Universidad Tecnológica Nacional
Facultad Regional Buenos Aires

Tesis para el Magíster Ingeniería de Sistemas de Información

MODELO DE PROTOCOLO PARA ANÁLISIS Y EVALUACIÓN DEL SOFTWARE DESARROLLADO EN LAS PYMES ARGENTINAS

Martín Alejo VALIENTE

Directora Dra. Zulma CATALDI
Co-Director Mg. Germán KRAUS

2008

Índice

Resumen

La función que tiene la ingeniería de software es producir programas de alta calidad, para eso se deben aplicar los métodos más efectivos y las herramientas más modernas en el proceso de desarrollo. En otras palabras, lo que va a buscar crear son tecnologías que hagan más sencillo, rápido y menos costoso la construcción de sistemas de computación de alta calidad.

El objetivo de esta tesis se orienta al estudio de dos aspectos que son las metodologías de desarrollo con las herramientas asociadas y las normativas de calidad, teniendo en cuenta especialmente la repercusión sobre los recursos humanos ante los posibles cambios en los modelos de proceso.

Con datos tomados de las Pymes acerca de sus necesidades se propuso un modelo de protocolo que es aplicable al proceso de desarrollo de software para las Pymes. El mismo consiste en un cuestionario que permite un seguimiento del cumplimiento de las actividades necesarias en la construcción de programas dado que se busca que los productos sean más competitivos al ser más confiables que el seguimiento sistemático hace posible. Para su validación se consultó a especialistas en el tema.

Abstract

Dedicatoria

Un agradecimiento especial a mi familia y mis directores de tesis.

Quisiera hacer una dedicatoria a quienes confiaron y me apoyaron siempre.

1.1. Justificación

La tecnología que comprende el proceso de construcción se basa en diversas herramientas metodológicas[1] posibles de utilizar en ingeniería de software, con base en diferentes paradigmas de desarrollo o modelos de proceso. Estas herramientas evolucionan hacia el desarrollo de productos de software cada vez más confiables y de menor costo. [Pressman, 2005, Sommerville, 2002; Pfleeger, 2002]. Como el objetivo de la ingeniería de software es producir productos de alta calidad, se deben aplicar los métodos más efectivos y las herramientas más modernas en el proceso de desarrollo. En este sentido, se cuenta con diferentes normativas para asegurar y certificar la calidad tales como la serie ISO 9000 o el CMM (Capability Madurity Model) y sus derivados. [Senlle, 2000, Kaplan, 1995; Jenner, 1995]

El interés de esta tesis se centrará en el estudio de estos dos aspectos: las metodologías de desarrollo con las herramientas asociadas y las normativas de calidad, teniendo en cuenta la repercusión sobre los recursos humanos ante los posibles cambios de paradigmas.

Por otra parte, se debe destacar que no se han detectado investigaciones publicadas que aborden la temática desde está óptica y la escasa información a la que se tiene acceso es parcial y poco estructurada.

Las pequeñas y medianas empresas[2] (Pymes) argentinas tienen el desafío de aumentar la competitividad en la industria del software y los sistemas informáticos (SSI) de manera de ampliar el mercado interno y las exportaciones. Para eso hay que estudiar y considerar las ventajas comparativas con que cuenta el país de manera de generar una política que se pueda desarrollar con continuidad.

En vista de ello y de las oportunidades que se pueden lograr el Ministerio de Economía y Producción a través de la Secretaría de Industria, Comercio y de la Pequeña y Mediana Empresa desarrolló el Plan Estratégico de SSI 2004-2014 y el Plan de Acción 2004-2007. Este trata nueve grupos temáticos los cuales son: Recursos Humanos, Investigación y desarrollo, Observatorio de oferta y demanda, Exportaciones, Calidad, Financiamiento e inversiones, Propiedad intelectual y software libre, El Estado y el desarrollo del software, y El software embebido y la industria electrónica. Para eso se reunieron El Estado Nacional, el sector privado y el académico de manera de encargarse de realizar el análisis y desarrollo para generar los objetivos generales y la forma de llevarlos a la práctica.

Se puede decir que en los últimos años se han producido un conjunto de transformaciones económicas y sociales que están vinculadas con la sociedad del conocimiento. Las tecnologías de la información y las comunicaciones (TICs) y están en el centro de dichas transformaciones. Esto es así tanto en los países desarrollados como en los que están en desarrollo.

[1] Por ejemplo, las CASE (Ingeniería de software asistida por computadora), para conseguir buena calidad y productividad.
[2] En el ámbito de la Subsecretaría de la Pequeña Empresa y Desarrollo Regional se llama Pyme, de acuerdo a la norma vigente definida por la Resolución 675/2002, Resolución 22 y 24 / 2001 y el Artículo 33 de la Ley 19.550 es por el monto de ventas. Anteriormente se regía por el número de empleados, metros cuadrados y otras variables.

Se pueden mencionar ciertos factores que hacen atractiva la expansión de la industria de SSI para países como el nuestro y son los siguientes:

- El uso de SSI contribuye a tener mejores indicadores económicos, además de mostrar mejores impactos a nivel social y sobre el funcionamiento de las instituciones. La producción local puede potenciar estas ventajas a través del desarrollo de soluciones nuevas adaptadas a las necesidades locales y más baratas.
- Las exportaciones vienen creciendo en los últimos años y es posible que se continúe con esta tendencia.
- La industria informática tiende a pagar mejores salarios que los promedio, genera más empleo de alta calificación.
- Brinda al resto de la industria valores positivos al basar su desarrollo en la capacidad de innovación y la formación de capital humano.

Entre los países que concentran la mayor producción y exportación de bienes software están los Estados Unidos, Japón y las naciones más avanzadas de Europa. Le siguen en una posición consolidada Israel, India e Irlanda y buscan posicionarse mostrando algún éxito Rusia, China y Filipinas. Además hay que tener en cuenta que varias naciones (como EEUU) terciarizan la provisión de servicios informáticos, lo que genera una expansión del mercado internacional de outsourcing.

Un estudio que se menciona en el Plan Estratégico de SSI 2004-2014 indicaba que en los países consumidores del mercado TI estaba primero en la lista los Estados Unidos seguido por los países desarrollados. Después se puede ver que le siguen en Latinoamérica a Brasil y México, y en Asia a China, Corea, Taiwán, India y Hong Kong. La Argentina estaba en el puesto 26 del ránking mundial con un 0,3% del mercado mundial de TI y servicios informáticos, y un 0,2% en software en 2001.

Si bien, históricamente el hardware ha sido el impulsor de la tecnología de computación (y aunque se cree que seguirá evolucionando rápidamente, proporcionando mayor capacidad de cálculo y de almacenamiento), hoy día se considera que las tendencias futuras en computación se verán impulsadas principalmente por las tecnologías del software, donde la actual ingeniería de software brinda un marco de trabajo basado en métodos modernos y herramientas que le son propias.

Una indagación acerca de la situación actual de las Pymes en cuanto al paradigma de desarrollo que las mismas han adoptado y el nivel de calidad de los productos permitirá visualizar el estadío en que se hallan respecto de las tendencias mundiales. Un buen nivel de calidad a través del uso de herramientas metodológicas adecuadas, permitirá un desarrollo del sector productivo de software, a través de la obtención de mejoras competitivas con el consecuente desarrollo económico y social que estas mejoras conllevan.

Se debe tener en cuenta además, que los cambios en la tecnología informática son cada vez mayores y se producen muy velozmente, por lo que las consecuencias de la falta de actualización repercuten directamente sobre la competitividad de la empresa al punto de provocar su desaparición del mercado. En este sentido las Pymes se ven obligadas a cambiar y a adaptarse a

las nuevas tendencias, para sobrevivir, lo que exige a los profesionales una actualización constante a los efectos de incorporar rápidamente las novedades tecnológicas.

De los párrafos anteriores se puede resumir que: *la importancia de la incorporación rápida de las nuevas tendencias en cuanto a la concreción de desarrollos con calidad, es una idea que avanza exigiendo la aceptación de cambios en los recursos humanos que finalmente redundarán en ventajas competitivas en los mercados tanto nacional como internacional.*

1.2. Objetivos del trabajo de tesis

1.2.1. Objetivos Generales

El objetivo general es:

Determinar cuáles son los paradigmas de diseño y las metodologías consecuentes de la ingeniería de software que las Pymes aplican actualmente, el nivel de calidad en que trabajan y cuál es la actitud de los recursos humanos frente a los posibles cambios en la forma de desarrollo de los sistemas.

1.2.2. Objetivos Específicos

El objetivo general se desglosa en los siguientes objetivos específicos:

- Describir cuáles son las nuevas herramientas cuyo uso prevé la ingeniería de software[3] y que permitirían obtener ventajas competitivas de mercado.

- Describir las normativas de calidad existentes para software, a nivel mundial y en cuáles podrían centrarse las Pymes argentinas.

- Efectuar un estudio de tipo exploratorio y descriptivo acerca del estado actual de las Pymes tomando como ejes básicos de indagación los paradigmas de desarrollo y las herramientas metodológicas asociadas, la calidad del producto y la actitud de los recursos humanos ante los posibles cambios.

- Evaluar cuáles de los nuevos paradigmas con las metodologías asociadas se podría incorporar a los procesos de desarrollo de las Pymes argentinas a fin de lograr la menor resistencia al cambio en los equipos de desarrollo para obtener un producto confiable y exportable.

1.3. Metodología de trabajo

[3] Tales como modelos de procesos evolutivos, la reutilización de componentes y el uso de prototipos.

En el marco situacional descripto surge como *hipótesis* que la aplicación de las metodologías más modernas en los desarrollos de software[4] y el nivel de calidad obtenido a través de certificaciones, sumados a un equipo de trabajo flexible, permitiría a las Pymes argentinas acceder a la posibilidad de ganar gradualmente mercados internacionales.

Etapa 1: Durante la misma se llevará a cabo la búsqueda documental a través del relevamiento en libros de textos, revistas especializadas, eventos nacionales e internacionales y centros de investigación accesibles por Internet. Se indagará acerca del estado actual de la ingeniería de software en las Pymes Argentinas, respecto de las metodologías de desarrollo, las herramientas, y su relación con la calidad del producto. En este sentido, la investigación documental se conducirá a fin de considerar la inserción local dentro del marco de tendencias actuales a nivel mundial.

Por otra parte, se buscarán estudios previos acerca de cómo reaccionan los miembros de la organización ante los cambios de paradigmas. Luego, se efectuará el análisis del material recogido que conducirá a la confección del Estado del Arte.

Etapa 2: Una vez determinado el estado actual respecto de las metodologías en uso, se procederá a determinar cuales podrían ser las herramientas, los métodos y las técnicas más adecuados de introducir desde el punto de vista de obtener un nivel de calidad que permita obtener un producto exportable. Para ello, se tendrán en cuenta las tendencias actuales a nivel mundial, ya descriptas en el estado del arte.

Etapa 3: Una vez delimitada la investigación en las etapas anteriores, se procederá a encuestar y a entrevistar a profesionales del área de sistemas, para lo cual se utilizarán las técnicas de entrevistas de tipo estructurada y de encuestas (con preguntas cerradas, abiertas y de opción múltiple) a muestras significativas respecto de la población de Pymes productoras de software[5], con el fin de indagar acerca del nivel de calidad con que el software se desarrolla, que metodologías se aplican y cuáles de última generación estarían dispuestos a utilizar. O sea, aquellas que produzcan menos resistencias en los equipos de trabajo relacionadas con la mejor calidad a obtener.
Se determinará la validez y la confiabilidad del instrumento para la toma de datos, mediante el uso de los métodos más apropiados. La encuesta se distribuirá a través de correo electrónico,

[4] Respecto de las tendencias mundiales

[5] Como se sabe que la población es de unas 400 Pymes según Calvi [2004], para obtener el tamaño de la muestra a relevar, de manera que en el 80% de los casos se obtenga un error estándar menor al 0,035 (3,5%), partiendo de la formula que presenta Hernández Sampieri [2001] se tiene: $n' = \frac{S^2}{\sigma^2}$; $S^2 = p(1-p) = 0{,}8(1-0{,}8) = 0{,}16$;

$$V = 0{,}035 \Rightarrow V^2 = (0{,}035)^2 = 0{,}001225 \quad n' = \frac{S^2}{\sigma^2} = \frac{0{,}16}{0{,}001225} = 131$$

Efectuando la corrección, ya que se tiene el dato de la población N, se tendrá: $n = \frac{n'}{1 + n'/N} = \frac{131}{1 + 131/400} = 99$ Pymes..

Análogamente, trabajando con un error mayor el tamaño de la muestra bajaría.

postal y personalmente. En caso se de ser necesario se utilizarán los servicios de una encuestadora.
Las encuestas se procesarán a través de la categorización de los resultados obtenidos para las preguntas abiertas y mediante el análisis estadístico para las preguntas cerradas y de opción múltiple usando algunos de los paquetes estadísticos disponible en el mercado.
A través del registro de las entrevistas, y la posterior desgrabación de las mismas, se llevará a cabo el análisis cualitativo, usando algunos de los software disponibles para ello.
El hito de esta etapa lo constituye la información estructurada que permitirá visualizar las tendencias, es decir hacia donde se encamina la ingeniería de software, a fin de poder capturar las experiencias en el tema.

Etapa 4: Se elaborarán las conclusiones y las repercusiones del trabajo de tesis en el ámbito académico, generador de investigaciones y los aportes para el ámbito empresarial como receptor de las investigaciones. Será necesario relacionar la información adquirida cotejándola con no sólo con la teoría, sino con las investigaciones similares llevadas a cabo en otros países (ámbitos), elaborándose conclusiones que puedan ser relevantes para los ingenieros de sistemas, y en software.

Etapa 5: En esta etapa se elaborarán las conclusiones, se establecerán los aportes y las futuras líneas de investigación que se pueden derivar de este estudio. Finalmente, se confeccionará el informe final de tesis.

1.4. Visión global de la tesis

Si bien el software comenzó a usarse durante la Segunda Guerra Mundial, era en sus principios una actividad artesanal. Por este motivo, en el simposio de Roma en 1969, debido principalmente a la influencia de la NATO (Organización del Tratado del Atlántico Norte) que era el principal comprador, se requirió que el producto fuera más *confiable*. Desde entonces, empezó a usarse el término *ingeniería de software*. Hoy en día, casi 40 años después, la práctica de esta técnica ha mejorado notablemente. Sin embargo, todavía se pide mayor control de los costos del proyecto, de los tiempos que insume y se pone énfasis en que el producto sea de *calidad*. Para poder cumplir con estas metas se han desarrollado normas de calidad y de seguridad que certifican distintos organismos.

La importancia del tema se puede resumir en la frase siguiente:
"El impacto del software en nuestra cultura y en la sociedad continúa siendo profundo. Al mismo tiempo que crece su importancia, la comunidad del software, trata continuamente de desarrollar tecnologías que hagan más sencillo, rápido y menos costosa la construcción de programas de computadora de alta calidad" [Pressman, 2005].

Hoy día, cualquier Pyme puede pedir una certificación específica como por ejemplo ISO-IEC 12207[6] y de esta manera podrá acceder a mercados que la requieren como condición para poder

[6] El ISO/IEC 12207 es el estándar referente a los Procesos del ciclo de vida del software.

comerciar. El modelo CMM, (modelo de capacidad de madurez) desarrollado por el SEI (Software Engineering Institute), como otra opción, permite evaluar a través de cinco etapas definidas; por lo tanto una pequeña empresa podría centrarse solamente en algunas de ellas, por ejemplo hasta nivel 3, siendo de esta manera accesible a las economías de algunas organizaciones[7]. Otra elección es la ISO/IEC 15504 SPICE Trials que es una norma que podría ser utilizada por una Pyme de la que se puede conseguir ayuda para su implementación a través del IRAM (Instituto Argentino de Racionalización de Materiales).

Aparece luego, el MMCGP (Modelo de madurez de la capacidad de gestión de personal) compañero del CMM que fue desarrollado para aumentar la preparación de las organizaciones en las cada vez más complicadas aplicaciones, para ayudar a atraer, motivar, desplegar y retener el talento con el fin de mejorar su capacidad de desarrollo de software. El modelo de gestión de personal define cinco áreas claves para el personal que realiza programas: rendimiento, entrenamiento, retribución, desarrollo de la carrera, diseño de la organización y del trabajo y desarrollo cultural y de espíritu de equipo [Pressman, 2005].

Será de interés en este trabajo, efectuar consecuentemente, un estudio acerca de las dificultades que produce el cambio de paradigmas en los equipos de desarrollo de software y evaluar cuál es la posibilidad de incorporar las nuevas metodologías a los procesos de desarrollo en la Pymes con la menor resistencia al cambio por parte del personal empleado. En este sentido, la motivación puesta de manifiesto por los gestores de proyectos de cualquier sistema es de suma importancia para que el personal trabaje eficientemente con los recursos disponibles: dicho personal que participa del proyecto, la manera en que se organizan los equipos de software y los aspectos sobre la coordinación y la comunicación ya que son temas que deben ser evaluados y valorados.

Para la toma de decisiones, las empresas necesitan información básica, la cual debe ser completa y debe estar disponible a tiempo. Cada área de la organización debe tener acceso a la información que le concierne y a su vez debe interactuar con las restantes. También se debe considerar la vinculación con los clientes y los proveedores donde la tecnología de comunicaciones facilita las relaciones acercando a las partes. Este esquema, plantea una diversidad de problemas que el área de sistemas debe resolver, para lo cual deberá formar personal altamente calificado en el uso de tecnologías modernas que ayuden a trabajar con los datos de las empresas con eficiencia, además de incorporar nuevas herramientas que sean las más apropiadas para alcanzar el objetivo propuesto.

Es importante que el área de sistemas mantenga una visión global acerca del propósito de la organización o del negocio de la misma. De esta forma podrá colaborar eficientemente con la estrategia de la Pyme. Los procesos que trabajan lentamente o que generan información redundante, generalmente causan malestar en el personal involucrado con la empresa. Es por ello, que el punto de partida a través de un buen análisis y desarrollo posterior de los ingenieros de

[7] EL CMM tiene 5 niveles de madurez que se van superando en cuanto la organización va "madurando", proceso que se determina mediante evaluaciones. Desde la óptica de los desarrolladores permite evaluar las capacidades e implementar las mejoras continuas y desde el punto de vista de los clientes permite identificar las debilidades de los proveedores.

software, permitirá un funcionamiento eficiente del sistema que satisfaga las necesidades del usuario.

El Plan Estratégico de SSI 2004-2014 menciona que el nivel de empleo en el sector SSI está en el orden de los 40000 empleados en Argentina lo que le da una especial importancia. Y cuando lo compara con la industria automotriz dice que en los últimos 20 años la cifra máxima de ocupación supero apenas las 26000 personas y que en 2002 empleaba a poco más de 12000 personas.

También destacaba que en una encuesta realizada en el 2002 mostraba que más de un 60% de las empresas de SSI esperaban que las exportaciones desempeñaran un papel muy importante. Las firmas locales han dirigido su esfuerzo exportador fundamentalmente hacia países latinoamericanos y en menor medida hacia España. Aunque Estados Unidos ha sido también un destino para nuestros productos.

En cuanto a la posibilidad de efectuar exportaciones se busca la asociación de Pymes de manera de facilitar las transacciones para que resulten más económicas. Por otra parte, se espera que a partir del Gobierno surjan políticas de promoción para la creación y el fortalecimiento de las empresas productoras de software asistiendo en el asesoramiento técnico y financiero para la implementación y elaboración de proyectos, y con capacitación del personal.

En base a las experiencias de los países de ingreso tardío al sector SSI se pueden considerar cinco alternativas de inserción las cuales son:

- Exportación de productos
- Exportación de servicios
- Creación de productos para el mercado interno
- Provisión de servicios para el mercado interno
- Una estrategia combinada y evolutiva de la oferta de productos y/o servicios para el mercado interno y luego para le penetración del mercado internacional.

2.1. Las Pymes Argentinas productoras de software

2.1.1. Introducción

Las pequeñas y medianas empresas (Pymes) argentinas enfrentan en la actualidad una situación muy delicada, fundamentalmente después de los cambios que se produjeron en la política económica de los últimos años. Esta misma coyuntura en que se encuentran, parece no favorecer a las nuevas inversiones; ya que prácticamente no hay créditos. Sin embargo, para la producción de software, en particular, es posible ser optimista debido a que se abren algunas posibilidades interesantes.

Por un lado, en el mercado nacional algunos sistemas como los legales y contables para el uso local son un producto específico que los profesionales argentinos pueden realizar con la ventaja de una asistencia técnica inmediata. Esta necesidad se ha puesto de manifiesto a través de la inflación y la pesificación asimétrica que obligaron a actualizar algunos de esos programas en forma rápida.

Por otro lado, la posibilidad de contar con profesionales capacitados para realizar sistemas de calidad, sumada al parque computacional de última generación[8] que se incorporó en los últimos años, permite considerar a la Argentina como un país que se pueda perfilar como productor de software con fines de exportación en un futuro inmediato. Se puede incluir en este análisis a los ingenieros que como única herramienta cuentan con su computadora y como materia prima su inteligencia, que han creado pequeñas empresas que prometen tener un buen futuro, pensando en que la pesificación asimétrica podría ser una ventaja competitiva en el mercado internacional.

Según un pronóstico de Jorge Casino presidente de la CESSI (Cámara de Empresas de Software y Servicios Informáticos) [Calvi, 2004], se espera la situación mejore en los próximos años tanto para las Pymes que trabajan en Argentina como para las consultoras que prestan servicios en el área de los sistemas de información respecto de la incorporación de las nuevas herramientas a los procesos productivos. Esto se fundamenta en la importancia que tiene la información estratégica que obliga a utilizar las mejores herramientas en áreas como son el comportamiento organizacional, estrategia organizacional, tecnología de la información (TI), marketing, e-learning, e-business y bussiness intelligence.

La situación económica actual podría funcionar como impulsora de nuevos productos más que como freno para el desarrollo de negocios informáticos. Durante el año 2001 se exportaron entre 45 a 100 millones de dólares [Sametband, 2002]. Actualmente se considera que hay 70 productos

[8] Es importante considerar que para el 2003, según la consultora Prince & Cooke las ventas de computadoras en el mercado local serián de unas 420000 unidades, cifra que se acerca a la que brinda la consultora IDC[8] que estimaba serián 380000 máquinas. Para la consultora Gartner la venta de computadoras creció un 233% más que el último trimestre del año 2002 con casi 100000 unidades, mientras que las ventas de notebooks aumentó un 302% en este trimestre respecto del año 2002 [Sametband, 2003]. Para Alejandro Oliveros, director de Internacional Data Corp de Argentina (IDC), las ventas de computradoras personales ha crecido en el 2004 a unas 685033 PC. En el 2000 se vendieron 1 millón de equipos; en 2001, 750000 y en 2002 apenas 150000 [Torres, 2004]
Carlos Pallotti del CESSI mencionó que las exportaciones en el 2003 fueron de $2830 millones un 0.66 % del PBI y se esperaba que para el 2004 fueran de $ 3254 millones un 0.71 % del PBI. Esto es un 15 % más que el año anterior de facturación, un 10 % más de empleados y un 29 % más de exportaciones. Publicado como "El software y la informática esperan facturar 3254 millones de pesos" [Clarín, 22/02/2004]

tecnológicos exportables en nuestro país que hoy día podrían generar 10000 puestos de trabajo. Algunos pronósticos optimistas como el de Carlos Zárate, de Motorola, indican que en 10 años se podría tener el 3% del mercado mundial, siendo esta una meta muy ambiciosa pero factible, que permitiría conseguir un ingreso mayor que el del agro [La Nación, 2002].

Si se analiza el marco legal de Argentina relacionado, se verá que se busca favorecer los nuevos emprendimientos. El Gobierno Nacional, ya aprobó la Ley 25.856 que es la *"Ley de Consideración de la producción de software como actividad industrial"* y la Cámara de Senadores dio media sanción al proyecto de *"Ley de la Industria IT como actividad estratégica para el desarrollo nacional"* [Cessi, 2003]. Además se sancionó el 18 de agosto de 2004 y fue promulgada parcialmente el 7 de septiembre de 2004 la ley 25.922 que es la *"Ley de Promoción de la Industria del Software"*[9].

Las normas de calidad que se aceptarán a los beneficiarios de esta Ley en sistemas de gestión de la calidad son CMM, CMMI, IRAM-ISO 15504 (ISO/IEC 15504), IRAM-ISO 9001:2000 (con la Guía de Implementación IRAM 90003, ISO/IEC 90003) e IRAM 17601 (CMMI del SEI) y en calidad de software: ISO/IEC 9126 (IRAM-ISO/IEC 9126). La aplicación de estas exigencias los beneficia con el 70% sobre las contribuciones patronales y 60% de exención en el impuesto a las ganancias.
Existen otros proyectos de leyes con media sanción en la Cámara de Diputados, como lo son la *Ley de Delitos Informáticos y la Ley de Protección del Correo Electrónico.* Estas leyes se suman a las ya existentes en la materia[10] .

Por otra parte, el Artículo Nº 86 de la Ley Nº 11.723 (Régimen Legal de la Propiedad Intelectual), favoreció la creación del "Registro Nacional de Propiedad Intelectual". En la actualidad dicho organismo es la Dirección Nacional del Derecho de Autor (DNDA), dependiente del Ministerio de Justicia de la Nación, de manera que se pueden registrar legalmente obras de software y contratos de obras de software [Cessi, 2003].

Para el *comercio exterior*, el marco jurídico con el que se cuenta es la Resolución 856/1995 y para el *tratamiento arancelario específico* la Resolución 8/2001 y modificatorias y el Decreto 690/2002. Además dentro del marco propicio para la inversión tecnológica está incluido el proceso de desregulación total del mercado de las telecomunicaciones.

---

[9] Los capítulos que trataba el proyecto de esta última son:

Capítulo I	: Definición, ámbito de aplicaciones y alcances
Capítulo II	: Tratamiento fiscal para el sector
Capítulo III	: Importaciones
Capítulo IV	: Comisión Consultiva
Capítulo V	: Fondo de Promoción de la Industria del Software
Capítulo VI	: Infracciones y sanciones
Capítulo VII	: Incentivos a la formación d recursos humanos
Capítulo VIII	: Disposiciones Generales

[10] Algunas de las leyes y normas que figuran en nuestro sistema legal son: la Ley 11.723 de Régimen Legal de la Propiedad Intelectual vigente desde el 28 de septiembre de 1989, Ley 25.036 Ley de Propiedad Intelectual modificatoria de la Ley 11.723 vigente desde noviembre de 1998 (Dec. Reg. 2628/2002), Ley 24.766 Ley de Confidencialidad sobre Información y Productos vigente desde el 20 de diciembre de 1996, Ley 25.326 Ley de Protección de los Datos Personales (Habeas Data) vigente desde el 30 de octubre de 2000 y la Ley 25.506 de Firma Digital aprobada el 11 de diciembre de 2001. La Ley 25856 sancionada el 4 de diciembre de 2003 y promulgada el 6 de enero de 2004.

Se puede considerar en principio que a diferencia de otros bienes, el software puede ser exportado de tres formas distintas, cada una de ellas con su respectivo marco legal: a) Por vía carga aérea (con un permiso de exportación vía Aduana), b) por correo/courrier y c) por Internet.

El Decreto 1214/2005 (Ley Nº 22415 - Código Aduanero) modifica al Decreto 1001/82 con la finalidad que deberán reunir los exportadores e importadores para cumplir con los requisitos de solvencia económica ventas brutas por un importe no menor a los $300000 en el año anterior o a través de un patrimonio de igual monto; o sino, deberán constituir una garantía por un valor de $30000.

Muchos productos pueden ser enviados y recibidos puerta a puerta por el sistema de courrier. Como regla general, se puede decir que los productos trasladados por courrier deben ser de un valor menor a los $3000 y a un peso menor a los 50 kg. Como excepción la Aduana a los productos que requieren un estampillado (por ejemplo el caso del calzado, los textiles, los electrodomésticos y otros) los retiene y los toma como "carga". Antes de realizar una compra por Internet es necesario interiorizarse de los costos de seguro internacional, flete internacional e impuestos y tasas de nacionalización que deberán abonarse si se trasladan por courrier. Generalmente el courrier se ocupa de abonar las tasas de aduana e impuestos correspondientes, que después cobrará en un importe desglosado que incluye el precio neto del producto, el flete, los gastos e impuestos de aduana, la tasa de estadística y los impuestos al valor agregado.

En cuanto a la posibilidad de efectuar exportaciones se busca la asociación de Pymes de manera de facilitar las transacciones para que resulten más económicas. Por otra parte, se espera que a partir del Gobierno surjan políticas de promoción para la creación y el fortalecimiento de las empresas productoras de software asistiendo en el asesoramiento técnico y financiero para la implementación y elaboración de proyectos, y con capacitación del personal.

En este sentido, la CESSI en el año 2002 informó que al menos 255 empresas ofrecieron sus productos en el exterior, 125 de las cuales habían concretado negocios esporádicamente y 35 de manera frecuente. Por lo cual, a fines del 2002 unos 3000 profesionales se encontraban trabajando en áreas relacionadas con el mercado externo, cifra que se calcula en 5000 personas actualmente [CICOMRA, 2003]. Se estima la existencia de unas 400 Pymes productoras de software, tomando como fuente a Pablo Calvi [Calvi, 2004]. Según las cifras del Banco de la Provincia de Buenos Aires, las Pymes representan el 90 % de las empresas en todo el país, generando el 50 % del PBI y ocupando el 65 % de la mano de obra, si se consideran a las mismas como empresas de producción, de servicios e incluso el comercio.

Las Pymes deben evolucionar hacia una incorporación rápida de los cambios en las reglas de juego económicas, y para ello, deben ajustar su modelo de gestión hacia un aprendizaje rápido[11] *a fin de enfrentar dichos cambios para superar las debilidades estructurales de los modelos actuales. Esto sumado a una política de otorgamiento de préstamos y subsidios y otras facilidades permitirían las ventajas competitivas que le darían inserción en el mercado mundial.*

[11] Este aprendizaje rápido se logra, entre otras cosas, a través de personal altamente capacitado que incorpore rápidamente nuevas herramientas tecnológicas con la menor resistencia.[Guns, 1996]
[12] Tanto en los costos de insumos como también la mano de obra en término de dólares.
[13] La oferta de científicos, ingenieros y técnicos especializados en tecnología y soluciones informática a costos competitivos.
[14] Existen productos y precios ajustados a las condiciones de diferentes segmentos. Los precios están al alcance de muchas Pymes.

