MODELOWANIE I SYMULACJA SYSTEMÓW KOLEJKOWYCH W ŚRODOWISKU FLEXSIM – STUDIUM PRZYPADKU

MONIKA KLAŚ, KRZYSZTOF JURCZYK

Streszczenie

W pracy zaproponowano model symulacyjny systemu transportowego dwóch rodzajów komponentów na dwie stacje przetwarzające przez jeden podajnik automatyczny. Model został utworzony w programie FlexSim 3d Simulation Software. W kolejnych akapitach artykułu szczegółowo omówiono zasadę działania utworzonego modelu oraz przedstawiono wyniki testów numerycznych, jakie przeprowadzono w celu identyfikacji wąskiego gardła analizowanego systemu.

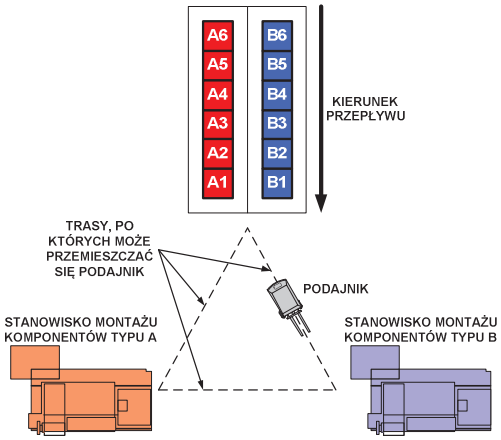
Słowa kluczowe: modelowanie i symulacja, specjalistyczne pakiety obliczeniowe, systemy kolejkowe, testowanie usprawnień

Wprowadzenie

Podstawowym elementem dynamicznie zmieniających się systemów produkcyjnych są tworzące się kolejki. Zrozumienie zasady ich funkcjonowania i zachowania jest niezbędnie do rozwiązywania problemów i podejmowania decyzji dotyczących takich systemów, jak również do tworzenia i analizowania reprezentujących je modeli [1]. Formowane kolejki umożliwiają urządzeniom i/lub instytucjom wykonywanie zadań lub świadczenie oferowanych usług w uporządkowany sposób [2]. Literatura związana z modelowaniem systemów kolejkowych jest bardzo bogata i obszerna [3–10]. W niniejszym artykule przedstawiono sposób wykorzystania profesjonalnego oprogramowania do symulacji zdarzeń dyskretnych – FlexSim 3D Simulation Software – w modelowaniu złożonych systemów kolejkowych.

1. Założenia do budowy modelu symulacyjnego

W artykule analizie poddano system kolejkowy dla montażu dwóch komponentów, składający się z wejścia do systemu, kolejki rozdzielającej zadania na poszczególne stanowiska, buforów przystanowiskowych, dwóch stacji obróbczych, podajnika automatycznego realizującego transport pomiędzy buforami i stacjami obróbczymi oraz wyjścia z systemu. Jeden podajnik automatyczny obsługuje dwie stacje obróbki. Podajnik może transportować maksymalnie jedną sztukę danego komponentu. Stanowiska montażowe są dedykowane dla dwóch różnych rodzajów komponentów, dla których przyjęto oznaczenia A oraz B. Znane są czasy pomiędzy kolejnymi przybyciami komponentów A i B do systemu, czasy montażu elementów na dedykowanych stanowiskach obróbczych, czasy transportu komponentów z buforów na stanowiska oraz czasy transportu komponentów ze stanowisk na wyjście z systemu. Na rysunku 1 zaprezentowano schemat analizowanego systemu kolejkowego.

