IE Interfaces
Vol. 20, No. 2, pp. 112-120, June 2007.

대한산업공학회 제2차 석사논문경진대회 수상논문

다중 프로젝트 상황에서 제품개발 업무의 동적 순서결정

강창묵 · 홍유석†

서울대학교 산업공학과

Dynamic Task Sequencing of Product Development Process in a Multi-product Environment

Chang Muk Kang · Yoo Suk Hong

Department of Industrial Engineering, Seoul National University, Seoul 151-742

As the market rapidly changes, the speed of new product development is highlighted as a critical element which determines the success of firms. While firms endeavor to accelerate the development speed, frequent iterations in a development process hinders the effort of acceleration. For this reason, many previous researches tried to find the optimal structure of the development process which minimizes the number of iterations. However, such researches have a limitation in that they can be applied to only a single-project environment. In a multi-project environment, waiting time induced by lack of resources also delays the process as well as the iterations do. In this paper, we propose dynamic sequencing method focusing on both iterations and waiting time for reducing the durations of development projects in a multi-project environment. This method reduces the waiting time by changing the sequence of development tasks according to the states of resources. While the method incurs additional iterations, they are expected to be offset by the reduced waiting time. The results of simulation show that the dynamic sequencing method dramatically improves the efficiency of a development process. Especially, the improvement is more salient as projects are more crowded and the process is more unbalanced. This method gives a new insight in researches on managing multiple development projects.

Keyword: product development project, multi-project environment, dynamic sequencing, iteration, waiting time

1. 서 론

현대 기업에 있어 속도의 향상은 가장 다급한 명제중 하나이다. 특히 신제품 개발의 속도는 신제품의 성공 여부 자체를 결정짓는 요인으로 작용한다. 이 때문에 산업계와 학계에서는 제품개발 기간을 단축시키기 위해 다양한 노력을 기울이고 있으며, 본 연구 또한 여기에 초점을 맞추고 있다.

제품개발 기간의 단축이 쉽게 이루어지지 못하는 이유는 제품개발 프로젝트에서 빈번히 발생하는 재설계 때문이다. 제품개발 프로젝트는 설계에 대한 의사결정을 내리는 여러 작업들로 이루어져있다(Smith and Eppinger, 1997; Ulich and Eppinger, 2003). 제품의 여러 부분들은 서로 유기적인 연결관계를 가지고 있기 때문에, 하나의 의사결정을 내리기 위해서는 다른 의사결정에 대한 정보가 필요하다. 이처럼 설계 작업간에 복잡하게 얽혀있는 정보의 종속성(information dependency)은 재설계(iteration)를 유발하는 원인이 된다(Ahmadi *et al.*, 2001; Eppinger *et*

본 연구는 2006년 정부(교육인적자원부)의 재원으로 학술진흥재단의 연구비지원에 의해 연구되었음(KRF-2006-331-D00680).
본 연구의 원활한 수행을 위한 서울대학교 공학연구소의 지원에 감사드립니다.
†연락저자 : 홍유석 교수, 151-742 서울시 관악구 신림동 산56-1번지 서울대학교 산업공학과, Fax : 02-889-8560, E-mail : yhong@snu.ac.kr
2007년 01월 접수, 1회 수정 후 2007년 05월 게재확정.

al., 1994; Smith and Eppinger, 1993, 1997b). 재설계가 적절히 관리되지 않으면 아무리 많은 자원을 투입한다고 해도 개발 기간의 단축은 쉽게 이룰 수 없다. Ahmadi *et al.*(2001)도 정보의 종속성으로 인한 재설계가 프로젝트의 비용과 기간을 늘리고, 비효율적인 프로세스를 만드는 주요한 원인이라고 언급한 바 있다.

때문에 이전의 많은 연구들이 제품개발 프로세스를 모델링하여 재설계를 최소화하는 프로세스를 설계하기 위하여 노력해왔다. 이러한 노력이 어느 정도 성과를 거두기는 하였으나, 단일 프로젝트 상황에서만 적용이 가능하다는 한계점 또한 지니고 있었다. 하지만 실제 기업의 개발조직에서는 다수의 개발 프로젝트가 동시에 수행되고 있으며, 무리하게 발주된 프로젝트로 인해 개별 프로젝트의 대기시간이 길어지고 효율성이 저하되는 문제를 겪고 있다(Wheelright and Clark, 1992). 본 연구는 그동안 단일 프로젝트 상황에만 초점이 맞추어져 왔던 개발 프로젝트 수행에 관한 연구를 다중 프로젝트 상황으로 확장하여, 다중 프로젝트 상황에서의 프로젝트 관리 방법론을 제안하고자 한다. 다중 프로젝트 상황에서는 재설계뿐만 아니라 자원의 부족으로 인한 대기시간이 프로젝트의 기간을 연장시키는 주요 원인으로 작용한다. 따라서 재설계와 함께 대기시간을 줄일 수 있는 방법론이 필요하다.

본 연구에서 제안하는 동적작업순서결정 방법은 다음 순서의 작업을 처리해야할 자원이 다른 프로젝트에 의해 점유되고 있는 경우, 설계작업의 순서를 바꾸어 지금 처리할 수 있는 작업부터 처리하도록 하는 프로젝트 관리 방법이다. 이를 통해 프로젝트의 대기 시간을 줄임으로써 빠른 시간안에 프로젝트를 끝마칠 수 있다. 이러한 방법론의 효과를 검증하기 위하여 시뮬레이션을 사용하였으며, 프로젝트의 효율성을 측정하기 위한 수단으로써 프로젝트들의 평균 수행시간을 사용하였다.

