Arhitecturi Orientate pe Servicii (SOA) si Sisteme Multi-Agent (MAS) pentru Controlul Distribuit al Productiei Agile

Theodor Borangiu

Centrul de Cercetare si Instruire CIMR, cimr@cimr.pub.ro
Universitatea Politehnica din Bucuresti

Sumar

1. **Arhitectura de sistem, integrare ERP-proces si instrumente de simulare dinamica**
2. **Holarhia si proiectarea HMES**
3. **Planificarea cu Holoni Expertiza:**
   - **Planificarea bazata pe cunostinte**
   - **Planificarea pas cu pas cu simulare dinamica**
4. **Gestiunea schimbarilor (comenzi, avarii, realimentari) prin contracte de negociere incorporate (CNP) si urmarirea productiei dirijata de evenimente. Integrarea sistemului**
5. **Rezultate experimentale. Concluzii si dezvoltari viitoare**

1. Arhitectura de sistem, integrare ERP-proces si simulator dinamic

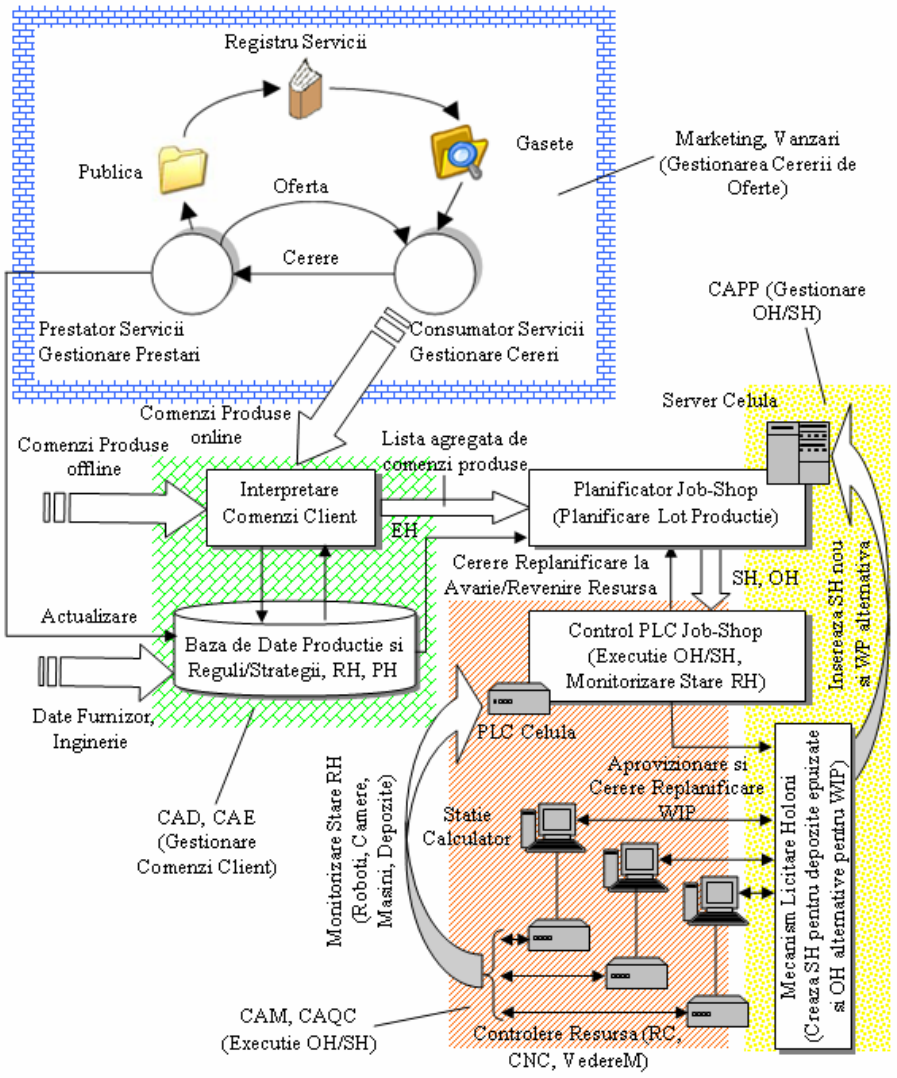
Trasaturi de baza ale controlului holonic al fabricatiei:

1. **Proces controlat**: fabricatie cu robot interconectati, cu prelucrari si alimentare incluse
2. **Structura de fabricatie**: tip job-shop, transport prin conveior in bucla inchisa
3. **Vedere artificiala (VA)**: viteza inalta, timp real pt conditionare materiale (GVR, AVI):
   - ✓ Califica, recunoaste, localizeaza vizual obiecte in scena (post de lucru, depozit)
   - ✓ Masuratori bazate pe caracteristici ale pieselor, extrase prin VA: *in-line* CAQC
   - ✓ Autorizeaza accesul la piese bazat pe test "amprente vizibile": evitare coliziuni
4. **Infrastructura fizica vizata**: roboti industriali, masini unelte CNC, VA, depozite de materiale, dispozitive inteligente de alimentare, magazine de scule, senzori, RD/WR cod
5. **Control**: distribuit, semi-heterarhic (comuta ierarhic/heterarhic), proiectat ca HMES
6. **Arhitectura de referinta**: PROSA (H. Van Brussel, P. Valkenaers, HMS Leuven, 1998)
7. **Holonii**:
   - ✓ Sunt agenti autonomi si cooperanti
   - ✓ Incapsuleaza o componenta informationala si o componenta fizica
   - ✓ Tipuri de holoni: Produs- (PH), Resursa- (RH), Comanda- (OH) si Expertiza (EH)
8. O **Arhitectura Orientata pe Servicii** (SOA) cu 4 zone: (i) Gestiune Cereri de Oferta; (ii) Gestiunea Comenzilor; (iii) Gestiunea OH, SH; (iv) Executia si Tracking OH, SH.
9. **Toleranta la defect**: Redundanta HW (SPOF), Multi-LAN, Replicare SW

