SEGESOFT: un entorno de entrenamiento para la gestión de proyectos software

J. Riquelme, I. Ramos, J. Aguilar, F. Ferrer, M. Toro (U. de Sevilla)
J. Dolado, A. Ruiz de Infante (U. del País Vasco)
J. Tuya, P. Fernández, M.A. Prieto (U. de Oviedo)
M. Ruiz-Carreira (U. de Cádiz),
D. Rodriguez-García, M. Satpathy, R. Harrison (U. of Reading),
R. Matilla, M.A. Álvarez (THALES INFORMATION SYSTEM)

Resumen

La gestión de proyectos se puede considerar todavía como un arte en el cual el uso de la información cuantitativa tiende a fomentar un enfoque más riguroso de la gestión. En este trabajo presentamos la estructura y los elementos principales de un entorno de entrenamiento para directores de proyectos. El objetivo del sistema es proporcionar una estructura uniforme para que se puedan incorporar nuevas técnicas en la estructura de forma gradual. El sistema reúne y almacena los datos del proyecto real y del simulado e implementa diferentes técnicas basadas en aprendizaje automático, modelado dinámico, monitorización de proyectos, etc. El propósito básico de este trabajo es el de presentar un entorno que facilite la toma de decisiones integrando diferentes técnicas y líneas de investigación.

Palabras claves: gestión de proyectos software, simulación, minería de datos

1 Un entorno de entrenamiento

La gestión de proyectos es una de las actividades de la ingeniería del software que aún necesita de unas bases técnicas sólidas. Cada paso que se da en las áreas de estimación, seguimiento, interpretación de los datos, etc., acercan un proyecto al objetivo de calidad y cumplimiento de tiempo y presupuesto estimado. Los sistemas de entrenamiento que presentan diferentes escenarios a los directores de proyectos permiten superar la ausencia de datos relacionados con la gestión de proyectos.

En los actuales entornos de gestión, el director tiene que tomar decisiones en función de una descripción aproximada del entorno del proyecto. El hecho de que la mayoría de la información de los proyectos finalizados es desconocida o contiene un elevado grado de incertidumbre hace difícil averiguar los parámetros del proyecto. Es más, los datos disponibles provienen de múltiples fuentes y la integración de todos esos datos no es una tarea sencilla de realizar con las herramientas software actuales.

Con estas ideas en mente, hemos desarrollado los principales módulos del sistema SEGESOFT para poder apoyar el entrenamiento de la gestión del software. El sistema implementa diferentes técnicas para manejar diversos usos de los datos disponibles en la herramienta de gestión de proyectos.

2 Integración de técnicas para la gestión del software

Nuestra herramienta integra diversos módulos que corresponden a diferentes técnicas utilizadas en la gestión de proyectos. En las siguientes subsecciones presentamos brevemente algunas de las técnicas implementadas y la estructura general del sistema.

2.1 La simulación y los modelos dinámicos

Los modelos dinámicos y la aparición de entornos de simulación potentes y amigables (Stella, Vensim, iThink, Powersim, etc.) aplicados a la gestión de proyectos software al comienzo de los años 90 permitieron la creación de herramientas (llamadas normalmente simuladores de proyectos), que posibilitan la simulación del comportamiento de dichos proyectos. Con estas herramientas de simulación, los responsables de proyectos software pueden "experimentar" sin ningún tipo de coste el efecto que tiene sobre el proyecto la aplicación o no de diferentes políticas de gestión [2][5].

Un modelo dinámico para proyectos software permite conocer la evolución del proyecto. Pero es importante resaltar que dicha evolución y, por tanto, la consecución de los objetivos del proyecto va a depender de: a) las estimaciones iniciales de los recursos necesarios, b) las políticas de gestión que se apliquen, c) las características del proyecto y d) las características de la empresa de desarrollo.

