제약이론을 활용한 업무프로세스의 효율적 실행 방법

Efficient Execution Method for Business Process Management using TOC Concepts

이승현(Seung-Hyun Rhee)*, 배혜림(Hyerim Bae)**†, 원형준(Hyungjun Won)***,
김훈태(Hoontae Kim)****, 강석호(Suk-Ho Kang)*

초 록

업무프로세스관리(BPM) 시스템은 업무프로세스의 효율적인 실행, 통제 및 관리를 지원하는 소프트웨어 시스템으로서, 복잡한 프로세스를 자동화하고, 이를 효과적으로 관리하여 기업의 생산성 향상에 기여한다. 그러나, 상용시스템은 주로 프로세스의 자동화에 초점을 두고 있어 프로세스와 업무참여자 관점에서 효율적인 관리 방법을 제공하지 못하고 있다. 따라서, BPM 시스템은 프로세스 참여자와 전체 프로세스 효율성 향상의 관점에서 개선의 여지가 있다. 이에 본 논문에서는 업무프로세스를 보다 효율적으로 관리하는 새로운 방법을 제안한다. 제안한 방법은 업무참여자의 업무부하를 고려하여 전체 프로세스를 실행하는 방식으로서, 프로세스 참여자원 가운데 가장 큰 제약요소를 집중관리한다. 이러한 방식은 경영혁신 이론으로 알려진 제약이론(Theory of Constraints)의 DBR (Drum-Buffer-Rope) 방법론을 기반으로 하고 있다. DBR 방법론을 활용하기 위해 우선, 업무프로세스와 DBR의 주적용모델의 차이점을 분석하고 업무프로세스에 적합한 드럼, 버퍼, 로프를 개발한다. 이를 토대로, 업무참여자의 업무부하와 프로세스 인스턴스 발생 패턴을 통제하는 BPM만의 DBR 방법론(BP-DBR, Business Process-DBR)을 제안한다. BP-DBR은 업무참여자의 업무환경과 전체프로세스의 효율성 개선에 기여할 것으로 기대한다. 이러한 기대효과를 증명하기 위해 기존 BPM의 프로세스 관리방법과 제안한 방법의 효과를 시뮬레이션 실험을 통해 비교한다.

ABSTRACT

Business Process Management (BPM) System is a software system to support an efficient execution, control and management of business processes. The system automates complex business processes and manages them effectively to raise productivity. Traditional commercial systems mainly focus on automating processes and do not have methods for enhancing process performances and task performer's efficiency. Therefore, there is room for enhancement of task performers' productivities and efficiency of business processes. In this paper, we propose a new method of executing business processes more efficiently in that a whole process is scheduled considering the degree of participants' workload. The method allows managing the largest constraints among constituent resources of the process. This method is based on the DBR (Drum-Buffer-Rope) in TOC (Theory of Constraints) concepts. We first consider the differences between business process models and DBR application models, and then develop the modified drum, buffer and rope. This leads us to develop BP-DBR (Business Process-DBR) that can control the proper size of task performers' work list and arrival rate of process instances. Use of BP-DBR improves the efficiency of the whole process as well as participants' working condition. We then carry out a set of simulation experiments and compare the effectiveness of our approach with that of the scheduling techniques used in existing systems.

키워드 : 업무프로세스관리 시스템, 제약이론, 드럼-버퍼-로프, 시뮬레이션
Business Process Management, Theory of Constraints, Drum-Buffer-Rope, Simulation

---

본 연구는 2004년도 부산대학교 교내학술연구비(신임교수연금정착금)에 의한 연구임
* 서울대학교 산업공학과
** 부산대학교 산업공학과 †교신저자
*** (주)핸디소프트 BPM컨설팅팀
**** 대진대학교 산업시스템공학과

1. 서 론

오늘날 기업의 비즈니스 상황은 시장 환경이 철저하게 고객중심으로 변화함에 따라 기업 내, 기업 간 거래 등에서 더욱 복잡한 프로세스의 수행을 요구하고 있다. 이에 따라, 기업의 프로세스를 효과적으로 관리하는 것이 기업 경쟁력 강화에 필수적이라는 인식이 자리잡게 되었다. 실제로 복잡한 프로세스를 정형화된 형태로 모델링하고 이를 명확히 하여 기업을 운영하면, 시장의 요구에 민첩하게 반응하여 궁극적으로 보다 많은 이윤 창출에 기여할 수 있다[11]. 프로세스 중심 경영 하에서는 참여자들이 전체 프로세스 목표를 이해하고 자신의 업무가 줄 수 있는 영향에 관심을 가지며, 프로세스 효율성과 성과를 염두에 두어야 한다. 이러한 경영 방식의 필요성이 대두되고 있는 가운데, 최근 많은 기업들이 업무프로세스관리 (Business Process Management, BPM) 시스템을 실천의 도구로 선택하고 있다. 이 시스템은 기업 안팎으로 산재한 프로세스를 정의, 구현, 개선하는 등의 모델링을 수행하고, 그 결과를 통해 정확한 순서대로 프로세스를 진행하고, 자동화함은 물론 다른 조직과의 상호연동을 편리하게 해준다[20].

그러나, 기존 업무프로세스관리 시스템은 프로세스의 정확한 수행을 담당함에도 불구하고 전체 프로세스의 효율적인 관리라는 측면에서는 개선의 여지가 있다. BPM을 사용하는 업무담당자들은 각자 고유의 능력을 갖고 프로세스 내 서로 다른 업무에 참여하고, 동일한 단위업무를 여러 번 처리한다. 이러한 환경에서 특정 업무담당자는 주어진 업무에만 주목하기 때문에, 자신의 업무처리가 다른 업무담당자의 업무 처리, 전체 프로세스 진행에 영향을 준다는 것을 구체적으로 인식하지 못한다. 다시 말해, 프로세스 참여자들은 전체 프로세스의 목적을 이해하지 못하는 경향이 있다[11][12]. 요컨대, 기존 업무프로세스관리 시스템은 업무담당자의 능력을 고려하여 전체 프로세스를 효율적으로 관리하는 기능을 제공하지 못하며, 프로세스의 진행 상황을 파악할 수 있는 정보를 효과적으로 활용하지 못한다.

본 논문에서는 각 업무담당자들이 전체 프로세스를 이해하고 이에 따라 프로세스에 참여하도록 유도하기 위하여 제약이론(TOC, Theory of Constraints)의 개념을 도입한다. 제약이론은 모든 자원의 효율성을 추구하는 것이 아니라, 프로세스 진행을 저해하는 자원을 발견하여 이를 집중 관리하는 이론이다. 이 자원은 전체 프로세스의 진행을 방해하기 때문에, 이 자원의 효율성이 전체 프로세스의 효율성으로 이어질 수 있기 때문이다. 본 논문에서는 지금까지 생산공정의 효율성 제고에 주로 응용되던 DBR (Drum-Buffer-Rope) 방법을 업무프로세스관리 영역에 적용함으로써 프로세스의 전체적인 효율을 높이데 기여한다. 이를 위하여 DBR 적용 시 고려해야 할 사항을 논의하고 변형된 DBR 방법론을 제안한다. 또한, 제안한 방법의 효과를 시뮬레이션 실험을 통해 확인한다.

2. 배경이론

2.1 업무프로세스관리 (Business Process Management, BPM)

BPM은 기업의 생산성을 제고하기 위해 업무절차를 체계적으로 설계, 관리, 개선하는 활동을 지원하는 총체적인 관리 방법론이다[20]. 미리 정의된 정형화된 프로세스 모델을 따라 실제 업무를 프로세스 참여자원에 부여함으로써 업무를 진행하는 방식으로[2][13], 기술적으로는 비즈니스 프로세스의 자동화를 의미하는 워크플로우(Workflow)를 포괄하고 있다[14][20][22][23]. BPM 시스템은 이러한 업무프로세스관리 방법을 구체화하여 효율적인 업무 환경을 지원하는 소프트웨어 시스템이다[20].

