목 차

Project 개요

Project 대상 기업 및 업체 선정 배경

Project 대상 기업

- 본 프로젝트는 원사를 가공하여 원단을 제작하는 섬유 공장을 대상으로 함
- 인원 : 정직원 10명(영업:2, 경리:1, 기사:7명) 임시 고용직 12명
- 매출 : 연 30억원
- 거래처 수 : 총 10여 곳
- 창고 현황 : 약 2000평
- 원자재 : 자체 구매 및 거래처에서 원사제공
- 재고 : 완제품 재고, 원자재
- 송장관리 : Excel
- 기계 : 7대 (빔제작), 16대 (원단제작)
- 공장 가동시간 : 24시간 (3교대)
- 주 생산품 : 벨벳

업체 선정 배경

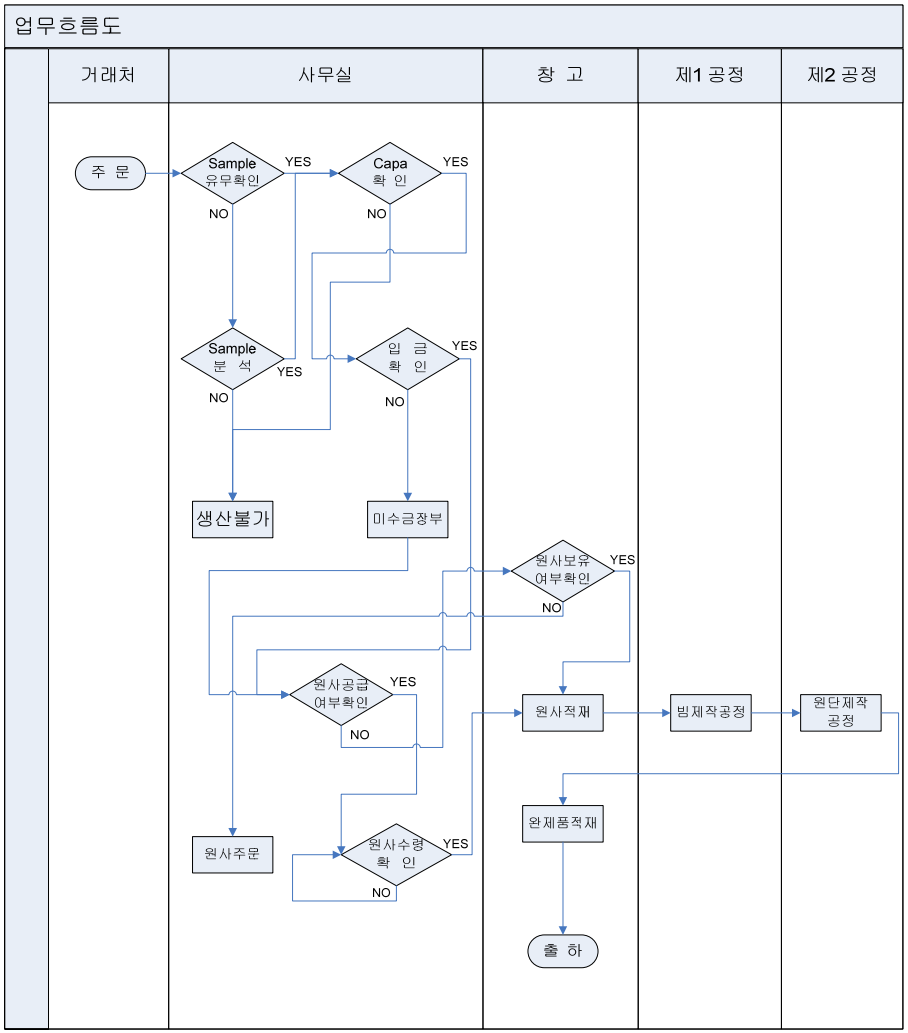
- 국내 현황
  - ▸경제적 불황
  - ▸섬유제품에 대한 수요 둔화
  - ▸수출품목 대외 경쟁력 약화
  - ▸인건비, 생산비의 증가

