Journal of the Society of Korea Industrial and Systems Engineering
Vol, 31, No. 4, pp.168－176, December 2008.

소프트웨어 재사용 기법을 이용한 현장제어시스템의 구현 방법에 관한 연구

김종수† · 오현승

한남대학교 산업경영공학과

A Study on Implementation Method of Manufacturing Control Systems using Software Reuse

Chong-Su Kim† · Hyun-Seung Oh

Dept. of Industrial and Management Engineering, Hannam University

In an environment where manufacturing conditions such as product lines keep changing, manufacturing control systems need to be continually maintained and upgraded according to the change. This requires an effective system implementation methodology. In this paper, a methodology based on the multi-criteria similarity comparison of manufacturing processes is proposed. A newly introduced process is compared with existing ones based on the multi-criteria similarity, and candidates for software reuse are selected from stored modules according to the overall process characteristics. The proposed methodology has been tested for an electronics manufacturing company's manufacturing control system, and the result has been satisfactory in that it can save time and efforts for system implementation.

Keywords : software reuse, production control, RAS, similarity

1. 서 론

현장제어시스템은 컴퓨터통합생산의 핵심적인 부분으로 낮은 수준의 공정제어와 상위 업무시스템의 생산지시 사이에서 이루어지는 각종 생산관련 작업을 통제하는 역할을 한다. 데이터 가공의 측면에서 보면 현장제어시스템은 하위의 수집 장치에서 얻은 생산관련 데이터를 처리하여 상위의 업무시스템에 전달하고 상위로부터 받은 생산지시를 받아 추가로 가공하여 조업자에게 전달하는 기능을 제공한다[8]. 생산시점관리(POP, Point of Production), 생산현장제어(SFC, Shop Floor Control) 그리고 제조실행시스템(MES, Manufacturing Execution Systems)은 소프트웨어적 관점에서 현장제어시스템의 범주에 포함된다고 할 수 있다. 현장제어시스템의 구현 형태는 초기의 단독적인 시스템 방식에서 벗어나, 전사적 자원관리(ERP)나 제조실행시스템(MES) 등의 상위시스템 및 PLC와 같은 하드웨어와 실시간으로 연계되어 생산정보를 수집하고 설비를 제어하는 양방향 시스템으로 진화하고 있다[5].

이와 같은 기능을 지닌 현장제어시스템은 생산제품 포트폴리오의 변경과 같은 환경의 변화에 대응하여 지속적으로 갱신되고 유지, 보수될 필요가 있다. 예를 들어 신규 제품이 도입되면 그에 따라 기존의 생산설비가 재구성되어야 하며 경우에 따라서는 신규설비가 도입된다. 그러므로 생산설비의 구성에 의존하는 생산정보시스템은 변화된 환경에 맞추어 수정, 확장되는 것이 일

논문접수일 : 2008년 11월 03일 논문수정일 : 2008년 12월 09일 게재확정일 : 2008년 12월 11일
† 교신저자 ckim@hnu.kr
※ 이 논문은 2007년도 한남대학교 교비학술연구비 지원에 의한 것임.

반적이다. 특히, SFC나 POP와 같은 현장관리 수준의 시스템은 생산설비의 구성 및 운용방식에 직접적으로 영향을 받는데 생산 환경의 변화에 따라 시스템이 변경되는 경우 높은 비용과 많은 시간이 소요된다. 또한 유연생산을 지향하는 생산 체제에서는 상대적으로 짧은 시간 내에 현장제어시스템을 구축하여 제품구현 리드타임을 줄이는 것이 필수적이다[16]. 그런데 현장제어시스템의 구현에서는 대개 소프트웨어 개발이 병목공정이 된다는 경험적 사실을 고려하면 생산환경 변화에 대응하여 신속하게 현장제어시스템을 구축할 수 있는 방법론이 필요하며, 이를 위해서는 기존의 소프트웨어 자산을 재사용하는 것이 현실적인 접근방법이라고 할 수 있다.

또한 이러한 구축방법론은 소프트웨어를 재사용하는 기법뿐 아니라 재사용을 용이하게 하는 설계기법도 포함할 필요가 있다. 현장제어시스템의 구성형태와 운용요구조건들이 지속적으로 변화하는 환경에서는 제어에 사용되는 소프트웨어 시스템이 주변 환경의 변화에 유연하게 대처할 수 있도록 변경이나 재구성이 용이한 구조로 설계되어야 한다[14]. 이는 단순히 코드 수준의 재사용을 염두에 둔 구조적 고려에 그치지 않는다. 코드의 재사용을 위해서는 설계 자체의 재사용이 필요하고 이는 다시 해당 영역의 제품계열에 대한 분석이 요구된다. 즉, 재사용이 용이한 시스템을 설계하기 위해서는 소프트웨어의 수명주기상의 한 단계로서의 재사용을 염두에 두고 전체 제품라인을 구성할 필요성이 제기된다[10].

이상에서 기술한 요구사항에 대응하기 위하여 본 논문에서는 현장제어시스템 구축의 비용 절감과 개발기간 단축을 위하여, 컴포넌트의 기능적 다면성에 기반을 둔 소프트웨어 재사용 기법을 이용한 현장제어시스템 구현 방법이 제안된다. 제어 측면에서의 공정특성과 소프트웨어 모듈의 기능적 유사성을 복합적으로 고려하여 기존 모듈과의 유사성을 평가한 후 재사용하는 방법이 제시된다. 단위공정 사이의 유사성을 다면적 기준에 의해 평가한 후 공정별로 재사용할 소프트웨어 자산(모듈)의 후보군을 도출하고, 그 후보군 중에서 전체공정의 일관성을 충족시키는 재사용 모듈의 조합을 결정한다. 제안된 방법론을 적용하여 시스템 구축에 소요되는 시간 및 비용을 절감하고 산출물인 현장제어시스템의 품질에 대한 개선효과를 추구하는 것이 본 연구의 목적이다.