BPM 시스템이 프로세스를 자동으로 수행하기 위해서는 컴퓨터가 이해할 수 있는 형태로 프로세스가 표현되어 있어야 한다[4]. 이렇게 표현한 것을 본 논문에서는 '프로세스 모델' 이라고 한다[3]. 다시 말해, 업무프로세스들은 BPM 시스템 기반에서 프로세스 모델로 표현된 것으로서, 본 논문에서 언급하는 업무프로세스를 프로세스 모델로 이해할 수 있다. 본 논문은 프로세스 모델 자체의 정확성 문제를 다루는 것은 아니므로 구조적으로 최소화된 형태의 모델만을 정의하기로 한다. 프로세스 모델은 프로세스 구조에 의해 표현되며, 그 구조는 프로세스를 이루는 단위업무와 이들간의 선후 관계를 말한다[3][22].

정의 1 (프로세스 모델)

프로세스 모델은 다음을 만족하는 유방향 그래프 P = (A, L)이다.

- A = $\{a_i \mid i = 1, \cdots, N\}$는 단위업무의 집합이다. a_i는 i-번째 단위업무이며, N은 P에 포함된 단위업무의 총개수이다.
- L⊆ $\{(a_i, a_j) \in a_i, a_j \text{ I } A \text{ and } i \neq j\}$는 링크 집합이다. 원소 (a_i, a_j)는 단위업무 a_i가 단위업무 a_j보다 바로 앞서 실행돼야 함을 의미한다.

프로세스 모델은 정의시(build-time) 단계와 실행시(run-time) 단계로 나누어 고려하는 것이 일반적이다[19]. 정의시 단계는 프로세스 수행을 준비하는 과정으로, 프로세스를 이루는 단위업무와 이들 간의 선후 관계 그리고 각 단위업무를 수행하는데 필요한 세부 속성들을 정의한다. 실행시 단계에서는 정의시 단계의 프로세스 모델을 해석하여 실제 프로세스를 실행, 관리, 통제한다[3][16]. 특히, BPM에서 실행시 단계를 담당하는 부분을 BPM 엔진(engine)이라 한다[16][23]. 실제로 처리해야 할 프로세스가 발생하면 BPM 엔진은 해당되는 정의시의 프로세스 모델을 바탕으로 실제 프로세스를 발생시키게 되는데, 이 때 생성되는 개별 프로세스를 프로세스 인스턴스(instance)라고 한다[23].

BPM 엔진은 시스템의 중심부에 위치하며, 실행시에 이루어지는 활동을 관리한다. 정의된 프로세스를 해석하고, 프로세스를 초기화하며 프로세스 인스턴스를 발생시킨다. 또한, 단위업무들의 순서를 조정하며, 프로세스 수행시 필요한 애플리케이션을 호출한다. 업무담당자와의 상호 작용을 통해 업무 수행에 필

요한 정보를 제공하기도 한다[20].

한편, 엔진은 수행해야 할 단위업무들을 적절한 업무담당자에게 할당하여, 이들을 처리하도록 한다. 이 때, 개별 업무담당자에 할당된 업무집합을 업무목록(work list)이라 한다. 그리고, 이러한 업무 할당은 보통 어떤 규칙에 따라 이루어지는데 이를 업무할당규칙(Task Allocation Rule)이라 한다. 이 규칙은 BPM 엔진의 관점에서는 효과적 또는 효율적 업무할당규칙이거나, 업무프로세스 특성상 필요한 적임자에 할당하는 규칙이다. 이 규칙은 보통 두 단계를 거쳐 적용되는데, 먼저 정의시에 미리 정의된 역할을 기반으로 일차적으로 특정 단위업무 수행을 담당할 업무담당자 후보군을 정의한다. 이를 바탕으로, 실행시에는 후보군 가운데 해당 시점에서 단위업무 수행에 가장 적합한 업무담당자를 결정하여 할당한다[19]. 보통은 업무 부하가 가장 작은 사람 또는 업무 실적이 가장 뛰어난 후보에게 업무가 할당된다.

2.2 제약이론

제약이론(Theory of Constraints, TOC)은 시스템의 목적 달성을 저해하는 제약자원을 찾아내어 이를 극복하여 기업의 수익창출에 기여하는 경영기법이다[8][9][10]. 이를 위하여, 현금창출율 증가, 재고 감소, 운영비용 절감에 주력한다[10].

일반적으로 제약이론은 5단계의 절차를 거쳐 진행된다. 첫 번째 단계는 제약자원(Constraint Capacity Resource, CCR)을 발견하는 단계이다. 제약자원이란, 현금창출율을 결정하는 요인이며 나아가서는 기업의 목적인 이익증대에 결정적으로 영향을 미치는 요인이다. 두 번째 단계는 제약자원을 철저히 활용하는 단계로, 총 경비의 증가나 추가적인 투자 없이 기업의 이익을 증대시킬 수 있다. 다음 단계는 비제약자원들을 제약자원에 종속시키는 단계인데, 제약자원의 활용능력이 기업 전체의 성과와 직결되기 때문에, 제약자원의 효율성이 극대화될 수 있도록 비제약자원들을 운영한다. 즉, 전체에 결정적인 영향을 주는 곳의 효율만을 극대화시킨다는 것이다. 이는 부분의 효율이 전체의 효율로 이어지지 않는다는 사실에 기인한다. 네 번째 단계는 제약자원의 숨겨진 능력까지 철저히 활용한 후에 투자를 통해 이 자원 자체의 능력을 향상시키는 과정이다. 마지막으로, 제약자원이 다른 공정으로 바뀌는지 주의해가며 첫 단계부터 반복한다. 제약자원의 능력이 점점 개선되면 다른 공정이 제약자원으로 되고 마지막에는 결국 시장이 제약자원이 된다. 이 단계에서는 제약자원이 바뀜에 따라 개선활동의 내용도 대폭 달라져야 하므로, 타성에 젖어 태만해지지 않도록 주의하는 것을 강조한다[8][9][21].

한편, 제약이론 가운데 효율적인 스케줄링 관리를 지원하는 방법론으로 DBR (Drum-Buffer-Rope)이 있다[15]. DBR은 생산과정을 군대의 행진에 비유한 방법론으로, 드럼(Drum)은 제약자원의 업무처리속도를 의미하며, 전체 공정의 생산 속도를 결정한다. 버퍼(Buffer)에는 생산의 통계적 변동으로 인한 제약자원의 작업 중단을 방지하는 제약버퍼(Constraint Buffer)와 조립 공정이 존재할 때

출하 시기가 지연되는 것을 방지하는 조립버퍼(Shipping Buffer) 등이 있다[9][10][15]. 버퍼의 설정과 관리는 제약자원을 활용하는데 굉장히 중요한 방법이며, 버퍼로부터 공정의 진행 상황 등을 파악할 수 있는 정보를 제공받을 수 있다. 로프(Rope)는 제약자원과 시작 공정 간의 통신 장치로서 제약자원의 공정 속도에 맞춰 시작 공정에 자재를 투입하는 속도를 조절한다.

DBR은 효율적인 시간관리와 자원관리를 목적으로 한다. 이 기법은 주로 제조 공정에 적용되는 스케줄링 방법으로서, 버퍼 크기나 우선순위규칙과 같은 할당 문제와 관련해서 연구되어 왔다[6][7][18]. 한편, 상용 업무프로세스관리 시스템은 아직까지 프로세스의 자동화를 지원하는데 중점을 두고 있어, 업무담당자의 업무능력이나 부하를 고려한 효과적인 관리 기법을 체계적으로 지원하지 않고 있다. 이에 본 연구에서는 제약이론의 5단계 절차에 기반한 DBR 기법을 적용하여 효율적인 업무프로세스 관리 기법을 제안한다.

3. DBR을 업무프로세스에 적용할 때 고려사항

제약이론에서 DBR 기법은 주로 생산 공정의 효율화를 위한 방법론으로 사용되어 왔으나[7][15], 지금까지 업무프로세스에 적용된 예는 없었으며[19], 관련연구 또한 체계적으로 이루어지지 못하고 있다[1]. 이에 본 논문에서는 업무프로세스에 DBR 방법론을 새로이 도입하여 효율적인 프로세스 관리방법을 제안한다. 이를 위하여 본 장에서는 업무프로세스와 생산공정 간의 차이를 논의하고, 업무프로세스에서 필요한 DBR 관련 개념을 정의한다.

