IE Interfaces
Vol. 12, No. 1, pp. 143-149, March 1999.

통합 물류 시스템 구축을 위한 워크플로 프레임워크

서용원 · 성제훈 · 함주호

서울대학교 산업공학과

A Workflow Framework for the Integrated Logistics Systems

Yong-Won Seo, Che-Hoon Sung, Ju-Ho Hahm

In this paper, we propose a workflow framework for the integrated logistics systems. We modelled the logistics activities into workflows, which allow clear specification and flexible modification of business processes. To specify workflows, we used petri-net that is extended to handle with information flows. A framework to implement the integrated logistics system using workflow management technology is proposed. Since logistics is closely related with other parts of enterprise such as manufacturing and accounting, the framework could capture a rich features required for enterprise-wide systems.

1. 서 론

거품경제의 붕괴와 더불어 시장개방압력으로 인해 기업의 생산성 향상은 구호를 넘어선 절실한 명제로서 자리잡게 되었다. 기업들은 한계사업 포기, 인원감축 등의 감량경영을 통해 생산성 향상을 꾀하고 있으나, 이는 일시적인 방법일 뿐 근본적인 생산성 향상은 기업 시스템 차원의 재설계를 통해서만이 가능하다는 사실을 인지해야 할 것이다.

선진 국가들의 경우, 기업 시스템 재설계를 위해 혁신적 업무 프로세스 재설계 또는 BPR(Business Process Reengineering)라는 방법론을 채택하였다. 이는 생산성의 점진적인 개선에는 한계가 있음을 파악하여, 기업 시스템을 프로세스 단위로 재해석하고 이들 업무 프로세스를 기초부터 재설계하고자 하는 방법론이다[5]. 그러나 재설계된 프로세스가 정착되지 못하고 과거로 회귀하는 문제가 종종 발생하여, 재설계된 프로세스를 정보 시스템으로 지원하기 위한 요구가 대두되었으며, 이에 대한 해답으로서 워크플로 관리시스템에 대한 관심이 고조되었다. 그러나 각 기업의 프로세스에 맞는 정보시스템을 구축하는 것은 상당한 시간과 비용을 요구하는 문제점이 있다.

다른 접근은, 일반적으로 뛰어난 업무관습(best practice)을 내장한 전사적 정보시스템인 ERP(Enterprise Resource Planning)를 도입하는 것이다. 이 경우 기존의 프로세스를 ERP상에 고착화된 업무 프로세스에 따라 변경해야 하나 현실적으로는 문화 또는 경영환경의 차이로 인해 ERP상의 프로세스를 그대로 따르는 것이 곤란한 경우가 많다는 문제가 있다.

대부분의 ERP 시스템은 내장된 프로세스의 유연한 변경을 염두에 두고 설계된 것이 아니므로, ERP 패키지를 실제에 적용하는 과정에서 많은 비용이 발생하고 신뢰성이 감소하는 문제점이 있었다.

두 접근방법을 절충하는 데에 워크플로 기반의 접근방법이 사용될 수 있다. 워크플로 관리시스템은 주로 BPR을 통해 재설계된 프로세스의 정보시스템 지원방법으로서 주목되어 왔다. 워크플로 관리시스템은 재설계된 업무 프로세스의 정착을 쉽게 할 뿐만 아니라, 부서간, 조직간을 넘나드는 프로세스상에서의 정보흐름을 자동화하여 프로세스의 효율을 향상시킬 수 있다. 뿐만 아니라, 워크플로 기반으로 설계된 정보시스템은 작업과 프로세스 수준이 명확히 구분되어, 프로세스의 변경이 요구될 경우 각각의 작업은 그대로 두고 워크플로 명세

(Workflow Specification)만 변경하면 된다는 장점이 있다. 최근 많은 워크플로 관리시스템(Workflow Management System)들이 상용화되는 등, 워크플로 관련 시장 규모도 확장 일변도에 있는 추세이다[19].

본 연구에서는 기업의 물류 시스템을 대상으로, 워크플로 기반으로 물류 시스템을 구축하는 프레임워크를 제시하고자 한다.

