dr inż. Wiesław Pierzchała
Politechnika Krakowska
Katedra Systemów Wytwarzania

Otwarty układ sterowania zautomatyzowanym wytwarzaniem

Streszczenie: W referacie przedstawiono oryginalną koncepcję układu sterowania zautomatyzowanym wytwarzaniem. Omówiono główne moduły oprogramowania tego układu. Zasadniczą rolę w prezentowanej koncepcji pełni nadrzędny program sterujący, który działa w oparciu o macierzowy model procesu wytwarzania.

Open Control System of Automated Manufacturing

Summary: The paper presents the original concept of automated manufacturing control system. Fundamental modules of control software are described. Key role in presented concept plays the supervisory control program based on real-time processed matrix model of manufacturing process.

1. WPROWADZENIE

System zarządzania produkcją ma strukturę hierarchiczną [4]. *Poziom strategiczny* (najwyższy) odnosi się do stosunkowo długiego okresu (zazwyczaj kilku lat) i obejmuje między innymi określanie asortymentu wyrobów, które będą produkowane, oraz planowanie potrzeb zasobowych. Na *poziomie taktycznym* odbywa się planowanie zadań produkcyjnych dla poszczególnych modułów systemu wytwarzania oraz nadzorowanie przebiegu ich realizacji. Średnioterminowy plan produkcyjny dla zautomatyzowanego systemu wytwarzania określa między innymi orientacyjne terminy uruchamiania oraz liczebność serii wyrobów, które będą równocześnie produkowane w rozpatrywanym horyzoncie czasu. Plan ten powinien zapewniać zrównoważone i możliwie wysokie wskaźniki obciążenia maszyn (obrabiarek). Wymaganie to niełatwo spełnić, ponieważ w fazie planowania zadań raczej unika się sztywnego przydziału operacji do stanowisk. Zmierza się do projektowania procesów technologicznych z alternatywnymi marszrutami lub o strukturze sieciowej (*net-like structure*). W rezultacie nie wiadomo, które operacje będą wykonywane na poszczególnych stanowiskach. Ostateczny przydział dokonywany jest bezpośrednio przed rozpoczęciem operacji, stosownie do aktualnego stanu systemu. Stąd możliwe jest jedynie oszacowanie średnich wskaźników obciążenia maszyn. Sterowanie produkcją na *poziomie operacyjnym* realizuje cztery główne cele: bezkolizyjną współpracę urządzeń, unikanie zastojów, zapewnienie wysokiej efektywności systemu wytwarzania oraz monitorowanie funkcji i usuwanie skutków zakłóceń. Najniższy poziom, *poziom sterowania urządzeniami* zazwyczaj wymaga jedynie przygotowania niezawodnego oprogramowania sterowników CNC i PLC.
W referacie jest prezentowana oryginalna koncepcja sterowania wytwarzaniem w systemach zautomatyzowanych, opracowana i zweryfikowana w Katedrze Systemów Wytwarzania Politechniki Krakowskiej. Odnosi się ona do dwóch najniższych poziomów omówionej struktury systemu zarządzania wytwarzaniem.

2. STRUKTURA OTWARTEGO UKŁADU STEROWANIA WYTWARZANIEM

Zadania oprogramowania komputerowego układu sterowania wytwarzaniem podzielono pomiędzy szereg aplikacji, które mogą pracować na różnych komputerach (choć nie jest to konieczne) i wymieniać pomiędzy sobą informacje za pośrednictwem sieci *Ethernet* (protokół *TCP/IP*). Wyróżniono cztery podstawowe jego poziomy, z których pierwsze dwa mają charakter uniwersalny, na trzecim pracują aplikacje dedykowane dla konkretnego systemu wytwarzania, natomiast poziom czwarty - najniższy - reprezentuje sterowanie lokalne poszczególnymi urządzeniami i modułami:

Poziom I: Nadrzędny program sterujący (**WinESP/MM**), działający w oparciu o model macierzowy realizowanego procesu wytwarzania, przetwarzany współbieżnie z jego przebiegiem. Może on sterować dowolnym systemem, dla którego opracowano dane wymagane przez ten model. Program wysyła polecenia wykonania czynności (w formie ciągów znaków będących ich nazwami) i oczekuje na potwierdzenia ich zakończenia.