1. Arhitectura de sistem, integrare ERP-proces si simulator dinamic

SOA a productiei tip job-shop cu control distribuit, holonic al RC, CNC, VA si ASRS

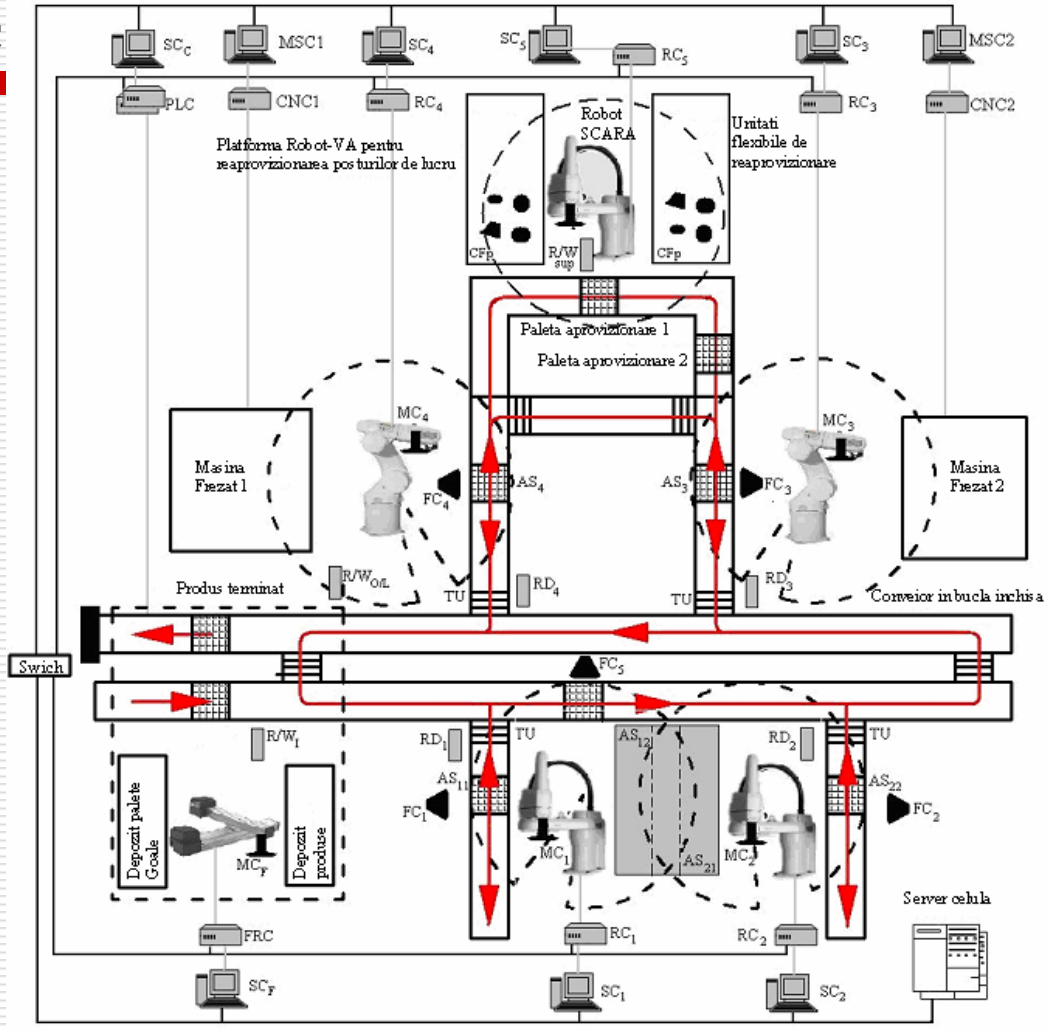
- Conveior in bucla inchisa
- Statii de lucru robot-vedere (RV) conectate in retea
- Camere duale RV:
  - fixa (*down looking*)
  - mobila (*arm-mounted*)
- Masini CNC 3/4 axe, ASRS centrale (2) / locale (robot), Magazine de scule
- Dispozitive RD/WR magnetice pentru urmarire *produs_pe_paleta*
- Holoni de alimentare (SH) creati on-line
- Gestiunea KB a Comenzilor Client
- Toleranta la defect si Inalta disponibilitate



