전사 수준의 통합 비즈니스 룰 리포지토리 구축을 위한 비즈니스 룰 관리 아키텍처에 관한 연구

허종원*, 최상호*

* 코리아엑스퍼트㈜

서울 영등포구 여의도동 24-1 율촌빌딩 14층

Tel: +82-2-782-5200, Fax: +82-2-782-5203

Abstract

최근 기업 경영활동의 의사결정시 사용되는 비즈니스 룰(Business Rule)을 정형화하고 하나의 시스템으로 구축하여 효과적으로 기업의 경쟁력을 제고하기 위한 노력이 다양하게 시도되고 있다. 비즈니스 룰 시스템 구축 작업의 경우 기업 내부에 비정형적으로 존재하는 비즈니스 룰을 체계적으로 관리하기 위해 BRMS(Business Rule Management System)와 같은 전문 관리 도구를 도입하나, 대부분의 경우 비즈니스 룰 리포지토리(Repository)를 단순히 기능별 혹은 업무별로 구성함으로 인해 동일한 내용의 룰이 서로 다른 룰 리포지토리에 중복 존재하게 되는 등 구조상의 문제점을 발생시킨다. 이로 인해 각 어플리케이션 간의 룰 또는 룰 세트(Rule Set) 공유 관계가 수동 관리되거나 중복 룰의 수정으로 인한 룰 세트별 버전 관리 문제 등 비즈니스 룰 리포지토리 운영의 어려움에 봉착하게 된다. 본 연구에서는 금융보험사의 룰웨어하우스 구축 사례를 통해 다양한 어플리케이션에서 참조되는 전사 수준의 비즈니스 룰 관리 아키텍처 구성 방법 및 각 방법이 지닌 장단점에 대해 분석한다. 본 연구의 결과를 토대로 다양한 어플리케이션에서 참조되고 수시로 변경되는 전사 수준의 통합 비즈니스 룰 관리 시스템 구축 방안에 대한 연구가 활성화되기를 기대한다.

Keywords:

Business Rule, Business Rule Management System, Rule Repository, Rule Management Architecture, BRE, BRMS

1. 서론

오늘날과 같이 급변하는 국내외의 경영 환경 속에서 안정적으로 기업 활동을 영위하고, 존속하기 위해서는 기업이 경영 환경의 변화에 효과적으로 그리고 민첩하게 대응하는 것이 매우 중요하다. 기업은 경영 활동 중에 필연적으로 발생하는 중요 의사결정을 성공적으로 수행하기 위하여 의사결정을 위한 자료를 신속히 수집하고, 적절한 지식을 적시에 활용해야 한다. 하지만, 대부분의 기업 정보 시스템은 기업이 활용하여야 하는 비즈니스 관련 지식을 집중적이고 체계적으로 관리하지 못하고 있는 실정이다. 이들 비즈니스 관련 지식은 대개 별도 구축된 여러 개의 기업 정보 시스템에 걸쳐 산재해 있거나 때로는 기업 구성원의 메모나 머릿속에 존재하기도 한다[1].

BRE(Business Rule Engine)는 기업이 가지고 있는 비즈니스 관련 지식을 체계적이며 집중적으로 관리하도록 해주는 기업형 솔루션이다. BRE를 이용하면 기업의 경영자, 분석가, 업무 담당자 등이 IT 전문가의 도움 없이도 비즈니스 관련 지식을 직접 저장, 수정, 삭제, 활용할 수 있게 된다. BRE를 활용하면 기업은 비즈니스 관련 지식과 같은 눈에 보이지 않는 지적 자산에 대한 관리 수준을 대폭 향상시킬 수 있으며 경쟁 기업에 대한 전략적 우위도 확보할 수 있을 것으로 기대된다.

