Una Propuesta para la Enseñanza de Aspectos Integrales de la Ingeniería de Requerimientos en las Carreras de Computación

Alejandra Cechich, Juan Manuel Luzuriaga, Rodolfo Martínez
GIISCo Research Group, Departamento de Ciencias de la Computación
Facultad de Economía y Administración
Universidad Nacional del Comahue
{acechich, jluzuria, rodom}@uncoma.edu.ar

Resumen

La enseñanza de la Ingeniería de Requerimientos en particular no es una tarea fácil en el ámbito de las carreras de Computación. En general, los temas relacionados con este tópico se abordan en una única asignatura limitando la enseñanza a un método de modelado y documentación de requerimientos (usualmente siguiendo el paradigma de orientación a objetos); y lamentablemente, asumiendo que el conocimiento sobre la construcción de modelos en sí mismos ya ha sido adquirido. En este artículo, presentamos una propuesta integral que ha evolucionado después de más de quince años de experiencia en la enseñanza de la ingeniería de requerimientos. Nuestra propuesta ha sido aplicada sistemáticamente y revisada a fin de obtener el máximo de transferencia dadas las limitaciones de tiempo que implica disponer de una única asignatura para formar "ingenieros de requerimientos software".

Palabras claves: ingeniería de requerimientos; elicitación; UML; ingeniería de software

Ingeniería de Requerimientos: Un tópico complejo a ser abordado

La ingeniería de software tradicionalmente ha abordado el modelado de requerimientos como un proceso temprano en las fases del ciclo de vida de desarrollo de un producto software (aunque evolucione constantemente). Aunque el énfasis sea meramente técnico y funcional a veces, los procesos para obtener y modelar requerimientos correctos no son fáciles. Los requerimientos pueden ser tan diversos como complejos (desde negocios a aspectos técnicos); asociados a distintos puntos de vista y problemas (con participantes de diversos niveles de una organización); altamente influenciados por aspectos sociales y cognitivos (como diferencias en percepción); etc. Esto hace que las técnicas y métodos necesarios para el modelado de requerimientos sean cada vez más complejos, especialmente si tenemos en cuenta el carácter multi-dominio que deben enfrentar los ingenieros de software. Por ejemplo, crear un producto software requiere identificar necesidades del negocio, traducirlas en una visión y alcance del producto y detectar requerimientos relevantes en ese contexto. Esos requerimientos, tanto funcionales como no funcionales, se traducen en distintos elementos de un modelo; lo que requiere de decisiones en las que intervienen aspectos de priorización de requerimientos (Ej., ¿Cuán importante un requerimiento es? ¿De acuerdo a la perspectiva de qué participantes?); capacidad de cambio (Ej., ¿Cuán modificable es un requerimiento? ¿Puede reemplazarse? ¿Puede ser atendido por productos ya realizados, ej. componentes existentes?); selección de participantes (Ej. ¿Son estos los participantes que deberían asistir en la elaboración de requerimientos? ¿Quién debería participar en calidad de experto?)

Independientemente de las actividades técnicas asociadas a la especificación y documentación de requerimientos, la mayoría de los procesos involucrados requieren de un contacto entre

personas definido en un espacio comunicacional. En otras palabras, los procesos relacionados con la ingeniería de requerimientos demandan el establecimiento de una comunicación efectiva – independientemente de que esos procesos sean de carácter distribuido o no. Lograr una comunicación efectiva no es tarea fácil y es esencial establecer pautas de gestión de los procesos para facilitar la comunicación entre los participantes. En estos casos, el conocimiento de pautas sobre flujos de información en una organización podría ser de utilidad, especialmente durante la elicitación de requerimientos [11].

De la misma manera, existen factores que influencian la elaboración de modelos y que afectan indirectamente la calidad de la especificación producida. Por ejemplo, las diferencias en interpretación de conceptos debido a percepciones diferentes de una realidad común, hace que el nivel de consenso entre los participantes disminuya – o aún que no se alcance consenso en absoluto [2].

