Una propuesta de Análisis de Puntos Función aplicado a LEL y Escenarios

Mabel Angélica Bertolami

Director: Lic. Alejandro Oliveros

Tesis presentada a la Facultad de Informática de la Universidad Nacional de La Plata como parte de los requisitos para la obtención del título de Magíster en Ingeniería de Software

Facultad de Informática

Universidad Nacional de La Plata - Argentina

Junio de 2003

Una propuesta de Análisis de Puntos Función aplicado a LEL y Escenarios

Resumen

En esta tesis se presenta un enfoque para medir la funcionalidad de un sistema de información a partir del modelo de requerimientos basado en Lenguaje Natural. Concretamente, se propone utilizar la documentación del Léxico Extendido del Lenguaje (LEL) y Escenarios para la aplicación del Análisis de Puntos Función (FPA). El objetivo es obtener una medida de la funcionalidad de un sistema desde las primeras etapas del proyecto de software, independientemente de la tecnología usada para el desarrollo o implementación. Para lograr ese objetivo se analizan las distintas alternativas para el cálculo de Puntos Función (FP) desde LEL y Escenarios (L&E), se selecciona un método de FPA y se estudia la relación entre los conceptos del FPA y L&E. Sobre esa base se establecen un conjunto de reglas y procedimientos que soportan el proceso de medición. Para examinar la aplicabilidad del enfoque se utiliza para el cálculo de los FP en varios casos de estudio y se obtienen conclusiones acerca de los resultados.

Índice

1 Introducción 1

1.1 Motivación 1

1.2 Objetivos 1

1.3 Propuestas relacionadas 2

1.4 Organización de la tesis 3

2 Léxico Extendido del Lenguaje y Escenarios 5

2.1 La Ingeniería del Software 5

2.2 La Ingeniería de Requerimientos 7

2.3 LEL y Escenarios 10

2.3.1 LEL 11

2.3.2 Escenarios 13

2.3.3 Ventajas del uso de escenarios en la Ingeniería de Requerimientos 19

2.3.4 Herramientas 20

3 Métricas del Software 24

3.1 Las medidas: qué son y por qué hacerlas 24

3.2 Las métricas del software 25

3.2.1 Beneficios de la aplicación de métricas 27

3.2.2 Medición del tamaño del sistema 28

3.3 Clasificación de las medidas del software 28

3.4 Determinar qué medir 29

3.5 Líneas de código fuente (SLOC) 30

3.6 SLOC vs. Puntos Función 31

3.7 Métodos de Análisis de Puntos Función 32

3.7.1 Introducción 32

3.7.2 Proceso de medición del tamaño funcional 35

3.7.3 FPA y los requerimientos 37

3.7.4 Beneficios de la aplicación del FPA 38

3.7.5 Método de Albrecht 40

3.7.6 Método MarkII 45

3.7.7 Comparación de los métodos de Albrecht y MarkII 50

4 Aplicación del Análisis de Puntos Función a LEL y Escenarios 53
4.1 Introducción 53
4.2 Beneficios de la aplicación del enfoque 53
4.3 Análisis de las alternativas para el cálculo de FP desde L&E y fundamentación de la elección 54
4.4 Modelo de medición del tamaño funcional desde L&E 57
4.5 ¿Qué método de FPA elegir? 58
4.6 Consideraciones respecto al ajuste de complejidad técnica 61
5 Medición de funcionalidad de L&E 63
5.1 Introducción 63
5.2 Medir la funcionalidad 64
5.3 Tamaño funcional y episodios 65
5.3.1 Clasificación de los episodios según su tipo y estructura 67
5.3.2 Restricciones 72
5.3.3 Comparación de los conceptos de transacción lógica y episodio 72
5.3.4 Clasificación de episodios según el procesamiento requerido 74
5.3.5 Componentes de los episodios 74
5.4 El límite del sistema 82
6 Proceso de medición de funcionalidad de L&E 85
6.1 Introducción 85
6.2 Etapas del proceso de medición 86
6.3 Formularios para documentar el proceso 94
6.3.1 Formularios 94
6.3.2 Normas para completar los formularios 95
6.3.3 Plantillas de formularios 97
7 Aplicación del enfoque propuesto 100
7.1 Aplicación del enfoque propuesto al caso de estudio Recepción del Hotel 100
7.2 Aplicación a otros casos 122
8 Conclusiones 130
Bibliografía 133
Anexo 1 Modelo de LEL y Escenarios para un caso de estudio 137
Anexo 2 Modelo de reglas 159

Lista de figuras

Figura 1 – Componentes del desarrollo de software [Loucopoulos'95]

Figura 2 - Procesos de la Ingeniería de Requerimientos [Loucopoulos'95]

Figura 3 - Etapas para la construcción del LEL [Hadad'97]

Figura 4 - Diagrama ER para el Modelo de Escenarios [Hadad'97]

Figura 5 - Ventana principal BMW

Figura 6 - Ventana de la vista Full Browser

Figura 7 - Ventana de edición de una entrada del LEL

Figura 8 - Ventana de edición de un escenario

Figura 9 - Vista de hipertexto para una entrada del LEL

Figura 10 - Vista de hipertexto para un escenario

Figura 11 - Los componentes del tamaño del sistema [Symons'91]

Figura 12 – Ámbito de aplicación de los métodos de FPA [Symons'91]

Figura 13 – Visión del sistema desde la perspectiva del usuario [Goodman'99]

Figura 14 - La evolución de los métodos de medición del tamaño funcional [Fetcke'99]

Figura 15 - Los dos pasos de abstracción de las variantes del FPA [Fetcke'99]

Figura 16 – La abstracción orientada a los datos del FPA [Fetcke'99]

Figura 17 - Componentes del Método de Puntos Función de Albrecht [Symons'91]

Figura 18 - EI que actualiza 2 ILF [Longstreet'96^{a}]

Figura 19 - EO que referencia y deriva información de 2 ILF [Longstreet'96^{a}]

Figura 20 - EQ con 2 ILF y sin datos derivados [Longstreet'96^{a}] [Longstreet'96^{b}]

Figura 21 - Componentes del Método de Puntos Función MarkII [Symons'91]

Figura 22 - Posibles enfoques de aplicación de FPA a L&E

Figura 23 - Modelo de FPA sobre L&E

Figura 24 - Modelo de cálculo de FP

Figura 25 – Esquema de clasificación de los episodios

Figura 26 - Transacciones lógicas [GIFPA'98]

Figura 27 - Clasificación de los objetos del LEL

Figura 28 - Etapas del proceso de medición

Figura 29 - Límite del sistema

Figura 30 - Gráfico de los casos de estudio de la Tabla 16

Lista de tablas

Tabla 1 – Comparación entre los métodos [Symons'91]

Tabla 2 - EI [Longstreet'96ª]

Tabla 3 – EO y EQ [Longstreet'96ª]

Tabla 4 – Transacciones [Longstreet'96ª]

Tabla 5 - Tipo registro [Longstreet'96ª]

Tabla 6 – ILF y EIF [Longstreet'96ª]

Tabla 7 – Cálculo de Puntos Función de Albrecht [Longstreet'96ª]

Tabla 8 – Características Generales del Sistema para Albrecht FPA [Longstreet'96ª]

Tabla 9 - Criterios para distinguir entidades primarias [Symons'91]

Tabla 10 – Características Generales del Sistema adicionales para MKII [Symons'91]

Tabla 11 - Resumen de alternativas – artefactos

Tabla 12 - Resumen de alternativas - proceso

Tabla 13 - Cuadro comparativo de los métodos Albrecht y MKII

Tabla 14 - Matriz de relaciones Etapa - Regla

Tabla 15 - Resumen del uso de formularios por etapa

Tabla 16 - Cuadro resumen de los casos de estudio analizados

Capítulo 1

Introducción

1.1 Motivación

El objetivo de la industria de desarrollo de sistemas de software es producir productos de calidad adecuada, mediante un proceso eficiente y con un costo razonable. Este objetivo encierra por sí mismo los conceptos de productividad, esfuerzo y mejoramiento del proceso. La única manera de disponer de parámetros que sirvan de referencia para hacer un seguimiento del proceso, es a través de la obtención de mediciones de los productos resultantes en las distintas etapas del proceso.

La posibilidad de aplicar alguna técnica de medición en las distintas etapas del desarrollo de software ha sido y es una preocupación de los desarrolladores e investigadores. Tradicionalmente se han utilizado métricas basadas en cantidad de líneas de código, sólo aplicables al producto resultante de la etapa de codificación. Estas métricas no se pueden aplicar a los productos que se obtienen en las etapas iniciales del proceso de desarrollo. La disponibilidad de alguna medida de estos productos es de gran utilidad para hacer estimaciones de tamaño, costo y esfuerzo.

Un enfoque alternativo a la medición de líneas de código es el de medir la funcionalidad del producto y se conoce como Análisis de Puntos Función (FPA). Tiene la ventaja que se puede aplicar a los productos obtenidos a lo largo de todo el ciclo de desarrollo y a medida que se avanza en el proceso se puede ir refinando hasta obtener mediciones más precisas. En el caso particular de esta tesis el producto que se desea medir es LEL y Escenarios (L&E).

El propósito de los métodos de FPA es medir la funcionalidad requerida por el usuario en forma independiente de la tecnología que se utilice para el desarrollo del software. Las aplicaciones de software están representadas mediante un modelo abstracto que contiene los ítems que contribuyen al *tamaño funcional* [Fetcke'98].

En este trabajo se propone un enfoque que permita aplicar los conceptos del modelo de FPA al L&E. Para ello se establecen un conjunto de reglas que sustentan el proceso de medición. El objetivo es demostrar la aplicabilidad del FPA para calcular la funcionalidad de un sistema de software a partir de L&E. Este enfoque complementa el concepto de medición independiente de los métodos o técnicas usados para desarrollar o implementar el software.

1.2 Objetivos

Los objetivos de este trabajo de tesis son:

- Medir la funcionalidad de un sistema a partir de la documentación de L&E.
- Aplicar un método de Análisis de Puntos Función.

- Obtener una medida del sistema al principio de ciclo de vida, en la etapa previa a la Especificación de Requerimientos.

1.3 Propuestas relacionadas

En la actualidad han surgido varias propuestas para aplicar el modelo de Puntos Función (FP) al software orientado a objetos. Éstas se basan en el modelo de *Casos de uso*. Este modelo tiene dos tipos de entidades: *actores* y *Casos de uso*. Los *actores* representan sujetos que interactúan o intercambian información con el sistema. Ellos permanecen fuera del sistema que se describe. Cuando un actor usa el sistema, sigue un curso de acciones relacionadas en su comportamiento con el sistema. Cada secuencia de acciones se llama *Caso de uso* y define una forma específica de usar el sistema. El conjunto de todos los *Casos de uso* representa la funcionalidad total del sistema [Fetcke'98].

La aplicación de FPA a *Casos de uso* resulta muy directa a partir que hay una relación "natural" entre ambos. FPA es un método para medir software desde una perspectiva de los requerimientos y *Casos de uso* es un método para desarrollar requerimientos. Ambos tratan de ser independientes de la tecnología usada para implementar la solución de software. Los *Casos de uso* se usan para validar un diseño propuesto y asegurar que se alcanzan todos los requerimientos. Un beneficio de los *Casos de uso* es intentar definir requerimientos muy temprano en el ciclo de vida. Si los *Casos de uso* se definen al principio del ciclo de vida y se puede aplicar el FPA, entonces las estimaciones del proyecto son más precisas. Puesto que el FPA tiene grandes bases históricas se puede comparar la productividad de usar los métodos de *Casos de uso* con otros métodos [Longstreet'01] [Tavares'02].

Thomas Fetcke, Alain Abran y Tho-Hau Nguyen en su trabajo "Mapping the OO-Jacobson Approach into Function Point Analysis" proponen un mapeo del método OOSE (Ingeniería del Software Orientada a Objetos) basado en *Casos de uso* de Jacobson con el modelo abstracto de Puntos Función. El mapeo está formulado como un conjunto conciso de reglas que sustentan el proceso de medición. Este trabajo demuestra la aplicabilidad del FPA como una medida de la funcionalidad del software para el enfoque OO-Jacobson y soporta la hipótesis que el FPA mide en forma independiente de la tecnología usada para la implementación y puede ser usada en el paradigma orientado a objetos [Fetcke'98].

El FPA fue inventado por Alan Albrecht[1] como un medio para medir aplicaciones de software para gestión independientemente de la tecnología a usar para su desarrollo [Symons'01]. Esta técnica está formulada como un método de varios pasos que se basa implícitamente sobre un modelo de alto nivel de las aplicaciones de software. La medición tiene lugar en el contexto del modelo abstracto. Los ítems en el modelo abstracto son transacciones y archivos. Esos ítems comúnmente se identifican desde los documentos obtenidos por las técnicas tradicionales del diseño estructurado. Los métodos de diseño OO modelan los sistemas de software como colecciones de objetos

[1] Algunos de los trabajos principales fueron [Albrecht'79], [Albrecht'83], pero no han podido ser consultados por la autora. En cada oportunidad que se haga mención a Albrecht, será a través de referencias indirectas.

cooperantes. Los modelos creados con estos métodos son diferentes, particularmente en las primeras etapas. Sin embargo, sigue siendo válido el objetivo de medir la funcionalidad que requiere el usuario. Los distintos métodos OO difieren en la manera de formalizar los modelos. Particularmente el método OOSE define un proceso para transformar los requerimientos en una secuencia de modelos. Los pasos incluyen los modelos de requerimientos, análisis, diseño, implementación y prueba. El modelo de *Casos de uso* es la base para el desarrollo de los modelos [Fetcke'98].

Para aplicar las reglas del FPA a las aplicaciones de software desarrolladas con OOSE, se deben establecer relaciones entre los conceptos y terminologías de FPA y OOSE. Se identifican estas relaciones y luego se transforman en un conjunto de reglas y procedimientos. Hay dos factores que tienen impacto sobre la facilidad de uso del FPA sobre OOSE: el primero de ellos es la facilidad de aplicación del mapeo del modelo OOSE a los conceptos de FPA, es decir cuán fácil resulta usar las reglas y procedimientos diseñados para identificar y medir, desde los documentos OOSE, los componentes que contribuyen al *tamaño funcional*. El segundo es el grado de conformidad entre la documentación del proyecto y los estándares de OOSE tanto como su calidad y completitud. El proceso de medición es muy dependiente de la calidad y completitud de la documentación del proyecto [Fetcke'98].

Desde lo expuesto se observa una proximidad entre el enfoque mencionado en el párrafo previo y el propuesto en esta tesis, en tanto los *Casos de uso* y los escenarios presentan características similares y ambos son una representación del comportamiento.

1.4 Organización de la tesis

Esta tesis está organizada de la siguiente manera:

En el Capítulo 2 se introducen los conceptos básicos del enfoque Léxico Extendido del Lenguaje y Escenarios y ciertos elementos de Ingeniería del Software e Ingeniería de Requerimientos.

En el Capítulo 3 se describen conceptos generales acerca de la medición del software y los conceptos propuestos por los métodos de Análisis de Puntos Función.

En el Capítulo 4 se enuncian los beneficios de la aplicación del enfoque propuesto. Se analizan las alternativas para el cálculo de los Puntos Función desde L&E y se describe el modelo de medición del tamaño funcional en el marco de L&E. Por último se analizan los métodos de Albrecht y MKII para determinar cuál se adapta mejor al contexto de L&E.

En el Capítulo 5 se describe la propuesta para medir la funcionalidad desde L&E tomando como unidad los episodios. Se presenta una clasificación de los episodios y un análisis comparativo de conceptos del L&E y del método de FPA que permiten definir un conjunto de reglas. Por último se define el límite del sistema.

En el Capítulo 6 se presenta el proceso de medición de funcionalidad de L&E. Se incluye además la descripción de los formularios que sirven de soporte para documentar el proceso de medición.

En el Capítulo 7 se expone la aplicación del enfoque propuesto a casos de estudio. Se presenta en detalle completo la aplicación al caso de estudio Recepción del Hotel, un resumen de la aplicación a otros casos y un cuadro con los resultados obtenidos.

En el Capítulo 8 se obtienen conclusiones acerca de la aplicabilidad del FPA a L&E.

En el Anexo 1 se adjunta el modelo de LEL y Escenarios para el caso de estudio Recepción del Hotel.

En el Anexo 2 se incorpora una síntesis del modelo de reglas utilizado para la descripción de la propuesta.

Capítulo 2

Léxico Extendido del Lenguaje y Escenarios

Este capítulo está organizado de la siguiente manera. En la sección 2.1 se presenta una breve reseña acerca de la Ingeniería del Software. En la sección 2.2 se describe la Ingeniería de Requerimientos y sus procesos. En la sección 2.3 se incluye la descripción de los conceptos básicos de LEL y Escenarios y en la sección 2.4 se expone una síntesis del uso de herramientas para la construcción del L&E.

2.1. La Ingeniería del Software

Una de las primeras definiciones de la Ingeniería del Software fue propuesta por Bauer en 1969, según se menciona en [Pressman'93]:

"El establecimiento y uso de principios de ingeniería robustos, orientados a obtener software económico que sea fiable y funcione de manera eficiente sobre máquinas reales".

Se han propuesto más definiciones, pero en todas se refuerza la importancia de utilizar un enfoque ingenieril para el desarrollo del software.

A lo largo de las dos últimas décadas la Ingeniería del Software ha crecido hasta convertirse en una verdadera disciplina, derivada de una investigación seria, un estudio minucioso y un debate interdisciplinario. En la industria, el término "ingeniero del software" ha reemplazado al de "programador" como categoría de trabajo principal. Actualmente el software ha superado al hardware como la clave del éxito de muchos sistemas basados en computadoras. El software es el que marca la diferencia [Pressman'93]. Como consecuencia de ese crecimiento de la industria del software, en algunos casos desordenado, que se caracterizó por los altos costos de mantenimiento, se llegó a la conocida "crisis del software", que provocó una revisión de las prácticas realizadas hasta entonces y el nacimiento de una nueva etapa que se caracteriza por una búsqueda incesante de calidad en los productos desarrollados. Esto llevó al surgimiento de múltiples estrategias para mitigar los errores cometidos en las etapas previas.

Los sistemas de información se han convertido en una parte integral de nuestra vida diaria, al punto que el bienestar de los individuos, la competencia en el mundo de los negocios y la efectividad de las instituciones públicas muchas veces dependen de la correctitud y eficiencia con que funcionan estos sistemas [Loucopoulos'95].

Existe por lo tanto una tarea del proveedor del sistema de entregar una solución que reúna el nivel de funcionalidad esperado y asegure una integración exitosa del sistema técnico en un marco organizacional [Loucopoulos'95].

La Ingeniería del Software abarca un conjunto de tres elementos clave –*métodos*, *herramientas* y *procedimientos* – que permiten controlar el proceso de desarrollo de software y proveer las bases para construir software de calidad de manera eficiente.

De manera sintética se describen cada uno de esos elementos [Pressman'93] [Loucopoulos'95].

Los *métodos* indican "cómo" construir el software. Abarcan un conjunto de tareas que incluyen: planificación y estimación de proyectos, análisis de requerimientos, diseño de las estructuras de datos, arquitectura de programas y procedimientos algorítmicos, codificación, prueba y mantenimiento [Pressman'93].

Las *herramientas* suministran un soporte automático o semiautomático para los *métodos*. Es el caso de las herramientas CASE (Ingeniería del Software asistida por Computadoras) [Pressman'93].

Los *procedimientos* son el nexo entre los *métodos* y las *herramientas*. Éstos definen la secuencia en la que se aplican los *métodos*, las entregas requeridas, los controles que favorecen el control de calidad, la gestión de cambios y las directivas para la evaluación del proyecto [Pressman'93].

Existen muchos modelos diferentes de desarrollo de software, sin embargo la mayoría de ellos reconoce la existencia de los componentes que se muestran en la Figura 1.

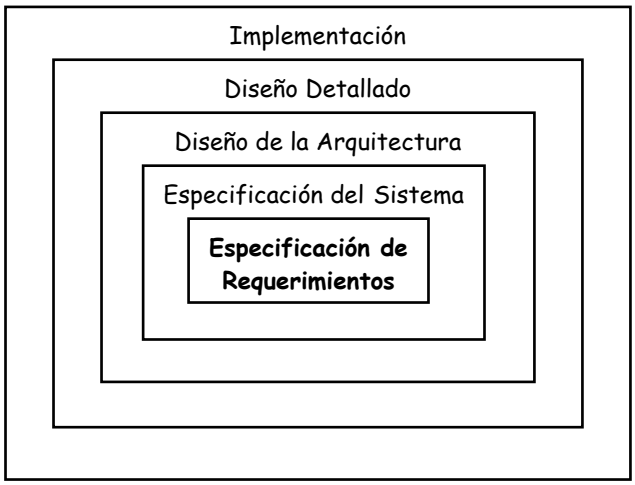


Figura 1 – Componentes del desarrollo de software [Loucopoulos'95]

Hace mucho tiempo que se ha establecido que la efectividad y flexibilidad de un sistema están intrínsecamente relacionadas con la interpretación correcta de las necesidades de los clientes o usuarios del sistema. Hay un componente clave, por lo tanto, de cada proceso de desarrollo, que juega un rol central en este proceso, llamado la **Especificación de Requerimientos**. El proceso de desarrollar una Especificación de Requerimientos de Software (SRS) se ha denominado **Ingeniería de Requerimientos** [Loucopoulos'95].

En este punto cabe preguntarse: ¿cuál es el propósito de la SRS? Hay varias razones que justifican el esfuerzo para el desarrollo de la SRS: provee un punto de partida para el proceso de comunicación y comprensión del dominio, el negocio y el sistema deseado, puede formar parte de acuerdos contractuales y se puede usar para la evaluación del producto final y jugar un rol relevante en la prueba de aceptación acordada entre el usuario y el proveedor del sistema [Loucopoulos'95].

La visión tradicional de la SRS es la de una especificación funcional, es decir, la descripción de las funciones fundamentales de los componentes del sistema de software. El interés está centrado en el procesamiento que deberán realizar los componentes del sistema sobre las entradas para producir alguna salida. Las funciones se especifican en términos de entradas, procesamiento y salidas. Sin embargo, una visión más dinámica de la funcionalidad del sistema también debe considerar aspectos referidos al control, temporalidad de las funciones y comportamiento del sistema en situaciones de excepción. Además, como las funciones tratan con datos, cualquiera sea la naturaleza de ellos (de entrada, salida, almacenados o temporales), esos datos requieren estar definidos y formar parte de la especificación funcional [Loucopoulos'95].

Tradicionalmente predominó la tendencia a darle relevancia a las especificaciones funcionales en desmedro de otros aspectos importantes, tales como los objetivos del sistema en sí mismo y la relación de éstos con los objetivos de la organización, la especificación de otras propiedades deseables del sistema (performance[2], seguridad, usabilidad, etc.) y las restricciones en su desarrollo (uso de herramientas, restricciones económicas, etc.). Por mucho tiempo se aceptó que la SRS debe definir el "qué", es decir, la descripción del problema y no el "cómo", es decir, la forma en que será resuelto el problema. Hay muchos factores que dificultan esa distinción entre el "qué" y el "cómo". En la práctica hay evidencias de proyectos en los que se requiere cierta comprensión de la arquitectura del sistema para poder articular, representar y evaluar los requerimientos [Loucopoulos'95].

Una visión más amplia de la SRS es la que va más allá de la descripción de las funcionalidades del sistema. Una SRS debe complementarse con otras perspectivas: debe incorporar la comprensión del dominio dentro del cual estará inserto el sistema y las restricciones que pueden determinar el sistema, su entorno o su desarrollo; esto se conoce como requerimientos no-funcionales (NFR - seguridad, portabilidad, usabilidad, performance, disponibilidad, etc.) [Loucopoulos'95].

El término SRS puede usarse desde una perspectiva aún más amplia para referirse a la descripción de los requerimientos en la empresa, expresada en términos de los fenómenos que son comunes a la empresa y al dominio del sistema [Loucopoulos'95].

2.2. La Ingeniería de Requerimientos

Los sistemas y requerimientos del software inadecuados, incompletos, erróneos y ambiguos son una fuente de problemas importante y continua en el desarrollo de sistemas. Esos problemas se manifiestan en agendas y presupuestos excedidos y sistemas que, en grado variable, son insensibles a las verdaderas necesidades del cliente. Esas dificultades son atribuidas a los procesos, pobremente definidos y mal comprendidos, usados para elicitar, especificar, analizar y validar requerimientos [SEI'91].

[2] Esta palabra se puede traducir como "rendimiento", "desempeño". Se consideró adecuado utilizar la palabra en inglés debido al uso habitual de la misma.

Es interesante en este punto definir el concepto de *requerimiento*.

Una definición de los requerimientos: (IEEE-Std. '610' -1990-):

1. *Una condición o capacidad requerida por el usuario para resolver un problema o alcanzar un objetivo.*
2. *Una condición o capacidad que debe ser alcanzada o procesada por un sistema o componente del sistema para satisfacer un contrato, un estándar, una especificación u otros documentos formalmente impuestos.*
3. *Una representación documentada de una condición o capacidad como en 1. o 2* [Loucopoulos'95].

La *Ingeniería de Requerimientos* es un subcampo de la *Ingeniería del Software* que surgió como una necesidad a partir de la "crisis del software".

"Ingeniería de Requerimientos (RE) se puede definir como:

El proceso sistemático de desarrollar los requerimientos a través de un proceso iterativo cooperativo de analizar el problema, documentar las observaciones resultantes en una variedad de formatos de representación y verificar la exactitud de la comprensión obtenida." [Loucopoulos'95].

Esta definición permite visualizar que la SRS implica una interacción de varios factores que incluyen aspectos relacionados con la forma de representación, la comunicación y cooperación entre los diferentes actores que participan del proceso y la comprensión del proceso por parte de los mismos.

Existen muchas formas de especificar requerimientos, sin embargo deben considerarse algunas características que debe satisfacer una SRS: correcta, completa, no ambigua, modificable, comprensible, rastreable, consistente, verificable e independiente del diseño [Loucopoulos'95].

La realización de un proceso exitoso para la obtención de la SRS dependerá de la habilidad del desarrollador para capturar las necesidades y restricciones del entorno y reflejarlas adecuadamente en la documentación, mediante un proceso de retroalimentación que permita la validación.

La calidad de la SRS y en última instancia del sistema de información depende mucho de la habilidad del desarrollador para extraer y comprender el conocimiento acerca del dominio modelado y del sistema de información en sí mismo [Loucopoulos'95]. La producción de una descripción del comportamiento del sistema que sea completa, sin inconsistencias ni ambigüedades requiere establecer el contexto: lugar en el que se desarrollarán las tareas de ingeniería, recursos disponibles, objetivos del producto y sus límites. Este contexto se denomina *Universo del Discurso* (UD) y no debe establecerse una visión estática del mismo: evoluciona a lo largo del ciclo de vida del producto. La mayor precisión que se pueda introducir en la definición del UD incrementará las posibilidades de alcanzar un sistema bien definido [Hadad'96].

Para lograr sus objetivos, la Ingeniería de Requerimientos describe tres subprocesos denominados *elicitación*, *especificación* y *validación de requerimientos* [Loucopoulos'95]. En la Figura 2 se muestra un esquema del proceso.

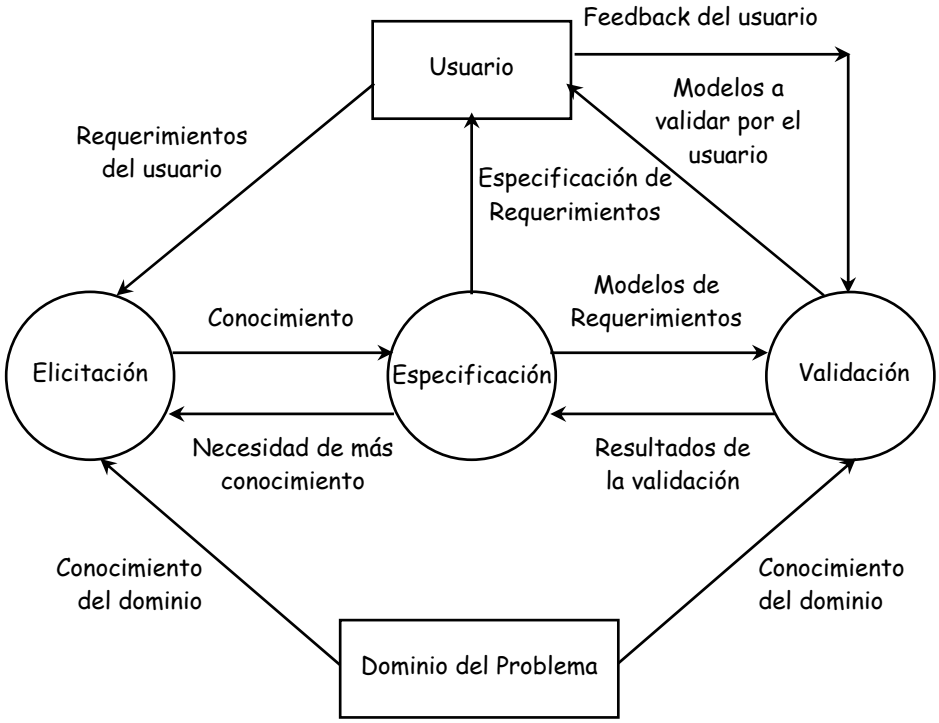


Figura 2 - Procesos de la Ingeniería de Requerimientos [Loucopoulos'95]

Elicitación

Se va a dedicar especial atención a este subproceso, pues en este trabajo se ha desarrollado el proceso de elicitación para un caso de estudio real, con el objetivo futuro de utilizarlo para la aplicación de un enfoque de medición de la funcionalidad.

Definiciones del término *elicitar:*

Diccionario Oxford Advanced Learner's [Hornby'95]
"Elicit: to draw facts, a response, etc. from somebody, sometimes with difficulty."

Loucopoulos [Loucopoulos'95]
"La elicitación de requerimientos se define como el proceso de la adquisición de todo el conocimiento relevante necesario para producir un modelo de requerimientos del dominio de un problema."

CREWS Glossary [CREWS'99]
"Elicit: the process of systematically obtaining from people new facts (scenarios, requirements) about the domain / business processes / the system under consideration".

Sinónimos: Acquire, Discover, Capture

El proceso de elicitación debe obtener la información acerca del sistema en el UD.

Existen muchas técnicas para realizar el proceso de elicitación. Cada una de las técnicas tiene ventajas y desventajas y es más adecuada a algunos tipos de problemas

que otros. Una de las técnicas más comúnmente usada es la adquisición de los requerimientos a partir de entrevistas a los usuarios y el reuso del conocimiento adquirido en dominios de problemas similares. Otra técnica es la de Léxico Extendido del Lenguaje (LEL) y Escenarios, sobre la que se tratará en forma detallada en la sección 2.3.

Especificación

La especificación se puede visualizar como un contrato entre los usuarios y los desarrolladores, el cual define el comportamiento funcional deseado del artefacto de software (y otras propiedades del mismo tales como la performance, fiabilidad, etc.), sin mostrar cómo se alcanzará dicha funcionalidad. El proceso de especificación de requerimientos construye representaciones de los requerimientos del software a partir del conocimiento obtenido en el proceso de elicitación para ser usadas en etapas subsiguientes del desarrollo [Loucopoulos'95].

Validación

La validación de requerimientos es el proceso de certificación de la exactitud del modelo de requerimientos frente al propósito del usuario. Las estadísticas han probado que no es suficiente validar el software después que ha sido desarrollado. Cuanto más tiempo se pospone la validación del software, más caro es corregir los errores (en términos de prueba, corrección y redesarrollo). Por lo tanto, la fase de validación de la especificación de requerimientos puede ayudar a evitar la corrección de errores costosos en etapas avanzadas del desarrollo. El propósito de esta tarea es identificar la existencia de una serie de propiedades deseables en el modelo de requerimientos, tales como: consistencia interna, no-ambigüedad, consistencia externa, minimalidad, completitud, redundancia. En este proceso el desarrollador y el usuario alcanzan un acuerdo acerca que el modelo de los requerimientos especifica una solución de software satisfactoria [Loucopoulos'95].

2.3. LEL y Escenarios

Una de las propuestas planteadas por la comunidad de Ingeniería del Software con el objetivo de dar respuesta a las necesidades surgidas ha sido el uso del lenguaje natural como medio de comunicación y representación, puesto que el mismo facilita enormemente la comprensión entre los distintos participantes del proceso; estos generalmente no tienen un vocabulario común, lo que dificulta el proceso de trasmisión de información en ambos sentidos.

La investigación se concentra en la utilización del LEL y los Escenarios para la elicitación de los requerimientos y su uso a lo largo del ciclo completo de desarrollo de software [Leite'97].

Este nuevo enfoque presenta una metodología basada en el uso de LEL para registrar el vocabulario del macrosistema y escenarios para modelar el comportamiento. Ambas herramientas presentan la ventaja de utilizar el lenguaje

natural para las descripciones, lo que favorece la comunicación y validación con el usuario [Hadad'96].

2.3.1. LEL

Una cuestión fundamental en el desarrollo de software es la comunicación. Usuarios y desarrolladores usualmente utilizan un lenguaje distinto que dificulta ese proceso. Es importante encontrar alternativas que permitan el uso de un mismo lenguaje que facilite la comunicación, comprensión, rastreo, revisión y validación de los productos resultantes en las diferentes etapas del desarrollo. Naturalmente el medio más simple de resolver ese problema surge a partir de la utilización del lenguaje natural y un enfoque orientado en esa dirección es el modelo propuesto por el LEL.

El LEL tiene como finalidad la comprensión del vocabulario de la aplicación, sin considerar en esta etapa la comprensión del problema. Se propone construir el LEL como primer paso en la fase de elicitación de requerimientos, tratando de conocer el vocabulario que utiliza el usuario en su mundo real y despreocuparse durante este procedimiento de entender el problema. Una vez que se está familiarizado con ese léxico, la comunicación con el usuario y la comprensión del problema tendrá un obstáculo menos: el vocabulario [Hadad'96].

El LEL es una representación en forma de hipertexto de los símbolos del lenguaje del cliente en el contexto de la aplicación. Cada símbolo podrá ser una palabra o frase, frecuentemente las más repetidas por el cliente o de importancia relevante para el sistema. A partir de las experiencias realizadas en el uso de esta técnica, se ha concluido que pueden no incluirse aquellas palabras o frases que son de conocimiento general, las que son demasiado específicas y las que tienen un significado muy amplio [Hadad'97]. La Figura 3 muestra gráficamente el proceso de construcción del LEL.

Para adquirir conocimiento del vocabulario, el ingeniero de software planifica alguna estrategia para recolectar información, principalmente entrevistas con el usuario o en caso de imposibilidad de hacerlas, se trata de reemplazar con la lectura de documentos descriptivos de la aplicación. Durante esta etapa se registran las frases o palabras que parecen tener un significado especial en la aplicación. Como resultado de esta fase se obtiene una *lista candidata* formada por *símbolos o entradas candidatos*, que el ingeniero de software utilizará como base para realizar nuevas entrevistas con el cliente, tratando de comprender el significado de cada símbolo.

Los símbolos se clasifican en las categorías generales *Sujeto*, *Verbo*, *Objeto* y *Estado* y se las adapta al dominio del problema [Hadad'97].

A partir del conocimiento obtenido, cada símbolo se describe en términos de *nombre*, *noción* e *impacto*. El *nombre* identifica el símbolo, la *noción* denota lo que significa el símbolo y el *impacto* cómo repercute en el sistema. Al describir *noción* e *impacto* deben tenerse en cuenta dos principios: *circularidad* que implica maximizar el uso de los símbolos en el significado de otros símbolos y *vocabulario mínimo* que

significa minimizar el uso de símbolos externos al lenguaje de la aplicación. La imposición de ambos principios da como resultado la formación de una red de elementos vinculados entre sí, que se representa como un documento de hipertexto [Leite'97].

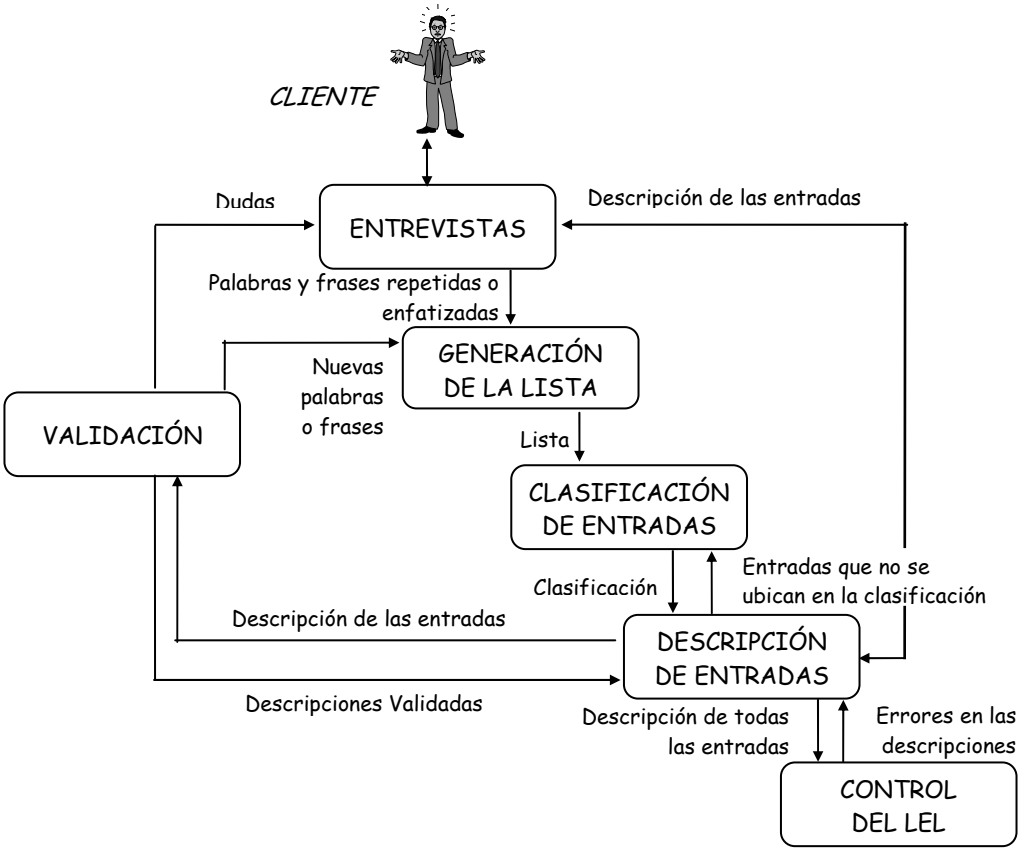


Figura 3 - Etapas para la construcción del LEL [Hadad'97]

La *lista candidata* se somete a validación con el cliente a fin de rectificar o ratificar su descripción. Esta etapa dará lugar a modificaciones en los símbolos, incorporación de nuevos símbolos y eliminación de otros. Se trata éste de un proceso iterativo de interacción con el cliente que va refinando la lista hasta que se obtiene la lista definitiva. Forma parte de esta etapa de validación el desarrollo de *escenarios*.

Se presenta un ejemplo de símbolo del LEL[3]. (los textos subrayados representan símbolos del LEL).

no show

- Noción
  - Es la baja de la *solicitud de reserva* por no presentación del *pasajero / huésped / pax*.
  - Lo hace el *recepcionista*.
  - Se hace en la *Recepción*.
- Impacto
  - Si el *pasajero / huésped / pax* no se presenta el *día de ingreso* hasta las 06 hs. del día siguiente entonces se elimina la *solicitud de reserva* en la *planilla de reservas*.

[3] Los ejemplos corresponden al caso de estudio Recepción del hotel. Ver Anexo 1.

- Se actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*

El documento hipertextual obtenido como resultado del desarrollo del LEL se puede navegar a través de los diferentes vínculos para conocer todo el vocabulario registrado.

2.3.2. Escenarios

Los escenarios juegan un rol importante en el proceso de desarrollo de software. La especificación convencional y los lenguajes de implementación no están diseñados para soportar la descripción de sistemas a través de escenarios. Las especificaciones describen los requerimientos que el sistema debe satisfacer en todos los casos. Igualmente el código es un conjunto de instrucciones suficientes para manejar todas las situaciones posibles que el sistema encontrará. Este énfasis sobre las descripciones generales en la Ingeniería del Software está en severo contraste con las habilidades humanas. La ciencia cognitiva ha demostrado que el conocimiento humano está organizado de manera que favorece el reconocimiento y respuesta a situaciones específicas. Esta cierta incompatibilidad entre la Ingeniería del Software y las capacidades humanas causa severas dificultades en la elicitación de requerimientos y validación de sistemas. En ese marco los escenarios han surgido como una manera alternativa de describir secuencias de interacciones entre sistemas y usuarios [Benner'93].

El escenario es una técnica de descripción que ha ganado la atención de profesionales e investigadores. La comunidad de sistemas de información ha dedicado especial atención a las posibilidades que brinda esta técnica para mejorar la comprensión de los aspectos relacionados al entorno de un sistema requerido y facilitar la comunicación entre los participantes, favoreciendo el acuerdo entre los requerimientos del usuario y el soporte provisto por el sistema [Sutcliffe'97].

El uso de ejemplos, escenas, descripciones narrativas de contextos, maquetas y prototipos han atraído considerable atención en las comunidades de la Ingeniería de Requerimientos, Interacción Humano Computadora (HCI) y Sistemas de Información. Aproximadamente todas esas ideas pueden llamarse *enfoques basados en escenarios*, aunque las definiciones precisas no son fáciles, más allá de decir que esos enfoques ponen énfasis en descripciones del mundo real. La experiencia parece indicar que la gente reacciona ante situaciones reales y que esto ayuda a clarificar los requerimientos. La amplia aceptación del prototipado en el desarrollo de sistemas apunta a la efectividad de los enfoques basados en escenarios. Lo que se representa en los escenarios varía desde descripciones concretas de la realidad hasta sistemas diseñados que se ejecutan como simulaciones para presentar una visión de cómo se comportará el sistema [Sutcliffe'97].

Los escenarios pueden representar el comportamiento de sistemas existentes, pero más comúnmente son útiles durante la planificación y diseño de nuevos sistemas. Los escenarios se pueden usar para facilitar la comprensión de las prácticas de trabajo y procesos del negocio, para representar requerimientos de comportamiento,

para validar requerimientos, para evaluar alternativas de requerimientos para componentes arquitecturales, para generar y validar diseños orientados a objetos, para explorar la significación de las decisiones de diseño y para generar y revisar casos de prueba del sistema [Potts'98].

Se dice que los escenarios son descripciones parciales porque no necesitan especificar completamente todos los estados que comprende un comportamiento ni especificar todos los atributos de un estado dado. Se trata de permitir que el analista provea sólo aquellas partes que son relevantes para el escenario [Benner'93].

Respecto al nivel de abstracción en que se describe un escenario, es crucial que los escenarios se puedan expresar en términos concretos. Esta capacidad permite al analista describir un comportamiento en el mayor nivel de generalidad, sin detenerse en detalles, pero considerando los factores importantes que tienen influencia en situaciones realistas. Esto refleja nuevamente el rol que juegan los escenarios para determinar qué es lo importante y por lo tanto, qué debería incluirse o no en el escenario [Benner'93].

En el contexto del presente trabajo los escenarios se usarán como un medio para favorecer el proceso de elicitación de requerimientos y se considerará la definición de Leite:

Los escenarios representan descripciones parciales del comportamiento del sistema en un momento específico de la aplicación, se representan en lenguaje natural, están vinculados naturalmente al LEL, evolucionan durante el proceso de desarrollo del software y se representan de manera estructurada [Leite'97].

Un escenario se compone de: *nombre*, *objetivo*, *contexto*, *recursos*, *actores* y *episodios*. A continuación se reproduce el Modelo de escenarios propuesto en [Leite'00]; en cada caso se explica el componente, su sintaxis (usando sintaxis BNF) y se provee un ejemplo.