2. 관련 연구동향 및 문제점

2.1 소프트웨어 재사용

소프트웨어 재사용은 지식 자산의 형태로 축적된 이전의 개발 경험을 새로운 소프트웨어의 신규개발 및 업데이트에 다시 적용하는 개발방법이다. 이미 작성된 코드를 재사용하면 소프트웨어 개발의 생산성을 높이고 그 품질을 향상시키는데 도움이 된다. 소프트웨어 재사용의 방법론적 측면에서는 컴포넌트 기반의 개발기법 (CBSD, Component-based Software Development)이 가장 널리 사용되고 있다. 그런데 CBSD와 같은 방법론이 실제로 소프트웨어 생산성을 현저하게 향상시킬 수 있는가에 대하서는 많은 의문이 제기되어 왔다. 이는 컴포넌트의 구축시 가정된 조건과 재사용에 필요한 정보들이 컴포넌트와 분리된 채 존재하는 경우가 많아 실제로 개발자들이 얻을 수 있는 기간단축의 효과가 크지 않기 때문으로 분석된다. 그 결과 재사용할 수 있는 컴포넌트의 설계 및 개발은 아직 크게 활성화되어 있지는 않다[6, 11].

이러한 문제점을 해결하기 위해서는 재사용에 필요한 정보들을 구조적이고 명시적으로 표현하여 이용하는 체계가 필요하다. RAS(Reusable Asset Specification)는 재사용을 위한 소프트웨어 자산의 명세화를 위해 제정된 규약으로 자산기반의 소프트웨어 개발과 관련된 용어들을 표준화하고 소프트웨어 자산을 정의하고 분류하기 위한 모델을 제공한다[15]. RAS를 컴포넌트 기반 기법에 적용하여 기능, 납기, 품질 등의 요구사항을 모형화하고 이를 기반으로 컴포넌트 사이의 유사도를 측정하여 재사용 자산을 탐색하는 기법이 제시되었다[3]. 이들의 연구는 재활용의 주요 요소인 기능적 다면성을 명세화 표준을 통해 방법론에 수용했다는 점에서 주목할 만하지만 컴포넌트 사이의 유사성을 코드에 의존하여 평가하므로, 제어와 같이 하드웨어의 기능적 측면이 소프트웨어의 구성에 밀접하게 결합에 분야에 그대로 적용하기에는 여러 문제점이 있다고 판단된다.

현장제어 분야의 시스템 구축에 있어서 소프트웨어를 재사용하는 다양한 방안들과 적용 사례들이 제시되었다. 실시간 제어에서 소프트웨어를 입력과 출력, 그리고 데이터 처리 기능을 갖춘 필터블록으로 분해하여 이들 블록의 재구성을 통해 재사용성을 높이려는 컴포넌트 기반의 접근 사례와[4], 웨이퍼 가공과 같은 전자제조공정에서 현장제어전략을 고려한 제어시스템의 구성 전략[7] 등이 그러한 예이다. 또한 전체 제품계열의 공통점과 차이점을 분석하여 피처를 모델링하고 이를 기반으로 설계 단계에서부터 재사용을 위한 소프트웨어를 작성하는 기법인 FORM(Feature-Oriented Reuse Method)이 제철소 생산라인의 공정제어에 적용되었다[1, 10]. 이러한 기법들 대부분은 미리 재사용이 가능한 소프트웨어 혹은 컴포넌트의 구조를 설정하고 이에 맞추어 개발 작업을 수행한 후 그 결과를 향후의 개발에 재사용하는 Pre-project Analysis의 성격을 띠고 있다. 이들은 아키텍처

상에서 소프트웨어의 재사용을 촉진하므로 재활용의 효율성을 극대화할 수 있다는 장점이 있는 반면, 개발 프로젝트의 착수 이전에 대규모의 노력이 소요되고 기존에 이미 개발되어 있는 소프트웨어 자산들을 활용하는데 적절하지 않다는 문제점도 있다.

2.2 제조공정의 분류

현장제어시스템에서 소프트웨어 구성의 기본 단위는 제조공정이다. 따라서 소프트웨어 모듈이나 컴포넌트는 제조공정의 특성을 상당한 수준으로 반영하게 되는데, 이는 재활용이 가능한 자산 중 필요한 모듈을 찾는 작업이 제조공정을 비교, 분류하는 것과 밀접한 연관이 있음을 의미한다. 제조공정의 분류는 공정의 특성을 비교하는 직접적인 방법과 공정에 의해 생산되는 산출물의 특성을 비교하여 공정을 분류하는 간접적인 방법이 있다.

GT(Group Technology)는 공정 배치 등의 측면에서 제조 현장의 생산성을 증가시키기 위한 운용기법의 하나로서, 공정 산출물인 제품 및 부품의 형상에 따라 공정을 간접적으로 분류하는 방법이다. 유사한 제조설비를 필요로 하는 제품 혹은 부품들을 그룹으로 묶고 이의 생산에 필요한 기계들을 한 곳에 모아 활용함으로써 다품종 소량생산을 가능케 하는 것이 GT기술의 철학이다[12, 17]. GT기술을 적용하면 제품생산의 유연성을 증가시키면서도 대량생산의 이점인 규모의 경제를 극대화할 수 있는데, 구체적으로 생산 준비시간의 단축과 비용절감, 공정재고 감소와 같은 효과가 있다고 알려져 있다. 공정분류 방법으로서의 GT기술이 이용되는 대표적인 분야는 배치유형의 생산에 있어서 생산성 향상을 목적으로 하는 셀 제조이다. 셀 제조에서 주요 계획 작업은 부품그룹과 그에 연관된 기계장비들을 그룹화시키는 일인데, GT기술을 이용하여 일단의 부품 그룹을 생산하기 위하여 셀 사이의 상호작용이 가장 낮은 상호독립적인 기계 셀을 찾는다[13].