2. 관련연구

앞서 언급하였듯이 설계상의 재작업 관리는 제품개발 프로젝트 관리에 있어서의 핵심 요소이다. Smith and Eppinger(1993)는 재설계를 예상된 재설계와 예상치 못한 재설계로 구분하였다. 이들의 정의에 따르면 예상된 재설계는 서로 연계된 설계 작업들을 동시에 수행할 때에 필연적으로 발생하게 되는 재작업을 말한다. 설계 작업의 오버래핑(overlapping)에 대한 많은 모델들이 예상된 재설계를 가정하고 있다(Krishnan *et al.*, 1997; Roemer *et al.*, 2000; Roemer and Ahmadi, 2004). Smith and Eppinger (1997a)는 DSM(Design Structure Matrix)를 확장한 WTM(Work transformation matrix)를 개발하여 예상된 재설계를 많이 발생시키는 설계 작업을 찾아내었다.

반면에 예상치 못한 재설계는 설계를 위해 필요한 정보가 충분하지 않은 상황에서의 예측치가 이후 밝혀진 실제값과 다를 경우 이전의 설계를 수정하기 위해 발생하는 재설계를 말한다. 이러한 재설계는 확률적으로 발생하며, 설계 작업의 순서가 어떻게 이루어졌는지에 따라 발생확률과 이로 인한 재설계의 범위가 달라질 수 있다. 따라서 기존의 많은 연구들에서는 재설계를 최소화하는 설계 작업의 순서를 찾아내기 위해 노력해왔다.

Smith and Eppinger(1997b)는 정량적 DSM을 사용하여 설계작업간 종속성의 강도를 재설계의 발생확률(rework probability)로 나타내고, 이를 보상 마코프체인(reward Markov chain)으로 모델링하여 개발 프로젝트의 예상기간을 알아내는 방법론을 제안하였다. Ahmadi *et al.*(2001)은 학습효과(learning effect)를 반영하여 재설계가 반복될수록 재설계 확률이 줄어들도록 개발 프로세스를 모델링하고, 이때의 최적 작업순서를 결정하는 방법론을 개발하였다. Abdelsalam and Bao(2006)는 이러한 과정을 거치지 않고 설계 작업사이에 예상되는 재설계의 횟수를 재설계 인수로 나타내어 시뮬레이션 모델을 개발하였다. 이러한 연구들이 재설계의 발생확률만을 고려한 반면, Browning and Eppinger (2002)와 Cho and Eppinger(2006)는 재설계 확률과 함께 재설계가 발생하였을때의 양을 재설계 영향(rework impact)으로 정의하여 보다 현실적인 개발 프로세스 모델을 제안하였다.

기존의 많은 연구들이 단일 프로젝트 상황에서의 최적 프로세스를 설계하기 위하여 노력한 반면, 실제 기업에서는 다수의 개발 프로젝트가 동시에 수행되고 있다. Elonen and Artto (2003)는 다수의 개발 프로젝트를 수행하는 데 있어 문제를 발생시키는 6가지 원인을 제시하였다. 이 중 자원의 부족과 잘못된 자원의 분배는 다중 프로젝트의 관리를 어렵게 만드는 가장 중요한 원인으로 꼽힌다.

다중 프로젝트 관리에 관한 연구들 중 가장 주목받는 연구는 프로젝트를 하나의 개체(entity)로 보고 이 개체들이 지정된 순서로 작업장(workstation)을 거치는 프로세스로 프로젝트의 수행을 모델링한 Adler *et al.*(1995)의 연구이다. 이 모델을 이용하면 전통적인 프로세스 관리기법들을 프로젝트 관리에 이용할 수 있다는 장점이 있다. 이를 이용하여 Anavi-Isakow and Golany(2003)는 생산 프로세스에서의 CONWIP(constant work-in-process) 기법을 확장한 CONPIP(constant project-in-process) 기법과 CONTIP(constant time-in-process) 기법을 제안하였다. Cohen *et al.*(2005)는 이에 덧붙여 CONPIP에서의 최적 프로젝트 수를 찾는 알고리즘을 개발하였다. Narahari *et al.*(1999)는 개발 프로세스의 모델링에 보다 초점을 맞추어 대기행렬 네트워크로 개발 프로세스를 모델링하고 이를 이용하여 여러 프로젝트 관리 기법들의 유용성을 검증하였다.

이처럼 단일 프로젝트 상황에서의 작업순서 결정에 대한 문제와 다중 프로젝트 상황에서 여러 프로젝트들의 관리기법 각각에 대해서는 활발한 연구가 이루어져 왔다. 하지만 다중 프로젝트 상황에서 각각의 프로젝트들이 어떠한 작업순서로 수행되어야 하는지에 대해서는 아직까지 연구가 이루어지지 못했다.

앞서 언급하였듯이 개발 프로세스에서는 필연적으로 재설

계가 요구되기 때문에 설계 작업의 순서는 재설계를 최소화하기 위한 의사결정의 결과물일 뿐이다. 만약 다중 프로젝트 상황에서는 단일 프로젝트 상황에서 얻어진 최적 작업순서가 오히려 프로젝트의 기간을 길어지게 만든다면, 이를 개선하여 개발 프로세스의 효율화를 꾀할 수 있을 것이다.

실제로 다중 프로젝트 상황에서는 병목작업에서 긴 대기시간이 발생하기 때문에 작업 순서의 변화를 통해 대기시간을 줄이게 되면 프로젝트의 완료 시점을 단축시킬 수 있을 것이다. 본 연구에서는 이러한 작업 순서의 변화가 실제로 다중 프로젝트 상황에서 어떠한 효과를 나타내는지를 시뮬레이션을 통해 분석하고, 프로젝트 관리에 있어서의 활용방안을 논의하고자 한다.

3. 개발 프로세스 모델링

개발 프로젝트에서의 작업 순서의 결정 문제를 풀기 위해서는 개발 프로세스에 대한 모델이 필요하다. 본 연구에서는 Adler *et al.*(1995)가 제안한 프로세스 관점에서의 모델을 채용하여 개발 프로세스를 모델링하였다. 또한 DSM 모델을 변형하여 다양한 프로세스 구조 대안들을 비교할 수 있도록 하였다.