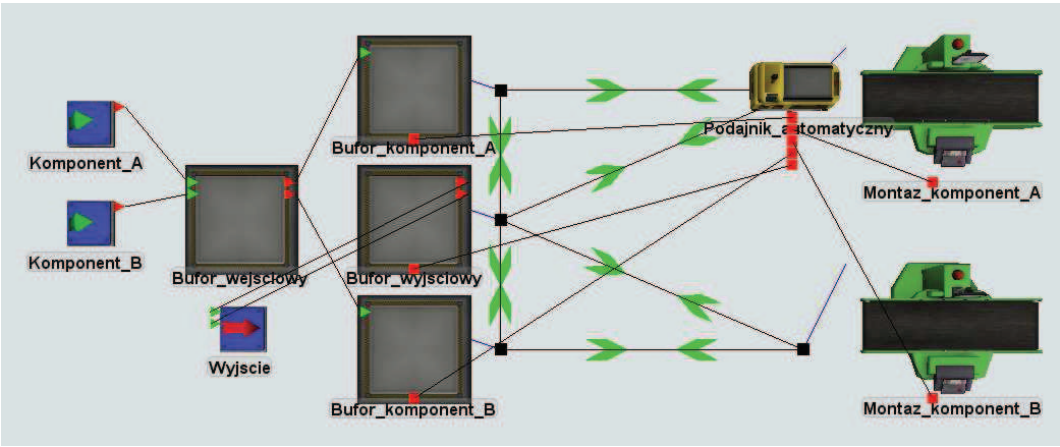


Rysunek 1. Schemat analizowanego systemu kolejkowego

Celem niniejszego artykułu jest analiza takich parametrów jak średnie czasy oczekiwania na transport przez poszczególne komponenty $\overline{T_{oczA}}$ oraz $\overline{T_{oczB}}$, stopień wykorzystania stanowisk obróbczych U_{rA} oraz U_{rB}, stopień zblokowania stanowisk obróbczych U_{bA} oraz U_{bB}, liczba kilometrów przebywanych przez podajnik automatyczny w ciągu jednej zmiany roboczej DT oraz całkowite obciążenie podajnika automatycznego U_t. Na potrzeby modelu i powtarzalności wyników symulacji przyjęto, że stanowiska pracują w systemie jednozmianowym przez 8 godzin.

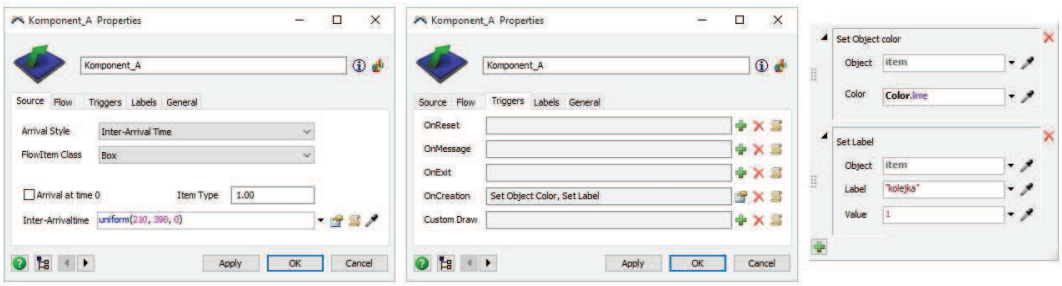
2. Model symulacyjny utworzony w programie FlexSim 3D Simulation Software

Model symulacyjny systemu kolejkowego opisanego w poprzednim rozdziale został wykonany w programie FlexSim 3D Simulation Software. Na rysunku 2 zaprezentowano widok modelu.



Rysunek 2. Model symulacyjny analizowanego systemu kolejkowego w programie FlexSim

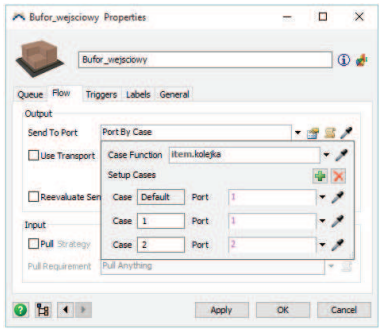
Wejście do modelu stanowią dwa źródła (obiekty typu „Source") – „Komponent_A" oraz „Komponent_B", na których generowane są elementy przepływu zgodnie z zadanym rozkładem czasu pomiędzy kolejnymi przybyciami. Komponenty A i B pojawiają się na wejściu systemu zgodnie z rozkładem jednostajnym, odpowiednio z częstotliwością od 210 do 390 sekund i od 120 do 360 sekund. Wprowadzenie tych wartości jest możliwe w ustawieniach obiektów „Komponent_A" oraz „Komponent_B" co pokazano na rysunku 3 – w polu Inter-Arrivaltime wprowadzamy odpowiednio zapisy `uniform(210, 390, 0)` oraz `uniform(120, 360, 0)`.



Rysunek 3. Ustawienia obiektu „Komponent_A"

Wygenerowanym elementom nadawany jest kolor za pomocą polecenia „Set Object Color" dostępnego w zakładce Triggers w sekcji „On Creation". Polecenie to pozwala na wskazanie jaki kolor powinny przyjmować nowo wygenerowane elementy przepływu. Dodatkowo każdy utworzony komponent otrzymuje etykietę (Label) o nazwie „kolejka" – polecenie „Set Label" Wartość tej etykiety pozwala na identyfikację poszczególnych elementów w systemie – jako komponenty typu A będą oznaczane te, dla których wartość etykiety „kolejka" będzie równa 1, natomiast komponenty typu B te, dla których ta wartość będzie równa 2.