1. Arhitectura de sistem. Integrare ERP-proces si simulator dinamic

Arhitectura de Control Distribuit

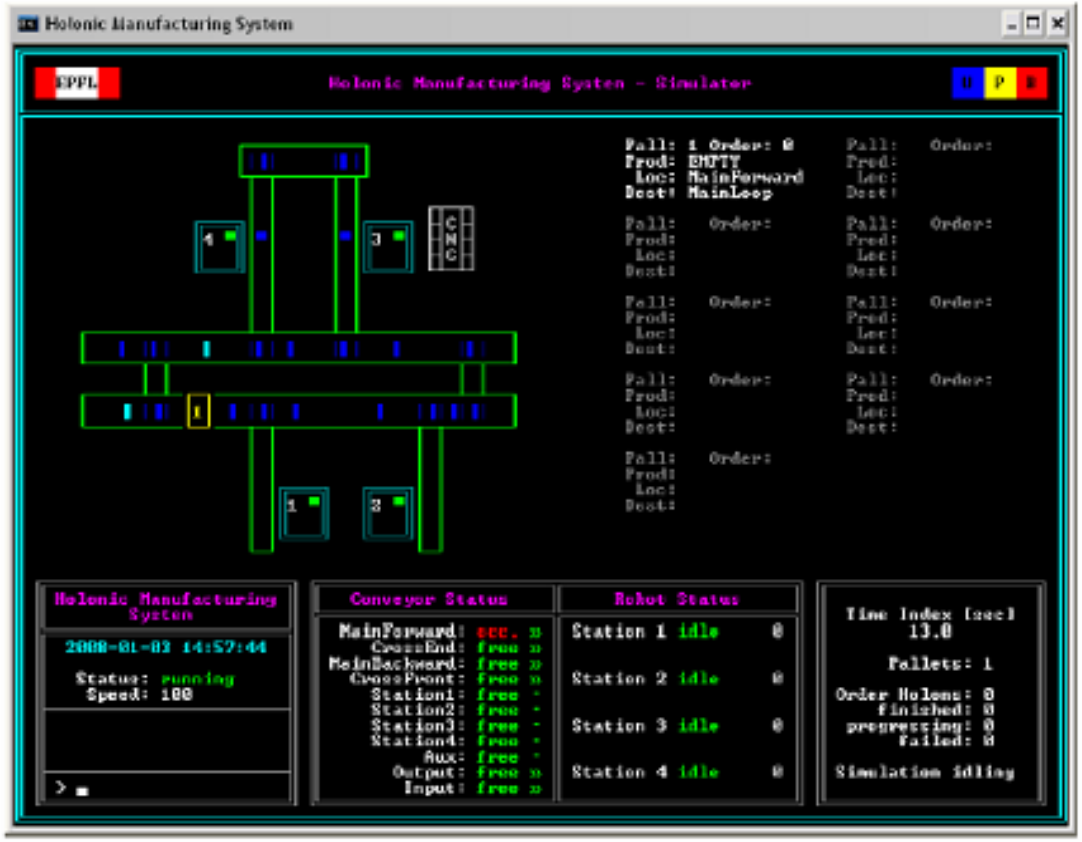
- Arhitectura Distribuita:
  Entitati informationale (holoni) cu corespondent fizic: Server Celula si Post, Calculator, Controller
- Comunicatie pe retele multiple:
  Ethernet, seriala, inel, I/O
- Server Celula (IBM xSeries) pentru:
  - Gestiunea comenzilor client
  - Planificarea loturilor in job-shop
- Statii de Calcul tip IBM PC pentru:
  - Furnizare date CNP
  - Replicare date si programe
  - Terminale Controller statie: editare pg., set up, tracking
- PLC pentru:
  - Executarea OH si SH
  - Urmarirea produselor



1. Arhitectura de sistem. Integrare ERP-proces. Simulare dinamic

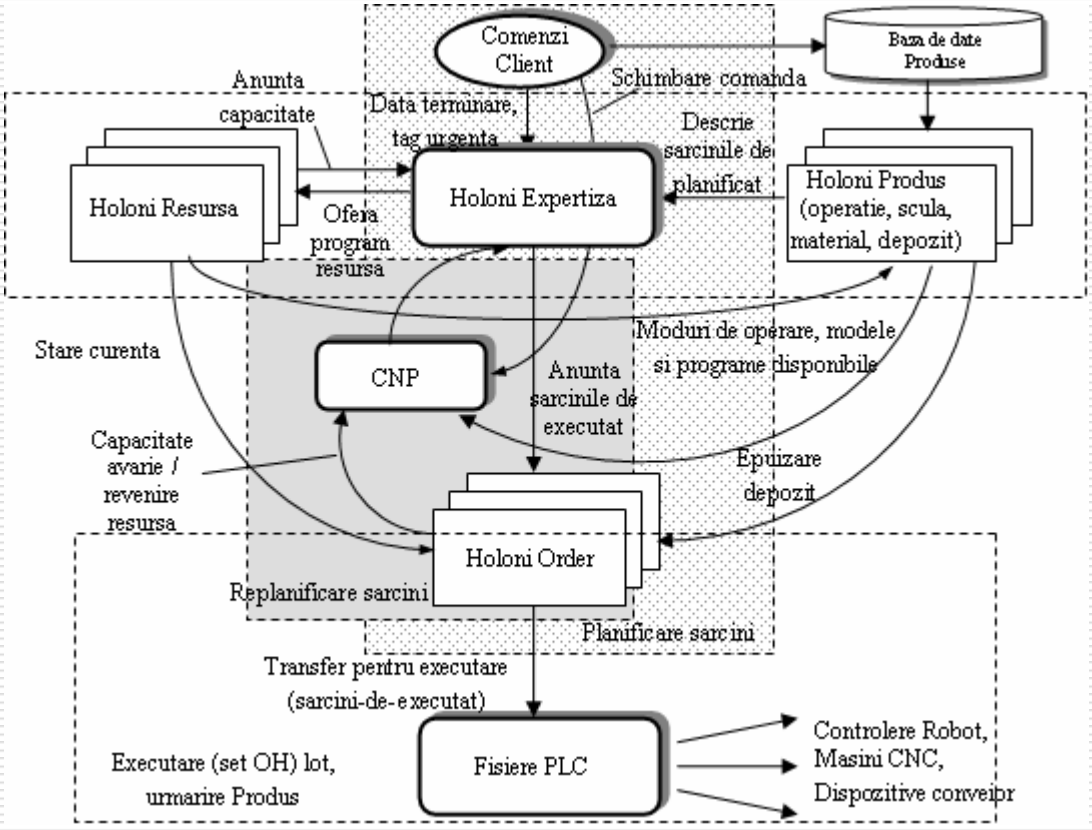
Instrumente de simulare dinamica (DST)

- DST asista si valideaza incremental:
  - crearea OH (*off-line*)
  - urmarirea productiei (*on-line*)
- DST simuleaza transportul produselor:
  - *Matrice a timpilor de transport* (TTM) creata prin masurarea timpului de miscarea a unei palete intre puncte (opritoare) conveior
  - Index minim de timp (esantionare timp transport) $t.s = 0.5$ sec
- Folosirea rutinelor nucleului DST cu *baza de timp variabila*:
  - Simulare vizuala: mod rulare cronometrat; TTM dirijata de evenimente - 0.5
  - (Re) planificare OH : mod rulare calculat, TTM declansat instanteu
- GUI – conveior divizat in sectoare