우선, 업무프로세스와 생산공정의 차이를 살펴보면, 생산 공정에는 단위 공정과 필요한 라우팅이 고정되어 있고, 자원도 명시적으로 할당되어 있다. 그러나, 업무프로세스에서는 진행 상황에 따라 다양한 대안이 실행되도록 표현되어야 하기 때문에, 자원이 명시적으로 배치되거나 할당되지 않는다. 한편, 생산 공정에서는 로트 크기를 결정하여 이를 공정에 투입하며, 상황에 따라 예측 생산을 통해 주문을 고정시킬 수 있다. 업무프로세스에서도 로트는 존재하는데, 고유한 인스턴스들이 로트의 역할을 하며 그 크기는 1이다. 그리고, 업무프로세스에서 자원은 주로 업무담당자 혹은 업무관련 애플리케이션인 반면에, 생산 공정은 생산설비를 비롯하여 공정 참여자 등 다양하다. 이러한 자원들에 업무를 할당할 때, 업무프로세스에서는 정의시 1차 할당을 명세한 뒤, 실행시에 업무를 실제 수행할 업무담당자에게 전달한다. 생산 공정에서는 생산설비에 고정, 혹은 그룹별 할당이 이루어진다. 또한, 생산공정은 많은 연구를 통해 효과적인 스케줄링 방법들이 고안되어 계획적이고 체계적으로 관리되고 있지만, 업무프로세스는 체계적인 관리 방법이 부족하며, 특히 전체 프로세스 관점에서 관리가 제대로 이루어지지 않고 있다[5]. 지금까지 언급한 바와 같이 업무프로세스와 일반적인 DBR 적용 모델인 생산 공정 간에는 차이점이 존재하며, 업무프로세스에 DBR을 적용할 때 이러한 점들이

우선 고려되어야 한다.

업무프로세스의 드럼, 버퍼, 로프를 정의하기 앞서, 본 연구에서 고려하는 업무 환경은 많은 프로세스 인스턴스가 발생하는 환경만을 고려함을 밝힌다. 즉, 많은 인스턴스가 발생하고 전체의 효율을 증진시키는 것이 중요한 업무환경을 고려한다. 한편, 업무프로세스에 참여하는 자원은 업무담당자에 한정하여, 제약자원은 업무담당자가 되도록 한다.

업무담당자는 프로세스 내 다수개의 단위업무를 수행하는 것이 일반적이다[23]. 그리고, 업무를 할당할 때에 적용되는 할당규칙에 따라 특정 단위업무에 참여하는 업무담당자는 발생하는 인스턴스마다 달라질 수 있다. 이 때, 인스턴스 별로 제약자원도 달라질 수 있기 때문에, 하나의 인스턴스마다 제약자원을 발견하는 것은 무의미하다. 따라서, 다수개의 인스턴스가 발생하는 특정 기간(DCI, Duration for CCR Identification)을 두고 제약자원을 찾아야 한다. 일정량의 인스턴스가 발생하면 하나의 인스턴스 완료시간은 다른 인스턴스들의 진행 상황에 영향을 받는다. 업무담당자가 처리하는 단위업무 완료가 지체되면, 할당된 또 다른 단위업무들은 대기 해야 하므로, 결과적으로 다른 인스턴스들의 진행을 방해하게 된다. 이를 통해 현 시점에서 업무목록에 많은 단위업무가 대기하고 있는 업무담당자가 프로세스 진행에 장애가 된다고 이해할 수 있다. 본 연구에서는 업무프로세스의 제약자원을 'BP-CCR'이라 명명하고, 일정 기간(DCI)동안 발생한 결과를 통해서 업무부하가 가장 큰 업무담당자로 정의한다. 제약자원을 정의하기에 앞서 먼저 프로세스의 자원인 업무담당자의 업무부하를 다음과 같이 정의한다.

정의 2 (업무부하).

현재 시점을 기준으로 한 i-번째 업무담당자가 J개의 처리되지 않은 업무를 가지고 있을 때, 업무부하 WL_i는 다음과 같이 정의한다.

$$WL_i = \sum_{j=1}^{J} ET_j \qquad (1)$$

ET_j: j-번째 단위 업무(a_j)의 기대처리시간
J: i-번째 업무담당자에 할당되는 업무의 수

위에서 정의한 것과 같이 각 업무 담당자의 업무부하는 할당되었으나 현재 시점에서 처리되지 않은 업무들의 기대처리 시간의 합이다. 다음으로 제약자원을 아래와 같이 정의한다.

정의 3 (BP-CCR).

$WL_R = max\{WL_i \mid WL_i$는 i-번째 업무담당자의 업무부하, $i=1, \cdots, I\}$ (2)

I: 프로세스에 참여하는 업무담당자의 수

라고 할 때, BP-CCR은 업무부하가 WL_R인 업무담당자이다.

BP-CCR을 바탕으로 제약자원의 DBR에 근거한 업무프로세스만의 드럼 및 버퍼, 로프 등을 정의할 수 있다. 드럼은 BP-CCR의 업무처리속도로 정의한다. 한편, 버퍼는 BP-CCR이 예기치 못한 상황에서도 끊임없이 가동될 수 있도록 하는 기능을 가져야 하므로,

〈표 1〉 업무프로세스의 드럼, 버퍼, 로프

항목	설 명
드럼	BP-CCR의 업무처리속도
버퍼	적절한 양의 단위업무가 적재되어 있는 BP-CCR의 업무목록
로프	BP-CCR의 업무부하 상태에 따른 프로세스 인스턴스 발생의 조절

적절한 양의 단위업무가 적재되어 있는 BP-CCR의 업무목록으로 정의한다. 업무프로세스의 로프는 BP-CCR의 업무부하 상태에 따라 프로세스 인스턴스 발생을 조절하는 역할로서, 프로세스 인스턴스 발생량에 따라 각 업무담당자의 업무부하가 영향을 받기 때문에, BP-CCR의 업무부하를 조절할 수 있다. 로프의 기능은 BPM의 엔진을 통해서 구현할 수 있다. 〈표 1〉은 업무프로세스의 DBR을 정리한 것이다.

지금까지 DBR을 업무프로세스에 적용하기 위해 고려해야 할 사항을 논의하였다. 업무프로세스 환경에서 드럼, 버퍼, 로프의 기능 또한 확인하였다. 다음 장에서는 이러한 논의사항을 바탕으로 DBR 기법을 업무프로세스에 적용하는 과정을 설명한다.

4. BP-DBR (Business Process-DBR)

본 장에서는 전술한 업무프로세스에 적용되는 DBR 기법의 구체적인 내용을 설명한다. 먼저, 업무프로세스 환경에서 드럼, 버퍼, 로프를 정의하는 방법을 설명하고, 이를 토대로 새로운 업무할당규칙을 제안한다.

