DISEÑO E IMPLEMENTACION DE UN AMBIENTE CASE PARA SISTEMAS DE TIEMPO REAL.

Miriam Judit Marinangelli [1]

José Oscar Spognardi [1]

Ing. Armando De Giusti[2]

C.C. Patricia M. Pesado [3]

Laboratorio de Investigación y Desarrollo en Informática [4]
Departamento de Informática - Facultad de Ciencias Exactas
Universidad Nacional de La Plata

RESUMEN

Se presenta el análisis e implementación de una herramienta CASE para Sistemas de Tiempo Real basada en la metodología SSEM de Keller y Shumate, la cual ha sido desarrollada como Trabajo de Graduación en la Universidad Nacional del Centro de la Pcia. de Bs. As. con la dirección de Profesores de la UNLP.-

Asimismo se discute el sistema de simulación asociado con la herramienta CASE que permite representar situaciones de funcionamiento (normal y anormal) a fin de evaluar la respuesta del sistema y el cumplimiento de los requerimientos establecidos por el usuario.-

---

[1] Alumno de Trabajo Final del Dto. Computación y Sistemas de la UNCPBA.

[2] Investigador Principal CONICET. Prof. Titular Ded. Exclusiva, Dpto. de Informática, Fac. de Ciencias Exactas, UNLP. Director del LIDI.

[3] Profesional Adjunto CIC. Prof. Adjunto Dedicación Exclusiva, Dpto. de Informática, Fac. de Ciencias Exactas, UNLP

[4] Calle 50 y 115 - 1er piso - (1900) La Plata - Buenos Aires
Tel: 54-21-22-7707 - 54-21-4-2738
Fax: 54-21-22-7707
E-mail: lidi@ada.info.unlp.edu.ar

INTRODUCCIÓN

Sistemas de Tiempo Real

Puede definirse un sistema de tiempo real (STR) como un sistema controlado por software o firmware, que realiza funciones de procesamiento cumpliendo restricciones de tiempo especificadas. Normalmente incluye un conjunto de dispositivos de hardware independientes que operan a distinta velocidad [DeGi91] [Maga86]

El objetivo principal del control de un STR es optimizar el uso de estos recursos, cumpliendo las restricciones de performance requeridas. Dentro de los STR se encuentran los sistemas de control y supervisión de procesos, que normalmente deben correr en forma continua en un modo automático y con alta confiabilidad.

El software para STR comprende funciones tales como: control digital directo, registro de datos, gerenciamiento de la información, interfase con el operador, monitoreo de alarmas.

En el desarrollo de STR se encuentran algunas dificultades que lo diferencian de otro tipo de aplicaciones. Entre estos "problemas" se encuentran [Niel90] [Fort85]:

- Manejo de restricciones de tiempo y performance.
- Control directo de recursos de hardware.
- Procesamiento de mensajes asincrónicos y de distintas prioridades.
- Mantenimiento de colas y buffers de almacenamiento.
- Modelización de la concurrencia de procesos.
- Protección de datos compartidos por los procesos concurrentes.
- Comunicaciones entre procesos y entre procesadores.
- Scheduling y dispatching de procesos que utilizan recursos compartidos.
- Relación con un reloj de tiempo real.
- Detección y corrección de condiciones de falla.
- Testeo y puesta a punto de un sistema que puede estar distribuido en distintos procesadores.
- Generación de herramientas de simulación cuando el hardware no está disponible durante el desarrollo.
- Selección de la estructura de hardware adecuada.

Características principales de los STR y su efecto

1) No pueden volver atrás ⇒

- Es obligatorio asegurar confiabilidad.
- Se debe prever algún grado de tolerancia a fallas.

- El costo de los errores post-implementación es mayor.
- Debe ponerse mayor esfuerzo de verificación en el desarrollo.

2) La secuencia temporal de entradas está determinada total o parcialmente por el mundo real, no por el operador ⇒

- Hay sincronización de eventos.
- Se debe evitar deadlock.
- Se debe evitar el overloading.
- Se deben administrar los conflictos por recursos.

3) Las demandas del ambiente suelen ser paralelas, no secuenciales ⇒

- Distribución del procesamiento.
- Comunicación de procesadores y procesos.
- Scheduling / dispatching de procesos.

4) Existen restricciones de tiempo y performance ⇒

- La verificación o validación funcional no es suficiente.
- Se necesita modelizar las restricciones.
- Es necesario buscar las condiciones de máximo riesgo.