2. Holarhia si proiectarea HMES

Holarhia



- Tipuri de Holoni (referinta PROSA):
  - *de Baza*: Produs-, Resursa-, Comanda-
  - *Supervizor*: Expertiza-
- Tipuri Holarhii (set de reguli de baza pentru cooperare intre holoni pentru gestiunea si controlul sarcinilor de productie):
  - Ierarhica (planificare optimala)
  - Heterarhica (flexibilitate, agilitate la schimbari)
- Comutare automata intre cele doua holarhii:
  - *declansata* de: defectare resursa / revenire, esuare operatie, cerere de realimentare si comanda urgenta
  - bazata pe *mecanism CNP*
- Proiectare OO a holonilor: o clasa ce contine campuri de date, functionalitati si info wip
- HolonManager: coordoneaza schimburile de date intre holoni

2. Holarhia si proiectarea HMES

Structura unui Holon (selectie, exemplu):

Holon Produs (PH):

Cod Produs: #IDpr*k*;

Set operatii (vector operatii): [o_*k*1, o_*k*2, ... , o_*k*f]

Ordine operatii (vector predecesor/operatie): [p(o_*k*1), p(o_*k*2), ... , p(o_*k*f)]

unde:

p(o_*ki*) {Ø, ... , (f – 1)} sunt operatii predecesoare

Materiale (componente): vector de materiale [tip; numar.] / operatie):

[m (t, no.m)] (o_*k*1), [m (t, no.m)] (o_*k*2), ... , [m(t, no.m)] (o_*k*f),

unde

[m (t, no.m)] (o_*ki*) = [t_1*i* | no.m_1*i*, t_2*i* | no.m_2*i*, ... , t_fi | no.m_f*i*]

Scule (instrumente): vector de scule necesare [tip; numar] / operatie:

[s (t, no.s)] (o_*k*1), [s (t, no.s)] (o_*k*2), ... , [s (t, no.s)] (o_*k*f),

unde

[s [t, (no.s)] (o_*k*i) = [t_1*i* | no.s_1*i*, t_2*i* | no.s_2*i*, ... , t_u*i* | no.s_u*i*]

Programe: Vector de programe (vector de #ID programe / operatie):

IDprg(o_*k*1), IDprg(o_*k*2), ... , IDprg(o_*k*f),

unde

IDprg(o_*ki*) = IDprg_1*i*, IDprg_2*i*, ... , IDprg_v*i*

2. Holarhia si proiectarea HMES

Structura unei operatii (selectie, exemple):

MACH(*mt*, *pr*): [#IDpr; #IDmat; [mac.seq], messg()] : *prelucrare* produs/material *pe masina CNC*, cu
[mac_seq]: [... ; mac.op_*i*, time_*i*, [tool_*i*1, … , tool_*i*f_{last}, …], #IDprg_*i* , …], 1≤ *i* ≤ last.op

FEED(*pr*, *mt*, *rob*)**:** *alimentare masina unealta* cu produs/ material folosind un robot :
[pr(#IDpr, loc.pr); mat(mat.type, mat.loc); robot(rob_type, grip_type, TOOL); mt(#IDmt, wp.loc), messg ()]

ASTO(*mat*, *stor*, *rob*)**:** *alimentare depozit de materiale / componente* de la paleta de alimentare folosind un robot:
[mat(mat.type, *k*, supply_pal, mat.loc); stor(stor.type, stor.loc); robot(rob.type, grip.type, TOOL); messg()]

MNTC(*pr*, [*comp*], *rob*)**:** *montare* uneia sau mai multor *componente pe un produs* folosind un robot:
[pr(#IDpr, pr.loc); [; (type_*k*.mat, n_*k*, [mat_*ki*.loc], [tr.mat_*ki*)])..]; [..;(type_*k*.stor, stor_*k*.loc);..]; rob(); mssg()]

CHGT(*old.tool, new.tool, rob*): *schimbare scula* in gripperul robotului:
[old.tool(type.otl, otl.loc); new.tool(type.ntl, ntl.loc); robot(rob.type, grip.type, old.TOOL, new.TOOL); messg()]

FAST(*pr*, [*sub*], [*fast.el*], [*tool*], *rob*): *fixare* unul sau mai multe *subansambluri montate pe produs*, eventual cu unul sau mai multe elemente de fixare (surub, nit, capac), eventual folosind una sau mai multe *unelte*:
[pr(#IDpr, loc.pr); [sub(t, sub.loc, [fix.mod, fix.el, tol])]; [el.fix(t, stor.fix, loc.fix)]; [tol(t, tol.hld, loc.tol)]; rob(); m]

DRLA(*funct*, [*parts*[ ]], *strategy*, AOI, *video.cam*[ ]): *Detectia, Recunoasterea, Localizarea* si *Accesul pieselor bazat pe vedere artificiala*:

- **detectia**: [all(blob, AOI); all(model_AOI); all((blob + model)_AOI]
- **recunoasterea**: [blob; model_instance]
- **localizarea**: [any / particular, order]
- **accesul**: [strategy(o_1, …, o_f); [MG_*i*], [MF_*i*]; dest(o_*i*), 1≤ *i*≤ f]

2. Holarhia si proiectarea HMES

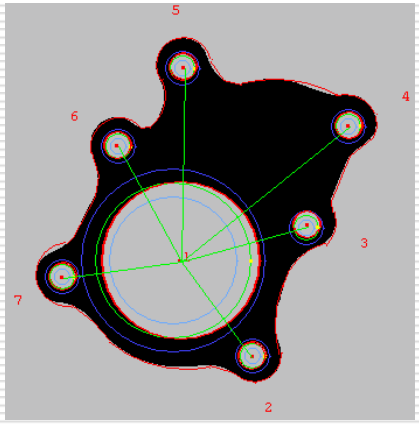
Structura unei operatii (cont'):