Poziom II: Program zarządzający systemem (**System Server**). Jego zadaniem jest zapewnienie połączenia nadrzędnego programu sterującego z III poziomem układu sterowania. W tym celu przetwarza on nazwy czynności na szereg poleceń wysyłanych do poszczególnych elementów systemu wytwarzania oraz identyfikuje odbierane stamtąd potwierdzenia i przesyła je do programu nadrzędnego. Ponadto program zarządzający systemem zapewnia współpracę z aplikacjami pomocniczymi pracującymi na poziomie III. Jest on uniwersalny a wszystkie swoje funkcje wykonuje w oparciu o utworzony przez użytkownika program napisany w specjalnym języku skryptowym.

Poziom III: Aplikacje dedykowane, specyficzne dla konkretnego systemu wytwarzania, np.:

- Program komunikacyjny zapewniający przekazywanie danych do poszczególnych elementów systemu produkcyjnego. W przypadku zastosowania lokalnej sieci przemysłowej stanowi on połączenie (ang. *gateway*) pomiędzy nią a siecią *Ethernet* wykorzystywaną na wyższych poziomach.
- Programy obsługujące poszczególne moduły systemu wytwarzania (np.: magazyn, podsystem narzędziowy, zarządzanie przyrządami).
- Aplikacje obsługujące inne funkcje układu sterowania (np. przechowywanie i przesyłanie do sterowników CNC i PLC żądanych przez nie programów).

Poziom IV: Lokalne sterowniki urządzeń i modułów systemu wytwarzania (w tym PLC i CNC ze stosownym oprogramowaniem).

Poszczególne poziomy i aplikacje dla nich przeznaczone zostaną krótko omówione, za wyjątkiem poziomu IV. Ze względu na dość powszechne obecnie stosowanie na tym poziomie sterowników programowalnych (CNC i PLC), jego zakres merytoryczny ogranicza się do wyspecyfikowania typów i konfiguracji sterowników oraz ich oprogramowania.

3. GŁÓWNY PROGRAM STERUJĄCY PRZEPŁYWEM PRODUKCJI (WINESP/MM) – POZIOM I

Najważniejszą cechą, wyróżniającą opisywane podejście do sterowania systemami zautomatyzowanymi, jest stosowanie modelu macierzowego (MM) jako wzorca procesu wytwarzania. Sekwencja poleceń wykonywania określonych czynności jest wynikiem przetwarzania tego modelu w czasie rzeczywistym, współbieżnie z realizowanym procesem. Model tworzony jest na drodze dyskretyzacji procesu produkcyjnego, to znaczy wyodrębnienia (arbitralnego)

czynności elementarnych realizowanych w systemie, oraz obiektów, które w nich uczestniczą. Na każdym etapie przetwarzania modelu specyfikowane są wszystkie te czynności, które w danej chwili są *możliwe do wykonania*. Jedynym kryterium branym pod uwagę jest dostępność wszystkich niezbędnych do wykonania czynności obiektów w wymaganej liczbie. Z tego względu fundamentalne znaczenie ma *macierz stanu* $[S_{jk}]$. Kolumny tej macierzy odpowiadają poszczególnym obiektom systemu (k), natomiast jej wiersze – czynnościom elementarnym (j). Wartość każdego elementu macierzy określa *nadmiar* obiektów, którym odpowiada dana kolumna, w stosunku do liczby niezbędnej do wykonania czynności, której odpowiada dany wiersz. Oznacza to, że jeśli odpowiadający określonemu obiektowi i określonej czynności element macierzy przyjmuje wartość ujemną, czynność ta nie może być wykonana ze względu na niedostępność (niedobór) danego obiektu. Należy podkreślić, iż fakt *dostępności* obiektu k dla czynności j (wtedy $S_{jk} \geq 0$) oznacza jedynie, że obiekt ten może w niej wziąć udział, ponieważ nie uczestniczy w żadnej innej czynności. I odwrotnie: obiekt *niedostępny* to taki, który w danej chwili bierze udział w czynności innej niż rozważana.

