mgr inż. Mateusz Pierzchała
ASTOR Poznań
dr inż. Wiesław Pierzchała
Politechnika Krakowska im. T. Kościuszki
Katedra Systemów Wytwarzania

INTEGRACJA OPERATYWNEGO STEROWANIA PROCESEM WYTWARZANIA Z SYSTEMAMI ZARZĄDZANIA PRODUKCJĄ

W pracy przedstawiono sposób powiązania układu operatywnego sterowania procesem wytwarzania z wyższymi poziomami zarządzania jego przebiegiem. W opracowanym układzie poziom zarządzania produkcją jest reprezentowany przez pakiet InTrack™ firmy Wonderware. Przedstawiona metoda polega na zapewnieniu przepływu danych pomiędzy układem sterowania operatywnego, wykorzystującym Skalowalny Model Obiektowy systemu wytwarzania, a bazą danych programu InTrack, zawierającą informacje o złożonych zamówieniach oraz o przebiegu ich realizacji.

INTEGRATION OF PRODUCTION CONTROL WITH MANUFACTURING EXECUTION SYSTEMS

The paper presents the way of connecting the operational level of the production control with the higher levels of manufacturing management – the manufacturing execution systems (MES). In developed system the MES level is represented by Wonderware® InTrack™ package. Presented method consists in providing the data flow between manufacturing control system (which utilizes the Scalable Object Model of production process) and InTrack database (which contains the information about orders and the production advance).

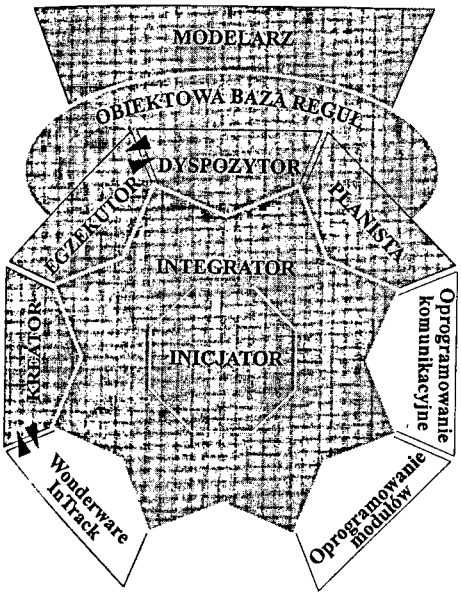
1. WPROWADZENIE

Prezentowany układ komputerowego sterowania przepływem produkcji zautomatyzowanej jest wynikiem wieloletniego udziału autorów w pracach z tego zakresu, prowadzonych w Katedrze Systemów Wytwarzania Politechniki Krakowskiej. Wcześniejsze doświadczenia oraz rezultaty badań były prezentowane m. in. w pracach [2], [3], [4]. Uzyskane, satysfakcjonujące efekty skłoniły do podjęcia próby powiązania układu operatywnego sterowania produkcją z wyższymi poziomami zarządzania jej przebiegiem. W referacie omówiono stan realizacji tego zamierzenia.

Główne zadanie sterowania operatywnego, polegające na podejmowaniu decyzji zapewniających właściwe i pożądane współdziałanie urządzeń wytwórczych, jest realizowane w prezentowanym układzie na podstawie przetwarzanego w czasie rzeczywistym oryginalnego, Skalowalnego Modelu Obiektowego (SMO) systemu wytwarzania. Z kolei, na taktycznym poziomie zarządzania produkcją zastosowano program InTrack 7.0, który jest jednym z najnowszych przykładów oprogramowania MES (*ang. Manufacturing Execution Systems*) przeznaczonego na komputery klasy PC, wchodzącym w skład pakietu