Entre las ventajas competitivas que encontramos en Argentina podemos enumerar los bajos costos de producción[12], el capital humano altamente calificado y con gran capacidad creativa[13], capacidad para exportar software de alto contenido innovador y demanda en el mercado interno[14]. Hay una gran oportunidad para la localización de inversiones orientadas a la producción y exportación de software y servicios informáticos para América Latina y el mundo hispano parlante. También se pueden considerar las nuevas posibilidades frente al uso de software de código abierto. [ADI, 2004]

Por otra parte, la importancia de la formación de recursos humanos es fundamental. Actualmente existen alrededor de 200 carreras de grado en el área de informática y sistemas (el 50 % son de corta duración 2 o 3 años). Existen aproximadamente 40 carreras de postgrado, mayoritariamente maestrías. Se estima que existen más de 20000 graduados en el Área Metropolitana de Buenos Aires (AMBA).

Se incorporó al Presupuesto Nacional 2003 el plan estratégico para el desarrollo de software y servicios informáticos (ARGENTEC). Que tiene por objeto el apoyo técnico en el proceso de certificación de software, la exportación de software y productos tecnológicos asociados de origen nacional y la promoción y capacitación en innovaciones tecnológicas, dirigida a la empresa.

En cuanto al contexto internacional se puede decir que el sector del software ha crecido un 100% entre 1995 y 2001, por encima de otros sectores de tecnologías de información. Se estimó que el mercado mundial de software en el 2001 representaba 196 mil millones de dólares. Siendo los principales exportadores Irlanda, Israel, India y USA. Los importadores más importantes son Reino Unido, Alemania, Japón, Canadá y USA. [ADI, 2004]

Actualmente se le está dando importancia a la seguridad informática, en parte debido a la conectividad que brindan las comunicaciones modernas y la necesidad de proteger la información. Esto ha producido una especialidad en el área de sistemas que ha sido acompañada por la generación de nuevas industrias que proveen productos cada vez más competitivos.

2.1.2. Seguridad Informática

La seguridad en la Ingeniería de Software es un tema muy amplio. Se ocupa de proteger la información que se encuentra en un sistema de computación, incluyendo el acceso a todos los recursos del sistema. La seguridad del software aplica los principios de la seguridad de información (Information security) al desarrollo del software. Generalmente se refiere a la modificación de los datos, si está en la fase de almacenamiento, procesamiento o tránsito.

También se ocupa de cuidar el acceso a la información contra la negación de servicios a usuarios autorizados y la no provisión de servicio a usuarios desautorizados, incluyendo las medidas necesarias para detectar, documentar, y contra actuar tales amenazas.

La seguridad en el área informática a crecido enormemente. En la Argentina se ha dado este fenómeno igual que en el resto del mundo. Si bien muchas Pymes cuentan con medidas de seguridad es de esperar que la aplicación de estándares de seguridad aumente significativamente en el futuro próximo. Por ejemplo se puede mencionar la Norma ISO 17799 de Seguridad de la Información[15] homologada por el IRAM.

El mercado de software y hardware específico para tareas de seguridad está proveyendo de tecnología cada vez más sofisticada. Debido a que continuamente se intenta sobrepasar las mismas, estas tecnologías son mejoradas constantemente. La posibilidad de proveer bienes y servicios en esta especialidad abre más posibilidades a la industria de nacional que cuenta con profesionales aptos para dicha tarea.

Los sistemas que son teóricamente seguros pueden ser inseguros en la práctica real. Además los sistemas son cada vez más complejos, esto proporciona más oportunidades para los ataques. Es mucho más fácil probar que un sistema es inseguro que demostrar que uno es seguro. Para probar la inseguridad, simplemente se toma ventaja de ciertas vulnerabilidades del sistema. Por otra parte, probar que un sistema es seguro, requiere demostrar que todas las opciones posibles de ataque puedan ser defendidas, siendo una muy desalentadora tarea cuando no imposible. Además la confiabilidad del software significa que un programa en particular debe seguir funcionando aún con la presencia de errores. Los errores pueden estar relacionados con el análisis, el diseño, la implementación, la programación, o el uso de la aplicación.

Las amenazas al sistema de cualquier organización pueden ser internas o externas. Los ataques internos, son más difíciles de detectar y repeler que los ataques externos. El atacante interno ya tiene acceso al sistema. Por cuestiones de seguridad y control las empresas monitorean las terminales y PC de los trabajadores. Como ejemplo el uso de Internet y e-mail mediante programas específicos. Según el informe de la American Management Association (AMA) unos 30 millones de empleados son espiados en todo el mundo durante su jornada laboral por medio se software de vigilancia (14 millones son Norteamericanos). También esto plantea un tema moral debido a que puede afectar la privacidad de la persona.

En cuanto a la seguridad informática se puede diferenciar la seguridad física (control de acceso físico, políticas de seguridad y normativas, plan de continuidad del negocio) y la seguridad lógica

[15] La misma está organizada en 10 capítulos y son:
1. Política de Seguridad
2. Organización de Seguridad
3. Clasificación y Control de Activos
4. Aspectos humanos de la seguridad
5. Seguridad Física y Ambiental
6. Gestión de Comunicaciones y Operaciones
7. Sistema de Control de Accesos
8. Desarrollo y Mantenimiento de Sistemas
9. Plan de Continuidad del Negocio
10. Cumplimiento.

(protección de la información en su propio ámbito como por ejemplo mediante el enmascaramiento y usando técnicas de criptografía).

El Plan de Continuidad del Negocio es el plan de contingencia que formula el comité de seguridad al realizar una política afín donde se define que se va a proteger. Durante el análisis se identifican los riesgos, se establecen medidas de seguridad apropiadas que puedan llegar a ser manejables. Hay que buscar un equilibrio entre la flexibilidad y la seguridad. Es importante pensar cuáles son los puntos débiles y a partir de ahí mejorar el sistema. Será necesario contar con recursos humanos capacitados y actualizados constantemente.

Hay tres principios básicos que se deben considerar. El del acceso más fácil; si bien hay varios frentes en los que se puede producir un ataque seguramente se hará en el punto más débil o donde sea más fácil el acceso (incluyendo el uso de cualquier artilugio para facilitarlo). El de la caducidad del secreto; que significa que hay un tiempo donde debe mantenerse la confidencialidad o el secreto del dato, esto se consigue fortaleciendo el cifrado. El de la eficiencia de las medidas tomadas; deben ser eficientes para que funcionen en el momento oportuno, fáciles de usar y que pasen desapercibidas por el usuario, y apropiadas al medio optimizando los recursos del sistema.

Un concepto importante es el de los datos seguros que se cumple si hay confidencialidad, integridad, disponibilidad, no repudio origen y destino.
Una solución para la comunicación que utilizan las empresas medianas es la red privada virtual o en inglés Virtual Private Netware (VPN), que es una red segura de datos al aplicar una caño virtual. Una aplicación posible es la conexión encriptada entre redes privadas usando la red pública como por ejemplo Internet.

2.1.3. Facilidades de apoyo económico a las Pymes Argentinas

El Banco Central de la República Argentina (B.C.R.A.) es quien habilita a las entidades financieras a otorgar préstamos y define las medidas para estimular el crédito.

La Agencia Nacional de Promoción Científica y Tecnológica dependiente del Ministerio de Educación, Ciencia y Tecnología se dedica a la promoción de actividades relacionadas a la ciencia, la tecnología y la innovación productiva. El apoyo económico lo hace a través de sus dos Fondos; los cuales son el Fondo para la Investigación Científica y Tecnológica (FONCyT) y el Fondo Tecnológico Argentino (FONTAR) que promueve el financiamiento de proyectos en el marco de la Modernización Tecnológica III.

Los beneficiarios de este Fondo pueden ser empresas productoras de bienes y servicios que facturen hasta $30000000. El límite que financian es de hasta $600000 que no podrá exceder el 50% del costo total del proyecto. Es interesante tener en cuenta que el sueldo máximo para las siguientes categorías es: junior $2809, semisenior $3785 y señor $4927.

Otra opción es la convocatoria abierta a emprendedores a través del Fondo Fiduciario de Promoción de la Industria del software (FONSOFT), previsto por la ley 25922, que puede destinar prestamos de hasta $100000 por proyecto.

Para poder financiar sus proyectos las Pymes deben acudir a inversionistas para que las apoyen. Por lo que es importante que las mismas cuenten con un plan de negocios de manera de vender sus ideas con una formalidad conocida. Por ese motivo es conveniente saber los factores que consideran algunos de ellos. Para resumir debería tener cuatro componentes: la declaración de la visión, las personas que gerencian el proyecto, perfil del negocio, y test económico.

Hay varios factores que son considerados por los inversores en las empresas de software como por ejemplo el equipo (Management Team), el tamaño del mercado porque es más probable que acepten invertir si es potencialmente amplio ya que tienen más chances de recuperar lo invertido y obtener ganancias, las conexiones personales siguen siendo un aspecto de importancia porque si alguien puede presentar a los inversores es un gran comienzo y al presentar la misión de la empresa conviene que sea específica es decir se debe ser concreto en qué va a hacer la misma. Las firmas de capital de riesgo además se fijan en las tendencias del sector y se focalizan en sectores específicos.

Además se tiene en cuenta el modelo de negocio (que puede incluir el resumen ejecutivo para captar el interés con los aspectos más importantes del negocio, la descripción del producto y el valor distintivo, ventaja competitiva del producto, la misión y los objetivos, oportunidad y mercado potencial, estrategia de desarrollo, requerimientos tecnológicos, integrantes del equipo y organización, evaluaciones preliminares económico-financieras, estrategia de marketing y comercialización, principales riesgos y estrategias de salida, etc.) que suele ser muy atractivo en la industria del software pero debe incluir una explicación de como se sustenta y los resultados a largo plazo que se esperan obtener.

Hay otros factores que también se estudian. El proyecto debe ser coherente con la realidad económica actual; o sea, debe tener sensibilidad económica (si tienen menos dinero para software también lo tendrán para ventas y marketing). La afinidad con los fondos, como los recursos son escasos es mejor si las inversiones tienen impacto en los resultados de un fondo de capital de riesgo. El plan financiero debe explicar cómo se hará para generar ganancias tanto para la Pyme como para los inversores. Reconocer la existencia de riesgos es importante pero también lo son más las consideraciones para sortear los mismos o solucionar las consecuencias.

2.1.4. Nuevas Oportunidades de Negocios

En Argentina se están generando cada vez más opciones de realizar negocios debido a que las nuevas tecnologías crean áreas donde aplicar la informática como por ejemplo con Internet y ASP, también existen mercados internos potencialmente significativos, la disponibilidad recursos humanos con un alto nivel multicultural lo favorece, la disponibilidad de profesionales muy capacitados y con capacidad creadora, la posibilidad para exportar software de alto contenido innovador, bajos costos de producción, demanda en el mercado interno y además las fuertes inversiones en el área de las telecomunicaciones.

Entre las oportunidades de inversión tenemos que analizar las ventajas competitivas del software en nuestro país además de las mencionadas anteriormente. Como es la de ser un polo de producción para el mercado de habla hispana. Un segmento importante de la oferta se beneficia directamente por la modernización empresarial en Argentina. En la medida que las empresas del sector manufacturero incursionan en mercados externos también se abren nuevas áreas de negocio para sus proveedores de software y servicios informáticos.

Entre estas nuevas oportunidades de negocios se puede mencionar:

- Software educativo y de entretenimiento (edutainment) para el mercado de habla hispana. [ADI, 2004]
- Software agropecuario con aplicaciones dirigidas tanto a productores como al sector público.
- Computación industrial (especialmente son desarrollos a medida).
- Aplicaciones médicas, en administración de servicios de salud y telemedicina.
- Oportunidades de sustitución de paquetes de software de firmas internacionales por desarrollos locales a costos menores.
- Establecimiento de áreas de desarrollo de firmas internacionales por desarrollos locales a costos menores.
- Posibilidad frente al uso de software de código abierto.

[ADI, 2004]

2.2. Las metodologías de la ingeniería de software

2.2.1. Introducción y definición de metodología

Para el desarrollo de un proyecto de software se necesita establecer un enfoque disciplinado y sistemático. Las metodologías de desarrollo influyen directamente en el proceso de construcción y se pueden elaborar a partir del marco definido por uno o más ciclos de vida. Según Piattini no hay un consenso entre los autores sobre el concepto de metodología, y por lo tanto no existe una definición universalmente aceptada. Pero sí hay un acuerdo en considerar a la metodología como un conjunto de pasos y procedimientos que se deben seguir para el desarrollo del software. [Piattini, 1996]

También se puede definir a la metodología como un conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentación y aspectos de formación para los desarrolladores de sistemas de información. [Maddison, 1983].

Por lo tanto, una metodología es un conjunto de componentes que especifican:

- Cómo se debe dividir un proyecto en etapas.
- Qué tareas se llevan a cabo en cada etapa.
- Qué salidas se producen y cuándo se deben producir.
- Qué restricciones se aplican.

– Qué herramientas se van a utilizar.
– Cómo se gestiona y controla un proyecto.

Generalizando, se puede llegar a la definición de metodología de desarrollo como "un conjunto de procedimientos, técnicas, herramientas, y un soporte documental que ayuda a los desarrolladores a realizar nuevo software". Normalmente consistirá en fases o etapas descompuestas en subfases, etapas, módulos, pasos, etc. Esta descomposición ayuda a los desarrolladores en la elección de las técnicas a utilizar en cada estado del proyecto, facilitando la planificación, gestión, control y evaluación de los proyectos. Para resumir se puede decir que una metodología va a representarse como el camino para desarrollar software de una manera sistemática. [Piattini, 1996]

Las metodologías persiguen cubrir tres necesidades principales:

– Mejores aplicaciones, tendientes a una mejor calidad, aunque a veces no es suficiente.
– Un proceso de desarrollo controlado, que asegure uso de recursos apropiados y de costo adecuado.
– Un proceso estándar en la organización, que no sienta los cambios del personal.

Las metodologías a veces tienen diferentes objetivos, pero los más representativos que se mencionan pueden ser:

– Brindar un método sistemático, de modo de controlar el progreso del desarrollo.
– Especificar los requerimientos de un software en forma apropiada.
– Construir productos bien documentados y de fácil mantenimiento.
– Ayudar a identificar las necesidades de cambio lo más pronto posible.
– Proporcionar un sistema ágil que satisfaga a todas las personas involucradas.

Los procesos se descomponen hasta el nivel de tareas o actividades elementales, donde cada tarea está identificada por un procedimiento que define la forma de llevarla a cabo. Para aplicar un procedimiento se pueden usar una o más técnicas. Estas pueden ser gráficas con apoyos textuales, formales y determinan el formato de los productos resultantes en la tarea. Para llevar a cabo las tareas se pueden usar herramientas software que automatizan la aplicación en determinado grado.

2.2.2. Evolución de las metodologías de desarrollo

Los primeros desarrollos de software no tuvieron una metodología definida, fueron totalmente artesanales y se los llamó desarrollos convencionales. Se caracterizaron por el aspecto monolítico de los programas y por una falta de control de lo que sucede en un proyecto. Estas limitaciones condujeron a una serie de problemas y por lo tanto a una búsqueda de hacer el desarrollo de forma sistemática.

Como una posibilidad de "nuevo orden", surge el desarrollo estructurado, sobre la base de la programación estructurada, los métodos de análisis y el diseño estructurado. Esta nueva etapa es la piedra fundamental para la construcción de programas con métodos ingenieriles. Aparece en

los sesenta en el ámbito científico y en los setenta pasa al ámbito empresarial. Tiene como punto de partida el establecimiento y uso de normas para la aplicación de estructuras de control y de datos.

En los setenta, el enfoque estructurado, se extiende de la fase de análisis a la fase de diseño y las técnicas estructuradas se dirigen tanto a los aspectos técnicos como los relacionados con la gestión en la construcción de software. El módulo del programa es el componente básico de la construcción del producto software, después venía la normalización de la estructura de los módulos de programa y finalmente el posterior refinamiento. En este período se comienzan a aplicar medidas de calidad en los programas.

Según Gane y Sarsons [Gane, 1977] y DeMarco [DeMarco, 1979], consideraron que uno de los problemas de los programas desarrollados monolíticamente era que se necesitaba una comprensión total de las especificaciones, por parte de los analistas, que en muchos casos eran ambiguas, y en otros eran obsoletas al llegar al final del proyecto.

La base de la programación y el diseño estructurados, es un análisis del problema usando el diseño top-down o descendente, con énfasis en las especificaciones funcionales. Se compone de diagramas, con textos de referencia de los mismos, y con independencia para que se puedan leer en forma parcial, con una redundancia mínima, de modo que los cambios no lo afecten en forma notable. También, se puede destacar, que ha habido una evolución en cuanto al modelado de sistemas en tiempo real, modelado de datos y estudio de eventos. [Piattini, 1996]

El paradigma orientado objetos, que aparece más tarde, trata a los procesos y los datos en forma conjunta, modulariza la información y el procesamiento. El Smalltalk, pone énfasis en la abstracción de datos, considerando a los problemas como un conjunto de objetos de datos a los que se les adicionaba un conjunto de operaciones, pero se fundamenta en la abstracción, la modularidad y el ocultamiento de la información, derivados del diseño estructurado. En general para los desarrollos de las metodologías orientadas al objeto se tomaron de los conceptos de técnicas estructuradas.

En los ochenta, aparecen el C++ y otros como el Lenguaje ADA del Departamento de Defensa de los Estados Unidos para la definición y mejora de tecnologías basada en este lenguaje.

2.2.3. Características de las metodologías

Los métodos vienen a proporcionar el "cómo" hacer la construcción de un producto en este caso del software. Estos abarcan una amplia diversidad de tareas que incluyen la comunicación, el análisis de requisitos, el modelado del diseño, la construcción del programa, la realización de pruebas y el soporte [Pressman, 2005].

Se pueden mencionar una serie de características que debe tener una metodología y que influirán en el entorno de desarrollo. Estas características pueden ser reglas predefinidas, determinación de los pasos del ciclo de vida, realizar verificaciones en cada etapa, la planificación y control del proyecto, la comunicación efectiva entre desarrolladores y usuarios, la flexibilidad y aplicación

en un amplio espectro de casos, y la fácil comprensión por quien las usa. La comunicación es uno de los aspectos más relevantes cuando se desarrolla un sistema de información, todo lo que permita motivar la buena disposición entre los miembros del equipo de desarrollo y promover el diálogo entre estos con los usuarios ayudará obtener un software de calidad.

El soporte de herramientas automatizadas es otra característica que debe ser tenida en cuenta porque otorga facilidades en la implementación de una metodología. Se puede mencionar la importancia de que los miembros del equipo tengan un verdadero conocimiento de la herramienta para que pueda ser amplia y correctamente usada.

Además se pueden señalar otras características tales como que se permita definir mediciones que indiquen mejoras, que se permitan modificaciones y que soporte la reusabilidad del software. El hecho del uso de métricas puede indicar áreas que requieran más atención, motivo por el cual la posibilidad de realizar modificaciones es necesaria para aprovechar esta ventaja. La reusabilidad del software está muy ligado con la calidad del producto por lo tanto el soporte dado se trasladará a los sistemas que se desarrollen.

2.2.4. Métodos convencionales para la ingeniería de software

Los sistemas de información comprenden varios componentes como son el hardware, el software, las personas, las bases de datos, la documentación y los procedimientos. La ingeniería de sistemas comienza tomando una visión global del problema a resolver, donde se analiza el dominio del producto o negocio con el fin de establecer los requisitos básicos. Después el enfoque se estrecha hasta llegar a una visión de cuando se analizan los elementos individualmente.

Los conceptos y principios del análisis estructurado son aplicados por la ingeniería de requisitos; que a través del uso de técnicas y herramientas hace un estudio, refinación y especificación del problema. Se puede decir que el análisis cumple con cinco áreas que son: el reconocimiento del problema, la evaluación y síntesis, el modelado, la especificación y revisión. Para obtener como resultado un correcto análisis de la especificación de los requisitos, esta debe ser revisada por las partes para asegurarse que el cliente y el analista tengan el mismo concepto del sistema. Igualmente hay que tener en cuenta que el problema puede seguir cambiando en el transcurso del tiempo.

Los elementos que se usan en el análisis buscan hacer un modelo para representar los requisitos de datos, funciones y comportamientos con diagramas y textos. Estos son: los diagramas de entidad-relaciones que describen los objetos de datos, los diagramas de flujos de datos que hacen una especificación del proceso (indican cómo se transforman los datos y representan las funciones que transforman los flujos de datos), los diagramas de transición de estados que hacen la especificación de control y el diccionario de datos que describe los elementos de datos. Las especificaciones del proceso y de control proporcionan una especificación adicional de los detalles. Hay varias herramientas CASE que pueden facilitar la creación y mantenimiento de estos diagramas [Pressman, 2005].

En la siguiente Tabla se presentan las diferencias más importantes de esta metodología en la fase de análisis estructurado enseñadas por Gane y Sarsons [Gane, 1977], DeMarco [DeMarco, 1979] y Yourdon [Yourdon 1989], las cuales tienen como base la utilización de un método descendente para la descomposición funcional del problema y se apoyan en técnicas gráficas de especificación estructurada.

Método de Gane y Sarsons	Método de DeMarco	Método de Jourdon
– Construir un modelo lógico actual. – Construir un modelo lógico del nuevo sistema. – Seleccionar un modelo lógico. – Crear un nuevo modelo físico del sistema. – Empaquetar la especificación.	– Construir un modelo físico actual. – Crear un conjunto de modelos alternativos. – Examinar los costos y tiempos de cada opción. – Empaquetar la especificación.	– Realizar los diagramas de flujo del sistema. – Realizar el diagrama de estructuras evaluar el diseño, midiendo la calidad cohesión y el acoplamiento. – Preparar el diseño para la implantación.

Tabla 1: Diferencias entre las metodologías convencionales de Gane y Sarsons, DeMarco y Yourdon

Los conceptos y principios del diseño estructurado buscan hacer un plano del sistema igual que cuando se va a construir una casa. La modularidad y la abstracción permiten al diseñador simplificar y reutilizar los componentes que elaboró; y el refinamiento le da un mecanismo para representar las capas y mejorar las hechas. El diseño estructurado se concentra en cuatro áreas: datos, arquitectura, interfaces y componentes del software. El diseño de la arquitectura del software nos da una visión del sistema porque describe la estructura y organización de los componentes del software, sus propiedades y conexiones. Cuando se diseña la interfaz de usuario se está creando la comunicación entre el hombre y la máquina, se tiene en cuenta al principio los requisitos del usuario, de la tarea y del entorno. Hay tres principios que conviene tener en consideración para hacer un diseño de interfaz eficaz: poner el control en manos del usuario, reducir la carga de memoria del usuario y construir una interfaz que considere estos puntos. El diseño de componentes o procedimental consiste en convertir el diseño de datos, interfaces y arquitectura en un software operacional, es una abstracción muy cercana al código fuente. Las notaciones usadas representan los detalles a nivel de componentes, con formatos gráficos, tabulares o basados en texto como son la tabla de decisiones y lenguaje de diseño de programas. [Pressman, 2005]

Cuando se lleva a cabo la implementación en el estructurado para hacer un buen software se busca que sea fácilmente legible, reutilizable (dividido en módulos), y que las condiciones y bucles cumplan con los requisitos del sistema como así también los indicados para una correcta práctica.

Cuando se hacen los casos de pruebas en el estructurado se busca obtener un el conjunto de pruebas que descubran la mayor cantidad de defectos. Para eso hay diferentes técnicas de diseño como son: prueba de caja blanca y prueba de caja negra. Las pruebas de caja blanca se basan en la estructura de control; o sea, aseguran que se han ejecutado todas las sentencias del programa o todos los caminos independientes dentro del módulo, que se han probado todas las condiciones lógicas, que se ejecuten todos lo bucles en sus límites y dentro de sus límites operacionales, y se hayan ejercitado las estructuras de datos internos de manera de asegurar su validez. Las prueba de caja negra validan los requisitos funcionales; los errores que busca encontrar son: funciones incorrectas o faltantes, errores de interfaz, errores en estructuras de datos o acceso a bases de datos externas, errores de comportamiento o desempeño, y errores de inicialización y

terminación. La verificación del software es cuando se evalúa si el software producido es un producto construido correctamente; en otras palabras, hace bien lo que se propuso hacer. La validación es cuando se evalúa si el software producido se ajusta a los requisitos del cliente. Para conseguir hacer un sistema de pruebas se recomienda una estrategia de pruebas del software como son las pruebas de unidad (que hace la verificación funcional de cada módulo de software), de integración (verifica la incorporación de los módulos en la estructura del programa), de validación (prueba el seguimiento de los requisitos) y del sistema (valida el software cuando se lo incorpora en un sistema superior). [Pressman, 2005]

El uso de las métricas en el estructurado son un elemento clave como en cualquier ingeniería, ya que proporcionan una manera cuantitativa de valorar la calidad de los atributos del producto. Las métricas que hay permiten una visión interna de los modelos que se crean durante el análisis (ej. métricas de Punto de Función y métricas bang), el diseño (ej. métricas del diseño arquitectónico y otras), la implementación (ej. Teoría de Halstead) y las pruebas (ej. Uso de las métricas de Punto de Función y B77ang). [Pressman, 2005]

2.2.5. Métodos de la ingeniería de software Orientada a Objetos

El concepto de las tecnologías de objetos es reflejar una visión natural del mundo, permiten encapsular funcionalidades para ser usadas. Los objetos son clasificados en clases que se organizan en jerarquías. Las clases tienen un conjunto de atributos que las describen y también las operaciones que definen su comportamiento. El encapsulamiento, la herencia y el polimorfismo son algunas ventajas que ofrece esta metodología. El análisis orientado a objetos (AOO) permite modelar un problema representando las características estáticas de las clases y sus relaciones. El lenguaje unificado de modelado, mejor conocido como UML, tiene las siguientes características: representación de las clases y jerarquías de clases, creación de modelos objeto-relación (establece las relaciones de clases y objetos) y obtención de modelos objeto-comportamiento (indica como responderá el sistema a sucesos externos). [Pressman, 2005]

El diseño orientado a objetos traduce el modelo del AOO al mundo real. Se pueden describir cuatro capas para el proceso. Hay una capa que es fundamental para el diseño de subsistemas que implementa las funciones principales del sistema, la capa de clases especifica la arquitectura de objetos global y la jerarquía de clases, la capa de mensajes indica como debe ser realizada la colaboración entre los objetos, y la capa de responsabilidades identifica las operaciones y atributos que caracterizan a cada clase.

La implementación o programación orientada a objetos convierte el diseño en un código de programa donde las clases son expresadas en un lenguaje orientado a objetos.

El objetivo de las pruebas con esta metodología también es la de encontrar el mayor número de errores con el mínimo esfuerzo. Las pruebas pueden empezar con la revisión del modelo de análisis y diseño orientado a objetos, después de generado el código las pruebas pasan a los objetos comenzando por lo pequeño. Se trabaja con el concepto de clase y es aplicable la modalidad de caja negra como prueba de validación.

Las métricas que se usan para los sistemas orientados a objetos se enfocan en la medición que puede aplicarse a cada clase y a las características de diseño (encapsulación, herencia, etc.). Hay muchas de ellas, pero una que se destaca es la serie de métricas CK (de Chidamber y Kemerer) que definen seis métricas de software orientadas a clase (métodos ponderados de clase, árbol de profundidad de herencia, número de descendientes, acoplamiento entre clases objeto, respuesta para una clase, y falta de cohesión en métodos) centrada en la clase y la jerarquía de clases. [Pressman, 2005]

2.2.5.1. RUP (Rational Unified Process)

Dentro de la metodología orientada a objeto el RUP es actualmente la más popular. Este proceso fue creado por el Rational Software, que es una división de IBM, es un marco de referencia y adaptable para desarrollar software con los estándares más altos.

El ciclo de vida que utiliza RUP es un prototipado que se compone de cuatro fases básicas:

- Fase de inicio.
- Fase de elaboración.
- Fase de construcción.
- Fase de transición.

A continuación se analiza en profundidad cuáles son las actividades que se realizan en cada una de las fases:

En la fase de inicio o concepción se desarrolla el concepto del sistema, se analiza el escenario en el cual se pondrá en funcionamiento el futuro sistema analizando los riesgos posibles del proyecto, también se define un plan para llevar a cabo el desarrollo y se asigna los diferentes recursos para cada una de las tareas. Además aquí se define la terminología que se va a utilizar, se identifican los posibles usuarios del sistema como así también de forma general los bloques fundamentales que constituirán el futuro sistema.

La fase de elaboración del proyecto es cuando comienza a tomar forma. Se analiza completamente el dominio de la futura aplicación y se define la arquitectura sobre la cual se montará el futuro sistema. Se definen los casos de uso, que son documentos que contienen los distintos pasos que debe seguir el sistema en cada una de las determinadas situaciones (como por ejemplo altas de usuarios, listados, etc.). Además los casos de uso son invocados por los actores, que son quienes interactuarán con el sistema. Se revisan los riesgos planteados en la fase de inicio y se actualizan en caso de ser necesario. Se generan especificaciones complementarias que contienen las restricciones no funcionales, como el consumo de ancho de banda, la disponibilidad requerida, los problemas en la conexión, etc.

Una vez comenzada esta fase de construcción, los cambios en la base del sistema se vuelven muy costosos, tanto en tiempo como en dinero, ya que volver atrás implica revisar todos los documentos y diagramas generados en la fase de elaboración, se analizan los nuevos riesgos que pueden surgir aparejados con las modificaciones a introducir y también el impacto de los agregados sobre los módulos ya existentes. En esta etapa se desarrolla el sistema, por medio de

módulos y/o prototipos y es en esta fase donde se generan los primeros programas ejecutables que se les presentan a los usuarios.

La fase de transición es importante porque los nuevos sistemas a desarrollar deben reemplazar a los actuales que se encuentran funcionando en las distintas organizaciones. Es en esta fase en la cual el software se transporta desde los servidores de desarrollo a los servidores de producción, donde quedarán funcionando. Esta transición no es trivial, ya que se requiere capacitar a los usuarios, al personal de sistema, a las mesas de ayuda y por otra parte, se terminan de validar los requerimientos generados en la fase de inicio buscando que se vean plasmados realmente en el sistema final.

2.2.6. Métodos de la Ingeniería de Software Orientados a la Web

Las aplicaciones Web (WebApps) están basadas en principios, conceptos, procesos y métodos de la ingeniería de software. Son diferentes de otras categorías de software por el intensivo uso de la red, por estar gobernadas por los datos y por la continua evolución que tienen las aplicaciones. La inmediatez es una característica común por la apremiante necesidad de poner software rápidamente en el mercado. Además se debe poner énfasis especialmente en la seguridad. La estética debe ser atractiva a los usuarios y la entrega del contenido funcional. También tiene otros atributos como la concurrencia de un gran número de usuarios al mismo tiempo, una carga impredecible ya que los usuarios la visitan en distintas cantidades para un tiempo comparable, el desempeño debe ser rápido para evitar que el usuario decida irse a otro lado y la evolución continua.