Macierz stanu, przekształcana algorytmicznie po każdym rozpoczęciu lub zakończeniu czynności, pozwala na każdym etapie sterowania wyznaczyć podzbiór wszystkich czynności *możliwych do wykonania* (tzn. takich, dla których wszystkie elementy w odpowiadających im wierszach macierzy stanu $[S_{jk}]$ są nieujemne), z pozostawieniem najszerszej swobody wyboru jednej z nich – tej, która ma być rozpoczęta. Otwiera to możliwość przydzielania operacji do stanowisk i określania porządku ich wykonywania na bieżąco, w trakcie trwania procesu produkcyjnego, gdy znany jest aktualny stan systemu. Warto tutaj podkreślić, że model jest otwarty na współdziałanie z różnymi algorytmami z tego zakresu, rozważanymi w teorii szeregowania zadań, m. in. heurystycznymi.

Model macierzowy dyskretnego procesu produkcyjnego i algorytm jego przetwarzania zostały dokładnie przedstawione w monografii [2].

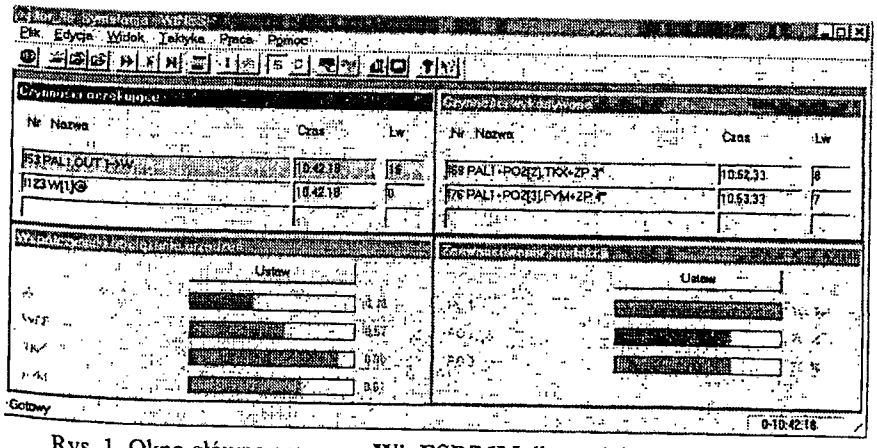
Oryginalność opisywanego rozwiązania polega na jednolitym (w sensie modelowym) traktowaniu symulacji i sterowania. Ta jednolitość jest tym bardziej wyrazista, że w obu przypadkach stosowany jest ten sam program **WinESP/MM** (*Visual C++*, środowisko *Windows NT*) przełączany do trybu symulacji lub sterowania (rys. 1). W przypadku symulacji kolejno są rozpoczynane czynności wybierane według zastosowanej heurystyki ze zbioru czynności możliwych, aż do wyczerpania tego zbioru. Wtedy następuje zakończenie czynności, której przewidywany termin zakończenia jest najwcześniejszy, i przesunięcie czasu bieżącego. W przypadku sterowania systemem wytwarzania przekształcenia modelu są identyczne, tyle że wymuszają je zdarzenia rzeczywiste. Polecenia rozpoczynania czynności możliwych są wysyłane do sterowników poszczególnych urządzeń, po czym program oczekuje na potwierdzenia ich zakończenia. Model przekształcany jest tutaj współbieżnie ze sterowanym procesem.