그러나 GT기술은 본 논문의 관심분야인 현장제어시스템 구축에서의 소프트웨어 재활용을 위한 공정분류의 방법으로는 적합하지 않다. GT기술에서는 제조 셀에서의 공정의 흐름이 일차적인 분류기준이 되는 반면, 현장제어시스템에서는 단위공정들 사이의 실행순서보다는 단위공정 내부에서의 각종 하드웨어 구성 및 공정제어 및 데이터 수집용 모듈의 특성이 공정을 분류하는 척도가 된다. 또한, GT에서의 분류는 가공해야 할 부품의 형상에 의존하므로 제조공정이 제품의 형상에 의존하지 않는 비기계 분야에서 적용하기는 쉽지 않다. 그리고 정태적인 부품집합과 공정집합을 가정하는 GT기술과는 달리, 현장제어시스템은 신규 제품이나 신규 단위공정이 도입되는 상황에 초점을 맞추므로 변화하는 공정에 점진적으로 적용할 수 있는 동적인 분류체계가 요구된다.

3. 소프트웨어 재사용 방법론의 설계

3.1 접근 방법

제 2장에서 설명된 바와 같이 현장제어시스템의 구현에 있어서 소프트웨어를 재사용하는 경우 기존의 연구에서 제시된 방법들을 그대로 적용하기는 어렵다. 코드 위주의 유사도 비교는 하드웨어에 밀접하게 연관된 제어 소프트웨어의 특성상 효과가 적고, 현장제어 전략을 적절하게 모형화하고 분석하는데 사용할 수 있는 공정 분류방법도 존재하지 않는다. 이러한 문제점을 고려하여 전체 공정 간의 유사도를 비교하여 현장제어의 하위 시스템인 생산시점관리시스템을 구축하는 방법론이 제시되었다[2]. 이 방법에서는 신규로 도입되는 공정 T가 기존의 공정 Q와 비교되고 그 비슷한 정도가 공정 유사도 $s(T, Q)$의 형태로 계산된다. $s(T, Q)$는 T의 각각의 단위공정 T_i를 Q의 개별공정들과 비교한 결과에 의해 산출된다. 이렇게 신규 도입공정 T를 Q와 같이 기존에 구축된 공정들과 각각 비교한 후 유사도가 가장 높은 기존 공정을 선택하여 그에 해당하는 소프트웨어 자산을 재사용하게 된다.

그러나 이 방법은 다음과 같은 한계점을 지닌다. 우선 개별 공정이나 개별 소프트웨어 모듈을 여러 공정에서 선별적으로 골라 재사용하지 않고 전체 제조공정의 단위에서 상호 비교하여 가장 유사한 공정을 선택한 후 사용하므로 소프트웨어 재사용의 효율성을 극대화하기 어렵다. 또한 유사도 평가에 있어서는 프로세스의 특성만을 단순 비교하므로 공정순서의 의미가 제대로 고려되지 않고, 단위공정들이 선형으로 결합되는 전체공정을 가정하기 때문에 비선형 공정에 적용하기 어렵다. 그리고 단위공정 사이의 유사도 평가가 평가자의 경험과 직관에 의해 이루어지므로 평가의 객관성이 결여되기 쉽고 향후 유사한 문제를 해결할 때 축적된 평가 결과를 시스템적으로 활용하기 어렵다.

위와 같은 한계를 극복하고 보다 효율적인 현장제어시스템의 구현 방법론을 제시하기 위하여 본 논문에서는 아래와 같이 소프트웨어 재사용에 대한 새로운 접근방법을 설정하였다. 유사도 평가에 있어서 다면적 컴포넌트 평가를 기반으로 하는 소프트웨어 재사용 기법을 응용하였고, 향후의 자동화를 고려하여 체계적이고 절차적인 방법론이 될 수 있도록 세부사항을 구성하였다.

소프트웨어 공학의 측면에서 보면 본 논문의 접근방법은 재활용 기법상 Post Project Generalization에 해당된다[9]. 즉, 기존에 이미 작성된 컴포넌트 혹은 모듈과 같은 소프트웨어 자산이 향후 재사용의 가능성이 있다고 판단되는 경우에는 별도의 메커니즘을 통해 이를 자산으로 등록하고 재사용시 이들 자산을 활용한다. 이런 접근방법을 통해 제 2장에서 설명한 기존의 제어 소프트웨어 재사용 기법들의 문제점을 상당 부분 해결할 수 있을 것으로 기대한다.

① 개별공정 사이의 유사도를 공정특성 및 해당 소프트웨어 모듈의 기능에 의거하여 소프트웨어 재사용의 관점에서 다면적으로 평가한다.
② 개별공정 사이의 유사도를 평가하는데 있어서 인접한 공정의 요소를 포함시키는 방법으로 공정의 순서를 고려한다.
③ 공정을 전체로서 이용하는 장점을 유지하기 위하여 동일한 제조공정에 속하는 단위공정의 모듈수가 최대가 되도록 재활용 모듈을 결정한다.