Se hace uso de un modelo de proceso interactivo e incremental porque el tiempo de vida de un desarrollo para la Web es corto.

Un modelo de proceso que sirva de marco de trabajo cumple con los siguientes puntos: comunicación con el cliente (análisis del negocio y formulación del proyecto), planeación (plan del proyecto), modelado (del análisis y del diseño), construcción (codificación y prueba), y despliegue (prueba de aceptación y evaluación del consumidor). Seguramente se aplica empleando un flujo incremental.

Al equipo que trabaja en proyectos Web se les debe dar autonomía y flexibilidad. Los lineamientos para la construcción de un equipo de trabajo exitoso incluyen establecer un conjunto de directrices del equipo donde se aclara que se espera de cada persona y como se lidiará con los problemas, un liderazgo que guíe mediante el ejemplo y el contacto, un especial respeto a los talentos individuales, un verdadero compromiso de cada miembro del equipo, y mantener el ímpetu que les permite superar problemas.

Un método que sirva de marco de trabajo puede incluir las siguientes etapas: la formulación (que identifica las necesidades del negocio, las metas y objetivos), la planificación (estima el costo global y evalúa los riesgos), el análisis (para establecer los requisitos técnicos e identificar los elementos que va a contener), la actividad de ingeniería (que incluye el diseño y la producción del contenido), la generación de páginas y pruebas, y finalmente la evaluación del cliente.

El modelado del análisis para una aplicación Web tiene como propósito describir las metas y objetivos, definir las categorías de usuarios, señalar los requisitos de contenido y función para la aplicación, y establecer porque se construirá, quién la usará y que problemas les resolverá al usuario. Hay cuatro actividades que permiten la creación de este modelo como son el análisis de contenido para identificar lo que proporcionará la aplicación, el análisis de interacción para describir como interactuar el usuario con la aplicación el análisis de funciones para definir las operaciones y el análisis de la configuración para describir el ambiente de la infraestructura.

Durante el diseño se introduce la calidad buscada definida como facilidad de uso, funcionalidad, confiabilidad, eficiencia, facilidad de mantenimiento, seguridad, escalabilidad y tiempo en el mercado. Los atributos que se valoran son la simplicidad, consistencia, identidad, robustez, navegabilidad y apariencia visual.

El fin de probar una aplicación Web es encontrar errores o descubrir conflictos que pudieran conducir a fallas en la calidad. Se centra en el contenido, función, estructura, facilidad de uso, navegabilidad, desempeño, compatibilidad, interoperabilidad, capacidad y seguridad. Las pruebas comienzan con examinando las unidades individuales y después que ejerciten la aplicación como un todo. El proceso de prueba abarca siete diferentes tipos de pruebas como son del contenido (con el fin de descubrir errores semánticos o sintácticos), de interfaz (valida los aspectos estéticos de la interfaz), de navegación (es para identificar los errores que impiden el completar un caso de uso), de componentes (se ejercitan y evalúan las unidades de contenido y funcionales), de configuración (busca descubrir los errores p problemas de compatibilidad), de seguridad (se diseñan para explorar vulnerabilidades) y de desempeño (para evaluar el tiempo de respuesta). [Pressman, 2005]

2.2.7. Nuevas tendencias metodológicas

El método ágil busca una entrega rápida del software que procure la satisfacción del cliente, la importancia de la organización propia de los equipos que tienden a ser pequeños en número y con alta motivación, la comunicación y colaboración tanto entre los miembros del equipo como entre los profesionales y los clientes, y un reconocimiento especial de que el cambio representa una oportunidad. En 2001 Kent Beck y otros desarrolladores firmaron el Manifiesto para el desarrollo ágil de software que establecía mejores formas de construirlo y por medio de ese trabajo valoraron:

- A los individuos y sus interacciones sobre los procesos y las herramientas
- Al software en funcionamiento sobre la documentación extensa
- A la colaboración del cliente sobre la negociación del contrato
- A la respuesta al cambio sobre el seguimiento de un plan. [Pressman, 2005].

La “programación extrema” conocida como XP (Extreme Programming) es el proceso ágil que más se utiliza. Es un enfoque formulado por Kent Beck quien escribió su libro en 1999. Surge motivado por los problemas del desarrollo de software como son los retrasos en la planificación, tasa de defectos, requisitos mal comprendidos, cambios de personal, etc. [Pressman, 2005] Está

orientado a la producción de código y sus pruebas; poniendo énfasis en la satisfacción del cliente y en promover una estrecha colaboración del equipo de desarrollo. Debido a sus principios presenta inconvenientes para ser utilizados en grandes proyectos. [Piattini, 2003, p.17]

La "programación extrema" cumple con un marco de trabajo de cuatro actividades que son: planeación, diseño, codificación y pruebas. La planeación es la descripción de las características y las funcionalidades requeridas para el producto software que se construirá, el diseño debe ser simple y es una guía de implementación para una historia, la codificación comienza con el desarrollo de pruebas de unidad de cada historia (incremento de software) y que después debe implementarse proponiéndose la programación en parejas y una estrategia de integración continua, y finalmente están las pruebas como las pruebas de unidad pueden crearse antes de la codificación y deben implementarse en un marco de trabajo que permita automatizarlas de tal forma que puedan ejecutarse de manera fácil y repetida o como las pruebas de aceptación que las especifica el cliente y se enfocan en las características generales y la funcionalidad del sistema. [Pressman, 2005]

Entre las principales características del método fundamentales podemos mencionar:

- Desarrollo iterativo e incremental para obtener pequeñas mejoras
- Pruebas unitarias continuas
- Frecuente interacción del equipo de programación con el cliente para mejor comunicación
- Corrección de todos los errores antes de escribir más funcionalidad
- Reciclado continuo del código para que el código reescrito sea más legible y mantenible
- Simplicidad en el código

El "desarrollo de software adaptativo" se basa en la colaboración de las personas y la organización propia del equipo. Su marco de trabajo cumple con tres actividades: especulación (inicio del proyecto y el ciclo de planeación), colaboración (se busca que la gente esté motivada para multiplicar los talentos y las salidas creativas además de que haya comunicación y confianza entre los miembro del equipo) y aprendizaje (que permita la satisfacción de las necesidades del negocio con grupos enfocados, revisiones técnicas formales, y después mejorar su desempeño y proceso de su aprendizaje).

El "método de desarrollo de sistemas dinámicos" proporciona un marco de trabajo para desarrollar y mantener sistemas con poco tiempo y mediante el empleo de prototipos incrementales en un ambiente de proyecto controlado. Comienza con las actividades de estudio de factibilidad (que establece los requisitos y evalúa si la aplicación es viable para este proceso) y estudio de negocios (que establece los requisitos funcionales y de información que le den valor al negocio, además de la arquitectura básica de la aplicación), después define tres ciclos iterativos: iteración de modelo funcional (produce una serie de prototipos incrementales que demuestren la funcionalidad), iteración de diseño y construcción (revisa la construcción de prototipos para asegurar que proporcionen un valor operativo para los usuarios finales), e implementación (que coloca el último incremento de software al ambiente operativo).

El "melé" propone que se use un conjunto de patrones de proceso de software que han probado su efectividad en proyectos con límites de tiempo muy ajustados, requisitos cambiantes y que son

críticos para el negocio. El sprint consiste en el euquipo de trabajo que se requiere para satisfacer un requisito definido en un período de tiempo definido con anterioridad el cual generalmente es de 30 días. Las reuniones en el melé son a diario y por un breve tiempo que por lo general es de 15 minutos, procuran ponerse al tanto de que progresos hubo desde la última reunión, que obstáculos nuevos encontró y que espera lograr para la próxima reunión.

El "cristal" es una familia de procesos de los métodos ágiles que pueden adaptarse a las características de un proyecto. Premia la manejabilidad planteándola como un juego de inventiva y comunicación con recursos limitados para cumplir con la meta primaria de entregar software útil y en funcionamiento, después como meta secundaria pasan a prepararse para el siguiente juego.

El "desarrollo conducido por características" enfoca al equipo de proyectos en el desarrollo de características, como son las funciones que se pueden implementar como mucho en un par de semanas y que las evalúa el cliente. Para eso definieron una plantilla para definir una característica y cinco actividades de de colaboración dentro del marco de trabajo (desarrollar un modelo general, elaborar una lista de características, plan por característica, diseño por característica y construcción por característica).

El "modelado ágil" propone que el modelado es esencial para el desarrollo de sistemas pero este modelo debe adaptarse al software que será construido. El modelado debe permitir entender mejor lo que se debe lograr, el problema se divida de la mejor manera para ser efectivo en lo que se debe resolver y la calidad se evalúe en cada paso a medida que el sistema se desarrolla. Se puede decir que es una guía útil para los profesionales tanto en el análisis como en el diseño. [Pressman, 2005]

2.2.8. Métrica Versión 3

La metodología Métrica v3 tiene por objeto ser útil para la sistematización de las actividades que dan soporte al ciclo de vida del software. Se la aplica tanto para procesos de desarrollo estructurado como también orientado a objetos. Cuenta con una guía que facilita a través de interfases descritas la realización de los procesos.

La Métrica v3 es una metodología que permite alcanzar los siguientes objetivos:

- Proporcionar o definir sistemas de información que ayuden a conseguir los objetivos de la organización mediante la definición de un marco estratégico.
- Proveer a la organización de productos software que satisfagan las necesidades de los usuarios dándole importancia al análisis de requisitos.
- Mejorar la productividad de los departamentos de sistemas y tecnologías de la información y las comunicaciones.
- Facilitar la comunicación y entendimiento entre los distintos participantes en la producción de software a lo largo del ciclo de vida del proyecto.
- Facilitar la operación, mantenimiento y uso de los productos software obtenidos.

Siguiendo el modelo se puede distinguir los procesos principales como son la planificación, el desarrollo y el mantenimiento. [Piattini, 2004]

La metodología Métrica v2.1 fue creada por el personal de varios organismos de administración de manera de constituir una guía sencilla y útil para la planificación, análisis, diseño, construcción e implantación de sistemas de información. Para cada tarea se detallan los participantes que intervienen, los productos de entrada y salida, y las técnicas y prácticas.

Para la construcción de esta metodología Métrica v3 se ha tenido en cuenta las normas ISO 9000:2000, ISO 12207, ISO/IEC 15504 (SPICE), estándar IEEE y otras más. Se ha aplicado el enfoque orientado al proceso que se centra en el ciclo de vida del software. La metodología descompone cada uno de los procesos en actividades y estas a su vez en tareas. Para cada tarea se describe su contenido como son por ejemplo sus principales acciones, productos, técnicas, prácticas y participantes.

Los procesos de la estructura principal son los siguientes:

- Planificación de Sistemas de Información
- Desarrollo de Sistemas de Información
- Mantenimiento de Sistemas de Información

El Plan de Sistemas de Información (PSI) tiene por objeto proporcionar un marco estratégico de referencia para los sistemas de información en un ámbito específico de la organización. Busca orientar las acciones de desarrollo de sistemas colaborando con la estrategia corporativa, elaborando una arquitectura de información y un plan de proyectos de manera de dar apoyo a los objetivos estratégicos. Los productos del PSI son el catálogo de requisitos y la arquitectura de información. El catálogo de requisitos va a surgir del estudio de la situación actual, del diagnóstico que se realizó y de las necesidades de información de la organización afectada por el plan de sistemas. La arquitectura de información se compone a su vez en los siguientes productos: modelo de información, modelo de sistemas de información, arquitectura tecnológica, plan de proyectos y plan de mantenimiento del PSI.

En cuanto al proceso de Desarrollo de Sistemas de Información contiene todas las actividades y tareas para desarrollar un sistema y se lo ha subdividido en cinco procesos para mayor claridad los cuales son: el Estudio de viabilidad del Sistema (EVS) que analiza un conjunto concreto de necesidades para proponer una solución que puede estar relacionada con aspectos económicos, técnicos, legales y operativos. Es un proceso obligatorio pero el nivel de profundidad depende de cada caso. Para valorar las alternativas que fueron planteadas y llegar a una solución se estudia el impacto en la organización de cada propuesta, el contexto del sistema, el costo/beneficio de cada solución, la valoración de los riesgos, etc.; el Análisis de Sistemas de Información (ASI) busca conseguir la especificación del sistema de información a través de un catálogo de requisitos y una serie de modelos que obtienen las necesidades de información, también delimita el alcance y las restricciones, además adquiere los requisitos funcionales y no funcionales del sistema. Se puede facilitar el análisis de sistemas identificando los subsistemas y elaborando los modelos de Casos de Uso y de Clases para el desarrollo orientado a objetos, o los modelos de Datos y Procesos para los desarrollos estructurados; el Diseño del Sistemas de Información (DSI) que define la arquitectura del sistema y del entorno tecnológico como así también las especificaciones de los

componentes del sistema de información; la Construcción del Sistemas de Información (CSI) es la programación y prueba (unitaria y de integración) de los componentes del sistema de información. También se preparan los procedimientos de operación, administración del sistema y seguridad; los manuales y las especificaciones para la formación de los usuarios; el código fuente de los componentes de migración y los procedimientos de migración; y la carga inicial de datos; y la Implantación y Aceptación del Sistema (IAS) es la entrega y aceptación del sistema de información, y la implantación en su totalidad que consiste en procesar las actividades para el paso a producción del sistema.

El mantenimiento de sistemas de información tiene como objetivo la obtención de una nueva versión de un desarrollo que puede ser justificado por una nueva necesidad o por un problema detectado. Los tipos de mantenimiento pueden ser correctivo o evolutivo, excluyendo se los adaptativos y perfectivos como pueden ser los de migración y la retirada del software. El responsable del mantenimiento debe encarar la definición de la solución que incluirá el estudio del impacto que tendrá, la valoración del esfuerzo y costo, las actividades y tareas a desarrollar, y el plan de pruebas de regresión.

Las Interfaces en la Métrica V3 definen una serie de actividades organizativas o de soporte para el proceso de desarrollo y a los productos. La aplicación de estas medidas es para proporcionar sistemas con calidad y seguridad. Estas interfaces son Gestión de Proyectos, Seguridad, Aseguramiento de la Calidad y Gestión de la Configuración.

La Gestión de Proyectos (GP) tiene como objetivo la planificación, el seguimiento y el control de las actividades, de los recursos humanos y de los otros elementos del desarrollo de los sistemas de información. Al controlar el desarrollo se conocen los problemas y esto permite resolverlos o minimizar el efecto evitando desviaciones del lineamiento marcado. Las actividades de esta interfaz son: de inicio del proyecto (para estimar el esfuerzo y establecer la planificación del proyecto), de seguimiento y control (supervisa las tareas y gestiona el proyecto en todo momento), de finalización del proyecto (registra la documentación de gestión y cierre).

La Seguridad (SEG) es el análisis de riesgos naturales (inundaciones, incendios, etc.) y lógicos (ataques externos, fallos del sistema, etc.) como así también establecer los mecanismos que aseguren el desarrollo. Las valoraciones de la seguridad corresponde hacerlas en función de las características del sistema.

El Aseguramiento de la Calidad (CAL) proporciona un marco de referencia para verificar la calidad de un producto en un proyecto concreto. Estas actividades son realizadas por el grupo de aseguramiento de la calidad trabajando de manera independiente de los desarrolladores.

La Gestión de la Configuración (GC) es la aplicación de los procedimientos técnicos y administrativos durante el desarrollo y el mantenimiento del sistema brindando información y haciendo el control. En otras palabras permite conocer el estado de los productos y facilita el mantenimiento del sistema aportando información del impacto de los cambios.

2.2.9. Tecnologías de Cuarta Generación

Las tecnologías de cuarta generación (T4G) son las que facilitan a los profesionales las especificaciones de algunas características del software a alto nivel. Se orienta hacia la posibilidad de especificar el software usando formas de lenguaje especializado o notaciones gráficas que describen el problema que hay que resolver. Existen datos en compañías que lo usan donde parece indicar que para aplicaciones pequeñas y medianas se reduce el tiempo de la producción de software. [Pressman, 2005]

La primera generación contaba con computadoras con válvulas y se programaban en lenguaje binario (unos y ceros) con tarjetas perforadas. Eran difíciles de aprender y nula la portabilidad entre distintas máquinas.

La segunda generación tenía computadoras con transistores y se programaban con lenguajes simbólicos que permitían mayor claridad para la escritura y lectura del código que con la generación anterior. Se empezó a ser más eficiente con el lenguaje ensamblador.

La tercera generación poseía computadoras con circuitos integrados y lenguajes de alto nivel que son independientes de la máquina y basados en procedimientos. Se empezó a utilizar la programación estructurada y las sentencias complejas. Se consiguió mayor abstracción y más funcionalidades que en las generaciones anteriores.

La cuarta generación cuenta con computadoras de circuitos integrados a gran escala y con herramientas de software que permiten construir aplicaciones sencillas combinando módulos predesarrollados. Los lenguajes de cuarta generación están orientados a resolver problemas y reducen el tiempo, costo y esfuerzo de programación.

La quinta generación cuenta con computadoras de circuitos integrados a gran escala y respecto del software con lenguajes de inteligencia artificial, aplica la lógica computacional y está basado en reglas; como ventaja se puede señalar los avances en robótica.

Los lenguajes de cuarta generación (4GLs) minimizan la cantidad de código que hay que escribir para obtener un sistema de información. Los programadores usan expresiones de alto nivel, especifican qué es lo que quieren que la computadora realice y no cómo. Estos lenguajes suelen incluir interfaces gráficos y capacidad de gestión avanzada. También permiten involucrar mucho al usuario que en el futuro será quien utilizará el sistema. Entre las áreas de aplicación podemos mencionar la consulta a bases de datos y la generación de informes. Una ventaja que se puede señalar es la facilidad de aprenderlos.

2.2.10. Generación de prototipos y modelos evolutivos

Cuando se habla de “modelos de procesos evolutivos” se reconoce que la incertidumbre domina la mayoría de los proyectos, el tiempo asignado es poco y la iteración es una solución parcial en cuanto a la ventaja que la misma otorga. Se procura llegar a modelos evolutivos con los que se

puedan obtener productos de trabajo incrementales, análisis de riesgo, planificación, revisión de planes y realimentación por parte del cliente. Para eso se estudian métodos más eficaces y herramientas más potentes que ayuden a la solución de problemas.

Al referirse a la "generación de prototipos" se puede considerar que el uso de los métodos formales dentro de la Ingeniería del Software no está centrado solamente en aspectos de especificación, podríamos aplicarlos también en la obtención de programas que resuelvan los problemas especificados. De esta manera decimos que el interés se centra en explotar los formalismos de especificación junto con mecanismos que automaticen la construcción de programas a partir de las especificaciones. Estos mecanismos de refinamiento nos permitirían no sólo obtener programas correctos sino también cubrir aspectos no funcionales como por ejemplo los de rendimiento. Las tareas de interés son:

– Soporte en la validación de especificaciones.
– Automatización en derivación de programas.
– Transformaciones horizontales: los programas se transforman en programas semánticamente equivalente usando un lenguaje común con vistas a mejorar el rendimiento.
– Transformaciones verticales: la posibilidad de transformar programas escritos en un lenguaje a programas en otro lenguaje preservando la semántica con múltiples objetivos (rendimiento, portabilidad, etc).
– Registrar diferentes versiones de una misma entidad: especificación y programas en diferentes estados de implementación (programas iniciales, programas finales más eficientes, etc). Aspectos necesarios como documentación de diseño en el desarrollo del software.

Una herramienta que se está usando mucho en la ingeniería de software son los "frameworks". Estos son Generadores de Aplicación que se relacionan directamente con un dominio específico, es decir con una familia de problemas relacionados. El desarrollo del framework está ganando rápidamente aceptación debido a su capacidad para promover la reutilización del código de diseño y el código fuente (source code) [Markiewicz, 2003]. El mismo consiste de las siguientes etapas: dominio del análisis donde se captura los requisitos, diseño del framework, e instanciación del framework (aplicación1, ... , aplicación N).

2.3. La calidad en la ingeniería de software

2.3.1. Definición de Calidad

El término calidad deriva de la raíz latina *qualitas* que significa condición, se la usa para caracterizar propiedades valiosas. Es la satisfacción de expectativas, en otras palabras podemos decir que es un término relativo. Tiene sentido únicamente cuando se compara o en relación a necesidades, expectativas pretensiones o especificaciones.

En la Norma ISO 8402 se definió a la calidad como: *"la totalidad de las características de una entidad que le confieren la aptitud para satisfacer las necesidades establecidas e implícitas"*.

Se define la calidad como el: *"grado en el que un conjunto de características inherentes cumple con los requisitos"*. Inherente significa que existe en algo, especialmente como una característica permanente. [IRAM-ISO 9000].
En el sentido más amplio, calidad del software es el *cumplimiento de los requisitos de funcionalidad y desempeño explícitamente establecidos, de los estándares de desarrollo explícitamente documentados y de las características implícitas que se esperan de todo software desarrollado profesionalmente*. [Pressman, 2005]

Un concepto de calidad para una empresa productora se refiere a obtener un producto perfecto, sin defectos, de bajo costo, seguridad del cliente, entrega en el lugar exacto, entrega en el plazo exacto y entrega en la cantidad exacta. [Alfaro, 2003]
En particular para la ingeniería de software cuando se propone construir un producto de calidad se refiere a cumplir con las especificaciones, entregarlo en el plazo convenido y de la manera que se comprometió a hacerlo.
Para alcanzar esta meta los empleados de la empresa deben tener un compromiso con la calidad. Para conseguir esto hay que formar a los directivos, como también es necesario educar y motivar a los empleados. [Kaplan, 1995]

2.3.2. Factores de Calidad

Los factores de calidad sirven para valorar lo que se construye y usa del software, permite establecer requisitos que se pueden medir y finalmente esta medida debe ser concordante con la calidad esperada. Hay factores que se pueden medir directamente (como por ejemplo los defectos por punto de función) y otros indirectamente (como por ejemplo facilidad de uso). La relación entre los factores de calidad y las métricas de la calidad del software del Modelo de Calidad según Mc Call tienen las siguientes características (métrica de la calidad del software/factor de calidad):

Corrección	Compleción, consistencia, trazabilidad
Confiabilidad	Exactitud, complejidad, consistencia, tolerancia a errores, modularidad, simplicidad
Eficiencia	Concisión, eficiencia de ejecución, Operatividad
Integridad	Facilidad de auditoría, instrumentación, seguridad
Facilidad de uso	Facilidad de operación, facilidad de formación
Facilidad de mantenimiento	Concisión, consistencia, instrumentación, modularidad, autodocumentación, simplicidad
Flexibilidad	Complejidad, concisión, consistencia, simplicidad, facilidad de expansión, generalidad, modularidad, autodocumentación, simplicidad
Facilidad de prueba	Facilidad de auditoría, complejidad, instrumentación, modularidad, autodocumentación, simplicidad
Portabilidad	Generalidad, independencia del hardware, modularidad, autodocumentación, independencia del sistema
Facilidad de Reutilización	Generalidad, independencia del hardware, modularidad, autodocumentación, independencia del sistema
Interoperabilidad	Estandarización de comunicaciones, estandarización de datos, generalidad, modularidad

Tabla 2: Modelo de Calidad según Mc Call

Una clasificación de los factores de calidad del software según McCall nos muestra como se ve afectado en el producto software: revisión del producto (facilidad de mantenimiento, flexibilidad y facilidad de prueba), operación del producto (corrección, confiabilidad, facilidad de uso, integridad y eficiencia), y transición del producto (portabilidad, facilidad de reutilización e

interoperabilidad). Estos factores junto con un conjunto de métricas nos permiten identificar y describir aspectos que afectan a la calidad del software. [Pressman, 2005, p.646 y 465].

Mc Call describe los factores de calidad que propone de la siguiente manera:

- Corrección. El grado en que un programa satisface sus especificaciones y consigue los objetivos propuestos por el cliente. Responde a la pregunta: ¿Hace lo que quiero?
- Confiabilidad. El grado en que se puede esperar que un programa lleve a cabo sus funciones esperadas con la precisión requerida (Se puede decir que se han propuesto otras definiciones de fiabilidad más completas). Responde a la pregunta: ¿Lo hace en forma fiable todo el tiempo?
- Eficiencia. La cantidad de recursos de computadora y de código requeridos por un programa para llevar a cabo sus funciones. Responde a la pregunta: ¿Se ejecutará en mi hardware lo mejor que se pueda?
- Integridad. El grado en que puede controlarse el acceso al software o a los datos, por personal no autorizado. Responde a la pregunta: ¿Es seguro?
- Facilidad de uso. El esfuerzo requerido para aprender a trabajar con un sistema, preparar las entradas e interpretar las salidas del programa. Responde a la pregunta: Está diseñado para ser usado?
- Facilidad de mantenimiento. El esfuerzo requerido para localizar y arreglar un error en un programa (Se puede decir que se trata de una definición muy limitada). Responde a la pregunta: ¿Permite ser corregirlo con relativa facilidad?
- Flexibilidad. El esfuerzo requerido para modificar un programa que se encuentra operativo. Responde a la pregunta: ¿Permite ser cambiado con relativa facilidad, es flexible para modificarlo?
- Facilidad de prueba. El esfuerzo requerido para probar un programa de forma que se asegure que realiza la función requerida. Responde a la pregunta: ¿Permite ser probado con relativa facilidad?
- Portabilidad. El esfuerzo requerido para transferir el programa desde un hardware y/o software a otro entorno. Responde a la pregunta: ¿Podré usarlo en otra computadora?
- Facilidad de Reutilización. El grado en que un programa (o partes de un programa) se puede volver a usar en otras aplicaciones. Esto está relacionado con el empaquetamiento y el alcance de las funciones que realiza el programa. Responde a la pregunta: ¿Podré reusar alguna parte del software?
- Interoperabilidad. El esfuerzo requerido para acoplar un sistema con otro. Responde a la pregunta: ¿Podré hacerlo interactuar con otros sistemas?

Mc Call describe los atributos de la siguiente manera (se puede decir que cada uno de estos atributos puede corresponder a más de un factor de calidad):

- Facilidad de auditoría. La facilidad con que se puede comprobar la conformidad con los estándares.
- Exactitud. La precisión de los cálculos y del control.
- Estandarización de las comunicaciones. El grado en que se usan el ancho de banda, los protocolos y los estándares de interfaces.

- Compleción. El grado en que se ha conseguido la implementación total de las funciones.
- Concisión. Lo compacto que es el programa en términos de líneas de código.
- Consistencia. El uso de un diseño uniforme y de técnicas de documentación a lo largo del proyecto de desarrollo.
- Estandarización en los datos. El uso de estructuras y de tipos de datos estándar a lo largo de todo el programa.
- Tolerancia al error. El daño que se produce cuando el programa encuentra un error.
- Eficiencia en la ejecución. El rendimiento del funcionamiento de un programa.
- Capacidad de expansión. El grado en que se puede ampliar el diseño arquitectónico, de datos o procedimental.
- Generalidad. La amplitud de aplicación potencial de los componentes del programa.
- Independencia del hardware. El grado con que se desacopla del software donde opera.
- Instrumentación. El grado en que el programa estudia su propio funcionamiento e identifica los errores que pueden ocurrir.
- Modularidad. La independencia funcional de los componentes del programa.
- Operatividad. La facilidad de operación de un programa.
- Seguridad. La disponibilidad de mecanismos que controlan o protejan los programas y/o los datos.
- Autodocumentación. El grado en que el código fuente proporciona documentación significativa.
- Simplicidad. El grado en que un programa puede ser entendido sin dificultad.
- Independencia del sistema de software. El grado en que el programa es independiente de características no estándares del lenguaje de programación, de las características del sistema operativo y de otras restricciones del entorno.
- Trazabilidad. La posibilidad de seguir una representación del diseño o de un componente real del programa hasta los requisitos.
- Formación. El grado en que el software ayuda a manejar el sistema a los nuevos usuarios.

[Pressman, 2005].

El modelo de calidad del producto según Boehm (1978), es útil porque ayuda a la comprensión de las formas en que las diversas facetas contribuyen al todo, también muestra que la calidad es algo mas amplio que limitar la crítica a defectos y fallas. Comienza con una utilidad general ya que todo software debe ser útil para algo; después considera que un primer tipo de usuario como el caso del cliente se puede interesar en la portabilidad; otro usuario como por ejemplo de la gerencia se fija en la utilidad percibida (confiabilidad, eficiencia, e ingeniería humana); y finalmente un tercer tipo de usuario como puede ser el programador en la facilidad de mantenimiento (facilidad de prueba, de comprensión y de modificación). Este modelo tiene una jerarquía de características que se terminan definiendo así: [Pfleeger, 2002, p.604]

Portabilidad	Independencia del dispositivo, Autocontención
Confiabilidad	Autocontención, Correctitud, Completitud, Robustez/Integridad, Consistencia
Eficiencia	Capacidad de contabilidad, Accesibilidad, Eficiencia de dispositivo
Ingeniería humana	Robustez/Integridad, Accesibilidad, Comunicación
Facilidad de prueba	Capacidad de contabilidad, Accesibilidad, Comunicación, Autodescripción, Estructura
Facilidad de comprensión	Consistencia, Autodescripción, Estructura, Concisión, Legibilidad
Facilidad de modificación	Estructura, Extensibilidad

Tabla 3: Modelo de calidad del producto Boehm

El modelo de calidad según ISO 9126 de 1990 posee un conjunto de características y atributos. Es un modelo jerárquico con seis atributos que contribuyen a la calidad.
La *funcionalidad* define a un conjunto de atributos que afectan la existencia de un conjunto de funciones y sus propiedades especificadas. Estas funciones satisfacen las necesidades enunciadas o implícitas. La funcionalidad comprenderá los atributos de adaptabilidad, exactitud, interoperación y seguridad.
La *confiabilidad* es un conjunto de atributos que pueden incidir sobre la aptitud del software para mantener su nivel de desempeño bajo las condiciones establecidas y por un período de tiempo establecido. Comprenderá los atributos de madurez, tolerancia a defectos y facilidad de recuperación.
La *facilidad de uso* es un conjunto de atributos que determinan el esfuerzo necesario para el uso y la valoración individual de tal uso por parte de un conjunto de usuarios declarados o implicados. Incluye los atributos de facilidad de comprensión, de aprender y de operar.
La *eficiencia* es un conjunto de atributos que afectan la relación entre el desempeño del software y la cantidad de recursos utilizados bajo condiciones establecidas. Comprende los atributos de comportamiento en el tiempo y de los recursos.
La *facilidad de mantenimiento* es un conjunto de atributos que determinan el esfuerzo necesario para realizar modificaciones especificadas. Incluye correcciones, mejoras o adaptaciones del software a los cambios ambientales y de requerimientos y en las especificaciones funcionales. También comprende los atributos de facilidad de análisis, facilidad de cambios, estabilidad y facilidad de prueba.
La *portabilidad* es un conjunto de atributos que inciden sobre la aptitud del software para ser transferido de un ambiente a otro; tanto del ambiente organizacional, de hardware o de software. Incluye los atributos de adaptabilidad, facilidad de instalación, conformidad y facilidad de reemplazo. [Pfleeger, 2002, p.606]

La norma IRAM 36052:1996 (ICS 35080 CNA 0000) Informática. Tecnología de la Información. Evaluación de productos de soporte lógico (software). Características de la calidad y guía para su uso.

