F3.** **…** **Fn.**	**D1.** **D2.** **D3.** **…** **Dn.**
Relación de Oportunidades **O1.** **O2.** **O3.** **…** **On.**	***FO (Maxi – Maxi)*** Estrategias para Maximizar las Fortalezas (F) y las Oportunidades (O)	***DO (Mini – Maxi)*** Estrategias para Minimizar las Debilidades (D) y Maximizar las Oportunidades (O)
Relación de Amenazas **A1.** **A2.** **A3.** **…** **An.**	***FA (Maxi – Mini)*** Estrategias para Maximizar las Fortalezas (F) y Minimizar las Amenazas (A)	***DO (Mini – Mini)*** Estrategias para Minimizar las Debilidades (D) y Minimizar las Amenazas (A)

CUADRO 4.1 MATRIZ FODA

4.2.1.4 ESTRATEGIAS DERIVADAS DE LA MATRIX FODA

Como resultado del análisis de las variables involucradas, se obtuvieron las siguientes estrategias para la CCI:

Estrategias DA – Debilidades vs. Amenazas (Mini-Mini). En general, el objetivo de estas estrategias es el de minimizar tanto las debilidades como las amenazas.

- **DA1** - Evaluar los procesos actuales de la CCI y redefinirlos de ser necesario (D1, D2, D3 - A2, A3).
- **DA2** - Revisión inmediata y coordinada de los prerequisitos de los cursos y proyectos de la CCI en sus diferentes niveles (D1 - A2, A3).
- **DA3** - Revisión inmediata y coordinada de las entradas y salidas requeridas en los proyectos integrados de cursos del mismo nivel en la CCI (D1 - A2, A3).
- **DA4** - Implementar un plan de desarrollo personal y capacitación continua para todo el personal de la CCI, acorde a sus funciones y responsabilidades dentro de la misma. (D4, D5 - A3, A4).

Estrategias DO – Debilidades vs. Oportunidades (Mini-Maxi). Intentan minimizar las debilidades y maximizar las oportunidades.

- **DO1** - Fomentar la investigación de docentes y estudiantes a través de proyectos de investigación de aplicación práctica dentro del campo de la ingeniería de software (D5 - O1, O2).
- **DO2** - Fomentar el desarrollo de tareas de investigación del equipo docente de la CCI mediante la asignación de horas como parte de su carga académica ordinaria (D5 - O1, O2).
- **DO3** - Implementar una metodología académica de desarrollo de software que integre vertical y transversalmente los proyectos de software de los cursos de la CCI (D1, D2, D3 - O2, O4).
- **DO4** - Definir un procedimiento de evaluación y mejora continua de la CCI, que cuente con indicadores y métricas para cada uno de los Objetivos de Aprendizaje de la Carrera. (D1, D2, D3 - O1, O4).
- **DO5** - Normar la publicación obligatoria de los resultados de las investigaciones académicas realizadas por alumnos y docentes de la CCI (Revistas científicas internas y externas) (D5 - 01, 02).

- **DO6** - Fomentar la capacitación del equipo docente de la CCI bajo condiciones preferenciales dentro de la institución y organizaciones pertenecientes a la red Laureate (D4 - O1, O5).

Las estrategias **DO3** y **DO4** se encuentran directamente relacionadas con la Metodología Académica de Desarrollo de Software MEDESOFT, y se convertirán posteriormente en Líneas de Acción concretas que contribuyan al logro de los objetivos de la CCI.

Estrategias FA – Fortalezas vs. Amenazas (Maxi-Mini). Estas estrategias se basan en las fortalezas de la organización que pueden hacer frente a las amenazas del medio ambiente externo.

- **FA1** - Difundir las políticas y reglas de confidencialidad de cumplimiento obligatorio, por parte del personal administrativo y docente, de la CCI. (F6 - A4).

- **FA2** - Formalizar el procedimiento de actualización curricular semestral así como la existencia de un comité consultivo para la CCI integrado por empleadores, docentes, egresados y estudiantes destacados (F4 - A2).

- **FA3** - Difundir los convenios existentes de complementación académica nacionales e internacionales de la CCI y establecer nuevos de ser factible. (F6 - A5, A6).

- **FA4** - Desarrollar capacitaciones y actividades para que el personal docente y administrativo de la CCI, entienda e interiorice la importancia del servicio al cliente (F4, F6, F7 - A2, A3, A5).

- **FA5** - Desarrollar actividades de integración para la comunidad estudiantil, docentes y personal administrativo de la CCI que fomenten los valores de la institución (F1,F2, F4, F6 - A3, A5, A6)

Estrategias FO – Fortalezas vs. Oportunidades (Maxi-Maxi). Estrategias orientadas a maximizar tanto las fortalezas como las oportunidades de la organización.

- **FO1** - Fomentar el intercambio estudiantil y de docentes de la CCI con las entidades académicas pertenecientes a la red Laureate, como parte integral de la formación de los estudiantes de la CCI (F4, F6 – O5).

- **FO2** - Firmar convenios académicos con las principales casas de software a nivel mundial (MICROSOFT, CISCO, SUN, IBM entre otros) para

obtener beneficios de capacitación y descuentos en exámenes de certificación internacionales (F3 - 04).

- **FO3** - Estudios del mercado en el extranjero, en nuevas zonas urbanas de Lima y principales ciudades en provincias, evaluar su potencialidad para definir la apertura de nuevas sedes (F1, F2, F4, F6 – O2, O3).

- **FO4** - Afianzar las relaciones con entidades educativas de nivel secundario y el marketing interno y externo de la CCI para promocionarla dentro de nuestro mercado objetivo (F1, F2, F4, F6 – O1).

Se muestra a continuación, en los cuadros 4.2 y 4.3, el resumen de las estrategias generadas en cada uno de los cuadrantes de la matriz FODA:

		Fortalezas	**Debilidades**
		F1. Material Didáctico y Bibliográfico actualizado para la CCI. F2. Excelente infraestructura de soporte para el dictado de la carrera de CCI. F3. Experiencia en implementación de proyectos por alumnos para empresas reales dentro de los cursos de la CCI. F4. Staff de docentes altamente calificado y de comprobada experiencia profesional. F5. Alto nivel de especialización y actualización en los cursos de la CCI F6. Prestigio de la CCI y de sus egresados dentro del mercado peruano. F7. Identificación de los estudiantes y docentes con la institución y la CCI.	D1. Insuficiente integración vertical y transversal entre los cursos de la CCI a nivel de prerrequisitos y proyectos. D2. Necesidad de mayores evidencias, indicadores y métricas para verificar los logros de aprendizaje de la CCI. D3. No existe un procedimiento formal para el desarrollo de proyectos de software por alumnos dentro de la CCI. D4. Los profesores de CCI carecen en general, de tiempo para dedicarse a actividades de investigación. D5. No se cuenta con un apoyo formalizado de la organización para tareas de investigación en la CCI.
Oportunidades		**FO (MAXI - MAXI)**	**DO (MINI - MAXI)**
	O1. Contexto mundial fomenta el uso intensivo de las nuevas Tecnologías de Información y Comunicación (TIC)	FO1. Fomentar el intercambio estudiantil y de docentes (F4,F6-O5)	DO1. Fomentar la investigación de docentes y estudiantes a través de proyectos de investigación (D5 - O1,O2)
	O2. Pocas organizaciones con carreras de TI y con un prestigio ganado como el de CIBERTEC y la CCI	FO2. Firmar convenios académicos con las principales casas de software a nivel mundial (F3 - 04)	DO2. Asignación de carga horaria para tareas de investigación del equipo docente (D5 - O1,O2)
	O3. Mercados potenciales en diferentes zonas urbanas de la capital, provincias y en el extranjero	FO3. Estudios del mercado en el extranjero, nuevas zonas urbanas de Lima y en provincias (F1,F2,F4,F6 - O2,O3)	DO3. Implementar una metodología académica de desarrollo de software (D1,D2,D3 - O2,O4)
	O4. Posibilidad de brindar nuevos y variados servicios a los estudiantes y docentes de la CCI .	FO4. Afianzar relaciones con entidades educacativas de nivel secundario y el marketing interno y externo de la carrera de CCI (F1,F2,F4,F6- O1)	DO4. Definir un procedimiento de evaluación y mejora continua de la CCI. (D1,D2,D3 - O1,O4) DO5. Publicación de los resultados de las investigaciones (Revistas científicas internas y externas). (D5 - 01,02)
	O5. Ser parte de la red internacional de educación Laureate Internacional Universities.		DO6. Fomentar la capacitación del equipo docente (D4 - O1,O5)

CUADRO 4.2 MATRIX FODA – CUADRANTES FO Y DO

		Fortalezas	Debilidades
		F1. Material Didáctico y Bibliográfico actualizado para la CCI. F2. Excelente infraestructura de soporte para el dictado de la carrera de CCI. F3. Experiencia en implementación de proyectos por alumnos para empresas reales dentro de los cursos de la CCI. F4. Staff de docentes altamente calificado y de comprobada experiencia profesional. F5. Alto nivel de especialización y actualización en los cursos de la CCI F6. Prestigio de la CCI y de sus egresados dentro del mercado peruano. F7. Identificación de los estudiantes y docentes con la institución y la CCI.	D1. Insuficiente integración vertical y transversal entre los cursos de la CCI a nivel de prerrequisitos y proyectos. D2. Necesidad de mayores evidencias, indicadores y métricas para verificar los logros de aprendizaje de la CCI. D3. No existe un procedimiento formal para el desarrollo de proyectos de software por alumnos dentro de la CCI. D4. Los profesores de CCI carecen en general, de tiempo para dedicarse a actividades de investigación. D5. No se cuenta con un apoyo formalizado de la organización para tareas de investigación en la CCI.
Amenazas		**FA (MAXI - MINI)**	**DA (MINI - MINI)**
	A1. Posible Inestabilidad económica del país producto de la actual recesión mundial.	Difundir las políticas y reglas de confidencialidad de cumplimiento obligatorio (F6 - A4)	Evaluar los procesos actuales de la CCI y redefinirlos de ser necesario. (D1,D2,D3 - A2,A3)
	A2. Perder cuota de mercado por insatisfacción de estudiantes de la CCI .	Formalizar el procedimiento de actualización curricular semestral (F4 - A2)	Revisión de los prerequisitos de los cursos y proyectos de la CCI (D1 - A2,A3)
	A3. Interés de entidades académicas en ingresar al mercado con carreras similares a la CCI: nuevos competidores.	Difundir los convenios existentes de complementación académica (F6 - A5,A6)	Revisión de las entradas y salidas requeridas en los proyectos integrados de cursos en la CCI (D1 - A2,A3)
	A4. Fuga de información: procedimientos, esquema de trabajo utilizado dentro de la CCI.	Desarrollar capacitaciones para el personal de la CCI (F4,F6,F7 - A2,A3,A5)	Implementar un plan de desarrollo personal y capacitación continua para todo el personal de la CCI (D4,D5 - A3,A4)
	A5. Competencia capte recursos humanos calificados de la CCI.	Desarrollar actividades de integración para la comunidad estudiantil y personal de la CCI (F1,F2,F4,F6 - A3,A5,A6)	
	A6. Los estudiantes pueden preferir una carrera universitaria de Ingeniería a una técnica de Computación e Informática.		

CUADRO 4.3 MATRIX FODA – CUADRANTES FA Y DA

4.2.2 PLANIFICACIÓN ESTRATÉGICA

Dentro del proceso de planificación estratégica, la declaración de la visión y la misión es el paso más importante. Una declaración efectiva de la filosofía de la organización sirve de marco de referencia para la adopción de decisiones estratégicas.

La visión y la misión, constituyen para la carrera de Computación e Informática – CCI, el nivel más alto de una pirámide de Resultados de Aprendizaje, la cual se muestra a continuación:

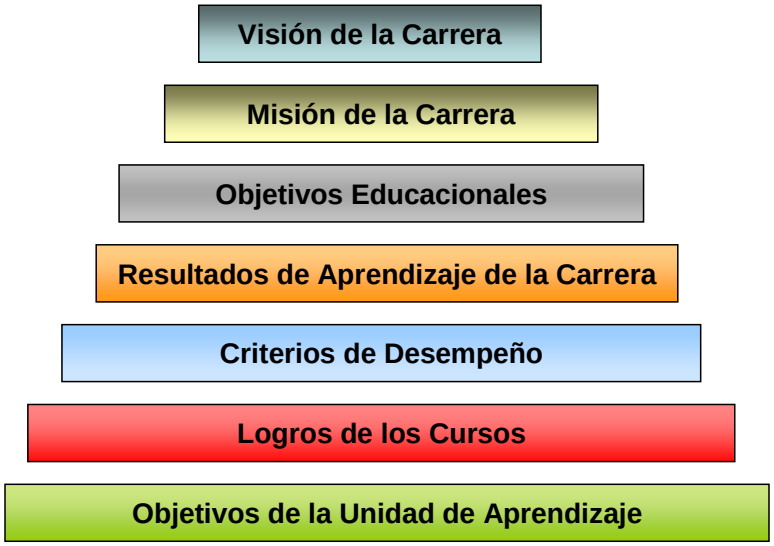


FIGURA 4.4 PIRÁMIDE DE RESULTADOS DE APRENDIZAJE DE LA CARRERA DE COMPUTACIÓN E INFORMÁTICA - CCI

La pirámide de resultados de aprendizaje, refleja como desde los objetivos de la unidad de aprendizaje de un curso en particular, se contribuye al logro de los objetivos educacionales de la carrera y en consecuencia el logro de su misión y de la visión de la institución.

Durante el presente trabajo de investigación, se definen en orden descendente, la visión de CIBERTEC, la misión de la carrera de Computación e Informática, sus Objetivos Educacionales y los Resultados de Aprendizaje que los soportan. Finalmente se detallan los Criterios de Desempeño definidos para los resultados de aprendizaje relacionadas con MEDESOFT.

Los criterios de desempeño permitirán identificar indicadores de medición que evidencien el logro de los Resultados de Aprendizaje relacionados con la Metodología Académica de Desarrollo de Software a implementar.

4.2.2.1 LA VISIÓN

La visión es la imagen futura que una organización desarrolla sobre sí misma y sobre la realidad sobre la cual trabaja. Por lo general la visión incluye tanto los cambios que deseamos lograr en el seno de nuestra población objetivo, como la imagen objetivo de la propia institución. La visión de la CCI es la siguiente:

> "Somos reconocidos como la carrera técnica líder, a nivel nacional e internacional, en la formación de profesionales con un alto valor ético y competentes en la implementación de sistemas de información, en la concepción, diseño y construcción de productos software de alta calidad, con una sólida base en las tecnologías existentes en las ciencias de la computación."

La visión previamente descrita, expresa las aspiraciones colectivas de todas las personas que trabajan en la CCI, las mismas que son convergentes con las aspiraciones de sus directivos. Esta visión los une, dirige y diferencia de otras entidades académicas.

4.2.2.2 LA MISIÓN

La misión: La misión de la organización debe reflejar lo que la organización es, haciendo alusión directa a la función general y específica que cumple como instancia de gestión empresarial. La aplicación principal de la misión es servir como una guía interna para quienes toman las decisiones importantes, y para que todos los proyectos y actividades puedan ser puestos a prueba en su compatibilidad con la misma.

En este contexto, se definió la misión de la CCI:

> "Formamos profesionales técnicos competentes capaces de desarrollar sistemas de información para lograr soluciones integrales que contribuyan con el incremento de la productividad de las organizaciones, sobre la base del manejo de las tecnologías de la información y el desarrollo de competencias personales y laborales".

4.2.2.3 OBJETIVOS EDUCACIONALES

Los Objetivos Educacionales (OE) son parte del perfil del egresado y definen el desempeño profesional esperado de los graduados. Están basados en las competencias obtenidas por el graduado.

Contexto: Entre uno y tres años iniciales de ejercicio profesional.

A continuación se describen los Objetivos Educacionales de la carrera de Computación e Informática:

- OE-1. Profesionales competentes. Los egresados de la CCI implementan las etapas del desarrollo de los sistemas de software (captura de requisitos, análisis y diseño, construcción, pruebas, despliegue y mantenimiento) con una visión integral del entorno en que se desenvuelven.

- OE-2. Desarrollo de soluciones integrales. Los egresados de la CCI desarrollan soluciones de software alineadas con los objetivos estratégicos del negocio.

- OE-3. Incremento de la productividad de las organizaciones. Los egresados de la CCI proponen soluciones tecnológicas integrales que contribuyen a elevar la eficiencia de los procesos de las organizaciones.

- OE-4. Dominio de las TI. Los egresados de la CCI son capaces de aplicar las herramientas de TI más convenientes de acuerdo con las necesidades de la organización.

- OE-5. Competencias personales y laborales. Los egresados de la CCI poseen habilidades comunicativas y analíticas que les permiten solucionar problemas relacionados con su área de conocimiento. Toman decisiones con sentido ético respetando los valores y principios de la organización, y sus acciones se orientan a su desarrollo integral y permanente.

4.2.2.4 RESULTADOS DE APRENDIZAJE DE LA CARRERA - RAC

Los Resultados de aprendizaje de la carrera (RAC) son parte del perfil del egresado al culminar el sexto ciclo de estudio de la carrera. Definen lo que el estudiante conoce y es capaz de hacer al momento de graduarse.

Contexto: Al momento de graduarse.

A continuación se describen los Resultados de Aprendizaje de la carrera de Computación e Informática:

- RAC.1 Conocimiento del ciclo de desarrollo de software. Se desempeñan asumiendo diferentes roles, en los equipos de proyectos de desarrollo de las soluciones de software que realizan.

- RAC.2 Participación en el aseguramiento de la calidad del software. Participa de la planificación, diseño y comprobación de la calidad del proceso de desarrollo y del producto software mediante actividades de verificación, validación y pruebas que permitan asegurar la calidad de los entregables.

- RAC.3 Define estrategias de implementación de soluciones de Software. Participa en la definición de las mejores estrategias de implementación de una solución de software en base a los requisitos y expectativas del área usuaria.

- RAC.4 Innovación y orientación a resultados. Plantean propuestas innovadoras de software que se integran con los sistemas existentes.

- RAC.5 Trabajo en equipo. Trabajan como parte de un equipo de desarrollo utilizando las buenas prácticas de ingeniería software para lograr sinergias entre todos los involucrados con el propósito de poder entregar software de calidad.

- RAC.6 Aprendizaje autónomo. Manejan nuevas y diversas tecnologías para crear aplicaciones que se requieran, y utilizan diversas herramientas de desarrollo, y de base de datos con la finalidad de mejorar su competencia profesional.

- RAC.7 Desempeño profesional. Posee y valora hábitos de trabajo efectivo, así como evidencia el liderazgo necesario para comunicarse en diversos niveles con los diferentes involucrados de un entorno laboral común.

- RAC.8 Análisis y comunicación asertiva. Comunican eficazmente sus propuestas de solución sobre la base del análisis de información requerida para la toma de decisiones en la organización.

- RAC.9 Responsabilidad ética y profesional. Toman decisiones con sentido ético considerando el desarrollo social y el principio del bien común, y sus acciones están orientadas al permanente crecimiento personal y profesional.

Los Resultados de Aprendizaje de la Carrera (RAC) contribuyen a alcanzar los Objetivos Educacionales (OE) de la misma. Se deben definir líneas de acción y metas concretas para los Resultados de Aprendizaje (Objetivos Específicos) en función de los Objetivos Educacionales a los que aportan (Objetivos Generales).

Se muestra en el cuadro 4.4, la relación entre los Resultados de Aprendizaje y los Objetivos Educacionales de la CCI:

Resultados del Aprendizaje de La Carrera de Computación e Informática - CCI		Objetivos Educacionales				
		OE-1	OE-2	OE-3	OE-4	OE-5
RAC.1	Conocimiento del ciclo de desarrollo de software	✓	✓			
RAC.2	Participación en el aseguramiento de la calidad del software	✓	✓	✓		
RAC.3	Estrategias de Implementación de soluciones de software		✓	✓		
RAC.4	Innovación y orientación a resultados		✓	✓		
RAC.5	Trabajo en equipo			✓		
RAC.6	Aprendizaje Autónomo	✓		✓	✓	✓
RAC.7	Desempeño Profesional	✓				✓
RAC.8	Análisis y Comunicación Asertiva	✓	✓	✓		✓
RAC.9	Responsabilidad ética y Profesional	✓				✓

CUADRO 4.4 RELACIÓN ENTRE RESULTADOS DE APRENDIZAJE Y OBJETIVOS EDUCACIONALES

4.2.2.5 LÍNEAS DE ACCIÓN Y METAS

Dentro del contexto de la carrera de Computación e Informática, se seleccionaron los Resultados de Aprendizaje que estén directamente relacionados con la implementación de la metodología de Desarrollo de Software Académica MEDESOFT:

- **RAC.1** Conocimiento del ciclo de desarrollo de software. Se desempeñan asumiendo diferentes roles, en los equipos de proyectos de desarrollo de las soluciones de software que realizan.

- **RAC.2** Participación en el aseguramiento de la calidad del software. Participa de la planificación, diseño y comprobación de la calidad del proceso de desarrollo y del producto software mediante actividades de verificación, validación y pruebas que permitan asegurar la calidad de los entregables.

Como siguiente paso, se definen las líneas de acción y metas específicas que contribuyen a evidenciar el logro de los Resultados de Aprendizaje seleccionados.

LÍNEAS DE ACCIÓN

LA1. Implementar una metodología académica de desarrollo de software, a la cual se denominará MEDESOFT, que integre vertical y transversalmente los proyectos de software de los cursos de la CCI, que cuente con niveles de madurez y que permita evidenciar los resultados de Aprendizaje de la Carrera.

LA2. Definir un procedimiento de evaluación y mejora continua de la CCI, que cuente con indicadores y métricas para cada uno de los Resultados de Aprendizaje de la Carrera.

METAS

M1. Culminar el procedimiento de Desarrollo de Software por Alumnos de MEDESOFT al finalizar el ciclo académico 2008 - 1. MEDESOFT versión 1.0.

M2. Poner en práctica MEDESOFT versión 1.0, durante el ciclo académico 2008-2, con los alumnos pertenecientes al Sexto de la CCI.

M3. Culminar el procedimiento de Gestión de Proyectos de Software por Alumnos de MEDESOFT al finalizar el ciclo académico 2009 - 2. MEDESOFT versión 1.1

M4. Poner en práctica MEDESOFT versión 1.1, durante el ciclo académico 2010-1, con los alumnos pertenecientes a la CCI.

M5. Culminar el procedimiento de evaluación y mejora continua de la CCI en su primera versión, al iniciar el ciclo académico 2010-1.

4.2.2.6 LOS INDICADORES DE MEDICIÓN

Dada la complejidad que puede presentar un plan estratégico, no sería factible evaluarlo tomando sólo como base las proposiciones literales de los Objetivos Educacionales y Resultados de Aprendizaje de la carrera. Debido a ello, es necesario recurrir a expresiones susceptibles de ser medidas, estas expresiones son conocidas como indicadores.

Se definieron los indicadores de medición de los Resultados de Aprendizaje **RAC.1** y **RAC.2** como parte de un enfoque general de calidad implementado para la CCI.

Este enfoque consta de las siguientes etapas:

- ✓ Definición de Criterios de Desempeño.
- ✓ Definición de Indicadores de Medición e Identificación de Requerimientos de Calidad del Usuario.
- ✓ Especificación y Diseño de la Evaluación.
- ✓ Ejecución de la Evaluación.
- ✓ Retroalimentación a la Organización.

Definición de Criterios de Desempeño

Los criterios de desempeño describen los requisitos de calidad para el resultado obtenido por un alumno en su desempeño académico. Permiten establecer si el estudiante alcanza o no el resultado de aprendizaje (RAC) al cual están asociados.

Se detallan a continuación los criterios de desempeño directamente relacionados con la metodología MEDESOFT:

RAC.1 Conocimiento del ciclo de desarrollo de software. Los criterios de desempeño definidos para este resultado de Aprendizaje son los siguientes:

> **CD1.** Identifica el modelo de ciclo de vida de software más apropiado a ser implementado en el desarrollo del producto software, considerando las características del proyecto y las necesidades del cliente.

CD2. Define los procesos de desarrollo y de gestión que deberán ejecutarse durante el proyecto de software sobre la base de estándares universalmente aceptados.

