공학석사학위논문

실현 가능한 설계 규칙의 수립을 위한 선제적 설계 방법론

Preemptive Design for Establishing Feasible Design Rule

2020 년 2 월

서울대학교 대학원

산업공학과

임 종 욱

실현 가능한 설계 규칙의 수립을 위한 선제적 설계 방법론

Preemptive Design for Establishing Feasible Design Rule

지도교수 홍 유 석

이 논문을 공학석사 학위논문으로 제출함

2019 년 12 월

서울대학교 대학원

산업공학과

임 종 욱

임종욱의 공학석사 학위논문을 인준함

2020 년 1 월

위 원 장 이 경 식 (인)

부위원장 홍 유 석 (인)

위 원 박 건 수 (인)

초록

급변하는 시장 상황 아래서 제품의 출시 시점은 제품의 성공을 결정짓는 중요한 요인이다. 기업은 그 시점을 앞당기기 위해 제품을 설계하는 데 소요되는 기간을 단축하고자 하는 노력을 계속하고 있다. 그러나 설계 과정 중 발생하는 설계 반복과 그로 인한 재작업은 전체 설계 기간을 연장하는 원인이 된다.

설계 반복을 줄이기 위한 방안으로써 특정한 설계 변수의 값을 고정하는, 설계 규칙이라는 방안이 활용되고 있다. 그러나 설계 규칙을 정함에 있어 설계 변수 값을 단독으로 고정할 수 있는지 혹은 그 주변 설계 변수의 값을 함께 고정해야 하는지에 대한 고려가 필요하다. 중요한 정보를 주고 받아 함께 설계해야하는 설계 변수들임에도 설계 규칙으로 인해 따로 설계를 실시할 경우, 원치 않는 설계 규칙의 수정 또는 제품 품질의 저하라는 결과를 낳을 수 있기 때문이다.

따라서 본 연구는 설계 규칙의 논의를 확장하여 일정 범위의 설계 변수를 선정하고 그에 대한 설계 작업을 우선적으로 수행하는 선제적 설계라는 전략적 의사결정을 제안한다. 설계 변수 간 연관 관계를 고려하여 설계 규칙의 대상 범위를 정함으로써 실현 가능한 설계 규칙을 정하는 것을 목표로 한다.

선제적 설계의 도입으로 인해 기존 진행하던 설계 프로세스가 어떻게 바뀌어 진행되는지 그래프 형태로 나타낸 설계 네트워크를 통해 해석한다. 설계 네트워크를 기반으로 설계 변수 간 연관 관계의 중요성을 고려한 선제적 설계의 대안을 생성한다. 나아가 생성된 대안 각각을 설계 프로세스에 대해 적용했을 때 효과가 어떻게 나타나는지 설계 프로세스 내 발생하는 재작업량의 관점에서 분석한다.

주요어: 선제적 설계, 설계 반복, 설계 규칙, 설계 네트워크

학번: 2018-25523

목차

표 목차

그림 목차

제 1 장 서론

일반적으로 복잡한 제품 개발 과정 내에서 설계 반복(design iteration)은 피할 수 없는 현상이다(Wynn and Eckert, 2017). 설계 작업(design task)을 순차적으로 한 번만 수행하는 것이 아니라, 이후에 진행되는 설계 작업으로부터 새로운 정보를 전달 받아 이미 수행했던 설계 작업에 대해 재작업(rework)을 하게 된다. 이와 같은 설계 프로세스 내의 현상을 설계 반복이라 한다(Smith and Eppinger, 1997b).

설계 반복은 설계 문제 해결의 관점에서 창의적인 해답을 찾는 과정이라는 점(Wynn and Eckert, 2017), 추상적인 수준의 설계를 구체화시켜 나가는 과정이라는 점(Denker et al., 2001; Costa and Sobek, 2003)에서 긍정적인 의미를 가진다. 그러나 재작업으로 인한 제품 개발 기간 상의 낭비가 크다는 점(Ballard, 2000), 빠르게 변화하는 소비자의 요구사항 및 기술 환경 아래서 제품 출시 시점은 제품의 성공에 큰 영향을 미치는 요인 중 하나라는 점(Cohen et al., 1996; Bayus, 1997; Chen et al., 2005)과 같은 이유로 부정적인 의미를 보다 크게 가지며 제품 개발 과정 내에서 관리해야 할 대상으로 인식되고 있다

그 동안 설계 프로세스 내 설계 반복을 관리하기 위한 방안이 여러 연구들에 의해 제시되었다. Smith and Eppinger(1997a)에 따르면 반복적인 설계 프로세스를 가속 하기 위한 방안으로 설계 반복을 더욱 빠르게 수행하는 것과 설계 반복의 횟수를 줄이는 것을 고려할 수 있다. 이 중 학문적으로 주된 관심의 대상이 되어 왔던 것은 설계 반복의 횟수를 줄이는 것으로, 본 연구는 설계 반복을 관리하기 위한 전략적인 의사결정의 하나로써 Baldwin and Clark

(2000)가 제안한 설계 규칙(design rule)에 주목한다.

설계 규칙이란 특정한 설계 변수(design parameter)에 대해 그 값을 고정하는 내용의 전략적인 의사결정이다. 이 때 설계 변수란 제품을 물리적으로 구현하기 위해 정해야 하는 수치적인 값으로 정의한다(Suh, 1998). 변수에 대해 특정한 값을 고정함으로써 대상이 된 설계 변수에 대해 더 이상 재작업을 수행하지 않겠다는 의미이며, 나머지 설계 변수들은 그 고정된 값에 종속된 결정을 내리게 된다. 이러한 설계 규칙의 설정으로 인해 설계 프로세스 내의 설계 반복은 그 패턴이 단순해진다.

하지만 잘못된 설계 규칙의 설정은 설계 규칙의 수정 또는 제품 품질의 저하라는 설계 실패(design failure)로 이어질 수 있다. 설계 규칙은 특정 설계 변수의 값을 고정하기 때문에 나머지 설계 변수가 그 변수의 값을 주어진 것으로 받아들이도록 인위적인 제약을 만든다. 이 때 나머지 설계 변수의 값을 찾을 수 있는지, 설정한 설계 규칙대로 설계 프로세스를 완료할 수 있는지가 문제 된다. 예를 들어 설계 규칙으로 정해둔 대로 설계를 진행하였으나 설계 변수 간 호환성에 문제가 생기거나 설계의 결과물로 나온 제품의 성능이 기대치에 미치지 못하는 경우가 발생할 수 있다.

설계 규칙을 제안한 Baldwin and Clark(2000)에 따르면 이러한 설계 실패는 제품의 기저에 깔려 있는 설계 변수 간 연관 관계를 고려하지 않을 때 발생한다. 설계 규칙의 대상이 되는 설계 변수의 값을 정함에 있어 중요한 정보를 주는 다른 설계 변수가 존재함에도 그 정보를 반영하지 않고 설계 규칙을 정하는 것은 실현 가능하지 않은 의사결정이 된다. 따라서 설계 규칙을 정하는 데 있어 설계 변수 간 얼마나 중요한 정보가 오고 가는지 그 연관 관계의 중요성을

기반으로 의사결정을 내리는 것이 요구된다. 즉 설계 규칙의 대상이 될 변수에 대해 그 값을 정하고자 할 때 그 변수와 밀접한 관계를 가지는 변수들과 함께 그 값을 정해야 하며, 그 때 어느 설계 변수까지 설계 규칙의 대상으로 선정할 것인지 판단이 필요한 것이다.

이와 같은 맥락에서 본 연구는 설계 실패를 야기하지 않을, 실현 가능한 설계 규칙을 정립하는 방법으로써 선제적 설계(preemptive design)의 개념과 방법론을 제안하고자 한다. 이는 설계 규칙을 정하는 데 있어 그 대상이 될 설계 변수의 선정 범위를 찾는 것을 목표로 하며, 범위 선정에 있어 설계 변수 간 연관 관계의 중요성을 고려하는 것에 그 의의가 있다.

이후 논문의 구성은 다음과 같다. 2장에서는 관련 연구로써 설계 반복을 관리하기 위한 전략적인 의사결정을 다루었던 논문들에 대하여 서술하고자 하며, 3장에서는 본 논문에서 중심으로 다루는 선제적 설계의 개념과 선제적 설계가 설계 프로세스에 미치는 영향을 분석하고자 한다. 4장에서는 선제적 설계를 수행하기 위해 선제적 설계의 대상이 되는 설계 변수의 묶음의 대안을 생성하고 생성된 대안들을 재작업량 측면에서 효과를 분석하는 일련의 프로세스를 서술한다. 마지막으로 5장은 토의와 결론, 향후 연구 과제에 관한 정리로 마무리할 것이다.

제 2 장 선행 연구

Ulrich and Eppinger(1998)에 따르면 제품의 설계 변수 또는 설계 작업 간의 관계는 정보 전달 관계에 따라 순차적(sequential), 병렬적(parallel), 상호종속적(coupled)으로 나눌 수 있다. 이 중 관심이 되는 대상은 상호종속적 관계로 설계 작업을 수행하는 데 있어 다른 작업과 서로의 정보를 필요로 하는 경우이다. 상호종속적 관계를 가지는 설계 작업끼리는 지속적으로 서로의 정보를 교환하면서 동시에 또는 반복적으로 설계 작업을 수행해야만 한다. 이러한 설계 변수 간의 상호종속적 연관 관계가 설계 프로세스 내 설계 반복을 발생시켜 설계 프로세스 내의 정보의 흐름을 복잡하게 만드는 원인이 된다.

설계 반복으로 인해 제품 설계는 일반적인 프로젝트와는 다른 관리 방안을 필요로 한다. 프로젝트를 관리하기 위한 일반적인 방법론으로써 PERT(Project Evaluation Review Technique) 또는 CPM(Critical Path Method)이 사용되지만, 제품 설계 프로세스 중 반복되는 설계 작업을 묘사하기는 어렵다(Black et al., 1990; Tang et al., 2000). 따라서 설계 반복이 반영된 설계 프로세스를 나타내기 위한 모델링이 어떻게 진행되었는지, 설계 반복의 관리를 위해 설계 프로세스에 대해 어떠한 내용의 의사결정을 내릴 수 있는지 참고가 필요하다.

본 장은 설계 반복을 관리하기 위한 전략적인 의사결정을 제안한 연구들에 대해 서술한다. 구체적으로는 설계 반복을 줄이기 위해 설계 작업 간 상호종속성을 제거할 수 있는 방안, 설계 프로세스 내 설계 작업을 나누어서 진행하는 것을 제안한 연구들을 살펴볼 것이다. 특히 본 연구에서 논의를 확장

하고자 하는 설계 규칙에 주목하여 그 내용과 한계점을 들여다보고자 한다.

Steward(1981)는 설계 반복을 고려한 설계 프로세스의 구조를 표현하기 위해 설계 체계 매트릭스(Design Structure Matrix; DSM)를 제안하였다. 설계 프로세스를 설계 변수를 정하는 과정으로 보아, 전기 자동차의 설계 과정을 예시로 든 그림 2.1과 같이 설계 작업 간 선후 관계를 매트릭스의 형태로 표현하였다. 매트릭스의 'X'표시는 변수 간의 연관 관계 및 변수 결정의 선후 관계를 나타낸다. 예를 들어 2번 변수인 '크기와 기체 역학'은 1번 변수인 '승객 수용 능력 사양'의 결정에 영향을 받아 1번 변수의 결정 이후에 이루어지는 의사 결정이다.

Steward는 특히 DSM의 상삼각행렬에 위치한 'X'표시에 주목해 상호종속적 관계를 갖는 설계 작업들을 파악하여 설계 프로세스의 관리 대상으로 삼았다. DSM에서 'X'표시들이 하삼각행렬로 정렬된 경우 정렬된 순서에 따라 설계를 진행하면 되는 이상적인 설계 프로세스로 볼 수 있다. 그러나 작업의 순서를

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Passenger Capacity Spec.	1	·															
Size and Aerodynamics	2	X	·	X				X				X	X				
Motor Spec. and Weight	3		X	·	X		X	X				X					
Total Weight	4	X	X	X	·			X				X	X				
Stored Energy Req.	5			X		·	X		X	X	X			X			
Battery Type & Energy Density	6						·										
Battery Size and Weight	7					X	X	·									
Cruising Speed Spec.	8								·								
Speed & Accel. Perf. vs Power	9	X	X		X					·			X				
Acceleration Spec.	10										·						
Speed& Accel. Conformance	11								X	X	X	·					
Structural & Suspension Design	12	X	X	X	X			X				X	·				
Range Spec.	13													·			
Cost	14		X	X	X		X	X					X		·		
Consumer Demand vs Cost	15	X	X						X		X		X	X		·	
Profit	16														X	X	·

그림 2.1: 전기 자동차에 대한 DSM 예시(Steward, 1981)

바꾸어도 상삼각행렬에 ‘X’가 남아있을 수 있으며, 이는 나중에 진행하는 설계 작업이 이전에 진행된 설계 작업에 피드백(feedback)을 주는 상황을 말한다. 피드백의 정보가 앞서 진행된 설계 작업이 기존 가정했던 정보와는 다른 새로운 내용을 담고 있을 때 재작업이 발생하며 이전에 진행되었던 설계 프로세스가 재차 반복된다. 그림 2.2 왼쪽 하단 매트릭스의 2, 3, …, 12번 설계 작업들과 같이 어떻게 순서를 바꾸어도 상삼각행렬에 ‘X’가 존재하게 되는 설계 작업의 묶음이 상호종속적 설계 작업이며 이를 관리 대상으로 삼는다.

설계 프로세스 내 상호종속적인 설계 작업을 관리하기 위해 파티셔닝(partitioning) 과 테어링(tearing)의 도입을 통해 설계 프로세스를 간소화하고자 하였다. 파티셔닝은 서로 관련 있는 설계 작업들을 그룹화하여 상호종속적인 설계 작업의 범위를 파악하는 작업이자 설계 반복의 범위를 파악하는 작업으로, DSM에서 하위 매트릭스로의 클러스터링으로 표현된다. 테어링은 설계 작업의 결과물을 추정하거나 작업 간의 연관 관계를 무시함으로써 설계 반복의 범위를 줄이는 작업을 말하며, 매트릭스 내 ‘X’표시의 제거로 표현된다.