3.2 구성개념의 표현

일반적으로 물성을 변화시키는 단위조작의 집합을 제조공정이라고 한다. 기계부품과 같은 파트의 제조에서는 이들 단위조작들이 순차적으로 실행되는 것이 보통이지만 복잡한 조립품의 경우 병렬적 혹은 반복적으로 적용되기도 한다. 즉, 공정을 구성하는 단위조작 사이에는 순서 개념이 존재한다. 따라서 정보가공 및 유통의 측면에서 보면 제조공정은 단위조작과 그들 사이의 관계로 이루어진다. 이는 제조공정은 단위조작들의 집합에 대한 함수로 표현될 수 있음을 의미한다. 본 논문에서는 전체적인 제조활동을 나타내는 경우를 공정 혹은 전체공정이라고 칭하고 재료의 물성을 변화시키는 단위조작을 의미하는 경우를 단위공정이라고 한다.

본 논문에서 제안되는 방법론을 적용하기 위해서는 전체 공정을 일련의 단위공정으로 분해해야 한다. 이 단위공정을 나누는 기준은 해당 어플리케이션의 맥락에 의해 결정된다. 예를 들어, 생산제어형 어플리케이션에서는 단위공정이 일반적으로 가공의 단위체로 나눠진다. 반면 물류관리를 위한 어플리케이션에서는 loading, unloading, buffer와 같은 물류개념이 단위공정 구분의 기준이 된다. 본 논문의 목적은 현장관리시스템을 구축하기 위한 소프트웨어 재사용 방법론을 제시하기 위한 것이므로 각 단위공정은 해당 단위에 대한 현장제어 소프트웨어가 하나의 모듈로 구성될 수 있도록 설정된다. 즉, 단위공정과 소프트웨어 모듈은 일대일로 대응한다. 여기서 모듈은 컴포넌트와 같이 재사용의 단위가 되는 소프트웨어 개체이다.

위에서 설명된 전체공정 및 단위공정의 개념과 다음 절에서 제시되는 방법론의 기술에 사용되는 개념들을 표시하는 기호들은 다음과 같이 정리될 수 있다.

$P(i)$: 전체공정 i
P_{ij} : 전체공정 $P(i)$의 j번째 단위공정
M_{ij} : 단위공정 P_{ij}에 대응되는 소프트웨어 모듈
$s(P_{0k}, P_{ij})$: 단위공정 P_{0k}와 P_{ij}와의 유사도
c_k : P_{0k}의 재사용 후보군 선정을 위한 유사도의 최저 기준값
C_{ij} : 단위공정에 대한 구분(기능구분)
A_{ij} : 인접공정의 구분
B_{ij} : 단위공정 P_{ij}에 대응되는 모듈 M_{ij}의 특성

3.3 방법론

앞에서 기술한 새로운 접근방법에 의한 현장제어시스템의 구축을 위한 소프트웨어 재사용 방법론이 아래에 설명된다. 이를 요약하면 <표 1>과 같다.

<표 1> 소프트웨어 재사용 방법론

신규도입공정 $P(0)$가 n개의 단위공정 P_{0k}(k = 1, 2, ···, n)로 구성되어 있을 때,

단계 1 : 단위공정 사이의 유사도의 계산
단위공정 P_{0k}와 기존의 모든 공정 P_{ij}와의 유사도를 계산한다.
$s(P_{0k}, P_{ij}) = f(C_{0k}, A_{0k}, B_{0k}, C_{ij}, A_{ij}, B_{ij})$

단계 2 : 재사용 모듈 후보군 선정
$s(P_{0k}, P_{ij}) >= c_k$인 공정들을 선택하고 그들의 대응모듈집합 $\{M_{ij}\}$을 재사용 모듈 후보군으로 선정한다.

단계 3 : 재사용 모듈의 결정
- 재사용 모듈 후보군 중 기존의 공정들 각각이 나타나는 빈도를 측정하여 가장 빈도가 높은 기존 공정을 주공정 $P(m)$로 선정한다.
- 신규도입공정 $P(0)$의 각 단위공정 P_{0k}에 대해 다음의 기준을 순서대로 적용하여 재사용 모듈을 결정한다.
  (1) 주공정 $P(m)$에 속하는 모듈
  (2) $s(P_{0k}, P_{ij})$값이 최대인 모듈

단계 1에서는 신규로 도입된 공정 $P(0)$의 k번째 단위공정 P_{0k}를 기존의 단위공정들과 비교하여 유사한 공정들을 찾아낸다. 유사성은 해당 단위공정의 기능적 속성들의 값에 의해 산정되는데, 이들 속성항목들과 유사도 계산에서 사용되는 가중치는 응용영역별로 달라질 수 있다. 본 논문의 사례연구에서 사용된 전자부품제조의 경우 <표 2>와 같은 속성들을 유사도 계산에 사용한다.

<표 2> 단위공정의 속성항목(전자부품제조)

기호	항목명	설 명
C_{ij}	공정구분	해당 단위공정의 종류를 나타낸다(예 : 자재, 측정, 증착, 가공, 검사, 수입, 포장, 물류 등).
A_{ij}	인접공정	인접한 공정의 종류를 나타낸다.
B_{ij}	모듈특성	입력점의 숫자(b1), 입력장치의 종류(b2) 및 인터페이스 되는 상위 시스템의 특성(b3)을 포함한다.

단위공정 P_{0k}와 비교의 대상이 되는 공정 P_{ij}와의 유사도는 $s(P_{0k}, P_{ij})$로 표시하며 0과 1사이의 값을 갖도록 정규화한다. 즉, 단위공정 사이의 유사도는 다음과 같이 표현될 수 있다.

