Introducción a Herramientas CASE

11056 - Sistemas de Información I

Universidad Nacional de Luján

Departamento de Ciencias Básicas
División Computación
Área de Sistemas de Información

Introducción

Desde sus orígenes la actividad de desarrollar software ha motivado el desarrollo de herramientas de software que den apoyo al proceso de la ingeniería de software. Uno de los primeros indicios de herramientas CASE (***C**omputer **A**ided **S**oftware **E**ngineering)* fue el proyecto ISDOS que se desarrolló durante la década de los 70 el cual tenía por objetivos analizar la relación entre los requisitos para un producto de software y los costos así como las necesidades de los diseñadores para dar respuesta a esos mismos requisitos.

No obstante, varios autores coinciden en que la primera herramienta CASE fue *Excelerator* la que se puso en producción en el año 1984 y corría en las pc´s de aquella época. A principios de los ´90 las CASE aparecieron con fuerza para los mainframes de la mano de IBM quien trabajó para desarrollar una CASE que abarcara todo el ciclo de vida del proceso de desarrollo de software.

A mediados de los ´90 ya se empezó a trabajar en herramientas que den soporte a parte del proceso y no al proceso en forma completa. De esta manera los distintos equipos de desarrollo podrían valerse de distintas herramientas de software para dar soporte a las distintas fases del proceso de desarrollo.

OBJETIVOS DE LAS CASE

Las herramientas CASE se producen para dar respuesta a alguno de los siguientes objetivos o a algún subconjunto de ellos:

- Mejorar la productividad en el desarrollo y mantenimiento del software.
    1. Las herramientas CASE deben dar visibilidad al proceso de desarrollo de software así como a la documentación del mismo. En la fase de mantenimiento contar con documentación actualizada del producto facilita esta actividad así como asiste en la evaluación del impacto de los cambios.
- Aumentar la calidad de los productos de software.
    1. Un producto desarrollado aplicando un proceso controlado y visible tiene mayor probabilidad de tener una adecuada calidad que uno que se desarrolla sin aplicar modelos de proceso preestablecidos.
- Reducir el tiempo y costo del proceso de desarrollo y facilitar el mantenimiento de los productos de software.
    1. La capacidad de dotar de visibilidad al proceso y a la documentación del producto resulta esenciales al momento de estimar el costo y el tiempo para la fase de mantenimiento.
- Mejorar la planificación y administración de los proyectos de desarrollo de software.
    1. Las herramientas CASE facilitan la administración de los proyectos posibilitando el trabajo en equipo y la coordinación de tareas entre quienes participan del proceso.
- Aumentar la biblioteca de conocimiento de una organización ayudando a la búsqueda de soluciones para los requisitos.
    1. Administrar una biblioteca de requerimientos es el punto de partida para la reutilización de requerimientos y la reutilización del software.
- Automatizar el desarrollo del software, la documentación, la generación de código, las pruebas de errores y la gestión del proyecto.

1. Todas las tareas que puedan automatizarse del proceso de desarrollo de software nos permiten disminuir el tiempo de desarrollo así como contar con el resultado de esas actividades sin necesidad de la intervención humana.

- Ayudar a la reutilización del software, portabilidad y estandarización de la documentación.
    1. La unificación de la documentación de los productos que se administran con una herramienta CASE facilita la intervención de distintos profesionales en el proceso de desarrollo de software que no participaran en el proceso desde su generación. Esta característica disminuye fuertemente la dependencia de algunas personas en particular para la evolución de los productos de software.
- Posibilitar la gestión de todas las fases de desarrollo de software con una misma herramienta.
    1. Las herramientas CASE unifican en un solo entorno las distintas tareas que se llevan a cabo durante el proceso de desarrollo de software.
- Facilitar el uso de las distintas metodologías propias de la ingeniería del software.
    1. Por lo general, las herramientas CASE trabajan aplicando las metodologías de uso frecuente en la Ingeniería de Software unificando el lenguaje mediante el que se comunican las ideas y modelos.