Nombre: es el título del escenario. En el caso de un sub-escenario es el mismo que el de la sentencia episodio, sin las excepciones y/o restricciones.

Sintaxis:

Phrase | ([Actor | Resource] + Verb + Predicate)

Ejemplo:

solicitud de reserva

Objetivo: es la finalidad a alcanzar en el macrosistema.

Sintaxis:

[Actor | Resource] + Verb + Predicate

Ejemplo:

– Atender una *solicitud de reserva* realizada por una persona, *agencia* u otro hotel.

Contexto: describe la ubicación geográfica/temporal y/o estado inicial del escenario.

Sintaxis:

{Geographical Location} + {Temporal Location} + {Precondition}
where Geographical Location is:
Phrase + {Constraint}
where Temporal Location is:
Phrase + {Constraint}
where Precondition is:
[Subject | Actor | Resource] + Verb + Predicate + {Constraint}

Ejemplo:

- Se realiza en la *Recepción* del Hotel

Recursos: medios de soporte, dispositivos u otros elementos pasivos necesarios para el escenario.

Sintaxis:

Name + {Constraint}

Ejemplo:

- *lista de precios*
- *planilla de reservas*
- *planilla de ocupación de habitaciones*
- Teléfono
- Fax
- e-mail

Actores: personas o estructuras organizacionales que cumplen un rol en el escenario.

Sintaxis:

Name

Ejemplo:

- *recepcionista*
- *agencia*
- otro hotel
- *pasajero / huésped / pax*

Episodios: una serie de acciones que detallan al escenario y describen su comportamiento.

Sintaxis:

<episodes> ::= <group series> | <episode series>
<group series> ::= <group> <group> | <non-sequential group> | <group series> <group>
<group> ::= <sequential group> | < non-sequential group>
<sequential group> ::= <basic sentence> | <sequential group> <basic sentence>
<non-sequential group> ::= # <episode series> #
<episode series> ::= <basic sentence> <basic sentence> | <episode series> <basic sentence>
<basic sentence> ::= <simple sentence> | <conditional sentence> | <optional sentence>

<simple sentence> ::= <episode sentence> CR
<conditional sentence> ::= **IF** <condition> **THEN** <episode sentence> CR
<optional sentence> ::= **[** <episode sentence> **]** CR
where <episode sentence> is described:
(([Actor | Resource] + Verb + Predicate) | ([Actor | Resource] + [Verb] + Title)) + {Constraint}

Ejemplo:

- El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo y modalidad de la *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

Excepciones: usualmente reflejan la falta o mal funcionamiento de un recurso necesario. Una excepción impide alcanzar el objetivo de un escenario. El tratamiento de la excepción puede ser expresado a través de otro escenario.

Sintaxis:

Cause [**(**Solution**)**]
where Cause is:
Phrase | ([Subject | Actor | Resource] + Verb + Predicate)
where Solution is:
Title

Ejemplo:

- **excepción:** Faltan tarjetas magnéticas para codificar

Restricciones: un requerimiento de alcance o calidad referido a una entidad dada. Es un atributo de los recursos, episodios básicos o subcomponentes del contexto.

Sintaxis:

([Subject | Actor | Resource] + **Must** [**Not**] + Verb + Predicate) | Phrase

Ejemplo:

- **restricción:** El *voucher* no cumple con los requisitos.

En la Figura 4 se presenta el modelo ER para el Modelo de Escenarios.

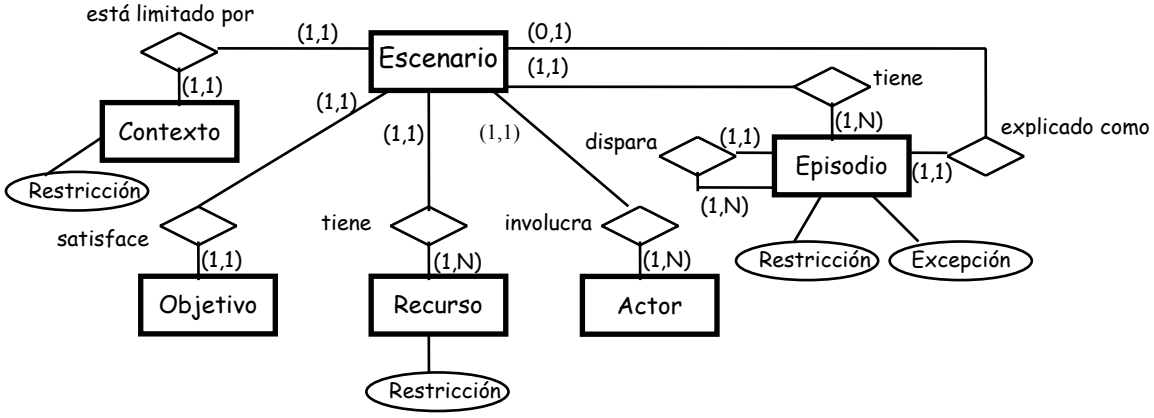


Figura 4 - Diagrama ER para el Modelo de Escenarios [Hadad'97]

Para la construcción de los escenarios, se usa exclusivamente la lista de símbolos del LEL. El proceso consta de varias etapas [Hadad'97]:

1. ***Identificación de los actores de la aplicación***: en el LEL se detectan las entradas que representan a los **actores** del UD. Se identifican los **actores principales** (los que realizan acciones directas sobre la aplicación) y los **actores secundarios** (los que reciben y/o brindan información, pero no ejecutan acciones directas sobre la aplicación).
2. ***Generación de la lista de escenarios candidatos***: los **impactos** de los símbolos del LEL de cada actor principal representan un escenario potencial, de esta lista se eliminan los escenarios repetidos.
3. ***Descripción de los escenarios candidatos***: a partir de la lista de escenarios candidatos, se describe cada escenario basándose en las siguientes reglas:
   - *Contiene un símbolo del LEL que pertenece a la clasificación **Verbo***
     - ⇒ Se busca dicho símbolo en el LEL.
     - ⇒ Se define el **objetivo** del escenario basándose en su nombre, el punto de vista de la aplicación y la **noción** de este símbolo, que representa una actividad.
     - ⇒ Se identifican los **actores** y los **recursos**, que pertenezcan a la clasificación ***Sujeto*** y ***Objeto*** respectivamente.
     - ⇒ A partir de cada **impacto** del símbolo se definen los **episodios**.
     - ⇒ Para definir el **contexto** se observa si el impacto del símbolo que lo originó tenía alguna secuencia con respecto a otros impactos, en caso afirmativo estos se incorporan como precondición. En el contexto también se define la ubicación física o temporal para la realización del escenario.
     - ⇒ Se completa el ítem **dudas**.

   A medida que se describen los episodios pueden surgir las **excepciones**.

   - *No contiene un símbolo del LEL que pertenezca a la clasificación **Verbo***
     - ⇒ Se identifican **símbolos** dentro del **impacto**.
     - ⇒ Se extrae del LEL la información sobre esos símbolos
     - ⇒ Se define el **objetivo** en función del nombre del escenario y el punto de vista de la aplicación.
     - ⇒ Se identifican posibles **actores** y **recursos**, a través de los símbolos asociados, pertenecientes a la clasificación ***Sujeto*** y ***Objeto*** respectivamente.
     - ⇒ No se describen episodios partiendo del LEL.
     - ⇒ Se completa el ítem **dudas**.
4. ***Ampliación de la lista de escenarios candidatos***: se aplica el procedimiento descripto en el ítem 2 a los **actores secundarios**.
5. ***Descripción de los Escenarios candidatos***: se aplica el procedimiento del ítem 3 a los nuevos escenarios incorporados a la lista en el ítem 4.
6. ***Revisión de los escenarios***: se aplica el siguiente proceso de revisión:

⇒ **Detección** de escenarios candidatos como episodios simples dentro de otros escenarios, con el cliente se determina si involucran más de un episodio.
- En caso afirmativo, se lo incorpora a la lista y reemplaza a los episodios en los escenarios en que figuraban (detección de sub-escenarios).
- En caso contrario, se lo elimina de la lista.

⇒ **Detección** de escenarios candidatos como un conjunto de episodios dentro de otros escenarios. Si en escenarios provenientes de actores principales, un conjunto de episodios corresponde a un escenario proveniente de un actor secundario, se los reemplaza por este escenario (detección de sub-escenarios)

⇒ **Unificación** de escenarios. Dos o más escenarios con episodios comunes o el mismo objetivo y contexto se agrupan en un único escenario. Si se requiere, se utiliza la forma condicional para describir episodios diferentes.

⇒ Definición de **restricciones**. Provienen del contexto del problema y del contexto del escenario

⇒ **Detección** de escenarios que corresponden a **excepciones**. Si en un escenario su contexto no identifica los anteriores o no mantiene ninguna relación con otro, se verifica si puede o no corresponder a una excepción.

El producto de este proceso es la lista de escenarios a validar con los clientes.

7. ***Validación de escenarios***:

⇒ Se valida con los **clientes** la lista definitiva, especialmente el ítem dudas. Este proceso permite detectar errores, omisiones o ampliar información en los episodios.

⇒ Se realizan **correcciones**. En caso de ampliar información de un episodio, se puede:
- descomponer en varios episodios dentro del mismo escenario o
- reemplazar por un nuevo escenario que involucre el conjunto de acciones detectado y se lo incorpora a la lista.

⇒ Se realiza una **revisión** del contexto, los actores y los recursos de cada escenario, en función de las correcciones anteriores.

Como resultado de este proceso se obtiene la lista de escenarios definitiva.

LEL y Escenarios constituyen el producto obtenido como resultado del proceso de elicitación. Si bien ni el LEL ni los Escenarios son los requerimientos, disponer de ellos representa una gran ayuda para la futura descripción de los requerimientos de un sistema de software. Su uso ayudará a establecer no sólo los requerimientos funcionales sino también los requerimientos no funcionales [Hadad'96].

Mientras las restricciones reflejan requerimientos no funcionales, los episodios y los objetivos de los escenarios son una fuente para derivar los requerimientos funcionales [Hadad'96].

2.3.3. Ventajas del uso de escenarios en la Ingeniería de Requerimientos

Considerar el punto de vista del usuario. Un escenario siempre visualiza el sistema desde el punto de vista del usuario. Esta es una ventaja fundamental cuando se valida la satisfacción de los requerimientos. Los escenarios dan a los usuarios una sensación de lo que obtendrán, mientras que las técnicas clásicas como las listas narrativas de requerimientos, los diagramas Entidad-Relación o los diagramas de clases orientados a objetos no lo hacen [Glinz'00].

Especificaciones parciales. Los escenarios son un medio natural para escribir especificaciones parciales. Cada escenario captura una secuencia de interacciones usuario-sistema que representa una transacción o una función del sistema desde la perspectiva del usuario. La fortaleza particular de los escenarios se basa en el hecho que proveen una descomposición de un sistema en funciones desde la perspectiva del usuario y que cada una de esas funciones puede ser tratada separadamente [Glinz'00].

Fácil de entender. Los escenarios simplifican la elicitación y validación de requerimientos, porque la noción de interacción usuario-sistema es una forma natural de entender y discutir los requerimientos entre usuarios e ingenieros de requerimientos. Los escenarios ofrecen la comodidad y facilidad de las especificaciones en lenguaje natural y evitan muchos otros problemas de una especificación expresada en forma puramente narrativa [Glinz'00].

Ciclo de retroalimentación breve. La combinación de la habilidad de tratar con cada función del usuario separadamente y la forma de representar requerimientos orientados al usuario en los escenarios permite ciclos cortos de retroalimentación entre usuarios e ingenieros de requerimientos [Glinz'00].

El uso de escenarios cuando fracasa el modelo abstracto. El uso de escenarios para representar modelos abstractos complejos ha demostrado ser una alternativa exitosa para elicitar, documentar y validar los requerimientos del cliente [Weidenhaupt'98].

Los escenarios son un medio para alcanzar acuerdos y consistencia. Los usuarios afectados por el desarrollo del sistema tienen diferentes objetivos acerca del futuro sistema y sus visiones pueden variar en forma significativa. Lograr el acuerdo completo entre todos acerca del sistema, es una tarea que lleva tiempo y muchas veces es imposible. Igualmente asegurar que el sistema a construir es consistente con todos los sistemas existentes en la organización a veces es poco factible. Los escenarios sirven como un medio para plantear soluciones alternativas, basando la discusión y negociación en casos reales [Weidenhaupt'98].

Base para la prueba del sistema. Las secuencias de interacción capturadas en los escenarios son una base ideal para definir la prueba del sistema. Los casos de prueba se pueden derivar directamente desde los escenarios, mejorando la verificabilidad de los requerimientos [Glinz'00]. Para ello es necesario mantener actualizados los escenarios a medida que avanza el proyecto [Weidenhaupt'98].

Traceability[4]. La traceability es un prerrequisito para establecer el ciclo de vida de los escenarios definidos durante la fase de Ingeniería de Requerimientos. La falta de traceability es una de las causas de la desactualización de los escenarios y por lo tanto su inconsistencia, que deriva en la imposibilidad de generar casos de prueba del sistema. La traceability es la principal fuente para permitir la integración de los cambios y para mantener actualizados los escenarios. Mantener la traceability requiere una mejor comprensión de las relaciones entre los diferentes artefactos producidos durante el desarrollo del software y los escenarios [Weidenhaupt'98].

Modularidad. El uso de escenarios representa un aporte a la práctica habitual del conocido "divide y vencerás", que consiste en descomponer un problema en partes más pequeñas para facilitar su resolución. La representación de las funciones fundamentales de la aplicación en forma de escenarios será de ayuda significativa cualquiera sea el futuro modelo de desarrollo que se adopte.

2.3.4. Herramientas

Para la documentación del caso de estudio en la primera etapa se utilizó un procesador de textos común. Dada la característica hipertextual del documento, fue necesario establecer los vínculos entre los diferentes símbolos del LEL y cada una de sus apariciones en las descripciones de otros símbolos y en los escenarios, para permitir la materialización de la red navegacional. Es de destacar lo tediosa que resultó la tarea, pues por cada cambio que se fue planteando a medida que evolucionaba el trabajo, fue necesario modificar manualmente cada uno de los vínculos.

Una herramienta desarrollada a efectos de facilitar la tarea es Baseline Mentor. Esta aplicación permite trabajar con proyectos. Cada proyecto contiene varias versiones. Las versiones están compuestas de LEL, Escenarios y tarjetas CRC. Baseline Mentor permite editar y navegar a través de esta información, haciendo verificaciones de integridad y consistencia, derivando automáticamente las tarjetas CRC desde el LEL y Escenarios y exportando información en formato HTML y RTF [Antonelli'99].

En las siguientes figuras se muestran diferentes pantallas de Baseline Mentor.

En la Figura 5 se visualiza la barra de menús de la ventana principal.

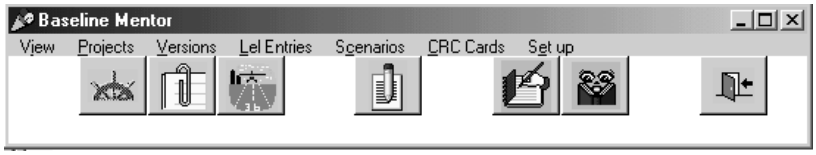


Figura 5 - Ventana principal BMW

[4] La palabra "traceability" se puede traducir como "rastreabilidad", habilidad para "rastrear". Debido a que su uso es poco habitual, se utiliza la palabra en inglés.

En la Figura 6 se visualiza la vista Full Browser para el archivo correspondiente al caso de estudio.

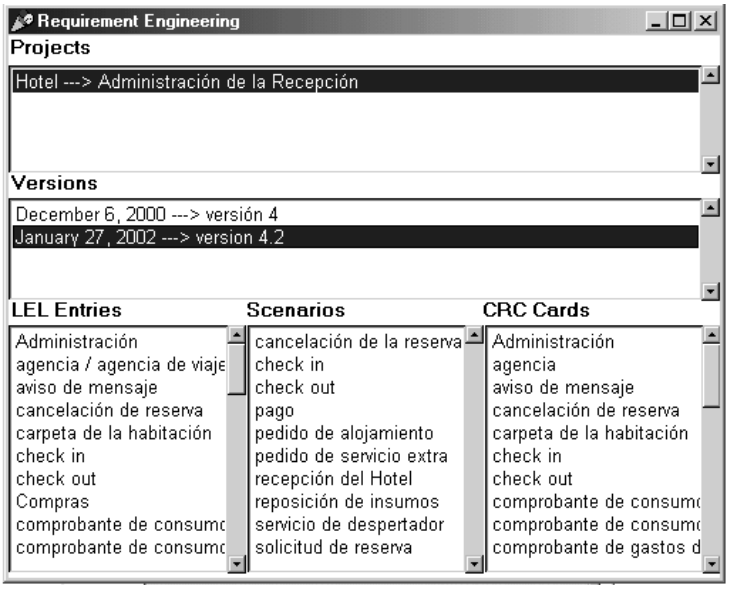


Figura 6 - Ventana de la vista Full Browser

En la Figura 7 se visualiza la ventana de edición de una de las entradas del LEL.

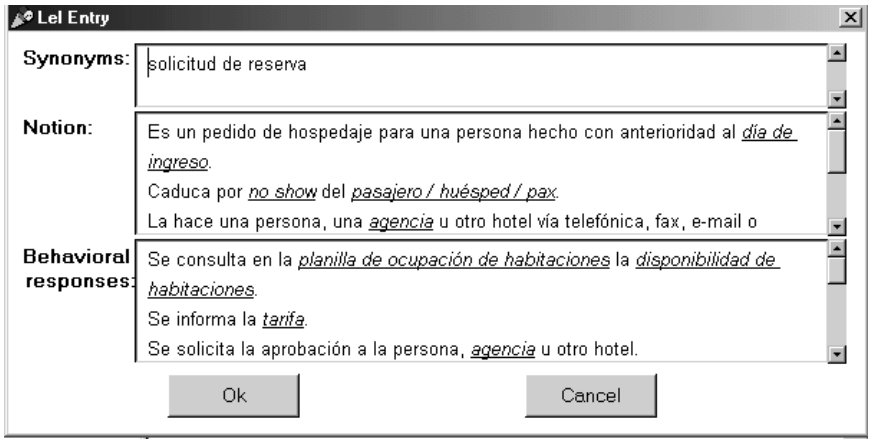


Figura 7 - Ventana de edición de una entrada del LEL

En la Figura 8 se visualiza la ventana de edición de uno de los escenarios.

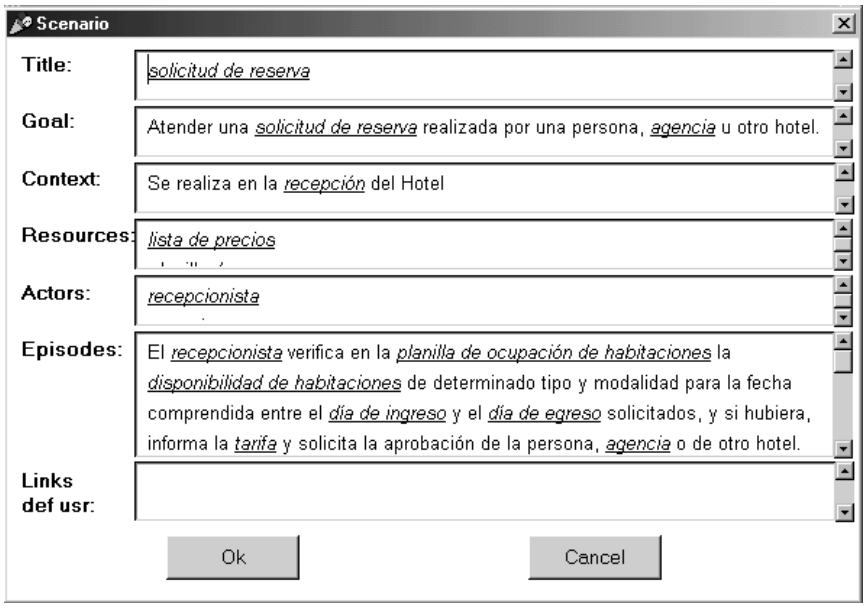


Figura 8 - Ventana de edición de un escenario

En la Figura 9 se visualiza la vista de hipertexto de una entrada del LEL.

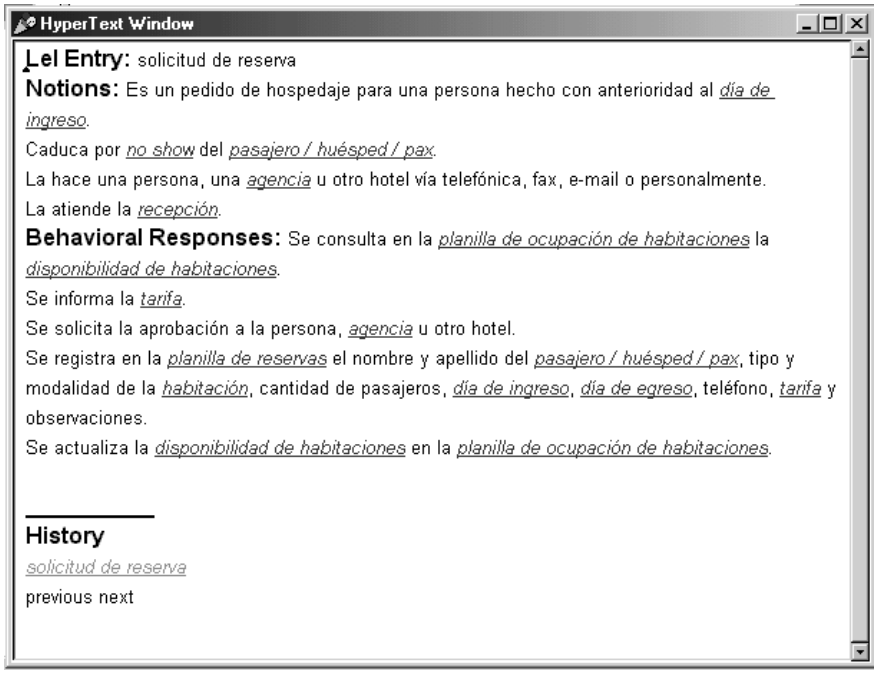


Figura 9 - Vista de hipertexto para una entrada del LEL

En la Figura 10 se visualiza la vista de hipertexto para uno de los escenarios.

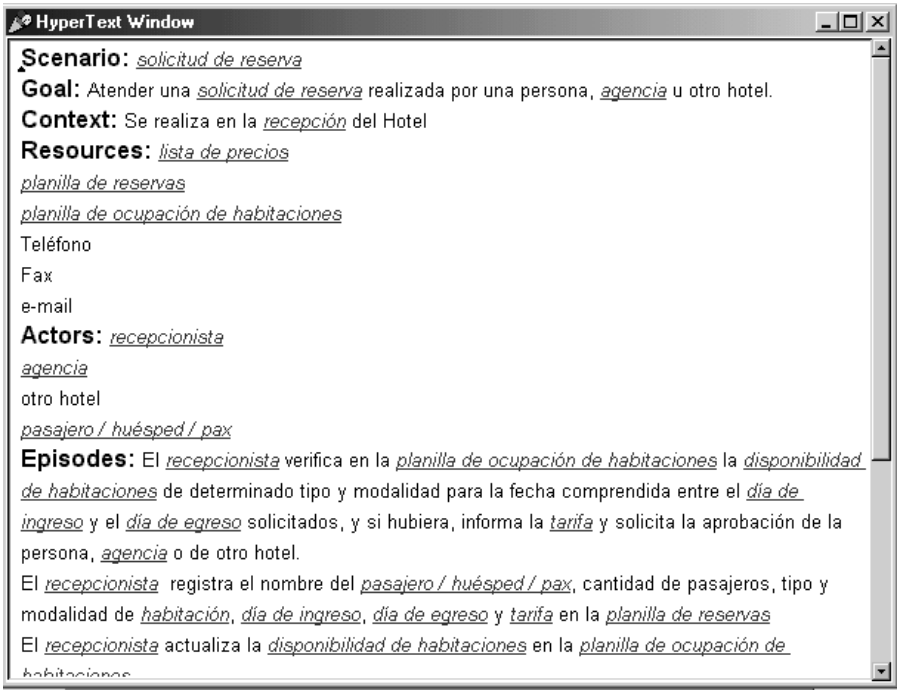


Figura 10 - Vista de hipertexto para un escenario

Capítulo 3

Métricas del software

Este capítulo está organizado de la siguiente manera. En la sección 3.1 se introducen algunos conceptos acerca de la medición. En la sección 3.2 se describen las métricas del software y sus beneficios. En las secciones 3.3 y 3.4 se presentan la clasificación de las medidas del software y los diferentes enfoques de la medición. En la sección 3.5 se analizan las desventajas de las líneas de código y en la sección 3.6 se comparan con los FP. En la sección 3.7 se describen en detalle los métodos de FPA.

3.1. Las medidas: qué son y por qué hacerlas

Nuestra vida cotidiana está permanentemente rodeada de situaciones que dependen de mediciones: determinar precio, tamaño y cantidad de objetos, establecer los parámetros que requiere un avión para sus diferentes operaciones en despegue, vuelo y aterrizaje, decidir según ciertos valores el estado de salud o enfermedad de una persona o ajustar instrumental para realizar prácticas de distintos tipos en cualquier actividad científica. Éstos y muchísimos más son ejemplos de cómo inciden las medidas en nuestras vidas.

¿Por qué medir? [IFPUG'00ª]

La medición es una práctica de administración probada en el tiempo, porque:

- No se puede administrar lo que no se puede medir.
- Aproximadamente el 40% de los proyectos fracasan debido a la falta de control en la administración.
- La medición también brinda una herramienta para comunicar a los clientes el tamaño de su petición y extrapolar productividad, calidad y estimación.
- Se mide para entender y mejorar los procesos.

En este punto corresponde establecer el significado de las palabras medir, medida y medición. El *Diccionario Kapelusz de la lengua española* [Kapelusz'79] los define:

Medir: Comparar una magnitud con otra de su misma clase, tomada como unidad.

Medición: Acción y efecto de medir.

Medida: Acción y efecto de medir. Expresión numérica del resultado de una medición.

En [Fenton'96] se define formalmente de la siguiente manera:

Medición: proceso de asignar números o símbolos a los atributos de las entidades del mundo real de manera de describirlos de acuerdo a un conjunto de reglas claramente definidas.

Por lo tanto, una medición captura información de atributos de entidades, entendiendo por entidad un objeto o evento del mundo real. Para ello es necesario

distinguir los objetos según ciertas propiedades que los identifican. Esas propiedades o características de las entidades se conocen como atributos. En nuestra vida cotidiana hablamos indistintamente de medir entidades y atributos, pero, si bien es aceptable en el ámbito informal, esa forma de expresión es errónea y no válida para la ciencia. Estrictamente no se miden entidades, se miden atributos de entidades. Esos atributos se expresan mediante números o símbolos. Se representa el precio de un objeto como un número que significa la cantidad de pesos, la medida de la vestimenta mediante un símbolo (L, M, S) o un número que indica el talle o la dificultad para resolver un problema con un símbolo (fácil, medio, difícil). Esos números y símbolos son de gran importancia: nos permiten establecer comparaciones, obtener conclusiones y tomar decisiones acerca de los atributos de las entidades [Fenton'96].

En este campo se plantean una serie de interrogantes como: ¿Se pueden medir todos los atributos? ¿De qué depende la precisión de una medida? ¿Cuál es la escala adecuada? ¿Qué acciones se pueden aplicar a los resultados de la medición? ¿Cuál es el margen de error aceptable? [Fenton'96]

Para responder a esas preguntas, primero debe establecerse qué clases de cosas pueden medirse y aquí es interesante observar que en muchas ciencias en la actualidad se pueden medir atributos que en otros tiempos se creyeron no mensurables [Fenton'96].

En el campo de las mediciones existen dos clases de cuantificación: medición y cálculo. La medición implica la cuantificación directa, por ejemplo, medir la estatura de una persona. El cálculo es una forma de cuantificación indirecta, es decir, la medida se obtiene a partir de tomar medidas y combinarlas para obtener el valor de un atributo de interés, por ejemplo, medir la velocidad de movimiento de un objeto usando las medidas de distancia y tiempo. Esas dos formas de cuantificación se conocen como medidas *directas* e *indirectas* [Fenton'96].

3.2. Las métricas del software

La medición del software se ha convertido en parte esencial de la Ingeniería del Software. Muchos de los desarrolladores de software miden características del software para saber si los requerimientos son consistentes y completos, si el diseño es de alta calidad y si el código está listo para ser probado. Los administradores de proyectos miden atributos de procesos y productos para ser capaces de decir cuando el software estará en condiciones de ser entregado y si el presupuesto estará excedido [Fenton'96].

La Ingeniería del Software describe la colección de técnicas que aplican un enfoque ingenieril a la construcción y soporte de productos de software. Las actividades de la Ingeniería del Software incluyen la administración, cálculo de costos, planificación, modelado, análisis, especificación, diseño, implementación, prueba y mantenimiento. Con la expresión enfoque ingenieril, se quiere significar que cada actividad es comprendida y controlada. La importancia de la Ingeniería del Software no se puede subestimar, debido a que los productos de software invaden nuestras vidas [Fenton'96].

Las disciplinas de la ingeniería se basan en modelos y teorías, aplican métodos científicos sobre la base de hipótesis, diseñan y realizan experimentos para probar su verdad y analizan sus resultados. La medición representa el soporte para el proceso científico. La utilidad y efectividad de la ciencia y la ingeniería dependen fuertemente de las mediciones. Así como nadie tiene dudas acerca del rol principal de las mediciones en las diferentes ramas de la ingeniería, durante mucho tiempo fueron consideradas un lujo en el ámbito de la Ingeniería del Software y cuando se hicieron, a menudo fueron inconsistentes e incompletas. Está muy claro desde las otras disciplinas de la ingeniería que las mediciones pueden ser efectivas, si no esenciales, para valorar la dimensión y decidir soluciones a los problemas [Fenton'96].

La medición debe considerarse una actividad científica motivada por objetivos claros, comprensibles y ajustados a las necesidades de usuarios, desarrolladores y administradores.

Las métricas del software incluyen muchas actividades, que comprenden estimaciones de costo y esfuerzo, medidas de productividad, recolección de datos, medidas de calidad y confiabilidad, evaluación de performance, métricas de complejidad, evaluación de métodos y herramientas [Fenton'96].

Las mediciones son necesarias para valorar el estado de un proyecto, de un proceso, producto o recurso. Es esencial medir y registrar características, tanto de buenos como de malos proyectos. Necesitamos documentar las tendencias, la magnitud de las acciones correctivas y de los cambios resultantes. Tom DeMarco, un gran defensor de las mediciones, afirmó en 1982: "No se puede controlar lo que no se puede medir" [Fenton'96].

Debido a la gran importancia económica de la industria del software y a la necesidad de disponer de medidas de performance y de métodos de estimación confiables, los ingenieros del software se han visto forzados a pensar en nuevos enfoques con el fin de lograr esos objetivos. Durante muchos años se pensó que los programas de mediciones ocupan demasiado tiempo y que hay otras prioridades; con el correr del tiempo y los problemas surgidos con los sistemas de software, se comenzó a valorar la importancia de introducir programas de medición en las organizaciones, tratando que los mismos sean una actividad complementaria de los procesos de desarrollo y mantenimiento. Un beneficio de introducir programas de medición es que la acumulación de medidas de performance a largo plazo, puede ser extremadamente valiosa para ayudar a mejorar la estimación de proyectos. Producir estimaciones confiables es uno de los ingredientes claves para un mejor control del proyecto [Symons'91].

En el párrafo anterior se ha mencionado el término *performance*, es importante tener claro cuál es su significado. Hay tres aspectos a considerar con relación a la performance del desarrollo y mantenimiento, dependiendo de los objetivos. Un primer objetivo puede ser mejorar la *productividad*, es decir producir más para un cierto recurso dado o producir más con menor costo, el cual en términos de desarrollo de sistemas significa menos esfuerzo u horas trabajadas. La productividad se define como [Symons'91]:

Productividad = salida / entrada = tamaño del sistema / horas trabajadas

Otro objetivo vinculado a la performance está relacionado con la *entrega del sistema* más rápido, para esto la medida apropiada de la performance es entrega:

Entrega = tamaño del sistema / semanas consumidas

El tercer aspecto de la performance es el mejoramiento de la *calidad*. La calidad es difícil de definir y hay varias medidas posibles. Las más importantes son:

Densidad de defectos = cantidad de defectos / tamaño del sistema

También es pertinente hablar de calidad funcional, es decir la visión del cliente acerca de la calidad, en términos de facilidad de uso, confiabilidad, seguridad, precisión o de calidad técnica, es decir, la visión del proveedor acerca de la calidad del sistema, en términos de su diseño, mantenibilidad y operabilidad [Symons'91].

Cualquiera sea el objetivo planteado, en todos ellos el tamaño del sistema es una medida clave.

Si se piensa acerca del problema de estimar el tamaño total de la tarea de desarrollar un sistema, medido en horas de trabajo, entonces el tamaño tiene tres componentes [Symons'91]:

- Alguna medida de la cantidad de información a procesar por el sistema: entradas, salidas, etc.
- Alguna medida del tamaño de los requerimientos técnicos: batch vs. on-line, facilidad de uso, etc.
- Los controladores de performance (performance-drivers). Este grupo comprende todos aquellos aspectos que no son parte de los requerimientos funcionales y técnicos: administración del proyecto, objetivos de calidad, métodos, herramientas, lenguajes de programación, etc.

3.2.1. Beneficios de la aplicación de métricas

La aplicación de mediciones es muy importante porque favorece tres actividades básicas [Fenton'96]:

1. Las mediciones ayudan a *entender* qué está ocurriendo durante el desarrollo y mantenimiento. La evaluación de la situación actual permite establecer lineamientos que ayudan a fijar objetivos para futuros comportamientos. En este sentido las mediciones hacen visibles los aspectos de productos y procesos y por lo tanto mejoran la comprensión de las relaciones entre actividades y las entidades que ellos afectan.
2. Las mediciones permiten *controlar* lo que está ocurriendo en un proyecto. En base a los lineamientos, objetivos y comprensión de las relaciones se puede predecir qué puede ocurrir y realizar los cambios a procesos y productos que ayuden a alcanzar los objetivos.

3. Las mediciones estimulan a *mejorar* los procesos y productos. Se puede aumentar la cantidad o tipo de revisiones del diseño basándose en las medidas de calidad de la especificación y las predicciones de calidad del diseño.

3.2.2. Medición del tamaño del sistema

La Especificación de Requerimientos del Software es el primer producto tangible de la mayoría de los ciclos de vida del desarrollo y una de las principales fuentes de problemas en etapas posteriores. Aunque la mayoría de los atributos de calidad de los requerimientos documentados son subjetivos, hay aspectos de la documentación que se pueden medir, que son indicadores de atributos relacionados a la calidad. El tamaño es uno de los primeros usados en muchas métricas de calidad y se puede medir directamente [Wilson'97].

Cuando se va a estimar el tamaño de la tarea de desarrollo, tenemos que tener en cuenta los tres componentes mencionados anteriormente, mientras que para las mediciones de productividad, el tamaño del sistema que necesitamos es función solamente de los dos primeros [Symons'91].

La Figura 11 representa los dos componentes a considerar:

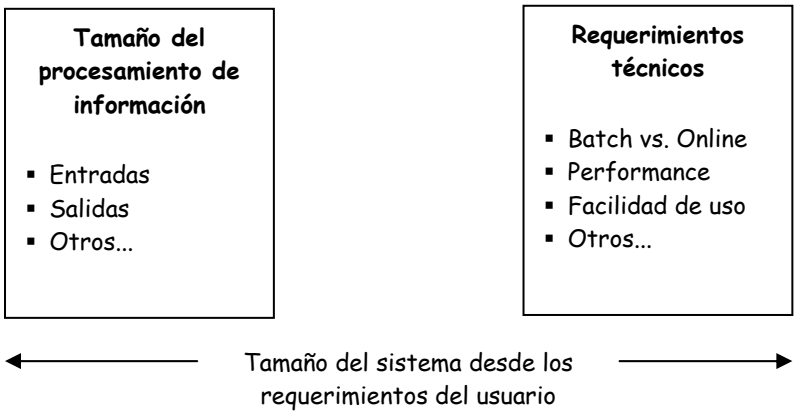


Figura 11 - Los componentes del tamaño del sistema [Symons'91]

3.3. Clasificación de las medidas del software

Teniendo conciencia de la necesidad de las mediciones, la primera tarea consiste en identificar las entidades y atributos que se pueden medir. Hay tres tipos de entidades [Fenton'96]:

Proceso: una colección de actividades relacionadas al software.

Producto: cualquier artefacto que resulta de una actividad de proceso.

Recurso: entidad requerida por una actividad de proceso.

Dentro de cada entidad se distinguen:

Atributos internos: son aquéllos que se pueden medir en términos de la entidad en sí misma.

Atributos externos: son aquéllos que solamente se pueden medir con respecto a cómo se relaciona la entidad con su entorno, o sea, su comportamiento.

Los atributos internos se pueden medir simplemente examinando la entidad, por ejemplo, el tamaño de un módulo de código o su complejidad, en tanto los externos sólo pueden evaluarse cuando el módulo se ejecuta, como es el caso de la cantidad de fallas o el tiempo que toma realizar una búsqueda y retorno de información. En general es más difícil medir atributos externos que internos y suelen medirse más tarde en el proceso de desarrollo. Puede complicarse aún más el panorama cuando esos atributos externos que se desean medir se pueden definir en términos de otros atributos que son más fáciles de medir en forma directa y no hay acuerdo en cuanto a cuáles son esos atributos, tal es el caso de evaluar el atributo externo calidad del software. Por lo tanto, es esencial identificar correctamente las relaciones entre los atributos internos y externos, así como encontrar métodos para medir en forma directa aquellos atributos de interés. La importancia de la medición de atributos internos para predecir atributos externos durante el proceso de desarrollo, se basa en la necesidad de monitorear, controlar y mejorar el proceso [Fenton'96].

3.4. Determinar qué medir

Una vez que se han identificado las entidades y los atributos, se debe determinar cuál es el objetivo de la medición. Establecer el objetivo ayudará a decidir qué entidades y atributos de las mismas deben medirse.

Debe determinarse en cada caso qué entidades son relevantes, qué atributos se van a medir y si la medición se utilizará para valoración o predicción.

Como se mencionó en el Capítulo 1, los enfoques tradicionales se han centrado en la medición del tamaño expresado en cantidad de líneas de código. Sin embargo existen otras formas de medir el tamaño de cualquier producto del desarrollo. Hay una cantidad de importantes aspectos específicos del tamaño, denominados longitud, funcionalidad, complejidad y también se puede considerar el reuso. La longitud es el tamaño físico del producto, la funcionalidad mide las funciones suministradas por el producto al usuario y la complejidad puede ser interpretada en diversas formas: complejidad del problema, complejidad algorítmica, complejidad estructural y complejidad cognitiva. El reuso mide cuánto del producto fue copiado o modificado desde otro producto existente [Fenton'96].

En este trabajo se realizarán mediciones sobre las entidades obtenidas como producto del proceso de elicitación, denominadas LEL y Escenarios. El atributo que nos interesa medir es un atributo interno, el tamaño del sistema en términos de su funcionalidad.

3.5. Líneas de código fuente (SLOC)

La cantidad de SLOC es una medida del tamaño de los requerimientos del usuario después que se ha construido el sistema y los requerimientos se han traducido a expresiones lógicas en la sintaxis de un lenguaje de programación. La gran ventaja es que las SLOC se pueden medir objetiva y automáticamente, pero tiene varias desventajas [Symons'91].

La primera de ellas y la principal es la definición de reglas para contar las SLOC en un lenguaje de programación dado y cómo sumarlas para un sistema escrito en más de un lenguaje. Este problema de la definición de reglas, que a primera vista pareciera simple, es bastante complejo, debido a que la cantidad de SLOC de un mismo lenguaje puede variar mucho según las convenciones adoptadas. Entre las convenciones que se deben establecer, pueden incluirse [Symons'91] [Fenton'96]:

- ¿Contar líneas de código procedural y/o sentencias declarativas?
- ¿Contar líneas en blanco?
- ¿Contar líneas de comentarios?
- ¿Contar las sentencias de declaración de datos solamente una vez o por cada módulo en que están incluidas?
- ¿Contar sentencias de expansión de macros y/o contar las expansiones de esas macros?
- ¿Contar las SLOC de funciones auxiliares necesarias para el procesamiento, que no son parte de los requerimientos del usuario?
- ¿Contar las SLOC de las funciones de backup y recuperación?

El problema de las convenciones es aún más complicado cuando el sistema utiliza más de un lenguaje. Se han establecido equivalencias entre distintos lenguajes, pero no hay convenciones aceptadas universalmente acerca de las mismas. Dentro de una misma organización es posible establecer e imponer algún conjunto de convenciones, pero esto se complica en general cuando se utilizan repositorios, herramientas CASE, etc. y se vuelve difícil determinar el significado de las SLOC [Symons'91].

Otra desventaja importante de SLOC es que no pueden medirse hasta que el sistema está completamente desarrollado, lo que implica que cualquier método de estimación que requiera el tamaño del sistema al principio del desarrollo se ve obstaculizado. Solamente aquellos profesionales con gran experiencia y habilidad para estimar por analogía con otros proyectos podrán utilizar esos métodos de manera confiable [Symons'91].

De cualquier manera, si se deben desarrollar sistemas en tiempo real, sistemas de control, sistemas operativos, herramientas de software, para los que algunos métodos de Análisis de Puntos Función (FPA) no son aplicables, las mediciones utilizando SLOC pueden ser el único enfoque disponible [Symons'91].

3.6. SLOC vs. Puntos Función

La cultura de muchas organizaciones de software está más cómoda haciendo estimaciones ad-hoc basándose en la experiencia previa o contando las SLOC nuevas o modificadas. Se han desarrollado muchas herramientas para contar o predecir la cantidad de SLOC, pero aún con los métodos más sofisticados, las estimaciones tempranas del tamaño y esfuerzo de los proyectos usando SLOC, especialmente aquéllas que implican un nuevo lenguaje o metodología, son inadecuadas y a menudo llevan a un exceso en costos y tiempo. Los Puntos Función (FP) son una unidad de medida lógica de las funciones del sistema de software desde la visión del usuario. Éstos proveen el valor esencial de lo que es el software y qué hace con los datos desde el punto de vista del usuario. Su potencia proviene del énfasis sobre el punto de vista externo. Debido a que la estimación de esfuerzo y costo basada en FPA no depende del lenguaje y tecnología utilizados, se puede disponer del cálculo desde las primeras etapas del desarrollo, particularmente los FP pueden calcularse desde los requerimientos. Solamente esto último ha promovido la amplia aceptación de esta técnica [Bernstein'95].

Las métricas orientadas al tamaño son objeto de polémicas y no están aceptadas universalmente como el mejor modo de medir el proceso de desarrollo de software. La mayor parte de la discusión gira en torno al uso de las SLOC como medida clave. Sus defensores afirman que es un artificio que se puede calcular fácilmente para todos los proyectos de desarrollo de software, que muchos modelos de estimación del software existentes las utilizan como elemento de entrada y que existe un amplio conjunto de datos y de literatura basados en SLOC. En el extremo opuesto, los detractores afirman que las medidas en SLOC son dependientes del lenguaje de programación, que perjudica a los programas más cortos pero bien diseñados, que no se pueden adaptar fácilmente a lenguajes no procedimentales y que su utilización en la estimación requiere un nivel de detalle que puede ser difícil de conseguir (puede ser necesario estimar las SLOC a producir antes de completar el análisis y diseño) [Pressman'93].

Muchos ingenieros de software prefieren estimar la funcionalidad en vez del tamaño físico. El concepto de funcionalidad captura la noción de cantidad de función contenida en un producto entregado o en la descripción de lo que se supone será el producto [Fenton'96]. Las métricas orientadas a la función son medidas indirectas del software y del proceso por el cual se desarrolla. Estas métricas se centran en la "funcionalidad" o "utilidad" del programa [Pressman'93].