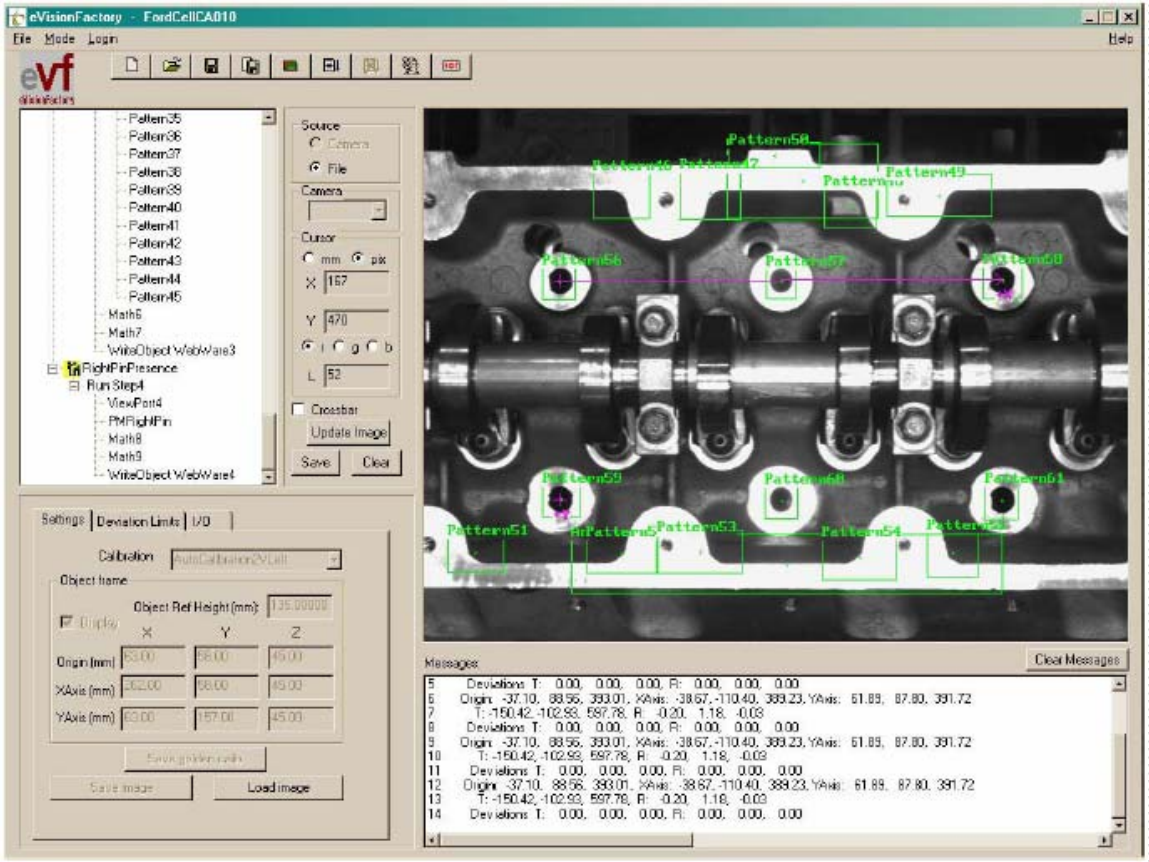
MEAS(*pr*, [*meas*], *valid*( ), *action*( ), …
… *message*( )):

inspecteaza produsul cu *masuratori vizuale*:

- GLOBAL (frontiera si regiune, descriptori scalari si spatiali)
- LOCAL (gaseste puncte-ancora, linii, arce, masoara distanta, unghi):

meas_i (meas_type, AOI, crit[cr_typ, [marking], std.val], …, res[cr_typ, [marking], meas.val])





3. Planificarea cu Holoni Expertiza

➢ **Algoritmul de planificare KB**:

Pas 0: Indexul de timp este setat la zero, se considera toate operatiile posibile bazate pe starea robotilor si restrictii predecesor ale tuturor produselor, se stocheaza in lista S (lista de operatii posibile)

Pas 1: Se aplica urmatoarele **reguli de prioritate** pentru a selecta o operatie:

- P0: daca vreuna dintre operatii apartine unui item deja introdus in sistem, se iau in considerare doar aceste operatii, altfel se iau in considerare toate operatiile gasite (*itemurile in executie au prioritate*)
- P1: se alege itemul (urile) cu cel mai mare numar de operatii succesive
- P2: se alege itemul (urile) cu cel mai mic numar de operatii in S
- P3: se alege itemul (urile) cu cel mai mare numar de operatii imediate succesive
- P4: se alege itemul (urile) cu cel mai mare numar de operatii neterminate
- P5: se alege itemul (urile) cu cel mai scurt timp de procesare
- P6: daca raman mai multe itemuri in S, atunci unul este ales aleator

Pas 2: Se planifica operatia aleasa mai sus si reactualizeaza starea resurselor

Pas 3: Se reactualizeaza lista de operatii S, daca lista este goala atunci salt la Pas 4, altfel salt la Pas 1

Pas 4: Se reactualizeaza simularea de transport si se creste indexul de timp cu un pas

Pas 5: Daca un robot a terminat la acest index de timp, se marcheaza operatia ca terminata si ca planificata

Pas 6: Se reactualizeaza starea robotilor si lista operatiilor S; daca S este goala, salt la Pas 1, altfel salt la Pas 4