COMPONENTES DE LAS HERRAMIENTAS CASE

Una herramienta CASE debe contar, como mínimo, con los siguientes componentes para que tenga la capacidad de manifestar las características que describimos en el apartado anterior:

- Diccionario de Datos o Repositorio, en el cual se almacenarán todos los elementos que se definan desde la herramienta. Este diccionario debiera ser administrado por la CASE como un sistema de base de datos
- Meta Modelo, es la definición de las técnicas y metodologías que soporta la herramienta CASE. Podemos representarlo como la configuración de la manera en que operará la herramienta.
- Facilidad para la carga de datos, la herramienta debiera tener la posibilidad de importar datos provenientes de otro software y prever la descarga de información de manera automática. Podemos definir a esta característica como la capacidad de interactuar con otros sistemas.
- Capacidad de comprobar errores de consistencia, efectuando análisis de exactitud, integridad y consistencia de los esquemas generados por la herramienta.
- Interfaz de usuario amigable, la interfaz de usuario deberá permitir editar texto, dibujar modelos, etc.

TIPO DE HERRAMIENTAS CASE

Las Herramientas CASE pueden dar soporte a distintas fases del proceso de desarrollo y suelen clasificarse según su alcance:

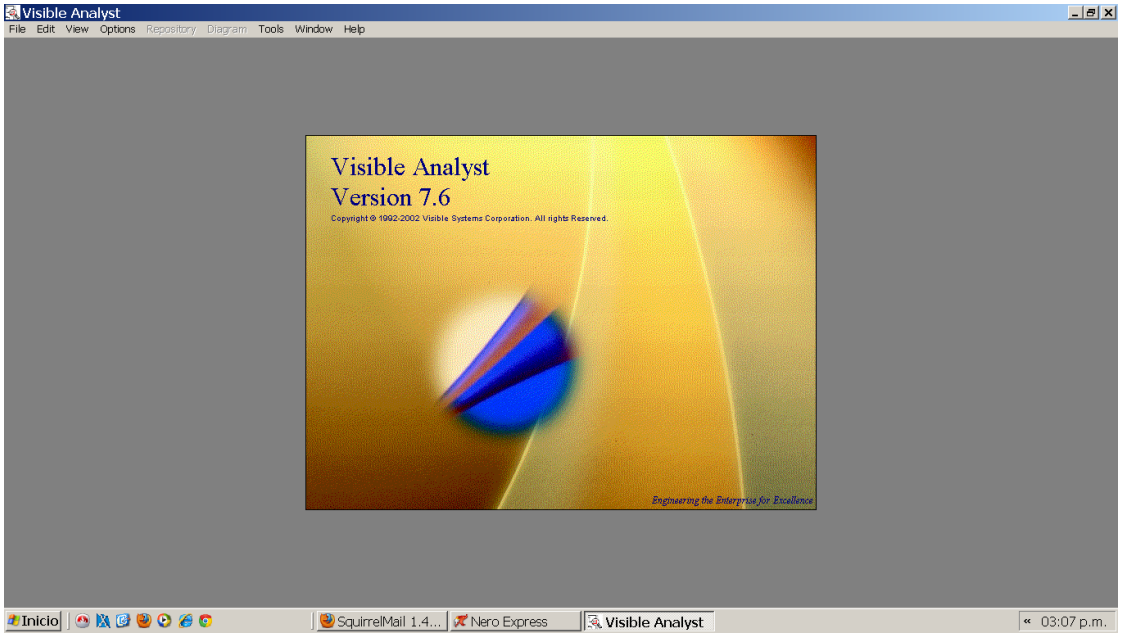
- CASE de alto nivel: son las herramientas que automatizan o apoyan las fases finales o superiores del ciclo de vida del desarrollo de sistemas como la planificación de sistemas, el análisis y el diseño de sistemas.

- CASE de bajo nivel: son las herramientas que automatizan o apoyan las fases finales o inferiores del ciclo de vida como el diseño detallado de sistemas, la implantación de y las tareas de soporte.
- CASE de ciclo de vida: son las herramientas que apoyan las actividades que tienen lugar a lo largo de todo el ciclo de vida, se incluyen actividades como la gestión de proyectos y la estimación de esfuerzo.

