1. INTRODUCCIÓN.

No existe un único enfoque para mejorar el mal del software. Sin embargo, mediante la combinación de métodos completos para todas las fases del desarrollo del software, mejores herramientas para automatizar estos métodos, bloques de construcción más potentes para la implementación del software, mejores técnicas para garantía de calidad del software y una filosofía predominante para coordinación, control y gestión, podemos conseguir una disciplina para el desarrollo del software – una disciplina llamada ingeniería del software.

Roger S. Pressman

La ingeniería de software abarca un conjunto de tres elementos claves [Pressman]: métodos, herramientas y procedimientos. Los métodos indican "cómo" construir técnicamente el software y abarcan un amplio espectro de tareas que incluyen: planificación y estimación de proyectos, análisis de los requerimientos del sistema y del software, diseño de estructuras de datos, arquitectura de programas y procedimientos algorítmicos, codificación, prueba y mantenimiento. Las herramientas suministran un soporte automático o semiautomático para los métodos mencionados anteriormente. Los procedimientos de la ingeniería de software son los que permiten relacionar los métodos y las herramientas para facilitar un desarrollo racional y oportuno del software.

De acuerdo con Boehm [Boehm79], la ingeniería de Software incluye la aplicación práctica del conocimiento científico en el diseño y construcción de programas para computadoras y la documentación asociada requerida para desarrollarlos, operarlos y mantenerlos. Otra definición de Ingeniería de Software es la entregada en el glosario de la IEEE, titulado "Standar Glossary of Software Engineering Terminology" [IEEE 90], en el cual se define como el enfoque sistemático para el desarrollo de operación, mantenimiento y eliminación del software donde software se define como aquellos programas, procedimientos, reglas y documentación posible asociada con la computación, así como los datos pertenecientes a la operación de un sistema de cómputo.

La Ingeniería de Software surge como disciplina en la década de los sesenta, cuando se hacen presentes los graves problemas existentes en la producción y, sobre todo, mantención de software, situación que se conoce como la "crisis del software". Las prácticas "artesanales" de programación y la escasa documentación que se utilizaban en ese entonces hicieron crisis y determinaron que se pensara en el desarrollo de software como un problema que necesariamente debería ser abordado de manera más sistemática. El modelo básico de esta sistematización corresponde a una variación de un modelo desarrollado en la década de los treinta en los Laboratorios Bell, método que en el ámbito del software se conoce como el ciclo de vida tradicional. Históricamente han surgido varios enfoques que buscan abordar de manera sistemática, la planificación, análisis, diseño e implementación de los proyectos de desarrollo de software, sean estos de gran escala y pequeñas aplicaciones, software a medida o productos de software. Cada uno de estos enfoques tiene su raíz en las preconcepciones dominantes en su época y, sobre todo, en la búsqueda incesante de mejoras a los enfoques precedentes.

Una constante que se observa en esta búsqueda, es que siempre incluyen una fase para alcanzar un acuerdo explícito sobre responsabilidades del cliente y del desarrollador, fase que conocemos como Especificación de Requerimientos.

Entre las tareas que hay que realizar en esta fase, está la definición de los requerimientos del software y del sistema del que el software forma parte, esta tarea debe realizarse al comienzo del proyecto, pero el principal problema que se nos presenta es que, en estos momentos iniciales, es difícil tener una

idea clara (o al menos, es difícil llegar a expresarla), de cuáles son los requerimientos del sistema y del software, y llegar a comprender en su totalidad la función que el software debe realizar. Por esto, algunos de los modelos de ciclo de vida proponen enfoques cíclicos de refinamiento de los requerimientos o incluso de todo el proceso de desarrollo de software.

El análisis de requerimientos es el primer paso en el proceso de ingeniería del software. Es aquí donde se refina la declaración general del ámbito del software en una *especificación* concreta que se convierte en la base de todas las actividades de ingeniería del software que siguen cumpliendo con los siguientes dos roles [Norris]:

- Proporciona la primera entrada para la fase de diseño.
- Da un lineamiento contra el cual las pruebas de aceptación son llevadas a cabo.

La forma de especificar un sistema tiene una gran influencia en la calidad de la solución implementada finalmente. Es ampliamente aceptado que el costo de corregir-modificar un sistema después de su instalación, o incluso después de las primeras etapas de la fase de diseño, es mucho mayor que el costo de preparar una especificación de requerimientos inicial [Davis].