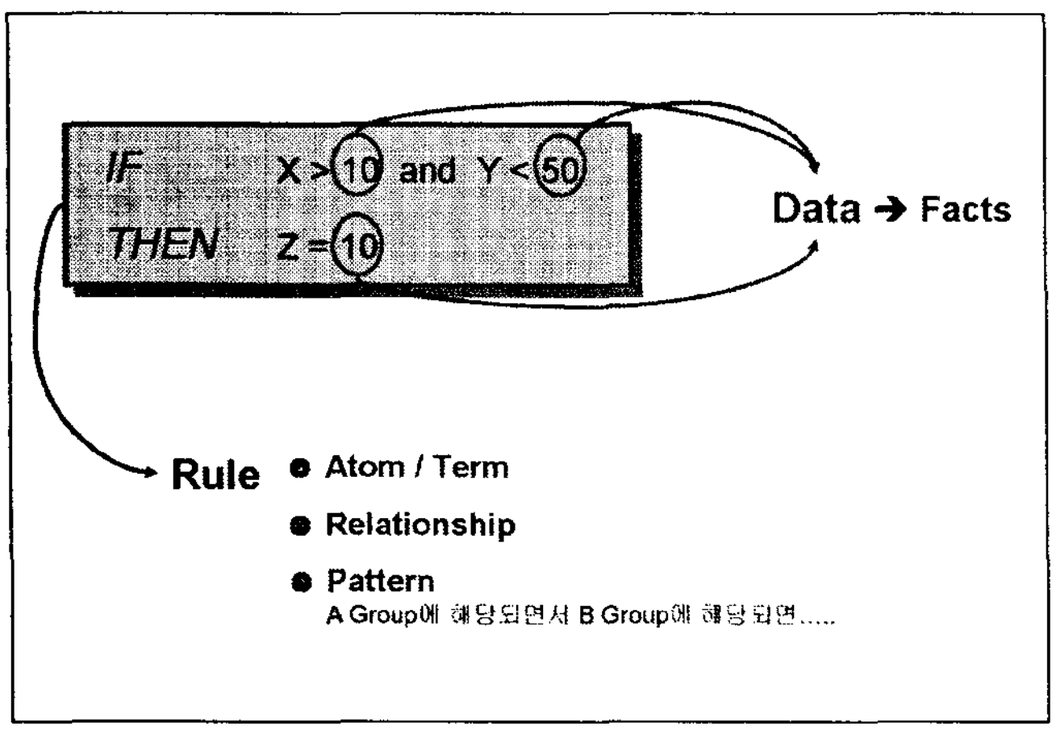
룰 웨어하우스 또는 리포지토리에 대한 개념 설명 추가

본 논문에서는 금융보험사의 룰 웨어하우스(Rule Warehouse) 구축 사례를 통하여 전사 수준의 비즈니스 룰 관리 아키텍처의 다양한 구성 방법 및 각 방법이 지닌 장단점에 대해 분석한다. 본 연구의 결과를 토대로 향후 전사 수준의 통합 비즈니스 룰 관리 시스템 구축 방안에 대한 보다 다양한 연구가 활성화되기를 기대한다.

2. 룰 웨어하우스(Rule Warehouse)

2.1 비즈니스 룰(Business Rule)

비즈니스 룰이란 기업 경영 활동 중 발생하는 의사결정 시점에서 사용될 수 있는 규칙과 업무수칙, 수행방안, 규정, 정책 등 업무 수행 활동들을 지도, 감독 및 관리하는 제반 규칙을 의미한다. 이를 보다 구체적으로 정의하자면 비즈니스를 정의하고 이를 통제하는 문장(statements that defines or constrains some aspect of the business)으로, 이는 비즈니스 구조를 규정하거나(asserting business structure), 비즈니스 행위를 제어하고자 하는 구문(controlling the behavior of the business)을 의미한다[2]. 비즈니스 구조를 규정한다는 것은 비즈니스에 관한 사실(facts)이나 용어(terms)를 정의하는 것이고 행위를 제어한다는 것은 비즈니스 행위에 대한 규칙(rules)을 정의하는 것이다. 비즈니스 룰은 일반적으로 <IF 조건 THEN 결과> 형태로 표현할 수 있으며, 기업 내에 존재하는 업무 규칙이나 절차, 규정, 정책, 담당자의 지식 및 노하우 등이 해당된다. [그림 1]은 일반적인 비즈니스 룰의 구조를 예시한 것이다.



[그림 1] 비즈니스 룰 구조

IF (금월실적 > 전월실적 * 1.5)

THEN (실적평가 = 실적평가 + 1)

위의 예는 "금월실적을 전월실적 보다 150% 이상 초과 달성하는 경우 실적평가에 가산점을 준다"는 비즈니스 룰이다. 이는 금월실적, 전월실적, 실적평가 등 비즈니스 도메인 상에 존재하는 개념을 예시하며, 이들 개념이 모여 해당 비즈니스의 구조를 정의한다. 또한 이 비즈니스 룰은 실적 평가 및 관리에 관한 정책 규정으로서 비즈니스 행위를 제어한다.

