Journal of the Korea Academia-Industrial cooperation Society
Vol. 13, No. 1 pp. 355-363, 2012

http://dx.doi.org/10.5762/KAIS.2012.13.1.355

블랙보드 구조와 다중 에이전트 구조의 통합

장혜진[1*]
[1]상명대학교 컴퓨터소프트웨어공학과

Integration of Blackboard Architecture into Multi-Agent Architecture

Hai Jin Chang[1*]
[1]**Dept. of Computer Software Engineering, Sangmyung University**

요 약 다중 에이전트 구조와 블랙보드 구조를 통합하면 두 구조의 특징들과 장점들을 필요로 하는 응용 분야에 대응할 수 있는 가능성이 생긴다. 본 논문은 Rete 네트워크에 기반을 둔 블랙보드 이벤트 탐지 메커니즘과 블랙보드 이벤트 기반의 암시적 호출 구조 패턴을 함께 사용하여 다중 에이전트 구조에 블랙보드 구조를 통합하는 방안을 제안한다. 다중 에이전트 구조에 블랙보드 구조를 통합하기 위하여 이벤트 기반의 암시적 호출 구조 패턴을 사용하는 것은 구성 요소들 간의 결합도(coupling)의 감소와 지식 원천 에이전트들의 제어의 융통성의 증대 등의 면에서 바람직하다. 하지만 이벤트 기반의 암시적 호출 구조 패턴 자체는 그것을 사용하는 구조의 성능을 고려하고 있지 않다. 본 논문이 제안하는 통합 구조의 성능을 향상시키려면 지식 원천 에이전트들을 활성화시킬 수 있는 블랙보드 이벤트들의 발생을 신속하게 탐지할 수 있어야 한다. 본 논문이 제안하는 통합 방안은 Rete 네트워크 기반의 블랙보드 이벤트 탐지 메커니즘을 사용하여 블랙보드 이벤트 기반의 암시적 호출 구조 패턴을 이용한 통합 구조에서 지식 원천 에이전트들을 활성화시킬 수 있는 블랙보드 이벤트들이 효율적으로 탐지될 수 있도록 한다.

Abstract The Integration of multi-agent architecture and blackboard architecture may lead to a new architecture to cope with new application areas which need some good and strong points of both the architectures. This paper suggests an integrated architecture of blackboard architecture and multi-agent architecture by using event-based implicit invocation pattern and a blackboard event detection mechanism based on Rete network. From the viewpoints of weak couplings of system components and flexible control of knowledge source agents, it is desirable to use the event-based implicit invocation pattern in the integrated architecture. But the pattern itself does not concern the performance of the architecture, and it is very critical to the performance of the integrated architecture to detect efficiently the blackboard events which can activate knowledge source agents which can contribute to the problem-solving processes of the integrated architecture. The integrated architecture suggested in this paper uses a blackboard event detection mechanism based on Rete network to detect efficiently blackboard events which can activate knowledge source agents.

Key Words : Blackboard architecture, Multi-agent architecture, Event-based implicit invocation pattern, Rete network, Blackboard event detection

1. 서론

블랙보드 구조와 다중 에이전트 구조는 규모가 크고 복잡한 문제들에 대응하기 위한 공동 작업 소프트웨어(collaborating software) 구조들이다[1]. 블랙보드 구조는 정형화된 해법이 없는 문제들을 푸는 데 유용하다. 블랙보드 구조는 주어진 문제에 대한 부분적인 해답 또는 완전한 해답을 구하기 위하여 각 분야의 전문 지식을 가진

본 논문은 상명대학교 교내선발과제로 연구되었음.
*교신저자: 장혜진(hjchang@smu.ac.kr)
접수일: 11년 11월 04일 수정일 (1차 11년 12월 27일, 2차 12년 01월 03일) 게재확정일 12년 01월 05일

전문가들에 해당하는 여러 지식 원천들이 협동하기 위한 구조이다[2]. 블랙보드 구조에서 지식 원천들(knowledge sources)이 공유해야 하는 모든 데이터는 블랙보드에 저장된다. 지식 원천들은 다른 지식 원천들과 공유해야 하는 데이터를 서로 복사하여 소유하지 않고 대신 공유 저장소인 블랙보드에 넣어 공유한다. 전통적인 블랙보드 시스템들에서 블랙보드와 지식 원천들은 공유 메모리 기술이나 원격 메서드 호출과 같은 프로세스들 간의 통신(interprocess communication) 기법을 통하여 상호 협동한다.

다중 에이전트 구조는 분산 환경 속의 크고 복잡한 문제들을 풀기 위한 구조이다. 다중 에이전트 구조는 블랙보드 구조에 비하여 네트워크상에 존재하는 다수의 에이전트들의 분산성(distribution)과 자동성(autonomy)과 같은 특성을 강조한다[1]. 일반적으로 다중 에이전트 구조의 에이전트들은 블랙보드 구조의 블랙보드와 같은 중앙 집중적 데이터 보관소를 갖지 않으며, 에이전트들 간의 통신은 문자열 형태의 에이전트 메시지들을 TCP와 같은 네트워크 프로토콜을 사용하여 전송하는 방법을 사용한다.

다중 에이전트 구조에 블랙보드 구조를 통합하면 두 구조의 특징들과 장점들을 융합할 수 있는 가능성이 생긴다. 특히 다수의 에이전트들이 다량의 데이터를 서로 공유해야 하는 응용 분야에서는 다중 에이전트 구조에 블랙보드와 같은 공유 저장소를 결합하는 것이 효과적일 수 있다. 왜냐하면, 블랙보드와 같은 공유 저장소를 사용하지 않는 전통적인 다중 에이전트 구조에서 다수의 에이전트들이 다량의 데이터를 공유하려면, 에이전트들이 어떤 데이터를 어떤 다른 에이전트들과 공유할 것인가를 결정해야 하며 다른 에이전트들과 공유하고자 하는 데이터를 에이전트 메시지들에 담아 해당 에이전트들 모두에게 전송해야 하기 때문이다. 공유 저장소를 갖는 다중 에이전트 구조는 화이트보드가 있는 효율적인 회의실에 비유될 수 있다[3].