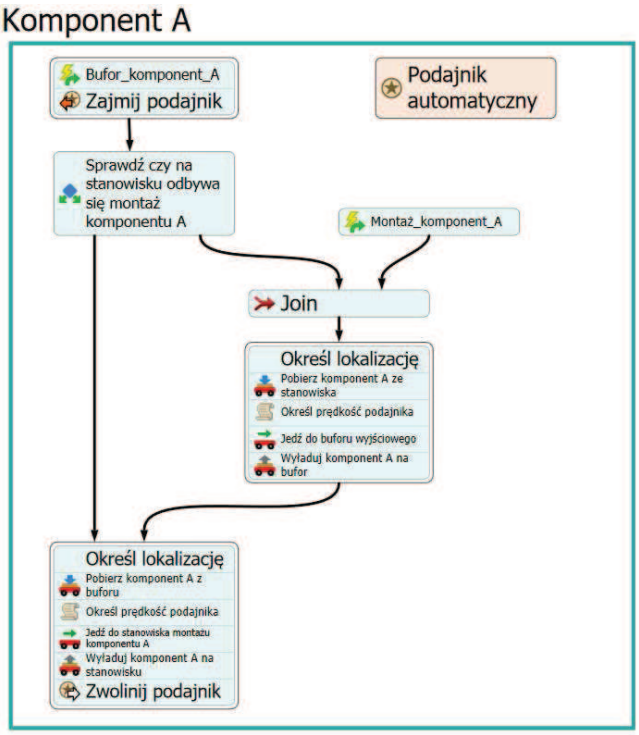
Wygenerowane elementy trafiają na wspólną kolejkę „Bufor_wejsciowy" (obiekt typu „Queue") o maksymalnej pojemności 1000 sztuk. Jest to element przejściowy, który pozwala na wysyłkę komponentów na dedykowane jednoelementowe bufory – „Bufor_komponent_A" oraz „Bufor_komponent_B". Elementy są wysyłane do odpowiednich buforów na podstawie wartości przechowywanej w etykiecie „kolejka". W tym celu w zakładce „Flow", w sekcji „Send To Port" wybrano funkcję „Port By Case". Dokładny zapis ustawień na buforze wejściowym przedstawiono na rysunku 4.



Rysunek 4. Ustawienia logiki przepływu obiektu „Bufor_wejsciowy"

Jednoelementowe bufory „Bufor_komponent_A" oraz „Bufor_komponent_B" to miejsca, z których podajnik automatyczny odpiera komponenty i zawozi na odpowiednie stanowiska montażowe – odpowiednio „Montaz_komponent_A" oraz „Montaz_komponent_B". Czasy przetwarzania komponentów na poszczególnych stanowiskach zadane są rozkładem jednostajnym. Czas montażu komponentu A mieści się w granicach od 120 do 360 sekund, natomiast czas montażu komponentu B wynosi od 138 do 318 sekund. Po zakończeniu procesu montażu wózek odbiera wyrób gotowy ze stanowiska montażowego i zawozi go na „Bufor_wyjściowy", skąd wyroby trafiają bezpośrednio do wyjścia z systemu – „Wyjscie" (obiekt typu „Sink"). Należy zaznaczyć, że jeżeli do transportu pomiędzy poszczególnymi elementami analizowanego systemu kolejkowego nie jest wykorzystywany podajnik automatyczny, to czas przejścia wynosi zero sekund, co oznacza, że elementy są od razu przekazywane na kolejne stadium procesu (jeżeli nie wystąpią inne ograniczenia związane na przykład z maksymalną pojemnością buforów).

Logika pracy podajnika automatycznego została opisana za pomocą technologii „Process Flow", umożliwiającej budowę modelu symulacyjnego w postaci schematu blokowego. Dzięki „Process Flow" można w prosty sposób adaptować działanie modelu 3D do bieżących warunków, co daje większą elastyczność. Na rysunku 5 przestawiono schemat logiki działania podajnika automatycznego dla transportu komponentu A zbudowany za pomocą „Process Flow". Analogiczny schemat logiki działania podajnika automatycznego opracowano dla transportu komponentu B.