CD3. Elabora los entregables pertenecientes a los procesos de identificación y análisis de requisitos, pertenecientes a la etapa de análisis del ciclo de vida de desarrollo del software.

CD4. Elabora los entregables correspondientes a los procesos de diseño de arquitectura y de diseño detallado del software pertenecientes a la etapa de diseño del ciclo de vida de desarrollo de software.

CD5. Elabora los entregables correspondientes a los procesos de construcción y aseguramiento de calidad pertenecientes a las etapas de desarrollo y pruebas del ciclo de vida de desarrollo de software.

RAC.2 Participación en el aseguramiento de la calidad del Software. Se describen a continuación los criterios de desempeño definidos para este resultado de Aprendizaje:

CD1. Participa de la planificación de un proyecto de desarrollo de software considerando el alcance, el tiempo y el costo del mismo, utilizando buenas prácticas de gestión de proyectos.

CD2. Identifica entregables y documentos del ciclo de vida que deberán ser verificados para asegurar la calidad del proceso de desarrollo de software; identificando para cada caso el tipo de revisión que se aplicará.

CD3. Diseña casos de prueba que permitan validar las funciones implementadas por un producto software.

CD4. Identifica métricas de calidad de producto software que deberán ser medidas como parte de las pruebas de elementos no funcionales.

CD5. Identifica, prioriza y analiza riesgos que pueden afectar a un proyecto de software. Define estrategias y líneas de acción para maximizar las oportunidades y minimizar las amenazas que puedan afectar el éxito del proyecto.

CD6. Planifica las pruebas de software que permiten validar el producto software contra los requerimientos funcionales y no funcionales definidos conjuntamente con los usuarios finales.

CD7. Propone acciones correctivas frente a posibles desviaciones con la planificación del alcance, el tiempo y costo del proyecto.

Los criterios de desempeño son la base para que un evaluador juzgue si un estudiante es, o aún no, competente. Permiten precisar acerca de lo que se hizo y al asociarlos a indicadores, permiten evaluar la calidad con que fue realizado.

Definición de Indicadores de Medición e Identificación de Requerimientos de Calidad del Usuario

Dentro del contexto académico, cada criterio de desempeño ha sido expresado como un indicador de medición. Cada indicador cuenta con un rango de valores posibles.

Se determinan también las necesidades del área usuaria, es decir, el área de Acreditación de CIBERTEC. Esta área determinó el valor esperado del conjunto de valores posibles del indicador. El valor esperado es el requerimiento de calidad del Usuario.

Se muestra a continuación el detalle de indicadores, valores posibles y valor esperado para los resultados de aprendizaje RAC.1 y RAC.2:

RAC1. Conocimiento del ciclo de desarrollo de software Se desempeña asumiendo diferentes roles en los equipos de proyectos de desarrollo de las soluciones de software que realizan.			
Criterios de Desempeño (Indicadores)	Alternativas de Selección	Valores Posibles	Valor Esperado Area Usuaria
CD1. Identifica el modelo de ciclo de vida de software más apropiado a ser implementado en el desarrollo del producto software, considerando las características del proyecto y las necesidades del cliente.	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD2. Define los procesos de desarrollo y de gestión que deberán ejecutarse durante el proyecto de software sobre la base de estándares universalmente aceptados	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD3. Elabora los entregables pertenecientes a los procesos de identificación y análisis de requisitos, pertenecientes a la etapa de análisis del ciclo de vida de desarrollo del software	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD4. Elabora los entregables correspondientes a los procesos de diseño de arquitectura y de diseño detallado del software pertenecientes a la etapa de diseño del ciclo de vida de desarrollo de software.	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD5. Elabora los entregables correspondientes a los procesos de construcción y aseguramiento de calidad pertenecientes a las etapas de desarrollo y pruebas del ciclo de vida de desarrollo de software	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	

CUADRO 4.5 INDICADORES Y REQUERIMIENTOS DE CALIDAD DEL RESULTADO DE APRENDIZAJE RAC.1

RAC2. **Participación en el aseguramiento de la calidad del Software** Participa de la planificacion diseño y comprobacion de la calidad del proceso de desarrollo y del producto software mediante actividades de verificacion, validacion y pruebas que permitan asegurar la calidad de los entregables			
Criterios de Desempeño (Indicadores)	Alternativas de Selección	Valores Posibles	Valor Esperado Area Usuaria
CD1. Participa de la planificación de un proyecto de desarrollo de software considerando el alcance, el tiempo y el costo del mismo, utilizando buenas prácticas de gestión de proyectos	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD2. Identifica entregables y documentos del ciclo de vida que deberán ser verificados para asegurar la calidad del proceso de desarrollo de software; identificando para cada caso el tipo de revisión que se aplicará	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD3. Diseña casos de prueba que permitan validar las funciones implementadas por un producto software.	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD4. Identifica métricas de calidad de producto software que deberán ser medidas como parte de las pruebas de elementos no funcionales.	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD5. Identifica, prioriza y analiza riesgos que pueden afectar a un proyecto de software. Define estrategias y líneas de acción para maximizar las oportunidades y minizar las amenazas que puedan afectar el éxito del proyecto.	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD6. Planifica las pruebas de software que permiten validar el producto software contra los requerimientos funcionales y no funcionales definidos conjuntamente con los usuarios finales.	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	
CD7. Propone acciones correctivas frente a posibles desviaciones con la planificación del alcance, el tiempo y costo del proyecto.	Cumple Satisfactoriamente	17 - 20	✓
	Cumple	13 - 16	
	Cumple Medianamente	10 - 12	
	No Cumple	0 - 9	

CUADRO 4.6 INDICADORES Y REQUERIMIENTOS DE CALIDAD DEL RESULTADO DE APRENDIZAJE RAC.2

Estos indicadores son evaluados inicialmente, a través de una encuesta académica aplicada a los alumnos del Sexto Ciclo de la CCI. La información obtenida, proporciona a la institución una primera apreciación de cómo los estudiantes se ven a sí mismos dentro del proceso de aprendizaje. Sin embargo esta información es subjetiva.

Por lo tanto, los resultados obtenidos serán contrastados con los resultantes de aplicar las métricas del proceso de Desarrollo de Software por Alumnos (PDSA).

Especificación y Diseño de la Evaluación

En esta etapa se identifican las métricas a ser aplicadas para determinar si se logran satisfacer las necesidades de calidad definidas por el área usuaria en el paso previo.

A continuación se detallan los elementos constitutivos de cada métrica correspondiente a la encuesta académica a ser aplicada a los alumnos al finalizar el Sexto Ciclo de la CCI:

Nombre de la métrica. Es el nombre del indicador asociado a los resultados de aprendizaje RAC.1 o RAC.2.

Audiencia objetivo. Alumnos del sexto ciclo de la carrera de Computación e Informática.

Método de Aplicación. Se redacta el indicador en forma de pregunta, dentro del contexto de una encuesta académica dirigida a la audiencia objetivo.

Se muestra a continuación, la redacción resultante del Criterio de Desempeño CD4 asociado al Resultado de Aprendizaje RAC.1:

Considera que, al finalizar la Carrera de Computación e Informática, usted conoce y es capaz de:

1. Elaborar los entregables correspondientes a los procesos de diseño de arquitectura y diseño detallado de software pertenecientes a la etapa de diseño del ciclo de vida de desarrollo de software.

A) Cumple Satisfactoriamente

B) Cumple

C) Cumple Medianamente

D) No cumple

Frecuencia de evaluación. Una vez por ciclo, al culminar el ciclo.

Entrada para la evaluación. Encuesta Académica.

Medición, Fórmula y Cálculo de elementos de datos. Se definió la siguiente fórmula de evaluación:

$$\mathbf{X} = \frac{A * (n1) + B * (n2) + C * (n3) + D * (n4)}{N}$$

Donde:

N = Número total de alumnos encuestados.

A = Peso asociado a la alternativa A: 20 puntos.

B = Peso asociado a la alternativa B: 15 puntos.

C = Peso asociado a la alternativa C: 10 puntos.

D = Peso asociado a la alternativa D: 5 puntos.

n1 = número de alumnos que seleccionaron la alternativa A.

n2 = número de alumnos que seleccionaron la alternativa B.

n3 = número de alumnos que seleccionaron la alternativa C.

n4 = número de alumnos que seleccionaron la alternativa D.

X = Resultado de la evaluación expresado en una escala vigesimal.

Interpretación del valor medido.

$5 <= X <= 20$

Donde:

Veinte o lo más cercano a veinte es lo mejor. El valor esperado por el área usuaria oscila entre Diecisiete y Veinte.

Los resultados obtenidos de la aplicación de la encuesta académica, se contrastan con los obtenidos de las métricas correspondientes al proceso de Desarrollo de Software por Alumnos (PDSA): métricas relacionadas con actividades de verificación, validación y pruebas realizadas por los alumnos de la CCI durante la implementación de un proyecto de Desarrollo de Software.

Las métricas son aplicadas al finalizar cada etapa del ciclo de desarrollo de software y al finalizar el ciclo de desarrollo completo.

Ejecución de la evaluación

Se debe ejecutar la evaluación usando como referencia la especificación y diseño de la evaluación definidas en el paso previo. La ejecución de la encuesta académica es parte de los procedimientos pertenecientes al área usuaria.

La ejecución de las métricas del Proceso de Desarrollo de Software por Alumnos (PDSA), es parte del conjunto de actividades regulares, realizadas dentro de los cursos involucrados en un proyecto de Desarrollo de Software utilizando MEDESOFT.

Después de aplicar la encuesta académica, y obtener los resultados por cada Criterio de Desempeño, se debe calcular el promedio respectivo por Resultado de Aprendizaje. La fórmula a aplicar será la siguiente:

$$Z_j = \frac{\sum CD_i.R_j}{N_j}$$

Donde:

$CD_i.R_j$ = Promedio de evaluación obtenido por el Criterio de desempeño **i** del Resultado de Aprendizaje **j**.

N_j = Número de Criterios de Desempeño del Resultado de Aprendizaje **j**.

Z_j = Promedio de evaluación obtenido por el Resultado de Aprendizaje **j**.

La variable i oscila entre uno y cinco para el RAC.1

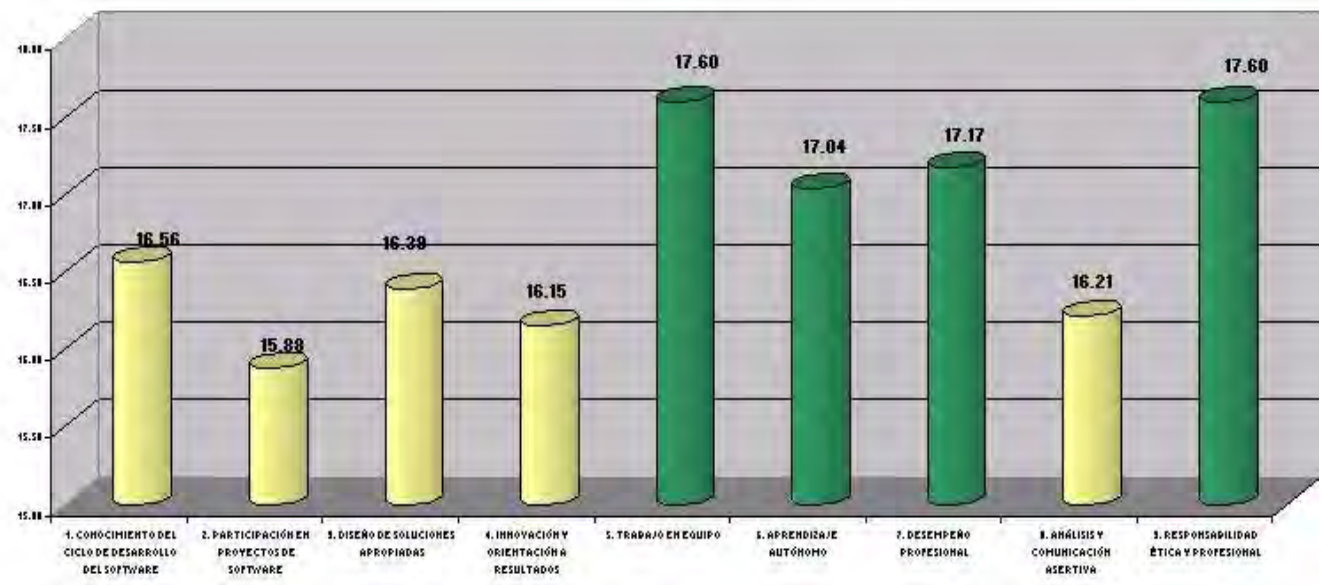
La variable i oscila entre uno y siete para el RAC.2

De manera análoga, los resultados obtenidos de las métricas del Proceso de Desarrollo de Software (PDSA) son consolidadas para su posterior comparación e interpretación. El detalle de las métricas, su forma de aplicación y consolidación se define en la descripción detallada de la metodología MEDESOFT realizada en el presente documento.

Retroalimentación a la organización

Una vez que todas las mediciones han sido completadas, estas deben ser documentadas así como las conclusiones generadas en un reporte de evaluación. Este reporte debe incluir tanto los resultados de la encuesta académica como los obtenidos de las métricas pertenecientes a los procesos de la metodología de desarrollo de software MEDESOFT.

Se muestra a continuación el resultado obtenido de las encuestas a los alumnos, para cada uno de los Resultados de Aprendizaje de la carrera (RACs) durante el ciclo académico 2008-1.



CUADRO 4.7 RESULTADOS DE ENCUESTAS DE CRITERIOS DE DESEMPEÑO – CICLO ACADÉMICO 2008-1

Se evidencia que entre otros, los resultados de aprendizaje:

- RAC.1 **Conocimiento del ciclo de desarrollo de software**. Se desempeñan asumiendo diferentes roles, en los equipos de proyectos de desarrollo de las soluciones de software que realizan.

- RAC.2 **Participación en el aseguramiento de la calidad del software**. Participa de la planificación, diseño y comprobación de la calidad del proceso de desarrollo y del producto software mediante actividades de verificación, validación y pruebas que permitan asegurar la calidad de los entregables

Son susceptibles de mejora (16.56 y 15.88 puntos respectivamente), a la cual contribuirá la metodología MEDESOFT con su aplicación a través de:

- Las buenas prácticas a ejecutar por los alumnos dentro de su proceso de desarrollo de Software.
- La evaluación del proceso a través de métricas objetivas que serán contrastadas con los resultados de la encuesta académica realizada a los alumnos al culminar la carrera. Esto, en el caso particular de CIBERTEC, a partir del ciclo académico 2008-2.

4.2.3 PROCEDIMIENTO DE EVALUACIÓN Y MEJORA CONTINUA

Con el objetivo de institucionalizar una cultura de calidad y gestionar adecuadamente los procedimientos académicos de la carrera de Computación e Informática, se definió un procedimiento de evaluación y mejora continua. Se describen a continuación y de manera general, los miembros participantes y actividades básicas.

4.2.3.1 CONSTITUYENTES

Son todos aquellos involucrados con la CCI. Son consultados sobre la misión, visión, objetivos educacionales, resultados de aprendizaje y en general, cualquier tema académico de la carrera.

Los constituyentes de la carrera de Computación e Informática están integrados por diferentes comités, los cuales se enumeran a continuación:

- Comité de Egresados
- Comité de Alumnos
- Comité de Docentes
- Comité de Empleadores

4.2.3.2 ACTIVIDADES PRINCIPALES

a) Definición y/o Revisión de la Visión y Objetivos Educacionales de la carrera.

Participantes: Todos los Comités y Autoridades de la Carrera

Periodicidad: Cada dos años o en plazos previos a solicitud de las Autoridades de la Carrera.

Los participantes definen y/o revisan la misión y objetivos Educacionales de la carrera. Los objetivos educacionales se derivan del profesional ideal propuesto por los participantes, tomando en cuenta las necesidades del mercado actual.

Se evalúa el logro de un objetivo educacional en base a los reportes de evaluación de los Resultados de Aprendizaje asociados a dicho Objetivo Educacional.

b) Definición y/o Revisión de Resultados de Aprendizaje

Participantes: Comité de Docentes, Comité de Empleadores y Autoridades de la Carrera

Periodicidad: Cada 6 meses.

Los participantes definen y/o revisan los Resultados de Aprendizaje de la Carrera.

Se evalúa el logro de un Resultado de Aprendizaje en base a los reportes de evaluación de los criterios de desempeño asociados a dicho Resultado de Aprendizaje.

c) Definición y/o Revisión de los Cursos de la CCI.

Participantes: Comité de Docentes, Coordinadores de Cursos y Autoridades de la Carrera

Periodicidad: Durante el ciclo Académico.

Los participantes definen y/o revisan los Logros y Contenidos de los Cursos de la Carrera.

El logro del curso debe contribuir al logro de uno o más Resultados de Aprendizaje.

Se evalúa el logro de un curso en particular en base a los reportes de evaluación de los logros parciales del curso específico.

Los resultados generados en cada una de las actividades principales se reflejarán directamente como mejoras en:

- El plan de estudios de la carrera,
- La estructura y contenidos de los cursos (sílabos, material de apoyo, etc.)
- Plana Docente.
- Equipamiento e Infraestructura.
- Actividades de Bienestar Estudiantil y Calidad Educativa en beneficio de docentes y alumnos de la carrera.

Se muestra a continuación en la figura 4.5 el resumen del procedimiento propuesto:

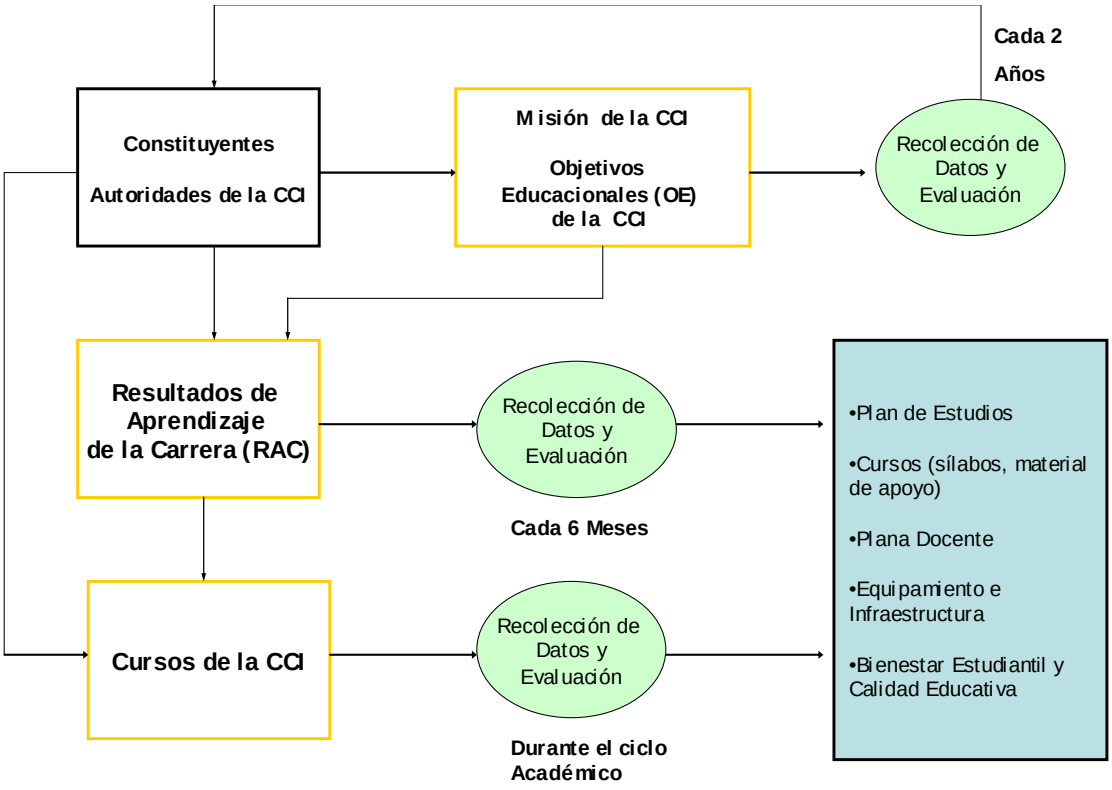


FIGURA 4.5 PROCEDIMIENTO DE EVALUACIÓN Y MEJORA CONTINUA DE LA CCI

4.3 PLAN TÁCTICO PARA IMPLEMENTAR LA METODOLOGÍA MEDESOFT

El plan táctico para implementar la metodología propuesta consiste de un proyecto técnico que involucra varias fases, las cuales permitirán implementar los procesos de la metodología, identificando niveles de madurez dentro de ella, que deberán ser alcanzados progresivamente por los alumnos de la institución en que se implemente, en este caso, los alumnos de la CCI.

4.3.1 ESTRATEGIA DE IMPLEMENTACIÓN DE LA METODOLOGÍA MEDESOFT

Se definen el tiempo y el alcance como factores de éxito en la implementación de MEDESOFT: hasta que no esté plenamente definido un proceso, dentro de los plazos establecidos, no se proseguirá con otro. Un proceso se considera plenamente definido cuando ha sido probado y se ha comprobado su correcto funcionamiento a través de los resultados obtenidos de la ejecución de sus métricas de evaluación.

La planificación de procesos a conseguir en cada plazo establecido debe ser incremental, de tal manera que lo conseguido en una fase sea la base para la siguiente.

Las fases a seguir son las siguientes:

- Establecer el proceso de Desarrollo de Software por Alumnos
- Establecer el proceso de Gestión de Proyectos Específicos por Alumnos
- Establecer el proceso de Gestión de Proyectos
- Establecer el Proceso de Gestión de Procesos

4.3.1.1 ESTABLECER EL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS

La culminación exitosa de esta fase representará la versión 1.0 de la metodología Académica de Desarrollo de Software MEDESOFT.

Propósito:

-Definir el proceso de desarrollo de software por alumnos (PDSA).

-La realización sistemática por los alumnos, dentro del ciclo académico de:

- ✓ Actividades de aseguramiento de calidad de productos software, cumpliendo con los requerimientos especificados para un proyecto de desarrollo de software.
- ✓ Actividades de captura de requisitos, análisis de requisitos, diseño de la solución, construcción, integración y pruebas de productos de software, cumpliendo con los requerimientos especificados para un proyecto de desarrollo de software.

-Formalizar la implementación de proyectos integradores dentro de los cursos de la carrera de computación e informática.

-Definir niveles de madurez para el proceso PDSA de modo que pueda ser utilizado por los alumnos de la carrera de computación e informática.

Plazo para culminar la definición del proceso: Al finalizar el ciclo académico 2008-1.

Plazo para la implementación del primer proyecto piloto: Al finalizar el ciclo académico 2008-2

Plazo para la definición de niveles de madurez para el proceso PDSA: Al culminar el ciclo académico 2009-1.

4.3.1.2 ESTABLECER EL PROCESO DE GESTIÓN DE PROYECTOS ESPECÍFICOS POR ALUMNOS

La culminación exitosa de esta fase representará la versión 1.1 de la Metodología Académica de Desarrollo de Software MEDESOFT.

Propósito:

Definir el proceso de Gestión de Proyectos Específicos por Alumnos (PGPA).

Definir niveles de madurez para el proceso PGPA de modo que pueda ser utilizado por los alumnos de la carrera de computación e informática.

La realización sistemática por los alumnos, dentro del ciclo académico, de:

- ✓ Actividades de auditoria que permitan determinar hasta que punto se han cumplido con las actividades de aseguramiento de calidad y de ingeniería de software del proyecto integrado correspondiente al quinto ciclo de la CCI.

- ✓ Actividades que permitan cumplir con los objetivos de un proyecto en tiempo y costo esperados

Plazo para culminar la definición del proceso: Al finalizar el ciclo académico 2009-2.

Plazo para la definición de niveles de madurez para el proceso: Al iniciar el ciclo académico 2010-1

Plazo para la implementación del primer proyecto piloto: Al finalizar el ciclo académico 2010-1. El proyecto incluye los procesos de las fases previas.

4.3.1.3 ESTABLECER EL PROCESO DE GESTIÓN DE PROYECTOS

La culminación exitosa de esta fase representará la versión 1.2 de la Metodología Académica de Desarrollo de Software MEDESOFT.

Propósito:

Definir el proceso de Gestión de Proyectos (PGPY).