본 논문은 블랙보드 구조를 다중 에이전트 구조에 통합하기 위하여 이벤트 기반의 암시적 호출 구조 패턴을 사용한다. 만일 블랙보드 구조와 다중 에이전트 구조의 통합을 위하여 이벤트 기반의 암시적 호출 구조 패턴을 사용하지 않고 제어 에이전트가 명령을 담은 에이전트 메시지들로 지식 원천 에이전트들을 직접 평가하거나 제어하도록 한다면 제어 에이전트와 지식 원천 에이전트들 간의 결합도(coupling)가 커지며 지식 원천들의 제어의 융통성이 줄어들 것이다. 이벤트 기반의 암시적 호출 구조 패턴을 적용하면, 시스템을 구성하는 호출 모듈들과 피호출 모듈들 간의 결합도(coupling)가 감소하며, 이벤트들의 제어를 통하여 시스템 전체를 보다 유연하게 제어할 수 있는 가능성이 생기며, 모듈들의 재사용성이 높아지고, 시스템의 다른 모듈에 영향을 주지 않으면서 모듈을 수정하거나, 모듈들의 동적인 추가와 삭제를 통한 시스템의 동적 재구성이 쉬워진다고 알려져 있다[2].

본 논문이 제안하는 통합 방안은 이벤트 기반의 암시적 호출 구조 패턴 뿐 아니라 Rete 네트워크 [5]에 기반을 둔 블랙보드 이벤트 탐지 메커니즘을 함께 사용한다. 이벤트 기반의 암시적 호출 구조 패턴은 여러 가지 장점을 갖지만, 그 패턴 자체는 블랙보드의 상태 변화가 있을 때마다 지식 원천들을 활성화시킬 수 있는 블랙보드 이벤트들의 발생을 탐지하는 문제를 해결하지 않는다. 하지만, 블랙보드 구조 또는 블랙보드 구조를 통합하는 다중 에이전트 구조에서 지식 원천 에이전트들을 활성화시킬 수 있는 블랙보드 이벤트들의 발생을 탐지하는 부분의 성능은 구조 전체의 성능에 큰 영향을 미친다. 본 논문의 블랙보드 이벤트 탐지 메커니즘은 Rete 네트워크 기술을 사용하여 이벤트 기반의 암시적 호출 구조 패턴을 사용하는 통합 구조에서 지식 원천들을 활성화시킬 수 있는 블랙보드 이벤트들이 효율적으로 탐지되도록 한다.

본 논문이 제안하는 통합 구조는 특정한 응용 분야나 응용 시스템만을 위한 구조가 아니지만, 처음에 지능형 로봇 시스템[6, 16]의 개발을 위한 프레임워크 용도로 개발되었다. 본 논문이 제안하는 구조는 로봇 시스템을 구성하는 다양한 센서들에 관련된 모듈들, 모터와 같은 액추에이터(actuator)들에 관련된 모듈들, 그리고 그들을 합목적적으로 제어하기 위한 지능적 요소들을 가진 모듈들을 모두 다중 에이전트 구조의 에이전트들로 표현하고, 블랙보드 에이전트를 이용하여 에이전트들 간에 문제 풀이를 위한 데이터를 공유하도록 할 수 있다.

본 논문의 제 2장은 블랙보드 구조와 다중 에이전트 구조의 통합에 관련된 개념들과 연구들에 대한 것이다. 제 3장은 본 논문이 제안하는 통합 구조를 위한 블랙보드 데이터와 에이전트 메시지에 대한 내용을 다룬다. 제 4장은 본 논문이 제안하는 통합 다중 에이전트 구조에 대한 것이다. 제 5장은 결론이다.

2. 관련 연구

2.1 이벤트 기반의 암시적 호출 패턴

블랙보드 구조를 통합하는 다중 에이전트 시스템에 이벤트 기반의 암시적 호출 구조 패턴을 적용하면 여러 가지 이점을 취할 수 있다. 이벤트 기반의 암시적 호출 구조 패턴을 사용하는 시스템은 개념적으로 두 가지 종류의 모듈들로 구성된다. 첫 번째 종류는 다른 모듈들에게

발생된 모든 이벤트들을 통보(notification)하는 모듈들이고, 두 번째 종류는 자신이 관심을 가진 이벤트의 타입과 그 이벤트를 처리하기 위한 절차의 쌍들을 시스템에 등록(subscription)하는 모듈들이다. 이벤트 기반의 암시적 호출 구조 패턴에서는 다양한 타입의 이벤트들이 사용될 수 있다. 암시적 호출 구조 패턴을 사용하는 시스템에는 모듈이 다른 모듈의 절차를 직접 호출하는 하드 코딩 방식이 사용되지 않으며, 대신 어떤 모듈이 이벤트를 발생시키면 시스템에 의해 그 이벤트의 타입에 따라 미리 정해져 있는 절차들이 간접적으로 호출된다. 이 때 두 번째 종류의 모듈들이 등록한 이벤트의 타입과 그 이벤트에 대응하는 절차의 쌍에 대한 정보가 사용된다. 이벤트를 발생시키는 모듈은 그 이벤트에 의해 어떤 다른 모듈의 어떤 절차가 호출되는가를 알 필요가 없다[4].

블랙보드 구조를 통합하는 다중 에이전트 구조에 이벤트 기반의 암시적 호출 구조 패턴을 적용하면 제어 에이전트와 지식 원천 에이전트들 간의 결합도의 감소, 시스템 구성 요소들의 동적 재구성의 용이성, 그리고 지식 원천 에이전트들의 제어에 대한 융통성의 증대와 같은 여러 가지 바람직한 특성을 기대할 수 있다. 왜냐하면, 암시적 호출 구조 패턴을 사용하는 시스템에서는 시스템을 구성하는 요소들 간에 직접적인 호출이 일어나지 않으며 대신 이벤트들의 통보를 통한 간접적인 호출이 일어나기 때문이다.

