통합 Database를 기반으로한 다중 Process Engine 개발

박상민, 배재호, 구상엽, 왕지남, 김광섭
Department of Industrial Engineering
Ajou University Suwon 441-749, Korea
(fax) 0331-219-1610 (tel) 0331-219-2429
(Email) gnwang@madang.ajou.ac.kr

Abstract

본 논문에서는 통합 Logistics Center라는 새로운 조직을 통해서, 생산, 영업 및 물류를 총괄하여, 전체 공장과 전체 DC/SB의 재고량을 줄이고 결품현상 방지, 수송 비용 절감을 도모할 수 있는 통합 물류 시스템을 구축하는데 필요한 각 Process 및 이를 수행할 수 있는 발견적 해법을 개발하는 것을 연구한다.

생산(공장)에는 가장 정확한 각 DC/SB의 수요 예측량을 토대로 수립된 발주량과, 이 발주량을 통해서 결정된 생산의뢰 계획을 제공한다. 이 생산 계획은 제품이 생산되어서 출고될 때, 근거리 지역에 있는 DC/SB에 출고될 수 있도록, 각 공장의 전용라인에 대해서는 최소 수송 비용을 고려한다. 제품의 혼합생산 라인의 정보(생산 Capa.)를 이용하여, 나머지 발주량을 생산 의뢰 계획으로 바꾸게 된다. 공장에서는 이 계획을 바탕으로 생산 계획을 수립하여, 그 계획과 생산된 제품을 DC/SB로 출고시킨다. 공장의 출하가능량과 각 DC/SB의 2주전의 발주량은 실제 많은 차이가 발생하므로, 이를 재조정하여, 공장의 재고를 최소로 하는 방향으로 공장의 출하가능량을 조절한다. 조절된 출하가능량은 2일간의 근거리 수송 원칙에 의해서 입고될 DC/SB를 결정한다. 공장의 출고를 돕기위해서 배차계획을 수립한다.

본 연구에서는 각각의 Processs Engine들의 활동과 정보의 흐름이 서로 상이한 시간 간격으로 발생할 때 각 Process Engine 의 Input/Output으로 사용되는 정보의 효율적인 Systematic 동기화 및 Database와 interface 및 Multi-Tasking Database Transaction을 고려할 것이다.

1. 서 론

1.1 연구 배경 및 목적

지역적으로 분산되어 있는 식품 생산 공장 및 DC/SB간의 효율적인 물류 시스템을 구축하기 위해서는 생산,영업,물류 조직을 총괄할 수 있는 새로운 Logistics Center가 필요하게 되었다. 지금까지의 물류(제품의 수,배송)문제를 효율적으로 관리하지 못한 것은 물류를 각 부문에서 가지는 입장에 따라서 서로간의 목적이 다르기 때문이었다. 이에 본 연구에서는 각 부문이 아닌 기업 전체에 부합되는 목표를 설정하고, 이를 수행할 수는 있는 다중 Process를 설계하게 되었다. 또한, 각 Process는 Logistics Center에서 통합적으로 관리할 수 있도록 구현될 수 있도록 하였다.본 연구에서 설계한 Process Engine은 다음과 같이 나타낼 수 있다.

1. 과거 제품 판매의 지역별,기간별, 제품별 수요 예측 Process Engine
2. 재고 자동 발주 Process Engine
3. 생산 의뢰 계획 Process Engine
4. 재고 자동 보충 Process Engine

1.2 용어 설명

Sij(t+k) t+k 시점에서 j^{th} 공장에서 i^{th} DC/SB로의 수송량

Pj(t+k) t+k 시점에서 j^{th} 공장에서 출하 가능량 $Pj(t) = \sum_{i=0}^{I} Sij(t)$

Xi(t+k) t+k 시점에서 i^{th} DC/SB에서의 예상 요구량 $Xi(t) = \sum_{j=0}^{J} Sij(t)$

Cij j^{th} 공장에서 i^{th} DC/SB로 제품 1팔레트를 수송하는 수송 비용.