Takie podejście ma kilka istotnych zalet, z których najważniejsze to:

1. Możliwe jest dowolnie długie eksperymentowanie na symulowanym systemie produkcyjnym, z uwzględnieniem wprowadzanych modyfikacji w zakresie jego struktury i sposobu działania (kolejności realizowanych czynności). Po stwierdzeniu poprawnego i zadowalającego (w sensie przyjętych kryteriów efektywności) działania systemu, ten sam model można bezpośrednio przenieść do komputerowego układu sterowania.
2. Istnieje możliwość uruchomienia pracującego w tle procesu wirtualnego, symulującego działanie systemu produkcyjnego począwszy od jego aktualnego stanu. Proces taki należy rozumieć jako mogący zaistnieć przebieg zdarzeń w rzeczywistym systemie. Uruchamiając proces wirtualny można sprawdzić np. skuteczność, w aktualnych warunkach, stosowanej heurystyki wyboru czynności do wykonania.
3. Istnieje możliwość szkolenia personelu i uczenia go zasad działania systemu na jego modelu symulacyjnym.

4. Można oczekiwać, że jeżeli czasy trwania rzeczywistych czynności nie będą istotnie odbiegały od czasów przyjętych w eksperymentach symulacyjnych, to system wytwarzania będzie działał dokładnie tak, jak spodziewa się projektant.
5. Istnieje możliwość stopniowego uruchamiania lub modernizowania systemu wytwarzania przez włączanie kolejnych jego modułów i obserwowanie ich współpracy z innymi, nieistniejącymi modułami, których działanie jest symulowane.

Prawidłowo zaprojektowany i przygotowany proces w zautomatyzowanym systemie wytwarzania przebiega zgodnie z poleceniami realizacji poszczególnych czynności, wydawanymi przez układ sterowania. Jeżeli zawsze byłyby spełnione wszystkie warunki realizowalności czynności zlecanych do wykonania, to nie musiałyby do niego docierać jakiekolwiek dodatkowe informacje. W istocie, moduły pomocnicze (np. narzędziowy, transportowy), odpowiedzialne za spełnianie tych warunków, ze względu na swą służebną rolę wobec głównych modułów produkcyjnych (np. stacji obróbkowych), powinny nadążać. Wtedy, gdy nie jest to możliwe, pojawia się informacja istotna dla sterowania. To z pozoru oczywiste spostrzeżenie prowadzi do stwierdzenia, że skutkiem uwzględniania wpływu informacji wewnątrzsystemowych i pochodzących z otoczenia może być jedynie blokowanie wykonywania niektórych czynności, dla których warunki realizowalności, najczęściej tylko okresowo, nie są spełnione. Informacje o czynnikach mogących mieć wpływ na przebieg procesu produkcyjnego (takich jak np. brak potrzebnego narzędzia, niedostępność programu NC, awaria obrabiarki, wadliwe wyroby) są uwzględniane w omawianym układzie sterowania za pośrednictwem tzw. *macierzy warunków lokalnych* $[L_{jk}]$ [1] uwzględnianej przez algorytm przetwarzania modelu macierzowego sterowanego systemu. Ujemna wartość elementu L_{jk} tej macierzy oznacza, że obiekt k, któremu odpowiada dana kolumna, z jakiegokolwiek powodu nie jest gotowy do wykonania czynności j, której odpowiada dany wiersz, natomiast wartość 0 – że jest gotowy. Czynność, która może zostać wykonana ze względu na gotowość wszystkich niezbędnych obiektów, nazywana jest *realizowalną*. Dzięki macierzy warunków lokalnych układ sterowania może uwzględniać praktycznie każdą informację o okresowej nierealizowalności niektórych czynności. Liczba i miejsce umieszczenia w macierzy wartości ujemnych zależą wyłącznie od aplikacji odpowiedzialnych za dostarczanie tych informacji. Wartości elementów macierzy warunków lokalnych $[L_{jk}]$ stanowią w pewnym sensie iloczyn logiczny czynników, od których uzależniona jest realizowalność czynności: jeżeli wszystkie warunki są spełnione, element przyjmuje wartość 0, jeśli co najmniej jeden warunek nie jest spełniony, element ma wartość –1. Liczba tych czynników dla każdej czynności może być inna.