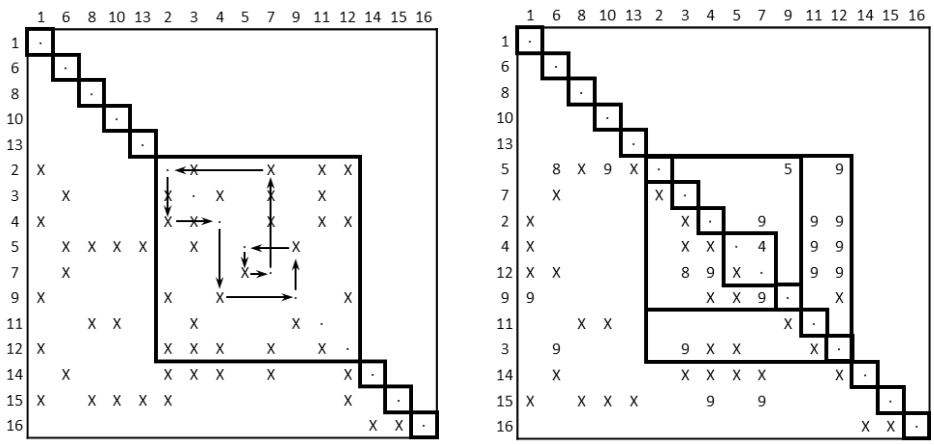


그림 2.2: DSM에 대한 파티셔닝과 테어링 예시(Steward, 1981)

이 때 테어링을 할 설계 작업을 정하기 위해 설계 작업 간 정보 추측의 난이도를 기준으로 삼았다. 설계 작업 간의 결과물을 추정하기 어려운 대상을 '1', 쉬운 대상을 '9'로 난이도를 단계별로 수치화하여 그 결과물을 추측하기 쉬운 설계 작업을 대상으로 테어링을 수행할 수 있다고 보았다. 하지만 어떤 난이도를 기준으로 테어링을 할 수 있는지 그 기준의 내용이 분명하지 않음을 지적할 수 있다. 그 기준에 대한 의사결정자의 전략적인 판단에 따라 테어링을 수행하게 될 설계 작업의 대상이 달라질 수 있으며, 그림 2.2의 오른쪽 하단 예시처럼 다른 설계 작업에 대한 테어링의 적용에 대해 DSM 클러스터링 결과가 다르게 나타남을 확인할 수 있다. 이후 테어링의 기준을 보다 명확히 하고자 다음과 같은 연구들이 수행되었다.

Gebala and Eppinger(1991)는 DSM을 기반으로 한 파티셔닝과 테어링을 설계 프로세스에 적용하는 알고리즘을 제안하였다. 파티셔닝을 위해 DSM 내의 순환을 찾고자 하였으며, 순환을 찾기 위한 방법으로 DSM 내의 'X'표시를 따라가는 방법과 DSM 내의 'X'를 '1'로 대체한 인접 행렬(adjacency matrix)의 거듭 제곱을 이용하는 방법을 이용하였다. 테어링에 대해서는 가장 적은 수의 설계 작업에게 영향을 받는 설계 작업, 즉 매트릭스 상에서 행에 위치한 'X'가 가장 적은 설계 작업을 선정하여 파티셔닝된 매트릭스 내부에서 가장 먼저 수행하도록 하는 휴리스틱을 고안하였다.

Yassine et al.(1999)은 테어링을 하는 기준으로 설계 작업의 민감도와 변동성을 고려하였다. 설계 작업의 민감도는 선행하는 설계 작업으로부터 정보를 받았을 때 해당 설계 작업의 결과물이 얼마나 변하는지를 나타내며, 설계 작업의 변동성은 후행 설계 작업의 입장에서 선행하는 설계 작업 결과물의

범위가 얼마나 넓은지 나타내는 수치이다. 두 가지 수치를 작업과 작업 사이 연관 관계에 부여하여 숫자로 표현한 DSM을 제안하였으며, 두 가지 수치를 기반으로 테어링에 대한 기준을 설정하였다. 연구에서 제안한 기준은 설계 작업 사이의 민감도가 낮아 선행하는 설계의 변화에 큰 영향을 받지 않고, 변동성이 낮아 선행하는 설계 작업의 결과를 예측하기 쉬운 연관 관계를 찾는 것이다.

피드백을 해결하기 위한 다른 방향의 접근으로 León et al.(2013)은 상호종속적인 작업 안에서 우선적으로 순환을 해결해야 할 설계 작업을 찾아내고자 하였다. 각 피드백에 대하여 주대각선(diagonal)을 기준으로 대칭된 'X'표시가 있는지 여부를 확인하여 대칭된 'X' 표시가 있는 경우를 제거할 수 없는 피드백, 없는 경우를 가짜 피드백으로 구분하였다. 그리고 Thomke and Fujimoto(2000)가 설계 프로세스 내에서 발생할 수 있는 문제를 최대한 이른 시점에 발견하여 해결하는 전략으로 제안한 프론트로딩의 개념을 이용하였다. 즉 제거할 수 없는 피드백을 상호종속적인 설계 작업 묶음의 앞에 위치시켜 설계 프로세스 내의 순환을 이른 시점에 종료시키고자 하였다.

언급한 연구들은 설계 프로세스를 관리하기 위한 방안으로 이후 진행되는 피드백의 내용을 무시하거나 피드백의 내용을 우선적으로 해결함으로써 설계 반복이 일어나는 범위를 축소하고자 하였다. 그러나 제안된 의사결정 방안은 설계 작업 간의 연관 관계를 파악하여 그 관계를 조율했다는 점에서 의의가 있으나, 설계 프로세스 내에서 제품의 구조와 그로 인한 설계 프로세스 내 의사소통 상에서의 문제점을 반영하지 못하였다. 또한 의사결정을 통해 달성하고자 하는 목적이 명확하지 않아 설계 프로세스 내에서의 설계 반복을 축소한다는 것 이상의 의미를 도출하기 어렵다.

설계 반복의 축소를 위해 설계 작업들을 나누어 진행하는 설계 프로세스의 분해(decomposition)를 제안한 연구들도 있다. 설계 프로세스의 분해는 설계 프로세스 내 해결해야 할 복잡한 문제를 하위 문제로 나누어 각 문제를 해결하기 위한 설계 작업이 동시적으로 수행되도록 하기 위한 목적을 가진다 (Eppinger, 1991; Chen and Lin, 2002; Yassine and Braha, 2003). Kusiak and Wang(1993)은 설계 작업 묶음 간 겹치는 변수가 최소가 되도록 설계 프로세스를 분해하였다. 설계 작업을 설계 작업 내 포함된 변수를 정하는 과정으로 보아 작업과 변수 간의 연관 관계를 매트릭스 형태로 파악하였다. 겹치는 변수가 최소가 되도록 구성된 작업의 묶음은 독립적으로 설계 작업을 수행할 수 있다는 전제 하에 전체 설계 프로세스의 동시성(concurrency)을 제고하였다. David(2013)은 상호종속적인 설계 작업을 두 개의 묶음으로 분해하여 설계 프로세스의 작업량을 평균화할 수 있도록 하였다. DSM에 대해 스펙트럴 알고리즘을 이용하여 묶음 내의 연관 관계는 최대로, 묶음 간의 연관 관계는 최소로 하는 방향으로 설계 작업을 묶었으며, 분해된 프로세스에 대한 작업량 변화를 살폈다.

한편, Eppinger et al.(1994)는 서로 관련 있는 작업들의 연관 관계 혹은 부품 간 인터페이스에 관한 작업을 별도로 추가함으로써 역설적으로 설계 프로세스의 속도가 빨라질 수 있음을 제안하였다. 연구에서 제시한 자동차의 브레이크 시스템 중 전기기계 장치의 일부에 대한 설계를 보면, 상호종속적으로 연관되어 있는 설계 작업들 중 와이어의 상세 설계를 첫번째 프로토타입을 만든 이후로 미루었다. 와이어 설계에 있어 와이어 수정이라는 추가적인 작업이 생겼지만 그럼에도 기존 설계 반복으로 인한 6개월이라는 기간이 좁아진 설계 반복 범위

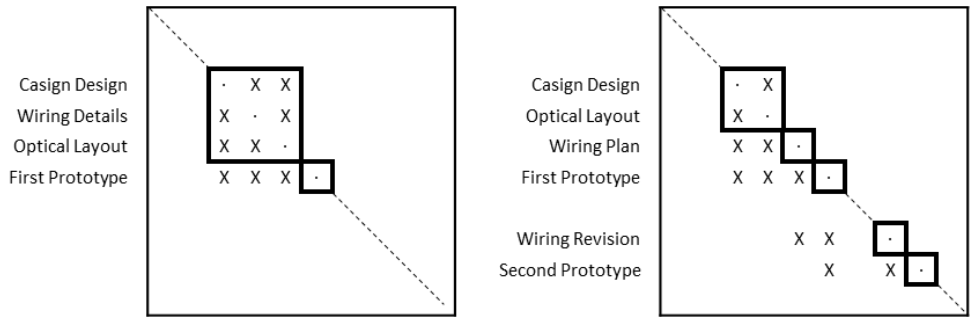
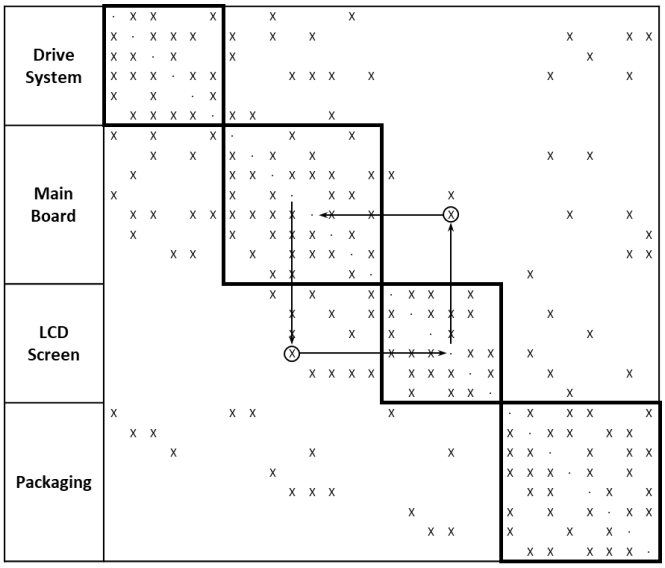


그림 2.3: 작업의 추가를 통한 설계 반복 범위 축소(Eppinger et al., 1994)

내에서 몇 주 간의 기간으로 줄어들었음을 언급하였다. 이는 경험적으로 수행한 사례를 통해 도출해낸 결과로 다른 일반적인 설계 프로세스에 대해 어떤 추가적인 작업이 필요한지 이끌어내기 어렵다.

Baldwin and Clark(2000)는 특정한 설계 변수에 대해 그 값을 고정하는 전략적인 의사결정인 설계 규칙을 제안하였다. 설계 규칙은 설계 프로세스 관리 대상으로 부품 간 설계 반복(inter-component design iteration)에 주목하였다. 그 이유로 1) 설계 팀의 구성이 제품의 구조를 반영하여 구성된다는 점, 2) 부품 사이의 연관 관계가 설계 팀 사이의 의사소통으로 이어진다는 점, 3) 설계 팀 내에서 이루어지는 의사소통보다 서로 다른 설계 팀 간 의사소통이 어렵다는 점을 언급하며 부품 간의 연관 관계로 인한 설계 반복이 전체 설계 프로세스를 더디게 만드는 요인으로 보았다.

따라서 설계 규칙의 목적은 부품 간 관계를 본격적인 설계 이전에 정해주어 부품을 담당하는 각 팀이 자신들의 업무를 독립적으로 수행할 수 있도록 하는 것이며 이와 같은 설계 규칙의 도입 과정을 설계 모듈화(design modularization)라 하였다.



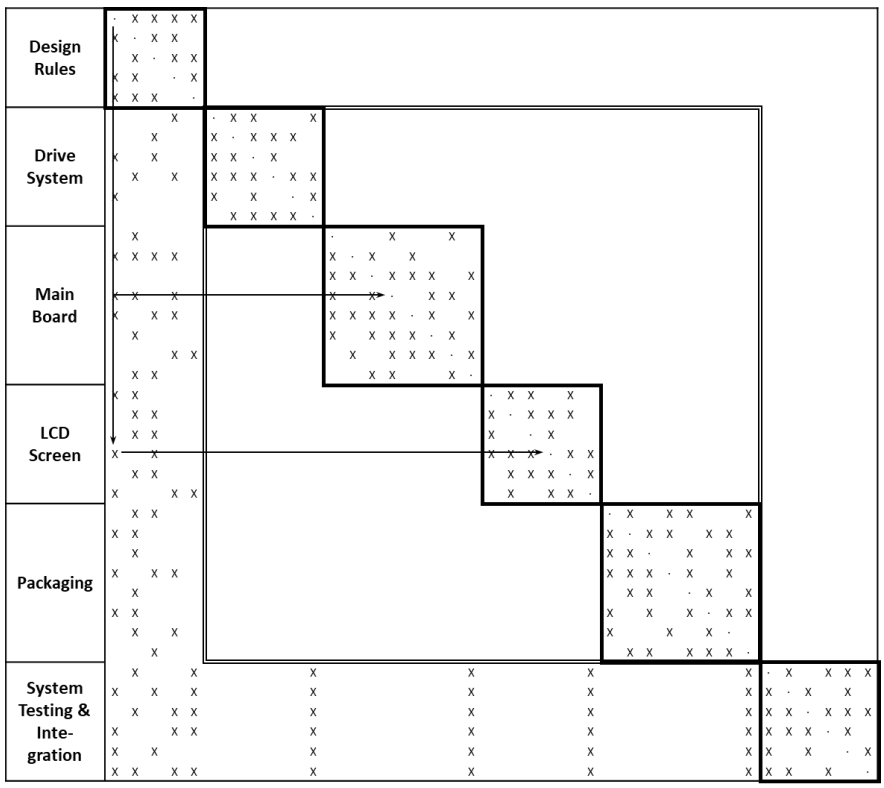


그림 2.4: 휴대용 컴퓨터에 대한 설계 규칙 적용 예시(Baldwin and Clark, 2000)

Baldwin and Clark(2000)의 설계 규칙에 대한 논의를 중심으로 앞서 설계 반복을 관리하기 위한 의사결정 방안을 제안한 연구들을 되짚어 볼 때, 설계 프로세스 내 의사소통의 양상과 의사결정의 실현 가능성 양측을 함께 고려한 연구가 부족하다.