5) Suelen ser sistemas de tiempo infinito ⇒

- Es conveniente prever más de un modo de operación (por ej, normal y en falla).
- Exigen cierta redundancia de hardware y software.

Procesos del mundo real

Dado el modelo abstracto de los sistemas de tiempo real y el hecho de que las aplicaciones son descompuestas (de alguna manera) en procesos, consideremos las distintas clases de procesos que se encuentran en ambientes de tiempo real. Las clases de procesos a ser soportados son factores determinantes al seleccionar medios de descripción de comportamiento apropiados, mapeo de procesos, y estructura de recursos [Laws92].

Procesos periódicos y aperiódicos

Podemos caracterizar a los procesos por su periodicidad, es decir, en que instante de tiempo el proceso ocurre:

Procesos periódicos: son ejecutados repetidamente a intervalos regulares.

Procesos aperiódicos: son ejecutados en momentos de tiempo arbitrarios.

Los procesos periódicos se relacionan con eventos determinísticos, típicamente donde se requiere alguna forma de sampling para adquisición de datos y/o control. Por otro lado, los procesos aperiódicos ocurren no determinísticamente debido a la existencia de alguna condición específica (evento).

Procesos estáticos y dinámicos

Los procesos pueden ser partes permanentes del STR o pueden nacer, vivir y morir dinámicamente, lo que lleva a las siguientes propiedades:

Procesos estáticos: son parte permanente del STR durante su ejecución

Procesos dinámicos: son creados durante la operación del STR como se requiera y removidos cuando ya no se necesitan.

Una aproximación más simple puede aplicarse en el caso estático, por ejemplo, vía mapeo directo del problema a una estructura de recursos apropiada. Por el otro lado, a medida que los requerimientos de flexibilidad crecen (particularmente en presencia de estructura de recursos limitada) se requieren facilidades de alocación de procesos. La combinación de procesos dinámicos y requerimientos de procesos periódicos introduce complejidad en la solución.

Importancia relativa de procesos

En aplicaciones de tiempo real es posible categorizar la importancia de los procesos [Rama89]:

Procesos críticos: deben cumplir indefectiblemente con las restricciones; en otro caso los resultados podrían ser catastróficos.

Procesos esenciales: tienen restricciones y son importantes en la operación del sistema, pero no causan una catástrofe si no terminan a tiempo.

Procesos no esenciales: pueden no cumplir sus restricciones sin afectar al sistema en el futuro próximo, pero podrían afectarlo a largo plazo (por ej, procesos de mantenimiento y backup).

Aquí también, basado en la importancia de procesos y la mezcla de varias formas de procesos que deben implementarse, pueden requerirse soluciones variables de descripción de comportamiento, mapeo de procesos y estructuras de recursos. La presencia de todas las formas en el mismo sistema requiere normalmente un mecanismo de prioridad y el uso de soluciones de scheduling y alocación de recursos sofisticadas.

Ejecución concurrente y paralela

Los términos concurrente y paralelo [Laws92] con frecuencia son usados intercambiablemente:

Ejecución concurrente: la ejecución de múltiples procesos que han comenzado pero no se completaron, y están activos en el mismo punto de tiempo. Los procesos pueden ser ejecutados por un único o múltiples elementos de procesamiento (PEs).

Ejecución paralela: la ejecución de múltiples procesos (o porciones de procesos) por múltiples PEs. Los PEs pueden estar poco o muy acoplados.

Importancia y Complejidad de una herramienta CASE para Tiempo Real

Los sistemas de tiempo real necesitan de herramientas mas potentes que las convencionales dadas sus características intrínsecas. Los CASE orientados deben soportar las posibilidades de especificación, prototipacion y simulación. [Ward85]

En tanto que el prototipo es un modelo del sistema de software, la simulación es un modelo del entorno al cual el sistema del software debe responder y controlar. La simulación proporciona una visión previa del comportamiento del sistema de software en su entorno verdadero de tiempo real. Las entradas verdaderas en tiempo real desde el entorno deben estar provistas de mecanismos para verificar la exactitud e integridad del modelo de diseño del sistema. Con la ayuda de las herramientas CASE de simulación, el programador puede encontrar los errores de diseño incluso antes del comienzo de la fase de codificación [McCl92].