널리 알려진 바와 같이, 물류부문은 기업비용의 상당부분을 차지하면서 아직 개선 가능성이 많이 남아 있는 부문이다. 또한, 물류부문은 생산, 회계 등 기업의 다른 부문과 밀접하게 연관되어 향후 전사적 규모의 시스템에 요구되는 다양한 특성들을 제시할 수 있을 것으로 기대된다.

2. 관련연구 현황

업무절차나 프로세스를 묘사를 통한 워크플로 모형은 단순한 묘사가 아닌, 업무절차를 관리하기 위한 시스템으로 발전하였다. 이러한 워크플로 관리시스템은 이미지 프로세싱, 문서관리, 전자우편, 그룹웨어, 트랜잭션 관리, 프로젝트 지원, BPR 등의 다양한 분야에서 발전되어 왔다[20]. 특수목적의 시스템에서 시작하여 범용 워크플로 관리시스템으로 진화하는 과정에서 워크플로의 모형을 체계적으로 나타내기 위한 워크플로 명세 언어(workflow specification language)의 필요성이 부각되었다.

워크플로 연구의 초창기에는 고전적 TM(Transaction Model)의 확장된 형태인 ATM(Advanced Transaction Model)과 밀접한 관련을 가지며 연구가 진행되었다[12]. 따라서 ATM이 일반적인 워크플로의 모형화에 사용될 수 있을 것으로 기대되었으나, 다양한 ATM 모형이 제시되었음에도 불구하고 실용적인 구현이 이루어진 예는 드물어[14], 이러한 관점의 워크플로 시스템 접근 실용성에 의문을 갖게 하였다.

이후의 많은 워크플로 관리시스템들에서는 대개 자체적인 워크플로 명세 언어를 제공했으며, 이들은 대개 ECA(Event-Condition-Action) 규칙에 기초를 두고 있다[16]. 그러나 많은 시스템들에서 표현능력의 한계와 의미상 모호를 보이는 문제점이 있었다[8].

이러한 문제의식에 입각하여, 워크플로의 명세에 페트리넷을 사용할 것을 제안하는 움직임이 나타나게 되었고, [6] ~ [8]에서는 페트리넷의 엄밀한 의미상의 정확성과 표현능력, 그리고 몇 가지 특성들이 워크플로의 명세에 적합하다는 점을 지적하였다.

한편으로는 워크플로 명세에 의해 주어진 워크플로를 실시간에 관리하기 위한 워크플로 기동 서비스(workflow enactment service)[20]의 구조에 관한 연구가 활발히 진행되었으며, 최근에는 분산객체시스템을 워크플로 관리시스템으로 통합하기 위한 구조가 제시되고 있는 추세이다[1][18].

3. 페트리넷을 이용한 워크플로

워크플로의 명세에 페트리넷(petrinet)을 이용하는 방안의 장점에 대해 앞서 간략히 언급한 바 있다. 그러나 페트리넷은 주로 제조 프로세스의 용도에 적합하게 설계되었으므로, 워크플로의 명세에 사용하기 위해서는 몇 가지 고려가 이루어져야 한다.

첫째로, 페트리넷은 작업의 흐름을 명확히 지시할 뿐, 자료(data)의 흐름을 정의할 수 있는 방법이 없다. 반면, 워크플로의 명세를 위해서는 자료의 흐름을 명세하는 것이 대단히 중요하므로, 페트리넷상에서 자료의 흐름을 정의할 수 있는 방법을 설계해야 한다. 페트리넷의 구성요소인 플레이스(place)와 트랜지션(transition) 중 작업(task)의 수행을 나타내는 것은 트랜지션에 해당한다[2]. 작업의 수행을 위해서는 입력자료가 필요하고, 수행이 끝나면 새로운 출력자료가 발생하므로, 트랜지션상에는 해당작업의 수행에 필요한 입력자료와 출력자료가 정의되어야 한다. 이 때, 플레이스는 자료를 보관하는 보관장소(container)로서의 역할을 수행하게 된다. 각 트랜지션은 앞선 플레이스들에서 필요로 하는 자료를 가져온 후, 해당 작업을 수행하고, 다음에 연결된 플레이스들에 출력자료를 보관한다. 페트리넷에서 작업의 수행은 반드시 토큰(token)을 필요로 하므로, 자료와 토큰은 동시에 움직이게 된다.