La realización sistemática por los docentes y autoridades de la organización de:

- ✓ Actividades que asegurar que los proyectos implementados por los alumnos contribuyan al cumplimiento de los objetivos y estrategias de la organización.

Plazo para culminar la definición del proceso: Al finalizar el ciclo académico 2010-1.

Plazo para la implementación del primer proyecto piloto: Al finalizar el ciclo académico 20010-2. El proyecto incluye los procesos de las fases previas.

4.3.1.4 ESTABLECER EL PROCESO DE GESTIÓN DE PROCESOS

La culminación exitosa de esta fase representará la versión 1.3 de la Metodología Académica de Desarrollo de Software MEDESOFT.

Propósito:

Definir el proceso de Gestión de Procesos (PGPR).

La realización sistemática por los docentes, autoridades y especialistas de la organización de:

- ✓ Actividades que permitan establecer los procesos de la organización, en función de los procesos requeridos identificados en el Plan Estratégico. También definen, planifican, e implantan las actividades de mejora en los mismos.

Plazo para culminar la definición del proceso: Al finalizar el ciclo académico 2011-1.

Plazo para la implementación del primer proyecto piloto: Al finalizar el ciclo académico 20011-2. El proyecto incluye los procesos de las fases previas.

4.4 METODOLOGÍA ACADÉMICA DE DESARROLLO DE SOFTWARE - MEDESOFT

4.4.1 PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS (PDSA)

En conformidad con la estrategia de implementación descrita en el plan táctico MEDESOFT, se definió el proceso de desarrollo de software por alumnos (PDSA) dentro de los plazos establecidos para la FASE 1.

El proceso referencia documentos de entrada, documentos de salida y documentos internos cuyos formatos y ejemplos serán detallados en los anexos del presente trabajo de investigación.

Se describe a continuación el detalle del proceso:

4.4.1.1 DEFINICIÓN GENERAL DEL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS (PDSA)

I. Propósito

La realización sistemática por los alumnos, dentro del ciclo académico, de:

- ✓ Actividades de aseguramiento de calidad de productos software, cumpliendo con los requerimientos especificados para un proyecto real de desarrollo de software.
- ✓ Actividades de captura de requisitos, análisis de requisitos, diseño de la solución, construcción, integración y pruebas, de productos de software, cumpliendo con los requerimientos especificados para un proyecto real de desarrollo de software.

II. Objetivos

O1 Lograr que los productos de salida sean consistentes con los productos de entrada en cada fase de un ciclo de desarrollo mediante las actividades de verificación, validación o prueba.

III. Descripción

El proceso de Desarrollo de Software por alumnos (PDSA) se compone de uno o más ciclos de desarrollo. Cada ciclo está compuesto de las siguientes fases:

III.a Inicio:

Revisión del Plan de Proyecto – PP por los alumnos miembros del Equipo de trabajo para lograr un entendimiento común del proyecto.

III.b Identificación de requerimientos:

- *Actividades de Ingeniería de Software:* Con el documento Reporte de Solicitud de Requerimientos – RSRQ elaborado por el líder usuario, los alumnos identifican los requisitos funcionales y no funcionales, a partir de los cuales se determinan las funciones de software que les darán el soporte, logrando de esta forma, conseguir un entendimiento común entre el cliente y el proveedor. También se identifican las reglas de negocio. A partir de las funciones identificadas, los alumnos, especifican la implementación de cada una de ellas. En dicha especificación los alumnos deberán hacer uso de las reglas de negocio. Acompañarán a dicha especificación, los prototipos y el detalle de los controles correspondientes. Como resultado se obtiene el documento:
    - Reporte de Especificaciones de Software v1 – RES v1

- Se verifica y valida el Reporte de Especificaciones de Software v1 – RES v1 sobre la base de los siguientes documentos:
    - Reporte de Solicitud de Requerimientos - RSRQ

  Los alumnos identifican y documentan los Casos de Prueba en base a la especificación de casos de uso y a los prototipos previamente definidos en el Reporte de Especificaciones de Software v1 - RES v1. Los Casos de Prueba se anexarán al Plan de Pruebas del Sistema - PPRS.

  Como resultado se obtienen los documentos:
    - Plan de Pruebas del Sistema – PPRS
        - Casos de Prueba
    - Reporte de Verificación - RVE
    - Reporte de Validación – RVA

III.c Análisis de Requisitos:

- *Actividades de Ingeniería de Software:* A partir de las especificaciones definidas en la etapa anterior, los alumnos, complementan las mismas con diagramas de clase y de secuencia (Modelo de análisis). Como resultado se obtiene la siguiente documentación:
    - Reporte de Especificaciones de Software v2 – RES v2

- *Actividades de Aseguramiento de Calidad:* Se verifica y valida el Reporte de Especificaciones Software v2 sobre la base del siguiente documento:
    - Reporte de Especificaciones de Software v1 –RES v1

  Como resultado se obtiene la siguiente documentación:
    - Reporte de Verificación - RVE
    - Reporte de Validación - RVA

III.d Definición de Arquitectura y Diseño Detallado de la Solución de Software:

- *Actividades de Ingeniería de Software:* Los alumnos identifican y documentan la arquitectura de hardware y software base sobre la cual se implementará la solución. Usando patrones de diseño y tomando como referencia el modelo de análisis del Reporte de Especificaciones de Software v2 - RES v2, identifican las clases de diseño y las interfaces necesarias para implementar el modelo de análisis propuesto. Como resultado se obtiene la siguiente documentación:
    - Reporte de Diseño de Software – RDS
        - Arquitectura de Software

- *Actividades de Aseguramiento de Calidad*: Se Verifica y valida el Reporte de Diseño de Software – RDS, sobre la base del siguiente documento:
    - Reporte de Especificaciones de Software v2 – RES v2

III.e Construcción:

Actividades de Ingeniería de Software: Los alumnos programan cada uno de los componentes de software diseñados en la etapa anterior. La

programación se realiza utilizando estándares previamente identificados y aprobados. Como resultados de esta etapa se obtienen:

- Los componentes de software (programas fuente).

Actividades de Aseguramiento de Calidad: Los alumnos aplican lo estipulado en el plan de pruebas de Sistema – PPRS, utilizando los casos de prueba. El objetivo es probar los casos de uso de manera unitaria.

Luego de ejecutar las pruebas unitarias se deberá generar el documento de:

- Reporte de pruebas unitarias – RPRU

III.f Pruebas:

Actividades de Ingeniería de Software: Una vez que los componentes de software han sido programados se procede con la integración de los mismos. Se debe considerar una arquitectura modular, generando el producto software que debiera satisfacer las necesidades del área usuaria.

Como resultado de esta etapa se deben producir los siguientes documentos:

- Manual de usuario - MUSU

Actividades de Aseguramiento de Calidad: Los alumnos deberán ejecutar los casos de prueba, definidos en el plan de pruebas de Sistema - PPRS.

Como resultado de las actividades de aseguramiento de calidad se debe generar el documento:

- Reporte de Pruebas de Sistema - RPRS

III.g Cierre:

Se procede con la integración final de la configuración de software generada en cada una las fases para su entrega final.

Se identifican y documentan las Lecciones Aprendidas, generando el reporte de mediciones y sugerencias de Mejora.

Como resultado de esta etapa se deben producir los siguientes documentos:

- Manual de usuario - MUSU

Para generar los productos de cada una de estas fases se realizan las siguientes actividades:

- ✓ Distribución de tareas. Se asignan las responsabilidades de cada alumno miembro del equipo de trabajo de acuerdo al Plan de Proyecto - PP.
- ✓ Producción, verificación, validación o prueba de los productos entregables, así como su corrección correspondiente.
- ✓ Generación del reporte de actividades - RACT al finalizar cada una de las fases.

IV. Indicadores y métricas

Una vez determinado el objetivo del proceso, se definieron indicadores que faciliten su evaluación. Los indicadores del proceso son los siguientes:

- ✓ I1. En cada fase de un ciclo de desarrollo se efectúan todas las actividades de verificación, validación o prueba correspondientes.
- ✓ I2. En cada fase de ciclo de desarrollo se generan productos que incluyen características mínimas requeridas para su aprobación.

Cada indicador contará con un conjunto de métricas asociadas las cuales se ejecutan durante el ciclo de desarrollo del software. Se determinó también el valor esperado de cada métrica, el cual representa el requerimiento de calidad de la organización para un ciclo académico específico. Se muestran a continuación las métricas definidas para el proceso:

I1 En cada fase de un ciclo de desarrollo se efectúan todas las actividades de verificación, validación o prueba correspondientes.						
Métrica	Medición y Fórmula	Fuente de Verificación	Valor esperado por Ciclo Académico			
			2008 -2	2009 - 1	2009 -2	2010 - 1
M1.1 Verificaciones realizadas durante cada fase del ciclo de desarrollo de software	X = A / B A: Cantidad de informes de verificación generados por fase B: cantidad de informes de verificación requeridos por cada fase.	Verificaciones y Validaciones requeridas del proceso Base de Conocimiento: Productos Verificados Reportes de Verificación – RVE realizados	> = 0.8	> = 0.9	1	1

M1.2 Validaciones realizadas durante cada fase del ciclo de desarrollo de software	X = A / B A: Cantidad de informes de validación generados por fase B: cantidad de informes de validación requeridos por cada fase.	Verificaciones y Validaciones requeridas del proceso Base de Conocimiento: Productos Verificados Reportes de Validación -RVA realizados	> = 0.8	> = 0.9	1	1
M1.3 Pruebas de Sistema realizadas durante el ciclo de desarrollo de software	X = A / B A: Cantidad de casos de prueba ejecutados B: cantidad de casos de prueba planificados	Reportes de Pruebas de Sistema - RPRS Plan de Pruebas de Sistema - PPRS: Casos de Prueba	> = 0.8	> = 0.9	1	1

CUADRO 4.8 MÉTRICAS DE VERIFICACIÓN, VALIDACIÓN Y PRUEBA DEL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS - PDSA

Las métricas **M1.1** y **M1.2** se ejecutan al finalizar cada fase del ciclo de desarrollo de software. La métrica **M1.3** se ejecuta al culminar el ciclo de desarrollo de software.

I2. En cada fase de ciclo de desarrollo se generan productos que incluyen características mínimas requeridas para su aprobación.						
Métrica	Medición y Fórmula	Fuente de Verificación	Valor esperado por Ciclo Académico			
			2008 -2	2009 - 1	2009 -2	2010 - 1
M2.1 Nivel de éxito en la evaluación del Reporte de Especificaciones de Software v1 – RES v1.	X = A / B A: Cantidad de características evaluadas de manera exitosa B: cantidad de características definidas para el entregable.	Lista de chequeo del producto Reportes de Verificación – RVE y Validación RVA realizados	> = 0.75	> = 0.85	> = 0.95	> = 0.95
M2.2 Nivel de éxito en la evaluación del Reporte de Especificaciones de Software v2 – RES v2.	X = A / B A: Cantidad de características evaluadas de manera exitosa B: cantidad de características definidas para el entregable.	Lista de chequeo del producto Reportes de Verificación – RVE realizados	> = 0.75	> = 0.85	> = 0.95	> = 0.95
M2.3 Nivel de éxito	X = A / B	Lista de chequeo	> =	> = 0.85	> =	> = 0.95

en la evaluación del Reporte de Diseño de Software – RDS.	A: Cantidad de características evaluadas de manera exitosa B: cantidad de características definidas para el entregable.	del producto Reporte de Verificación – RVE realizado	0.75		0.95	
M2.4 Nivel de éxito en la evaluación del Código Fuente.	X = A / B A: Cantidad de características evaluadas de manera exitosa B: cantidad de características definidas para el entregable.	Lista de chequeo del producto Reporte de Verificación – RVE realizado	> = 0.75	> = 0.85	> = 0.95	> = 0.95
M2.5 Nivel de éxito de las Pruebas realizadas durante el ciclo de desarrollo de software	X = A / B A: Cantidad de características evaluadas de manera exitosa B: cantidad de características definidas para el entregable.	Reportes de Pruebas de Sistema - RPRS Plan de Pruebas de Sistema - PPRS: Casos de Prueba	> = 0.75	> = 0.85	> = 0.95	> = 0.95

CUADRO 4.9 MÉTRICAS DE CARACTERÍSTICAS POR PRODUCTO DEL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS - PDSA

Las métricas **M2.1**, **M2.2**, **M2.3**, **M2.4** y **M2.5** se ejecutan al culminar cada fase del ciclo de desarrollo de software.

V. Responsabilidad y Autoridad

Responsable:

- Responsable de Desarrollo de software (RD)

Autoridad

- Responsable de Gestión del Proyecto Específico (RGPE)

VI. Procesos Relacionados

- Gestión de Proyectos Específicos de alumnos
- Gestión de Procesos

VII. Entradas del Proceso

1. Reporte de Solicitud de Requerimientos	
Fuente: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** RSRQ

Este documento se compone de los siguientes elementos:

1. Datos Generales
   - 2.1 Descripción del Requerimiento
   - 2.2 Área de negocio
   - 2.3 Tipo de Requerimiento
2. Beneficios
   - 2.4 Descripción de la problemática actual.
   - 2.5 Justificación del Requerimiento.
   - 2.6 Beneficios
   - 2.7 Como va a funcionar la solución.
3. Alcance
   - 3.1 Funcionalidad Requerida Detallada
   - 3.2 Restricciones
   - 3.3 Supuestos

2. Plan de Proyecto	
Fuente: Gestión de Proyectos Específicos de alumnos	Abreviatura: PP

Este documento se compone de los siguientes elementos:

4 Definición

1.1 Objetivos del Proyecto. Se enuncian los objetivos por los que se lleva a cabo el proyecto.

1.2 Alcance. Describe lo que se quiere lograr en el proyecto y sus limitaciones. Por ejemplo, el alcance puede identificar las unidades de negocio y las áreas funcionales que se incluirán (o excluirán), o los sistemas que se afectarán (o no serán afectados).

- Dentro del Alcance
- Fuera del Alcance
- Restricciones

1.3 Factores Críticos de éxito. Enunciado de aquellos factores que influyen sobre la marcha del proyecto. Son factores cuyos efectos podrían atentar contra la calidad, plazos o costos. Por lo general expresan los compromisos o requerimientos necesarios para el éxito del proyecto.

2. Procesos Identificados en el Proyecto
3. Estructura General del Proyecto
4. Cronograma
5. Organización del Proyecto
6. Análisis de Beneficios
7. Estimaciones
8. Administración de Riesgos Potenciales

VIII. Salidas del Proceso

1. Reporte de Especificaciones Software v1	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** RES v1

Este documento en su primera versión se compone de los siguientes elementos:

1. Antecedentes. Describe la situación actual y las necesidades o problemas que se pretenden atender.

2. Objetivos. Es la explicación resumida, pero clara, de lo que se pretende con el requerimiento, es decir la visión de éste.

3. Alcance. En caso que no varié el alcance definido en el Plan de Proyecto no se detalla en el presente documento.

4. Fuera de Alcance. En caso que no varié lo especificado como alcance en el Plan de Proyecto no se detalla en el presente documento.

5. Procesos de Negocio

5.1 Descripción de los procesos de Negocio. Describe los procesos de negocio a los cuales se les dará soporte con la solución de software. De ser necesario se incorpora un diagrama de casos de uso de negocio.

5.2 Reglas de Negocio. Identifica las reglas que regulan la estructura de los negocios y cómo ellas operan afectándolo. Describe los procesos de negocio que se considerarán para el diseño del sistema

6. Requisitos Funcionales. De acuerdo a lo solicitado por el área usuaria, lista todos los requisitos funcionales del producto software. Los requisitos funcionales que listados deben estar asociados a los casos de uso.

7. Requisitos No funcionales. Lista los requisitos no funcionales, los mismos que serán considerados para el modelo de calidad de producto.

8. Modelo de Casos de Uso de Sistema. En esta sección muestra el modelo de sistema o modelo de requisitos. Para ello se identifican los actores de sistemas, la arquitectura del sistema (organizada en paquetes) y la relación de casos de uso por cada paquete.

8.1. Lista de Casos de Uso del Sistema por Paquete.
8.2. Diagrama de Casos de Uso por Paquete.
8.3. Priorización de los Casos de Uso del Sistema.
8.4. Realización de los Casos de Uso del Sistema.

8.4.1. Especificación de Alto Nivel
En esta sección se incluye la especificación de alto nivel de los casos de uso del sistema.

8.4.2 Especificación Expandida de Casos de Uso y Prototipos de Interfaz Gráfica. Por cada caso de uso de sistema especificado a un alto nivel se deberá incluir la especificación expandida de casos de uso. Para ello deberá indicar el flujo básico y los flujos alternos e incorporará el prototipo con la inclusión de los controles.

2. Reporte de Especificaciones Software v2	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** RES v2

9. Navegación

9.1. Flujo General de Navegación. Incluye el árbol de navegación que permite entender el flujo que se seguirá en la navegación por el aplicativo.

10. Esquema de Seguridad. En esta se documenta los esquemas de seguridad en base a perfiles y su acceso a su información.

10.1. Matriz de perfiles de usuario y accesos por Aplicativo / Módulo / Función

3. Reporte de Diseño de Software	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** RDS

Este documento contiene la descripción textual y gráfica de la estructura de los componentes de software. Esta estructura consta de las siguientes partes:

Arquitectónica. Contiene la estructura interna del sistema, es decir la descomposición del sistema en subsistemas. Así como la identificación de los componentes que integran los subsistemas y las relaciones de interacción entre ellos.

Detallada. Contiene el detalle de los componentes que permitan de manera evidente su construcción y prueba en el ambiente de programación.

4. Componente	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** -

Conjunto de unidades de código relacionados.

5. Software	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** -

Sistema de software, destinado a un cliente o usuario, constituido por componentes agrupados en subsistemas, posiblemente anidados.

6. Configuración de Software	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** -

Conjunto consistente de productos de software, que incluye:

- Reportes de Especificaciones de Software: RES v1 y RES v2
- Reporte de Diseño de Software - RDS

- Software
- Matriz de trazabilidad
- Plan de pruebas de sistema
- Reporte de pruebas de sistema
- Manual de usuario

7. Manual de Usuario	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** - MUSU

Documento electrónico o impreso que describe la forma de uso del software con base a la interfaz del usuario. Éste deberá ser redactado en términos compresibles a los usuarios.

8. Reporte de Actividades	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** RACT

Registro periódico de actividades, fechas de inicio y fin, responsables y mediciones, tales como:

- Tiempo de producción, de corrección, de verificación y de validación.
- Defectos encontrados en verificación, validación o prueba
- Tamaño de productos

9. Lecciones Aprendidas	
Destino: Gestión de Procesos	**Abreviatura:** - LAPR

Registro de mejores prácticas, problemas recurrentes y experiencias exitosas en la solución de problemas, encontrados en un ciclo de desarrollo.

10. Reporte de Mediciones y Sugerencias de Mejora	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** - RMSM

Registro que contiene:
Mediciones de los indicadores del proceso de desarrollo y mantenimiento de software por alumnos.

Sugerencias de Mejora al proceso de Desarrollo de Software por Alumnos – PDSA (métodos, herramientas, formatos, estándares, etc.).

11. Matriz de Trazabilidad	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** - MTRZ

Relación entre los requerimientos, elementos de análisis y diseño, componentes y planes de pruebas

12. Plan de Pruebas de Sistema	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** - PPRS

Identificación de pruebas requeridas para el cumplimiento de los requerimientos especificados.

13. Reporte de Pruebas de Sistema	
Destino: Gestión de Proyectos Específicos de alumnos	**Abreviatura:** - RPRS
Registro de participantes, fecha, lugar, duración y de defectos encontrados.	

IX. Productos Internos

1. Reporte(s) de Verificación	
Uso: Durante el ciclo de desarrollo del Software	**Abreviatura:** RVE
Registro de participantes, fecha, lugar, duración y de defectos encontrados	
2. Reporte(s) de Validación	
Uso: Durante el ciclo de desarrollo del Software	**Abreviatura:** RVA
Registro de participantes, fecha, lugar, duración y de defectos encontrados	

4.4.1.2 PRÁCTICAS

I. Roles involucrados y capacitación

Los alumnos, integrantes de un equipo de proyecto y principales ejecutores del proceso, asumirán diferentes roles dependiendo de la etapa en la que se encuentren dentro del ciclo de vida de desarrollo del software.

Son partícipes también del proceso:

- Docentes. A cargo de los cursos relacionados con el PDSA.
- Comité de Proyectos. Integrado por los docentes coordinadores de los cursos relacionados con el PDSA.
- Líder Usuario. Representante de la organización cliente ante CIBERTEC y la organización proveedora, es decir, el equipo de proyecto integrado por alumnos.
- Usuario. Utiliza el producto software elaborado por la organización proveedora.

Se describen en el siguiente cuadro los roles involucrados en el PDSA:

Rol	Abreviatura	Capacitación
Responsable de Gestión del proyecto específico	RGPE	Capacidad de liderazgo y de toma de decisiones, trabajo en equipo y desarrollo de software
Responsable de Desarrollo software	RD	Conocimiento en el desarrollo de software.
Analista	AN	Conocimiento en la obtención, especificación y análisis de requerimientos.
Diseñador de Interfaz de usuario	DU	Conocimiento en diseño de interfaces de usuario.
Diseñador	DI	Conocimiento en el diseño de la estructura de los componentes de software.
Programador	PR	Conocimiento en la programación, integración y pruebas unitarias
Responsable de pruebas	RPU	Conocimiento en la planificación y realización de pruebas de integración y de sistema.
Revisor	RE	Conocimiento en las técnicas de redacción y en el desarrollo y mantenimiento de software.
Responsable de manuales	RM	Conocimiento en las técnicas de redacción y en el desarrollo y mantenimiento de software
Equipo de trabajo	ET	Conocimiento de acuerdo a su rol.
Líder Usuario	LU	Interpretación del estándar de la especificación de requerimientos
Usuario	US	Ninguna
Comité de Proyecto	CP	Conocimiento y experiencia de acuerdo a su rol
Docente	DCE	Conocimiento y experiencia de acuerdo a su rol

CUADRO 4.10 ROLES PARTICIPANTES EN EL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS – PDSA

II. Actividades

El proceso de desarrollo de software por alumnos, involucra la ejecución secuencial de las siguientes fases correspondientes al ciclo de vida de desarrollo de software:

A1. Fase de Inicio
A2. Fase de Identificación de Requerimientos
A3. Fase de Análisis de Requisitos
A4. Fase de Definición de Arquitectura y Diseño detallado de la solución de software
A5. Fase de Construcción
A6. Fase de Pruebas
A7. Fase de Cierre

La ejecución de todas las fases representa una iteración dentro del ciclo de vida de desarrollo, pudiéndose realizar varias iteraciones para obtener el producto software deseado.

La duración de una iteración (realización de todas las fases) corresponderá a la duración de un ciclo académico en CIBERTEC: una iteración se inicia al comenzar el ciclo académico y culmina al terminar el ciclo académico.

Se detallan a continuación las actividades por cada fase del ciclo de desarrollo identificando los roles involucrados, entradas y salidas por cada fase, así como los controles de calidad respectivos.