Para sustentar este enfoque de funcionalidad se han desarrollado métodos que han sido probados y revisados sobre la base de la experiencia. Estos métodos se conocen como Métodos de Análisis de Puntos Función [Fenton'96].

Una cuestión muy importante es el ámbito de aplicación de las métricas. La medida de FP se diseñó para ser utilizada en aplicaciones de sistemas de información de gestión [Pressman'93]. Hay ciertos tipos de software en que las medidas en FP como están definidas por algunos métodos no son confiables (ver Figura 12) [Symons'91].

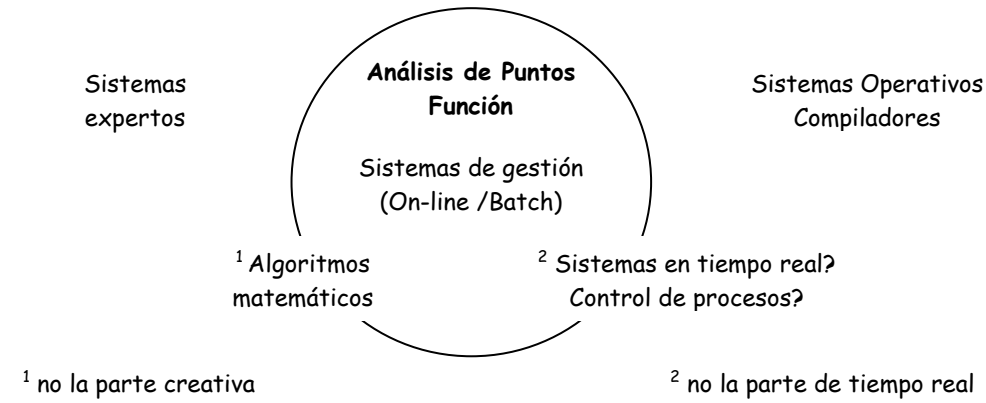


Figura 12 – Ámbito de aplicación de los métodos de FPA [Symons'91]

En la actualidad esta restricción ha sido superada por el método COSMIC FFP que puede aplicarse también a software para aplicaciones en tiempo real [Symons'01].

3.7. Métodos de Análisis de Puntos Función

3.7.1. Introducción

La definición estándar de productividad es "bienes o servicios producidos por unidad de trabajo o costo". Hasta 1979, cuando Albrecht de IBM publicó su métrica de FP no había una definición de exactamente qué bienes o servicios eran el resultado de un proyecto de software. La métrica previa era "costo por línea de código", la cual no se relaciona completamente a la definición económica de productividad. Los administradores de fábricas entienden que si un proceso de fabricación implica un porcentaje sustancial de costos fijos y disminuye la cantidad de unidades fabricadas, el costo por unidad debe aumentar. El software implica un porcentaje sustancial de costos fijos que no están asociados con el código. Cuanto más poderosos son los lenguajes de programación que se usan, el resultado es la reducción de las unidades que se deben producir para un sistema. Sin embargo, los requerimientos, especificaciones, documentación del usuario y muchos otros elementos del costo tienden a comportarse como costos fijos y por lo tanto, provocan que métricas tales como "costo por líneas de código", paradójicamente, aumente en lugar de disminuir [Jones'97].

Albrecht estableció que la unidad económica resultante de los proyectos de software debía ser válida para todos los lenguajes y representar tópicos concernientes a los usuarios del software. Brevemente, deseaba medir la funcionalidad del software. Albrecht consideraba que los aspectos externos visibles del software que se podrían enumerar con precisión consistían en cinco ítems: las entradas hacia la aplicación, las salidas desde ella, las consultas de los usuarios, los archivos de datos que pudieran actualizarse y las interfases hacia otras aplicaciones. A partir de un proceso de prueba y error, surgieron los factores de peso empíricos para los cinco ítems, que representan la dificultad de implementar cada uno de ellos. Esto marcó un hito en la historia de la computación: que se pudiera medir la productividad económica del software [Jones'97].

El FPA es una medida de clara significación para los sistemas administrativos. La técnica FPA cuantifica las funciones contenidas dentro del software en términos que son significativos para los usuarios del software. Esta medida se relaciona directamente a los requerimientos de ese tipo de aplicaciones. Se puede aplicar a un amplio rango de entornos de desarrollo y a través del ciclo de vida de un proyecto de desarrollo, desde la primera definición de los requerimientos hasta el uso operacional completo, permitiendo refinar la estimación de tamaño y por lo tanto, las estimaciones de costo o productividad [IFPUG'00^{b}] [Fenton'96].

Los FP miden el tamaño del software cuantificando la funcionalidad provista al usuario basándose solamente en el diseño lógico y las especificaciones funcionales [IFPUG'00^{a}].

El cálculo de FP se puede realizar con documentación mínima. Sin embargo, la precisión y eficiencia de las cuentas mejoran con documentación apropiada. Ejemplos de documentación apropiada son [Heller'95]:

- Especificación de diseño.
- Visualización del diseño.
- Requerimientos de datos (internos y externos).
- Descripción de las interfases del usuario.

En general se tiende a pensar en FPA como una técnica para realizar mediciones de tamaño que permite realizar comparaciones de productividad a través de proyectos, sistemas y empresas o para estimaciones de costos, sin embargo, el factor más importante es que FPA visualiza a los sistemas desde la perspectiva del usuario. Los usuarios ingresan información a la aplicación, obtienen información y reconocen grupos discretos de datos, entidades o archivos que el sistema usa y mantiene. Al descomponer el sistema en sus partes, se puede alcanzar el más bajo nivel de funcionalidad reconocible por el usuario (Figura 13) [Goodman'99].

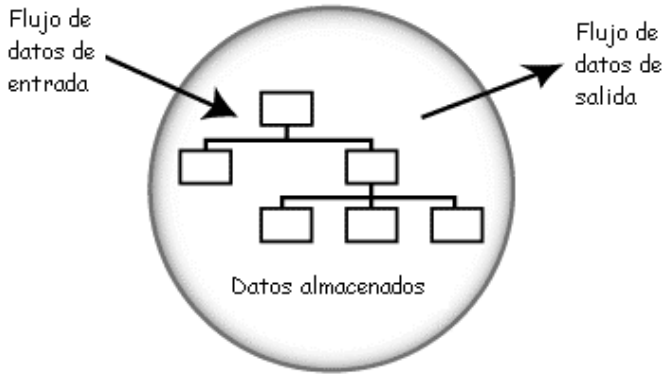


Figura 13 – Visión del sistema desde la perspectiva del usuario [Goodman'99]

La medición de los requerimientos ayuda a los ingenieros a producir requerimientos no ambiguos y medibles, además de mejorar el rigor con que éstos son documentados. En consecuencia, pueden usarse como base para los acuerdos entre clientes y proveedores [Grant Rule'01^{b}].

Con respecto a esto último, es oportuno destacar los siguientes párrafos [Grant Rule'01ª]:

- Los usuarios pagan para que sean satisfechos sus requerimientos; el código no es su preocupación, para un usuario el producto entregable es la satisfacción de sus requerimientos.
- Las medidas del *tamaño funcional* expresan la cantidad de procesamiento de información entregada: la comunidad del software necesita una medida de la cantidad de funcionalidad que el cliente requiere del software, independientemente de la tecnología usada y de la gente que lo produce. Esas medidas son de importancia crucial, no sólo para la administración del proyecto, sino para tomar decisiones económicas.
- El tamaño es relevante desde la definición de requerimientos hasta la entrega final: los proveedores de software usualmente tienen presiones para estimar el esfuerzo y los costos de desarrollo al principio del ciclo de vida de un proyecto. A menudo hay necesidad de establecer compromisos basados en una inadecuada comprensión de los requerimientos del cliente. Cualquier técnica que ayude a remover la ambigüedad, mejorar la definición de los requerimientos y que permita a clientes y proveedores acordar los términos y controlar el alcance y progreso de los contratos, será bienvenido por la comunidad del software.

Se han desarrollado varios métodos de FPA, entre ellos el de Albrecht, MarkII, COSMIC y unas cuantas variaciones propuestas por otros autores. En la Figura 14 se muestra la evolución de los principales métodos.

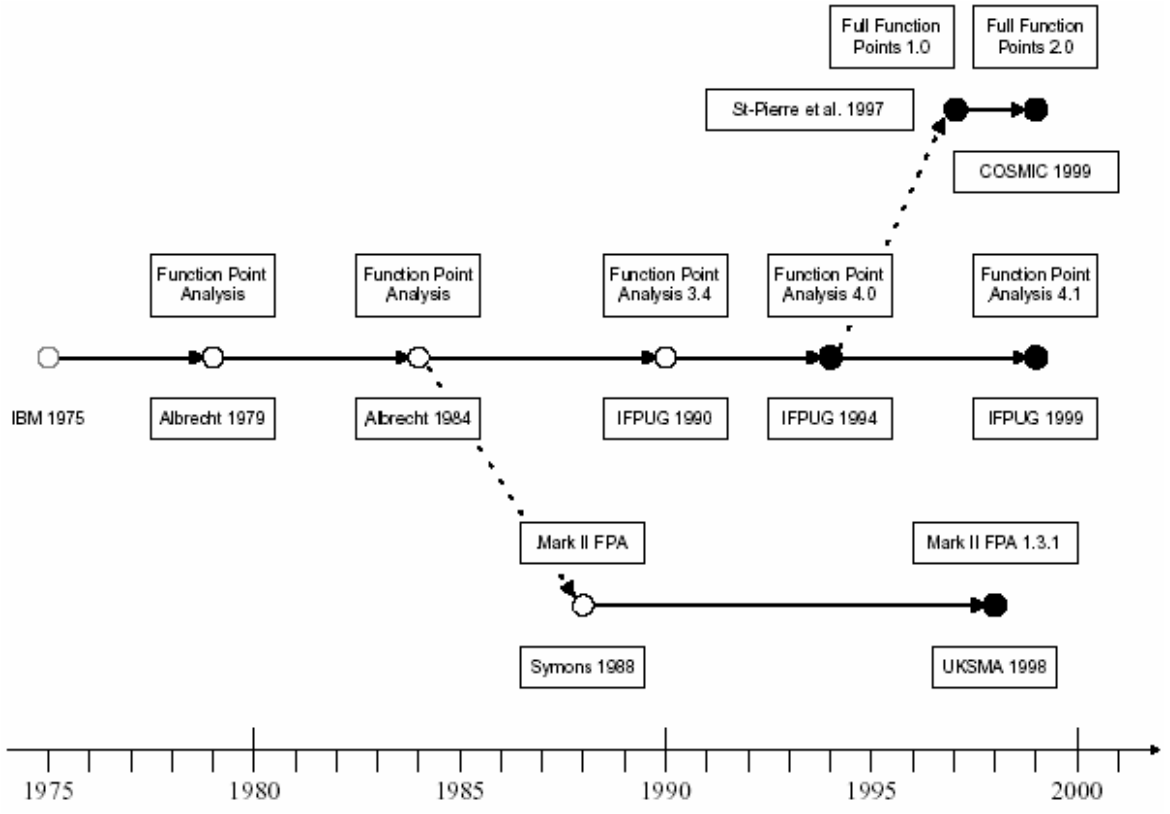


Figura 14 - La evolución de los métodos de medición del tamaño funcional [Fetcke'99]

Debido a la amplia difusión de las métricas de FP surgió la necesidad de definir estándares. En 1996 la International Standards Organisation inició un grupo de trabajo (ISO/IEC JTC1 SC7 WG12) con la participación de expertos en métricas del software para establecer los principios comunes de la medición del *tamaño funcional*. La primera publicación que estableció los principios básicos fue ISO/IEC 14143-1 [Symons'01]. Los métodos IFPUG, MKII FPA y NESMA (variante del método IFPUG mantenido por Netherlands Software Metrics Association) han sido aprobados por ISO. En febrero de 2003 se aprobó el estándar ISO/IEC 19761:2003 para el método COSMIC –FFP [COSMIC-FFP'03].

Se describirán en detalle los métodos de Albrecht por tratarse del precursor de la idea y MarkII que será la base para el desarrollo del enfoque propuesto en esta tesis.

La Tabla 1 muestra un cuadro comparativo con algunos aspectos de los métodos.

Consideración	SLOC	Albrecht FP	MarkII FP
Definición			
¿Estándar aceptado?	No, varios	Sí, se está refinando	Sí, recomendado por CCTA para UK Gov.
¿Claridad de la definición?	Potencialmente preciso	Subjetivo en parte	Razonablemente objetivo
¿Basado en métodos estructurados?	No	No	Sí
Usabilidad			
Facilidad de uso	Fácil	← No tan fácil como parece →	
¿Automatizado?	Sí	Muy difícil	Sí (herramienta CASE)
¿Usable para estimación?	En algunas circunstancias	Sí	Sí
Disponibilidad			
Público/propietario	Ambas	Público	Público desde 1991
Base de usuarios	Muy amplio	Muy amplio	Adecuado
Grupo de usuarios	Varios (Ej. COCOMO User Groups)	Varios (Ej. IFPUG)	Sí (Ej. EFPUG)
Entrenamiento público	No disponible	Sí	Sí

Tabla 1 - Comparación entre los métodos [Symons'91]

3.7.2. Proceso de medición del tamaño funcional [Fetcke'99]

La medición puede entenderse como una abstracción que captura ciertos atributos de los objetos de interés. La teoría de la medición visualiza esta abstracción como un

mapeo que asigna objetos numéricos a objetos empíricos. En la medición de *tamaño funcional*, esos objetos empíricos son las aplicaciones de software. Esta técnica de medición requiere dos pasos de abstracción.

Los dos pasos de abstracción

En lugar de usar los conceptos y modelos de un método de desarrollo particular, cada variante del FPA define sus propios conceptos para la representación de una aplicación de software. Esos métodos, por lo tanto, definen una abstracción del software que representa los ítems considerados relevantes para el *tamaño funcional*. Las variantes de FPA que se muestran en la Figura 14 se basan en una abstracción orientada a los datos. Esos métodos definen los dos pasos de abstracción siguientes:

1. Los requerimientos funcionales del usuario están representados en la abstracción orientada a los datos.
2. Los ítems de la representación orientada a los datos se asocian con números.

El primer paso de la abstracción se aplica a la documentación de la funcionalidad del software. El resultado es una representación que contiene los ítems significativos para el *tamaño funcional*. Este paso requiere de la evaluación de reglas por parte del profesional.

El segundo paso es la medición, es decir, el mapeo de esos ítems con números. La fuente para este paso es la abstracción orientada a los datos.

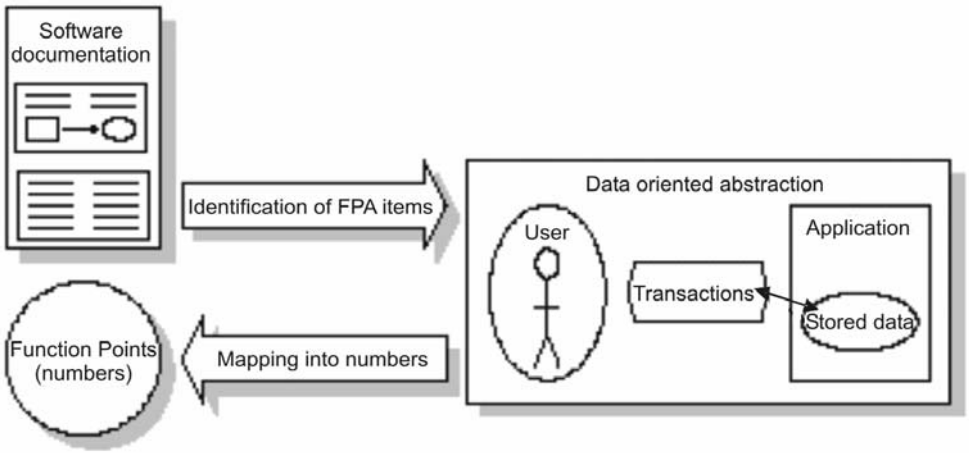


Figura 15 - Los dos pasos de abstracción de las variantes del FPA [Fetcke'99]

La Figura 15 ilustra la visión de los dos pasos de la abstracción orientada a los datos. Ambos pasos están definidos por las reglas de los métodos de medición del *tamaño funcional*. Sin embargo, en la documentación de los mismos no suele aparecer tan claramente la separación entre esos dos pasos de la abstracción.

La abstracción orientada a los datos

El IFPUG FPA plantea una abstracción orientada a los datos de la aplicación medida. Se han propuesto variantes del FPA para mejorar el enfoque original. Estas variantes difieren en los ítems identificados y en las funciones medidas.

Particularmente el enfoque MarkII se basa en los mismos conceptos centrales que el estándar IFPUG [Fetcke'99].

- *Concepto de usuario*. El usuario interactúa con una aplicación y no necesariamente se restringe a usuarios humanos, sino que puede incluir aplicaciones de software y hardware.
- *Concepto de aplicación*. La aplicación es el objeto de la medición. Las aplicaciones proveen funciones al usuario. Esas funciones son los atributos de interés, precisamente, los requerimientos que especifica el usuario para esas funciones.
- *Concepto de transacción*. Las *transacciones* son procesos de interacción del usuario con la aplicación desde una perspectiva lógica o funcional. En términos de IFPUG FPA, las *transacciones*:
  - son las unidades de actividad más pequeñas significativas para el usuario.
  - son autocontenidas, es decir lógicamente completas y
  - dejan la aplicación en estado consistente.
- *Concepto de dato*. Los datos son almacenados por la aplicación. Los elementos dato representan los ítems de datos más pequeños significativos para el usuario y están estructurados en grupos lógicamente relacionados, similares a las tablas en una base de datos.
- *Concepto de tipo*. En el nivel más alto de la abstracción orientada a los datos, todas las variantes de FPA identifican tipos de *transacciones* y tipos de grupos de datos. El término *tipo* se refiere al principio que múltiples instancias del mismo componente lógico se identifican sólo una vez. Este concepto es esencial para la visión del *tamaño funcional* en las variantes de FPA.

  La Figura 16 ilustra estos conceptos de la abstracción orientada a los datos.

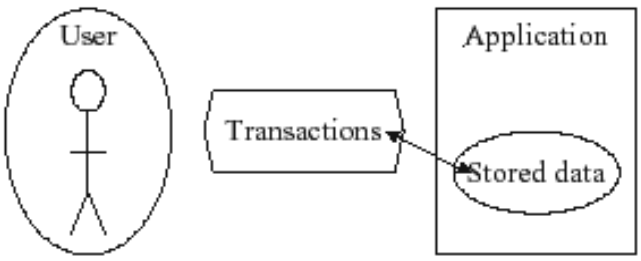


Figura 16 – La abstracción orientada a los datos del FPA [Fetcke'99]

3.7.3. FPA y los requerimientos

Hay una diversidad de enfoques tradicionales para identificar y obtener los requerimientos del software que usados adecuadamente permiten entregar una forma documentada de los requerimientos del usuario. Después de una serie de revisiones con el usuario, se asume que esos requerimientos formales representan el conjunto completo de requerimientos del usuario. Sin embargo, el conjunto completo de requerimientos del usuario generalmente no está completo hasta el final del proyecto y continúa evolucionando mientras éste progresa. La experiencia indica que FPA es

efectivo para identificar el conjunto completo de requerimientos funcionales del usuario y descubrir defectos potenciales. De hecho, los beneficios obtenidos por la aplicación de FPA a los requerimientos funcionales del usuario pueden ser más valiosos que el mero tamaño en FP del software [Dekkers'01].

En este punto vale la pena identificar los principales tipos de requerimientos del software y el aporte de FPA en cada caso. Primero están los requerimientos funcionales del usuario, que son las funciones que el sistema debe ejecutar. Todos los sistemas desde los de tiempo real hasta los sistemas administrativos tienen requerimientos funcionales. Estos son los procesos elementales que deben realizarse para la entrada, proceso, manipulación, salida e interfases de datos desde y hacia el software. FPA soporta este tipo de requerimientos del usuario. Los segundos son los requerimientos no-funcionales. Éstos son las restricciones independientes de la tecnología y de la empresa, que el software debe considerar. Estos requerimientos incluyen la calidad y los requerimientos de performance como portabilidad, usabilidad, seguridad, confiabilidad y velocidad. Parte de las técnicas de FPA pueden asistir con ese tipo de requerimientos. Por ultimo, los requerimientos técnicos que son los requerimientos del usuario para una determinada configuración de hardware/software o una configuración técnica que debe ser entregada. Por ejemplo, los requerimientos técnicos pueden especificar una base de datos determinada o una solución de hardware concurrente. FPA no soporta este tipo de requerimientos [Dekkers'01].

3.7.4. Beneficios de la aplicación del FPA [Goldfarb'00]

El FPA provee las bases para:

Mejorar la definición de los requerimientos. Las técnicas FPA ayudan a los ingenieros a producir requerimientos del software que son mensurables. Reducen la ambigüedad y mejoran el rigor con que son documentados los requerimientos. Se pueden usar como base para el acuerdo entre clientes y proveedores [Grant Rule'01ª].

Comunicar requerimientos funcionales. Este fue el objetivo original subyacente del desarrollo de los FP. A partir de evitar la terminología técnica y enfocar sobre los requerimientos del usuario, resulta un excelente vehículo para comunicarse con los usuarios. La técnica puede usarse para realizar entrevistas a los clientes y documentar los resultados obtenidos. La documentación resultante provee un marco (framework) que describe los requerimientos del usuario y técnicos [Heller'95].

Estimar el esfuerzo, agenda y costos basándose en los requerimientos. Cuando un cliente requiere una nueva aplicación de software desarrollada por un proveedor, necesita una estimación del costo de desarrollo a medida que evolucionan los requerimientos, para asegurar la decisión acerca de la óptima inversión costo beneficio. Para determinar el precio el proveedor debe estimar el esfuerzo de desarrollo, el personal necesario y por lo tanto, el costo resultante, comenzando solamente desde el tamaño de los requerimientos. Esas estimaciones se necesitan al principio, pero claramente cuanto más temprano se hacen son más inciertas. En consecuencia, la estimación necesita repetirse, mejorando la precisión cuando la

comprensión de los requerimientos es más detallada [Grant Rule'01ª]. Dentro de los varios factores que necesitan considerarse para estimar un proyecto de software, los dos puntos claves esenciales son: el tamaño del producto entregable y cuánto de ese producto se puede producir en un período definido de tiempo. El tamaño se puede derivar desde los FP y el segundo requerimiento para la estimación se determina a partir del tiempo que toma producir un FP [Heller'95].

Evaluar y administrar la factibilidad de un proyecto. En una organización cuyos recursos están limitados en términos de fondos, habilidades, disponibilidad, tiempo u otro recurso, lo que pueda lograrse estará determinado por la cantidad de recursos y la capacidad de la organización para producir software. La factibilidad de un proyecto estará restringida por la capacidad en términos de cantidad de resultados que se pueden producir para un cierto desembolso en un tiempo dado. La experiencia adquirida con el FPA en proyectos previos permite hacer estimaciones antes de considerar en detalle los requerimientos de uno nuevo [Grant Rule'01ª].

Administrar los cambios. Este es otro beneficio clave del FPA para un proyecto. Una vez que el proyecto ha sido aprobado y se han calculado los FP, es una tarea relativamente fácil identificar, rastrear y comunicar los requerimientos nuevos o que han sido cambiados. Cuando el usuario propone un cambio en los requerimientos, se desarrollan los FP y el resultado se usa para determinar el impacto en el esfuerzo y presupuesto del proyecto. El usuario y el equipo de proyecto pueden determinar la importancia del cambio requerido frente al impacto en el presupuesto y la agenda. A la finalización del proyecto se puede evaluar la cuenta final de FP frente a la estimación inicial para determinar la efectividad de las técnicas para obtener requerimientos. Este análisis ayuda a identificar oportunidades para mejorar el proceso de definición de requerimientos [Heller'95].

Mejorar el mantenimiento y soporte de la aplicación. Los FP se pueden usar para evaluar el soporte a los requerimientos de mantenimiento de sistemas. En este análisis la productividad se determina calculando la cantidad de FP que un individuo puede mantener para un sistema dado en el término de un año. Cuando se compara con otros sistemas, esa tasa ayuda a identificar qué sistemas requieren mayor soporte. Ese análisis ayuda a la organización a desarrollar una estrategia de mantenimiento y reemplazo para esos sistemas [Heller'95].

Medir la productividad. Es una salida natural del FPA. Puesto que los FP son independientes de la tecnología, se pueden usar para comparar la productividad a través de diferentes herramientas y plataformas. Más importante aún, pueden usarse para establecer una tasa de productividad para un conjunto de herramienta y plataforma específica. Una vez que esa tasa de productividad está establecida, puede usarse para estimar proyectos y rastrear a lo largo del tiempo para determinar el impacto que tienen las iniciativas de mejoramiento del proceso sobre la productividad [Heller'95].

Verificar la completitud de los requerimientos. Para verificar la completitud, uno de los problemas es asegurar que se ha identificado el conjunto completo de requerimientos. Para ello, los analistas toman como marco de referencia los modelos

teóricos del análisis estructurado, modelado de datos y/o la experiencia adquirida a partir de otros sistemas similares. FPA provee un marco de referencia adicional para verificar la completitud de los requerimientos funcionales basado en una perspectiva enfocada al usuario. FPA examina el conjunto de requerimientos en términos de datos y movimiento/manipulación de los mismos *(transacciones)*, tal como son entendidos y expresados por los usuarios y sobre esta base determina el *tamaño funcional* del software. Cada uno de los pasos del FPA ayuda a clarificar los requerimientos, permitiendo confirmar los requerimientos originales o revelando las inconsistencias en la comprensión. El aporte a la completitud va mucho más allá de la fase de requerimientos. Simplemente revisando el listado de FP se puede determinar si una cierta funcionalidad está incluida y ayudar a tomar decisiones acerca de incluirla o no [Dekkers'01].

3.7.5. Método de Albrecht

El primer método basado en métricas de funcionalidad fue propuesto por Alan Albrecht. La propuesta partía de un concepto distinto, el desarrollo de un índice de la funcionalidad de un sistema como una medida asociada a su tamaño [Symons'91]. Desde entonces, ha aumentado el uso del FPA en el mundo del desarrollo de sistemas y los métodos se han perfeccionado basándose en la experiencia acumulada. Existe una organización, el International Function Point Users Group (IFPUG) que se ocupa de mantener, actualizar y difundir la información y la práctica.

La idea es que se pueden calcular FP sin forzar a la especificación a ajustarse a un modelo o técnica de especificación [Fenton'96]. El proceso de medición se realiza en el contexto de un modelo abstracto del sistema. Ese modelo se compone de *transacciones* y *archivos*. Estos elementos se identifican a partir de los documentos de requerimientos, técnicas de diseño estructurado u otros modelos.

Cuando Albrecht desarrolló el FPA tenía por objetivos lograr [Symons'91]:

- Una medida consistente del tamaño del sistema para usar en mediciones o estimaciones de productividad.
- Un método significativo para el usuario final. Al estar relacionados directamente a los requerimientos del usuario, los FP deberían ser más fáciles de comprender que las SLOC.
- Reglas para contar los FP que deberían ser fáciles de aplicar.
- FP que deberían ser estimables a partir de la especificación de requerimientos.
- FP que deberían ser independientes de la tecnología usada para desarrollar el sistema.

Los seres humanos resuelven los problemas descomponiéndolos en piezas más pequeñas que les resultan más comprensibles. Los problemas que pueden parecer difíciles, son simples una vez que son divididos en partes. Clasificar las cosas, colocarlas en alguna categoría, es un proceso corriente. Cuando los objetos a clasificar son los contenidos de sistemas, se debe usar un conjunto de definiciones y

reglas para colocar a esos objetos en la categoría apropiada. El FPA es una técnica estructurada de clasificación de los componentes de un sistema [Longstreet'96ª].

En el mundo del FPA, los dos componentes del tamaño del sistema de la Figura 11, están representados por cinco grandes clases el primero y por las denominadas características generales del sistema el segundo [Longstreet'96ª].

El componente tamaño del procesamiento se descompone en: *entradas externas (External Inputs)*, *salidas externas (External Outputs)* y *consultas externas (External Inquires)*, cada una de las cuales interactúa con archivos y son llamadas *transacciones*. Los otros dos son los *archivos lógicos internos (Internal Logical Files)* y los *archivos de interfase externa* hacia otras aplicaciones *(External Interface Files)* [Symons'91].

El componente requerimientos técnicos está representado por las características generales del sistema que valoran la funcionalidad general del sistema [Symons'91].

En la Figura 17 se visualizan gráficamente esos componentes.

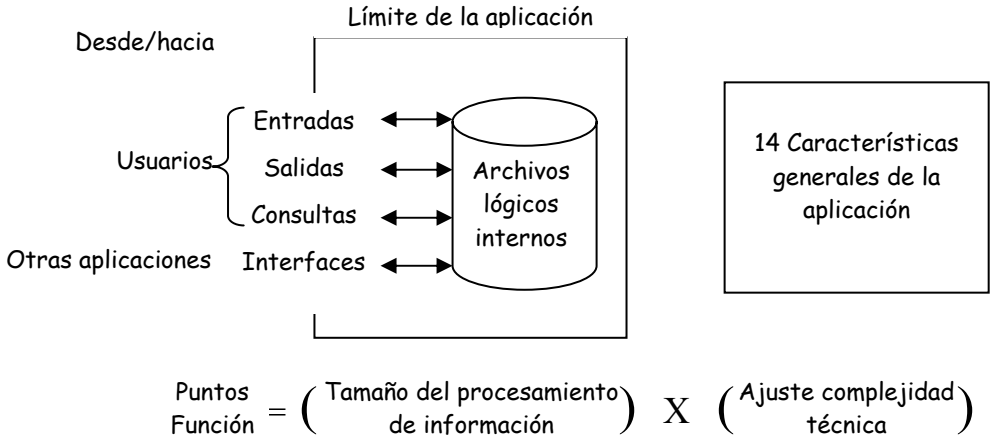


Figura 17 - Componentes del Método de Puntos Función de Albrecht [Symons'91]

Debido a que los FP miden los sistemas desde una perspectiva funcional, son independientes de la tecnología. Independientemente del lenguaje, método de desarrollo o plataforma de hardware usados, la cantidad de FP permanecerá constante. La única variable es la cantidad de esfuerzo necesario para entregar un conjunto de FP. Por lo tanto, el FPA se puede usar para determinar si una herramienta, un entorno o un lenguaje es más productivo comparado con otros, dentro de una organización o entre organizaciones [Longstreet'96ª].

El componente tamaño del procesamiento: los cinco principales componentes

Debido a que es común que los sistemas interactúen con otros sistemas, debe trazarse un *límite* alrededor del sistema a medir, previo a clasificar sus componentes. Este *límite* se debe trazar de acuerdo al punto de vista del usuario e indica el borde entre el proyecto o aplicación que se está midiendo y las aplicaciones externas o el dominio del usuario [Longstreet'96ª].

Entradas externas (*EI*) es un proceso elemental en el cual los datos cruzan el *límite* desde el exterior hacia el interior del sistema. Estos datos pueden venir desde una pantalla de entrada de datos o desde otra aplicación. Los datos pueden ser usados para mantener uno o más *archivos lógicos internos*. Ver Figura 18 [Longstreet'96ª] [Longstreet'96ᵇ].

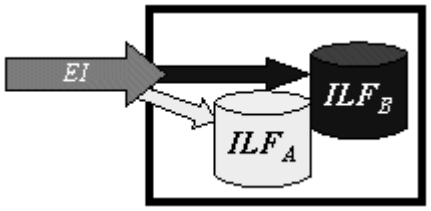


Figura 18 - EI que actualiza 2 ILF

Salidas externas (*EO*) es un proceso elemental en el cual los datos derivados atraviesan el *límite* desde el interior hacia el exterior del sistema. Además, un *EO* puede actualizar un *archivo lógico interno*. Los datos crean reportes o archivos de salida hacia otras aplicaciones. Esos reportes y archivos son creados a partir de uno o más *archivos lógicos internos* y *archivos de interfase externa*. Ver Figura 19 [Longstreet'96ª] [Longstreet'96ᵇ].

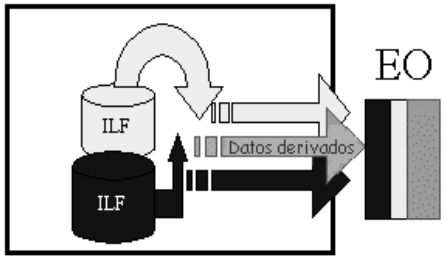


Figura 19 - EO que referencia y deriva información de 2 ILF

Consultas externas (*EQ*) es un proceso elemental con componentes de entrada y salida que resulta en el retorno de datos desde uno o más archivos lógicos internos y archivos de interfases externas. El proceso de entrada no actualiza archivos *ILF* y la salida no contiene datos derivados. Ver Figura 20 [Longstreet'96ᵇ].

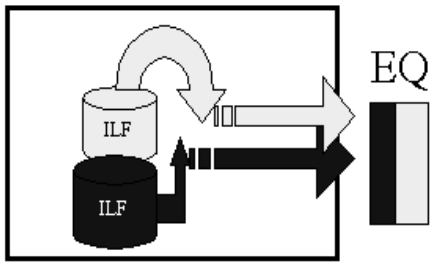


Figura 20 - EQ con 2 ILF y sin datos derivados

Archivos lógicos internos (*ILF*) es un grupo de datos lógicamente relacionados, identificable por el usuario, que residen completamente dentro del *límite de la aplicación* y es mantenido a través de entradas externas [Longstreet'96ª] [Longstreet'96ᵇ].

Archivos de interfase externa (*EIF*) es un grupo de datos lógicamente relacionados, identificable por el usuario, que se usa solamente con propósitos de *referencia*. Los datos residen completamente fuera de la aplicación y son mantenidos por otra aplicación [Longstreet'96ª] [Longstreet'96ᵇ].

Una vez que todos los componentes han sido clasificados como *EI*, *EO*, *EQ*, *ILF* o *EIF*, se les asigna una posición en la escala que puede ser: *bajo*, *medio* o *alto (Low, Average o High)*. Para las *transacciones* (*EI, EO, EQ*) esa posición se basa en la cantidad de *archivos actualizados o referenciados* (*FTR*) y la cantidad de *tipos de elemento dato (Data element type - DET)*. Un *DET* es un campo único, no recursivo, reconocible por el usuario. En el caso particular de las *EQ* se ubican en la escala *(bajo, medio o alto)* como *EO*, pero se les asigna un factor como *EI*. Si el mismo *FTR* se usa en la parte de entrada y salida se cuenta una sola vez. Si el mismo *DET* se usa en la parte de entrada y salida, también se cuenta sólo una vez [Longstreet'96ª].

Cada una de las siguientes tablas se utiliza en el proceso de asignación de posiciones en la escala y sus factores asociados:

FTR	Elementos dato		
	1-4	5-15	>15
0-1	Bajo	Bajo	Medio
2	Bajo	Medio	Alto
3 o más	Medio	Alto	Alto

Tabla 2 - EI [Longstreet'96ª]

FTR	Elementos dato		
	1-5	6-19	>19
0-1	Bajo	Bajo	Medio
2-3	Bajo	Medio	Alto
> 3	Medio	Alto	Alto

Tabla 3 - EO y EQ [Longstreet'96ª]

Posición	Factor		
	EO	EQ	EI
Bajo	4	3	3
Medio	5	4	4
Alto	7	6	6

Tabla 4 - Transacciones [Longstreet'96ª]

Para los archivos *ILF* y *EIF* la posición en la escala *(bajo, medio, alto)* se basa en los *tipos de elementos registro* (*RET*) y *DET*. Un *RET* es un subgrupo de datos reconocible por el usuario dentro de un *ILF* o *EIF* [Longstreet'96ª].

RET	Elementos dato		
	1-19	20-50	>50
1	Bajo	Bajo	Medio
2-5	Bajo	Medio	Alto
>5	Medio	Alto	Alto

Tabla 5 – Tipo registro [Longstreet'96ª]

Posición	Factor	
	ILF	EIF
Bajo	7	5
Medio	10	7
Alto	15	10

Tabla 6 – ILF y EIF [Longstreet'96ª]

Las cuentas para cada nivel de complejidad de cada tipo de componente se pueden ingresar en una tabla como la siguiente (Tabla 7). Cada cuenta se multiplica por el puntaje numérico indicado para determinar el puntaje total. Los valores de cada fila se suman a través de la tabla resultando el valor total para cada tipo de componente. Esos totales luego se suman a través de la tabla para obtener la cantidad total de *FP sin ajustar* (*Unadjusted Function Points - UFP*) [Longstreet'96ª].

Tipo de componente	Complejidad de los componentes			
	Bajo	Medio	Alto	Total
Entradas Externas (EI)	x 3 =	x 4 =	x 6 =	
Salidas Externas (EO)	x 4 =	x 5 =	x 7 =	
Consultas Externas (EQ)	x 3 =	x 4 =	x 6 =	
Archivos lógicos internos (ILF)	x 7 =	x 10 =	x 15 =	
Archivos de interfase externa (EIF)	x 5 =	x 7 =	x 10 =	
		UFP		
		TCA		
		FP		

Tabla 7 – Cálculo de Puntos Función de Albrecht [Longstreet'96ª]

El componente complejidad técnica

El valor de *Ajuste de complejidad técnica (TCA)* se obtiene a partir de las 14 características generales del sistema que asignan un puntaje a la funcionalidad general de la aplicación que se está midiendo. Esas características tienen descripciones asociadas que ayudan a determinar los *grados de influencia (DI)* de las características. Los *grados de influencia* varían en una escala de cero a cinco, desde ninguna a fuerte influencia. El objetivo de este factor es considerar la influencia sobre el *tamaño funcional* de los requerimientos de calidad y técnicos. En la Tabla 8 se detallan las características generales [Longstreet'96ª].

Características Generales del Sistema			
1	Comunicación de datos	8	Actualización interactiva
2	Procesamiento de datos distribuido	9	Complejidad de procesamiento
3	Performance	10	Reusabilidad
4	Entorno operativo muy utilizado	11	Facilidad de instalación
5	Frecuencia de transacción	12	Facilidad de operación
6	Entrada de datos interactiva	13	Múltiples instalaciones
7	Eficiencia usuario final	14	Facilidad de cambios

Tabla 8 - Características Generales del Sistema para Albrecht FPA [Longstreet'96ª]

Una vez que han sido valorados los 14 factores, debe aplicarse la ecuación de *Ajuste de complejidad técnica (TCA)* del IFPUG.

TCA = 0.65 + 0.01 * Σ DI

donde *DI* son los *grados de influencia* dados por cada una de las 14 características generales del sistema.

La cuenta final de FP se obtiene multiplicando el valor UFP por TCA.

FP = UFP * TCA

Se completa la Tabla 7 con el valor TCA y se obtiene FP.

Algunas críticas al Método de Albrecht

A pesar de ser una idea muy interesante y novedosa, al experimentar con el uso del método de Albrecht surgieron una cantidad de dificultades y anomalías. Entre ellas se puede mencionar [Symons'91]:

- Los componentes propuestos por Albrecht a veces son difíciles de identificar en la práctica del desarrollo de sistemas.
- Es razonable preguntarse cómo determinó Albrecht los pesos (o puntajes) usados para los diferentes tipos de componentes y para los diferentes niveles de complejidad cuando se calculan los *UFP*.

- Una crítica similar se puede hacer respecto a la elección de los componentes y los pesos para las características generales de la aplicación del *Ajuste de complejidad técnica*.
- Otra cuestión que se plantea es si este método trata adecuadamente con la complejidad de procesamiento interno del sistema.
- Por último, las reglas de Albrecht parecen favorecer a los sistemas discretos frente a los sistemas integrados.

Las más serias deficiencias del Método de Albrecht son aquéllas que surgen a partir del estrecho rango de puntajes en FP que se puede adjudicar a *transacciones* con un rango muy amplio de variación en la cantidad de *DET* en la entrada/salida y en la complejidad de procesamiento interno [Symons'91].

A partir de este método y de las críticas que se han expuesto, Charles Symons desarrolló un nuevo enfoque basándose en el mismo concepto. Este método se conoce como Análisis de Puntos Función MarkII o simplemente MKII FPA.

3.7.6. Método MarkII

Este método fue desarrollado por Symons a fines de los '80 y tiene la misma finalidad y estructura que el de Albrecht, pero supera las dificultades antes enunciadas. La organización internacional que se ocupa de mantener, actualizar y difundir la información es United Kingdom Software Metrics Association (UKSMA). El método MKII cumple con la norma ISO 14143/1 [MKII FPA CPM'98].

"Simple in concept, easy to apply, aligned with modern systems analysis methods, for intelligent software sizing and estimating" [MKII FPA CPM'98]

Este método puede usarse para medir el *tamaño funcional* de cualquier aplicación de software que se pueda describir en términos de *transacciones lógicas*, cada una comprendiendo componentes de entrada, proceso y salida. Las reglas fueron diseñadas para aplicarlas a sistemas de información de gestión, donde el componente procesamiento de cada *transacción* tiende a estar dominado por consideraciones de almacenamiento y recuperación de datos. Puede ser aplicable a otros dominios de software, pero debe notarse que las reglas no toman en cuenta las contribuciones al tamaño de algoritmos complejos como aquéllos que se encuentran en el software científico y de ingeniería, ni toman en cuenta los requerimientos de tiempo real [MKII FPA CPM'98].

En la Figura 21 se describen los componentes del Método MKII. Los cambios más significativos respecto al método descripto anteriormente están relacionados al tamaño del componente procesamiento de información. La idea es considerar al sistema como una colección de *transacciones lógicas* [Symons'91]. Para ello es necesario determinar previamente el *límite del sistema*, a fin de establecer cuales pertenecen al sistema que se medirá y cuales quedarán excluidas. La aplicación o parte de la aplicación incluida dentro del *límite* debe ser un cuerpo coherente de funcionalidad que comprende una o más *transacciones lógicas* [MKII FPA CPM'98].

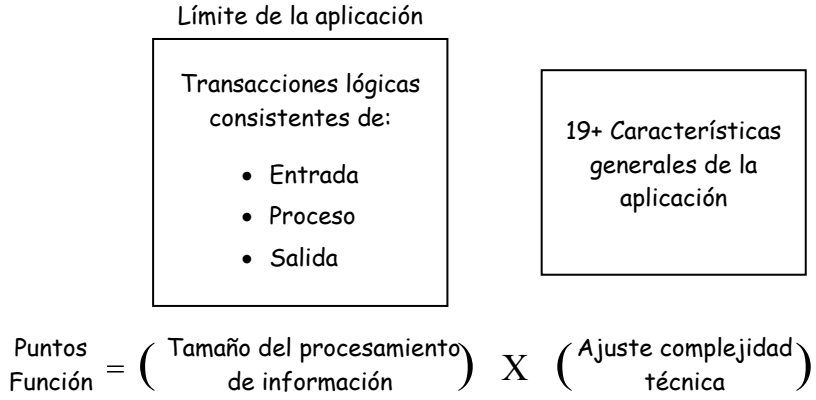


Figura 21 - Componentes del Método de Puntos Función MarkII [Symons'91]

Una *transacción lógica* se define como una combinación de tres componentes [MKII FPA CPM'98]:

- Un elemento de entrada a través del *límite*.
- Un elemento de procesamiento de los datos dentro del *límite*.
- Un elemento de salida a través del *límite*.

Tamaño funcional y transacciones lógicas

El *tamaño funcional* de una aplicación es la suma de los tamaños de cada *transacción lógica*, cuyos componentes de entrada y salida atraviesan el *límite de la aplicación*.

La *transacción lógica* es el proceso autoconsistente de más bajo nivel. Corresponde a un proceso lógico único e independiente. Es disparada por un único *evento* del mundo externo significativo para el usuario, que al completarse deja la aplicación en estado consistente con relación al *evento*. También puede ser disparada por un requerimiento del usuario para extraer información desde la aplicación sin modificar los datos almacenados [MKII FPA CPM'98]. Las *transacciones* son procesos completos. Una *transacción lógica* se cuenta sólo una vez, aún aunque pueda ser ejecutada en más de un punto en la aplicación.