Un modelo dinámico para proyectos software incorpora una serie de parámetros que permiten definir los aspectos anteriores, es decir, estimaciones iniciales, políticas de gestión y características del proyecto y de la organización. Un simulador de proyectos software permite realizar diferentes análisis del proyecto dependiendo del estado en el que se encuentra:

1. Un análisis a priori del proyecto antes de comenzar su ejecución.
2. Una monitorización del proyecto durante la ejecución del mismo para ir adaptando las estimaciones iniciales a la evolución del proyecto.
3. Un análisis post-mortem del proyecto una vez finalizado para conocer cómo podrían haberse mejorado los resultados del mismo.

En definitiva, un simulador de proyectos software permite responder a las siguientes cuestiones: "¿qué ocurrirá si..?" antes de comenzar, "¿qué está ocurriendo..?" durante la ejecución y "¿qué habría ocurrido si..?" una vez que el proyecto ha finalizado.

2.2 Descubriendo conocimiento o aprendizaje automático

Las técnicas y herramientas computacionales diseñadas para sustentar la extracción de conocimiento útil a partir de bases de datos se conocen tradicionalmente como aprendizaje automático (machine learning en inglés). Recientemente, términos como minería de datos (data mining) o descubrimiento de conocimiento en bases de datos (knowledge discovery in databases o KDD) se utilizan frecuentemente en vez de aprendizaje automático. En general, las técnicas así denominadas tratan de extraer automáticamente una información útil para la toma de decisiones o la exploración del origen de los datos [1].

Un proceso estándar de aprendizaje automático está compuesto por varios pasos tales como preparación de los datos, selección, limpieza, minería propiamente dicha e interpretación de los resultados. Por tanto, la minería de datos se puede considerar como un paso dentro de un proceso más global; este paso consistiría en aplicar algoritmos específicos para extraer patrones a partir de los datos.

2.3 Otras técnicas

Las medición es esencial para gestionar, evaluar, predecir y mejorar la calidad de los procesos y productos software. Las actuales herramientas de gestión de proyectos trabajan con tiempo y recursos pero no consideran los aspectos de calidad. La herramienta SEGESOFT proporciona facilidades para el control de calidad.

Para controlar la calidad de los productos y procesos es necesario definir modelos de calidad para ambos. Además es necesario definir diferentes modelos para los diversos componentes generados durante el ciclo de vida del software. Algunos de los modelos siguientes son necesarios para analizar el estado del proyecto.

Entre los modelos disponibles para la evaluación de los procesos, los más influyentes son el *Capability Maturity Model* (CMM), ISO/IEC 12207, ISO 9000, el modelo BOOTSTRAP y el ISO 15504 (SPICE). Los modelos para evaluar la calidad de los productos varían desde los modelos jerárquicos fijos como Boehm, FCM (Factor Criteria Metric) de McCall e ISO 9126 (actualmente bajo revisión) a enfoques más flexibles como QMS (Quality Management Subsystem) de Kitchenham. Todos estos modelos ayudan a clarificar qué aspectos de calidad son considerados y porqué.

Aunque los modelos de proceso ayudan a incrementar la calidad de los productos que producen, la relación entre modelos de procesos y calidad de productos no está usualmente clara en los modelos descritos anteriormente. Para aliviar este problema, Satpathy et al. [7] definen un modelo genérico como una plantilla que puede ser instanciada para ser el modelo de calidad de cualquier proceso individual.

3 Estructura del Sistema

3.1 Arquitectura General

La arquitectura de la versión actual del sistema está formada por un conjunto de componentes desarrollados sobre una base de datos para el almacenamiento de diferentes tipos de métricas. La importancia de disponer de un módulo que permita registrar todo tipo de métricas ha sido reseñada en los trabajos de Paul et al. [3], donde se propone una estructura general para una base de datos de proyectos con una estructura relacional de la información, aunque no se presenta ninguna herramienta que materialice dichas ideas.

En este artículo se intenta dar un paso más allá integrando sobre una misma base de datos de métricas diferentes herramientas analíticas y proporcionando una estructura abierta que permita incorporar futuras necesidades. Desde el punto de vista de la implementación se ha tratado de reutilizar en lo posible software desarrollado o en desarrollo, licenciado como "freeware" en numerosas ocasiones. Se ha utilizado Java 2 SDK (versión 1.2 ó 1.3) para el desarrollo.