음향 탐지기의 음향 신호들을 해석하기 위한 블랙보드 시스템인 HASP 시스템[7]은 이벤트 기반의 암시적 호출 구조를 사용하고 있다. HASP 시스템은 미리 정의된 세 가지 종류의 이벤트 타입들을 이용하여 시스템을 제어한다. HASP 시스템에서는 지식 원천들의 제어를 위한 제어 지식이 코드가 아니라 생성 규칙(production rule)의 형태로 표현되며, 블랙보드 수행 사이클의 매 단계마다 이벤트들의 발생이 탐지되고 그들 중 선정된 이벤트에 대응하는 절차들이 호출된다. HASP 시스템은 이벤트 기반의 암시적 호출 구조 패턴을 사용하였지만 사용할 수 있는 이벤트 타입들이 제한되어 있었다는 제약을 갖는다. HASP 시스템은 Rete 네트워크 기술과 같은 범용의 패턴 매칭 기술을 사용하지 않았다. Rete 네트워크는 미리 정해진 이벤트들만이 아니라 다양한 데이터 패턴들 그리고 그들의 조합에 대응하는 다양한 블랙보드 이벤트 타입들을 지원할 수 있다.

블랙보드 시스템에서 이벤트 기반의 암시적 호출 구조 패턴을 사용하는 개념을 기술한 기존 연구[8]에서는, 블랙보드의 상태가 변화할 때마다 지식 원천을 활성화시키는 블랙보드 이벤트들이 생성될 수 있으며, 각 지식 원천은 자신의 활성화 여부를 판단하기 위하여 블랙보드를 검색하는 대신 자신이 관심을 가진 블랙보드 이벤트 타입들을 시스템에게 등록한다. 이 후 블랙보드 이벤트가 발생하면 그 블랙보드 이벤트의 타입에 따라 해당 지식 원천 에이전트에게 그 이벤트가 통보된다. 또한 본 연구와 마찬가지로 블랙보드 시스템과 다중 에이전트 시스템을 이벤트 기반의 암시적 호출 구조 패턴을 이용하여 통합하는 방안을 기술한 기존 연구[3]에서는 그 통합 방안 자체의 장점들에 대하여 기술하고 있다. 하지만 기존 연구들[3, 8]은 본 논문과 달리 지식 원천들 또는 지식 원천 에이전트들을 활성화시킬 블랙보드 이벤트들을 효율적으로 탐지하기 위한 메커니즘에 대하여 언급하지 않는다.

2.2 Rete 네트워크와 생성 규칙 시스템들

Rete 네트워크는 다량의 패턴들과 다량의 데이터간의 패턴 매칭을 신속하게 수행하기 위한 범용의 패턴 매칭 기술이다[5]. Rete 네트워크 기술은 이론적으로 패턴들의 수가 늘어나도 패턴 매칭의 성능이 감소하지 않는다는 특징을 갖고 있다. Rete 네트워크 기술은 데이터와 패턴들의 매칭의 결과를 계산할 때 이전에 수행한 매칭 작업들의 중간 결과를 내부적으로 기억하여 새롭게 변화된 데이터에 대해서만 매칭 작업이 수행되도록 하므로, 시간의 흐름에 따라 부분적이고 점진적인 상태 변화를 일으키는 데이터에 대한 반복적인 패턴 매칭 작업을 매우 효과적으로 수행한다. 예를 들어, 규칙 시스템의 작업 메모리(working memory)와 같이 데이터의 삽입, 삭제, 수정 연산에 의하여 데이터의 일부분만이 점진적으로 변화하는 경우 Rete 네트워크 기술은 매우 효과적으로 패턴 매칭을 수행한다.

따라서 Rete 네트워크가 발명된 후 JESS[9], OPS5[10], 그리고 Drools[11]와 같은 큰 규모의 규칙 베이스(rule base)를 지원하는 생성 규칙 시스템들이 Rete 네트워크 기술을 사용하여 개발되었다. 생성 규칙 시스템들은 충돌 집합(conflict set) 즉 작업 메모리의 현재 상태에서 잠재적으로 활성화되어 수행될 수 있는 규칙들의 집합을 신속하게 탐지하기 위하여 Rete 네트워크 기술을 사용한다. 생성 규칙 시스템에서 충돌 집합을 탐지하는 메커니즘의 성능은 전체 시스템의 성능을 결정하는 매우 중요한 요소이다.

2.3 지식 원천들을 활성화시킬 수 있는 블랙보드 상태 변화의 탐지

생성 규칙 시스템에서 규칙들의 조건부를 구성하는 패턴들과 작업 메모리의 사실들 간의 패턴 매칭을 통하여 충돌 집합을 계산하는 메커니즘의 성능이 전체 시스템의

성능에 큰 영향을 주는 것과 마찬가지로, 블랙보드 구조 또는 블랙보드 구조를 통합하는 다중 에이전트 구조에서는 블랙보드의 상태가 변할 때마다 지식 원천 에이전트들을 활성화시킬 수 있는 블랙보드 이벤트들의 발생을 효과적으로 탐지하기 위한 메커니즘의 성능이 매우 중요하다. 왜냐하면, 블랙보드 이벤트들의 탐지 작업은 블랙보드 수행 사이클에서 반복적으로 수행되어야 하며, 많은 량의 데이터와 많은 량의 패턴들 간의 매칭 작업을 필요로 할 수 있기 때문이다.

본 논문에서 블랙보드 이벤트란 지식 원천 에이전트를 활성화시킬 수 있는 블랙보드의 상태 변화를 표현하는 자료 구조를 의미한다. 블랙보드 이벤트에 대한 자세한 내용은 제 3.3절에서 기술된다.

Unification 알고리즘[12]과 같은 일반적인 패턴 매칭 함수를 사용하여 지식 원천들을 활성화시키는 블랙보드 상태 변화 또는 그 상태 변화에 대응되는 블랙보드 이벤트들을 반복적으로 탐지하는 방법은 효과적이지 않다. 왜냐하면 데이터와 패턴들 간의 패턴 매칭의 결과를 반복적으로 계산해야 하는 경우, 새롭게 변화한 데이터에 관련된 패턴 매칭만을 선별적으로 수행하는 Rete 네트워크와 달리, Unification 알고리즘은 주어진 모든 데이터와 모든 패턴들에 대한 매칭을 반복 수행하기 때문이다.

