EDA(Event-driven Architecture)를 적용한 BPMS

구훈영[1], 윤영하[2], 홍정식[3]
1) 한국전자통신연구원, koohy@etri.re.kr
2) BPMS 프리랜서, mail2yyh@gmail.com
3) 서울산업대학교, hong@snut.ac.kr

Abstract

BPMS는 애플리케이션으로부터 프로세스 논리를 분리한다는 혁신적인 사상으로 시작했지만, 이를 어떻게 이룰 것인지에 명확하지가 않다. 여기서는 EDA(Event Driven Architecture)를 프로세스 논리의 분리 기준으로 제시한다.

1. BPMS와 EDA 개요

BPMS는 비즈니스 프로세스와 애플리케이션을 분리하여 유연성과 확장성을 갖는 IT 시스템을 구축, 운영, 개선하는 데 도움을 주는 솔루션이다. 그러나, 기존의 BPMS는 주로 워크플로우 관리 기반의 솔루션으로 순차적인 업무 처리에 초점을 맞추어 개발되었다. 이는 또한 BPMS 개발 초기의 소프트웨어 엔지니어의 성향에서도 기인하는데, 이는 시스템의 프로세스 지원 보다는 에러없는 동작을 위한 시스템의 통제성에 초점을 맞추었던 것이 하나의 중요한 원인이라 할 수 있다 [Aalst 2002].

비즈니스 프로세스, 특히 사람까지 참여하는 비즈니스 프로세스는 복잡한 병렬 이벤트 처리가 필요하다. 이 경우, 프로세스 수행을 위한 태스크의 처리논리와 태스크 수행 완료 후의 프로세스 진행 논리가 연계된다면, 복잡한 동시성 제어 문제를 해결하기가 매우 어려워진다. 따라서 두 논리의 처리를 이벤트 큐 등을 이용해 단절시킬 필요가 있다. 이러한 복잡한 이벤트의 처리를 위해 이벤트와 큐의 개념을 중심으로 하는 이벤트 기반 아키텍쳐(Event-driven architecture)가 관심을 모으고 있다 [Michelson 2006].

2. EDA를 적용한 BPMS

시스템에서 대기(Queueing) 상황은 매우 흔한 현상이다. 이러한 대기(Queueing) 상황의 처리에 『Event-Driven』 방식은 어떤 식으로든 개입하게 된다. 『Event-Driven』 방식이 적용된 경우를 모두 EDA라 정의하는 것은 부적절하기 때문에, 다음과 같은 조건을 만족하는 경우를 EDA로 정의한다.

(1) Decoupled Interaction : 이벤트의 수신과 이벤트 처리논리가 분리되어야 함

(2) No Dependency on Event Processing : 이벤트 소스의 정의만을 활용하여 이벤트 처리가 가능해야 함

두 번째 조건을 만족하는 경우, 이벤트 소스가 이벤트 프로세싱 큐의 존재만을 인식하면 되기 때문에, 이벤트 프로세싱에 대한 개입을 없앨 수 있다. 즉 이벤트 소스의 처리논리와 이벤트 프로세싱의 처리논리가 완벽하게 분리된다는 것이다. BPMS의 경우, 이벤트 소스에서, (프로세스를 인식하여) 프로세스 시작 이벤트를 발생시키는 등의 상황은 이벤트 소스가 이벤트 프로세싱에 개입한 것으로 두 번째 조건을 만족하지 못한 것으로 본다. BPMS가 별도의 프로세스 실행 엔진과, 프로세스 정의 및 실행이 이벤트 소스의 정의만으로 가능한 경우, 프로세스 논리가 분리되었다고 정의한다.

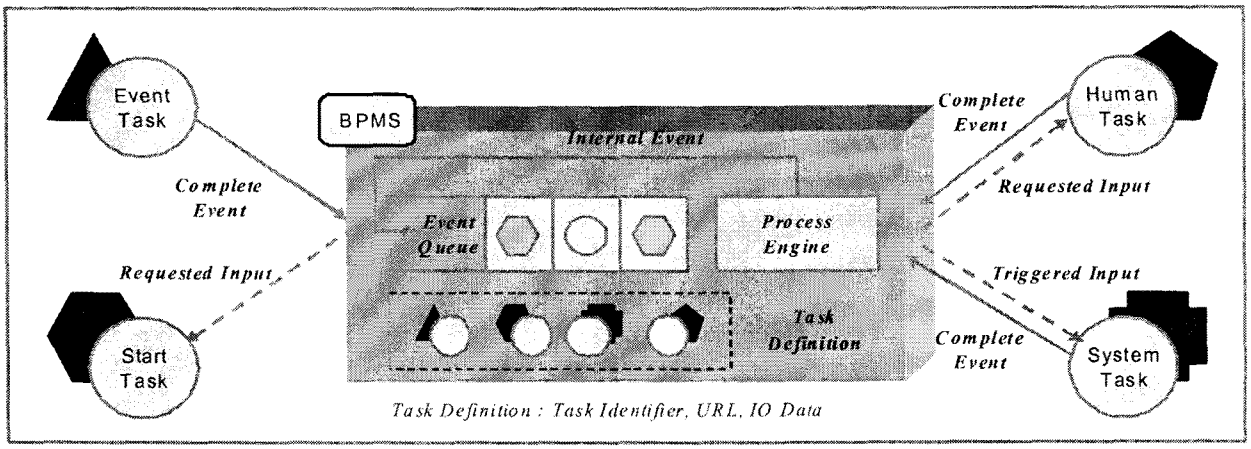
BPMS의 경우 이벤트 소스는 애플리케이션 또는 태스크, 이벤트 처리 시스템은 프로세스 실행 엔진으로 볼 수 있다. BPMS를 EDA로 정의하기 위해서는 이벤트 소스의 정의 즉 애플리케이션 또는 태스크의 정의만으로 프로세스 정의 및 실행이 가능해야 한다. BPMS가 EDA라면, 이벤트 소스의 처리논리(애플리케이션, 태스크 논리)와 프로세스 논리가 완벽하게 분리된다.