제안된 설계 규칙은 제품 구조가 반영된 설계 팀 간의 의사소통 패턴을 명시적으로 고려하였다. 이는 기존 설계 작업 간 연관 관계만을 토대로 정보의 흐름을 통제하고자 했던 테어링의 개념을 확장한 것이다. 더하여 설계 프로세스를 나누어 진행하는 목적을 설계의 모듈화라는 개념으로 분명히 하여 프로세스 분해의 방향성을 제시하였다. 제품 개발 과정 내 설계 작업의 숫자와 그로 인한 정보의 흐름이 증가하고 복잡해지는 추세임을 고려할 때(Braha, 2016), 설계 규칙의 제안은 설계 반복 관리의 기준을 명확히 하였다는 점에서 의의를 가진다.

그러나 설계 규칙은 그 의사결정을 도입함에 있어 실현 가능성에 대한 내용을 반영하지 않아 한계점을 가진다. 언급한 설계 규칙의 방향성은 그 의미가 있으나, 설계 규칙의 구현을 위한 방법을 고려하지 않고 부품 사이의 연관 관계를 모두 사전에 제거한다는 선언적인 내용에 그쳤기 때문이다. 또한 현실의 설계 프로세스에서 설계 규칙을 어떻게 설정할 수 있는지, 설계 규칙을 정했을 때 설계 프로세스 상에서 효과가 구체적으로 어떻게 나타나는지에 대한 분석이 미흡하다.

따라서 설계 규칙이 실제로 설계 프로세스에 적용되기 위해 테어링과 프로세스의 분해와 관련한 연구에서 주목하였던 설계 작업 간 연관 관계를 고려하는 것이 필요하다. 본 연구는 이러한 점에 착안하여 설계 규칙의 목표인

설계 프로세스의 모듈화를 달성하는 동시에 그 실현가능성을 제고하기 위한 선제적 설계라는 방안을 제안한다. 나아가 선제적 설계를 이행하기 위한 프로세스를 구체화하며 그에 의한 효과를 분석하고자 한다.

제 3 장 선제적 설계

3.1 개요

제품의 설계 프로세스는 부품의 설계 변수를 정하는 일련의 설계 작업 활동으로 볼 수 있다. 설계 변수와 설계 작업 사이의 관계는 다양하게 규정될 수 있으나(Kusiak and Park, 1990; Suh, 1998; Clarkson and Hamilton, 2000), 본 연구에서는 설계 변수(design parameters)와 설계 작업(design tasks)이 일대일로 대응되는 설계 프로세스를 가정한다. 즉, 설계 작업 하나가 설계 변수 하나를 담당하는 단위로 구성되어 있으며 설계 변수 사이의 연관 관계가 곧 설계 작업 간의 연관 관계가 되는 것으로 간주한다.

선제적 설계는 설계 프로세스 내에서 특정한 설계 변수들의 설계 작업을 선제적으로 수행함으로써 그 설계 변수들의 값을 고정하는 전략적인 의사결정이다. 설계 변수들 중에서 다른 부품과 연관이 있는 설계 변수를 후보 설계 변수로 정하고, 이와 중요한 연관 관계를 가지는 설계 변수들을 탐색하면서 설계 프로세스 상에서 선제적으로 설계를 마쳐야 할 설계 변수들을 선정한다. 후보 설계 변수마다 함께 설계를 수행해야하는 설계 변수의 묶음이 대안으로 생성되며, 그들 중 실제로 선제적 설계의 대상이 될 대안을 고르는 것이 최종적인 의사결정의 내용이 된다.

본 연구는 설계 변수 간의 관계를 네트워크 형태로 표현하여 선제적 설계 도입 시 네트워크가 어떻게 변화되는지 그 효과를 시각적으로 나타낸다. 그림 3.1과 같이 선제적 설계에 따라 설계 반복이 단순하게 바뀌는 것을 관찰할 수

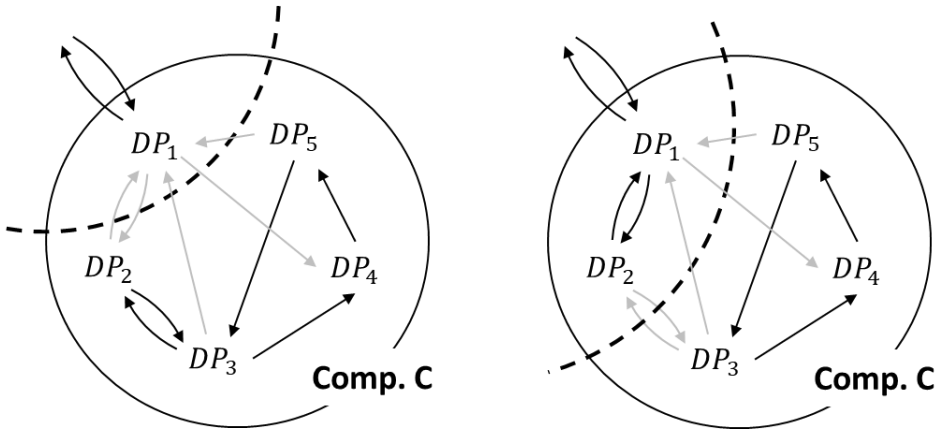


그림 3.1: 선제적 설계 범위 선정 및 그에 따른 효과

있다. 특정한 설계 변수의 값을 고정함으로써 나머지 설계 변수들은 그 값을 주어진 것으로 받아들이고 설계 작업을 이행하게 되어 설계 프로세스가 보다 순차적으로 진행되기 때문이다.

그림 3.1의 왼쪽 그림은 DP_1에 대해 선제적 설계를 수행하기로 했을 때의 작업의 흐름을 나타내었다. DP_1에 대해 먼저 설계 작업을 수행하여 그 값을 고정한다는 것은 연관 관계를 가지는 나머지 설계 변수로부터의 재작업 요구를 받지 않겠다는 것이며 기존에 있던 작업의 흐름이 단절됨을 의미한다. DP_1의 값이 고정된 이후 DP_2부터 DP_5는 본래 설계 변수의 연관 관계에 따라 정보를 주고 받으며 설계 작업을 수행하게 된다. 이 때 작업의 흐름이 단절됨으로 인해 설계 프로세스 중 수행되어야 할 설계 반복이 이전보다 줄어들고 단순해진다.

그러나 단절되는 작업의 흐름 중 무시할 수 없는 내용의 흐름이 존재하는 경우, 이는 설계 실패를 낳는 의사결정이 된다. 예를 들어 DP_1과 DP_2 사이의 정보 교환이 DP_1의 내용을 정하는 데 있어 중요한 내용을 담고 있다고 할 때, DP_1과 DP_2 사이 작업의 흐름을 단절시키는 이와 같은 선제적 설계의 수행은

DP_1 에 대한 적절한 값을 정하지 못하는 결과를 낳으며, 이는 잘못된 설계 규칙의 수립이 된다. 잘못된 설계 규칙의 수립은 이후 남은 설계를 진행하면서 미리 정했던 설계 규칙의 내용의 변경을 야기하는 원인이 되거나 설계 결과물의 품질이 저하되는 문제를 가져올 수 있다.

그림 3.1의 오른쪽 그림은 이러한 문제에 대응하여 선제적 설계의 대상을 DP_2 까지 포함했을 때의 작업의 흐름을 나타낸 것이다. DP_1 을 선제적으로 설계할 때 DP_2 를 함께 설계함으로써 DP_1 설계에 필요한 DP_2 의 설계 내용을 반영할 수 있다. 또한 만약 새로이 끊어지는 DP_2 과 DP_3 사이 작업의 흐름이 단절되어도 괜찮은, 중요하지 않은 내용을 담고 있다면 DP_2 를 DP_1 과 함께 선제적 설계의 대상으로 삼음으로써 덜 중요한 작업의 흐름을 끊어내되 중요한 작업의 흐름이 유지된다. 이렇듯 적절한 범위의 설계 변수를 선정하여 선제적으로 설계를 수행하는 것을 통해 실현 가능한 설계 규칙을 수립할 수 있다.

선제적 설계는 설계 변수 간 연관 관계를 고려하여 함께 그 내용을 정해야 할 설계 변수들을 선정하여 실현 가능한 설계 규칙을 정하도록 하는 데 그 의의가 있다. 설계 규칙은 설계 반복을 줄이기 위한 전략적인 의사결정이지만, 그에 따르는 위험인 설계 실패의 가능성을 함께 고려해야 한다. 이와 같은 내용을 반영한 선제적 설계의 도입은 설계 프로세스에 대한 전략적인 의사결정을 체계화할 수 있는 구체적인 이행 방안이 될 것이다.

3.2 설계 네트워크

설계 변수 간 복잡한 연관 관계를 구조적으로 표현하기 위해 변수 단위의 DSM(parameter-based DSM)과 유향 그래프(directed graph)를 이용한다. 변수 단위의 DSM은 정방형의 매트릭스로써 설계 변수, 혹은 설계 변수를 정하는 낮은 단위의 설계 작업(low-level design task)간의 연관 관계를 나타낸다(Black et al., 1990; Browning, 2001; Pektaş and Pultar, 2006). 설계 변수를 정하는 설계 작업이 DSM의 행과 열을 구성하며, 매트릭스 안의 'X'표시는 열에 위치한 설계 작업이 행에 위치한 설계 작업에게 영향을 줌을 의미한다. 이 때 영향의 의미는 열에 위치한 설계 작업의 내용이 바뀌었을 때 행에 위치한 설계 작업이 그 내용을 수정해야하는, 즉 재작업을 수행해야하는 관계를 가짐을 말한다. 또 다른 한 가지 표현 방법인 유향 그래프의 경우 각 노드는 설계 변수, 간선은 설계 변수 사이의 연관 관계를 나타낸다. 이 때 연관 관계의 의미는 앞선 DSM 표기에서 'X'의 의미와 동일하다.

동일한 의미를 가지는 두 가지 표현 방법으로 설계 변수 사이의 연관 관계를 네트워크 형태로 나타낼 수 있으며, 이를 설계 네트워크(design network)라 부르기로 한다. 그림 3.2는 상기한 두 가지 표현 방법으로 나타낸, 1번부터 15번까지의 설계 변수를 정하기 위한 설계 작업과 그 사이의 연관 관계를 표현한 설계 네트워크의 예시이다. 이후 연구에서는 그래프 형태의 표기를 주로 이용하여 선제적 설계가 설계 네트워크에 대해 어떤 효과를 가져오는지 분석할 것이다.

여러 부품으로 구성되어 있는 제품의 구조 아래서 설계 변수를 부품 단위로 묶어 구분할 수 있다. 기업 내에서 조직의 구조는 대개 제품의 구조를 반영하여 구성됨을 고려할 때(Browning, 2001), 이와 같은 구분은 각 설계 팀이 어떠한

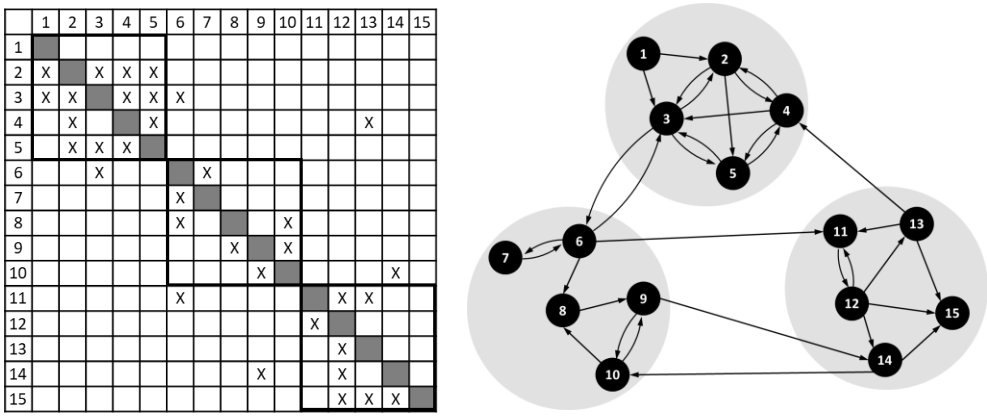


그림 3.2: 매트릭스와 그래프로 나타낸 설계 네트워크 예시

설계 변수를 담당하여 설계하는지를 조사함으로써 파악할 수 있다. 설계 네트워크 상으로는 DSM 상에서 굵은 색으로 표현한 구획과 그래프 상에서의 음영으로 표현된다. 이는 설계 변수 사이의 연관 관계를 부품 내와 부품 사이의 연관 관계로 구별하기 위한 목적이며 이후 설계적 설계를 수행하는 데 있어 필요한 정보가 된다. 예를 들어 구획 내의 3번 – 5번 설계 변수 간의 관계는 부품 내 연관 관계, 구획 사이를 연결 짓는 3번 – 6번 설계 변수 간의 관계는 부품 사이의 연관 관계로 해석할 수 있다.

3.3 선제적 설계의 효과

3.3.1 설계 프로세스 변화

선제적 설계의 도입으로 인해 설계 프로세스는 선제 설계 단계(preemptive design phase), 후속 설계 단계(subsequent design phase)의 두 단계로 나누어져

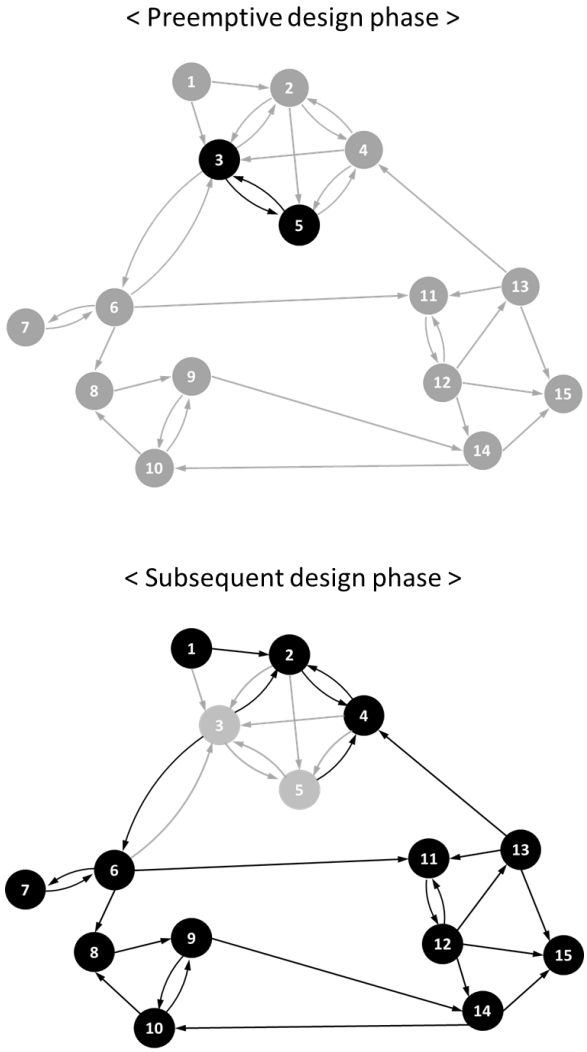


그림 3.3: 설계 네트워크에 대한 선제적 설계 적용

진행된다. 이러한 설계 프로세스의 진행을 앞서 표현한 설계 네트워크에 반영하면 다음 그림 3.3과 같다.