비즈니스 룰은 대부분의 경우 다음과 같은 특징을 갖는다. 첫째, 비즈니스 룰은 비정형적인(unstructured) 경우가 많다. 비즈니스 룰은 실제 경험을 통해 획득된 담당자의 업무 노하우를 포함하기도 하는데, 업무 처리 지식은 순서가 불규칙하고 비정형 구조를 갖는 것이 일반적이다. 둘째, 비즈니스 룰은 변경이 잦다(frequently changing). 비즈니스 룰은 기업 관련 운영 정책이나 마케팅 캠페인, 서비스 관리 규정 등 기업 내 의사결정을 위한 모든 지식을 포괄하는데 이는 시장 상황이나 경쟁사 동향, 계절적 요인 등 다양한 환경 변화에 따라 수시로 변경되는 특성을 지닌다. 마지막으로 비즈니스 룰은 다양한 소스에 분산되어 저장되는 경우가 많다. 비즈니스 룰은 데이터베이스 내에 데이터로 저장되거나 업무 지원 시스템의 처리 로직, 사용자 인터페이스 또는 인적 자원 등 다양한 형태로 기업 내부에 분산되어 존재한다.

비즈니스 룰은 자주 변경되므로 기업이 경쟁력을 유지하기 위해서는 신속하고 정확한 변경 관리가 필요하지만 비즈니스 룰의 비정형적 형태와 여러 곳에 분산되어 존재한다는 특징 때문에 유지 보수가 어렵다. 즉, 변경이 필요한 비즈니스 룰의 위치를 찾기 힘들 뿐 아니라, 만일 비즈니스 룰이 여러 모듈에 분산되어 존재하는 형태로 각 모듈간에 상호 의존적 관계가 존재하는 경우 룰의 추가, 수정, 삭제 행위에 따른 전체 비즈니스 룰의 일관성(consistency) 및 무결성(integrity) 유지가 쉽지 않다.

만일 업무 시스템을 기존 범용 프로그램 방식을 통해 구축된 일반적인 시스템 구조로 관리하는 경우에는 업무 절차를 규정하는 비즈니스 로직과 프로그램을 제어하는 제어 로직이 하나의 모듈에 혼재하게 된다. 이러한 경우 해당 업무 절차가 복잡해지거나 자주 변경되는 경우 개발 공수가 증대되고, 변경된 업무 시스템의 품질 관리가 용이치 않으며, 개발자의 작업 스킬에 따라 업무 시스템의 품질이 크게 좌우되고, 전체적인 운영 비용이 증대되는 등의 문제점이 발생한다. 또한, 만일 비즈니스 룰을 프로그램 방식으로 어느 정도 자동화하는 형태로 관리하는 경우에는 실제 업무를 담당하는 담당자가 직접 비즈니스 룰을 변경하기가 어렵고, 비즈니스 룰을 변경할 때마다 시스템 코드를 이해하고 수정할 수 있는 IT 전문가의 도움을 받아 시스템을 정지하고 비즈니스 로직을 정비해야 하는 등의 비즈니스 운영 상의 문제점이 발생한다[3].

BRMS(Business Rule Management System)란 이러한 문제점을 해결하기 위한 시스템[4]으로, BRMS를 도입하면 기업은 비즈니스 로직, 데이터베이스, 사용자 인터페이스 등에 분산되어 존재하는 비즈니스 룰을 통합적으로 관리할 수 있으며 이를 통해 룰 관리의 효율성이 증가하고 일관성과 무결성 유지가 보장된다.

BRMS를 위해서 과거에는 주로 DBMS(Database Management System)을 활용하였다. 데이터베이스의 고유 기능인 데이터 관리 기능을 이용하여 비즈니스 룰을 추가, 수정, 삭제 조회하였으며 데이터베이스의 제어 기능인 일관성 무결정 유지 기능을 활용해 룰 베이스를 제어하였다. 하지만, 최근에는 비즈니스 룰만의 관리를 위한 전문적인 솔루션으로 BRE가 소개되어 사용되기 시작하였다. BRE 는 BRMS의 룰 관리 기능 이외에 추론 기능이 추가되었기 때문에 선언적인 지식인 비즈니스 룰만 시스템에 입력해 놓으면 사용자는 절차적 지식의 코딩으로부터 자유로울 수 있게 된다[4].

