Arquitecturas de Software para el diseño de Robots Móviles Autónomos

Erika Michalczewsky Pablo R. Fillottrani

{emichal,prf}@cs.uns.edu.ar

Departamento Ciencias de la Computación
Universidad Nacional del Sur

1. Introducción

El área de las arquitecturas de software dentro de la ingeniería de software es una disciplina que evolucionó naturalmente a partir del diseño de abstracciones. Es decir, surgió como consecuencia de la búsqueda, por parte de los diseñadores de sistemas, de mejores formas para entender el software y de nuevos métodos que permitan la construcción de sistemas más grandes y más complejos. Ofrece la base para el inicio del ciclo de vida del software: a partir de la arquitectura se determina si el sistema tiene la robustez para soportar la carga actual y futura del mismo, la flexibilidad para adaptarse a nuevas tecnologías y requerimientos y la capacidad de afrontar, sin riesgos, un desarrollo con el presupuesto disponible y en el tiempo planificado [BAS98][SHA96].

La arquitectura de software es en gran parte el estudio de la estructura del software, y como tal puede aplicarse en el desarrollo de sistemas pertenecientes a diversos dominios. En este caso se consideraron los sistemas para el control de robots móviles autónomos, los cuales presentan estructuras particulares. El objetivo de este trabajo es evaluar las alternativas de diseño para arquitecturas de robots móviles autónomos, aspectos que permitieron la propuesta de una arquitectura de control para robots y un ambiente de trabajo para la evaluación de algoritmos de planificación y navegación utilizando esta arquitectura. Estos sistemas se encuentran en la etapa de implementación.

2. Requerimientos de las Arquitecturas para Robots Móviles Autónomos

Cuando se contruyen sistemas para el control de robots debe tenerse en cuenta la facilidad con que los mismos pueden ser desarrollados, testeados, depurados y entendidos. Con el constante progreso que existe en el área de la robótica, no es razonable suponer que todas las componentes del sistema se van a construir desde cero. Por el contrario, la amplia variedad de módulos desarrollados independientemente pueden ser integrados dentro de su estructura. Para ello, los sistemas deben imponer restricciones mínimas en la naturaleza de los datos, las representaciones y los algoritmos. Una arquitectura, para ser útil, debe poder ser utilizada por diversos programadores en sitios diferentes y con métodos distintos. Además, debe ser factible la incorporación de subsistemas proporcionando así un desarrollo evolutivo sin afectar la funcionalidad ya lograda.

La arquitectura debe ofrecer los medios por los cuales el sistema pueda lograr sus metas eficientemente, satisfacer restricciones de tiempo real, promover la tolerancia a las fallas y proporcionar seguridad para vehículo y su entorno. Debe ofrecer una distribución apropiada para los procesos de sensado y razonamiento de manera que éstos se realicen en tiempos razonables para conseguir los objetivos propuestos. En general, como los sensores utilizados operarán con una frecuencia diferente, así como también los procedimientos que procesan sus datos, se debe permitir la operación asincrónica para maximizar el resultado y la respuesta del sistema.

3. Alternativas de Diseño

Para cumplir con los requerimientos mencionados, el diseño de arquitecturas para el control de robots móviles autónomos tiene en cuenta cuatro características principales sobre el sistema que se pretende desarrollar: el tipo de razonamiento (reactivo o deliberativo), la naturaleza del procesamiento

(centralizado o distribuido), la manera de combinar la información de entrada (fusión de sensores o arbitraje de comandos) y la estructura de control (*top-down* o *bottom-up*). Estas características conforman las decisiones más importantes en un sistema para el control de robots. Existen arquitecturas que resuelven tomar una posición extrema en estas cuestiones, pero han demostrado ser muy limitadas. El pragmatismo indica que existe una extensión intermedia muy amplia que debe ser explorada, es decir, que existen equilibrios razonables que deben considerarse para el diseño de estos sistemas. A continuación se analizan cada una de estas alternativas.