3.1 모델링 가정

3.1.1 확률적 네트워크 가정

Adler *et al.*(1995)는 개발 프로세스를 개별적인 프로젝트들이 다수의 워크스테이션을 확률적으로 거치게 되는 네트워크로 가정하였다. 각각의 워크스테이션에서는 작업자 혹은 자원을 이용하여 정해진 작업을 수행하게 된다. 이러한 가정하에서는 네트워크 내에 여러개의 프로젝트가 존재하기 때문에 다중 프로젝트 상황에서의 개발 프로세스를 모델링하는데에 적합한 가정이라고 할 수 있다. 추가로, 본 연구에서는 모든 프로젝트가 동일한 작업과 프로세스 구조를 가진다고 가정하였다.

3.1.2 순차적 작업 수행 가정

본 연구에서는 모든 작업이 순차적으로 이루어진다고 가정하였다. 여러 작업을 병렬적으로 수행하는 것은 프로젝트의 기간을 단축시키기 위하여 일반적으로 사용되는 방법이지만, 이에 대한 연구는 본 연구에서 논의하고자 하는 범위를 벗어나기 때문에 본 연구에서는 다루지 않는다.

3.1.3 유연 종속성 가정

Ahmadi *et al.*(2001)은 작업간 정보의 종속성을 유연한 종속성(soft dependency)와 엄격한 종속성(hard dependency)로 나누었다. 엄격한 종속성하에서는 종속된 작업이 반드시 종속하는 작업의 이후에 수행되어야 하지만, 유연한 종속성하에서는 작업의 선후관계가 단지 재작업의 가능성이나 영향 등에 의해서만 결정된다. 본 연구에서는 모든 종속성이 유연한 종속성이라고 가정하여 작업간의 순서변화가 언제나 가능하다고 가정하였다. 다음 절에서 밝히겠지만 이러한 유연한 종속성만을 가지고도 엄격한 종속성 관계까지 표현이 가능하다.

3.2 DSM 기반 재설계 모델

본 연구에서는 Browning and Eppinger(2002)가 제안한 대로 재설계의 발생 확률과 재설계의 영향을 나타내는 두개의 DSM 기반 매트릭스 ***RP***와 ***RI***로 재설계를 모델링하였다. 단 본 연구는 작업순서의 변화가 프로세스에 끼치는 영향을 알아보기 위한 것이므로 특정한 작업순서를 가정하고 작성된 Browning and Eppinger(2002)의 매트릭스를 수정하여 사용하였다.

재설계 확률 매트릭스 ***RP***의 (i, j)원소는 설계단위 i가 먼저 설계되었을 때 후행하는 설계단위 j에 대한 설계 결과 때문에 설계단위 i를 재설계하게 될 확률을 나타낸다. 재설계 영향 매트릭스 ***RI***의 원소들은 이러한 재설계가 발생하였을 때 재설계 작업의 양을 원래 설계 작업량에 대한 비율로써 나타내게 된다.

<Figure 1>에서와 같은 ***RP***와 ***RI***가 주어졌을 경우, RP_{AB}= 0.5, RI_{AB} =0.7이므로 설계단위 A를 먼저 설계하고, B를 나중에 설계하게 되면 0.5의 확률로 0.7만큼의 A에 대한 설계작업을 반복해야 한다.

앞서도 언급하였듯이 이러한 모델링 방법은 유연한 종속성의 가정에 기반하고 있다. 하지만 엄격한 종속성 또한 이 모델링 방법을 통해 표현할 수 있다. 엄격한 종속성이 존재하는 두 작업이 있을 경우 하나의 작업은 반드시 다른 작업에 앞서 수행되어야 한다. 만약 이러한 순서가 바뀐다면, 앞서 한일이 거의 모두 쓸모없게 되어 다시 작업을 수행해야 하는 일이 발생할 것이다. 이는 곧, 엄격한 종속성 하에서 뒤에 수행되어야 할 설계작업이 앞서서 수행될 때 그 설계작업에 대한 재설계의 확률과 영향이 1에 가깝다는 의미로 해석될 수 있다. 즉, 본 연구에서 제안한 모델에서 재설계의 확률과 영향을 1에 가깝게 설정하게 되면 엄격한 종속성까지도 표현할 수 있게 되는 것이다.

3.3 재설계 특성의 동적 갱신

개발 프로세스 모델링에 있어서의 또 다른 문제는 재설계 확률과 영향 등의 특성을 동적으로 갱신하는 문제이다. 실제 설계 상황에서는 재설계를 반복할수록 그 발생확률이 점차 줄어드는 것이 일반적이다. Browning and Eppinger(2002)는 이러한 현상의 원인이 학습효과에 있다고 보고, 학습효과 계수를 산정하여 이를 프로세스 모델링에 반영하였다. 하지만 학습효과 계수라는 것이 다분히 주관적인 판단에 의해 주어지는 것이기 때문에 이 값에 대한 논리적인 근거를 마련하기 어렵다.

따라서 본 연구에서는 다른 방식으로 이 문제에 대해 접근하였다. 오버래핑(overlapping)에 관한 기존의 연구들에서 오버

래핑시 발생하게 되는 재작업의 양을 결정하기 위해 사용한 논리를 차용하는 것이다. 오버래핑시에 이후단계의 작업은 이전단계 작업의 결과를 필요로 함에도 이를 모르는 상태에서 작업을 하기 때문에, 차후에 이전단계의 작업이 완료되고 나면 그 내용을 이후단계의 작업에 다시 반영하는 과정이 필요하다. 이러한 이후단계 작업에 대한 재작업은 피할 수 없는 것으로 여겨지며, 그 양은 오버래핑된 양, 즉 이전단계 작업의 결과를 모르고 행해진 이후작업의 양에 비례한다. 따라서 <Figure 2>와 같이 재작업의 양 $h(y_i)$는 오버래핑된 작업의 양 y_i에 대한 함수로 나타나게 된다.