그림 3.3은 설계 변수 3번과 5번의 묶음을 선제적 설계의 대상으로 하여 설계 네트워크에 반영한 예시이다. 첫번째 단계인 선제 설계 단계에서는 선제적

설계의 대상으로 삼은 설계 변수들에 대해서 설계 작업을 수행한다. 선제적 설계의 대상이 된 설계 변수들은 설계에 필요한 정보를 주고 받으며 그 내용이 정해질 때까지 설계 작업을 진행한다. 다만 선제적 설계는 이후 설계 프로세스에 대해 제약으로 작용할 설계 변수의 값을 고정하는 의사결정이기에 선제적 설계 이전에 연관된 설계 작업들로부터 필요한 입력 정보를 수렴하는 과정이 필요하다. 설계 변수 3번과 5번의 값을 어떤 정보를 활용하여 정할 것인지에 대한 합의가 도출된 이후 선제적 설계를 진행할 수 있다. 이 때 협의에 필요한 설계 팀 혹은 설계 변수의 선정은 정량화하여 분석하기 어려우며, 설계자들의 경험을 통해 보충되어야 한다.

선제적 설계의 대상 범위에 있는 설계 변수들의 의사결정이 수렴되어 고정되고 나면 두번째 단계인 후속 설계 단계가 이어지며 선제 설계 단계에서 정한 설계 변수 외 나머지 설계 변수들에 대해서 설계 작업을 수행하게 된다. 후속 설계 단계에서의 설계 작업은 선제 설계 단계에서 결정한 설계 변수를 토대로 의사결정을 내리게 된다. 후속 설계 단계에서의 설계 작업은 선제 설계 단계에서의 설계 작업에 영향을 주지 않으며 단지 앞서 정해진 설계 변수의 값을 제약 조건으로 받아들여 그 결정을 구속 받게 된다. 이와 같은 정보의 흐름을 반영한 그림 3.3의 후속 설계 단계에서의 네트워크를 살펴볼 때, 선제적 설계를 마침으로 인해 설계 변수 3번과 5번으로 향하는 간선은 사라졌으나 이들로부터 나오는 간선은 여전히 존재하는 것으로 표현된다.

특정한 설계 변수의 값을 고정함으로 인해 후속 설계 단계에 대한 제약이 생겨 설계 결과물의 내용에 영향을 미칠 수 있음에 유의한다. 설계 상에 제약을 안게 된 설계 프로세스의 결과물과 간섭을 하지 않은 설계 프로세스의 결과물은

차이가 있기 때문이다(Schätz et al., 2010; Biskjaer et al., 2014). 그러나 그러한 결과물의 차이가 얼마나 큰지, 또 차이가 발생한 결과물의 품질에 대해 우열을 가릴 수 있는지 답하기엔 어려움이 따른다. 설계 상의 제약으로 인해 설계자의 자유의지가 훼손될 수 있으며(Amabile, 1982) 설계 결과물의 품질이 낮아질 가능성이 발생한다는(Thomke, 1997) 부정적인 시각으로 볼 수 있으나, 설계 상에서의 제약은 창의적인 설계 결과물을 이끌어 내는 데에 도움을 준다는 주장(Johnson-Laird, 1988; Onarheim, 2012) 또한 존재하기에 일반화하기에는 어려운 문제이다.

3.3.2 설계 클러스터 단위로 본 효과

설계 프로세스는 상호종속적인 설계 작업 묶음 안에서 설계 작업들이 반복적으로 수행되는 흐름을 가진다. 그러한 묶음 안에서 설계 작업 순서를 정하는 것은 그 자체로 하나의 전략적인 의사결정이나(Meier et al., 2006; León et al, 2012; Attari-Shendi et al., 2019), 설계 작업의 선후 관계가 분명한 경우, 설계 작업 묶음 간 선후 관계가 분명한 경우에 대해서는 그에 따라 순서가 정해질 수 있다. 본 연구에서는 그래프 내의 강연결 요소(strongly connected component; SCC)로 정의된 설계 클러스터(design cluster)를 설계 프로세스가 진행되는 단위로 보아 설계 프로세스의 흐름을 살피고자 한다.

설계 프로세스 내 설계 반복의 변화는 그래프 내 존재하는 강연결 요소의 양상 변화로 파악할 수 있다. 강연결 요소란 강하게 연결 되어 있는 최대 부분 그래프(maximal strongly connected subgraph)를 의미한다. 이 때 '강한 연결'의

의미는 유향 그래프 내의 임의의 두 노드 사이에 경로가 존재함을 말한다. 두 개의 노드 사이 양방향으로 가는 경로를 찾을 수 있기에 강연결 요소 내 두 노드 사이에는 항상 순환이 존재한다. 주어진 그래프 상에서 강연결 요소의

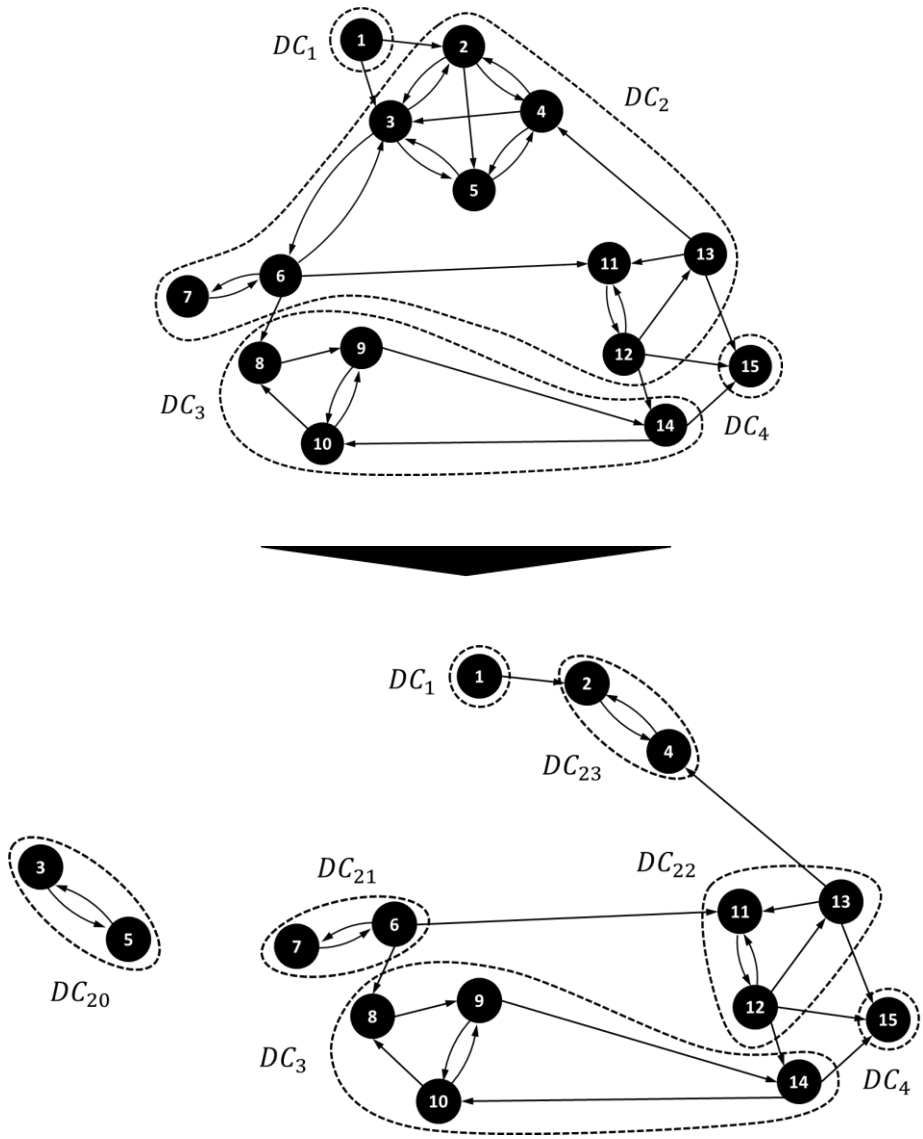


그림 3.4: 선제적 설계로 인한 설계 네트워크의 변화

형태는 유일하며 그래프 내의 강연결 요소를 찾는 알고리즘은 Tarjan(1972)과 Sharir(1981) 등에 의해 연구된 바 있다. 강연결 요소는 최대 특성(maximal property)을 가지고 있기에 서로 다른 강연결 요소 사이에는 순환이 존재하지 않으며, 강연결 요소는 설계 반복이 일어나는 경계의 의미를 가진다(Steward, 1981).

강연결 요소를 설계가 진행되는 묶음인 설계 클러스터로 해석하였을 때, 선제 설계 단계와 후속 설계 단계의 두 단계로 진행토록 하는 선제적 설계의 도입으로 인해 설계 클러스터가 분할된다. 그림 3.4와 같이 그래프로 표현된 설계 네트워크가 두 개의 연결된 부분 그래프로 나누어지며 원래의 그래프 내의 설계 클러스터가 분할된다. 설계 클러스터의 분할 결과 기존에 존재하던 2-3-6-11-12-13-4-2와 같은 긴 길이의 순환이 사라진 것을, 그래프 내에서 설계 반복이 일어날 수 있는 영역이 줄어들었음을 확인할 수 있다. 이는 넓은 영역에서 설계 반복이 발생하고 있던 DC_2 가 선제 설계 단계에서의 DC_{20} , 후속 설계 단계에서의 DC_{21}, DC_{22}, DC_{23}로 나뉘었기 때문이다.

설계 클러스터의 분할은 설계 변수들에 대한 재작업에 대해 영향을 미친다. 분할 이전의 DC_2에서는 설계 변수 6번으로부터 출발한 설계 정보가 여러 설계 변수를 거치는 복잡한 경로를 거쳐 다시 돌아와 설계 변수 6번에 재작업을 야기하는 넓은 범위에 걸친 설계 반복이 나타났다. 그러나 클러스터의 분할로 인해 설계 변수 6번은 DC_{21} 내에서 설계 변수 7번과의 상호작용으로 인한 재작업만을 고려하게 되며 반복된 재작업 끝에 얻어진 결과물을 설계 변수 11번의 입력 변수로서 넘겨주게 된다. 넓은 범위의 설계 변수 영역에 대해 진행되던 설계 반복이 선제적 설계의 도입으로 인해 작은 크기의 클러스터로

나누어지게 되었고 그 결과 이전보다 순차적으로 설계 프로세스를 진행할 수 있게 된다.

더하여, 설계 네트워크를 설계 클러스터 단위로 보았을 때 설계 프로세스는 단방향의 흐름을 가진다. 이는 강연결 요소의 응축(condensation)을 통해 표현되는 개념으로, 그래프 내 존재하는 강연결 요소를 하나의 노드로 응축시키면 원래의 유향 그래프가 유향 비순환 그래프(directed acyclic graph; DAG)로 변환됨을 이용한다. 유향 비순환 그래프 내에선 사이클이 존재하지 않는 것을 고려할 때, 설계 클러스터 사이에 설계 반복이 일어나지 않음을 반영한다. 그림 3.4를 응축하여 나타낸 그림 3.5에서도 역행하는 흐름 없이 단방향의 정보 흐름만 존재할 뿐, 클러스터 사이 순환이 존재하지 않게 된 것을 확인할 수 있다.

정의된 설계 클러스터의 흐름대로 설계 변수의 값을 정하는 것은 설계 프로세스 내 재작업량을 줄이기 위한 설계 작업의 순서를 정하는 의미를 갖는다. 만약 그 흐름을 따르지 않은 설계 프로세스를 수행할 시 설계 클러스터 사이의 정보의 전달로 인한 재작업을 추가적으로 수행하게 되며, 이는 설계 반복 및 재작업량 측면에서 손해가 되기 때문이다. 이러한 접근은 기존 연구 중 DSM 내에서 설계 작업의 순서를 상호종속적인 작업을 기준으로 묶어 정하는 것과 같다(Steward, 1981; Yassine and Braha, 2003).