4.1 드럼

먼저 업무프로세스의 드럼을 계산하는 방법을 설명한다. 이를 위하여 우선 BP-CCR을 판단할 수 있을 만큼 충분한 시간으로 적용기간(*DCI*)을 결정해야 한다. 전술한 바와 같이 BP-CCR은 현재 시점에서 업무부하가 가장 큰 업무담당자이다. BP-CCR이 결정되면 다음과 같이 해당 업무담당자의 업무처리속도인 드럼을 계산해야 한다.

$$\text{드럼}(d) = \frac{\text{총단위업무 처리개수}}{\text{총업무참여시간}} \qquad (3)$$

그러나 이 때, BP-CCR은 일반적으로 다수개의 단위업무에 참여하고 이들 단위업무는 그 처리시간이 서로 다르다. 따라서, 속도를 계산하기 위하여 단위시간당 처리한 업무의 개수를 적용하는 것은 서로 다른 단위의 수치를 단순합으로 계산하는 것이기 때문에 불합리하다. 이를 해결하기 위하여 단위업무의 처리 개수는 처리시간이 가장 짧은 단위업무의 시간으로 나눈 값을 적용한다. 만일 처리시간 1인 단위업무(a_1)가 최소시간을 가진 단위업무라면 처리시간 2인 단위업무(a_2) 는 2개로 계산되어야 한다. 그러므로, 단위업무 a_j의 단

위개수는 다음과 같이 환산하여 사용한다.

$$a_i\text{의 단위개수}(UN_j) = \frac{ET_i}{ET_{\min}} \qquad (4)$$

ET_i : a_i의 예상처리시간

ET_{min} : BP-CCR에 할당된 최소 단위업무 처리시간

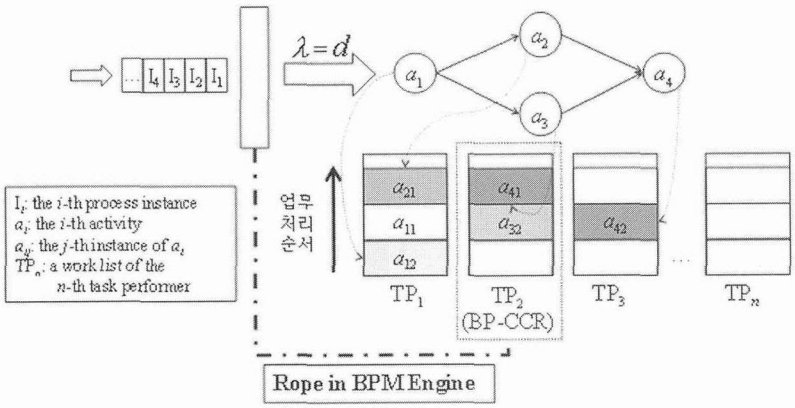
이 때, 특정 단위업무의 처리개수는 이 단위업무의 단위개수와 실제 처리된 단위업무 인스턴스 개수의 곱으로 계산된다. 주어진 시간 *DCI* 내에 *K*개의 단위 업무를 처리한다고, 가정할 때, 드럼(*d*)은 다음과 같이 계산한다.

$$\text{드럼}(d) = \frac{\sum_{k=1}^{K} UN_k \cdot n_k}{\sum_{k=1}^{K} PT_k \cdot n_k} \text{ (개/시간)} \qquad (5)$$

한편, BP-CCR의 업무할당은 정의시에 결정되기 때문에, 단위업무들의 예상처리시간을 통해 각 단위업무의 단위개수를 식 (4)와 같이 미리 계산할 수 있다. 단위업무의 단위개수(*UN*)는 일정한 값에 의해 결정된 결과로서, 실행시에도 그 값을 유지한다. 그러므로, 드럼(*d*)은 식 (5)의 n_j와 PT_k의 값에 따라 변화하며 특히, 드럼(*d*)은 PT_k에 반비례한다. 다시 말해, 업무처리시간이 길다는 것은 속도가 느리다는 것을 의미한다.

4.2 로프

드럼이 계산되면, BPM 엔진은 이를 고려하여 프로세스 인스턴스 발생간격을 조절할 수 있다. 즉, BPM 엔진이 로프의 기능을 수행하게 된다. BP-CCR이 효율적으로 업무에 참여하면 전체 프로세스 진행은 개선될 수 있기 때문에, 이러한 관점에서 BPM 엔진이 인스턴스 투입 속도를 드럼과 동기화한다. 뿐만 아니라, BP-CCR 이외의 각 업무담당자들은 처리할 단위업무의 프로세스 인스턴스 순서를 BP-CCR이 처리할 단위업무의 프로세스 인스턴스 순서로 일치시킨다. 예를 들어, 〈그림 1〉에서 보는 바와 같이 BP-CCR은 인스



〈그림 3〉 자체 구축을 통한 전환 구조

턴스 2번의 a_3을 처리한 후 인스턴스 1번의 a4를 처리한다. 여기에 맞춰 업무담당자 1 역시 인스턴스 2번의 a_1을 처리한 후에 인스턴스 1번의 단위업무들을 처리한다. 이는 BP-CCR이 처리한 단위업무가 속해 있는 프로세스 인스턴스의 진행이 다른 업무담당자로 인해 지연되는 것을 방지하기 위함이다. 〈그림 1〉은 로프의 기능을 실행시에 BPM 엔진이 하나의 프로세스를 관리하는 가상의 시나리오를 통해 보여주고 있다. 로프란 BPM 엔진이 이미 발생한 프로세스 인스턴스를 BP-CCR의 업무처리속도(드럼)에 맞춰 투입하는 기능을 말한다. 이 때, 인스턴스가 발생되는 간격 λ는 다음과 같이 정한다.

$$\lambda = d \text{ (개/시간)} \qquad (6)$$

한편, 프로세스가 진행됨에 따라 드럼은 지속적으로 변화하기 때문에, 식 (5)에서 n_j와 PT_k을 현 시점의 결과를 이용하여 갱신한다. 갱신된 드럼은 식 (6)의 동기화 방법에 반영되어 지속적으로 인스턴스 발생간격을 재계산할 수 있다. 이와 같은 방식으로 로프는 실행시에 지속적으로 프로세스의 효율적인 진행을 지원한다.

4.3 버퍼

프로세스 인스턴스가 실제 발생하면, 정의시에 정의한 프로세스 관련 속성들은 실제 프로세스의 진행에 따라 변화한다. 특히, 속성들 가운데 프로세스의 진행 정도는 BP-CCR의 업무부하를 통해 확인할 수 있다. BP-CCR의 업무부하가 크면 전체 프로세스의 진행이 지체되고 있는 것이고, 업무부하가 작으면 프로세스의 진행이 순조롭다는 것을 의미한다. 업무부하 정보는 드럼과 로프가 적용된 이후 BP-CCR의 업무목록 즉, 버퍼를 통해 확인할 수 있는데, 이 버퍼를 관리하면 전체 프로세스를 효율적으로 운영할 수 있다.

제약이론에서 제시하는 버퍼는 프로세스가 지속적으로 진행하되, 재고를 최소화하고 운영비용을 절감하기 위해 존재한다[8]. 그러므로, 원활한 프로세스 운영을 위해 BP-CCR 앞에 시간과 비용의 관점에서 적정한 크기의 버퍼를 두어야 한다. 버퍼크기 계산은 이미 많은 문헌 등을 통해 연구되고 있으며 [7][9][15][17], 제약자원의 버퍼크기를 비용의 관점에서 계산한 연구[18] 또한 존재한다. 이 연구에서는 제약자원을 K 크기의 유한 큐를 보유한 단일서버시스템($M/M/1/K$)으로 보았는데, 프로세스 효율의 관점에서 접근하지 않아 본 연구에 적용하기에는 어려움이 있다. 또한, 프로세스 효율을 최적화하는 버퍼크기는 상황과 관점에 따라 달리 접근하여 계산되는 성질을 지닌다. 따라서, 본 논문에서는 경험에 따른 적절한 버퍼크기가 이미 존재한다고 가정하고 이 크기를 통해 버퍼를 관리하는 방법을 논의한다.

업무프로세스에서는 버퍼를 통해 크게 두 가지 프로세스 진행 상황을 관리한다. 프로세스의 진행이 지체됐을 때, BP-CCR의 업무부하가 지나치게 증가하는 상황과 BP-CCR의 업무부하가 감소하여 처리할 단위업무가 고갈되었을 때 프로세스 진행이 중단되는 상황이다. 이러한 상황을 처리하기 위해, BP-CCR

```
INTEGER λ, d, SS, WL_R;
WHILE (λ > 0)
  IF WL_R > SS
      BLOCK BP-CCR'S Work List;
  ELSE IF WL_R < SS
      PUT INTO BP-CCR'S WORK LIST UNTIL REACHING THE SS;
  ELSE
      λ = d;
END;
```

〈그림 2〉 업무목록 관리

이 업무목록에 적정한 수준의 단위업무들을 대기시켜 일정 크기로 유지한다. 이 크기를 기준크기(Standard Size, SS)라 하며, 이 기준크기를 통해 다음 PSEUDO 코드와 같이 BP-CCR 업무부하를 조정한다. 아래 코드에서 λ는 0보다 큰 값을 유지해야 BPM이 실행되고 있다고 이해할 수 있다.