Un esfuerzo a escala global

Para abordar la diversidad y complejidad de la enseñanza de temas como los enunciados en la sección anterior, diversas universidades y organizaciones en todo el mundo han establecido programas para la enseñanza de tópicos de ingeniería de software en general y de ingeniería de requerimientos en particular, que complementan aspectos sociales y técnicos. En ese sentido, para apoyar mediante guías para el diseño de curricula efectiva, tanto IEEE Computer Society[1] como ACM[2] han desarrollado el conjunto de recomendaciones SE2004 [9]. Este documento comienza con una presentación de la ingeniería de software como unidad disciplinaria tanto de aspectos científicos como de ingeniería. Luego remarca los principios que guían el desarrollo del documento y describe el perfil deseado en el estudiante introduciendo a continuación lo que se conoce como SEEK (Software Engineering Education Knowledge) – el cuerpo de conocimiento que debería enseñarse en el área Ingeniería de Software.
En particular, los tópicos referidos a ingeniería de requerimientos son abordados en un total no menor a 53 horas abarcando los siguientes sub-tópicos: Principios y lenguajes de modelado, sintaxis vs. semántica, modelado de comportamiento, estructural y de dominio, análisis de correctitud, completitud y calidad (atributos no funcionales); e ingeniería de requerimientos propiamente dicha (elicitación, especificación, validación, etc.).
En un esfuerzo similar, el SWEBOK (Software Engineering Body of Knowledge) [15] detalla las áreas principales a ser abordadas durante la Ingeniería de Requerimientos: Elicitación, Análisis, Especificación y Validación. También incluye apartados con aspectos fundamentales del proceso como definiciones, participantes, etc.
Finalmente, diversas recomendaciones para la conformación de curricula en el área ingeniería de software destacan la importancia, casi al punto de necesidad, de la existencia de un proyecto real asociado a la industria durante el proceso de aprendizaje [13][16]. La motivación es clara: el estudiante debería ser preparado para "el mundo real", que suele ser más complejo, con inconsistencias y cambiante que una simulación controlada en aulas.

Tomando como base estas recomendaciones, así como lectura básica en el área y experiencia en casos reales, durante más de quince años hemos incrementalmente construido una estrategia para la enseñanza de conocimientos ligados a la ingeniería de requerimientos, siguiendo principios que entendemos básicos para la incorporación no sólo del concepto sino de su uso. Estos principios han sido aplicados tanto con métodos de análisis estructurado (experiencias a principios de los '90), como a métodos orientados a objetos (experiencias a partir de

[1] www.ieee.org
[2] www.acm.org

1998). En términos generales, hemos basado el proceso de enseñanza en:

- Establecimiento de dos tipos de práctica de conceptos: en aula y en caso de estudios reales.
- Establecimiento de ciclos teórico-prácticos guiados, de manera que cada nuevo conocimiento tenga inmediata aplicación y debate. Dependiendo del tema y momento particular, los ciclos se establecen entre profesor-estudiante o profesor-estudiante-participante caso de estudio.
- Incorporación de conceptos relacionados de manera incremental y como evolución natural del proceso.
- Establecimiento de relaciones entre conceptos desde lo general a lo particular (ej. relaciones entre canales de comunicación en una organización y determinación de procesos para la elicitación de requerimientos).
- Establecimiento del proceso en dos estadios diferenciados y relacionados, abordando problemáticas diferentes: modelado y especificación.
- Validación mediante experimentación en casos de estudio.

Para clarificar el proceso, las siguientes secciones describen las técnicas de enseñanza-aprendizaje para los dos estadios, detallando aspectos y ejemplificando los principios enunciados. Para la aplicación en el caso de estudio, en ambos casos, los estudiantes tienen a disposición un conjunto de técnicas y métodos; guías específicas sobre el estadio en particular; y supervisión directa por parte de un tutor (Figura 1 – (1)(2)(3)). Como producto del aprendizaje, se generan dos elementos diferentes pero integrados: el Modelo y la especificación (Figura 1- (4)(5)).