설계 클러스터의 분할로 설계 반복이 줄어들었다는 해석 외에도 설계 프로세스가 이전보다 모듈화 되었음을, 각 설계 팀이 이전보다 독립적으로 작업을 수행하게 되었음을 확인할 수 있다. DC_2 내의 설계 변수 3번과 5번에 대한 선제적인 결정으로 인해 후속 설계 단계에서 DC_1을 담당하는 설계 팀과

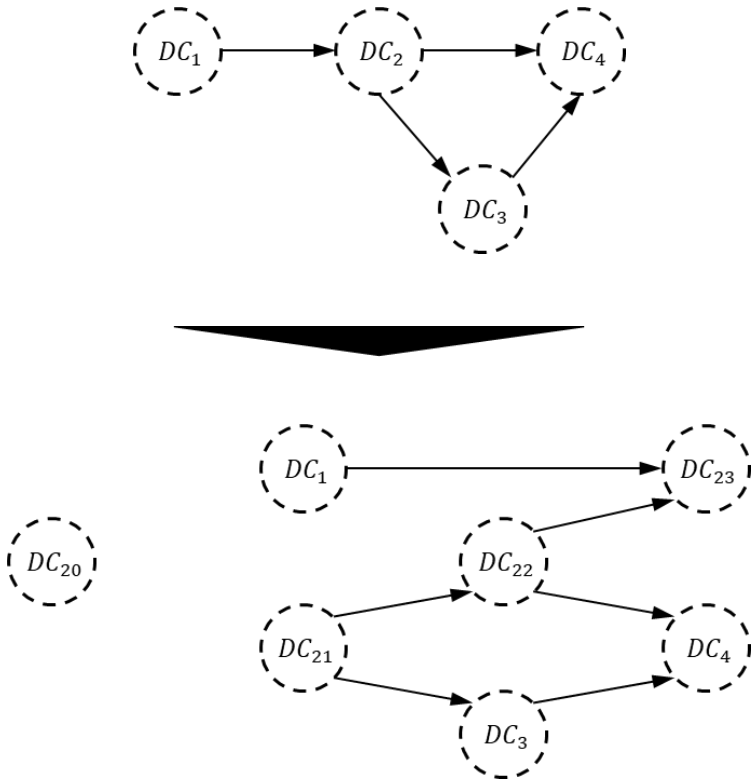


그림 3.5: 유향 비순환 그래프로 응축된 설계 네트워크

DC_{21}을 담당하는 설계 팀이 자신들의 일을 각자 시작할 수 있게 된다. 이는 설계 팀 사이 상호작용을 야기하는 설계 변수에 대해 선제적으로 의사결정을 내림으로써 설계 팀이 자신의 업무를 수행 하는 데 필요한 정보의 내용이 분명해졌기 때문이다.

선제적 설계로 인하여 설계 프로세스의 진행 및 설계 작업의 순서가 바뀌는 것을 고려할 때, 설계 작업의 순서가 제품의 품질에 어떻게 영향을 주는지 분석한 연구를 주목할 만하다(Badhrinath and Jagannatha Rao, 1996; Krishnan et al., 1997; Lewis and Mistree, 1998). 이들 연구에서는 제품의 품질과 관련한 제품의

사양 및 특성 값을 목적 함수로 설정하여 설계 변수가 설계 프로세스 내에서 정해지는 과정을 순차적으로 진행되는 최적화 또는 게임이론적인 접근을 통하여 도출하였다. 그러나 이는 제품의 품질을 설계 변수의 수치적인 값으로부터 분명하게 정의할 수 있을 때 적용 가능한 방법론이기에 제품의 품질을 정의하기 어려운 일반적인 설계 상황을 해석하는 데에는 무리가 있다. 따라서 선제적 설계가 설계 결과물의 품질에 미치는 영향은 추후 연구 과제로 남겨두기로 하며 본 연구는 이후 선제적 설계라는 전략적 의사결정이 설계 프로세스 내 재작업량 측면에서 어떤 효과를 가져오는지를 중심으로 살펴볼 것이다.

제 4 장 수행 및 평가

4.1 선제적 설계 프로세스

본 항목에서는 선제적 설계의 대상이 될 설계 변수를 선정하는 프로세스를 설명한다. 이를 위해 선제적 설계의 고려 대상이 될 설계 변수의 대안을 생성하며 나아가 각 대안을 선제적 설계의 대상으로 삼았을 때 설계 프로세스 내의 재작업량이 얼마나 줄어드는지 그 변화를 평가하는 것까지 논의의 대상으로 삼는다.

선제적 설계의 목표는 설계 반복의 관리, 그 중 부품 간 일어나는 설계 반복을 관리하는 것에 있다. 부품 간 설계 반복은 부품 간 상호작용을 일으키는 설계 변수를 정하는 과정 안에서 정보의 흐름이 순환하는 형태를 가질 때 발생하게 된다. 여러 부품에 걸쳐진 설계 변수들의 거듭된 조정은 설계자가 자신의 의사 결정이 제품에 미치는 영향을 파악하기 어렵게 만들며, 자신이 수행해야할 재작업의 내용 또한 예측하기 어렵게 만든다. 이러한 문제를 해결하기 위해 부품 간 인터페이스를 사전적으로 정의하는 작업을 통해 부품 간 연관 관계를 제거하고자 하는 연구들이 제안된 바 있다(Eppinger, 1991; Baldwin and Clark, 2000).

본 연구 또한 선제적 설계의 대상으로써 부품 간 상호작용을 일으켜 부품 간 설계 반복을 야기할 수 있는 설계 변수에 주목한다. 부품 간 상호작용을 일으키는 설계 변수의 값을 고정함으로써 나머지 설계 변수는 그에 종속된 의사결정을 내릴 수 있으며, 부품 사이 발생하는 설계 변수의 조정 과정이 단순

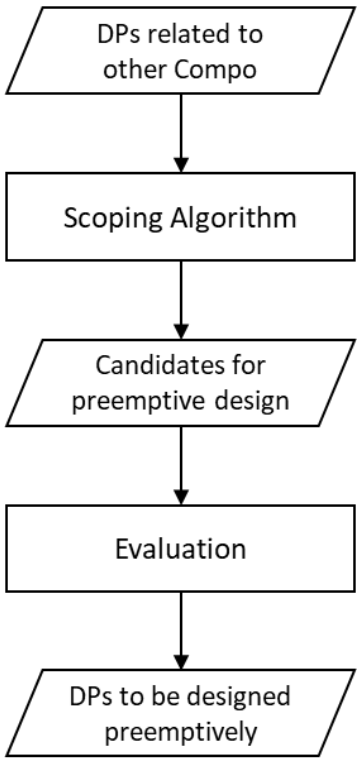


그림 4.1: 선제적 설계 프로세스 흐름도

해진다. 선제적 설계는 여러 부품에 걸쳐 발생하던 설계 반복을 각각의 부품 내로 한정되도록 하기 위함이며, 이는 Baldwin and Clark(2000)이 제안한 설계 모듈화의 목적과 일치한다.

선제적 설계의 수행을 위해 다른 부품과 연관되어 있는 설계 변수와 함께 설계를 수행해야 할 설계 변수를 찾아 선제적 설계를 위한 대안들을 생성한다. 이를 위해 다른 부품과 연관된 설계 변수에 대해 단독으로 값을 고정할 수 있는지 살펴보아야 한다. 만약 해당 설계 변수의 값을 정하는 데 있어 함께 설계를 진행해야하는 다른 설계 변수가 존재하는 경우, 그러한 설계 변수는 선제적 설계의 대상에 포함하는 것이 설계 실패의 가능성을 줄일 수 있기

때문이다. 이와 같은 내용을 반영하여 다른 부품과 연관되어 있는 설계 변수 각각으로부터 파생되는 설계 변수의 묶음을 선제적 설계의 대안이라 부른다.

마지막으로 선제적 설계의 대안으로 생성된 설계 변수의 묶음 중 어느 것을 선제적 설계의 이행 대상으로 삼아야 할지 평가한다. 그 평가 기준은 정량적이고 정성적인 내용을 모두 포함할 수 있다. 선제적 설계를 통해 달성하고자 했던 설계 프로세스내의 설계 반복을 효율적으로 줄일 수 있는지, 선제적으로 설계 작업을 수행하는 것이 실제로 가능한 설계 변수들인지 등 다각적인 평가가 가능하다. 평가 과정 끝에 정해진 설계 변수의 묶음은 다른 설계 변수들보다 앞서 설계 작업이 수행되는 대상이 된다.

본 논문에서 제안하는 설계 변수 선정 알고리즘은 부품 간 상호작용을 야기하는 설계 변수부터 시작하여 이와 연관 관계를 가지는 주변 설계 변수를 탐색해 나간다. 특히 함께 설계를 진행해야 할 설계 변수의 기준을 설계 작업 간 재작업 임팩트(rework impact)로 삼아 그 중요도를 판단하여 선정 여부를 정한다. 이러한 판단 과정을 반복하여 더 이상 선정된 설계 변수들에 대해 중요한 연관 관계를 가지지 않는 범위까지 그 선정 범위를 확장한다. 그 결과 부품을 연결 짓는 설계 변수 각각으로부터 파생되는 설계 변수의 묶음이 정해지며, 이들은 부품 간 설계 반복을 줄임과 동시에 설계 실패를 야기하지 않을 선제적 설계의 대안이 된다.

또한 선제적 설계의 대안으로 생성된 설계 변수의 묶음을 평가하는 기준은 생성된 설계 변수 묶음 각각을 설계 프로세스에 적용했을 때의 재작업량으로 삼는다. 이는 앞서 살펴보았던 설계 네트워크 내의 설계 클러스터 분할이 재작업량에 어떤 영향을 끼치는지 분석하는 정량적인 분석 과정이 될 것이다.

선제적 설계의 목적이 설계 프로세스 내 설계 반복을 관리하는 것에 있음을 고려하였을 때, 설계 반복으로 인해 발생하는 재작업량이 얼마나 줄어드는지를 평가하는 것은 그 목적에 부합한다고 말할 수 있다.

4.2 대안 생성

4.2.1 후보 설계 변수

여러 부품으로 구성되어 있는 제품의 구조 나아가 그를 구현하고자 하는 설계 변수의 구조 내에서 선제적 설계의 후보 설계 변수는 각 부품마다 다른 부품과 연관이 있는 설계 변수이다. 이 때 정보를 주는 설계 변수와 받는 설계 변수 사이에 차이는 두지 않으며 이들 설계 변수가 부품 사이 정보의 흐름을 발생시킨다는 사실에 주목 하기로 한다. 이는 부품 별로 음영 처리가 되어 있는 설계 네트워크 구조에서 음영 사이를 연결 짓는 설계 변수를 파악하는 작업이다.

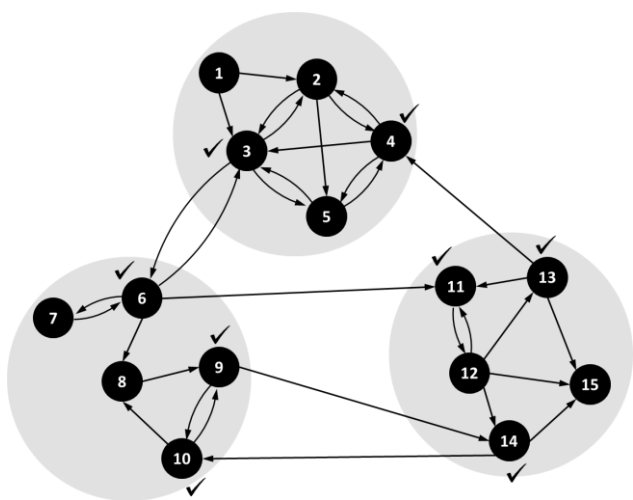


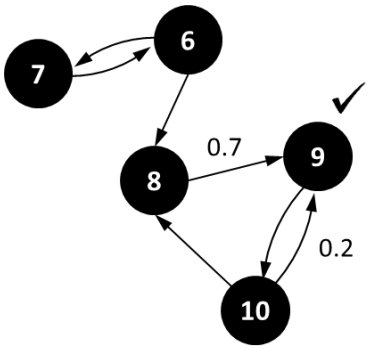
그림 4.2: 선제적 설계를 위한 후보 설계 변수

그림 4.2에는 이와 같은 접근을 통해 총 8개의 설계 변수가 선정되어 표시되었다. 구한 $S = \{3,4,6,9,10,11,13,14\}$는 선제적 설계를 위한 후보 설계 변수의 집합 이며 이후 진행될 대안 생성 알고리즘의 입력 변수가 된다.

4.2.2 대안 생성 알고리즘

집합 S의 각 원소를 시작으로 하여, 선정된 설계 변수에 대해 중요한 연관 관계를 가지는 주변 노드를 탐색하여 그러한 주변 노드를 선정 범위에 포함하는 것으로 선제적 설계를 이행할 수 있는 설계 변수의 묶음을 구한다. 이 때 연관 관계의 중요성에 대한 판단 기준은 재작업 임팩트를 기준으로 삼기로 한다. 재작업 임팩트란 어떤 설계 작업의 결과물에 변화가 생겼을 때, 그 결과물이 다른 설계 작업에게 얼만큼의 재작업을 야기하는지 비율로써 나타낸 지표로, 0부터 1까지의 숫자로 나타내어진다(Carrascosa et al., 1998; Maier et al., 2014).

재작업 임팩트의 크기가 클수록 한 설계 작업이 다른 설계 작업에 대해 재작업을 크게 야기한다는 의미이며 그만큼 중요한 정보를 전달하는 것이라 해석할 수 있다. 그림 4.3의 그래프는 설계 변수 9번을 향한 재작업 임팩트 값을 간선에 반영하였다. 설계 변수 8번으로부터 설계 변수 9번으로의 간선의 비중이 0.7이라는 것의 의미는 설계 변수 8번을 정하는 작업의 내용이 바뀌었을 때 설계 변수 9번을 정하는 작업이 이전에 수행했던 작업량에 대하여 0.7의 비율만큼 재작업을 해야함을 뜻한다. 이는 설계 변수 8번으로부터 새로운 정보를 받게 된 설계 변수 9번이 그에 대응하는 재작업을 얼마나 수행해야 하는지 나타낸 수치이다. 따라서 재작업 임팩트는 설계 변수 간 정보의 중요성에 대한 지표로 사용할 수 있으며 이 값은 설계자들의 경험과 기존에



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	30														
2	0.7	40	0.5	0.2											
3	0.3	0.4	25	0.2	0.8	0.4									
4		0.4		30	0.7								0.8		
5		0.2	0.5	0.3	70										
6			0.3			40	0.8								
7						0.3	30								
8						0.6		90		0.4					
9								0.7	50	0.2					
10									0.3	25				0.6	
11						0.5					30	0.7	0.5		
12											0.4	40			
13												0.4	20		
14									0.3			0.8		15	
15												0.5	0.3	0.4	30

그림 4.3: 재작업 임팩트 값을 반영한 그래프와 DSM

쌓여온 설계 프로세스에 대한 데이터 분석을 통해 도출할 수 있다.

‘X’대신 재작업 임팩트 값을 매트릭스의 원소로 갖는, 숫자로 나타낸 DSM은 그림 4.3의 오른쪽 매트릭스와 같다. 더하여 DSM의 대각 원소는 각 설계 변수를 설계할 때 소요되는 시간 단위로 표현된 초기 작업량으로, 이후 설계 변수를 정하는 설계 작업에 대해 발생하는 재작업량의 크기를 도출하는 데 이용될 것이다.