oprogramowania przemysłowego *FactorySuite 2000* firmy *Wonderware*. Program ten umożliwia m. in. śledzenie przepływu materiałów (zarówno surowców, jak i gotowych wyrobów), gromadzenie informacji o przebiegu procesu produkcyjnego w bazie danych oraz udostępnianie tych informacji w celach analitycznych lub dokumentacyjnych (np. w formie generowanych automatycznie raportów). InTrack posiada ponadto możliwość wymiany danych z najwyższym poziomem zarządzania przedsiębiorstwem (*ang. Enterprise Resource Planning – ERP*). Stąd też, uznano, że InTrack może stanowić bardzo dobry, a jednocześnie stosunkowo tani „pomost" pomiędzy najniższymi a najwyższymi poziomami w Systemie Zarządzania Produkcją.

2. STRUKTURA UKŁADU STEROWANIA PRZEPŁYWEM PRODUKCJI



Rys. 1. Otwarty układ sterowania przepływem produkcji

Opracowany układ sterowania przepływem produkcji rys. 1, umożliwiający płynny przepływ zleceń produkcyjnych w zautomatyzowanym systemie wytwarzania, niezależnie od dynamiki zmian asortymentowych wynikających z bieżącego zapotrzebowania na wyroby, jest rozproszonym układem typu klient / serwer, złożonym z szeregu aplikacji przeznaczonych dla systemu Windows NT 4.0, napisanych w języku C++ z wykorzystaniem kompilatora Micosoft Visual C++ 6.0.

2.1. MODELARZ

Jak napisano wcześniej, w układzie tym wykorzystywany jest przetwarzany współbieżnie z realizowanym procesem, Skalowalny Model Obiektowy (SMO) sterowanego systemu wytwarzania, którego funkcjonowanie jest zdyskretyzowane poprzez wyodrębnienie skończonego zbioru czynności elementarnych. SMO jest budowany dla zadanego zbioru zleceń przez program DYSPOZYTOR, w trybie samokreowania na podstawie Obiektowej Bazy Reguł (OBR). Baza ta jest tworzona przez program MODELARZ i opisuje wszystkie elementy systemu wytwarzania, ich funkcje, reguły działania oraz potencjalne możliwości współpracy [1].

MODELARZ jest wieloetapowym kreatorem, w którym każdy etap stanowi okno dialogowe przeznaczone do wprowadzania poszczególnych elementów bazy. Pierwszym etapem budowania OBR jest wprowadzenie listy *typów elementów* systemu produkcyjnego. Przez *typ elementów* rozumie się wszystkie elementy (maszyny, urządzenia, moduły) pełniące identyczne funkcje (identycznie „zachowujące się") w systemie wytwarzania. Następnie sporządza się listy *wzorców czynności elementarnych* realizowanych przez te elementy. *Wzorzec* reprezentuje wszystkie czynności o identycznym charakterze, w których biorą udział elementy tych samych typów. Na przykład, wszystkie czynności obróbki wszystkich przedmiotów jednego typu na jednego typu obrabiarkach są reprezentowane przez jeden wzorzec. Po sporządzeniu list wzorców czynności elementarnych projektant definiuje porządek ich wykonywania, przez co określa sposób działania poszczególnych urządzeń.

Polega to na tym, że każdemu ze zdefiniowanych wzorców przyporządkowuje się listę jego następników. W końcu, dla algorytmicznie wyznaczonych przez MODELARZA *wzorcowych cykli elementarnych* definiowane są *warunki dopuszczalności* (*realizowalności*), które pozwalają unikać *zastojów* podczas eksploatacji systemu wytwarzania.
Dla konkretnych elementów przypisanych do poszczególnych ich typów, OBR jest przekształcana («Kompiluj») w plik **.smo*, akceptowany przez program DYSPOZYTOR.

2.2. INICJATOR

INICJATOR umożliwia skonfigurowanie i uruchomienie wszystkich komponentów układu. W oknie tego programu operator wskazuje, które aplikacje powinny być uruchomione, podaje adresy komputerów, na których one się znajdują, oraz numery portów, które wykorzystują do odbierania informacji.