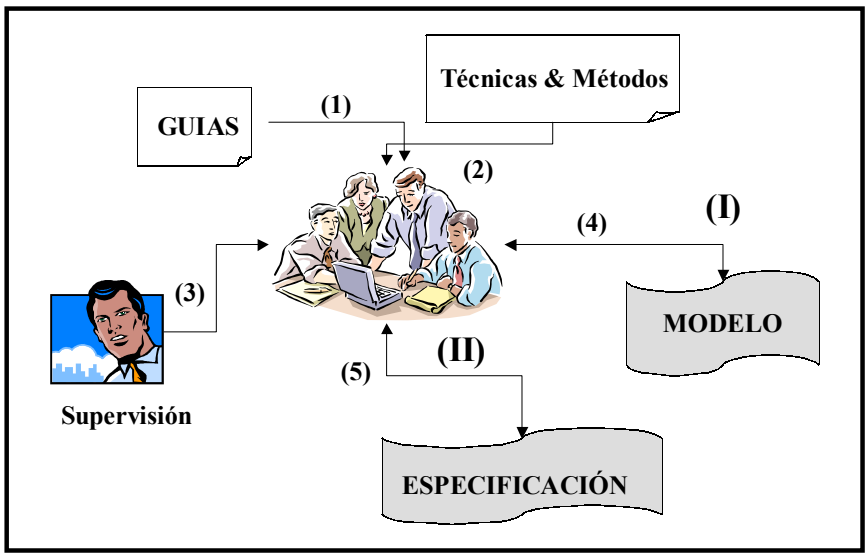

Figura 1. Claves del Proceso de Trabajo de Campo

Enseñanza Paso-a-paso: del dominio al modelo

En este estadio, el énfasis se establece en mejorar la capacidad del estudiante para identificar aspectos relevantes de la realidad, en relación a objetivos específicos. En preparación a su futuro rol en la obtención de requerimientos, asumimos como esencial el reforzar las habilidades relacionadas con los procesos de abstracción/compresión de datos relevantes (entendiendo aquí como "dato" a todo elemento de la realidad que pueda afectar al producto software – desde restricciones de dominio a información propiamente dicha).

En esta primer fase, el estudiante se enfrenta con una cantidad de datos que requieren de técnicas para gestionar su complejidad, al mismo tiempo que debe tomar decisiones sobre la relevancia de los mismos con respecto al modelo que debe ser construido.
En este sentido, incorporamos técnicas que favorezcan el pensamiento creativo y que sirvan para estructurar modelos, conceptuar y resolver problemas. En general, se distinguen cuatro fases de la creatividad: preparación, incubación, iluminación y verificación [12]. De ellas, este estadio hace énfasis en incubación e iluminación, ya que son las fases que generan ideas creativas o innovadoras. En cuanto a los tipos básicos de proceso creativo adoptado, el estadio favorece una creatividad exploratoria (explora un posible espacio de soluciones y descubre nuevas ideas) y de combinación (combina ideas que ya existen para crear nuevas soluciones). Este último tipo favorece especialmente la incorporación de principios sobre reusabilidad de productos y soluciones software.
Específicamente sobre el proceso de construcción de modelos, se proporciona en esta fase una guía clara sobre alternativas y consecuencias de las decisiones adoptadas. Por ej., comparación de modelos resultantes al aplicar procesos de abstracción o de compresión sobre un mismo dominio dado. Por otra parte, los dominios son lo suficientemente diversos como para estimular al estudiante desarrollando su capacidad de comprensión y de interpretación de conceptos. Por ejemplo, la construcción de modelos con datos de ámbitos artísticos (pintura, poesía), científicos (geología, arqueología), etc., favorece la separación de los métodos del ámbito del software, identificando los procesos por sus características intrínsecas.

Una visión integral del proceso de obtención de requisitos software debe estimular también la capacidad de evaluación de prioridades y de verificación del modelo construido [5][10][14]. El contraste de resultados se promueve mediante evaluaciones periódicas en una relación tutor-aprendiz (más que tutor-estudiante). El sentido de este proceso es transmitir no sólo el conocimiento asociado a la formulación de modelos sino favorecer la discusión constructiva, cuestionando diversos aspectos y ayudando en la formulación de criterios. En otras palabras, es un mecanismo para transferir conocimiento sobre el proceso de modelado, no limitando la enseñanza a la mera transmisión de técnicas.
Finalmente, se espera que como resultado de este estadio el estudiante haya adquirido la capacidad de establecer objetivos y alcance, valorar elementos del dominio y abstraer aquellos relevantes.
Al mismo tiempo que se introduce al estudiante en este estadio, se definen prácticas en industria en las que se llevarán a cabo aplicaciones en campo. Para ello, el tutor nuevamente tiene un rol protagónico en la guía suministrada al grupo de estudiantes (los trabajos de campo deben obligatoriamente ser grupales para estimular el proceso de comunicación efectiva y la discusión en toma de decisiones). De esta manera, se selecciona una pequeña o mediana empresa de la zona para realizar las tareas de ingeniería de requerimientos que se distribuyen a manera de guías (con fechas y entregables).