Componente procesamiento en las transacciones lógicas

El tamaño de este componente es proporcional a la cantidad de entidades de datos *referenciadas* en cada *transacción*, entendiéndose como *referencia* a los procesos de creación, lectura, actualización, eliminación, listado. Se cuenta la cantidad de *tipos de entidades primarias* que son *referenciadas* por la *transacción lógica*, más la *referencia* a una entidad denominada *System* (que se describirá más adelante), si es necesario [MKII FPA CPM'98].

El concepto de *entidad* se define como: alguna cosa en el mundo real (objeto, transacción, período de tiempo, tangible o intangible) acerca de la cual se requiere mantener información. Una *entidad* es un elemento cuyas ocurrencias son identificables en forma individual o colectiva [Symons'91].

Todos los accesos a *tipos de entidades no primarias* se consideran *referencias* a una única entidad llamada *System*. Dentro del *límite de la aplicación* puede haber solo una entidad *System* y como máximo se puede incluir una *referencia* en el componente procesamiento de una *transacción lógica* [MKII FPA CPM'98].

Distinguir los tipos de entidad primaria y no primaria: la entidad System

Las *entidades primarias* son aquéllas que representan elementos principales para el sistema en estudio. Las *entidades no primarias* son aquéllas que tienen pocos atributos y generalmente tienen relación con tablas maestras que sirven para propósitos de validación. Se agrupan en una única entidad *System*. La distinción entre ambas clases de *entidades* no es absoluta, puede haber casos en el límite entre ambas [Symons'91].

Hay varios criterios para distinguir *entidades primarias* y *no primarias* [MKII FPA CPM'98] [Symons'91]:

Criterio	Entidad primaria	Entidad no primaria
Número de atributos	Varios, con valores que cambian frecuentemente	Muy pocos; los valores cambian raramente
Frecuencia de ocurrencia	Puede cambiar a menudo	Fijo permanentemente o cambia raramente
Tiempo de cambio de los valores atributos	Durante la operación normal del sistema	Usualmente fuera de la operación normal del sistema
Autorización de cambios de los valores atributos	Ninguna especial, ejecutados por el usuario normal del sistema	Ejecutados por el administrador del sistema
Ciclo de vida de la entidad	Usualmente varias etapas o estados	Usualmente sólo "vive" o no existente

Tabla 9 - Criterios para distinguir entidades primarias [Symons'91]

Componentes de entrada y salida en las transacciones lógicas

Este método asume que el tamaño relativo de los componentes entrada y salida en cada *transacción lógica* es proporcional a la cantidad de *DET* en cada componente respectivo. Un *DET* es un ítem único de información que es indivisible para el propósito de la *transacción lógica* y es parte de un flujo de datos en la *entrada* y *salida* a través del *límite* [MKII FPA CPM'98].

Pasos para calcular el Índice de Puntos Función

Para evaluar el *Índice de Puntos Función (FPI)*, se deben seguir los siguientes pasos [Symons'91] [MKII FPA CPM'98]:

1. *Obtener una comprensión general del sistema al cual se aplicará el método*. Para lograr el conocimiento del problema y del sistema necesario para cumplir con los requerimientos del usuario, se realiza la Especificación de Requerimientos.

2. *Definir el punto de vista, propósito de la medición y límite del sistema* que se medirá, para establecer qué elementos estarán incluidos dentro del mismo.
3. *Determinar las entidades primarias y no primarias.* Es necesario distinguir las *entidades primarias* y *no primarias* agrupadas en la entidad *System*.
4. *Determinar las transacciones lógicas*.

   Siempre hay que tener presente: *¿Qué requiere el usuario?*

   Hay algunas reglas que permiten diferenciar algunos procesos para determinar si se consideran o no *transacciones* [Symons'91].

   - Los procesos internos que se repiten con frecuencia deben ser contados en cada *transacción* que ocurren. No deben contarse como *transacciones lógicas* separadas, a menos que sean disparados en forma independiente por un *evento* significativo para el usuario.
   - Los cambios en el valor de un *DET* de entrada no pueden disparar *transacciones lógicas* separadas.
   - Los *eventos* que son significativos para el diseñador físico del sistema no se cuentan como *eventos* que disparan *transacciones lógicas*.
5. *Determinar los componentes de las transacciones lógicas*. Cada una de ellas consta de tres componentes: entrada, procesamiento y salida [MKII FPA CPM'98]:

   5.1. El componente entrada a través del *límite* consiste de la adquisición y validación de los datos ingresados que describen un *evento* de interés en el mundo externo o de los parámetros de una *consulta* para obtener información de salida desde la aplicación.

   5.2. El componente procesamiento consiste del almacenamiento y recuperación de información que describe el estado de las entidades de interés en el mundo externo.

   5.3. El componente salida a través del *límite* consiste del formateo y presentación de información hacia el mundo externo.

   Para organizar el trabajo se utiliza una tabla de *transacciones lógicas* con el siguiente encabezado:

ID	Nombre transacción	Evento o consulta	N° de DET entrada	Respuesta	N° de DET salida	Tipo de entidad referenciada	N° de ER[5]	MKII FP

6. *Aplicar la fórmula para calcular FPI.*

   El método MKII utiliza la siguiente fórmula para determinar el *Índice de Puntos Función* o *tamaño funcional* [MKII FPA CPM'98]:

$$FPI = W_I * \Sigma N_I + W_E * \Sigma N_E + W_O * \Sigma N_O$$

   donde W_I, W_E y W_O representan los *pesos promedio en la industria (industry average weights)* para los componentes *entrada* (I), *entidades referenciadas* (E) y

[5] ER: entidad referenciada

de *salida* (O). Sus valores son: $W_I = 0.58$, $W_E = 1.66$ y $W_O = 0.26$ y

N_I: cantidad de *DET* de entrada
N_E: cantidad de *entidades referenciadas*
N_O: cantidad de *DET* de salida

Los *pesos promedio en la industria* suman 2.5 para mantener la correspondencia con los FP de Albrecht. En promedio los métodos dan aproximadamente los mismos tamaños hasta cerca de los 400 FP. Para sistemas más grandes, MKII tiende a producir tamaños mayores que el método de Albrecht [Grant Rule'01[c]].

Hay algunos casos en que es aconsejable obtener los pesos mediante un proceso de calibración. En general, si el objetivo es comparar productividad a través de diferentes entornos, se recomienda usar los *pesos promedio en la industria*. Sin embargo, para estimar dentro de un entorno específico, lo adecuado es obtener los pesos por calibración de proyectos dentro de ese mismo entorno [Symons'91].

7. *Opcional*. En caso de necesitarse calcular los FP considerando los requerimientos técnicos, se debe calcular el *TCA*.

Para el cálculo del factor *TCA*, se utiliza una lista de características generales de la aplicación que se basa en la que usa Albrecht, pero se extiende a 19 características o más si realmente se justifica. Las características adicionales de MKII son:

Características Generales del Sistema			
15	Requerimientos de otras aplicaciones	18	Uso directo por terceras partes
16	Seguridad, privacidad, auditabilidad	19	Documentación
17	Necesidad de entrenamiento al usuario	20	Características definidas por el usuario

Tabla 10 - Características Generales del Sistema adicionales para MKII [Symons'91]

$$TCA = (TDI * C) + 0.65 \text{ [MKII FPA CPM'98]}$$

donde el valor actual promedio en la industria de *C* es 0.005 y *TDI* es el total de los puntajes para cada una de las 19 características, llamado *grado de influencia total*. Los grados de influencia al igual que en el método de Albrecht varían en una escala de cero a cinco, desde ninguna a fuerte influencia [Symons'91].

Luego, el *Índice de Puntos Función Ajustado*, se expresa como:

$$AFPI = FPI * TCA \text{ [MKII FPA CPM'98]}$$

Al comienzo del desarrollo de este punto se ha establecido que es opcional. El *Índice de Puntos Función* mide el *tamaño funcional* de la aplicación desde la visión del usuario, pero el FPA también ofrece un medio para tener en cuenta la complejidad técnica y ciertos requerimientos de calidad, conocidos como requerimientos no funcionales [MKII FPA CPM'98]. Mediante este factor se intenta considerar esos requerimientos. Sin embargo, no hay un acuerdo general acerca de su validez. De hecho en las nuevas versiones del método se considera

que el TCA no contribuye al *tamaño funcional*. La razón es que hay estudios estadísticos que sugieren que este factor no representa en forma adecuada la influencia sobre el tamaño de las características que intenta considerar. En general se podría ignorar cuando se trabaja en un único entorno técnico. Cuando se necesitan comparaciones de tamaño entre aplicaciones construidas con técnicas y requerimientos de calidad muy diferentes puede ser preferible usar un valor de ajuste calculado localmente [MKII FPA CPM'98]. Debe notarse que el valor propuesto para el factor *C* en el cálculo del *TCA* es la mitad del propuesto por Albrecht. La interpretación es que esos factores técnicos actualmente se obtienen con menor esfuerzo que muchos años atrás. Se podría esperar que si esa tendencia continúa, a largo plazo solamente se deberá pensar en los requerimientos de procesamiento de información del problema [Symons'91]. El *TCA* no está incluido dentro de la norma ISO/IEC 14143 y generalmente no se recomienda su uso [MKII FPA CPM'98].

3.7.7. Comparación de los métodos de Albrecht y MarkII [Grant Rule'01c]

Este análisis apunta a mostrar las similaridades y diferencias entre las dos variantes del FPA estudiadas. Las técnicas de FPA intentan cuantificar los requerimientos funcionales y no-funcionales. En ambas técnicas esto se logra mediante la determinación inicial del *tamaño funcional* y la posterior aplicación de un factor de ajuste al tamaño, para atender el esfuerzo adicional necesario para cumplir los requerimientos de calidad. Las principales diferencias entre IFPUG (Albrecht) y MKII FPA se encuentran en la forma en que se determina el *tamaño funcional*.

En cualquier técnica de medición del *tamaño funcional* se miden los requerimientos funcionales de una aplicación en términos de uno o más tipos de componentes lógicos, cada uno de los cuales debe tener una relación explícita con el *límite de la aplicación*. Además, se proveen reglas para identificar los tipos de componentes lógicos del software y asignarles un valor numérico que representa su contribución al *tamaño funcional*. Después de identificar los componentes, se determina el *tamaño funcional* de la aplicación mediante la suma del tamaño de cada uno de los tipos de componente lógico.

MKII y IFPUG expresan los requerimientos funcionales en términos de componentes lógicos. MKII usa un único tipo de componente lógico y expresa todos los requerimientos funcionales como *transacciones lógicas*. IFPUG usa cinco tipos de componentes lógicos: *EI*, *EO*, *EQ*, *ILF* y *EIF*.

Los tipos de componentes lógicos están formados por varias partes. En MKII cada *transacción lógica* se compone de entrada, procesamiento y salida. En IFPUG, *EI y EO* se componen de mensajes de entrada y salida, respectivamente, *EQ* por un par entrada/salida, *ILF* y *EIF* por datos persistentes mantenidos por la aplicación u otra aplicación, respectivamente.

En MKII, cada *tipo de entidad* se trata como independiente y se cuentan las *referencias* a las *entidades* por cada *transacción lógica*. En IFPUG, los *tipos de entidad* se agrupan para formar *ILF* si están dentro del *límite de la aplicación* o *EIF* si

están fuera del *límite*. Las *referencias* a los *tipos de entidad* se cuentan como *FTR* por cada *EI*, *EO* o *EQ*.

En ambas técnicas hay una relación entre los tipos de componentes lógicos y el *límite de la aplicación*.

En MKII, por cada *transacción lógica*, el mensaje de entrada y de salida atraviesa el *límite* cuando llega y sale de la aplicación respectivamente y la parte de proceso está completamente retenida dentro del *límite*. En IFPUG, cada *EI* y *EO* atraviesa el *límite* cuando llega y sale de la aplicación, por cada *EQ* la parte de entrada y salida atraviesa el *límite* cuando llega y sale y los *ILF* y *EIF* están completamente retenidos dentro del *límite de la aplicación* que se analiza o de otra aplicación, respectivamente.

Ambas técnicas basan sus reglas para evaluar el tamaño de sus componentes lógicos en las *cuentas base*. En ambas se cuenta la cantidad de *DET* en las partes del mensaje del componente lógico y la cantidad de *referencias* a datos persistentes dentro de la aplicación. Aunque usan diferente terminología, las definiciones son similares.

Mientras en MKII las cuentas resultan de las entradas y salidas de las *transacciones* y de las *referencias* a los datos persistentes, en IFPUG se identifican *ILF* y *EIF* y se cuentan sus partes constituyentes que contribuyen a la funcionalidad.

En MKII se cuenta una *referencia* a una *entidad* por cada *tipo de entidad* accedida durante el curso de una *transacción lógica*, mientras que en IFPUG se cuenta una *FTR* por cada *ILF* o *EIF* accedido durante el curso de una entrada, salida o consulta.

Los objetos de la especificación que se deben identificar son los mismos para ambas técnicas: *mensajes de entrada* y *salida, mensajes de error* y *tipos de entidad*. Además en IFPUG se consideran los *DET* almacenados en los *tipos de entidad*.

Las principales diferencias entre las dos técnicas surgen desde *cómo* se construyen las *cuentas base* y no desde *qué* es lo que se cuenta.

Para determinar las *cuentas base*, se identifican los respectivos componentes lógicos y se cuenta la cantidad de *DET* en los mensajes asociados. Los *mensajes de error* se tratan de manera diferente en las dos técnicas. En IFPUG los atributos de un *mensaje de error* se tratan como atributos de un mensaje de entrada, en MKII se tratan como atributos adicionales de un mensaje de salida. A continuación se cuenta la cantidad de accesos a datos persistentes. En IFPUG, además, se deben ejecutar pasos adicionales para contar la cantidad de *tipos de entidad* en los grupos de *tipos de entidad* que forman cada *ILF* y *EIF* y para contar la cantidad de *DET* almacenados en los *tipos de entidad*.

Hay algunas otras variaciones. MKII usa el concepto de entidad *System* para agrupar las *entidades no primarias* que pueden contener información dependiente de la implementación. IFPUG no cuenta datos dependientes de la implementación.

En MKII las *cuentas base* se usan directamente en los cálculos del *tamaño funcional* expresado en FP sin ajustar. En IFPUG las *cuentas base* se usan para determinar la magnitud de cada componente lógico. Usando las tablas provistas por el

método se valora la magnitud de cada componente como *bajo*, *medio* o *alto*, en base a los valores de las *cuentas base* de *DET* y *referencias* a *tipos de entidad* o la cantidad de *RET* en el caso de *ILF* y *EIF*. IFPUG usa el término *complejidad* en vez de magnitud. No debe confundirse con la *complejidad técnica*.

Los mensajes de *entrada*, *salida* y *referencias* a datos persistentes tienen su propia contribución a una aplicación, pero la contribución de cada una es de diferente clase. Para combinarlas y derivar un único valor numérico del *Índice de tamaño funcional*, se deben normalizar las *cuentas base* para usar una única unidad. Esto se logra a través de los *pesos relativos*.

En MKII, se usan tres pesos: *Wi*, *Wo* y *We*. En IFPUG, el sistema de pesos es más complicado, debido a la gran cantidad de tipos de componentes lógicos. Cada tipo de componente lógico tiene asignado un peso que depende del tipo de componente y su magnitud.

En ambas técnicas el *Índice de tamaño funcional* de la aplicación expresado en FP se obtiene a partir de la suma de las cuentas afectadas por los pesos.

Los dos métodos usan enfoques similares para considerar los requerimientos técnicos. Se evalúa una lista de características no-funcionales en una escala de cero a cinco. IFPUG usa 14 características; MKII usa las mismas 14, pero agrega otras cinco o más.

Una vez evaluado el *grado de influencia* de las características, se suman para dar el *grado de influencia total*. Este valor se usa para ajustar el *Índice de tamaño funcional* y obtener un nuevo valor que representa el *tamaño funcional* y los requerimientos técnicos combinados.

Capítulo 4

Aplicación del Análisis de Puntos Función a LEL y Escenarios

Este capítulo está organizado de la siguiente manera. En la sección 4.1 se presenta la propuesta para aplicar FPA a L&E. En la sección 4.2 se analizan los beneficios de la aplicación de este enfoque. En la sección 4.3 se presentan las distintas alternativas para el cálculo de FP desde L&E y se selecciona una de ellas. En la sección 4.4 se describe el modelo de medición del *tamaño funcional* en el marco de L&E. En la sección 4.5 se evalúa cuál de los métodos estudiados - Albrecht y MKII - se adapta mejor al contexto de L&E y en la sección 4.6 se analizan algunos aspectos referidos al Ajuste de complejidad técnica.

4.1. Introducción

Si bien en la bibliografía se ha propuesto la aplicación de los métodos de FPA a partir de la especificación de requerimientos [Symons'91], la propuesta de esta tesis es aplicarlos en una fase anterior, previa a la especificación. La idea es calcular la funcionalidad a partir del LEL y los Escenarios.

Los Puntos Función (FP) miden el tamaño del software cuantificando la funcionalidad provista al usuario basándose solamente en el diseño lógico y las especificaciones funcionales [IFPUG'00ª].

Los escenarios se usan para entender la aplicación y su funcionalidad: cada escenario describe una situación específica de la aplicación centrando la atención en su comportamiento [Leite'00].

Esto significa que L&E se puede utilizar como documentación base para el cálculo de FP.

4.2. Beneficios de la aplicación del enfoque

Antes de analizar los beneficios se debe tener presente que cuando un cliente requiere una nueva aplicación de software - o el reemplazo de una existente - desarrollada por un proveedor, tanto el cliente como el desarrollador necesitan una estimación del costo de desarrollo: el cliente para asegurar la óptima decisión de su inversión según la relación costo/beneficio y el desarrollador para evaluar las necesidades inherentes al desarrollo. Para determinar el costo, el proveedor tiene que estimar el esfuerzo de desarrollo, el personal necesario y por lo tanto los costos resultantes, comenzando solamente desde el tamaño de los requerimientos [Grant Rule'01ª]. Justamente uno de los dilemas que se les presenta a los directores de proyectos de software al comienzo del desarrollo es que se le requieren estimaciones cuantitativas "ahora", pero en esa etapa no disponen de información sólida [Pressman'93]. Han sido ellos los principales promotores del uso de medidas del software que les permitan hacer estimaciones en las primeras fases del ciclo de vida

del proyecto [Fenton'95]. Es evidente que la posibilidad de disponer cuanto antes de esas medidas favorecerá la realización de estimaciones.

En el Capítulo 3 sección 3.2.1 se describen los beneficios de la aplicación de métricas y en 3.7.4 los beneficios de aplicar FPA. En esta sección se destacan algunos beneficios adicionales que se pueden obtener a partir del enfoque propuesto.

La ventaja decisiva de este enfoque consiste en la disponibilidad de una estimación del tamaño del producto a construir en una etapa muy temprana del proyecto de construcción. Esta disponibilidad temprana permite anticipar una amplia gama de decisiones con las consiguientes ventajas para la gestión del proyecto de desarrollo. Más específicamente, los beneficios del enfoque son:

- Obtener una medición del tamaño del sistema en la etapa inicial del desarrollo, no requiriendo esperar hasta la especificación de los requerimientos. Esto hace que a la ventaja de tener una medida de tamaño suministrada por el FPA se suma la posibilidad de tenerla antes.
- Utilizar esta medida para la estimación de costos, esfuerzo, duración. En consecuencia permite tomar decisiones vinculadas con esas estimaciones de un proyecto antes de avanzar en su desarrollo. Esperar hasta que la especificación esté completa significa invertir tiempo y esfuerzo en un proyecto que quizás no podrá concretarse.
- Usar la estimación para favorecer la toma de decisiones acerca de desarrollar o comprar un producto de software. En ese caso se ahorraría el esfuerzo de avanzar hasta obtener una especificación cuya única finalidad sería evaluar la conveniencia de una u otra opción. Esto es especialmente ventajoso en el caso de evaluarse el reemplazo de un sistema en funcionamiento, pues en ese caso se dispone de un conocimiento elevado de la funcionalidad del mismo.
- Ampliar el conocimiento acerca del sistema con una medida de su funcionalidad, posibilitando una gestión temprana de las métricas del proyecto.
- Producir una representación comprensible para desarrolladores y usuarios complementaria de la visión de L&E. La comparación de métricas de tamaño con otros proyectos ayuda al desarrollador a disponer de elementos de evaluación de su proyecto más finos.

4.3. Análisis de las alternativas para el cálculo de FP desde L&E y fundamentación de la elección

La idea central de esta propuesta es desarrollar una métrica de Puntos Función a partir de L&E. Comparte con los métodos de FPA los objetivos de medir la funcionalidad de un sistema y la factibilidad de aplicación a los productos obtenidos a lo largo del ciclo de vida del desarrollo [Fenton'96]. La diferencia radica en que se propone medir L&E, producto resultante de la elicitación, etapa previa a la Especificación de Requerimientos.

El FPA considera al sistema como un conjunto de *transacciones lógicas* que interactúan con *entidades* y/o archivos lógicos. Para obtener la funcionalidad aplica un conjunto de reglas a las *entidades* y *transacciones lógicas* obtenidas a partir del diseño del sistema. Los requerimientos funcionales usualmente son modelados en los documentos de Especificación de Requerimientos y diseño lógico del sistema. Ellos son una expresión de *qué* debe hacer la aplicación en vez de *cómo* lo hace y por lo tanto son buenos documentos fuente para el propósito de contar los FP [MKII FPA CPM'98]. Los métodos de FPA no establecen cómo obtener esos requerimientos.

L&E son representaciones auxiliares que ayudan a establecer no sólo los requerimientos funcionales sino también los requerimientos no funcionales[6] [Hadad'96].

Para el desarrollo de la propuesta se analizan las siguientes alternativas para obtener los FP desde L&E. En base a la afirmación del párrafo previo, una posibilidad es intentar obtener los requerimientos desde L&E y desde ellos definir el diseño. Otra alternativa es evitar el paso de obtención de requerimientos y definir el diseño desde L&E. Otro camino posible es obviar los pasos de obtención de requerimientos y diseño. Esto significa tener la certeza que L&E contiene estrictamente lo que requiere la aplicación. En todos los casos, sea el diseño o L&E, se descomponen en *entidades* y *transacciones lógicas* y luego se aplica el método de la manera convencional. Por último, la otra alternativa es derivar los FP directamente desde L&E.

En la Figura 22 se representan las alternativas potenciales:

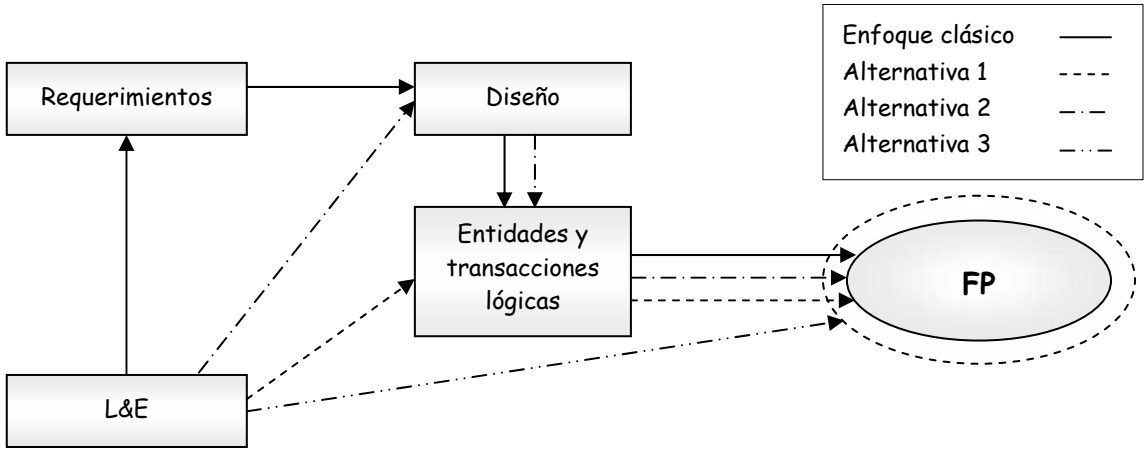


Figura 22 - Posibles enfoques de aplicación de FPA a L&E

Resumen de las alternativas:

1. El enfoque *clásico* deriva los FP desde el diseño. Se acepta que a partir de L&E se pueden construir los requerimientos. La afirmación anterior significa que a partir de L&E habrá una forma de calcular los FP.
2. Desde el L&E se derivan las *entidades* y *transacciones lógicas* y a partir de las

[6] La incorporación de requerimientos no funcionales se logra mediante la extensión al LEL propuesta por Cysneiros, ver sección 4.6.

mismas se aplica el método de FPA. (Alternativa 1)

3. Desde el L&E se obtiene el diseño, desde allí se derivan las *entidades* y *transacciones lógicas* y a partir de estas últimas se aplica el método de FPA. (Alternativa 2)
4. Obtener directamente los FP a partir de L&E. (Alternativa 3)

El primer paso será optar por alguna de las alternativas planteadas.

El enfoque clásico no resulta un aporte novedoso relacionado con el objetivo de este trabajo, pues una vez obtenido el diseño se deben seguir los pasos propuestos por los métodos de FPA.

La Alternativa 1 equivale a omitir el diseño y la Alternativa 2 equivale a construir un sistema sin considerar los requerimientos. Sin embargo, debe tenerse presente que: el desarrollo de software es un proceso que tiene como propósito el desarrollo de un sistema de software completo. La determinación de los requerimientos es un subproceso dentro del proceso. Otros procesos dentro del proceso de desarrollo del software incluyen diseño, codificación y prueba. Cada uno de estos procesos tiene un único propósito que contribuye al propósito general de desarrollar software [Loucopoulos'95]. Tal como se menciona en la sección 2.1, los diferentes modelos de desarrollo de software reconocen la existencia de los componentes especificación, diseño e implementación (ver Figura 1) [Loucopoulos'95]. En este contexto ambas alternativas parecen poco posibles.

Las Tablas 11 y 12 muestran un resumen de las alternativas desde el punto de vista de los productos requeridos y de los procesos involucrados para el desarrollo del cálculo de los FP.

Alternativa	Artefactos requeridos				Cálculo FP
	L&E	Requerimientos	Diseño	Entidades y transacciones	
Enfoque clásico	x	x	x	x	x
Alternativa 1	x			x	x
Alternativa 2	x		x	x	x
Alternativa 3	x				x

Tabla 11 – Resumen de alternativas – artefactos

Alternativa	Proceso
Enfoque clásico	• Desde L&E se derivan los requerimientos. • Desde los requerimientos se desarrolla el diseño del sistema. • A partir del diseño se determinan las entidades y transacciones lógicas. • Se calculan los FP a partir de las entidades y transacciones lógicas.
Alternativa 1	• Desde L&E se derivan las entidades y transacciones lógicas. • Se calculan los FP a partir de las entidades y transacciones lógicas.
Alternativa 2	• Desde L&E se deriva el diseño del sistema. • Desde el diseño se determinan las entidades y transacciones lógicas. • Se calculan los FP a partir de las entidades y transacciones lógicas.
Alternativa 3	• Se calculan los FP a partir de L&E.

Tabla 12 – Resumen de alternativas - proceso

En esta tesis se considerará la Alternativa 3. Se tratará de calcular los FP basándose en el L&E. Este enfoque para medir la funcionalidad supone establecer un modelo de cálculo de FP, analizar las relaciones entre los elementos del L&E y los conceptos del FPA, enunciar las reglas y fórmulas para calcular los FP.

Debe advertirse que tomando como base del análisis a L&E se está considerando todo el sistema, no exclusivamente las funciones asignadas al software y por lo tanto esto llevará a obtener una cantidad de FP presumiblemente mayor que la que se obtendría a partir del enfoque tradicional. ¿Aun así, cómo se puede sostener la validez de la propuesta? Desde la fundamentación del análisis de L&E surge que un escenario comienza en el macrosistema, para el cual el software es un componente y continúa evolucionando a medida que el "artefacto" de software se va construyendo [Hadad'96]. Este es el punto de apoyo, se considera que la cantidad de FP obtenida inicialmente se puede ir refinando a medida que evolucionan los escenarios, para tender a equiparase al cálculo que resultaría de la aplicación del enfoque clásico. También se puede avalar esta afirmación basándose en [Fenton'96]: al referirse al concepto de funcionalidad y examinar tres enfoques para medirla (DeMarco, COCOMO y Albrecht), indica que los tres miden la funcionalidad desde los documentos de especificación, pero también se pueden aplicar a productos posteriores en el ciclo de vida para refinar las estimaciones de tamaño y por lo tanto, las estimaciones de costo o productividad. Una visión intuitiva de la funcionalidad dice que si un programa P es una implementación de la especificación S entonces P y S deberían tener la misma funcionalidad. Al mencionar los problemas que surgen al cambiar los requerimientos, menciona que los FP son una medición de tamaño atractiva pues pueden recalcularse a medida que continúa el desarrollo. Si se compara el cálculo de FP generado desde la especificación inicial con el obtenido desde el sistema resultante a veces se encuentra un incremento importante. La diferencia puede deberse a nueva funcionalidad incluida cuando progresa el desarrollo o al menor nivel de detalle de la especificación respecto a la implementación real.

Derivar los FP directamente desde L&E representa un aporte novedoso y complementario del FPA. Un aspecto que vale destacar es que debería analizarse la factibilidad de comparar los resultados con los obtenidos por los métodos convencionales de FPA, cuestión que escapa al alcance de esta tesis.

4.4. Modelo de medición del tamaño funcional desde L&E

Partiendo del concepto que un método de medición del *tamaño funcional* define una medida que asigna un número de FP a las aplicaciones de software y considerando que el objeto de la medición es L&E, la propuesta es definir un enfoque que permita identificar dentro del mismo aquellas partes que representan funciones y asociarlas con números para obtener el *tamaño funcional* de la aplicación. Este proceso requiere dos pasos de abstracción.

La primera meta en la definición del enfoque de cálculo de los FP sobre L&E requiere establecer una abstracción del sistema que representa los ítems que contribuyen al *tamaño funcional*. Pueden definirse dos pasos:

1. Identificar los componentes de L&E que contribuyen a la funcionalidad.
2. Asociar con números a cada uno de los componentes identificados en L&E.

El primer paso de la abstracción se aplica a L&E. El resultado es una representación que contiene los elementos relevantes para el *tamaño funcional*.

El segundo paso es asociar estos elementos con números. La documentación fuente para este paso son los componentes identificados en el primer paso.

Considerando los dos pasos de la abstracción mencionados en los párrafos anteriores, se puede describir el modelo del proceso en los siguientes términos: en la documentación de L&E se identifican los componentes que contribuyen a la funcionalidad, los escenarios en nuestro caso. Éstos a su vez pueden descomponerse en un conjunto de *episodios*. El objetivo es analizar los *episodios* e identificar dentro de ellos los ítems de funcionalidad de menor nivel. El paso siguiente será asociarlos con números para determinar el número de FP del sistema.

En la Figura 23 se ilustra el modelo de FPA sobre L&E.

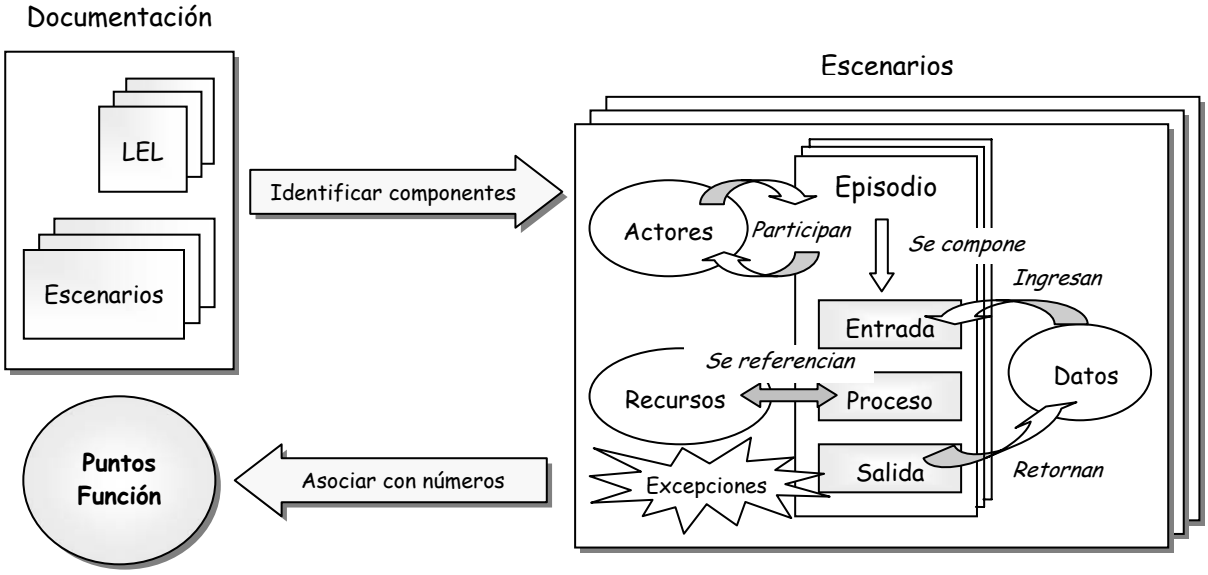


Figura 23 - Modelo de FPA sobre L&E

4.5. ¿Qué método de FPA elegir?

Para avalar la propuesta de utilización de los métodos de FPA sobre L&E pueden mencionarse los casos de aplicación sobre el modelo de *Casos de uso*. Como se mencionó en el Capítulo 1, el trabajo "Mapping the OO-Jacobson Approach into Function Point Analysis" de Thomas Fetcke, Alain Abran y Tho-Hau Nguyen propone el mapeo del método OOSE basado en *Casos de uso* de Jacobson con el modelo abstracto de Puntos Función y establece en los objetivos la utilización del estándar IFPUG FPA [Fetcke'98]. En [Longstreet'01] también se propone usar IFPUG FPA sobre *Casos de uso*.

En el Capítulo 3 secciones 3.7.5 y 3.7.6 se han incorporado los fundamentos de los métodos de Albrecht y MKII. En 3.7.7 se presenta una comparación entre ambos

métodos. Una cuestión importante es determinar cuál de los métodos estudiados presenta un marco conceptual más adecuado para facilitar la identificación de los componentes que aportan funcionalidad. Ambos métodos presentan similitudes y diferencias para abordar el proceso de medición de la funcionalidad.

En la sección 3.7.5, bajo el título "Algunas críticas al método de Albrecht" se han expuesto algunas debilidades del método de Albrecht que el método MKII trata de superar. A continuación se analizan algunas de ellas para determinar su influencia en la aplicación a L&E:

1. Con respecto a la facilidad para identificar los componentes *entrada*, *salida* y *consultas*, la aplicación de cualquiera de los métodos resulta de la misma complejidad.
2. Respecto a los componentes *archivos lógicos internos* y *archivos de interfase externa* de Albrecht, se dificulta la identificación debido al escaso nivel de detalle que se obtiene en la documentación de L&E.
3. La valoración de la complejidad de los diferentes tipos de componentes de Albrecht, en *baja*, *media* o *alta* es difícil establecerla a partir de L&E, en cambio MKII asume que el tamaño relativo de los componentes *entrada* y *salida* es proporcional a la cantidad de *DET* y el componente *proceso* es proporcional a las *referencias* [7] a *entidades*. Esa información se puede obtener de manera relativamente sencilla a partir de los *episodios* de los escenarios.
4. La complejidad de procesamiento para los componentes de tipo archivo en el método de Albrecht se evalúa usando la misma valoración *baja*, *media* o *alta*, según la cantidad de *archivos referenciados*. Para tratar con la misma cuestión, MKII cuenta cada una de las *entidades referenciadas* en una *transacción*. Como en el punto anterior, para el caso de L&E una vez determinados los *episodios* resulta más sencillo analizar su parte de procesamiento y contar las *referencias* a *recursos*.
5. Con respecto a las características generales de la aplicación y su influencia sobre el *Ajuste de complejidad técnica*, si bien el método MKII propone algunas mejoras frente al método de Albrecht, en esta propuesta no se va a considerar; en la sección 4.6 se exponen los fundamentos que sustentan la decisión.

[7] Referencia indica acciones de creación, lectura, actualización, borrado, listado

	Componente	Albrecht	MKII
Facilidad para identificar componentes	Transacciones	Igual complejidad	
	Entidades	Difícil para identificar ILF y EIF	---
Valoración de la complejidad de los componentes	Transacciones	Difícil determinar valoración baja, media, alta para EI, EO, EQ	Proporcional a la cantidad de DET en componentes de entrada y salida y a la cantidad de entidades referenciadas
	Entidades	Difícil determinar valoración baja, media, alta para ILF y EIF	---
Complejidad de procesamiento	Proceso	Difícil determinar valoración baja, media, alta según cantidad de ILF y EIF referenciados	Contabilizar la cantidad de entidades referenciadas
Ajuste de complejidad técnica	No tiene influencia en ninguno de los métodos		

Tabla 13 – Cuadro comparativo de los métodos Albrecht y MKII

En favor de MKII se puede mencionar que en [Grant Rule'01[b]] se sugiere la incorporación del concepto de *transacción lógica* de MKII en la descripción de *Casos de uso* como solución para contrapesar algunas deficiencias (falta de rigor en la aplicación de la técnica, falta de un nivel de granularidad consistente, mayor preocupación por las oportunidades de reuso que con el análisis del problema y la descripción de los requerimientos, optimización demasiado temprana de la solución sin considerar otras opciones de solución). Grant Rule en el artículo "Scenarios, Use Cases & MKII FPA" expresa: la definición de *transacción lógica* de MKII parece muy similar al concepto de *Caso de uso* en el nivel de granularidad de detalles. La integración de esos dos distintos enfoques permite derivar mediciones de tamaño desde los desarrollos Orientados a Objetos basados en *Casos de uso* fácilmente y con bajo costo [Grant Rule'98].

En resumen, según lo expuesto se puede pensar que MKII presenta un marco conceptual que será más sencillo para abordar a partir de L&E. Parece más natural descomponer el modelo L&E en "partes" que se asemejan a las *entidades* y *transacciones lógicas* de MKII y definir un conjunto de reglas para su tratamiento y cálculo de la funcionalidad. La propuesta es desarrollar un enfoque similar a MKII, basándose exclusivamente en la información disponible en L&E, que brinde una serie de reglas para identificar, clasificar y cuantificar los diferentes tipos de componentes lógicos del L&E para obtener una medida del *tamaño funcional* del sistema.

4.6. Consideraciones respecto al Ajuste de complejidad técnica

Además de las razones expuestas en el Capítulo 3 sección 3.7.6 -Pasos para calcular el Índice de Puntos Función en el Método MKII-, otra consideración importante tiene relación particular con el enfoque propuesto: la documentación de L&E contiene información relativa al dominio de aplicación, no es posible identificar allí características técnicas que tengan relación con aspectos vinculados a la implementación. Tal como se menciona en el Capítulo 5 sección 5.3.2 al tratar con las restricciones, Cysneiros [Cysneiros'01] propone una extensión al L&E para favorecer el proceso de elicitación de requerimientos no funcionales. La utilización de ese enfoque permitiría contar con alguna información vinculada a las características generales del sistema que se utilizan para la determinación del factor de *Ajuste de complejidad técnica*, como, por ejemplo, requerimientos de *performance* provenientes de una exigencia de reducido tiempo de respuesta o una *frecuencia de transacción* establecida a partir de la necesidad de ejecutar las transacciones diariamente, pero habrá otras características que será muy difícil o aún imposible ponderarlas a partir de L&E, por ejemplo, la *facilidad de instalación* o la capacidad de soportar *múltiples instalaciones* en distintas organizaciones.

Con referencia al tratamiento de los requerimientos técnicos y de calidad Symons dice: el enfoque de Albrecht que considera los requerimientos técnicos y de calidad mediante un Factor de Ajuste que multiplica el tamaño obtenido desde los requerimientos funcionales puros, no es la mejor manera de tratar con esos requerimientos. Es como decir que se estimará el tamaño de una casa en metros cuadrados usando una escala que varía desde *No presente* a *Impacto significativo* para una serie de factores tales como si el lugar de la construcción es de fácil acceso o no, si los materiales son escasos, etc. Esos factores claramente necesitan tenerse en cuenta cuando se estima el esfuerzo para construir la casa, pero ellos no alterarán su tamaño. El *tamaño funcional* debería basarse puramente en los requerimientos funcionales del usuario. Los requerimientos técnicos y de calidad no deberían combinarse con los requerimientos funcionales del usuario para producir un tamaño compuesto. El resultado es muy difícil de interpretar y los factores inevitablemente dependen de suposiciones acerca de la tecnología a usar [Symons'01].

En el contexto de MKII, "requerimientos de procesamiento de información" significa el conjunto de funciones requeridas por el usuario del producto de software de aplicación (excluyendo requerimientos técnicos y de calidad) o dicho de otra forma: una medida del *tamaño funcional* MKII debería ser verdaderamente independiente de la tecnología o métodos usados para desarrollar o implementar el software [MKII FPA CPM'98].

Es por ello que las nuevas versiones del estándar del método MKII, para cumplir con la norma ISO 14143/1, consideran que el *Ajuste de complejidad técnica* no contribuye al "*tamaño funcional*" [MKII FPA CPM'98]. La norma ISO 14143-1 indica que los requerimientos funcionales del usuario excluyen las características de calidad como se describen en ISO 9126 y las características técnicas. Ejemplos de requerimientos técnicos o de implementación son: restricciones de performance,

especificaciones de confiabilidad en tiempo de ejecución y tecnologías de interfase de usuario [ISO/IEC'95].

En consonancia con lo anterior cabe mencionar que en "Mapping the OO-Jacobson Approach into Function Point Analysis" [Fetcke'98] como en el caso de aplicación "The Warehouse Software Portfolio: A Case Study in Functional Size Measurement" [Fetcke'99] no se consideran las características generales del sistema.

Las razones enunciadas en los párrafos previos sirven como soporte para la decisión de basar la medición exclusivamente en las funciones del sistema obtenidas desde L&E.

Capítulo 5

Medición de funcionalidad de L&E

Este capítulo esta organizado de la siguiente manera. En la sección 5.1 se presenta la estrategia a desarrollar para la definición de conceptos y reglas. En la sección 5.2 se describe la propuesta basada en el *enfoque de reducción a episodios*. Éste consiste en obtener el conjunto de escenarios desde la documentación de L&E y descomponerlos en un conjunto de *episodios* no repetidos. Cada uno de esos *episodios* se toma como unidad para la medición de la funcionalidad. En la sección 5.3 se estudian las relaciones entre los conceptos del L&E y del método de FPA y se establecen un conjunto de reglas para aplicar a los *episodios* y sus componentes. En la sección 5.4 se define el *límite del sistema*, concepto esencial para medir la funcionalidad ya que permite determinar qué partes del sistema deben considerarse en la medición.

5.1. Introducción

En este capítulo se propone estudiar el L&E teniendo como marco de referencia el método MKII. En la estrategia se plantea la búsqueda de los conceptos de L&E vinculados con los conceptos de MKII, el análisis de las similitudes y diferencias y la definición de nuevos conceptos y reglas que resultan de la adaptación al contexto.

Durante la exposición se hace un uso intensivo de los términos *acción* y *evento*, por lo tanto a continuación se incluye la definición de ambos términos:

Para el *Diccionario Kapelusz de la lengua española* [Kapelusz'79]:
Acción: cosa que se lleva a cabo [sinónimo: hecho]. Efecto producido por la actividad de una cosa en otra.
Evento: suceso, acontecimiento. Hecho imprevisto o que puede llegar a suceder.

Por su parte el *Oxford Advanced Learners Dictionary* [Hornby'95]:
Action: the process of doing something; activity; a thing done; a deed; an act.
Event: a thing that happens, especially something important.

Mc.Menamin [Mc.Menamin'84] asigna los nombres genéricos *evento* y *respuesta* a las interacciones entre el sistema y su entorno. Un *evento* es algún cambio en el entorno del sistema y una *respuesta* es el conjunto de acciones ejecutadas por el sistema siempre que ocurre un cierto evento.