Tradicionalmente los ingenieros de software han venido trabajando con especificaciones incompletas, inconsistentes o erróneas, lo que invariablemente lleva a la confusión y a la frustración en todas las etapas del ciclo de vida. Como consecuencia de esto, la calidad, la corrección y la completitud del software disminuyen.

Existen varios enfoques para analizar los requerimientos, dependiendo del ciclo de vida de desarrollo que se esté utilizando. Cada enfoque implica diferentes necesidades en términos del tipo de información que se captura. El diseño centrado en el punto de vista del usuario para definir los requerimientos es el enfoque al que adhieren recientes metodologías, por ejemplo en [Hsia94], [Rubin92] y [Jacobson92], que proponen la especificación de "casos de uso" los cuales son una descripción de las acciones de un sistema desde el punto de vista del usuario, los cuales complementan satisfactoriamente las características que propone el mencionado enfoque. Durante esta etapa la interacción con el usuario es indispensable, por lo cual estas herramientas deben ser relativas al lenguaje que ellos manejan para facilitar la comunicación.

En [Leite90], [Leite95], [Leite97] se propuso una metodología que propone trabajar con una documentación integrada en una estructura llamada Requirements Baseline, que acompaña al proceso de desarrollo de software. El elemento central de esta metodología es el LEL (Language Extended Lexicon) para modelar el vocabulario de un macrosistema y propone el uso de Escenarios para representar el comportamiento.

1.1. Motivación.

La Ingeniería de Requerimientos es el primer paso esencial para entregar lo que el cliente desea, ya que enfoca un área fundamental: la definición de lo que se desea producir. Como ya se mencionó, proporciona la primera entrada para la fase de diseño y da un lineamiento contra el cual las pruebas de aceptación son llevadas a cabo.

Se debe involucrar efectivamente a los usuarios, para conseguir identificar necesidades y/o problemas específicos y se puedan establecer mecanismos de resolución adecuados y apoyar cada una de las fases en sólidos principios de comunicación humana. La ingeniería de software como disciplina ha evolucionado significativamente en lo que se refiere a modelos conceptuales y herramientas de trabajo, que hacen del proceso de desarrollo y mantenimiento de software una actividad cada vez menos dependiente del arte de quienes llevan a la práctica un diseño elaborado.

Un enfoque novedoso para sistematizar el proceso de análisis de requerimientos, se propone en **A client oriented requirements baseline** [Leite 90], que está compuesto por el LEL (Language Extended Lexicon) que permite representar y documentar, con tecnología hipertextual un conjunto de símbolos que representan el lenguaje de la aplicación y por los Escenario que hacen una descripción parcial del comportamiento de la aplicación en un momento específico.

El objetivo principal de este Trabajo de Tesis esta centrado en la creación de una herramienta automatizada para soportar este enfoque, es decir, que permita implementar el LEL, con tecnología de hipertexto, que soporte las relaciones dinámicas entre los símbolos. También realizará una derivación de escenarios sobre la base de las heurísticas desarrolladas en el trabajo de investigación de [Hadad96], cuyas conclusiones denotan la necesidad de una herramienta de este tipo.

Con el propósito de asistir al desarrollador en el proceso de documentación de requerimientos, la herramienta también incorpora un editor de texto para documentar el Universo del Discurso, crea link a símbolos y escenarios en forma automática, realiza controles de integridad cuando se modifica o elimina algún símbolo o escenario existente, permite consultas de símbolos por categoría y estados, administra distintas versiones para un proyecto, exporta la información de la versión a formato html para que pueda ser recorrida por medio de navegadores de internet ordinarios o en el navegador incorporado en ella, todas estas prestaciones, se analizara en detalle en el capitulo 5 de esta tesis.

Este trabajo se centró en la fase de elicitación de requerimientos, utilizando una metodología basada en el uso del Léxico Extendido del Lenguaje (LEL) y Escenarios. Utilizó como caso de estudio, para ejemplificar y verificar las heurísticas existentes, el Sistema de Registración y Producción del Instituto de Hemoterapia de la provincia de Buenos Aires, el cual resume una experiencia en

la que el Lenguaje del dominio contiene un gran volumen de palabras de uso muy específico que se podría denominar “lenguaje técnico", de difícil comprensión para los desarrolladores.

