PROGRAMA EN MÉTODOS PARA EL DESARROLLO DE SOFTWARE FIABLE, DE ALTA CALIDAD Y SEGURO DE LA COMUNIDAD DE MADRID

PROMESAS

PROMESAS

http://www.promesas-cm.org/

RESUMEN Y OBJETIVOS

El principal objetivo del Programa es contribuir de forma decidida al desarrollo de productos software de alta calidad, seguros y fiables.

Un control global de calidad sólo puede ser alcanzado mediante un proceso riguroso que cubra todas las fases de desarrollo de software. Esto supone la integración de diferentes mecanismos que cubran técnicas relacionadas con entornos de desarrollo de software, lenguajes de especificación, generación de código a partir de especificaciones, interpretación abstracta, lenguajes declarativos, seguridad por código con demostración ('Proof-Carrying Code'), programación semántica, métodos formales, etc.

El programa incluye tanto actividades de investigación en las temáticas citadas como acciones más globales relacionadas con la formación, contratación, transferencia de tecnología y lanzamiento de proyectos de colaboración a nivel nacional e internacional.

SOCIOS

Coordinador

The Computational logic, Languages, Implementation, and Parallelism Laboratory (CLIP)
Universidad Politécnica de Madrid
Responsable: MANUEL HERMENEGILDO SALINAS

Socios

Desarrollo de Software Fiable y de Alta Calidad a partir de Tecnología Declarativa / Programming Languages and Reliable Software (BABEL)
Universidad Politécnica de Madrid
Responsable: JUAN JOSÉ MORENO NAVARRO

Diseño y análisis formal de sistemas de software (FADOSS)
Universidad Complutense de Madrid
Responsable: NARCISO MARTÍ OLIET

Grupo de Programación Declarativa (GPD)
Universidad Complutense de Madrid
Responsable: FRANCISCO JAVIER LÓPEZ FRAGUAS

Figura 1. La tecnología de compilación y verificación desarrollada permite programar aplicaciones empotradas en lenguajes de alto nivel y garantizar no sólo su corrección sino también que el gasto de recursos (memoria, tiempo, energía) se ajusta a las limitaciones de la plataforma. Esto se ha aplicado en el sistema auditivo humano para un escenario de realidad virtual, en cuyo muy reducido procesador corren dichas aplicaciones.

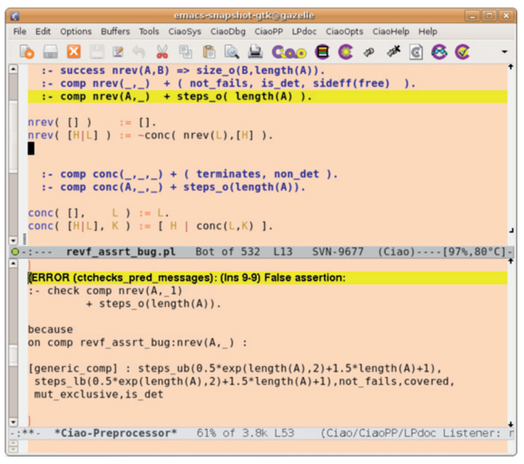

Figura 2. Las herramientas desarrolladas ilustran que es posible detectar errores de consumo de recursos no triviales durante el proceso de compilación. Esto supone identificar dichos errores mucho antes de las pruebas (testing) lo que puede suponer un ahorro significativo en el coste de desarrollo de un sistema empotrado.

Figura 3. Las aplicaciones para los ferrocarriles exigen un nivel de integridad en la seguridad del software muy alto, dado que los daños que podría causar una avería pueden ser muy graves.

Las técnicas de análisis estático mediante interpretación abstracta desarrolladas pueden ser de mucha utilidad en este campo de aplicación. De hecho, el standard europeo EN50128 [CENELEC 2000] no admite que ninguna parte del software relacionado con la seguridad pueda omitir el análisis estático ni durante la verificación y comprobación, ni durante la fase de evaluación del software sin una justificación detallada. Dependiendo del nivel de integridad en la seguridad del software, pueden ser obligatorias la especificación formal y la verificación.

Se ha evaluado hasta qué punto las herramientas y las técnicas desarrolladas pueden ayudar a cumplir estos requisitos extremos en un tiempo y con unos esfuerzos mínimos y cómo podrían usarse para resolver tareas de prueba simples de manera automática.