Está basada en la ISO 9126:1991. Define seis características que describen la calidad del soporte lógico. También describe como se usan estas características para la evaluación de la calidad del software.

La norma tiene siguientes títulos:

1. Objeto y campo de aplicación
2. Normas para consulta
3. Definiciones
4. Características de la calidad del soporte lógico (“software”)
5. Guía para el uso de las características de la calidad
6. Anexo A Subcaracterísticas de la calidad
7. Anexo B
8. Bibliografía

Cuando se trata el punto 4 titulado Características de la calidad del soporte lógico ("software") menciona las características de la calidad de la siguiente manera:

4.1.Funcionalidad	Adaptabilidad, exactitud, interoperabilidad, conformidad, seguridad
4.2.Confiabilidad	Madurez, tolerancia a fallas, recuperabilidad
4.3.Usabilidad	Inteligibilidad, facilidad de aprendizaje, operatividad
4.4.Eficiencia	Comportamiento temporal, utilización de recursos
4.5.Mantenibilidad	Facilidad de análisis, facilidad de cambios, estabilidad, verificabilidad
4.6.Portabilidad	Adaptabilidad, facilidades de instalación, conformidad, facilidad de reemplazo

Tabla 4: Características de la calidad del software

Según Piattini deberíamos hablar de *"funcionamiento",* el software debe funcionar siempre debe permitirnos usarlo cuando es necesario; *"funcionalidad*" el software debe cumplir con las funcionalidades que publica; o sea, debe hacer lo que dice que hace; y *"usabilidad debe permitirnos hacerlo de forma adecuada y natural*" [Piattini, 2003, p.3].

En los párrafos anteriores se describieron una serie de factores cualitativos para la medición de la calidad de software pero muchas veces se busca desarrollar medidas más precisas que no se vean afectadas por la naturaleza subjetiva de la actividad. Para eso es posible aplicar un conjunto de métricas que permiten una evaluación cuantitativa de la calidad del software.

2.3.3. Métrica del Punto de Función

Definida por Allan J. Albrecht, de IBM, en 1979, es un método para medir el tamaño del sistema a partir del modelo de análisis. Esta métrica es útil para establecer el tamaño (funcional o lógico) y la complejidad de los sistemas informáticos basada en la cantidad de requerimientos funcionales de los usuarios. Por lo tanto se puede predecir el esfuerzo y costo para diseñar codificar y probar el software, además de valorar el desarrollo, como así también predecir el número de errores que se encontrarán durante la prueba y pronosticar el número de componentes, de líneas de código o ambas en el sistemas implementado. [Pressman, 2005, .474]

Tiene de bueno su simplicidad y no requiere mucho esfuerzo para su uso. Cuando se habla de funcionalidad se hace referencia a la capacidad del software para que un usuario pueda realizar transacciones (lectura, escritura, etc.) y de guardar datos. Si se lo analiza en detalle con estos elementos se puede describir cualquier sistema.

La principal crítica que recibe esta métrica es la de requerir una dedicación adicional en los proyectos de desarrollo de software, los que normalmente suelen desenvolverse con presupuestos ajustados. Se cuestiona la precisión cuando se trata de proyectos pequeños, como los que están por debajo de los 100 PF que puede resultar en poco fiables.

La fórmula para Calcular los puntos de función sin ajustar (UFPC) o conteo total va a ser:

$$UFPC = \Sigma FP (ILF) + \Sigma FP (EIF) + \Sigma FP (EI) + \Sigma FP (EO) + \Sigma FP (EQ)$$

Donde:
Número de entradas externas (External Input - EI)

Número de salidas externas (External Output - EO)
Número de peticiones o consultas externas (External Inquiry - EQ)
Número de archivos lógicos internos (Internal Logical File - ILF)
Número de interfaces de internfas externas (External Interface File - EIF)

Para Calcular el Factor de Ajuste se tiene en cuenta la complejidad del sistema los F sub i (siendo i entre 1 y 14). Según las repuestas que se obtengan a las siguientes preguntas:

1. ¿Requiere el sistema copias de seguridad y de recuperación confiables?
2. ¿Se requiere comunicación de datos especializadas para transferir información a la aplicación u obtenerla de ella?
3. ¿Existen funciones de procesamiento distribuido?
4. ¿Es crítico el rendimiento?
5. ¿Se ejecutará el sistema en un entorno operativo existente y fuertemente utilizado?
6. ¿Requiere el sistema entrada de datos en línea?
7. ¿Requiere la entrada de datos en línea que las transacciones de entrada se lleven a cabo sobre múltiples pantallas u operaciones?
8. ¿Se actualizan los archivos maestros de forma interactiva?
9. ¿Son complejas las entradas, las salidas, los archivos o las peticiones?
10. ¿Es complejo el procesamiento interno?
11. ¿Se ha diseñado el código para ser reutilizado?
12. ¿Están incluidas en el diseño la conversión y la instalación?
13. ¿Se ha diseñado el sistema para soportar múltiples instalaciones en diferentes organizaciones?
14. ¿Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizada por el usuario?

Las respuestas a las preguntas anteriores es respondida usando una escala con rangos desde 0 a 5. Se pueden aplicar factores de peso a las cuentas de los dominios de información empíricamente.

La fórmula para Calcular los Puntos de Función Ajustados (AFPC) va a ser:

$$AFPC = UFPC \times [0{,}65 + 0{,}01 \times \Sigma (Fi)]$$

El factor de ajuste se obtiene sumando 0,65 a la sumatoria de los grados de influencia de las 14 características generales del sistema, multiplicado por 0,01. Dentro de las características hay criterios tales como: complejidad del proceso, facilidad de instalación, entrada de datos en línea y otros.

En la Tabla 5 llamada Hoja de Complejidad se observa el valor recomendado a los elementos respecto de los registros o archivos siendo las designaciones simple, media o compleja.

FILES	**ELEMENTOS:1-19**	**ELEMENTOS:20-50**	**ELEMENTOS:>50**
RECORDS:1	LOW	LOW	AVERAGE
RECORDS:2-5	LOW	AVERAGE	HIGH
RECORDS:>5	AVERAGE	HIGH	HIGH
INPUT	**ELEMENTOS:1-4**	**ELEMENTOS:5-15**	**ELEMENTOS:>15**
FTR:0-1	LOW	LOW	AVERAGE
FTR:2	LOW	AVERAGE	HIGH
FTR:>2	AVERAGE	HIGH	HIGH
OUTPUT	**ELEMENTOS:1-5**	**ELEMENTOS:6-19**	**ELEMENTOS:>19**
FTR:0-1	LOW	LOW	AVERAGE
FTR:2-3	LOW	AVERAGE	HIGH
FTR:>3	AVERAGE	HIGH	HIGH

Tabla 5: Hoja de Complejidad

	LOW	AVERAGE	HIGH
EXTERNAL INPUT	3	4	6
EXTERNAL OUTPUT	4	5	7
EXTERNAL INQUIRY	3	4	6
INTERNAL LOGICAL FILE	7	10	15
EXTERNAL INTERFACE FILE	5	7	10

Tabla 6: Hoja de Contribución

donde:
low significa simple; average significa medio y high significa complejo.

Existen varios estándares de la ISO para la métrica punto de función, como son por ejemplo los siguientes:
ISO/IEC 14143 – Information Technology – Software Measurement – Functional Size Measurement.
Este estándar define los conceptos de una métrica de tamaño basada en la funcionalidad y en las características que debe cumplir un método para poder estar homologado al estándar y ser considerado una medida del tamaño de la funcionalidad.

Para poder establecer la cantidad de puntos función que tiene una aplicación existen en la actualidad varios métodos. En general, todos ellos establecen un conteo basado en la identificación del tipo de funciones que tiene la aplicación, y a cada una le asocia un número de puntos función tomando en cuenta su complejidad. Las variantes surgen al buscar conteos más precisos en puntos función conforme al tipo de aplicación. Como ejemplo se puede decir que un sistema en tiempo real tiene una complejidad muy distinta al un sistema tradicional de negocio, como también a un sistema operativo o a una aplicación científica que realiza muchos cálculos, pero el resultado puede ser un solo dato. Los métodos que están homologados con el ISO 14143, aunque no todos son públicos, son:

- ISO/IEC 20926:2003 Software engineering -- IFPUG 4.1 Unadjusted functional size measurement method. Este método ha sido definido por el Internacional Function Point

Users Group (IFPUG[13]) y ha evolucionado, a partir de la propuesta original de Allan Albrecht en IBM, por lo que es el más conocido y más utilizado, sobre todo en Estados Unidos que es el mercado más grande de SW. Esto es muy importante porque se está convirtiendo de facto en el estándar de la industria.

- ISO/IEC 24570. Software engineering -- NESMA -- Function Point Analysis. El estándar fue definido por la NEtherlands Software Metrics Users Association. Esta es una pequeña variante del método del IFPUG.
- ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis. Puntos de Función Mark II (MKII) es un método que ha sido desarrollado por la UKSMA (United Kingdom Software Metrics Association), es la simplificación del método haciéndolo compatible con ideas de análisis y diseño estructurado.
- ISO/IEC 19761:2003 Software engineering -- COSMIC-FFP -- A functional size
- measurement method. Este método fue sido definido por el COSMIC (COmmon Software Measurement International Consortium), integrado por expertos de Australia, Canadá, Finlandia, Alemania, Irlanda, Italia, Japón, Holanda y el Reino Unido. La idea principal que persigue es adecuarse mejor a la medición de los sistemas en tiempo real.

2.3.4. Métrica Bang

La métrica de Bang fue definida por De Marco en 1982. Está orientada a la función, por lo tanto es indicativa del tamaño funcional del sistema y es independiente de todo tipo de cuestiones de implementación. Es un método para medir el tamaño del software después del análisis con el diagrama de flujos. Sugiere que todo el software puede agruparse en dos grandes dominios: Dominio de Función o Dominio de Datos.

Se consideran en la implementación los componentes elementales (que no pueden dividirse en más pequeños) o primitivos. En cada etapa del modelo de especificación (modelo funcional, modelo de datos y modelo de comportamiento) a cada parte se la descompone hasta el nivel más bajo donde todos los componentes son elementales.

De Marco contabiliza seis tipos de componentes: primitivas funcionales (son los componentes elementales de las descomposiciones del modelo funcional, ej. descomposición del DFD), datos elementales (son los datos concretos e indivisibles, ej. los del Diccionario de Datos), objetos (son los componentes elementales de las descomposiciones del modelo de datos, ej. un diagrama de E/R), interrelaciones (son los componentes elementales derivados de las relaciones de unos objetos y otros), estados (son los componentes elementales de las descomposiciones del modelo de comportamiento, ej. diagrama de estados/transiciones) y transiciones (son las acciones que pasan al sistema de un estado a otro del modelo de comportamiento).

De todas maneras De Marco agregó seis componentes elementales más para la contabilización quedando entonces todos los siguientes números de: primitivas funcionales (FP), primitivas funcionales modificadas (FPM), datos elementales (DE), datos elementales de entrada (DEI), datos elementales de salida (DEO), datos elementales retenidos (DER), objetos (OB), interrelaciones (RE) del modelo de datos del sistema, estados (ST), transiciones (TR), tokens

(TC_i) implicados en la i-ésima primitiva funcional y el número de interrelaciones (RE_i) en que está involucrado el i-ésimo objeto del modelo de datos.

Para los sistemas orientados a procesos el FP es el indicador principal que es cuando hay un alto grado de componente funcional y para los orientados a datos será el OB. Se proponen criterios para clasificar los sistemas los cuales son:

Si RE / FP < 0.7 entonces el sistema está orientado a proceso y transformación de datos
Si RE / FP > 1.5 entonces el sistema está fuertemente orientado a datos
Si (RE / FP >= 0.7) y (RE / FP <= 1.5) entonces el sistema es híbrido

Además se tiene el Ratio DEO / FP que es un ratio indicativo del trasiego de datos desde dentro del sistema hacia fuera (es elevado para sistemas comerciales y de gestión, y pequeño para los de ingeniería y científicos).

Para la formulación de la métrica se distinguen los tres tipos de sistemas software (orientados a la función, orientado a los datos y los híbridos). Estos son:

Para sistemas orientados a la función (proceso y transformación de datos):
El factor $TC_{medio} = \Sigma\ TC_i / FP$ es en el proceso de descomposición de un modelo cuando se deja un componente como elemental porque no se lo puede dividir más o porque la descomposición no reduce este factor. Siendo el tamaño corregido de cada primitiva funcional:

$$CFPI_i = TC_i \times \log_2 (TC_i)$$

Finalmente se puede decir que el tamaño total corregido de todas las primitivas funcionales del sistema es:

$$CFP = \Sigma\ CFPI_i$$

Se puede mencionar que De Marco propuso una clasificación para la variación de las primitivas funcionales en diferentes categorías a las que les asignaba un peso o factor de corrección. Los pesos asignados dependen del desarrollo y están afectados por el lenguaje de programación por lo que un experto puede desarrollar su propia tabla.

CATEGORÍA	PESO
Separación	0.6
Unión	0.6
Encauzado de datos	0.3
Actualización simple	0.5
Almacenamiento	1.0
Edición	0.8
Verificación	1.0
Manipulación de texto	1.0
Sincronización	1.5
Generación de salidas	1.0
Muestreo	1.8
Análisis tabular	1.0
Aritmética	0.7
Iniciación	1.0
Cálculo matemático	2.0
Manipulación de dispositivos	2.5

Tabla 7: Categorías y peso según la complejidad de las primitivas funcionales

Para sistemas orientados a los datos:
Teniendo en cuenta que el indicador principal es el OB se propone un factor RE para la corrección. Por eso se usa la tabla para los factores de corrección (COBI: Corrected Object Increment) para los objetos de un sistema en función de sus interrelaciones con otros que es la siguiente:

REi	COBI
1	1.0
2	2.3
3	4.0
4	5.8
5	7.8
6	9.8

Tabla 8: Peso para los objetos de un sistema en función de sus interrelaciones con otros.

Entonces el tamaño funcional total de la aplicación medida está representado por el COB que es:

$$COB = \Sigma\ COBI_i$$

Para sistemas híbridos:
Considera calcular independientemente las dos métricas Bang de manera que una representa el esfuerzo de desarrollo de la parte funcional y la otra la parte orientada a los datos.

Para concluir se puede decir que la propuesta especifica una serie de factores que se calculan en el entorno de trabajo del propio programador, por lo tanto termina siendo poco confiable al no haber valores independientes para comparar estos valores.

2.3.5. Métrica Halstead

Esta métrica fue definida por Halstead a fines de los 70, cuando propuso las primeras leyes analíticas para el software de computadora como lo es el código fuente. Es un método que deriva de la teoría general de la complejidad y también de otras teorías de la producción de la programación. Utiliza medidas primitivas para desarrollar expresiones para obtener la longitud global del programa y la complejidad de este. El token es la unidad más elemental distinguible por un compilador, se los puede clasificar como operadores u operandos. La longitud del programa sería la cantidad de tokens usados.

Para medir el código fuente de un programa bien estructurado propone que la longitud N se puede determinar por:

$$N = n1 \log2\ n1 + n2 \log2\ n2$$

donde :
n1 = el número de operadores distintos que aparecen en un programa.
n2 = el número de operandos distintos que aparecen en un programa.
N1 = el número total de veces que aparece un operador.

N2 = el número total de veces que aparece un operando.

El volumen de programa se puede definir como:

$$V = N \log2 (n1 + n2)$$

Como debe haber un volumen mínimo para un programa define la relación de volumen L que debe ser menor que 1. La relación de volumen para las medidas primitivas es:

$$L = 2/n1 \times n2/N2$$

Si aplicamos las métricas de Halstead a las pruebas podemos estimar el esfuerzo.
Siendo V el volumen, NP el nivel de programa y e el esfuerzo tenemos que:

$$NP = 1 / [(n1 / 2) (N2 / n2)]$$

$$e = V / NP$$

(donde el uso promedio de cada operador es N2 / n2 y medir cuanto más difícil es el programa respecto de invocar una función es n1 / 2).
A la ecuación que calcula el esfuerzo general de prueba se le debe agregar un k que se estima así:

$$\text{Porcentaje de esfuerzo de prueba } (k) = e(k)/ \Sigma e(i)$$

[Pressman, 2005].

2.3.6. Introducción a la Normalización

La normalización implica la participación de personas que representan a distintas organizaciones de los tres sectores involucrados: productores, consumidores y de intereses generales. Estos representantes aportan su experiencia y sus conocimientos para establecer soluciones a problemas reales o potenciales. De acuerdo con la ISO (International Standards Organization que tiene su base en Ginebra Suiza) la normalización es la actividad que tiene por objeto establecer, tanto frente a problemas reales como potenciales, disposiciones destinadas a usos comunes y repetidos, con el fin de obtener un nivel de ordenamiento óptimo en un contexto dado. Este contexto puede ser tecnológico, político o económico. Se hace referencia, entonces, a una actividad que se plasma en un hecho práctico, que luego hay que concretar en un documento que se pone a disposición del público.

Un tema que se considera cuando se trata el aseguramiento de la calidad en el costo de certificar a la organización como si fuera un lujo. En realidad, un costo inadmisible que no se puede dar son las fallas en los productos.

Como ventajas de la aplicación de normas de calidad podemos señalar el acceso a los mercados y el funcionamiento confiable. Es importante que se pueda recomendar usarlas para ingresar a mercados que lo exigen, por cuestiones de calidad y seguridad. Como desventaja está el costo económico. Cualquier empresa que trabajaba sin estar normalizada al incorporar esta modalidad de trabajo afecta la cultura de la misma, entonces puede disminuir la productividad.

2.3.7. La Norma ISO 9000

La norma IRAM-ISO 9000: Sistema de gestión de la calidad, fundamentos y vocabulario; describe los fundamentos de los sistemas de gestión de la calidad y especifica el vocabulario. Reemplaza a la ISO 8402:1994 y a la ISO 9000-1:1994. La ISO 9000-3:1997 trata las directrices para la aplicación de la 9001 en desarrollo, suministro, instalación y mantenimiento de software. Es importante mencionar que para ISO siempre hay un contrato que puede ser formal o informal donde hay un proveedor y un adquiriente, cada uno con derechos y obligaciones. Las partes definen que documentación va a emplear, debido a que no hay una metodología específica. También exige que se publique lo que se va a hacer y después demuestre que lo ha hecho.
La estrategia está basada en tres puntos: sistemas de calidad, documentos de control y auditorias. [Kaplan, 1995]

El contenido de la norma IRAM-IACC-ISO 9000-1:1994 abarca los siguientes temas:
0 Introducción
- 0.1 Generalidades
- 0.2 Principios de gestión de la calidad

1 Objeto y campo de aplicación
2 Fundamentos de los sistemas de gestión de la calidad
- 2.1 Base racional para los sistemas de gestión de la calidad
- 2.2 Requisitos para los sistemas de gestión de la calidad y requisitos para los productos
- 2.3 Enfoque de sistemas de gestión de la calidad
- 2.4 Enfoque basado en procesos
- 2.5 Política de la calidad y objetivos de la calidad
- 2.6 Papel de la alta dirección dentro del sistema de gestión de la calidad
- 2.7 Documentación
- 2.8 Evaluación de los sistemas de gestión de la calidad
- 2.9 Mejora continua
- 2.10 Papel de las técnicas estadísticas
- 2.11 Sistemas de gestión de la calidad y otros sistemas de gestión
- 2.12 Relación entre los sistemas de gestión de la calidad y los modelos de excelencia

3 Términos y definiciones
- 3.1 Términos relativos a la calidad
- 3.2 Términos relativos a la gestión
- 3.3 Términos relativos a la organización
- 3.4 Términos relativos al proceso y al producto
- 3.5 Términos relativos a las características
- 3.6 Términos relativos a la conformidad
- 3.7 Términos relativos a la documentación
- 3.8 Términos relativos al examen
- 3.9 Términos relativos a la auditoria
- 3.10 Términos relativos al aseguramiento de la calidad para los procesos de medición

Anexo A (Informativo) Metodología utilizada en el desarrollo del vocabulario
Anexo B (Informativo) Bibliografía
Anexo C (Informativo) Índice alfabético
Anexo D (Informativo) Integrantes del organismo de estudio

2.3.8. La Norma ISO 9001

La norma IRAM-ISO 9001: Sistema de gestión de la calidad, requisitos; especifica los requisitos para los sistemas de gestión de la calidad aplicables a toda organización. Es la única de esta serie que puede certificarse. Reemplaza ISO 9001:1994, la ISO 9002:1994 y la ISO 9003:1994.

ISO 9001 de 1994 y 2000 (Requisitos), Model for Quality Assurance in Design, Development, Production, Installation and Servicing. Esta norma de calidad enfocada al software, concierne al diseño, desarrollo, producción instalación y servicio de manutención del software.

Tiene como objetivo lograr de forma coherente la satisfacción del cliente con los productos y servicios de la organización, cuando se necesita manifestar la capacidad para demostrar la conformidad con los requisitos del cliente y los requisitos reglamentarios aplicables. También para mejorar continuamente el sistema de gestión de calidad.

El ISO 9001 adoptó un ciclo de planear, hacer, revisar y actuar que se aplica a los elementos de gestión de la calidad. En un proyecto de software planear establece los objetivos, las actividades y tareas del proceso para conseguir un producto de calidad que satisfaga al cliente; hacer es la implementación del proceso de software; revisar es cuando se monitorea y mide el proceso para asegurarse de que todos los requisitos establecidos para la gestión de calidad hayan sido cumplidos; actuar inicia las actividades de mejoramiento del proceso de software que debe tener una continuidad de trabajo para mejorar el proceso. [Pressman, 2005]

El contenido de la norma IRAM-IACC-ISO 9001 abarca los siguientes temas como lo describe el índice:

Introducción
1 Objeto y campo de aplicación
    1.1 Generalidades
    1.2 Aplicación
2 Referencias normativas
3 Términos y definiciones
4 Sistemas de gestión de la calidad
    4.1 Requisitos generales
    4.2 Requisitos de la documentación
5 Responsabilidad de la dirección
    5.1 Compromiso de la dirección
    5.2 Enfoque al cliente
    5.3 Política de la calidad
    5.4 Planificación
    5.5 Responsabilidad, autoridad y comunicación
    5.6 Revisión por la dirección
6 Gestión de los recursos
    6.1 Provisión de recursos
    6.2 Recursos humanos
    6.3 Infraestructura

6.4 Ambiente de trabajo
7 Realización del producto
7.1 Planificación de la realización del producto
7.2 Procesos relacionados con el cliente
7.3 Diseño y desarrollo
7.4 Compras
7.5 Producción y prestación del servicio
7.6 Control de los dispositivos de seguimiento y de medición
8 Medición, análisis y mejora
8.1 Generalidades
8.2 Seguimiento y medición
8.3 Control del producto no conforme
8.4 Análisis de datos
8.5 Mejora
Anexo A (Informativo) Correspondencia entre las normas IRAM-ISO 9001:2000 e IRAM-ISO 14001:1996
Anexo B (Informativo) Correspondencia entre las normas IRAM-ISO 9001:2000 e IRAM-IACC-ISO E 9001:1994
Anexo A (Informativo) Bibliografía
Anexo A (Informativo) Integrantes del organismo de estudio

2.3.9. La Norma ISO 9004

La norma IRAM-ISO 9004 Sistemas de gestión de la calidad, Directrices para la mejora del desempeño; proporciona las directrices que consideran tanto la eficacia como la eficiencia del sistema de gestión de la calidad. Las directrices son para la mejora del desempeño. Reemplaza ISO 9004-1:1994.

El contenido de la norma IRAM-ISO 9004:2000 abarca los siguientes temas:
0 Introducción
0.1 Generalidades
0.2 Enfoque basado en procesos
0.3 Relación con la norma IRAM-ISO 9001
0.4 Compatibilidad con los otros sistemas de gestión
1 Objeto y campo de aplicación
2 Referencias normativas
3 Términos y definiciones
4 Sistema de gestión de la calidad
4.1 Gestión de sistemas y procesos
4.2 Documentación
4.3 Uso de los principios de gestión de la calidad
5 Responsabilidad de la dirección
5.1 Orientación general
5.2 Necesidades y expectativas de las partes interesadas
5.3 Política de la calidad

ISO 9004-4 – 1993 y 2000, Quality management and quality system elements - Part 4. Proporciona las directrices para las facilidades del servicio de software como soporte a usuarios. De manera de mantener la satisfacción del cliente. Abarca la eficiencia y la eficacia. Los requisitos son los siguientes puntos:

- Responsabilidad de la gestión
- Inspección, medición y equipo de pruebas
- Sistema de calidad
- Inspección y estado de pruebas
- Revisión de contrato
- Acción correctiva
- Control de diseño
- Control de producto no aceptado
- Control de documento
- Tratamiento, almacenamiento, empaquetamiento y entrega
- Compras

- Producto proporcionado al comprador
- Registros de calidad
- Identificación y posibilidad de seguimiento del producto
- Auditorias internas de calidad
- Formación
- Control del proceso
- Servicios
- Inspección y estado de prueba
- Técnicas estadísticas

2.3.10. La Norma ISO 19011

La norma IRAM-ISO 19011:2002 proporciona orientación relativa a las auditorías de sistemas de gestión de la calidad y ambiental. Reemplaza a la ISO 10011-1:1994, la ISO 10011-2:1994 y a la ISO 10011-3:1994 y otras tres directrices para auditorías de sistemas de gestión ambiental.

2.3.11. Norma ISO 12207

Tecnología de información (Information technology) – Proceso del ciclo de vida del software (Software life cycle process)

La Norma ISO/IEC 12207 (IEC: the International Electrotechnical Comisión) de 1995 y su Enmienda 1 de 2002 es un estándar internacional que propone una estructura para el ciclo de vida del Software y crear una terminología que sea utilizable por los ingenieros de software. Abarca desde el inicio de la idea de un sistema informático hasta el retiro del Software. Esta consiste en procesos para adquirir y suministrar productos de software. Los procesos que están incluidos en esta norma constituyen un conjunto de referencia para cualquier organización. Para eso la misma seleccionará el subconjunto de los procesos que sean adecuados para su proyecto de acuerdo con los objetivos que tenga. Es bueno considerar que ISO (International Standar Organization) habla de funcionalidad y eficiencia.
Un ciclo de vida consiste en las siguientes fases: adquisición, provisión (suministro), desarrollo, operación y mantenimiento.

La estructura está compuesta de procesos, actividades y tareas que deben realizarse para el cumplimiento del estándar. El procesos se subdivide en actividades, estos pueden ser primarios (son 5), de provisión o soporte (son 8) y organizativos (son 4). Los procesos incluidos en esta norma constituyen un conjunto de referencia. Una actividad se subdivide en tareas; o sea, la realización de una actividad es la terminación de todas las tareas que incluyen. Las tareas son unidades de trabajo que producen un resultado visible. Se especifican entradas, salidas, procedimientos, recursos y herramientas.

Procesos → actividades → tareas

El contenido de la norma ISO/IEC 12207 abarca los siguientes temas:
Prefacio

Introducción
1.Alcance
2.Referencia a Normativa
3.Definiciones
4.Aplicación del standar internacional
5.Procesos primarios del ciclo de vida (procesos de adquisición, suministrador, desarrollo, operación, mantenimientos)
6.Procesos de soporte del ciclo de vida (documentar, administrar configuraciones, asegurar la calidad, verificar, validar, revisión conjunta, auditoría y resolución de problemas)
7.Procesos organizacional del ciclo de vida (proceso de administración, mejora de infraestructura, procesos de instrucción)
8.Anexos.

2.3.12. La Norma ISO 15504 (SPACE)

La Norma ISO/IEC 15504 de1998 que fue actualizada como 15504-2 en el 2003 y es más conocida como SPACE (Software Process Improvement and Capability Determination).[SQI, 2004] o ASSESMENT

Esta norma define un conjunto de requisitos para la evaluación del proceso de software. Busca ayudar a las organizaciones en el desarrollo de una evaluación que sea objetiva en todos los procesos de software definido. [Pressman, 2005, p.37]

La Norma Space se compone de 9 partes:
Parte 1: Guía de conceptos e introducción
Parte 2: Un modelo para administrar procesos
Parte 3: Clasificación de procesos
Parte 4: Guía para conducir accesoria
Parte 5: Construcción, selección y uso de instrumentos y herramientas de accesoria
Parte 6: Calificación y entrenamiento de asesores
Parte 7: Guía para el uso de desarrollo de proyectos.
Parte 8: Guía para el usar en el proceso de determinación en las capacidades de provisionamiento
Parte 9: Vocabulario.

2.3.13. CMM Modelo de madurez de la capacidad

El CMM (Capability Maturity Model) del SEI (Software Engineering Institute) fue desarrollado por CMU (Cannegie Mellon University) es un modelo completo que se basa en un conjunto de funciones de ingeniería del software que debería estar en las organizaciones que alcanzan un nivel de madurez. La organización maneja su proceso de software por niveles (KPA: Key Process Area).

Tiene procesos que deben estar vigentes en todos los proyectos. Trata sobre 5 niveles de madurez que son: 1.- por default están todas, 2 y 3.- apuntan al management, y 4 y 5.- a nivel tecnológico.

El modelo CMM permite certificar por etapas, por lo tanto una pequeña empresa podría interesarse solamente en la primera o segunda de las ellas. De esta manera es accesible a las economías de algunas organizaciones.

Inicial: El *nivel 1* representa a las organizaciones que sin esfuerzo en la garantía de la calidad y gestión del proceso desarrollan software de cualquier manera eligiendo los métodos, estándares y procedimientos con absoluta libertad. Se lleva a cabo cuando se definen pocos procesos y el éxito depende del esfuerzo individual.

Repetible: En el *nivel 2* se establecen los procesos de gestión del proyecto para hacer seguimiento del coste, de la planificación y de la funcionalidad.