Pj(t+k-1) t+k시점에서 j^{th} 공장에서 당일 출하된량

Fj(t+k) t+k시점에서 j^{th} 공장의 당일 재고량

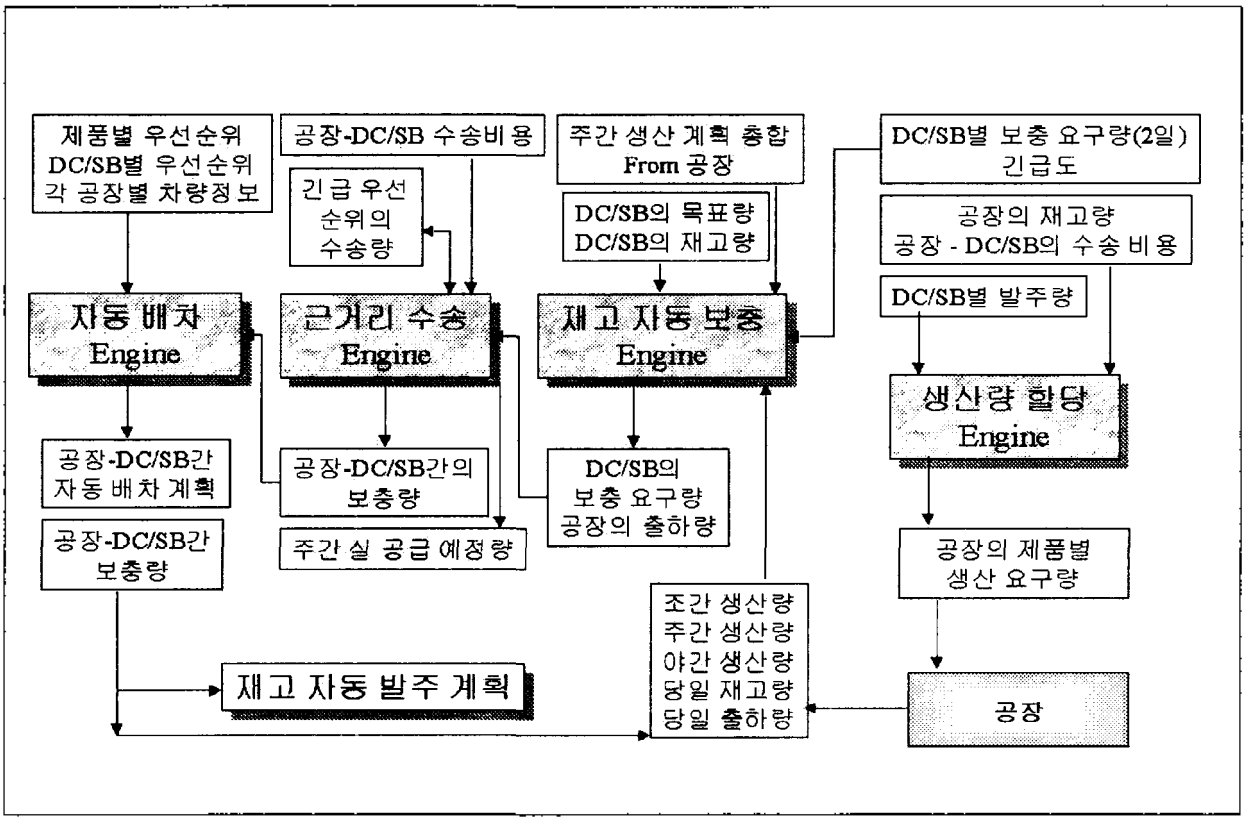
Oj(t+k) t+k시점에서 j^{th} 공장의 당일 생산량및 생산 예정량.

Qij(t+k) t+k시점에서 j^{th} 공장에서 i^{th} DC/SB로의 긴급 수송량.

본 논문에서는 [3,4]의 Process Engine에 관해서 그 내용을 기술하고, 전체 시스템의 흐름도 및 시스템 구현(Database, 분산 처리,시스템 동기화)에 관해서 논의하고자 한다.

2. 본론

2.1 시스템 흐름도.



[그림1.] 시스템 정보 흐름도

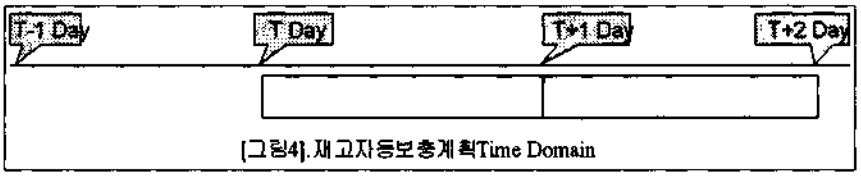
2.2. 재고 자동 보충 계획

2.2.1 재고 자동 보충 계획 구성

ㄱ. 재고 자동 보충 Process Engine
- T+1시점(내일), T+2시점(모레)의 공장의 출하 가능량과 DC/SB의 보충 요구량을 일치 시킨다.

