

Interpretación Abstracta de Múltiples Vistas en el Diseño de Sistemas*

Cuauhtémoc Lemus Olalde[†]
Dr. Boumediene Belkhouche[‡]
Universidad de Tulane
Nueva Orleans, Louisiana. E.U.A.

Marzo, 1996

Resumen

*El proceso de diseño de sistemas permite a los diseñadores formular un conjunto de especificaciones que describen la forma en la que se implementará un sistema computacional. Estas especificaciones describen la estructura de la solución y la manera en que los componentes se conectarán entre ellos, actuando como un conjunto de **planos** en términos de estructura, comportamiento, datos, módulos y funciones, que una vez consolidados constituyen los planos del diseño.*

Durante la formulación del diseño, éste pasa a través de varios procesos de evaluación y verificación, donde la identificación de errores es el principal objetivo. Se propone considerar los diferentes planos o "vistas" del sistema como una herramienta para una mejor identificación de defectos en el diseño.

*El propósito de esta investigación es el de proveer las bases formales de **Múltiples Vistas (MV)** a través de **Interpretación Abstracta (IntAbs)**. Este enfoque facilitará*

el análisis semántico de las vistas, proporcionando las apropiadas transformaciones entre las definiciones concretas y abstractas de los diseños.

1 Introducción

La fase de diseño del ciclo de desarrollo de sistemas implica un alto grado de creatividad. Durante ésta fase, el diseñador crea, documenta y verifica la arquitectura del sistema, componentes, módulos e interfaces. El diseñador hace todo lo posible por entender el problema (sistema), produciendo un conjunto de planos que corresponden a los diferentes atributos del sistema. Este principio, de usar múltiples planos o vistas para tener un mejor entendimiento de un problema ha sido empleado exitosamente en otras áreas ingenieriles. Por ejemplo, un Arquitecto o Ingeniero Civil, aplicando varias técnicas, reglas y métodos ingenieriles, obtiene diferentes planos, tales como, distribución de cargas, distribución eléctrica, distribución de ductos de aire, etc., y cada uno mostrando diferentes atributos del proyecto. En contraste, un diseñador construye varias vistas o modelos del sistema aplicando uno o varios métodos de

*Tesis Doctoral. Derechos Reservados, Mayo 1996.

[†]Candidato a Doctor en Ciencias Computacionales, becario del Convenio SEP-Tulane.

[‡]Asesor de Tesis.

diseño.

A pesar de la existencia de varios métodos de diseño en el desarrollo de sistemas, el proceso de diseño todavía tiene algunas áreas que necesitan mejorarse [1, 2], en particular, el manejo de múltiples vistas [3, 4, 5, 6, 7], integración de múltiples vistas [8, 9] y el Análisis de Múltiples Vistas (AMV).

2 Aspectos y Consideraciones Importantes

El uso de MV en el diseño de sistemas no es una técnica nueva. Representaciones canónicas han sido usadas para proporcionar soporte [10, 9]. La técnica de Análisis y Diseño Estructurado utiliza MV para obtener un modelo más cercano de la realidad [11].

MV han sido empleadas en la recolección de requerimientos [13], modelaje de dominios [14], definición de requerimientos y especificaciones [3, 4, 5, 6], sistemas de desarrollo de programas [15], y reuso de diseños de sistemas [10]. A continuación se listan los aspectos importantes que se deben de tomar en cuenta en la utilización de MV en el diseño de sistemas:

- Un método de diseño de sistemas aplica un cierto punto de vista para la solución de un problema [16].
- La mayoría de las metodologías de diseño se basan en el uso de notaciones gráficas [17, 18].
- Una notación hace una abstracción del sistema, enfocándose en ciertos atributos del sistema [2].

Un aspecto importante de la etapa de AMV, son las condiciones necesarias en las cuales las vistas serán analizadas. El ambiente adecuado será proporcionado por IntAbs. Por ello la siguiente lista presenta una perspectiva de los aspectos a considerar:

- AMV requiere de una representación formal de las vistas [14].
- AMV requiere de un análisis sistemático de las vistas formalizadas [13].
- IntAbs ha sido aplicada en el área de lenguajes de programación [19, 20, 21, 22].

Consecuentemente, las siguientes consideraciones tienen efecto con el fin de apoyar el uso de MV y AMV en la fase de diseño de sistemas:

1. **Definición de una representación formal de notaciones gráficas.** Esta representación formal garantizará la correspondencia uno-a-uno entre notaciones gráficas..
2. **Falta de un procedimiento sistemático para realizar AMV.** Una vez que las vistas han sido formalizadas, un análisis sistemático de las vistas tiene que ser realizado.
3. **Falta de AMV en la fase de diseño.** Hacer uso de AMV en la detección de errores de diseño.
4. **Falta de soporte en la aplicación de Interpretación Abstracta.** No existen las condiciones necesarias para realizar Interpretación Abstracta de múltiples notaciones de diseño.