Definido: En el *nivel 3* al proceso del software de las actividades de gestión y de ingeniería se lo documenta, se estandariza y se integra dentro de un proceso de software de toda la organización. Todos los proyectos utilizan una versión documentada y aprobada del proceso de la organización para el desarrollo y mantenimiento del software. Se incluyen todas las medidas del nivel anterior.

Administrado: En el *nivel 4* se recopilan medidas detalladas del proceso del software y de la calidad del producto. De manera que las métricas favorezcan el comprender y controlar cuantitativamente tanto los productos como el proceso. Se incluyen todas las medidas del nivel anterior.

Se pueden manejar distintas métricas en el nivel 4 para supervisar y controlar un proyecto de software, como por ejemplo:

- una métrica de prueba puede usarse para determinar cuándo finalizar una prueba de cierto elemento de un código.
- una métrica de legibilidad puede usarse para algún documento en lenguaje natural.
- una métrica de comprensión del programa puede utilizarse para proporcionar un dato numérico que los programadores no puedan cruzar.

Optimizado: El *nivel 5* trata la optimización mediante una retroalimentación cuantitativa del proceso, ideas y tecnologías innovadoras que posibilitan una mejora del mismo. Se incluyen todas las medidas del nivel anterior.

Para su aplicación se puede hacer "evaluación interna" o aplicar el "SCAMPI"[16].

El estándar SW-CMM fue propuesta por el SEI en el 1993. [Oktaba, 2004] El Modelo de Capacidad de Madures del Software (Capability Maturity Model for Software) ha sido retirado y reemplazado por el CMMI. Establecía un conjunto de buenas prácticas las cuales son: definición de un proceso documentado; proveer a la organización de los medios y formación necesarios; prácticas ejecutadas de un modo sistemático, universal y uniforme; aplicación de medidas y verificaciones.

El SEI además de proponer el modelo SW-CMM también desarrolló el SE-CMM, SA-CMM, P-CMM, IPD-CMM. Finalmente propuso migrar desde el 2003 al 2005 al CMMI-SW.
Donde las abreviaciones significan lo siguiente:
SE-CMM: Systems Engineering Capability Maturity Model
SA-CMM: Software Acquisition Capability Maturity Model
P-CMM: People Capability Maturity Model
IPD-CMM: Integrated Product Developmen

2.3.14. CMMI Integración del modelo de madurez de la capacidad

El estándar CMMI (Capability Maturity Model Integration) fue propuesto por el SEI en el 2002. [Oktaba, 2004] Este modelo se desarrolla en base a los conceptos del CMM, en particular al Capability Maturity Model for Software (SW-CMM). El cuerpo de conocimientos incluyen la ingeniería de sistemas (SE), ingeniería de software (Software engineering - SW), productos integrados y desarrollos de procesos (Integrated product and process development - IPPD), y Supplier Sourcing - SS. Además del CMMI para desarrollo (Development – CMMI DEV) que va por la Versión 1.2, hay otros 2 nuevos trabajos para la comisión del CMMI que son el para Servicios (Services – CMMI SVC) y adquisiciones (Acquisition – CMMI ACQ). Los títulos que tiene el CMMI-AM Versión 1.1 (Acquisition Module) son los siguientes:

- Tables of Contents
- Executive Summary
- 1 Introduction
- 2 Executive Questions
- 3 Acquisition Process Areas
- 4 Generic Practices
- Appendix Improving Acquisition Practices
- Bibliographi

y trata sobre la eficiencia y la eficacia en los proyectos de adquisición.

El CMMI provee:

- Guía para la eficiencia y la eficacia en el desarrollo de disciplinas de procesos múltiples en una organización
- Mejoras improvenments para la incorporación de las mejores prácticas desde los modelos tempranos
- Una visión integrada del desarrollo para todos los elementos de una organización

Los beneficios de trabajar con este estándar son los que se mencionan:

- Incrementa la productividad
- Provee un ciclo de tiempo
- Provee un esquema a la planificación
- Provee calidad (como medir los defectos)
- Incrementa la satisfacción del cliente
- Incrementa la moral de los empleados

- Incrementa el retorno a la inversión

A nivel nacional a este estándar se la conoce como norma IRAM 17601.

2.3.15. Comparación de las Normas

Podrían compararse las normas ISO con CMM. Pero si se estudia como mercado a cual exportar la norma ISO sería conveniente para el europeo mientras que la CMM es mejor para el norteamericano.
El nivel III de CMM es equivalente al estándar que pretenden la norma ISO 9001. La norma Spice es una opción conveniente para incorporarla a pequeñas empresas, igual que conseguir el Nivel 2 de CMM.

Una crítica habitual que se les hace es que en el ámbito de la mejora de los procesos es que por fijar la atención en los medios se olvidad del fin. También que las normas reducen la autonomía de los desarrolladores y establecen restricciones a la creatividad. Pero en la práctica se ha observado que los procesos maduros con un esfuerzo colaborativo aumentan la eficiencia y la eficacia.

Los principales modelos de la calidad además de ISO 9000, ISO 12207, ISO 15504 (SPICE) Y CMM para el proceso de desarrollo en la ingeniería de software son TickIT, Boostrap y Trillium. [Piattini, 2003]

a) TICKIT

En el Reino Unido se creó en 1991 el programa TickIT como respuesta a las críticas emitidas por organizaciones con respecto a la calidad del software y al hecho de que había una fuerte resistencia para la certificación ante la norma ISO 9001 porque se la consideraban demasiado general. El objetivo buscado con este programa era ayudar a las organizaciones desarrolladoras de software a crear sistemas de calidad que agregaran valor a sus empresas y que cumplieran con la norma ISO.

El programa TickIT se completa en la ISO 9001:2000 y exige que al evaluar la conformidad con dicha norma se tome en consideración la guía ISO 9000-3:2004. Por otra parte, también se agregan requisitos administrativos para el organismo de certificación acreditado en relación con la calificación del auditor y una nueva evaluación completa cada tres años. También se observan las definiciones de los procesos del ciclo de vida del software publicados en la ISO/IEC 12207.

Las certificaciones ante TickIT han mostrado un rápido crecimiento. Recientemente se expidieron cerca de 1400 nuevos certificados, de los cuales el 50% fueron al exterior del Reino Unido. Actualmente, alrededor del 80% de los certificados pertenecen a empresas localizadas en el Reino Unido, y el resto pertenece a Europa continental y Norteamérica. En octubre de 2005 había 1093 certificaciones de las cuales 807 (73.8%) son del Reino Unido.

Los objetivos principales de TickIT son desarrollar un sistema de certificación que sea aceptable en el mercado y estimular a los desarrolladores de software a implementar sistemas de calidad, dando la dirección y guías necesarias para tal efecto. El objetivo de la certificación es demostrar que las prácticas necesarias para el aseguramiento de la calidad durante el desarrollo de software existen y son verificables.

El modelo TickIT promueve estimular a los desarrolladores de software a pensar acerca de que la calidad existe en el contexto de los procesos de desarrollo de sistemas, de cómo puede ser lograda la calidad, además en la manera en que los sistemas de información de la calidad pueden ser mejorados en forma continua. Se estableció un método para la organización, con reglas y procedimientos para un esquema de certificación del sector que trabaja en la industria de la tecnología de información (productos software, productos que incluyeran software, y sistemas de software que facilitan la prestación de servicios).

La guía de TickIT provee el enlace necesario para que la conformación, del sistema auditado respecto al modelo TickIT, pueda ser expresada en función de los criterios de la ISO 9001, logrando así la aplicación de esta última al desarrollo de software. Los requisitos de competencia y conocimientos que se les piden a los auditores hacen posible la aplicación del modelo gracias a la formación y habilidades adecuadas que le dan confianza, suponiendo que la experiencia y conocimientos de los auditores no se vean obsoletos frente a las prácticas y técnicas nuevas dentro del medio de desarrollo de software.

Esta guía TickIT Versión 5.0 de enero de 2001 está compuesta por las siguientes partes:
Parte A. Introducción al TickIT y el proceso de certificación
Parte B. Guía para los Clientes
Parte C. Guía para los Distribuidores
Parte D. Guía para los Auditores
Parte E. Requerimientos del Sistema de Administración de la Calidad del Software - Perspectivas estándares
Parte F. Requerimientos del Sistema de Administración de la Calidad del Software - Perspectiva del Proceso
Apéndice 1: Administración y valoraciones (assessment) de los procesos TI
Apéndice 2: Caso de estudio: Uso del Modelo de Excelencia EFQM
Apéndice 3: Caso de estudio: ISO/IEC TR 15504 – Valoraciones(assessment) de Procesos Compatibles
Apéndice 4: Caso de estudio: Mejora(improvement) del Proceso de Software el camino CMM-SW
Apéndice 5: Referencias estándares
Glosario de términos
Índice de términos

b) BOOTSTRAP

El modelo Bootstrap fue creado por la Comisión Europea como parte del programa ESPRIT (European Strategic Program for Research) con el interés principal de evaluar y mejorar la

capacidad de los procesos de desarrollo de software. La metodología ha sido desarrollada conforme el estándar ISO para el seguimiento del proceso como el SPICE y la metodología ISO 12207 de los procesos del ciclo de vida del software. Esta metodología mediante prácticas, herramientas y estándares de calidad internacional mide, evalúa y propone mejoras al proceso de desarrollo de software que siguen las Unidades de Producción de Software (SPU, Software Producing Units) de las empresas.

Los principios del Instituto Bootstrap sostienen que la metodología sea accesible a todos y crezca de forma que permita mejoras. Se convierta en un servicio a la industria Europea y opere como una empresa no lucrativa. Se puede decir que el Instituto tiene como objetivos la mejora continua de la metodología para la valuación de la calidad de los procesos de desarrollo de software (tomando en cuenta los estándares relevantes internacionales en el área incluyendo la forma de distribuirlo y el material de entrenamiento), la promoción para ampliar su cobertura en la industria europea de manera de consolidarse como estándar, la licencia a terceros, el manejo apropiado de las bases de datos de resultados de las evaluaciones llevadas a cabo por asesores certificados, la certificación de los asesores y la certificación de las organizaciones evaluadas.

Las principales actividades del Instituto son medir el estado actual de la práctica de desarrollo de software, buscando evaluar a empresas y planificando la acción, capacitar en la metodología y mejoras de la misma, certificar a los asesores, recolección y administración de los datos de las evaluaciones, definir mecanismos para mantener la confidencialidad de los datos, representación en otros trabajos de estandarización, coordinación de evaluaciones multinacionales y foro para obtener licencias y asesores independientes.

El modelo se basa en los siguientes puntos según la versión 3.2: organización (ingeniería de negocios, gestión de recursos humanos y gestión de infraestructura), metodología (funciones del ciclo de vida dependiente, funciones del ciclo de vida independiente y funciones relacionadas al proceso) y tecnología (innovación tecnológica, soporte tecnológico para los procesos del ciclo de vida, soporte tecnológico para los procesos independientes del ciclo de vida e integración de herramientas). Entre las funciones del ciclo de vida dependiente tenemos: el modelo de desarrollo, análisis, diseños, implementación, pruebas y mantenimiento; de las funciones del ciclo de vida independiente consideramos: administración de la configuración, de riesgos, de proyectos, de calidad y de subcontratistas; y como funciones relacionadas al proceso consideramos la descripción, la medición y el control de procesos.

Los modelos de procesos BOOTSTRAP definen procesos y niveles de capacidad como los siguientes:
Nivel 0: Incomplete Process
Nivel 1: Performed Process
Nivel 2: Managed Process
Nivel 3: Established Process
Nivel 4: Predictable Process
Nivel 5: Optimising Process

Hay otros estándares internacionales más como por ejemplo TRILLIUM y JTC 1 (Joint Technical Commitee) para el software y también modelos de aplicación más amplia como por ejemplo Six Sigma.

2.3.16. Aseguramiento de la calidad del software

La garantía de la calidad del software es una actividad de protección según la Gestión de Calidad del Software (SQA Software Quality Assurance) y se debe aplicar a lo largo de todo el proceso de software y abarca los siguientes puntos:

- Un enfoque de gestión de la calidad.
- Tecnología de ingeniería del software efectiva (métodos y herramientas).
- Revisiones técnicas formales que se aplican durante el proceso del software.
- Una estrategia de prueba en múltiples niveles.
- El control de la documentación del software y de los cambios realizados.
- Un procedimiento que asegure un ajuste a los estándares de desarrollo del software
- (cuando sea posible).
- Mecanismos de medición y de generación de informes.

Un plan de la Gestión de Calidad del Software proporciona un mapa para institucionalizar la garantía de la calida. El IEEE en 1994 ha recomendado un estándar donde se describen el propósito y el alcance del proyecto, los documentos del plan y también una sección para la gestión que describe la situación dentro de la estructura organizativa, las tareas y actividades a lo largo del proceso, tanto como las funciones y responsabilidades organizativas respecto de la calidad del producto.
Cuando describe la documentación incluyen:

- Documentos del proyecto (por ejemplo el plan del proyecto)
- Modelos (por ejemplos el Diagrama de Entidad Relación DER, o Jerarquía de Clases)
- Documentos Técnicos (por ejemplo planes de prueba)
- Documentos del usuario (por ejemplo archivos de ayuda)

Además de un conjunto de productos de trabajo que se pueden aceptar para lograr alta calidad.

2.3.17. Factores que afectan la calidad en una Pyme productora de software

Se pueden mencionar algunos factores que afectan la calidad en una Pyme productora de software. Estos son [Piattini, 2003]:

a) Los recursos humanos: son la base para sustentar una propuesta de calidad y muy especialmente en esta ingeniería donde el objetivo es crear programas. Un concepto que tiene importancia es la carrera profesional, ya que el sistema educativo forma profesionales con una competencia específica. También hay que considerar el perfil de la persona, un buen

desarrollador no necesariamente es un buen líder. Así como es importante que el personal sea adecuado también es importante que sea estable.

b) Infraestructura: es necesario que la organización tenga una instalación propia, que sea adecuada y optimizada para el trabajo que realiza. Estos incluyen a las computadoras de escritorio, los servidores, las líneas y dispositivos de comunicación, el equipamiento de la oficina, etc. Las actualizaciones de los equipos deben llevarse a cabo en un tiempo aceptable además de disponer de soporte para el correcto funcionamiento de los mismos. Si bien está claro que hace falta inversión en infraestructura también se debe mencionar que puede ser el aspecto diferenciador.

c) La formación: debido a lo competitivo del sector y lo cambiante que es, se busca una continua formación del personal para mantener el nivel que se busca. Hay muchos cursos en el mercado y en alguna medida también está la opción de la autoformación.
Metodología: apunta al concepto de organización y la secuenciación de los trabajos. Nos debe decir que hacer y con que contenido. Es necesario que se aplique con sensatez, para que con las tareas asociadas se mantengan controlados los costos y el tiempo. Los documentos que se generen deben ser útiles, es un aspecto importante para gestionar.
Arquitectura: al principio la arquitectura era centralizada, después se empezó a usar Cliente-Servidor y actualmente aparecieron modelos mas globales y distribuidos debido especialmente a Internet. Actualmente se busca desarrollar aplicaciones multiplataforma con un único producto (lenguaje) de manera de poder encarar diferentes clientes y negocios. La arquitectura está asociada a la normalización para trabajar con objetivos concretos de calidad.

d) La gestión: es el conocimiento y administración del negocio; o sea, es la planifica y seguimiento de este. Cuando la planificación es realista y se realiza un seguimiento de las tareas, entonces se tiene control sobre el desarrollo. Una técnica que ayuda a la medición de una aplicación es Puntos de Función, por ejemplo para estimar el tamaño.

2.4. El efecto del cambio en los programadores

La actitud de los recursos humanos frente al cambio de la forma en que se desarrollan los sistemas. Diferentes personas tienen distintos estilos para interactuar con otras y entender los problemas que surgen durante el trabajo. Algunas personas prefieren hacer un análisis detallado antes de tomar una decisión mientras que otras son más intuitivas. También se puede pensar en el estilo de trabajo preferido en términos de dos componentes. Uno es la forma en la cual se comunican los pensamientos y se reúnen las ideas, el otro es el grado en que las emociones afectan la toma de decisiones. Los primeros son los llamados *extrovertidos* mientras que los segundos son los *introvertidos*. Además hay personas *intuitivas* que basan sus decisiones en sentimientos y reacciones emocionales frente a un problema, los *racionales* deciden principalmente examinando los hechos y considerando cuidadosamente todas las opciones. Por lo tanto podemos describir un conjunto de estilos de trabajo considerando la personalidad de los recursos humanos. [Pfleeger, 2002]

El personal debe estar organizado en equipos eficientes de manera de maximizar las capacidades y habilidades de cada uno, motivados para hacer el trabajo con alta calidad según sus capacidades y coordinados para lograr una comunicación eficaz. En particular el líder del equipo debe tener capacidad de resolución de problemas diagnosticando los conflictos técnicos y organizativos además de buscar la solución para los mismos, aplicando las soluciones aprendidas de la experiencia de otros proyectos, y manteniéndose flexible para hacer los cambios necesarios para llegar a la solución de problemas; debe tener dotes de gestión para dirigir proyectos, mostrando confianza para asumir el control cuando es necesario y la seguridad de delegar responsabilidades en los profesionales y técnicos; manejar incentivos para recompensar la iniciativa y los logros como también que la toma de riesgos controlada no será penalizada; y fomentar la cultura de equipo, influenciando para resolver en conjunto los problemas con la mayor cooperación posible además de mantener el control en situaciones de alta tensión emocional. [Pressman, 2005]

El *ámbito del cambio* de la informática en los últimos 50 años ha sido promovido y controlado por los avances en las ciencias experimentales duras (física, química, ciencia y tecnología de los materiales). Durante las próximas décadas, los avances en la informática serán regidos por las ciencias suaves no experimentales (sicología humana, biología, neurofisiología, sociología, filosofía y otras). También los cambios que afectarán a la ingeniería del software próximamente se verán influenciados por cuatro fuentes que son: las personas que realizan el trabajo, el proceso que aplican, la naturaleza de la información y la tecnología informática empleada. [Pressman, 2005].

Las personas que construyen el sistema y la forma en que lo construyen son un tema de estudio. Debido a que aumenta el tamaño del software, también aumenta la cantidad de profesionales dedicados a realizar esta tarea. La experiencia demuestra que cuánto más personas forman un equipo la productividad del mismo sufre. Un método para mejorar esta situación es dividirlos en equipos, pero se requiere que los mismos estén comunicados. Para favorecer una mejor comunicación los ingenieros disponen de nuevas herramientas como son el correo electrónico, los grupos de noticias, y los sistemas de videoconferencias en una red de información que cada vez será más inteligente. Internet permite la suscripción a un grupo de noticias centrándose en un área tecnológica y una consulta podrá ser respondida desde cualquier lugar del globo. La World Wide Web es y seguirá siendo la biblioteca más grande del mundo. La cultura del desarrollo del software cambiará significativamente en el futuro próximo.

El proceso que se aplica en sistemas complejos es iterativo e incluso a veces incremental. Antes se lo consideraba de pensamiento lineal. Como los modelos de procesos evolutivos reconocen que la incertidumbre domina la mayoría de los proyectos se hace hincapié en productos de trabajo incrementales, planificación, análisis de riesgo y realimentación por parte del cliente. El modelo de madurez de capacidad del SEI ha tenido un gran impacto en la ingeniería de software. El MMC proporciona buenos principios para trabajar en software. Las tecnologías de objetos pueden llevar a una tendencia hacia los procesos de modelos evolutivos. Con la reutilización de componentes y la aplicación de herramientas CASE para el uso de prototipos de una aplicación permitirá construir soluciones más rápidamente. El aumento de las aplicaciones Web (WebApps) está creciendo. Los equipos que trabajan en esta área muchas veces son interdisciplinarios. Es software que ha surgido de productos Web a favorecido tanto cambios económicos como culturales. El proceso de la ingeniería de software deberá adaptarse para trabajar en la Web. Los

equipos ágiles trabajan en pequeños grupos muy motivados, destacándose la competencia individual y la colaboración de todos como factor de éxito del equipo, son autoorganizados.

En los proyectos para desarrollo de software modernos los equipos de ingeniería de sistemas deben establecer métodos eficientes para coordinar al personal que realiza el trabajo. Para eso es necesario establecer mecanismos para la comunicación formal e informal. La comunicación formal se logra por medio de escritos, reuniones estructuradas y otras formas de comunicación impersonal. La comunicación informal es más personal, los integrantes del equipo comparten ideas, piden ayuda e interactúan diariamente.

La ingeniería de software va a cambiar, mas allá de lo radicales que sean los cambios seguramente la calidad nunca perderá su importancia, y el análisis y diseño efectivos siempre tendrá su lugar en los sistemas basados en computadoras.

3. Descripción del Problema

3.1. Contextualización del problema

Se puede decir que en los últimos años se han producido un conjunto de transformaciones económicas y sociales vinculadas a la sociedad el conocimiento. Dentro de ellas están las tecnologías de la información y las comunicaciones (TICs) que han generado profundos cambios tanto en el mundo desarrollado con en los países en desarrollo. El sector de software y servicios informáticos (SSI) es un segmento clave dentro de las TICs que ha venido creciendo más que el hardware y se prevé que va a continuar así en el futuro. Los países desarrollados son los principales productores y consumidores pero se abren posibilidades que pueden ser aprovechadas por otros como lo demuestran los casos de la India (que comenzó con la exportación de servicios y continuó hacia la programación y los servicios offshore), Irlanda (gracias a la Comunidad Económica Europea) e Israel (porque tiene como nicho el software orientado a la seguridad, las tecnologías antivirus y militares). La economía Argentina puede enfrentar el desafío de ingresar a la economía basada en el conocimiento gracias a las buenas condiciones que tiene como lo demuestra el hecho de que hace varias décadas que el sector SSI viene creciendo. Para eso es necesario encontrar estrategias que le permitan penetrar en los mercados.

Los estudios realizados del sector SSI en la Argentina demuestran que ha estado creciendo en los últimos 40 años de manera continua lo que ha llegado a definir un lugar en la economía nacional. Empieza en la década del 60 y se desarrolla a base de una estrategia basada marcadamente en el mercado interno, la cual continúa así por las décadas siguientes. A mediados de los 80, se producen los primeros diagnósticos sobre la situación actual que mostraba el predominio del uso de software importado, como es por ejemplo el software de base en especial los sistemas operativos y los utilitarios, y un cierto grado de desarrollo como en los casos de software aplicativo para fines administrativos, contables e impositivos. Entonces, habían unas 300 firmas de SSI de las cuales 200 realizaban desarrollos de software (en algunos casos embebidos en distintas clases de equipos). Para la década de los 90 había unas 300 empresas dedicadas a la producción y distribución del software con unas 3000 personas vinculadas a la actividad, además de 1500 en la provisión de servicios. Ya en el 2002 los trabajos más recientes demuestran que había más de 500 empresas en el sector de producción que facturaban más de U$S 500 millones y que llegaban a U$S 750 millones si se consideraban la licencias por ventas de productos extranjeros. Estimaciones más recientes indican que las ventas del sector rondarían los U$S 1100 (caracteriza alrededor del 0,7 % del PBI).

La caracterización del sector SSI en Argentina considerándola como oferta local está dividida en tres grupos:

- Un pequeño número de empresas de gran tamaño, mayormente de capital extranjero, dedicadas principalmente a la comercialización de productos y a la prestación de servicios informáticos. Los paquetes de software que ofrecen son para grandes clientes incluyendo al Estado.
- Un pequeño número de empresas de tamaño mediano que desarrollan software y servicios informáticos principalmente para el área de gestión empresaria.

- Un numeroso conjunto de empresas de tamaño pequeño, de capitales locales y de poco tiempo en el mercado. Se dedican a la producción de software como así también a la provisión de servicios informáticos. Se mantienen atendiendo demandas variadas por la dificultad de encontrar nichos sustentables de especialización.

[Foro de Software y Servicios Informáticos, 2004]

También hay que considerar los inconvenientes propios de la industria del software que debe encontrar una solución a los problemas que plantean las siguientes preguntas:
¿Por qué lleva tanto tiempo terminar los programas?
¿Por qué son tan elevados los costos de desarrollo?
¿Por qué no se pueden encontrar todos los errores antes de entregar el software a nuestros clientes?
¿Por qué nos resulta difícil constatar el progreso conforme se desarrolla el software?
La ingeniería del software en busca de la calidad viene a responder a estas preguntas con modelos de desarrollo que deben ser estudiados y utilizados.

A raíz de esto conviene tener en cuenta que el software es un producto de la lógica, no restringido por las leyes de la física o por los límites de los procesos de fabricación del hardware. Como es abstracto, entonces también su calidad lo es. Por lo tanto se pueden considerar algunas características del software:

- Se desarrolla, no se fabrica. El costo está fundamentalmente en el proceso de análisis y diseño, no en el desarrollo. Y por lo tanto los errores se introducen mayormente en las etapas previas a la producción.
- El software no se estropea con el tiempo. No es susceptible a los efectos del entorno como los desarrollos físicos, y la curva de fallos es distinta a la del hardware. No se generan nuevos errores pero el software se deteriora debido a los cambios. Es engañosamente fácil realizar modificaciones sobre un software, pero los efectos de estos cambios se pueden propagar de forma explosiva e incontrolada.
- Es artesanal en gran medida. El software se construye en general a medida, en vez de ser construido ensamblando componentes existentes y ya probados. Esto dificulta más el control de la calidad. El software se debería diseñar y también implementarse para poder volver a reutilizarlo. [Pressman, 2005].

Existe un conjunto de tareas genéricas para la construcción de software las cuales son:

- Construir la infraestructura arquitectónica (que incluye codificar y probar los componentes arquitectónicos, adquirir patrones arquitectónicos reutilizables y probar la infraestructura).
- Construir un componente del software (que viene a ser crear una serie de pruebas de unidad, codificar las estructuras de datos y la interface del componente, codificar los algoritmos internos y las funciones de procesamiento).
- Realizar pruebas de unidad al componente y corregir los errores descubiertos.

- Integrar el componente terminado a la infraestructura arquitectónica.

[Pressman, 2005]

3.2. Delimitación del problema

La Argentina debe buscar su propio camino para el desarrollo del sector SSI y en particular el de la industria productora de software. Para eso se debe tener en cuenta las experiencias de los países de ingreso tardío, sin intentar copiarlas. Las características propias culturales, sociales, económicas, científicas y tecnológicas, sumado a las circunstancias externas pueden dar la posibilidad para acceder a los mercados internacionales, para lo cual será necesario detectar las oportunidades concretas que existen en aquellos ámbitos. La definición de un modelo de desarrollo puede ser el resultado de los encuentros entre los distintos sectores involucrados como el empresario, el Estado y las instituciones educativas y de investigaciones científicas tecnológicas. Más allá de esto, difícilmente será algo más que la definición de un objetivo deseable y factible, pero nunca deberá apuntar a cerrar actividades de ningún tipo de emprendimientos. A su vez el crecimiento de la industria irá determinando nuevas oportunidades, analizar amenazas y además de cómo seguir con las líneas de avance en las cuales se esté trabajando. En principio, el modelo debe orientarse a productos de aquellos segmentos específicos donde se poseen competencias claves, las que a su vez puedan ser reconocidas internacionalmente y al mismo tiempo son una oportunidad en cuanto a los requerimientos comerciales y tecnológicos locales.

Se han detectado una serie de problemas en el desarrollo de software que pueden ser tenidas en cuenta por la Pymes argentinas dedicadas a esta actividad:

- Falta de capacitación de los técnicos que llevan a cabo la construcción de programas sin fundamentos metodológicos. Además de profesionales de otras disciplinas que trabajan como informáticos debido a que hay cargos mejor remunerados que necesitan ser ocupados en diversos proyectos y ellos no están adecuadamente preparados para atender los problemas informáticos que se les presentan.
- Se debe poner un interés especial en la configuración de los perfiles de los diferentes profesionales del equipo de desarrollo para que cubran satisfactoriamente los objetivos propuestos.
- Algunos desarrollos de sistemas en la actualidad no se implementan con las metodologías provistas por la ingeniería de software y por ese motivo los resultados son que no se terminan en tiempo y forma, o deben hacerse modificaciones significativas. Motivo por el cual se justifica el ciclo de vida para la construcción de un sistema de software. [Cataldi, 2000]

Se desea presentar la problemática que responde al problema de la Pyme Argentina productora de software. Dentro del contexto de posicionar a la Argentina como productora de software de alta calidad con la creación de nuevas empresas especializadas en informática. Para eso se deben construir cadenas de valor multiempresas para la oferta exportable y focalizar el desarrollo en

sectores definidos. Entre las fortalezas está el hecho de que contamos con recursos profesionales capacitados y actualizados (en gran parte universitarios), se produjo una concientización inicial sobre el tema de los estándares de calidad con dificultades para la implementación por los costos y las posibilidades de la exportación por ventajas cambiarias y de infraestructura.
[Foro de Software y Servicios Informáticos, 2004]

Finalmente se considera que cualquier Pyme que produzca software deberá utilizar los paradigmas de diseño y las metodologías correspondientes que ofrece la ingeniería de software poniendo interés en el nivel de calidad con el que trabajan. Para eso un riguroso y sencillo esfuerzo debe aplicarse para el seguimiento del trabajo realizado de manera de llegar a cumplir con los objetivos buscados.

4. Solución propuesta

4.1. Entrevistas realizadas

Se realizaron una serie de entrevistas para conocer la opinión de profesionales con años de experiencia en proyectos de sistemas de información. Con el objetivo de aprovechar el saber que adquirieron de sus trabajos que sea útil para el modelo de protocolo propuesto. Un resumen de estas entrevistas se transcribe en las siguientes líneas:

Entrevista 1

Dra. Lic. Catalina Mostaccio
Lugar: La Plata
Fecha: 24/05/2007
Horario: de 09:10 a 10 hs.

Durante la entrevista se conversó sobre su experiencia como programadora, en particular destacó cuando integró el equipo que participaba en la Facultad de Astronomía de la UNLP (Universidad Nacional de La Plata). Entonces comentó que durante esos años cuando trabajaba ahí verificaban los resultados que daba la corrida del software con una tabla de datos conocidos de manera de tener seguridad con esta prueba de que los programas hacían lo correcto.

Actualmente como docente de la Facultad de Informática de la UNLP enseña y evalúa que un buen software es fácilmente legible, es reutilizable (dividido en módulos), y las condiciones y bucles cumplen con los requisitos. Decía que, cuando empezó, a los alumnos se los formaba en la programación estructurada mientras que ahora se les enseña orientado a objetos.

También mencionó una cartilla para chequear software para la Web que fue desarrollada en la Universidad Nacional de La Pampa. Sabe esto porque está en contacto con el grupo Gidis (Grupo de Investigación y Desarrollo en Ingeniería de Software / ingeniería Web) de la Facultad de Ingeniería que realiza investigaciones y capacitación en ingeniería Web.