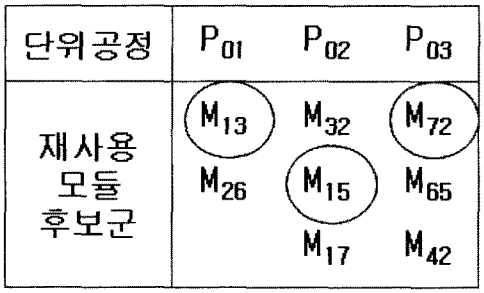
$$s(P_{0k}, P_{ij}) = f(C_{0k}, A_{0k}, B_{0k}, C_{ij}, A_{ij}, B_{ij})$$
$$0 =< s(P_{0k}, P_{ij}) =< 1$$

위와 같은 방법으로 모든 유사도 값이 계산된 후 재사용 모듈의 후보군을 선정할 수 있다(단계 2). 본 논문에서 제안되는 방법론에서는 두 단위공정 사이의 유사도$s(P_{0k}, P_{ij})$가 미리 정해진 특정값 c_k 이상인 공정의 모듈집합$\{M_{ij}\}$를 재사용이 가능한 후보군으로 간주한다.

마지막으로 단계 2에서 선정된 후보군 중에서 재사용할 모듈을 결정한다(단계 3). 본 방법론에서는 동일한 제조공정에 속하는 단위공정의 모듈수가 최대가 되도록 재사용 모듈을 결정한다. 이러한 제조공정을 주공정이라 칭한다. 주공정에 속하는 모듈수를 최대화시키는 이유는 개별공정에 대한 유사도가 상대적으로 낮더라도 공통의 전역적 특질(Global Properties)을 공유하는 경우에는 개발소요시간의 측면에서 코드 재사용의 효율성을 높일 수 있기 때문이다. 예를 들어 소프트웨어 모듈의 경계를 넘어서는 전역변수와 같은 데이터 구조가 동일하거나 유사하다면 소프트웨어를 수정하는 작업이 용이해진다.

단계 3은 다음과 같이 진행된다. 신규로 도입되는 공정 $P(0)$의 단위공정 P_{0k} 각각에 대하여 기존 공정 $P(m)$의 단위공정인 P_{mj}가 나타나는 빈도를 측정한다. 이때 어떤 단위공정 P_{0k}에 대하여 $P(m)$에 속하는 단위공정이 2회 이상 나타나는 경우에는 처음의 경우를 제외한 나머지는 빈도에 포함시키지 않는다. 모든 기존 공정에 대하여 이러한 빈도측정을 실시한 후 빈도가 가장 높은 공정 $P(m)$를 주공정으로 선택한다. 주공정이 정해지면 재사용할 모듈을 결정할 수 있다. 신규 도입공정인 $P(0)$의 각각의 단위공정 P_{0k}에 대하여 주공정 $P(m)$에 속하는 단위공정이 있으면 그 중 유사도가 가장 높은 공정을 선택하고 그 공정의 모듈을 재사용한다. 만일 주공정 $P(m)$에 속하는 단위공정이 존재하지 않는 경우에는 전체공정과 관계없이 유사도가 가장 높은 단위공정과 그 모듈을 택한다.

이상에서 기술한 단계 3의 재사용 모듈의 결정 과정을 예를 통하여 도식적으로 설명하면 <그림 1>과 같다. 신규도입공정 $P(0)$의 세 단위공정 P_{01}, P_{02} 및 P_{03}에 대한 재사용 모듈의 후보들 중 기존 공정 $P(1)$의 출현빈도가 2회로 가장 높다. 따라서 $P(1)$이 주공정이 된다. 제안된 방법론에 따르면 P_{01} 및 P_{02}의 구현을 위해 재사용되는 모듈은 각각 M_{13}, M_{15}이다. P_{01}에 대해서는 주공정인 $P(1)$에 속하는 단위공정의 모듈은 M_{13}이 유일하며, P_{02}의 경우에는 M_{15}와 M_{17}이 모두 $P(1)$에 속하지만 유사도가 더 높은 M_{15}가 선택된다. P_{03}에 대해서는 주공정인 $P(0)$에 속하는 모듈이 존재하지 않으므로 유사도가 가장 높은 모듈인 M_{72}를 선택한다.



<그림 1> 재사용 모듈의 결정 과정의 예

4. 적용 사례

본 논문에서 제안하는 방법론을 통신용 전자부품을 제조하는 H사의 현장제어시스템 개발 사례에 적용한다. 신규로 도입한 공정에 대한 시스템 개발에 있어서 실제로 재사용된 모듈과 제안된 방법론에 의해 도출된 결과를 비교하여 제안된 방법론의 타당성을 분석하는데 사용한다.

4.1 개요

H사는 생산현장을 제어하기 위한 시스템을 구축한 후 수년 간 운영해 오고 있다. 상위 시스템에서 생산지시가 입력되면 생산라인 별로 작업자에게 생산지시가 내려지고 작업을 위해 장비가 가동되면 장비 이력, 작업자 정보 등이 시스템의 서버로 전송된다. 시스템은 생산 실적정보 및 원자재 불량률을 포함한 생산과 관계된 정보

를 취합하여 생산성 분석, 비가동 분석, 불량 분석 등의 작업을 행하며 자재 재고, 공정 재고 및 수불 현황 등을 파악하여 상위 시스템에 전달한다. H사는 이러한 정보들을 생산운용뿐 아니라 작업자 교육 및 작업장 환경 개선, 설비개선, 설비 이력관리 등에도 이용하고 있다.

이 회사의 제품 구성은 수시로 변동되는 특성을 지닌다. 따라서 그 동안 개발된, 그리하여 재사용의 대상이 되는 단위공정에 대한 소프트웨어 모듈의 수는 320여 개에 이른다. 본 논문에서는 제안된 방법론의 적용성을 검토하고 그 효과를 측정하기 위하여 320여 개의 모듈 중 92개를 선정하여 제안된 방법론을 적용할 수 있도록 공정특성을 부여하였다. 그리고 가장 최근에 도입된 컷오프 필터의 생산 공정 $P(0)$중 4개의 단위공정을 골라 제안된 방법론을 적용시켜 보았다.