- 기업의 현 현황
  - ▸공장 규모 축소
  - ▸ERP시스템 소멸
  - ▸공장중심의 근시안적인 생산

한국외국어대학교
산업정보시스템공학부

Project 현황 분석

현 업무흐름도 및 현황 분석



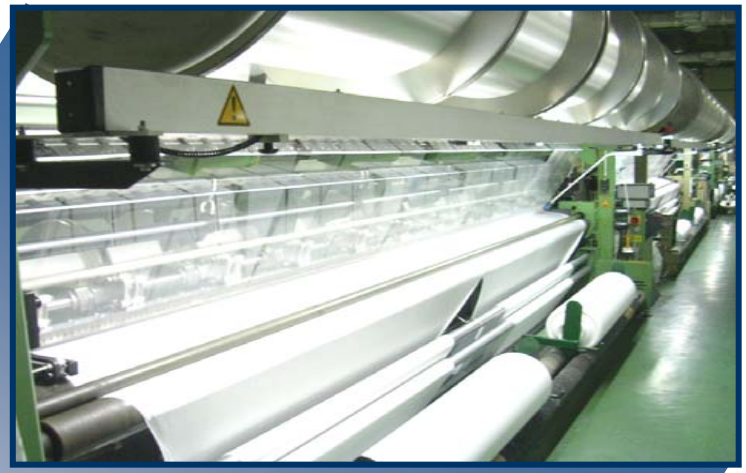
- 주문 : 영업사원이 거래처와 전화 및 대면을 통해 주문 접수

- 사무실 : 영업사원으로부터 샘플 받아 생산 가능여부 판단 및 Capacity 확인, 총 생산량 예측하여 스케줄링
  거래처로부터 입금 확인 선수금, 미수금을 통한 장부작성
  원단 제작에 필요한 원사를 창고재고로부터 확인하여 부족량을 주문

- 창고 : 원사와 원단(완제품)을 보관하고 실사 확인 완성된 원단을 출하 (Forklift 사용, 주문을 발주한 업체에서 직접 수령)

Project 현황 분석

현 업무흐름도 및 현황 분석



생산공정

- 제 1공정
  - 원사를 선반에 수작업으로 설치 (1200개)
    셋업타임 : 약 3~4시간
  - 원사를 가지고 원단을 만들기 위한 빔 제작공정
    공정 시간 : 큰 빔(3시간), 작은 빔(1시간30분)
    (작은빔은 특정 원사에 사용)
  - 원사 보유량이 1200개 이하일 때, 작은 빔을 사용하고 그 이상일 경우는 큰 빔을 사용
  - 큰 빔은 5EA, 작은 빔은 10EA의 단위로 일괄 생산 (길이/무게 통일)
  - 완성된 빔을 생산 완료 시기에 맞추어 제 2공정으로 운반
  - 이동수단 : 이동식 레일카(1대)
  - 소요시간 : 약 15분, 최대운반개수 : 큰빔 4EA, 작은빔 6EA
- 제 2공정
  - 원단의 특성에 맞게 빔을 조합하여 거치시킴
  - 셋업타임 : 2시간 (기중기 활용)
  - 공정기간 : 4일~2주
  - 원단에 들어가는 실의 종류에 따라 최대 3가지 종류의 다른 실을 걸 수 있음
  - 원사의 혼합율(원사의 소모량)에 따라 각 Bar에 걸려있는 빔의 회전속도에 차이
  - 빔에 감겨있는 원사 전부 소모되면 기계는 멈추고 빔 교체작업을 실시
  - 완성된 원단을 포장(수작업) 후 창고로 이동