Rysunek 5. Schemat logiki działania podajnika automatycznego dla transportu komponentu A zbudowany za pomocą „Process Flow"

Podstawowym, elementarnym składnikiem „Process Flow" jest token, czyli jednostka, która przechodzi przez kolejne etapy schematu blokowego (activities) podczas przebiegu symulacji. Funkcje tokenów są bardzo podobne do funkcji realizowanych przez elementy przepływu w standardowym modelu symulacyjnym – tak, jak element przepływu przechodzący od źródła do kolejki, procesora, itd., token przechodzi pomiędzy kolejnymi działaniami w schemacie blokowym. W przeciwieństwie do tradycyjnego elementu przepływu, token nie musi jednak reprezentować rzeczywistego, fizycznie istniejącego obiektu przechodzącego przez system. Tokeny mogą być abstrakcyjne, co oznacza, że mogą reprezentować wszystko, czego potrzebuje użytkownik modelu. Często są one logicznie powiązane z obiektami istniejącymi fizycznie w modelu (również z elementami przepływu), jednak nie stanowi to konieczności i jest w całości zależne od modelującego. Na najbardziej podstawowym poziomie, token to tylko nośnik danych, który przechodzi przez kolejne etapy schematu „Process Flow".

Logika pracy podajnika automatycznego została oparta na elementach z biblioteki obiektów „Process Flow", które szerzej opisano w kolejnych podrozdziałach niniejszej pracy.

2.1. Bloki typu „Resource", „Acquire Resource", „Release Resource"

Blok typu „Resource" to zasób dzielony o ograniczonej dostępności, który można pobrać i zwolnić, czego przykładem może być sytuacja, w której trzy stanowiska montażowe wykorzystują jedno narzędzie – to narzędzie stanowi wówczas zasób dzielony. Jeżeli stanowisko używa tego narzędzia to pobiera zasób. Po uzyskaniu zasobu żadne inne stanowisko nie może go nabyć, dopóki token posiadający zasób go nie zwolni. Zasoby są przydatne do modelowania ograniczeń czasowych, gdy wiele tokenów musi korzystać z ograniczonej puli zasobów.

W przypadku systemu kolejkowego zasobem ograniczonym jest podajnik automatyczny, który obsługuje jednocześnie transport komponentu A i komponentu B. Blok typu „Resource" jest połączony z blokami typu „Acquire Resource" oraz „Release Resource".

Blok typu „Acquire Resource" pozwala na pobranie zdefiniowanego zasobu dzielonego. Po umieszczeniu tej czynności w obszarze „Process Flow" po prawej stronie bloku pojawia się czerwony wykrzyknik, który informuje, że to działanie należy połączyć z konkretnym zasobem. Jeżeli nie zdefiniowano inaczej to tokeny zajmują zasób dzielony zgodnie z regułą FIFO (first in first out – pierwszy przyszedł pierwszy wyszedł).

W modelu symulacyjnym blok ten wykorzystywany jest do zajęcia podajnika automatycznego po pojawieniu się komponentu (A lub B) na dedykowanym buforze. Token, który zajmuje zasób dzielony (podajnik automatyczny) otrzymuje od razu nową etykietę (Label) o nazwie „resource". Za pomocą tej etykiety możemy się odwołać do zasobu dzielonego i zlecić mu wykonywanie konkretnych czynności.

Blok typu „Release Resource" pozwala na zwolnienie wcześniej zajętego zasobu dzielonego. Zasób dzielony jest wtedy zwracany do bloku typu „Resource", co umożliwia jego ponowne zajęcie przez kolejny token.

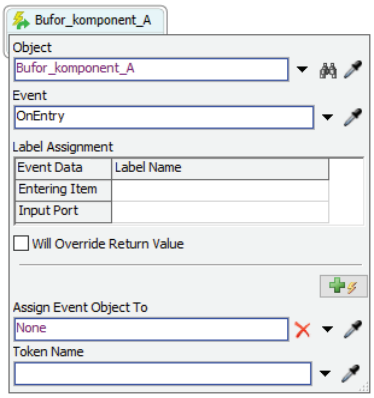
W analizowanym przypadku blok ten wykorzystywany jest do zwolnienia podajnika automatycznego. Dzięki temu podajnik może być wykorzystany do transportu kolejnej sztuki komponentu A lub B.

Dla czytelności schematu przedstawionego na rysunku 5 blokom tym nadano nazwy odpowiednio: „Podajnik automatyczny", „Zajmij podajnik" oraz „Zwolnij podajnik".