3. Planificarea cu Holoni Expertiza. Planificatorul "pas-cu-pas"

- **Principii de baza ale planificarii**:
  - Rutinele programului de simulare joaca un rol esential in acest tip de planificare (folosita de asemenea in timp real cu P0 ca regula de prioritate unica in mecanismul de replanificare CNP)
  - La fiecare moment de timp este imperativ sa se stie locatia exacta a fiecarui produs_pe_paleta, astfel incat paletele sa nu se suprapuna sau ca sa nu depaseasca una pe alta. Simulatorul este folosit:
    - *fara vizualizare grafica* si
    - *ruland la viteza maxima* data de frecventa procesorului pentru un emulare realista a transportului
  - Variabilele de simulare contin informatii despre:
    - *starea curenta a sistemului*;
    - *numarul de palete in sistem* la indexul curent de timp
  - Indexul de timp este incrementat in perioade de esantionare comprimate ce corespund pasilor de 0,5 sec; astfel la orice punct al planificarii indexul curent de timp poate fi citit si memorat.
  - Acesti indexi de timp (decomprimati la 0.5 sec) declanseaza executia productiei in timp real; indecsi de timp de interes memorate in timpul generarii planificarii vor dicta:
    - *timpii de introducere (intrare)e* a itemurilor (palete goale) in sistem;
    - *timpul* la care o anumita *operatie este inceputa*;
    - *timpul* la care o anumita *operatie ar trebui terminata*.
  - Algoritmul are o comportare de tip planificator incremental de secvente de productie (RSRP)

3. Planificarea cu Holoni Expertiza. Planificatorul "pas-cu-pas"

Pas 1: Pentru mai putin de 2 palete in sistem, salt la Pas 2, altfel salt la Pas 3

Pas 2: Se alege un element din lista S folosind o regula de prioritate *proprie* sau se alege *una dintre P1-P6*

2.1: pentru itemul ales se genereaza o lista a tuturor operatiilor posibile bazata pe restrictiile de predecesor

2.2: pentru fiecare operatie posibila se gasesc toti robotii capabili sa o execute si se calculeaza timpul de asteptare pentru fiecare robot inainte ca sarcina sa poata fi executata cind itemul ajunge la postul robot

2.3: se alege operatia cu cel mai mic timp de asteptare; se introduce acest item pe o noua paleta in sistem cu destinatia precis stabilita si se memoreaza indexul curent de timp ca timpul introducerii in sistem

Pas 3: Se executa simularea conveiorului cu un increment de index de timp si operatiile la robot. Daca un robot termina o operatie, salt la Pas 4. Daca o paleta ajunge la un robot, salt la Pas 6, altfel salt la Pas 5.

Pas 4: Pentru itemul care tocmai a terminat o operatie, se memoreaza indexul de timp curent ca *timpul de terminare a operatiei*, se marcheaza robotul ca *liber*, apoi:

4.1: se determina daca acest item este terminat; daca da, atunci este marcat si se scoate paleta, salt la Pas 1

4.2: pentru itemul ales, se genereaza o lista cu toate operatiile posibile bazata pe constrangerile de predecesor

4.3: pentru fiecare operatie posibila, se gasesc toti robotii capabili sa o execute si se calculeaza timpul de asteptare pentru fiecare robot de cand paleta ajunge la el pana cand operatia ar putea fi executata

4.4: se alege operatia cu cel mai scurt timp de incepere si se trimite paleta la acea statie robot

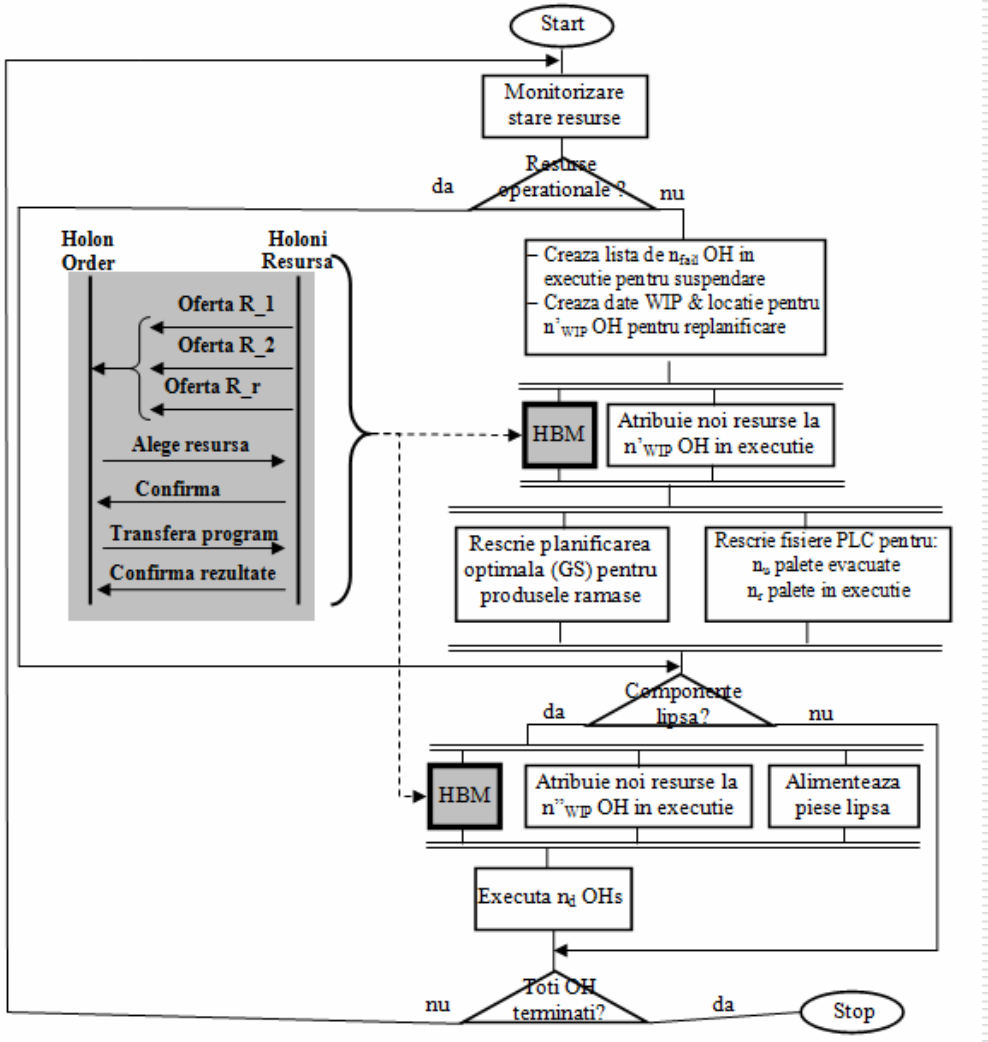
Pas 5: Daca inca mai sunt item-uri in lista OH sau palete in sistem atunci salt la Pas 1, altfel inchide algoritmul