1.2. Publicaciones y trabajos relacionados.

Los siguientes artículos publicados y trabajos de investigación son algunos de los resultados obtenidos respecto al tema de esta tesis:

Director en el Trabajo de Investigación “ESTUDIO DE UN CASO REAL APLICANDO LEL Y ESCENARIOS EN EL ANÁLISIS ORIENTADO A OBJETOS”.
Consejo de Investigaciones de la UNSa – Nota Nro. 113 C.I./2000
Período: 01/01/2000 al 31/12/2000.

“Léxico Extendido del Lenguaje y Escenarios del Instituto de Hemoterapia de la Provincia de Buenos Aires”. CIUNSa – Consejo de Investigación de la Universidad de Salta, 2000. Gil, Gustavo, Arias Figueroa, D.

“Gestión General de un Banco de Sangre. Universo del Discurso”, CIUNSa – Consejo de Investigación de la Universidad de Salta, 2000.
Gil, Gustavo, Arias Figueroa, D.

“PRODUCCIÓN DEL LEL EN UN DOMINIO TÉCNICO. INFORME DE UN CASO”. Presentado en WER 2000 – Workshop en Ingenieria de Requerimientos – Río de Janeiro – Brasil – 13 y 14 de julio del 2000.
L.I.D.T.I. – Gustavo D. Gil – Daniel Arias Figueroa – Alejandro Oliveros

“ADQUISICIÓN DE REQUISITOS USANDO LEL Y ESCENARIOS”. 2das. JORNADAS DE INFORMÁTICA DEL NEA. Del 25 al 27 de octubre de 2001 - Corrientes, Argentina. Invitado como Disertante.

“TOOLS FOR THE IMPLEMENTATION OF LEL AND SCENARIOS (TILS)”. Presentado en el marco de la 31 JAIIO – ASSE 2002 (Simposio Argentino de Ingeniería de Software) Santa Fe – Argentina – 9 al 11 de septiembre de 2002.
Gustavo D. Gil – Daniel Arias Figueroa – Loraine Gimson.

“EVALUACIÓN DE UNA HERRAMIENTA PARA IMPLEMENTAR LEL Y ESCENARIOS”. Presentado en el marco del CACIC 2002 (VIII Congreso Argentino de Ciencias de la Computación). Buenos Aires – Argentina. 15 al 18 de octubre de 2002.
Gustavo D. Gil – Daniel Arias Figueroa – Loraine Gimson.

Director en el Trabajo de Investigación “LEL Y ESCENARIOS EN EL ANÁLISIS Y DISEÑO ESTRUCTURADO”.
Consejo de Investigaciones de la UNSa – Nota Nro. 502 C.I./2002
Período: 01/07/2002 al 30/06/2003.

1.3. Estructura de la Tesis.

La tesis se divide básicamente en seis grandes partes: Introducción, Estado del Arte: Ingeniería de requerimientos, Léxico Extendido del Lenguaje, Escenarios, Funcionalidad y Arquitectura de la Herramienta, Utilización de la Herramienta y Conclusiones.

A continuación se describe las restantes partes de lo que sigue a esta tésis.

ESTADO DEL ARTE

Que incluyen los siguientes capítulos:

El capítulo 2, llamado "INGENIERÍA DE REQUERIMIENTOS", en donde se analiza algunos modelos para el análisis de requerimientos, áreas de problemas y técnicas de elicitación.

En los capítulo 3 y 4, "LA INGENIERÍA DE SOFTWARE" se describe cada uno de los pilares del modelo propuesto: Léxico Extendido del Lenguaje y Escenarios, respectivamente.

FUNCIONALIDAD Y ARQUITECTURA DE LA HERRAMIENTA

En capítulo 5 se describe la funcionalidad y arquitectura de la herramienta desarrollada, que automatiza el modelo presentado.

UTILIZACIÓN DE LA HERRAMIENTA Y CONCLUSIONES

En el capítulo 6 se muestra la utilización de la herramienta, se realiza una evaluación contextualizada, contrastando un caso de estudio realizado con la herramienta para implementar LEL y escenarios, con otro desarrollo echo manualmentea.

Finalmente se presentan las conclusiones a esta propuesta y experiencia.

CONTRIBUCIONES

Por último, en el capítulo 7 se describe las contribuciones de esta tesis y se dejan bosquejadas algunas posibles líneas de investigación futuras.