2.2. Blok typu „Event-Triggered Source”

Blok typu „Event-Triggered Source” pozwala na wygenerowanie tokenu w momencie, w którym nastąpi konkretne zdarzenie w modelu symulacyjnym. Jeżeli zdarzenie wystąpi, wówczas tworzy się jeden lub więcej tokenów, które są przesyłane do kolejnych bloków schematu. Jeżeli zdarzenie wyzwalające powinno zainicjować przepływ procesu (przez co należy rozumieć pojawienie się tokenów w „Process Flow”) należy użyć źródła wyzwalającego to zdarzenie.

W analizowanym modelu symulacyjnym zdarzeniem inicjującym tokeny w „Process Flow” jest pojawienie się komponentu A na obiekcie „Bufor_komponent_A” (analogicznie „Bufor_komponent_B” dla komponentu B) – w ustawieniach bloku w polu „Event” wybieramy opcję „OnEntry”. Szczegóły zobrazowano rysunkiem 6.



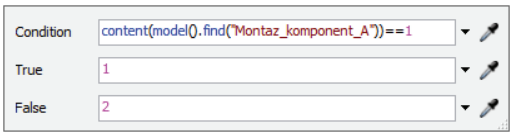
Rysunek 6. Ustawienia bloku „Bufor_komponent_A”

Dla czytelności schematu przedstawionego wcześniej na rysunku 5 nazwę bloku zmieniono na „Bufor_komponent_A”.

2.3. Blok typu „Decide”

Blok typu „Decide” pozwala na przesłanie tokena do jednego z dwóch lub więcej wyjść na podstawie zdefiniowanego warunku. Blok ten określa kolejną czynność, do której token powinien zostać przekierowany.

W analizowanym systemie kolejkowym blok typu „Decide” jest wykorzystywany do określenia miejsca, z którego podajnik powinien pobrać komponent do transportu. Dla czytelności schematu przedstawionego wcześniej na rysunku 5 nazwę bloku zmieniono na „Sprawdź czy na stanowisku odbywa się montaż komponentu A”. Następnie sprawdzamy, czy na dedykowanym stanowisku odbywa się montaż komponentu A lub B. W tym celu wykorzystywane jest polecenie edytujemy zawartość pola „Conditional Decide”. Sprawdzanym warunkiem jest pojemność stacji montażowej – korzystamy z funkcji `content`, której argumentem jest obiekt „Montaz_komponent_A” z modelu 3D. Jeżeli na stanowisku montażowym obrabiany jest komponent, wówczas podajnik powinien poczekać na zakończenie obróbki na tym stanowisku, a następnie przetransportować wyrób gotowy na bufor wyjściowy, a następnie pobrać z bufora kolejny komponent i zawieźć go na stanowisko montażowe. Jeżeli na stanowisku montażowym nie ma żadnego elementu, to podajnik powinien jedynie wykonać transport z bufora komponentu A lub B na stanowisko montażowe (rys. 7).



Rysunek 7. Decyzja warunkowa dla bloku „Sprawdź czy na stanowisku odbywa się montaż komponentu A”

2.4. Blok typu „Join”

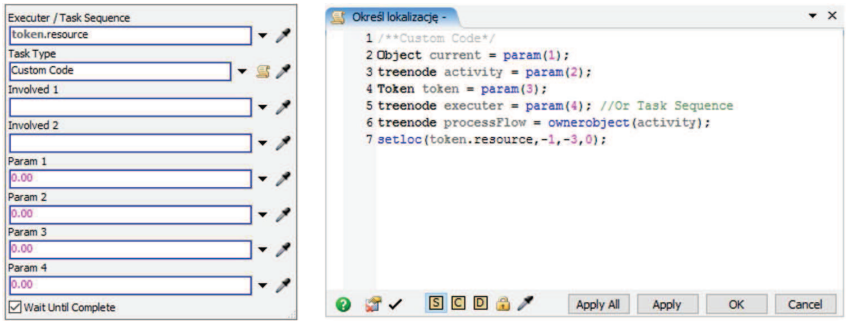
Blok typu „Join” przechowuje tokeny dopóki z każdego bloku wejściowego nie pojawi się po jednym tokenie. Do kolejnej czynności przekazywany jest token pochodzący z pierwszego bloku wejściowego, zaś pozostałe tokeny są usuwane.