BPMS에서 이벤트 소스의 처리논리를 관여하지 않는 것처럼, 업무수행자가 어떤 능력을 가지고 있는지 등을 알지 못한다. BPMS에서는 이벤트 소스의 정의와 사용자와 사용자들이 어떤 역할(Role)에 속해 있는지를 알고 있을 뿐이다.

BPMS에 사용자를 등록할 때, 해당 사용자가 어떤 프로세스에 속하는 업무를 수행하는지를 사전에 알 필요가 없듯이, 애플리케이션을 등록할 때도 어떤 프로세스에 해당 애플리케이션이 사용되는지를 알 필요가 없어야 한다. 애플리케이션이나 역할이 프로세스와 관련되어 정의되지만, 이들을 BPMS에 등록할 때, 프로세스에 종속적으로 정의할 필요가 없다는 의미이다. 기존의 BPMS들은 사람에 대해서는 이와 같은 처리를 하지만, 애플리케이션은 프로세스에 종속적으로 정의하고 있다.

EDA 기반의 BPMS에서는 업무수행자는 이벤트 소스 즉 애플리케이션을 시작시키는 역할을 하며, BPMS는 애플리케이션을 시작할 수 있는 정보를 제공하고, 애플리케이션으로부터 프로세스 상태변화와 관련된 이벤트를 수신한다. 프로세스의 상태변화를 일으키는 이벤트는 인스턴스 생성, 소멸과 관련된 이벤트로 한정하며, 업무시작과 업무수행자 변경 등의 BPMS 운영과 관련된 이벤트는 성격이 다른 이벤트로 본다. 프로세스 상태변화를 일으키는 이벤트는 외부 이벤트 소스 뿐만 아니라 BPMS 자체에서도 발생하며, 이들을 특별히 구별할 필요는 없다.

[그림 1]은 BPMS에 적용된 EDA를 나타낸 것으로, 이벤트 소스를 시작시키기 위한 정보의 흐름을 추가하였다. 여기서 순수 이벤트 소스는 이벤트 태스크(Event Task)이고, Human Task나 System Task의 경우는 BPMS로부터 입력정보를 수신한다. 시작 태스크(Start Task)는 프로세스 상태변화를 일으키지 않는 더미(Dummy) 이벤트 소스로, 해당 업무를 수행할 수 있는 사람에게 업무 시작을 위한 워크아이템을 제공한다. Start Task는 해당 워크아이템의 관점에서는 더미(Dummy) 이벤트 소스이지만, 내부적으로 여러 이벤트 소스를 구현할 수 있기 때문에 프로세스 실행과 관련이 있다. 실제 많은 이벤트 소스들이 내부적으로 여러 이벤트 소스를 구현하기 때문에, 하나의 이벤트 소스만을 구현하는 단순 이벤트 소스의 관점이 아니라 『복합 이벤트 소스의 관점』에서 접근해야 한다. 기존의 워크플로우 제품들이 특정한 영역에만 적용되었던 것은 단순 이벤트 소스의 관점에서 접근했기 때문이다. 복합 이벤트 소스는 프로세스 중심의 애플리케이션을 구현하는 방식으로 설명하기 어렵다.



[그림 1] BPMS에 적용된 EDA

3. 추후 연구

이상에서 기술된 EDA의 장점을 활용하면 BPMS를 이용한 애플리케이션으로부터의 프로세스 분리가 가능하다. 애플리케이션으로부터 분리된 프로세스는 복잡한 인벤트 처리가 가능해지고 프로세스 수정에 따른 애플리케이션 수정이 용이해 지는 장점을 갖는다. 이러한 EDA의 장점을 활용한 새로운 BPMS의 개념적, 아키텍쳐를 보다 구체적으로 정의하고, 모델링 및 실행 측면 등에서의 연구가 필요하다.

Acknowledgement

본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술 개발사업의 일환으로 수행하였음. [2006-X-001-03, 실시간 우편물류 요소기술개발]

참고 문헌

[1] Brenda M. Michelson, Event-Driven

Architecture Overview, Patricia Seybold Group, February 2, 2006

[2] W.M.P. van der Aalst, Making Work Flow: On the Application of Petri Nets to Business Process Management, Lecture Notes In Computer Science, Vol. 2360 (Proceedings of the 23rd International Conference on Applications and Theory of Petri Nets table of contents, pp. 1~22), 2002