Rol	**Descripción de la Actividad**
A1. Realización de la fase de inicio	
ET	A.1.1 Revisar con los miembros del equipo de trabajo el Plan de Proyecto - PP para lograr un entendimiento común del proyecto.
RD	A.1.2 Elaborar el Reporte de Actividades – RACT, registrando las actividades realizadas, fechas de inicio y fin, responsable por actividad y mediciones requeridas. En base a este documento el docente del curso evalúa el avance de cada uno de los miembros del equipo de trabajo.
A2. Realización de la fase de Identificación de Requerimientos	
RD AN	A.2.1 Distribuir tareas a los miembros del equipo de trabajo según su rol, de acuerdo al Plan de Proyecto - PP actual.
AN CL US DU	A2.2. Documentar o modificar el Reporte de Especificaciones de Software – RES v1: • Identificar y consultar fuentes de información (Exposiciones del líder usuario, entrevistas a usuarios, cuestionarios, sistemas previos, documentos, etc.) para obtener nuevos requerimientos. • Analizar los requerimientos identificados para delimitar el alcance y su factibilidad, considerando los supuestos y restricciones del proyecto identificados en el Plan de Proyecto - PP. • Generar o actualizar el Reporte de Especificaciones de Software v1 - RES v1, en base al Reporte de Solicitud de Requerimientos –RSRQ elaborado por el líder usuario y al Plan de Proyecto - PP actuales. Este documento en su primera versión incluye: o Reglas de Negocio: que regulan el comportamiento de los procesos de negocio considerados para el diseño del sistema. o Requisitos funcionales del producto software. o Requisitos no funcionales que deben ser considerados para el modelo de calidad del producto. o Modelo de casos de Uso del Sistema: ▪ Lista de Casos de Uso del Sistema por paquete ▪ Diagrama de Casos de Uso por Paquete ▪ Priorización de Casos de Uso de Sistema ▪ Realización de Casos de Uso de Sistema – Especificación de Alto Nivel ▪ Realización de Casos de Uso de Sistema – Especificación Expandida de Casos de Uso

	▪ Prototipos de interfaz gráfica por cada caso de uso.
RE DCE	A.2.3 Verificar el Reporte de Especificaciones de Software v1 – RES v1 (**Ver1**) : • El alumno revisor comprueba el Reporte de Especificaciones de Software v1 - RES v1 utilizando las técnicas de verificación y la lista de chequeo definidas para esta actividad. Se debe generar el Reporte de verificación respectivo – RVE : o El alumno revisor verifica el Reporte de Especificaciones de Software - RES v1 contra el Reporte de Solicitud de Requerimientos - RSRQ y el Plan de Proyecto - PP actual. o El Reporte de Verificación - RVE se redacta de acuerdo al formato definido para esta actividad. • Verifican los docentes de los cursos relacionados con el proyecto. Los docentes generan el Reporte de verificación – RVE o Con el reporte de verificación – RVE del alumno revisor, el docente verifica el Reporte de Especificaciones de Software - RES v1. Se complementa el Reporte de Verificación – RVE del alumno revisor generándose una versión definitiva en base a la apreciación del docente.
AN DU	A.2.4 Corregir defectos encontrados en el Reporte de Especificaciones de Software - RES v1, en base al reporte de verificación - RVE, y obtener la aprobación de las correcciones por el docente del curso.
DCE CP	A.2.5 Los docentes elijen el mejor Reporte de Especificaciones de Software - RES v1 (el docente elige sólo un Reporte de Especificaciones de Software - RES v1 de todas las secciones a su cargo). El documento seleccionado se envía al comité de proyectos.
LU US RPU CP	A.2.6 Validar el Reporte de Especificaciones de Software - RES v1 (**Val1**): • Valida el líder usuario: o El comité de proyectos envía los Reportes de Especificaciones de Software – RES v1 al líder usuario. o El líder usuario recibe los Reportes de Especificaciones de Software v1 – RES v1 y los valida usando la lista de chequeo definida para esta actividad. o El líder usuario retorna al comité de proyectos los Reportes de Especificaciones de Software - RES v1 validados. o El líder usuario visita cada una de las secciones involucradas en el proyecto y junto a los equipos de alumnos revisa los prototipos del sistema.
AN DU	A.2.7 Corregir los defectos encontrados por el usuario en el Reporte de Especificaciones de Software v1 - RES v1.
AN RPU	A.2.8 Elaborar o modificar Plan de Pruebas del sistema – PPRS. En base al Reporte de Especificaciones de Software v1 - RES v1. Se deberá desarrollar el Plan de Pruebas del sistema – PPRS.
RE	A.2.9Verificar el plan de pruebas de sistema – PPRS (**Ver2**)
RPU	A.2.10 Corregir los defectos encontrados en el Plan de pruebas de sistema – PPRS, con base en el Reporte de verificación - RVE y obtener la aprobación de las correcciones.
RD	A.2.11 Incorporar el Reporte de Especificaciones de Software - RES v1 y el Plan de Pruebas del Sistema - PPRS como de líneas base a la configuración de Software. ▪ Generar o actualizar el Reporte de línea base del Proyecto – RLB v1.

RD	A.2.12 Elaborar el Reporte de Actividades – RACT registrando las actividades realizadas, fechas de inicio y fin, responsable por actividad y mediciones requeridas.
A3. Realización de la fase de Análisis de Requisitos	
RD AN DI	A.3.1 Distribuir tareas a los miembros del equipo de trabajo según su rol, de acuerdo al Plan de Proyecto - PP actual.
AN DU	A.3.2 Documentar o modificar el Reporte de Especificaciones de Software v2 – RES v2. • Elaborar o modificar las especificaciones de Casos de Uso, las cuales incluyen las realizaciones (Diagramas de Secuencia con el patrón MVC) • Generar o actualizar el Reporte de Especificaciones de Software - RES v2 en base al Reporte de Especificaciones de Software - RES v1
RE DCE	A.3.3 Verificar el Reporte de Especificaciones de Software v2 - RES v2(**Ver3**) • El alumno revisor comprueba el Reporte de Especificaciones de Software - RES v2 utilizando las técnicas de verificación y la lista de chequeo definidas para esta actividad. Se debe generar el Reporte de verificación respectivo – RVE : o El alumno revisor verifica el Reporte de Especificaciones de Software - RES v2 contra el Reporte de Especificaciones de Software - RES v1. • Verifican los docentes de los cursos relacionados con el proyecto. Los docentes generan el Reporte de verificación - RVE. o Con el reporte de verificación - RVE del alumno revisor, el docente verifica el Reporte de Especificaciones de Software - RES v2 contra el Reporte de Especificaciones de Software - RES v1. El docente verifica diagramas de secuencia de Análisis: realizaciones de casos de uso implementadas utilizando patrones de diseño. Por ejemplo, el patrón MODEL VIEW CONTROLLER (MVC). o Se complementa el Reporte de Verificación – RVE del alumno revisor con el Reporte de verificación - RVE del docente.
AN DU DI	A.3.4 Corregir defectos encontrados en el Reporte de Especificaciones de Software - RES v2 con base en el reporte de verificación - RVE y obtener la aprobación de las correcciones por los docentes.
RD	A.3.5 Incorporar el Reporte de Especificaciones de Software - RES v2 como de línea base a la configuración de Software. • Generar o actualizar el Reporte de línea base del Proyecto – RLB v2.
RD	A.3.6 Elaborar el Reporte de Actividades – RACT, registrando las actividades realizadas, fechas de inicio y fin, responsable por actividad y de mediciones requeridas.
A4. Realización de la fase de Definición de Arquitectura y Diseño detallado de la solución de software	
RD AN DI	A.4.1 Distribuir tareas a los miembros del equipo de trabajo según su rol, de acuerdo al Plan de Proyecto - PP actual.
AN DI	A.4.2 Documentar o modificar el Reporte de Diseño de Software - RDS.

DU	• Analizar el Reporte de Especificaciones de Software v2 - RES v2 para generar la descripción de la estructura interna del sistema y su descomposición en subsistemas, y estos a su vez en componentes, definiendo las interfaces entre ellos. • Describir el detalle de la apariencia y el comportamiento de la interfaz con base en la especificación de requerimientos de forma que se puedan proveer los recursos para su implementación. • Describir el detalle de los componentes que permita su construcción de manera evidente. • Generar o actualizar el Reporte de Diseño de Software - RDS. Este documento contiene los siguientes elementos: o Arquitectura de Software (Base de Datos, Servidores). o El modelo Lógico y Físico de la base de datos. o Diseño detallado de Software. Implica las realizaciones de casos de uso de Diseño incluyendo los patrones de Arquitectura utilizados. Por ejemplo, se podrán utilizar los patrones DATA ACCESS OBJECT (DAO), VIEW HELPER, FRONT CONTROLLER y frameworks tales como: STRUTS, IBATIS y/o SPRING. • Generar o modificar la matriz de trazabilidad - MTRZ.
RE DCE	A.4.3 Verificar el Reporte de Diseño de Software - RDS y Matriz de Trazabilidad - MTRZ (**Ver4**) • El alumno revisor comprueba el Reporte de Diseño de Software - RDS y el Registro de Rastreo – RRAS utilizando las técnicas de verificación y la lista de chequeo definidas para esta actividad. Se debe generar el Reporte de verificación respectivo – RVE : o El alumno revisor verifica el Reporte de Diseño de Software - RDS contra el Reporte de Especificaciones de Software v2 - RES v2. • Verifican los docentes de los cursos relacionados con el proyecto. Los docentes generan el Reporte de verificación – RVE. o Con el Reporte de Verificación - RVE del alumno revisor, el docente verifica el Reporte de Diseño de Software - RDS contra el Reporte de Especificaciones del Software v2 - RES v2 . El docente verifica: -Arquitectura de software. Debe ser la definida en el curso, es decir, se verifica el uso FRAMEWORKS, Patrones de Diseño y diversas tecnologías involucradas. -Realizaciones de Casos de Uso de Diseño (diagramas de Secuencia): deben incorporar los patrones de diseño enseñados en los cursos de diseño y programación. o Se complementa el Reporte de Verificación – RVE del alumno revisor con el Reporte de verificación – RVE del docente.
AN DI DU	A.4.4 Corregir los defectos encontrados en el Reporte de Diseño de Software - RDS y en la matriz de Trazabilidad – MTRZ con base en el Reporte de Verificación – RVE y obtener la aprobación de correcciones por parte del docente.
RD	A.4.5 Incorporar el Reporte de Diseño de Software – RDS y la matriz de Trazabilidad - MTRZ como líneas base a la configuración de software.

	• Generar o actualizar el Reporte de línea base del Proyecto – RLB v3.
RD	A.4.6 Elaborar el Reporte de Actividades – RACT, registrando las actividades realizadas, fechas de inicio y fin, responsable por actividad y de mediciones requeridas.
A5. Realización de la fase de Construcción	
RD	A.5.1 Distribuir tareas a los miembros del equipo de trabajo según su rol, de acuerdo al Plan de Proyecto – PP actual.
PR	A.5.2 Construir o modificar el (los) Componente(s) de software: • Implementar o modificar componente(s) con base en el Reporte de Diseño de Software - RDS. • Verificar el componente software (**Ver5**): o El alumno revisor comprueba el componente software utilizando las técnicas de verificación y la lista de chequeo definidas para esta actividad. Se debe generar el Reporte de verificación respectivo – RVE o Corregir defectos encontrados en el componente software con base en el reporte de verificación – RVE • Aplicar pruebas unitarias para verificar que el funcionamiento de cada componente esté acorde con la parte detallada del Reporte de Diseño de Software - RDS. • Corregir los defectos encontrados hasta logras pruebas unitarias exitosas (sin defectos) • Actualizar la matriz de Trazabilidad - MTRZ, incorporando los componentes construidos o modificados.
RE DCE	A.5.3 Verificar la matriz de trazabilidad - MTRZ (**Ver6**). • El alumno revisor verifica la matriz de trazabilidad - MTRZ y genera el Reporte de Verificación – RVE
PR	A.5.4 Corregir los defectos encontrados en la matriz de trazabilidad – MTRZ con base en el Reporte de Verificación – RVE y obtener la aprobación de las correcciones.
RD	A5.5 Incorporar Componentes y matriz de trazabilidad - MTRZ como líneas base a la Configuración de Software. • Generar o actualizar el Reporte de línea base del Proyecto – RLB v4.
RD	A5.6 Elaborar el Reporte de Actividades – RACT, registrando las actividades realizadas, fechas de inicio y fin, responsable por actividad y mediciones requeridas.
A6. Realización de la fase de Pruebas	
RD	A6.1 Distribuir tareas a los miembros del equipo de trabajo según su rol, de acuerdo al Plan de Proyecto – PP actual.
PR RPU	A6.2 Realizar integración • Integrar los componentes en subsistemas o en el sistema de software.
RPU	A6.3 Realizar las pruebas de sistema siguiendo el Plan de Pruebas de Sistema - PPRS, documentando los resultados en un reporte de Pruebas de Sistema – RPRS.
PR	A6.4 Corregir los defectos encontrados en las pruebas de sistema con base en el Reporte de Pruebas de Sistema – RPRS, y obtener la aprobación de las correcciones.
RM	A6.5 Documentar el Manual de Usuario – MUSU o modificar el existente.
RE	A6.6 Verificar el Manual de Usuario - MUSU (**Ver7**). • El alumno revisor verifica el Manual de Usuario – MUSU y genera el Reporte de Verificación – RVE

RM	A6.7 Corregir los defectos encontrados en el Manual de Usuario - MUSU con base en el Reporte de verificación – RVE y obtener la aprobación de correcciones.
RD	A6.8 Incorporar Software y Manual de Usuario – MUSU como líneas base a la configuración de Software. • Generar o actualizar el Reporte de línea base del Proyecto – RLB v5
RD	A6.9 Elaborar el Reporte de Actividades- RACT registrando las actividades realizadas, fechas de inicio y fin, responsable por actividad y mediciones requeridas.
A7. Realización de la fase de Cierre	
RD ET	A7.1 Identificar las lecciones aprendidas - LAPR e integrarlas a la base de Conocimiento. Como ejemplo, se pueden considerar mejores prácticas, experiencias exitosas de manejo de riesgos, problemas recurrentes, entre otras.
RD ET	A7.2 Generar el Reporte de Mediciones y Sugerencias de Mejora – RMSM.
RD	A7.3 Elaborar el Reporte de Actividades – RACT, registrando las actividades realizadas, fechas de inicio y fin, responsable por actividad y mediciones requeridas.

III. Verificaciones y Validaciones

Son realizadas durante todo el ciclo de desarrollo para garantizar que los productos generados en cada fase satisfacen los requerimientos especificados por la organización cliente.

A continuación se muestra un cuadro resumen con las verificaciones y validaciones realizadas durante el Proceso de Desarrollo de Software por Alumnos – PDSA:

Verificación o validación	Actividad	Producto	Rol	Descripción
Ver1	A2.3	Reporte de Especificaciones de Software v1 – RES v1	RE	Verificar la claridad de redacción del Reporte de Especificaciones de Software v1 – RES v1 y su consistencia con la descripción del Producto software. Adicionalmente revisar que los requerimientos sean completos y no ambiguos o contradictorios. Los defectos encontrados se documentan en un Reporte de Verificación - RVE.
Val1	A2.6	Reporte de Especificaciones de Software v1 – RES v1	LU, US, RPU	Validar que el Reporte de Especificaciones de Software v1 - RES v1 cumple con las necesidades y expectativas acordadas, incluyendo los prototipos de interfaz de usuario. Los defectos encontrados se documentan en un Reporte de Validación - RVA.
Ver2	A2.9	Plan de Pruebas de Sistema –	RE	Verificar la consistencia del Plan de Pruebas de Sistema - PPRI con el

		PPRS		Reporte de Especificaciones de Software v1 – RES v1. Los defectos encontrados se documentan en un Reporte de Verificación – RVE.
Ver3	A3.3	Reporte de Especificaciones de Software v2 – RES v2	RE	Verificar claridad de la documentación del Reporte de Especificaciones de Software v2 – RES v2 (Análisis de Requisitos, su factibilidad y la consistencia) con la especificación de requerimientos, es decir, el Reporte de Especificaciones de Software v1 – RES v1. Los defectos encontrados se documentan en un reporte de Verificación - RVE.
Ver4	A4.3	Reporte de Diseño de Software - RDS y Matriz de Trazabilidad - MTRZ	RE	Verificar claridad de la documentación del Reporte de Diseño de Software – RDS, su factibilidad y la consistencia con el Reporte de Especificaciones de Software v2 - RES v2. Verificar que el Registro Rastreo - RRAS contenga las relaciones adecuadas entre los elementos de Análisis y los elementos de Diseño. Los defectos encontrados se documentan en un reporte de Verificación – RVE.
Ver5	A5.2	Componente Software	RE	Verificar que el componente software cumpla con los estándares de codificación y criterios definidos en la lista de chequeo respectiva. Los defectos encontrados se documentan en un reporte de verificación – RVE.
Ver6	A5.3	Matriz de Trazabilidad - MTRZ	RE	Verificar que la Matriz de Trazabilidad – MTRZ contenga las relaciones adecuadas entre los elementos de análisis y Diseño y los componentes. Los defectos encontrados se documentan en un reporte de verificación – RVE.
Ver7	A6.6	Manual de Usuario – MUSU	RE	Verificar consistencia del Manual de Usuario – MUSU con el Reporte de Especificaciones de Software v1 - RES v1 y con el el sistema de software. Los defectos encontrados se documentan en un reporte de verificación – RVE.

CUADRO 4.11 VERIFICACIONES Y VALIDACIONES DEL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS - PDSA

IV. Incorporación a la base de conocimiento

Se incorporan en este repositorio todos los productos generados durante el Proceso de Desarrollo de Software por Alumnos – PDSA.

El cuadro 4.11 resume todos los productos incorporados a la base de conocimiento:

Producto	Forma de aprobación
Reporte de Especificaciones de Software v1 – RES v1	Ver1, Val1
Plan de Pruebas de Sistema – PPRS	Ver2
Reporte de Especificaciones de Software v2 – RES v2	Ver3
Reporte de Diseño de Software – RDS y Matriz de Trazabilidad – MTRZ	Ver4
Componente(s)	Ver5 y Prueba unitaria exitosa
Matriz de Trazabilidad – MTRZ	Ver6
Software	Prueba de integración exitosa, prueba de sistema exitosa
Manual de Usuario – MUSU	Ver7
Reporte de Pruebas de sistema – RPRS	Ninguna
Reporte(s) de Actividades – RACT	Ninguna
Lecciones aprendidas – LAPR	Ninguna
Reporte(s) de Verificación – RVE	Ninguna
Reporte(s) de validación – RVA	Ninguna

CUADRO 4.12 PRODUCTOS GENERADOS EN EL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS - PDSA

V. Recursos de Infraestructura

Las actividades de cada una de las fases del ciclo de desarrollo de software pueden ser ejecutadas utilizando recursos de hardware y/o software que faciliten su ejecución o la hagan más eficiente.

Se muestran a continuación, los recursos disponibles por actividad, en la primera versión de este proceso:

Actividad	Recurso
A1, A2, A3, A4, A5, A6, A7	Herramienta para documentación: MS OFFICE
A2	Herramienta para la captura de Requisitos: RATIONAL ROSE
A3	Herramientas para el Análisis de Requisitos:
A4	Herramientas para el Diseño de la solución: RATIONAL ROSE
A5	Herramientas para la construcción: IDE de desarrollo: Eclipse Lenguaje de Programación: Java (Java Enterprise Edition – JEE) Servidor de Base de Datos: MySQL Servidor de Aplicaciones: TOMCAT
A5,A6	Herramientas para la realización de pruebas: PRUEBAS MANUALES

CUADRO 4.13 RECURSOS DISPONIBLES POR ACTIVIDAD PARA EL PROCESO DE DESARROLLO DE SOFTWARE POR ALUMNOS - PDSA

VI. Situaciones Excepcionales

Los roles participantes del Proceso de Desarrollo de Software por Alumnos – PDSA, deberán notificar al rol Responsable de Desarrollo de Software (RDS), de manera oportuna, las situaciones que les impidan el desarrollo de las actividades asignadas.

VII. Lecciones Aprendidas

Antes de iniciar las actividades asignadas, los roles participantes en el Proceso de Desarrollo de Software por Alumnos – PDSA deberán consultar las Lecciones Aprendidas - LAPR de la Base de conocimiento para aprovechar la experiencia de la organización y disminuir la posibilidad de incurrir en problemas recurrentes.

4.5 RESUMEN DEL CAPÍTULO

En este capítulo se detalla en primer lugar, el proceso de planeamiento estratégico realizado para la carrera de computación e informática de CIBERTEC. En este proceso se evidencia la relación entre la misión, visión y objetivos de la carrera con la metodología de desarrollo de software a implementar: MEDESOFT. MEDESOFT se constituye en una de las principales líneas de acción que contribuyen a alcanzar los resultados de aprendizaje de la carrera RAC.1 y RAC2: estos resultados se encuentran relacionados con el conocimiento y participación de los alumnos en actividades de desarrollo y calidad de software.

Posteriormente se describe el plan táctico asociado al planeamiento estratégico elaborado. Este plan está dividido en fases y permitirá la implementación exitosa de la metodología MEDESOFT, la cual en su fase final contará con procedimientos de desarrollo, gestión y mejora continua de procesos.

Como parte de la primera fase del plan táctico, se definió el proceso de desarrollo de software por alumnos – PDSA, el cual cuenta con objetivos, indicadores y métricas de modo que pueda ser evaluado y ser parte de un proceso de mejora continua. Es el objetivo de esta primera fase, implementar un proyecto piloto durante el ciclo académico 2008-2. El objetivo ha sido alcanzado y es detallado en el siguiente capítulo.

Capítulo 5
Implementación de proyecto Piloto

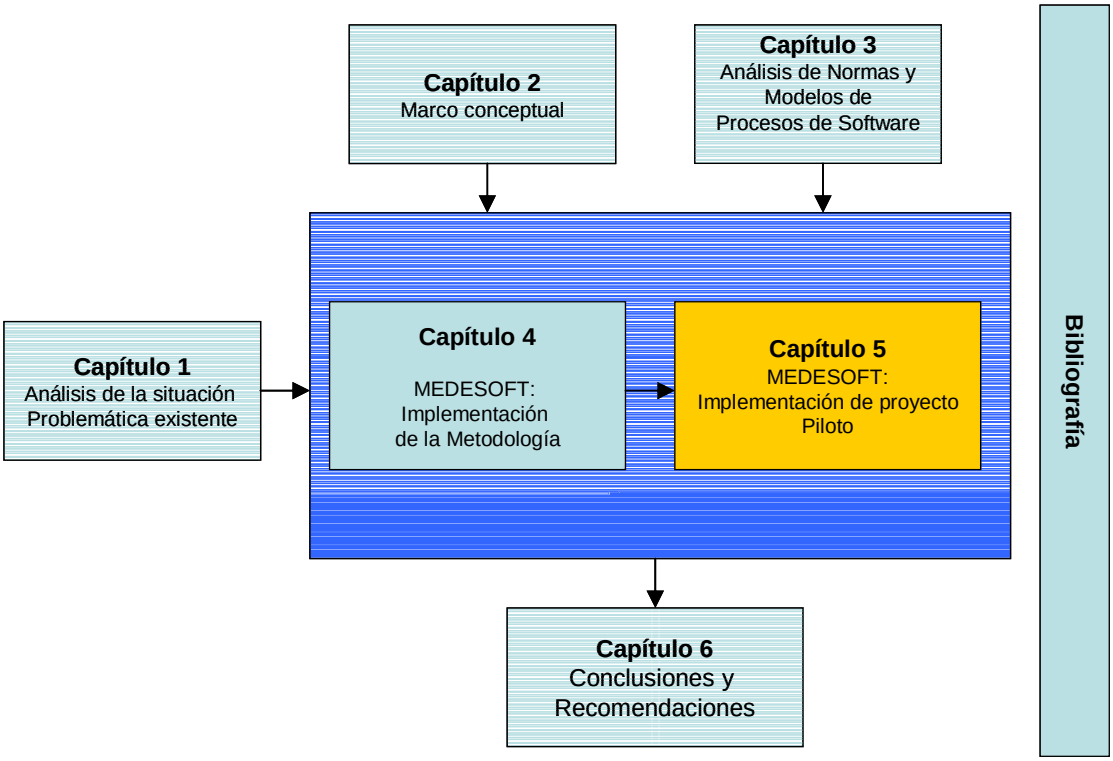


FIGURA 5.1 UBICACIÓN EN LA LECTURA DEL DOCUMENTO

En el presente capítulo se describe el proceso de implementación del primer proyecto piloto utilizando la metodología académica de desarrollo de software MEDESOFT. El proyecto se implementó durante el ciclo académico 2008-2, con los alumnos del último ciclo de la carrera de Computación e Informática.

Se denominó a este proyecto: Sistema de Votación Electrónica no Presencial, el cual está basado en un proyecto real (actualmente en desarrollo) perteneciente a la Oficina Nacional de Procesos Electorales: ONPE.

El alcance y documentación inicial de este proyecto fueron validados por el Jefe del Área de Proyectos Informáticos de la ONPE, quien asumió el rol de líder usuario y se desempeña actualmente como coordinador del curso de Calidad de Software de la carrera de Computación e Informática.