Blok „Join” jest wykorzystywany do zapewnienia, że w przypadku, gdy na stanowisku montażowym obrabiany jest komponent, to podajnik musi poczekać na zakończenie czasu obróbki. W bloku „Montaż_komponent_A” generowany jest token po zakończeniu procesu montażu komponentu A. Blok „Join” łączy token pochodzący z bloku „Sprawdź czy na stanowisku odbywa się montaż komponentu A” z tokenem z bloku „Montaż_komponent_A”. Jeśli pojawią się oba tokeny podajnik może pobrać wyrób gotowy.

2.5. Blok typu „Custom Task”

Blok typu „Custom Task” to polecenie wykorzystywane w momencie, gdy konieczne jest zlecenie realizatorowi zadań konkretnej czynności, do której nie zdefiniowano specjalnego bloku w bibliotece obiektów „Process Flow”.

W analizowanym przypadku blok typu „Custom Task”, któremu nadano nazwę „Określ lokalizację” jest wykorzystywany do określenia lokalizacji, z której podajnik automatyczny powinien rozpocząć realizację zadania. W tym celu w ustawieniach tego bloku w polu „Task Type” dokonujemy edycji kodu programu – wykorzystywana jest funkcja `setloc`, za pomocą której określane jest położenie podajnika automatycznego, którego adres jest przechowywany w etykiecie `token.resource`. Szczegóły zobrazowano rysunkiem 8.

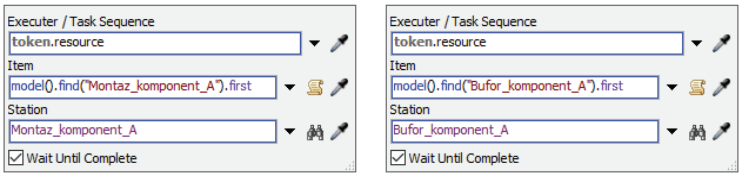


Rysunek 8. Ustawienia bloku „Określ lokalizację”

2.6. Bloki typu „Load”

Blok typu „Load” służy do zlecania realizatorowi zadań pobrania obiektu (załadunku) w modelu 3D. W tym celu należy wskazać realizatora zadań (Executer) oraz element, który powinien pobrać (Item). Ponieważ zawsze wiadomo, gdzie znajduje się dany obiekt w modelu symulacyjnym w programie FlexSim to nie ma konieczności określania stanowiska (Station), z którego należy pobrać element.

Bloki typu „Load” są wykorzystywane do zlecenia podajnikowi automatycznemu (wpisanemu do etykiety `token.resource`) czynności pobrania komponentu z bufora (wówczas komponent traktowany jest jako pierwszy element na buforze – korzystając z próbnika w polu „Item” wskazujemy obiekt „Bufor_komponent_A” z modelu 3D, a następnie do zapisu `model().find("Bufor_komponent_A")` dodajemy funkcję `first`) oraz czynności pobrania wyrobu gotowego ze stacji montażowej (wówczas wyrób gotowy traktowany jest jako pierwszy element na stacji montażu – korzystając z próbnika w polu „Item” wskazujemy obiekt „Montaz_komponent_A” z modelu 3D, a następnie do zapisu `model().find("Montaz_komponent_A")` dodajemy funkcję `first`). Szczegóły zobrazowano rysunkiem 9.



Rysunek 9. Ustawienia bloków „Pobierz komponent A ze stanowiska” oraz „Pobierz komponent A z buforu”

2.7. Bloki typu „Custom Code”

Bloki typu „Custom Code” pozwalają na zdefiniowanie specjalnych zadań przez użytkownika modelu. Użytkownik ma możliwość wykorzystania istniejących opcji poprzez wybranie ich z listy rozwijanej lub może wprowadzić własny kod programu. Po pojawieniu się tokena w tym bloku automatycznie wykonywane jest zdefiniowane przez użytkownika polecenie, a następnie token przekazywany jest do kolejnej czynności bez opóźnienia czasowego.

Za pomocą bloków typu „Custom Code” określana jest prędkość podajnika automatycznego (rys. 10).

