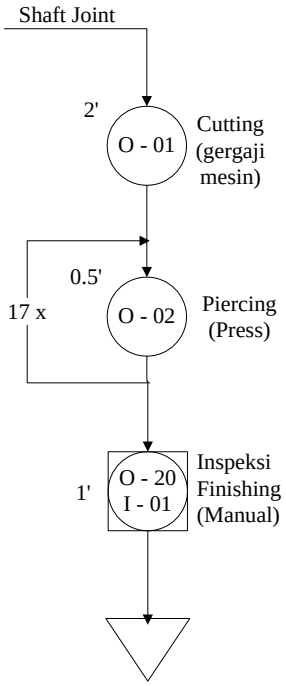
LAMPIRAN 1

PETA PROSES OPERASI

PETA PROSES OPERASI

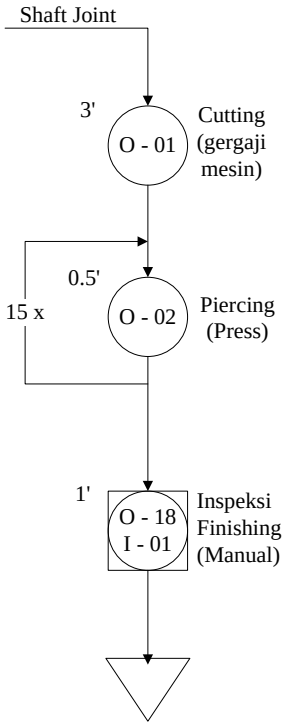
Nama Obyek : Shaft Joint
No. Gambar : 01
Digambar Oleh : Adam Setia Salim
Tanggal digambar : 14 –0 9 - 2013



Ringkasan		
Kegiatan	Jumlah	Waktu (detik)
Proses	20	12
Inspeksi	1	0'
Jumlah	21	12

PETA PROSES OPERASI

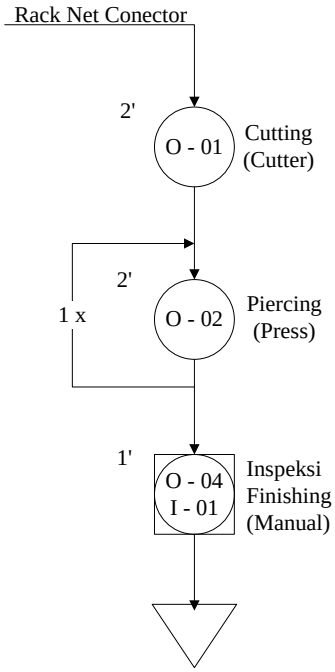
Nama Obyek : Shaft Support
No. Gambar : 02
Digambar Oleh : Adam Setia Salim
Tanggal digambar : 14 –0 9 - 2013



Ringkasan		
Kegiatan	Jumlah	Waktu (detik)
Proses	18	13
Inspeksi	1	0"
Jumlah	21	13

PETA PROSES OPERASI

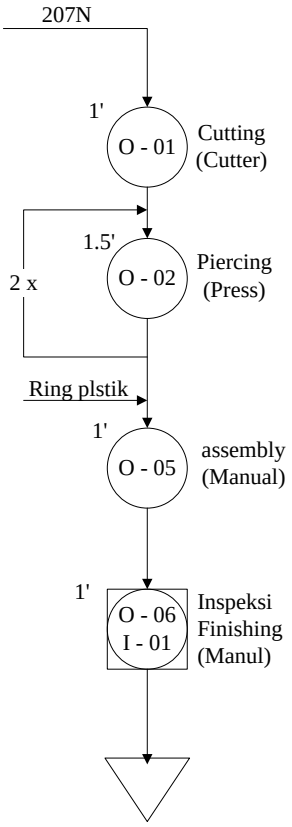
Nama Obyek : Rack Net Conector
No. Gambar : 03
Digambar Oleh : Adam Setia Salim
Tanggal digambar : 14 – 09 - 2013



Ringkasan		
Kegiatan	Jumlah	Waktu
Proses	4	6'
Inspeksi	1	0"
Jumlah	5	6'

PETA PROSES OPERASI

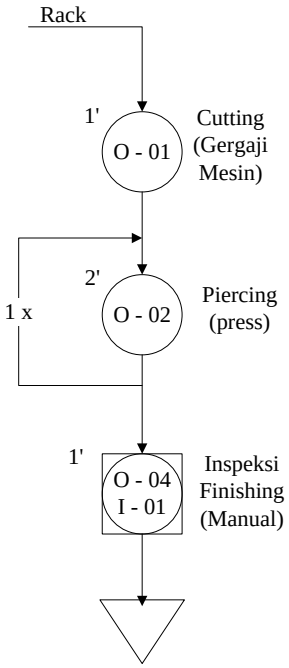
Nama Obyek : 207N
No. Gambar : 04
Digambar Oleh : Adam Setia Salim
Tanggal digambar : 14 –0 9 - 2013