워크플로 정의의 편의성과 이미 정의된 워크플로의 재사용을 위해서는 중첩 워크플로(nested workflow)를 가능하도록 해야 한다.

이 경우, 하나의 작업은 하나의 클라이언트(client) 또는 데이터베이스 트랜잭션 등의 원자적인 활동(atomic activity)과 연관될 수도 있으나, 그 자체로서 하위 워크플로(sub-workflow)를 가질 수도 있다. 따라서, 각 트랜지션은 SQL(Structured Query Language)문이나 수행할 클라이언트의 이름을 갖는 이외에도, 다른 워크플로의 이름을 가질 수도 있다. 이 때, 하위 워크플로의 시작 플레이스는 상위 워크플로의 해당 트랜지션에 주어진 입력자료를 가지게 되고, 하위 워크플로의 종료 플레이스에 보관되는 자료가 상위 워크플로의 해당 트랜지션의 출력자료가 된다.

하나의 플레이스에서 둘 이상의 트랜지션으로 분기되는 경우에는 토큰이 어디로 향할 것인지를 정해 주어야 한다. 따라서 분기가 일어나는 플레이스는 현재 보관중인 자료에 대한 조건문을 가져야 하며, 이 조건에 따라 분기할 수 있도록 해야 한다. 다음의 <표 1>에 워크플로 명세를 위해 플레이스, 트랜지션들이 가져야 할 속성을 정리하였다.

표 1. 플레이스와 트랜지션의 속성

플레이스의 속성	트랜지션의 속성
• 이전, 이후의 트랜지션 • 보관 자료 • 분기를 위한 조건문	• 이전 이후의 플레이스 • SQL 트랜잭션, 작업을 수행하는 클라이언트의 이름, 또는 하위 워크플로의 이름 • 입력자료 • 출력자료

4. 페트리넷기반 워크플로 관리시스템 구현시의 고려사항

본 절에서는 페트리넷을 워크플로 명세 언어로 사용하는 경우의 워크플로 관리시스템 구현시의 기술적 고려사항에 대해 다룬다. 일반적인 워크플로 관리시스템은 크게 프로세스 정의 부분(process definition part), 워크플로 기동 서비스(workflow enactment service), 인간/IT자원과의 상호작용(interactions between human/IT resources)으로 구성된다[20].

4.1 프로세스 정의 부분

페트리넷에 의해 워크플로를 명세할 수 있도록 하기 위해, 프로세스 정의 모듈은 페트리넷 편집기를 내장해야 한다. 여기서 편집된 페트리넷은 일반적인 페트리넷에 관련된 정보들 이외에도 앞의 <표 1>에서 설명된 속성들을 포함하고 있어야 하므로, 이의 표현 및 보관을 위한 적절한 표현방법은 객체지향개념을 적용하는 것이다. 즉, 플레이스, 트랜지션 클래스를 구성하고 <표 1>에 표현된 속성들을 이들 클래스의 속성으로 갖도록 하는 것이다. 이렇게 정의된 객체들의 저장을 위해서는 자바(Java)의 객체 직렬화(Object Serialization)등의 방법을 통해 바이트 배열화하여 파일 시스템을 이용할 수도 있고[10], 이미 상용화된 객체지향 데이터베이스를 이용할 수도 있을 것이다.

4.2 워크플로 엔진

워크플로 엔진은 워크플로 기동 서비스의 핵심을 구성한다. 이는 정의된 페트리넷을 읽어들여 해석하고, 이에 따라 관련 클라이언트들을 실시간에 제어해야 한다.