En la actualidad, podrían clasificarse a las herramientas CASE de la siguiente manera:
TOOLKIT: es una colección de herramientas integradas que permiten automatizar un conjunto de tareas de algunas de las fases del ciclo de vida: Planificación estratégica, Análisis, Diseño, Generación de código.
WORKBENCH: Son conjuntos integrados de herramientas que dan soporte a la automatización del proceso completo de desarrollo del producto de software. Permiten cubrir el ciclo de vida completo. El producto final aportado por ellas es el código ejecutable y su documentación.

Otra clasificación que se aplica es en función de las fases de proceso de desarrollo que abarca la herramienta:
UPPER CASE: Planificación estratégica, Requerimientos.
MIDDLE CASE: Análisis y Diseño.
LOWER CASE: Generación de código, test e implantación.

EJEMPLO DE HERRAMIENTA CASE



CREAR UN NUEVO PROYECTO

Cada proyecto que se crea representa un sistema completo. Un proyecto puede emplearse también para describir una unidad dentro de un sistema muy grande. Si se mantiene el sistema completo dentro de un sólo proyecto, VisibleAnalyst asegura que el sistema en su conjunto permanece consistente a lo largo de todo el proceso de desarrollo en lugar de tener que

chequear la consistencia global una vez que todas las piezas o unidades hayan sido juntadas. La versión LAN de Visible Analyst permite que múltiples diseñadores participen en los diagramas dentro de un mismo proyecto.

Existen diferentes tipos de líneas para cada tipo de diagrama. Se puede elegir el tipo de línea de cualquier diagrama disponible para usar en un diagrama no estructurado. Esta selección debe realizarse antes de crear el diagrama. (Figura1)