En este trabajo se expone en forma muy sintética el desarrollo de una herramienta CASE que trata de cubrir algunos de los requerimientos mencionados precedentemente y que se combina con un simulador que trabaja integrado en el ambiente.[Mari95]

El desarrollo constituye una continuación de una serie de trabajos de investigación en el área de Sistemas de Tiempo Real realizados por el LIDI con participación de alumnos avanzados de la UNLP y la UNCPBA.

EL AMBIENTE ISTRAC

El ambiente desarrollado, llamado Ingeniería de Sistemas de Tiempo Real Asistida por Computadora (ISTRAC), permite especificar STRs desde el punto de vista del análisis estructurado clásico, con algunas variantes menores; y a partir de dicha especificación, permite definir las restricciones temporales críticas utilizando las herramientas adecuadas y luego asignar ese modelo abstracto a una red de procesadores y elementos de hardware (sensores, teleactuadores, etc.) detallando los medios de comunicación entre elementos de hardware, la redundancia, etc.; también aquí se da un tratamiento especial a los requerimientos temporales críticos.[Hatl88][Kell92]

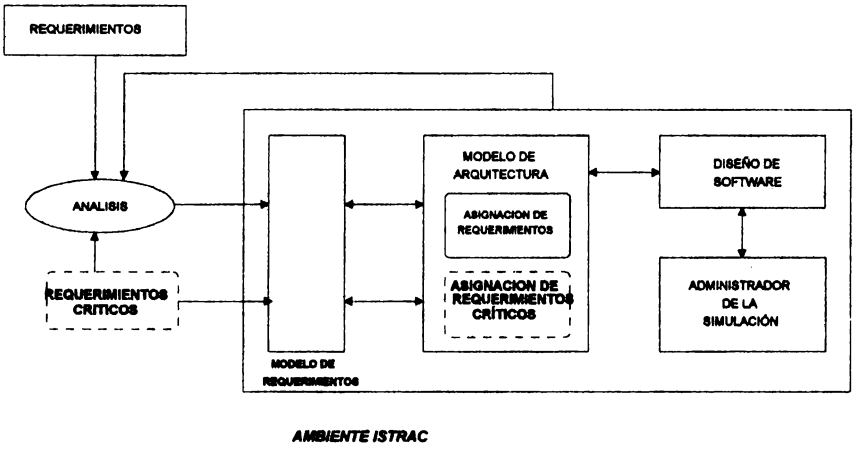

fig. 1 El modelo especificado por ISTRAC

Finalmente, a partir del diseño especificado en el ambiente, se pasa a diseñar el software de cada componente del diseño en forma individual. En este sentido, se determina la concurrencia dentro de cada componente y a partir de aquí, es posible simular el comportamiento de los módulos de software en función de los requerimientos temporales críticos asignados a cada componente.

Entre las principales características del ambiente podemos mencionar: las facilidades para iterar hacia atrás y hacia adelante entre las diversas fases durante la especificación, las reglas de auto-chequeo y validación de las especificaciones a medida que se crean, y los procedimientos de auto-consistencia en una fase a medida que se modifican los productos de otra fase.

En cuanto al programa, responde al modelo "GUI, Graphical User Interfaces" tipo menu-driven y form-driven, es decir, presenta una interfase gráfica y es manejado por medio de menús y formas o cajas de dialogo. [Grub95][Hami93]

Sintéticamente ISTRAC nos provee de un manejo de edición gráfica, basado en la metodología SSEM que facilita la especificación con:

- entorno de desarrollo gráfico flexible y fácil de utilizar
- reglas automáticas de chequeo sintáctico y semántico del producto

- estructura adecuada al producto (por ejemplo, jerarquía de niveles en el DFD)
- producto resultante consistente y válido, como criterio de salida para la transición a la siguiente fase
- comunicación con los otros editores y actualización automática (por ejemplo, si eliminamos un proceso en el DFD, luego deberá desaparecer del módulo al cual estaba asignado en el AFD)

Todos los editores están ligados y relacionados internamente (es decir, en forma trasparente al usuario), de manera que las modificaciones y/o "vuelta atrás" (feedback) son efectuadas automáticamente; facilitando así las frecuentes iteraciones entre las etapas y editores.

Los editores disponibles en la fase de Análisis del Sistema son:

- Editor de Diagramas de Flujos de Datos y de Control(DFD)
- Editor de Especificaciones de Tiempo de Respuesta (RTS)
- Diccionario de Datos (DD)

En la fase de Diseño del Sistema:

- Editor de Diagramas de Flujos de Arquitectura (AFD)
- Editor de Diagramas de Interconexión de Arquitectura (AID)
- Editor de Asignación de Tiempos de Respuesta (RTA)

En las fases de Análisis y Diseño del Software:

- Editor de Diagramas de Comunicación de Tareas (TCG)

En cuanto a los diagramas de contexto, se supone que el usuario puede crearlo mediante el mismo editor de DFD, por lo tanto, no existe el editor de diagramas de contexto explícitamente.