ㄴ. 근거리 수송 Process Engine
- 공장에서 DC/SB로의 재고 보충시 일정 기간내에서 전체 수송비용이 최소가 될 수 있도록 공장-DC/SB의 제품별 수송 계획을 수립한다.

ㄷ. 자동 배차 Process Engine
- 각 공장별로 제품을 수송하기 위해서, 팔레트 단위의 제품을 차량단위로 배차시키는 배차 계획을 자동으로 수립한다.

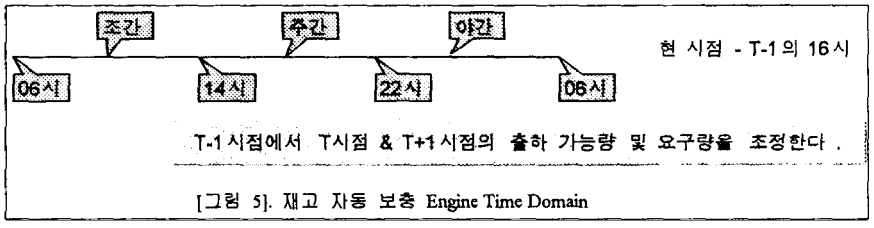


[그림4].재고자동보충계획Time Domain

2.2.2 Time Domain및 필요 정보

T Day 시점에서 [T+1,T+2]의 재고 자동 보충 Engine,근거리 수송 Engine을 수행한다.
T Day 시점에서 [T+1]의 자동 배차 Engine을 수행한다.

2.2.3 재고 자동 보충 Process Engine



[그림 5]. 재고 자동 보충 Engine Time Domain

1.Engine 목적
*.2일간의 전체 공장의 출하 가능량과 전체 DC/SB의 예상 요구량을 정확히 일치 시킨다.
*.각 공장의 재고량을 최대한 줄이는 방향으로 할당을 진행한다.

2.Engine 내용
1.IF Pj(t+k) = Xi(t+k) then Engine 완료
2.IF Pj(t+k) > Xi(t+k) then
1.First, assign Xi(t+k)
2.남은 Pj(t+k)는 DC/SB의 전체 최대재고량까지 기간별 목표량 대비 할당한다.
3.Database의 Xi(t+k)를 강제적으로 변경시킨후 완료.
3.IF Pj(t+k) < Xi(t+k) then
1.DC/SB의 기간별 목표량 대비 할당한다.
2.Database의 Xi(t+k)를 강제적으로 변경시킨후 완료.

2.2.4 근거리 수송 Process Engine

1.Engine 내용
여기에 사용한 Algorithm은 Transportation Problem을 사용하였다.

또한, [t,t+1]시점에 대해서 동시에 고려한 이유는 다음과 같다.

*.매일의 Optimal solution합은 일정기간 동안의 Optimal solution이 아니다.

*.만약 2일 이상까지 고려하면, 결품 현상이 빈번해지며,
또한,제품의 수송 lead time이 1일이기 때문이다.

아울러서, [t,t+1]시점에 대하여 각각 Optimal solution을 [t,t+1]을 동시에 고려한 algorighm수행의 Initial solution으로 사용한 이유는 다음과 같다.

*.[t,t+1]간의 물량 이동은 Main Database가 저장해야 하므로, Initial solution을 사용하지 않으면, algorighm수행에 부담이 크다.

2.Algorithm수행 절차.