본 연구에서 제시하는 알고리즘은 설계 변수에 대해 영향을 주는 다른 설계 변수 중 재작업 임팩트를 기준으로 중대한 영향을 주는 다른 설계 변수를 찾는 것을 수행한다. 그래프의 표기 아래선 노드로 들어오는 방향의 간선(incoming edge)의 가중치에 대해 판단하는 과정이다. 이 때 중요한 영향에 대한 판단에 있어 재작업 임팩트에 대해 특정한 임계값을 도입하여 그 임계값을 넘는 재작업 임팩트를 주는 주변 설계 변수를 선제적 설계 대상에 포함하는 것으로 한다. 만약 그림 4.3의 예시에서 재작업 임팩트의 임계값을 0.5로 정했다면, 설계 변수

9번으로부터 시작하는 선제적 설계 대안에 설계 변수 8번은 포함되나 설계 변수 10번은 포함되지 않는다.

서술한 판단 과정을 새롭게 선정된 설계 변수에 대해서도 재귀적으로 수행하여 선정된 설계 변수들에 대해서 중요한 영향을 주는 설계 변수가 더 이상 존재 하지 않을 때까지 탐색 과정을 반복한다. 설계 변수 8번이 설계 변수 9번으로부터 시작한 선제적 설계 대상에 포함된다면 설계 변수 8번에 대해서도 들어오는 방향의 간선을 탐색하여 중요한 영향을 주는 주변 설계 변수들에 대해 판단하는 과정을 반복한다.

탐색 과정이 종료된 후 설계 변수 9번으로부터 시작하여 선정된 설계 변수들은 설계 변수 9번을 설계 규칙으로 삼고자 할 때 그 값을 정해야 할, 선제적 설계의 대안이 된다. 알고리즘을 통해 찾아낸 설계 변수의 묶음은 그 주변 설계 변수들의 작업으로부터 임계값 이하의 재작업 임팩트를 받는 성질을 가진다. 즉 주변 설계 변수들로부터 중대한 영향을 받지 않아 이후 그 값을 수정해야 할 필요가 적은 설계 묶음이 구성된다.

지금까지 설명한 메커니즘을 담은 알고리즘을 의사코드로 표현하면 다음 알고리즘 1과 알고리즘 2와 같다. 이 때 $N^{-}(x)$는 노드 x에 대하여 들어오는 방향의 간선을 가진 이웃노드들의 집합, $e(y,x)$는 노드 y로부터 노드 x로의 간선의 가중치, θ는 간선의 가중치에 대한 임계값을 의미한다.

알고리즘 1은 앞서 후보 설계 변수 집합 S의 원소, 각 노드 x부터 유도되는 설계 변수의 묶음을 찾기 위해 그래프 구조 G와 노드 x를 변수로 가지는 SEARCH 함수를 호출한다. SEARCH 함수는 노드 x에게 영향을 주는 노드에 대해 그 가중치를 판단하는 과정을 거치며 이는 설계 변수 x에 대해 중대한

Algorithm 1 Find subsets from initial candidates

procedure FINDSUBSETS(G, S)
  $P = \emptyset$
  for $x \in S$ ***do***
    $P \leftarrow P \cup \text{SEARCH}(G, x)$
  return P

Algorithm 2 Search by DFS

procedure SEARCH(G, x)
  θ *is given as threshold*
  $Q = \{x\}$
  for $y \in N^{-}(x)$***do***
    if $e(y, x) > \theta$ **then**
      $Q \leftarrow Q \cup \text{SEARCH}(G \backslash x, y)$
  return Q

그림 4.4: 대안 생성 알고리즘 의사코드

영향을 주는 설계 변수를 찾기 위한 것이다. 만약 노드 x에 대하여 일정한 크기 이상의 영향을 주는 노드가 존재한다면, 그 노드에 대해 다시 SEARCH 함수를 호출하는 깊이 우선 탐색(depth first search; DFS)을 수행한다.

초기 노드 x로부터 시작한 재귀적인 SEARCH 함수의 호출 끝에 도출되는 결과는 노드 x를 포함한 일정 영역의 노드들의 집합 Q이다. 이러한 과정을 집합 S에 속한 모든 초기 노드들에 대해 수행하여 각 초기 노드들로부터 유도된 집합 Q를 원소로 가지는 집합 P를 도출할 수 있다. 이 때 집합 P는 후보 설계 변수 집합 S의 각 원소들이 확장된 집합을 원소로 가지게 된다.

4.2.3 대안 생성 결과

초기의 후보 설계 변수 집합 S에 대한 알고리즘의 수행 결과로써, 선제적

설계를 수행하기 위한 설계 변수 묶음의 대안을 원소로 가지는 집합 P를 구할 수 있다. 예시로 든 그래프에 대해 임계값 $\theta = 0.5$를 적용하여 대안을 생성하면 $P = \{\{3,5\},\{4,5,13\},\{6,7\},\{6,7,8,9\},\{10,12,14\},\{11,12\},\{13\},\{12,14\}\}$ 와 같이 도출된다.

이 때 $\{3,5\}$원소의 의미는 만약 설계 변수 3번에 대해 그 값을 고정하려는 의사결정을 내리고자 한다면 설계 변수 5번을 함께 선제적으로 설계해야 한다는 의미이다. 만약 이 두 설계 변수를 따로 설계한다면 설계 변수 3번에 대해 내린 결정이 이후 설계 변수 5번에 의해 재작업에 대한 요구를 강하게 받아 설계 규칙의 실패를 야기할 수 있는 원인이 된다. 선제적 설계의 범위는 하나의 부품 내로 한정되지 않으며 $\{4,5,13\},\{10,12,14\}$와 같이 다른 부품 사이의 설계 변수를 함께 설계해야하는 결과 또한 나올 수 있다.

알고리즘에 의해 하나의 설계 변수를 원소로 갖던 S의 각 원소에 대해 확장된 설계 변수들의 집합이 도출된다. 이 때 설계 변수의 집합은 주변 설계

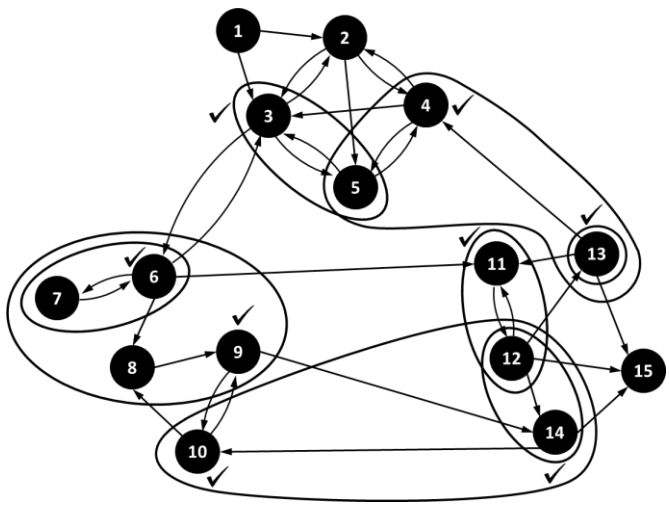


그림 4.5: 선제적 설계 대안 생성 결과

변수에 대한 탐색 과정을 거쳤기에, 유향 네트워크 상에서 연결된 부분 그래프(connected subgraph) 의 형태를 지닌다. 또한 결과로써 생성된 집합 P는 실체적인 의미에서 선제적 설계를 수행할 수 있는 최소한의 설계 변수의 집합을 담고 있다는 점에서 최소 집합(minimal set)을 집합의 원소로 가진다.

4.3 대안 평가

본 항목에서는 앞서 생성된 선제적 설계의 각 대안이 선택되어 설계 프로세스에 적용되었을 때 재작업량 관점에서 어떤 효과를 갖는지 평가한다. 이를 위해 설계 클러스터 내에서 일어나는 설계 반복으로 인해 각 설계 작업에 대해서 얼만큼의 재작업을 수행해야 하는지를 추정하는 메커니즘을 설명코자 한다. 이러한 메커니즘을 토대로 선제적 설계로 인해 설계 클러스터가 분할되는 것의 효과를 분석할 것이며 각 대안이 적용되었을 때의 재작업량의 변화를 계산 결과로써 구할 것이다.

4.3.1 설계 프로세스 내 재작업량 추정

설계 프로세스 내 설계 반복으로 인한 재작업량을 추정하기 위해 몇 가지 가정을 세우기로 한다.

i) 설계 프로세스는 설계 네트워크 내의 강연결 요소로 정의된 설계 클러스터의 흐름으로 진행한다.

ii) 각 클러스터는 클러스터 내의 설계 작업이 모두 완료된 후 자신의 결과물을 다음 클러스터에게 전해준다.

iii) 각 클러스터는 다른 클러스터로부터 필요한 모든 결과물을 전달받았을 때 자신의 설계 작업을 시작한다.

위와 같은 가정 아래서 클러스터 사이의 정보 흐름은 클러스터 내의 설계 변수의 내용이 모두 수렴이 된 후 발생하는 한 번의 결과물 전달로 마무리되어 클러스터 사이 정보의 흐름으로 인한 재작업은 발생하지 않는다. 즉 설계 프로세스 내의 설계 반복은 설계 클러스터 내부에서만 일어나게 되며, 각 클러스터 내부에서 발생하는 설계 반복으로 인한 재작업량의 합을 설계 프로세스가 진행되며 발생하는 재작업량으로 해석할 수 있다.

설계 변수에 대한 설계 작업은 다른 설계 변수의 수정에 의해 재작업 요구를 받게 되고, 특히 상호종속적인 설계 변수 간 연관관계 내에서는 이러한 재작업 요구가 반복되어 수행된다. 이는 상호종속적인 설계 변수들의 설계는 그 내용을 정하기 위해 서로의 정보가 필요하기 때문이며 이로 인해 설계 변수 간 정보의 흐름이 순환하는 형태를 가진다. 이러한 재작업은 서로의 정보가 충분하게 성숙되고 상호종속적인 변수들에 대한 결정이 모두 합의될 때까지 반복하게 된다. 다만 재작업이 반복되어 수행될수록 1) 설계 작업에 대한 설계자들의 숙련도가 올라가고 2) 설계 작업에 대해 바꾸어야 할 변경 내용이 줄어들게 되면서 점차 재작업에 소요되는 작업량이 감소하게 된다.

충분한 설계 반복이 진행된 후에는 재작업이 더 이상 필요하지 않은 시점, 설계 변수에 대한 의사결정이 수렴하게 되는 시점에 도달하게 된다. 제품 내

모든 설계 변수에 대해 의사결정이 완료되는 상황에 대하여 설계 수렴(design convergence)이라 말할 수 있고 이는 제품의 설계가 완료되어 종료되는 시점을 일컫게 된다. 이러한 설계 수렴 과정이 어떻게 이루어지는지, 수렴이 종료되는 시점이 언제인지를 분석하기 위해 여러 연구들이 각기 다른 모델링 방법을 통해 접근하였으며(Smith and Eppinger, 1997a; Chanron and Lewis, 2005; Braha and Bar-Yam, 2007; Maier et al., 2014) 설계 반복 과정을 통해 설계 프로세스 내에서 수행해야 하는 작업량이 줄어드는 과정을 반영하였다는 점에서 공통점을 지닌다.

설계 변수의 수렴은 설계 변수에 대한 작업을 처음 수행하는 데 필요한 시간, 그리고 해당 설계 작업이 다른 설계 작업들로부터 받는 영향에 따라 발생하는 재작업에 의해 정해진다. 설계 작업에 대하여 발생하는 재작업량은 어떤 설계 작업으로부터 재작업 요구를 받았는지에 따라 그 내용이 다르며 그 내용에 따라 수행해야 하는 재작업량의 크기도 다르다. 중대한 내용의 새로운 정보를 받은 경우 많은 양의 재작업을 수행해야할 것이며 그 내용이 수렴되어 가는 속도 또한 느릴 것이다. 반대로 새로운 정보를 받아도 그 내용이 사소한 것이라면 요구되는 재작업량도 적을 것이며 수렴 속도도 빠를 것이라 예상할 수 있다.

이러한 설계 작업 간 관계에 따라 달리 발생하는 재작업량의 크기를 고려하기 위해 재작업 임팩트 값을 이용하도록 한다. 재작업 임팩트는 선제적 설계의 범위를 도출하는 데 사용하였던 값으로 설계 작업에 변화가 생겼을 때, 그 결과물이 다른 설계 작업에 대하여 얼만큼의 재작업을 야기하는지 그 정도를 나타내었다.

더하여 설계 프로세스 상에서 발생하는 비용으로써 설계 작업을 처음

수행하는 데 필요한 시간을 제외한, 설계 작업에 발생하는 재작업량의 크기에 중점을 두어 분석하기로 한다. 설계 작업을 처음 수행하는 데 필요한 시간은 어떠한 설계 프로세스 내에서도 반드시 소요되는 시간으로 고정적인 값이기에 설계 프로세스에 대해 발생하는 비용 상의 손해로 간주하지 않을 수 있다. 설계 작업 단위의 재작업량 추산을 위해 아래와 같은 가정을 통해 설계 작업 간의 관계를 단순화하도록 한다.

i) 한 설계 작업의 변경이 발생한 경우, 그 설계 작업에 영향을 받는 다른 설계 작업은 그 변경으로 인해 확정적으로 재작업을 수행한다.
ii) 설계 프로세스는 고정된 시간 간격에 따라 진행되며, 각 설계 작업은 기마다 수행해야 할 작업량을 완수한다.
iii) 다른 설계 작업으로부터 재작업 요구가 들어온 경우, 그로부터 발생하는 재작업량은 그 설계 작업으로 인해 수행했던 이전 기 재작업량의 일정 비율(재작업 임팩트의 크기)만큼이다.
iv) 서로 다른 설계 작업으로부터 요구 받는 재작업의 내용은 독립적이다.

가정 i)은 설계 프로세스의 움직임에 대한 제약이다. 현실의 설계 프로세스 안에서 설계 변수가 자신의 내용을 수정하였을 때 다른 설계 변수의 내용을 수정을 해야하는지 아니면 그 내용을 유지해도 되는지는 불확정적인 사건이다. 이러한 설계 프로세스 내의 움직임에 대해 설계 작업 간 재작업이 일어날 확률 값을 도입하여 분석한 연구가 진행된 바 있다(Yassine et al., 2001; Cho and Eppinger, 2005; Karniel and Reich, 2013; Hossain and Chua., 2014). 그러나 본 연구에서는 한 설계 변수의 수정 작업이 관련한 다른 설계 변수의 작업에 대해 반드시

재작업을 야기하는 것으로 가정하여 설계 프로세스를 확정적인 상황으로 보아 재작업량을 계산하기로 한다.