고려사항 중의 하나는 다중 워크플로 인스턴스(multiple instance)이다. 일반적으로 하나의 워크플로 인스턴스가 진행중일 때라도 동일 워크플로의 다른 인스턴스의 발생을 가능하도록 하는 것이 바람직하다. 워크플로의 정의 자체를 다중 인스턴스를 고려한 것으로 하는 것은 번거로운 일이므로, 워크플로의 정의는 단일 인스턴스에 대한 것으로 주어진다고 가정한다. 단일 인스턴스의 경우에는 주어진 토큰을 처리하고 토큰의 위치에서의 작업을 수행, 자료를 플레이스에 보관하는 것으로 충분하지만, 다중 인스턴스의 경우 각 인스턴스들에 대한 토큰들이 여러 개 존재하므로, 각 토큰은 고유의 인스턴스 ID(instance ID)를 가진다. 자료들도 각 인스턴스에 관련되므로, 자료에도 동일한 ID가 첨부된다. 존재할 수 있는 인스턴스의 개수는, 시작 플레이스(initial place)가 발생시킬 수 있는 토큰의 개수와 동일하게 되므로, 인스턴스 개수에 제한을 두지 않기 위해서는 편의상 시작 플레이스가 무한 개의 토큰을 가진다고 생각하는 것이 편리하다. 반면, 종료 플레이스에 토큰이 도달하면 해당 인스턴스의 종료를 알리고, 토큰을 폐기처분하면 된다. 따라서, 플레이스에는 일반적인 플레이스와 무한 개의 토큰을 가지는 플레이스, 그리고 토큰을 폐기하는 플레이스가 존재한다.

또한, 워크플로 엔진은 실시간 제어에 필요한 정보들을 관리해야 한다. 워크플로가 기동되는 과정에서 필요한 정보들은 각 플레이스에 보관되는 자료들, 각 트랜지션의 입/출력 자료 등과 현재 토큰의 분포상태 등이다. 일반적으로 다중 인스턴스를 가정하므로, 플레이스와 트랜지션들은 복수의 인스턴스에 대한 자료와 토큰들을 대기열(queue)의 형태로 저장하도록 한다.

4.3 엔진과 클라이언트의 네트워크 연결

트랜지션이 수행될 때는 해당작업을 수행하는 클라이언트에 작업과 관련자료를 넘겨주어야 한다. 클라이언트는 일반적으로 네트워크상에 분산되어 있으며, 광범위한 접속지역과 적은 네트워크 비용을 위해 네트워크는 TCP/IP 프로토콜을 사용하는 것으로 가정한다.

이 때, 워크플로 엔진과 관련 클라이언트들을 연결하는 방법에는 TCP/IP 소켓(Socket)을 이용하는 방법에서부터 RMI

(Remote Method Invocation), CORBA(Common Object Request Broker Architecture) 등에 이르기까지 여러 가지가 가능하나, 클라이언트 구현 환경에 대한 독립성과 유지 보수상 많은 이점을 제공하는 CORBA 환경을 이용하는 것이 바람직할 것이다[2][18]. 이 경우 엔진의 IP 주소(IP address)와 클라이언트가 작업을 요청하거나 완료한 작업을 넘겨줄 방법(method) 등에 대한 명세만 가지면 클라이언트의 통합이 가능하므로, 클라이언트의 개발과 기존 응용프로그램의 통합을 용이하게 할 수 있다. 사용자 교육비용과 유지보수비용의 절감을 위해 클라이언트는 인트라넷 개념을 적용, 웹 브라우저 상에서 수행되도록 하는 것이 좋을 것이다.

5. 물류 시스템의 통합

앞에서 페트리넷을 통해 워크플로를 정의하는 방안을 언급하였다. 이러한 방법에 의해, 물류 시스템이 어떻게 워크플로로 나타내어질 수 있는지 살펴보도록 하겠다.

물류 시스템을 워크플로로 나타내기 위해서는, 기업의 물류활동을 프로세스(process)와 작업(task)으로 나누어 분석해야 한다. 여기에는 계층적 상세화(hierarchical detailing)의 개념이 적용되어, 최상위수준에서는 각 작업이 비교적 큰 규모로서, 그 자체가 하위 워크플로(sub-workflow)의 성격을 갖고, 수준이 내려갈수록 작업의 크기는 줄어들어 하나의 응용 프로그램 또는 데이터베이스 트랜잭션에 이르게 된다.