2.3. DYSPOZYTOR i EGZEKUTOR

Skalowalny Model Obiektowy (SMO) systemu wytwarzania jest budowany dla zadanego zbioru zleceń przez program DYSPOZYTOR, w trybie samokreowania na podstawie Obiektowej Bazy Reguł. Model jest przekształcany przez program EGZEKUTOR w momencie zaistnienia zdarzeń rozpoczęcia lub zakończenia czynności elementarnych. EGZEKUTOR wyznacza czynność dopuszczalną, którą DYSPOZYTOR zleca do wykonania, korzystając z odpowiedniego *oprogramowania komunikacyjnego*. Jeżeli żadna czynność nie może być rozpoczęta, DYSPOZYTOR oczekuje na potwierdzenie czynności wcześniej zleconej i gdy je otrzyma, przekazuje do EGZEKUTORA, który dokonuje stosownego przekształcenia SMO. DYSPOZYTOR współpracuje z pakietem do zarządzania produkcją InTrack. Z jego bazy danych pobiera zlecenia skierowane do realizacji i w miarę postępu procesu produkcyjnego dokonuje w niej stosownych modyfikacji. Współdziałanie wymienionych programów odbywa się za pośrednictwem programu INTEGRATOR.
DYSPOZYTOR i zintegrowany z nim EGZEKUTOR zostały znacznie zmodyfikowane w stosunku do swoich poprzednich wersji, przedstawionych m. in. w pracy [2]. Ich rola i idea działania pozostały jednakże niezmienione, toteż nie będą tutaj szerzej opisywane.

2.4. INTEGRATOR

INTEGRATOR jest programem mającym kluczowe znaczenie dla współpracy poszczególnych komponentów układu sterowania. Pełni on funkcję uniwersalnego, konfigurowalnego interfejsu pomiędzy poszczególnymi aplikacjami i zapewnia każdej z nich dostęp do wszelkich niezbędnych danych, niezależnie od źródła ich pochodzenia, w odpowiednim formacie. Zadania INTEGRATORA zostaną uporządkowane poprzez wydzielenie *powiązań komunikacyjnych* przezeń obsługiwanych, z których każde służy oddzielnemu celowi. Powiązania te zostały wyszczególnione poniżej, wraz z opisem ich roli.

DYSPOZYTOR ↔ INTEGRATOR ↔ Oprogramowanie komunikacyjne ↔ Sterowniki

INTEGRATOR odbiera polecenia wykonania czynności, wysyłane przez DYSPOZYTORA w postaci ich nazw, tłumaczy te polecenia na postać bloków danych „zrozumiałych" dla sterowników urządzeń systemu oraz przesyła wygenerowane bloki do tych urządzeń za pośrednictwem *oprogramowania komunikacyjnego*. Bloki danych generowane są na podstawie programu skryptowego, w którym dla każdej czynności (lub grupy czynności) zdefiniowany jest skrypt wypełniający pola bloku danych i zlecający wysłanie go do właściwego adresata (te skrypty opisano np. w pracach [3], [4]). To powiązanie służy również

do przesyłania potwierdzeń wykonania czynności. INTEGRATOR dopasowuje odebrany ze sterownika blok do czynności będącej poleceniem i odsyła jej nazwę do DYSPOZYTORA.

DYSPOZYTOR ↔ INTEGRATOR ↔ Program InTrack

Powiązanie wykorzystywane do informowania programu InTrack o przebiegu procesu produkcyjnego, a także do pobierania z bazy danych tego programu informacji o zamówieniach skierowanych do realizacji. Pierwsza z tych funkcji realizowana jest za pośrednictwem programu skryptowego. W skryptach realizowanych podczas wysyłania poleceń rozpoczęcia czynności oraz podczas odbierania potwierdzeń, można wykorzystywać polecenia służące do komunikowania się z programem InTrack. Polecenia te zostaną poniżej opisane. Druga funkcja realizowana jest za pomocą zapytania SQL zadawanego do bazy danych programu InTrack, które selekcjonuje z niej listę zamówień skierowanych do realizacji. Uzyskane dane są udostępniane przez INTEGRATORA DYSPOZYTOROWI.