En la figura 5.2 se muestra la secuencia de actividades realizadas para implementar el proyecto piloto:

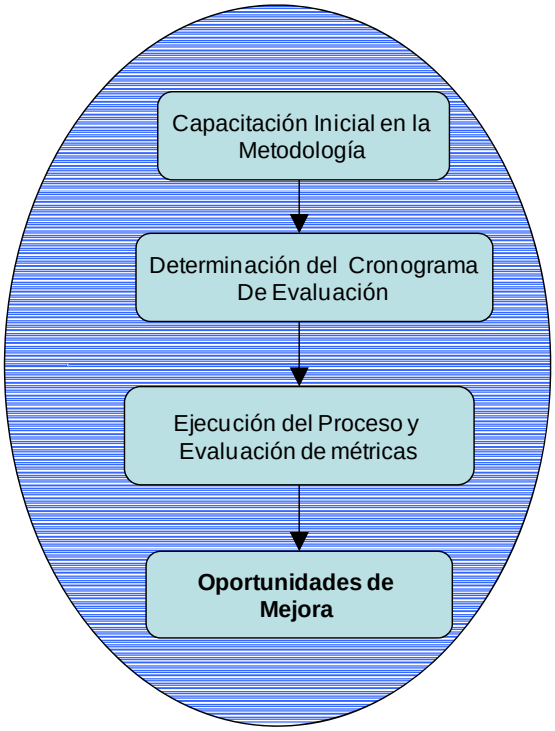


FIGURA 5.2 SECUENCIA DE ACTIVIDADES PARA IMPLEMENTAR EL PROYECTO PILOTO CON MEDESOFT

5.1 CAPACITACIÓN INICIAL EN LA METODOLOGÍA MEDESOFT

Con el auspicio de la Dirección académica de la Escuela de Tecnología, se realizó un taller de capacitación sobre la metodología MEDESOFT v1.0 al equipo docente integrante de los siguientes cursos:

- **Quinto Ciclo**: Desarrollo de Aplicaciones Web 1, Diseño de Proyectos y Calidad de Software
- **Sexto Ciclo:** Desarrollo de Aplicaciones Web 2 y Certificación de Calidad

El objetivo de la capacitación se orientó a facilitar la comprensión y el desarrollo de habilidades en el equipo docente para poner en práctica el Proceso de Desarrollo de Software por Alumnos – PDSA. Cada docente debía aplicar el proceso en las secciones a su cargo.

Una vez explicado y comprendido el procedimiento PDSA, se realizaron las siguientes actividades adicionales como parte del taller de capacitación MEDESOFT:

- Selección de los productos software a generar en el primer proyecto piloto a implementar
- Capacitación en el manejo de los formatos de Verificación y/o validación de los productos generados por el proceso PDSA
- Determinación de responsabilidades de evaluación de métricas y formato de seguimiento

5.1.1 SELECCIÓN DE LOS PRODUCTOS SOFTWARE A GENERAR

Para la implementación del primer proyecto piloto, se determinó de común acuerdo con el equipo docente involucrado, generar los siguientes entregables:

- Reporte de Especificaciones de Software v1 – RES v1
- Reporte de Especificaciones de Software v2 – RES v2
- Plan de Pruebas del Sistema – PPRS
- Reporte de Diseño de Software - RDS
- Casos de Prueba del Sistema
- Componentes Software y Solución de Software Integrada

El criterio de selección de los entregables para esta primera implementación, se basó en los contenidos actuales de los sílabos de los cursos:

- Quinto ciclo: Diseño de Proyectos Calidad de Software y Desarrollo de Aplicaciones Web I. Cursos prerrequisitos de los cursos Certificación de Calidad y Desarrollo de Aplicaciones Web II

- Sexto ciclo: Certificación de Calidad y Desarrollo de Aplicaciones Web II.

Bajo el mismo criterio, se determinó también realizar las siguientes verificaciones y/o validaciones a lo largo del ciclo de desarrollo de software:

- **Ver1:** Verificación del Reporte de Especificaciones de Software – RES v1
- **Val1:** Validación del Reporte de Especificaciones de Software – RES v1
- **Ver2:** Verificación del Plan de Pruebas del Sistema - PPRS
- **Ver3:** Verificación del Reporte de Especificaciones de Software – RES v2
- **Ver4:** Verificación de Diseño de Software – RDS
- **Ver5:** Verificación del Componente Software

5.1.2 CAPACITACIÓN EN EL MANEJO DE LOS FORMATOS DE VERIFICACIÓN Y/O VALIDACIÓN

Una vez definidos los productos software a generar se seleccionó por consenso del equipo docente, la técnica de revisión de pares como técnica de verificación para el proceso PDSA.

Los profesores coordinadores de los cursos Calidad de Software y Certificación de Calidad realizaron una capacitación en la que se destacó la importancia de las actividades de verificación y validación: se detalló a través de un ejemplo práctico, el flujo de actividades que se realizan durante una revisión de pares y como se debe llenar correctamente el formato de verificación.

Para implementar la revisión de pares durante las actividades de verificación **Ver1, Ver2, Ver3, Ver4** y **Ver5** se definieron los siguientes pasos:

- Realizar una asignación al azar entre los diversos equipos de alumnos que llevan los cursos Certificación de Calidad y Desarrollo de Aplicaciones Web II, de modo que entre si se realicen las revisiones de pares de los productos involucrados.

- Luego de entregado a cada equipo el documento a revisar, se realizará una presentación de las observaciones y defectos encontrados en el mismo, en el cual participarán el equipo revisor (el grupo que hizo la revisión de pares), el equipo revisado (el grupo que elaboró el documento) y el moderador (el profesor del curso).

- Se realizará un debate entre los grupos revisores y revisados para llegar a un consenso de cuáles son las correcciones definitivas a implementar, las mismas que deberán ser presentadas en la siguiente sesión de clase.

A continuación se muestran en las figuras 5.3, 5.4 y 5.5 los resúmenes de los formatos de verificación de los productos:

- Reporte de Especificaciones de Software – RES,
- Reporte de Diseño de Software – RDS y
- Componente Software (Código Fuente).

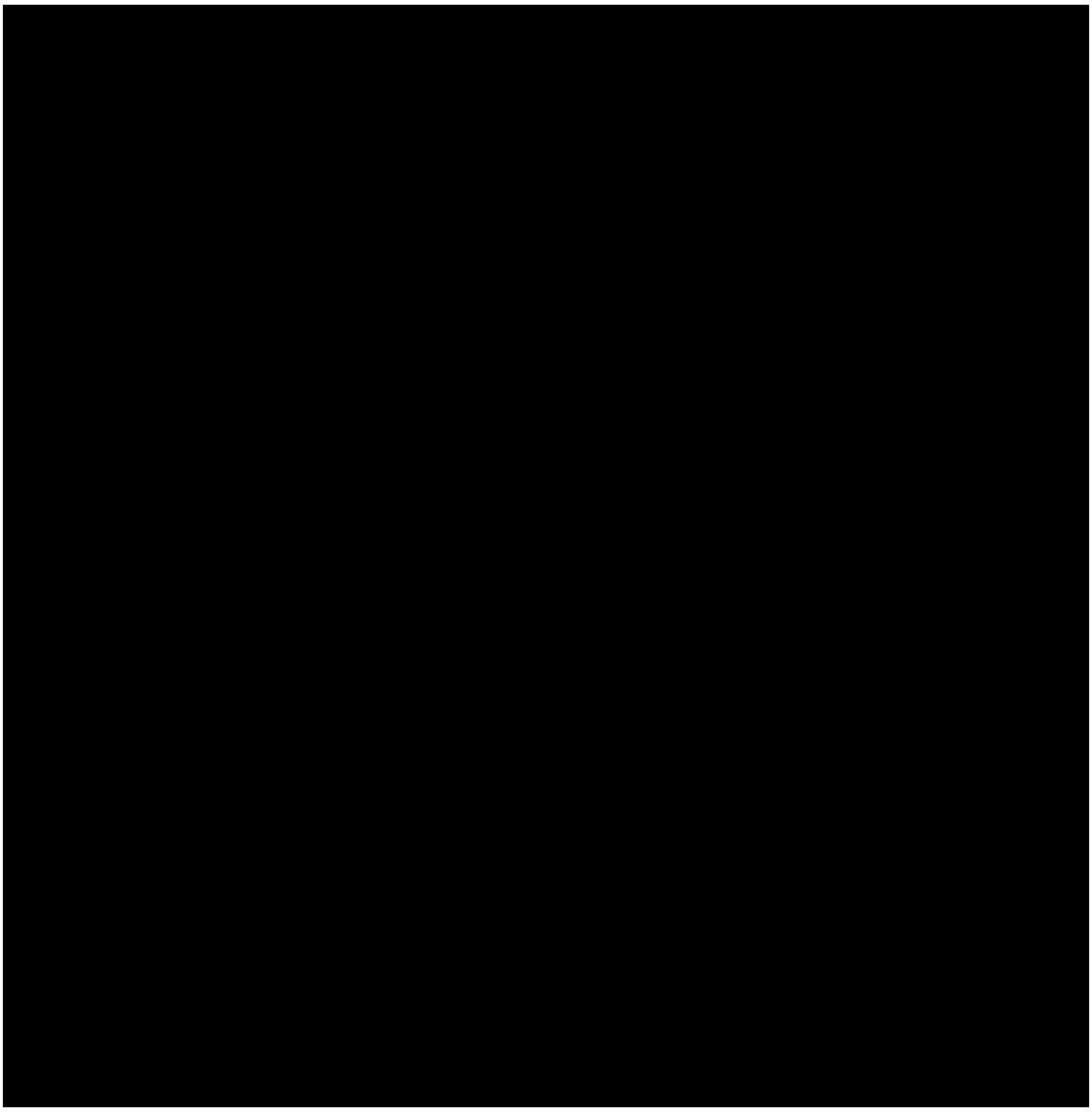


Figura 5.3 Resumen del formato de verificación del documento RES

REVISIÓN DE PARES DEL RDS - LISTA DE CHEQUEO PARA VERIFICACIÓN

I. Datos Generales

Documento

Código :		Elaborado por :	

Proyecto

Código :		Nombre :	

II. Resumen de las revisiones

N°.Revisión	Fecha de Revisión	Fecha estimada de corrección	Cantidad de defectos	Horas	Observaciones

III. Lista de Criterios de Verificación

Ítem	Preguntas iniciales	Referencia	Cumple	Revisión de reporte	Revisión de corrección	Observaciones
1	¿Se encuentra claramente identificado el Nombre del Sponsor, Nombre y Fecha de Inicio del Proyecto?	Caratula				
2	¿Se identifica los responsables de Elaboración, Revisión y Aprobación del documento?	Caratula				
3	¿El historial de versiones se encuentra correctamente llenado?	Caratula				
4	¿El indice se concuerda con la documentación brindada en el proyecto?	Indice				
5	¿Se ha descrito claramente el propósito del RDS?	1.1				
6	¿Se ha colocado el alcance que contempla este documento?	1.2				
7	¿Se han colocado las principales definiciones, acrónimos y abreviaturas utilizadas a lo largo del RDS?	1.3				
8	¿Se han identificado las referencias a otros documentos que se utilizarán desde el RDS?	1.4				
9	¿Se ha colocado la Vista General de la Arquitectura?	2				
10	¿La Vista General de la Arquitectura concuerda con lo plasmado a lo largo del RDS?	2				
11	¿Se han descrito las diversas capas que componen las arquitectura planteada?	2				
12	¿Se han identificado las posibles restricciones de la arquitectura que se tomaron en cuenta?	3				
13	¿Se muestra correctamente el diagrama de paquetes de casos de uso?	4				

Figura 5.4 Resumen del formato de verificación del documento RDS

LISTA DE REVISIÓN DE PARES - CÓDIGO FUENTE JAVA

I. Datos Generales

Componente:

Código :		Elaborado por :	

II. Resumen de las sesiones

N°.Sesión	Fecha de sesión	Fecha estimada de corrección	Cantidad de defectos	Horas	Observaciones

III. Lista de Verificación

Ítem	Preguntas iniciales	Referencia	Cumple	Sesión de reporte	Sesión de corrección	Observaciones
	Variables y constantes					
1	¿Los nombres de las variables son descriptivos?					
2	¿Existen nombres de variables que podrían causar confusión?					
3	¿Está cada variable local inicializada?					
4	¿Existen variables que hayan sido declaradas y no se estén usando?					
5	¿Son los nombres de las constantes descriptivas?					
6	¿Existen variables que deberían ser constantes?					
	Expresiones					
7	¿Se está usando paréntesis en las expresiones complejas para la claridad del código?					
8	¿Para expresiones con más de un operador se esta asumiendo el orden de evaluación?					
9	¿Existen pérdidas de información por hacer un cast. a una variable ?					
10	¿Se están comparando los operadores correctamente?					
11	¿Todos los loops terminan?					
12	¿Es el mejor loop utilizado para el cada caso?					
13	¿Cada switch tiene un valor por default?					
	Comentarios					
14	¿Cada función, módulo y archivo tiene una cabecera de comentarios apropiada?					
15	¿Cada variable y constante tiene un comentario?					

Figura 5.5 Resumen del formato de verificación del Código Fuente

5.1.3 DETERMINACIÓN DE RESPONSABILIDADES DE EVALUACIÓN DE MÉTRICAS Y FORMATO DE SEGUIMIENTO

El proceso de Desarrollo de Software – PDSA cuenta con los siguientes indicadores y métricas:

Indicador 1: I1 En cada fase de un ciclo de desarrollo se efectúan todas las actividades de verificación, validación o prueba correspondientes.

Las métricas asociadas a este indicador son las siguientes:

- M1.1 Verificaciones realizadas durante cada fase del ciclo de desarrollo de software
- M1.2 Validaciones realizadas durante cada fase del ciclo de desarrollo de software
- M1.3 Pruebas de Sistema realizadas durante el ciclo de desarrollo de software

Indicador 2: I2 En cada fase de ciclo de desarrollo se generan productos que incluyen características mínimas requeridas para su aprobación.

Las métricas asociadas a este indicador son las siguientes:

- M2.1 Nivel de éxito en la evaluación del Reporte de Especificaciones de Software v1 – RES v1.
- M2.2 Nivel de éxito en la evaluación del Reporte de Especificaciones de Software v2 – RES v2.
- M2.3 Nivel de éxito en la evaluación del Reporte de Diseño de Software – RDS.
- M2.4 Nivel de éxito en la evaluación del Código Fuente
- M2.5 Nivel de éxito de las Pruebas realizadas durante el ciclo de desarrollo de software

Para la implementación del primer proyecto piloto, se determinó de común acuerdo con el equipo docente involucrado, asignar las siguientes responsabilidades de evaluación de métricas:

- Los profesores del curso **Desarrollo de Aplicaciones Web 2** evaluarán las métricas: M1.1, M2.3 y M2.4

- Los profesores del curso **Certificación de Calidad** evaluarán las métricas: M1.1, M1.2, M1.3, M2.1, M2.2, M2.5

5.1.3.1 Actividades aprobadas por el equipo docente

1. Las métricas serán aplicadas por los docentes a cada una de las secciones que tengan a su cargo.

2. Se definió para esta actividad un formato de seguimiento de métricas.

3. Estas métricas deberán ser reportadas al coordinador del curso al finalizar la fase del ciclo de desarrollo de software que le corresponda.

Se muestra a continuación en la figura 5.6 un ejemplo de aplicación del formato de seguimiento de métricas:

FORMATO DE SEGUIMIENTO DE MÉTRICAS

T6AC - Grupo 2 - Luis García

I1 En cada fase de un ciclo de desarrollo se efectúan todas las actividades de verificación, validación o prueba correspondientes

	Verificaciones Proyectadas (B)	Verificaciones Realizadas (A)	Porcentaje de éxito (A/B)
Identificación de Requerimientos			
Verificación del RES v1	3	3	100.00%
Análisis de Requisitos			
Verificación del RES v2	3	3	100.00%
Definifición de Arquitectura y Diseño Detallado			
Verificación del RDS	3	3	100.00%
Costrucción			
Verificación código Java	3	2	66.67%

Observaciones de uso del formato:
Se registran las incidencias ocurridas en cada fase en relación al no cumplimiento del valor esperado.

Ejemplo:
En la fase de Construcción, de los tres equipos de proyecto en la sección, solo dos realizaron la verificación del código fuente en la fecha establecida porque no habían culminado con la elaboración del componente software

A/B = 2/3 = 0.66 = 66.67%

I2. En cada fase de ciclo de desarrollo se generan productos que incluyen características mínimas requeridas para su aprobación.

	Características Definidas para el entregable (B)	Características evaluadas exitosamente (A)	Porcentaje de éxito (A/B)
Análisis de Requisitos (RES v2)			
G1	34	27	79.41%
G2	34	20	58.82%
G3	34	18	52.94%
		Promedio:	63.73%

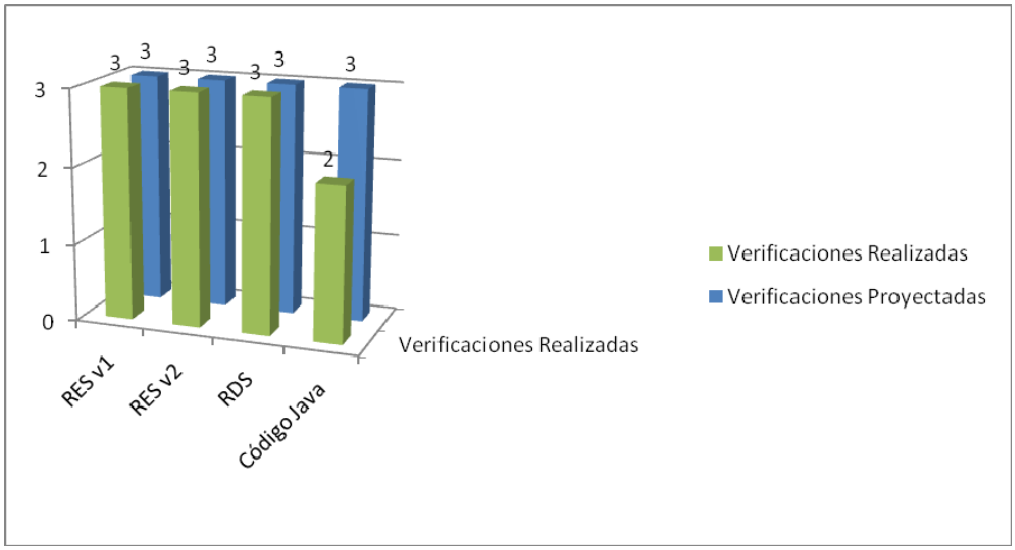
Figura 5.6 Formato ejemplo de seguimiento de métricas

4. Una vez evaluadas las métricas de un indicador, el docente actualizará el formato de seguimiento con los gráficos respectivos para proporcionar mayor claridad en su evaluación posterior.

Se muestra a continuación un ejemplo de gráfico generado para el formato de seguimiento mostrado en la figura anterior:

I1 En cada fase de un ciclo de desarrollo se efectúan todas las actividades de verificación, validación o prueba correspondientes.

T6AC - Grupo 2 - Luis García



Cuadro 5.1 Ejemplo de Verificaciones realizadas

5. Los coordinadores de los cursos Certificación de Calidad y Desarrollo de Aplicaciones Web 2, generarán al finalizar el ciclo de desarrollo de software, un consolidado final en base a todos los formatos enviados por los docentes de sus respectivos equipos.

5.2 DETERMINACIÓN DEL CRONOGRAMA DE EVALUACIÓN

Una vez culminado el proceso de capacitación del equipo docente en la metodología MEDESOFT, la Dirección académica de la Escuela de Tecnología así como el área de acreditación de CIBERTEC, convocaron a una reunión final de coordinación a los docentes coordinadores de los siguientes cursos:

- **Quinto Ciclo**: Desarrollo de Aplicaciones Web 1 y Calidad de Software
- **Sexto Ciclo:** Desarrollo de Aplicaciones Web 2 y Certificación de Calidad

Como resultado de esta reunión se definió el cronograma de verificaciones y/o validaciones a ser ejecutado durante el proyecto piloto para el ciclo académico 2008-2. Este ha sido expresado en semanas académicas, iniciándose la primera semana de clases el 16 de Octubre de 2008.

Verificación o validación	Semana Académica	Producto
Ver1	Semana 04	Reporte de Especificaciones de Software v1 – RES v1
Val1	Semana 05	Reporte de Especificaciones de Software v1 – RES v1
Ver2	Semana 06	Plan de Pruebas de Sistema – PPRS
Ver3	Semana 06	Reporte de Especificaciones de Software v2 – RES v2
Ver4	Semana 09	Reporte de Diseño de Software - RDS
Ver5	Semana 11	Componente Software

CUADRO 5.2 CRONOGRAMA DE VERIFICACIONES Y VALIDACIONES DEL PROYECTO PILOTO VOTACIÓN ELECTRÓNICA NO PRESENCIAL

5.3 EJECUCIÓN DEL PROCESO Y EVALUACIÓN DE MÉTRICAS

Se contó con un universo integrado por los alumnos del sexto ciclo de la carrera de computación Informática, matriculados en los cursos Certificación de Calidad y Desarrollo de Aplicaciones Web II. Las secciones involucradas fueron las siguientes:

> **Sede Central. Av. Salaverry 2255. San Isidro:** Secciones T6AC, T6BC y T6CC
>
> **Sede Norte. Av. Carlos Izaguirre 233. Independencia:** Sección T6AN

Cada sección cuenta aproximadamente con 40 alumnos matriculados.

5.3.1 EJECUCIÓN DEL PROCESO

Se implementó exitosamente el primer proyecto piloto integrador: **Votación Electrónica no Presencial**, utilizando el proceso de Desarrollo de Software por Alumnos – PDSA.

El proceso en mención utilizó dentro de los cursos de sexto ciclo 0268 Desarrollo de Aplicaciones Web II y 0124 Certificación de Calidad los siguientes formatos:

- Reporte de Solicitud de Requerimiento - RSRQ
- Plan de Proyecto - PP
- Reporte de Especificación de Software –RES, en sus dos versiones.
- Reporte de Diseño de Software - RDS
- Lista de Chequeo del documento RES
- Lista de Chequeo del documento RDS
- Lista de Chequeo del Código Fuente
- Reporte de Actividades - RACT

Los formatos fueron utilizados por TODOS los equipos de alumnos contándose con la evidencia de los mismos ya que cada equipo debió hacer entrega en formato digital (CD) de toda la documentación y solución de software implementada en sus proyectos.

Para esta primera versión, todos los equipos implementaron la misma solución de software, compitiendo entre ellos por brindar al líder usuario del proyecto: el profesor coordinador del curso Calidad de Software, la mejor solución a las necesidades planteadas.

El equipo docente, en general, cumplió con la implementación del proceso PDSA en sus respectivas secciones, así como la elaboración y envío del formato de seguimiento de métricas del proceso a los coordinadores del curso.