En el contexto de esta tesis se considerará:
Acción: actividad realizada por un usuario sobre el sistema.
Evento: suceso que ocurre en el mundo externo al sistema y que produce un efecto sobre el mismo.

Nota: La presentación de las reglas se realiza según el siguiente esquema: primero se enuncia cada regla y luego se exponen los fundamentos.

5.2. Medir la funcionalidad (enfoque de reducción a episodios)

Teniendo en cuenta que los escenarios representan las funciones requeridas por el usuario y considerando la definición de *evento*, la idea es utilizar un enfoque similar al planteado en [Symons'91] al describir el proceso de "descomposición funcional", donde se explicita que las funciones del sistema son descompuestas o refinadas reiteradamente hasta obtener -en el menor nivel- combinaciones entrada-proceso-salida únicas e independientes.

La propuesta a desarrollar consiste en descomponer cada escenario en un conjunto de *episodios* que representen acciones únicas, determinar el conjunto unión de todos los *episodios* asegurando que no haya *episodios* repetidos, calcular los FP de cada uno de ellos y luego a partir de la sumatoria obtener el total de FP del sistema. Ver Figura 24.

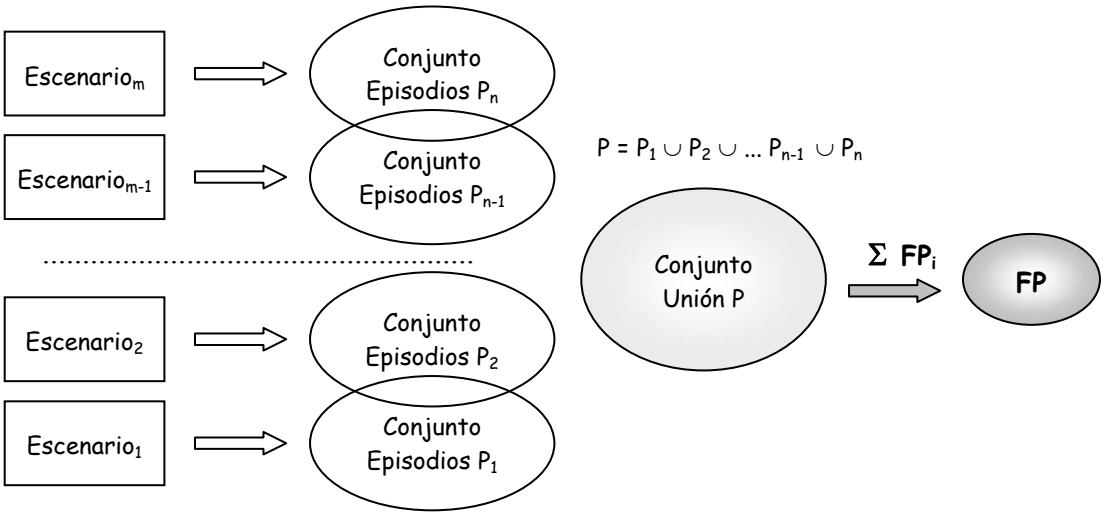


Figura 24 - Modelo de cálculo de FP

Los escenarios mencionados son *escenarios particulares*, que son aquéllos que describen un momento específico de la aplicación y usualmente sus *episodios* corresponden a simples acciones. Con respecto a los *escenarios generales*, que son aquéllos que representan las funciones fundamentales de la aplicación y usualmente sus *episodios* referencian a otros escenarios [Hadad'96], se debe verificar si existen *episodios* que representen simples acciones y si así fuera deberán incluirse en el respectivo conjunto de *episodios*.

Descripción de la propuesta:

a. Sea ***E*** una colección de M escenarios E_i, i=1 a M. Cada escenario tiene un conjunto de *episodios*, sólo se consideran aquéllos que no son escenarios.

b. Se eliminan las repeticiones de *episodios* y se obtiene el conjunto ***P*** de N *episodios* P_i, i=1 a N.

c. Se indica con FP_i los FP del *episodio* P_i

d. Se define con $FP = \sum_{i=1}^{N} FP_i$ la cantidad total de FP del sistema.

Una vez obtenido el conjunto de *episodios* P, se trata de calcular los FP de cada *episodio*. Esto significa que una cuestión fundamental es establecer un conjunto de reglas y procedimientos que sirvan como guía para la realización de ese proceso.

Para el método MKII el *Índice de Puntos Función* de una aplicación es la sumatoria de los *tipos de elementos dato* (*DET*) de entrada (N_I), los *tipos de entidades referenciadas* (N_E) y los *DET* de salida (N_O), cada uno de ellos afectado por el respectivo *peso promedio en la industria* W_I, W_E, W_O para todas las *transacciones lógicas* [Symons'91] [MKII FPA CPM'98].

$$FPI = W_I \sum_{i=1}^{N} N_{I_i} + W_E \sum_{i=1}^{N} N_{E_i} + W_O \sum_{i=1}^{N} N_{O_i}$$

donde: W_I = 0.58, W_E = 1.66 y W_O = 0.26 y N es la cantidad de *transacciones*.

La hipótesis de MKII es que si el propósito de la medición es medir la productividad y usarla en estimación, entonces los pesos relativos deberían reflejar el esfuerzo relativo promedio necesario para analizar, diseñar, programar, probar e implementar esas características, donde el promedio se obtiene desde algún ámbito apropiado para la tarea. Para comparaciones de productividad en la industria el promedio debería obtenerse en base a la industria en general, para estimación de proyectos en un entorno técnico particular debería ser medido en ese entorno [Symons'91].

La propuesta de esta tesis consiste en obtener los FP sin tener en cuenta el esfuerzo, que en la fórmula anterior está reflejado por la aplicación de los pesos estándar. Por lo tanto, los FP de cada *episodio* se calculan mediante la siguiente fórmula:

$$FP = N_I + N_R + N_O$$

donde N_I es la cantidad de *DET* de entrada, N_R la cantidad de *recursos referenciados* y N_O la cantidad de *DET* de salida.

El *Índice de Puntos Función* se calcula mediante:

$$FP = \sum_{i=1}^{N} N_{I_i} + \sum_{i=1}^{N} N_{R_i} + \sum_{i=1}^{N} N_{O_i}$$

donde N es el total de *episodios*.

5.3. Tamaño funcional y episodios

En MKII durante las primeras etapas de análisis del problema se confecciona una lista de todas las *transacciones candidatas*. En las siguientes etapas se pueden confirmar o rechazar las *transacciones candidatas* e incorporar nuevas *transacciones* [MKII FPA CPM'98].

De aquí en adelante se trabaja con el concepto de *episodio candidato*:

Definición: se denomina *episodio candidato* al *episodio* que puede estar incluido en la lista de *episodios* elaborada inicialmente, pero aún se debe establecer el compromiso definitivo para determinar su inclusión en la lista definitiva.

A los efectos de estudiar la funcionalidad de la aplicación se calcularán los FP de los *episodios candidatos*, para lo cual se formularán un conjunto de reglas[8] que establezcan los criterios para determinar qué debe y qué no debe considerarse un *episodio candidato*.

Regla 1

Situación: un *episodio* representa un escenario.
Decisión: no se lo considera para contabilizar los FP.
Ejecución: no incluir en la lista de *episodios* candidatos.

Según se adelantó, no se incluyen los *episodios* que son escenarios.

Ejemplo: [9]

– **check in** [10]

Aclaración del ejemplo: en el L&E se observa que **check in** *es un episodio de un escenario que a su vez es un escenario.*

Regla 2

Situación: un *episodio* representa acciones que no aportan funcionalidad al sistema.
Decisión: no se lo considera para contabilizar los FP.
Ejecución: no incluir en la lista de *episodios candidatos*.

En los escenarios se encuentra una descripción completa de las interacciones del actor con el sistema. Dentro de ellas existen algunas que representan acciones que no resultan en un aporte a la funcionalidad del sistema, pues si bien es necesaria su realización, no implican procesamiento de interés. Esto significa que esos *episodios* se deben excluir de la lista de *episodios candidatos*.

Ejemplo:

– El *recepcionista*[11] le provee al *pasajero / huésped / pax* la *tarjeta codificada* de la *habitación* y el control remoto TV.

Aclaración del ejemplo: en el L&E se detecta que este episodio, si bien es necesario para completar el escenario, no aporta funcionalidad al sistema pues consiste en manipular objetos físicos sin requerimientos de procesamiento de interés.

[8] Para la definición de las reglas se utiliza el modelo descripto en [Rolland'93]. Ver Anexo 2.
[9] Los ejemplos corresponden al L&E del caso de estudio Recepción del Hotel incluido en el Anexo 1, caso contrario se indica la referencia.
[10] Este formato de texto corresponde al utilizado por BMW para representar los escenarios.
[11] Este formato de texto corresponde al utilizado por BMW para representar los símbolos del LEL.

5.3.1. Clasificación de los episodios según su tipo y estructura

Los *episodios* son el centro de este análisis debido a que *cada episodio representa una acción realizada por un actor, donde participan otros actores y se utilizan recursos* [Leite'96]. Como lo indica la definición, dado que representan *actividad*, puede esperarse que sean el camino hacia las funciones buscadas. Para reafirmar esta idea, según consta en [Hadad'96], a partir de ellos se derivarán requerimientos funcionales.

La idea es buscar una analogía de conceptos entre los *episodios* y las *transacciones lógicas*. Una vez descartados los *episodios* que son escenarios, los repetidos y los que no incluyen funcionalidad, los restantes *episodios* que se pueden encontrar en un escenario se deben clasificar según su tipo y estructura.

Los *episodios* son de tipo *simple*, *condicional* y *opcional*. Independientemente de su tipo, un *episodio* puede ser expresado como una única acción o puede ser un escenario. El modelo de escenarios provee la descripción de comportamientos con diferentes órdenes temporales. Una secuencia de *episodios* implica un orden de precedencia, pero un orden no secuencial requiere el agrupamiento de dos o más *episodios*. Esto se usa para expresar un orden paralelo o secuencial indistinto [Leite'00].

En la Figura 25[12] se presenta un esquema de clasificación de los *episodios*:

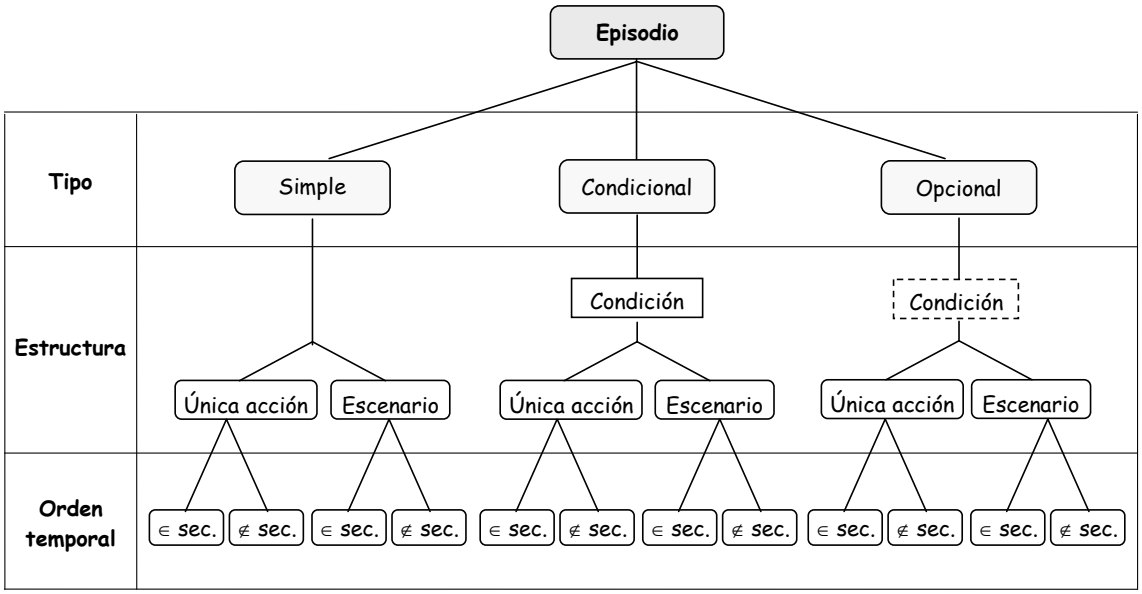


Figura 25 – Esquema de clasificación de los episodios

Debido al papel que juegan los *episodios* en cuanto a describir acciones, se debe hacer un análisis detallado de los mismos.

Desde la definición de *episodio*, se puede ver que cualquiera sea el tipo del mismo se puede expresar como una o varias acciones o un escenario.

[12] La línea punteada en *Condición* para episodio Opcional indica condición no explícita. En Orden temporal ∈ sec. significa que el episodio pertenece a un grupo secuencial, ∉ sec. el episodio pertenece a un grupo no secuencial.

En lo que sigue de este parágrafo se analizarán los tres tipos de *episodios* mencionados y en cada caso se hará explícita la *única acción* que le corresponde según la Figura 25. Luego se analizarán los *conjuntos de episodios.* Se excluyen del análisis los *episodios* que son *escenarios*.

Nota: En los siguientes párrafos se adoptan las definiciones correspondientes a los *episodios* del *Modelo de escenarios* propuesto por Leite [Leite'00], cuyo detalle se reproduce en la sección 2.3.2 del Capítulo 2 de esta tesis.

i) Episodios simples

Son aquéllos que son necesarios para completar el escenario [Leite'00].

Definición: a partir del Modelo de escenarios propuesto por Leite [Leite'00], un *episodio simple* se define como:
<episode > ::= <basic sentence>
<basic sentence> ::= <simple sentence>
<simple sentence> ::= <episode sentence> CR

Para facilitar el análisis se incluye la siguiente descripción [Leite'00]:

<episode sentence> is described:
(([Actor | Resource] + Verb + Predicate) | ([Actor | Resource] + [Verb] + Title)) + {Constraint} (1)

La primera parte (hasta la "|") de la expresión (1) en la definición de [Leite'00] se refiere al *episodio* cuya estructura se corresponde con una única acción según la clasificación, la segunda a un escenario.

Regla 3

Situación: un *episodio simple* representa una única acción, esta acción es atómica y se realiza en forma completa e independiente de otras.
Decisión: se lo considera para contabilizar los FP.
Ejecución: incorporarlo a la lista de *episodios candidatos*.

Única acción: representa una acción atómica -no puede descomponerse en otras- disparada por algún *evento* del mundo externo o por una *consulta* para obtener información, que se realiza en forma completa e independiente de otras. Las *consultas* difieren de los *eventos* debido a que éstos ocurren independientemente de la existencia de una aplicación de software, mientras las *consultas* son una consecuencia de la existencia de la aplicación [MKII FPA CPM'98].

Ejemplo de *evento*:

- El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo y modalidad de la *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

Aclaración del ejemplo: en el L&E se detecta que como consecuencia de un suceso del mundo externo (la llegada del pasajero) se dispara la realización del procesamiento descripto por el episodio (el registro de sus datos).

Ejemplo de *consulta*:

– El *recepcionista* busca el nombre del *pasajero / huésped / pax* en la *planilla de reservas*.

Aclaración del ejemplo: en el L&E se detecta una actividad que implica el retorno de información (resultado positivo o negativo de la búsqueda) desde la aplicación sin modificar los datos existentes.

ii) Episodio condicional

Los *episodios condicionales* son aquéllos cuya ocurrencia depende de una condición específica. La condición puede ser interna o externa al escenario. Las condiciones internas pueden deberse a precondiciones, restricciones de actores o *recursos* y *episodios* previos [Leite'00].

Definición: según el modelo de [Leite'00] un *episodio condicional* se define como:
<episode > ::= <basic sentence>
<basic sentence> ::= <conditional sentence>
<conditional sentence> ::= **IF** <condition> **THEN** <episode sentence> CR

Regla 4

Situación: un *episodio condicional* representa una única acción que puede o no ejecutarse según se cumpla una condición.
Decisión: se lo considera para contabilizar los FP.
Ejecución: incorporarlo a la lista de *episodios candidatos*.

Única acción: representa un *episodio simple* que sólo se ejecuta si se cumple la condición asociada.

Ejemplo:

– **if** existe *disponibilidad de habitaciones* y la persona estuviera de acuerdo **then** el *recepcionista* registra los datos personales del *pasajero / huésped / pax* en la *planilla de reservas*.

Aclaración del ejemplo: en el L&E se detecta que existe una condición que determina la realización o no del episodio.

iii) Episodio opcional

Los *episodios opcionales* son aquéllos que pueden o no tener lugar dependiendo de condiciones que pueden no estar explícitamente detalladas [Leite'00].

Definición: según el modelo de [Leite'00] un *episodio opcional* se define como:
<episode > ::= <basic sentence>
<basic sentence> ::= <optional sentence>
<optional sentence> ::= **[** <episode sentence> **]** CR

Regla 5

Situación: un *episodio opcional* representa una única acción que puede o no ejecutarse.
Decisión: se lo considera para contabilizar los FP.
Ejecución: incorporarlo a la lista de *episodios candidatos*.

Única acción: representa un *episodio simple* que puede o no ejecutarse.

Ejemplo:

– [El *convocante* confecciona el *temario*] [Hadad'97]

Aclaración del ejemplo: en el L&E se detecta que existen condiciones que pueden no ser explícitas que determinen la realización o no del episodio.

iv) Conjunto de episodios

Un *conjunto de episodios* puede ser: un grupo *secuencial* de *episodios* o un grupo *no secuencial* de *episodios*. Este último requiere del agrupamiento de dos o más *episodios* [Leite'00].

Definición: según el modelo de [Leite'00] se define como:
<episodes> ::= <group series> | <episode series>
<group series> ::= <group> <group> | <non-sequential group> | <group series> <group>
<group> ::= <sequential group> | < non-sequential group>
<sequential group> ::= <basic sentence> | <sequential group> <basic sentence>
<non-sequential group> ::= # <episode series> #
<episode series> ::= <basic sentence> <basic sentence> | <episode series> <basic sentence>

iv.1. Episodio perteneciente a un grupo secuencial

Los *episodios* que pertenecen a un grupo *secuencial* son aquéllos que se ejecutan según el orden de la secuencia. Cada uno de los *episodios* de la secuencia debe clasificarse según las Reglas 3 a 5.

iv.2. Episodio perteneciente a un grupo no secuencial

Los *episodios* que pertenecen a un grupo *no secuencial* son aquéllos que tienen un orden de ejecución paralelo o no secuencial.

Regla 6

Situación: un grupo de *episodios no secuencial* representa dos o más acciones.
Decisión: cada una de las acciones que cumpla con los criterios de las Regla 3 a 5 se considera para contabilizar los FP.
Ejecución: incorporar cada una de las acciones a la lista de *episodios candidatos*.

Dos o más acciones: Representa dos o más acciones que se ejecutan en orden no secuencial o paralelo.

Ejemplo:

– # El *solicitante* completa la *solicitud de adhesión* con sus datos personales.
El *solicitante* elige un *bien* # [Mauco' 97]

Aclaración del ejemplo: la especificación de este conjunto de episodios establece que se ejecutarán en orden paralelo o no secuencial.

v) Episodios que corresponden a excepciones

Las *excepciones o casos alternativos* se refieren a los casos de excepción que pueden interrumpir la evolución normal del escenario. Cada excepción se describe como una sentencia simple que especifica la causa de la interrupción. Si incluye el título de otro escenario, la excepción será tratada por ese escenario. Usualmente estos reflejan la imposibilidad de realización de un *episodio* o del escenario por la no disponibilidad de uso de un *recurso* [Leite'00].

Definición: a partir del modelo propuesto en [Leite'00], las excepciones se definen como:

Exceptions:
Syntax: Cause [(Solution)]
where Cause is:
Phrase | ([Subject | Actor | Resource] + Verb + Predicate)
where Solution is:
Title

Las excepciones o casos alternativos deben analizarse según corresponda a uno de los casos siguientes:

Sentencia simple: describe el motivo de la interrupción del *episodio* o escenario.

Ejemplo:

– **excepción:** Faltan tarjetas magnéticas para codificar

Aclaración del ejemplo: en el L&E se detecta una situación inesperada o inusual.

Por tratarse de la descripción de una situación inusual no incluye funcionalidad. No se consideran para el cálculo de FP.

Iincluye escenario: describe el tratamiento de la excepción. Según la Regla 1 no se incluyen los *episodios* que son escenarios.

Utilizando las reglas propuestas a partir del análisis previo se determinan todos los *episodios* que contienen las funciones requeridas, los que deberán ser contabilizados en la medición que se realice.

5.3.2. Restricciones

Las restricciones provienen del contexto del problema y del contexto del escenario. Es un atributo de los *recursos*, *episodios* básicos o subcomponentes del contexto [Leite'00]. Son aquellos aspectos que al presentarse impiden la acción. Las restricciones reflejan requerimientos no funcionales [Hadad'97]. No se considerarán pues no representan funcionalidad. Para avalar esta afirmación, cuando Cysneiros [Cysneiros'01] propone la extensión del LEL como una herramienta de soporte para elicitar requerimientos no funcionales (NFR), menciona que un tipo de objetivo importante, llamado *requerimiento no funcional*, se centra en *cómo* debe hacer algo el software en lugar de *qué* debe hacer el software. Algunos métodos de desarrollo de software mencionan los NFR, entre otras maneras, como restricciones o atributos de calidad del software. Establece además que: toda información incluida en un escenario que satisface un NFR debe ser definida como una restricción.

Definición: en el modelo propuesto en [Leite'00] las restricciones se definen como:
Constraint

Syntax: ([Subject | Actor | Resource] + **Must** [**Not**] + Verb + Predicate) | Phrase

Regla 7

Situación: un *episodio* representa una *restricción*.
Decisión: no se considera para contabilizar los FP.
Ejecución: no incorporar a la lista de *episodios candidatos*.

Desde el punto de vista del FPA, según se detalla en la sección 3.7.6 al describir el método MKII, si bien el método ofrece un factor para considerar la complejidad técnica y ciertos requerimientos de calidad, actualmente se recomienda no incluirlos [MKII FPA CPM'98].

Ejemplo:

– **restricción**: No hay *disponibilidad de habitaciones*

Aclaración del ejemplo: desde la especificación de esta sentencia en el L&E se detecta un impedimento para realizar una determinada acción.

5.3.3. Comparación de los conceptos de transacción lógica y episodio

Las *transacciones lógicas* son los procesos de más bajo nivel soportados por una aplicación de software [MKII FPA CPM'98]. MKII utiliza el concepto de *transacción lógica* definido como: *"una combinación de uno o más tipos de entrada, algún procesamiento y uno o más tipos de salida, correspondientes a un proceso único e*

independiente". Es disparada por un *único evento del mundo externo significativo para el usuario* o es el resultado de *una consulta para retornar o extraer información* que no modifica los datos almacenados [Symons'91]. En la Figura 26 se muestra un esquema representativo de las *transacciones lógicas*.

Figura 26 - Transacciones lógicas [GIFPA'98]

¿Qué se quiere significar con el término *disparada*? Se refiere a que la parte de procesamiento y salida de la *transacción* ocurren como respuesta al estímulo provisto por la parte de entrada que corresponde en sí misma al *evento* del mundo externo [MKII FPA CPM'98].

¿Qué es un *evento del mundo externo significativo para el usuario*? son aquellas situaciones del mundo externo -todo aquello que está fuera del *límite* se considera el mundo externo- acerca de las cuales el usuario requiere que la aplicación procese información [MKII FPA CPM'98].

¿Qué es *una consulta para retornar o extraer información* que no modifica los datos almacenados? para recuperar la información desde una aplicación es necesaria la realización de *consultas* (por un usuario en el mundo externo). Por definición no provocan cambios de estado [MKII FPA CPM'98].

El análisis de la definición de *episodio* sugiere que para que un actor pueda realizar una acción sobre el sistema debe trasmitir algún mensaje que provoque la realización del procesamiento correspondiente a esa única acción. Durante el procesamiento la aplicación utiliza *recursos*. Evidentemente para que la acción realizada por la aplicación tenga sentido, es necesario que produzca algún resultado como salida. El análisis de los diferentes tipos de *episodios* permite establecer como concepto que el *episodio simple que representa una única acción es equivalente al concepto de transacción lógica*. En MKII a los *eventos* candidatos a *transacciones* se les aplica la prueba *ACID* para determinar si cumplen cuatro condiciones, en caso contrario se descartan del cálculo de los FP. Esto significa que los *eventos* [MKII FPA CPM'98]:

- Son *Atómicos*: un *evento* no puede descomponerse en otros *eventos*.
- Preservan la *Consistencia* y son *Instantáneos*: cada *evento* se completa satisfactoriamente o falla completamente, no hay posibilidades de puntos intermedios.
- Son *Detectables*: el *evento* debe ser observable o al menos se debe poder inferir.

Es razonable suponer que un *episodio simple* cumple con las condiciones citadas. Por un lado se ha establecido que representan acciones únicas, no pueden descomponerse en otras, significando con este concepto que se trata de acciones atómicas. Además, una vez comenzadas, estas acciones se realizan o fallan completamente. Por otro lado, los *episodios* provienen del impacto del símbolo del LEL de un actor principal, por lo tanto son acciones detectadas por su repercusión sobre el sistema.

Cualquier *episodio candidato* que no cumpla alguna de las condiciones anteriores, debe descartarse para el cálculo de los FP. Si los hubiere, será necesario revisar los *episodios* para verificar si están correctamente definidos.

Hasta ahora se han identificado todos los *episodios* que resultan de interés para el análisis de la funcionalidad. Los escenarios no se considerarán como *episodios* en ninguno de los casos, pues deben ser evaluados en la oportunidad que se estudie el escenario, de otra manera se duplicarían los cálculos y se fundamenta en las mismas razones que motivaron la propuesta de utilizar el conjunto unión de todos los *episodios*.

5.3.4. Clasificación de episodios según el procesamiento requerido

Para clasificar los *episodios* según este criterio, se debe determinar qué necesidades de procesamiento satisface la realización del *episodio*, considerando que las interacciones de un actor con el sistema son provocadas por:

- La ocurrencia de algún suceso en el mundo externo que requiere la realización de algún procesamiento.
- La necesidad de recuperar información.

Teniendo en cuenta las definiciones de *evento* y *consulta* de MKII y la equivalencia entre *transacción lógica* y *episodio simple*, si se considera que las únicas interacciones posibles entre un sistema y un actor son las mencionadas en los párrafos previos, entonces se puede establecer la correspondencia de la primera de ellas con el concepto de *evento* y de la segunda con el de *consulta*. Este análisis avala la siguiente conclusión:

Todo *episodio* definido según la Regla 3 debe pertenecer a la clasificación *evento* o *consulta*.

5.3.5. Componentes de los episodios

En esta sección, teniendo presente el objetivo de calcular la cantidad de FP del sistema, se debe determinar dentro de los *episodios* cuáles son los componentes que contribuyen a ese cálculo. En pos de ese objetivo, se debe buscar un modelo que se ajuste a las necesidades para describir la realidad.

Una abstracción de la interacción entre usuario y sistema se puede describir como: el usuario se comunica con el sistema para requerir un servicio, el sistema realiza el proceso requerido y retorna al usuario los resultados. En términos más concretos esta interacción consiste en las siguientes actividades: ingreso de datos desde el usuario, realización de algún procesamiento por parte del sistema y el retorno de datos hacia el usuario. Para modelizar esta interacción usuario-sistema se puede usar el modelo *entrada-proceso-salida (E-P-S)* formado por tres componentes que representan: el ingreso de un flujo de datos proveniente del usuario que atraviesa el *límite del sistema (entrada)*, el procesamiento sobre los datos ingresados dentro

del *límite del sistema (proceso)* y el retorno hacia el usuario de un flujo de datos que cruza el *límite del sistema (salida).*

De lo anterior se desprende la similitud del modelo propuesto con la definición de MKII: Cualquier *transacción lógica* debe tener alguna entrada del usuario que atraviesa el *límite*, debe hacer algún procesamiento sobre los *tipos de entidades* dentro del *límite* y retornar la salida a través del *límite* [MKII FPA CPM'98].

Las interacciones usuario-sistema están representadas por los *episodios*. En base al modelo seleccionado se deben descomponer los *episodios* en partes representativas de los componentes *entrada*, *proceso* y *salida*. Una cuestión relevante dentro de este modelo es la cantidad de elementos que deben manipularse, tanto sea para el ingreso y validación, procesamiento y formateo de la salida. Es una convención en MKII FPA que como mínimo cada *transacción lógica* debe tener un *DET* de entrada, debe hacer *referencia* a un *tipo de entidad* y debe tener un *DET* de salida [MKII FPA CPM'98]. La equivalencia entre *transacción* y *episodio simple* soporta la adopción del mismo criterio para evaluar los componentes *entrada*, *proceso* y *salida* de un *episodio*. Para ello se considerarán las definiciones de *DET* y *entidad* de MKII:

DET: un ítem de información único, no recursivo, reconocible por el usuario [MKII FPA CPM'98].

Entidad: alguna cosa en el mundo real (estrictamente hablando algún tipo de cosa) tangible o intangible, acerca de la cual se desea mantener información [13] [MKII FPA CPM'98].

Los *DET* mantienen información acerca de *tipos de entidades* [MKII FPA CPM'98].

El siguiente ejemplo de *episodio* servirá para aclarar los conceptos:

– El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo de *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

Utilizando el modelo *E-P-S* para describir el *episodio* resulta:

- *Entrada:* consiste en la adquisición de los datos provistos por el usuario que llegan al sistema, sea para describir un *evento* de interés en el mundo externo o una *consulta* para recuperar información del sistema [MKII FPA CPM'98].

  En el ejemplo está representada por los datos que provee el recepcionista: nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo de *habitación*, *día de ingreso*, *día de egreso* y *tarifa*

- *Proceso:* consiste en las acciones de almacenamiento y/o recuperación de información que describe el estado de los elementos de interés para el mundo exterior [MKII FPA CPM'98].

[13] En el método MKII interesa distinguir las entidades *primarias* y *no primarias*. Estas últimas se agrupan en una única entidad *System*. Las referencias a las entidades *primarias* y *no primarias* agrupadas en *System*, se consideran en el componente procesamiento.
MKII también establece que pueden definirse subtipos de entidades cuando se requiera alguna variación específica de una entidad primaria, para la que se requieren reglas de procesamiento distinto [Symons'91].

En el ejemplo está representada por la acción de almacenamiento de los datos provistos a través de la parte de entrada en la *planilla de reservas*. En este caso la entidad es *planilla de reservas* y la acción es actualizar.

- *Salida:* son los ítems provistos al usuario, sea cual fuere la categoría de la acción. Esta parte comprende las tareas de presentación de la información hacia el mundo externo [MKII FPA CPM'98].

  En el ejemplo está representada por los datos actualizados en la *planilla de reserva* (nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo de *habitación*, *día de ingreso*, *día de egreso* y *tarifa*)

Clasificación de los componentes E-P-S de los episodios

Al definir los componentes *E-P-S* del *episodio*, se utilizaron los términos *DET* y *entidad* de MKII. Se debe establecer la equivalencia conceptual entre estos conceptos y los objetos[14] que integran los *episodios*.

Los componentes según el modelo *E-P-S* de un *episodio* contienen una serie de objetos que se requiere clasificar para luego contabilizar. Esos objetos se pueden clasificar en dos grupos: símbolos del LEL y palabras o frases que no han sido definidas dentro de las entradas del LEL [Hadad'96]. A su vez dichos símbolos pueden subclasificarse en *Simples* y *Complejos*. En la Figura 27 se muestra un esquema de la clasificación propuesta.

Definido en el LEL { Simple / Complejo

No definido en el LEL

Figura 27 – Clasificación de los objetos del LEL

Definido en el LEL: se corresponde con las entradas del LEL.

- *Simple*: un ítem atómico de información reconocible por el usuario. Este a su vez puede ser:
  - independiente, no depende ni participa en la definición de otros símbolos.

    Ejemplo: *tarjeta codificada*

    Aclaración del ejemplo: en el L&E se observa que esta entrada está definida por un ítem atómico de información, no utiliza otra entrada y no participa en la definición de otra entrada.

  - no depende de otros símbolos y es una entrada de otro símbolo.

    Ejemplo: *tarifa* forma parte de *lista de precios*

[14] En el contexto de esta tesis se considera la definición: Un objeto es cualquier entidad del mundo real, importante para la discusión de los requerimientos, con un borde claramente definido. [Davis'93]

Aclaración del ejemplo: en el L&E se observa que esta entrada está definida por un ítem atómico de información, no utiliza otra entrada y participa en la definición de otra entrada.

- *Complejo*: es un conjunto formado por uno o más símbolos pertenecientes a la clasificación *Simple* y/o uno o más *No definido en el LEL*.

  Ejemplo: *planilla del pasajero* consiste de: nombre, número de documento, datos personales y comerciales, firma, *día de ingreso*, procedencia del *pasajero / huésped / pax / huésped / pax* y datos del vehículo

  Aclaración del ejemplo: en el L&E se observa que esta entrada está definida por un conjunto compuesto por otras entradas y palabras/frases No definidas en el LEL.

No definido en el LEL: palabras o frases que por sus características no se describen como entradas del LEL, pero son necesarias para el desarrollo del *episodio*. Estos pueden representar objetos que intuitivamente se reconocen como atómicos, por ejemplo: número de documento; compuestos, por ejemplo: datos personales o *recursos*, por ejemplo: teléfono o formulario.

Tipos de elemento dato

En los componentes *entrada* y *salida* del *episodio* se detecta que los objetos generalmente pertenecen a la clasificación *Simple*, en algunos casos a *Complejo* y a *No definido en el LEL*.

Regla 8

Situación: el componente *entrada/salida* incluye una entrada del LEL perteneciente a la clasificación *Simple*.
Decisión: se debe clasificar como *DET*.
Ejecución: incluir en la lista de *DET* de entrada/salida.

Regla 9

Situación: el componente *entrada/salida* incluye un objeto perteneciente a la clasificación *No definido en el LEL*.
Decisión: se debe clasificar como *DET*.
Ejecución: incluir en la lista de *DET* de entrada/salida.

La similitud de las definiciones *Definido en el LEL/Simple* y objeto atómico *No definido en el LEL* con la definición de *DET* de MKII soporta establecer la equivalencia entre los conceptos y se define:

Definición: se denomina *tipo de elemento dato (DET)* a toda entrada del LEL perteneciente a la clasificación *Simple* y a los objetos atómicos *No definido en el LEL* que integran los componentes *entrada* y *salida* del *episodio*.

Ejemplo:

Entrada Definido en el LEL/Simple: *tarifa*

– El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo de *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

Aclaración del ejemplo: en el episodio se observa que este símbolo clasificado como Simple participa del componente entrada para un cierto procesamiento, por lo tanto se debe incluir en la lista de DET de entrada.

Ejemplo:

Objeto No definido en el LEL: cantidad de pasajeros

– El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo de *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

Aclaración del ejemplo: en el episodio se observa que este objeto No definido en el LEL participa del componente entrada para un cierto procesamiento, por lo tanto se debe incluir en la lista de DET de entrada.

Regla 10

Situación: el componente *entrada/salida* incluye una entrada del LEL perteneciente a la clasificación *Complejo*.
Decisión: se deben identificar los ítems que lo componen.
Ejecución: buscar en la noción del símbolo del LEL e incluir cada uno de los ítems que lo componen en la lista de *DET* de entrada/salida.

Cuando en el componente *entrada* se detecta una entrada del LEL perteneciente a la clasificación *Complejo*, se debe buscar el símbolo del LEL y a partir de la *noción* obtener los *DET* que lo componen.

Ejemplo:

Entrada del LEL Definido en el LEL/Complejo: *disponibilidad de habitaciones*.

– El *recepcionista* actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*.

La noción del símbolo *disponibilidad de habitaciones* establece: Contiene la cantidad y tipo de *habitación* desocupada. Esto significa considerar los *DET* cantidad y tipo de *habitación*.

Aclaración del ejemplo: en el episodio se observa que este símbolo clasificado como Complejo participa del componente entrada para un cierto procesamiento, por lo tanto se debe buscar en la noción del símbolo los DET que lo integran e incluirlos en la lista de DET de entrada.

Regla 11

Situación: un *episodio* se inicia "automáticamente".
Decisión: identificar el ítem disparador y clasificarlo como *DET* de entrada.
Ejecución: agregar a la lista de *DET* de entrada.

Los *eventos* que un sistema puede responder son de dos tipos: *eventos externos*, que son iniciados por entidades en el entorno y *eventos temporales* que son iniciados por el pasaje del tiempo [Mc.Menamin'84].

Cuando las acciones se inician "automáticamente" se debe encontrar el disparador real. Una fecha puede ser un disparador válido. Un cambio de fecha, por ejemplo, el final de un mes, es un *evento* externo válido y se trata como una entrada a la aplicación que provoca una *transacción* [MKII FPA CPM'98]. Esta es una situación posible en el marco de este enfoque y se va a considerar teniendo en cuenta que el *evento* externo de las características indicadas será una entrada al sistema que disparará un *episodio*.

Ejemplo de esta situación provocado por el *no show*[15] del pasajero:

– **if** el *pasajero / huésped / pax* no se presenta en el período comprendido entre las 12 hs del *día de ingreso* establecido en la *solicitud de reserva* y las 06 hs. del día siguiente **then** el *recepcionista* elimina la *solicitud de reserva* de la *planilla de reservas*.

Aclaración del ejemplo: en el episodio se observa que la ocurrencia de una fecha y hora disparan la realización de un cierto procesamiento, en este caso representan los DET de entrada, por lo tanto se deben agregar a la lista de DET de entrada.

Recursos

Durante la realización de un *episodio* se utilizan *recursos*. Éstos son los medios de soporte, dispositivos u otros elementos pasivos que deben estar disponibles en el escenario [Hadad'97]. Los *recursos* son entradas del LEL que pueden pertenecer a la clasificación *Simple*, *Complejo* o *No definido en el LEL*, estos últimos generalmente representan objetos físicos. Si bien hay alguna similaridad entre los conceptos de *entidad* de MKII y *recurso* no se puede establecer una relación uno a uno entre ambos, pues el *recurso* tiene un significado más amplio, involucra objetos que no son considerados entidades en ese método. Debe recordarse que usualmente las entidades se derivan desde el modelo Entidad-Relación de la aplicación [MKII FPA CPM'98]. Esto significa que el hecho de tomar como base del cálculo de los FP a L&E pueda derivar en una mayor cantidad de FP que si el cálculo se realizara por un método clásico. En el marco de este enfoque se considerará que las *entidades* de MKII que componen la parte de *proceso* son los *recursos*.

[15] Es la baja de la reserva por no presentación del pasajero

Regla 12

Situación: en el componente *proceso* de un *episodio* se identifica una entrada perteneciente a la clasificación *Simple*, *Complejo* o un objeto *No definido en el LEL*.
Decisión: se debe clasificar como *recurso*.
Ejecución: incorporar a la lista de *recursos* del *episodio*.

Definición: se denomina *recurso* a toda entrada del LEL perteneciente a la clasificación *Simple* o *Complejo* y a los objetos *No definido en el LEL* que forman parte del componente *proceso* del *episodio*.

Ejemplo:

- *Recurso* perteneciente a la clasificación *Simple*: *tarjeta codificada*
  - El *recepcionista* hace una *tarjeta codificada* para el número de *habitación*.

  Aclaración del ejemplo: en el componente proceso del episodio se detecta una entrada clasificada como Simple que es necesaria para realizar el episodio, debe considerarse recurso e incluirse en la lista de recursos.

- *Recurso* perteneciente a la clasificación *Complejo*: *planilla de reservas*
  - El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo de *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

  Aclaración del ejemplo: en el componente proceso del episodio se detecta una entrada clasificada como Complejo que es necesaria para realizar el episodio, debe considerarse recurso e incluirse en la lista de recursos.

- *Recurso* perteneciente a la clasificación *No definido en el LEL*: sello de Pagado
  - El *recepcionista* agrega el sello de Pagado en la factura

  Aclaración del ejemplo: en el componente proceso del episodio se detecta un objeto No definido en el LEL que es necesario para realizar el episodio, debe considerarse recurso e incluirse en la lista de recursos.

Aporte a la funcionalidad de cada uno de los componentes

Una vez identificados los *DET* en los componentes *entrada* y *salida* y los *recursos* en el componente *proceso* del *episodio* se deben establecer reglas para valorar su contribución a la cuenta de FP del *episodio*.

Regla 13

Situación: el componente *entrada/salida* del *episodio* incluye un *DET* clasificado según las Reglas 8 a 11.
Decisión: contabilizar cada *DET* distinto y no las ocurrencias de cada uno de ellos.
Ejecución: sumar 1 (por cada *DET*) a la cuenta de *DET* de entrada/salida.

Los *DET* en el componente entrada y salida del episodio: la cantidad de *DET* se usa para determinar el tamaño de la parte de *entrada* y *salida* de cada *transacción lógica*. Se deben contar únicamente los *DET*, no las ocurrencias individuales [MKII FPA CPM'98]. La equivalencia conceptual entre *transacción* y *episodio* permite establecer que el tamaño de los componentes *entrada* y *salida* de un *episodio* es proporcional a la cantidad de *DET* en los componentes *entrada* y *salida*.

Ejemplo:

- El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo y modalidad de la *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

Aclaración del ejemplo: el componente entrada del episodio incluye una lista de 4 DET clasificados según la Regla 9 y 3 por la Regla 8, luego la cuenta de DET de entrada es 7.

Regla 14

Situación: el componente *proceso* de un *episodio* incluye una *referencia* a un *recurso* clasificado según la Regla 12.
Decisión: contabilizar una única *referencia* al *recurso*.
Ejecución: si es la primera vez que aparece la *referencia* al *recurso* sumar 1 (por cada recurso referenciado) a la cuenta de *recursos*.

Las referencias a *recursos*: el tamaño del componente *proceso* de una *transacción lógica* es proporcional a la cantidad de *entidades referenciadas* durante la realización de las acciones [MKII FPA CPM'98]. La equivalencia conceptual establecida en esta sección (apartado Recursos) entre *entidad* y *recurso* permite establecer que el tamaño del componente *proceso* de un *episodio* es proporcional a la cantidad de *recursos referenciados* durante la realización de las acciones.

En una aplicación que almacena información acerca de un conjunto de *entidades*, la presencia de cada *entidad* automáticamente implica la existencia de un conjunto mínimo de *transacciones lógicas*, conocido en forma coloquial con el acrónimo CRUDL, que significa Create, Read, Update, Delete, List [MKII FPA CPM'98]. Cuando en el componente *proceso* se habla de *entidades referenciadas*, implícitamente se están considerando esas operaciones sobre las *entidades*.

Vale analizar si existe una relación uno a uno entre ellas y las operaciones posibles en el entorno de este enfoque. Si bien se consideran válidas, deben ampliarse para incluir también el concepto de "uso" del *recurso*, cuestión que deriva directamente de la definición de *recurso* y que no necesariamente es alguna de las operaciones CRUDL, simplemente puede ser manipularlo para realizar alguna actividad, por ejemplo, utilizar el teléfono. Por lo tanto, el concepto de *referenciar* el *recurso* incluye también el uso del mismo. Esto remite nuevamente a la cuestión mencionada antes acerca de la probabilidad de obtener un número mayor de FP usando este enfoque.

Definición: las operaciones de creación, lectura, actualización, borrado, listado y uso de un *recurso* se denominan genéricamente *referenciar* un *recurso*.

En el componente *proceso* se debe considerar una única *referencia* por cada *recurso* distinto accedido durante la realización del *episodio*. Si hay múltiples *referencias* a un mismo *recurso* se cuenta sólo una vez. Por ejemplo, si una operación de actualización requiere el acceso a un *recurso* para una operación de lectura previa a la actualización debe considerarse una única *referencia* [MKII FPA CPM'98].

Ejemplo:

– El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo y modalidad de la *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*

Aclaración del ejemplo: en el componente proceso del episodio se accede (referencia) a un recurso para actualizarlo, por lo tanto la cuenta de recursos es 1.