2.2 BRE(Business Rule Engine)

BRE는 룰 관리 기능 뿐만 아니라 인공지능에서 개발된 추론 기능도 제공하는 시스템으로, 이것은 전신은 규칙 기반 전문가 시스템(Rule-based Expert System)이다. BRE에서 비즈니스 룰은 <If 조건 Then 결과> 또는 의사결정 테이블, 스코어카드 등의 형태로 표현된다. BRE는 룰베이스 모듈(Rulebase Module), 팩트베이스 모듈(Factbase Module), 설명 모듈(Explanation Generator)등으로 구성된다([그림3]). 룰베이스는 기업 정책이나 실무자 지식 등 기업이 가지고 있는 비즈니스 룰을 통합적으로 보관 관리하는 부분이며, 팩트베이스는 기업, 고객, 상품, 거래 등에 관한 정보를 포함하는 워킹 메모리(Working Memory) 역학을 한다. 추론엔진은 전방향 추론, 후방향 추론 등의 기법이 구현된 모듈이며, 설명 모듈은 추론 결과가 어떤 과정을 거쳐 나오게 되었는지 근거를 사용자가 이해하기 쉽게 제시해 주는 모듈이다.

BRE는 추론엔진을 이미 갖추고 있기 때문에, 개발자로 하여금 절차적 지식의 구현 즉, 프로그래밍 과정으로부터 자유롭게 한다. BRE를 도입하면 일반적인 시스템 개발에 필요했던 절차적 지식의 코딩 부분 중 상당 부분이 생략되며, 특히 비즈니스 룰은 거의 선언적 지식으로 입력된다. 따라서 비즈니스 룰의 관리에는 프로그래밍 기술이나 경험을 전혀 요구하지 않기 때문에 일반 실무자들도 IT 전문가의 도움 없이 쉽게 비즈니스 룰을 입력, 수정, 조회, 삭제할 수 있다. 따라서 시스템의 유지보수 시간을 줄이고, 비용을 절감하는데 매우 효과적이다[1].

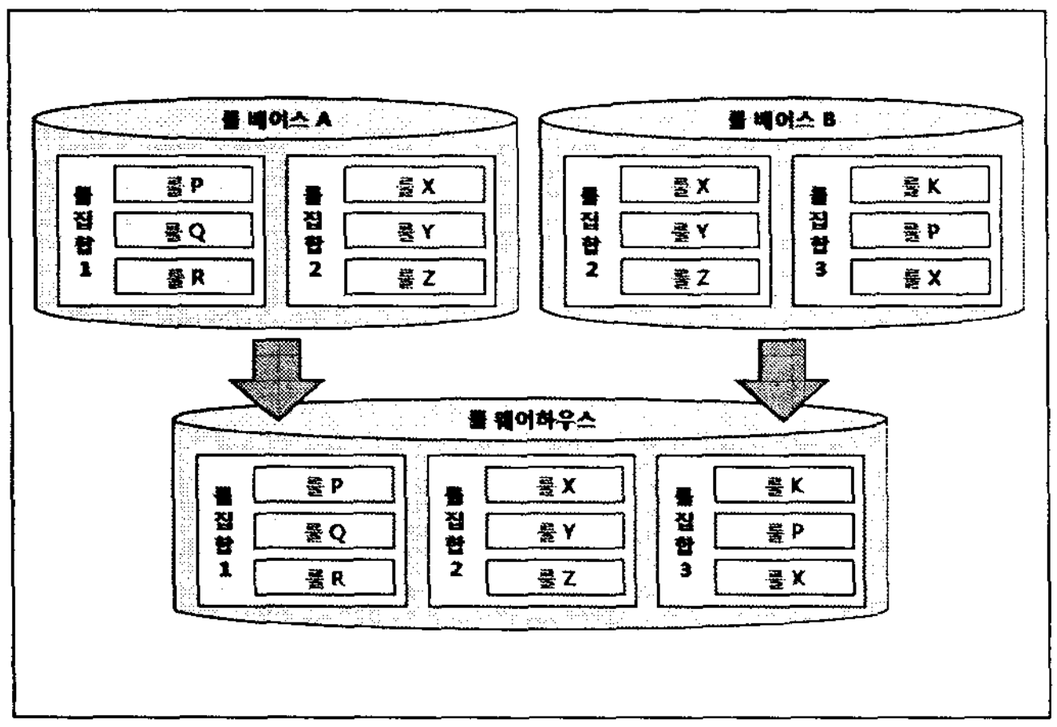
BRE는 경영자나 실무자들이 가지고 있는 비구조적인 지식을 시스템에 표현하고 이를 근거로 추론을 수행할 수 있기 때문에, BRE를 도입하면 기업은 지식관리의 수준을 한층 높일 수 있게 된다. BRE는 기업의 전략이나 전술상의 변화를 손쉽게 시스템에 반영할 수 있도록 하기 때문에 기업의 시장 대응 능력이 향상된다. BRE는 새로운 상품이나 서비스를 개발하기 위한 유연하고 신속한 IT 인프라를 제공하기 때문에 시장화 시간(time-to-market)을 단축할 수 있다. 또한 BRE의 태생적 기능이 의사결정 지원이기 때문에 경영자나 실무자가 일관성 있고 신속한 경영 의사결정을 효과적으로 내릴 수 있도록 한다. 시스템의 유지보수 측면에서도 BRE는 매우 유용하다. BRE로 구현된 경영 정책이나 프로세스는 이를 수정하는데 정보 기술자나 개발자의 도움을 필요로 하지 않는다. 실무자가 변화 내용을 직접 시스템에 반영할 수 있다. 따라서 고객 중심의 시장 환경에서 고객의 다양하고 역동적인 요구에 신속하게 반응할 수 있다[5].