La estructura general del sistema se representa en la Figura 1, donde se incluyen los principales módulos del sistema, los cuales son suficientemente coherentes e independientes entre sí. En dicha figura las flechas representan una relación de uso de la información. A continuación se describen brevemente la función de cada módulo:

- El módulo de *Guiones de Entrenamiento* organiza las lecciones, datos y guías para el uso de la herramienta.
- El módulo de *Medición* permite recoger la información relativa a las métricas de los proyectos.
- El módulo de *Estimación* incluirá algunos métodos típicos de estimación de proyectos.
- El módulo de *Seguimiento* está específicamente diseñado para monitorizar la evolución del proyecto. Podría considerarse como parte del módulo de medición, pero se diferencia por su interés en resaltar información relativa al estado de un proyecto en curso durante y al final del mismo.
- El módulo de *Simulación* permite al usuario modelar la organización utilizando un enfoque de modelos dinámicos para obtener datos simulados.

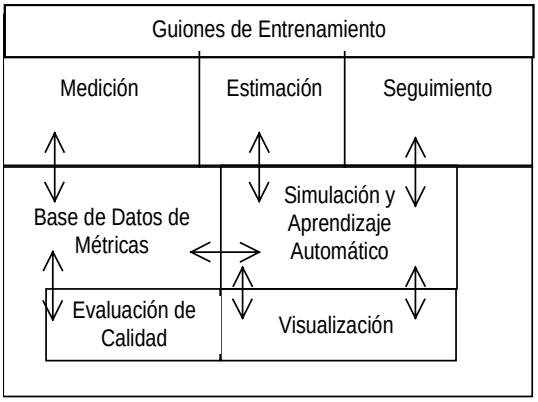

Figura 1. *Estructura general de los componentes del sistema*

- El módulo de *Aprendizaje Automático* implementa algunos algoritmos utilizados para resumir grandes volúmenes de datos.
- La *Base de Datos de Métricas* que es el corazón del sistema donde se almacena toda la información.
- El módulo de *Visualización*, está siendo desarrollado con el objetivo de permitir la presentación de información de interés para el gestor en forma de gráficos y diagramas
- El módulo de *Evaluación de Calidad* implementa criterios para la evaluación continua de la calidad de un proyecto.

3.2 Base de Datos

El modelo de la base de datos permite almacenar diferentes proyectos ejecutados por una organización. La Figura 2 muestra de forma esquemática e informal la estructura de la información. La raíz presenta la organización en su globalidad, que se estructura por una parte en el conjunto de proyectos (diferenciando entre activos y pasados) y en la línea base de proyectos (que guarda las mediciones globales para proyectos finalizados) desglosadas por tipos de proyectos. Cada uno de los proyectos tendrá diferentes versiones correspondientes cada una a las diferentes planificaciones que se han realizado a lo largo del tiempo.

Cada una de las versiones del proyecto continúa representándose de forma jerárquica. Por ejemplo, en el caso de un proyecto activo, este se desglosa en diferentes versiones para reflejar las replanificaciones que se realizan durante el transcurso de la ejecución del proyecto. Dependiendo de éste se encuentran las estructuras jerárquicas básicas siguientes:

- RBS (Resource Breakdown Structure): Recursos disponibles para el proyecto.
- WBS (Work Breakdown Structure): Tareas del proyecto (desglosadas de forma jerárquica, a partir de las cuales se construye el diagrama Gantt de planificación).
- PBS (Product Breakdown Structure): Productos generados a lo largo del proyecto.
- PRBS (Process Breakdown Structure): Procesos del proyecto (tipos de actividades del proyecto).
- MBS (Metrics Breakdown Structure): Información de métricas de diferentes tipos.
- SBS (Simulation Breakdown Structure): Parámetros y estructuras utilizadas para la simulación del comportamiento del proyecto.
- MLBS (Machine Learning Breakdown Structure): Parámetros de los algoritmos de aprendizaje automático utilizados.