가정 ii)와 iii)은 설계 작업에 대한 진행과 설계 변수에 대해 재작업이 발생하는 상황에 대한 가정이다. 설계 변수가 수행해야 하는 재작업량은 설계 반복이 진행되어 설계 프로세스가 성숙되어 가면서 점차 줄어들게 됨을 앞서 언급하였다. Smith and Eppinger(1997a)와 Yassine et al.(2003)과 같은 연구는 이를 반영하기 위하여 설계 프로세스를 이산적인 간격으로 시간을 구분하고 각 기마다 다른 설계 작업으로부터 새로운 정보를 받아 발생하는 재작업량이 점차 감소하는 것을 반영하고자 하였다.

선행 연구의 발상을 바탕으로 본 연구는 이번 기에 발생하는 재작업량이 이전 기에 수행했던 해당 설계 작업으로 인해 수행했던 작업량에 의존하며, 그 값이 줄어드는 비율은 재작업 임팩트 값으로 고정된다고 가정한다. 기를 거듭할수록 줄어드는 재작업량의 크기를 시간의 진행에 따른 함수의 형태로 표현하여 모델링할 수 있으나(Carrascosa et al., 1998), 재작업 임팩트의 고정된 비율만큼 재작업량이 줄어드는 것으로 설계 프로세스를 단순화하도록 한다. 이 때 설계 작업에 대해 매 기 사용할 수 있는 시간과 인력 자원이 충분하여 요구되는 재작업량을 충분히 완수할 수 있다는 가정이 전제되어 있음에 유의한다.

또한 가정 iv)에 따라 한 설계 작업에 대해 발생하는 재작업이 어떤 설계 변수로부터 요구되어 온 것인지 명확히 구분한다. 따라서 여러 작업이 한 설계 작업에 대해 요구하는 재작업 내용과 그 재작업량은 개별적으로 계산될 수 있다.

위와 같은 가정 아래, 설계 작업 b로 인해 설계 작업 a가 수행해야 하는

재작업량 r_{ab}의 크기는 다음과 같이 정해진다.

$$r_{ab} = \sum_{n=1}^{\infty} w_a (i_{ab})^n \tag{4.1}$$

이 때 w_a는 설계 작업 a를 처음 수행하는 데 필요한, 시간 단위로 나타낸 초기 작업량이며 i_{ab}는 설계 작업 b로부터 설계 작업 a로의 재작업 임팩트의 값을 뜻한다. 가정 iii)에 따라 재작업 임팩트만큼의 고정된 비율로 설계 작업 a가 설계 작업 b로 인해 수행해야 하는 작업량이 감소하며, 수식적으로는 멱급수의 형태를 가지게 된다.

4.2.1에서 가정한 설계 프로세스의 흐름 아래서, 어떤 임의의 설계 작업에 대해서 재작업을 발생시키는 것은 같은 설계 클러스터에 속하며 그 설계 작업에 대해 영향을 주는 설계 작업들이다. 설계 작업 a와 같은 클러스터에 속한 설계 작업의 집합을 $C(a)$라 할 때, 설계 작업 a에 대해 발생하는 전체 재작업량 r_a은 다음과 같다.

$$r_a = \sum_{x \in N^-(a) \cap C(a)} r_{ax} \tag{4.2}$$

식 (4.2)는 설계 작업 a가 수행해야하는 재작업량을 구하기 위해 때 앞선 문단의 조건을 만족하는 설계 작업 $x \in N^-(a) \cap C(a)$에 대하여 식 (4.1)의 계산을 수행한다. 이를 각 설계 작업에 대해 수행하고 계산된 재작업량을 설계 프로세스 내 존재하는 모든 설계 작업 $v \in V$에 대해 모두 합한 값이 전체 설계 프로세스 내의 재작업량의 크기 r이 된다. 이를 정리하면 식 (4.3)와 같다.

$$r = \sum_{v \in V} r_v \tag{4.3}$$

4.3.2 대안 평가 결과

대안을 평가하기 위해 각 대안 별로 계산된 설계 프로세스 내의 재작업량 크기를 비교한다. 이를 위해 앞서 제안한 방법대로 노드들의 재작업량을 계산하고 합하는 과정을 거친다. 예시로 선제적 설계를 도입하지 않은 설계 프로세스 내 DC_3 의 노드 중 설계 작업 9번에 대한 재작업량 크기를 구하면 아래 식 (4.4)와 같다.

$$\begin{aligned} r_{8,9} &= 50 \times 0.7 + 50 \times 0.7^2 + \cdots = \textstyle\sum_{n=1}^{\infty} 50 \times 0.7^n = 116.67 \\ r_{10,9} &= \textstyle\sum_{n=1}^{\infty} 50 \times 0.2^n = 12.5 \\ r_9 &= \textstyle\sum_{x \in N^-(9) \cap C(9)} r_{9x} = r_{8,9} + r_{10,9} = 116.67 + 12.5 = 129.17 \end{aligned} \tag{4.4}$$

식 (4.4)와 같이 식 (4.1)과 식 (4.2)의 계산을 각 노드에 대해 수행하여 클러스터 단위로 값을 묶으면 그림 4.6과 같다. 괄호 안에 있는 숫자가 해당 클러스터에서 소요되는 시간 단위로 표현한 재작업량을 나타낸다. 이러한 표기는 클러스터 내에서 소요되는 재작업량을 파악함을 통해 어떤 설계 클러스터에 대해 선제적 설계를 이행해야 하는지 직관을 준다. 예시의 경우 DC_2 에서 소요되는 재작업량의 크기가 다른 클러스터보다 상당히 큼을 고려하였을 때, DC_2에 속한 설계 변수에 대해 설계를 이행하는 것이 효율적인 의사결정일 것이라 예상할 수 있다.

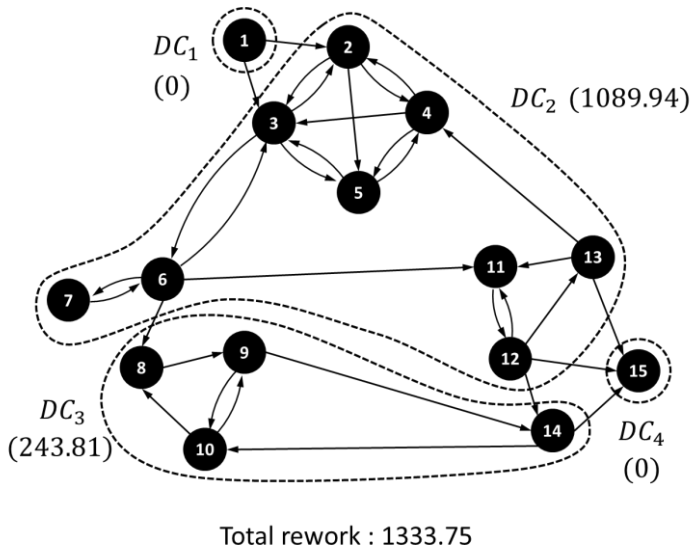


그림 4.6: 설계 클러스터에서 발생한 재작업량의 크기

설계 프로세스 상에서 발생하는 재작업량은 설계 네트워크 내 클러스터의 구조에 의해 달라지게 된다. 이는 앞서 정의한 조건 아래서 설계 변수가 같은 클러스터 내의 설계 변수로부터 영향을 받을 때에만 재작업을 수행하기 때문이다. 수식적으로는 식 (4.2)의 $C(\cdot)$ 에 대한 고려로 인한 것으로, 선제적 설계로 인해 일부 설계 클러스터가 나누어져 작아진 클러스터로 인해 설계 변수의 입장에서 재작업을 수행하도록 요구해오는, $C(\cdot)$ 조건을 만족하는 설계 변수가 이전보다 줄어들기 때문이다. 이는 설계 프로세스가 이전보다 순차적인 흐름으로 바뀌어 상호종속적인 설계 작업의 범위가 줄어들었음을 반영한다.

서술한 계산 과정을 이용하여 선제적 설계의 대안 각각을 설계 프로세스에 적용한 결과는 표 4.1과 같다. 선제 설계 단계과 후속 설계 단계를 수행하면서 발생하는 재작업량을 각각 구하여 합한 값이 전체 설계 프로세스를 진행하면서 발생한 재작업량이 된다. 대안 중 {10,12,14}를 선제적 설계의 대상으로 하였을

때 전체 설계 프로세스 내의 재작업량이 639.94 단위의 시간이 소요되어 도출된 다른 대안을 적용하였을 때보다 작은 값을 가지게 되었다. 그림 4.7으로 나타낸 최종 대안은 선제적 설계 프로세스의 결과물로써 선제적으로 설계를 수행하게 될 설계 변수의 묶음이 된다.

표 4.1: 선제적 설계 도입 후 설계 프로세스 내 재작업량

	Workload of Additional Rework		
Preemptive design set	Preemptive Design Phase	Subsequent Design Phase	Total
Ø	0	1333.75	1333.75
{3,5}	130.00	613.33	743.33
{4,5,13}	176.19	603.81	780.00
{6,7}	172.86	817.08	989.94
{6,7,8,9}	172.86	573.27	746.13
{10,12,14}	**0**	**639.94**	**639.94**
{11,12}	96.67	883.75	980.42
{13}	0	980.42	980.42
{12,14}	0	839.82	839.82

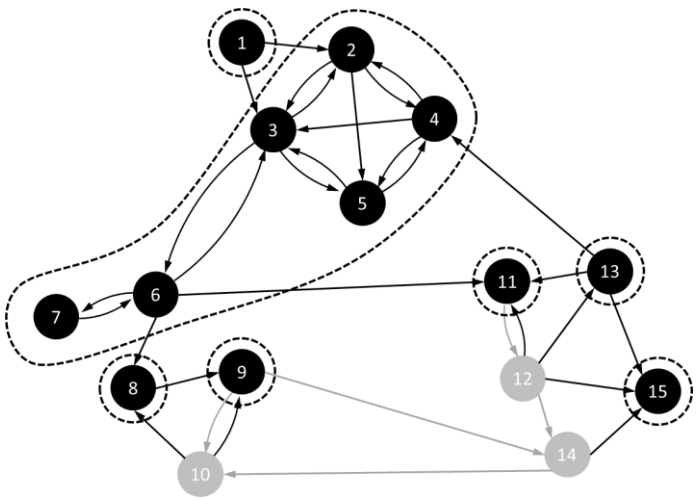


그림 4.7: 재작업량을 최소화하는 선제적 설계 대안

제 5 장 결론

5.1 결론 및 토의

본 논문에서는 실행 가능한 설계 규칙의 설정을 위한 방안으로써 선제적 설계의 개념과 선제적 설계를 이행하기 위한 프로세스를 제안하였다. 선제적 설계는 부품 간 설계 반복을 줄이기 위한 전략적인 의사결정으로 다른 설계 변수들보다 우선하여 그 내용을 고정할 설계 변수의 선정 범위를 찾아 먼저 설계를 수행하는 작업이다. 특히 실현 가능한 설계 규칙의 설정을 위해 설계 변수 간 연관 관계, 본 연구에서는 설계 변수를 정하는 작업 간 재작업 임팩트를 설계 변수 사이의 중요성으로 고려하여 선제적 설계의 대안을 생성하였다. 나아가 도출된 각각의 대안을 설계 프로세스에 적용하였을 때의 효과를 설계 프로세스 내의 재작업량을 기준으로 비교하여 선제적 설계의 대상이 될 설계 변수의 묶음을 도출하였다.

다만 선제적 설계는 제품의 인터페이스, 나아가 제품 개발 순서에 영향을 미치는 전략적인 의사결정인만큼 선제적 설계를 행하기 위한 설계 부문 전반의 노력이 필요하다. 앞서 언급하였듯 선제적 설계의 대상이 되는 설계 변수에 대한 결정을 먼저 내리기 위해 그 설계 변수와 관련한 설계 부서가 협의를 거쳐 필요한 정보를 합의하는 과정이 요구된다. 이는 선제적 설계로 인해 끊어지게 될 정보의 흐름에 대한 내용을 정하는 과정으로 선제적 설계를 수행하기 위해 필요한 정보를 수렴하는 내용이다.

재작업량을 추정하는 데 사용한 방법론은 선제 설계 단계에서의 작업량에

대해 선제적 설계의 대상이 되는 작업 간 발생하는 재작업량만을 추정하였다. 그러나 설계 부서 간 협의를 거치는 과정 또한 선제 설계 단계에 반영이 되어야 할 작업량에 속해야 하는 내용이다. 선제적 설계에 앞서 선행해야 하는, 선제적 설계에 필요한 정보를 모으는 작업의 양과 난이도에 따라 어떤 설계 변수에 대해 선제적 설계를 수행해야하는지 그 결정이 달라질 수 있다. 이는 정량적으로 그 값을 이끌어내기에는 어려운 정보로써 설계 부문을 아우르는 의사결정자의 전략적인 판단을 통해 보완되어야 하는 부분이다.

추가적으로 본 연구의 주제로 삼지는 않았으나 연구의 각론으로써 선제적 설계이라는 의사결정 메커니즘을 실제 적용할 때 고려해야 할 사항들이 있다. 재작업 임팩트 임계값에 대한 의사결정, 도출된 선제적 설계 범위 여러 개를 설계 프로세스에 적용할 수 있다고 하였을 때 최적의 선제적 설계 범위는 무엇일지에 대한 의사결정이다. 이는 각각 선제적 설계 범위를 도출할 때 적용할 재작업 임팩트 임계값 θ에 대한 민감도 분석, 최종 선제적 설계 범위의 합집합을 설계 네트워크에 적용하는 내용이다. 현재의 의사결정 모델 아래에서 이러한 사항에 대해 최적화된 값을 찾을 수 있는 알고리즘을 연구 범위에 포함하지는 않았으나, 이에 대한 추가적인 고려를 통해 보다 합리적인 의사결정을 내리는 것에 도움을 줄 수 있을 것으로 예상한다.