Pas 6: Pentru itemul sosit se memoreaza indexul de timp curent ca momentul inceperii operatiei, se asigneaza acest item robotului si se marcheaza robotul ca *ocupat*, salt la Pas 1.

4. Gestiunea schimbarilor cu CNP incorporat

Diagrama de gestiune a schimbarilor:

- HBM: Mecanism de Licitare Holonic de tip CNP, asigneaza operatii resurselor
- N: toti OH planificati
- n_{WIP}: OH in executie curenta
- n_{fin}: OH terminati
- n_d: OH intarziati din cauza lipsei de piese
- n_{fail}: OH nereusiti (in executie, nu mai pot fi terminati)
- n_e: OH neinceputi inca, nu mai pot fi executati
- $n'_{WIP} = n_{wip} - n_{fail}$: OH in executie, pot fi continuati fara replanificare
- $n''_{WIP} = n_{wip} - n_d$: OH ce pot fi executati prin replanificare pe resurse ce dispun de piese
- $n_c = N - n_{fin} - n_{WIP} - n_e$: OH in asteptare, pot fi procesati



4. Gestiunea schimbarilor cu CNP incorporat. Defectare resurse si realimentare

Defectarea unei resurse (replanificare OH):

1. Se suspenda temporar fabricatia
2. Se reactualizeaza RH cu noile stari ale robotilor
3. Se citesc comenzile din sistem
4. Se evalueaza toate comenzile daca mai pot fi produse de sistem
5. Se verifica starea fiecarei comenzi
   - daca comanda a fost la robotul avariat, se marcheaza produsul ca esuat si se scoate din sistem
   - daca comanda este in sistem, dar nu poate fi terminata din cauza resursei avariate, se marcheaza ca esuata si se evacueaza din sistem
   - daca comanda nu este inca in sistem, dar nu poate fi terminata din cauza resursei avariate, atunci se marcheaza ca esuata
6. Produsele ramase in sistem sunt localizate si se initiaza simularea de transport. Contract Net Protocol este inclus in aceasta etapa
7. Se ruleaza algoritmul de planificare luand comenzile terminate si esuate (care nu mai au nevoie de replanificare)
8. Se sterg comenzile anterioare stocate in sistem
9. Se transfera sistemului comenzile reactualizate
10. Se reia productia

Realimentare posturi de lucru (generare SH):

1. Un SH este creat prin specificarea *pieselor ce trebuiesc extrase* din 1 sau 2 dispozitive flexibile de alimentare de catre un robot SCARA ghidat prin VA si *ruta paletei de aprovizionare* catre depozitul golit

2. Dintre OH in executie curenta, n_d vor fi intarziati pana cand depozitul gol este realimentat, si $n''_{WIP} = n_{wip} - n_d$ OH vor fi replanificati de mecanismul holonic de licitare la robotii ce dispun de toate piesele necesare.

4. Managementul schimbarilor cu CNP incorporat. Integrarea in sistem

Integrarea AP in sistem pentru controlul executiei OH, SH:

- Aplicatia de planificare comunica cu AP al liniei de transfer prin *protocolul OPC*; conexiune Ethernet GPS – PLC. GPS transmite catre AP lista OH si primeste starea RH (AP interogheaza periodic RC)
- Fiecare paleta are propriul cod; codul poate fi scris/citit in puncte de bifurcatie a benzii de dispozitive R/W
- O matrice dubla a fost definita pentru AP, continand 256 de matrici (max. 256 palete in fiecare lot), fiecare
- constind din 16 structuri (un *produs_pe_paleta* poate suporta maxim 16 operatii)

```
Pallets_array: ARRAY [0..255,1..16] OF PalletData;
```

- `PalletData`: structura informationala de baza (despre operatii ce pot fi facute pe un produs la statii robot) :

```
STRUCT
    station: BYTE;(* ID al statiei unde va fi facuta operatia *)
    operation: BYTE;(* tipul operatiei *)
    timemin: WORD; (* durata minima a operatiei *)
    timemax: WORD;(* durata maxima a operatiei *)
    report: BYTE;(* un bit cod ce indica cum a fost executata operatia-ok,esec *)
      END_STRUCT
```

- Aplicatia de planificare trimite la PLC o matrice cu *timpi propusi* ai inceperii executiei OH (insertia paletei):

```
time_insertion_array[0..255]: word
```