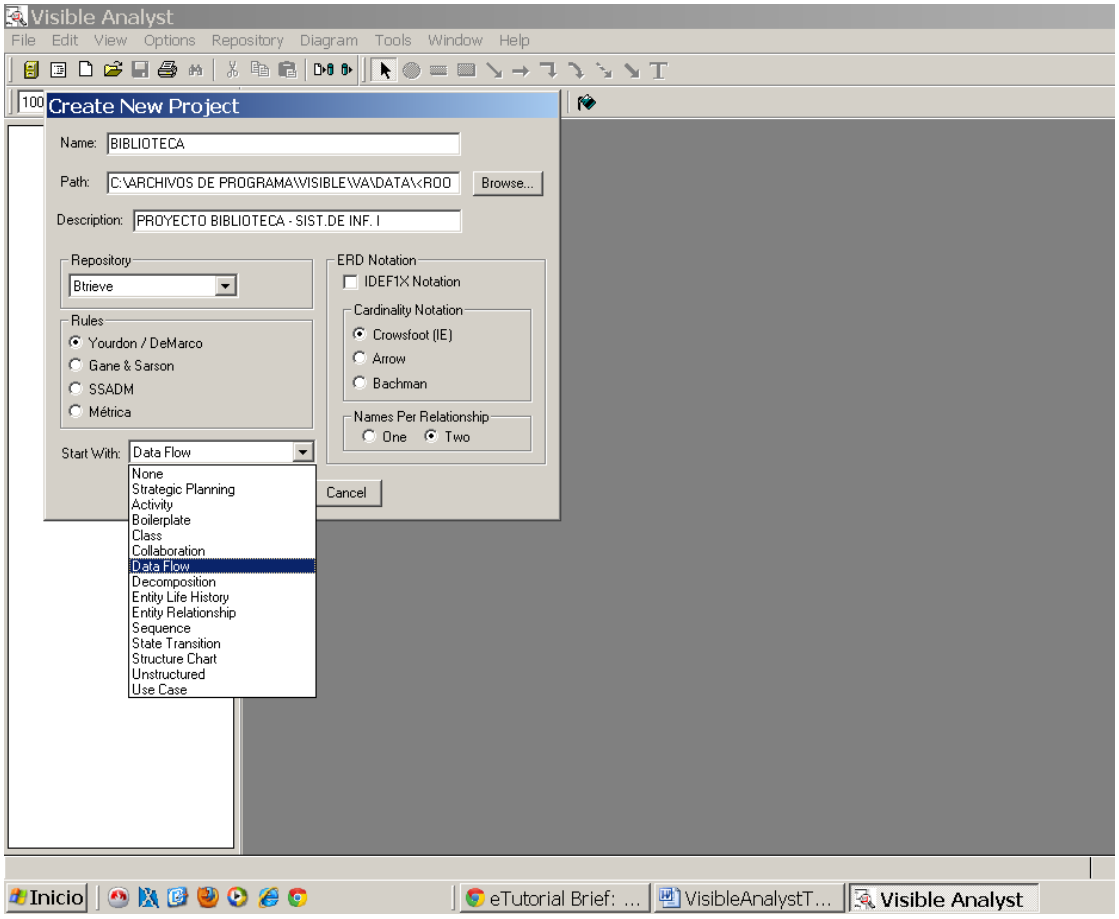


Figura 1

CREAR UN NUEVO DIAGRAMA

Después de crear un proyecto creamos un nuevo Diagrama del tipo Flujo de Datos. Para tal fin, utilizaremos la resolución del Trabajo Práctico Nº1 “Biblioteca” y comenzaremos dibujando el Diagrama de Contexto, tal como se muestra en la Figura 2.