Rysunek 10. Kod programu do określania prędkości podajnika automatycznego

W liniach kodu 2–5 znajduje się standardowy kod deklaracji parametrów w Process Flow. Są to wartości domyślne, które pozwalają się odwołać do bieżącego obiektu (linia 2), bieżącej czynności (linia 3), tokena (linia 4) i głównego właściciela czynności, czyli Process Flow (linia 5). Warto tutaj nadmienić, że „Object" stanowi klasę obiektów w programie FlexSim, pozwalającą na odwołanie się do zapisanej w drzewie FlexSim zmiennej z danymi obiektowymi. Z kolei „treenode" to zmienna zawierająca odwołanie do konkretnego węzła dostępowego (node) w drzewie FlexSim. Warto również zaznaczyć, że klasa „treenode" jest klasą bazową dla wszystkich obiektów i węzłów dostępowych w drzewie FlexSim. Chociaż można uzyskać dostęp do większej liczby danych i funkcji specyficznych dla węzłów z danymi obiektu przy użyciu klasy „Object", „treenode" jest najbardziej podstawowym interfejsem. Po pierwsze daje dostęp do danych i funkcji związanych z strukturą drzewa. Po drugie daje dostęp do podstawowych danych przechowywanych na węzłach, takich jak nazwy węzłów i wartości danych. Po trzecie, ponieważ tak wiele logiki modelowania używa etykiet, klasa „treenode" zapewnia mechanizm łatwej manipulacji etykietami na obiektach, tokenach i sekwencjach zadań.

W linii kodu nr 6 następuje przypisanie wartości zwartej w `token.resource` do zmiennej `wozek`. W etykiecie o nazwie `token.resource` przechowywany jest adres podajnika automatycznego pobranego do realizacji zadania. W linii 7 do zmiennej `dystans` przypisywana jest odległość, jaką podajnik automatyczny ma do pokonania pomiędzy stacją, z której pobiera komponent, a stacją na którą ten komponent zawozi. Zmienna `dystans` jest zmienną typu double. Odległość do przejechania jest określana za pomocą funkcji `distancetotravel(object traveler, object destination)`, gdzie `object traveler` to miejsce, z którego pobierany jest komponent, a `object destination`, to miejsce do którego komponent należy dostarczyć. W liniach 8 i 9 do zmiennych `minczas` oraz `maxczas` wpisany został odpowiednio minimalny i maksymalny czas przejazdu wózka na trasie pomiędzy `object traveler` a `object destination`. W liniach 12 i 13 do zmiennych `vmin` i `vmax` przypisano minimalną i maksymalną prędkość z jaką powinien poruszać się podajnik automatyczny. Prędkość minimalną rozumieć należy jako iloraz odległości koniecznej do pokonania i maksymalnego czasu przejazdu podajnika automatycznego. Z kolei prędkość maksymalna to iloraz odległości i minimalnego czasu transportu. W 14 linii kodu ustawiana jest prędkość podajnika automatycznego za pomocą funkcji

`setvarnum(object, variable, numeric value)`. W analizowanym przypadku na podajniku automatycznym określana jest wartość zmiennej `maxspeed` – prędkość maksymalna. Zmienna `maxspeed` może przyjmować wartości z rozkładu jednostajnego (`uniform`) pomiędzy prędkością minimalną a prędkością maksymalną.

2.8. Bloki typu „Travel”

Blok typu „Travel” pozwala na zlecenie realizatorowi zadań (Task Executer) polecenia przemieszczenia się do określonego obiektu w modelu symulacyjnym 3D.

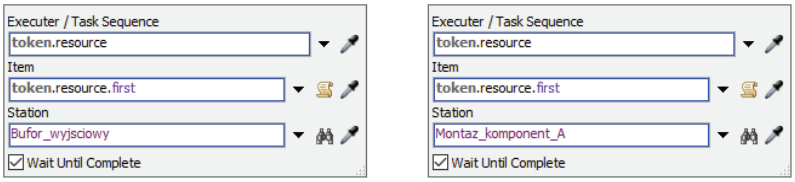
Bloki typu „Travel” są wykorzystywane do zlecenia podajnikowi automatycznemu zadania przejazdu do stanowiska montażu komponentu A (wówczas w polu „Station” korzystając z próbnika należy wskazać obiekt „Montaz_komponent_A” z modelu 3D) oraz do zlecenia zadania przejazdu do bufora wyjściowego (wówczas w polu „Destination” korzystając z próbnika należy wskazać obiekt „Bufor_wyjsciowy” z modelu 3D. Szczegóły zobrazowano rysunkiem 11.

Rysunek 11. Ustawienia bloków „Jedź do buforu wyjściowego” oraz „Jedź do stanowiska montażu komponentu A”

2.9. Bloki typu „Unload”

Blok typu „Unload” pozwala na zlecenie realizatorowi zadań polecenia wyładunku obiektu, który transportuje/przechowuje w modelu symulacyjnym 3D.