따라서 기존 블랙보드 시스템들은 지식 원천들을 활성화시키는 블랙보드 상태 변화 또는 그 상태 변화에 대한 블랙보드 이벤트를 탐지하는 부담을 줄이기 위하여 트리거링 조건(triggering condition), 식별 네트워크(discrimination network), 데몬(demon)과 같은 기법들을 사용하고 있다. HEARSAY-II [13]는 활성 지식 원천들의 탐지를 위하여 데몬 기반의 블랙보드 활성화 메커니즘을 사용한다. 데몬은 블랙보드 데이터의 특정한 변화를 감시하고 그런 변화가 발생하였을 때 자동으로 활성화되는 작은 프로세스를 의미한다. 데몬들은 지식 원천들이 활성화될 수 있는 지를 평가하기 위한 인터럽트가 통보되도록 하는 역할을 한다. HEARSAY-II의 데몬 기반의 활성 지식 원천탐지 메커니즘은 이후에 개발된 블랙보드 시스템들에 많은 영향을 미쳤다. Poligon 시스템[14]도 블랙보드의 상태 변화를 감시하는 데몬들을 이용하여 지식 원천들을 활성화시킨다. HEARSAY-II와 달리 Poligon 시스템은 지식 원천들 중에서 가장 유망한 지식 원천을 선정하기 위한 중앙 집중적 기능을 갖지 않는다. BB1 시스템 [15]는 지식 원천들의 활성화 단계의 효율을 높이기 위하여 트리거링 조건을 사용하고 있다. 또한 BB1 시스템의 초기 버전은 트리거링 조건을 식별 네트워크로 컴파일 하여 트리거링 단계를 효율적으로 수행하는 기법을 사용하였다. 하지만 활성화된 지식 원천들을 탐지하기 위하여 데몬이나 트리거를 사용하는 메커니즘들의 공통적인 문제는 데몬이나 트리거들의 수가 많아지거나 데몬이나 트리거들이 감시해야 하는 조건 패턴들이 복잡해지면 많은 자원을 소모하게 될 수 있다는 것이다.

본 논문이 제안하는 구조 통합 방안은 기존 블랙보드 시스템들처럼 데몬 기반의 활성 지식 원천 탐지 메커니즘을 사용하거나 Unification 알고리즘과 같은 일반적인 패턴 매칭 함수를 직접 호출하여 블랙보드 이벤트를 탐지하는 대신 Rete 네트워크 기반의 블랙보드 이벤트 탐지 모듈을 사용한다. Rete 네트워크 기술은 범용의 패턴 매칭 기술이지만 주로 생성 규칙 시스템들을 위한 기술로 사용되었다. Rete 네트워크 기술은 전통적인 블랙보드 시스템들에서는 사용되지 않았다. HEARSAY-II나 HASP와 같은 초기의 블랙보드 시스템들이 개발되었던 시점보다 Rete 네트워크 기술이 발명된 시점이 늦었다는 점도 고려되어야 한다.

블랙보드 시스템의 블랙보드 데이터와 블랙보드 데이터 패턴들 간의 매칭은 생성 규칙 시스템의 작업 메모리의 사실들과 규칙의 조건부를 구성하는 패턴들 간의 매칭과 유사성을 갖는다. 그 유사성은 생성 규칙 시스템에서의 작업 메모리의 상태 변화와 마찬가지로 블랙보드 구조에서의 블랙보드의 상태 변화도 부분적이고 점진적으로 일어나는 경향이 있다는 것이다. 따라서 블랙보드 데이터와 블랙보드 데이터 패턴들 간의 패턴 매칭에 Rete 네트워크 기술을 적용하는 것은 합리적이라고 할 수 있다.

3. 블랙보드 데이터와 에이전트 메시지

이 장에서는 본 논문이 제안하는 통합 방안에서 사용되는 블랙보드 데이터와 에이전트 메시지에 대한 내용을 기술한다.

3.1 블랙보드 데이터와 블랙보드 데이터 패턴의 표현

본 논문에서는 GL(Generalized List) 언어[16]를 사용하여 블랙보드 데이터와 블랙보드 데이터 패턴들을 표현한다. GL 언어는 심벌, 변수, 상수, 수식 등을 원소로 갖는 중첩 리스트 구조의 일종이라고 할 수 있다. GL 언어에서 변수를 나타내는 심벌은 $로 시작된다. GL 언어로 표현된 블랙보드 데이터나 블랙보드 데이터 패턴의 첫 번째 요소는 심벌이어야 한다. GL 언어로 표현된 블랙보드 데이터의 예는 다음과 같다.

- (position 10 20)
- (robot (position 10 20)(battery “stable”))

블랙보드 데이터 패턴이란 블랙보드 데이터에 매칭될 수 있는 데이터 패턴을 의미한다. 블랙보드 데이터는 변수나 수식을 갖지 못하지만 블랙보드 데이터 패턴은 변수나 수식을 포함할 수 있으며 and와 같은 논리 접속사를 포함할 수 있다. GL에서 수식은 #로 시작하는 리스트 형태로 표현되며 수식은 참이나 거짓으로 평가된다. 수식에 포함된 어떤 변수가 어떤 값과 바인딩 되지 않는 경우 그 수식은 거짓으로 평가된다. GL 언어로 표현된 블랙보드 데이터 패턴의 예들은 다음과 같다. GL 언어에서는 괄호들의 시각적 구분을 편하게 위하여 소괄호 (와) 대신 중괄호 [와]를 사용할 수 있다.

- (person (name “kim”) (age 30))
- [tank (heat 300) (pressure $y)] and #(> $y 3.5))

블랙보드 데이터와 블랙보드 데이터 패턴과의 패턴 매칭의 결과는 바인딩(binding)이다. 바인딩이란 변수와 값의 쌍들의 목록을 의미한다. 예를 들어, (p (a $x) (b $y)) and (> $x 10) 이라는 블랙보드 데이터 패턴과 블랙보드 데이터 (p (a 11) (b 33.3)) 의 패턴 매칭의 결과로 산출되는 바인딩은 [($x 11) ($y 33.3)] 이다. 변수를 포함하지 않는 블랙보드 패턴과 블랙보드 데이터와의 매칭의 결과는 블랙보드 패턴과 블랙보드 데이터가 동일하면 빈 바인딩 [ ] 이고 아니면 False로 표현된다.