이와 마찬가지로, 일반적인 기업의 물류활동도 여러 물류 프로세스들을 포함하고 있는데, 본 연구에서는 각각의 물류프로세스의 하위 프로세스인 출하물류(outbound logistics)부문의 한 예를 들어 페트리넷으로 구성하는 예로 제시하고자 한다. 예에서는 주문에 의해 생산이 이루어지는 ATO (Assemble To Order) system 가정하였고, 이의 페트리넷 구성은 <그림 1>과 같다[11].

일련의 접수된 주문에 의해 견적서를 작성한 후, 견적서를 고객에게 다시 발송하여 고객의 주문이 있을 경우 물품출고 절차를 시작하게 된다. 물류센터 또는 창고에서 물품의 출고는 일단 창고의 재고를 확인한 후 고객이 주문한 제품의 재고가 있을 경우 곧바로 출고 프로세스를 진행하게 되지만, 재고가 없을 경우에는 생산에 들어가게 된다.

제품생산 후, 창고에 입고되기 전에 물품을 검사하고(검수) 입고가 이루어지게 된다. 이 때, 재고기록파일 또는 DB에 입고된 물품의 수량 및 명세가 기록되게 된다. 그리고 이러한 제품들의 출고 프로세스를 거쳐 배송이 이루어지며, 이에 따른 송

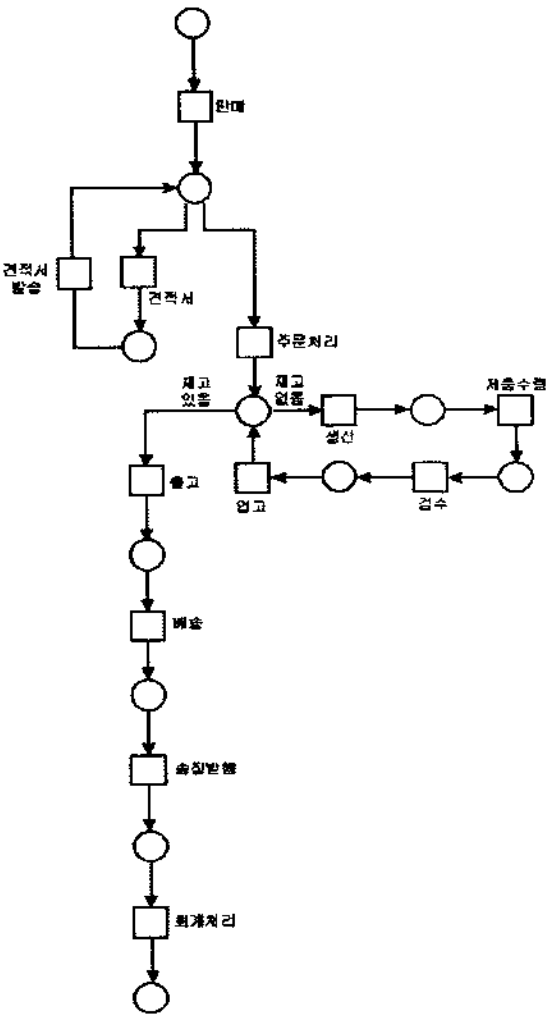

그림 1. 출하물류 워크플로.

장발행 및 회계처리가 이루어지는 것으로 주문접수에서 배송까지의 일련의 물류 프로세스가 이루어지게 된다.

<그림 1>에서, 몇 개의 플레이스에서는 조건에 따른 분기가 이루어지고 있으며, 각 트랜지션은 비교적 큰 규모의 작업에 관련되어, 대개 하위 워크플로를 갖는다. 또한, 그림에서 나타난 부분은 작업의 흐름을 나타내는 부분(control flow)이며 각 플레이스와 트랜지션에서는 <표 1>에서 정의된 속성을 명세해 주어야 한다. 예로서, 출고작업의 경우 다음 <그림 2>와 같이 하위 워크플로로 나타낼 수 있다[11].

출고할 제품내역에 따라 출고 지시서를 작성, 이를 확인하여 특정장소에서의 물품출고를 요구하는지에 따라 출고전략을 선택한다. 제품을 물리적으로 반출한 후 이 물품이 반출되었음을 확인하고 재고기록을 갱신하는 것으로 출고 프로세스는 종결되며 배송 프로세스에 물품을 인도하게 된다.