〈그림 2〉의 코드에서 확인할 수 있듯이, BP-CCR의 업무부하 크기가 SS보다 크면 엔진은 BP-CCR의 업무목록을 차단한다. 이는 인스턴스 투입으로 BP-CCR의 업무부하가 지나치게 증가하는 것을 방지하기 위함이다. 한편, BP-CCR의 업무부하 크기가 *SS*보다 작으면 다시 기준크기까지 단위업무가 누적되도록 버퍼에 투입한다. BP-CCR이 처리할 단위업무가 없다면 이는 제약이론에 의해 전체 프로세스의 진행이 중단됨을 의미하므로 BP-CCR은 항상 끊임없이 업무에 참여할 만큼의 업무부하를 필요로 한다. BP-CCR의 업무부하가 *SS*를 유지하면 엔진은 BP-CCR의 업무목록에 별다른 조치를 취하지 않고 드럼과 인스턴스 발생간격을 동기화하면 된다. 이와 같은 방법을 사용하면 *SS*부근에서 BP-CCR의 업무부하는 항상 안정적으로 유지될 것이라고 예상할 수 있다.

4.4 업무할당규칙 및 요약

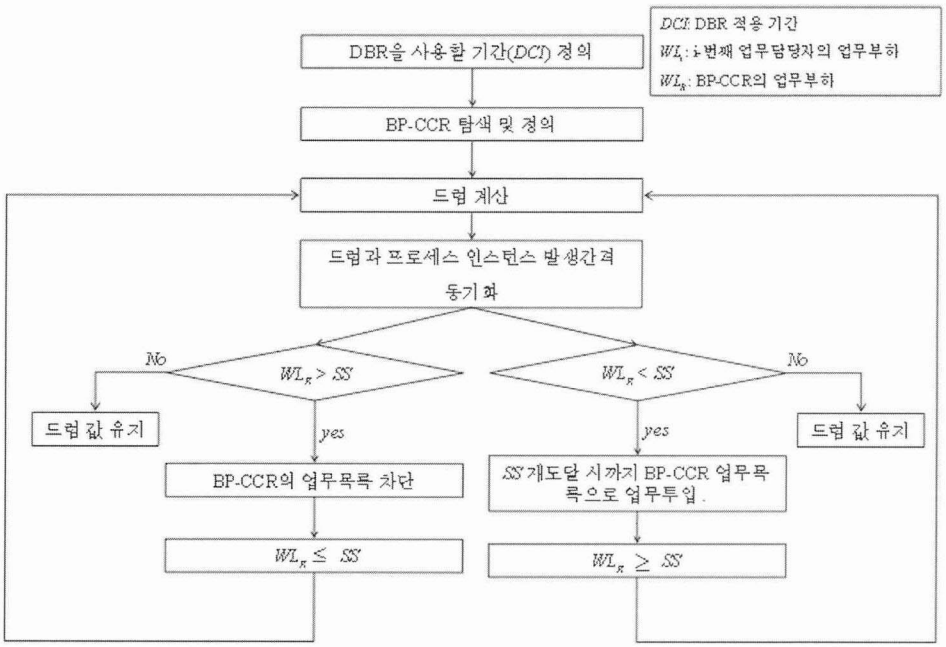
업무프로세스에 TOC 개념을 적용하여 프로세스의 효율을 높이는데 있어 BP-DBR 방법뿐 아니라, 업무할당규칙을 고려할 필요가 있다. 전술한 업무목록 관리방법에 의하면, BP-CCR의 업무부하가 *SS*를 초과하면 이 자원으로의 업무할당이 일시 중단된다. 이는 BP-CCR의 지나친 업무부하를 방지하지만, 할당되어야 할 단위업무들 관점에서는 효율을 떨어뜨리게 된다. 이를 해결하기 위해 BP-CCR이 담당해야 할 단위업무들을 여유능력을 갖춘 비 BP-CCR에 할당하는 업무할당규칙을 제안하며, 이 방식을 *'WL-TAR'* (Work List-Task Allocation Rule)이라 한다.

*WL-TAR*에 의하면 BP-CCR의 업무부하가 업무목록의 *SS*이상이 되면, 업무목록은 차단되기 때문에 대체자원에 업무를 할당한다.

BP-CCR의 능력은 한정되어 있기 때문에, 여유능력을 갖춘 대체자원이 BP-CCR의 업무를 일부 처리함으로써 BP-CCR의 능력을 향상시킨 효과를 기대할 수 있다. 이것은 잠재된 제약자원의 능력을 더 이상 발견할 수 없을 때까지 활용한 후, 이 자원의 능력 자체를 향상시킨다는 제약이론의 기본 원리를 따른 것이다.

*WL-TAR*은 독립적인 업무할당방식이 아니라 BP-DBR 기법 사용 시 본래의 업무할당방식을 보완하는 업무할당방식이다. BP-CCR이 그룹별 업무할당방식에 의해 이미 특정 단위업무를 전달 받고 있었다면, 그 업무를 대체하여 수행할 업무담당자들은 BP-CCR과 같은 그룹에 속한 업무담당자이다. 개인별 업무할당방식을 채택하고 있다면, 정의시 BP-CCR의 단위업무들을 대체 수행할 업무담당자를 미리 정의하면 된다.

지금까지 본 논문에서 BP-DBR 방법을 제안하였다. 이 방법은 아래 〈그림 3〉과 같이 일련의 절차에 따라 정의시와 실행시에 걸쳐 수행된다. 제안한 방법은 다음과 같은 개선 효과를 기대할 수 있다. BP-CCR의 효율성에 초점을 두어 궁극적으로 전체 프로세스의 효율성을 향상시킨다. 또한, 업무담당자의 업무부하를 관리하여, 낮은 수준으로 안정적인 업무부하를 유지할 수 있다. 마지막으로, 복잡한 다수 개의 프로세스에 참여하는 업무담당자들은 프로세스 목표를 이해하고 이의 효율성과 성과를 염두에 두기 때문에, 프로세스 진행에 기여하는 방향으로 업무에 참여할 것이다. 이러한 예상 효과를 다음 장에서 시뮬레이션 실험을 통해 검증한다.



〈그림 3〉 업무목록 관리

5. 시뮬레이션 실험 및 분석

BP-DBR의 효과를 검증하기 위해 시뮬레이션 실험을 실시한다. 실험을 통하여, 동일한 프로세스에 대해 BP-DBR을 적용한 프로세스 모델과 적용하지 않은 프로세스 모델을 비교, 분석한다. 본 실험에서는 두 프로세스 모델을 각각 'BP-DBR 프로세스', 'Non BP-DBR 프로세스'라 명명한다.

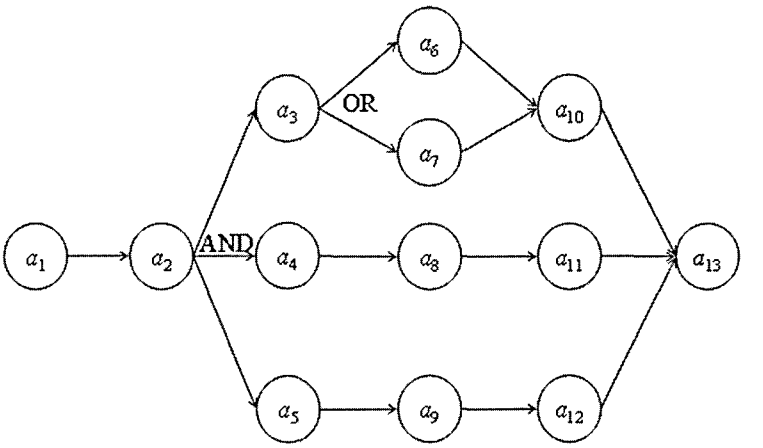
5.1 실험모델 및 환경

본 장에서는 구체적인 실험에 앞서, 사용한 프로세스 모델, 업무할당규칙과 실험환경을 설명한다. 먼저, 〈그림 4〉은 실험에서 사용한 프로세스 모델을 표현한 것이다. 실험모델은 13개의 단위업무로 구성되어 있으며, 하나의 AND 구조와 OR 구조를 가지고 있다.