1. t-1시점에 결정된 Qij(t)를 먼저 수행한다.
2. 공장의 출하 가능량을 변경한다.

$$Pj(t+k) - \sum_{i=1}^{n} Qij(t+k) \text{ For i=1,2,..n For j=1,2,...m}$$

3. t시점의 transportion problem algorithm을 수행한다.

$$\text{Minimize } Z = \sum_{i=1}^{n}\sum_{j=1}^{m} Cij * Sij(t) \text{ For i=1,2,..n For j=1,2,...m}$$

$$\text{Subject to } \sum_{i=1}^{n} Ei(t) = \sum_{j=1}^{m} Pj(t) \text{ For i=1,2,..n For j=1,2,...m}$$

4. t+1시점의 transportion problem algorithm을 수행한다.

$$\text{Minimize } Z = \sum_{i=1}^{n}\sum_{j=1}^{m} Cij * Sij(t+1) \text{ For i=1,2,..n For j=1,2,...m}$$

$$\text{Subject to } \sum_{i=1}^{n} Ei(t+1) = \sum_{j=1}^{m} Pj(t+1) \text{ For i=1,2,..n For j=1,2,...m}$$

5. [t,t+1]시점을 동시에 고려하여 transportion problem algorithm을 수행한다.

$$\text{Minimize } Z = \sum_{i=1}^{n}\sum_{j=1}^{m} Cij * Sij(t) + \sum_{i=1}^{n}\sum_{j=1}^{m} Cij * Sij(t+1) \text{ For i=1,2,..n For j=1,2,...m}$$

$$\text{Subject to } \sum_{i=1}^{n} Ei(t) + Ei(t+1) = \sum_{j=1}^{m} Pj(t) + Pj(t+1) \text{ For i=1,2,..n For j=1,2,...m}$$

6. [t,t+1]시점간의 물량이동에 대한 Database처리를 한다.
t+1시점의 Qij(t+k)도 계산하여 Database처리를 한다.

2.2.5 자동 배차 Process Engine

1.Engine 목적

각 공장별 차량 정보를 수집하여,위의 근거리 수송 Engine에 의해서 발생된 공장-DC/SB간의 수송량을 각 차량 단위별로 수송할 수 있는 chart생성을 그 목적으로 한다.

2.Engine 내용

1.동일 제품 1차량(8t,11t)은 우선적으로 결정한다.
2.나머지 물량은 우선 순위가 높은 DC/SB의 우선 순위가 낮은 제품부터 차량에 싣는다.

*참고 사항

우선 순위가 높은 제품은 자주 발주되고,수요량이 많은 것이다.
우선 순위는 제품 유형별로 그 기준이 다르며,tie가 발생할 수도 있으며,
사용자가 변경할 수도 있다.

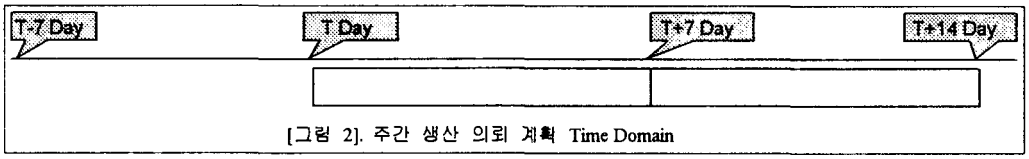
3.만약 1차량이 되지 못하는 제품이 발생하면, User option으로 처리한다.

1.내일 수송될 우선 순위가 높은 제품을 추가로 싣는다.
2.타 DC/SB에 제품을 넘긴다.

3.타 DC/SB의 남는 제품을 추가로 싣는다.
4.수송을 내일로 미루거나 취소한다.

2.3 생산 의뢰 계획

2.3.1 Time Domain및 필요 정보
ㄱ.월간 생산 의뢰 계획
매월초 한달간의 각 공장의 제품별 생산의뢰량을 결정한다.
ㄴ.주간 생산 의뢰 계획



[그림 2]. 주간 생산 의뢰 계획 Time Domain

T Day시점에서 [T+7 Day ~ T+13 Day]의 일(日)단위의 생산의뢰량을 결정.
현재는 매주 월요일에 시스템에 반영한다.
생산 의뢰 계획에서 공장에 제공되는 정보는 공장의 생산 계획을 작성하는데, 참고 자료로만 사용될 것이다. 이 계획이 가장 중요하게 반영되는 것은 예를 들면 다음과 같다.
*.제품 소비량이 증가하여 새로운 생산설비를 도입하게 되었을 때, 어느 공장에 도입할 것인가?.