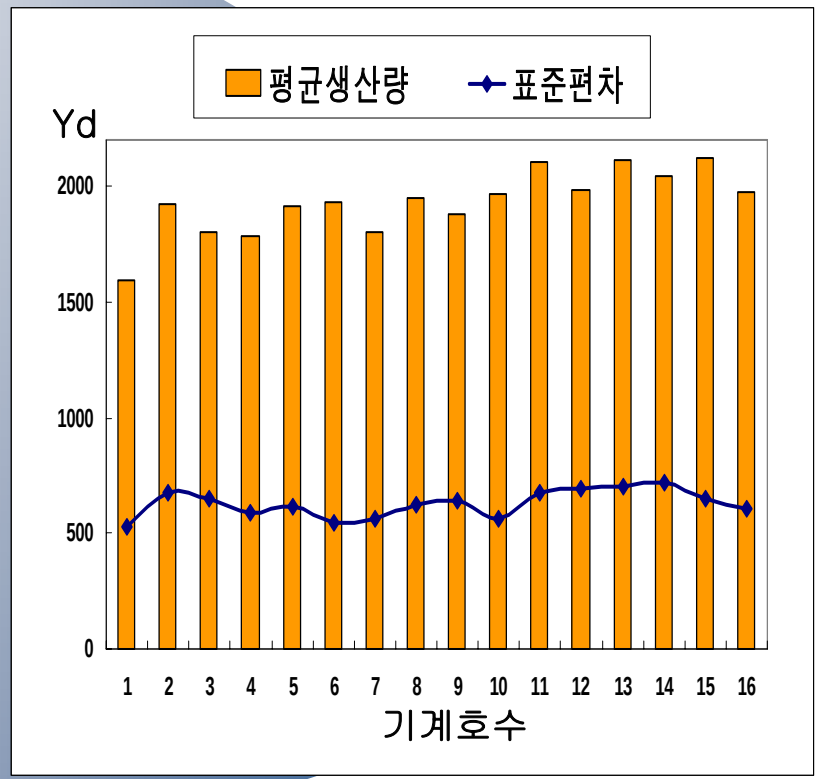
Project 필요성

문제점

- 한 달 총생산량을 대략적으로 계산하여 주문 접수로 인한 초과 주문 및 부족 주문 발생
  - 현재 총생산량 구하는 식= 2000yd × 기계 대수
  - 만드는 원단에 따라 기계속도인 rpm과 기계 밀도가 다르나 현재는 대당 2000yd로 일괄적으로 계산하여 부정확한 총생산량을 계산

대당 생산량 = 기계 rpm ÷ 기계 밀도 × 60(분) × 24(시간) ÷ 91.44(㎝) × (가동률) × 2(폭)

- 부정확한 총생산량 계산으로 인해 초과주문을 받거나 총생산량에 미치지 못하는 주문량 수주
- 여러 가지 종류의 원단을 주문 받았을 때 기계에 대한 할당을 직관으로 판단하여 불필요한 할당 발생
- 누적된 장부의 정보파악의 어려움으로 인해 과거 data 검색 어려움
- 제2공정과 제1공정 간의 스케줄링 오차 발생과 원사의 부적절한 공급으로 기계별 생산량 편차 발생



기계당 일일평균 생산량 및 표준편차(70일)