Ringkasan		
Kegiatan	Jumlah	Waktu
Proses	6	6'
Inspeksi	1	0"
Jumlah	8	6'

PETA PROSES OPERASI

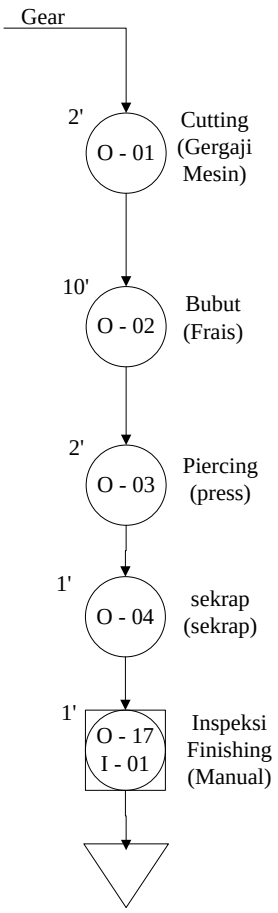
Nama Obyek : Rack
No. Gambar : 04
Digambar Oleh : Adam Setia Salim
Tanggal digambar : 14 –0 9 - 2013



Ringkasan		
Kegiatan	Jumlah	Waktu
Proses	4	6'
Inspeksi	1	0"
Jumlah	5	6'

PETA PROSES OPERASI

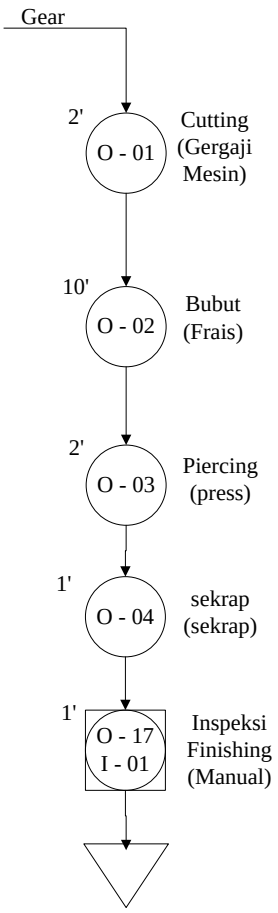
Nama Obyek : Gear
No. Gambar : 07
Digambar Oleh : Adam Setia Salim
Tanggal digambar : 14 –0 9 - 2013



Ringkasan		
Kegiatan	Jumlah	Waktu
Proses	5	16
Inspeksi	1	0"
Jumlah	6	16

PETA PROSES OPERASI

Nama Obyek : Gear
No. Gambar : 07
Digambar Oleh : Adam Setia Salim
Tanggal digambar : 14 –0 9 - 2013