Rys. 1. Okno główne programu **WinESP/MM** dla modelu systemu TOR [5]

4. SERWER SYSTEMOWY – POZIOM II

Program zarządzający systemem (**System Server** – serwer systemowy) pełni funkcję interfejsu pomiędzy programem nadrzędnym, przetwarzającym macierzowy model realizowanego procesu wytwarzania, a sterowanym systemem produkcyjnym. Ze względu na jego uniwersalność, rola programu nadrzędnego w sterowaniu sprowadza się do wysłania nazwy czynności, która ma się rozpocząć. Nazwa ta jest charakterystyczna dla modelu macierzowego i nie może być rozumiana w bezpośredni sposób przez żadnego z realizatorów zlecanych czynności. Rolą serwera systemowego jest przetłumaczenie nazwy czynności na polecenia „zrozumiałe" dla sterowników urządzeń pracujących w systemie. Polecenia te mają postać bloków danych o określonym przez projektanta formacie.

Podobnie jak nadrzędny program sterujący, tak i program zarządzający systemem jest uniwersalny, tzn. może być wykorzystany do sterowania każdym systemem. Zrealizowano to przez opracowanie specjalnego języka skryptowego umożliwiającego programowanie serwera systemowego. Dla każdego systemu można napisać program skrypt zawierający wszystkie działania, jakie muszą być wykonane dla poszczególnych czynności.

Każdy program skryptowy składa się z siedmiu części [3]. Są to:

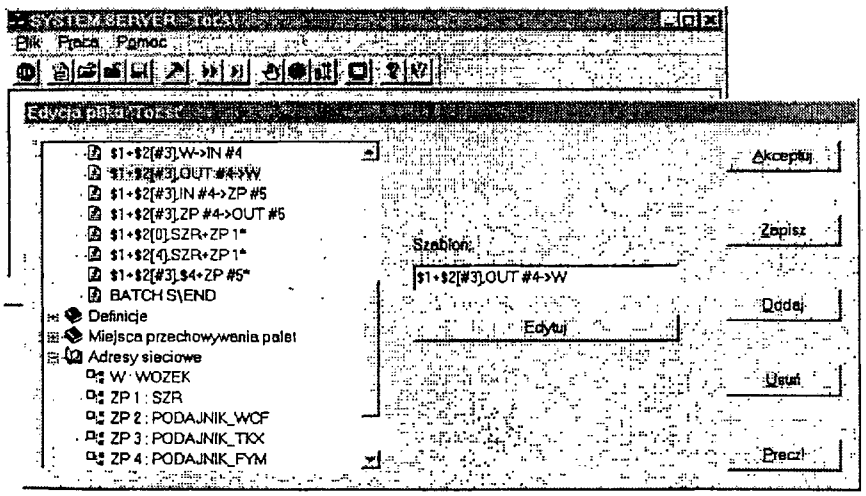
1. Definicja formatu bloku danych do wymiany informacji w układzie sterowania.
2. Skrypty opisujące operacje, które ma wykonać serwer systemowy po otrzymaniu polecenia wykonania czynności. Każdy skrypt składa się z dwóch części. Pierwsza zawiera instrukcje przeznaczone do wykonania po odebraniu polecenia z programu nadrzędnego, druga – instrukcje wykonywane po odebraniu potwierdzenia wykonania czynności. W języku skryptowym możliwe jest wykorzystanie zmiennych tekstowych i liczbowych (całkowitych) oraz 29 funkcji. Są wśród nich m.in. funkcje obsługi bloku danych, magazynu, transmisji programów NC i instrukcje warunkowe.