DYSPOZYTOR ↔ INTEGRATOR ↔ PLANISTA

Powiązanie jest wykorzystywane do realizowania wszystkich zadań związanych ze współpracą PLANISTY z DYSPOZYTOREM, opisanej w pacy [5]. Tą drogą przesyłane są polecenia uruchomienia procesu wirtualnego oraz wyniki weryfikacji wariantów planu.

PLANISTA ↔ INTEGRATOR ↔ InTrack

Powiązanie jest wykorzystywane do realizowania wszystkich zadań związanych ze współpracą PLANISTY z programem InTrack. Tą drogą przesyłane są listy zamówień wprowadzonych do bazy danych programu InTrack oraz wyniki planowania, tzn. informacje, które zlecenia zostały skierowane do wykonania.

INTEGRATOR ↔ Oprogramowanie modułów

Powiązanie służy do komunikacji z dodatkowym oprogramowaniem obsługującym różne moduły systemu (np. moduł magazynowania, moduł narzędziowy). Oprogramowanie takie może udostępniać INTEGRATOROWI dane wymagane przez inne komponenty układu.

INICJATOR ↔ INTEGRATOR ↔ Pozostałe aplikacje

Powiązanie służy do przekazywania informacji i poleceń pomiędzy programem INICJATOR a pozostałymi aplikacjami układu.

Można wyróżnić trzy główne moduły programu INTEGRATOR:

1. Moduł kliencki interfejsu OLE Automation. Zapewnia on wymianę informacji z programem InTrack. Dzięki niemu INTEGRATOR może zadawać bezpośrednie zapytania do bazy danych tego programu w celu wyselekcjonowania pożądanych informacji, może również wykonywać wszystkie jego funkcje i procedury, dzięki czemu może informować go o przebiegu produkcji (przepływie produktów, realizowanych operacjach itd.).
2. Moduł komunikacyjny dla protokołu 3P. Protokół ten wykorzystywany jest do przekazywania wszelkich danych pomiędzy komponentami układu.
3. Interpreter języka skryptowego. Dzięki skryptom INTEGRATOR jest w pełni konfigurowalny i niezależny od konkretnego systemu wytwarzania, dla którego jest zastosowany. Każdy skrypt składa się z dwóch części. Pierwsza zawiera instrukcje przeznaczone do wykonania po odebraniu polecenia z DYSPOZYTORA, druga – instrukcje wykonywane po odebraniu potwierdzenia wykonania czynności. W skryptach stosowane tzw. *szablony czynności*. Jeżeli w modelu zdefiniowanych jest kilka podobnych czynności, które wymagają wykonania podobnych operacji, można zdefiniować dla nich szablon i jeden skrypt (szczegóły tej koncepcji można znaleźć w pracach [3], [4]).

Język skryptowy wyposażono w polecenia umożliwiające powiązanie z programem InTrack:

- `_Connect(użytkownik,hasło)` – polecenie służące do nawiązania połączenia z bazą danych programu InTrack i zalogowania się do niej. Parametrami polecenia są: identyfikator użytkownika bazy danych oraz hasło.