Después de la entrevista me envió por e-mail un paper que estudia los atributos de calidad en sitios de e-commerce argentinos. Entre las conclusiones donde comparan el trabajo de investigación actual con otro realizado anteriormente observan un rápido crecimiento en las aplicaciones de correo electrónico para diferentes modelos de negocios. [Olivito,2004]

Entrevista 2

Lic. Fernando Aramburu
Lugar: La Plata
Fecha: 07/06/2007

Horario: de 15 a 15:55 hs.

Durante la entrevista se conversó sobre su experiencia como programador para consultoras, software factory y el Lifia (Laboratorio de Investigación y Formación en Informática Avanzada). Cuando trabajó en el Lifia de la Facultad de Informática de la UNLP junto con la organización Lumina se puso más interés en la calidad del software. Este aspecto de la calidad fue algo que destacó.

Actualmente programa en .Net en un emprendimiento para proveer software internacionalmente. Además prepara los presupuestos para proyectos de sistemas donde estima un cálculo de la cantidad de horas hombres necesarios incluyendo una persona más para la calidad y el costo.

Si bien conoce las normas CMI ya que las estudió no tuvo la ocasión de trabajar con ellas. Sin embargo me comentó que solo algunas empresas grandes la usan como por ejemplo, citó, el caso de IBM.

Las especificaciones para desarrollar el software las obtiene conversando con los clientes (a veces mediante SKY), después arma y envía una minuta como resumen de lo hablado. Cuando realiza reuniones con los clientes se prepara una minuta para dejar registrado lo que se trató en la misma.

Realizan pruebas unitarias de cada módulo una vez terminado. También pruebas unitarias del conjunto de todos los módulos como por ejemplo cada quince días y pruebas de integración previas y con el cliente.

Cuando proponía sus productos decía que son de calidad y baratos, pero le han replicado que el software no es barato sino que es caro, por lo tanto que aclare como garantiza que va a cumplir con lo que dice en tiempo y por lo tanto con el presupuesto acordado. Entonces les habla de los procedimientos que usan.

Considera que las leyes para la promoción de la industria del software son útiles para las grandes empresas y no para las pequeñas.

Entrevista 3

Ma. Ing. Paola V. Britos
Lugar: Buenos Aires
Fecha: 12/07/2007
Horario: de 09:55 a 10:15 hs.

Actualmente es docente en el Centro de Ing. del Software e Ing. del Conocimiento de la Escuela de Postgrado del ITBA (Instituto Tecnológico de Buenos Aires). También se desempeña en auditorías de software.

Durante la entrevista me recordó su experiencia durante el curso de Gestión de la Calidad, Normas ISO 9000 y Aplicaciones de Software dictado para el doctorado de la Universidad Nacional de La Plata.

Le comenté el tema de mi tesis y le mostré el cuestionario que estoy preparando, entonces me hizo las siguientes observaciones:

- Desglosar cada una de las preguntas.
- No está de acuerdo en que el programador participe en la verificación y validación del software aunque la teoría diga otra cosa.
- Usar terminología estándar ("gestión de configuración" y "pruebas de validación")
- Tomar cada actividad de Métrica y preguntarse que debería evaluarse.

Entrevista 4

Analista Prog. Victor Toledo
Lugar: Tolosa
Fecha: 24/08/2007
Horario: de 10 a 11 hs.

Durante la entrevista se conversó sobre su experiencia como analista programador en el Ministerio de Salud de la Provincia de Buenos Aires. También de la sociedad que construyó y resultó ser un emprendimiento para proveer sistemas de información a empresas y otras organizaciones. Además de las actividades de docencia que realiza en el área informática.

En el centro de cómputos de Ministerio las pruebas de software las hacía por cada rama del sistema. Los programas los probaba desde lo macro a lo más chico. El lote de prueba lo obtenía en el relevamiento, por cada operación de entrada o salida trataba de fotocopiar una planilla de estos trámites y si era posible con datos reales.

Cuando realizó su emprendimiento para terceros probaba el sistema en el lugar de la organización que contrató el trabajo.

Conoce las normas ISO para el área de informática y tecnologías de la información porque las estudio un poco pero no trabajó con ellas. Sin embargo me comentó de la experiencia de otro profesional certificado con quien se cruzó laboralmente y que las utilizó.

Respecto del cuestionario que estoy haciendo, le di una copia y me dijo que considerando que estaba orientado a los responsables de Pymes debía ser más sencillo en las entrevistas para que les resulte claro. Si bien el entendía a que me refería debía pensar que hay personas no tan especializadas y que igualmente pueden gestionar el negocio.

Entrevista 5

Analista Prog. Victor Toledo
Lugar: Tolosa
Fecha: 12/09/2007
Horario: de 09 a 10 hs.

Nuevamente entrevisté a Toledo después de haberle enviado por e-mail el cuestionario que estoy haciendo y parte de la documentación de la tesis para ampliarle más la idea de lo que necesitaba. Por ese motivo estuvimos conversando sobre el tema, los objetivos general y específicos como así también de las etapas y actividades del trabajo que estoy llevando a cabo.

Posteriormente señaló las ventajas de seguir una metodología para el desarrollo de software y de las ventajas de haberlo implementado en su experiencia profesional. Además mencionó los proyectos que está encarando actualmente.

Finalmente tratamos el cuestionario destacando la necesidad de mayor claridad en algunas preguntas para que pueda ser entendido por un usuario que no tenga formación en el desarrollo de sistemas de información. También me dijo que como buscaba una respuesta por Si o No (y No Aplica o No Sabe/No Contesta) debía buscar más precisión en las preguntas de manera de evitar que surjan dudas al momento de tomar una decisión.

4.2. Instrumento de toma de datos

ENCUESTA SOBRE LOS RRHH Y LA CALIDAD EN LA PRODUCCIÓN DE SOFTWARE EN ARGENTINA

Esta encuesta tiene como objetivo rescatar información de los recursos humanos, sobre las metodologías usadas y la calidad en la producción de software en especial en las PyMEs Argentinas.
Los datos son confidenciales y el uso estadístico es para un trabajo de tesis de Maestría en la Universidad Tecnológica Nacional – Facultad Regional Buenos Aires.
Marque con una x la/s respuesta/s a las preguntas.

1. ¿En qué área de desarrollo trabajan en la producción de software?

1. Web	
2. Sistemas para usuarios	
3. Juegos para PC	
4. Entretenimiento para celulares	
5. Otra	
6. Ninguna	

2. ¿La organización en la que trabaja está certificada en una norma de calidad?

1. ISO-IRAM	
2. CMM	
3. Otra	
4. Ninguna	

3. ¿Cree conveniente poner énfasis en la calidad cuando se desarrolla software?

1. Si	
2. No	

¿Por qué?

……

……

4. ¿Aplican alguna métrica en su organización para evaluar la planificación, el desarrollo y/o el mantenimiento de software?

1. COCOMO	
2. Puntos de Función	
3. Orientadas a clase	
4. Otra	
5. Ninguna	

5. ¿Qué metodología prefiere usar cuando desarrolla un sistema?

1. Orientada a Objetos	
2. Estructurada	
3. Funcional	
4. Otra	
5. Ninguna	

6. ¿Realizan encuestas para evaluar al personal?

1. Periódicamente	
2. Ocasionalmente	
3. Nunca	

7. ¿Realizan encuestas para conocer la opinión del cliente?

1. Periódicamente	
2. Ocasionalmente	
3. Nunca	

8. ¿Cuál cree que es el futuro de las Pymes productoras de software en la Argentina?

1. De mucho crecimiento	
2. De crecimiento continuo	
3. De poco crecimiento	
4. Decrecimiento	
5. Ninguna	

9. ¿Qué es lo que más valora del profesional de sistemas?

1. Capacidad	
2. Trabajo en equipo	
3. La actitud para encarar un desafío tecnológico	
4. La manera en que ve el negocio desde el área TI	

10. ¿Cuál piensa es la perspectiva para la ingeniería de software?

1. De profundos cambios	
2. De cambios relativos	
3. Sin cambios significativos	

¿Por qué?

. . . . . .

. . . . . .

11.¿Documentan el análisis, diseño y/o construcción de aplicaciones?

1. Totalmente	
2. Ampliamente	
3. Parcialmente	
4. Nada	

12. En caso de que sea afirmativa la respuesta ¿Utilizan una herramienta en su organización para documentar los sistemas?

1. Vicio	
2. Word	
3. Otra	
4. Ninguna	

13. ¿Qué herramientas usan en la organización en que trabaja?

1. Lenguajes de programación	
2. Sistemas aplicativos	
3. Software de ingeniería y/o científico	
4. Data Warehouse	
5. Otros	

14. ¿Usan la red para comunicación interna?

1. Utilizan correo electrónico	
2. Poseen acceso a Internet	
3. Poseen página Web con información institucional	
4. Ninguna	

Observaciones/Comentarios:

……
……
……

Gracias por su colaboración!!!

4.3. Justificación de las preguntas

1. Justificación: para estudiar la influencia del área de desarrollo en la encuesta y si hay mucho peso de un sector en esta última.

2. Justificación: para determinar si la organización donde trabaja el encuestado está certificada en una norma de calidad y estudiar que porcentaje está incorporado en alguna de ellas.

3. Justificación: pregunta cerrada relacionada directamente con el objetivo de la tesis de esta maestría referente a la calidad en la producción de software.
   Justificación: es una pregunta abierta para que desarrolle el tema con libertad y conocer la opinión del encuestado de manera de obtener más información sobre el tema de la calidad.

4. Justificación: para conocer el uso y la importancia que se les da a las distintas métricas.

5. Justificación: pregunta relacionada directamente con el objetivo de la tesis que busca conocer la preferencia de los métodos propuestos.

6. Justificación: permite conocer un poco más si la empresa u organización se interesa por sus RRHH siendo la persona un capital importante en lo referente a la producción de software.

7. Justificación: como el concepto de calidad está vinculado con la satisfacción del cliente, conocer su opinión muestra el interés puesto en la calidad de manera más objetiva.

8. Justificación: pregunta relacionada directamente con el objetivo de la tesis de esta maestría que busca conocer la opinión del encuestado.

9. Justificación: es para saber que es lo que más se valora de los RRHH en una organización.

10. Justificación: para conocer la opinión del encuestado respecto del futuro de la ingeniería de software de manera de percibir cambios que pueden darse.
    Justificación: es una pregunta abierta para que desarrolle el tema con libertad y conocer su opinión del futuro de la ingeniería de software.

11. Justificación: para conocer el grado de organización que se aplica en la empresa.

12. Justificación: para conocer el grado de tecnificación de la empresa.

13. Justificación: para conocer cuales herramientas son más usadas en las empresas y organizaciones.

14. Justificación: para conocer la conducta tecnológica de las empresas en Argentina.

4.4. Resultados obtenidos

Para hacer un análisis de las respuestas se tomarán los ejes siguientes:

- La calidad y los temas asociados para obtener un producto de exportación
- El sistema de documentación y las aplicaciones usadas
- Las formas de estimación de los parámetros del proyecto
- Los recursos humanos y su calificación

Respecto de la pregunta número 3: *¿Cree conveniente poner énfasis en la calidad cuando se desarrolla software?,* se puede observar a través de la Figura 1 que el 90 % responde en forma afirmativa considerando a la calidad como un factor esencial en el desarrollo.

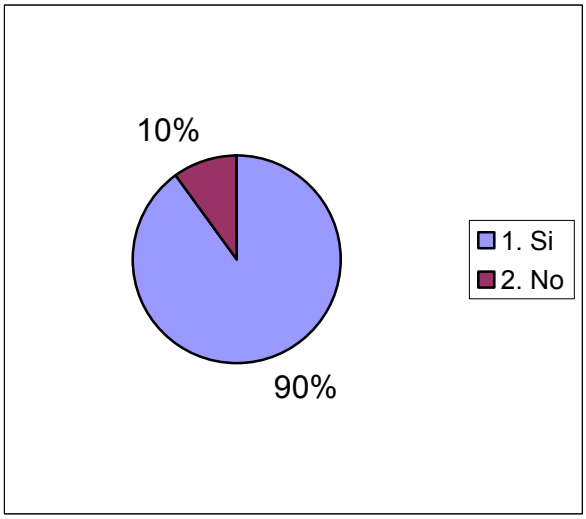

Figura 1: Importancia que se le da a la calidad en la producción de Software

Al analizar las respuestas a la pregunta abierta *Por qué?* justifica su elección muestra los siguientes casos de once encuestados.

1. *Porque la calidad también hace al producto terminado*
2. *Porque de esa manera se puede competir internacionalmente con otras software factory. Es la única forma de asegurar la calidad del producto entregado.*
3. *Una razón es que facilita muchísimo el mantenimiento de errores y desarrollo nuevo.*
4. *Si por calidad se refiere a la calidad del producto, la respuesta es ciertamente si, por razones obvias de competencia en el mercado. Si por calidad se refiere a cumplir con los procedimientos de certificación, la respuesta depende de quienes resulten los clientes finales; para el caso en que los productos resulten evaluados por gerentes de corporaciones, el producto deberá estar certificado, dado que es la nueva moda entre consumidores corporativos, en cambio si el producto será evaluado en función de resultados (clientes finales o evaluadores independientes) no hace falta certificarlo y se puede competir más agresivamente en precio y tiempo de respuesta.*
5. *Mejora la performance de los recursos (humanos, procesos, etc.)*
6. *Porque al desarrollar software de calidad es mucho más sencillo escalar las aplicaciones desarrolladas ya sea a nuevas versiones de lenguajes, sistemas operativos o incluso reutilización de código fuente para nuevos desarrollos.*
7. *Si por que genera una credibilidad mayor en los productos, y un mejor mantenimiento pre y post venta.*
8. *La calidad es fundamental en cualquier herramienta, en este caso particular, el software. Permite cumplir con las necesidades del usuario, teniendo en cuenta aspectos de escalabilidad, usabilidad, performance, etc.*
9. *Aplicar normas de Calidad llevan a normalizar los procesos, eso deviene en la necesidad del management y es lo que le da valor agregado al producto. Entonces la producción Argentina podría dejar de ser solo copia de productos para ponerse a innovar o descubrir.*
10. *Para poder exportar y asimismo cumplir con la Ley de promoción de software*
11. *Se debe brindar al usuario un servicio eficiente que cumpla con sus expectativas, facilite su tarea diaria y el margen de error sea el mínimo.*

Esto da una muestra de que la calidad es esencial en el desarrollo del software para ingresar a mercados internacionales. Esta idea está instalada en la PyMEs. Pero al observar las respuestas a la pregunta 2 :*¿La organización en la que trabaja está certificada en una norma de calidad?*
Solo un 30 % están certificadas por ISO-IRAM ó CMM, con un 50% sin certificar, lo que es un valor extremadamente grande como se ve en la Figura 2.

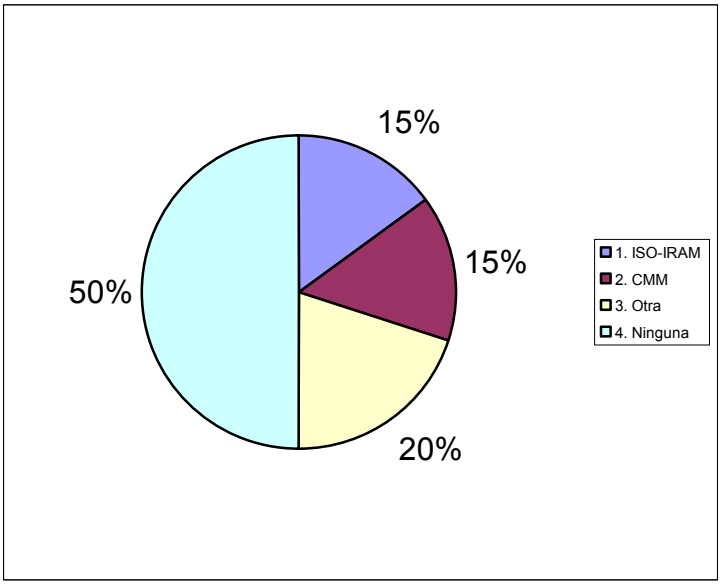

Figura 2: Certificaciones seleccionadas

Ahora bien, de acuerdo a la pregunta 1: *En qué área de desarrollo trabajan en la producción de software?* El software desarrollado se centra en aplicaciones para usuarios y para la web como se observa en la Figura 3 representando un 74%.

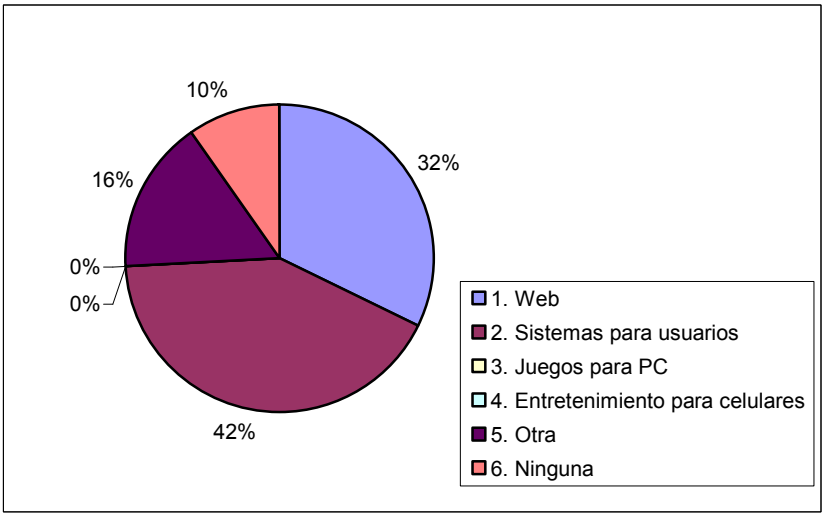

Figura 3: Áreas de desarrollo en la producción de software

Las metodologías más usadas resultan ser las orientadas a objetos con un 57 %, que se evidencian a partir de la respuesta a la pregunta 5: *¿Qué metodología prefiere usar cuando desarrolla un sistema?* Como se ve en la Figura 4.

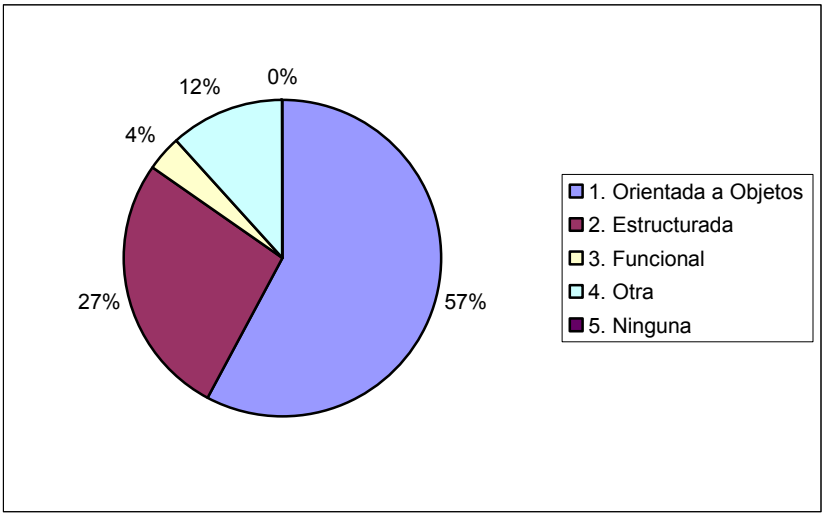

Figura 4: Metodologías preferidas por los profesionales

Las etapas previas de planificación desarrollo y mantenimiento no están estimadas a través de métricas. Se basan mayormente en la intuición y experiencia para prever los tiempos lo que la experiencia demuestra no es muy acertado.

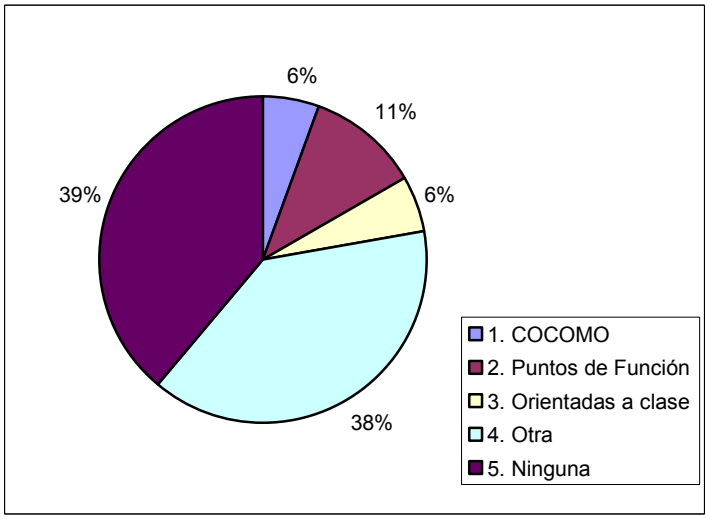

Figura 5: Métricas más usadas

No usan formas habituales para el cálculo de métricas.
En la Figura 5 se ve que un 39% no usa métricas y las más conocidas solo se aplican en un 17%.
Las preguntas 8 *¿Cuál cree que es el futuro de las Pymes productoras de software en la Argentina?* y 10 *¿Cuál piensa es la perspectiva para la ingeniería de software?* Como se observa en las Figuras 6 y 7 tanto para las PyMEs como para la ingeniería de software asociadas a las producciones se lo ve como un futuro con crecimiento continuo (55%) y con profundos cambios para la ingeniería de software (63%). Debido a la necesidad de comunicar sistemas en un mundo globalizado y de hacerlo cada vez más rápido es necesario llevar a cabo este trabajo de manera más confiable y económica lo que obliga a que se realicen cambios profundos en la ingeniería de

software. Además se debe considerar que la informática y las comunicaciones acompañan el crecimiento de una economía en expansión mundial lo que implica que la informática va a crecer.

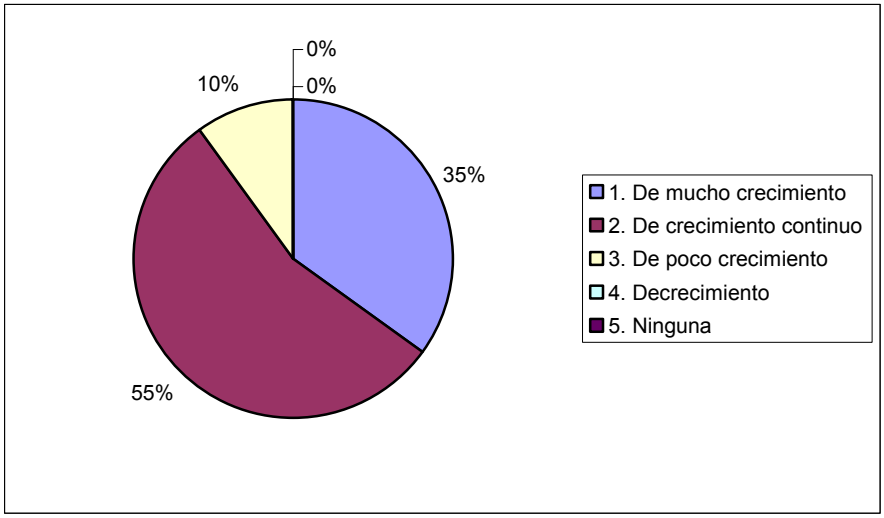

Figura 6: Opinión de los profesionales respecto del crecimiento de la Ingeniería de software

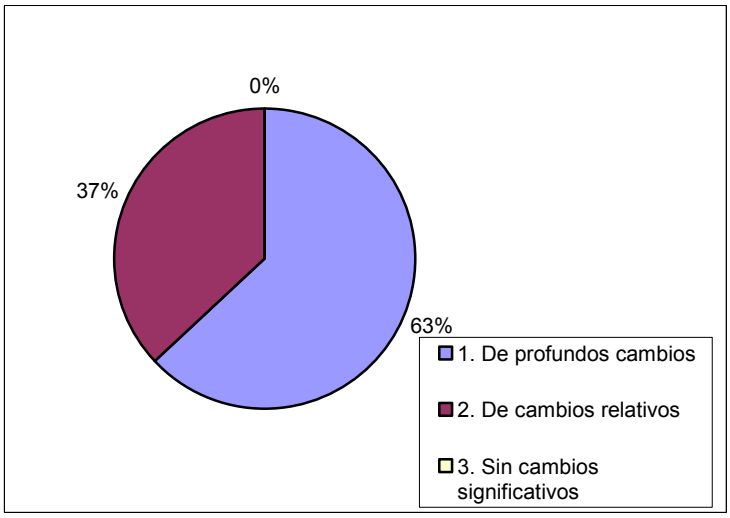

Figura 7: Opinión de los profesionales respecto del comportamiento de la Ingeniería de software

Los encuestados dan algunas explicaciones al respecto:

1. *Porque la tecnología avanza muy rápido*
2. *Se proponen un sinfín de nuevas metodologías y productos pero son muy pocos los que se traducen en dinero en los negocios (minimización de gastos, aumento de la rentabilidad).*
3. *Dada la alta integración de sistemas que se vislumbra*
4. *Mi respuesta está basada en la necesidad de aplicar e incrementar Ingeniería de Software y management en el desarrollo de Sistemas, eso generara valor en el producto y por ende mejor posicionamiento en el mercado, por lo cual, el necesario uso de estas técnicas llevará a su constante innovación*

5. *Porque la tecnología se encuentra en proceso de cambio continuo y la ingeniería debe seguir su camino*
6. *Porque la tecnología crece en forma continua y progresiva*

En cuanto a uno de los grandes pilares en que se basa la ingeniería de software que es la del sistema de documentación en la pregunta 11: *¿Documentan el análisis, diseño y/o construcción de aplicaciones?* Y la pregunta 12: *¿Utilizan una herramienta en su organización para documentar los sistemas?* Se observa que un 53% prácticamente no documenta (Figura 8). Un 60% usa una herramienta sencilla como un procesador de texto y alguna otra (Figura 9), aunque solo un 4% no usa ninguna.

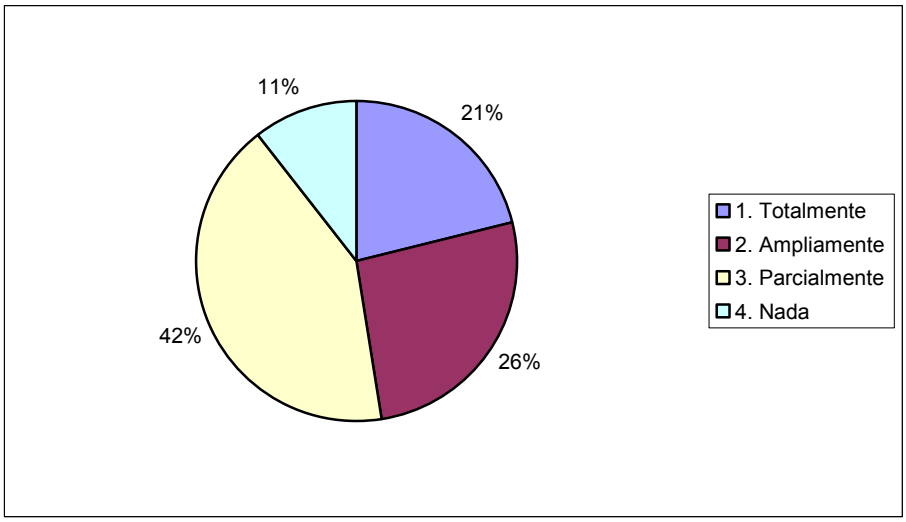

Figura 8: Ejercicio de la documentación en las etapas de construcción de sistemas

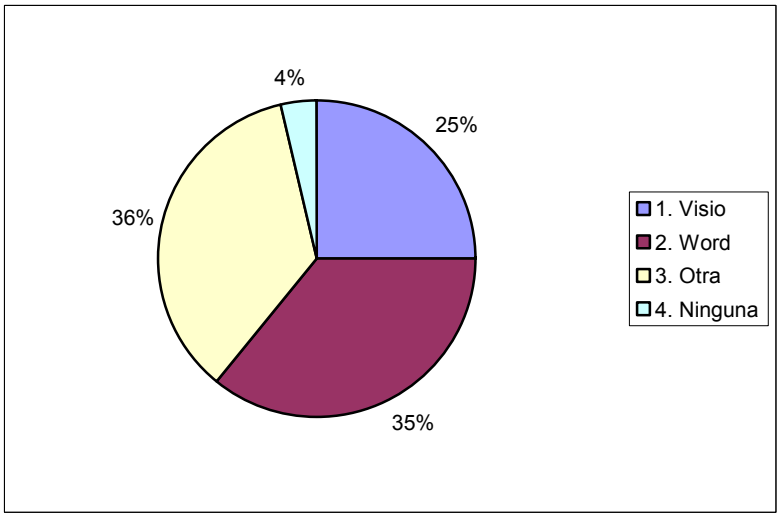

Figura 9: Herramientas usadas en la documentación durante la construcción de sistemas

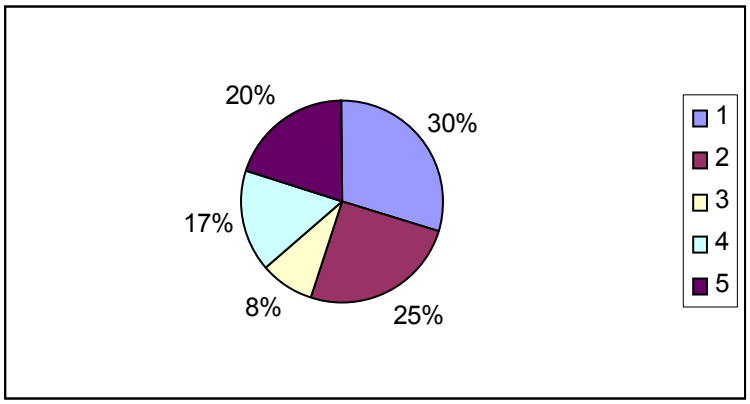

Figura 10: Herramientas usadas en las organizaciones

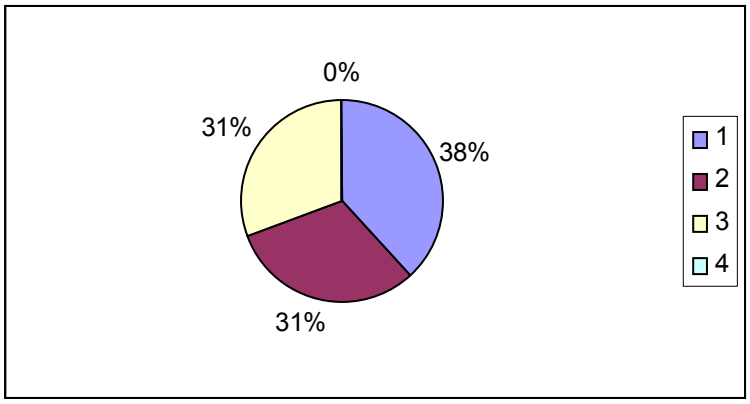

Figura 11: Porcentaje de formas de uso de la red (correo electrónico, acceso a la Web, página propia, y ninguna 0 %)

En cuanto a la tecnificación usada por los profesionales la pregunta 13, ¿Qué herramientas usan en la organización en que trabaja?. Y la pregunta 14, ¿Usan la red para comunicación interna? Se observa que los lenguajes de aplicación con un 30% seguido por los sistemas aplicativos con un 25 % son los más usados. Y que el 95 % usan el correo electrónico, además de que hay un importante acceso para el uso de Internet y páginas Web con información institucional.