한편, 위 프로세스에 참여한 업무담당자는 총 11명이며 이들은 각각 6개의 역할그룹으로 나뉘어 구성되어 있다. 아래 〈표 2〉는 역할그룹별 업무담당자의 구성을 나타낸 것이다. 실험에서 사용한 역할그룹은 주로 2명 이상으로 구성되어 있으며, 역할그룹 6에 한해서는 특정 단위업무를 전담으로 처리하는 상황을 고려하여 1명으로 지정하였다.

업무 할당 시에는 특정 단위업무를 정의시에 그룹별로 할당한 뒤, 실행시에 현 시점에서 그룹 내 최소 업무부하를 가진 업무담당자에게 할당하는 규칙을 사용한다. 〈표 3〉은 정의시에 단위업무별로 할당 받는 역할그룹을 제시한 것이다.

시뮬레이션 소프트웨어로는 Arena 6.0을 사용하였다. 단위업무의 수행시간 및 프로세스 인스턴스 발생 간격은 지수분포를 따른다고 가정하였다[5]. 이에 따라, 단위업무들의 예상 처리시간은 지 시뮬레이션 소프트웨어 Arena의 입력형식에 맞춰 expo(50)으로 동일하게 주었다. 이는 업무참여자들의 능력을 일정하게 고정시킴으로써 실험 결과에 영향을 주는 변수로 고려하지 않도록 하기 위함이다. 프로세스 인스턴스 발생간격 또한 expo(105)를 따



〈그림 4〉 프로세스 모델

〈표 2〉 역할그룹별 업무담당자 구성

역할그룹	업무담당자	역할그룹	업무담당자	역할그룹	업무담당자
1	1, 2, 3, 7	2	4, 7, 9	3	5, 6, 7, 8
4	4, 9, 10, 11	5	5, 7	6	7

〈표 3〉 단위업무별 역할그룹의 할당

단위업무	역할그룹	단위업무	역할그룹	단위업무	역할그룹
a_1	1	a_6	2	a_{11}	1
a_2	6	a_7	3	a_{12}	6
a_3	4	a_8	3	a_{13}	2
a_4	5	a_9	4		
a_5	5	a_{10}	4		

르도록 하였다. 한편, BP-CCR의 업무 버퍼(Buffer)의 기준크기(*SS*)는 실험의 경험치에 의존하여 200으로 주었다. 그리고, 제약자원을 발견하기 위한 일정 기간(*T*)은 Arena에서 replication time을 의미하며, 96,000을 그 기간으로 한다. 마지막으로 안정적인 결과를 위해 초기준비기간(warming up period)를 3,000로 주고, 실험을 30회 반복하였다.

업무할당규칙에 대해서 Non BP-DBR 프로세스는 정의시에 특정 단위업무를 처리할 후보군을 미리 정의한 뒤 실행시에 가장 업무부하가 작은 업무담당자에게 할당하는 규칙을 사용한다. 반면, BP-DBR 프로세스는 Non BP-DBR 프로세스의 방식과 *WL-TAR*을 병행 사용하여 BP-CCR의 업무부하를 중점적으로 고려한다. 마지막으로, 두 프로세스 모델에서는 업무담자들이 먼저 할당된 업무를 먼저 처리하는 선입선출 규칙을 사용한다. 이는 기존의 BPM 시스템이 업무 처리 규칙에 관한 정보를 체계적으로 제공하지 않으며, 일반적으로 선입선출을 사용하고 있기 때문이다[19].

5.2 프로세스 효율성 분석

먼저 프로세스 효율성에 대하여 BP-DBR 프로세스와 Non BP-DBR 프로세스의 결과를 비교하였다. 프로세스 효율은 프로세스 완료시간(PE1), 완료 인스턴스 개수(PE2)의 두 가지 측면을 사용하여 분석하였다. 그리고, 결과의 유의성을 검증하기 위해 유의수준 5%에서 t-검정을 수행하였다.

〈표 4〉는 BP-DBR 프로세스가 Non BP-DBR 프로세스보다 프로세스 효율성 관점에서 개선 효과가 있음을 보여준다. 프로세스 완료시간에 대해서는 36%의 개선 효과를 보였고, 완료 인스턴스 개수는 10%의 개선 효

〈표 4〉 프로세스 효율성 비교

Process Efficiency	PE1	PE2
BP-DBR 프로세스	758.30	828.27
Non BP-DBR 프로세스	1,190.95	743.79
비율(%)	36.33	10.48
t 통계량(기각치)*	8.52 (1.70)	3.69 (1.69)

* t-검정의 가설은 PE1에 대해 'H_0: (Non BP-DBR 프로세스의 PE1) - (BP-DBR 프로세스의 PE1) = 0 H_1: (Non BP-DBR 프로세스의 PE1) - (BP-DBR 프로세스의 PE1) 〉 0',
* PE2에 대해 'H_0: (BP-DBR 프로세스의 PE2) - (Non BP-DBR 프로세스의 PE2) = 0 H_1: (BP-DBR 프로세스의 PE2) - (Non BP-DBR 프로세스의 PE2) 〉 0'

〈표 5〉 업무담당자의 업무부하 비교

업무담당자	BP-DBR 프로세스	Non BP-DBR 프로세스
업무담당자 7 (8, 9)	203.55 (137.71, 137.86)	539.47 (139.20, 138.55)

과가 있음이 확인되었다. 한편, BP-DBR의 개선 효과가 통계적으로 유의함을 유의수준 5%에서 확인하였다. 이와 같은 결과는 BP-DBR의 특징으로 인해 나타난 결과이다. 즉, Non BP-DBR 프로세스에서는 BP-CCR이 지나치게 많은 업무를 적재하거나 비효율적으로 처리하기 때문에, 비제약자원들이 개인의 효율성만 증대한다고 전체 결과를 개선할 수 없다. 그러나, BP-DBR 프로세스에서는 BP-CCR에 업무부하가 편중되는 것을 방지하고 잉여 능력을 갖춘 자원으로 분산시켜주기 때문에, 프로세스 관점에서 참여자들이 협업할 수 있다. 이러한 BP-DBR의 운영 원리가 전체 프로세스의 효율성을 향상시키는 결과를 낳았다고 분석할 수 있다. 그러나, 이러한 분석은 좀 더 구체적인 프로세스 참여자의 효율성 분석을 요구한다. 다음 장에서는 각 업무담당자 관점에서 실험결과를 비교하도록 한다.

5.3 프로세스 참여자 관점의 효과 분석

프로세스에 참여하는 업무담당자의 업무부하를 비교하기 위해, 두 프로세스에서 BP-CCR 평균업무부하 크기를 측정하였다.