본 논문이 제안하는 통합 방안은 GL 이외의 다른 다양한 블랙보드 데이터 표현을 지원할 수 있다. Rete 네트워크는 특정한 지식 표현에 한정되지 않는 범용의 매칭 기술이다. Rete 네트워크를 사용하는 생성 규칙 시스템들[10, 11]은 다양한 지식 표현 방식들을 사용하고 있다.

3.2 에이전트 메시지의 표현

에이전트들 간의 통신을 위해서는 에이전트 메시지의 표현에 대한 기준이 필요하다. 그런 기준은 에이전트 통신 언어(agent communication language)의 규격에 의해 정해진다. 본 논문은 기술의 편의상 에이전트들이 에이전트 통신 언어로 KQML[17]을 사용한다고 가정한다.

지식 원천 에이전트들은 블랙보드 에이전트에게 블랙보드 데이터의 삽입, 삭제, 수정 명령을 담은 에이전트 메시지를 보내서 블랙보드의 상태를 변경시킬 수 있다. 예를 들어, 지식 원천 에이전트 KS1이 블랙보드 에이전트에게 블랙보드 데이터 (tank1 (heat 200) (pressure 5.5))를 블랙보드에 삽입해달라고 요청하는 KQML 에이전트 메시지는 다음과 같이 표현될 수 있다. 일반적으로 모든 에이전트는 분산 환경 속에서 서로 구분될 수 있도록 고유한 ID를 갖는다. 이후 블랙보드 에이전트의 ID를 BBAgent라고 가정한다.

```
(assert
   :sender “KS1”  :receiver “BBAgent”
   :language “GL”
   :content “(tank1 (heat 200) (pressure 5.5))” )
```

지식 원천 에이전트들은 블랙보드 에이전트에게 자신이 관심을 가진 블랙보드 데이터 패턴을 등록하기 위한 에이전트 메시지를 보낼 수 있다. 예를 들어, 지식 원천 에이전트 KS1이 블랙보드 에이전트에게 자신이 관심을 가진 블랙보드 데이터 패턴 (tank1 (heat $x) (pressure $y)) and #(> $x 190) and #(> $y 5.0)을 등록하기 위한 에이전트 메시지는 다음과 같이 표현될 수 있다.

```
(subscribe
   :sender “KS1”   :receiver “BBAgent”
   :language “GL”
   :content
     “(tank1 (heat $x) (pressure $y)) and #(> $x 190) and #(> $y 5.0)” )
```

본 논문에서 지식 원천 에이전트들은 블랙보드 데이터 패턴의 등록 및 등록 해지를 동적으로 요청할 수 있다. Rete 네트워크가 패턴의 동적인 등록과 등록 해지를 지원할 수 있기 때문이다.

3.3 블랙보드 이벤트와 블랙보드 이벤트 메시지

지식 원천 에이전트가 등록한 블랙보드 데이터 패턴과 블랙보드 데이터 간에 매칭이 발생하였음을 나타내는 자료 구조인 블랙보드 이벤트를 생성하는 것은 제 4장의 그림 3에서 기술될 블랙보드 이벤트 탐지 모듈이다. 본 논문에서 블랙보드 이벤트는 다음 요소들의 쌍을 의미한다.

- 블랙보드 패턴 자체
- 블랙보드 패턴과 블랙보드 데이터의 패턴 매칭의 결과로 생성된 바인딩

예를 들어, 어떤 지식 원천이 블랙보드 데이터 패턴 (p $x 44)를 등록하였으며, 블랙보드에 데이터 (p 33 44)와 (p 22 44)가 삽입된다면 패턴 매칭의 결과로 두 개의 바인딩들 [($x 33)]와 [($x 22)]가 생성되므로, 결과적으로

블랙보드 이벤트 탐지 모듈에 의해 블랙보드 이벤트 ((p $x 44) [($x 33)])와 ((p $x 44) [($x 44)]) 가 탐지된다.

제어 에이전트가 지식 원천 에이전트에게 그 지식 원천 에이전트가 등록한 블랙보드 데이터 패턴을 만족하는 상태 변화가 발생하였음을 통보하기 위한 메시지인 블랙보드 이벤트 메시지는 :content 파라미터 값으로 블랙보드 이벤트를 담고 있다. 예를 들어, 블랙보드 데이터 패턴 (p (heat $x) (state $y)) and #(> $x 22)에 대한 바인딩 [($x 25) ($y “stable”)] 을 담은 블랙보드 이벤트를 제어 에이전트가 지식 원천 에이전트 KS1에게 통보하기 위한 블랙보드 이벤트 메시지는 다음과 같이 표현될 수 있다. 제어 에이전트의 ID를 ControlAgent 라고 가정한다.

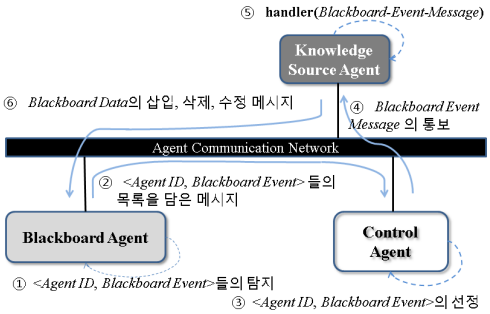
(notify
:sender “ControlAgent”
:receiver “KS1”
:language “GL”
:content “((p (heat $x) (state $y)) and #(> $x 22) [($x 25) ($y \“stable\”)])”)

본 논문이 제안하는 통합 구조에서 지식 원천 에이전트는 수신된 블랙보드 이벤트 메시지를 블랙보드 이벤트 메시지 처리기로 처리한다. 지식 원천 에이전트의 블랙보드 이벤트 메시지 처리기는 블랙보드 이벤트 메시지를 인자로 사용하여 자동 호출되는 콜백 함수(call-back function)이다. 지식 원천 에이전트의 세부적 기능은 블랙보드 이벤트 메시지 처리기의 내부 코드가 어떤 블랙보드 이벤트들을 어떻게 처리하도록 구현하는가에 따라 달라진다.

4. 통합 구조

4.1 블랙보드 이벤트 기반의 암시적 호출 구조 패턴

본 논문이 제안하는 통합 방안에서 사용하는 블랙보드 이벤트 기반의 암시적 호출 구조 패턴의 참여자는 그림 1과 같이 블랙보드 에이전트, 제어 에이전트(control agent), 그리고 지식 원천 에이전트(knowledge source agent)들이다.