4.2 단위공정 사이의 유사도 계산

제 3장에서 설명된 바와 같이 단위공정 사이의 유사도는 해당 소프트웨어 모듈의 구성에 영향을 미치는 공정특성을 비교하여 계산한다. 이 공정특성에 속하는 속성 항목들과 그들이 유사도에 미치는 영향의 정도를 나타내는 가중치는 응용영역별로 달라질 수 있다. 본 논문에서는 H사의 통신부품 제조공정에 대하여 적용되는 속성 항목 및 가중치를 <표 3>과 같이 정하였다.

<표 3> 공정특성의 속성항목 및 가중치

공정 특성	세부 항목	가중치
공정구분	-	30%
인접공정	-	20%
모듈특성	입력점수	20%
	입력장치 종류	20%
	상위 시스템	10%

이제 재사용 대상이 되는 모든 기존 공정에 대하여 $P(0)$의 4개의 단위공정 P_{01}, P_{02}, P_{03}, P_{04} 각각과의 유사도를 계산할 수 있다. 그 과정을 P_{01}을 예로 들어 설명하면 다음과 같다. 우선, P_{01} 공정은 검사공정으로 측정과 수입을 각각 Input, Output에 해당하는 인접공정으로 두고 있다. 입력점의 숫자는 12개이며 입력장치는 주로 키오스크 형태로 되어 있다. 이 공정의 모듈 M_{01}과 연결되는 상위 시스템은 생산 및 품질 시스템이다. P_{01}과 두 개의 기존 공정 P_{37}, P_{26}과의 유사도 계산방식 및 결과는 각각 <표 4> 및 <표 5>와 같다.

<표 4> 유사도 계산의 예($s(P_{01}, P_{37})$)

	항 목	값	점수 (X100)
C_{ij}		검사	100 X 0.3
A_{ij}		측정/포장	50 X 0.2
B_{ij}	b1	6개	100(1-(12-6)/12) X 0.2
	b2	바코드	30 X 0.2
	b3	품질	60 X 0.1
유사도			$s(P_{01}, P_{37}) = 0.72$

<표 5> 유사도 계산의 예 ($s(P_{01}, P_{26})$)

	항 목	값	점수 (X100)
C_{ij}		증착	20 X 0.3
A_{ij}		자재/가공	
B_{ij}	b1	14개	100(1-(14-12)/12) X 0.2
	b2	키오스크	100 X 0.2
	b3	생산/품질	100 X 0.1
유사도			$s(P_{01}, P_{26}) = 0.55$

C_{ij} 항목에 대한 유사도는 두 비교 공정의 공정구분이 동일하면 만점인 100점, 그렇지 않으면 공정의 닮음 정도에 따라 미리 정해진 값을 부여한다(증착의 경우는 20점). 이 점수는 해당 항목의 가중치로 곱해진 후 유사도 값을 얻기 위해 다른 항목들의 점수와 합산된다. A_{ij}는 인접공정의 차이에 따라 결정된다. Input/output이 모두 같은 경우에는 100점, 하나만 같은 경우에는 50점이 주어진다. B_{ij}의 b1은 양 단위공정의 입력점 개수의 차이가 클수록 그 크기에 비례하여 낮은 점수가 배정된다. P_{37}의 경우 입력점의 개수는 P_{01}의 12개의 절반인 6개이므로 100점의 절반인 50점이 배정된다. 다른 항목들도 이와 유사한 방법으로 계산되었다.

4.3 재사용 모듈의 결정

제 4.2절에서 설명한 방식으로 신규도입공정 $P(0)$의 4개 단위공정 각각에 대하여 재사용 후보 모듈과의 유사도를 계산하였다. 재사용 모듈을 결정하기 위한 단계 3의 방식을 적용함에 있어서 유사도의 최저기준값 c_k는 모든 k에 대해 일률적으로 0.6으로 설정하였다. 그 결과는<표 6>과 같다.

<표 6>의 결과에 방법론의 단계 3의 절차를 적용하면, 우선 $s(P_{0k}, P_{ij}) > c_k = 0.6$인 재사용 가능 모듈 중에서 단위공정 별로 출현 빈도가 가장 높은 공정은 $P(1)$으로 3번이다. P_{03}에서와 같이 전체공정 $P(1)$에 속하는 모듈이

<표 6> 단위공정별 재사용 후보 모듈

	P_{01}	P_{02}	P_{03}	P_{04}
1st	***M12*** ***(0.73)***	-	M_{41} (0.83)	***M17*** ***(0.75)***
2nd	M_{37} (0.62)	-	***M18*** ***(0.69)***	-
3rd	-	-	M_{14} (0.61)	-
< c_k		***M37*** ***(0.54)***		

2회 이상 나타나면(M_{18}과 M_{14}) 한 번만 출현빈도에 포함시킨다. 따라서 $P(1)$이 주공정이 되고, 주공정의 모듈을 포함하는 공정 P_{01}, P_{03}, P_{04}에 대해서는 각각 M_{12}, M_{18}, M_{17}을 재사용 모듈로 선정한다. P_{02}의 경우에는 주공정인 $P(0)$뿐만 아니라 다른 공정에서도 유사도가 최저 기준값 c_k를 넘는 모듈이 없다. 따라서 유사도가 계산된 모듈 중 가장 높은 값을 갖는 M_{37}을 재사용한다.