2.3 룰 웨어하우스(Rule Warehouse)

룰 웨어하우스는 개념적으로는 데이터 웨어하우스와 같은 개념이다. 기업 내에 산재된 데이터베이스들을 분석하여 통합 데이터 웨어하우스를 구축하는 것과 같이, 룰 웨어하우스는 기업 내에 다양한 시스템에서 사용되고 있는 룰 베이스를 하나의 룰 레파지토리, 즉 룰 웨어하우스에 저장하고자 하는 것이다. 데이터 웨어하우스를 구축하면서 중복된 데이터를 제거하는 것과 같이, 룰 웨어하우스에서도 중복되는 룰을 제거하고, 다양한 어플리케이션에서 이를 손쉽게 사용할 수 있도록 하는 작업이 필요하다. 그러나, 데이터와 달리, 비즈니스 로직을 표현하는 룰은 동일한 룰이 서로 다른 업무용 룰 베이스에서 다른 목적으로 사용될 수 있으며, 같은 목적으로 사용되나, 서로 다른 객체모형을 사용함에 따라 다른 객체명을 포함하고 있어 다른 룰로 인식되는 경우도 존재한다. 따라서, 로직을 포함하는 룰을 단순 데이터로 취급하여 룰 웨어하우스를 구축하는 것은 불가능하다고 할 수 있다.

룰은 데이터가 아닌 비즈니스 로직이라는 측면에서 룰 웨어하우스를 비즈니스 로직 웨어하우스라고 할 수 있다. 즉, 다양한 비즈니스 로직을 통합한 구조를 지니게 된다. 비즈니스 로직은 개별 룰로 표현되나, 실제 업무에서 이야기하는 비즈니스 로직은 하나의 룰이 아닌 여러 개의 룰이 모여 표현된다. 따라서, 각각의 룰 단위로 구성하는 것 보다는 하나의 비즈니스 로직을 표현한 하나의 룰 그룹, 즉 룰 집합(Rule Set)단위를 최소의 구분단위로 보는 것이 타당하다. 서로 다른 비즈니스 로직에 의하여 특정 업무를 수행하는 룰 집합들의 집합을 룰 웨어하우스로 정의하고, 개별 룰의 중복여부는 고려하지 않는다. 현실적인 측면에서도, 개별 룰 단위의 룰 웨어하우스 구축은 보다 복잡한 관리구조를 요구하게 되어, 사용이 어렵게 되어, 그 효과가 크게 줄어들 것으로 판단되나, 보다 구체적인 연구는, 보다 깊은 연구가 필요하다.