Los coordinadores de curso, elaboraron el informe final con el consolidado de las métricas obtenidas en el proceso PDSA. Este informe ha sido presentado a la dirección académica de la escuela de tecnología y al área de acreditación de CIBERTEC. Este documento será contrastado con los resultados obtenidos en las encuestas de criterios de desempeño por Resultado de Aprendizaje de la Carrera.
Es importante destacar que como resultado de esta primera implementación, tanto los docentes como los alumnos, generaron importantes sugerencias de mejora tales como:

- Simplificación de los formatos de documentos,
- Reformulación de algunos ítems de verificación en los formatos de chequeo,
- Mejora de la coordinación entre los cursos involucrados a nivel de entregables,
- Profundizar contenidos en cursos prerrequisitos que se asumen son dominados por los alumnos en el proceso PDSA

Estas sugerencias han sido evaluadas por los coordinadores de los cursos involucrados y la dirección académica de la escuela de tecnología, por lo que su implementación se concretará antes del inicio del ciclo académico 2009-1

5.3.2 RESULTADOS OBTENIDOS A TRAVÉS DE LAS MÉTRICAS DEL PROCESO

Una vez consolidadas las métricas del proceso, se obtuvieron los siguientes resultados finales:

5.3.2.1 Indicador 1 (I1)

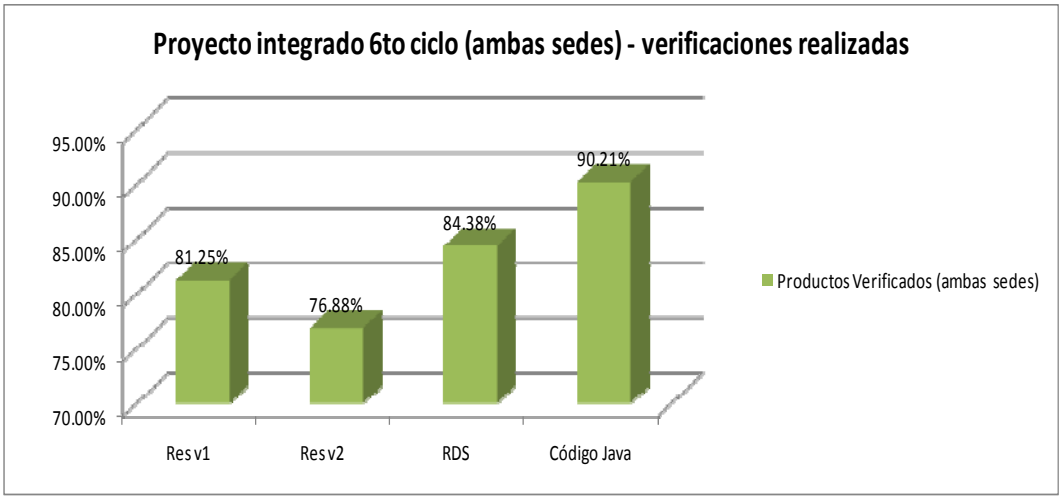
En cada fase de un ciclo de desarrollo se efectúan todas las actividades de verificación, validación o prueba correspondientes.

- Métrica M1.1: Verificaciones realizadas durante cada fase del ciclo de desarrollo de software:

		Productos Verificados (ambas sedes)	Productos Verificados Sede Norte	Productos Verificados Sede Central
Identificación de Requerimientos - *Verificación del RES v1*	Res v1	81.25%	70.00%	85.00%
Análisis de Requisitos - *Verificación del RES v2*	Res v2	76.88%	45.00%	87.50%
Definición de Arquitectura y Diseño de Detallado - *Verificación del RDS*	RDS	84.38%	72.50%	88.33%
Construcción - *Verificación del código Fuente*	Código Java	90.21%	83.33%	92.50%

FIGURA 5.7 VERIFICACIONES REALIZADAS DURANTE CADA FASE DEL CICLO DE DESARROLLO DE SOFTWARE

Se muestra a continuación en el cuadro 5.2, el consolidado de la métrica M1.1:



CUADRO 5.3 VERIFICACIONES REALIZADAS DURANTE CADA FASE DEL CICLO DE DESARROLLO DE SOFTWARE

Como se puede observar en el cuadro, se realizaron todas las actividades de verificación planificadas durante el presente proyecto piloto. Debemos tener presente que esta métrica mide la realización de las actividades de verificación, es decir el cumplimiento de la actividad, mas **NO**, su realización exitosa.

De todas las verificaciones realizadas, la segunda, (Verificación del documento Reporte de Especificaciones de Software – RES v2) fue la única que no alcanzó el nivel esperado establecido por la dirección de la Escuela de Tecnología: 80.00 %

En general, las verificaciones que no se pudieron realizar, no se concretaron debido a que los equipos de proyecto de alumnos no presentaron a tiempo el entregable por motivos ajenos al curso. Sin embargo, también se tuvieron casos, en especial en Sede Norte, en que las entregas no se concretaron por falta de coordinación entre los cursos involucrados, esto evidenció la importancia de la metodología MEDESOFT: al hacer como parte del proceso verificaciones y solicitarse reportes de seguimiento en cada fase del ciclo de vida de desarrollo del software, se detectó en **etapas tempranas** el problema, tomándose inmediatamente las medidas correctivas respectivas.

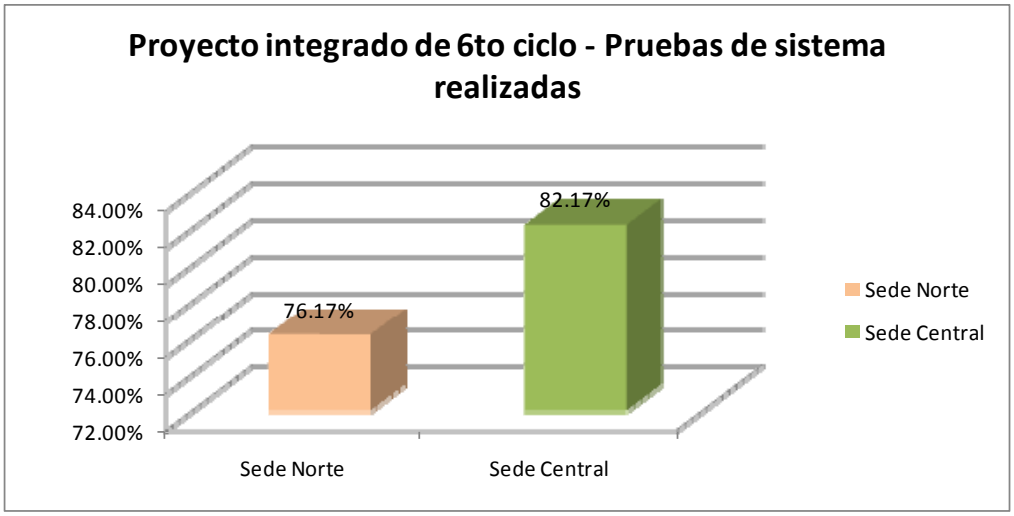
La detección temprana de problemas durante el desarrollo del proyecto es una de las importantes mejoras logradas con MEDESOFT. Esta mejora es conocida también como **visibilidad** del proyecto.

- Métrica M1.3 Pruebas de Sistema realizadas durante el ciclo de desarrollo de software:

	Pruebas de Sistema Realizadas
Sede Norte	76.17%
Sede Central	82.17%

FIGURA 5.8 PRUEBAS DE SISTEMA REALIZADAS

Se muestra a continuación en el cuadro 5.3, el consolidado de la métrica M1.3:



CUADRO 5.4 PRUEBAS DE SISTEMA REALIZADAS

La realización de las pruebas de sistema estuvo a cargo del curso de Certificación de Calidad, y como se puede observar en el cuadro, se logró una mayor efectividad en los equipos de proyecto de sede central aunque sólo por un ligero margen. Se realizó una capacitación a los alumnos dentro del curso, sobre la importancia de realizar las pruebas de sistema, tipos de prueba y herramientas existentes para generarlas así como para registrar los errores encontrados.

Fue una nota del curso Certificación de Calidad (Segunda evaluación continua de teoría) la presentación del plan de pruebas elaborado por cada equipo de proyecto. De la misma manera, fue también evaluado el correcto registro y seguimiento de los errores encontrados en la ejecución de las pruebas, para lo cual utilizaron las herramientas de registro de errores vistas durante las clases de laboratorio.
Se enfatizó en la correcta elaboración de los casos de prueba: se hizo énfasis en corregir la sintaxis que utilizaban los alumnos para describir los casos de pruebas.

5.3.2.2 Indicador 2 (I2)

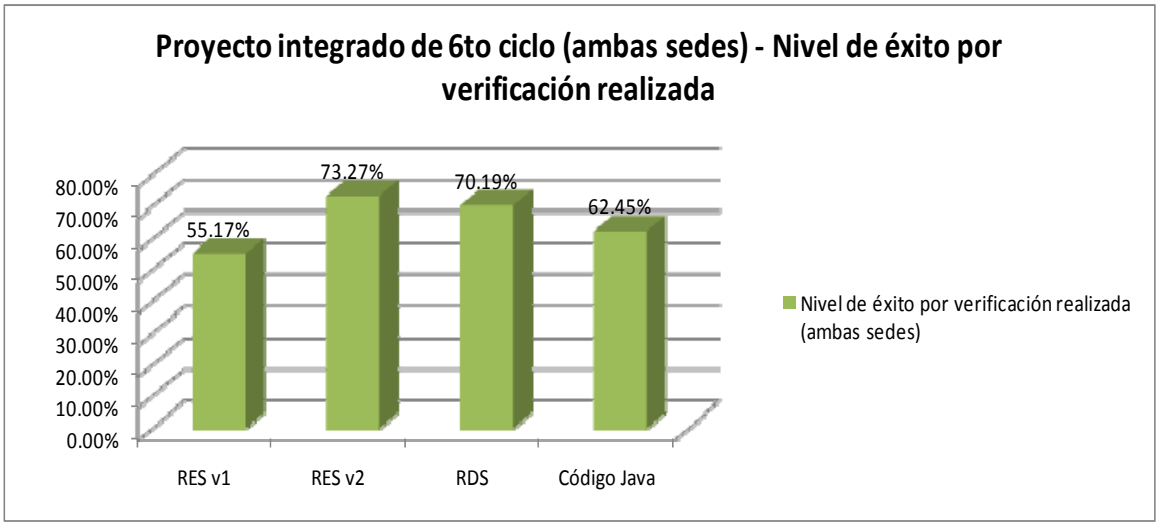
En cada fase de ciclo de desarrollo se generan productos que incluyen características mínimas requeridas para su aprobación.

- Métrica M2.1 Nivel de éxito en la evaluación del Reporte de Especificaciones de Software v1 – RES v1.
- Métrica M2.2 Nivel de éxito en la evaluación del Reporte de Especificaciones de Software v2 – RES v2.
- Métrica M2.3 Nivel de éxito en la evaluación del Reporte de Diseño de Software – RDS.
- Métrica M2.4 Nivel de éxito en la evaluación del Código Fuente

		Nivel de éxito por verificación realizada (ambas sedes)
Identificación de Requerimientos - *Verificación del RES v1*	RES v1	55.17%
Análisis de Requisitos - *Verificación del RES v2*	RES v2	73.27%
Definición de Arquitectura y Diseño de Detallado - *Verificación del RDS*	RDS	70.19%
Construcción - *Verificación del código Fuente*	Código Java	62.45%

FIGURA 5.9 NIVEL DE ÉXITO POR VERIFICACIÓN REALIZADA

Se muestra a continuación en el cuadro 5.3, el consolidado de las métricas M2.1, M2.2, M2.3 y M2.4:



CUADRO 5.5 NIVEL DE ÉXITO POR VERIFICACIÓN REALIZADA

Como se puede observar en el cuadro previo, se evaluó el **nivel de éxito** de cada una de las actividades de verificación planificadas durante el presente proyecto piloto. Es decir, para cada verificación realizada por un equipo de proyecto de alumnos, se aplicó la siguiente fórmula:

Nivel de éxito = A / B

A: Cantidad de características evaluadas de manera exitosa
B: cantidad de características definidas para el entregable.

Ejemplo: Si la lista de chequeo del documento Reporte de Diseño de Software – RDS cuenta con **23** elementos de verificación y de los 23 elementos el equipo de Proyecto de Alumnos: "Si se puede" ha cumplido con **18**, entonces el nivel de éxito obtenido por dicho equipo será: 18/23 = 78.26 %

Si bien los resultados obtenidos en las métricas del Indicador 2, se encuentran ligeramente por debajo del nivel esperado de la Dirección de la escuela de Tecnología y el área de Acreditación de CIBERTEC (75.00 %), estos son bastante alentadores y han sido recibidos con mucho optimismo por las autoridades de la institución en atención a que:

- Es la primera vez que se ha logrado **medir de manera objetiva** el proceso de desarrollo de software aplicado dentro de los proyectos integrados de la Carrera de Computación e Informática. Esta evidencia es muy importante para el proceso de acreditación ABET de la carrera.

- Se ha logrado detectar **problemas y corregirlos en etapas tempranas** del ciclo de desarrollo de software (análisis y diseño) y no recién en plena etapa de construcción del producto software.

En relación a los factores que influyeron en el nivel de éxito alcanzado en las métricas del Indicador 2, debemos destacar:

1. La mayor cantidad de errores encontrados en el documento Reporte de Especificaciones de Software – RES, se debió a la no correspondencia entre los prototipos y la descripción de los casos de uso, la no utilización y/o referencia de las reglas de negocio en los diversos casos de uso y la no correcta explicación de la funcionalidad de los casos de uso.

2. Tanto en los documentos Reporte de Especificaciones d Software – RES como en el Reporte de Diseño de Software - RDS se detectaron vacíos en la especificación de los requisitos No Funcionales del producto Software, los equipos no describían correctamente aspectos de seguridad, performance, etc.

3. A nivel de código fuente, se evidencia la necesidad de institucionalizar en los alumnos el uso de estándares de codificación, la mayor cantidad de errores se debieron a que los equipos de proyecto solo aplicaron parcialmente los estándares de codificación definidos para el proyecto piloto: así por ejemplo se detectó que los comentarios en las clases fuentes no eran completos, de la misma manera la definición de nombres de paquetes y variables estandarizadas.

Todas estas consideraciones han sido tomadas en cuenta como parte de la retroalimentación generada por el proceso y se deberán mejorar en la siguiente implementación durante el ciclo académico 2009-1.

5.4 OPORTUNIDADES DE MEJORA

Como resultado de la primera aplicación del proceso de Desarrollo de Software por Alumnos – PDSA, alumnos y docentes participantes del mismo, detectaron importantes oportunidades de mejora tanto en el proceso como en los productos generados, las cuales detallamos a continuación:

5.4.1 LECCIONES APRENDIDAS

Se detectaron oportunidades de mejora en las listas de chequeo de los entregables: algunas preguntas de verificación dentro de las listas de chequeo pueden causar confusión en el evaluador:

> Por ejemplo, en la lista de chequeo de código fuente: en la pregunta ¿Existen nombres de variables que podrían causar confusión? La columna de verificación indica si cumple o no cumple. El evaluador puede marcar **NO** como respuesta a la pregunta, cuando en realidad la respuesta debe ser **Sí CUMPLE**.

Se detectaron oportunidades de mejora en el formato Reporte de Diseño de Software – RDS, existen secciones que implican excesiva documentación por parte de los equipos de proyecto, las cuales podrían ser simplificadas para agilizar el proceso:

> Por ejemplo, la especificación detallada de cada uno de los diagramas de secuencia (realizaciones de diseño) para cada caso uso del sistema.

Se requiere mayor capacitación a los **alumnos y docentes** en la definición e implementación de requisitos no funcionales:

> Por ejemplo, en los documentos Reporte de Especificaciones de Software - RES y Reporte de Diseño de Software - RDS se hace referencia al siguiente requisito: "alta disponibilidad", sin embargo, la mayoría de equipos de proyecto no sabe cómo se implementa ese requerimiento.

Se han detectado algunas deficiencias en los alumnos en relación a la correcta diagramación de las vistas de despliegue, implementación e integración de software. Estas debieron ser aprendidas en el curso **Diseño de Software** (cuarto ciclo), sin embargo, al parecer no se llegaron a tratar dichos temas.

Se requiere mayor capacitación y compromiso dentro del equipo docente en relación a la correcta aplicación de la metodología MEDESOFT y el Proceso de Desarrollo de Software por alumnos – PDSA. Algunos docentes aún no entienden

el proceso, por lo cual, no lograron asesorar correctamente a los alumnos que tenían a su cargo.

5.5 RESUMEN DEL CAPÍTULO

En este capítulo se han detallado las actividades realizadas para la implementación del primer proyecto piloto de desarrollo de software utilizando la metodología MEDESOFT. MEDESOFT en su primera versión (1.0), cuenta con el proceso de Desarrollo de Software por Alumnos – PDSA, el cual fue aplicado por los alumnos y el equipo docente de los cursos Certificación de Calidad y Desarrollo de Aplicaciones Web II.

El éxito alcanzado con esta primera implementación requirió de un alto nivel de compromiso por parte del equipo docente, el cual debió participar de actividades de inducción para la comprensión del proceso y su posterior difusión con los alumnos, definición de entregables a evaluar y las técnicas de revisión a aplicar, así como la aplicación ordenada de las métricas del proceso para su posterior análisis. Durante la ejecución del proceso se identificaron por cada etapa del ciclo de desarrollo importantes oportunidades de mejora, evidenciándose también la importancia de las actividades de verificación, validación y pruebas para detectar errores en etapas tempranas del desarrollo y no durante la entrega del producto final.

Tanto el equipo docente como los alumnos integrantes de los equipos de proyecto generaron importantes recomendaciones las cuales se han consolidado como parte de las lecciones aprendidas del proceso, y serán implementadas en la siguiente versión del mismo como evidencia de un proceso continuo de mejora.

Capítulo 6
Conclusiones y Recomendaciones

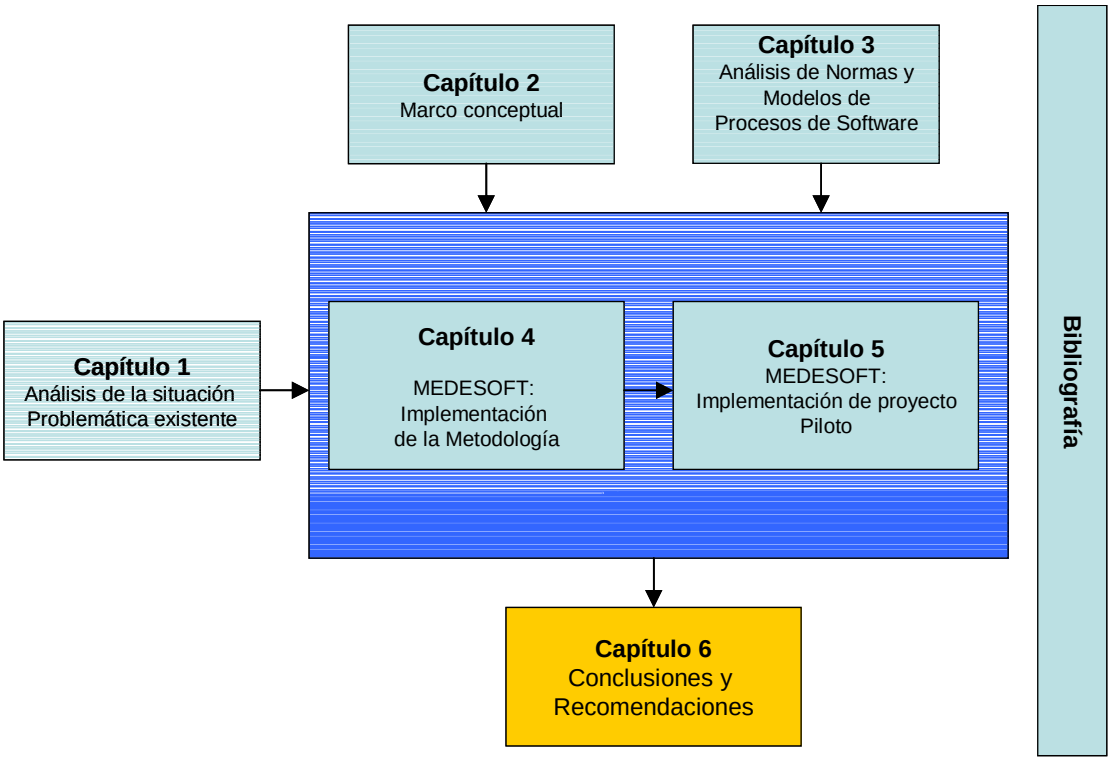


FIGURA 6.1 UBICACIÓN EN LA LECTURA DEL DOCUMENTO

6.1 CONCLUSIONES

La implementación del Proceso de Desarrollo de Software por alumnos – PDSA, a través del primer proyecto piloto implementado durante el ciclo académico 2008-2, permitió generar las siguientes conclusiones:

1. Se cuenta por primera vez en CIBERTEC con un proceso de desarrollo de software, plenamente definido, ejecutado por alumnos, el cual es capaz de ser medido de manera objetiva a través de sus métricas, por lo tanto, también susceptible de ser mejorado constantemente.

2. A diferencia de ciclos académicos anteriores, los alumnos de manera planificada y en base al procedimiento, asumieron diferentes roles a lo

largo del ciclo de desarrollo de software. Por ejemplo, se desempeñaron adecuadamente en los roles de:

- Analistas,
- Diseñadores,
- Diseñadores de Interfaces de Usuario,
- Revisores,
- Responsable de Pruebas,
- Responsable de Desarrollo, etc.

Cada uno de estos roles está plenamente definido en el proceso PDSA así como sus responsabilidades.

3. El proceso de desarrollo de software – PDSA, facilita también la labor docente y el proceso de aprendizaje de los estudiantes. Permite poner en práctica todos los conocimientos adquiridos por los alumnos a lo largo de la carrera y facilita la evaluación de cada alumno dentro del equipo de proyecto:

    - Cada integrante del equipo siempre tiene un rol y una responsabilidad asignada
    - Cada integrante elabora siempre un reporte de Actividades en el que se contrasta lo planificado versus lo que verdaderamente ha realizado

4. A través de los documentos y componentes software generados como entregables en cada una de las fases del ciclo de desarrollo de software, se ha logrado **evidenciar** que efectivamente los alumnos de la carrera de Computación e Informática alcanzan los resultados de aprendizaje de la carrera, en especial los asociados a la metodología MEDESOFT:

    - **RAC.1 Conocimiento del ciclo de desarrollo de software**. Se desempeñan asumiendo diferentes roles, en los equipos de proyectos de desarrollo de las soluciones de software que realizan.

    - **RAC.2 Participación en el aseguramiento de la calidad del software**. Participa de la planificación, diseño y comprobación de la calidad del proceso de desarrollo y del producto software mediante actividades de verificación, validación y pruebas que permitan asegurar la calidad de los entregables.

5. Se ha logrado reducir el tiempo de desarrollo de un proyecto por parte de los equipos de proyecto de alumnos y mejorar su calidad. Esto gracias a

que ahora se detectan y corrigen los errores en etapas tempranas del ciclo de desarrollo en base a las verificaciones, validaciones y /o pruebas preestablecidas en el proceso:

- Muchas correcciones al proyecto se hacen "en el papel" (documentos de las primeras fases), lo cual es mucho más económico y rápido a tener que hacerlo en la "codificación del programa".
- El ahorro en actividades de re-trabajo, ha facilitado el poder hacer otras: por **primera ve**z se ha logrado durante el ciclo académico 2008-2 hacer **pruebas funcionales** al sistema.

6. Los alumnos participantes del proyecto han logrado comprobar la importancia y las ventajas de desarrollar software utilizando un proceso especialmente definido, estandarizado y con técnicas y métodos de desarrollo y calidad de software.

7. Alumnos y docentes se encuentran más comprometidos en el proceso de aprendizaje continuo, dado que es parte del proceso rescatar las sugerencias, críticas y mejoras implementadas. Esto como parte de las lecciones aprendidas y mejora continua del proceso.

6.2 RECOMENDACIONES

1. En atención a las sugerencias generadas por el equipo docente durante la primera implementación del Proceso de Desarrollo de Software por Alumnos – PDSA, se sugiere en el corto plazo (antes del inicio del ciclo académico 2009-1):

   Convocar a una capacitación en la metodología MEDESOFT, con carácter de obligatoria, a todo el equipo docente de los cursos:

   - Cuarto ciclo: Análisis y Diseño II y Lenguaje de Programación II
   - Quinto ciclo: Diseño de Proyectos, Calidad de Software y Desarrollo de Aplicaciones Web I
   - Sexto ciclo: Certificación de Calidad y Desarrollo de Aplicaciones Web II

   De la misma manera se requiere capacitar al equipo docente de los ciclos Quinto y Sexto en estrategias de implementación de requisitos No Funcionales, tales como:
   - Alta disponibilidad
   - Escalabilidad horizontal y vertical
   - Seguridad Web
   - Performance, entre otros

2. Seleccionando actividades específicas dentro del Proceso de Desarrollo de Software por Alumnos – PDSA, se propone para el ciclo 2009-1, realizar proyectos integrados en los siguientes cursos:

   Proyecto de 4to Ciclo: 0258 Análisis y Diseño de Sistemas II + 0480 Lenguaje de Programación II

   Proyecto de 5to ciclo: 0265 Desarrollo de Aplicaciones Web I + 0301 Diseño de Proyectos + 0302 Calidad de Software.

3. Culminar con la definición de los formatos Matriz de Trazabilidad – MTRZ y Manual de Usuario – MUSU, que quedaron pendientes de definición en el proceso de Desarrollo de Software por Alumnos - PDSA para el ciclo académico 2008-2. Deben ser definidos para aplicar las verificaciones **Ver6** y **Ver7** durante el ciclo académico 2009-1.