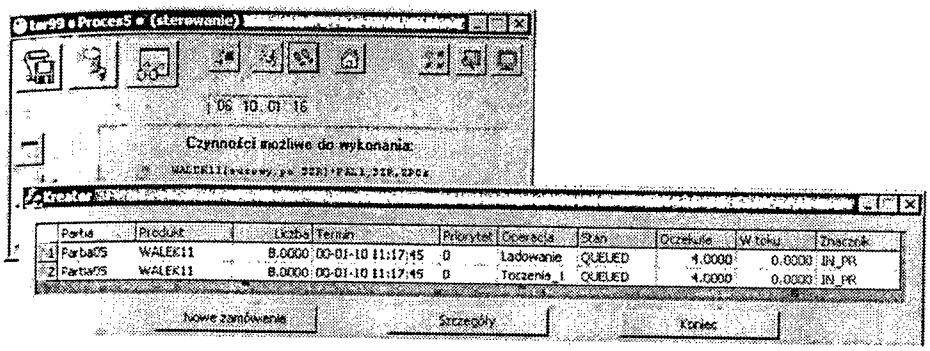
- `_Disconnect()` – polecenie służące do zakończenia połączenia z bazą.
- `_Start(partia,marszruta,operacja,maszyna,liczba)` – polecenie służące do przesłania programowi InTrack informacji o rozpoczęciu wykonywania określonej operacji technologicznej. Jego parametrami są: nazwa partii produktu, dla którego rozpoczyna się operacja, nazwa marszruty technologicznej, numer kolejny operacji w ramach marszruty, maszyna, na której odbywa się operacja, oraz liczba sztuk produktu.
- `_Consume(partia,ilość)` – polecenie służące do przesłania programowi InTrack polecenia pobrania określonej ilości materiału wejściowego (półfabrykatu) niezbędnego na aktualnym etapie produkcji. Jego parametrami są: nazwa partii półfabrykatu, z której będzie on pobierany, oraz pobierana ilość (lub liczba sztuk) półfabrykatu.
- `_Complete(partia,marszruta,operacja,maszyna,dyspozycja)` – polecenie służące do przesłania programowi InTrack informacji o zakończeniu wykonywania określonej operacji technologicznej. Jego parametrami są: nazwa partii produktu, dla którego kończona jest operacja, nazwa marszruty technologicznej, numer kolejny operacji w ramach marszruty, maszyna, na której odbywała się operacja, oraz zdefiniowana w programie InTrack dyspozycja, która zostanie nadana partii w bazie danych.
- `_Close(partia,marszruta,operacja,maszyna,dyspozycja, nowa_partia)` - polecenie służące do przesłania programowi InTrack informacji o zakończeniu wykonywania określonej operacji technologicznej, która jest jednocześnie ostatnią operacją marszruty technologicznej. Jego parametry są identyczne, jak dla polecenia `_Complete()`. Dodatkowy parametr to nazwa partii gotowego produktu, która zostanie utworzona w efekcie zakończenia produkcji.
- `_GetLotID()` – polecenie zwracające nazwę partii produktu, dla której wykonywana jest czynność przetwarzana aktualnie przez skrypt. Z każdym poleceniem wykonania czynności INTEGRATOR otrzymuje informację o nazwie partii, do której należą przedmioty biorące udział w tej czynności. Informacja ta jest niezbędna do prawidłowego wykonywania poleceń `_Start()`, `_Consume()`, `_Close()`, oraz `_Complete()`.

Wykonanie każdego z tych poleceń skutkuje wywołanie odpowiedniej funkcji programu InTrack, za pośrednictwem interfejsu OLE Automation.