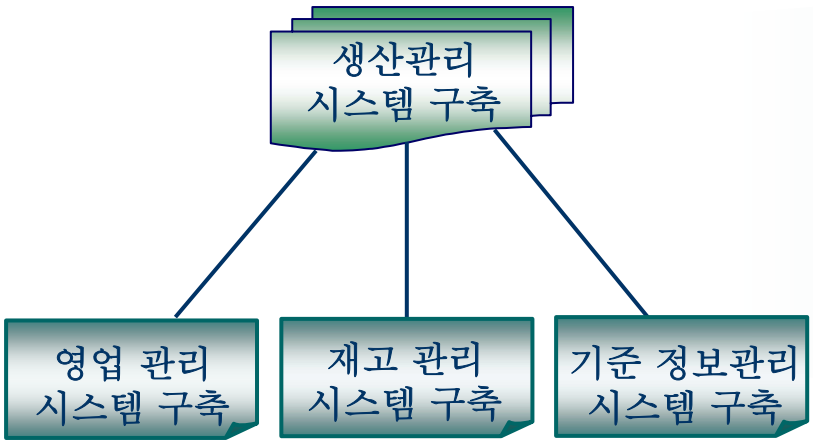
Project 목적 및 기대효과

목적 및 기대효과

목 적

- 생산 스케줄링 개선
  - 총생산량 계산방법 개선
  - 납기일, 가동률 고려
  - 빔공정과 원단제작 공정의 동기화
  - 원사 보유시기 파악
- 생산 스케줄링에 필요한 기본적인 재고관리, 영업 관리 시스템 구축으로 인한 손쉬운 업무 파악
- 수작업으로 이루어지는 문서 작업의 전산화

기대효과



- RPM, 기계밀도 고려한 총생산량 계산으로 인해 정확도 향상
- 납기일, Overtime의 최적화를 고려한 스케줄링
- 기준정보 체계확립
- 시스템구축으로 인한 정보의 DB화

Project 범위 및 내용 / 산출물

범위 및 내용 / 산출물

범위 및 내용

영업 관리 시스템

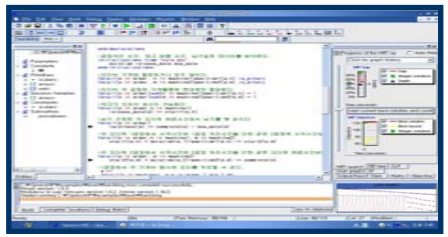
생산관리 시스템 구축
-스케줄링을 통한 두 공정 원활한 연계
-납기지연 최소화
-Over Time 최소화
-특이사항 발생시 고려할 수 있는 스케줄링 구축
-생산 지시서 출력

재고 관리 시스템

기준 정보 관리

산출물

1 Optimization S/W를 이용한 스케줄링 엔진 구현



2 Visual Basic을 이용한 사용자 인터페이스

3 스케줄링 엔진과 인터페이스 프로그램 연동

Project 산출물

1 Optimization S/W를 이용한 스케줄링 엔진 구현

상황 분석

- 원사가 2개의 공정을 거쳐 원단을 생산하는 과정
- 1공정은 원사를 빔으로 만드는 공정이고, 2공정은 빔을 원단으로 만드는 공정이다.
- 각 공정에는 기계가 각각 n, r, m이 있다.
  - n과 r 기계는 같은 공정에서 작은 빔과 큰 빔으로 분류된다.
- order는 각각의 공정 시간이 있다.
- 각 order는 원사가 도착해야 생산을 시작할 수 있다.
- 목적 1.
  - 기계에 order들을 잘 분배해서 최소시간에 주문 모두를 생산할 수 있도록 스케줄링

모델링 Formulation

DATA

due_date_{o} = 각 order의 납기일

$duration_{o,p}$ = 각 order의 공정별 처리시간

$stock\_date_{o}$ = 각 order의 원사 확보 시점

M = 매우 큰 수

$o \in order \quad p \in process$

Variable

$position1\_1_{o,n}$ = 1_1공정에서 어떤 order가 어떤 기계에 할당여부(이진변수)

$position1\_2_{o,r}$ = 1_2공정에서 어떤 order가 어떤 기계에 할당여부(이진변수)

$position2_{o,m}$ = 2공정에서 어떤 order가 어떤 기계에 할당여부(이진변수)

$start1\_1_{o,n}$ = order가 1_1공정 기계에서 작업을 시작하는 시간

$start1\_2_{o,r}$ = order가 1_2공정 기계에서 작업을 시작하는 시간

Project 산출물

모델링 Formulation (계속)