RP	A	B
A	-	0.5
B	0.8	-

RI	A	B
A	-	0.7
B	0.6	-

Figure 1. RP and RI

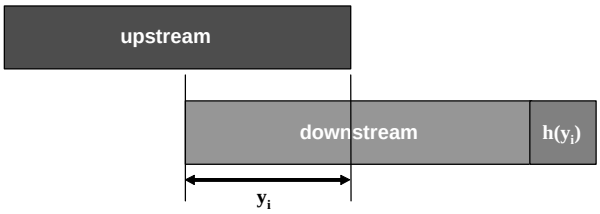

Figure 2. Rework of overlapping

그런데 순차적인 프로세스 상에서 발생하는 재작업 또한 오버래핑에 의해 발생하는 재작업과 그 발생원인이 동일하다. 순차적인 프로세스에서도 종속하는 설계작업이 이루어지지 않아 그 결과를 모르는 상태에서 종속된 설계작업을 수행함으로써, 종속하는 설계작업이 이루어진 후에 이전의 설계작업을 반복하게 되는 것이다. 따라서 순차적인 프로세스에서도 재설계의 확률과 영향이 이후에 이루어지는 설계작업의 양에 비례해야 할 것이다.

<Figure 3>에서 두 작업 A와 B는 <Figure 1>과 같은 ***RP***와 ***RI***를 가지고 있다고 가정하자. A가 먼저 설계되었으므로 B가 설계된 이후에 A가 재설계될 확률은 0.5이고 이때, 재설계의 양은 원래의 0.7만큼이다. 이 0.7A만큼의 재설계는 B를 처음 설계하던 시점에서는 알 수 없는 정보였다. 따라서 이제는 B가 A보다 먼저 설계된 셈이 된다. 이 경우 B가 재설계될 확률은 0.8인데 A가 완전히 재설계된 것이 아니라 0.7만큼 재설계된 것이므로 B가 다시 재설계될 확률은 0.7*0.8, 즉 0.56이 된다. 재설계 영향 또한 마찬가지 방식으로 계산하게 되면 0.42(=0.7*0.6)가 된다. 이러한 논리에 따르면, RP와 RI의 초기값이 RP^0와 RI^0이고, 설계작업 j에 의한 설계작업 i의 재작업양이 RI_{ij}일 때, RP_{xi}와 RI_{xi}는 $RI_{ij}\cdot RP^0_{xi}$와 $RI_{ij}\cdot RI^0_{xi}$로 갱신된다.

두개 이상의 설계작업이 연관되어 있을 때에는 또다른 문제가 제기된다. <Figure 4>와 같이 A, B, C 세 작업이 서로 연관되어 있고, 설계작업 C에 의해 설계작업 A와 B가 재작업되어야 한다고 하자. 만약 이 상황에서 설계작업 A의 재설계가 B의 재설계보다 먼저 이루어진다면, B의 재설계량은 C에 의한 양인 0.5와 A에 의한 양인 0.3 중 하나가 될 것이다. 본 연구에서는 이러한 상황에서 둘 중 더 큰 쪽을 선택하는 것으로 가정하였다.

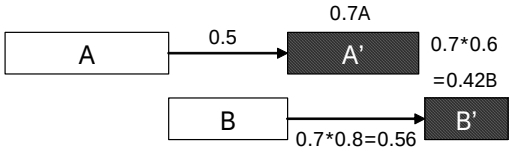

Figure 3. Updates of rework characteristics

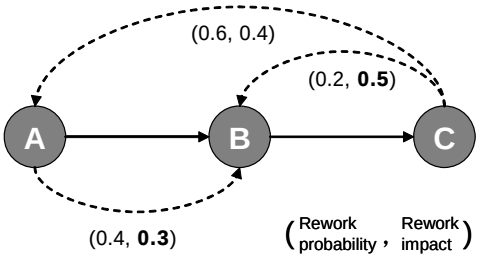

Figure 4. Rework of three tasks

이상의 원리에 따라 RP와 RI를 갱신하면 초기의 재설계 확률 및 영향에 대한 정보와 실제 재설계의 발생여부만을 가지고도 재설계의 특성을 동적으로 갱신해 갈 수 있다.

4. 다중 프로젝트 상황에서의 동적 순서결정

앞서 설명한 개발 프로세스 모델에 따라 개발 프로젝트의 수행을 시뮬레이션 해 보면, 프로젝트의 기간을 늘리는 주요 원인이 병목 자원의 부족으로 인한 대기시간임을 알 수 있다. 대기시간은 프로젝트가 자주 발주될수록 점점 더 길어져 프로젝트 기간이 무한정 길어지는 상황에까지 이르게 한다.

하지만 프로젝트들이 병목 작업의 수행을 위해 기다리지 않고 자원이 넉넉한 다른 설계작업부터 먼저 수행할 수 있다면 대기시간이 현저히 줄어들 것이다. 이러한 방식은 재료의 물리적인 변화가 요구되는 전통적인 생산 프로세스에서는 불가능한 것이었다. 하지만 개발 프로세스에서의 작업 순서는 정해져 있다기 보다는 재설계를 최소화하는 방향으로 선택되는 사항이기 때문에 프로젝트를 수행하는 도중에도 순서가 바뀔 수 있다. 우리는 이를 제품개발 업무의 동적 순서결정(dynamic sequencing) 방법이라고 명명하였다.

동적 순서결정 방법이 대기시간을 줄일 수 있을 것이라는 것은 상식적인 선에서도 충분히 예상할 수 있는 것이지만, 전체적인 프로젝트 기간의 관점에서 보면 항상 더 좋은 결과를 보여준다고 장담할 수는 없다. 작업순서의 변화가 대기시간을 단축시키기는 하지만 또한 프로세스에 있어서의 추가적인 재설계 발생시키기 때문이다. 추가적인 재설계는 개별 프로젝트의 기간을 늘릴 뿐만 아니라 자원을 추가적으로 소모하게 함으로써 다른 프로젝트들의 대기시간까지 늘리는 결과를 가져온다.