최근 기업들은 단일 제품의 설계를 넘어 여러 제품을 보다 효율적으로 설계하고자 노력을 기울이고 있다. 그러나 여러 제품을 효율적으로 개발하는 것은 부품 공용화, 제품 포트폴리오의 구성 등 새로운 문제를 야기한다. 실제 자동차의 제품군 설계 중 부품 공용화 단계에 설계 규칙 단계를 따로 두어 성공적인 결과를 거두었던 Tomoatsu(2018)의 사례, 아키텍처-모듈-포트폴리오에

대한 설계 규칙을 통해 제품 개발과 생산을 아우르는 의사결정을 내리고자 제안한 Løkkegaard et al.(2018)의 연구, 제품군 내에서 여러 부품 간 상호종속성을 해결하고자 하였던 Cheng et al.(2018)의 연구들을 참고할 때, 설계 규칙은 여러 제품을 개발할 때 발생할 수 있는 문제들을 해결할 수 있는 전략적인 의사결정이며 설계 규칙의 실현 가능성을 제고한 본 연구 또한 의미를 더할 것이라 기대할 수 있다.

5.2 향후 연구

본 연구에서 명시적으로 고려하지 않았으나, 선제적 설계의 도입에 있어 추가적으로 고려해야할 사항이 있다.

우선 선제적 설계로 인해 발생하는 설계 실패와 관련한 부분이다. 설계 실패의 위험성을 고려한 선제적 설계이지만, 이 또한 특정 설계 변수를 고정하는 결과를 의사결정인 바 다른 설계 변수의 값을 결정하는 것을 방해하거나 불가능하게 만든다. 이와 같은 문제의 발생에 대하여 두가지 해결 방안으로 1) 설계 규칙으로 고정한 설계 변수의 값을 바꾸거나 2) 추가적인 인력과 시간을 들여 발생한 문제에 대한 해결책을 찾는 것을 생각해볼 수 있다. 이는 각각 추가적인 설계 반복의 발생, 설계 과정을 수행하는 데 드는 비용의 증가 등의 역효과로써 나타난다. 이와 같은 역효과는 선제적 설계의 다각적인 분석을 위해 고려해야 할 문제이나 현재의 재작업량을 중심으로 본 연구의 방법론 아래에서는 다루기 어려운 내용이기에 추가적인 연구의 진행이 필요하다.

이에 더하여 선제적 설계의 대상이 되는 설계 변수를 고르는 과정에 있어

각 설계 변수가 가질 수 있는 값의 범위가 설계 실패의 중요한 요인으로 작용할 수 있다. 값을 고정하는 의사결정이 넓은 범위의 값을 가질 수 있는 설계 변수에 대해 이루어진 경우, 한편으로는 효율적인 의사결정이 됨과 동시에 다른 한편으로 다른 설계 변수의 설계를 크게 제약하는 것이 될 수 있다. 이러한 내용의 분석을 위해선 설계 변수 사이의 관계를 어떻게 정의할 것인지, 정의된 관계 아래서 설계 변수 사이의 값의 조정이 어떻게 이루어지는지에 대한 연구가 필요할 것이다.

설계 작업의 진행을 보다 구체적으로 모델링하여 현실에 가까운 설계 작업의 흐름 추적과 그에 따른 재작업을 파악할 수 있다. 본 연구의 모델링에서는 설계 작업이 완료가 되면 그와 연관 관계를 가지는 설계 작업이 반드시 재작업을 수행한다고 가정하였으나 실제 설계 작업은 설계 작업 결과물에 따라 재작업을 수행해야할 수도, 그렇지 않을 수도 있는 확률적인 움직임을 따른다. 즉 설계 프로세스는 확정적이기 보다는 확률적인 움직임을 가지며 이를 모델링에 반영하는 것은 의미 있는 작업이 될 것이다. 이에 더하여 재작업이 발생하는 정도에 대하여 설계 작업의 재작업량이 수행될 때마다 재작업 임팩트에 따르는 고정된 비율로 줄어든다고 가정하였으나 이러한 가정을 완화할 수 있다. 설계 작업을 거듭하여 수행할수록 작아지는 재작업량의 크기를 학습효과를 반영한 특정한 함수의 형태로 구체화할 수 있다. 다만 위와 같은 내용을 고려함에 있어 분석적인 접근으로는 한계가 있을 것으로 보이며 시뮬레이션 기법을 이용하여 설계 프로세스를 모델링하고 구하고자 하는 결과를 도출할 수 있을 것으로 예상한다.

참고 문헌

Amabile, T. M.(1982). Social psychology of creativity: A consensual assessment technique. *Journal of personality and social psychology*, *43*(5), 997.

Attari-Shendi, M., Saidi-Mehrabad, M. and Gheidar-Kheljani, J.(2019). A Comprehensive Mathematical Model for Sequencing Interrelated Activities in Complex Product Development Projects. *IEEE Transactions on Engineering Management*.

Badhrinath, K. and Jagannatha Rao, J. R.(1996). Modeling for concurrent design using game theory formulations. *Concurrent Engineering*, *4*(4), 389-399.

Baldwin, C. Y. and Clark, K. B.(2000). *Design rules: The power of modularity*(Vol. 1). MIT press.

Ballard, G.(2000, July). Positive vs negative iteration in design. In *Proceedings Eighth Annual Conference of the International Group for Lean Construction, IGLC-6, Brighton, UK*(pp. 17-19).

Bayus, B. L.(1997). Speed-to-market and new product performance trade-offs. *Journal of product innovation management*, *14*(6), 485-497.

Black, T. A., Fine, C. H. and Sachs, E. M.(1990). A method for systems design using precedence relationships: An application to automotive brake systems.

Biskjaer, M. M., Dalsgaard, P. and Halskov, K. (2014, June). A constraint-based understanding of design spaces. In *Proceedings of the 2014 conference on Designing interactive systems* (pp. 453-462). ACM.

Braha, D. (2016). The complexity of design networks: Structure and dynamics. In *Experimental Design Research* (pp. 129-151). Springer, Cham.

Braha, D. and Bar-Yam, Y.(2007). The statistical mechanics of complex product

development: Empirical and analytical results. *Management Science*, *53*(7), 1127-1145.

Browning, T. R.(2001). Applying the design structure matrix to system decomposition and integration problems: a review and new directions. *IEEE Transactions on Engineering management*, *48*(3), 292-306.

Carrascosa, M., Eppinger, S. D. and Whitney, D. E.(1998, September). Using the design structure matrix to estimate product development time. In *Proceedings of the ASME design engineering technical conferences(design automation conference)*(pp. 1-10).

Chanron, V. and Lewis, K. (2005). A study of convergence in decentralized design processes. Research in Engineering Design, 16(3), 133-145.

Chen, S. J. and Lin, L.(2002). A project task coordination model for team organization in concurrent engineering. *Concurrent Engineering*, *10*(3), 187-202.

Chen, J., Reilly, R. R. and Lynn, G. S.(2005). The impacts of speed-to-market on new product success: the moderating effects of uncertainty. *IEEE Transactions on engineering management, 52*(2), 199-212.

Cheng, X., Xiao, R. and Wang, H.(2018). A method for coupling analysis of association modules in product family design. *Journal of Engineering Design*, *29*(6), 327-352.

Cho, S. H. and Eppinger, S. D.(2005). A simulation-based process model for managing complex design projects. *IEEE Transactions on engineering management*, *52*(3), 316-328.

Clarkson, P. J. and Hamilton, J. R. (2000). 'Signposting', a parameter-driven task-based model of the design process. *Research in Engineering Design*, *12*(1), 18-38.

Cohen, M. A., Eliasberg, J. and Ho, T. H.(1996). New product development: The performance and time-to-market tradeoff. *Management Science*, *42*(2), 173-186.

Costa, R. and Sobek, D. K.(2003, January). Iteration in engineering design: inherent and unavoidable or product of choices made?. In *ASME 2003 International design engineering technical conferences and Computers and information in engineering conference*(pp. 669-674). American Society of Mechanical Engineers Digital Collection.

David, M.(2013). Organising, valuing and improving the engineering design process. *Journal of Engineering Design*, *24*(7), 524-545.

Denker, S., Steward, D. V. and Browning, T. R.(2001). Planning concurrency and managing iteration in projects. *Project Management Journal*, *32*(3), 31-38.

Eppinger, S. D.(1991). Model-based approaches to managing concurrent engineering. *Journal of Engineering Design*, *2*(4), 283-290.

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

Gebala, D. A. and Eppinger, S. D.(1991). Methods for analyzing design procedures.

Hossain, M. A. and Chua, D. K. H.(2014). Overlapping design and construction activities and an optimization approach to minimize rework. *International journal of project management*, *32*(6), 983-994.

Johnson-Laird, P. N.(1988). Freedom and constraint in creativity. *The nature of creativity: Contemporary psychological perspectives*, *202*.

Karniel, A. and Reich, Y.(2013). Multi-level modelling and simulation of new product development processes. *Journal of Engineering Design*, *24*(3), 185-210.

Krishnan, V., Eppinger, S. D. and Whitney, D. E.(1997). Simplifying iterations in cross-functional design decision making. *Journal of Mechanical Design*, *119*(4), 485-493.

Kusiak, A. and Park, K.(1990). Concurrent engineering: decomposition and scheduling of design activities. *THE INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH*, *28*(10), 1883-1900.

Kusiak, A. and Wang, J.(1993). Decomposition of the design process.

León, H. C. M., Farris, J. A. and Letens, G.(2012). Improving product development performance through iteration front-loading. *IEEE Transactions on Engineering Management, 60*(3), 552-565.

Lewis, K. and Mistree, F.(1998). Collaborative, sequential, and isolated decisions in design.

Løkkegaard, M., Mortensen, N. H. and Hvam, L.(2018). Using business critical design rules to frame new architecture introduction in multi-architecture portfolios. *International Journal of Production Research*, *56*(24), 7313-7329.

Maier, J. F., Wynn, D. C., Biedermann, W., Lindemann, U. and Clarkson, P. J.(2014). Simulating progressive iteration, rework and change propagation to prioritise design tasks. *Research in Engineering Design, 25*(4), 283-307.

Meier, C., Yassine, A. A. and Browning, T. R.(2006). Design process sequencing with competent genetic algorithms.

Onarheim, B.(2012). Creativity from constraints in engineering design: lessons learned at Coloplast. *Journal of Engineering Design*, *23*(4), 323-336.

Pektaş, Ş. T. and Pultar, M.(2006). Modelling detailed information flows in building design with the parameter-based design structure matrix. *Design Studies*, *27*(1), 99-122.

Schätz, B., Hölzl, F. and Lundkvist, T. (2010, March). Design-space exploration through constraint-based model-transformation. In *2010 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems* (pp. 173-182). IEEE.

Sharir, M.(1981). A strong-connectivity algorithm and its applications in data flow analysis. *Computers and Mathematics with Applications*, *7*(1), 67-72.

Smith, R. P. and Eppinger, S. D.(1997). Identifying controlling features of engineering design iteration. *Management science, 43*(3), 276-293.

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

Steward, D. V.(1981). The design structure system: A method for managing the design of complex systems. *IEEE transactions on Engineering Management*,(3), 71-74.

Suh, N. P.(1998). Axiomatic design theory for systems. *Research in engineering design*, *10*(4), 189-209.

Tang, D., Zheng, L., Li, Z., Li, D. and Zhang, S. (2000). Re-engineering of the design process for concurrent engineering. Computers & Industrial Engineering, 38(4), 479-491.

Tarjan, R.(1972). Depth-first search and linear graph algorithms. *SIAM journal on computing*, *1*(2), 146-160.

Thomke, S. H. (1997). The role of flexibility in the development of new products: An empirical study. *Research policy*, *26*(1), 105-119.

Thomke, S. and Fujimoto, T.(2000). The effect of "front-loading" problem-solving on product development performance. *Journal of Product Innovation Management: AN INTERNATIONAL PUBLICATION OF THE PRODUCT DEVELOPMENT and MANAGEMENT ASSOCIATION*, *17*(2), 128-142.

Tomoatsu, S.(2018). Managing new product development process and project leaders for modular architecture development: Framework of two-stage NPD process and module leaders.

Ulrich, K. T. and Eppinger, S. D.(1988). Product design and development, 1995.

Singapore. McGraw.

Wynn, D. C. and Eckert, C. M.(2017). Perspectives on iteration in design and development. *Research in Engineering Design, 28*(2), 153-184.

Yassine, A. A. and Braha, D.(2003). Complex concurrent engineering and the design structure matrix method. *Concurrent Engineering, 11*(3), 165-176.

Yassine, A. A., Falkenburg, D. and Chelst, K.(1999). Engineering design management: an information structure approach. *International Journal of production research*, *37*(13), 2957-2975.

Yassine, A. A., Joglekar, N., Braha, D., Eppinger, S. D. and Whitney, D. E.(2003). Information hiding in product development: the design churn effect. *Research in Engineering Design*, *14*(3), 145-161.

Yassine, A. A., Whitney, D. E., and Zambito, T.(2001). Assessment of rework probabilities for simulating product development processes using the design structure matrix(DSM).

Abstract

Preemptive Design for Establishing Feasible Design Rule

Jongwook Lim
Department of Industrial Engineering
The Graduate School
Seoul National University

Under rapidly changing market conditions, the release timing of a product is an important factor determining its success. Firms are continuing their efforts to shorten the design phase to advance the release. However, iterations and resulting reworks in the design process delays the whole design process.

As a way to reduce wasteful design, the establishment of design rule is currently used to determine the value of a particular design parameter. However, it is necessary to consider whether the value of design parameter should be fixed alone or together with those surrounding the design parameter. Such carefully orchestrated design rule is desired since doing so otherwise may result in unwanted modification of the design rule or in the degradation of product quality.

Therefore, this study expands the discussion of design rule in order to propose a strategic decision called preemptive design, which selects a certain range of design parameters and prioritizes the design tasks on them. The objective is to define feasible design rule by setting its scope, taking into account the associations between design parameters.

A graphical design network interprets how the existing design process changes and progresses as a result of the introduction of preemptive design. Based on the design network represented, the scope of preemptive design considering the importance of the association among design parameters is derived. The effects of applying each scope for preemptive design to the design process are analyzed from the perspective of rework in the design process.

Keywords: Preemptive design, Design iteration, Design rule, Design network
Student Number: 2018-25523