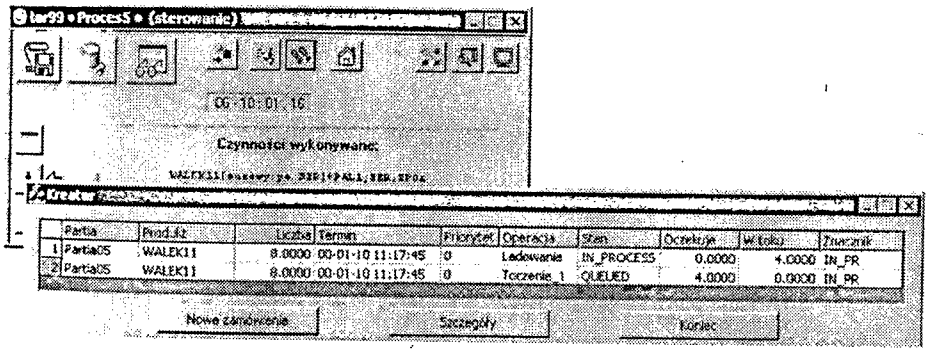
Oto przykład skryptu dla szablonu czynności, do którego pasuje np. czynność elementarna: WALEK11[surowy:po SZR] + PAL1;SZR, ZP0& (ładowanie półfabrykatów WALEK11 na paletę PAL1, na stacji załadowczo rozładowczej SZR, ze zmieniaczem palet ZP0).

```
/$1[$2:$3]+$4;SZR,ZP0&        ; skrypt wykonywany w momencie rozpoczynania czynności
  @Act_code = 1
  @Pal_num = _GetObjectPaletteNum([ZP0])
  @Pal_type = [$4]
  @Wp_num = [$1]
  @Wp_state = [$2]
  _Send("SZR")                ; wysłanie przygotowanego bloku danych
  _Start(_GetLotID(),[ROUTE$1],1,"SZR",[NUM$1])     ; polecenie InTrack
  _Consume("polfabr@magazyn_polfabrykatow",[NUM$1]) ; polecenie InTrack
CONFIRM                       ; skrypt wykonywany w momencie zakończenia czynności
  State = [$3]
  _SetObjectState([ZP0],@Pal_num,@Pal_type,@Wp_num,State)
  _Complete(_GetLotID(),[ROUTE$1],1,"SZR","OK")     ; polecenie InTrack
```

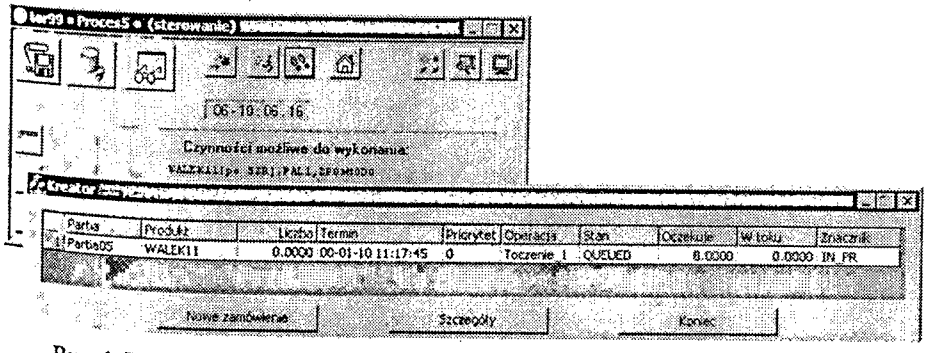
Zamieszczone poniżej trzy rysunki ilustrują skutki działania tego skryptu, pokazując okno DYSPOZYTORA i podgląd zawartości bazy danych programu InTrack, kolejno: przed rozpoczęciem czynności, w trakcie jej wykonywania (zrealizowane zamieszczone w skrypcie polecenia `_Start` i `_Consume`) oraz po jej zakończeniu (zrealizowane polecenie `_Complete`).



Rys. 2. Przed rozpoczęciem czynności WALEK11[surowy:po SZR]+PAL1;SZR,ZP0&



Rys. 3. W trakcie wykonywania czynności WALEK11[surowy:po SZR]+PAL1;SZR,ZP0&



Rys. 4. Po zakończeniu czynności WALEK11[surowy:po SZR]+PAL1;SZR,ZP0&

3. MODUŁ KOMUNIKACYJNY 3P

Protokół 3P (Protokół Przekazywania Poleceń) jest standardem przekazywania danych i poleceń, opracowywanym na potrzeby opisywanego układu. Obecna postać protokołu realizuje aktualne potrzeby w zakresie wymiany informacji pomiędzy wszystkimi opisanymi

powyżej komponentami. Protokół ten ma jednak strukturę otwartą i razie potrzeby będzie rozbudowywany tak, by mógł realizować kolejne zadania.