5.4. El límite del sistema

MKII FPA se usa para medir la funcionalidad requerida por los usuarios de la aplicación dentro de un *límite* definido por el *propósito de la medición* y el *punto de vista* [MKII FPA CPM'98].

Propósito de la medición: está determinado por el objetivo de la medición. Puede ser, por ejemplo, medir la productividad de un grupo de desarrolladores, estimar el esfuerzo para desarrollar el proyecto, medir la funcionalidad entregada al usuario [MKII FPA CPM'98].

Punto de vista: comúnmente se distinguen tres puntos de vista, *Project*, *Manager Application* y *Business Enterprise* [MKII FPA CPM'98]. Éste ayudará a determinar qué funcionalidad debe incluirse o no en la medición.

Límite de la aplicación: define un borde conceptual continuo entre el software y el usuario. En el contexto de FPA un usuario puede ser: *usuario humano*, por ejemplo un empleado, alguna persona que interactúa con la aplicación o *usuario automatizado*, otra aplicación o dispositivo automático de recolección de datos que suministra o recibe un servicio de la aplicación en estudio. La aplicación incluida dentro de ese *límite* debe conformar un cuerpo consistente de funcionalidad, significando con esto que dicho límite debe incluir operaciones (o procesos) completos [MKII FPA CPM'98].

Regla 15

Situación: los escenarios representan las funciones del sistema.
Decisión: establecer como límite del sistema al borde conceptual que encierra todos los escenarios.
Ejecución: considerar a cada uno de los escenarios dentro del límite del sistema y todo lo que no está contenido en los escenarios está fuera del sistema.

La visión del modelo de escenarios se corresponde con el concepto de *límite* de MKII, en tanto los *actores* están fuera de la aplicación y el conjunto de todos los escenarios representa la funcionalidad de la aplicación. Cualquier otra función que pueda considerarse necesaria, pero no fue requerida por el usuario debe descartarse a los fines del cálculo de los FP.

Definición: se denomina *límite del sistema* al borde conceptual que encierra a todos los escenarios y todo aquello que no es un escenario se considera fuera del *límite del sistema*.

El *límite del sistema* elicitado utilizando L&E y el del software a desarrollar no necesariamente deben ser coincidentes debido a que puede ser que no todas las funciones del sistema sean automatizadas. En consecuencia, el *límite* definido a partir de L&E probablemente será mayor que el del software. El cálculo de los FP se realizará en base al *límite* determinado a partir del L&E.

Una vez definido el *límite del sistema*, es necesario analizar los conceptos de *usuario* en FPA y *actor* en los escenarios para estudiar la equivalencia entre ambos.

Regla 16

Situación: la descripción de un escenario incluye uno o más actores.
Decisión: considerar *usuario* a cada uno de los actores presentes en los escenarios.
Ejecución: armar una lista de usuarios del sistema con todos los actores de los escenarios y solamente éstos.

Vale mencionar que un análisis similar se realiza en el contexto del modelo de *Casos de uso* y FPA, donde los términos *actor* y *usuario* se consideran sinónimos [Longstreet'01]. Según Fetcke, en el marco del enfoque OO-Jacobson el conjunto de actores da una visión de la funcionalidad completa de los usuarios y aplicaciones externas, pero este conjunto puede incluir actores que no son considerados usuarios en el entorno de FPA, pues OO-Jacobson visualiza la funcionalidad del sistema subyacente como un *actor* (bases de datos, dispositivos de impresión). En consecuencia para permitir el mapeo de conceptos se deben excluir estos últimos y seleccionar aquéllos que se corresponden con el concepto de *usuario* en FPA [Fetcke'98].

En el marco de esta propuesta deben considerarse:

Concepto de *usuario* en FPA: ISO 14143-1 define un *usuario* como "cualquier persona que especifica requerimientos funcionales del usuario y/o cualquier persona o cosa que se comunica o interactúa con el software en cualquier momento durante el tiempo de vida del software" [MKII FPA CPM'98].

Concepto de *actor* en los escenarios: un *actor* se define como la persona, dispositivo o estructura organizacional que tiene un rol en el escenario [Leite'00]. Un *actor* se identifica a partir de su interacción con el sistema y por los eventos que envía hacia o recibe desde el sistema. No siempre un usuario es necesariamente un *actor*, ya que un

mismo usuario puede representar varios papeles en el sistema, teniendo así más de un *actor* para representarlo [Caldas'98].

El análisis de las definiciones de *usuario* en FPA y *actor* en los escenarios sugiere la equivalencia de ambos conceptos y soporta la decisión: el conjunto de todos los *actores* de los escenarios representan adecuadamente el concepto de *usuario* en FPA.

Capítulo 6

Proceso de medición de funcionalidad de L&E

Este capítulo está organizado de la siguiente manera. En la sección 6.1 se presenta el esquema del proceso de medición. En la sección 6.2 se describen las etapas que componen el proceso de medición. En la sección 6.3 se proponen los formularios para documentar el proceso de medición.

6.1. Introducción

El proceso de medición de funcionalidad de L&E se describe como una serie de etapas en secuencia cuyo esquema se muestra en la Figura 28. En la columna a la izquierda del gráfico se indican las etapas y a la derecha de cada etapa se representa cada uno de los subprocesos en que se descompone cada etapa. El sentido de las flechas en el gráfico significa que el producto resultante de cada etapa es utilizado como entrada por la etapa siguiente. La Etapa 4 debe realizarse para cada *episodio* obtenido como producto de la Etapa 3. La Etapa 5 se debe realizar para cada componente de cada *episodio* que se obtiene como producto de la Etapa 4. El subproceso 6.1 de la Etapa 6 se debe realizar para cada *episodio*.

Durante las diferentes etapas del proceso se aplican las reglas que se definieron en las secciones 5.3 y 5.4 del Capítulo 5. En la Tabla 14 se visualiza una síntesis de las relaciones entre las etapas del proceso de medición y las reglas que se usan en cada una de ellas. Una X en la intersección de la fila Etapa con la columna Regla significa que en esa etapa se utiliza la regla correspondiente a la columna.

		Regla															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Etapa	1.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	1.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2	2.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-
	2.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X
3	3.1	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	3.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	3.3	-	-	X	X	X	X	X	-	-	-	-	-	-	-	-	-
	3.4	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	3.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4	4.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	4.2	-	-	-	-	-	-	-	X	X	X	X	-	-	-	-	-
	4.3	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-
5	5.1	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-
	5.2	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-
	5.3	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-
6	6.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	6.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tabla 14 – Matriz de relaciones Etapa - Regla

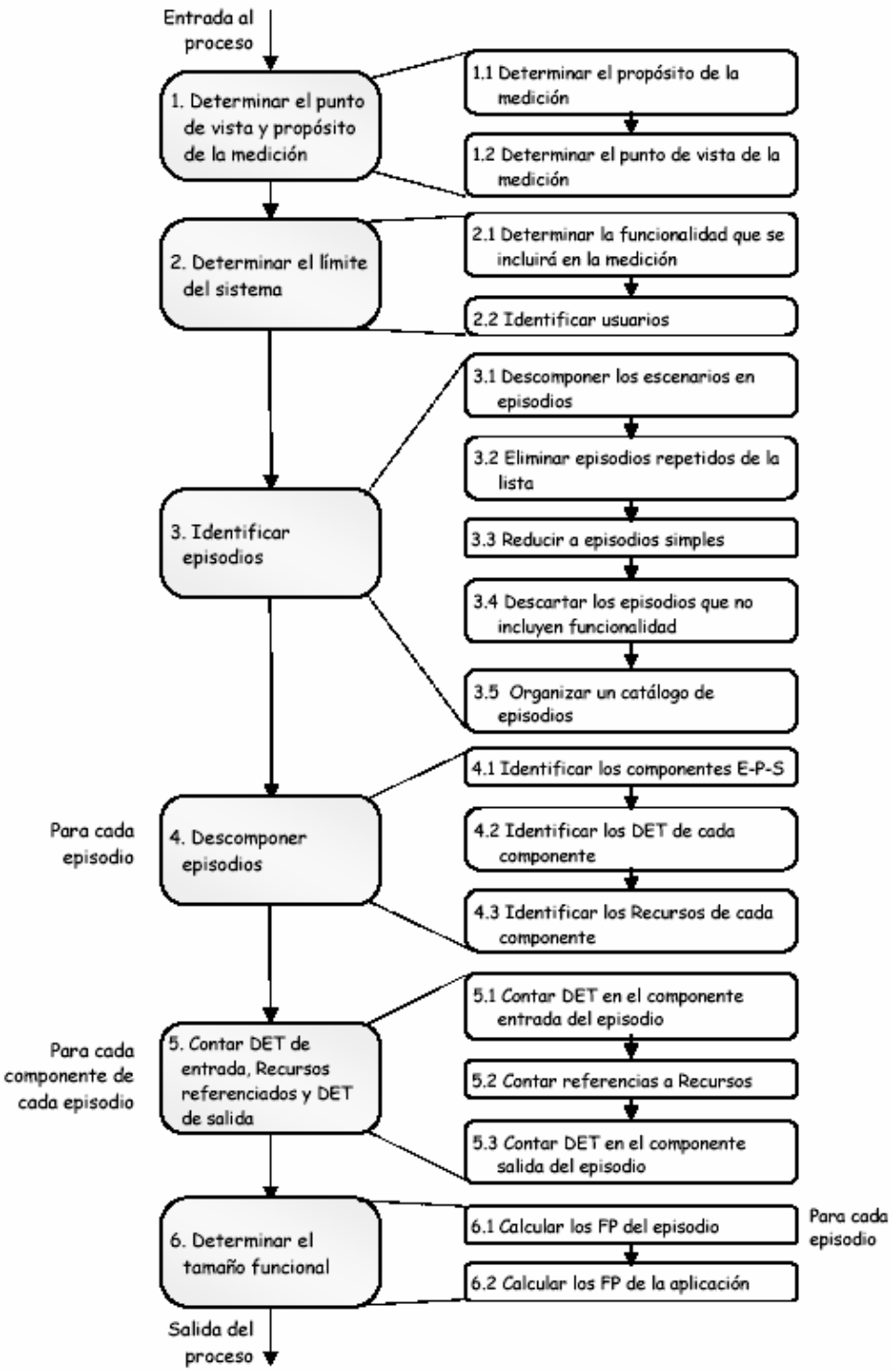


Figura 28 – Etapas del proceso de medición

6.2. Etapas del proceso de medición

Etapa 1 Determinar el punto de vista y propósito de la medición

El proceso de contar FP requiere establecer el *límite de la aplicación* que se va a someter al proceso de medición. Esto servirá para determinar qué parte de la funcionalidad va a estar incluida en el proceso y cuál quedará excluida. La elección del *límite* depende del *punto de vista* y *propósito de la medición* [MKII FPA CPM'98].

Esta etapa requiere:

1.1 Determinar el propósito de la medición

El propósito se determina a partir del objetivo de este trabajo y es: medir la funcionalidad requerida por el usuario a partir de la documentación de L&E.

1.2 Determinar el punto de vista

Es el punto de vista desde el cual se miden los FP. Debido a que el propósito de la medición es medir la funcionalidad entregada al usuario, se considera el punto de vista del usuario.

Para la descripción del proceso, a través de las Etapas 2 a 6 se utiliza el modelo *Entrada-Proceso-Salida,* para representar que en cada etapa se recibe un producto como entrada, se realiza un determinado proceso con/sobre él y se obtiene un nuevo producto como salida.

Etapa 2 Determinar el límite del sistema

Esta etapa requiere los subprocesos:

2.1. Determinar la funcionalidad que se incluirá en la medición

En base al *punto de vista* y *propósito de la medición* se determina incluir todos los escenarios que representan la funcionalidad del sistema desde la visión del usuario. Según la Regla 15 se establece el límite que incluye a todos los escenarios. En la Figura 29 se representa gráficamente.

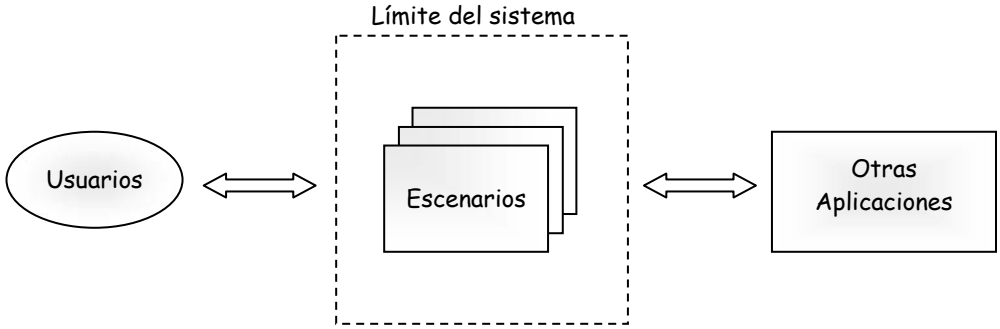


Figura 29 - Límite del sistema

Se describe como:

Entrada: *punto de vista*, *propósito de la medición*, L&E.
Proceso:

Determinar la funcionalidad que se incluirá en la medición considerando la Regla 15.

Salida: el *límite del sistema* que incluye todos los escenarios.

El *límite* trazado implica la existencia de algún mecanismo de comunicación que permita la interacción entre el usuario y el sistema. Las entradas desde el *usuario* cruzan el *límite* para "entrar" a la aplicación, dentro del *límite* se realiza algún

procesamiento y los productos resultantes cruzan el *límite* para alcanzar al *usuario*. Es necesario pues identificar los *usuarios* que interactúan con el sistema.

2.2. Identificar usuarios

En base a la Regla 16 se determina la lista de *usuarios* que interactúan con el sistema a través del *límite del sistema*.

Se describe como:

Entrada: *límite del sistema*, L&E
Proceso:
Identificar *usuarios* desde la documentación de L&E
Salida: lista de *usuarios*

En la próxima etapa se establecen los procesos para identificar los *episodios* que se incluyen en la medición.

Etapa 3 Identificar episodios

Luego de decidir el propósito y elegir el *límite* apropiado, la siguiente decisión más crítica es identificar correctamente el conjunto de *episodios* asociados con los *usuarios* identificados.

Esta etapa requiere los subprocesos:

3.1. Descomponer los escenarios en *episodios*

Los escenarios comprendidos dentro del *límite del sistema* se deben descomponer en los *episodios*. Se requiere analizar cada uno de los escenarios e incorporar cada uno de los *episodios* que no son escenarios a una lista.

Se describe como:

Entrada: escenarios
Proceso:
Generar una lista con todos los *episodios* de todos los escenarios excluyendo los *episodios* que son escenarios según la Regla 1
Salida: lista_de_episodios_candidatos1

Ejemplo [Leite'00]:

– # **notify assistance.**
notify absence.
[ask for equipment.]
if the convoking date was made with anticipation **then remind the meeting**.
[The *secretary* assures that the *equipment* is available for the *meeting date.]*
The *secretary* assures that the *physical space* is available for the *meeting date*. #

Aclaración del ejemplo: representa un grupo de episodios no secuencial que incluye episodios que son escenarios, por lo tanto se excluyen de la lista los cuatro primeros.

3.2. Eliminar episodios repetidos de la lista

De la lista de *episodios* obtenida en la etapa previa se deben eliminar aquellos *episodios* que aparecen repetidos. Esto significa comparar cada uno de los *episodios* con los otros que están en la lista y si representa la misma funcionalidad que otro existente quitarlo de la lista.

Se describe como:

Entrada: lista_de_episodios_candidatos1
Proceso:
Comparar los *episodios* y eliminar de la lista los que están repetidos
Salida: lista_de_episodios_candidatos2

3.3. Reducir a episodios simples

Los *episodios* de la lista de *episodios candidatos* obtenida en la etapa previa deben analizarse según los criterios y reglas establecidos en la sección 5.3.1 y 5.3.2 para determinar su permanencia o no en la lista.

Se describe como:

Entrada: lista_de_episodios_candidatos2
Proceso:
Analizar cada uno de los *episodios* según las Reglas 3 a 7 y determinar si corresponde mantenerlo o no en la lista de *episodios candidatos*
Salida: lista_de_episodios_candidatos3

Ejemplo: continuando con el ejemplo de la Etapa 3.1.

– # **[**The *secretary* assures that the *equipment* is available for the *meeting date*.**]**
The *secretary* assures that the *physical space* is available for the *meeting date*. #

Aclaración del ejemplo: representa un grupo de episodios no secuencial que incluye episodios que son opcionales.

Luego se reduce a los siguientes *episodios simples*:

- The *secretary* assures that the *equipment* is available for the *meeting date*.
- The *secretary* assures that the *physical space* is available for the *meeting date.*

3.4. Descartar los episodios que no incluyen funcionalidad

Se requiere eliminar de la lista de *episodios candidatos* a aquéllos que no aportan funcionalidad.

Se describe como:

Entrada: lista_de_episodios_candidatos3
Proceso:
Analizar los *episodios* según la Regla 2 y descartar los que no incluyen funcionalidad
Salida: lista_de_episodios_candidatos4

Los *episodios* que resultan del ejemplo analizado en la Etapa 3.3 no aportan funcionalidad del sistema. Luego, se eliminan de la lista de *episodios candidatos*.

3.5. Organizar un catálogo de episodios

Un enfoque recomendado cuando se analizan y definen los requerimientos del software es producir versiones refinadas sucesivamente de un catálogo de las *transacciones lógicas* que se requiere que realice la aplicación (o proyecto). En todos los casos los datos provenientes del *usuario* que atraviesan el *límite* para ser procesados [MKII FPA CPM'98]:

- causarán que alguna de las partes de la aplicación cambie de estado de acuerdo a los requerimientos del usuario
  y/o
- causarán que los datos estén disponibles para el usuario.

Por lo tanto, durante las primeras etapas de análisis del problema el catálogo de *transacciones* consistirá de una lista de todos los *eventos* y *consultas* candidatos [MKII FPA CPM'98].

Se describe como:

Entrada: lista_de_episodios_candidatos4
Proceso:

Asignar a cada *episodio*: un código de identificación (ID), un nombre representativo de la acción que realiza y un atributo *evento* (E) o *consulta* (C) según el procesamiento requerido

Salida: un catálogo de N *episodios*

Para ello es útil disponer del Formulario 1 –en la sección 6.3 se detallan los Formularios- para registrar los *episodios*. A continuación se muestra una sección del Formulario 1 como ejemplo.

Escenario	**Episodio**		
	ID	**Nombre**	**E / C**
….	…	…	…
Solicitud de reserva	16	Consultar disponibilidad de habitaciones	C
	17	Registrar la reserva	E
….	…	…	…

Etapa 4 Descomponer episodios

Un paso previo para la realización del cálculo de la funcionalidad de un *episodio* es reconocer sus componentes *E-P-S* y en cada uno de ellos determinar sus *DET* y *recursos* según corresponda.

Se requieren los siguientes subprocesos:

4.1 Identificar los componentes E-P-S

Es necesario analizar la estructura de cada *episodio* para descomponerlo en la parte correspondiente al ingreso de datos desde el usuario, el proceso que se realiza dentro del *límite* y el retorno de información hacia el *usuario*.

Se describe como:

Para cada *episodio* P_i, i=1 a N, N: total de *episodios*

Entrada: un *episodio*
Proceso:
Identificar los componentes *E-P-S* del *episodio* P_i
Salida: lista de componentes del *episodio* P_i

4.2 Identificar los DET de cada componente

Tanto el componente *entrada* como el componente *salida* de un *episodio* incluyen datos que provienen desde o se retornan hacia el *usuario*. Como el tamaño de cada uno de estos componentes es proporcional a la cantidad de *DET*, el primer paso es identificarlos.

Se describe como:

Para cada componente del *episodio* {este subproceso se realiza para el componente *E* y para el componente *S* de cada *episodio* P_i, i=1 a N, N: total de *episodios*}

Entrada: un componente
Proceso:
Identificar *DET* según las Reglas 8 a 11
Salida: lista de *DET* del *episodio* P_i

En el Formulario 2 del *episodio* se registran los *DET* identificados en cada uno de los componentes *E* y *S* del *episodio* P_i

4.3 Identificar los recursos de cada componente

El componente *proceso* de un *episodio* incluye *referencias* a *recursos*. Como el tamaño de este componente es proporcional a la cantidad de *recursos referenciados*, el primer paso es identificar los *recursos*.

Se describe como:

Para cada componente del *episodio* P_i, i=1 a N, N: total de *episodios* {este subproceso se realiza para el componente *P* del *episodio*}

Entrada: un componente
Proceso:
Identificar *recursos* según la Regla 12
Salida: lista de *recursos* del *episodio* P_i

En el Formulario 2 del *episodio* se registran los *recursos* identificados en el componente *P* del *episodio* P_i

Al finalizar la Etapa 4, parte del Formulario 2 de cada *episodio* se habrá completado como se muestra en el ejemplo siguiente.

EPISODIO		**CATEGORÍA**			
ID: 6	**Nombre:** Consultar fichero de pasajeros	**Evento**		**Consulta**	X

DET Entrada	**Recurso referenciado**	**DET Salida**
nombre del *pasajero / huésped / pax*	*fichero de pasajeros*	nombre del *pasajero / huésped / pax*
...	...	...

Etapa 5 Contar DET de entrada, recursos referenciados y DET de salida

Una vez identificados los *DET* y *recursos* en los componentes correspondientes deben contabilizarse.

Se requieren los siguientes subprocesos:

5.1 Contar DET en el componente entrada del episodio

El tamaño del componente *entrada* de un *episodio* es proporcional a la cantidad de *DET*.

Se describe como:

Para cada *episodio* P_i, i=1 a N, N: total de *episodios*

Entrada: lista de *DET* de entrada del *episodio* P_i
Proceso:
Aplicar la Regla 13 a cada uno de los *DET* del *episodio* P_i
Salida: la cuenta N_{I_i} de los *DET* de entrada

Se completa el campo TOTAL de la columna DET ENTRADA del Formulario 2 correspondiente al *episodio* P_i con la cuenta N_{I_i} .

5.2 Contar referencias a recursos

El tamaño del componente *proceso* de un *episodio* es proporcional a la cantidad de *recursos referenciados* durante la realización del mismo.

Se describe como:

Para cada *episodio* P_i, i=1 a N, N: total de *episodios*

Entrada: lista de *recursos* del *episodio* P_i
Proceso:
Aplicar la Regla 14 a cada uno de los *recursos referenciados* del *episodio* P_i
Salida: la cuenta N_{R_i} de los *recursos referenciados*

Se completa el campo TOTAL de la columna RECURSO REFERENCIADO del Formulario 2 correspondiente al *episodio* P_i con la cuenta N_{R_i} .

5.3 Contar DET en el componente salida del episodio

El tamaño del componente *salida* de un *episodio* es proporcional a la cantidad de *DET*.

Se describe como:

Para cada *episodio* P_i, i=1 a N, N: total de *episodios*

Entrada: lista de *DET* de salida del *episodio* P_i
Proceso:
Aplicar la Regla 13 a cada uno de los *DET* del *episodio* P_i
Salida: la cuenta N_{O_i} de los *DET* de salida

Se completa el campo TOTAL de la columna DET SALIDA del Formulario 2 correspondiente al *episodio* P_i con la cuenta N_{O_i}

Al final de la Etapa 5 se habrán completado tantos Formularios 2 como *episodios* haya en el *catálogo de episodios* (si N es la cantidad total de *episodios* en el catálogo habrá N Formularios 2). El Formulario 2 de cada *episodio* se visualizará como en el ejemplo siguiente.

EPISODIO				**CATEGORÍA**			
ID: *6*	**Nombre:** *Consultar fichero de pasajeros*			**Evento**		**Consulta**	*X*
DET Entrada		**Recurso referenciado**		**DET Salida**			
nombre del *pasajero / huésped / pax*		*fichero de pasajeros*		nombre del *pasajero / huésped / pax*			
...		...		...			
TOTAL	*1*	**TOTAL**	*1*	**TOTAL**			*7*

En este punto del proceso se procede al llenado del Formulario 3 con la información asociada a cada *episodio* obtenida desde el Formulario 2 (ID, Nombre, etc.) y se completan las columnas CANTIDAD DET ENTRADA, CANTIDAD REC. REF. y CANTIDAD DET SALIDA de cada uno, según los valores almacenados en los respectivos campos TOTAL de los Formularios 2.

Etapa 6 Determinar el tamaño funcional

El *tamaño funcional* FP de un *episodio* resulta de la suma de los correspondientes N_I , N_R y N_O. Este proceso se realiza para cada *episodio* y la sumatoria de los mismos produce los FP del sistema.

Se requieren los siguientes subprocesos:

6.1 Calcular los FP del episodio

Se describe como:

Para cada *episodio* P_i, i=1 a N, N: total de *episodios*

Entrada: N_{I_i} , N_{R_i} y N_{O_i}

Proceso:

Aplicar la fórmula $FP_i = N_{I_i} + N_{R_i} + N_{O_i}$

Salida: FP_i del *episodio* P_i

Durante este subproceso se completa el campo FP de cada *episodio* P_i en el Formulario 3.

6.2 Calcular los FP de la aplicación

Se describe como:

Entrada: FP_i de todos los *episodios*.

Proceso:

Aplicar la fórmula $FP = \sum_{i=1}^{N} FP_i$

Salida: FP del sistema.

En este punto del proceso se completa el campo TOTAL FP del Formulario 3 y representa la finalización del proceso con la obtención de los FP del sistema.

6.3. Formularios para documentar el proceso

Es fundamental que el proceso de FPA sobre L&E esté debidamente documentado. Es conveniente disponer de formularios estándar para registrar detalladamente el proceso de FPA, que deben adjuntarse a la documentación del proyecto, incluyendo la documentación propia de L&E.

Algunas razones que sustentan la necesidad de documentación:

- Facilitar y organizar el proceso de recolección de datos.
- Favorecer el proceso de revisión y el control de los cambios proporcionando información acerca de qué es lo que se contabilizó durante el análisis.
- Establecer las bases para el análisis del sistema en el futuro.

Dentro de la documentación se debe incluir:

- Toda documentación relacionada con el proyecto.
- La documentación completa de L&E.
- Los formularios propuestos debidamente completados.

6.3.1. Formularios

Se proponen los siguientes formularios:

Formulario 1: CATALOGO DE EPISODIOS.

Formulario 2: DETALLE DEL ANÁLISIS POR EPISODIO.

Formulario 3: RESUMEN DEL CÁLCULO DE LOS FP

En la Tabla 15 se visualiza un resumen del uso de los formularios en cada una de las etapas del proceso. En las intersecciones de la fila Etapa con la columna Formulario se usa la siguiente nomenclatura:

- E: significa que el formulario (en parte o todo) se usa en la etapa como una entrada.
- S: significa que el formulario (en parte o todo) es una salida de la etapa.
- - : etapa y formulario no tienen relación.

		Formulario		
		1	2	3
1	1.1	-	-	-
	1.2	-	-	-
2	2.1	-	-	-
	2.2	-	-	-
3	3.1	-	-	-
	3.2	-	-	-
	3.3	-	-	-
	3.4	-	-	-
	3.5	S	-	-
4	4.1	-	-	-
	4.2	-	S	-
	4.3	-	S	-
5	5.1	-	E	S
	5.2	-	E	S
	5.3	-	E	S
6	6.1	-	-	E
	6.2	-	-	E

Etapa

Tabla 15 – Resumen del uso de formularios por etapa

6.3.2. Normas para completar los formularios

En cada formulario deben completarse todos sus campos.

Para todos los formularios. En el encabezado de cada formulario se debe completar:

1. Título de la aplicación: debe indicarse el nombre de la aplicación.
2. Autor: indicar nombre del autor del proyecto.
3. Etapa del proyecto: indicar etapa del proyecto, por ejemplo, elicitación.
4. Revisor: indicar nombre del revisor del documento.
5. Fecha de revisión: indicar la fecha de realización de la revisión del documento.
6. Nº revisión: indicar el número de revisión del documento.

Formulario 1 CATALOGO DE EPISODIOS. Deben completarse tantos como sean necesarios para incluir todos los *episodios candidatos*, colocando numeración en cada página. Los campos se deben completar como se indica:

1. Escenario: incluir el nombre de cada escenario. Debido a que la cantidad de *episodios* es variable para cada escenario, en cada caso particular deben establecerse las líneas horizontales de separación entre escenarios.
2. Episodio: incluir ID, nombre del *episodio* y E/C según corresponda a *evento* o *consulta*.

Formulario 2 DETALLE DEL ANÁLISIS POR EPISODIO. Debe completarse uno por cada *episodio* del Formulario 1, colocando la correspondiente numeración en cada página. Los campos deben completarse como sigue:

1. Escenario: incluir el nombre del escenario.
2. Episodio: incluir ID y nombre del *episodio*.
3. Clase: incluir X en el campo correspondiente a *Evento* o *Consulta*.
4. DET Entrada: incluir la lista con los nombres de cada *DET* de entrada.
5. Recurso referenciado: incluir la lista con el nombre de cada *recurso* referenciado.
6. DET Salida: incluir la lista con los nombres de cada *DET* de salida.
7. Total: en cada uno de los campos calcular la cuenta de DET Entrada N_{I_i}, Recurso referenciado N_{R_i} y DET Salida N_{O_i}.

Formulario 3 RESUMEN DEL CÁLCULO DE LOS FP. En él deben registrarse los *episodios* analizados en cada uno de los Formularios 2 DETALLE DEL ANÁLISIS POR EPISODIO.

1. Nombre Escenario: indicar el nombre de cada escenario. Líneas horizontales de separación entre escenarios ídem Formulario 1.
2. ID - Nombre episodio: incluir ID y nombre del *episodio*.
3. Evento / Consulta: indicar E o C.
4. Cantidad DET entrada: incluir el valor almacenado en el campo Total de la columna DET Entrada del Formulario 2.
5. Cantidad Rec. ref.: incluir el valor almacenado en el campo Total de la columna Recurso referenciado del Formulario 2.
6. Cantidad DET salida: incluir el valor almacenado en el campo Total de la columna DET Salida del Formulario 2.
7. FP: aplicar la fórmula $FP_i = N_{I_i} + N_{R_i} + N_{O_i}$ a cada *episodio*.
8. Total FP: calcular la suma de la columna FP, que representará los $FP = \sum_{i=1}^{N} FP_i$ de la aplicación.

6.3.3. Plantillas de formularios

Formulario 1	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS** **CATALOGO DE EPISODIOS**		
TÍTULO DE LA APLICACIÓN: **AUTOR:** **ETAPA DEL PROYECTO:**	**REVISOR:** **FECHA DE REVISIÓN:** **Nº REVISIÓN:**		
Escenario	**Episodio**		
	ID	**Nombre**	**E / C**
			Hoja ... de ...

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS** **DETALLE DEL ANÁLISIS POR EPISODIO**

TÍTULO DE LA APLICACIÓN:

AUTOR: **REVISOR:**

ETAPA DEL PROYECTO: **FECHA DE REVISIÓN:**

Nº REVISIÓN:

ESCENARIO:

EPISODIO		**CATEGORÍA**			
ID:	**Nombre:**	**Evento**		**Consulta**	

DET Entrada	**Recurso referenciado**	**DET Salida**
TOTAL	**TOTAL**	**TOTAL**

Hoja ... de ...

Formulario 3	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS** **RESUMEN DEL CALCULO DE LOS FP**

TÍTULO DE LA APLICACIÓN:

AUTOR: **REVISOR:**

ETAPA DEL PROYECTO: **FECHA DE REVISIÓN:**

Nº REVISIÓN:

Nombre Escenario	**Episodio**						
	ID	**Nombre**	**Evento / Consulta**	**Cantidad DET entrada**	**Cantidad Rec. ref.**	**Cantidad DET salida**	**FP**
Total FP							

Capítulo 7

Aplicación del enfoque propuesto

Este capítulo está organizado de la siguiente manera. En la sección 7.1 se presenta en detalle completo la aplicación del enfoque al caso de estudio Recepción del Hotel. En la sección 7.2 se exponen los resultados de su aplicación a otros casos de estudio y se adjunta el correspondiente Formulario 3 de cada caso.

7.1. Aplicación del enfoque propuesto al caso de estudio Recepción del Hotel

Se aplicará el enfoque propuesto para el cálculo de los Puntos Función sobre el L&E desarrollado para el caso de estudio de la Recepción del Hotel.

Etapa 1 Determinar el punto de vista y propósito de la medición

El propósito es medir *la funcionalidad requerida por el usuario* a partir de la documentación de L&E. Se adopta el punto de vista del usuario.

Etapa 2 Determinar el límite del sistema

El límite encierra los siguientes escenarios:	Lista de usuarios:
cancelación de reserva	*Recepcionista*
check in	*Agencia*
check out	otro hotel
pago	*pasajero / huesped / pax*
pedido de alojamiento	Departamento *Mantenimiento*
pedido de servicio extra	Departamento *Mucamas*
recepción del Hotel	Departamento *Administración*
reposición de insumos	Departamento *Compras*
servicio de despertador	
solicitud de reserva	

Etapa 3 Identificar episodios - Subprocesos 3.1 – 3.4

A efectos de visualizar la evolución de la lista de *episodios* se han mantenido los *episodios* que se han descartado y se indican con formato de fuente tachado. A la derecha de cada uno de ellos se indica el motivo de la eliminación usando la siguiente nomenclatura: R: episodio repetido, E: incluye un escenario, N: no incluye funcionalidad de interés, X: excepción, T: restricción.

Escenario	Episodio
cancelación de la reserva	– **if** el *recepcionista* recibe el pedido de anulación de una *solicitud de reserva* o el *pasajero / huesped / pax* no se presenta en el período comprendido entre las 12 hs del *día de ingreso* establecido en la *solicitud de reserva* y las 06 hs. del día siguiente **then** el *recepcionista* elimina la *solicitud de reserva* de la *planilla de reservas*. – El *recepcionista* actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*. – **exception:** El teléfono, el fax o el e-mail no funcionan. X
check in	– **if** el *pasajero / huesped / pax* presenta un *voucher* **then** el *recepcionista* registra los datos incluidos en el mismo y lo almacena en la *carpeta de la habitación*. N – El *recepcionista* busca el nombre del *pasajero / huesped / pax* en la *planilla de reservas*. – **if** el *pasajero / huesped / pax* se presenta por primera vez al Hotel **then** el *recepcionista* le provee la *planilla del pasajero* que debe completar con sus datos y firmar. – El *recepcionista* controla que la *planilla del pasajero* esté completa y la almacena en el *fichero de pasajeros*. – **if** el *pasajero / huesped / pax* ya estuvo en el Hotel **then** el *recepcionista* busca los datos personales por nombre en el *fichero de pasajeros*. – **if** el *pasajero / huesped / pax* tiene acompañantes **then** cada uno debe registrarse en forma individual. R – El *recepcionista* asigna el número de *habitación* al *pasajero / huesped / pax*. N – Al actualizar el *recepcionista* la *disponibilidad de habitaciones*, se actualiza la *planilla de ocupación de habitaciones*. R – El *recepcionista* agrega el número de *habitación* asignado al *pasajero / huesped / pax* en la *planilla de reservas*. – El *recepcionista* hace una *tarjeta codificada* para el número de *habitación*. – El *recepcionista* le provee al *pasajero / huesped / pax* la *tarjeta codificada* de la *habitación* y el control remoto TV. N – **if** el *pasajero / huesped / pax* lo solicita **then** el *recepcionista* le provee la llave del *minibar / frigobar* y/o de la caja de seguridad. N – **restriction:** El *voucher* no cumple con los requisitos. T – **exception:** Faltan tarjetas magnéticas para codificar X – La máquina generadora de tarjetas codificadas no funciona. X
check out	– El *recepcionista* entrega al *pasajero / huesped / pax* junto con la factura: *comprobante de gastos de lavandería*, *comprobante de consumos de cafetería y/o restaurant*, *comprobante de consumos del minibar* y *comprobante de llamados telefónicos*. N – El *recepcionista* agrega el sello de Pagado en la factura – El *recepcionista* envía la copia de la factura del *pasajero / huesped / pax* al Departamento *Administración*. N – El *recepcionista* recibe la *tarjeta codificada* y el control remoto TV y la llave del *minibar / frigobar* y/o caja de seguridad si estuviera en poder del *pasajero / huesped / pax*. N – Al actualizar el *recepcionista* la *disponibilidad de habitaciones*, se actualiza la *planilla de ocupación de habitaciones*. R – **restriction:** El *recepcionista* no recibe el *pago* porque el *pasajero / huesped / pax* no cuenta con los medios necesarios. T

pago	~~– **if** el *pasajero / huesped / pax* paga con dinero en efectivo o con cheque **then** el *recepcionista* recibe el dinero o el cheque y lo guarda en la caja de la *Recepción*.~~ N ~~– **if** el *pasajero / huesped / pax* paga con tarjeta de crédito **then** el *recepcionista* solicita la autorización al emisor de la tarjeta de crédito.~~ N – **if** el *pasajero / huesped / pax* está habilitado **then** el *recepcionista* hace el cupón con el importe correspondiente, el *pasajero / huesped / pax* lo firma, se guarda en la *Recepción* y se le provee la copia. ~~– **if** el *pasajero / huesped / pax* pertenece a una empresa que tiene cuenta corporativa en el Hotel **then** firma la factura y se envía a la *Administración*.~~ N
pedido de alojamiento	~~– El *recepcionista* consulta la *planilla de ocupación de habitaciones* para conocer la *disponibilidad de habitaciones*.~~ R ~~– **if** existe *disponibilidad de habitaciones* y la persona estuviera de acuerdo **then** el *recepcionista* registra los datos personales del *pasajero / huesped / pax* en la *planilla de reservas*.~~ R ~~– **restriction:** No hay *disponibilidad de habitaciones*~~ T
pedido de servicio extra	~~– El *recepcionista* registra un *pedido de limpieza extra o pedido de mantenimiento extra* que hace el *pasajero / huesped / pax* para su *habitación*.~~ R – El *recepcionista* registra el número de *habitación* y el tipo de servicio a efectuar. (reparaciones o limpieza extra) ~~– El *recepcionista* hace la lista de los pedidos de reparaciones o limpieza extra.~~ R ~~– El *recepcionista* envía la lista al Departamento *Mantenimiento* o *Mucamas*.~~ N ~~– **restriction:** No se registró el número de *habitación* en la lista de pedidos de reparaciones o limpieza extra.~~ T ~~– No se registró el tipo de servicio.~~ T
reposición de insumos	– El *recepcionista* hace una lista de insumos faltantes. ~~– El *recepcionista* envía el pedido al Departamento *Compras*.~~ N ~~– El *recepcionista* recibe el pedido que envía el Departamento *Compras*.~~ N ~~– El *recepcionista* firma el conforme en el recibo.~~ N ~~– **restriction:** No se registraron los insumos faltantes~~ T
servicio de despertador	– El *recepcionista* registra el número de *habitación* y la hora solicitada por el *pasajero / huesped / pax*. – El *recepcionista* llama por teléfono al *pasajero / huesped / pax* cuando llega la hora. ~~– **restriction:** El *recepcionista* no consulta la lista de pedidos de despertador.~~ T
solicitud de reserva	– El *recepcionista* verifica en la *planilla de ocupación de habitaciones* la *disponibilidad de habitaciones* entre *día de ingreso* y *día de egreso* y si hubiera, informa la *tarifa* y solicita la aprobación de la persona, *agencia* o de otro hotel. – El *recepcionista* registra el nombre del *pasajero / huesped / pax*, cantidad de pasajeros, tipo y modalidad de *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas* ~~– El *recepcionista* actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*.~~ R ~~– **restriction:** No hay *disponibilidad de habitaciones*~~ T ~~– **exception:** El teléfono, el fax o el e-mail no funcionan.~~ X
recepción del Hotel (general)	~~– Si el *pasajero / huesped / pax* lo solicita entonces **pedido de servicio extra**~~ E ~~– Si el *pasajero / huesped / pax* lo solicita entonces **servicio de despertador**~~ E ~~– La *carpeta de la habitación* se envía a la *Administración* junto con las fechas de ingreso y egreso, para la emisión de la *facturación*.~~ N

Etapa 3 - Subproceso 3.5 Catálogo de episodios

En el Formulario 1 se organiza el *catálogo de episodios*.

Formulario 1	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS CATALOGO DE EPISODIOS**		
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *6/12/02* **Nº REVISIÓN:** *1*			
Escenario	**Episodio**		
	ID	**Nombre**	**E / C**
Cancelación de la reserva	1	Cancelar la reserva (explícita o no show)	E
	2	Actualizar planilla de ocupación de habitaciones	E
Check in	3	Consultar planilla de reservas	C
	4	Completar planilla del pasajero	E
	5	Actualizar fichero de pasajeros	E
	6	Consultar fichero de pasajeros	C
	7	Actualizar planilla de reservas	E
	8	Generar tarjeta codificada	E
Check out	9	Sellar la factura	E
Pago	10	Confeccionar cupón de tarjeta de crédito	E
Pedido de servicio extra	11	Registrar pedido de limpieza extra	E
	12	Registrar pedido de mantenimiento extra	E
Reposición de insumos	13	Generar lista de pedidos de insumos	E
Servicio de despertador	14	Registrar pedido de despertador	E
	15	Activar despertador	E
Solicitud de reserva	16	Consultar disponibilidad de habitaciones	C
	17	Registrar la reserva	E
Hoja 1 de 1			

Etapas 4 y 5 – Descomponer episodios y Contar DET de entrada, recursos referenciados y DET de salida

Antes de estudiar cada *episodio* vale desarrollar la siguiente fundamentación para el criterio adoptado en aquéllos casos en que el *DET* de salida está representado por el conjunto de datos necesarios para actualizar un *recurso*. Si la *transacción lógica* procesa los datos como un único *DET*, sin importar su estructura, hay un único *DET*: contar 1 de entrada o salida. Ejemplo: archivar registros de datos. La estructura del registro puede comprender muchos *DET*, pero el registro completo podría ser almacenado sin validación o cambios al formato [MKII FPA CPM'98]. Por lo tanto, se va a considerar un único *DET* de salida representado por un conjunto de elementos que se denominará *"vector de actualización"*.

***1. Cancelar la reserva**: la cancelación por no show se dispara automáticamente al no presentarse el pasajero el día de ingreso a la hora 06. Según la Regla 11 ambos son DET de entrada. Éstos junto con el nombre del pasajero componen los DET de entrada. En el proceso se elimina la reserva de la planilla de reservas (1 recurso). Para la salida se consideran los datos de interés para el usuario, que en este caso es el vector que actualiza la planilla de reservas:* tipo de habitación + modalidad de habitación + día de ingreso + día de egreso.