출고 프로세스는 상위인 물류 프로세스에서 이미 출고할

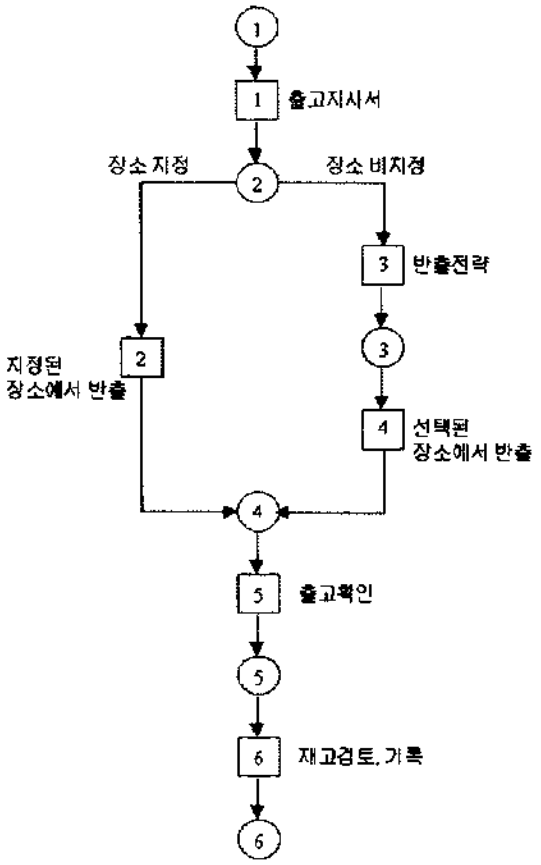

그림 2. 출고 워크플로.

제품에 대한 내역이 주어져 있는 것을 가정하며, 결과로서 출고된 물품에 대한 정보가 상위 프로세스로 전달된다. 자료에 대한 명세를 위해, 각 플레이스와 트랜지션이 갖는 속성을 표로 나타내면 <표 2>와 같다. <표 2>에서 각 트랜지션은 해당활동을, 각 플레이스는 활동전후의 입/출력 자료의 보관장소 역할을 하고 있음을 확인할 수 있다. 또한, 조건이 존재하는 2번 플레이스의 경우 자료(출고지시서)의 내용에 의한 분기조건문을 기술하였다.

<표 2>에 따라, 각각의 하위 워크플로에 대한 명세가 끝나면, 상위 워크플로에서의 자료 흐름에 대한 상세 명세를 구할 수 있고, 물류 워크플로에 대한 명세는 완료된다. 여기에, 신규 개발 또는 기존응용의 통합을 통해 각 작업을 수행하는 클라이언트가 연결되면 물류 프로세스 전반을 워크플로 관리시스템을 통해 통합할 수 있다.

이렇게 이루어진 통합 물류 시스템은 그 자체로서 하나의 패키지와 같이 작동할 뿐만 아니라, 프로세스에 대한 변경이 필요한 경우 워크플로 명세만 바꾸어 주면 되므로 유연한 프로세스 변경을 지원할 수 있는 장점이 있다.

6. 결 론

본 연구에서는 통합 물류 시스템 구축을 위해 워크플로 관리

표 2. 출고 workflow상의 플레이스, 트랜지션의 속성

번호	플레이스		트랜지션			
	보관자료	분기조건문	이름	작업	입력자료	출력자료
1	출고할 제품 내역	없음	출고 지시서	출고지시서 생성	출고할 제품 내역	출고지시서
2	출고지시서	특정 장소에서 출고 지시 여부	지정된 장소에서 반출	지정된 장소에서 제품 반출	출고지시서	출고지시서, 출고된 제품 정보
3	출고지시서, 반출전략에 따라 선택된 장소	없음	반출전략	반출 전략에 따라 반출 장소 선택	출고지시서	출고지시서, 선택된 반출 장소 정보
4	출고지시서, 출고된 제품 정보	없음	선택된 장소에서 반출	선택된 반출 장소에서 제품 반출	출고지시서, 선택된 반출 장소 정보	출고지시서, 출고된 제품 정보
5	출고 확인 여부, 출고된 제품 정보	없음	출고확인	출고된 제품과 출고지시서의 대조 확인	출고지시서, 출고된 제품 정보	출고된 제품 정보, 출고 확인 여부
6	출고된 제품에 대한 정보	없음	재고 검토, 기록	현 재고 검토하여 재고관리, 기록	출고된 제품 정보, 출고 확인 여부	출고된 제품 정보