3P jest protokołem sieciowym, opartym na standardzie sieci lokalnej Ethernet, a w sensie programowym – na protokole sieciowym TCP/IP i oferowanym przezeń mechanizmie *gniazd* (*ang. sockets*). W każdej aplikacji zdefiniowane są gniazda nadawcze i odbiorcze, pomiędzy którymi przesyłane są bloki danych zgodnie z protokołem TCP. Adresowanie komputerów i gniazd odbywa się zgodnie z protokołem IP: adres komputera ma postać czterech liczb oddzielonych kropkami, numer gniazda jest dodatkową piątą liczbą w adresie.

Protokół 3P opisuje format poleceń i bloków danych przesyłanych pomiędzy aplikacjami otwartego układu sterowania oraz zapewnia dostarczanie tych informacji odpowiednim aplikacjom we właściwej postaci. W tym sensie jest więc względem TCP/IP protokołem wyższego poziomu. Zadania te są realizowane przez moduły komunikacyjne protokołu 3P, dołączone do wszystkich komponentów układu.

Moduł komunikacyjny 3P zajmuje się:

- obsługą gniazd nadawczych i odbiorczych aplikacji;
- wysyłaniem poleceń i bloków;
- akceptowaniem połączeń nawiązywanych przez inne aplikacje;
- odbieraniem bloków i poleceń oraz buforowaniem ich;
- udostępnianiem aplikacji odebranych bloków i poleceń na żądanie.

Polecenia i dane są przesyłane w ramach protokołu 3P jako ciągi znaków o ustalonym formacie. Ciąg taki ma następującą postać:

```
_POLECENIE(parametry)
```

Poniżej przedstawiono najważniejsze polecenia obecnie obsługiwane w protokole:

`_SELECTORDERS()` – polecenie udostępnienia listy zamówień wprowadzonych do bazy danych programu InTrack. Polecenie to jest kierowane przez DYSPOZYTORA oraz PLANISTĘ do INTEGRATORA, który je realizuje.

`_ORDERS(lista_zamówień)` – w takiej postaci INTEGRATOR przesyła listę zamówień. Poszczególne zamówienia oddzielone są znakiem „|", a dla każdego z nich określony jest szereg parametrów, w kolejności: nazwa partii, nazwa produktu, liczba sztuk, priorytet, najwcześniejszy i najpóźniejszy termin wykonania, termin krytyczny oraz najwcześniejszy termin rozpoczęcia realizacji. Przykład listy trzech zamówień:

```
_ORDERS(Partia1,WALEK11,80,0,00-01-05,00-01-05,00-01-03,
    00-01-01|Partia2,WALEK12,40,0,00-01-08,00-01-08,00-01-07,
    00-01-03|Partia3,WALEK21,160,0,00-01-09,00-01-09,00-01-06,
    00-01-01)
```

`_MAKEPLAN(lista_zamówień)` – polecenie weryfikacji wariantu planu, przesyłane przez PLANISTĘ do DYSPOZYTORA. Lista zamówień przesyłana jest w identycznej postaci, jak w przypadku polecenia `_ORDERS()`.

`_ACCEPTPLAN(lista_zamówień)` – polecenie akceptacji wariantu planu, przesyłane przez PLANISTĘ do INTEGRATORA. Lista kierowanych do wykonania zamówień przesyłana jest w identycznej postaci, jak w przypadku polecenia `_ORDERS()`. INTEGRATOR dla każdego zaakceptowanego zlecenia ustawia odpowiedni znacznik w bazie danych programu InTrack.

`_PERFORM(nazwa_czynności)` – polecenie wykonania czynności, przesyłane przez DYSPOZYTORA do INTEGRATORA.