Cabe señalar que, según la metodología explicada, el diagrama de flujo de control puede ir o no separado del DFD, en nuestro caso, decidimos editor conjunto. Por lo tanto, el DFD es también el diagrama de flujo de control CFD.

EL SIMULADOR ASOCIADO CON EL CASE

El Administrador de la simulación ADSIM es una herramienta externa del ISTRAC, es decir, se ejecuta como un programa individual y se encarga de la simulación. Mediante un conjunto de facilidades es posible crear situaciones críticas, las cuales se pueden presentar en cualquier momento de la operación del sistema.[Kuma94]

Es importante mencionar que la simulación se efectúa antes de la etapa de diseño detallado, vale decir, que aún no se ha especificado el software cuando utilizamos el ADSIM. Esta característica permite estimar si el diseño del sistema cumple con las restricciones especificadas e incluso los resultados de la simulación ayudan a comparar los diferentes diseños candidatos o alternativos.

Los editores de ISTRAC permiten especificar un proyecto hasta la etapa de diseño top-level, logrando un modelo consistente y bien definido. En esa etapa tenemos dos especificaciones que son la entrada para el administrador de la simulación: las especificaciones de tiempos de respuesta y los CSCIs (definidos mediante tareas que operan concurrentemente).

Durante las fases de análisis y diseño los requerimientos fueron asignados a módulos (de hardware y software); entre esos requerimientos es importante destacar los requerimientos temporales críticos. En este caso, los requerimientos críticos han sido asignados a CSCIs, por lo tanto, dentro de algunos CSCI estará centrado el procesamiento para que la salida se pueda efectuar en los límites de tiempo establecidos.

a) El camino o thread

El administrador de la simulación permite marcar el camino o thread que hace un requerimiento crítico dentro de un CSCI, pasando de una tarea a otra, y a partir de allí, mediante especificaciones adicionales es posible simular el comportamiento del CSCI.

Luego, a partir del thread, especificamos para cada tarea el tiempo que insumirá en atender la señal. Esto permite simular cada señal crítica que debe ser atendida por el CSCI.

b) Condiciones externas

Las condiciones externas son aquellas situaciones externas al thread (que pueden venir desde otro CSCI o desde él mismo) que lo afectan durante su "ejecución". Por ejemplo, una sincronización entre tareas o una llegada de determinado evento, una alarma, que interrumpe por prioridad la ejecución de una tarea del thread.

Antes de comenzar la simulación, es posible establecer este tipo de condiciones, el instante preciso de tiempo en que se darán, la duración y la tarea en la que se presentan las condiciones externas.

c) Parámetros de la simulación

Los parámetros de la simulación son los valores necesarios para determinar el tipo y duración de la simulación. El primero es el número de corridas, que permite efectuar varias corridas continuas de la simulación para obtener valores más aproximados a la realidad.

También se debe especificar la duración de cada corrida, es decir, el tiempo que creemos necesario que el sistema este funcionando para encontrar situaciones reales que puedan evaluarse.

Otro parámetro es la carga, es decir, la frecuencia a la que llega la señal a simular, que puede ser o no la tasa de repetición especificada en el DD. Para definir la carga hay que indicar el tipo de distribución probabilística y el tiempo medio y desviación estándar, si corresponde (por ejemplo, exponencial con tiempo medio de 5 segs., o normal con tiempo medio de 27 segs. y desviación 2 segs.)

d) Resultados de la simulación

Los resultados de la simulación permiten observar el comportamiento global e individual de las tareas. Los tiempos totales medios y máximos de servicio (de las tareas) y espera (de las señales) y los tiempos individuales para cada tarea, tanto de servicio como de espera. Luego, comparando estos resultados con los tiempos especificados en el RTS y RTA el diseñador sacará sus conclusiones.

ASPECTOS DE IMPLEMENTACION

El programa fue desarrollado en el ambiente Microsoft Visual C++ [Krug94][Micr94] versión 1.0, utilizando un equipo 486 DX40 con 8 MB de RAM, disco rígido de 260 MB y monitor SVGA, utilizando una resolución de 800 x 600.