En cuanto a los recursos humanos a partir de la pregunta 9, *¿Qué es lo que más valora del profesional de sistemas?* Se observa que en un 72 % se puntualiza la capacidad y el trabajo en equipo como se ve en la Figura 12.

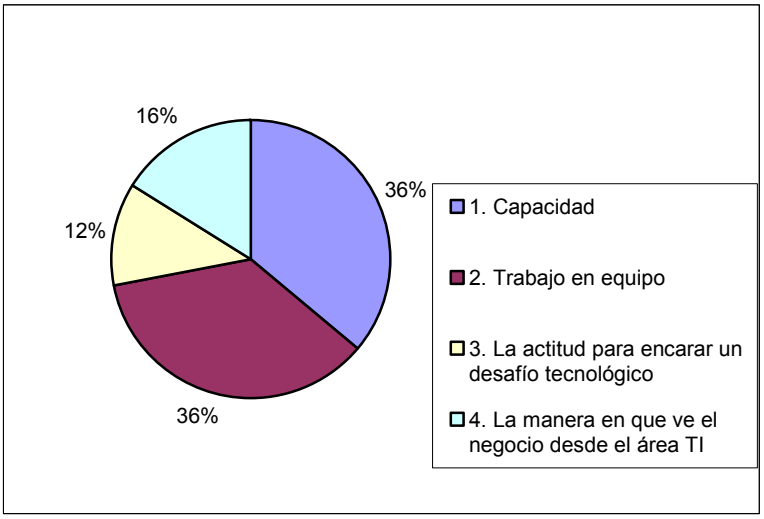

Figura 12: Características más valoradas por los profesionales de sistemas

Pero a través de las preguntas 6 *¿Realizan encuestas para evaluar al personal?* y 7 *¿Realizan encuestas para conocer la opinión del cliente?* En general solo el 30% hacen encuestas periódicas para evaluar al personal y un 39% para conocer la opinión del cliente en forma periódica.
Aquí hay que señalar que el encuestado 1 comenta: *Yo haría más hincapié en los recursos humanos, lejos, por sobre productos y/o tecnología física. El problema de los RR.HH. está dado por lograr conocer la verdadera calificación de los mismos.*

De la pregunta 13 observa en las respuestas que las herramientas más usadas en las organizaciones son los lenguajes de programación seguido por los sistemas aplicativos. Y en la pregunta 14, cuando se los consultan para qué se usa la red de comunicación interna, la respuesta más numerosa es para el correo electrónico seguido por igual porcentaje por los que poseen acceso a Internet como por los que poseen página Web con información institucional.

En las observaciones de la encuesta se han registrado las siguientes expresiones:

1. Yo haría más hincapié en los recursos humanos, lejos, por sobre productos y/o tecnología física. El problema de los RR.HH. está dada por lograr conocer la verdadera calificación de los mismos
2. La herramienta utilizada para la documentación de los proyectos es el Enterprise Architec 6.0, que sigue la metodología RUP.

4.5. Modelo de protocolo propuesto

Le agradecería responda el siguiente cuestionario donde: Si es la respuesta correcta, No la negativa, N/A es No Aplica, y NS/NC es No Sabe o No Contesta.	Si	No	NA	NS/ NC
DOCUMENTACIÓN INTERNA Y EXTERNA				
1. ¿El usuario revisó el sistema y el manual del usuario considerando la completitud y alcance?				

2. ¿La terminología, las descripciones del menú y las respuestas del sistema son consistentes con el programa?				
3. ¿Es realmente fácil encontrar una guía dentro de la documentación o de la ayuda del sistema?				
4. ¿El diseño del documento (tipo de formato, tipo de letra, etc.) es apropiado para comprender y asimilar rápidamente la información?				
ASEGURAMIENTO DE LA CALIDAD				
5. ¿Están documentadas todas las etapas del desarrollo?				
6. ¿Se siguió un modelo de sistema documental?				
7. ¿Se consideró cómo puede incorporarse la tolerancia a los defectos para reducir al mínimo la probabilidad de que un defecto se transforme en una falla?				
8. ¿Es fácil mover el sistema desde una ubicación a otra o de un tipo de computadora a otra?				
9. ¿Se respetan los requerimientos definidos para la confiabilidad, disponibilidad, facilidad de mantenimiento, seguridad y los restantes atributos de la calidad?				
10. ¿Existen actividades que puedan hacer que nuestro proceso sea más eficiente para asegurar la calidad?				
DEFINICIÓN DE REQUISITOS				
11. ¿Se han especificado las interfaces externas necesarias?				
12. ¿Se han especificado todos los recursos de hardware necesarios?				
13. ¿Los requisitos están bien definidos? (no hay contradicciones, son comprensibles, se especifica el rendimiento)				
14. ¿Si el requisito se puede probar se han especificado las pruebas de validación?				
15. ¿Se han definido los criterios de aceptación para cada una de las funciones definidas?				
16. ¿Los requerimientos definidos fueron verificados con la implementación del software?				
DISEÑO				
17. ¿Cubre el diseño todos los requisitos funcionales definidos?				
18. ¿Es el diseño suficientemente detallado y sin ambigüedades como para que sea posible implementarlo?				
PRUEBAS FUNCIONALES (CAJA NEGRA)				
19. ¿Es consistente el comportamiento del sistema con la información que debe procesar y las funciones que se desarrollaron?				
20. ¿Se probaron combinaciones específicas de datos (condiciones válidas y no válidas) para evaluar el efecto que tienen sobre el sistema?				
21. ¿Se han desarrollado casos de prueba en los valores límites, y apenas arriba y debajo de estos?				
22. ¿Se probaron una tasa de datos y un volumen de datos qué debe tolerar el sistema?				
23. ¿El sistema es sensible a determinados valores de entrada?				
PRUEBAS ESTRUCTURALES (CAJA BLANCA)				
24. ¿Se han probado en cada módulo todos los caminos independientes?				
25. ¿Se corrió el software probando todas las condiciones lógicas?				
26. ¿Se han ejecutado todos lo bucles en sus límites y dentro de sus límites operacionales?				
27. ¿Se han probado las estructuras de datos para asegura su validez?				
OBJETOS				
28. ¿Se han probado las operaciones de cada clase?				
29. ¿Se examinó el desempeño al colaborar una clase con otra?				
APLICACIONES WEB				
30. ¿Tiene el sitio una forma de navegación fácil de entender?				
31. ¿El sitio usa CSS (hojas de estilo en cascada) para todos los aspectos de la presentación (fuentes, color, borde, etc.)?				
32. ¿El estilo estético del contenido entra en conflicto con el estilo estético de la interfaz?				
33. ¿La Web utiliza en forma óptima el tamaño y la resolución de la pantalla?				
34. ¿Se realizaron pruebas para la compatibilidad de diferentes configuraciones de				

ambiente del cliente?				
35. ¿Se realizaron pruebas para evaluar la facilidad de uso de una página Web completa?				
36. ¿La USN (Unidad Semántica de Navegación) se logra en su totalidad sin error?				
37. ¿Se han probado todas las rutas para acceder a la USN?				
38. ¿Todo nodo de navegación se puede acceder dentro del contexto de las rutas de navegación que define la USN?				
39. ¿Existe un mecanismo para regresar al nodo precedente o al comienzo de la ruta de navegación?				
40. ¿Los mecanismos de navegación funcionan adecuadamente?				
41. ¿Todos los nodos se pueden alcanzar desde el mapa del sitio?				
42. ¿La Web es totalmente compatible con el sistema operativo del servidor?				
43. ¿La Web se ha probado con la configuración de servidor distribuido (si existe uno) que se haya elegido?				
44. ¿Se hicieron pruebas para descubrir errores entre la Web y la BD (Bases de Datos) remota?				
45. ¿Se realizaron pruebas que demuestren la validez de los datos brutos que recibe el servidor Web?				
46. ¿La Web está adecuadamente integrada con el software de base de datos? ¿La Web es sensible a diferentes versiones del software de BD?				
47. ¿El tiempo de respuesta del servidor se reduce hasta un punto donde es apreciable e inaceptable?				
48. ¿El sistema se degrada fácilmente o el servidor se desconecta cuando se rebasa su capacidad?				
CLIENTE/SERVIDOR				
49. ¿Se ha probado las funciones de coordinación y de manejo de datos del servidor?				
50. ¿Se ha considerado el desempeño del servidor? (tiempo de respuesta y procesamiento total de los datos)				
51. ¿Se ha probado la exactitud e integridad de los datos en las transacciones y almacenamientos del servidor?				
52. ¿Se han realizado las pruebas de comunicaciones de red? (mensajes, transacciones y tráfico de red de manera correcta)				
SISTEMAS DE TIEMPO REAL				
53. ¿Se han hecho pruebas para el manejo de eventos (procesamiento de interrupciones, temporización de datos, paralelismo entre las tareas que manejan datos)?				
54. ¿Se han asignado y manejado apropiadamente las propiedades de interrupción?				
55. ¿Se ha manejado correctamente el procesamiento de cada interrupción?				
56. ¿El desempeño de cada procedimiento de manejo de interrupciones cumple con los requisitos?				
57. ¿Un elevado número de interrupciones que llega en momentos críticos crea problemas en la función o el desempeño del sistema?				
AMBIENTE DE RED				
58. ¿Se ha probado la memoria requerida durante un enlace?				
59. ¿Se han probado los recursos requeridos durante un enlace?				
60. ¿Se ha probado la arquitectura y el desempeño de la red?				
VERIFICACIÓN				
61. ¿Se han creado los datos de prueba y métodos de prueba necesarios para probar adecuadamente el software?				
62. ¿Se ha desarrollado un proceso razonable para determinar si los defectos fueron corregidos satisfactoriamente?				
63. ¿Se puede llegar a un estado del sistema que debe ser evitado (como por ejemplo por razones de seguridad)?				
64. ¿Se consideró al programador en las pruebas de verificación y validación del software de manera no exclusiva?				

65. ¿La información respecto de cambios del software es recopilada y documentada?				
MANTENIMIENTO				
66. ¿El software es fácil de mantener?				
67. ¿Se hacen muchos cambios a los requisitos del software después de la entrega del mismo?				

Tabla 9: Modelo de protocolo propuesto

4.6. Justificación cuestionario y descripción página Web

Para la formulación de las preguntas usadas en el cuestionario se tomó como modelo algunas que se enuncian en los libros de Ingeniería de Software [Pfleeger, 2002] y [Pressman, 2005], también se valoraron las del CMMI-SE/SW V.1.1 to SW-CMM V1.1 y de otras encuestas. Se buscó un conjunto de preguntas reducido en número que abarquen todo los aspectos relacionados con la calidad del software. De esta manera el tiempo empleado en responderlas puede ser breve y entonces de manera económica se puede cumplir con los aspectos relacionados con la calidad.

Las preguntas se agruparon por temas para mayor facilidad de comprensión. Estos son: Documentación Interna y Externa, Aseguramiento de la Calidad, Definición de Requisitos, Diseño, Pruebas Funcionales (Caja Negra), Pruebas Estructurales (Caja Blanca), Objetos, Aplicaciones Web, Cliente/Servidor, Sistemas de Tiempo Real, Ambiente de Red, Verificación, y Mantenimiento.

A su vez, se pide que las respuestas sean: Correcta (Si), incorrecta o Negativa (No), No Aplica (N/A), No Sabe o No Contesta (NS/NC). Abarcando con las múltiples opciones todas las posibilidades que puede considerar quien completa el cuestionario. También pensando en la claridad se usó un vocabulario sencillo y al mismo tiempo estándar.

A partir de las preguntas que se observan en el instructivo de toma de datos (Tabla 9) en que se observa un "Modelo de protocolo propuesto", para el seguimiento del desarrollo de programas se realizó una aplicación Web que permita facilitar la toma de decisiones a los responsables de hacerlas.

Primero se hizo una página de prueba donde se podía bajar el cuestionario de la red haciendo clic en una de las opciones como un archivo de texto; con otra opción, facilitaba enviar las respuestas a mi dirección de correo para poder hacer un seguimiento. También había información en la pantalla que permitía identificar este sistema y una explicación breve de cómo completar el cuestionario. De esta manera se pudo estudiar el uso del mismo gracias a lo que contestaron unos profesionales compañeros y colegas. En la Figura xx se puede observar la página de inicio.

Tesis para la Maestría de Ing. en Sistemas de Información

UTN – FRBA

"Modelo de Protocolo para el Análisis y Evaluación del Software Desarrollado en las PyMES Argentinas"

Martín A. Valiente

e-mail: mvaliente@lpsat.com

Le agradecería responda las preguntas del cuestionario de opciones múltiples marcando con una X, donde: "Si" es la respuesta correcta, "No" la negativa, "N/A" No Aplica, y "NS/NC" es No Sabe o No Contesta. Y lo envíe adjunto a la siguiente dirección de correo electrónico: mvaliente@lpsat.com

Pulse aquí para ver Cuestionario

Mandar un Email

Figura xx. descripción

Posteriormente se desarrolló una nueva página que se subió a Internet con la facilidad de completar el cuestionario y datos a cerca de quien responde de manera de capturar las respuestas y guardarlas en una base de datos. Estos datos pueden ser utilizados para realizar estadísticas y así sacar conclusiones que puedan ser aprovechadas en el futuro. La pantalla que corresponde a este nuevo desarrollo se puede ver en las páginas siguientes.

CUESTIONARIO PRUEBA DE SOFTWARE

Carrera: Universidad:

Especialidad: PROGRAMADOR

Hacer click donde la columna: *Si es la respuesta correcta, No la negativa, NA es No Aplica, y NS es No Sabe o No Contesta.*	**Selección**			
DOCUMENTACIÓN INTERNA Y EXTERNA	**Si**	**No**	**NA**	**NS**
1. ¿El usuario revisó el sistema y el manual del usuario considerando la completitud y alcance?	☐	☐	☐	☐
2. ¿La terminología, las descripciones del menú y las respuestas del sistema son consistentes con el programa?	☐	☐	☐	☐
3. ¿Es realmente fácil encontrar una guía dentro de la documentación o de la ayuda del sistema?	☐	☐	☐	☐
4. ¿El diseño del documento (tipo de formato, tipo de letra, etc.) es apropiado para comprender y asimilar rápidamente la información?	☐	☐	☐	☐
ASEGURAMIENTO DE LA CALIDAD	**Si**	**No**	**NA**	**NS**
5. ¿Están documentadas todas las etapas del desarrollo?	☐	☐	☐	☐
6. ¿Se siguió un modelo de sistema documental?	☐	☐	☐	☐
7. ¿Se consideró cómo puede incorporarse la tolerancia a los defectos para reducir al mínimo la probabilidad de que un defecto se transforme en una falla?	☐	☐	☐	☐
8. ¿Es fácil mover el sistema desde una ubicación a otra o de un tipo de computadora a otra?	☐	☐	☐	☐
9. ¿Se respetan los requerimientos definidos para la confiabilidad, disponibilidad, facilidad de mantenimiento, seguridad y los restantes atributos de la calidad?	☐	☐	☐	☐
10. ¿Existen actividades que puedan hacer que nuestro proceso sea más eficiente para asegurar la calidad?	☐	☐	☐	☐
DEFINICIÓN DE REQUISITOS	**Si**	**No**	**NA**	**NS**
11. ¿Se han especificado las interfaces externas necesarias?	☐	☐	☐	☐
12. ¿Se han especificado todos los recursos de hardware necesarios?	☐	☐	☐	☐
13. ¿Los requisitos están bien definidos? (no hay contradicciones, son comprensibles, se especifica el rendimiento)	☐	☐	☐	☐
14. ¿Si el requisito se puede probar se han especificado las pruebas de validación?	☐	☐	☐	☐
15. ¿Se han definido los criterios de aceptación para cada una de las funciones definidas?	☐	☐	☐	☐
16. ¿Los requerimientos definidos fueron verificados con la implementación del software?	☐	☐	☐	☐
DISEÑO	**Si**	**No**	**NA**	**NS**
17. ¿Cubre el diseño todos los requisitos funcionales definidos?	☐	☐	☐	☐
18. ¿Es el diseño suficientemente detallado y sin ambigüedades como para que sea posible implementarlo?	☐	☐	☐	☐

PRUEBAS FUNCIONALES (CAJA NEGRA)	**Si**	**No**	**NA**	**NS**
19. ¿Es consistente el comportamiento del sistema con la información que debe procesar y las funciones que se desarrollaron?	☐	☐	☐	☐
20. ¿Se probaron combinaciones específicas de datos (condiciones válidas y no válidas) para evaluar el efecto que tienen sobre el sistema?	☐	☐	☐	☐
21. ¿Se han desarrollado casos de prueba en los valores límites, y apenas arriba y debajo de estos?	☐	☐	☐	☐
22. ¿Se probaron una tasa de datos y un volumen de datos qué debe tolerar el sistema?	☐	☐	☐	☐
23. ¿El sistema es sensible a determinados valores de entrada?	☐	☐	☐	☐
PRUEBAS ESTRUCTURALES (CAJA BLANCA)	**Si**	**No**	**NA**	**NS**
24. ¿Se han probado en cada módulo todos los caminos independientes?	☐	☐	☐	☐
25. ¿Se corrió el software probando todas las condiciones lógicas?	☐	☐	☐	☐
26. ¿Se han ejecutado todos lo bucles en sus límites y dentro de sus límites operacionales?	☐	☐	☐	☐
27. ¿Se han probado las estructuras de datos para asegura su validez?	☐	☐	☐	☐
OBJETOS	**Si**	**No**	**NA**	**NS**
28. ¿Se han probado las operaciones de cada clase?	☐	☐	☐	☐
29. ¿Se examinó el desempeño al colaborar una clase con otra?	☐	☐	☐	☐
APLICACIONES WEB	**Si**	**No**	**NA**	**NS**
30. ¿Tiene el sitio una forma de navegación fácil de entender?	☐	☐	☐	☐
31. ¿El sitio usa CSS (hojas de estilo en cascada) para todos los aspectos de la presentación (fuentes, color, borde, etc.)?	☐	☐	☐	☐
32. ¿El estilo estético del contenido entra en conflicto con el estilo estético de la interfaz?	☐	☐	☐	☐
33. ¿La Web utiliza en forma óptima el tamaño y la resolución de la pantalla?	☐	☐	☐	☐
34. ¿Se realizaron pruebas para la compatibilidad de diferentes configuraciones de ambiente del cliente?	☐	☐	☐	☐
35. ¿Se realizaron pruebas para evaluar la facilidad de uso de una página Web completa?	☐	☐	☐	☐
36. ¿La USN (Unidad Semántica de Navegación) se logra en su totalidad sin error?	☐	☐	☐	☐
37. ¿Se han probado todas las rutas para acceder a la USN?	☐	☐	☐	☐
38. ¿Todo nodo de navegación se puede acceder dentro del contexto de las rutas de navegación que define la USN?	☐	☐	☐	☐
39. ¿Existe un mecanismo para regresar al nodo precedente o al comienzo de la ruta de navegación?	☐	☐	☐	☐
40. ¿Los mecanismos de navegación funcionan adecuadamente?	☐	☐	☐	☐
41. ¿Todos los nodos se pueden alcanzar desde el mapa del sitio?	☐	☐	☐	☐
42. ¿La Web es totalmente compatible con el sistema operativo del servidor?	☐	☐	☐	☐

43. ¿La Web se ha probado con la configuración de servidor distribuido (si existe uno) que se haya elegido?	☐	☐	☐	☐
44. ¿Se hicieron pruebas para descubrir errores entre la Web y la BD (Bases de Datos) remota?	☐	☐	☐	☐
45. ¿Se realizaron pruebas que demuestren la validez de los datos brutos que recibe el servidor Web?	☐	☐	☐	☐
46. ¿La Web está adecuadamente integrada con el software de base de datos? ¿La Web es sensible a diferentes versiones del software de BD?	☐	☐	☐	☐
47. ¿El tiempo de respuesta del servidor se reduce hasta un punto donde es apreciable e inaceptable?	☐	☐	☐	☐
48. ¿El sistema se degrada fácilmente o el servidor se desconecta cuando se rebasa su capacidad?	☐	☐	☐	☐
CLIENTE/SERVIDOR	**Si**	**No**	**NA**	**NS**
49. ¿Se ha probado las funciones de coordinación y de manejo de datos del servidor?	☐	☐	☐	☐
50. ¿Se ha considerado el desempeño del servidor? (tiempo de respuesta y procesamiento total de los datos)	☐	☐	☐	☐
51. ¿Se ha probado la exactitud e integridad de los datos en las transacciones y almacenamientos del servidor?	☐	☐	☐	☐
52. ¿Se han realizado las pruebas de comunicaciones de red? (mensajes, transacciones y tráfico de red de manera correcta)	☐	☐	☐	☐
SISTEMAS DE TIEMPO REAL	**Si**	**No**	**NA**	**NS**
53. ¿Se han hecho pruebas para el manejo de eventos (procesamiento de interrupciones, temporización de datos, paralelismo entre las tareas que manejan datos)?	☐	☐	☐	☐
54. ¿Se han asignado y manejado apropiadamente las propiedades de interrupción?	☐	☐	☐	☐
55. ¿Se ha manejado correctamente el procesamiento de cada interrupción?	☐	☐	☐	☐
56. ¿El desempeño de cada procedimiento de manejo de interrupciones cumple con los requisitos?	☐	☐	☐	☐
57. ¿Un elevado número de interrupciones que llega en momentos críticos crea problemas en la función o el desempeño del sistema?	☐	☐	☐	☐
AMBIENTE DE RED	**Si**	**No**	**NA**	**NS**
58. ¿Se ha probado la memoria requerida durante un enlace?	☐	☐	☐	☐
59. ¿Se han probado los recursos requeridos durante un enlace?	☐	☐	☐	☐
60. ¿Se ha probado la arquitectura y el desempeño de la red?	☐	☐	☐	☐
VERIFICACIÓN	**Si**	**No**	**NA**	**NS**
61. ¿Se han creado los datos de prueba y métodos de prueba necesarios para probar adecuadamente el software?	☐	☐	☐	☐
62. ¿Se ha desarrollado un proceso razonable para determinar si los defectos fueron corregidos satisfactoriamente?	☐	☐	☐	☐
63. ¿Se puede llegar a un estado del sistema que debe ser evitado (como por ejemplo por razones de seguridad)?	☐	☐	☐	☐
64. ¿Se consideró al programador en las pruebas de verificación y validación del software de manera no exclusiva?	☐	☐	☐	☐

65. ¿La información respecto de cambios del software es recopilada y documentada?	☐	☐	☐	☐
MANTENIMIENTO	**Si**	**No**	**NA**	**NS**
66. ¿El software es fácil de mantener?	☐	☐	☐	☐
67. ¿Se hacen muchos cambios a los requisitos del software después de la entrega del mismo?	☐	☐	☐	☐

Finalizar

5. Conclusiones

5.1. Análisis y conclusiones

A partir del análisis efectuado se puede destacar la importancia del software en las organizaciones porque éste sirve de soporte a los procesos de negocio, y también es parte integral de las estrategias corporativas que tiene como objetivo la generación de ventajas competitivas.

Entre las conclusiones parciales, se puede resaltar de los siguientes puntos:

- La ingeniería de sistemas busca al desarrollar software, especialmente construir programas de manera más sencilla, rápida y menos costosa para de esta manera tener más posibilidades de introducirlos en los mercados.

- Las organizaciones requieren que el software sea más confiable, para eso pueden buscar la calidad a nivel proceso y a nivel producto. Con esa meta por ejemplo pueden implementar normas como son CMMI y la ISO 9001. Si bien hay muchas normas de calidad hay que tener en cuenta los objetivos de la empresa y donde colocar los productos a la hora de elegir una de ellas.

- Las empresas que están certificadas con normas de calidad tienen beneficios tales como el reconocimiento en los mercados y una manera de encarar la toma de decisiones que corresponde a la filosofía de la calidad del software.

- Se puede destacar que la calidad es uno de los pilares sobre los que se sostienen los productos en el mercado. Para el logro de un producto software de calidad es necesario una gestión de la calidad que sea integral. Debido a ello es conveniente utilizar modelos y estándares de calidad del software, los cuales consisten en reunir todas las actividades y funciones de manera tal que cada una de ellas se planee, controle y ejecute de un modo formal y sistemático.

- Con este fin se crearon herramientas como por ejemplo las aplicaciones Web (WebApps) que permiten un uso intensivo de la red y una continua evolución de las aplicaciones que son gobernadas por el contenido, de tal modo de poder acceder más fácilmente a la estrategia de practicidad Web.

- De las normativas existentes a nivel mundial en Pymes se puede usar la ISO/IRAM 9001:2000 de la serie 9000 (ISO/IEC/IRAM 90003) para la gestión de la calidad, IRAM-ISO 15504 conocida también como SPACE, IRAM 17601 (CMMI del SEI), ISO/IEC/IRAM 9126 para la calidad del software y la ISO 17799 para la seguridad de la informática.

- En el estudio sobre el estado actual de la Pymes realizado se observó cuáles son los paradigmas de desarrollo que se usan, entre los que se puede mencionar a las metodologías orientadas a objeto y las herramientas usadas son las de más fácil uso como por ejemplo para el diseño el uso de del Visio o del Word lo que refleja una actitud especial por ser sencillas y rápidas de aprender a usar.

- Finalmente, se analizaron y evaluaron los paradigmas más recientes como son las metodologías ágiles que permiten un desarrollo iterativo e incremental para obtener mejoras, pruebas unitarias continuas, simplicidad en el código y frecuente interacción del equipo de programación.

- En consecuencia, se logra el objetivo general que es determinar cuáles son los paradigmas de diseño y las metodologías que usan las Pymes argentinas actualmente, además del nivel de calidad que se aplica en el trabajo. También se observó la actitud positiva que tienen los recursos humanos frente a los cambios en la forma de desarrollo de los sistemas.

5.2. Consideraciones y limitaciones

Se pueden realizar una serie de consideraciones. Por un lado, están los recursos humanos que deben ser profesionalmente competentes, adecuando las tareas que desempeñan a su perfil y estar continuamente capacitados. Por otro lado, hay que tener en cuenta la infraestructura y la inversión que deberá estar adecuada y se podrá optimizar para cumplir con los objetivos de la Pyme. También hay que consoderar ~~a~~ la gestión del proyecto que incluye el conocimiento y la administración del negocio, y es muy importante para un buen desempeño la planificación y el seguimiento de las tareas que se realizan.

Entre las limitaciones que se deben tener en cuenta está presente el tema económico ya que las Pymes como toda empresa se requiere generar ganancias para sostener los emprendimientos y por ello los costos deben ser estudiados minuciosamente para evitar los riesgos financieros que comprometan a la organización.

El mercado tiene necesidades de productos software que hay que satisfacer para generar ganancias. La detección de esos nichos y una rápida respuesta favorecen a las Pymes que compiten para que se integren a la economía real global. Un problema que se les presenta a las Pymes Argentinas es la dificultad de acceder al crédito que les dificulta la construcción de sistemas de cierta magnitud.

También hay una limitación dado las restricciones legales que pueden exigir el cumplimiento de obligaciones en determinadas áreas como por ejemplo la seguridad sin afectar la privacidad de los empleados, o de guardar determinada información por varios años por lo que hay que conservar la información digital y el hardware necesario de tal manera de poder mostrarla si esta es requerida.

5.3. Líneas futuras de investigación

A partir de la problemática detectada y de los resultados obtenidos se puede continuar con las siguientes líneas de trabajo:

- Desarrollo de otras metodologías basadas en los ciclos de vida del software y los procedimientos para el desarrollo del software en aplicaciones Web.
- Desarrollo de un software especial en español para el sector de Pymes que producen software a los efectos de favorecer el seguimiento de los procedimientos o de las normas de calidad.
- Otra posible línea, es la adopción de aplicaciones que existen en el mercado o la incorporación de nuevos programas que cubran las necesidades específicas de las Pymes productoras de software teniendo en cuenta el idioma y las herramientas informáticas específicas que se usan en el mercado.
- También se puede desarrollar software que utilice métodos estadísticos para el estudio de los datos obtenidos con el objetivo de obtener información útil de ser usada.
- Desarrollo de una aplicación que se base en una línea de investigación con base algorítmica como a través de la clasificación usando redes neuronales.

6. Bibliografía

ADI, "Invertir en Argentina" Tecnología/Software , Consultado el 13/12/2008 en: http://www.inversiones.gov.ar/documentos/inf_sect_software.pdf

Amescua, Ingeniería del Software de Gestión (Edit. Paraninfo, 1995)

Alfaro, A. P**., "E**l control de gestión y el turn around management", UTN Seminarios de MISI (2003)

Arbones Malirani, E.A., Ingeniería de Sistemas (Edit. Marcombo, 1991)

Autor Centro Computación Profesional México, Análisis y Diseño de Sistemas (Edit. McGraw Hill, 2001)

Braude Eric, J., Ingeniería de Software. Una perspectiva Orientada a Objetos (Edit. AlfaOmega, 2003)

Briand, L. C., Bunse, C., y Daly, J. W. , A controlled experiment for evaluating quality guidelines on the maintainability of object-oriented designs. (Software Engineering, IEEE Transactions on**,** Jun 2001). Volume: 27, Issue: 6, On page(s): 513-530

Brown, B., "Top five RUP implementation process killers". The Rational Edge. (2003). Consultado el 05/03/04 en: http://www.therationaledge.com/content/jun_03/f_topfive_bb.jsp

Burch, J. G., Diseño de Sistemas de Información (Edit. Limusa, 1999)

Calvi, P., "Las Pymes de software, especialistas en exportar" http://www.pymesdigital.com.ar/secciones/exportaciones/exp11.htm , Clarín (Febrero, 13, 2004)

Carballo, R., Ponencia S12B6 "El ciclo de vida de las métricas del software". Segundas Jornadas de ingeniería del software Universidad Empresa. 15 al 27 de setiembre, (Universidad de la Coruña, 2003).