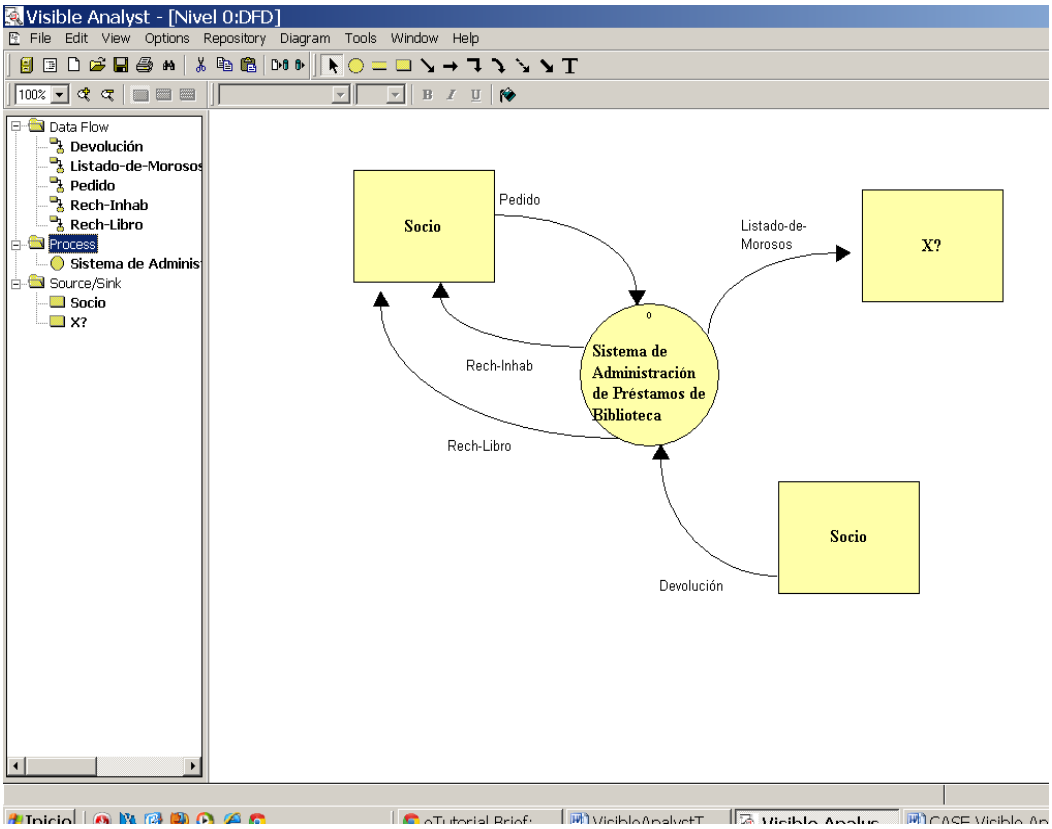
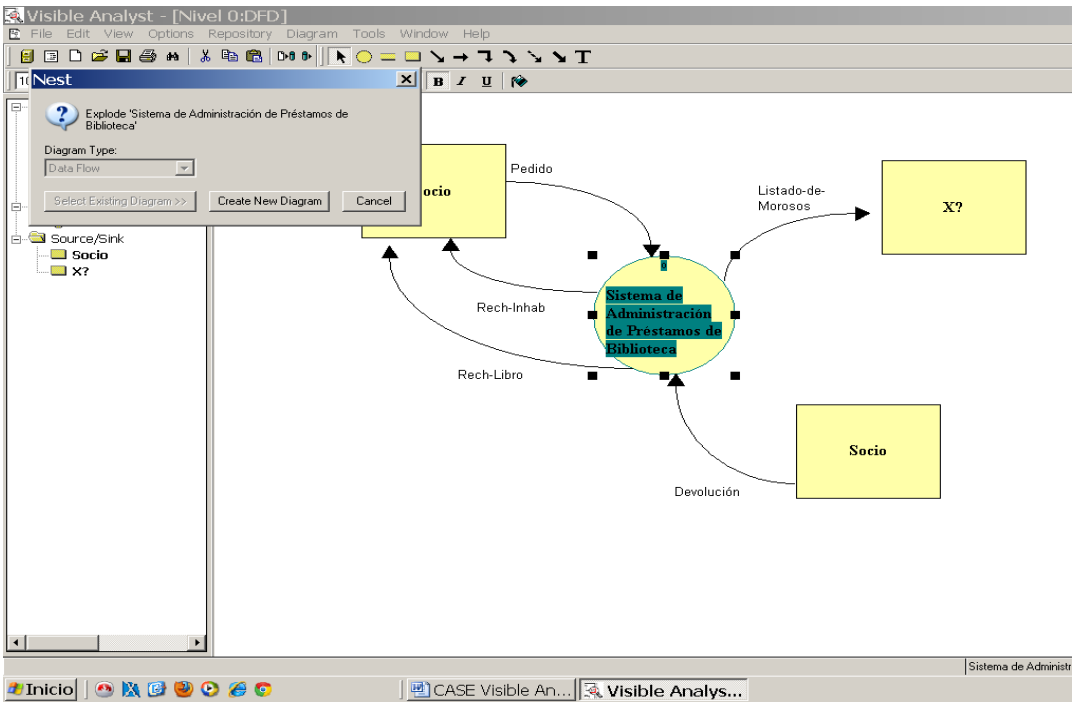


Figura 2

CREAR UN DFD PARTCICIONADO POR EVENTOS

Después de crear el Diagrama de Contexto, podremos "explotarlo" para dar origen al Diagrama de Flujo de Datos particionado por Eventos. Ver Figura 3.



Al crear un Nuevo Diagrama, los Flujos de Datos definidos en el nivel anterior se mostrarán automáticamente con el fin de garantizar la consistencia entre niveles de la técnica y de esta manera, colaborar con el diseñador para que no cometa errores. Se dibuja un nuevo diagrama mostrando los flujos de entrada en la esquina superior izquierda del diagrama, y los flujos de salida en la esquina superior derecha, tal como aparecen en la figura 4.