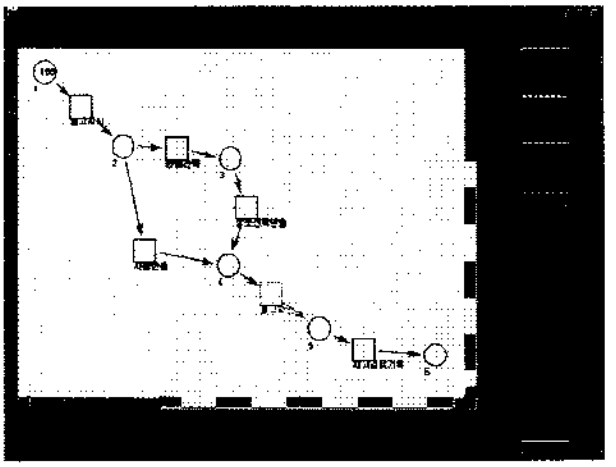

그림 3. 프로세스 정의 모듈.

시스템을 이용하는 프레임워크를 설계하였다. 이를 위한 워크플로 명세방법으로 페트리넷을 사용할 경우의 고려사항에 대해 다루었으며, 워크플로 관리시스템 구현시의 고려사항에 대해 언급하였다. 또, 이를 일반적인 기업의 물류 시스템에 적용하여 통합 물류 시스템을 구축하는 방안을 다루었다.

현재 이 페트리넷 기반의 워크플로 관리시스템은 프로토타입을 구현한 상태이며(<그림 3>, <그림 4> 참조), 소규모의 프로세스에 대한 클라이언트의 개발과 이의 통합실험이 이루어진 바 있다.

프로토타입은 자바언어로 구현되었고, 네트워크 연결부분에는 자바의 RMI[10]가 사용되었다. RMI는 CORBA와 유사한 구조를 가지므로, 향후 CORBA 환경으로의 변경이 용이할 것으로 기대할 수 있다.

본 연구에서 제시한 프레임워크는 물류 시스템의 통합 뿐 아니라, 향후 분석범위를 확대하여 기업시스템에 대한 워크플로 관점의 분석이 이루어진다면, 물류를 포함한 전사적 규모의 ERP 시스템에도 적용가능할 것으로 기대된다.

이러한 프레임워크는 유연한 프로세스 변경을 지원하므로, 기업 특성상 특수한 프로세스가 불가피한 경우에 효과적으로 사용될 수 있을 것이다.

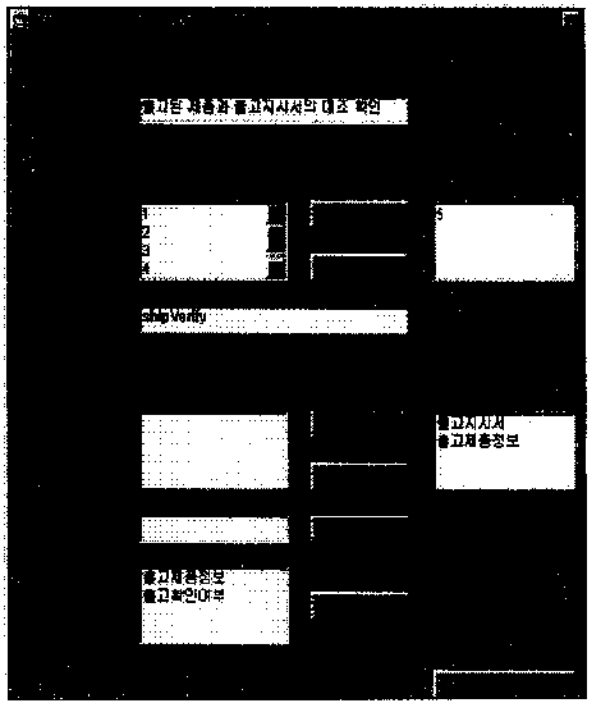

그림 4. 트랜지션의 자료 속성.