4. Incluir como parte del proceso de Desarrollo de Software por Alumnos - PDSA la elaboración de un Plan de Pruebas de Integración – PPRI y sus respectivos casos de prueba de modo que puedan ser ejecutados por los alumnos antes de las pruebas de Sistema. Se sugiere, a diferencia del ciclo

académico 2008-2, que los equipos de alumnos desarrollen módulos de software diferentes que deban ser luego integrados para obtener la solución de software final.

Bibliografía

PIATTINI, Mario. Fábricas de Software: experiencias, tecnologías y organización, Editorial Alfaomega – Rama, México, 2007.

PRESSMAN S. Roger. "Ingeniería de Software" (5ta Edición), Editorial McGraw-Hill, España, 2002.

Norma Técnica Peruana NTP-ISO/IEC 9126: Calidad del Producto Software. Lima, 2004. Indecopi.

Norma Técnica Peruana NTP-ISO/IEC 12207: Procesos del ciclo de vida del Software. Lima, 2006. Indecopi.

Norma Técnica Peruana NTP-ISO/IEC 14598: Proceso de Evaluación del Producto Software. Lima, 2006. Indecopi.

Oktaba Hanna. MOPROSOFT: Modelo de Procesos para la Industria del Software. [En línea] Agosto de 2004. <http://www.software.net.mx/desarrolladores/directorios/asociaciones/amcis/MoProSoft.htm> [Consulta: Julio de 2007]

Softex. MPS.BR. Mejora del proceso de Software Brasileño. [En línea]. <http://www.softex.br/portal/softexweb/uploadDocuments/_mpsbr/Apresentação%2024ABR2008%20MPS.BR%20Forum%20TIC%20Governo%202008.pdf >. [Consulta: Julio de 2007]

Software Engineering Institute. CMMI: Capability Maturity Model Integration. [En línea]. <http://www.sei.cmu.edu/cmmi/general/index.html>. [Consulta: Agosto de 2007]

Programa iberoamericano de ciencia y tecnología para el desarrollo. COMPETISOFT: Mejora de Procesos Software para pequeñas empresas. [En línea]. < http://alarcos.inf-cr.uclm.es/Competisoft/>. [Consulta: Enero de 2008]

Anexo A
Reporte de Especificaciones de Software - RES

Reporte de Especificación de Software (RES)

[Nombre de Gerencia Sponsor del proyecto]

[Nombre del proyecto]

[Mes y Año Inicio del Proyecto]

[Este documento es la plantilla base para elaborar el documento Reporte de Especificación de Software. Los textos que aparecen entre paréntesis rectos son explicaciones de que debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda. En caso que alguna de las secciones del presente documento no aplique a su proyecto pueden usarse las frases "No hay cambios", "No hay impacto en esta sección", "La solución que se está implementando no tiene impacto en esta sección", "No aplican para el proyecto" (No borrar secciones del documento)]

Elaborado por:	Revisado por:	Aprobado por:
Fecha: / /	Fecha: / /	Fecha: / /

HISTORIAL DE REVISIONES

Versión	Autor	Descripción	Fecha de Elaboración	Fecha de Revisión	Revisado por
<x.x>	<Persona que elabora el documento>	<Detalles>	<Fecha de Elaboración>	<Fecha de Revisión>	<Persona(s) que revisa(n) el documento>

Contenido

1. Antecedentes

[Describa la situación actual y las necesidades o problemas que se pretende atender. Recuerde que debe tomar como información base lo registrado en la solicitud de requerimiento (RSRQ) y documento de planificación del proyecto (PP). Recuerde que es posible que en esta sección se pueda complementar la información de los documentos base RSRQ y PP.]

2. Objetivos

[Es la explicación resumida, pero clara, de lo que se pretende con el requerimiento, es decir la visión de éste. Recuerde que debe tomar como información base lo registrado en la solicitud de requerimiento (RSRQ) y documento de planificación del proyecto (PP). Recuerde que es posible que en esta sección se pueda complementar la información de los documentos base RSRQ y PP.]

3. Alcance

3.1. Dentro del Alcance

[En esta sección deberá incluir el alcance definido en el PP. Es posible detallar el alcance siempre y cuando no varíe en cuanto al original definido en el PP.]

3.2. Fuera del Alcance

[En esta sección deberá incluir lo que no es parte del alcance del proyecto definido en el PP. Es posible detallar lo que queda fuera del alcance siempre y cuando no varíe en cuanto al original definido en el PP.]

3.3. Restricciones

[En esta sección deberá incluir las restricciones del proyecto definidas en el PP. Es posible detallar más restricciones relacionadas con los requisitos siempre y cuando no varíen las que se definieron originalmente en el PP.]

3.4. Supuestos

[En esta sección deberá incluir los supuestos del proyecto definidos en el PP. Es posible detallar más supuestos relacionados con los requisitos siempre y cuando no varíen las que se definieron originalmente en el PP.]

4. Procesos de Negocio

4.1. Lista de Casos de Uso de Negocio

[En esta sección deberá listar los casos de uso de negocio que se obtuvieron a partir de los procesos de negocio identificados dentro del ámbito de la solución y a los cuales se les dará el soporte con el producto software. Cada Caso de Uso de Negocio deberá ser identificado con un código único y correlativo. Ejemplo CUN01. De ser necesario deberá incorporar un diagrama de casos de uso de negocio.]

4.2. Realización de los Casos de Uso de Negocio

[En esta sección deberá desarrollar los diagramas de actividades y diagrama de clases de negocio por cada Caso de Uso de Negocio identificado en la sección 4.1. Por cada juego de diagramas deberá identificar cuáles serán las actividades que serán automatizadas.]

4.3. Lista de Trabajadores de Negocio

[En esta sección deberá listar a los trabajadores de negocio incluyendo una descripción por cada uno.]

Trabajador del Negocio	Descripción
Asistente de Gestión	Trabajador encargado de procesar las rectificaciones de los ciudadanos que lo solicitan. También coordina las entregas de hologramas.

4.4. Reglas de Negocio

[En esta sección deberá identificar las reglas que regulan la estructura del negocio y cómo ellos operan afectando el funcionamiento de los procesos de negocio. Dichas reglas de negocio son las que se considerarán para el diseño del sistema. Cada Regla de Negocio deberá ser identificada con un código único y correlativo. Ejemplo: RN01. Para identificar las reglas de negocio puede considerar la siguiente clasificación:

Reglas de Estructura: Ejemplo (Todo pedido debe ser realizado por un cliente, y que el mismo debe estar dado de alta. Además una vez que el cliente haya hecho algún pedido, se deberá garantizar que no es posible eliminarlo, al menos que previamente se eliminen todos sus pedidos)

Reglas de Derivación: Ejemplo (El total de un pedido se puede calcular a partir de distintas líneas que lo componen, mientras que el total de cada línea se puede calcular a partir del número de unidades vendidas y el precio por unidad)

Reglas de Interfaz o de Modelo de Datos: Ejemplo (No hay precio de artículos negativos, el sexo de una persona sólo puede ser masculino o femenino, una fecha tiene que ser siempre una fecha válida - no existe 30 de febrero)

Reglas de Operación o Reglas de Flujo: Ejemplo (Un cliente puede hacer una petición de análisis al laboratorio que anota un encargado: hecho esto, se genera un parte para uno o más analistas, estos realizan las mediciones correspondientes y devuelven los partes con la información pertinente, a partir de la cual se genera un informe de análisis, que será un análisis válido solo cuando sea firmado por los responsables de garantizar su corrección)

Regla de Estímulo Respuesta: Ejemplo (Para un saldo existe una regla de interfaz que indica que éste debe ser un número, pero además puede haber una regla que indique que el saldo nunca puede ser menor que cierta cantidad tope establecida para cierto tipo de clientes)]

5. Requisitos Funcionales

[De acuerdo a lo solicitado explícitamente por el área usuaria, listar todos los requisitos funcionales del producto software. Considere que los requisitos funcionales que liste deberán ser asociados posteriormente a los casos de uso (funciones de software). Cada Requisito Funcional deberá ser identificado con un código único y correlativo. Ejemplo: RF01.]

Código	Descripción	Proceso de Negocio
[Código del requisito funcional]	[Descripción detallada del requisito funcional.]	[Identificador del proceso de negocio asociado]
RF-001	[Descripción detallada del requisito funcional 1.]	[CUN01]
RF-002	[Descripción detallada del requisito funcional 2.]	

Código	Descripción	Proceso de Negocio
...	....	
RF-00n	[Descripción detallada del requisito funcional n.]	

6. Requisitos No Funcionales

[Listar los requisitos no funcionales los mismos que deberán ser considerados para el modelo de calidad de producto. Cada Requisito No Funcional deberá ser identificado con un código único y correlativo. Ejemplo: RNF01.]

Tipo de Requisito	Código	Descripción
[Nombre del tipo de requisito no funcional]	[Código del requisito no funcional]	[Descripción detallada del requisito no funcional.]
Restricciones del Diseño [Definir cualquier tipo de restricción de diseño, tales como: proceso de desarrollo de software, sistemas operativos, lenguajes de programación, administrador de base de datos, conexión a la BD, generador de reportes, manejo de información, etc.]	RNF-001	[Descripción detallada del requisito no funcional 1.]
	RNF-002	[Descripción detallada del requisito no funcional 2.]
Componentes a Adquirir [Identificar los componentes que se deben adquirir o tener en cuenta, para llevar acabo el desarrollo y ejecución del sistema. Ejemplo: lenguajes de programación, servidores, estaciones de trabajo, etc.]	RNF-003	[Descripción detallada del requisito no funcional 3.]
	RNF-004	[Descripción detallada del requisito no funcional 4.]

Tipo de Requisito	Código	Descripción
Interfaces de Usuario [Describir las interfaces de usuario que serán implementados en el software. Esto incluye por ejemplo: formatos de la pantalla, página o esquemas de las ventanas, reportes, menús, etc.]	RNF-005	[Descripción detallada del requisito no funcional 5.]
	RNF-006	[Descripción detallada del requisito no funcional 6.]
Interfaces de Hardware [Definir cualquier interfase de hardware que será soportado por el software, incluyendo estructura lógica, direcciones físicas, etc.]	RNF-007	[Descripción detallada del requisito no funcional 7.]
	RNF-008	[Descripción detallada del requisito no funcional 8.]
Interfaces de Software [Especificar el uso de otros productos software requeridos e interfaces con otros sistemas de la aplicación.]	RNF-009	[Descripción detallada del requisito no funcional 9.]
	RNF-010	[Descripción detallada del requisito no funcional 10.]
Interfaces de Comunicaciones [Describir las interfaces de comunicación para otros sistemas ó dispositivos, tales como: redes de área local, dispositivos de serie remota.]	RNF-011	[Descripción detallada del requisito no funcional 11.]
	RNF-012	[Descripción detallada del requisito no funcional 12.]

Tipo de Requisito	Código	Descripción
Requerimientos de Licenciamiento [Identificar las licencias que se requieran para el desarrollo del sistema.]	RNF-013	[Descripción detallada del requisito no funcional 13.]
	RNF-014	[Descripción detallada del requisito no funcional 14.]
Seguridad [Describir como será controlada la seguridad del sistema.]	RNF-015	[Descripción detallada del requisito no funcional 15.]
	RNF-016	[Descripción detallada del requisito no funcional 16.]
Estándares aplicables [Especificar con qué estándares trabaja el sistema.]	RNF-017	[Descripción detallada del requisito no funcional 17.]
	RNF-018	[Descripción detallada del requisito no funcional 18.]
Requisitos del Sistema [Especificar los requerimientos de plataforma tecnológica necesarios para el diseño y el desarrollo del sistema.]	RNF-019	[Descripción detallada del requisito no funcional 19.]
	RNF-020	[Descripción detallada del requisito no funcional 20.]
Requisitos de Desempeño [Listar y especificar los requisitos de desempeño con los que debe trabajar el sistema. Ejemplo: Tiempo de respuesta en alguna consulta del sistema.]	RNF-021	[Descripción detallada del requisito no funcional 21.]

Tipo de Requisito	Código	Descripción
	RNF-022	[Descripción detallada del requisito no funcional 22.]

7. Modelo de Casos de Uso del Sistema

[En esta sección deberá desarrollar el modelo de sistema o modelo de requisitos. Para ello deberá indicar los actores de sistemas, la arquitectura de sistema (organizada en paquetes) y la relación de casos de uso por cada paquete. Cada Caso de Uso deberá ser identificado con un código único y correlativo. Ejemplo: CUS01.]

7.1. Lista de Actores de Sistema

[Listar a los actores de sistema.]

Actor del sistema	Descripción
Nombre del actor del sistema	Descripción del actor de sistema. En la descripción deberá indicar que participación tiene en el sistema

7.2. Diagrama de Actores del Sistema

[Incorpore el diagrama de actores del sistema.]

7.3. Arquitectura del Sistema – Diagrama de Paquetes

[Incorpore el diagrama de paquetes que representa la arquitectura modular del sistema. Cada Paquete deberá ser identificado con un código único y correlativo. Ejemplo: P01.]

7.4. Lista de Casos de Uso del Sistema por Paquete

[En esta sección deberá listar todos los casos de uso del sistema que se han identificado. Para hacerlo deberá tomar como referencia la organización del sistema de acuerdo al diagrama de paquetes del punto 7.3.]

Paquete: P01 – Nombre del Paquete

Caso de uso del sistema	Descripción
CUS01 – Nombre del Caso de Uso	Descripción del caso de uso. En la descripción deberá indicar las acciones que permitirá el caso de uso.

7.5. Diagrama de Casos de Uso por Paquete

[Incorpore el diagrama de casos del uso del sistema de acuerdo a los paquetes y la lista trabajada en el punto 7.4.]

Paquete: P01 – Nombre del Paquete

7.6. Priorización de los Casos de Uso del Sistema

7.6.1. Clasificación de los Casos de Uso del Sistema

[En esta sección deberá clasificar los casos de uso de sistema indicando si son principales, secundarios u opcionales.]

Nombre del caso de uso	Clasificación
CUS01 – Nombre del caso de uso	Primario
CUS02 – Nombre del caso de uso	Secundario
CUS03 – Nombre del caso de uso	Opcional

7.6.2. Ciclos de Desarrollo de los Casos de Uso del Sistema

[En esta sección deberá indicar en qué ciclo de desarrollo se trabajarán cada uno de los casos de uso del sistema.]

Ciclo de desarrollo	Nombre del caso de uso	Clasificación
Núcleo central o Ciclo 0	CUS01 – Nombre del caso de uso	Primario
Ciclo 1	CUS02 – Nombre del caso de uso	Secundario
	CUS03 – Nombre del caso de uso	Opcional

7.7. Matriz de Modelo de Negocio y Modelo de Sistema

[En esta sección deberá incluir una matriz en la que se pueda evidenciar la trazabilidad entre los procesos de negocio y las funciones del producto software.]

Caso del uso del negocio		Actividad a automatizar			Requerimiento funcional		Caso de uso del sistema		
Nº	Nombre	Nº	Nombre	Trabajador	Nº	Nombre	Nº	Nombre	Actor
1	Caso de Uso de Negocio (CUS01)	1	Actividad a ser automatizada	Trabajador de Negocio	1	Requisito Funcional (RF01)	1	Casos de Uso de Sistema (CUS01)	Actor de Sistema
		2	Actividad a ser automatizada	Trabajador de Negocio					
		3	Actividad a ser automatizada	Trabajador de Negocio					

7.8. Realización de los Casos de Uso del Sistema

7.8.1. Especificación de Alto Nivel

[En esta sección deberá incluir la especificación de alto nivel de los casos de uso del sistema. Asimismo deberá indicar que requisitos funcionales están asociados a cada caso de uso, tomando como referencia lo indicado en la matriz del punto 7.7.]

Caso de uso:	CUS01 – Nombre del Caso de Uso
Actor(es):	Nombre del actor
Propósito:	Indicar el propósito del caso de uso
Caso de uso asociado:	Indicar si existe algún caso de uso asociado. De no haber indicar No Aplica.
Resumen:	Describir brevemente el caso de uso. Para ello deberá indicar como empieza el caso de uso, que actividades desarrolla y como termina.
Clasificación	Indicar la clasificación del caso de uso
Requerimientos	Indicar el(los) códigos de requisitos funcionales asociados.

7.8.2. Especificación Expandida

[Por cada caso de uso de sistema especificado a un alto nivel deberá incluir la especificación expandida de casos de uso. Para ello deberá indicar el flujo básico y los flujos alternos e incorporará el prototipo con la inclusión de los controles. Deberá usar la plantilla que a continuación se detalla

CUS01 – Nombre del caso de Uso

1 Actores

1.1 Lista de actores

2 Propósito

Indicar el propósito

3 Breve Descripción

Reutilizar el resumen del punto 7.4

4 Flujo Básico de Eventos

1. Indicar el flujo básico de eventos
2. Es posible hacer referencia a las reglas de negocio.

5 Subflujos

Indicar los subflujos del flujo básico.

6 Flujos Alternativos

6.1 Nombre del subflujo

Breve descripción del suflujo.

1. Detalle del flujo alterno. Se pueden incluir reglas de negocio.

7 Precondiciones

7.1 Nombre de la precondición

Descripción de la precondición

7.2 Perfil de usuario

Indicar el perfil de usuario que interactúa con el caso de uso

8 Poscondiciones

8.1 Nombre de la poscondición

Descripción de la poscondición

9 Puntos de Extensión

Indicar si existen puntos de extensión.

10 Requerimientos Especiales

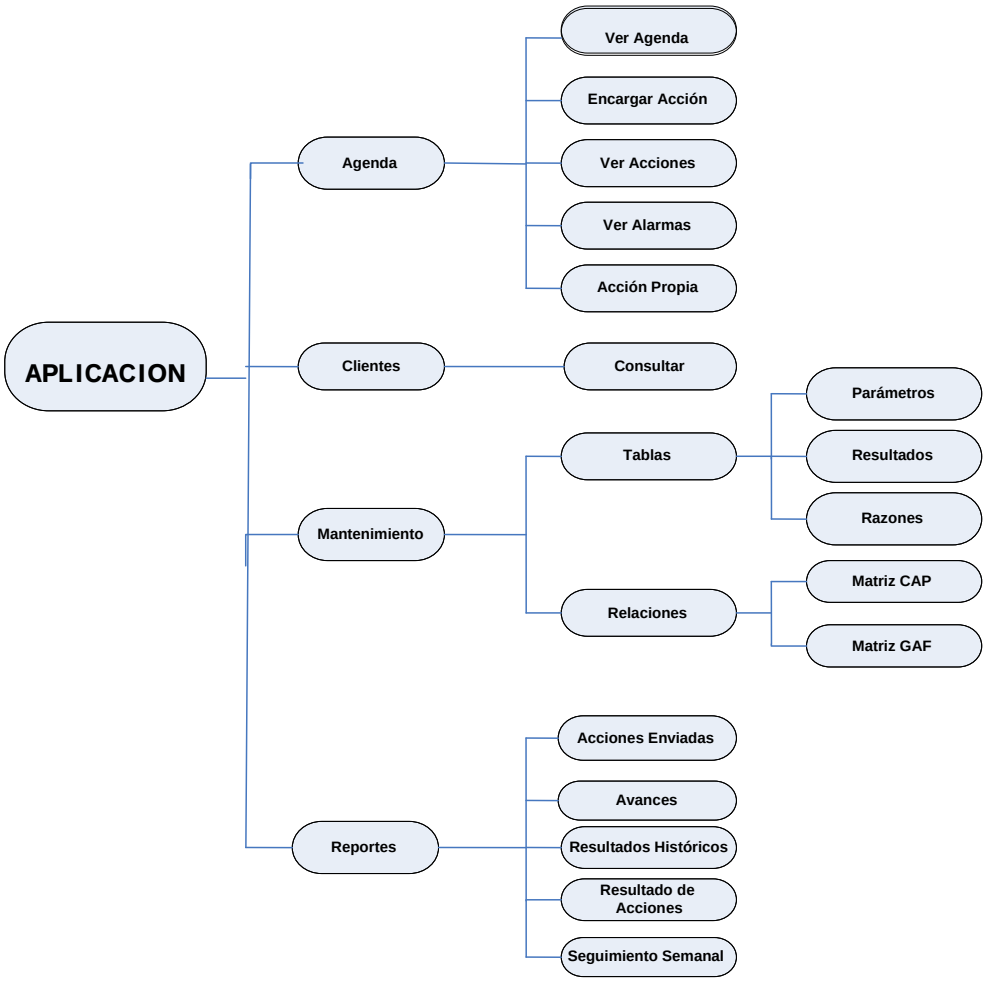
Indicar si existen requerimientos especiales.

11 Prototipos

Incluir los prototipos asociados al caso de uso

8. Flujo General de Navegación

[Incluir un árbol de navegación que permita entender el flujo que se seguirá en la navegación por el aplicativo. El siguiente ejemplo muestra un árbol de navegación:]



9. Esquema de Seguridad

[En esta se documenta los esquemas de seguridad en base a perfiles y su acceso a su información. Para ello se utiliza una matriz de perfiles de usuario y accesos por Aplicativo/Módulo/Función.]

Aplicativo				
Funciónes por Módulo	Perfil 1	Perfil 2	...	Perfil N
Módulo A	x	x	X	x
Consulta de información de empresas				
Consulta de operadores autorizados	x	x	X	x
Modificación de operadores autorizados	x	x	X	x
Módulo B				
Modificación de cuentas afiliadas	x	x	X	x
Modificación de combinaciones autorizadas	x	x	X	x

Anexo B
Reporte de Diseño de Software - RDS

Reporte de Diseño de Software (RDS)

[Nombre de Gerencia Sponsor del proyecto]

[Nombre del proyecto]

[Mes y Año Inicio del Proyecto]

[Este documento es la plantilla base para elaborar el documento Reporte de Diseño de Software. Los textos que aparecen entre paréntesis rectos son explicaciones de que debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda. En caso que alguna de las secciones del presente documento no aplique a su proyecto pueden usarse las frases "No hay cambios", "No hay impacto en esta sección", "La solución que se está implementando no tiene impacto en esta sección", "No aplican para el proyecto" (No borrar secciones del documento)]

Elaborado por:	Revisado por:	Aprobado por:
Fecha: / /	Fecha: / /	Fecha: / /

HISTORIAL DE REVISIONES

Versión	Autor	Descripción	Fecha de Elaboración	Fecha de Revisión	Revisado por
<x.x>	<Persona que elabora el documento>	<Detalles>	<Fecha de Elaboración>	<Fecha de Revisión>	<Persona(s) que revisa(n) el documento>

Contenido

1. Introducción

[Describa de manera breve el contenido del documento orientando la descripción hacia la utilidad que la misma busca. Recuerde que para la elaboración del documento debe considerar lo desarrollado en el Reporte de Especificación de Software (RES) y que es posible que en esta sección se pueda complementar la información del documento base RES.]

1.1. Propósito

[En esta sección se debe proporcionar una visión general de la arquitectura del sistema haciendo referencia a las diferentes vistas que implementarán los diferentes aspectos.]

1.2. Alcance

[Indicar cuál es el alcance del documento. Considere que en el RES se ha desarrollado la Vista de Sistema (funcional) y que para completar la visión total es necesario incluir la vista de arquitectura, vista lógica, vista de implementación, vista de despliegue y vista de datos. A menudo es necesario incluir una representación de la arquitectura con consideraciones de infraestructura tecnológica, relación con otros sistemas y una vista de procesos en donde se describe la descomposición del sistema en procesos y los métodos de comunicación del sistema.]

1.3. Definiciones, Acrónimos y Abreviaturas

[Proporcione las definiciones, acrónimos y abreviaturas que se utilizarán en el desarrollo del documento.]

1.3.1. Definiciones

[Indique cada una de las definiciones que son relevantes para el entendimiento del presente documento. Cada definición deberá ir acompañada de una breve descripción.]

Definición	Descripción
Asistente de Gestión	Trabajador encargado de procesar las rectificaciones de los ciudadanos que lo solicitan. También coordina las entregas de hologramas.