Figura 4. Las herramientas de verificación de programas basadas en técnicas formales de análisis mediante interpretación abstracta se utilizan para garantizar el buen funcionamiento del software empotrado crítico en aviones. Esto se consigue garantizando que dicho software cumple propiedades tales como la ausencia de errores en tiempo de ejecución, el consumo adecuado de recursos (tiempo de procesador y memoria), el cumplimiento de tiempos máximos de respuesta, etc, todas ellas propiedades cuyo no cumplimiento puede tener consecuencias desastrosas, sobre todo en daños humanos.

Figura 5. Las aplicaciones en el campo espacial cubren una amplia gama de características, en términos de complejidad o de arquitecturas software, pero las características comunes son el hardware de bajas prestaciones con grandes restricciones sobre el software (CPU, memoria, buses, ...) y un alto nivel de criticalidad en términos de mantenibilidad, fiabilidad (hasta quince años en vuelo) y disponibilidad (misiones y modos críticos). Los métodos y herramientas de análisis estático desarrollados pueden aplicarse tanto para aumentar la calidad final del software como para disminuir los costes de verificación.

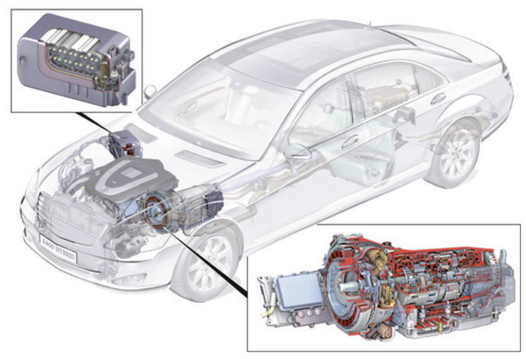

Figura 6. El software empotrado en el campo de la automoción cubre una amplia gama de aplicaciones, con diferentes niveles de complejidad y seguridad, en diferentes arquitecturas software admitidas por una gran diversidad de plataformas de hardware. Además, debido a que se trata de un negocio muy competitivo y con gran volumen de producción, el software tiene que optimizarse en términos de coste, manteniendo un alto nivel de calidad y de fiabilidad. A ello contribuyen las metodologías y herramientas para el desarrollo y verificación de software, las cuales podrían integrarse en los futuros procesos standard de la automoción que se están desarrollando (AUTOSAR, futuro ISO26262).

LÍNEAS DE TRABAJO DESTACADAS

1. Desarrollo de entornos de programación y lenguajes declarativos y de especificación con rigurosos fundamentos matemáticos que puedan modelar de forma eficiente problema reales de software de la industria. Se cubren aspectos de diseño, definición e implementación, que en la mayoría de los casos serán concebidos como extensiones o mejoras de lenguajes y sistemas ya desarrollados por los grupos del programa.
2. Implementación y fundamentos de herramientas para el desarrollo, validación y verificación de estos lenguajes, fomentando sus usos efectivos en entornos empresariales.
3. Metodologías para el desarrollo de software basadas en estos lenguajes y herramientas que garanticen calidad y seguridad de código.

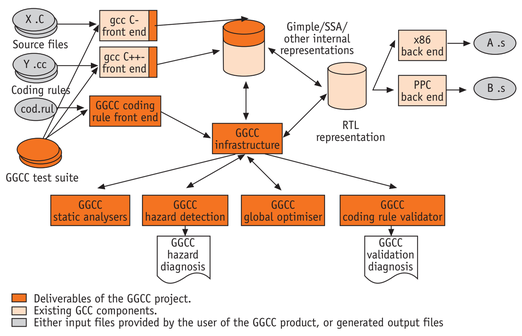

Figura 7. Diagrama del proyecto GGCC.

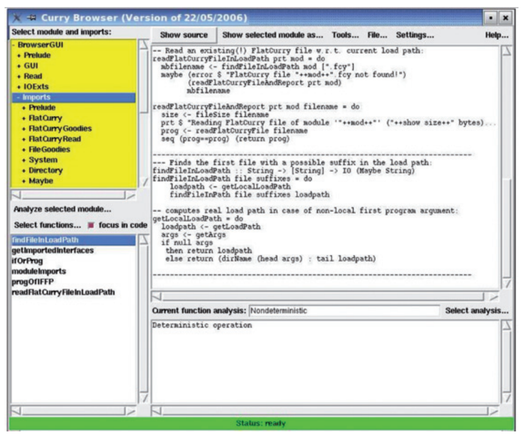

Figura 8. Curry browser.