3.1. Razonamiento

El objetivo es lograr un balance entre la necesidad de obtener un conjunto de acciones óptimas y planificadas cuidadosamente, para conseguir las metas, y la necesidad de tomar decisiones rápidas, que satisfagan reacciones a un entorno dinámico e incierto. El razonamiento deliberativo tiene la capacidad de lograr metas de alto nivel y evitar errores que podrían conducir a ineficiencias, mientras que las respuestas reactivas permiten al robot evitar perjudicarse o perjudicar el entorno. La ausencia de cualquiera de los dos tipos de razonamiento puede llevar a que el robot no sea capaz de satisfacer sus objetivos.

También es importante el tipo de representación utilizada o soportada por la arquitectura. Las estructuras definidas juegan un rol muy importante en determinar cuán apropiada es una arquitectura particular para diferentes clases de dominios y tareas, aunque no sea en la capacidad absoluta, al menos sí en la facilidad de uso. No existe una única respuesta; la conveniencia de una abstracción particular depende de la arquitectura en la cual es implementada, el nivel en el cual se describe mejor la tarea, las demandas de tiempo real de la misma, y las capacidades y preferencias del diseñador del sistema. Muchas arquitecturas obligan a los sistemas que las implementan a adoptar una posición u otra en el nivel de planificación y en el nivel de abstracción de la representación utilizada. Idealmente, una arquitectura debería posibilitar la inclusión, de manera eficiente y efectiva, de cualquier tipo de sistema de razonamiento del espectro deliberativo-reactivo, según sea apropiado para el dominio. Se evitará de esta forma forzar al programador a engañar a los diseñadores de la arquitectura para lograr sus objetivos.

3.2. Procesamiento

Los primeros sistemas de robots móviles operaban recolectando todos los datos disponibles de los sensores, creaban un modelo completo de su entorno estático, planificaban una serie óptima de acciones dentro del contexto de ese modelo, y ejecutaban ese plan . El robot luego se paraba para obtener más información y el proceso se repetía [MOR90][NIL80].

Las arquitecturas jerárquicas son un tipo de sistema distribuido en el cual los módulos se encuentran organizados en niveles de control múltiples que operan con granularidad, niveles de abstracción y escalas de tiempo diferentes, ofreciendo así un equilibrio entre la correctitud y completitud a largo término y la sobrevivencia y relevancia a corto tiempo. Las arquitecturas jerárquicas tradicionales usan una descomposición homogénea, en donde cada nivel es construido a partir de los mismos módulos, aunque con distintos niveles de razonamiento [ALB87]. Otro tipo de arquitecturas jerárquicas descomponen la tarea en sí misma de una manera recursiva. Se crea un plan para lograr una tarea del nivel más alto y se expanden subtareas recursivamente para lograr las submetas de ese plan hasta que las acciones más primitivas obtienen el resultado deseado [FIR94] .

Las arquitecturas basadas en esquemas o comportamientos constituyen una clase radicalmente diferente de sistemas para el control de robots. En lugar de diseñar el sistema con módulos funcionales, como por ejemplo módulos de percepción y planificación, el sistema está compuesto por comportamientos individuales, los cuales realizan una tarea específica y limitada [BRO86].

3.3. Combinación de la Información

Las arquitecturas también pueden ser caracterizadas por la forma en la cual combinan la información y los objetivos. Las arquitecturas que realizan fusión de sensores construyen un modelo del mundo único basándose en todos los datos disponibles desde los sensores [MOR85]. Esta

propuesta tiene la ventaja de ser capaz de combinar la información para sobrellevar las ambigüedades y el ruido, inherentes en el proceso de sensado, pero tiene la desventaja de crear un cuello de botella computacionalmente caro con los sensores, dado que todos los datos de los mismos deben ser recolectados e integrados antes de poder actuar. Además, un modelo único y monolítico del mundo es difícil de desarrollar, mantener y extender.