1.3.2. Acrónimos

[Indique cada una de los acrónimos que son relevantes para el entendimiento del presente documento, para el entendimiento de la arquitectura propuesta y para el diseño detallado. Cada acrónimo deberá ir acompañada de una breve descripción.]

Acrónimo	Descripción
RUP	Rational Unified Process

1.3.3. Abreviaturas

[Indique cada una de las abreviaturas que son relevantes para el entendimiento del presente documento, para el entendimiento de la arquitectura propuesta y para el diseño detallado. Cada abreviatura deberá ir acompañada de una breve descripción.]

Acrónimo	Descripción
SIGA	Sistema Integrado de Gestión Administrativa

1.4. Referencias

[Mencione los documentos que sirven como entrada, o salida, y herramientas que se usarán para el desarrollo del presente documento.]

2. Vista General de la Arquitectura

[En esta sección describa la arquitectura de software para el sistema actual, y cómo este será representado. De las vistas de casos de uso, lógica, procesos, despliegue e implementación; se enumerarán las vistas necesarias, y por cada vista, se deberá explicar qué tipo de elementos del modelo contienen.]

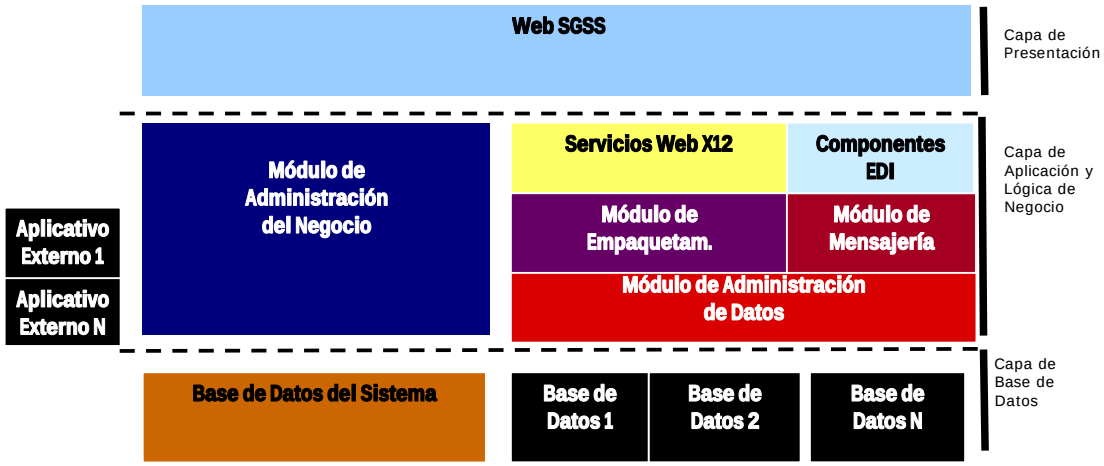
Ejemplo:

A continuación se muestra la Arquitectura del Sistema ABC, ésta está dividida en tres capas:

- Capa de Presentación
- Capa de Negocios
- Capa de Integración

Así mismo la aplicación por un criterio de funcionalidad se ha dividido en seis módulos:

- Módulo de Administración del Negocio
- Módulo de Empaquetamiento
- Módulo de Mensajería
- Módulo de Administración de Datos
- Servicios Web X12
- Componentes EDI



La figura anterior muestra la distribución de los módulos del software que tendrá el sistema, además de brindar una visión general del sistema. En el gráfico, se observa la distribución en tres capas de la arquitectura, las cuales se describen a continuación.

- **Capa de presentación**
  En esta capa se encuentra la aplicación Web dedicada a la administración y configuración DEL SISTEMA XYZ, la cual estará conformada por las pantallas de presentación al usuario. Estas aplicaciones Web serán páginas ASP.NET.

- **Capa de Lógica de Negocio**
  Esta capa provee todo lo que es la lógica del negocio, es decir la funcionalidad del sistema, ya sean los procesos administrativos, y los procesos referidos a la elegibilidad, crédito hospitalario (cartas de garantía) y crédito ambulatorio (pedidos de reembolso). Además, en esta capa se encuentran los servicios publicados, como los Web Services y los Servicios EDI sobre TCP/IP.

- **Capa de acceso a datos**
  Esta capa provee los servicios y conexiones a la base de datos requeridos por la capa de lógica de Negocio. Por otro lado, el manejador de base de datos utilizado para este sistema será Microsoft SQL Server 2000.

3. Metas y Restricciones de la Arquitectura

[En ésta sección se describe describen los requerimientos de software y objetivos que tienen algún significativo impacto sobre la arquitectura; por ejemplo: seguridad, privacidad de uso del producto, portabilidad, distribución y reuso. Esto también captura las restricciones especiales que quizás apliquen en la: Estrategia de diseño e implementación, herramientas de desarrollo, estructura del equipo, cronograma, leyes y regulaciones legales y otros. Las restricciones que aquí se recogen pueden complementar a las identificas en el RES a excepción de aquellas funcionales.]

4. Vista de Casos de Uso

[En ésta sección se listan los casos de uso o escenarios del modelo de casos de uso. Debe tomar como referencia lo desarrollado en el RES en los puntos punto 7.3, 7.4 y 7.5. Recuerde que debe respetar la codificación indicada en el RES para los paquetes y para los casos de uso. Si fuse necesario ampliar en esta sección recuerde que es necesario se actualice el RES.]

5. Vista Lógica

[La vista lógica está representada por los diagrama de clases del sistema donde se muestran sus relaciones estructurales y de herencia. La definición de clase incluye definiciones para atributos y operaciones. El modelo de casos del punto 3 del presente documento uso aporta información para establecer las clases, objetos, atributos y operaciones.]

5.1. Realización de Casos de Uso – Modelo de Análisis

[Esta sección ilustra cómo el software trabaja a partir de los casos de uso o escenarios seleccionados, y explica cómo varios elementos del modelo de análisis contribuyen con ellos funcionalmente. Por cada caso de uso deberá desarrollar un diagrama de secuencia y de clases de análisis. Para ello deberá usar el patrón MVC. Para la realización deberá identificar los escenarios. Dichos escenarios se obtienen de las combinaciones entre el flujo principal y flujos alternativos del la especificación expandida de casos de uso (ver punto 7.8.2 del RES).]

5.1.1. Código del CUS – Nombre del CUS

Nombre del Escenario

[Identifica el escenario a ser realizado y una breve descripción. Se recomienda identificar con un código único a cada escenario. Por ejemplo ESC01]

Diagrama de Secuencia de Análisis

[Incluya el diagrama de secuencia de análisis en el cual se observe el uso del patrón MVC que implementa el escenario identificado.]

5.1.2. Diagrama de Clases de Análisis

[Incluya el diagrama de clases de análisis obtenido del conjunto de diagramas de secuencia que se implementan por cada escenario.]

5.2. Modelo Conceptual

[Esta sección ilustra cómo a partir de las clases del tipo entidad se pueden identificar una primera propuesta de modelo de persistencia. Para ello se utiliza un diagrama clases por cada paquete que forma parte de la arquitectura del sistema. Se puede hacer uso de tarjetas CRC para documentar las responsabilidades y colaboraciones de cada clase de persistencia identificada.]

5.3. Modelo Lógico

[El modelo lógico es el refinamiento del Modelo Conceptual. Aquí se reducen y/o aumentan clases y sólo quedan aquellas que van a ser diseñadas como tablas de la Base de Datos. El modelo lógico debe representarse con un diagrama de clases de acuerdo a la arquitectura propuesta. Tenga presente que para la transformación del modelo conceptual al modelo lógico se debe tener en cuenta:

- Pasar las reglas de negocio
- Colocar las multiplicidades entre clases
- Identificar los atributos de Enlace o Clase de Enlace de las asociaciones de muchos a muchos
- NO INCLUIR los atributos identificadores de la clase (se agregarán en el modelo físico)
- Incluir los atributos de las clases que se necesitan para satisfacer los requerimientos del sistema
- Documentar un registro de glosario de términos
- Verificar que las reglas de negocio se sigan cumpliendo.

Se sugiere que por cada clase se tenga un diccionario que incluya el nombre, el tipo, la descripción, atributos, tipo de dato, visibilidad y valor inicial]

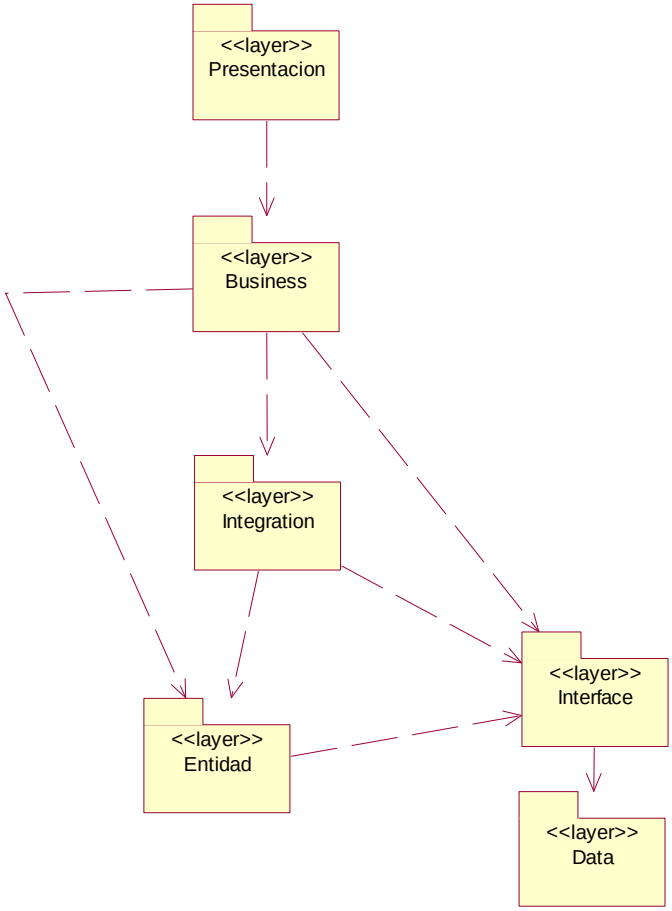
Nombre	Nombre de la Clase		
Tipo	Tipo de Clase (Ejemplo Entidad)		
Descripción	Descripción de la clase identificando que representa		
Atributo	Tipo de Dato	Visibilidad	Valor inicial
Nombre del atributo	Integer / String / Boolean	Público / Privado	

5.4. Modelo de Diseño

[En esta sección debe representar el refinamiento del modelo de análisis considerando los requisitos no funcionales identificados en el RES.]

5.4.1. Vista de Capas y Subsistemas

[Incluir el diagrama en el que se represente la arquitectura de diseño. Para ello puede usar un patrón en el cual se usen capas y subsistemas. Además deberá identificar subsistemas requeridos por el uso de algún patrón de diseño como el DAO Factory, Singleton, Front Controller, entre otros. Por cada capa y subsistema deberá identificar las clases de diseño que se implementarán]



5.4.1.1. Capa de Presentación

[Identifique las clases de diseño de la capa de presentación. Ordene dicha identificación utilizando los paquetes al interior de las capas denominados subsistemas.]

5.4.1.2. Capa de Negocio

[Identifique las clases de diseño de la capa de negocio. Ordene dicha identificación utilizando los paquetes al interior de las capas denominados subsistemas.]

5.4.1.3. Capa de Integración

[Identifique las clases de diseño de la capa de integración. Ordene dicha identificación utilizando los paquetes al interior de las capas denominados subsistemas.]

5.4.1.4. Capa de Datos

[Identifique las clases de diseño de la capa de datos. Ordene dicha identificación utilizando los paquetes al interior de las capas denominados subsistemas.]

5.4.1.5. Capa de Entidad

[Identifique las clases de diseño de la capa de entidad. Ordene dicha identificación utilizando los paquetes al interior de las capas denominados subsistemas.]

5.4.1.6. Capa de Interfaces o Elementos Comunes

[Identifique las clases de diseño de la capa de elementos comunes. Ordene dicha identificación utilizando los paquetes al interior de las capas denominados subsistemas.]

5.4.2. Realización de Casos de Uso – Modelo de Diseño

[Esta sección deberá desarrollar los diagramas de secuencia y de clases de diseño a partir de los requisitos no funcionales identificados en el RES y considerando los escenarios identificados en el punto 4.1 del presente documento. Debe asegurarse que las clases que se incorporen deben ser aquellas que se han identificado en el punto 4.4.1 del presente documento.]

5.4.2.1. Código del CUS – Nombre del CUS

[A partir de los casos de uso realizados del modelo de análisis deberá identificar los casos de uso que usará para las realizaciones de diseño.]

Nombre del Escenario

[Identifica el escenario a ser realizado y una breve descripción. Se recomienda identificar con un código único a cada escenario. Por ejemplo ESC01. Deberá reusar los escenarios identificados en el modelo de análisis]

Diagrama de Secuencia de Diseño

[Incluya el diagrama de secuencia de diseño en el cual se observe el uso de patrones de diseño para las clases que implementarán cada una de las clases lógicas.]

5.4.2.2. Diagrama de Clases de Diseño

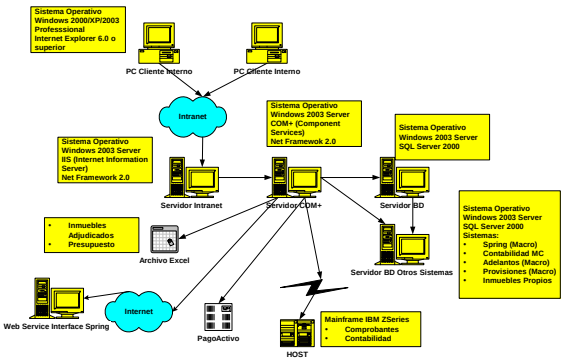
[Incluya el diagrama de clases de diseño obtenido del conjunto de diagramas de secuencia que se implementan por cada escenario.]

6. Vista de Procesos

[Esta sección describe la descomposición del sistema en procesos de primer nivel (un solo hilo de control) y los procesos de último nivel (grupos de procesos de primer nivel). También describe la ubicación de objetos y clases. Organizar la sección por los grupos de los procesos que se comunican u obran recíprocamente. Describir los modos principales de la comunicación entre los procesos, tales como el paso de mensajes, interrupciones y qué pasa, las interrupciones, y puntos de encuentro entre procesos.]

7. Vista de Despliegue

[En esta sección se describen unas o más configuraciones físicas de la red (hardware) que se usarán para el despliegue de los componentes de software que forman parte de la solución. Para ello puede usar un Diagrama de Despliegue indicando como mínimo, para cada configuración, en qué nodos físicos (computadoras, CPU) se ejecuta el software y sus interconexiones (bus, LAN, punto a punto, y así sucesivamente). De ser posible se debe incluir un mapeo de los procesos de la vista de procesos sobre los nodos físicos. Además deberá especificar los detalles técnicos de cada nodo en la vista de despliegue.]

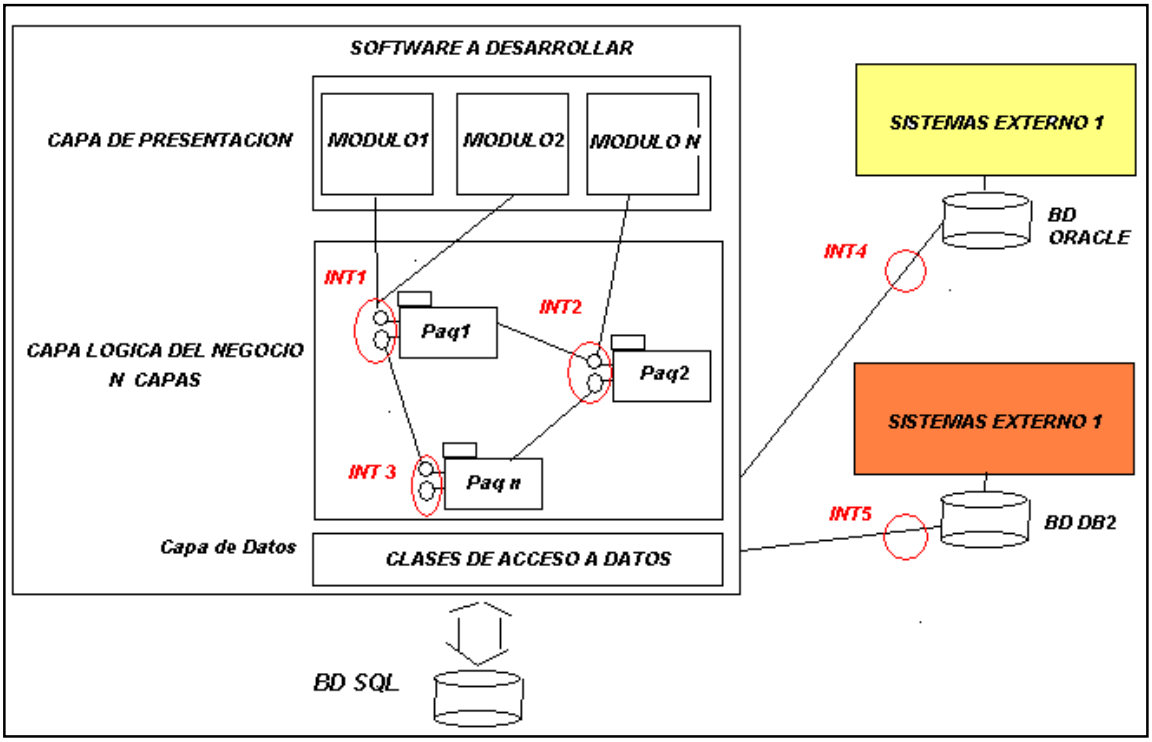


8. Vista de Implementación

[En esta sección se describe la estructura total del modelo de implementación, utilizando la descomposición del software en capas y subsistemas y cómo éste se pondrá en práctica. Deberá identificar cualquier componente arquitectónico significativo. Debe nombrar y definir las capas y contenidos de las mismas, las reglas que gobiernan la inclusión de una u otra capa, y las características entre capas. Incluya el diagrama de componentes que muestra las relaciones entre capas. Para cada capa, incluya una sub-sección con el nombre de la capa, una enumeración de los subsistemas localizados dentro de la capa y un diagrama de componentes.]

9. Vista de Integración del Software

[De requerirlo en esta sección pude incluir un diagrama integración del software desarrollado y su interacción con las diferentes interfaces identificadas en el modelo de diseño.]



INTERFAZ	DESCRIPCION BREVE	TIPO DE INTERFAZ	REFERENCIA
INT1	La interfaz 1 apoya la integración del Paquete 1 y el Paquete 2, incluye las clases C1, C2, etc….	Interfaz Interna	La Especificación de esta interfaz se encuentra en el documento de Especificación de Componentes
INT2	La interfaz 1 apoya la integración del Paquete 1 y el Paquete 2, incluye las clases C1, C2, etc….	Interfaz Interna	La Especificación de esta interfaz se encuentra en el documento de Especificación de Componentes
INT3		Interfaz Interna	La Especificación de esta interfaz se encuentra en el documento de Especificación de Componentes
INT4		Interfaz Externa	La Especificación de esta interfaz se encuentra en el documento de Especificación de Componentes
INT5		Interfaz Externa	La Especificación de esta interfaz se encuentra en el documento de Especificación de Componentes

9.1. Criterios de Integración de Software

[Identifique los criterios que se deberán considerar para la integración de los componentes de software y sus interfaces.

Ejemplo:

Para la óptima integración del Software se deberán tener que cumplir, considerar y evaluar los siguientes criterios:

- Antes de realizar la integración todos los componentes deberán haber pasado por pruebas unitarias.
- Antes de realizar la integración, todas las incidencias, errores u otras no conformidades encontradas durante las pruebas unitarias deberán estar cerradas.
- Se deberá tener preparado los ambientes y entornos para la integración (Entorno de Desarrollo o Entorno de Integración).
- Deberá haberse inicializado y migrado data consistente previa a la integración.
- Otros Criterios que apoyen a que la integración resulte un éxito.]

9.2. Secuencia de Integración

[Defina la secuencia de integración que se aplicarán a los componentes de software y sus interfaces.

Ejemplo:

Para que el Software se integre totalmente se seguirá la siguiente secuencia de integración:

- Realizar las pruebas unitarias a todos los componentes desarrollados (De todos los módulos).
- Levantar todos los errores e incidencias encontradas en las pruebas unitarias (De todos los módulos).
- Realizar revisión de pares al código fuente y levantar las no conformidades.
- Asegurarse que todos los componentes del Sistema estén completamente corregidos (Realización de nuevas pruebas sobre los errores encontrados).
- Validar que el entorno de integración este listo.
- Validar que la data haya sido migrada satisfactoriamente.
- Iniciar la integración
    - Integrar Modulo 1 y Modulo 2 - Realizar pruebas de integración entre ambos módulos.
    - Integrar Modulo 1 y Modulo 2 y Modulo3 - Realizar pruebas de integración entre módulos.
    - Integrar Modulo 1 y Modulo 2 y Modulo n - Realizar pruebas de integración entre módulos.
- Finalizada la Integración entre módulos, realizar la integración con aplicativos externos al sistema en desarrollo.
    - Integrar Sistema en desarrollo con Sistema Externo1 (Aplicativo Externo) y Realizar Pruebas.
    - Integrar Sistema en desarrollo con Sistema Externo2 (Aplicativo Externo) y Realizar Pruebas.
- Finalmente realizar las pruebas del Sistema y luego de ellas las Pruebas de Aceptación con los Usuarios Finales.]

9.3. Entorno Necesario para la Integración

[En esta sección se deberán identificar y especificar los diversos entornos que se usarán o que están involucrados en la integración del Software.]

NOMBRE DEL SERVIDOR	Serv_Desa
IP	1.1.15.50

DESCRIPCION Y OBJETIVO DEL SERVIDOR

En este servidor se almacenará el código fuente, en este entorno trabajaran los desarrolladores. Aquí se realizarán las pruebas unitarias.

SERVICIOS

NOMBRE DE SERVICIO	APLICACIÓN	FUNCIÓN	INICIO	USUARIO
Por Ejemplo: Asynchronous JavaScript + XML	Por ejemplo: AJAX	Por ejemplo: Presentación basada en estándares usando XHTML y CSS	Automático	Adminservice
<Servicio 1>	<Aplicación 1>	<Función 1>	Automático	Local System Account
<Servicio 2>	<Aplicación 2>	<Función 2>	Automático	Local System Account
<Servicio N>	<Aplicación N>	<Función 1>	Automático	Local System Account

CONFIGURACIÓN DE HARDWARE Y SOFTWARE

Nombre del Sistema Operativo	Microsoft (R) Windows (R) Server 200.Enterprise Edition
Version	2.2.3790 Service Pack 2 Build 3790
Proveedor del Sistema Operativo	Microsoft Corporation
Nombre del Sistema	DEIPSBATCH
Proveedor del Sistema	IBM
Modelo del Sistema	-[865811Y]-
Tipo del Sistema	X86 – based PC
Procesador	x86 Family 6 Model 8 Stepping 3 Genuineintel -664
BIOS Version/Date	IBM ILKT44AUS, 20/09/2001
SMBIOS Version	2.1
Total de Memoria Física	2,047.49 MB
Promedio de Memoria Física	1.37 GB

Total de Memoria Virtual	3.86 GB
Promedio de Memoria Virtual	3.47 GB
Tipo de Adaptador	Ethernet 802.3
Tipo de Producto	IBM Netfinity Fault Tolerante PCI Adapter
Nombre del Servicio	PCNet5
Dirección IP	10.203.32.9
Máscara de Sub Red IP	255.255.255.0
Gateway IP	10.203.32.254
DHCP Enabled	No
DHCP Server	Not Available
MAC Address	00:06:29:D5:38:0F
Memory Address	0XFEB7FC00-0XFEB7FC1F
SOFTWARE ADICIONAL	
USARIOS CON PERMISOS AL SERVIDOR	
RELACION CON OTROS SERVIDORES	

10. Vista de Datos

[Se debe describir la perspectiva persistente del almacenaje de datos del sistema. Deberá incluir el Modelo de Base de Datos Lógico y Físico, así como el diccionario de datos.]

11. Tamaño y Desempeño

[En esta sección se pueden incluir descripciones de las principales características del dimensionamiento del software que afectan la arquitectura, así como las restricciones de desempeño. Si trabajo estas características en el RES haga referencia a dicho documento.]