   W celu usprawnienia programowania wprowadzono możliwość zastosowania tzw. *szablonów czynności*. Jeżeli w modelu zdefiniowanych jest kilka podobnych czynności (np. analogiczne czynności obróbkowe dla kilku różnych przedmiotów), wygodnie jest zdefiniować dla nich *szablon* i napisać dla niego jeden skrypt. W szablonach można stosować *symbole ogólne* zastępujące dowolne ciągi znaków lub liczby (*$n* oznacza *n*-ty symbol tekstowy, *#n* oznacza *n*-ty symbol liczbowy). Na przykład, do szablonu `$1+$2[#3],$4+ZP #5*` pasuje między innymi czynność PAL1+PO1[2],TKX+ZP 3* (oznaczająca obróbkę - symbol * - przedmiotów *PO1* w stanie *2*, przemieszczanych na palecie *PAL1*, na stacji tokarskiej *TKX*, z udziałem zmieniacza palet *ZP 3*) i w skrypcie dla tego szablonu podczas realizowania tej czynności zamiast symbolu *$1* wstawione zostanie *PAL1*, zamiast *$2* - *PO1*, zamiast *#3* - *2* itd.
3. Skrypty specjalne opisujące operacje, które ma wykonać serwer systemowy:
   a) po wystąpieniu błędu,
   b) po otrzymaniu polecenia od jednego ze sterowników,
   c) podczas obsługi transmisji programu numerycznego.
4. Definicje stałych tekstowych i numerycznych wykorzystywanych w skryptach.
5. Definicje tymczasowych miejsc przechowywania palet. W systemie wytwarzania często istnieją miejsca poza magazynem, w których mogą być przechowywane palety. Takimi miejscami mogą być np. zmieniacze palet, wózek lub stoły. Zdefiniowanie ich jest konieczne, w przeciwnym bowiem razie informacja o położeniu i zawartości palety ginęłaby po umieszczeniu jej w takim miejscu.
6. Definicje identyfikatorów wykorzystywanych do komunikacji z systemem wytwarzania. Każdy blok danych wysyłany do systemu wytwarzania jest uzupełniony o tekstowy adres

odbiorcy bloku. Musi to być adres rozumiany przez sieciowy *program komunikacyjny* pracujący na III poziomie sterowania. Program pozwala na zdefiniowanie dowolnej liczby adresów i nadanie każdemu z nich identyfikatora używanego w skryptach.

7. Definicje komunikatów błędów, wyświetlanych na ekranie w przypadku odebrania przez serwer systemowy informacji o wystąpieniu błędu.

Wszystkie wymienione elementy składają się na komplet danych o sterowanym systemie. Jest on zapisywany na dysku jako plik z rozszerzeniem **.st* i ma format tekstowy. Można go więc edytować za pomocą dowolnego edytora tekstu ASCII, ale jest to niewskazane. Plik ten ma bowiem ściśle określony format i poprawiając go ręcznie bardzo łatwo o pomyłki, co może prowadzić do niepoprawnej pracy programu. Zalecane jest wykorzystywanie do tego celu wbudowanego do programu **System Server** edytora. Wygląd okna edytora jednego z szablonów czynności w przykładowym programie skryptowym przedstawiony jest na rys. 2.



Rys. 3. Ekran edytora programu **System Server** obsługującego system TOR [5]

Program **System Server** (*Visual C++*, środowisko *Windows NT*) może współpracować z programami obsługującymi niektóre moduły systemu wytwarzania i funkcje układu sterowania. Przykładami takich programów są: aplikacja zarządzająca magazynem oraz aplikacja przesyłająca do sterowników obrabiarek żądane programy numeryczne.

5. APLIKACJE DEDYKOWANE – POZIOM III

Aby polecenia generowane przez program zarządzający systemem mogły dotrzeć do właściwych sterowników, niezbędny jest *program komunikacyjny*. Jego zadaniem jest odbieranie bloków danych wysyłanych przez serwer systemowy za pośrednictwem sieci lokalnej wg protokołu *TCP/IP* i przesyłanie ich do odpowiednich sterowników urządzeń, w sposób właściwy dla zastosowanego w systemie standardu komunikacyjnego (np. lokalnej sieci przemysłowej, interfejsu szeregowego), a także odbieranie bloków danych przychodzących z systemu wytwarzania i odsyłanie ich do serwera systemowego.