Observación: la cancelación por pedido explícito del pasajero, si bien es una función del sistema, a los efectos del cálculo de funcionalidad no se contabiliza pues resultaría en una duplicación de FP, debido a que tiene un DET de entrada que es el nombre del pasajero y su aporte en FP es menor.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS** **DETALLE DEL ANÁLISIS POR EPISODIO**

TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*

AUTOR: *Mabel* **REVISOR:** *Mabel*

ETAPA DEL PROYECTO: *Elicitación* **FECHA DE REVISIÓN:** *14/01/03*

Nº REVISIÓN: *2*

ESCENARIO: **cancelación de la reserva**

EPISODIO		**CATEGORÍA**			
ID: *1*	**Nombre:** *Cancelar la reserva*	**Evento**	*X*	**Consulta**	

DET Entrada	**Recurso referenciado**	**DET Salida**
nombre del *pasajero / huesped / pax*	*planilla de reservas*	tipo de *habitación* + modalidad de *habitación* + *día de ingreso* + *día de egreso*
día de ingreso		
hora		
TOTAL *3*	**TOTAL** *1*	**TOTAL** *1*

Hoja 1 de 17

2. Actualizar planilla de ocupación de habitaciones: *los DET de entrada son: tipo de habitación, modalidad de habitación, día de ingreso, día de egreso, el recurso es la planilla de ocupación de habitaciones y el DET de salida es el vector que actualiza la planilla de ocupación de habitaciones:* tipo de habitación + modalidad de habitación.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*						
ESCENARIO: **cancelación de la reserva**						
EPISODIO			**CATEGORÍA**			
ID: *2*	**Nombre:** *Actualizar planilla de ocupación de habitaciones*		**Evento**	*X*	**Consulta**	
DET Entrada		**Tipo de recurso referenciado**		**DET Salida**		
tipo de *habitación*		*planilla de ocupación de habitaciones*		tipo de *habitación* + modalidad de *habitación*		
modalidad de *habitación*						
día de ingreso						
día de egreso						
TOTAL	*4*	**TOTAL**	*1*	**TOTAL**	*1*	
Hoja 2 de 17						

3. Consultar planilla de reservas: *el DET de entrada es el nombre del pasajero, el recurso es la planilla de reservas y como DET de salida los datos asociados al nombre del pasajero almacenados en dicha planilla: nombre del pasajero, tipo y modalidad de habitación, cantidad de pasajeros, día de ingreso, día de egreso, teléfono, tarifa.*

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**

TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*
AUTOR: *Mabel* **REVISOR:** *Mabel*
ETAPA DEL PROYECTO: *Elicitación* **FECHA DE REVISIÓN:** *8/12/02*
Nº REVISIÓN: *1*

ESCENARIO: **check in**

EPISODIO		**CATEGORÍA**			
ID: *3*	**Nombre:** *Consultar planilla de reservas*	**Evento**		**Consulta**	X

DET Entrada	**Recurso referenciado**	**DET Salida**
nombre del *pasajero / huesped / pax*	*planilla de reservas*	nombre del *pasajero / huesped / pax*
		tipo de *habitación*
		modalidad de *habitación*
		cantidad de pasajeros
		día de ingreso
		día de egreso
		teléfono
		tarifa
TOTAL *1*	**TOTAL** *1*	**TOTAL** *8*

Hoja 3 de 17

4. Completar planilla del pasajero: *los DET de entrada son: nombre del pasajero, número de documento, datos comerciales, firma, día de ingreso, procedencia del pasajero y datos del vehículo, el recurso es la planilla del pasajero y el DET de salida es el vector que actualiza la planilla del pasajero:* nombre del pasajero + número de documento + día de ingreso + procedencia + datos vehículo + datos comerciales + firma.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **REVISOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*						
ESCENARIO: **check in**						
EPISODIO			**CATEGORÍA**			
ID: 4	**Nombre:** *Completar planilla del pasajero*		**Evento**	X	**Consulta**	

DET Entrada		**Recurso referenciado**		**DET Salida**	
nombre del *pasajero / huesped / pax*		*planilla del pasajero*		nombre del *pasajero / huesped / pax* + número de documento + *día de ingreso* + procedencia + datos vehículo + datos comerciales + firma	
número de documento					
día de ingreso					
procedencia					
datos vehículo					
datos comerciales					
firma					
TOTAL	*7*	**TOTAL**	*1*	**TOTAL**	*1*

Hoja 4 de 17

5. Actualizar fichero de pasajeros: *los DET de entrada son: nombre del pasajero, número de documento, datos comerciales, firma, día de ingreso, procedencia del pasajero y datos del vehículo, los recursos son: planilla del pasajero y fichero de pasajeros y el DET de salida es el vector que actualiza el fichero de pasajeros:* nombre del pasajero + número de documento + día de ingreso + procedencia + datos vehículo + datos comerciales + firma.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*						
AUTOR: *Mabel*			**REVISOR:** *Mabel*			
ETAPA DEL PROYECTO: *Elicitación*			**FECHA DE REVISIÓN:** *14/01/03*			
			Nº REVISIÓN: *2*			
ESCENARIO: **check in**						
EPISODIO			**CATEGORÍA**			
ID: *5*	**Nombre:** *Actualizar fichero de pasajeros*		**Evento**	*X*	**Consulta**	

DET Entrada		**Recurso referenciado**		**DET Salida**	
nombre del *pasajero / huesped / pax*		*fichero de pasajeros*		nombre del *pasajero / huesped / pax* + número de documento + *día de ingreso* + procedencia + datos vehículo + datos comerciales + firma	
número de documento		*planilla del pasajero*			
día de ingreso					
procedencia					
datos vehículo					
datos comerciales					
firma					
TOTAL	*7*	**TOTAL**	*2*	**TOTAL**	*1*

Hoja 5 de 17

6. Consultar fichero de pasajeros: *el DET de entrada es nombre del pasajero, el recurso es el fichero de pasajeros y como DET de salida los datos asociados al nombre del pasajero almacenados en el fichero de pasajeros: nombre del pasajero, número de documento, datos comerciales, firma, día de ingreso, procedencia del pasajero y datos del vehículo.*

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**

TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*
AUTOR: *Mabel* **REVISOR:** *Mabel*
ETAPA DEL PROYECTO: *Elicitación* **FECHA DE REVISIÓN:** *8/12/02*
Nº REVISIÓN: *1*

ESCENARIO: **check in**

EPISODIO		**CATEGORÍA**			
ID: *6*	**Nombre:** *Consultar fichero de pasajeros*	**Evento**		**Consulta**	*X*

DET Entrada	**Recurso referenciado**	**DET Salida**
nombre del *pasajero / huesped / pax*	*fichero de pasajeros*	nombre del *pasajero / huesped / pax*
		número de documento
		día de ingreso
		procedencia
		datos vehículo
		datos comerciales
		firma
TOTAL *1*	**TOTAL** *1*	**TOTAL** *7*

Hoja 6 de 17

7. Actualizar planilla de reservas: *en este caso se trata de agregar a la planilla de reservas el número de habitación asignado al pasajero en el momento del check in. El DET de entrada es el número de habitación, el recurso la planilla de reservas y el DET de salida es el único DET que actualiza la planilla de reservas:* número de habitación.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación*				**REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*		
ESCENARIO: **check in**						
EPISODIO			**CATEGORÍA**			
ID: *7*	**Nombre:** *Actualizar planilla de reservas*		**Evento**		**Consulta**	*X*
DET Entrada		**Recurso referenciado**		**DET Salida**		
número de *habitación*		*planilla de reservas*		número de *habitación*		
TOTAL	*1*	**TOTAL**	*1*	**TOTAL**		*1*
Hoja 7 de 17						

8. Generar tarjeta codificada: *para el número de habitación se requiere crear una tarjeta que permita el acceso a la misma. Luego el DET de entrada es el número de habitación asignado al pasajero. El recurso es una tarjeta magnética en blanco y el DET de salida es el código generado sobre la tarjeta.*

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación*				**REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *8/12/02* **Nº REVISIÓN:** *1*		
ESCENARIO: **check in**						
EPISODIO			**CATEGORÍA**			
ID: *8*	**Nombre:** *Generar tarjeta codificada*		**Evento**	*X*	**Consulta**	
DET Entrada		**Recurso referenciado**		**DET Salida**		
número de *habitación*		tarjeta en blanco		código en tarjeta		
TOTAL	*1*	**TOTAL**	*1*	**TOTAL**		*1*
Hoja 8 de 17						

9. Sellar la factura: *cuando el pasajero realiza el pago se le entrega una factura sellada. Luego el DET será el número de factura, los recursos son la factura y el sello y el DET de salida es el sellado sobre la factura.*

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**

TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*
AUTOR: *Mabel* **REVISOR:** *Mabel*
ETAPA DEL PROYECTO: *Elicitación* **FECHA DE REVISIÓN:** *8/12/02*
Nº REVISIÓN: *1*

ESCENARIO: **check out**

EPISODIO		**CATEGORÍA**			
ID: 9	**Nombre:** *Sellar la factura*	**Evento**	X	**Consulta**	

DET Entrada	**Recurso referenciado**	**DET Salida**
número de factura	factura	sellado en factura
	Sello de Pagado	
TOTAL 1	**TOTAL** 2	**TOTAL** 1

10. Confeccionar cupón de tarjeta de crédito: *en este caso se requiere el número de la tarjeta de crédito, el nombre del titular y el importe a pagar, que son los DET de entrada. Los recursos son el cupón en blanco y la tarjeta de crédito y el DET de salida son esos mismos datos impresos en el cupón.*

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*						
ESCENARIO: **pago**						
EPISODIO			**CATEGORÍA**			
ID: *10*	**Nombre:** *Confeccionar cupón de tarjeta de crédito*		**Evento**	*X*	**Consulta**	
DET Entrada		**Recurso referenciado**		**DET Salida**		
número de tarjeta de crédito		cupón de tarjeta de crédito en blanco		número de tarjeta de crédito + nombre del titular + importe		
nombre del titular		tarjeta de crédito				
importe						
TOTAL	*3*	**TOTAL**	*2*	**TOTAL**		*1*
Hoja 10 de 17						

11. Registrar pedido de limpieza extra: *se requiere el número de habitación y el tipo de servicio que son los DET de entrada. El recurso es la lista de pedidos de limpieza extra y los DET de salida es el vector que actualiza la lista de pedidos de limpieza extra*: número de habitación + tipo de servicio.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*						
ESCENARIO: **pedido de servicio extra**						
EPISODIO			**CATEGORÍA**			
ID: *11*	**Nombre:** *Registrar pedido de limpieza extra*		**Evento**	*X*	**Consulta**	
DET Entrada		**Recurso referenciado**		**DET Salida**		
número de *habitación*		lista de *pedido de limpieza extra*		número de *habitación* + tipo de servicio		
tipo de servicio						
TOTAL	*2*	**TOTAL**	*1*	**TOTAL**	*1*	
Hoja 11 de 17						

12. Registrar pedido de mantenimiento extra: *se requiere el número de habitación y el tipo de servicio que son los DET de entrada. El recurso es la lista de pedidos de mantenimiento extra y el DET de salida es el vector que actualiza la lista de pedidos de mantenimiento extra:* número de habitación + tipo de servicio.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*						
ESCENARIO: **pedido de servicio extra**						
EPISODIO			**CATEGORÍA**			
ID: *12*	**Nombre:** *Registrar pedido de mantenimiento extra*		**Evento**	*X*	**Consulta**	
DET Entrada		**Recurso referenciado**		**DET Salida**		
número de *habitación*		lista de *pedido de mantenimiento extra*		número de *habitación* + tipo de servicio		
tipo de servicio						
TOTAL	*2*	**TOTAL**	*1*	**TOTAL**	*1*	
Hoja 12 de 17						

13. Generar lista de pedidos de insumos: *se requiere el nombre y la cantidad de los insumos que se deben reponer y estos son los DET de entrada. El recurso es la lista de pedidos de insumos y el DET de salida es el vector que actualiza la lista de pedidos de insumos:* nombre insumos + cantidad.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*						
AUTOR: *Mabel*			**REVISOR:** *Mabel*			
ETAPA DEL PROYECTO: *Elicitación*			**FECHA DE REVISIÓN:** *14/01/03*			
			Nº REVISIÓN: *2*			
ESCENARIO: **reposición de insumos**						
EPISODIO			**CATEGORÍA**			
ID: *13*	**Nombre:** *Generar lista de pedidos de insumos*		**Evento**	*X*	**Consulta**	
DET Entrada		**Recurso referenciado**		**DET Salida**		
nombre insumos		lista de pedido de insumos		nombre insumos + cantidad		
cantidad						
TOTAL	*2*	**TOTAL**	*1*	**TOTAL**	*1*	

14. Registrar pedido de despertador: *en este caso los DET de entrada son el número de habitación del pasajero y la hora para la que solicita el servicio. El recurso es la lista de pedido de servicio de despertador y el DET de salida es el vector que actualiza la lista de pedidos de servicio de despertador:* número de habitación + hora.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación*				**REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*		
ESCENARIO: **servicio de despertador**						
EPISODIO			**CATEGORÍA**			
ID: *14*	**Nombre:** *Registrar pedido de despertador*		**Evento**	*X*	**Consulta**	
DET Entrada		**Tipo de recurso referenciado**		**DET Salida**		
número de *habitación*		lista de *pedido de servicio de despertador*		número de *habitación* + hora		
hora						
TOTAL	*2*	**TOTAL**	*1*	**TOTAL**	*1*	
Hoja 14 de 17						

15. Activar despertador: *en este caso el DET de entrada es la hora. Los recursos son la lista de pedido de servicio de despertador y el teléfono y el DET de salida es la emisión sonora del teléfono.*

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**

TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*
AUTOR: *Mabel* **REVISOR:** *Mabel*
ETAPA DEL PROYECTO: *Elicitación* **FECHA DE REVISIÓN:** *9/12/02*
Nº REVISIÓN: *1*

ESCENARIO: **servicio de despertador**

EPISODIO		**CATEGORÍA**			
ID: *15*	**Nombre:** *Activar despertador*	**Evento**	*X*	**Consulta**	

DET Entrada	**Tipo de recurso referenciado**	**DET Salida**
hora	teléfono	emisión sonora del llamado telefónico
	lista de *pedido de servicio de despertador*	
TOTAL *1*	**TOTAL** *2*	**TOTAL** *1*

Hoja 15 de 17

16. Consultar disponibilidad de habitaciones: *en este caso los DET de entrada son: tipo y modalidad de las habitaciones (que surgen desde la disponibilidad de habitaciones) y fechas de ingreso y egreso. Los recursos son: planilla de ocupación de habitaciones y lista de precios y el DET de salida es tarifa.*

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**					
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **REVISOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **FECHA DE REVISIÓN:** *7/12/02* **Nº REVISIÓN:** *1*						
ESCENARIO: **solicitud de reserva**						
EPISODIO			**CATEGORÍA**			
ID: *16*	**Nombre:** *Consultar disponibilidad de habitaciones*		**Evento**		**Consulta**	*X*
DET Entrada		**Recurso referenciado**		**DET Salida**		
tipo de *habitación*		*planilla de ocupación de habitaciones*		cantidad de habitaciones de un tipo y modalidad		
modalidad de *habitación*		*lista de precios*		*tarifa*		
día de ingreso						
día de egreso						
TOTAL	*4*	**TOTAL**	*2*	**TOTAL**		*2*
Hoja 16 de 17						

17. Registrar la reserva: *en este caso los DET de entrada son los datos requeridos para completar la planilla de reservas: datos del pasajero, fechas de ingreso y egreso, tipo y modalidad de habitaciones y tarifa. El recurso es la planilla de reservas y el DET de salida es el vector que actualiza la planilla de reservas:* nombre del pasajero + cantidad de pasajeros + tipo de habitación + modalidad de habitación + día de ingreso + día de egreso + tarifa + teléfono del pasajero.

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS DETALLE DEL ANÁLISIS POR EPISODIO**

TÍTULO DE LA APLICACIÓN: *Recepción del Hotel*
AUTOR: *Mabel* **REVISOR:** *Mabel*
ETAPA DEL PROYECTO: *Elicitación* **FECHA DE REVISIÓN:** *14/01/03*
Nº REVISIÓN: *2*

ESCENARIO: **solicitud de reserva**

EPISODIO		**CATEGORÍA**			
ID: *17*	**Nombre:** *Registrar la reserva*	**Evento**	*X*	**Consulta**	

DET Entrada	**Recurso referenciado**	**DET Salida**
nombre del *pasajero / huesped / pax*	*planilla de reservas*	nombre del *pasajero / huesped / pax* + cantidad de pasajeros + tipo de *habitación* + modalidad de *habitación* + *día de ingreso* + *día de egreso* + *tarifa* + teléfono del *pasajero / huesped / pax*
cantidad de pasajeros		
tipo de *habitación*		
modalidad de *habitación*		
día de ingreso		
día de egreso		
tarifa		
teléfono del *pasajero / huesped / pax*		
TOTAL *8*	**TOTAL** *1*	**TOTAL** *1*

Hoja 17 de 17

Etapa 6 - Determinar el tamaño funcional

En el Formulario 3 se realiza el cálculo de los FP de los *episodios* y el total de FP de la aplicación.

Formulario 3	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS RESUMEN DEL CÁLCULO DE LOS FP**						
TÍTULO DE LA APLICACIÓN: *Recepción del Hotel* **AUTOR:** *Mabel* **ETAPA DEL PROYECTO:** *Elicitación* **REVISOR:** *Mabel* **FECHA DE REVISIÓN:** *14/01/03* **Nº REVISIÓN:** *2*							
Nombre Escenario	**Episodio**						
	ID	**Nombre**	**Evento / Consulta**	**Cantidad DET entrada**	**Cantidad Rec. ref.**	**Cantidad DET salida**	**FP**
Cancelación de la reserva	**1**	***cancelar la reserva***	E	3	1	1	5
	2	***actualizar planilla de ocupación de habitaciones***	E	4	1	1	6
Check in	**3**	***consultar planilla de reservas***	C	1	1	8	10
	4	***completar planilla del pasajero***	E	7	1	1	9
	5	***actualizar fichero de pasajeros***	E	7	2	1	10
	6	***consultar fichero de pasajeros***	C	1	1	7	9
	7	***actualizar planilla de reservas***	E	1	1	1	3
	8	***generar tarjeta codificada***	E	1	1	1	3
Check out	**9**	***sellar la factura***	E	1	2	1	4
Pago	**10**	***confeccionar cupón de tarjeta de crédito***	E	3	2	1	6
Pedido de servicio extra	**11**	***registrar pedido de limpieza extra***	E	2	1	1	4
	12	***registrar pedido de mantenimiento extra***	E	2	1	1	4
Reposición de insumos	**13**	***generar lista de insumos***	E	2	1	1	4
Servicio de despertador	**14**	***registrar pedido de despertador***	E	2	1	1	4
	15	***activar despertador***	E	1	2	1	4

Solicitud de reserva	**16**	***consultar disponibilidad de habitaciones***	C	4	2	2	8
	17	***registrar la reserva***	E	8	1	1	10
Total FP							**103**

Resumiendo, se analizaron 8 escenarios y del total de *episodios* quedaron 17, a partir de los cuales se obtuvieron 103 FP.

7.2. Aplicación a otros casos

A continuación se enumeran los casos de estudio a los que se aplicó el enfoque propuesto para calcular los FP a partir de la documentación de L&E y los resultados obtenidos.

- *Sistema nacional de obtención de pasaportes* [Leite'96]: se analizaron 14 escenarios y 24 episodios, obteniéndose 128 FP.
- *Administradora de Planes de Ahorro para automóviles* [Mauco'97]: de 12 escenarios y 23 episodios, resultaron 79 FP.
- *Estación de servicio* [Esteban'98]: de 17 escenarios y 45 episodios, resultaron 219 FP.

La Tabla 16 muestra un resumen de los casos de estudio y los resultados de la medición:

Caso	Escenarios	Episodios	FP	FP/Escenario	FP/Episodio
Recepción del Hotel	8	17	103	12,88	6,06
A. de Planes de ahorro para automóviles	12	23	79	6,58	3,43
S. N. de obtención de pasaportes	14	24	128	9,14	5,33
Estación de servicio	17	45	219	12,88	4,87

Tabla 16 - Cuadro resumen de los casos de estudio analizados

En la Figura 30 se muestran las relaciones Escenarios - FP y Episodios - FP para los casos de estudio de la Tabla 16, incluyendo la recta de regresión lineal en ambos casos.

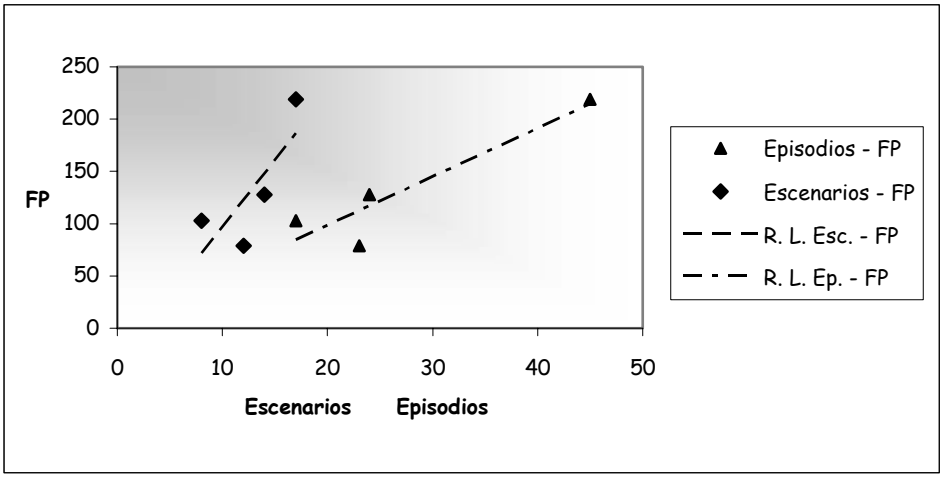


Figura 30 – Gráfico de los casos de estudio de la Tabla 16

En las páginas siguientes se presentan los formularios conteniendo la información detallada correspondiente a los tres casos de estudio mencionados.

Formulario 3	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS** **RESUMEN DEL CALCULO DE LOS FP**

TÍTULO DE LA APLICACIÓN: *Sistema nacional de obtención de pasaportes*

AUTOR:
ETAPA DEL PROYECTO: *Elicitación*

REVISOR: *Mabel*
FECHA DE REVISIÓN: *14/01/03*
Nº REVISIÓN: *2*

Nombre Escenario	Episodio						
	ID	**Nombre episodio**	**Evento / Consulta**	**Cantidad DET entrada**	**Cantidad Rec. ref.**	**Cantidad DET salida**	**FP**
Entrega y llenado de formularios vacíos	**1**	***completar el formulario requerido***	E	9	1	1	11
	2	***completar el formulario de donación de órganos***	E	1	1	1	3
Cobrar trámite	**3**	***consultar el importe***	C	1	1	1	3
	4	***pagar el importe***	E	1	2	1	4
Obtención de huellas digitales para control	**5**	***tomar huellas digitales del pulgar derecho***	E	2	1	1	4
Obtención de huellas digitales para legajo	**6**	***tomar todas las huellas digitales***	E	1	2	1	4
Sacar fotografía	**7**	***cobrar importe fotografía***	E	1	2	2	5
	8	***tomar fotografía***	E	1	1	1	3
Derivación a Cabina de Recepción	**9**	***controlar formulario***	E	4	1	1	6
Recepción y armado de pasaporte original	**10**	***completar pasaporte***	E	10	4	1	15
	11	***registrar número de control y nombre del solicitante***	E	2	1	1	4
	12	***confirmar opción en el formulario de donación de órganos***	E	1	3	1	5

Control de prontuario	**13**	***verificar número de identificación***	C	1	1	1	3
	14	***verificar prontuario***	C	1	2	1	4
	15	***verificar datos del legajo***	C	1	6	1	8
	16	***completar formulario*** *(casillero Índice general)*	E	1	2	1	4
Control dactiloscópico	**17**	***verificar huellas digitales***	C	1	3	1	5
	18	***archivar ficha dactiloscópica***	E	1	3	1	5
	19	***completar formulario*** *(casillero Dactiloscopía)*	E	1	2	2	5
Envío a incineración de pasaportes no retirados	**20**	***incinerar pasaportes***	E	1	3	1	5
Derivación de pasaportes observados	**21**	***observar pasaporte***	E	1	2	2	5
Recepción para reválida de pasaporte sin renovación total	**22**	***registrar número de control***	E	1	4	1	6
Inicio del trámite de reválida telefónica	**23**	***registrar número de identificación y asignar código***	E	1	2	3	6
Control de documentación	**24**	***registrar tipo de trámite***	E	1	3	1	5
Total FP							**128**

Formulario 3	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS** **RESUMEN DEL CALCULO DE LOS FP**

TÍTULO DE LA APLICACIÓN: *Administradora de Planes de Ahorro para automóviles*

AUTOR:
ETAPA DEL PROYECTO: *Elicitación*

REVISOR: *Mabel*
FECHA DE REVISIÓN: *14/01/03*
Nº REVISIÓN: *2*

Nombre Escenario	**Episodio**						
	ID	**Nombre episodio**	**Evento / Consulta**	**Cantidad DET entrada**	**Cantidad Rec. ref.**	**Cantidad DET salida**	**FP**
Adjudicación	**1**	***pagar derecho de adjudicación***	E	1	1	1	3
	2	***llenar un formulario de pedido de vehículo***	E	2	1	1	4
Armar un grupo	**3**	***armar lista de miembros del nuevo grupo***	E	4	1	1	6
Asegurar bien tipo	**4**	***elegir aseguradora***	E	1	1	1	3
	5	***endosar la póliza***	E	1	1	1	3
Cambiar bien tipo adjudicado	**6**	***presentar solicitud cambio de bien***	E	1	1	1	3
	7	***pagar diferencia***	E	2	1	1	4
	8	***acreditar diferencia***	E	2	1	1	3
Disolver un grupo	**9**	***devolver el importe por renuncia al grupo***	E	1	2	1	4
	10	***acreditar el importe por disolución del grupo***	E	1	1	1	3
Evaluar licitaciones	**11**	***crear ranking de adherentes***	E	2	1	1	4
	12	***asignar orden de sorteo a licitación con el mismo importe***	E	2	1	1	4
Licitar	**13**	***devolver diferencia***	E	2	1	1	4

Pagar cuota	**14**	***pagar cuota***	E	1	1	1	3
Pagar cuota vencida	**15**	***pagar cuota vencida***	E	1	1	1	3
Solicitar ingreso	**16**	***llenar planilla de adhesión***	E	1	1	1	3
	17	***pagar cuota de ingreso***	E	1	1	1	3
	18	***devolver cuota de ingreso por rechazo***	E	1	1	1	3
Sorteo	**19**	***asignar orden al azar***	E	1	1	1	3
	20	***verificar cuotas al día***	C	1	1	1	3
	21	***tomar al siguiente sorteado***	E	1	1	1	3
Transferir plan	**22**	***dar de baja en plan de ahorro***	E	1	1	1	3
	23	***dar de alta en plan de ahorro***	E	1	1	1	3
Total FP							**79**

Formulario 2	**ANÁLISIS DE PUNTOS FUNCIÓN SOBRE LEL Y ESCENARIOS** **RESUMEN DEL CALCULO DE LOS FP**

TÍTULO DE LA APLICACIÓN: *Estación de servicio*

AUTOR:
ETAPA DEL PROYECTO: *Elicitación*

REVISOR: *Mabel*
FECHA DE REVISIÓN: *14/01/03*
Nº REVISIÓN: *2*

Nombre Escenario	Episodio						
	ID	Nombre episodio	Evento / Consulta	Cantidad DET entrada	Cantidad Rec. ref.	Cantidad DET salida	FP
Vender combustible	**1**	***despachar combustible***	E	2	2	1	5
Cobro de la venta con tarjeta	**2**	***pasar tarjeta de crédito por máquina***	E	1	3	1	5
	3	***completar comprobante de pago***	E	6	1	1	8
	4	***archivar copia de comprobante de pago***	E	1	2	1	4
Cobro de la venta contado	**5**	***cobrar venta al contado***	E	1	1	1	3
Emisión de la factura	**6**	***completar factura en PC***	E	10	3	1	14
	7	***archivar factura***	E	1	2	1	4
Recambio de filtros y lubricante	**8**	***completar tarjeta de servicio***	E	1	2	1	4
	9	***completar orden de servicio***	E	2	1	1	4
	10	***consultar guía de lubricantes***	E	1	2	1	4
Recepción de combustible	**11**	***completar nota de pedido de combustible***	E	1	1	1	3
	12	***comparar volumen descargado con el indicado en nota de pedido y remito de entrega***	C	2	2	1	5
	13	***firmar conformidad***	E	1	2	1	4
	14	***firmar no conformidad***	E	1	3	1	5

	15	***ingresar al sistema la cantidad de combustible recibida***	E	1	1	1	3
	16	***archivar nota de pedido***	E	1	2	1	4
Recepción de filtros y lubricantes	**17**	***verificar pedido de filtros y lubricantes***	C	2	2	1	5
	18	***firmar conformidad***	E	1	1	1	3
	19	***firmar no conformidad***	E	1	2	1	4
	20	***actualizar nota de pedido de filtros y lubricantes***	E	2	2	1	5
	21	***archivar nota de pedido de filtros y lubricantes***	E	1	2	1	4
Cambio de precios	**22**	***emitir notificación de cambio de precio***	E	4	2	1	7
	23	***archivar notificación de cambio de precio***	E	1	3	1	5
Emisión y conformidad de la nota de liquido producto	**24**	***verificar nota de líquido producto***	C	5	1	1	7
Medición física de tanques	**25**	***anotar volumen en planilla de inventarios***	E	1	1	1	3
	26	***archivar planilla de inventarios***	E	1	2	1	4
	27	***archivar datos***	E	1	1	1	3
Cambio de producto en tanques	**28**	***Dar de baja el combustible en el sistema***	E	1	1	1	3
Verificaciones de stock	**29**	***calcular diferencia stock***	E	2	1	1	4
	30	***completar planilla de control de existencia***	E	1	1	1	3
	31	***archivar planilla de control de existencia de productos***	E	1	2	1	4

	32	***calcular diferencia stock de filtros y lubricantes combustible***	E	2	1	1	4
Cierre de turno	**33**	***tomar estado surtidores***	C	1	1	1	3
	34	***emitir reporte de cierre de turno***	E	11	1	1	13
	35	***archivar reporte de cierre de turno***	E	1	2	1	4
Cierre de día	**36**	***emitir reporte de cierre de día***	E	1	4	1	6
	37	***archivar reporte de cierre de día***	E	1	2	1	4
Cierre de mes	**38**	***emitir reporte de cierre de mes***	E	1	4	1	6
	39	***archivar reporte de cierre de mes***	E	1	2	1	4
Pedido de combustible	**40**	***completar nota de pedido de combustible***	E	5	1	1	7
	41	***enviar fax de pedido de combustible***	E	1	2	1	4
	42	***archivar nota de pedido de combustible***	E	1	3	1	5
Pedido de filtros y lubricantes	**43**	***completar nota de pedido de filtros y lubricantes***	E	5	1	1	7
	44	***enviar fax de pedido de filtros y lubricantes***	E	1	2	1	4
	45	***archivar nota de pedido de filtros y lubricantes***	E	1	3	1	5
Total FP							**219**

Capítulo 8

Conclusiones

La propuesta presentada en esta tesis pretende facilitar el proceso de obtención de mediciones de una aplicación desde las primeras etapas del desarrollo, en particular, a partir del modelo de L&E. Se desarrolla un enfoque orientado a la medición de la funcionalidad basado en el Método de Análisis de Puntos de Función MarkII, aprovechando las similitudes conceptuales que se han encontrado entre el modelo de *transacciones* de MKII y los *episodios* de los escenarios. Sin embargo, debe notarse que los métodos de FPA requieren la Especificación de Requerimientos, documentación no disponible en la etapa de elicitación. Por lo tanto, se considera que este enfoque representa un avance que puede resultar beneficioso para realizar estimaciones del tamaño de una aplicación al principio del proceso de desarrollo, que podrán refinarse en etapas posteriores. Por otro lado, la experiencia de haber realizado un análisis detallado de los escenarios para estudiar la funcionalidad, permite afirmar que los beneficios mencionados en la sección 4.2 pueden lograrse durante la descomposición del L&E:

- Se puede obtener una medida del sistema en FP usando solamente la documentación de L&E.
- Al descomponer los *episodios* para obtener sus componentes de *entrada*, *proceso* y *salida*, se realiza un proceso de revisión de los escenarios y su relación con los símbolos del LEL e inversamente, lo que representa una contribución a la tracebility.
- El ingeniero necesita comprender claramente cada una de las funciones relevantes para poder describirlas en términos de sus componentes tal como se propone, lo que determina una mejor comprensión del sistema bajo estudio.
- Se encuentran detalles no contemplados o interpretados en forma errónea al desarrollar el modelo de L&E. Esto permite detectar y corregir las falencias en la adquisición del conocimiento, permitiendo, si fuese necesario, incorporar aquellos elementos que estuviesen ausentes o incorrectos a través de nuevas sesiones con el usuario. Este proceso favorece la completitud, correctitud y consistencia de los requerimientos.
- Durante esta etapa se pueden detectar todos los *episodios* que se presentan repetidos en distintos escenarios, lo que en las fases futuras resultará beneficioso como aporte a la modularidad, sea cual fuere el modelo de desarrollo e implementación que se adopte.

Otra conclusión que surge al analizar los *episodios* de los escenarios es que la tarea de descomposición de los *episodios* se facilitaría si éstos estuvieran expresados en términos del modelo *entrada-proceso-salida*. Esto sugiere que si se desea aplicar MKII a L&E es recomendable representar los *episodios* de los escenarios usando ese modelo de representación.

A partir de los resultados obtenidos al analizar los cuatro casos de estudio e indicados en la Tabla 16, se pueden describir algunas conclusiones:

- En general se observa una tendencia creciente en el número de FP a medida que aumentan los escenarios y los *episodios*.
- Es importante notar que la variación en la relación FP/Episodio para los casos analizados resulta en un rango de valores que oscilan entre un mínimo de 3,43 FP y un máximo de 6,06 FP. Estos valores permitirían hacer estimaciones iniciales rápidas a partir de la obtención de la cantidad de *episodios* que cumplen con las imposiciones del modelo propuesto.
- En cuanto a la relación FP/Escenario, esta medida parece tener menor valor para las estimaciones, pues su rango de variación es bastante más amplio y por otro lado, puede tener más vinculación con el área de aplicación del sistema. Puede que haya ciertos sistemas cuyos escenarios requieran un mayor número de *episodios* para describir las acciones.
- Los resultados obtenidos para el caso de la Administradora de Planes de ahorro para automóviles son notablemente más bajos en relación a los otros casos. Una interpretación es que la diferencia posiblemente puede deberse a factores relacionados con el nivel de detalle de la documentación disponible. En varios *episodios* de los casos estudiados se manipulan planillas; al comparar el caso de la *Recepción del Hotel* o el del *S. N. de obtención de pasaportes*, los cuales disponen de información detallada acerca de los datos necesarios para procesarlas, con el de la *Administradora de Planes de ahorro para automóviles*, se observa que en este último esa información no está disponible en forma completa en el LEL. En consecuencia, al analizar los componentes de los *episodios*, la cantidad de *DET* de entrada/salida es sensiblemente menor. Una situación similar se produce con las *referencias* a los *recursos*, pues en el caso de la *Recepción del Hotel* o el *S. N. de obtención de pasaporte* se han contabilizado todos los *recursos*, aún aquéllos que resultan obvios, como por ejemplo, el sello de Pagado, no mencionado como *recurso* en el caso de la *Administradora de Planes de ahorro para automóviles*. Esto produce variaciones que tienen impacto en los resultados finales. Vale aclarar que el análisis se realizó respetando la información disponible en el L&E de cada uno de los casos, sin introducir ninguna variación que pudiese favorecer los resultados finales.

Una cuestión que se considera de importancia y debe destacarse es que en esta propuesta no considera el esfuerzo, tal como es valorado en el método MKII FPA a partir de los *factores de peso estándar* o calculados para un entorno específico. Esto puede pensarse como un impedimento para hacer comparaciones directas con los resultados obtenidos mediante la aplicación del mismo.

Por último cabe mencionar una conclusión que surge de la aplicación del enfoque propuesto a los diferentes casos de estudio y es que la tarea de contabilizar los FP "manualmente" resulta sumamente tediosa, lo que se vería grandemente facilitado si existiese una herramienta automatizada. Queda entonces planteado el desafío para el futuro, el desarrollo de dicha herramienta.

En la sección 4.3 se señala la necesidad de estudiar la posibilidad de comparar los resultados obtenidos por los métodos convencionales de FPA con los resultantes de la aplicación de esta propuesta. Es éste otro tema pendiente para abordar en trabajos futuros.

[Albrecht'79] Albrecht, A., "Measuring Application Development Productivity", *IBM Applications Development Symposium*, Monterey, CA, 1979.

[Albrecht'83] Albrecht, A., Gaffney, J., *Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation*, IEEE Transactions on Software Engineering, Vol. SE-9, no. 6, Nov. 1983.

[Antonelli'99] Antonelli, L., Rossi, G., Oliveros A., *Baseline Mentor, An Application that Derives CRC Cards from Lexicon and Scenarios,* L.I.F.I.A., Facultad de Informática, U. N. L. P., Argentina, 1999.

[Benner'93] Benner, K. M, Feather, M. S., Johnson, W. L., Zorman, L. A., *Utilizing Scenarios in the Software Development Process*, USC, Information Sciences Institute, Admiralty Way, Marina del Rey, USA, 1993.

[Bernstein'95] Bernstein, L., Lubashevsky, A., *Living with Function Points*, AT&T, 1995.

[Caldas'98] Caldas, J., Masiero, P., *Erace-Tool - Uma Ferramenta Baseada Em Cenários Para À Engenharia De Requisitos*, Fundação Paulista de Educação e Tecnología, ICMC, Universidade de São Paulo, 1998.

[COSMIC-FFP'03] COSMIC-FFP, *COSMIC-FFP method approved as an ISO standard*, Press Release, Feb. 2003.

[CREWS'99] CREWS (Cooperative Requirements Engineering with Scenarios) Glossary, http://panoramix.univ-paris1.fr/CRINFO/CMB/MethBaseDescro/glossary.html, 1999.

[Cysneiros'01] Cysneiros, L., Leite, J., "Using the Language Extended Lexicon to Support Non-Functional Requirements Elicitation", *Workshop on Requirements Engineering*, Buenos Aires, Argentina, 2001.

[Davis'93] Davis, A., *Software Requirements: Objects, Functions, and States*, Englewood Cliffs, Prentice Hall, 1993.

[Dekkers'01] Dekkers, C., Aguiar, M., *Applying Function Point Analysis to Requirements Completeness*, Software Technology Support Center, CrossTalk The Journal of Defense Software Engineering, Feb. 2001.

[Esteban'98] Esteban, N., Heidanowski, A., *LEL y Escenarios de una Estación de Servicio*, Trabajo Final de Tópicos I, U. N. L. P., 1998.

[Fenton'96] Fenton, N., Pfleeger, S., *Software metrics A rigorous and Practical approach,* Second Edition, PWS Publishing Company, 1996.

[Fetcke'98] Fetcke, T., Abran, A., Nguyen, T., *Mapping the OO-Jacobson Approach into Function Point Analysis*, Université du Québec à Montreal, Software Engineering Management Research Laboratory, 1998.

[Fetcke'99] Fetcke, T., *The Warehouse Software Portfolio: A Case Study in Functional Size Measurement,* Report No. 1999-20, Technische Universität Berlin, Fachbereich Informatik, 1999.

[GIFPA'98] GIFPA, *Introduction to function points analysis,* GIFPA (Guild of Independent Function Points Analysts), 1998.

[Glinz'00] Glinz, M., *Improving the Quality of Requirements with Scenarios*, Institut für Informatik, Universität Zürich, 2000.

[Goldfarb'00] Goldfarb, S., *Software Estimating with Functional Metrics*, Q/P Management Group, Inc, 2000.

[Goodman'99] Goodman, P., *Aspects of Function Point Analysis*, Software Measurement Services, 1999.

[Grant Rule'98] Grant Rule, P., *Scenarios, Use Cases & MKII FPA*, GIFPA (Guild of Independent Function Points Analysts), 1998.

[Grant Rule'01[a]] Grant Rule, P., "The Importance of the Size of Software Requirements", Software Measurement Services, *NASSCOM Conference*, India, 2001.

[Grant Rule'01[b]] Grant Rule, P., "Using measures to understand requirements", Software Measurement Services, *ESCOM 2001*, 2001.

[Grant Rule'01[c]] Grant Rule, P., "A Comparison of the Mark II and IFPUG Variants of Function Point Analysis", Software Measurement Services, *ESCOM 2001*, 2001.

[Hadad'96] Hadad, G., Kaplan, G., Oliveros, A., Leite, J., *Integración de Escenarios con el Léxico Extendido del Lenguaje en la elicitación de requerimientos. Aplicación a un caso real*, Universidad de Belgrano, 1996.

[Hadad'97] Hadad, G., Kaplan, G., Oliveros, A., Leite, J., "Construcción de Escenarios a partir del Léxico Extendido del Lenguaje", *26 JAIIO*, Buenos Aires, Argentina, 1997.

[Heller'95] Heller, R., *An Introduction to Function Point Analysis*, Q/P Management Group Inc, 1995.

[Hornby'95] Hornby, A. S., *Oxford Advanced Learner's Dictionary of Current English,* Oxford University Press, Fifth edition, 1995.

[IFPUG'00[a]] International Function Point Users Group, *Frequently Asked Questions*, 2000.

[IFPUG'00[b]] International Function Point Users Group, *About Function Point Analysis*, 2000.

[ISO/IEC'95] International ISO/IEC, standard CD 14143, Information Technology, Software measurement, Definition of functional size measurement, 1995.

[Jones'97] Jones, C., *What Are Function Points?*, Software Productivity Research Inc, 1997.

[Kapelusz'79] *Diccionario Kapelusz de la lengua española*, Editorial Kapelusz, 1979.

[Leite'96] Leite, J., Oliveros, A., Rossi, G., Balaguer, F., Hadad, G., Kaplan, G., Maiorana, V., *Léxico extendido del lenguaje y escenarios del sistema nacional para la obtención de pasaportes*, Universidad de Belgrano, 1996.

[Leite'97] Leite J., Rossi G., et al, "Enhancing a Requirements Baseline with Scenarios", *Proceedings of RE 97' International Symposium on Requirements Engineering*, IEEE, 1997.

[Leite'00] Leite, J., Hadad, G., Doorn, J., Kaplan, G., *A Scenario Construction Process*, Requirements Engineering Journal, 2000.

[Loucopoulos'95] Loucopoulos, P., Karakostas, V., *System Requirements Engineering,* McGraw-Hill, 1995.

[Longstreet'96[a]] Longstreet, D., *Fundamentals of Function Point Analysis*, Systems Development Management, 1996.

[Longstreet'96[b]] Longstreet, D., *Function Points Step by Step,* Longstreet Consulting Inc, Systems Development Management, 1996.

[Longstreet'01] Longstreet, D., *Use cases and function points*, Longstreet Consulting Inc, 2000.

[Mauco' 97] Mauco, V., Ridao, M., del Fresno, M., Rivero, L., Doorn, J., *Proyecto Sistema de Planes de Ahorro*, Facultad de Ciencias Exactas, U. N. C. P. B. A., Argentina, 1997.

[Mc.Menamin'84] Mc.Menamin, S., Palmer, J., *Essential Systems Analysis*, Yourdon Press, 1984.

[MKII FPA CPM'98] *MKII Function Points Analysis Counting Practices Manual,* Versión 1.3.1, UKSMA (United Kingdom Software Metrics Association), 1998.

[Potts'98] Potts, C., Antón, A., *A Representational Framework for Scenarios of System Use*, College of Computing College of Engineering, Georgia Institute of Technology, North Carolina State University, Engineering Graduate Research Center Atlanta, 1998.

[Pressman'93] Pressman, R., *Ingeniería del Software. Un enfoque práctico,* Tercera edición, 1993.

[Rolland'93] Rolland, C., *Modeling the Requirements Engineering Process*, Université de Paris I Pantheon-Sorbonne, 1993.

[SEI'91] SEI Requirements Engineering Project, *Requirements Engineering and Analysis Workshop Proceedings*, Software Engineering Institute, Pittsburgh, Pennsylvania, 1991.

[Sutcliffe'97] Sutcliffe, A., *Workshop Exploring Scenarios in Requirements Engineering*, Centre for HCI Design, City University, Northampton Square, 1997.

[Symons'91] Symons, C., *Software sizing and estimating MKII FPA*, John Wiley and Sons, 1991.

[Symons'01] Symons, C., *Come back function point analysis (modernised) – All is forgiven!*, Software Measurement Services Ltd, 2001.

[Tavares'02] Tavares, H., Carvalho, A., Castro, J., "Medição de pontos por função a partir da especificação de requisitos", *Workshop on Requirements Engineering*, Valencia, España, 2002.

[Weidenhaupt'98] Weidenhaupt, K., Pohl, K., Jarke, M., Haumer, P., *Scenario Usage in System Development: A Report on Current Practice*, IEEE Software, March, 1998, RWTH Aachen, Informatik V, Germany, 1998.

[Wilson'97] Wilson, W., "Writing Effective Requirements Specifications", GSFC (Goddard Space Flight Center's), Software Assurance Technology Center (SATC) NASA, *Software Technology Conference*, UTA, 1997.