[그림 1] 블랙보드 이벤트 기반의 암시적 호출 구조 패턴의 사용

[Fig. 1] Usage of Blackboard Event based Implicit Invocation Architectural Pattern

그림 1을 구성하는 에이전트들은 블랙보드 시스템에 주어진 문제에 대한 해답을 찾을 때까지 반복되는 블랙보드 수행 사이클을 통해 협동한다. 본 논문에서 블랙보드 수행 사이클은 다음에 기술되는 [단계 1]부터 [단계 4]까지를 반복적으로 수행한다. 블랙보드 수행 사이클의 반복 수행을 시작하기 직전에 모든 지식 원천들은 자신이 관심을 가진 블랙보드 데이터 패턴들을 블랙보드 에이전트에게 등록하고 블랙보드 에이전트는 블랙보드에 초기화 블랙보드 데이터를 적재한다. 그림 1에서 handler (*Blackboard- Event-Message*)는 지식 원천 에이전트의 블랙보드 이벤트 메시지 처리기를 의미한다.

[단계 1] 블랙보드 이벤트들의 탐지

블랙보드 에이전트는 어떤 지식 원천 에이전트가 등록한 어떤 블랙보드 데이터 패턴이 블랙보드의 현재 상태와 패턴 매칭이 일어나는가를 탐지한다. 그리고 탐지된 모든 지식 원천 에이전트의 ID와 그 에이전트가 등록한 블랙보드 데이터 패턴에 대한 블랙보드 이벤트의 쌍들의 목록을 제어 에이전트에게 전송한다. 이 단계는 그림 1의 ①번과 ②번에 해당하는 단계이다.

[단계 2] 스케줄링 및 블랙보드 이벤트의 통보

제어 에이전트는 블랙보드 에이전트로부터 수신한 지식 원천 에이전트의 ID와 블랙보드 이벤트의 쌍들의 목록으로부터 시스템의 문제 풀이 과정에 가장 기여도가 클 것이라고 평가되는 하나의 쌍을 선정한다. 그리고 그 쌍을 $<agentID_i, blackboardEvent_j>$라고 한다면, 제어 에이전트는 지식 원천 에이전트 $agentID_i$에게 $blackboardEvent_j$를 담은 블랙보드 이벤트 메시지를 통보한다. 이 단계는 그림 1의 ③과 ④번에 해당하는 단계이다.

[단계 3] 지식 원천 에이전트의 수행

블랙보드 이벤트 $blackboardEvent_j$를 담은 메시지를 통보받은 지식 원천 에이전트 $agentID_i$의 블랙보드 이벤트 메시지 처리기 handler()가 호출되어 주어진 블랙보드 이벤트 메시지를 처리한다. 그리고 블랙보드 이벤트 메시지 처리기의 수행 결과로 블랙보드 데이터의 삽입, 삭제, 수정 메시지가 블랙보드 에이전트에게 전달되어 블랙보드의 상태가 변화될 수 있다. 이 단계는 그림 1의 ⑤와 ⑥번에 해당하는 단계이다.

[단계 4] 반복

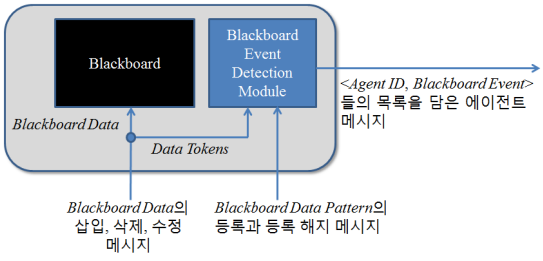
단계 1로 돌아간다.

블랙보드 수행 사이클의 [단계 3]에서 블랙보드 이벤트 메시지를 수신한 지식 원천 에이전트는 자신이 블랙보드 에이전트에게 등록한 어떤 블랙보드 데이터 패턴이 어떤 블랙보드 데이터와 어떻게 매칭이 되었는가를 파악할 수 있다. 왜냐하면 블랙보드 이벤트 처리기의 인자로 사용된 블랙보드 이벤트 메시지가 담고 있는 블랙보드 이벤트는 제 3.3절에서 기술한 바와 같이 지식 원천 에이전트가 등록한 블랙보드 데이터 패턴과 블랙보드 데이터와의 패턴 매칭의 결과로 생성되는 바인딩을 담고 있기 때문이다.

본 논문이 제안하는 통합 구조는 제 2.1 절에서 기술한 바와 같은 암시적 호출 구조 패턴을 사용하는 구조의 장점들을 반영할 수 있다.

4.2 블랙보드 에이전트

본 논문이 제안하는 통합 구조의 블랙보드 에이전트는 그림 2와 같이 블랙보드 뿐 아니라 블랙보드 이벤트 탐지 모듈(blackboard event detection module)을 가지고 있다.



[그림 2] 블랙보드 에이전트
[Fig. 2] Blackboard Agent

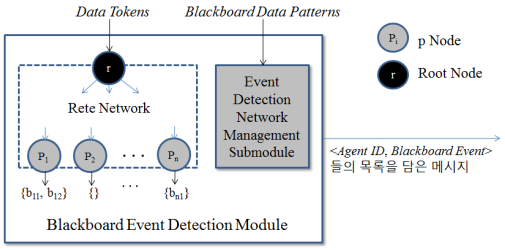
본 논문의 블랙보드 에이전트는 두 종류의 에이전트 메시지들을 수신한다. 그 중 한 종류는 블랙보드 데이터의 삽입과 삭제를 위한 에이전트 메시지들이다. 지식 원천 에이전트들은 블랙보드 에이전트에게 새로운 블랙보드 데이터를 삽입하기 위한 에이전트 메시지를 보내거나, 지정된 블랙보드 데이터 패턴을 만족하는 기존 블랙보드 데이터를 모두 삭제하기 위한 에이전트 메시지를 보낼 수 있다. 블랙보드 데이터의 수정은 블랙보드 데이터 삭제와 삽입의 결합으로 수행될 수 있다. 블랙보드에 새로운 데이터가 삽입되거나 삭제될 때마다 그에 대응되는 데이터 토큰(data token)이 생성되어 블랙보드 이벤트 탐지 모듈로 전달된다. 데이터 토큰은 데이터 자체의 복사를 피하기 위하여 데이터의 복사본 대신 데이터에 대한 참조를 담은 일종의 구조체이다.