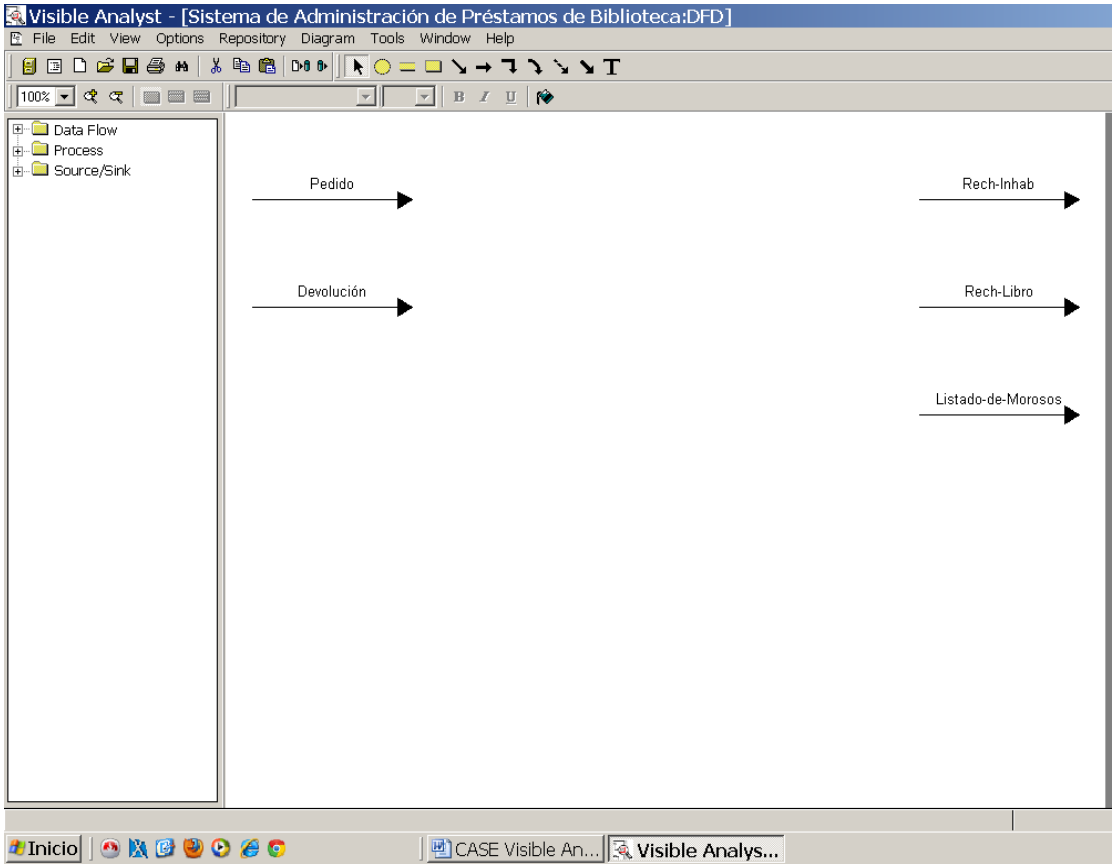


Figura 4

A continuación deberemos añadir los procesos que representa a cada evento de la Lista de Eventos que hemos definido en la clase. A modo de ejemplo, se utilizó la herramienta VA para representar el evento Nº1 que habíamos llamado “Atender Préstamos”. Luego debemos dibujar los Flujos de Datos y las Memorias de acuerdo al diagrama que definimos en la clase.

Si prestamos atención, a medida que se construye el Diagrama, el software genera automáticamente el registro de los componentes del mismo (Flujos de Datos, Memorias, Procesos y Entidades Externas), tal como se muestra en la Figura 5.