따라서 본 연구에서는 동적 순서결정 방식이 기존의 정적인 작업순서를 따라가는(static sequencing) 방식에 비해 얼마나 더 좋은 성능을 보이는 지를 시뮬레이션을 통해 검증해보고자 한다. 본 연구에서 사용하는 동적 순서결정 방법은 다음 작업을 수행할 자원을 다른 프로젝트에서 먼저 점유하고 있을 경우 이를 수행하지 않고 그 이후의 작업들 중 바로 수행이 가능한 작업부터 수행하는 방식으로 작업순서를 할당하게 된다. 이외에도 다양한 작업순서 결정 알고리즘이 존재할 수 있고, 그에 따라 방법론의 성능 또한 달라지겠지만, 이는 본 연구의 범위를 벗어나는 문제이기 때문에 추후 연구에서 다루도록 한다.

5. 시뮬레이션을 통한 검증

Narahari *et al.*(1999)는 대기행렬 네트워크 모델을 이용하여 다중 프로젝트 상황에서의 프로젝트 기간을 수리적으로 예측하였다. 하지만 본 연구에서 사용하는 프로세스 모델에서는 재설계의 확률과 영향이 동적으로 변화하고 작업의 순서 또한 자원의 상황에 따라 달라지기 때문에 이러한 수리적 모델을 사용할 수 없다. 따라서 본 연구에서는 시뮬레이션을 이용하여 동적 순서결정 방법의 효과를 검증하고자 한다.

5.1 사례연구

본 연구에서는 광학 마우스의 개발 프로세스를 대상으로 하여 시뮬레이션을 수행하였다. 개발 프로세스 상의 설계작업들과 이들의 예상 수행시간은 <Table 1>과 같이 주어진다. 모든 작업들은 각각 독립적인 자원들에 의해 수행되며, 작업시간은 불확실성을 방영하기 위해 예상 수행시간에서 ±20%의 최대, 최소값을 가지는 삼각분포를 가정하였다.

<Figure 5>와 <Figure 6>은 재설계 확률과 영향을 나타내는 RP, RI 메트릭스이다. 앞서 언급하였듯이 재설계 확률은 행에 위치한 설계작업을 먼저 수행하였을 때 열에 위치한 설계작업의 결과에 의해 행의 작업을 재설계하게 될 확률이고, 재설계 영향은 그에 따른 재설계 작업의 양을 나타낸다.

이러한 정보들은 보통의 설계 프로젝트에서 일반적으로 얻을 수 있는 정보들이기 때문에 꼭 마우스가 아닌 다른 제품의 개발 프로세스에서도 같은 방식의 분석이 가능하다.

Table 1. Development tasks and expected durations

No.	Development task	Exp. duration
1	Connection cable design	2
2	Top case design	10
3	Bottom case design	8
4	Wheel mechanism design	8
5	Main board design	3
6	Microprocessor design	9
7	Click mechanism design	7
8	Optical sensor design	2
9	Balancing weight design	7

		1	2	3	4	5	6	7	8	9
Connection cable	1		1	1			1			
Top case	2			0.1	0.5	0.4				
Bottom case	3		0.1				0.1			0.8
Wheel mechanism	4		1.0	0.2		0.8			0.1	
Main board	5				0.2		0.8			
Microprocessor	6	0.1		0.7		0.3		1.0		1.0
Click mechanism	7						0.1			
Balance weight	8			0.1						
Optical sensor	9			0.1			0.1			

Figure 5. Rework probability of optical mouse development tasks (***RP***)

		1	2	3	4	5	6	7	8	9
Connection cable	1		0.5	0.5			0.5			
Top case	2			0.1	0.5	0.3				
Bottom case	3		0.1				0.8			0.5
Wheel mechanism	4		0.8	0.2		0.2			0.1	
Main board	5				0.4		0.1			
Microprocessor	6	0.2		0.9		0.1		0.8		0.5
Click mechanism	7						0.1			
Balance weight	8			0.1						
Optical sensor	9			0.5			0.5			

Figure 6. Rework impact of optical mouse development tasks (***RI***)

5.2 최적 작업순서의 탐색

동적으로 작업순서를 변경하는 방식의 효과를 검증하기 위해서는 재설계를 최소화하는 최적 작업순서를 정적으로 따라갈 때와의 성능 차이를 비교해야 한다.

최적 작업순서를 결정하는 문제가 외판원 문제(traveling salesman problem)으로 변환된다는 것이 자명하므로, 이 문제는 NP- Hard 문제라고 할 수 있다. 따라서 유전 알고리즘을 이용하여 최적 작업순서를 찾았다. 하나의 작업순서를 유전자로 설정하였으며 그 밖의 유전알고리즘 파라미터들은 <Table 2>와 같다.

Table 2. Parameters of genetic algorithm

Parameter	Value
Number of populations	10
Population size	100
Reproduction ratio	10%
Crossover ratio	89%
Mutation ratio	1%
Crossover method	PMX method[a]
Stopping criterion	|Improvement|<0.2

[a]Murty(1995)

Table 3. Optimal and worst task sequences

Optimal sequence	5, 9, 3, 2, 7, 6, 4, 8, 1
Expected duration	59.074
Standard deviation	4.478
Worst sequence	1, 6, 8, 3, 9, 7, 4, 5, 2
Expected duration	108.099
Standard deviation	13.205

이를 이용하여 최적의 작업순서와 비효율적인 작업순서를 찾을 수 있었다. 비효율적인 작업순서는 예상 프로젝트 수행시간을 최대로 만드는 작업순서로서 <Table 3>과 같이 최적의 작업순서에 비해 2배 가까이 긴 수행시간을 나타내었다.