[그림 2]는 룰 집합을 기준으로 구성하는 룰 웨어하우스의 개념적 설명을 보여준다. 룰 집합 1과 룰 집합 3은 룰 A를 공통으로 지니고 있다. 룰 단위로 룰 웨어하우스를 구축한다면, 이는 중복되는 룰로 보아야 하겠지만, 여기서는 룰 집합을 룰 웨어하우스를 구성하는 최소의 비즈니스 로직 단위로 하고 있어, 이러한 중복이 허용된다. 따라서 일반적으로 룰 집합 내에서 허용되지 않는 룰들의 중복(Redundancy) 문제가 여기에서는 동일 비즈니스 로직을 표현하는 룰 집합의 중복여부로 판단의 기준이 변화하여야 한다.



[그림 2] 룰 웨어하우스 개념

룰 웨어하우스의 목적은 산재된 룰 베이스, 즉 비즈니스 로직을 하나로 모으는 것이다. 이는 기업 내의 다양한 비즈니스 로직을 중앙집중식으로 관리한다는 것을 의미한다. 정책의 변경이 발생하는 경우, 관련 프로그램의 코드를 수정하다 보면, 간혹 실수로 로직이 잘못 구현된 프로그램이 존재할 수 있다. 이는 그 프로그램이 룰 베이스로 로직을 관리하는 경우에도 발생할 수 있다. 그러나, 모든 로직을 일괄관리하고, 관련 프로그램들은 동일한 로직을 참조함으로써, 전사적 정책의 일관성을 확보할 수 있게 된다. 또한 신속하게 변화에 대처할 수 있게 된다. 이는 Agile Computing이 중요한 현재의 IT환경에서 매우 중요한 의미를 지닌다고 할 것이다.

3. 룰 웨어하우스의 관리(Rule Warehouse Management)

룰 웨어하우스는 개념적으로 간단한 내용이나, 실제 이의 운영과 관리는 기술관점과 관리관점에서 새로이 고려하여야 할 문제점들을 야기한다. 기술적 관점에서 고려하여야 하는 문제점들로는 다음과 같은 것들이 있다.

- 서로 다른 객체모형을 사용하는 룰 집합들의 통합
- 룰에서 호출되는 외부함수들의 통합
- 부분적으로 중복된 룰 템플릿들의 통합
- 룰 집합에서 실행 중에 사용하는 관리자원의 처리

이외에도 이미 구축된 룰 집합들을 통합하기 위하여는 구축된 룰 집합의 완성도도 매우 중요한 문제가 된다. 룰에 대한 이해가 부족한 프로그래머가 구축한 룰의 경우, 프로그램 코드와 비슷한 구조와 강제적인 룰 처리흐름을 갖게 되는 경우가 많아, 룰은 물론, 연결되는 함수, 자원들이 상호 의존성을 지니는 경우가 많아, 이들의 분리 및 타 룰 집합들과의 통합이 어려운 경우도 자주 발생된다. 이러한 문제점들을 피하려면, 처음 룰 어플리케이션을 개발하는 초기부터 룰 웨어하우스를 고려한 설계가 이루어져야 한다. 그렇지 않은 경우, 개별 관리는 물론, 룰 웨어하우스로의 통합은 많은 비용을 필요로 하게 될 것이다.

기술적 관점 못지않게 관리관점도 중요하다. 룰 웨어하우스의 관리는 중앙집중 방식으로 이루어지지만, 실제 단위 업무와 그 비즈니스 로직, 즉 단위 룰 집합의 관리는 해당부서에서 하여야 하는 경우가 더 많기 때문이다. 그림 2에서 룰 베이스 A는 A부서에서 관리하던 룰 베이스이고, 룰 베이스 B는 B부서에서 관리하던 룰 베이스라고 하면, 룰 집합 2는 A부서와 B부서에서 공동으로 관리되는 영역에 해당된다. 이는 A부서나 B부서 모두 단독으로 임의 수정을 허용해서는 안 된다는 것을 의미한다.