Como parte esencial del proceso de obtención de requisitos, se agregan en este estadio conceptos y prácticas sobre gestión de comunicación en grupos, dinámica y establecimiento de relaciones, y flujos de información en las organizaciones [3][11]. Estos conceptos facilitan el abordaje del trabajo de campo en la primer fase en la que se establecen roles, procedimientos de trabajo y responsabilidades.

En síntesis, al finalizar este estadio de aprendizaje el estudiante está capacitado para realizar el proceso de obtención de requerimientos, ya que ha tomado decisiones técnicas y de gestión del proceso y aplicado esas decisiones a un caso de estudio en campo.

Enseñanza paso-a-paso: del modelo a la especificación

El comienzo de este segundo estadio depende de la evolución del grupo de estudiantes en particular. Es decir, en algunos casos debe esperarse a concluir el modelo antes de abordar su especificación; en otros casos, es posible solapar ambos estadios parcialmente e iniciar la especificación durante el modelado. Aquí debemos llamar la atención a los riesgos que implica el establecer el límite entre ambos estadios. Es importante que se reconozca la diferencia entre modelado conceptual y representación o especificación como dos procesos diferentes, incluso con técnicas distintas. Una vez establecido esto, y asegurado que los estudiantes han aprehendido esas diferencias, entonces es posible abordar ambos procesos parcialmente en paralelo. De otra manera, si no quedan claramente establecidos, se corre el riesgo de que el estudiante confunda un modelo con su representación (lo que produciría por ejemplo "ingenieros de requerimientos UML" solamente).

Es así que la especificación se aborda desde un punto de vista general, incluyendo estándares (ej. IEEE 830 [7], UML [1]). Se analizan alternativas y se describe (por una cuestión de estado de la práctica actual) la notación UML en profundidad. De nuevo, se promueve un debate crítico sobre el uso de las herramientas suministradas por el lenguaje UML (diagramas, etc.) y se aplican en el caso de estudio en campo. El tutor acompaña al grupo de estudiantes con revisiones y guía permanente a fin de favorecer la incorporación de criterios.

En síntesis, al finalizar este estadio de aprendizaje el estudiante está capacitado para realizar el proceso de especificación de requerimientos ligado a un proceso de modelado evitando el riesgo de valorar la "especificación por sí misma". Al mismo tiempo, se espera haya adquirido criterios para evaluar especificaciones alternativas discutiendo ventajas y desventajas con un espíritu crítico. En este último aspecto, es de especial importancia no sólo conocer las posibilidades que brindan diversas notaciones [4][6] sino la relación existente entre la especificación y la capacidad de verificación del producto software, ligada a atributos de calidad estándares [8].

Experiencias

Aplicando los estadios anteriores y sujetos a los tópicos correspondientes al programa que hemos elaborado para mantener actualizada la enseñanza (ver Anexo I), hemos realizado experiencias que han permitido ajustar la propuesta hasta llegar a la que presentamos en este artículo.

Experiencias Fase 1: nuestro primer conjunto de experiencias (desde principios de los '90) se centró en cómo transferir no sólo conceptos sobre modelado funcional o de objetos, sino la complejidad inherente a los diversos dominios de aplicación que un ingeniero de requerimientos debe enfrentar. Propusimos desarrollar los conceptos de "agregación" y "desagregación" no relacionando el hecho de modelar con el de producir un sistema software. Por ejemplo, uno de los primeros modelos a ser construido correspondía a la identificación de funciones realizadas al extraer un dinosaurio del sitio donde había sido hallado; o la identificación de objetos en una descripción de cuadros de la época impresionista. De esta manera, obligamos al futuro ingeniero a enfrentarse con problemas relativos a comprensión del dominio, objetivo del estudio del dominio (modelar funciones u objetos), incorporación de vocabulario, y gestión de complejidad. Las descripciones no estructuradas (generalmente en texto) potenciaban la discusión entre ventajas y desventajas de una construcción de modelos top-down o bottom-up, así como la diferencia entre el proceso de construcción de modelos y su representación. La comparación entre los resultados producidos aportaba ventajas adicionales: mostrar la diversidad de posibles

interpretaciones y la ambigüedad del proceso de captura de requerimientos, siempre dependiente de la percepción y grado de conocimiento del dominio.