4.4 결과 분석

위의 적용 예에서 나온 결과는 [M_{12}, M_{37}, M_{18}, M_{17}]을 재사용하는 것이다. $P(1)$에서 3개의 모듈을, $P(3)$에서 1개의 모듈을 선택하여 재사용한다(<표 7>의 결과 A). 그런데 [2]에서 제안된 방법인 전체공정 비교방법론에서는 단 하나의 기존 전체공정을 선택하여 신규로 도입되는 공정의 모든 단위공정을 구축하는데 재사용한다. 위의 예와 같은 경우라면 주공정인 P(1)이 선택되며 M_{37} 대신 M_{16}이 재사용된다(결과 B). 또한 본 논문에서 제안된 방법론에서 단계 3을 적용하지 않는다면, 즉 공통의 전역적 특질을 고려하지 않는다면, 각 단위공정 별로 유사도가 가장 높은 모듈이 선택되므로[M_{12}, M_{37}, M_{41}, M_{17}]이 재사용된다(결과 C).

위의 세 가지 결과를, 실제로 신규도입공정 $P(0)$에 대한 현장제어 모듈을 개발할 때 개발자들이 위에서 언급된 어떤 방법론도 사용하지 않고 자신의 경험에 의하여 선택한 모듈의 조합과 비교하면 <표 7>과 같다. 실제로 시스템 구현을 수행한 개발자들에 의하면 그들이 실제로 재활용 모듈을 선택할 때 P_{02}는 유사한 모듈을 찾기 어려워 처음부터 새로 개발하는 방법을 택했다. 이를 바탕으로 판단하면 본 논문에서 제안된 방법론을 적용한 결과인 A가 전문성을 갖춘 개발자들이 경험에 의존하여 선택한 모듈의 조합에 가장 근사함을 알 수 있다. 비록 본 논문에서의 적용 예가 전체공정 $P(0)$의 일부 단위공정에 국한되어 있고 다른 선택과의 비교가 개발시간과 같은 정량적인 지표에 의한 것이 아니라 선택된 모듈들의 합치도를 사용한다는 한계는 있지만, 제안된 방법론의 가능성을 보여주는 데는 큰 문제가 없다고 판단된다.

<표 7> 결과 비교

	P_{01}	P_{02}	P_{03}	P_{04}
결과 A	M_{12}	M_{37}	M_{18}	M_{17}
결과 B	M_{12}	M_{16}	M_{18}	M_{17}
결과 C	M_{12}	M_{37}	M_{41}	M_{17}
개발자 선택	M_{12}	없음	M_{18}	M_{17}

위의 비교에서는 개발 시간이 실제로 어느 정도 단축되었는가라는 정량적 지표가 명시적으로 제시되지 않기 때문에 제안된 방법론이 어느 정도로 소프트웨어의 개발 효율성을 개선시키는지 알기 어렵다. 따라서 본 논문에서는 제안된 방법론이 실제로 개발자로 하여금 재사용 모듈 탐색시간과 개발시간을 단축시킬 수 있는지를 다음과 같이 알아보았다. 신규로 도입되는 공정 $P(0)$의 단위공정 P_{06}을 선택하여 제안된 방법론을 사용하지 않는 경우의 해당 개발자의 모듈탐색시간과 개발시간을 측정하였다(이 경우 개발자는 자신의 경험과 직관, 기타 방법들에 의거하여 재사용할 모듈을 탐색하고 수정하여 모듈을 개발하게 된다). 그리고 제안된 시스템이 추천하는 재사용 모듈 2가지를 제시하고 그를 수정하는 경우의 개발시간을 측정하여 비교하였다. 그 결과는<표 8>과 같이 나타난다.

<표 8> 탐색시간과 작성시간의 비교(단위 : 분)

구 분	재사용 모듈	개별 유사도	탐색 시간	작성 시간
독립적탐색	M_{14}	0.72	90	71
시스템추천	M_{67}	0.79	0	85
	M_{14}	0.72	0	93

위의 결과를 보면 개발자가 독립적으로 재사용 모듈을 찾는 경우에도 제안된 방법론이 추천하는 모듈 중의 하나인 M_{14}을 선택했음을 알 수 있다. 또한 해당 단위공정의 모듈을 구현하는데 소요되는 총 시간은 161분에서 90분 수준으로 단축되었다. 이 실험결과에서 동일한 모듈인 M_{14}에 대한 작성시간이 독립적 탐색인 경우 다소 짧게 나타나는 것은 개발자가 재사용할 모듈을 탐색하는 동안 코드의 내용을 어느 정도 숙지하게 되므로 실제 코드를 수정할 때 시간이 단축되는 학습효과가 발생하기 때문으로 생각된다.

다음으로 제안된 방법론의 품질적 측면을 살펴보기 위하여 이미 개발된 8개의 모듈에 대해 실제 개발자가 어떤 모듈을 재사용했는지 조사한 후 시스템이 추천하는 모듈과의 합치도를 비교하였다(<표 9>). 비교의 대상이 되는 8개의 모듈은 2개의 전체공정에 속하는 8개의 단위공정을 제어하는 모듈이며, 이의 개발에 참여한 개발자는 3인이다. <표 9>에 의하면 개발자들은 실제로 하나의 경우를 제외하고는 개별 유사도가 가장 높은 모듈들 중 하나를 선택하여 재사용했음을 알 수 있다. 합치되지 않은 모듈은 GT제품을 만드는 공정의 자재반출 단위공정인데, 자재반출은 전체 공정의 특성과 별 관계없이 표준화되어 있기 때문에 상위 유사도를 가지는 모듈과 그렇지 않은 것들과의 유사도의 차별성이 없어서라고 판단된다.