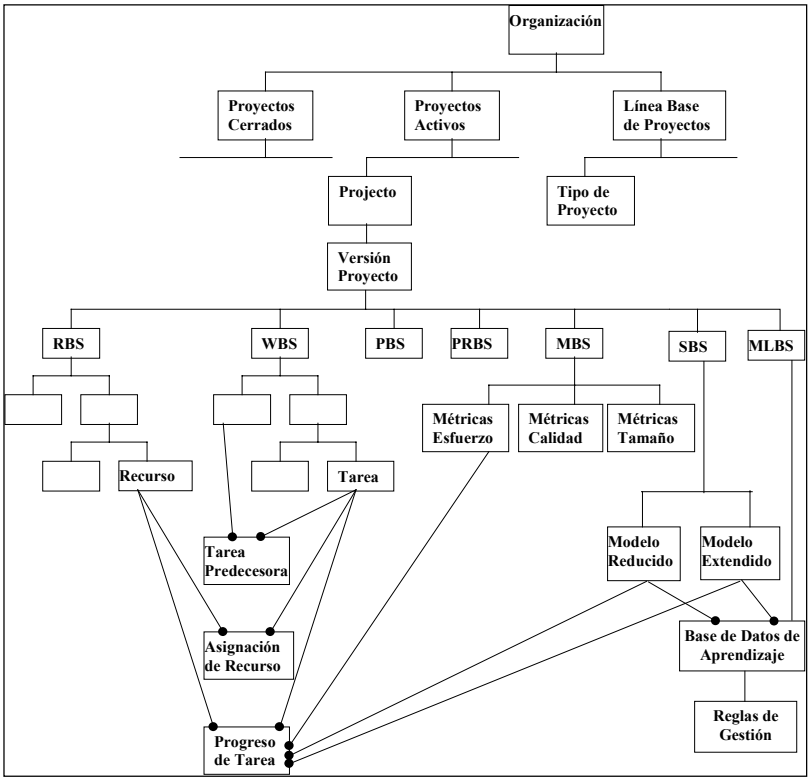

Figura 2. *Esquema de la Estructura de la Base de Datos*

Estas estructuras básicas jerárquicas se van desglosando en otros nodos que representan mayor nivel de detalle. Existe otra estructura que relaciona nodos correspondientes a la jerarquía. Como ejemplo, en la figura anterior se pueden ver las predecesoras de una tarea, los recursos planificados en una tarea, o el progreso y esfuerzo realizado en la ejecución de una tarea.

3.3 Interfaz de Usuario

En relación con la interfaz de usuario, se pueden apuntar una serie de líneas generales que homogeneizan el uso y navegación del usuario a través de las estructuras anteriores. En la Figura 3 se representa el aspecto de una pantalla.