Como conclusión, estas experiencias resultaron lo suficientemente exitosas como para continuar aplicándose a lo largo de estos años, con mínimas variaciones.

Experiencias Fase 2: en paralelo a las experiencias de Fase 1, comenzamos experiencias en el sentido de obtención de requerimientos para casos de estudio. Estas experiencias, a diferencia de las anteriores, sí necesitaron adaptación y cambios constantes. Es que realizar un caso de estudio conlleva a la necesidad de interactuar con participantes que se incorporan al proceso de enseñanza-aprendizaje pero que proceden de la industria local. Aunque en un primer momento evaluamos la posibilidad de "simular" la participación real de la industria, lo descartamos inmediatamente por entender que el proceso de aprendizaje sería inmensamente más rico si se basaba en casos reales.
Para llevar adelante casos de estudio, establecimos pautas en su elaboración:

1. Los casos deberían ser en empresas de la zona, de pequeña envergadura o mediana limitando el área de estudio.
2. El dominio a ser modelado debería ser de complejidad mediana y de tipo comercial.
3. La cantidad y/o complejidad del sistema debería poder abarcarse en el tiempo calendario indicado para la asignatura.

Estas tres pautas fueron (y son actualmente) cuidadosamente evaluadas por docentes de la asignatura. Con respecto a la pauta (3), se realizaron diversas experiencias; ej. duración mayor (el doble) del tiempo calendario de la asignatura; duración igual al tiempo calendario de la asignatura incluyendo exámenes finales; duración exactamente igual al tiempo calendario de la asignatura (sin incluir exámenes finales). De estas variaciones, actualmente adoptamos la última, aunque debemos hacer notar que la posibilidad de llegar a buenos resultados en el modelado de requerimientos durante el transcurso de una asignatura no sólo depende de las pautas enseñanza-aprendizaje sino de la experiencia del personal docente en la aplicación de las mismas. Es por ello que estas experiencias de Fase 2 resultan flexibles y la recomendación es adaptar su duración dependiendo del contexto de enseñanza-aprendizaje del sitio donde quieran aplicarse.

Experiencias Fase 3: Un tercer conjunto de experiencias se refirió a cómo establecer pautas de especificación que lleguen a un conjunto de documentos que puedan ser tomados como items de configuración de software. El motivo subyacente fue la visión integral del producto realizado en la etapa de requerimientos, como un elemento de entrada para la construcción de la arquitectura software (a realizarse en asignaturas posteriores). Después de una cuidadosa selección que tuvo en cuenta relaciones de complejidad-utilidad-tiempo, se decidió dividir la realización de la especificación en etapas, con entregables bien establecidos.

Experiencias Fase 4: Por último, estas experiencias apuntaron a la correcta selección de participantes y a la recepción efectiva de guías y procedimientos por parte de docentes de la asignatura. La figura de tutor de cada grupo para trabajos de campo fue esencial para llevar a cabo las metas fijadas en ambos estadios. Sin embargo, las posibilidades de distribución y tipo de entregables variaron (y a veces oscilaron) durante un período considerable antes de estabilizarse. Creemos que haber llegado a un estado estable, con pautas que varían muy poco entre una edición y otra del curso, se debe principalmente a: (1) la maduración alcanzada siguiendo los mismos objetivos de enseñanza en cuanto a la relación entre técnicas aplicadas y aprendizaje; y (2) la consolidación de docentes con conocimientos, experiencia y visión compartida. Aunque

durante quince años, como es de esperar, ha habido cambio de docentes en el área, y en consecuencia de tutores, la dirección del equipo se ha mantenido estable y coordinada hacia un crecimiento sostenido. En este proceso coordinado es esencial el trabajo cooperativo de todo el equipo docente alineando los esfuerzos en la consecución de una visión compartida.