INFRAESTRUCTURA CIENTÍFICO-TECNOLÓGICA

Además de las herramientas software mencionadas anteriormente, que constituyen la base de la investigación y oferta tecnológica del programa, se dispone de varios laboratorios y una infraestructura de servidores que facilita la investigación y el intercambio de resultados entre grupos y que incluye un multiprocesador Sunfire T2000 con 8 procesadores y capaz de gestionar hasta 32 hilos (threads) simultáneos. Esta infraestructura incluye un repositorio común de documentos y sistemas software de apoyo para el desarrollo colaborativo (control de accesos, control de cambios, control de versiones, etc.). Se está utilizando regularmente como herramienta de trabajo colaborativo tanto sobre sistemas como sobre documentos. También se ha implantado una infraestructura de videoconferencia basada en estándares abiertos que se utiliza tanto para la comunicación remota como para la grabación de conferencias.

PROMESAS

PUBLICACIONES Y PATENTES RELEVANTES

Publicaciones más relevantes

- E. Albert, M. Gómez-Zamalloa, L. Hubert, and G. Puebla. Verification of Java Bytecode using Analysis and Transformation of Logic Programs. In Int'l. Symp. on Practical Aspects of Declarative Languages, volume 4354 of LNCS, pages 124-139. Springer-Verlag, January 2007.
- R. Caballero, M. R. Artalejo, and R. del Vado Vírseda. Declarative diagnosis of wrong answers in constraint functional-logic programming. In S. Etalle and M. Truszczynski, editors, 22nd Int'l. Conf. on Logic Programming (ICLP 2006), volume 4079 of LNCS, pages 421-422. Springer-Verlag, 2006.
- M. Carro, J. Morales, H. Muller, G. Puebla, and M. Hermenegildo. High-Level Languages for Small Devices: A Case Study. In K. Flautner and T. Kim, editors, Compilers, Architecture, and Synthesis for Embedded Systems, pages 271-281. ACM Press / Sheridan, October 2006.
- M. Clavel, F. Durán, J. Hendrix, S. Lucas, J. Meseguer, and P. Ölveczky. The Maude formal tool environment. In Proceedings of CALCO 2007, LNCS. Springer-Verlag, 2007.
- M. Fernández and M. J. Gabbay. Nominal rewriting. Information and Computation, 2007.
- A. Fernández, T. Hortalá-González, F. Sáenz-Pérez, and R. del Vado-Vírseda. Constraint Functional Logic Programming over Finite Domains. Theory and Practice of Logic Programming, 2007.
- D. de Frutos Escrig and C. Rodríguez. Process equivalences as global bisimulations. Journal of Universal Computer Science, 12(11):1521-1550, 2006.
- A. Herranz and J. J. Moreno Navarro. Modeling and reasoning about design patterns in Slam-Sl. Design Pattern Formalization Techniques, 2007.
- M. Hidalgo-Herrero, F. Rubio, and Y. Ortega-Mallén. Analyzing the influence of mixed evaluation on the performance of eden skeletons. Parallel Computing, 32:523-538, 2006.
- F. López-Fraguas, M. Rodríguez-Artalejo, and R. del Vado Vírseda. A new generic scheme for functional logic programming with constraints. Higher Order and Symbolic Computation, 20(1/2), 2007.
- J. Mariño, A. Herranz, and J. J. Moreno-Navarro. Demand analysis with partial predicates. Theory and Practice of Logic Programming, 1&2:153-182, 2007.
- E. Mera, P. López-García, G. Puebla, M. Carro, and M. Hermenegildo. Combining Static Analysis and Profiling for Estimating Execution Times. In Ninth International Symposium on Practical Aspects of Declarative Languages, volume 4354 of LNCS, pages 140-154. Springer-Verlag, January 2007.
- J. Meseguer, M. Palomino, and N. Martí-Oliet. Equational abstractions. Theoretical Computer Science, 2007.
- J. J. Moreno-Navarro, N. Maya-Fernández, and A. Herranz. Towards semantically defined web services. In 11th International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems (IPMU 2006), 2006.

Otros resultados

Durante el primer periodo del programa, los diferentes grupos involucrados han desarrollado o mejorado hasta 32 sistemas o librerías software distintas, todas ellas distribuidas como software abierto.

Incorporación de personal investigador

A través del programa, y durante los primeros 18 meses, se han incorporado al sistema I+D de la Comunidad de Madrid un total de 24 investigadores o gestores. Esto incluye 6 doctores (2 a cargo del Programa Ramon y Cajal, 1 a cargo del Programa Juan de la Cierva, 2 a cargo del programa en sí, y uno más a través de otros programas), 18 investigadores predoctorales, un técnico de gestión y un técnico de apoyo a la investigación.