참고문헌

1. Sheth, A., *An Overview of Workflow Management : From Process Modeling to Workflow Automation Infrastructure,* Distributed and Parallel Database, Vol. 3, 1995.
2. Peterson, J. L., *Petrinet and the Modeling of Systems,* Prentice-Hall, 1981.
3. Orfali, R. and Harkey, D., *Client/Server Programming with JAVA and CORBA,* John Willey & Sons, 1997.
4. Kim, S., Seo, Y. and Hahm, J., "Development of an electronic data interchange support system based on workflow concepts," *The 14th International Conference on Production Research Proceedings,* August 1997.
5. Hammer, M., *et al.*, *Reengineering the Corporation,* 김영사, 1993.
6. Merz, M., Liberman, B., Muller-Jones, K. and Lamersdorf, W., "Interorganizational workflow management with mobile agents in COSM," *Institute for Information Management,* 1994.
7. Merz, M., Liberman, B., Muller-Jones, K. and Lamersdorf, W., "Workflow modeling and execution with coloured petrinets in COSM," *Institute for Information Management,* 1994.
8. Van der Aalst, W. M. P., "Three good reasons for using a petri-net-based workflow management system," *Eindhoven University of Technologies,* 1995.
9. Mohan, C., *Tutorial : State of the Art in Workflow Management System Research and Products,* IBM Research Center, 1996.
10. Flanagan, D., *JAVA in a Nutshell,* O'Relly, 1997.
11. *SAP-R3 Analyzer,* SAP, 1997.
12. Alonso, G., Agrawal, D., Abbadi, A., *et al.*, "Advanced transaction models in workflow contexts," *IBM Research Report,* 1996.
13. Ting, C., Gloor, P. A. and Nog, S., "Dartflow : a workflow management system on the web using transportable agents," *Technical Report, Dept. of CS., Dartmouth College,* 1996.

14. Database Systems Lab., "Workflow management and support for advanced database applications," *http://www.cs.umass.edu/db/wf.html*, 1995.
15. Georgakopoulos, D., Hornick, M. and Sheth, A., "An overview of workflow management : from process modeling to workflow automation infrastructure," *Distributed and Parallel Databases*, Vol. 3, 1995.
16. Kappel, G., Lang, P., Rausch-Schott, S. and Retschitzegger, W., "Workflow management based on objects," *Rules, Roles, Data Engineering*, Vol. 18, No. 1, pp. 11-18, 1995.
17. Krishnakumar, N. and Sheth, A., "Managing heterogeneous multi-system tasks to support enterprise wide operations," *Distributed and Parallel Databases*, Vol. 3, 1995
18. Miller, J. A., Sheth, A. P., Kochut, K. J. and Wang, X., "CORBA-based run-time architecture for work-flow management systems," *Technical Report, Large Scale Distributed Information Systems Lab., Dept. of CS, U. of Georgia*, 1995.
19. Mohan, C., "Tutorial : state of the art in workflow management system research and products," *ACM SIGMOD Intl. Conf. on Mgmt. of Data*, 1996.
20. Hollingsworth, D., "The workflow reference model," *Technical Report, Workflow Management Coalition*, 1994.

서용원
1994년 서울대학교 산업공학과 학사
1996년 서울대학교 산업공학과 석사
1998년 서울대학교 산업공학과 박사 수료
현재: 서울대학교 공과대학 산업공학과 박사 과정
관심분야: 물류 프로세스 분석 및 개선분야

함주호
1983년 서울대학교 산업공학과 학사
1985년 서울대학교 산업공학과 석사
1990년 University of Michigan at Ann Arbor 산업공학과 박사
관심분야: 재고관리, Supply Chain Management system, Enterprise Resource Planning, Business Process Reengineering

성계훈
1994년 서울대학교 산업공학과 학사
1996년 서울대학교 산업공학과 석사
1998년 서울대학교 산업공학과 박사 수료
현재: 서울대학교 산업공학과 박사과정
관심분야: Supply Chain Management System 개발 및 구현, Business Process Reengineering