Program obsługi magazynu jest aplikacją niezbędną do pracy serwera systemowego. Musi on zapewniać realizację poleceń skryptowych dotyczących obsługi magazynu, wykonując rozkazy i odpowiadając na zapytania serwera systemowego. Serwer pyta np. o to, jaka paleta jest umieszczona w danym miejscu, o to, które miejsce jest puste, lub gdzie znajduje się paleta określonego typu. Serwer może także polecić umieszczenie określonej palety w

zadanym miejscu, bądź też jej usunięcie. Dla każdego systemu obsługiwanego przez serwer systemowy należy stworzyć aplikację zarządzającą magazynem, zaprojektowaną dla konkretnego rozwiązania magazynu. Oczywiście musi ona być dostosowana do komunikacji z serwerem systemowym poprzez sieć lokalną w standardzie *TCP/IP* (*Windows Sockets*), rozpoznawać jego rozkazy i zapytania oraz prawidłowo na nie odpowiadać.

Program obsługi bazy programów numerycznych nie jest aplikacją niezbędną do pracy serwera systemowego. Pełni on funkcję pomocniczą, zapewniając możliwość przesłania do sterownika (na jego żądanie) programu numerycznego o określonym numerze. Z racji tego, że w systemie mogą być stosowane różne sterowniki, program musi być dostosowany do konkretnego sterownika CNC (lub ewentualnie kilku sterowników) i jego formatu transmisji programów. Transmisja jest organizowana przez serwer systemowy. Obsługę żądań sterowników dotyczących przesłania programu projektant musi umieścić w skryptach specjalnych serwera. Żądanie takie jest przekazywane (za pośrednictwem sieci) programowi obsługi bazy, którego zadaniem jest wyszukanie właściwego programu, w razie konieczności podzielenie go na mniejsze części i przesłanie sterownikowi za pośrednictwem serwera systemowego.

6. OTWARTOŚĆ UKŁADU STEROWANIA

W nowoczesnych koncepcjach układów sterowania dla zautomatyzowanych (w tym elastycznych) systemów wytwarzania, mocno akcentowana jest cecha *otwartości* kierowana jako ważny postulat pod adresem projektantów tych układów [6]. Otwartość omawianego tutaj układu, którego główny moduł oprogramowania (**WinESP/MM**) oparty jest na modelu macierzowym, przejawia się m. in. w następujących jego cechach:

1. Istnieje możliwość łatwego wprowadzania i testowania symulacyjnego zmian strukturalnych w modelu macierzowym (a więc pośrednio w układzie sterowania), obejmujących:
   - dodawanie, usuwanie, zwielokrotnianie liczby i funkcji obiektów,
   - dodawanie lub usuwanie wybranych czynności,
   - łączenie modeli systemów.
2. Istnieje możliwość wprowadzania zmian asortymentu produkowanych wyrobów.
3. Warunki realizowalności czynności są przekształcane do postaci macierzy warunków lokalnych $[L_{jk}]$ i wprowadzane do układu sterowania poprzez aktualizację jej zawartości. Umożliwia to uwzględnienie wpływu na przebieg procesu wytwarzania rozmaitych czynników i dodatkowych wymagań, zarówno wewnętrznych, jak i pochodzących z jego otoczenia. Macierz warunków lokalnych integruje wszelkie informacje ważne dla podejmowania decyzji odnośnie do sterowania. Nie wprowadza żadnych ograniczeń co do ich rodzaju, treści ani źródła pochodzenia. Pozwala także uwzględnić nowe wymagania i warunki, których nie przewidziano, lub które pominięto świadomie (np. w celu uproszczenia procedur uruchamiania i testowania) w projekcie układu sterowania.
4. Rozwój oprogramowania sterującego, w każdej jego części, jest oczywiście możliwy. Dopóki nie zostanie zaniechana koncepcja sterowania systemem zautomatyzowanym znamienna zastosowaniem jego modelu macierzowego, główna część (jądro) oprogramowania sterującego i przekształcenia w niej wykonywane nie ulegną zmianie. W szczególności, w odniesieniu do modułów monitorowania warunków realizowalności czynności, stopniowy rozwój oprogramowania może mieć wpływ jedynie na ich liczbę i sposób uzyskiwania informacji, na podstawie których modyfikowana jest macierz warunków lokalnych, analizowana potem w niezmienny sposób. Innym, niejako naturalnym kierunkiem rozwoju poprawnie działającego oprogramowania sterującego jest wzbogacanie interfejsu użytkownika i czynienia go wygodniejszym (prosta obsługa programów, wizualizacja, animacja, przekazywanie obrazu procesu wytwarzania do stanowiska dyspozytora itp.).