2.3.2 생산량 할당 Process Engine
- 각 공장, 각 제품별로 전용 생산라인의 정보를 이용하여, 근거리 수송 Process Engine의 Optimal solution을 결정한다. 단, 전체 공장의 전용 생산라인의 생산량의 합과 DC/SB의 발주량의 합이 대부분 서로 다르므로, 각각 Dummy source(공장) 또는 Dummy Destination(DC/SB)를 사용한다.
ㄱ.Dummy source를 사용하는 경우에는 DC/SB의 발주량이 더 많게 되므로 혼합 생산라인이 Dummy Source를 생산하는 것이다.
ㄴ.Dummy Destination을 사용하는 경우에는 공장의 최소 생산 Capa.를 생산하는 것이며, 시스템은 두가지 정보 모두를 제공한다.

2.4 시스템 구현

2.4.1 DataBase
본 시스템은 [Oracle 7.3 Workgroup Server For Windows NT]를 사용할 계획이다.
Application Server와 Database간의 Connection은 Proc를 사용하여 작성되고 있다.

2.4.2 Development Environment & Tools
Windows NT 4.0환경에서, 다중 Process Engine은 DLL(Dynamic Linked Library)로 작성되며, Application Server의 GUI는 Visual C++ / Visual Basic을 사용하여 개발하고 있다.

2.4.3 분산 처리
다음과 같은 사용자 요구로 인하여 분산 객체 또는 미들웨어의 도입을 고려하고 있다.
ㄱ. 각 Process Engine은 계산량이 많기 때문에 전용 Application에서 운영되기를 바란다.
ㄴ. 현재 각 업무를 담당하는 담당자는 LAN환경에서 Application을 필요한 시기에 실행하고 그 결과만 Database를 통해서 받고 자신의 PC (Personal Computer)에서 GUI환경으로 볼 수 있기를 바란다.
ㄷ. Application Server의 각 Process Engine간의 시간적인 동기화를 필요로한다.

따라서, 본 논문에서는 사용자 요구사항을 만족시키기 위해서 다음 3가지를 고려하고, 이를 적용하기 위한 연구를 진행중이다.
대안 1. 미들웨어(Tuxio) & PowerBuilder (- Client side)

대안 2. DCOM (Distributed Common Object Model)
- DCOM은 Microsoft의 독자적인 분산 객체 모델이다.

대안 3. CORBA (Common Object Request Broker Architecture)
- OMG(약 700여개 이상의 회사가 참여한 Consortium형태의 그룹) 의 표준 분산 객체 모델이다.
- 다양한 언어와(C/C++,Ada,Cobol,Java,...) IDL간의 변환을 지원한다.
- 풍부한 서비스를 지원한다.

*. Tuxio ,DCOM,CORBA간의 자세한 비교는 생략한다.

3.결론

본 연구에서는 DC/SB의 예상 요구량을 적정 시점에 가장 근거리 공장에서 출하시키기 위해서, 공장의 생산 의뢰 계획및 재고 자동 보충 계획과 이를 수행할 수 있는 Heuristic Algorithm 개발을 하였다. 본 시스템은 현재 개발되고 있는 상황이다. 따라서, 본 시스템의 평가는 최소한 Application Server의 개발이 완료되어야 내려질 수 있을 것이다.
하지만, 본 연구는 현장의 업무를 담당하는 직원들과 장기간 동안의 계속적인 회의를 통해서 수정 보완되었기 때문에, 이론적이면서도 현장의 많은 factor를 반영하였다고 생각된다.

참고 문헌

1. F.S. Hillier & G.j. Lieberman, "Introduction to Operations Research", McGraw Hill pp208-239.
2. Roger D.Eck, "Operations Research for Business" pp245-254
3. Katta G.Murty "Network Programming" prentice Hall, pp192-206
4. Hamdy A. Taha "Operations Research" macmillan, pp 208-225
5. 황흥석,류정철, "An Integrated Model for Inventory and Transportation Problems Based on Double- Compound Demand Distribution", '97춘계 학술대회 논문집,대한 산업공학회,pp 76-79
6. Charles Petzold, "Programming Windows 95",Microsoft press,1996
7. Jeffrey Prosisse, "Programming Windows 95 with MFC",Microsoft press,1996
8. Jeffrey Richter, "Advanced Windows NT",Microsoft press,1996
9. Peter Hipson,Roger Jennings "Database Developer's Guide with Visual C++ 4",Sams Publishing, 1996
10. OMG, CORBA services. New York, John & Sons, 1995
11. Amjad Umar, Distributed Computing and Client-Server Systems,Prentice Hall, 1993