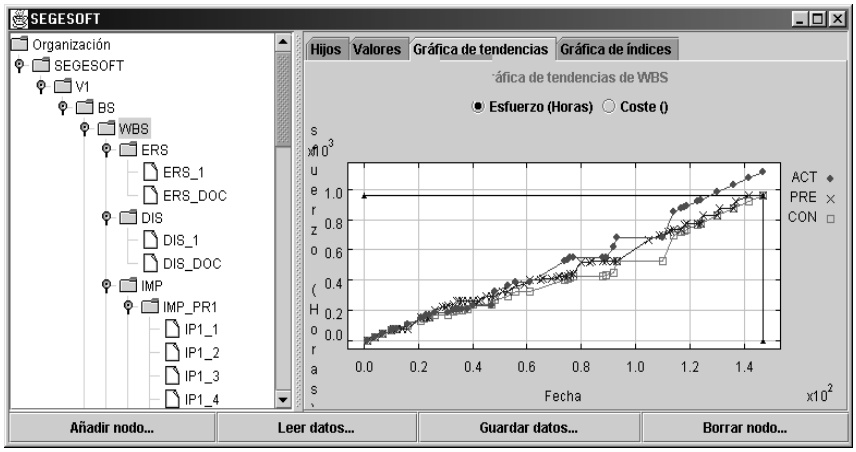

Figura 3. *Ejemplo del interfaz gráfico*

En la parte izquierda se dispone de un navegador que permite localizar rápidamente cada uno de los nodos de las estructuras anteriores y las opciones básicas de crear, borrar, mover y renombrar los diferentes elementos. En el panel de información se representará la información de cada objeto. En general, los tipos de información a mostrar son los siguientes:

- Datos correspondientes al objeto (sus atributos).
- Relaciones (una o más pestañas para permitir modificar los datos de las relaciones de este objeto).
- Estructura en forma de matriz (correspondientes al objeto y sus descendientes) de forma que se puedan realizar actualizaciones y visualización rápida. Por ejemplo, en una WBS se mostraría una matriz todos los descendientes en filas y en columnas los atributos más relevantes.
- Estructura en forma de gráfico (correspondiente al objeto y sus descendientes). Por ejemplo, mostrar gráficamente el PERT, GANTT o WBS.
- Otros datos en forma gráfica (por ejemplo, curvas de progreso).

Todos los objetos (navegador y cada uno de los paneles) están diseñados de forma que se puedan reutilizar en diferentes contextos. Por ejemplo, en la figura 3 se ha seleccionado un proyecto (SEGESOFT) al cual se accede en su versión primera (V1) a la estructura de partición del trabajo (WBS) de la cual dependen las tareas. A la derecha se han seleccionado las pestañas correspondientes a las curvas de valor acumulado que muestran la evolución temporal de esfuerzo o coste (actual o real, presupuestado y conseguido). Si se seleccionase una tarea (por ejemplo el diseño: DIS) se mostrarían los mismos datos relacionados solamente con dicha tarea a cualquier nivel de detalle.

3.4 Modelado y simulación

El objetivo de este módulo es el de recoger el funcionamiento y los principales componentes de entornos de simulación tan conocidos como Vensim, Stella, Powersim, etc...

Este módulo implementa un modelo dinámico simplicado que se puede utilizar para simular el comportamiento temporal de un proyecto. Para ello, se debe introducir los tres grupos de parámetros siguientes (ver figura 4): los parámetros relacionados con el proyecto (tamaño, número inicial de técnicos, etc.), los parámetros relacionados con la organización de desarrollo (retrasos medios en la contratación y despido de los técnicos, etc.) y los parámetros de control de la simulación (duración e incremento temporal que se utilizará en la simulación del modelo dinámico).