〈표 5〉에서 보는 바와 같이, BP-CCR의 업무부하는 BP-DBR 적용 시 절반 이상 단축됨을 볼 수 있다. 한편, 업무담당자 7과 같은 그룹에 속해 있는 업무담당자 8과 9의 업무부하는 BP-DBR 적용 전후 차이를 보이지 않

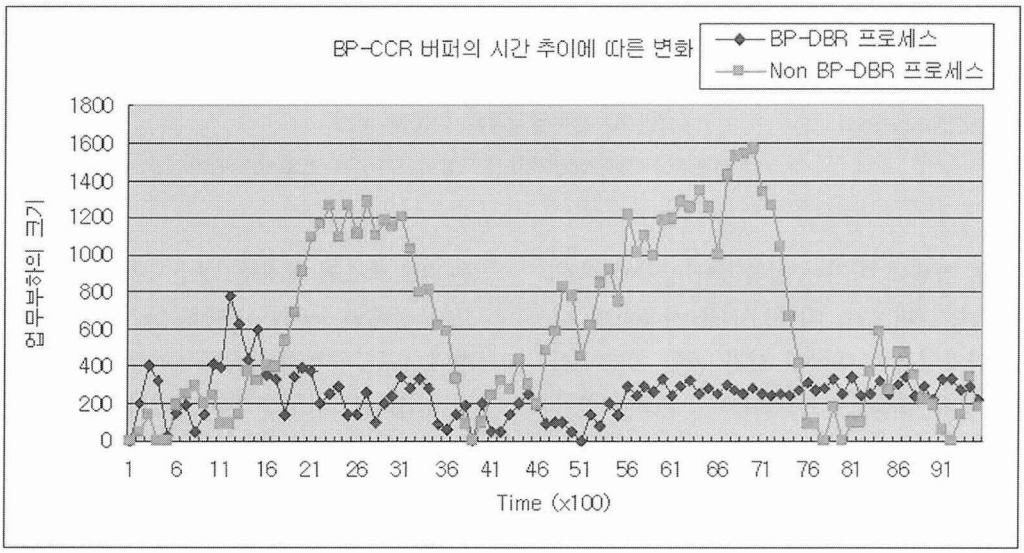
았다. 비제약자원들은 충분한 여유능력을 보유하기 때문에, BP-CCR로부터 분산된 업무부하를 담당한다고 하더라도 본래 업무능력을 초과하지 않는다. 결국, BP-DBR은 비제약자원의 업무 진행에 영향을 주지 않는 범위에서 BP-CCR의 업무부하를 감소시키는 효과를 가져다 준다.

업무담당자의 입장에서는 절대적인 업무부하와 함께 시간에 따른 업무부하의 편차도 중요하다. 이를 확인하기 위해 BP-CCR의 업무부하가 시간에 따라 어떻게 변화하는지 그 추이를 측정하였다. 〈그림 5〉는 96,000 (replication time) 동안 두 프로세스에서 수행된 결과를 보여준다.

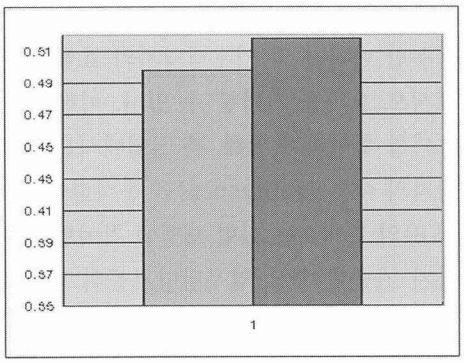
〈그림 5〉에서 보듯이, BP-DBR 프로세스에서 BP-CCR의 업무부하가 실험 기간 동안 안정적이면서, 상대적으로 작은 값을 유지하였다. BP-CCR의 업무부하가 지나치게 증가하거나, 감소하는 것을 막아주었기 때문이다.

다음으로, BP-DBR은 업무부하의 크기 관점뿐만 아니라 각 업무담당자의 효율성 관점에서도 효과를 기대할 수 있다. 이러한 기대효과를 확인하기 위해, 본 실험에서는 업무담당자의 이용률(utilization)을 분석하였다. 〈그림 6〉의 그래프는 실험 모델에 참여한 업무담당자 가운데 주요 업무담당자들의 이용률을 비교한 그래프이다. 각 그래프의 왼쪽은 본 논문이 제안한 BP-DBR 프로세스, 오른쪽은 Non BP-DBR 프로세스의 결과에 해당한다.

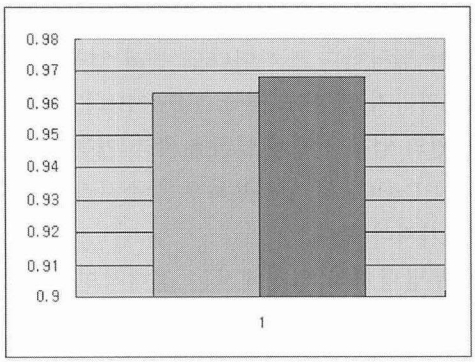
〈그림 6〉에서 보듯이, 양 프로세스 간에 업무담당자의 업무효율에는 유의한 차이가 없다. 프로세스 효율성과 비교하면, 업무담당자의 이용률이 프로세스 진행에 기여하는 정도는 BP-DBR을 사용할 때 상대적으로 높게 나타난다. 즉, 비슷한 시간 동안 일을 하고 있다고 해도 전체 프로세스 관점에서 보면 프로



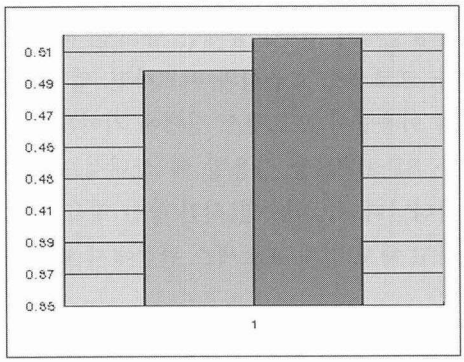
〈그림 5〉 BP-CCR의 업무부하 크기의 시간 추이



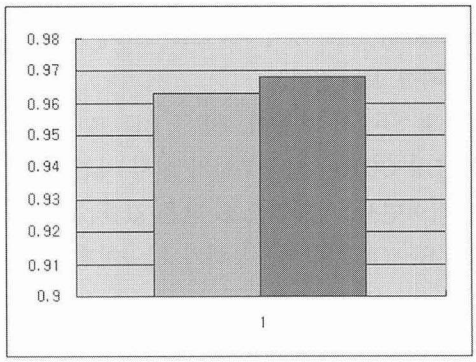
(a) Task Performer 3



(b) Task Performer 7



(c) Task Performer 8



(d) Task Performer 11

〈그림 6〉 업무담당자의 이용률* 비교

* 이용률 = 업무담당자의 업무참여시간 / 총 프로세스 운영시간(replication time)

세스의 진행에 미치는 실질적인 영향에 차이가 있다. 그러므로, BP-DBR을 사용하면 각 업무담당자들이 주어진 시간 동안 프로세스의 진행에 실질적으로 기여하는 업무에 참여함으로써, 프로세스의 효율을 높일 수 있다.

지금까지 프로세스 효율의 관점과 업무담당자 효율의 관점에서 BP-DBR을 효과를 시뮬레이션 실험을 통해 확인하였다. BP-DBR은 프로세스의 효율을 개선시켜 프로세스의 보다 신속한 처리와 정확한 납기준수를 유도한다. 또한, 안정적인 업무부하를 유지하여 보다 안정적인 업무환경을 조성하고, 업무담당자가 프로세스 관점에서 처리해야 할 단위업무를 우선 처리하게끔 한다. 결국 BP-DBR은 프로세스 참여자원들이 전체 프로세스의 공통 목표를 인식하여 이에 따라 업무에 참여

하도록 유도하는 효율적인 프로세스관리 방법이라고 할 수 있다.

6. 결론 및 추후 연구과제

본 논문에서는 TOC 개념을 적용하여 업무프로세스를 효율적으로 관리하는 BP-DBR 방법을 제안하고, 그 효과를 시뮬레이션 실험을 통해 보였다. 기존 업무프로세스관리 시스템은 기업 내 산재한 프로세스를 체계적으로 분리하고 이를 정형화하여 자동화하는데 초점을 맞추고 있기 때문에, 프로세스를 효율적인 관리하는 기능을 제공하지 않고 있고, 관련 연구 또한 없다. 본 연구에서는 생산공정에서 주로 이용되어 왔던 DBR 방법론을 업무프로세스의 환경에 맞게 변형하여 관리하는 방법인 BP-DBR을 고안하였다. BP-DBR은 프로세스 진행에 결정적인 영향을 주는 업무담당자의 업무부하를 지속적으로 관리함으로써, 발생한 인스턴스의 투입을 조절하고, 다른 업무담당자에게 업무를 분산하여 전체 프로세스의 효율성을 개선한다. BP-DBR의 여러 개선 효과는 시뮬레이션 실험을 통해서 확인할 수 있었다. 먼저, 프로세스 효율성 관점에서 프로세스 완료 시간과 완료된 인스턴스 개수에서 개선의 효과가 있었다. 두 번째로, 업무 환경 측면에서 프로세스 진행 기간 동안 업무담당자의 업무부하는 안정적으로 나타났다. 마지막으로, 업무담당자 효율성 관점에서 BP-DBR은 업무담당자가 진행에 필요한 업무에 참여하도록 하였다.