5.3 동적 순서결정 방법의 효과

모든 프로젝트가 최적의 작업순서를 따라 개발을 진행시켜 나가는 방식과 동적으로 작업순서를 변경하는 방식은 프로젝트의 수행시간 측면에서 큰 차이를 보였다. <Figure 7>은 평균 프로젝트 발생 간격을 변화시키면서 20,000개의 프로젝트를 발생시켰을 때, 두 관리 방식에 따른 평균 프로젝트 수행시간을 나타낸 것이다. 발생간격이 13일 때 동적 순서결정 방법이 37.1% 더 짧은 평균 프로젝트 기간을 나타내었고, 이때 수행시간의 표준편차도 44.5% 줄어들었다. 따라서 동적 순서결정 방법이 더 빠르고 안정적인 프로젝트 수행을 도모한다는 것을 알 수 있었다.

또 하나, 프로젝트 기간의 우위는 발생시간 간격이 길어질수록 점점 줄어들다가 발생간격 16에서 통계적으로 유의미한 차이가 사라지고, 그 이후로는 오히려 정적 작업순서를 따르는 것이 더 좋은 것으로 나타났다. 이는 발생간격이 길어질수록 대기시간이 줄어들어 동적인 작업순서 변경을 통해 줄어드는 대기시간이 이로 인해 늘어나는 재설계 시간을 상쇄하지 못하기 때문이라고 생각할 수 있다.

이처럼 프로젝트의 수행시간이 짧아지는 원인은 대기시간이 줄어들었기 때문이다. <Figure 8>을 보면 평균 프로젝트 발생간격이 13일때 동적 순서결정 방법을 사용하면 평균 대기시간이 50.4% 감소하는 것을 볼 수 있다. 그리고 이러한 감소폭은 프로젝트 발생간격이 길어지면서 점점 줄어들기는 하나 정적인 작업순서를 따라가는 것에 비해서는 항상 짧은 대기시간을 보였다.

이처럼 대기시간이 줄어들 수 있었던 원인은 동적 순서결정 방법이 병목자원에 집중되면 부하를 여러 자원들로 분배시켰기 때문이다. <Figure 9>를 보면 정적 작업순서에서는 자원 2에 평균 16개의 프로젝트가 대기하고 있던 것에 비해 동적 순서결정에서는 평균 대기 프로젝트 수가 6.6개로 줄어들고, 자원 2의 부하가 자원 6으로 많이 분배된 것을 알 수 있다. 이로부터 동적 순서결정 방법이 개발 프로세스 내의 자원 분배를 원활히 하여 프로세스의 효율성을 향상시킨다는 것을 알 수 있었다.

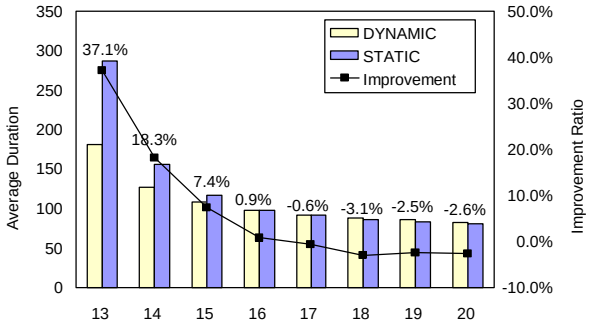

Figure 7. Comparison of average durations

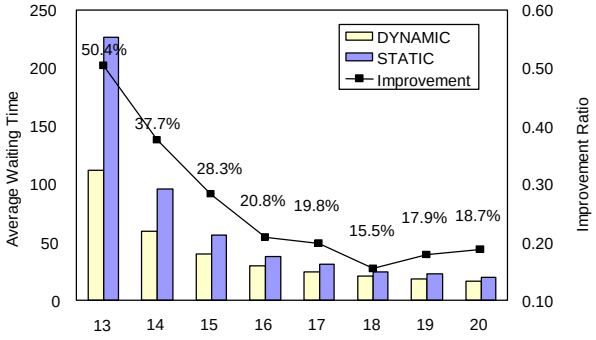

Figure 8. Comparison of waiting times

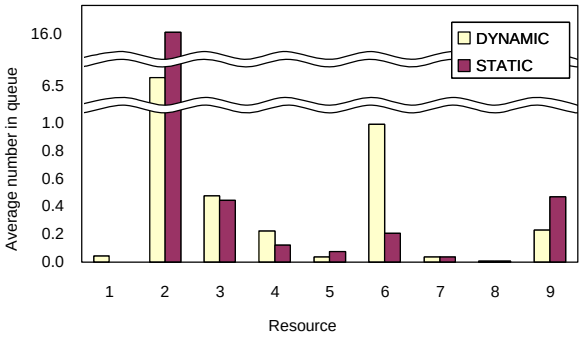

Figure 9. Average number of waiting projects

5.4 균형 프로세스와 비균형 프로세스

동적 순서결정 방법은 병목작업에서의 대기시간을 줄여 전체적인 프로젝트의 기간을 줄이는 방법이다. 이러한 관점에서 보면 동적 순서결정 방법의 효과는 프로세스가 균형을 이룬 정도에 따라 달라질 것이다. 이를 알아보기 위해 완전히 균형을 이룬 가상의 프로세스를 가정하였다. 이 프로세스의 전체 기간은 광학 마우스와 같은 54이지만, 모든 설계작업이 6씩 동일한 기간을 가진다. 이 때 최적 작업순서는(8, 7, 5, 9, 2, 3, 4, 6, 1)이 된다.

<Figure 10>을 보면 균형 프로세스에서는 동적 작업순서가 정적 작업순서에 비해 오히려 더 긴 프로젝트 기간을 나타내는 것을 알 수 있다. 즉 균형 프로세스에서는 특별한 병목작업이 없기 때문에 대기시간이 줄어든 효과가 감소된 것이다. 이 같은 현상은 <Figure 11>을 보면 보다 확실히 알 수 있다. 동적 순서결정에서는 정적 작업순서에 비해 전 자원의 이용률(utilization)이 전반적으로 높아진 것을 볼 수 있는데, 동적 작업순서가 대기시간은 거의 감소시키지 못하면서 재설계만을 증가시켜 전체적으로 자원의 소비가 많아졌기 때문이다.