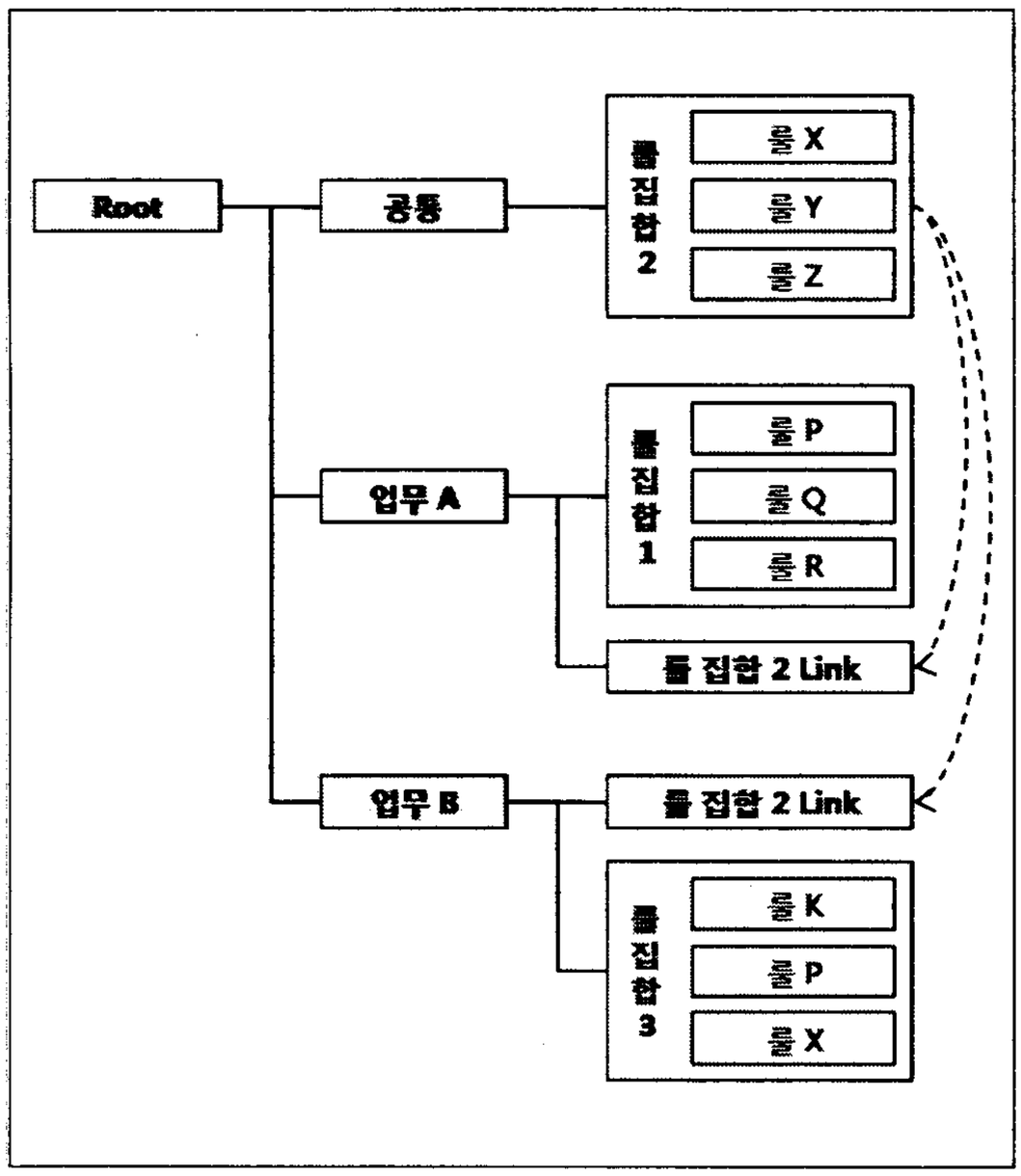
* 룰 웨어하우스 접속권한 및 접속이력 관리
* 직급, 직위, 부서, 등의 조건에 따른 조회 및 수정 범위
* 룰 웨어하우스의 각 단위 구성원들의 관리자에 대한 책임 및 역할
* 새로운 룰 집합과 같은 신규 단위구성원 등록에 대한 관리
* 복수 사용자가 동일한 룰을 동시 수정하고자 하는 경우의 처리 (Dead Lock과 유사한 문제)
* 룰 변경에 대한 검증 및 승인 프로세스
* 개발서버, 테스트서버, 운영서버에 설치된 룰 웨어하우스 사이의 배포관리
* 검증 및 감사 방안