La figura 1 muestra estas consideraciones, así como los objetivos de la investigación que los satisfacen.

3 Estrategia de Trabajo

El propósito de la investigación es el Análisis de Múltiples Vistas, para su posterior uso en la etapa de diseño. El AMV, compara dos diferentes vistas e identifica sus diferencias. En específico, dos vistas son comparadas y

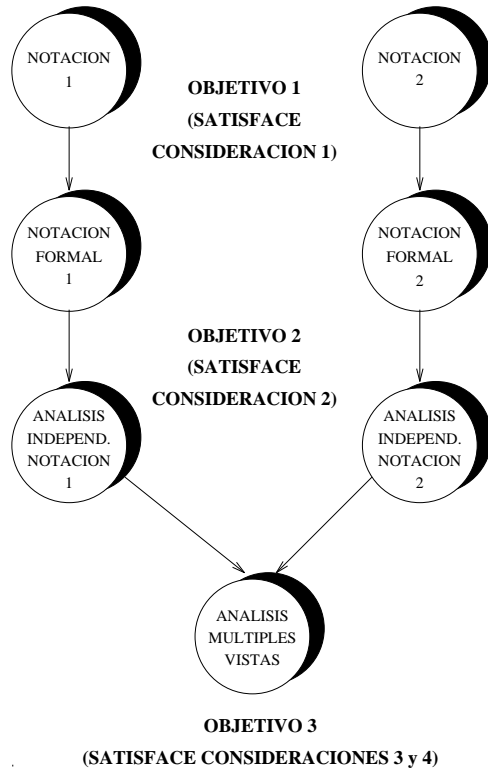


Figure 1: Objetivos Principales de la Investigación que satisfacen las Consideraciones de la sección 2.

analizadas con el propósito de obtener información complementaria o contradictoria de ellas. Las vistas utilizan dos notaciones diferentes, en particular se seleccionó una vista funcional provista por el Diagrama de Flujo de Datos y una vista funcional/estructural capturada por la notación de Diagrama Estructurado.

La estrategia de investigación se muestra gráficamente en la figura 1, y una breve descripción de los objetivos está a continuación:

1. Proveer especificaciones formales a dos notaciones de diseño.
 - (a) Seleccionar el formalismo que servirá para representar las notaciones de diseño.
 - (b) Seleccionar dos notaciones de diseño.
 - (c) Formalizar cada notación.
2. Proveer las bases para el análisis independiente y simultáneo de vistas.
 - (a) Realizar un análisis sintáctico independiente para cada diseño formal.
 - (b) Realizar un análisis simultáneo de los diseños formales.
3. Proveer los fundamentos matemáticos del AMV.
 - (a) Extender el enfoque IntAbs para que contemple múltiples notaciones de diseño.
 - (b) Realizar una interpretación abstracta independiente de cada diseño formal.
 - (c) Definir las transformaciones entre las definiciones formales de cada diseño.

4 Conclusiones

La contribución mas importante de la investigación es el desarrollo de las bases para el uso

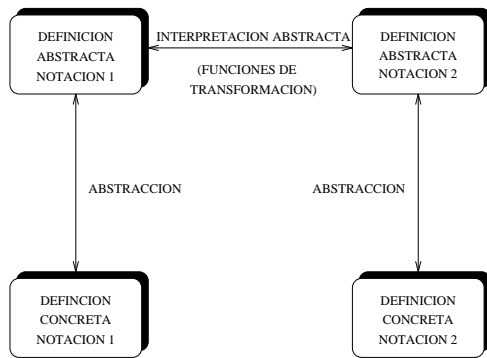


Figure 2: Análisis de Múltiples Vistas a través de Interpretación Abstracta.

de Múltiples Vistas en la fase de diseño de sistemas. Para lograr esta meta, se ha tomado como fundamento matemático el enfoque de Interpretación Abstracta, haciendo su aplicación en este campo específico de diseño de sistemas el primero en su haber. La figura 2 esquematiza la contribución.

Además, una gran parte del trabajo se dedicó en la definición del formalismo, en el cual las vistas son formalizadas y analizadas. También, se desarrolló un proceso sistemático y formal de proveer los fundamentos del AMV en la fase de diseño. Y por último, se confirmó lo útil del enfoque de IntAbs en el manejo de múltiples notaciones de diseño.