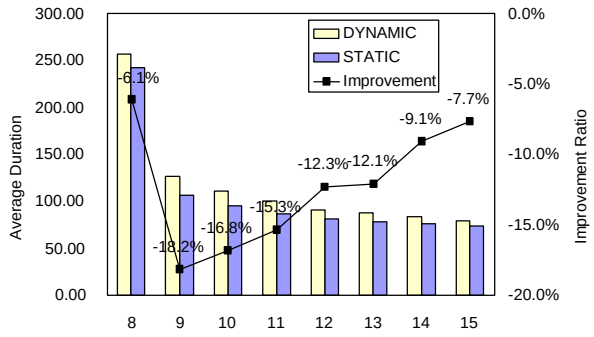

Figure 10. Comparison of average durations in a balanced process

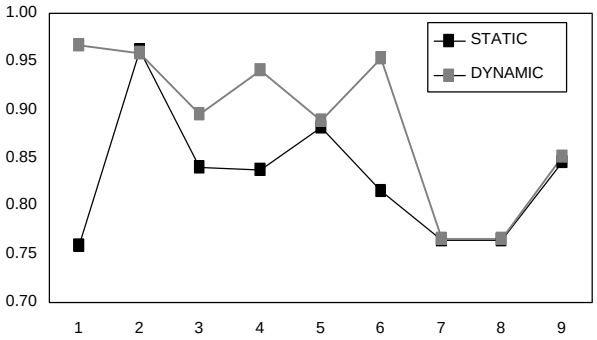

Figure 11. Comparison of utilizations in a balanced process

6. 시사점

이상의 모델링 과정과 시뮬레이션의 결과로부터 다음과 같은 시사점들을 찾을 수 있었다.

6.1 동적 순서결정 방법의 효과

<Figure 7>과 <Figure 8>을 보면 정적 작업순서에서 대기시간이 프로세스 전체 시간의 대부분을 차지하는 것을 알 수 있다. 따라서 동적 작업순서를 사용함으로써 재설계가 추가로 발생하더라도 줄어든 대기시간이 이를 상쇄할 수 있기 때문에 동적 순서결정 방법이 전체 프로젝트 기간을 단축시킨다.

그리고 그래프 상에는 나타나지 않았지만 발생 간격이 더 짧아졌을 때에도 동적 작업순서를 사용하였을 때 프로젝트 기간이 길어지는 속도가 훨씬 느리게 나타났다. 이를 통해 동적 순서결정 방법이 다중 프로젝트 상황에서 효과적이라는 것을 알 수 있다.

6.2 작업순서의 영향

<Table 3>을 보면 단일 프로젝트 상황에서는 비효율적인 작업순서를 사용했을 때 최적의 작업순서를 사용했을때 보다 2배 정도 긴 프로젝트 기간을 나타내었다. 하지만 다중 프로젝트 상황에서는 비효율적인 작업순서를 사용하였을 때 병목작업의 작업시간인 10보다 3배 이상 긴 36의 평균 발생시간 이전에는 프로젝트 수행시간이 무한히 증가하였다. 이를 통해 다중 프로젝트 상황에서 프로젝트의 작업순서가 훨씬 더 중요한 역할 한다는 것을 알 수 있다. 이러한 현상이 발생하는 이유는 추가적으로 발생되는 재설계가 하나의 프로젝트의 기간을 늘릴 뿐만 아니라 자원의 추가적인 소모를 통해 다른 프로젝트들의 대기 시간 또한 늘리는 결과를 가져오기 때문이다.

앞서도 언급하였듯이 최적의 작업순서를 결정하는 문제는 NP-Hard 문제이기 때문에 이를 찾아내는 쉽지 않다. 하지만 동적으로 작업 순서를 할당하면 자원의 상황에 따라 적절한 순서를 선택하게 되기 때문에 부적합한 작업순서를 선택하게 되는 위험을 피할 수 있다.

6.3 효율적 균형화(balancing) 방법

프로세스의 균형화는 프로세스의 효율성을 향상시키는 대표적인 방법이다. 하지만 이를 위해서는 병목작업에 추가적인 자원을 투입해야 하기 때문에 추가적인 비용을 필요로 한다. 동적 순서결정 방법은 병목작업에 집중된 부하를 여러 작업들로 분산시키기 때문에 이러한 균형화 방법의 하나라고 할 수 있다. 이 때문에 프로세스가 균형화 되어있지 않을수록 동적 작업순서의 효과가 커지는 것이다. 하지만 여기에는 추가적인 자원 투입이 필요하지 않기 때문에 효율적으로 프로세스를 균형화할 수 있다.

6.4 작업순서 결정 알고리즘의 개발

동적 순서결정 방법의 효과를 향상시키기 위해서는 보다 효

율적인 순서 결정 방법이 필요하다. 이러한 방법을 개발하는 데 있어 가장 중요한 요소는 작업순서 변경으로 인해 발생하는 추가적인 재설계의 최소화이다. 원론적으로는 순서를 변경함으로써 야기되는 재설계에 걸리는 시간이 순서 변경을 통해 줄일 수 있는 대기시간보다 길다면 순서를 변경하지 말아야 할 것이다. 순서를 변경할 때에도 재설계를 최소화하는 대안을 선택해야 한다.

순서 결정 방법을 개발하기 위해서는 추가적으로 발생하는 재설계를 예측할 수 있는 수리적인 모델이 필요하다. 본 연구에서는 시뮬레이션을 이용해 이러한 재설계를 예측하였는데, 여기에는 많은 시간이 소요되기 때문에 현실적으로 사용하기 힘들다. 수리적인 재설계 예측 모델의 개발과 작업순서 결정 방법의 개발은 향후 연구 과제이다.

7. 결 론

개발 프로젝트에 있어서의 재설계는 개발 프로젝트의 지연과 비용 초과의 주요 원인이다. 많은 기존의 연구들이 재설계와 프로젝트의 기간을 최소화하는 방법론을 개발하고자 하였다. 이러한 노력들은 단일 프로젝트를 최적화하는 데 있어서 어느 정도 성공적이었다.