본 논문은 몇 가지 추후 보완 연구를 필요로 한다. 먼저, 제약자원을 주어진 상황에 따라 제약자원의 정의 및 특성은 달라질 수 있음을 인식하고, 각 해당 상황에 맞게 제약자원을 구체적으로 정의하는 방식을 고안해야 한다. 또한, 이러한 상황에 따른 제약자원의 특성을 고려하여 드럼을 계산하는 방식도 세분화할 필요가 있다. 그리고, 실험을 통해 인스턴스 발생 패턴의 시간에 따른 변화, 업무부하의 정도 등 BP-DBR의 효과가 발생하는 가장 적합한 환경을 구체적으로 확인해볼 필요가 있다. 이를 통해 본 연구의 방법론이 적절한 프로세스 및 환경을 발견하여 실제 효과를 증명할 필요가 있다.

참 고 문 헌

[1] 김훈태, 안동준, 강석호, "워크플로우 관리시스템을 위한 제약이론 적용방안에 대한 연구", 대한산업공학회/한국경영과학회 춘계공동학술대회 발표논문집, 2004.

[2] van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B. and Barros, A.P., "Advanced workflow patterns", Proceedings of 7th Conference on Cooperative Information Systems of Lecture notes in Computer Science, vol. 1901, 2000, pp. 18-19.

[3] Bae, H., e-Business Document Management: Change Management of Business Documents for Workflow Processes, Ph. D. Dissertation, Seoul National Univ., Seoul, South Korea, 2002.

[4] Bae, Joon-Soo, Jeong, Seok-Chan, Seo, Yeongho, Kim, Yeongho and Kang, Suk-Ho, "Integration of workflow management and simulation", Computers & Industrial Engineering, Vol. 37, No. 1-2, 1999, pp. 203-206.

[5] Chang, D., Son, J.H. and Kim, M.H., "Critical path identification in the context of a workflow", Information and Software Technology, Vol. 44, No. 7, 2002, pp. 405-417.

[6] Daniel, V. and, Guide Jr., R., "Scheduling with priority dispatching rules and drum-buffer-rope in a recoverable manufacturing system", International Journal of Production Economics, Vol. 53, No. 1, 1997, pp. 101-116.

[7] Duclos, L. K. and Spencer, Michael S., "The impact of a constraint buffer in a flow shop", International Journal of Production Economics, Vol. 42, No. 2, 1995, pp. 175-185.

[8] Goldratt, E. M. The Haystack Syndrome, North River Press, New York, 1991.

[9] Goldratt, E. M. The Goal, North River Press, New York, 1992.

[10] Goldratt, E. M. and Fox, Robert E., The Race, North River Press, New York, 1986.

[11] Hammer, M., The Agenda: What Every Business Must Do to Dominate the Decade, Three Rivers Press, 2003.

[12] Handysoft, Collaborative content management, http://www.handysoft.com, 2000.

[13] Kim, Y., Bae, J., Bae, H. and Kang, S., "Automatic Control of Workflow Processes using ECA Rules", IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 8, 2004, pp. 1010?1023.

[14] Kim, Y., Kang, S., Kim, D., Bae, J. and Ju, K., "WW-Flow: Web-based workflow management with runtime encapsulation", IEEE Internet Computing, Vol. 4, No. 3, 2000, pp. 55-64.

[15] Klusewitz, G., Rerick, R., "Constraint management through the drum-buffer-

rope system", IEEE/SEMI Advanced Semiconductor Manufacturing Conference, 1996, pp. 7-12.

[16] Lawrence, P. Workflow Handbook, Wiley, 1997.

[17] Pierce, N. G. and Yurtsever, T., "Dynamic Dispatch and Graphical Monitoring System", IEEE International Symposium on Semiconductor Manufacturing Conference Proceedings, 1999, pp. 65-68.

[18] Radovilsky, Z. D., "A quantitative approach to estimate the size of the time buffer in the theory of constraints", International Journal of Production Economics, Vol. 55, No. 2, 1998, pp. 113-119.

[19] Rhee, S. -H, Bae, H. and Kim, Y., "A dispatching rule for efficient workflow", Concurrent Engineering-Research and Applications, Vol. 12, No. 4, 2004.

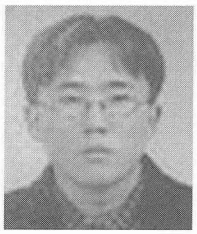
[20] Smith, H. and Fingar, P., Business Process Management - The Third Wave, Meghan-Kiffer Press, 2003.

[21] Wei, Chiu-Chi, Liu, Ping-Hung and Tsai, Ying-Chin, "Resource-constrained project management using enhanced theory of constraint", International Journal of Project Management, Vol. 20, No. 7, 2002, pp. 561-567.

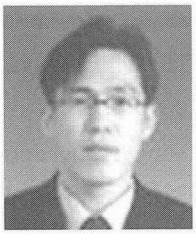
[22] WfMC-TC-1016-P, Process Definition Interchange, Workflow Management Coalition, Lighthouse Point, Fla, 1999.

[23] WFMC-TC-1011, Terminology & Glossary, Workflow Management Coalition, Lighthouse Point, Fla, 1999.

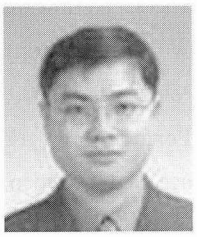
저 자 소 개



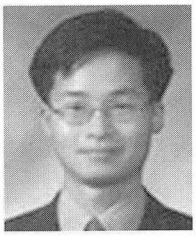
이승현 (E-mail : toy7sam@netopia.sun.ac.kr)
2001. 3 ~ 2003. 3 서울대학교 산업공학과 정보시스템 연구실
2004. 4 ~ 현재 (주)핸디소프트 BPM Professional Service팀
관심 분야 솔루션 컨설팅, 정보전략, BPR, 프로세스 분석 및 통합, 웹서비스



배혜림 (E-mail : hrbae@pusan.ac.kr)
1996. 2 서울대학교 산업공학과(학사)
1998. 2 서울대학교 산업공학과(석사)
2002. 8 서울대학교 산업공학과(박사)
2002. 8 ~ 2003. 2 삼성카드 정보기획팀
2003. 3 ~ 2004. 8 동의대학교 인터넷비즈니스학과 전임강사
2004. 9 ~ 현재 부산대학교 산업공학과 조교수
관심 분야 BPM, 다조직 프로세스 통합, XML, eAI, 웹서비스



원형준 (E-mail : hjwon@handysoft.co.kr)
1993. 2 서울대학교 산업공학과(학사)
1995. 2 서울대학교 산업공학과(석사)
1999.12 대우자동차 기술연구소 연구원
2000. ~ 현재 핸디소프트 BPM 컨설팅팀 팀장
관심 분야 Workflow, BPM, 프로세스 분석 및 통합



김훈태 (E-mail : hoontae@daejin.ac.kr)
1988. 2 서울대학교 산업공학과(학사)
1990. 2 서울대학교 산업공학과(석사)
1997. 2 서울대학교 산업공학과(박사)
1997. ~ 현재 대진대학교 산업시스템공학과 부교수
관심 분야 전자거래, 프로세스 분석 및 통합, 공급망관리

강석호 (E-mail : shkang@sun.ac.kr)
1970 서울대학교(학사)
1972 미국 University of Washington(석사)
1976 미국 Texas A&M University(박사)
1976. ~ 현재 서울대학교 산업공학과 교수
관심 분야 지능형생산시스템, 비즈니스 프로세스 관리, 정보시스템 통합