`_CONFIRM(nazwa_czynności)` – potwierdzenie wykonania czynności, przesyłane przez INTEGRATORA do DYSPOZYTORA.

- DATABLOCK(*blok_danych*) – w tej postaci przesyłany jest blok danych liczbowych. Poszczególne dane zapisane są w postaci dziesiętnej i oddzielone przecinkami. Tego rodzaju bloki przesyłane są przez INTEGRATORA do *programu komunikacyjnego*, którego zadaniem jest dostarczenie ich do odpowiednich urządzeń sterujących systemu wytwarzania. W takiej samej postaci bloki te powracają do INTEGRATORA jako potwierdzenia wykonania czynności.

Moduł komunikacyjny 3P, dołączony do każdej aplikacji otwartego układu sterowania, daje do dyspozycji zestaw funkcji zapewniających konwersję odebranych list zamówień oraz bloków danych z postaci opisanego powyżej ciągu znaków do postaci binarnej, oczekiwanej przez adresata. Stosowanie mechanizmów buforowania bloków i wykrywania błędów w przekazywaniu danych oraz wykorzystanie standardu TCP/IP, zapewniają protokołowi 3P duży stopień niezawodności, wymagany w zastosowaniach, do których został przeznaczony.

4. UWAGI KOŃCOWE

Przedstawiony w pracy układ operatywnego sterowania wytwarzaniem, zintegrowany z pakietem do zarządzania produkcją Wonderware® InTrack™, był testowany z wykorzystaniem graficznego symulatora zrestrukturyzowanego centrum produkcyjnego TOR, ponownie uruchamianego obecnie w laboratorium ITMiAP. Istotne jest to, że symulator odbiera i realizuje identyczne polecenia jak rzeczywisty system, a po zakończeniu zleconych czynności w identyczny sposób je potwierdza. Przeprowadzone próby wykazały poprawność programów i właściwe ich współdziałanie. Każde zlecenie oraz przebieg produkcji były rejestrowane w bazie danych programu InTrack.
Istnieją możliwości integracji układu z programami wyższego poziomu zarządzania przedsiębiorstwem, np. SAP R3/R4. Jedną z nich jest wykorzystanie faktu, że InTrack oparty jest na standardowych serwerach relacyjnych baz danych: Microsoft SQL Server lub Oracle. Struktura bazy, z którą on współdziała jest dobrze udokumentowana, toteż nietrudno zbudować interfejs, który udostępni jej zawartość innym aplikacjom. Ponadto zaczynają się pojawiać na rynku gotowe pakiety programów (wprawdzie drogie), umożliwiające takie połączenia. Przykładem może być VisualFlow, posiadający moduł do systemu SAP. Niebawem ma być także oferowany moduł do systemu Baan.

Publikacja powstała w wyniku realizacji projektu badawczego nr 1010/T07/98/15.

LITERATURA

[1] Pierzchała W.: *Coloured object-oriented model of manufacturing system.* Postępy Technologii Maszyn i Urządzeń, Oficyna Wydawnicza Politechniki Rzeszowskiej, vol. 23, Nr 4, Rzeszów 1999.

[2] Pierzchała W.: *Model przepływu produkcji w zautomatyzowanym systemie wytwarzania.* Materiały Konferencji AUTOMATION'99, PIAP, Warszawa 1999, s. 131-138.

[3] Pierzchała W.: *Modernizacja układu sterowania elastyczną linią obróbkową.* Materiały Konferencji AUTOMATION'98, PIAP, Warszawa, marzec 1998, s. 287-294.

[4] Pierzchała W.: *Otwarty układ sterowania zautomatyzowanym wytwarzaniem.* Materiały Konferencji AUTOMATION'98, PIAP, Warszawa, marzec 1998, s. 94-101.

[5] Pierzchała W.: *Planowanie zadań produkcyjnych w oparciu o wirtualny proces wytwarzania* (w niniejszych materiałach).