Ringkasan		
Kegiatan	Jumlah	Waktu
Proses	5	16
Inspeksi	1	0"
Jumlah	6	16

LAMPIRAN 2

PROGRAM GENERATE AND TEST

```
Program GenerateAndTest (input,output);
{$N+}
{$M 62000,0}
uses Wincrt;
{const
  jk=4;}
label
  DFSlagi, up;
var
  {maksimal 20 kota}
  jk,jm: integer;
  D,Due: array [1..20] of integer;
  level, selesai, u, pro: byte;
  S, F: array [0..20+1,1..20] of byte;
  R,Tipe: array [0..20] of byte;
  PathCostLkp, BestPathCost:extended;
  Fi, Fr: text;
  rt : array [1..3] of extended;
  rt1,rt2,rt3:extended;

Procedure Input;forward;
Procedure Inisialisasi;forward;
Procedure DFS;forward;
Procedure Backtracking;forward;
Procedure PCost;forward;

Procedure Input;
var
  i,j:integer;
Begin
   {Seting parameter awal}
   {jumlah kota dan jaraknya terdapat
pada file 'inputG&T.txt'}
   Assign(Fr,'inputAd3.txt');
   Reset(Fr);
   Read(Fr,jk);
   Write(jk,' ');
   Read(Fr,jm);
   Writeln(jm);
   Read(fr,rt1);
   read(fr,rt2);
   read(fr,rt3);
   writeln(rt1,' ',rt2,' ',rt3);
   readln;

   i:=1;

   While not Eof(Fr) do
   begin

    Read(Fr,D[i]);
    Read(Fr,Due[i]);
    i:=i+1;

   end;
   For i:=1 to jk do
   Begin
      Writeln('D[',i,']=',D[i],';
Due[',i,']=',Due[i]);
   end;
   readln;

end;

Procedure Inisialisasi;
var
  i:integer;

Begin
   Assign(Fi,'HasilAd.txt');
   Rewrite(Fi);
   Level:=0;
   S[0,1]:=0;

   For i:=1 to jk do
    F[0,i]:=i;

   BestPathCost:=10000000;

end;

Procedure DFS;
label
  akhir, sela;
var
```

```
  i,j,k,l,m,n,p,t:integer;
Begin

    if selesai=1 then
     goto akhir;

    level:=level+1;
    {writeln(level);}
    {writeln('level ',level);}

    {Tentukan State-state pada level
sekarang.}
    For i:=1 to level do
    begin
     if i<level then
     begin
      S[level,i]:=S[level-1,i];
      {writeln('S[',level,',',i,']=',S[level,i]);
      readln;}
     end
     else
     begin
      j:=1;
      while j <= jk-level+1 do
      begin
       if F[level-1,j]<>0 then
       begin
        S[level,i]:=F[level-1,j];
        F[level-1,j]:=0;
        j:=jk+1;

{writeln('S[',level,',',i,']=',S[level,i]);
         readln;}
       end;
       j:=j+1;
      end;
     end;
    end;
    {Write(level,jk);readln;}
    {Apakah GOAL tercapai?}
    if level = jk then
    begin
     {hitung pathcost}
     PCost;

     Write(Fi,'Rute ');
     For t:=1 to jk-1 do
      Write(Fi,S[level,t],'-');
     Write(Fi,S[level,jk],'   ');
     Writeln(Fi,'PathCost=
',PathCostLkp);
     {Readln;}

     {tetapkan BestPathCost}
     If PathCostLkp < BestPathCost then
     begin
      BestPathCost:=PathCostLkp;
      For l:=1 to jk do
       R[l]:=S[level,l];
     end;



     {lakukan backtracking}

     Backtracking;
     {pro:=2;
     goto sela;}
    end
    else
    begin
     {tentukan Fringe pada level
sekarang}
     {Write('Fringe: ');}
      k:=1;
      For m:= 1 to jk do
      begin
       p:=0;
       For n:= 1 to level do
        if m<>S[level,n] then
         begin
          p:=p+1;
          if p=level then
          begin
           F[level,k]:=m;
           {Write(F[level,k],' ');}
           k:=k+1;
          end;
         end;
      end;
     {lanjutkan untuk level berikutnya}
     {DFS;}
     pro:=1;
     goto sela;
    end;
```

```
    akhir:
    sela:
end;

Procedure Backtracking;
{label
    sela2;}
var
    i:byte;
Begin
    {writeln('backtracking');readln;}
    level:=level-1;
    if (F[0,jk]=0) and (level=0) then
    begin
      selesai:=1;
      DFS;
      {pro:=1;
      goto sela2;}
    end
    else
    begin
     For i:=level+1 to jk do
      if F[level,i-level]<>0 then
      begin
        DFS
        {pro2:=1;
        goto sela2;}
      end
      else
        if i=jk then
        begin
          backtracking;
          {pro:=2;
          goto sela2;}
        end;
    end;
    {sela2:}
End;

Procedure PCost;
var
  i,j,k,l:integer;

  makespan:extended;

  mmin:byte;
  rtmin:extended;

Begin
    {write('masuk pc');readln;}
    rt[1]:= rt1;
    rt[2]:= rt2;
    rt[3]:= rt3;
    makespan:=0;
    For i:=1 to jk do
    begin
      k:=S[level,i];
      {write('K=',K);READLN;}
      {pilih mesin}
      rtmin:=10000000;

      For j:=1 to jm do
        if rt[j]<rtmin then
        begin
          rtmin:=rt[j];
          mmin:=j;
        end;
      rt[mmin]:=rt[mmin]+D[k];
      {write(mmin,' ',rtmin:3:0,'
',rt[mmin]:3:0);readln;}
    end;
    For l:=1 to jm do
      if rt[l]>makespan then
        makespan:=rt[l];
        PathCostLkp:=makespan;
    {write('makespan=
',makespan);readln;}
end;

{Main Module}
BEGIN
  Input;
  Inisialisasi;
  pro:=0;
  {pro2:=1;}
  DFSlagi:
  DFS;
  {up:}
  If pro=1 then
  begin
    pro:=0;
    goto DFSlagi;
  end;
  {else
  begin
```

```
   backtracking;
   goto up;
  end;}

  {sela2:
  If pro2=1 then
   DFS
  else
   backtracking;}

  Write(Fi,'Rute terpilih adalah ');
  For u:=1 to jk-1 do
   Write(Fi,R[u],'-');
  Write(Fi,R[jk],'   ');
  Writeln(Fi,'BestPathCost=
',BestPathCost);
  Close(Fi);
END.
```