<표 9> 추천 모듈의 합치도 비교

단위공정	추천된 모듈			실제선택
	1st	2nd	3rd	
1	M_{26}	M_{73}	M_{23}	1st
2	M_{19}	M_{36}	M_{33}	2nd
3	M_{36}	M_{54}	M_{64}	1st
4	M_{63}	M_{15}	M_{25}	1st
5	M_{14}	M_{28}	-	1st
6	M_{49}	M_{38}	M_{77}	3rd
7	M_{16}	M_{32}	-	기타
8	M_{65}	M_{72}	M_{18}	1st

<표 7>, <표 8> 그리고 <표 9>에서 알 수 있듯이 제안된 방법론은 현장제어시스템의 구축에 있어서 개발시간 단축에 효과적임을 알 수 있다. 또한 개발자들의 실제 선택이 합리적이라는 가정 하에, 방법론의 품질에 해당하는 추천의 정확도 측면에서도 만족스러운 결과를 보여주고 있다. 비록 신규로 도입되는 공정 전체에 적용하여 얻는 결과는 아니지만 위와 같은 부분적 실험에서 개발시간을 40% 정도 단축시키고 높은 정확도를 보여주었다는 사실은 제안된 방법론의 효율과 품질에 대한 가능성을 보여준다고 할 수 있다.

5. 결 론

생산현장에서 이루어지는 각종 작업을 통제하는 역할을 하는 현장제어시스템은 생산 환경의 변화에 대응하여 지속적으로 갱신되고 유지, 보수될 필요가 있다. 이에 필요한 시간과 노력을 줄이기 위해서는 생산 환경의 변화에 대응하여 신속하게 현장제어시스템을 구축할 수 있는 방법론이 필요하다. 본 논문에서는 이를 위하여 공정 특성의 다면성에 기반을 둔 소프트웨어 재사용 기법을 이용한 현장제어시스템의 구현 방법을 제안하였다. 단위공정 사이의 유사성을 다면적 기준에 의해 평가한 후 전체공정의 일관성을 고려하여 재사용 모듈을 결정하는 방법론을 적용하면, 시스템 구축에 소요되는 시간 및 비용, 그리고 산출물인 현장제어시스템의 품질에 대한 개선효과를 가져올 수 있을 것으로 기대된다. 본 논문에서 제안된 방법론은 단위공정 간의 유사도 평가에 공정의 다면적 특성을 비교하는 방법을 이용하는데 그 구체적인 계산방법은 응용 분야에 따라 달라질 수 있다. 따라서 향후 분야별로 특화된 유사도 계산 프레임이 정립되어야 한다. 또한 소프트웨어 모듈 사이의 유사도를 비교함에 있어서 해당 공정의 기능적 측면 이외에 코드 자체의 특성도 고려되어야하며, 그 비교방법도 온톨로지 모델과 같은 구조적인 절차가 되도록 방법론을 보완할 필요가 있다.

참고문헌

[1] 김근배, 강교철, 김경석, 이재준, 고은만, 양강선; "소프트웨어 제품 라인 공학을 적용한 공정 제어 시스템 개발 : 사례 연구," 소프트웨어공학회지, 18(1) : 33-49, 2005.

[2] 김종수, 김경택; "유사도 평가 방법론을 이용한 POP 시스템의 구현", 산업경영시스템학회지, 29(4), : 91-99, 2006.

[3] 박수진, 박수용, "컴포넌트의 다면성과 서비스를 기반으로 하는 재사용 모델", 정보과학회논문지 : 소프트웨어 및 응용, 34(4) : 303-316, 2007.

[4] 신동익, 전태웅, "필터 블록 조립 방식에 의거한 실시간 공정 제어 소프트웨어 아키텍처 설계방법", 한국정보과학회 학술발표대회, 24(2) : 543-546, 1997.

[5] 중소기업청, "중소기업 생산정보화자원 구축사례집", 2004.

[6] Fichman, R. G. and Kemerer, C. F.; "Incentive Compatibility and Systematic Software Reuse," *Journal of Systems and Software*, 57(1) : 45-60, 2001.

[7] Hsu, S. Y. and Sha, D. Y.; "The integration of shop floor control in wafer fabrication," *Journal of Manufacturing Technology Management*, 18(5) : 598-619, 2007.

[8] Huang, C. Y.; "Distributed manufacturing execution: A workflow perspective," *Journal of Intelligent Manu-*

facturing, 12 : 485-497, 2002.

[9] IEEE; "IEEE Standard for Information Technology-Software Life Cycle Processes-Reuse Processes," IEEE Std 1517-1999(R2004), 2004.

[10] Kang, K. C., Lee, J., and Donohoe, P.; "Feature-Oriented Product Line Engineering," *IEEE Software*, : 58-65, 2002.

[11] Keller, R. K. and Schauer, R.; "Design Components : towards software composition at design level," *Proceedings of the 1998 International Conference : Software Engineering*, : 302-311, 1998.

[12] Kusiak, A.; "The Generalized Group Technology Concept," *International Journal of Production Research*, 28 : 185-198, 1987.

[13] Kusiak, A. and Chow, W. S.; "Efficient Solving of the Group Technology Problem," *Journal of Manufacturing Systems*, 6(2) : 117-124, 2002.

[14] Natarajan, S. and Zhao, W.; "Issues in Building Dynamic Real-Time Systems," *IEEE Software*, : 16-21, 1992.

[15] OMG, *Reusable Asset Specification (RAS)*, OMG Document # ad/2003-10-12, Ver. 2.1, 2003.

[16] SAP, Inc.; "Integration of Manufacturing Execution Systems in Mill Industries," SAP Technical Bulletin, 2004.

[17] Won, S.; "Two-phase Approach to GT Cell Formation using Efficient p-median Formations," *International Journal of Production Research*, 38(7) : 1601-1613, 2000.