$start2_{o,m}$ = order가 2공정에서 작업을 시작하는 시간

$complete_o$ = 각 order의 완료 시간

$y1\_1_d$ = 1_1공정 작업 순서를 정하기 위한 이진변수

$y1\_2_f$ =1_2공정 작업 순서를 정하기 위한 이진변수

$y2_c$ = 2 공정 작업 순서를 정하기 위한 이진변수

$tardiness_o$ =Order가 납기일을 넘긴 시간

$o \in order \quad n \in machine1\_1 \quad r \in machine1\_2$

$m \in machine2$

Object Function

minimize $\sum_{o \in order} complete_o$: 완료시간의 최소화

minimize $\sum_{o \in order} tardiness_o$: 납기지연의 최소화

Subject To

$\forall o \in order : position1\_1_{o,n} \in \{0,1\}$

$\forall n \in machine1\_1 : position1\_2_{o,r} \in \{0,1\}$

$\forall r \in machine1\_2 :$

$\forall m \in machine2 : position2_{o,m} \in \{0,1\}$

$$\forall o \in order : \sum_{n \in machine1\_1} position1\_1_{o,n} = 1$$

$$\forall o \in order : \sum_{r \in machine1\_2} position1\_2_{o,r} = 1$$

$$\forall o \in order : \sum_{m \in machine2} position2_{o,m} = 1$$

모든 order는 각 공정의 기계들 중에 한대에만 할당됨

$$\forall o \in order : stock\_date_o \le start1\_1_{o,n}$$
$$\forall n \in machine1\_1 :$$
$$\forall r \in machine1\_2 : stock\_date_o \le start1\_2_{o,r}$$

1공정에서의 각 order는 시작시간에 1공정 처리시간을 더한 값은 2공정의 시작시간보다 작거나 같음

$$\forall o \in order :$$
$$\forall m \in machine2 :$$
$$start2_{o,m} + duration_{o,2} * position2_{o,m} \le complete_o$$

2공정에서의 각 order는 시작시간에 2공정 처리시간을 더한 값은 order의 완료시간보다 작거나 같음

모델링 Formulation (계속)

$$\forall o \in order \quad :$$
$$\text{tardiness}_{o} \geq complete_{o} - due\_date_{o}$$

각 order의 완성시간이 Overtime보다 작거나 같아야 함

$$\forall o, q \in order (o \neq q) :$$
$$\forall n \in machine\ 1\_1 \ :$$
$$\text{start1\_1}_{o,n} + duration_{o,1} * position\ 1\_1_{o,n} \leq start\ 1\_1_{q,n} + M * (1 - y1\_1_{d})$$
$$\text{start1\_1}_{q,n} + duration_{q,1} * position\ 1\_1_{q,n} \leq start\ 1\_1_{o,n} + M * y1\_1_{d}$$
$$\forall o, q \in order (o \neq q) :$$
$$\forall r \in machine\ 1\_2 \ :$$
$$\text{start1\_2}_{o,r} + duration_{o,1} * position\ 1\_2_{o,r} \leq start\ 1\_2_{q,r} + M * (1 - y1\_2_{f})$$
$$\text{start1\_2}_{q,r} + duration_{q,1} * position\ 1\_2_{q,r} \leq start\ 1\_2_{o,r} + M * y1\_2_{f}$$
$$\forall o, q \in order (o \neq q) :$$
$$\forall m \in machine\ 2 \ :$$
$$\text{start2}_{o,m} + duration_{o,2} * position\ 2_{o,m} \leq start\ 2_{q,m} + M * (1 - y2_{c})$$
$$\text{start2}_{q,m} + duration_{q,2} * position\ 2_{q,m} \leq start\ 2_{o,m} + M * y2_{c}$$