Desafíos y Limitaciones

La enseñanza de la ingeniería de requerimientos no es ajena a los cambios a los que está sometida la enseñanza de la ingeniería de software en general. Nuevos paradigmas de desarrollo (orientado a servicios, ágiles, colaborativo, etc.) y mayor complejidad en los productos (ej. sistemas móviles, a escala global, etc.) desafían nuestra capacidad para formar ingenieros de requerimientos competentes.

Incorporar esta mayor complejidad implica que los ingenieros de requerimientos no sólo serán capaces de abordar la construcción y especificación de modelos (según los estadios presentados en este artículo), sino que podrán incorporar en cada caso factores de análisis pertinentes. Esto puede llevar a que nos cuestionemos sobre la necesidad de redefinir los estadios aquí propuestos (con las limitaciones temporales inherentes a los cursos curriculares); o a evaluar alternativas como incorporar cursos adicionales en la forma de orientaciones o especializaciones. También queda abierto el debate a si ya es tiempo de un cambio de este tipo para curricula de grado o si temas más complejos deberían ser materia de enseñanza en postgrado.

El aspecto práctico es un desafío a la vez que un riesgo. La relación con empresas donde realizar estudios de campo se ha ido construyendo en estos años hacia una conciencia compartida sobre la necesidad de establecer procesos teórico-prácticos reales durante el aprendizaje. Sin embargo, no es sencillo a veces que una organización destine recursos (especialmente tiempo) para actividades propias de la obtención y modelado de requerimientos. Es responsabilidad de ambas partes (academia-empresas) el establecer pautas claras que establezcan alcances, compromisos y beneficios para ambas partes. En nuestra experiencia, la cantidad de empresas que están dispuestas a participar del proceso es importante, incluso cuando el beneficio que reciben no es monetario. Es indudable que existe un beneficio de todas maneras para ambas partes: la enseñanza se ve favorecida por la posibilidad de experimentación real, y las empresas por la posibilidad de que sean aplicadas técnicas de manera guiada por tutores experimentados, lo que en muchos casos permite mejorar sus propios procesos de desarrollo de software o incluso iniciar uno a través de las prácticas realizadas.

Finalmente, debemos destacar que la formación de los tutores es también un aspecto clave. Los tutores han sido cuidadosamente seleccionados y entrenados en su papel. En principio, la recomendación ha sido contar con tutores que no sólo tengan conocimientos del tema tutelado, sino experiencia en el mismo. Luego de seleccionados, los candidatos a tutores pasan un período de entrenamiento a cargo de un "tutor senior" quién los introduce en los aspectos propios del seguimiento del trabajo de campo. Finalmente, una vez pasado su período de entrenamiento, su primera asignación como "tutores junior" es supervisada por un "tutor senior". Un tutor pasa a la categoría senior después de haber tenido al menos un año de tutelas exitosas

Conclusiones

En este artículo hemos presentado una propuesta para la enseñanza de aspectos integrales de la ingeniería de requerimientos, tanto técnicos como sociales. La propuesta se basa en la experiencia recogida durante los últimos quince años en la enseñanza de estos

tópicos en el marco de carreras de Ciencias de la Computación.

Hemos detallado también desafíos y limitaciones a la propuesta. Sin embargo, el éxito alcanzado nos permite confiar en que la base metodológica adoptada es lo suficientemente flexible y adecuada para sostener cambios que puedan producirse en el futuro. En esencia, la base metodológica ha sido la misma y ha sido utilizada tanto para el paradigma estructurado como para el orientado a objetos, lo que es una muestra de su bondad en la capacidad de adaptación.

Por supuesto, esa adaptación no es sencilla y requiere pruebas y ajustes en muchos casos (y especial atención a puntos clave), pero las consecuencias son lo suficientemente valiosas como para invertir en ese esfuerzo. Los resultados de aplicar esta base metodológica han mostrado que los ingenieros de requerimientos formados a través de ella se muestran receptivos ante cambios, con mayores ventajas para llevar adelante un proceso creativo y con capacidad crítica para fundamentar sus decisiones.