블랙보드 에이전트가 수신하는 다른 한 종류의 에이전트 메시지들은 지식 원천 에이전트가 자신이 관심을 가진 블랙보드 데이터 패턴을 등록하거나 자신이 등록한 블랙보드 데이터 패턴의 등록 해제를 위한 메시지들이다. 예를 들어, 지식 원천 에이전트 KS1이 (boiler1 (pressure $x)) and #(> $x 10) 이라는 블랙보드 데이터 패턴을 블랙보드 에이전트에게 등록하는 에이전트 메시지는 다음과 같이 표현될 수 있다.

```
(subscribe
   :sender "KS1"
   :receiver "BBAgent"
   :language "GL"
   :content "(boiler1 (pressure $x) and #(> $x 10)" )
```

블랙보드 이벤트 탐지 모듈은 지식 원천 에이전트들이 보낸 블랙보드 이벤트 패턴들로부터 블랙보드 이벤트들을 탐지하기 위한 Rete 네트워크를 생성한다. 일반적으로 Rete 네트워크는 하나의 루트 노드, 알파 노드들, 베타 노드들, 그리고 네트워크의 가장 말단에 위치하는 p 노드들로 구성된다[5]. 루트 노드는 블랙보드에 입력되는 모든 블랙보드 데이터에 대응하는 데이터 토큰들이 입력되는 노드이다. 알파 노드는 데이터 토큰들에 대한 필터 역할을 한다. 베타 노드들은 데이터 토큰들을 결합(join)하는 역할을 한다. 알파 노드들과 베타 노드들은 루트 노드와 p 노드들의 사이에 위치한다.

블랙보드 이벤트 탐지 모듈은 그림 3과 같이 지식 원천 에이전트들이 등록한 블랙보드 데이터 패턴들로부터 생성된 Rete 네트워크와 그 Rete 네트워크를 생성하고 관리하는 이벤트 탐지 네트워크 관리 부모듈(event detection network management sub-module)로 구성된다.



[그림 3] 블랙보드 이벤트 탐지 모듈
[Fig. 3] Blackboard Event Detection Module

Rete 네트워크를 생성하는 데 사용된 하나의 데이터 패턴으로부터 하나의 p 노드가 생성된다. 본 논문에서 각 p 노드의 출력은 개념적으로 그 p 노드에 대응하는 블랙보드 데이터 패턴과 블랙보드 데이터와의 패턴 매칭의 결과로 산출되는 바인딩들의 집합이다. 예를 들어, 그림 3에서 집합 $\{b_{11}, b_{12}\}$은 p 노드 P_1이 출력한 바인딩들의 집합을 의미한다. 참고로, GL 언어는 집합을 표현하는 기능이 없으므로 본 논문에서는 바인딩들의 집합 대신 바인딩들의 목록을 사용하고 있다.

제 3.3 절에서 기술한 바와 같이 블랙보드 데이터 패턴과 블랙보드 데이터간의 패턴 매칭의 결과로 p 노드들이 출력하는 바인딩들의 목록은 블랙보드 이벤트들을 생성하는 데 사용된다. 예를 들어, 지식 원천 에이전트 KS1이 블랙보드 에이전트에게 블랙보드 데이터 패턴(box $x 22)를 등록하였다면 블랙보드 이벤트 탐지 모듈의 Rete 네트워크에는 그 패턴에 대응되는 p 노드가 생성된다. 이제 블랙보드 데이터 (b 15 22)와 (b 16 22)가 어떤 지식 원천 에이전트들에 의해 블랙보드에 입력된다면 그 p 노드는 두 개의 바인딩을 담고 있는 목록 ([($x 15)] [($x 16)]) 을 출력할 것이고, 블랙보드 이벤트 탐지 모듈은 에이전트 ID와 블랙보드 이벤트의 쌍 (“KS1” ((b $x 22) [($x 15)])) 와 (“KS1” ((b $x 22) [($x 16)]))을 담은 다음과 같은 에이전트 메시지를 생성하여 제어 에이전트에게 보낼 것이다.

```
(tell
   :sender “BBAgent”
   :receiver “ControlAgent”
   :language “GL”
   :content
   “((\”KS1\” ((b $x 22) [($x 15)]))
     (\”KS1\” ((b $x 22) [($x 16)])))”)
```

본 논문이 이벤트 기반의 암시적 호출 구조 패턴을 사용한 블랙보드 구조와 다중 에이전트 구조의 통합 방안에 Rete 네트워크 기술을 함께 사용하는 것은 Rete 네트워크 기술이 다음과 같은 여러 가지 장점들을 갖기 때문이다.