각 공정에서 각 기계는 동시에 오더를 처리 할 수 없음

Project 산출물

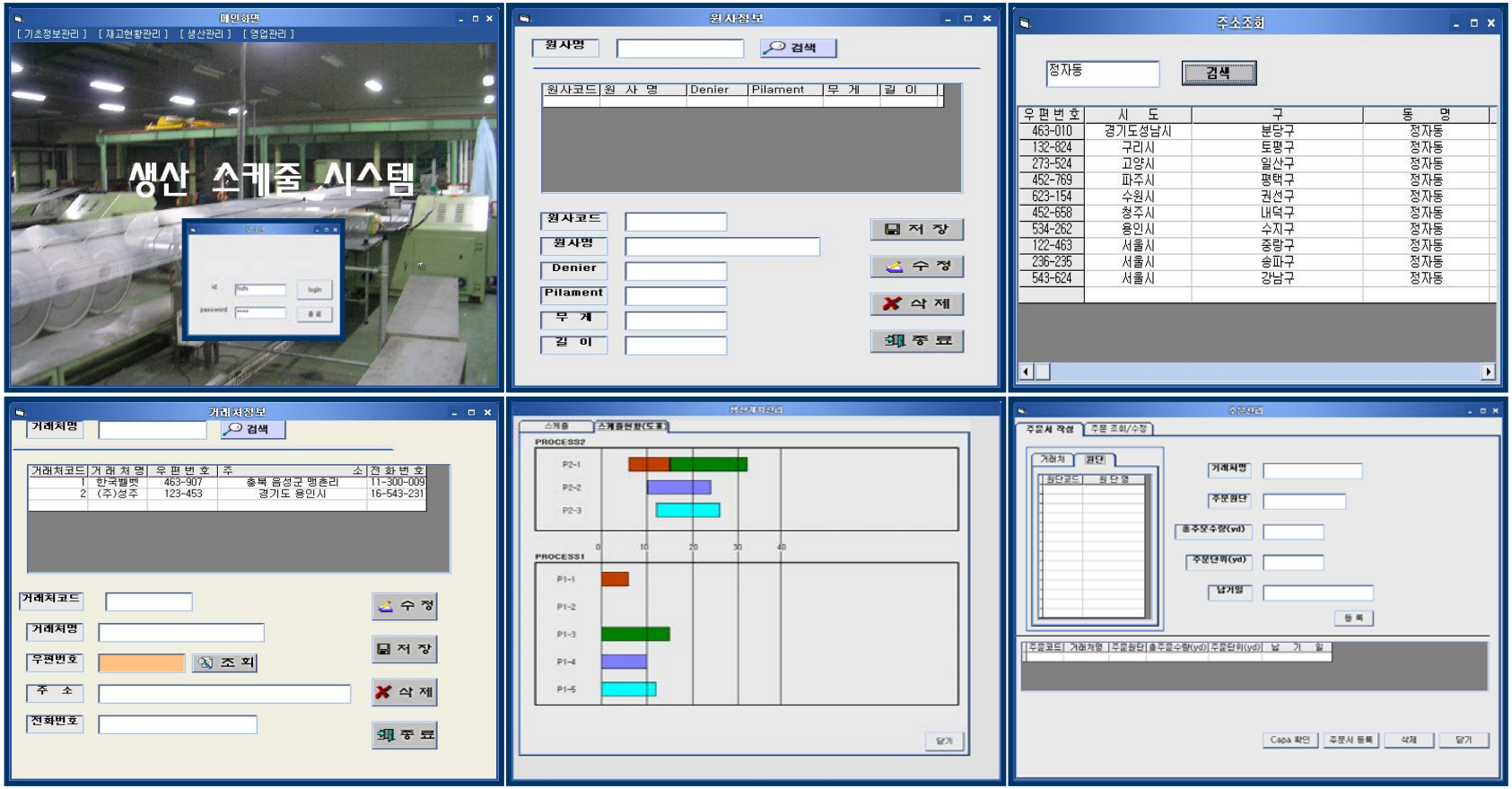
1 Optimization S/W를 이용한 스케줄링 엔진 구현 (계속)