5. Czteropoziomowa struktura układu sterowania w przejrzysty sposób rozgranicza funkcje poszczególnych jego modułów.

   Główny program **WinESP/MM** (poziom I) pozostaje niezależny od struktury i sposobu działania sterowanego systemu.

   Program zarządzający systemem – **System Server** (poziom II) – „dopasowuje" program główny do konkretnego systemu sterowanego, komunikując się z innymi aplikacjami za pośrednictwem sieci *Ethernet* według standardowego protokołu *TCP/IP*. Jest on na tyle ogólny i konfigurowalny, że po zaprogramowaniu go za pomocą specjalnie opracowanego języka skryptowego może być zastosowany w układach sterowania różnymi systemami. Tym niemniej, **System Server** może być zastąpiony innym programem (np. wykorzystującym inny protokół komunikacyjny), realizującym analogiczne funkcje. Na podstawie nazw czynności elementarnych buduje on bloki danych (ta funkcja wyróżnia poziom II) dla sterowników poszczególnych urządzeń, jednakże działa w oderwaniu od sposobu, w jaki zostaną one tam wprowadzone.

   Poziom III stanowią aplikacje ściśle powiązane z konkretnym systemem wytwarzania. Takie moduły oprogramowania muszą istnieć zawsze i z natury swej mogą, a często muszą być wymieniane (np. w przypadku zmiany protokołu komunikacyjnego). Tworzą one najniższy poziom dopasowania układu sterowania systemem zautomatyzowanym do warunków lokalnych i nierozerwalnie wiążą się z ich specyfiką.

7. ZAKOŃCZENIE

Przedstawiona struktura oprogramowania sterującego zautomatyzowanym systemem wytwarzania pozwala na wyraźne oddzielenie treści i merytorycznego znaczenia informacji przekazywanych pomiędzy poszczególnymi jego modułami od sposobu ich przesyłania.

W uzupełnieniu stwierdzeń odnośnie do jego otwartości należy podkreślić, że model macierzowy procesu wytwarzania, odgrywający tutaj fundamentalną rolę, nie posiada żadnych ograniczeń stosowalności do specyficznych systemów produkcyjnych, a forma zapisu modeli różnych procesów jest jednolita. Wymagana jest jedynie możliwość ich dyskretyzacji.

LITERATURA

[1] J. Cyklis, W. Pierzchała, J. Zając: Integration of the CIM information on the level of FMS control. Automation und Meßtechnik 4/1997, Springer-Verlag, Wien, s. 169-175.

[2] J. Cyklis, W. Pierzchała: Modelowanie procesów dyskretnych w elastycznych systemach produkcyjnych. Zeszyty Naukowe Politechniki Krakowskiej, Mechanika z. 77, Monografia nr 3, Kraków 1995.

[3] J. Cyklis, W. Pierzchała, J. Zając i in.: Opracowanie dyskretnego modelu przepływu informacji w systemie sterowania produkcją zautomatyzowaną. Raport końcowy z realizacji projektu badawczego Nr 7 T07D 026 08, ITMiAP, Politechnika Krakowska, Kraków 1997.

[4] Pierzchała W.: Hierarchical Approach to Production Control in CIM. Publ. Univ. of Miskolc, Series C. Mechanical Engineering. Vol. 45. (1995) No. 1, s. 241-248.

[5] Pierzchała W.: Modernizacja układu sterowania elastyczną linią obróbkową. Materiały Konferencji AUTOMATION'98.

[6] Szafarczyk M.: „Open Architecture Controllers and Automatic Supervision in Manufacturing". Proc. of the CIRP Seminars. Manufacturing Systems, Vol. 25 (1996) s. 37÷41.