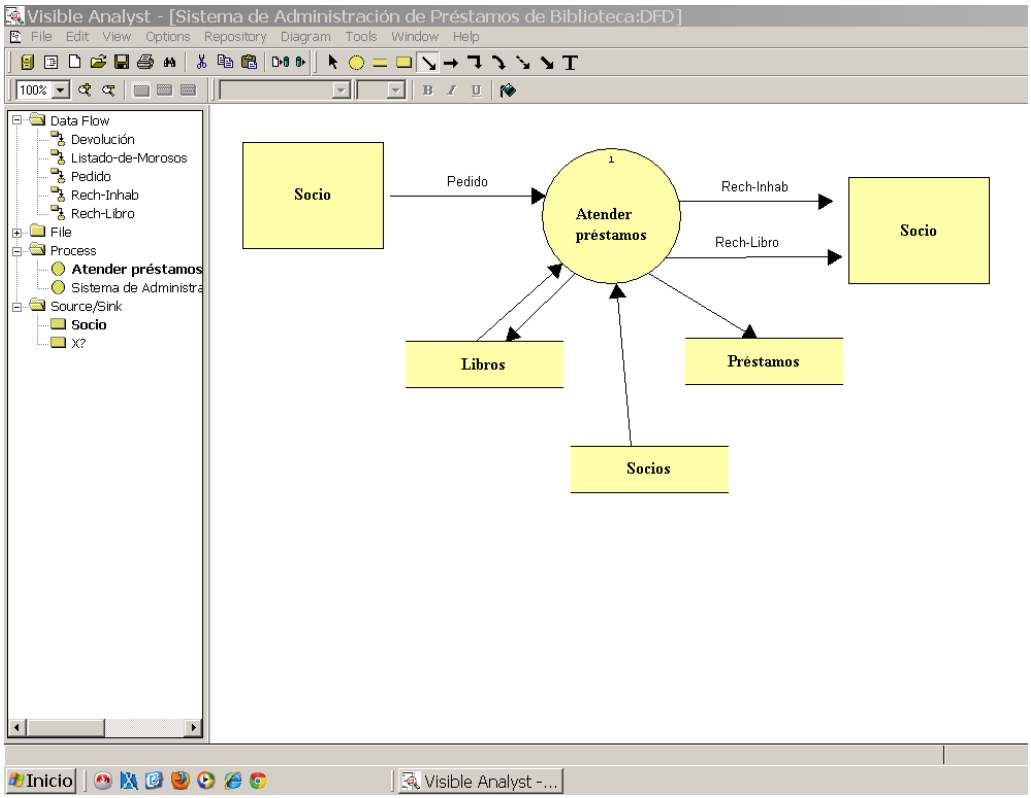


Figura 5

Luego se deberá definir cada componente del diagrama, con el fin de ir completando el Diccionario de Datos del Sistema. En la Figura 6 y en la Figura 7 se muestran las dos pantallas principales que se utilizan en la herramienta para definir los datos que componen la memoria "Préstamos" y los diferentes seteos que permite realizar. Fundamentalmente, se deberá definir el tipo de dato, la longitud y si puede ser (o no) nulo, respecto a cada dato componente de la memoria.