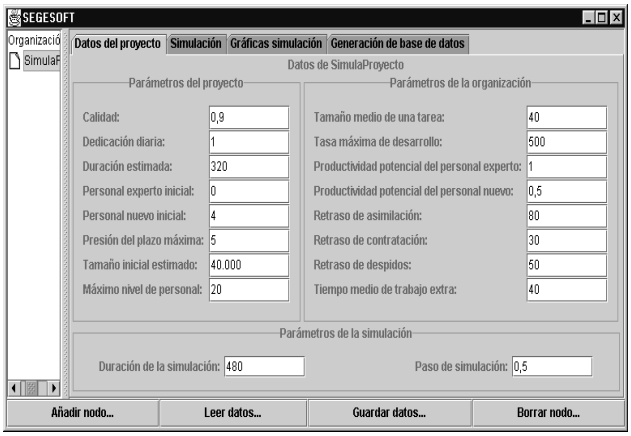

Figura 4. *Parámetros del proyecto*

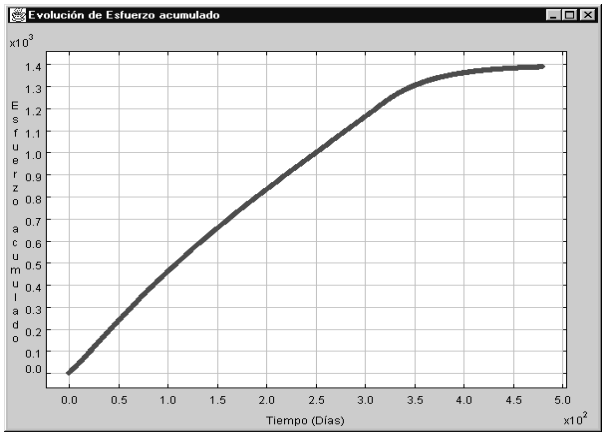

Figura 5. *Evolución del esfuerzo acumulado*

Una de la
varia vecto
que s

Este vecto
tanta os se
utiliz cada
simul tros,
obter ables
del m rvalo
de va

3.5 A

Usan r del
volur s de
apren listas
de de os de
regla radas
medi Esto
hace

Por e para
amba ra el
módu 6. En
ella RT y
DEL está
defin iable
EFFC iable
DEL ún se
puede 73] y
[149, de la
simul es de
los pa e las
varia este
prepr nente
menc amos
intere s de
apren caso
que r

Una vez que las etiquetas son generadas, se proporciona al algoritmo cuántas reglas deseamos generar y la tasa de error permitida. Por último, un valor debe indicar el número de pruebas a efectuar: dado que el proceso de búsqueda es probabilístico este factor indica la cobertura de la búsqueda realizada, es decir, el número de puntos seleccionados en el proceso de generación inicial de las reglas.

A continuación, describimos brevemente el algoritmo que se implementó en este módulo (ver figura 7). Las reglas de decisión son generadas para informar si la ejecución del proyecto se realiza conforme a las estimaciones iniciales y deben proporcionar un conjunto de decisiones a tomar para evitar la violación de éstas. El algoritmo sigue una estrategia de escalada monótona con multiarranque.

```
Para cada regla inicial r
  Seleccionar aleatoriamente un ejemplo e de BD
  Para cada parámetro p de BD
    Ordenar BD por p crecientemente
    Buscar los ejemplos más cercanos a e (menor y mayor) con diferente clase
    Establecer ambos valores como extremos del intervalo de ese parámetro
Para cada regla r que pueda cubrir más ejemplos de BD
  Elegir aleatoriamente un parámetro p
  Expandir el parámetro p por un extremo (izquierda o derecha)
```

Figura 7. *Pseudocódigo*

Un ejemplo de salida gráfica que se obtiene en este proceso de aprendizaje se muestra en la figura 8. Dado que estamos interesados en conseguir "buenos" proyectos, los intervalos resaltados en negro indican los posibles valores que deben tomar los parámetros del proyecto para finalizarlo en las condiciones iniciales estimadas al principio.