하지만 본 연구는 다중 프로젝트 상황에서의 프로젝트 관리에 초점을 맞추고 있다. 단일 프로젝트 상황에서는 자원의 부족으로 인한 대기시간이 발생하지 않지만 다중 프로젝트 상황에서는 이로 이한 대기시간이 발생하게 되고 이러한 대기시간이 재설계로 인한 추가적인 시간보다도 커지게 된다. 따라서 본 연구에서는 프로젝트의 작업순서를 동적으로 변경함으로써 대기시간을 줄이는 방법을 제안하고 이 효과를 시뮬레이션을 이용하여 검증하였다.

본 연구에서 사용된 프로세스 모델은 기존의 연구에서 사용한 DSM 기반 모델들에 기초를 두고 있지만, 재설계의 특성을 갱신하는 측면에서 차이가 있다. 기존의 연구들에서와는 다르게 본 연구에서는 현재 설계작업의 양에 따라 재설계의 확률과 영향을 갱신한다.

이 모델을 사용하여 동적 순서결정 방법이 가지는 이점을 검증하였다. 프로젝트 수행에 대한 성능척도로는 프로젝트의 평균 수행시간을 사용하였으며, 시뮬레이션을 통해 각 방법에서의 기대 프로젝트 수행시간, 시스템 이용률, 대기 시간 등을 예측하였다. 동적 작업순서는 프로젝트의 발생 간격이 짧을 때 정적 작업순서에 비해 짧은 프로젝트 기간을 나타냈다. 이는 동적 순서결정 방법이 일종의 프로세스 균형화 방법으로써 병목작업에 걸리는 부하를 다른 작업들로 배분하기 때문이다.

본 연구는 다중 프로젝트 상황에서 동적 순서결정 방식이 정적 작업순서에 비해 효과적이라는 것을 보이기 위한 것이다. 따라서 향후 연구에서는 효과적인 순서결정 방법을 개발해야 할 것이다. 이를 위해서는 본 연구에서 이용한 시뮬레이션 모델이 아니라 수리적인 모델을 이용해 재설계를 예측할 수 있어야 한다. 또한 예상 프로젝트의 기간뿐만 아니라 비용, 자원 이용률 등 다양한 기준에서의 최적화 방법의 개발이 필요하다.

참고문헌

Abdelsalam, H. and Bao, H. (2006), A simulation-based optimization framework for product development cycle time reduction, *IEEE Transactions on Engineering Management*, **53**(1), 69-85.

Adler, P., Mandelbaum, A., Nguyen, V., and Schwerer, E. (1995), From project to process management: An empirically-based framework for analyzing product development time, *Management Science*, **41**(3), 458-484.

Ahmadi, R., Roemer, T., and Wang, R. (2001), Structuring product development processes, *European Journal of Operational Research*, **130**(3), 539-558.

Anavi-Isakow, S. and Golany B. (2003), Managing multi-project environments through constant work-in-process, *International Journal of Project Management*, **21**(1), 9-18.

Browning, T. and Eppinger, S. (2002), Modeling impats of process architecture on cost and schedule risk in product development, *IEEE Transactions on Engineering Management*, **49**(4), 428-442.

Cho, S. and Eppinger, S. (2006), A simulation-based process model for managing complex design design projects, *IEEE Transactions on Engineering management*, **52**(3), 316-328.

Cohen, I., Golany, B. and Shtub, A. (2005), Managing stochastic, finite capacity, multi-project systems through the cross-entropy methodology, *Annals of Operations Research*, **134**(1), 183-199.

Elonen, S. and Artto K. (2003), Problems in managing internal development projects in multi-project environments. *International Journal of Project Management*, **21**(6), 395-402.

Eppinger, S., Whitney, D., Smith R., and Gebala, D. (1994), A model-based method for organizing tasks in product development, *Research in Engineering Design*, **6**(1), 1-13.

Krishnan, V., Eppinger, S., and Whitney, D. (1997), A model-based framework to overlap product development activities, *Management Science*, **43**(4), 437-451.

Murty, K. (1995), *Operations research: Deterministic optimization models*, Prentice Hall, Englewood Cliffs.

Narahari, Y., Viswanadham, N., and Kumar, V. (1999), Lead time modeling and acceleration of product design and development. *IEEE Transactions on robotics and automation*, **15**(5), 882-896.

Roemer, T., Ahmadi R., and Wang, R. (2000), Time-cost trade-offs in overlapped product development, *Operations Research*, **48**(6), 858-865.

Roemer, T. and Ahmadi R. (2004), Concurrent crashing and overlapping in product development, *Operations Research*, **52**(4), 606-622.

Smith, R. and Eppinger, S. (1993), Charateristics and models of iteration in engineering design, *International Conference on Engineering Design*, The Hague, Netherlands, 17-19, August.

Smith, R. and Eppinger, S. (1997a), Identifying controlling featues fo engineering design iteration, *Management Science*, **43**(3), 276-293.

Smith, R. and Eppinger, S. (1997b), A predictive model of sequential iteration in engineering design, *Management Science*, **43**(8), 1104-1120.

Ulich, K. and Eppinger, S. (2003), *Product design and development*, McGraw-Hill, New York.

Wheelwright, S. and Clark, K. (1992), Creating project plans to focus product development, *Harvard Business Review*, **70**(2), 70-82.

강 창 묵
서울대학교 산업공학과 학사
현재: 서울대학교 산업공학과 박사과정
관심분야: 플랫폼 개발, 개발 프로세스 효율화

홍 유 석
서울대학교 산업공학과 학사
서울대학교 산업공학과 석사
Purdue University 산업공학부 박사
현재: 서울대학교 산업공학과 조교수
관심분야: 제품공학, 제품 아키텍처 분석 및 설계, 지속가능 제품개발, 서비스 설계 및 개발