En las arquitecturas basadas en comportamientos el procesamiento perceptual se encuentra distribuido en múltiples módulos independientes. Cada comportamiento requiere sólo un conocimiento fragmentado del mundo y recibe exclusivamente los datos de los sensores que son directamente relevantes en la toma de decisiones particulares. Así, cada comportamiento se encuentra libre de utilizar la representación que considere más apropiada para el propósito.

3.4. Estructura de Control

En arquitecturas estrictamente jerárquicas, se espera que el diseño y operación del sistema para el control de robots sea *top-down*; cada nivel controla el nivel por debajo y asume que su comando se ejecutará como se previó.

En las arquitecturas basadas en comportamientos, los módulos se encuentran activos todo el tiempo, en el sentido que siempre procesan los datos disponibles. La estructura de control es *bottom-up* porque cada comportamiento determina por sí mismo si es relevante para la situación actual, basándose en los datos que recibe.

3.5. Otros aspectos

Además de los aspectos mencionados, existen otros que no son de menor importancia y deben tenerse en cuenta en el diseño de arquitecturas de software para robots.

1. **Selección de acciones**. En una arquitectura distribuida es necesario determinar en cada momento los comportamientos que controlan al robot y seleccionar entre las acciones propuestas por los mismos. Fusionando los comandos apropiadamente, o a través del arbitraje, un sistema para el control de robots puede responder a su entorno sin sufrir de los problemas inherentes a la fusión de sensores. En su lugar, el sistema combina los comandos para decidir un curso de acción apropiado; las formas más comunes son el arbitraje basado en prioridades y los campos potenciales.
2. **Entorno**. Existen diferentes tipos de entornos que afectan el diseño de estos sistemas y cada tipo requiere de un programa distinto que lo trate apropiadamente.
3. **Restricciones físicas**. Cuando se trabaja con sistemas físicos, como los robots móviles, también es importante considerar aspectos del control como la estabilidad, y la limitación y restricciones de la planta física. Por ejemplo, las restricciones no-holonómicas de un vehículo con ruedas; las capacidades finitas de un actuador; las demoras inherentes de cualquier sistema que surgen de las latencias en la adquisición de datos, en el procesamiento de los datos, en la comunicación entre módulos y en las respuestas del actuador; y el movimiento continuo del vehículo.

Reconocimientos

El estudio de las diferentes alternativas de diseño de sistemas para el control de robots móviles autónomos se enmarcó en el plan de estudio propuesto para la tesis de Magister en Ciencias de la Computación, en el Departamento de Ciencias de la Computación de la Universidad Nacional del Sur. Este trabajo fue realizado con el soporte de la Secretaría de Ciencia y Tecnología, a través de una Beca de Iniciaciación a la Investigación para Egresados de la UNS.

Bibliografía

[ALB87] J. Albus and H. McCain and R. Lumia. NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM), NBS, Technical Note 1235, 1987.

[BAS98] L. Bass and P. Clements and R. Kazman. Software Architecture in Practice, Addison-Wesley, 1998, Reading, Massachusetts.

[BRO86] R.A. Brooks. A Robust Layered Control System For A Mobile Robot, IEEE Journal of Robotics and Automation, 2(1), pages 14+, 1986.

[FIR94] J.R. Firby. Task Networks for Controlling Continuous Processes, Proceedings of the Second International Conference on AI Planning Systems, 1994, Chicago.

[MOR85] H.P. Moravec and A.E. Elfes. High Resolution Maps From Wide Angle Sonar, Proceedings of the IEEE International Conference on Robotics and Automation, IEEE Computer Society Press, pages = 116-121, 1985, Los Alamitos, CA.

[MOR90] H.P. Moravec. The Stanford Cart and the CMU Rover, Autonomouos Robot Vehicles, I. Cox and G. Wilfong editors, Springer-Verlag, 1990.

[NIL80] N.J. Nilsson. Principles of Artificial Intelligence, Tioga Press, 1980, Palo Alto, CA.

[SHA96] M. Shaw and D. Garlan. Software Architecture. Perspectives on an Emerging Discipline, Prentice Hall, 1996, Upper Saddle River, New Jersey.