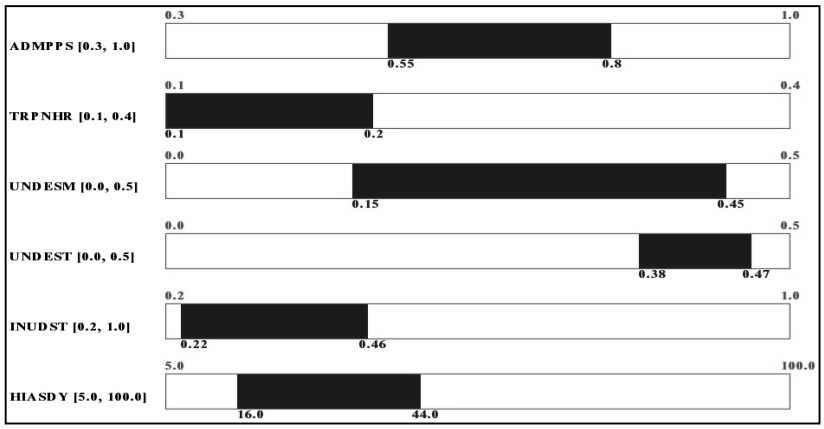

Figura 8. *Presentación gráfica de la primera regla*

El significado de la figura 8 es el siguiente: si el parámetro ADMPPS (dedicación media del personal) está en el 55% y el 80% y TRPNHR (dedicación media de los técnicos expertos a formación) está entre el 10% y el 20% y UNDESM (infraestimación inicial del esfuerzo estimado) está entre 15% y 45% y UNDEST (infraestimación inicial del número de tareas estimadas) está en [38%, 47%] y INUDST (infraestimación inicial del número de técnicos estimados) está en [22%, 46%] y HIASDY (retraso medio en la contratación y adecuación de los técnicos nuevos) está entre [16, 44] días entonces el proyecto en cuestión evolucionaría según el criterio definido como "bueno" por parte del responsable.

Además las reglas pueden filtrarse considerando sólo los atributos cuyos intervalos no recogen los valores de la estimación inicial para la simulación. Es decir, si el valor inicial del parámetro INUDST se encuentra en el rango [22%, 46%] entonces la condición sobre este parámetro puede desaparecer de la regla anterior, simplificándose su lectura.

4 Conclusiones y desarrollos adicionales

Un uso inteligente de los datos del proyecto puede tener considerables efectos en los resultados del proyecto. Esto a su vez depende de la habilidad y experiencia del director del proyecto para abordar múltiples fuentes de información. El entrenamiento en la gestión de proyectos software es un requisito para avanzar en la profesión de ingeniero del software. Este entrenamiento se puede conseguir mediante el uso de entornos educativos que incorporen métodos de extracción y análisis de información.

Hemos presentado en este trabajo la primera versión de un entorno que ha sido desarrollado con el propósito de integrar diferentes aspectos de la gestión de proyectos. Uno de los principales beneficios de nuestro enfoque es que se construye con la intención de explorar el uso de diferentes tipos de técnicas desarrolladas formalmente para otras disciplinas de la ingeniería.

Los posteriores desarrollos y esfuerzos se dirigirán hacia la mejora de la visualización de toda clase de datos, uso de la información de múltiples orígenes o fuentes, la introducción de nuevos métodos, etc. Hemos detectado como un prerrequisito básico la falta de una metodología para interpretar toda la información que el director de proyecto reúne. Nuestra herramienta, además de conseguir un incremento en la información presente en las herramientas anteriores, permite evaluar acciones correctivas en forma de reglas de gestión para alcanzar los objetivos deseados.

Agradecimientos

El proyecto SEGESOFT está subvencionado por CICYT: TIC 99-0351. La participación de SYSECA Cantábrico (ahora denominada THALES Information System) ha sido de gran ayuda para nuestro trabajo.

5 Referencias

[1] U. Fayyad, Piatetsky-Shapiro G, Smyth P. *The KDD Process for Extracting Useful Knowledge from Volumes of Data*. Comm. of the ACM 1996; 39(11): 27-34

[2] P. Mandl-Striegnitz et al., "*Simulating Software Projects - An Approach for Teaching project Management*", INSPIRE'98, pp. 87-98

[3] R. A. Paul, Tosiyasu L. Kunii, Y. Shinagawa, and M. F. Khan, *Software Metrics Knowledge and Databases for Project Management*, IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 1, 1999 255-264

[4] I. Ramos, J. Aguilar, J. C. Riquelme, M. Toro, *A new method for obtaining software project management rules*. Software Quality Management VIII (SQM2000). pp. 149-160. Greenwich (London, U.K.), 2000.

[5] A.G. Rodrigues and T.M. Williams, "*System dynamics in software project management: towards the development of a formal integrated framework*", European Journal of Information systems, Nº 6, pp. 51-66, 1997.

[6] M. Ruiz, I. Ramos, *A Dynamic Estimation Model for the Early Stages of a Software Project*, Software Process Simulation Modelling Workshop (Prosim2000), London (U.K.)

[7] M. Satpathy, R. Harrison, C. Snook, and M. Butler, "A Generic Model for Assessing Process Quality", presented at 10th International Workshop on Software.