- Rete 네트워크 기술은 다량의 데이터와 다량의 데이터 패턴들의 매칭을 신속하게 계산하기 위한 기술이며 Rete 네트워크의 이론적 수행 속도는 패턴들의 수가 늘어나도 변화하지 않는다고 알려져 있다. Rete 네트워크 기술은 기존의 블랙보드 시스템들이 사용하는 트리거나 데몬 기술과 달리 운영 체계의 특성에 영향을 받거나 감시 패턴들의 수가 많아져도 성능이 크게 저하되지 않는다. 따라서 본 논문의 Rete 네트워크 기반의 블랙보드 이벤트 탐지 모듈은 다량의 블랙보드 패턴들이 사용되어야 하는 블랙보드 응용 분야들을 지원할 수 있다.
- Rete 네트워크 기술의 특징은 패턴 매칭의 대상이 되는 데이터 전체가 아니라 새롭게 삽입되거나 삭제되거나 수정된 데이터에 대해서만 선택적으로 패턴 매칭을 수행하는 기법을 사용하여 패턴 매칭의 속도를 향상시키는 것이다. 블랙보드 구조에서, 블랙보드의 상태는 생성 규칙 시스템의 작업 메모리와 마찬가지로 새로운 데이터의 삽입과 기존 데이터의 삭제 및 수정에 의해서 부분적이고 점진적으로 변화하는 경향을 갖는다. 따라서 블랙보드 데이터와 블랙보드 데이터 패턴들 간의 매칭에 Rete 네트워크 기술을 사용하는 것은 합리적이라고 할 수 있다.
- Rete 네트워크 기술은 특정한 데이터 표현 기법들만을 지원하는 패턴 매칭 기술이 아닌 범용 패턴 매칭 기술이다. 따라서 본 논문이 제안하는 Rete 네트워크 기반의 블랙보드 이벤트 탐지 메커니즘은 다양한 종류의 데이터 표현 기법들을 지원할 수 있다.
- 또한 Rete 네트워크 기술은 데이터 패턴들의 동적인 등록 및 등록 해지를 지원할 수 있다. 예를 들어 Rete 네트워크를 사용하는 생성 규칙 시스템의 하나인 Drools [11]는 생성 규칙들의 동적 등록과 동적 등록 해지를 지원하고 있다. 본 논문이 제안하는 통합 다중 에이전트 구조도 블랙보드 데이터 패턴의 동적인 등록 과 등록 해지를 지원할 수 있으며, 따라서 지식 원천들을 블랙보드 시스템의 수행 중에 동적으로 문제 풀이 과정에 참여시키거나 참여를 중단시킬 수 있는 제어의 융통성을 갖게 된다.

5. 결론

본 논문은 블랙보드 구조를 다중 에이전트 구조에 통

합하기 위한 방안을 제안하였다. 제안된 방안은 Rete 네트워크 기반의 블랙보드 이벤트 탐지 메커니즘과 블랙보드 이벤트 기반의 암시적 호출 구조 패턴을 사용한다.

제안 통합 구조가 사용하는 블랙보드 이벤트 기반의 암시적 호출 구조는 제어 에이전트와 지식 원천 에이전트들 간의 결합도를 감소시키고 블랙보드 시스템의 제어에 대한 융통성을 증대시킨다. 또한 본 논문이 제안하는 통합 구조에서 사용되는 Rete 네트워크 기반의 블랙보드 이벤트 탐지 메커니즘은 블랙보드 수행 사이클의 블랙보드 이벤트 탐지 단계의 수행 성능을 높일 수 있도록 한다.

이벤트 기반의 암시적 호출 구조 패턴 자체와 Rete 네트워크 기술 자체는 각각 이미 잘 알려진 기술이지만 본 논문의 의의는 블랙보드 이벤트 기반의 암시적 호출 구조 패턴과 Rete 네트워크 기술을 결합하여 블랙보드 구조를 다중 에이전트 구조에 통합하는 효과적인 방안을 제시한 것이라 할 수 있다.

References

[1] Daniel D. Corkill, "Collaborating Software Blackboard and Multi-Agent Systems & the Future", In Proceedings of the International Lisp Conference, New York, New York, October, 2003.

[2] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Pattern-Oriented Software Architecture, Vol. 1, pp. 71 – 83, John Wiley & Sons Ltd, 1996.

[3] Jing Dong, Shanguo Chen, and Jun-Jang Jeng, “Event-based Blackboard Architecture for Multi-Agent Systems”, The Proceedings of the IEEE International Conference on Information Technology Coding and Computing, Vol. 2, IEEE, pp. 379-384, April, 2005.

[4] David Garlan, Mary Shaw, “An Introduction to Software Architecture,” Advances in Software Engineering and Knowledge Engineering, Vol. 1, World Scientific Publishing Company, New Jersey, 1993.

[5] Charles Forgy, "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", Artificial Intelligence, 19, pp.17 – 37, 1982.

[6] Byul-Saim Kwak, Jaeho Lee, “A System Integration Methodology for Control of Intelligent Robots”, Journal of The Korean Institute of Information Scientists and Engineers, v.24, No.3, pp.24-34, March, 2006.

[7] Penny Nii, Blackboard Model of Problem Solving, The AI Magazine, Vol. 7, No. 2, pp. 38-53, 1986.

[8] Daniel D. Corkill, “Blackboard Systems”, AI Expert 6(9), pp.40-47, September, 1991.

[9] Ernest Friedman-Hill, *Jess in Action: Rule Java Rule-Based Systems*, Manning Publications, ISBN 1-930110-89-8, 2003.

[10] Lee Brownston, Robert Farrell, Elaine Kant, Nancy Martin, *Programming Expert Systems in OPS5*, Addison-Wesley, ISBN 0-201-10647-7, 1985.

[11] http://www.jboss.org/drools/, the official website of Drools.

[12] George F. Luger, Artificial Intelligence Structures and Strategies for Complex Problem Solving, pp. 66-72, 5th Edition, Addison Wesley, 2005.

[13] Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser, and D. Raj Reddy, “The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty", Computing Surveys, 12(2), pp. 213-253, June, 1980.

[14] Poligon - J. Rice, N. Aiello and H.P. Nii, “See How They Run... The Architecture and Performance of Two Concurrent Blackboard Systems", *Blackboard Architectures and Applications*, V. Jagannathan, R. Dodhiawala and L. S. Baum, editors, Academic Press, pp. 153-178, 1989.

[15] Hayes-Roth, B. and Hewett, M., “BB1: An Implementation of the Blackboard Control Architecture", Blackboard Systems, Engelmore, R., and T. Morgan, editors, pp. 297-313, Addison-Wesley, 1988.

[16] SungKyu Kang, Jaeho Lee, “Knowledge management Framework 2.4 Manual”, Center for Intelligent Robitics, 2007.

[17] Tim Finin, Don McKay, Rich Fritzson and Robin McEntire, “KQML: An Information and Knowledge Exchange Protocol”, Int. Conf. on Building and Sharing of Very Large-Scale Knowledge Bases, Tokyo, December, 1993.

장 혜 진(Hai Jin Chang) **[정회원]**

- 1994년 2월 : 서울대학교 대학원 계산통계학과 박사 졸업(전산학 전공)
- 1994년 3월 ~ 현재 : 상명대학교 공과대학 컴퓨터소프트웨어 공학과

<관심분야>
다중 에이전트 프레임워크, 로봇, 분산 제어 시스템, DRM.