관리측면에서는 이외에도 여러 문제가 발생될 수 있다. 이러한 문제들은 기술적 해결보다는 관리적 해결이 우선적으로 요구된다. 즉, 룰 웨어하우스의 구축은 룰 관리정책의 수립 등 룰 거버넌스(Rule

Governance)에 대한 고려가 반드시 이루어져야 한다. 단일 룰 베이스의 관리에 대해서도 관리정책이 필요하지만, 룰 웨어하우스는 지니고 있는 비즈니스 로직의 다양성, 관련부서 및 담당자의 수, 관리 프로세스의 복잡함, 등이 단일 룰 베이스의 관리와는 비교가 되지 않게 많은 고려요소들을 지니고 있다. 따라서, 룰 웨어하우스의 구축은 기술적 측면의 개발과 동시에 관리적 측면에서의 컨설팅작업이 반드시 수반되어야 한다.

4. 룰 웨어하우스 구조 (Rule Warehouse Architecture)

룰 웨어하우스의 기본적 구조는 [그림 3]에서와 같이, Root 폴더를 정점으로 하는 트리구조로 구성한다. 이와 같은 트리구조는 사용자의 이해가 쉬워 관리가 편리한 장점을 지니고 있다.



[그림 3] 룰 웨어하우스의 구조

룰 웨어하우스의 트리구조는 Root에서 시작하여, 필요한 폴더를 계속 추가해 나아갈 수 있다. 그리고 트리의 Terminal Node에는 룰 집합과 같은 실제 구성원이 위치하게 된다.

[그림 3]에서 "룰 집합 2 Link"는 공통 폴더에 위치한 룰집합2와 업무 A, 혹은 B와 연결시켜준다. 즉, 업무 A에서는 룰 집합 1과 룰 집합 2가 함께 사용됨을 의미한다. 이와 같은 룰집합 2의 사용은 개발된 룰 집합에 대한 재사용성을 높여준다. 또한, 한번의 수정으로 두 개의 업무를 수정하는 효과를 가져온다. 또한, 룰집합 1과 룰집합 3을 묶어서 새로운 업무 C를 생성할 수도 있다. 이는 서비스기반 아키텍처(SOA: Service Oriented Architecture)에서 단위 서비스의 오케스트레이션(Orchestration)을 통하여 하나의 실용 서비스를 제공하는 것과 같이, 이미 등록되어 있는 룰 집합들을 조합하여 새로운 업무처리 룰 서비스를 정의할 수 있음을 의미한다. 이것 역시 Agile Computing을 위하여 필요한 기능이라고 할 수 있다.

5. 결론

룰 웨어하우스는 룰의 사용범위가 점차 확대되는 상황에서 그 필요성이 점차 대두되고 있는 기술이다. 이의 구축은 기술적 측면에서 다양한 업무를 포괄적으로 처리할 수 있는 구조와 환경을 제공하여야 하며, 관리적인 측면에서도 관리정책과 프로세스를 정립할 것을 요구한다. 룰 웨어하우스는 이와 같은 많은 복잡한 문제점들을 지니고 있으나, 현재 IT가 지향하고 있는 Agile Computing의 실현을 위한 최적의 구조를 제공하며, 중앙집중형의 일괄관리를 가능하게 한다. 따라서, 점차 이의 적용이 확산될 것으로 예상된다. 이와 함께, 최적의 룰 웨어하우스 구축을 위한 연구도 함께 진행될 것이다.

References ← 12pt, Times bold

[1] BRCommunity. (2005). "A Brief History of the Business Rule Approach," *Business Rules Journal, Vol. 6, No. 1.*

[2] the Business Rules Group. (2000). "Defining Business Rules~What Are They Really?," *the Business Rules Group Final Report revision 1.3,* http://www.BusinessRulesGroup.org.

[3] 박규호. (2001). "Rulebase Management System(Brokat Advisor)을 이용한 사례 소개," *한국지능정보시스템학회: 학술대회지 , 한국지능정보시스템학회 2001년도 춘계정기학술대회 , pp.111-135 ..*

[4] Ross, R. (2001). "What is Rule Management About?," *Business Rules Journal, 2004..*

[5] Lin, N. (2002). "Alternatives for Rule-based Application Development," *Business Rule Journal, 2002.*