스케줄링 엔진 구현

```
model "BoA"
uses "mmxprs"

forward procedure print_sol(obj:integer)

declarations
    number_of_order : integer
end-declarations

!총 오더의 갯수를 받아온다.
initializations from 'kuro.dat'
    number_of_order
end-initializations

!데이터 값 및 변수들을 생성한다.
declarations

    process = 1..2
    machine1 = 1..2
    machine2 = 1..4
    machine3 = 1..8
    order = 1..number_of_order
    M = 5000

    due_date : array(order) of integer
    duration : array(order,process) of integer
    release_date : array(order) of integer
    sequence :array(order) of integer

    tardiness : array(order) of mpvar
    position1 :array(order,machine1) of mpvar
    position2 :array(order,machine2) of mpvar
    position3 :array(order,machine3) of mpvar
    start3 : array(order,machine3) of mpvar
    complete : array(order) of mpvar
    p1ord : array(range) of mpvar
    p3ord : array(range) of mpvar
    y1 : array(range) of mpvar
    y2 : array(range) of mpvar
    y3 : array(range) of mpvar

end-declarations

!공정처리 시간, 재고 보충 시간, 납기일의 데이터를 받아온다.
initializations from 'kuro.dat'
    duration release_date due_date
end-initializations

forall(o in order)do
if sequence(o) = 1 then
a:=1
  create(p1ord(a))
  p1ord(a) = o
a+=1
  else
e :=1
  create(p3ord(e))
  p3ord(e) = o
a+=1
  end-if
  end-do

!오더는 기계에 할당되거나 되지 않는다.
forall(o in order, n in machine1)position1(o,n) is_binary
forall(o in order, m in machine2)position2(o,m) is_binary
forall(o in order, l in machine3)position3(o,l) is_binary


!오더는 각 공정의 기계들중에 한대에만 할당된다.
forall(o in order)sum(n in machine1)position1(o,n)+sum(l in machine3)position3(o,l) = 1
forall(o in order)sum(m in machine2)position2(o,m) = 1

!재고가 있어야 생산이 가능하다.
forall(o in order,n in machine1,l in machine3)do
    release_date(o) <= start1(o,n)
    release_date(o) <= start3(o,l)
end-do

!납기 지연은 각 오더의 완료시간에서 납기를 뺀 값이다.
forall(o in order)
    tardiness(o) >= complete(o) - due_date(o)

!각 오더의 1공정에서 시작시간에 1공정 처리시간을 더한 값은 2공정의 시작시간보다 작거나 같다.
forall(o in order,n in machine1, m in machine2, l in machine3)do
    start1(o,n) + duration(o,1)*position1(o,n) <= start2(o,m)
    start3(o,l) + duration(o,1)*position3(o,l) <= start2(o,m)
end-do

!각 오더의 2공정에서 시작시간에 2공정 처리시간을 더한 값은 오더의 완료시간보다 작거나 같다.
forall(o in order, m in machine2)
    start2(o,m) + duration(o,2)*position2(o,m) <= complete(o)

!1공정에서 각 기계는 동시에 오더를 처리할 수 없다.


 d:=1
 forall(n in machine1, h,i in p1ord | h<>i) do
  create(y1(d))
    y1(d) is_binary
        start1(h,n)+duration(h,1)*position1(h,n) <= start1(i,n)+M*(1-y1(d))
        start1(i,n)+duration(i,1)*position1(i,n) <= start1(h,n)+M*y1(d)
    d+=1
 end-do

 b:=1
 forall(l in machine3, j,k in p3ord | j<>k) do
        start3(j,l)+duration(j,1)*position3(j,l) <= start3(k,l)+M*(1-y3(b))
        start3(k,l)+duration(k,1)*position3(k,l) <= start3(j,l)+M*y3(b)
    b+=1
  end-do

!2공정에서 각 기계는 동시에 오더를 처리할 수 없다.
 c:=1
 forall(m in machine2 ,o,q in order| o<>q ) do
  create(y2(c))
    y2(c) is_binary
        start2(o,m)+duration(o,2)*position2(o,m) <= start2(q,m)+M*(1-y2(c))
        start2(q,m)+duration(q,2)*position2(q,m) <= start2(o,m)+M*y2(c)
    c+=1
  end-do


!모든 노드에서 오더들의 완료 시간의 합
finish := sum(o in order) complete(o)
finish <= M
minimize(finish)
print_sol(1)


!납기 지연 최소화
due := sum(o in order) tardiness(o)
minimize(due)
 print_sol(2)

!출력 값
 procedure print_sol(obj:integer)
    writeln("Objective ", obj, ": ", getobjval,
            if(obj>1, " completion time: " + getsol(finish), ""),
            if(obj<>2, " tardiness: " + getsol(due), ""))
 end-procedure
```