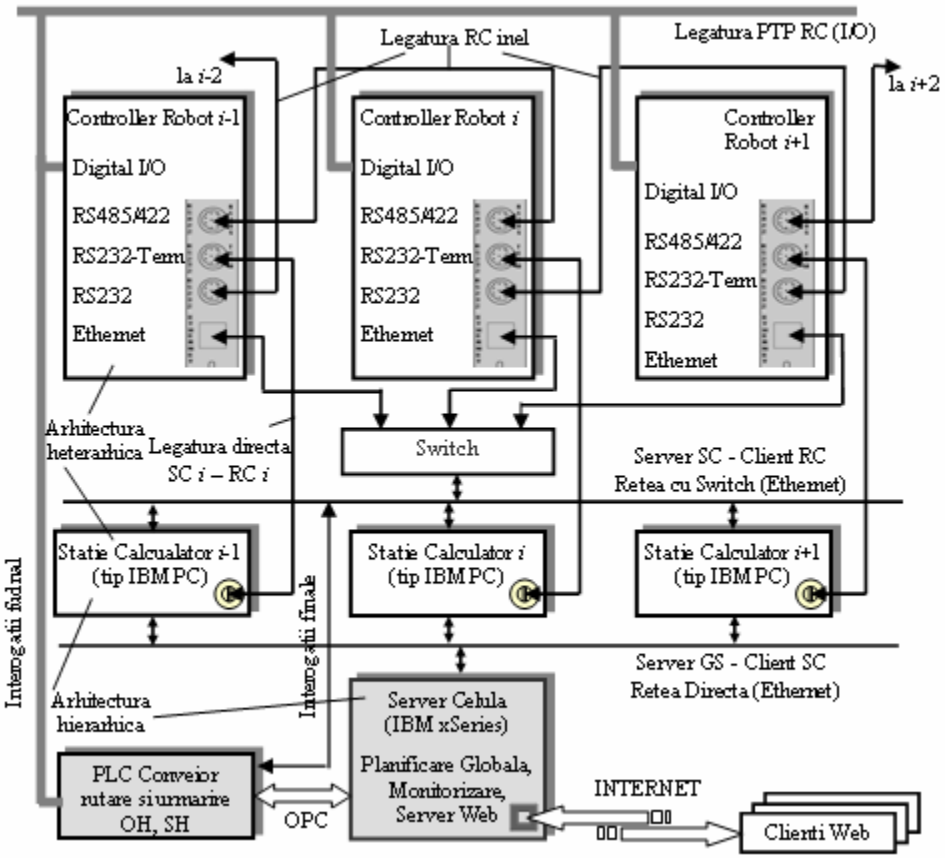
- Matrice cu 256-itemuri *index_array*; fiecare item *it* este asociat cu paleta avand cod ID indexul lui *it* in *index_array*. Valoarea acelui item este indexul *urmatoarei operatii* de executat pe paleta respectiva:

```
Index_array: ARRAY [0..255]OF BYTE;
```

4. Integrarea sistemului. Control si comunicatie tolerante la defect (FT)

Control si comunicatie FT

- **Ethernet cu comutare (switch)**: daca un Calculator de Statie (SC) se defecteaza, sarcina sa este preluata de o SC valida (datele SC defecte sunt *replicate* intre SC-uri)
- **Seriala Directa**: defectarea unui RC este detectata prin monitorizare continua prin *legaturi seriale directe* SC*i* – RC*i*
- **Inel Inter-Controllere Robot** : daca *Switch-ul* e avariat, comunicatia dintre clusterele SC si RC inca va functiona prin *inelul de legaturi* directe [SC*i* – RC*i*] si [RC*i* – RC*i+1*]
- **Punct-la-Punct I/O**: *interconditionarea operationala intre joburi RC* (excludere mutuala, sincronizare) este realizata de catre o retea I/O *conectata-incrucisat*



5. Rezultate experimentale. Concluzii si dezvoltari viitoare

5. Rezultate experimentale. Concluzii si dezvoltari viitoare

Assembly sequence H-product:
1. Mount axis
2. Mount axis
3. Mount axis
4. Mount T
5. Vision inspection
6. Mount I
7. Mill I
8. Vision inspection

Assembly sequence U-product:
1. Mount axis
2. Mount axis
3. Mount axis
4. Mount L
5. Vision inspection
6. Mount I
7. Mill I
8. Vision inspection

Assembly sequence L-product:
1. Mount axis
2. Mount axis
3. Mount axis
4. Mount I
5. Vision inspection
6. Mount I
7. Mill I
8. Vision inspection

Assembly sequence C-product:
1. Mount axis
2. Mount axis
3. Mount axis
4. Mount r
5. Vision inspection
6. Mount r
7. Vision inspection

Timpi de fabricatie pentru loturi de produse de tip H-, U-, L- and C-, si timpi de reluare la defectare resursa / golire depozit

Dim. Lot [buc]	Timp de fabricatie [unitati de timp]		Cel mai defavorabil timp [unitati de timp]	
	H-type [RSRP / CNP]	Numar egal de produse H-, U-, L-, C- [RSRP/Negociere CNP]	OH alternative la nivel [pachet = 5] (defectare resursa *i*: R*i*F)	SH nou pentru refacerea unui Depozit Local *i* (DL*i*)
4	684 / 734	663 / 687	6.4 (R1F)	97 (DL1)
20	2841 / 3112	2550 / 2712	6.5 (R2F)	112 (DL2)
40	5485 / 5962	4934 / 5288	6.8 (R3F)	136 (DL3)
60	8129 / 9089	7362 / 7902	6.5 (R4F)	83 (DL4)

5. Rezultate experimentale. Concluzii si dezvoltari viitoare

- Structura distribuita de conducere automata a productiei. Integrare business-proces
- Arhitectura de control semi-heterarhica, cu comutare automata ierarhic / heterarhic declansata de:
  - ✓ Aparitia de comenzi prioritare (rush orders –Earliest Deadline First)
  - ✓ Defectarea unei resurse
  - ✓ Realimentarea unui depozit local
- Agilitate prin reconfigurare, nu reprogramare
- Replanificare pe orizont [pachet = 5] *pipeline* Replanificare orizont [rest lot]
- AP: interogare *periodica* toate resursele | *ultimativa* resursa alocata jobului curent
- Noi strategii si algoritmi de negociere pe baza de contract
- Dezvoltare IMS – RFID (Enhanced Information Management System – RFID)
- Abordari noi: “Automatizarea fabricatiei dirijata de produs”, “Produs inteligent”

Sfarsit

Va multumesc !