Referencias

- [1] Sievert, G. E. and Mizell, T. A., "Specification-Based Software Engineering with TAGS," *IEEE Computer*, pp. 56–65, April 1985.
- [2] Budgen, D., *Software Design*. International Computer Science Series, Addison-Wesley Publishing Co., 1994.
- [3] Finkelstein, A., "A Framework for Expressing the Relations Between Multiple Views in Requirements Specification," *IEEE Transactions on Software Engineering*, September 1994.
- [4] Finkelstein, A., Gabbay, D., and Hunter, A., "Inconsistency Handling in Multi-Perspective Specifications," *Proceedings of 4th European Software Engineering Conference*, pp. 84–99, September 1993.
- [5] Finkelstein, A., "Viewpoints: A Vehicle for Method and Tool Integration," *IEEE Proceedings of International Workshop on CASE*, pp. 50–60, July 6-10 1992.
- [6] Finkelstein, A., "Co-ordinating Distributed Viewpoints: The Anatomy of a Consistency Check," *Concurrent Engineering: research and Applications*, CERA Institute, 1994.
- [7] Nuseibeh, B., Finkelstein, A., and Kramer, J., "Method Engineering for Multi-Perspective Software Development," tech. rep., Department of Computing, Imperial College of Science, Technology and Medicine. London, 1994.
- [8] Finkelstein, A., Kramer, J., Nuseibeh, B., Finkelstein, L., and Goedicke, M., "Viewpoints: A Framework for Integrating Multiple Perspectives in System Development," *International Journal of Software Engineering and Knowledge Engineering*, vol. 2, pp. 31–58, March 1992.
- [9] Meyers, S. D., *Representing Software Systems in Multiple-View Development Environments*. PhD dissertation, Department of Computer Science at Brown University, May 1993. CS-93-18.
- [10] Lowry, M. R. and McCartney, R. D., *Chapter 5 in Automating Software De-*

- sign.* Cambridge, England: MIT Press, 1991.
- [11] Marca, D., *SADT: Structured Analysis and Design Technique*. New York: McGraw Hill, 1988.
- [12] Budgen, D., "Augmenting the Design Process: Transformations from Abstract Design Representations," *Advanced Information Systems Engineering 4th International Conference CAiSE Manchester, UK*, May 12-15 1992.
- [13] do Prado Leite, J. C. S. and Freeman, P. A., "Requirements Validation Through Viewpoint Resolution," *IEEE Transactions on Software Engineering*, vol. 17, pp. 1253-1269, December 1991.
- [14] Easterbrook, S., "Domain Modeling with Hierarchies of Alternative Viewpoints," *IEEE International Symposium on Requirements Engineering*, pp. 65-72, January 4-6 1993.
- [15] Reiss, S. P., "PECAN: Program Development Systems that Support Multiple Views," *IEEE Transactions on Software Engineering*, vol. 11, pp. 276-285, March 1985.
- [16] Bergland, G. D., "A Guided Tour of Program Design Methodologies," *IEEE Computer*, pp. 13-37, October 1981.
- [17] Raeder, G., "A Survey of Current Graphical Programming Techniques," *IEEE Computer*, vol. 19, pp. 11-25, August 1985.
- [18] Tripp, L. L., "A Survey of Graphical Notations for Program Design-An Update," *ACM SIGSOFT Software Engineering Notes*, vol. 13, pp. 39-44, October 1988.
- [19] Tennent, R. D., "The Denotational Semantics of Programming Languages," *Communications of the ACM*, pp. 437-453, August 1976.
- [20] Barbuti, R., Giacobazzi, R., and Levi, G., "A General Framework for Semantics-Based Bottom-Up Abstract Interpretation of Logic Programs," *ACM Transactions on Programming Languages and Systems*, vol. 15, pp. 133-181, January 1993.
- [21] Charlier, B. L. and Hentenryck, P. V., "Experimental Evaluation of a Generic Abstract Interpretation Algorithm for Prolog," *ACM Transactions on Programming Languages and Systems*, vol. 16, pp. 35-101, January 1994.
- [22] Englebort, V., Charlier, B. L., Roland, D., and Hentenryck, P. V., "Generic Abstract Interpretation Algorithms for Prolog: Two Optimization Techniques and their Experimental Evaluation," *Software Practice and Experience*, vol. 23, pp. 419-459, April 1993.
- [23] Tse, T., "The Application of Prolog to Structured Design," *Software Practice and Experience*, vol. 24, pp. 659-676, July 1994.
- [24] Moriconi, M. and Hare, D. F., "Visualizing Program Designs Through PegaSys," *IEEE Computer*, pp. 72-85, August 1985.
- [25] Hekmatpour, S. and Woodman, M., "Formal Specification of Graphical Notations and Graphical Software tools," *Proceedings of 1st European Software Engineering Conference*, pp. 297-305, September 1987.
- [26] Hoare, C., "An Overview of Some Formal Methods for Program Design," *IEEE Computer*, pp. 85-91, September 1986.

- [27] Tse, T. and L.Pong, "Towards a Formal Foundation for DeMarco Data Flow Diagrams," *Computer Journal*, vol. 32, pp. 1–12, 1989.
- [28] Tao, Y. and Kung, C., "Formal Definition and Verification of Data Flow Diagrams," *Journal of Systems Software*, vol. 16, pp. 29–36, 1991.