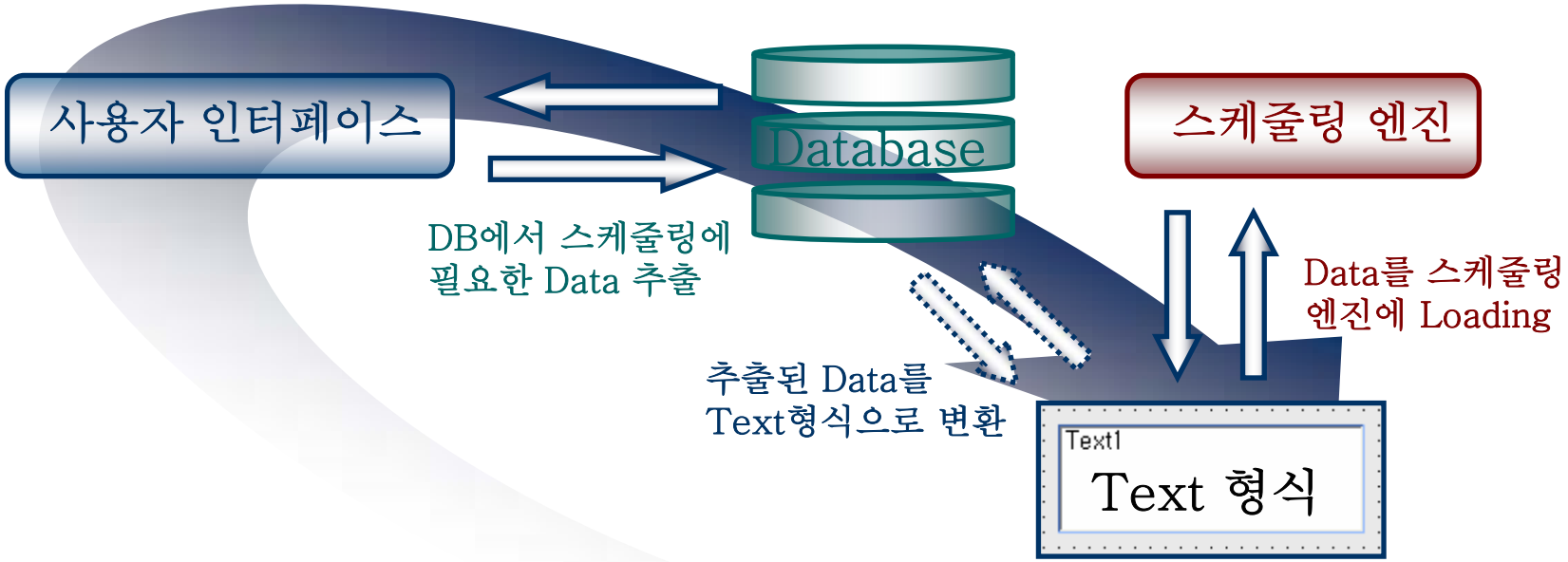
Project 범위 및 내용 / 산출물

2 Visual Basic을 이용한 사용자 인터페이스



Project 범위 및 내용 / 산출물

3 스케줄링 엔진과 사용자 인터페이스 연동



사용자가 스케줄링 엔진을 컨트롤하기 쉽게 하기 위하여 Database의 정보를 Text화한 후

스케줄링 엔진에서 읽어 결과값을 다시 Text형식으로 출력

Text형식의 결과값을 다시 Database로 저장한 후 사용자 인터페이스로 출력