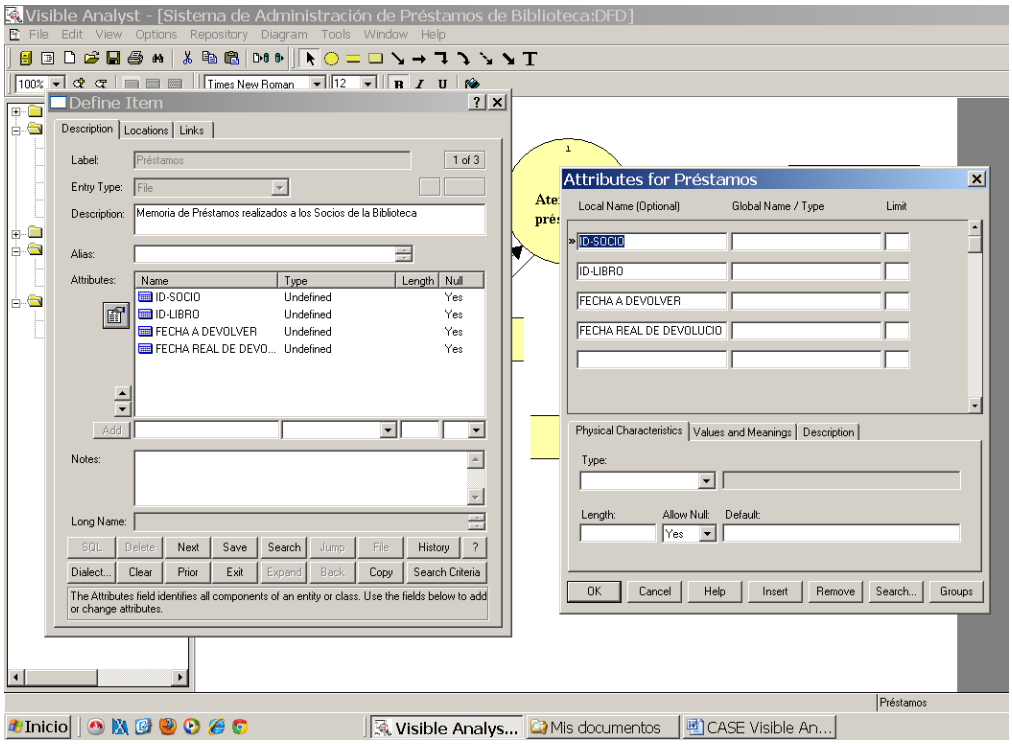
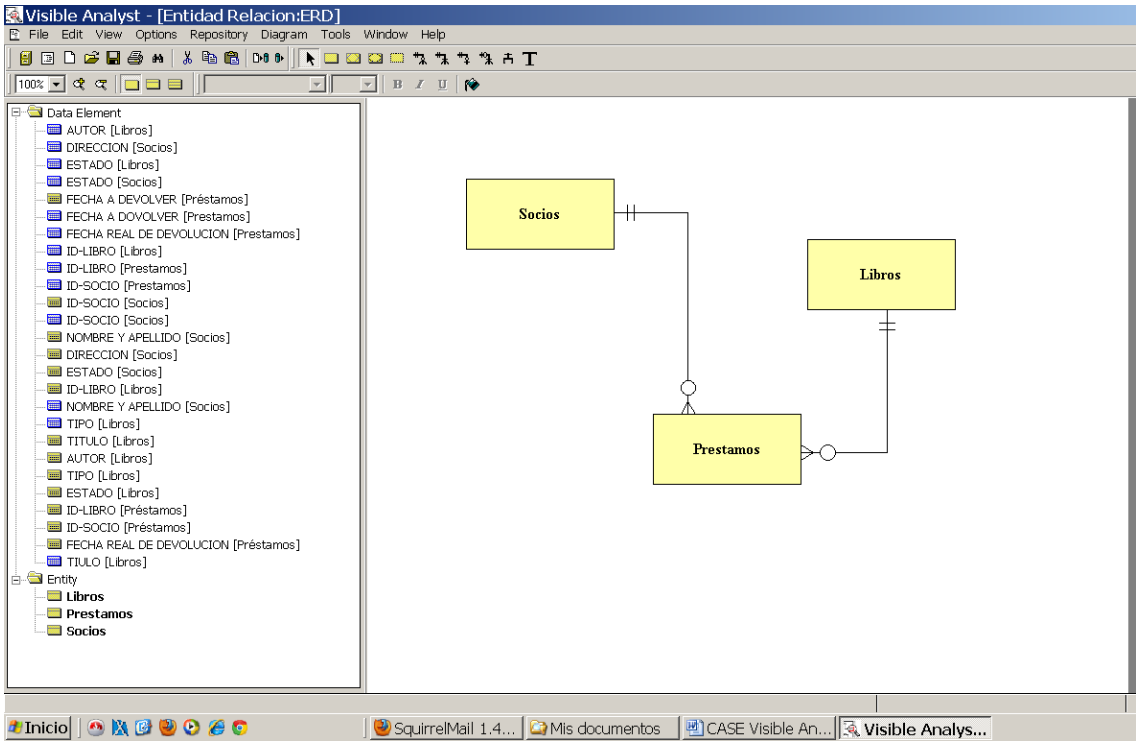


Figura 6

De esta manera se puede avanzar con la herramienta hasta la definición del modelo de datos mediante un Diagrama Entidad Relación.



Bibliografía

Kendall & Kendall, *Análisis y Diseño de Sistemas*, Sexta Edición, Editorial Pearson Educación, 2005

Ingeniería de Software de Ian Sommerville. Editorial Addison Wesley (2002).

Tutorial “http://es.scribd.com/doc/20369496/TutorialVA”