Esperamos que compartir nuestras experiencias contribuya al debate en la comunidad sobre cómo debemos instrumentar procesos de aprendizaje en áreas tan complejas y multi-disciplinarias como la ingeniería de requerimientos.

Referencias

[1] Booch G., Rumbaugh J., Jacobson I., *El Lenguaje Unificado de Modelado*,– Addison Wesley, 1999

[2] Cechich A., Luzuriaga J., Martínez R., *Classifying Requirements Perception to Guide Exploratory Processes in Requirements Elicitation,* V Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software, IDEAS 2002, La Habana, Cuba, 23-26 Abril 2002.

[3] Ciamberlani, *Comunicación para la transparencia* – Granica Ed., 1999

[4] Davis A., *Software Requirements: Objects, Functions & States*, – Prentice Hall, 1993

[5] Freeman, *Software Perspectives* - Addison Wesley, 1987

[6] Ghezzi, *Fundamentals of Software Engineering* – Prentice Hall, 1991

[7] *IEEE 830-1998 std.* Recommended practice for software requirements specification, http://ieeexplore.ieee.org/xpl/tocresult.jsp?isNumber=15571

[8] ISO International Standard ISO/IEC 9126. ISO/IEC 9126 - Information technology - Software product evaluation - Quality characteristics and guidelines for their use, 2001

[9] Lethbridge T., LeBlanc R., Sobel A., Hilburn T., Diaz-Herrera J., SE2004: Recommendations for Undergraduate Software Engineering Curricula, *IEEE Software special issue Curriculum Development*, Nov/Dec 2006, pp. 19-25.

[10] Loucopoulos P. and Karakostas V., *System Requirements Engineering*, Mc Graw-Hill, 1995

[11] Luzuriaga J. Martínez R., Cechich A., *Managing Enterprise Communication Networks to Improve the Requirements Elicitation Process,* ICEIS 2002, 4th International Conference on Enterprise Information Systems, Ciudad Real, España, 3-6 Abril 2002, pp. 770-775.

[12] Maiden N. and Grizikis A., Where Do Requirements Come From?, *IEEE Software*, 18(5), 2001, pp. 10-12.

[13] Saiedian H., Bagert D., Mead N., Software Engineering Programs: Dispelling the Myths and Misconceptions, *IEEE Software special issue Educating Software Professionals*, Sept/Oct 2002, pp. 35-41.

[14] Sommerville I., Integrated Requirements Engineering: A Tutorial, *IEEE Software*, Jan/Feb 2005, pp. 16-23

[15] *SWEBOK* – Guide to the Software Engineering Body of Knowledge, www.swebok.org

[16] Van Vliet H., Reflections on Software Engineering Education, *IEEE Software*, May/June 2006, pp. 55-61.

Anexo I

Unidad I
Formulación de modelos. Sistemas. Definición y conceptos. Ingeniería de Sistemas e Ingeniería de Software. El Software y su importancia. Características, componentes y aplicaciones del Software. La crisis del software.

Unidad II
Principios de la Ingeniería de Software. Abstracción. Técnicas de agregación y desagregación. El proceso de Desarrollo de Software. Definiciones y Conceptos (SDS). Planificación y control de un proceso de Desarrollo de Software. Organización, Staffing y Dirección

Unidad III
Elicitación de requerimientos. Especificación de requerimientos de software (SRS). Lenguaje de Modelado Unificado (UML). Principales enfoques para la especificación de software Orientado a Objetos

Unidad IV
Relación Cliente-Desarrollador. Análisis participativo. Calidad en el software. Definición y Clasificación. Requerimientos de calidad en diferentes. Áreas de aplicación. Medición de la Calidad.

Unidad V
La estructura del proceso de Software. Meta-modelos, modelos, enfoques, métodos, herramientas, entornos. Ciclos de vida para el desarrollo de software. Modelos de ciclos de vida en cascada, estructurado, Incremental, Evolutivo, Prototipos rápidos y evolutivos. Modelo Transformacional. Modelo Espiral. Uso del UML en las distintas etapas del ciclo de vida según el modelo

Unidad VI
Análisis y gestión de riesgo. Comunicación en las organizaciones. Comunicación del analista