Los requerimientos mínimos para ejecutar el programa son: Windows 3.1, 486 DX (o 486SX con co-procesador) y 3 MB de espacio en disco rígido. Recomendamos utilizar la resolución 800 x 600, aunque es posible utilizar cualquier otra resolución disponible.

En cuanto a la herramienta de simulación, es deseable disponer de 16 MB de RAM o como alternativa, la memoria virtual suministrada por Windows.

CONCLUSIONES

Este trabajo pone de manifiesto que la especificación de sistemas de tiempo real y el proceso de desarrollo pueden ser asistidos por una herramienta automatizada que guíe al ingeniero y le facilite la aplicación de una metodología con todos sus pasos en el orden correcto.

El ambiente desarrollado está orientado a la Ingeniería de Sistemas de Tiempo Real, permitiendo especificar las restricciones típicas de tiempo real y automatizando el proceso de verificación y chequeo de la asignación de esas restricciones al diseño logrado.

Además, se aplican reglas de auto-consistencia y chequeos en la mayoría de las operaciones válidas y se dispone de mecanismos de actualización automática de una componente de la especificación al modificar componentes relacionados. Estas características logran que el usuario se despreocupe de la consistencia de sus especificaciones y dedique toda su atención al problema en sí mismo.

Finalmente, la posibilidad de disponer de una herramienta que en función de la especificación permita estimar el comportamiento de un STR tiene evidentes consecuencias en ahorro de esfuerzos de implementación, ya que el diseñador, mediante la simulación tiene estimaciones del comportamiento del sistema durante el proceso de diseño y, evidentemente antes de la implementación del sistema. Al mismo tiempo, en esta etapa, los resultados de las simulaciones contribuyen a la comparación entre diseños candidatos o alternativos.

La herramienta CASE desarrollada está disponible en la UNLP y la UNCPBA.-

BIBLIOGRAFÍA

[DeGi91] De Giusti A., "Descripción y Validación de Hardware", EBAI, 1991.

[Fort85] Fortier P., "Design and analysis of distributed real-time systems", McGraw-Hill, 1985.

[Grub95] Grübel, G. "The ANDECS CASE Framework". IEEE Control Systems, Piscataway, abril 1995, págs. 8 a 13.

[Hami93] Hamidzadeh Babak y Shekhar Shashi. "Specification and Analysis of Real-Time Problem Solvers". IEEE Transaction of Software Engineering, volumen 19, número 8, Agosto 1993, págs. 788 a 803.

[Hatl88] Hatley, Derek y Pirbhai, Imtiaz. "Strategies for Real-Time System Specification". New York, Dorset House Publishing, 1988, 384 págs.

[Kell92] Keller, Marilyn y Shumate Ken. "Software Specification and Design, A Disciplined Approach for Real-Time Systems", New York, John Wiley & Sons, Inc., 1992, 405 págs.

[Krug94] Kruglinski, David. "Inside Visual C++,version 1.5". 2da. ed., Redmond, Washington, Microsoft Press, 1994, 732 págs.

[Kuma94] Kumar, P. y Kumar, Sunil. "Performance Bounds for Queueing Networks and Scheduling Policies". IEEE Transaction On Automatic Control, volumen 39, número 8, Piscataway, agosto 1994, págs. 1600 a 1611.

[Laws92] Lawson H., "Parallel processing in industrial real time applications", Prentice Hall 1992.

[Maga86] Magalhaes, "Software para tempo real", EBAI, 1986.

[Mari95] Marinangelli M., Spognardi J, "Ambiente para la especificacion de sistemas de tiempo real", Informe UNICEN, 1995.

[McCl92] McClure C., "CASE, la automatización del software", Ra-Ma, 1992.

[Micr94] Microsoft Corporation. "Class Library Reference, for The Microsoft Foundation Class Library, Microsoft Visual C++". EE.UU, Microsoft Corporation, 1175 págs.

[Niel90] Nielsen K., "Ada in Distributed Real-Time Systems", McGraw-Hill, 1990.

[Rama89] Ramamritham K., Stankovic J. A., Zhao W., "Distributed Scheduling of Tasks with Deadlines and Resource Requirements", IEEE Transactions on Computers, Vol. 38, No. 8, pp. 1110-1123.

[Ward85] Ward P. y Mellor, S. "Structured Development for Real-Time Systems". Volúmenes 1-3. New York, Yourdon Press, 1985.