Anexo 1

Modelo de LEL y Escenarios para un caso de estudio

Caso de estudio: Recepción del Hotel

Este caso de aplicación de LEL y Escenarios se desarrolló en el marco del Magíster en Ingeniería del Software de la Universidad Nacional de La Plata en forma conjunta con Elena Centeno.

Objetivo

El objetivo de este trabajo es desarrollar la tarea de elicitación de los requerimientos para el subsistema correspondiente a la Recepción de un Hotel de la ciudad de Comodoro Rivadavia. El primer paso fue obtener toda la información relevante del Universo del Discurso. A partir de la tarea de elicitación, en una etapa posterior se podrán establecer los requerimientos para desarrollar el subsistema para automatizar las tareas del Departamento Recepción.

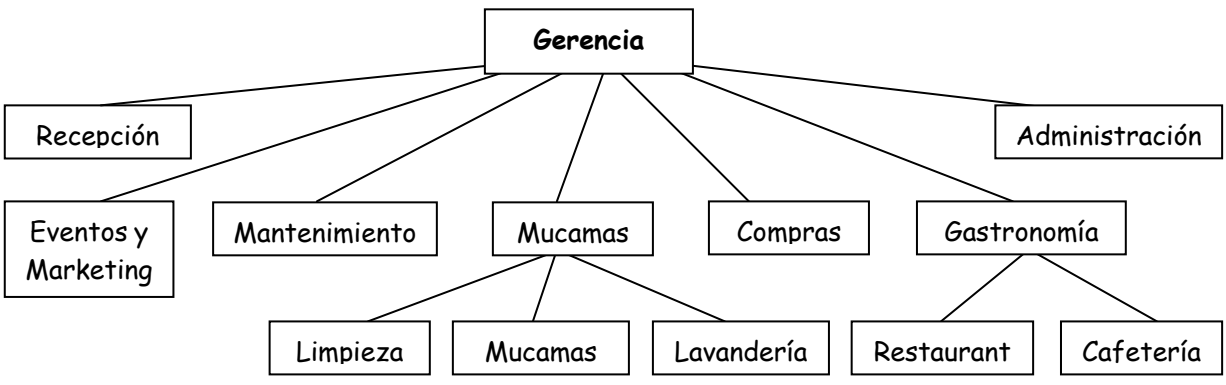
Proceso

Para realizar la tarea de elicitación se utilizó la técnica de LEL y Escenarios.

Con el objeto de conocer acerca del dominio del problema (UD), se realizaron entrevistas al Recepcionista (futuro usuario del subsistema) del Hotel. Durante las mismas se le consultó acerca de las tareas que se realizan en la Recepción, cuál es el funcionamiento de los diferentes Departamentos y su relación con Recepción. Además se tuvo acceso a las planillas que se usan habitualmente para registrar las diferentes actividades. A través de las entrevistas se pudo conocer el vocabulario usual y analizar las diferentes situaciones que se plantean en el trabajo cotidiano.

El UD requiere establecer el lugar donde se realizarán las tareas de ingeniería, los recursos disponibles, objetivos del producto y sus límites. [Hadad'96] Para el caso del subsistema de la Recepción, el macrosistema es el sistema anfitrión completo.

Pudo establecerse que si bien el Hotel no tiene un organigrama definido, funciona con los siguientes Departamentos:



La Recepción del Hotel es el lugar donde se realizan todas las actividades correspondientes a la reserva, ingreso, permanencia y egreso de los pasajeros al Hotel.

LEL

Con la información recolectada se generó una lista de frases, palabras o expresiones usuales en el lenguaje cotidiano dentro de la Recepción, que se denominan entradas o símbolos del LEL.

Sobre la base de su significado, se estableció una clasificación de las entradas del LEL.

Se hizo la descripción de cada uno de los símbolos del LEL, lo que permitió obtener diferentes versiones del LEL, que se fueron refinando a medida que se iba conociendo en forma más precisa el problema en estudio y a través de nuevas entrevistas con el Recepcionista.

Clasificación de entradas del LEL

Según su significado, las entradas se clasificaron de la siguiente manera:

Departamentos (T): en la noción se define, se indican las actividades que realiza y en el impacto se describen los efectos sobre Personas, Formularios, Datos, Ficheros, Objetos.

Actividades (A): en la noción se indica en qué consisten, quien las ejecuta y el lugar donde se realizan y en el impacto se describe el efecto que producen sobre Datos, Formularios, Personas, Objetos, Ficheros.

Datos (D): en la noción se indica su contenido, qué actividad, función o persona los provee y dónde se almacenan y en el impacto se describen las actividades en las que se usan.

Ficheros (F): en la noción se indica su contenido y ubicación y en el impacto se establece en que actividades se utilizan.

Objetos (O): en la noción se define el concepto, sus características, su utilidad y su ubicación y en el impacto se describen las actividades que se realizan sobre ellos.

Personas (P): en la noción se define y en el impacto se explica cuál es su acción sobre Datos, Actividades, Formularios, Ficheros, Objetos.

Lista definitiva de símbolos del LEL según su clasificación

Actividades

cancelación de reserva
check in
check out
facturación
no show

pago
pedido de alojamiento
pedido de limpieza extra
pedido de mantenimiento extra
pedido del servicio de despertador
reposición de insumos
solicitud de reserva

Datos

dia de egreso
dia de ingreso
disponibilidad de habitaciones
tarifa

Departamentos

Administración
agencia / agencia de viajes
Compras
Gastronomía
Gerencia
Mantenimiento
Mucamas
Recepción

Ficheros

carpeta de la habitación
fichero de pasajeros
planilla de ocupación de habitaciones
planilla de reservas

Formularios

comprobante consumos del minibar
comprobante de consumos de cafetería y/o restaurant
comprobante de lavandería
comprobante de llamados telefónicos
planilla del pasajero
aviso de mensajes
voucher

Objetos

habitación
lista de precios
minibar / frigobar
tarjeta codificada

Personas

pasajero / huésped / pax
recepcionista

Construcción de escenarios

Una vez que se definió la lista de símbolos del LEL, se procedió al desarrollo de los Escenarios, siguiendo el esquema propuesto en la bibliografía.

1. Identificación de los actores de la aplicación.

Principales:

Recepcionista
Pasajero / huésped / pax

Secundarios:

Departamento Administración
Departamento Gastronomía
Departamento Compras
Departamento Mantenimiento
Departamento Mucamas

2. Generación de la lista de escenarios candidatos.

A continuación se listan los impactos de las entradas del LEL correspondientes a los actores principales identificados y se eliminan los repetidos:

Del recepcionista

Atiende una *solicitud de reserva* o un *pedido de alojamiento*.

Hace el *check in*.

Atiende el *pedido del servicio de despertador*.

Recibe los *pedidos de limpieza extra* y los *pedidos de mantenimiento extra* del *pasajero/huésped/pax*.

Hace el pedido de *reposición de insumos*.

Hace el *check out*.

Ante la *cancelación de reservas* o el *no show* actualiza la *planilla de reservas*.

Del pasajero

Hace el *pago*.

3. Descripción de los escenarios candidatos.

A partir de cada uno de los impactos seleccionados en el ítem anterior y sucesivos proceso de revisión y validación con el Recepcionista se definieron los siguientes escenarios.

cancelación de la reserva
solicitud de reserva
pedido de alojamiento
check in
check out
servicio de despertador
pedido de limpieza extra
pedido de mantenimiento extra
reposición de insumos
no show
pago

4. Revisión de los escenarios.

En esta etapa, se procedió a unificar algunos escenarios que presentaban similitudes en sus características.

Unificación de escenarios

Cancelación de la reserva y **No show** en un único escenario denominado **Cancelación de la reserva**
Atención del pedido de mantenimiento y **Atención del pedido de limpieza extra** en un escenario **Pedido de servicio extra**

Escenario general **Recepción del Hotel**

Al final de proceso se obtuvieron 41 entradas del LEL y 10 escenarios.
A continuación se presenta el texto en formato rtf del modelo de LEL y Escenarios generado con la herramienta ***Baseline Mentor Workbench***.

Administración de la Recepción

January 27, 2002
version 4.2

LEL entries y Scenarios

Generated by BaselineMentor

Indice:

LEL Entries:

Administración
agencia / agencia de viajes
aviso de mensaje
cancelación de reserva
carpeta de la habitación
check in
check out
Compras
comprobante de consumos de cafetería y/o restaurant
comprobante de consumos del minibar
comprobante de gastos de lavandería
comprobante de llamados telefónicos
disponibilidad de habitaciones
día de egreso
día de ingreso
facturación
fichero de pasajeros
Gastronomía
Gerencia
habitación
lista de precios
Mantenimiento
minibar / frigobar
Mucamas
no show
pago
pasajero / huésped / pax
pedido de alojamiento
pedido de limpieza extra
pedido de mantenimiento extra
pedido del servicio de despertador
planilla de ocupación de habitaciones
planilla de reservas
planilla del pasajero
recepcionista
Recepción
reposición de insumos
solicitud de reserva
tarifa
tarjeta codificada
voucher

Scenarios:

cancelación de la reserva
check in
check out
pago
pedido de alojamiento
pedido de servicio extra
recepción del Hotel
reposición de insumos
servicio de despertador
solicitud de reserva

LEL entries

Administración

- Notion
  - Es un departamento del Hotel.
  - Administra todas las actividades relacionadas con el manejo económico del Hotel.
- Behavioral responses
  - Atiende todas las cuestiones relacionadas al personal.
  - Hace la *facturación* para el *pasajero / huésped / pax*.

---

agencia / agencia de viajes

- Notion
  - Es una organización externa e independiente del Hotel.
  - Administra las ventas de servicios turísticos a los pasajeros.
- Behavioral responses
  - Puede hacer una *solicitud de reserva* o una *cancelación de reserva* para el *pasajero / huésped / pax*.
  - Puede hacer un *voucher* para el *pasajero / huésped / pax*.

---

aviso de mensaje

- Notion
  - Contiene el registro de los mensajes recibidos cuando el *pasajero / huésped / pax* no se halla en el Hotel, junto con el número de *habitación*.
  - Lo registra la *Recepción*.
  - Se almacena temporariamente en la *Recepción*.
- Behavioral responses
  - Lo recibe el *pasajero / huésped / pax*.

---

cancelación de reserva

- Notion
  - Es la baja de la *solicitud de reserva* de un *pasajero / huésped / pax*.
  - La hace el *pasajero / huésped / pax*, una *agencia* u otro hotel vía telefónica, fax, e-mail o personalmente.
  - Se hace en la *Recepción*.
- Behavioral responses
  - Elimina la *solicitud de reserva* del *pasajero / huésped / pax* de la *planilla de reservas*.
  - Actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*.

---

carpeta de la habitación

- Notion

  - Contiene: *comprobante de consumos de cafetería y/o restaurant*, *comprobante de gastos de lavandería*, *comprobante de consumos del minibar*, *comprobante de llamados telefónicos* y *voucher*, si correspondiere.
  - Se guarda en un mueble de archivo para carpetas en la *Recepción*.
- Behavioral responses
  - La *Administración* la utiliza para hacer la *facturación*.

---

check in

- Notion
  - Es el conjunto de los trámites necesarios para el ingreso de un *pasajero / huésped / pax* al Hotel.
  - Lo atiende el *recepcionista*.
  - Se hace en la *Recepción*.
- Behavioral responses
  - Si el *pasajero / huésped / pax* presenta un *voucher* entonces el *recepcionista* lo almacena en la *carpeta de la habitación*.
  - Se busca el nombre del *pasajero / huésped / pax* en la *planilla de reservas*.
  - Si el *pasajero / huésped / pax* se presenta por primera vez al Hotel debe completar y firmar la *planilla del pasajero* sino se buscan sus datos personales por nombre en el *fichero de pasajeros*.
  - El *recepcionista* controla la *planilla del pasajero* y actualiza el *fichero de pasajeros*.
  - Se asigna un número de *habitación* al *pasajero / huésped / pax*.
  - Al actualizar la *disponibilidad de habitaciones*, se actualiza la *planilla de ocupación de habitaciones*.
  - Se agrega el número de *habitación* asignado en la *planilla de reservas*.
  - Se hace una *tarjeta codificada* para el número de *habitación*.
  - Se entrega la *tarjeta codificada* y el control remoto TV.
  - Si el *pasajero / huésped / pax* lo solicita se le provee la llave de la caja de seguridad de la *habitación* para guardar objetos de valor y/o dinero.
  - Si el *pasajero / huésped / pax* lo solicita se le entrega la llave del *minibar / frigobar*.

---

check out

- Notion
  - Es el conjunto de los trámites necesarios para el egreso del *pasajero / huésped / pax* del Hotel.
  - Lo atiende el *recepcionista*.
  - Se hace en la *Recepción*.
- Behavioral responses
  - Se provee al *pasajero / huésped / pax* junto con la factura: *comprobante de gastos de lavandería*, *comprobante de consumos de cafetería y/o restaurant*, *comprobante de consumos del minibar* y *comprobante de llamados telefónicos* .
  - Se recibe el *pago* del *pasajero / huésped / pax*.
  - Se agrega el sello de Pagado en la factura
  - Se envía la copia de la factura del *pasajero / huésped / pax* al Departamento *Administración*.
  - Se recibe la *tarjeta codificada*, el control remoto TV, la llave del *minibar / frigobar* si estuviera en poder del *pasajero / huésped / pax* y la llave de la caja de seguridad si estuviera en poder del *pasajero / huésped / pax*
  - Al actualizar la *disponibilidad de habitaciones*, se actualiza la *planilla de ocupación de habitaciones*.

Compras

- Notion
  - Es un departamento del Hotel.
  - Administra todas las cuestiones relacionadas a la *reposición de insumos* para todos los departamentos del Hotel.
- Behavioral responses
  - [no relevado]

---

comprobante de consumos de cafetería y/o restaurant

- Notion
  - Contiene el número de *habitación*, el detalle y el importe de los consumos hechos y la firma del *pasajero / huésped / pax* o acompañantes según corresponda.
  - Lo provee el Departamento *Gastronomía*.
  - Se almacena en la *carpeta de la habitación*.
- Behavioral responses
  - Lo hace el Departamento *Gastronomía* .
  - El *pasajero / huésped / pax* lo firma.
  - El Departamento *Gastronomía* lo envía a la *Recepción*.

---

comprobante de consumos del minibar

- Notion
  - Contiene el número de *habitación*, el importe y el detalle de los artículos del *minibar / frigobar* consumidos por el *pasajero / huésped / pax*.
  - Lo provee el Departamento *Mucamas*.
  - Se almacena en la *carpeta de la habitación*.
- Behavioral responses
  - Lo hace diariamente el Departamento *Mucamas* .
  - El Departamento *Mucamas* lo envía a la *Recepción*.

---

comprobante de gastos de lavandería

- Notion
  - Contiene el importe del trabajo de lavandería solicitado por el *pasajero / huésped / pax* y su número de *habitación*.
  - Lo provee el Departamento *Mucamas*.
  - Se almacena en la *carpeta de la habitación*.
- Behavioral responses
  - Lo hace el Departamento *Mucamas* .
  - El Departamento *Mucamas* lo envía a la *Recepción*.

---

comprobante de llamados telefónicos

- Notion
  - Contiene el importe de la llamada telefónica y el número de *habitación* del *pasajero / huésped / pax*.
  - Lo provee el sistema telefónico.
  - Se almacena en la *carpeta de la habitación*.
- Behavioral responses
  - El sistema telefónico lo hace en forma automática.

---

disponibilidad de habitaciones

- Notion
  - Contiene la cantidad y tipo de *habitación* desocupada.
  - Se obtiene de la *planilla de ocupación de habitaciones* como diferencia entre la capacidad total del Hotel y la suma de las habitaciones ocupadas y reservadas.
- Behavioral responses
  - Se actualiza cuando se actualiza la *planilla de ocupación de habitaciones*.

---

día de egreso

- Notion
  - Contiene la fecha en que el *pasajero / huésped / pax* deja el Hotel.
  - La provee el *pasajero / huésped / pax* o la obtiene el *recepcionista*.

  - Se almacena en la *planilla de reservas*.
- Behavioral responses
  - La diferencia entre esta fecha y el *día de ingreso* permite calcular los días que el *pasajero / huésped / pax* ha permanecido en el Hotel.
  - La *Administración* la utiliza para el cálculo de la *facturación*.

---

día de ingreso

- Notion
  - Contiene la fecha en que el *pasajero / huésped / pax* ingresa al Hotel.
  - La provee el *pasajero / huésped / pax* o la obtiene el *recepcionista*.
  - Se almacena en la *planilla de reservas*.
- Behavioral responses
  - La diferencia entre el *día de egreso* y ésta permite calcular los días que el *pasajero / huésped / pax* ha permanecido en el Hotel.
  - La *Administración* la utiliza para el cálculo de la *facturación*.

---

facturación

- Notion
  - Es el proceso de emisión de la factura del *pasajero / huésped / pax*.
  - Se hace en la *Administración*.
- Behavioral responses
  - Se busca el nombre del *pasajero / huésped / pax* en la *planilla de reservas* y se obtiene el número de *habitación*, el *día de ingreso* y el *día de egreso*.
  - Se envía a la *Administración* la *carpeta de la habitación* correspondiente al número de *habitación* del *pasajero / huésped / pax*, junto con el *día de ingreso* y el *día de egreso*.
  - Se recibe una factura original y una copia desde la *Administración*.

---

fichero de pasajeros

- Notion
  - Contiene el conjunto de *planilla del pasajero*.
  - Se guarda en un mueble de archivo en la *Recepción*.
- Behavioral responses
  - Se utiliza como archivo histórico para el *check in* a un *pasajero / huésped / pax*.

Gastronomía

- Notion
  - Es un departamento del Hotel.
  - Brinda el servicio de desayuno incluido en la *tarifa*, el servicio de Restaurant y Cafetería y el servicio de cafetería en la *habitación*.
- Behavioral responses
  - Envía el *comprobante de consumos de cafetería y/o restaurant* del *pasajero / huésped / pax* a la *Recepción*.

Gerencia

- Notion
  - Es un departamento del Hotel.
  - Establece las políticas de alto nivel del Hotel.
- Behavioral responses
  - [no relevado].

---

habitación

- Notion

  - Es un cuarto del Hotel.
  - Puede ser de tipo estándar, suite junior o suite presidencial, todas ellas en la modalidad single/doble.
  - Tiene una *tarifa* determinada y un número que la identifica.
  - Aloja al *pasajero / huésped / pax*.
- Behavioral responses
  - Se reserva al hacer una *solicitud de reserva*.
  - Se asigna en el *check in*.
  - Se desocupa en el *check out*.

---

lista de precios

- Notion
  - Es un listado que provee el Departamento *Gerencia*.
  - Se utiliza para almacenar la *tarifa* según el tipo y modalidad de la *habitación* .
  - Está en la *Recepción* y en la *Administración*.
- Behavioral responses
  - La utiliza la *Administración* para obtener la *tarifa* de una *habitación* .
  - La utiliza la *Recepción* para informar la *tarifa* de una *habitación* cuando se efectúa una *solicitud de reserva* o *pedido de alojamiento*

Mantenimiento

- Notion
  - Es un departamento del Hotel.
  - Hace las tareas de conservación y reparaciones programadas y solicitadas desde otros departamentos.
- Behavioral responses
  - Hace la tarea requerida en una *habitación*.

---

minibar / frigobar

- Notion
  - Es una pequeña heladera que tiene una llave.
  - Se utiliza para conservar bebidas y alimentos refrigerados.
  - Está en cada *habitación*.
- Behavioral responses
  - El *pasajero / huésped / pax* puede consumir los elementos que contiene.
  - Diariamente el Departamento *Mucamas* controla su contenido y repone los elementos faltantes.

Mucamas

- Notion
  - Es un departamento del hotel dirigido por la Gobernanta.
  - Atiende el servicio de lavandería de todo el Hotel, incluyendo los pedidos particulares del *pasajero / huésped / pax* .
  - Hace la limpieza de cada *habitación* .
  - Hace la limpieza de los sectores públicos.
  - Atiende el *pedido de limpieza extra* enviado por la *Recepción*.
  - Registra los consumos del *minibar / frigobar*.
- Behavioral responses
  - Maneja la *tarjeta codificada* maestra.
  - Envía a la *Recepción* *comprobante de gastos de lavandería* y *comprobante de consumos del minibar* del *pasajero / huésped / pax* .

---

no show

- Notion

  - Es la baja de la *solicitud de reserva* por no presentación del *pasajero / huésped / pax*.
  - Lo hace el *recepcionista*.
  - Se hace en la *Recepción*.
- Behavioral responses
  - Si el *pasajero / huésped / pax* no se presenta el *día de ingreso* hasta las 06 hs. del día siguiente entonces se elimina la *solicitud de reserva* en la *planilla de reservas*.
  - Se actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*

---

pago

- Notion
  - Es el proceso para cancelar el importe de la *facturación* al *pasajero / huésped / pax*.
  - Lo hace el *pasajero / huésped / pax*.
  - Se hace en la *Recepción*.
- Behavioral responses
  - Si es en efectivo o cheque se guarda en la caja de la *Recepción*.
  - Si es con tarjeta de crédito se solicita la autorización, se hace el cupón, el *pasajero / huésped / pax* lo firma y se guarda en la *Recepción*.
  - Si se debita de una cuenta corporativa el *pasajero / huésped / pax* firma la factura y se envía a la *Administración*.

---

pasajero / huésped / pax

- Notion
  - Es una persona que tiene una *solicitud de reserva* o hace un *pedido de alojamiento*
- Behavioral responses
  - Provee los datos requeridos por el *recepcionista*.
  - Puede presentar un *voucher*.
  - Completa la *planilla del pasajero* la primera vez que se presenta en el Hotel.
  - Recibe la *tarjeta codificada* y el control remoto TV.
  - Puede solicitar la llave de la caja de seguridad y/o *minibar / frigobar*.
  - Puede hacer consumos en Cafetería y/o Restaurant, consumos del *minibar / frigobar*, llamados telefónicos y gastos de lavandería.
  - Puede hacer un *pedido del servicio de despertador*.
  - Puede hacer un *pedido de limpieza extra* y/o un *pedido de mantenimiento extra*.
  - Puede recibir un *aviso de mensaje*.
  - Recibe la factura y los comprobantes de gastos.
  - Hace el *pago*.
  - Devuelve la *tarjeta codificada*, el control remoto TV y la llave del *minibar / frigobar* y/o de la caja de seguridad de la *habitación* si las solicitó.
  - Puede hacer una *cancelación de reserva*.

---

pedido de alojamiento

- Notion
  - Es el pedido de una *habitación* hecho en forma personal, sin una *solicitud de reserva*.
  - Lo hace una persona.
  - Se recibe en la *Recepción*.
- Behavioral responses
  - Se consulta la *planilla de ocupación de habitaciones*.
  - Si hay *disponibilidad de habitaciones* se registran sus datos personales en la *planilla de reservas*.
  - Se hace el *check in*.

---

pedido de limpieza extra

- Notion
  - Es la solicitud de un servicio de limpieza adicional.
  - Lo hace el *pasajero / huésped / pax*.

  - Se recibe en la *Recepción*.
- Behavioral responses
  - Se registra el número de *habitación* del *pasajero / huésped / pax* que solicitó el servicio.
  - Se hace la lista con los pedidos recibidos.
  - Se envía la lista al Departamento *Mucamas*.

---

pedido de mantenimiento extra

- Notion
  - Es la solicitud de un servicio de reparaciones.
  - Lo hace el *pasajero / huésped / pax*.
  - Se recibe en la *Recepción*.
- Behavioral responses
  - Se registra el número de *habitación* del *pasajero / huésped / pax* que solicitó el servicio y el tipo de reparación a efectuar.
  - Se hace la lista con los pedidos de reparaciones.
  - Se envía el pedido al Departamento *Mantenimiento*.

---

pedido del servicio de despertador

- Notion
  - Es la solicitud del servicio que brinda el Hotel para despertar al *pasajero / huésped / pax* en un horario determinado.
  - Lo hace el *pasajero / huésped / pax*.
  - Se atiende en la *Recepción*
- Behavioral responses
  - Se registra el número de *habitación* y la hora solicitada.
  - Llegada esa hora, se llama por teléfono al número de *habitación* del *pasajero / huésped / pax*.

---

planilla de ocupación de habitaciones

- Notion
  - Contiene el estado de ocupación según el tipo y modalidad de la *habitación*.
  - Se guarda en la *Recepción*.
- Behavioral responses
  - Se actualiza cuando se hace una *solicitud de reserva*, una *cancelación de reserva*, un *no show*, un *check in* o un *check out*.

---

planilla de reservas

- Notion
  - Contiene la *solicitud de reserva* de un *pasajero / huésped / pax*.
  - Se hace una por día y se conserva en una carpeta en la *Recepción* hasta el *check out* de todos los pasajeros que figuran en ella, luego se guardan en un archivo.
- Behavioral responses
  - Se utiliza para actualizar la *planilla de ocupación de habitaciones*.
  - En el *check in* se le agrega el número de *habitación* asignado al *pasajero / huésped / pax*.
  - Se actualiza cuando se hace una solicitud de *solicitud de reserva*, *cancelación de reserva*, *no show* o *pedido de alojamiento* del *pasajero / huésped / pax*.

---

planilla del pasajero

- Notion
  - Contiene el nombre, número de documento, otros datos personales y comerciales, firma, *día de ingreso*, procedencia del *pasajero / huésped / pax* y datos del vehículo, si correspondiere.
  - La completa el *pasajero / huésped / pax* en el *check in*.
  - Se almacena en el *fichero de pasajeros* en la *Recepción* del Hotel..
- Behavioral responses

- La completa el *pasajero / huésped / pax* la primera vez que se presenta en el Hotel.
- Se utiliza para actualizar el *fichero de pasajeros*.

---

recepcionista

- Notion
  - Persona que atiende el Departamento *Recepción* del Hotel.
- Behavioral responses
  - Recibe una caja con dinero cuando se hace cargo de la *Recepción*.
  - Atiende una *solicitud de reserva* o un *pedido de alojamiento*..
  - Controla la *disponibilidad de habitaciones*.
  - Hace el *check in*.
  - Recibe y almacena en la *carpeta de la habitación* : *comprobante de consumos de cafetería y/o restaurant*, *comprobante de gastos de lavandería*, *comprobante de consumos del minibar* y *comprobante de llamados telefónicos* del *pasajero / huésped / pax*.
  - Atiende el *pedido del servicio de despertador*.
  - Recibe los *pedido de limpieza extra* y *pedido de mantenimiento extra* del *pasajero / huésped / pax*.
  - Atiende los mensajes y si el *pasajero / huésped / pax* no está, completa el *aviso de mensaje*.
  - Envía la *carpeta de la habitación* a la *Administración* para la *facturación*.
  - Hace el *check out*.
  - Ante la *cancelación de reserva* o el *no show* actualiza la *planilla de reservas*.
  - Hace el pedido de *reposición de insumos*.

Recepción

- Notion
  - Es un departamento interrelacionado con todos los departamentos del Hotel.
  - Atiende todas las actividades relativas al ingreso, permanencia y egreso del *pasajero / huésped / pax*.
- Behavioral responses
  - Es el lugar de trabajo del *recepcionista*.
  - Almacena toda la información relativa al movimiento de pasajeros.

---

reposición de insumos

- Notion
  - Es la acción de pedir y recibir los elementos necesarios para el funcionamiento de la *Recepción*.
  - Lo hace el *recepcionista*.
  - Se hace en la *Recepción*.
- Behavioral responses
  - Se hace una lista de elementos faltantes.
  - Se envía la lista al Departamento *Compras*.
  - Se recibe el pedido desde el Departamento *Compras*.
  - Se firman los formularios de recibido.

---

solicitud de reserva

- Notion
  - Es un pedido de hospedaje para una persona hecho con anterioridad al *día de ingreso*.
  - Caduca por *no show* del *pasajero / huésped / pax*.
  - La hace una persona, una *agencia* u otro hotel vía telefónica, fax, e-mail o personalmente.
  - La atiende la *Recepción*.
- Behavioral responses
  - Se consulta en la *planilla de ocupación de habitaciones* la *disponibilidad de habitaciones*.
  - Se informa la *tarifa*.
  - Se solicita la aprobación a la persona, *agencia* u otro hotel.

- Se registra en la *planilla de reservas* el nombre y apellido del *pasajero / huésped / pax*, tipo y modalidad de la *habitación*, cantidad de pasajeros, *día de ingreso*, *día de egreso*, teléfono, *tarifa* y observaciones.
- Se actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*.

---

tarifa

- Notion
  - Es el costo de la *habitación* según sus características.
  - La provee la *lista de precios*.
  - Se almacena en la *lista de precios*.
- Behavioral responses
  - La utiliza la *Administración* para hacer la *facturación*.

---

tarjeta codificada

- Notion
  - Es una tarjeta magnética que tiene grabado un código.
  - Según su tipo se utiliza para abrir una o varias habitaciones.
  - Está en el Departamento *Mucamas* o en poder del *pasajero / huésped / pax*.
- Behavioral responses
  - Si está destinada al *pasajero / huésped / pax* tiene un código específico para abrir una determinada *habitación*.
  - Si es para el personal del Departamento *Mucamas* se trata de una tarjeta maestra que permite el acceso a las habitaciones de todo un piso o de todo el Hotel.

---

voucher

- Notion
  - Contiene los datos del *pasajero / huésped / pax*, el *día de ingreso*, el *día de egreso*, la cantidad de pasajeros, nombre y dirección del Hotel.
  - Lo provee la *agencia*.
  - Se almacena en la *carpeta de la habitación*.
- Behavioral responses
  - El *pasajero / huésped / pax* paga el importe del alojamiento en la *agencia* y éste se acredita en una cuenta corriente del Hotel.
  - El *recepcionista* lo recibe del *pasajero / huésped / pax* .
  - Se utiliza para cancelar el importe del ítem correspondiente a alojamiento en la *facturación*.

Scenarios

---

cancelación de la reserva

- Goal
  - Dar de baja una *solicitud de reserva* de un *pasajero / huésped / pax*.
- Context
  - Se realiza en la *Recepción* del Hotel. Existe una *solicitud de reserva* para un *pasajero / huésped / pax*.
- Resources
  - *planilla de reservas*
  - *planilla de ocupación de habitaciones*
  - Teléfono
  - Fax
  - e-mail
- Actors
  - *recepcionista*
  - *agencia*
  - otro hotel
  - *pasajero / huésped / pax*
- Episodes
  - **if** el *recepcionista* recibe el pedido de anulación de una *solicitud de reserva* o el *pasajero / huésped / pax* no se presenta en el período comprendido entre las 12 hs del *día de ingreso* establecido en la *solicitud de reserva* y las 06 hs. del día siguiente **then** el *recepcionista* elimina la *solicitud de reserva* de la *planilla de reservas*.
  - El *recepcionista* actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*.
  - **exception:** El teléfono, el fax o el e-mail no funcionan.
- Links defUsr

---

check in

- Goal
  - Efectuar el ingreso del *pasajero / huésped / pax* al Hotel.
- Context
  - Se realiza en la *Recepción* del Hotel el *día de ingreso* que figura en la *planilla de reservas*.
- Resources
  - *voucher*
  - *planilla del pasajero*
  - *planilla de reservas*
  - *planilla de ocupación de habitaciones*
  - *fichero de pasajeros*.
  - *tarjeta codificada* de la *habitación*
  - Llave del *minibar / frigobar*
  - Llave de la caja de seguridad

  - Control remoto TV
- Actors
  - *recepcionista*
  - *pasajero / huésped / pax*
- Episodes
  - **if** el *pasajero / huésped / pax* presenta un *voucher* **then** el *recepcionista* registra los datos incluidos en el mismo y lo almacena en la *carpeta de la habitación*.
  - El *recepcionista* busca el nombre del *pasajero / huésped / pax* en la *planilla de reservas*.
  - **if** el *pasajero / huésped / pax* se presenta por primera vez al Hotel **then** el *recepcionista* le provee la *planilla del pasajero* que debe completar con sus datos y firmar. El *recepcionista* controla que la *planilla del pasajero* esté completa y la almacena en el *fichero de pasajeros*.
  - **if** el *pasajero / huésped / pax* ya estuvo en el Hotel **then** el *recepcionista* busca los datos personales por nombre en el *fichero de pasajeros*.
  - **if** el *pasajero / huésped / pax* tiene acompañantes **then** cada uno debe registrarse en forma individual.
  - El *recepcionista* asigna el número de *habitación* al *pasajero / huésped / pax*.
  - Al actualizar el *recepcionista* la *disponibilidad de habitaciones*, se actualiza la *planilla de ocupación de habitaciones*.
  - El *recepcionista* agrega el número de *habitación* asignado al *pasajero / huésped / pax* en la *planilla de reservas*.
  - El *recepcionista* hace una *tarjeta codificada* para el número de *habitación*.
  - El *recepcionista* le provee al *pasajero / huésped / pax* la *tarjeta codificada* de la *habitación* y el control remoto TV.
  - **if** el *pasajero / huésped / pax* lo solicita **then** el *recepcionista* le provee la llave del *minibar / frigobar* y/o de la caja de seguridad.
  - **restriction:** El *voucher* no cumple con los requisitos.
  - **exception:** Faltan tarjetas magnéticas para codificar
  - La máquina generadora de tarjetas codificadas no funciona.
- Links defUsr

---

check out

- Goal
  - Efectuar el egreso del *pasajero / huésped / pax* del Hotel.
- Context
  - El *pasajero / huésped / pax* se presenta en la *Recepción* del Hotel el *día de egreso* acordado manifestando su deseo de retirarse. La *Administración* hizo la *facturación*.
- Resources
  - Factura
  - *tarjeta codificada* de la *habitación*
  - Llave del *minibar / frigobar*
  - Llave de la caja de seguridad
  - Control remoto TV
  - Sello de Pagado
  - *planilla de ocupación de habitaciones*
- Actors
  - *recepcionista*
  - *pasajero / huésped / pax*
- Episodes
  - El *recepcionista* entrega al *pasajero / huésped / pax* junto con la factura: *comprobante de gastos de lavandería*, *comprobante de consumos de cafetería y/o restaurant*, *comprobante de consumos del minibar* y *comprobante de llamados telefónicos*.
  - **pago**
  - El *recepcionista* agrega el sello de Pagado en la factura
  - El *recepcionista* envía la copia de la factura del *pasajero / huésped / pax* al Departamento *Administración*.

  - El *recepcionista* recibe la *tarjeta codificada* y el control remoto TV y la llave del *minibar / frigobar* y/o caja de seguridad si estuviera en poder del *pasajero / huésped / pax*.
  - Al actualizar el *recepcionista* la *disponibilidad de habitaciones*, se actualiza la *planilla de ocupación de habitaciones*.
  - **restriction:** El *recepcionista* no puede cobrar porque el *pasajero / huésped / pax* no cuenta con los medios necesarios.
- Links defUsr

---

pago

- Goal
  - Recibir el *pago* del *pasajero / huésped / pax*.
- Context
  - Se realiza en la *Recepción* del Hotel durante el *check out*. La factura está preparada
- Resources
  - Dinero en efectivo
  - Tarjeta de crédito
  - Cupón de tarjeta de crédito
  - Cheque
- Actors
  - *recepcionista*
  - *pasajero / huésped / pax*
- Episodes
  - **if** el *pasajero / huésped / pax* paga con dinero en efectivo o con cheque **then** el *recepcionista* recibe el dinero o el cheque y lo guarda en la caja de la *Recepción*.
  - **if** el *pasajero / huésped / pax* paga con tarjeta de crédito **then** el *recepcionista* solicita la autorización al emisor de la tarjeta de crédito. **if** el *pasajero / huésped / pax* está habilitado **then** el *recepcionista* hace el cupón con el importe correspondiente, el *pasajero / huésped / pax* lo firma, se guarda en la *Recepción* y se le provee la copia.
  - **if** el *pasajero / huésped / pax* pertenece a una empresa que tiene cuenta corporativa en el Hotel **then** firma la factura y se envía a la *Administración*.
- Links defUsr

---

pedido de alojamiento

- Goal
  - Atender un *pedido de alojamiento* realizada por una persona.
- Context
  - Se realiza en la *Recepción* del Hotel. Una persona se presenta para un *pedido de alojamiento* .
- Resources
  - *planilla de ocupación de habitaciones*
  - *planilla de reservas*
- Actors
  - *recepcionista*
  - *pasajero / huésped / pax*
- Episodes
  - El *recepcionista* consulta la *planilla de ocupación de habitaciones* para conocer la *disponibilidad de habitaciones*.
  - **if** existe *disponibilidad de habitaciones* y la persona estuviera de acuerdo **then** el *recepcionista* registra los datos personales del *pasajero / huésped / pax* en la *planilla de reservas*.
  - **check in**.
  - **restriction:** No hay *disponibilidad de habitaciones*
- Links defUsr

---

pedido de servicio extra

- Goal
  - Atender *pedido de limpieza extra* y *pedido de mantenimiento extra* .
- Context
  - Se realiza en la *Recepción* del Hotel. Existe un *pasajero / huésped / pax* que solicita un servicio para su *habitación*.
- Resources
  - Lista de *pedido de mantenimiento extra*
  - Lista de *pedido de limpieza extra*
- Actors
  - *recepcionista*
  - *pasajero / huésped / pax*
  - Departamento *Mantenimiento*
  - Departamento *Mucamas*
- Episodes
  - El *recepcionista* registra un *pedido de limpieza extra* o *pedido de mantenimiento extra* que hace el *pasajero / huésped / pax* para su *habitación*.
  - El *recepcionista* registra el número de *habitación* y el tipo de servicio a efectuar.
  - El *recepcionista* hace la lista de los *pedido de limpieza extra* o *pedido de mantenimiento extra* .
  - El *recepcionista* envía la lista al Departamento *Mantenimiento* o *Mucamas*.
  - **restriction:** No se registró el número de *habitación* en la lista de *pedido de limpieza extra* y/o *pedido de mantenimiento extra* .
  - No se registró el tipo de servicio.
  -
- Links defUsr

---

recepción del Hotel

- Goal
  - Atender las tareas referentes al movimiento de los pasajeros.
- Context
  - Se realiza en la *Recepción* del Hotel. Funciona las 24 horas del día.
- Resources
  - *planilla de reservas*
  - *planilla de ocupación de habitaciones*
  - *fichero de pasajeros*
- Actors
  - *recepcionista*
  - *pasajero / huésped / pax*
- Episodes
  - **solicitud de reserva** o **pedido de alojamiento**
  - **check in**
  - Si el *pasajero / huésped / pax* lo solicita entonces **pedido de servicio extra**
  - Si el *pasajero / huésped / pax* lo solicita entonces **servicio de despertador**
  - La *carpeta de la habitación* se envía a la *Administración* junto con el *día de ingreso* y el *día de egreso* , para la emisión de la *facturación*.
  - **check out**
  - # **reposición de insumos** #
  - # **cancelación de la reserva** #
- Links defUsr

---

reposición de insumos

- Goal
  - Reponer los elementos faltantes para el funcionamiento de la *Recepción*.

- Context
  - Se realiza en la *Recepción* del Hotel. El *recepcionista* tiene registradas las necesidades de insumos.
- Resources
  - Lista de pedidos de insumos
  - Insumos
  - Formularios de recibido
- Actors
  - *recepcionista*
- Episodes
  - El *recepcionista* hace una lista de insumos faltantes.
  - El *recepcionista* envía el pedido al Departamento *Compras*.
  - El *recepcionista* recibe el pedido que envía el Departamento *Compras*.
  - El *recepcionista* firma el conforme en el recibo.
  - **restriction:** No se registraron los insumos faltantes
- Links defUsr

---

servicio de despertador

- Goal
  - Registrar el *pedido del servicio de despertador* y controlar que se cumpla
- Context
  - Existe un *pasajero / huésped / pax* que requiere ser despertado a una hora determinada cuyo pedido se registra en la *Recepción*.
- Resources
  - Teléfono
  - Reloj
  - Lista de *pedido del servicio de despertador*
- Actors
  - *recepcionista*
  - *pasajero / huésped / pax*
- Episodes
  - El *recepcionista* registra el número de *habitación* y la hora solicitada por el *pasajero / huésped / pax*.
  - El *recepcionista* llama por teléfono al *pasajero / huésped / pax* cuando llega la hora.
  - **restriction:** El *recepcionista* no consulta la lista de *pedido del servicio de despertador* .
- Links defUsr

---

solicitud de reserva

- Goal
  - Atender una *solicitud de reserva* realizada por una persona, *agencia* u otro hotel.
- Context
  - Se realiza en la *Recepción* del Hotel
- Resources
  - *lista de precios*
  - *planilla de reservas*
  - *planilla de ocupación de habitaciones*
  - Teléfono
  - Fax
  - e-mail
- Actors
  - *recepcionista*
  - *agencia*
  - otro hotel

  - *pasajero / huésped / pax*
- Episodes
  - El *recepcionista* verifica en la *planilla de ocupación de habitaciones* la *disponibilidad de habitaciones* comprendida entre el *día de ingreso* y el *día de egreso* solicitados, y si hubiera, informa la *tarifa* y solicita la aprobación de la persona, *agencia* o de otro hotel.
  - El *recepcionista* registra el nombre del *pasajero / huésped / pax*, cantidad de pasajeros, tipo y modalidad de la *habitación*, *día de ingreso*, *día de egreso* y *tarifa* en la *planilla de reservas*
  - El *recepcionista* actualiza la *disponibilidad de habitaciones* en la *planilla de ocupación de habitaciones*.
  - **restriction:** No hay *disponibilidad de habitaciones*
  - **exception:** El teléfono, el fax o el e-mail no funcionan.
- Links defUsr

Anexo 2

Modelo de reglas

Para la descripción formal del enfoque se requiere la definición de un conjunto de reglas y esto determinó la necesidad de adoptar un modelo de representación para las mismas.

El modelo adoptado se basa en el utilizado por el proyecto NATURE. Este proyecto apunta a investigar tres aspectos básicos de la fase de Ingeniería de Requerimientos (RE) en el desarrollo de Sistemas de Información (IS): (1) representación del conocimiento, (2) definición y uso del conocimiento del dominio, (3) modelado del proceso de RE [Rolland'93].

El enfoque particular elegido para modelar los procesos de RE en NATURE asume que los bloques básicos de construcción de cualquier proceso pueden ser modelados como una 4-tupla: <*situación, decisión, argumento, acción*>. Asocia la situación que tiene que tratar el ingeniero de requerimientos con una de las decisiones que puede tomar para resolver el problema, el argumento que motiva la decisión y una de las acciones que se van a ejecutar para aplicar la decisión [Rolland'93].

Los cuatro conceptos básicos son [Rolland'93]:

- *situación* para explicar el contexto de la decisión,
- *decisión* para guiar el proceso de RE,
- *acción* para ejecutar las transformaciones del producto,
- *argumento* para soportar la toma de decisiones.

Una *situación* se trata frecuentemente de una parte del producto bajo desarrollo que justifica tomar una decisión sobre él. Representa el contexto de una decisión. Puede ser una sentencia acerca del dominio de la aplicación o un requerimiento. Puede referirse a parte de productos existentes como resultado de desarrollos previos o ser una porción de conocimiento del dominio.

Una *decisión* refleja una elección que hace un ingeniero de requerimientos en un momento dado del proceso de RE. Puede considerarse como la intención de transformar el producto. Explícitamente representa la resolución de una cuestión.

Una *acción* ejecuta una transformación sobre el producto. Es una materialización de la decisión. La ejecución cambia el producto y puede generar nuevas situaciones sujetas a nuevas decisiones.

Los *argumentos* son sentencias que soportan u objetan las decisiones dentro de un contexto dado.

En esta tesis se utilizó este modelo como base para la definición de las reglas considerando el conjunto <*situación, decisión, acción*>. A fin de evitar confusiones semánticas con las acciones que se realizan en relación al sistema bajo estudio se decidió reemplazar la palabra *acción* por *ejecución*. Por lo tanto las reglas se definen como la 3-tupla <*situación*, *decisión*, *ejecución* >.