Castro, H.M., "Data Warehouse y Sistemas de Soporte a la Decisión", UTN MISI (2002)

Castro, Vladimir Estivill, "Minería de Datos", Investigador visitante del LANIA, Universidad Tecnológica de Queensland, Australia. Consultado el 11/08/ 2008 en: http://www.lania.mx/spanish/actividades/newsletters/1997-otono-invierno/mineria.html , (Agosto, 11, 2003)

Cataldi, Z., "Metodología de diseño, desarrollo y evaluación de software educativo". Tesis de Magíster en Informática. Facultad de Informática. UNLP. (2000)

CESSI, "Nuevas Leyes para la producción de Software" en cessi-soft(nuevas leyes)_archivos.htm, (Agosto,23,2003)

CESSI, "Nuevas leyes para la producción de Software" (Septiembre, 23, 2003) http://www.cessi.org.ar/

CESSI, "El Gobierno de la Ciudad de Buenos Aires aprobó 23 proyectos de empresas del sector IT, que recibirán subsidios directos por un monto de $460.000". Berardo,D.A., Belotti,H.E., Vigetti.,O.G., (Agosto,04,2003) http://www.cessi.org.ar/

CICOMRA, Ejes del Trabajo (Febrero, 16, 2004)

http://www.cicomra.org.ar/novedades/documentos/ejesdetrabajo.PDF,

Collin, J. P., Certificaciones: un seguro de calidad para los clientes. Users.Code (Año III, Número 26, 2006)

Cumpe, O. E., "Aplicación del CMM en Empresas de Software de Argentina". Tesis de Maestría de la Universidad Tecnológica Nacional, Facultad Regional Buenos Aires, Argentina. (Septiembre 2003)

Davis William, S., Herramientas Case (Edit. Paraninfo, 1992)

Decreto 10/2004, Respecto de la Ley 25.856/04 (06/01/2004)

Decreto 165, Propiedad intelectual - Software y base de datos - Protección – Normas (Fecha: 3 febrero 1994, Publicación: B.O. 8/2/94)

Decreto 1.307/98, Reglamentario de la Ley 25.036/98 (06/11/1998)

Decreto 2628/02, Decreto Regulatorio de la Ley 11.723 (2002)

Decreto 690/2002, Decreto Regulatorio de Comercio Exterior (2002)

DeMarco, T., Structured analysis and systems specifications. (Prentice Hall, 1979).

DeVilliers, D. J. "Introducing the RUP into an Organization". The Rational Edge. (2002). Consultado el 05/03/08 en: http://www.therationaledge.com/content/jan_02/m_introducingTheRUP_dd.html

Diez, E., "Aseguramiento de la Calidad en la Construcción de Sistemas Basados en el Conocimiento un enfoque práctico". Trabajo de Especialización en Ingeniería de Sistemas Expertos. ITBA. (2003)

Feld, F., Introducción al Marketing en Internet para Pymes. (Edit. Compumagazine, 2000)

Fernández Delpech, H., Protección Jurídica del Software (Edit. Lexis Nexis, 2000)

Foro de Software y Servicios Informáticos, Libro Azul y Blanco, Plan Estratégico de SSI 2004 – 2014 y Plan de acción 2004-2007 (Ministerio de Economía y Producción) http://www.industria.gov.ar/foros/soft_inf/documentos/fssi_libro_ayb.pdf

Firtman, M., ASP.NET Aplicaciones WEB de Alto Rendimiento (Edit. MP Ediciones)

Gane, C., Sarsons, T., Análisis Estructurado de Sistemas. Quinta reimpresión. (El Ateneo, 1977).

Graña, M.T.M., "INFORMACIÓN GENERAL" SOBRE EL PROGRAMA DE POSTGRADO EN ING. DEL SOFTWARE,
Consultado el 04/04/04 en:
http://www.ls.fi.upm.es/master/sobre_e.htm

González Jardón, C. Ponencia S2B7 "Modelo CMM para pymes". Segundas Jornadas de ingeniería del software Universidad Empresa. 15 al 27 de setiembre, (Universidad de la Coruña, 2003).

Guerrieri, H.. Instrumentos para el cambio en la Provincia de Buenos Aires. (1999)

Guns, B. Aprendizaje organizacional. Cómo ganar y mantener la competitividad. (Prentice Hall, Simon and Schuster, 1996)

Hernández Sampieri, R. y otros. Metodología de la Investigación. (Mc Graw Hill. 2007).

Humphery Watts, S., A Discipline for Software Engineering (Edit. Addison-Wesley 2000)

Humphery Watts, S., Introducción al proceso software personal (Edit. Addison-Wesley Iberoa 2001)

Jacobson I., Booch Grady, Rumbaugh, J., El proceso unificado de desarrollo de software (Edit. Addisson-Wesley Iberoa, 2000)

Jacobson, I., Booch, G. y Rumbaugh, J., El Proceso Unificado de Desarrollo de Software. (Pearson Educación S. A., 2000).

Johansen, Teoría General de Sistemas. Introducción (Edit. Limusa 1998)

Jenner M., Software Quality Management and ISO 9001. (John Wiley, 1995)

Kaplan, C., Clatk R., y Tang G., Secrets of Software Quality.(Mc Graw Hill,1995)

Kendall, Análisis y Diseño de Sistemas. 3ra Edición (Prentice Hall, 1997)

Kroll, P. y Kruchten, P. The Rational Unified Process Made Easy: A Practitioner's Guide to RUP. (Addison Wesley, 2003)

Kruchten, P. "Going Over the Waterfall with the RUP". The Rational Edge. (2001). Consultado el 05/08/08 en: http://www.therationaledge.com/content/sep_01/t_waterfall_pk.html

Kruchten, P., "From Waterfalll to Iterative Lifecycle. A tough transition for project managers". Rational Software White Paper. (TP–173 5/00, 2000). Consultado el 05/08/08 en: http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/RUPwaterfalltoiterative.pdf

Ley 11723(235) Régimen Legal de la Propiedad Intelectual (1989)

Ley 19550 Ley de Sociedades Comerciales (03/04/1972, promulgada en Boletín Oficial con fecha 25/4/72,Decreto 841/84 B.O. 30/03/1984)

Ley 24.467 Régimen para pequeñas y medianas empresas - Derogación de la ley 23.020. (Sanción: 15 marzo 1995, Promulgada: 23 marzo 1995, Publicación: B. O. 28/3/95)

Ley 24.766, Ley de Confidencialidad sobre Información y Productos que estén legítimamente bajo control de una persona y se divulgue indebidamente de manera contraria a los usos comerciales honestos. (Sancionada: Diciembre 18 de 1996, Promulgada: Diciembre 20 de 1996)

Ley 25036 Ley de Propiedad Intelectual (Sancionada: Octubre 14 de 1998, Promulgada: Noviembre de 1998)

Ley 25300 Ley de Fomento para la Micro, Pequeña y Mediana Empresa (2002)

Ley 25326 Ley de Protección de los Datos Personales (2000)

Ley 25506 Firma Digital (2001)

Ley 25856 Consideración de la producción de software

Maddison, R. N., Information System methodologies. Wiley Henden (1993).

Markiewicz, M.E.,Carlos J.P. de Lucena, "El Desarrollo del Framework Orientado al Objeto" Consultado el 07/08/2008 en: http://www.acm.org/crossroads/espanol/xrds7-4/frameworks.html

Mende Ulrich, Software Development for SAP R/3 (Edit. Springer, 2000)

Norma IRAM-ISO 9000:2000 Sistemas de Gestión de la Calidad. Fundamentos y vocabulario. (30/05/2001)

Norma IRAM-ISO 9001:2000 Sistemas de Gestión de la Calidad. Requisitos. (30/05/2001)

Norma IRAM-ISO 9004:2000 Sistemas de Gestión de la Calidad. Directrices para la mejora del desempeño. (30/05/2001)

Norris, M., Rigby, P., Ingeniería del Software Explicada (Edit. Limusa, 1994)

Oktaba, H., "Estrategia de la normalización para la industria de software en México", Universidad Nacional Autónoma de México, Consultado el 06/05/2008 en: http://www.unlm.edu.ar/ritos/Documentos/softMex.ppt

Olivito, J., Lafuente, G., Bernabé Loranca, M. B., Olsina, L., "Estudio de Atributos de Calidad en Sitios E-commerce Argentinos". (2004)

Pfleeger, S. L., Ingeniería de Software. Teoría y Práctica. (Prentice Hall, Enero de 2002)

Piattini Velthuis, M. G., Garcia Rubio Felix, O., Calidad en el Desarrollo y Mantenimiento del Software (Edit. AlfaOmega, 2003)

Piattini, M., Bastanchury, T., Martinez, M. A., Nistal, C., Polo, M., Ruiz, F., Mantenimiento del software, (Edit. AlfaOmega, 2001)

Piattini, M., Mantenimiento del software (Edit. Ra-ma, 1998)

Piattini, M., Análisis y Diseño Detallado de Aplicaciones Informáticas de Gestión. (Rama, 1996) Madrid

Piattini Velthuis, M. G., Calvo-Manzano, J. A., Cervera, J., Fernández, L., Análisis y Diseño de Aplicaciones Informáticas de Gestión. Una perspectiva de Ingeniería del Software (Edit. Alfaomega Rama, 2004)

Pressman, R. S., Ingeniería de Software. Un enfoque práctico .(Sexta edición) (Edit. Mc Graw Hill, 2005)

Prince, A., Ponencia "La Era del Conocimiento y la Nueva Economía". Cuartas Jornadas Universitarias de Tecnologías de Internet. 23/10/2003, Bs. As..

Puigjaner, Evaluación y Explotación de Sistemas Informáticos (Edit. Síntesis, 1995)

Rational Software & Context Integration. "Building Web Solutions with the Rational Unified Process: Unifying the Creative Design Process and the Software Engineering Process". Rational Software White Paper. (1999). Consultado el 05/03/08 en: http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/76.pdf

Rumbaugh, J., Jacobson, I. y Booch, G.. El Lenguaje Unificado de Modelado, Manual de Referencia. (Pearson Educación S. A., 2000).

Resolución 8/2001 Tratamiento Arancelario Específico (2001)

Resolución 856/1995, comercio exterior (1995)

Resolución 22/2001 "Micro, Pequeñas y Medianas Empresas", Modificación (30/04/2001)

Resolución 24/2001 "Micro, Pequeñas y Medianas Empresas", Ventas Totales Anuales - Nivel Máximo de Valor (20/02/2001)

Resolución 675/2002 "Micro, Pequeñas y Medianas Empresas", Parámetros para su Consideración (30/10/2002)

Rumbaugh, J., Object Oriented modeling and design. Englewood Cliffs. (Prentice Hall, 1991) Nueva Jersey

Rumbaugh, J., Over waterfall and into the whirpool. En JOOP, mayo de 1992, págs. 23-26.

Sametband, P., Aseguramiento de la calidad. Users.Code (Año III, Número 26, 2006)

Sametband, R., "Software con calidad argentina". La Nación (Julio, 22, 2002) sección 5 p.1

Sametband, R., "Retrato de la Argentina digital". La Nación (Noviembre, 24, 2003) s.5 p.1

Scalone, F., "Estudio Comparativo de los Modelos y Estándares de Calidad del Software". Maestría en Ingeniería de Calidad. UTN-FRBA. (2006)

Schultz, M. R., "Abogados y Derecho de la Alta Tecnología. La ley de promoción del software". El Mes Judicial (Octubre, 2005) Año 2 Nº 12

Scolnik, Hugo D. Ponencia "Panel de Políticas de Promoción". Primeras Jornadas Latinoamericanas de Ingeniería y Desarrollo de Software: Teoría y Aplicaciones – IDS 2003

Segundas Jornadas de ingeniería del software Universidad Empresa. 15 al 27 de setiembre, (Universidad de la Coruña, 2003)

Senlle A., y Stoll G., ISO 9000: Las normas para la calidad en la práctica. (Gestión, 2000)

Sigwart C. et al. Software Engineering: a project oriented approach. Franklin, Beedle y Associates, Inc., Irvine, (California, 1990)

SQI, SPACE, Consultado el 25/02/2008 en: http://www.sqi.gu.edu.au/spice/

Somerville, I. Y Sawyer, P., Requirements Engineering (Wiley, 1997)

Sommerville Ian, Ingeniería del Software (Edit. Prentice Hall, 2002)

The RUP Implementation Workshop Group. "RUP implementation guide. Part I: Recommended strategy and typical issues and risks". The Rational Edge. (2003). Consultado el 05/03/08 en:
http://www.therationaledge.com/content/aug_03/f_implement_gp.jsp

Torres, A., "Se vendieron este año más de 685000 PC". Clarín (Dic.,19,2004) Información General.

Warmer, J. B., Kleppe, A. G., The object constraint language. Second Edition (Addison-Wesley, 2003)

Yourdon, E. y Constantine, L. Structured design, 2º Ed. Englewood Cliffs, (Prentice Hall, 1975).

Yourdon, E. Análisis Estructurado Moderno. (Prentice Hall, 1989)

Zavala, R., "Diseño de un Sistema de Información Geográfica sobre internet". Extracto de la Tesis de Maestría en Ciencias de la Computación. Universidad Autónoma Metropolitana-Azcapotzalco. México, D.F. (2000)
http://www.angelfire.com/scifi/jzavalar/apuntes/IngSoftware.html

Sitios Web de Institutos de investigación relacionados con la Ingeniería de Software y de normalización

Sitio de ACM (Association for Computing Machinery) http://www.acm.org
Sitio de ADI (Agencia de Desarrollo de inversions) www.inversiones.gov.ar
Sitio de AEA (Asociación Electrotécnica Argentina) www.aea.org.ar
Sitio de AEMES (Asociación Española de Métricas del Software) http://www.aemes.org/
Sitio de AENOR (Asociación Española de Normalización y Certificación) http://www.aenor.es/
Sitio de Agencia Nacional de Promoción Científica y Empleo (del Ministerio de Educación, Ciencia y Tecnología) http://www.agencia.secyt.gov.ar/
Sitio de AMN (Asociación Mercosur de Normalización) www.amn.org.br
Sitio de ANSI (American National Standards Institute) www.ansi.org
Sitio de ASQ (American Society for Quality) www.asq.org
Sitio de CABASE (Cámara Argentina de Bases de Datos y Servicios de Línea) http://www.cabase.org.ar/
Sitio de CADIE (Cámara Argentina de Industrias Electrónicas) http://www.cadie.org.ar/
Sitio de Casio http://www.casioargentina.com/
Sitio de CEN (European Committee for Standardization) www.cenorm.be
Sitio de CESSI (Cámara de Empresas de Software y Servicios Informáticos) http://www.cessi.org.ar/
Sitio de CICOMRA (Cámara de Informática y Comunicaciones de la República Argentina) http://www.cicomra.org.ar/
Sitio de Consejo Profesional de Ciencias Económicas de la Ciudad Autónoma de Bs. Aires www.cpcecf.org.ar
Sitio de Compaq http://welcome.hp.com/country/ar/es/welcome.html
Sitio de DIN (Deutsches Institut fur Normung) www.din.de
Sitio de EducAr (Ministerio de Educación, Ciencia y Tecnología) http://software.educ.ar/
Sitio de Epson http://www.epson.com.ar/v4/asp/home.asp
Sitio de HFES (Human Factors and Ergonomics Society) www.hfes.org

Sitio de Hewlett-Packard Development Company
http://welcome.hp.com/country/ar/es/welcome.html
Sitio de IBM http://www.ibm.com/ar/
Sitio de IEC (Comisión Electrotécnica Internacional, IEC: the International Electrotechnical Comisión) www.iec.ch
Sitio de INDEC (Instituto Nacional de Estadística y Censos) http://www.indec.mecon.ar/
www.ingenieriadelsoftware.org/web/programa2.html
Sitio de InfoLEG (Información Legislativa, MECON: Ministerio de Economía y Producción) http://www.infoleg.gov.ar/
Sitio de (The International Certification Network) www.iqnet-certification.com
Sitio de IAF (International Accreditation Forum, Inc.) http://www.iaf.nu/
Sitio de Iatca (International Auditor and Training Certification Association)
http://www.iatca.com/
Sitio de ICONTEC (Instituto Colombiano de Normas Técnicas y Certificación)
http://www.icontec.org.co/
Sitio de IEEE (Institute of Electrical and Electronics Engineers) http://www.ieee.org/portal/site
Sitio de IESE (Institut Experimentelles Software Engineering)
http://www.iese.fhg.de/fhg/iese_DE/
Sitio de IFPUG (International Function Point Users Group) http://www.ifpug.org/
Sitio de IGA (Independent Glass Association) http://www.iga.org/
Sitio de ITBA-CAPIS (Instituto Tecnológico de Buenos Aires-Centro de Actualización Permanente en Ingeniería del Software) www.itba.edu.ar
Sitio de INCITS (InterNational Committee for Information Technology Standars) www.ncits.org
Sitio del IRAM (Instituto Argentino de Normalización) http://www.iram.org.ar/
Sitio de ISA (The Instrumentation, System, and Automation Society) www.isa.org
Sitio de ISO (International Organization for Standardization) www.iso.org
Sitio de JISC (Japanese Industrial Standards Committee) www.jisc.org
Sitio de las Jornadas IDS (Ingeniería y Desarrollo de Software) http://www.jornadasids.net
Sitio de ISO/IEC JTC 1 (Tecnología de la Información) www.jtc1.org
Sitio de OMG (Object Management Group) http://www.omg.com./
Sitio de Plan Estratégico de SSI 2004 –2014 y Plan de acción 2004-2007-10-05 (Ministerio de Economía y Producción)
http://www.industria.gov.ar/foros/soft_inf/documentos/fssi_libro_ayb.pdf
Sitio de SEI (Software Engineering Institute) www.sei.cmu.edu
Sitio del SECYT (Secretaría de Ciencia, Tecnología e Innovación Productiva)
http://www.secyt.gov.ar/
Sitio de Xerox http://www.xerox.com/go/xrx/template/013.jsp?Xcntry=ARG&Xlang=es_AR
Sitio de SQI (Software Quality Institute) http://www.sqi.gu.edu.au/indexFrameset.html
Sitio de SUN Microsystems http://www.sun.com/
http://developers.sun.com/prodtech/index.html
Sitio de Microsoft
http://www.microsoft.com/spanish/MSDN/estudiantes/ingsoft/default.asp
Sitio de LSI (Universidad de Sevilla, Departamento de Lenguajes y Sistemas Informáticos)
http://www.lsi.us.es/is/
Sitio de PRAXIOM (Praxiom Research Group Limited) http://www.praxiom.com/
Sitio de PymesDigital (Posadas Misiones Argentina 2004)

http://www.pymesdigital.com.ar/
Sitio de UNI (Ente Nazionale Italiano di Unificazione) www.unicei.it
Sitio de UNLM (Universidad Nacional de La Matanza, 4ta reunión anual de la Red Iberoamericana de Tecnologías del Software para la década del 2000 (RITOS 2) http://www.unlm.edu.ar/ritos/
Sitio de HP Invent (LSF: Load Sharing Facility) http://hp.com/techservers/software/lsf.html
Sitio de 4abetterbusiness, Inc. http://www.4abetterbusiness.com/services.htm
Sitio del diario La Nación http://www.lanacion.com.ar/
Sitio del diario Clarín http://www.clarin.com/diario/hoy/index_diario.html

7. Anexos

Anexo 1
Normas IRAM de Informática, Computación y Tecnologías de la Información
Fuente IRAM (05/05/2004)

Norma	Título	Págs.	edición	Vigencia
IRAM 2144	Instrumentos eléctricos registradores.	18	1	04/11/50
IRAM-ISO-IEC 17799	Tecnología de la información. Código de práctica para la gestión de la seguridad de la información.	90	1	30/08/02
IRAM 34940	Tarjetas. Clasificación por diseño y por personalización.	5	1	04/11/94
IRAM 34941	Tarjetas de identificación de lectura directa o del tipo magnético. Características físicas.	8	1	02/12/94
IRAM 34942-1	Tarjetas de identificación. Técnicas de registro. Parte 1: Estampado.	32	1	13/11/00
IRAM 34942-3	Tarjetas de identificación. Técnicas de registro. Parte 3: Posición de los caracteres estampados en las tarjetas ID-1.	9	1	14/11/03
IRAM 34943-1	Tarjetas de identificación. Identificación de emisores. Parte 1: Sistema de numeración.	11	1	05/02/02
IRAM 34943-2	Tarjetas de identificación. Identificación de emisores. Parte 2: Procedimientos de solicitud y registro.	26	1	20/02/04
IRAM 36001	Procesamiento electrónico de datos. Caracteres a ser perfograboverificados.	4	2	01/03/78
IRAM 36004-1	Informática. Terminología de computación. Definiciones de términos fundamentales.	21	1	03/10/86
IRAM 36004-10	Terminología de computación. Términos alusivos relativos a las técnicas y recursos de operación del procesamiento de datos.	30	1	03/12/82
IRAM 36004-11	Informática. Sistemas de procesamiento de la información. Terminología. Unidades de procesamiento.	14	1	05/10/90
IRAM 36004-12	Informática. Sistemas de procesamiento de la información. Equipos periféricos. Definiciones.	51	1	03/11/89
IRAM 36004-13	Informática. Terminología de computación. Graficación por computadoras.	35	1	03/10/86
IRAM 36004-14	Terminología de computación. Confiabilidad, mantenimiento y disponibilidad.	17	1	03/12/82
IRAM 36004-15	Informática. Terminología de computación. Definiciones de lenguajes de programación.	31	1	07/11/86
IRAM 36004-16	Terminología de computación. Teoría de la información. Definiciones.	30	1	07/10/83
IRAM 36004-18	Informática. Procesamiento de la información. Terminología de computación. Definiciones de términos referentes al procesamiento distribuido de datos.	21	1	03/04/87

IRAM 36004-19	Informática. Sistemas de procesamiento de la información. Computación analógica. Definiciones.	19	1	03/11/89
IRAM 36004-2	Terminología de computación. Operaciones aritméticas y lógicas.	49	1	02/04/82
IRAM 36004-20	Informática. Terminología de tecnología de la información. Definiciones referentes al desarrollo de sistemas.	30	1	06/12/96
IRAM 36004-21	Informática. Terminología de computación. Interfaces entre los sistemas computarizados en procesos y los procesos técnicos. Definiciones.	24	1	03/04/87
IRAM 36004-22	Informática. Terminología de máquinas de oficina. Definiciones referentes a calculadoras.	31	1	01/05/87
IRAM 36004-3	Informática. Terminología de computación. Términos seleccionados relativos al material y a la tecnología del procesamiento de datos.	19	1	07/08/87
IRAM 36004-4	Informática. Sistemas de procesamiento de la información. Terminología. Organización de los datos.	19	1	02/11/90
IRAM 36004-5	Informática. Sistemas de procesamiento de la información. Terminología. Términos y definiciones referentes a la representación de los datos.	24	1	02/11/90
IRAM 36004-6	Terminología de computación. Términos y definiciones de preparación y procesamiento de datos.	20	1	05/08/88
IRAM 36004-7	Terminología de computación. Términos y definiciones de programación de computadoras digitales.	80	1	02/07/82
IRAM 36004-8	Informática. Terminología de computación. Sistemas de procesamiento de la información. Control, integridad y seguridad.	31	1	05/08/88
IRAM 36004-9	Informática. Terminología de computación. Definiciones de comunicación de datos.	38	1	07/11/86
IRAM 36005	Procesamiento electrónico de datos. Contenido de las carpetas de sistemas.	10	1	04/11/77
IRAM 36019	Informática. Procesamiento de la información. Juego de caracteres codificados en 7 bits para el intercambio de la información.	31	1	04/07/86
IRAM 36022	Informática. Procesamiento de la información. Símbolos de documentación y convenciones aplicables a los diagramas de flujo de datos, de programación y de análisis y a los gráficos de redes de programas y de recursos del sistema.	34	1	07/08/87
IRAM 36023	Informática. Procesamiento de la información. Pautas para la documentación de sistemas de aplicación basados en el uso de computadoras.	35	1	01/07/88
IRAM 36024	Informática. Sistemas de procesamiento de la información. Interconexión de sistemas abiertos. Modelo de referencia básico.	87	1	01/09/89
IRAM 36025	Informática. Sistemas de procesamiento de la información. Interconexión de sistemas abiertos. Servicio de transporte.	38	1	01/09/89
IRAM 36026	Informática. Interconexión de sistemas abiertos. Convenciones relativas a la definición del servicio de capa del modelo de referencia básica de interconexión	13	1	03/11/89

	de sistemas abiertos (OSI).			
IRAM 36027	Informática. Interconexión de sistemas abiertos. Definición del servicio de red.	56	1	05/04/91
IRAM 36031	Informática. Sistemas de procesamiento de la información. Símbolos y convenciones para diagramas de configuración de sistemas informáticos.	29	1	05/10/90
IRAM 36032	Informática. Interconexión de sistemas abiertos. Notación de sintaxis abstracta uno (ASN.1).	82	1	03/09/93
IRAM 36033	Informática. Sistemas de procesamiento de la información. Interconexión de sistemas abiertos. Reglas básicas de codificación de la Notación de Sintaxis Abstracta Uno (ASN.1) y sus extensiones.	31	1	03/09/93
IRAM 36041	Informática. Código de barras EAN. Información técnica básica sobre numeración y simbología.	36	1	02/08/91
IRAM 36042	Informática. Código de barras EAN. Especificaciones de los procesos de producción, impresión, control y ubicación de los símbolos.	39	1	02/08/91
IRAM 36051	Informática. Sistemas de procesamiento de la información. Documentación para el usuario e información en el embalaje para los consumidores de paquetes de soporte lógico (paquetes de "software").	17	1	06/12/96
IRAM 36052	Informática. Tecnología de la información. Evaluación de productos de soporte lógico ("software"). Características de la calidad y guía para su uso.	18	1	06/12/96
IRAM 36053	Proceso de la información. Tablas de decisión simples	23	1	25/06/99
IRAM 36054	Informática. Tecnología de la información. Guía para la gestión de la documentación del soporte lógico (software).	17	1	15/10/98
IRAM 36056	Tecnología de la información. Asignación y registro de direcciones de red de formato ISO DCC.	14	1	28/12/01
IRAM 36504	Controladores fiscales. Procedimientos operatorios por seguir cuando un comprobante no pueda ser emitido o cuando una vez emitido, su importe no pueda ser percibido.	5	1	04/11/94
IRAM 36505	Controladores fiscales. Bloqueo por recambio de memoria fiscal.	4	1	01/04/94

Anexo 2
Normas ISO de Ingeniería de Software y Tecnologías de la Información
Fuente IRAM (20/05/2004)

– ISO 2382-32 1999-02-01 Information technology Vocabulary Part 32: Electronic Mail First Edition
– ISO 9126-1 2001-06-15 Software Engineering - Product Quality -
– Part 1: Quality Model First Edition; Cancels and Replaces ISO/IEC 9126:1991

– ISO TR 9126-2 2003-07-01 Software Engineering - Product Quality - Part 2: External Metrics First Edition
– ISO TR 9126-3 2003-07-01 Software Engineering - Product Quality - Part 3: Internal Metrics First Edition
– ISO 10746-4 1998-12-15 Information Technology - Open Distributed Processing - Reference Model: Architectural Semantics First Edition; Amendment 1: 12/15/2001
– ISO 12207 1995-00-00 Information Technology - Software Life Cycle Processes First Edition; Amendment 1: 5/01/2002
– ISO FDIS 132512004-01-01 (Draft)Collection of graphical symbols for office equipment
– ISO 14143-2 2002-11-15 Information Technology - Software Measurement - Functional Size Measurement - Part 2: Conformity Evaluation of Software Size Measurement Methods to ISO/IEC 14143-1:1998 First Edition
– ISO TR 14143-32003-04-15 Information Technology - Software Measurement - Functional Size Measurement - Part 3: Verification of Functional Size Measurement Methods First Edition
– ISO TR 14143-42002-08-15 Information Technology - Software Measurement - Functional Size Measurement - Part 4: Reference Model First Edition
– ISO 14598-6 2001-06-01 Software Engineering - Product Evaluation - Part 6: Documentation of Evaluation Modules First Edition
– ISO 14769 2001-05-15 Information technology - Open Distributed Processing - Type Repository Function First Edition
– ISO 15288 2002-11-01 Systems Engineering - System Life Cycle Processes First Edition
– ISO 15414 2002-10-15 Information Technology - Open Distributed Processing - Reference Model - Enterprise Language First Edition
– ISO 15437 2001-08-15 Information Technology - Enhancements to LOTOS (E-LOTOS) First Edition
– ISO 15474-1 2002-10-15 Information Technology - CDIF Framework - Part 1: Overview First Edition
– ISO 15474-2 2002-10-15 Information Technology - CDIF Framework - Part 2: Modelling and Extensibility First Edition
– ISO 15475-1 2002-11-01 Information technology CDIF transfer format Part 1: General rules for syntaxes and encodings First Edition
– ISO 15475-2 2002-11-01 Information technology CDIF transfer format Part 2: Syntax SYNTAX.1 First Edition
– ISO 15475-3 2002-11-01 Information technology CDIF transfer format Part 3: Encoding ENCODING.1 First Edition
– ISO 15476-1 2002-11-01 (Draft)Information Technology - CDIF Semantic Metamodels - Part 1: Foundation First Edition
– ISO 15476-2 2002-11-01 (Draft)Information Technology - CDIF Semantic Meatmodels - Part 2: Common First Edition
– ISO 15504-2 2003-10-15 Software engineering Process assessment Part 2: Performing an assessment First Edition
– ISO 15504-3 2004-01-15 Information technology Process assessment Part 3: Guidance on performing an assessment First Edition
– ISO 15939 2002-07-15 Software Engineering Software Measurement Process First Edition

- ISO 18019 2004-01-15 Software and system engineering Guidelines for the design and preparation of user documentation for application software First Edition
- ISO 19500-2 2003-04-01 Information Technology - Open Distributed Processing - Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP) First Edition
- ISO TR 19760 2003-11-15 Systems engineering A guide for the application of ISO/IEC 15288 (System life cycle processes) First Edition
- ISO 19761 2003-02-15 Software Engineering - COSMIC-FFP - A Functional Size Measurement Method First Edition
- ISO 20926 2003-10-01 Software engineering IFPUG 4.1 Unadjusted functional size measurement method Counting practices manual First Edition
- ISO 20968 2002-12-01 Software Engineering - Mk II Function Point Analysis - Counting Practices Manual First Edition
- ISO FDIS 900032003-07-21 (Draft)Software and system engineering Guidelines for the application of ISO 9001:2000 to computer software