W analizowanym systemie kolejkowym za pomocą bloków typu „Unload” realizowany jest rozładunek podajnika automatycznego na stanowisku montażowym i na buforze wyjściowym. Każdorazowo podajnik automatyczny wyładowuje element, który transportuje – w polu „Item” wprowadzamy zapis `token.resource.first` oznaczający pierwszy element jaki znajduje się na `token.resource`. W polu „Station” przy użyciu próbnika wskazujemy stacje wyładunku na modelu 3D – obiekty „Montaz_komponent_A” oraz „Bufor_wyjsciowy”. Szczegóły zobrazowano rysunkiem 12.



Rysunek 12. Ustawienia bloków „Wyładuj komponent A na bufor” oraz „Wyładuj komponent A na stanowisku”

3. Testowanie modelu i analiza uzyskanych rozwiązań

Utworzony model symulacyjny przedstawionego na wstępie systemu kolejkowego został poddany testom zgodnie z przyjętymi wcześniej założeniami. W tabeli 1 przedstawiono uzyskane rozwiązania.

Tabela 1. Wyniki kolejnych dziesięciu powtórzeń analizowanego eksperymentu symulacyjnego

Kolejne powtórzenie	**Zmienna wynikowa**							
	$\overline{T_{oczA}}$ [s]	$\overline{T_{oczB}}$ [s]	U_{rA} [%]	U_{bA} [%]	U_{rB} [%]	U_{bB} [%]	DT [km]	U_t [%]
1	361,9	362,2	48,0	3,7	42,4	5,4	5,0	84,9
2	348,9	386,9	47,3	3,5	41,4	9,5	4,8	82,9
3	467,1	380,9	39,8	6,9	48,9	2,2	4,7	79,1
4	359,3	339,7	48,7	1,6	42,7	6,4	4,8	83,4
5	369,5	388,9	40,7	11,1	49,0	1,2	4,8	82,6
6	403,4	432,1	53,7	0,0	36,5	20,1	4,5	78,8
7	370,3	417,9	53,2	1,4	38,9	11,0	4,5	79,1
8	376,6	418,9	48,0	4,3	39,7	11,7	4,7	78,9
9	341,7	391,8	44,7	4,2	46,9	1,9	5,0	86,4
10	379,3	388,6	41,4	10,7	46,1	2,5	4,8	82,7
Średnia	**377,8**	**390,8**	**46,5**	**4,7**	**43,2**	**7,2**	**4,8**	**81,9**

Jak wynika z danych przedstawionych w tabeli 1 średni czas oczekiwania na transport z bufora komponentów A wynosi 377,8 sekund, zaś z bufora komponentów B 390,8 sekund. Wykorzystanie stanowisk montażowych nie jest zadowalające. Stanowisko A przetwarza komponenty przez 46,5% czasu pracy, zaś stanowisko B przez 43,2%. Stanowiska montażowe A i B są bezczynne przez odpowiednio 48,7% i 49,6%. Czas bezczynności należy rozumieć jako oczekiwanie na kolejny komponent do obróbki. Z kolei blokada stanowisk A i B wynosi odpowiednio 4,7% i 7,2%, czyli stanowiska skończyły już obrabiać komponent i oczekują aż podajnik zabierze element na bufor wyjściowy. Stopień wykorzystania podajnika automatycznego wynosi średnio 81,9% (przez co należy rozumieć, że podajnik automatyczny transportuje komponenty), a średnia liczba kilometrów przebywanych w ciągu zmiany roboczej to 4,8 km. Podajnik automatyczny stanowi bez wątpienia wąskie gardło w systemie. Należałoby się zastanowić, czy dodanie kolejnego podajnika automatycznego nie usprawniłoby działania systemu.

4. Podsumowanie

Symulacja komputerowa stanowi narzędzie pozwalające na weryfikację działania istniejących i planowanych systemów, dzięki czemu możliwe jest bezpieczne weryfikowanie zamierzonych zmian, co w znaczymy stopniu ogranicza ryzyko podejmowanych decyzji. W niniejszym artykule zaprezentowano model symulacyjny systemu transportowego zbudowany w programie FlexSim 3D Simulation Software. W bardzo czytelny sposób odwzorowano naturalną zmienność analizowanego

systemu, identyfikując przy tym jego wąskie gardło. Łatwość modelowania, bogata biblioteka obiektów oraz przyjazny użytkownikowi interfejs skłaniają do stosowania oprogramowania FlexSim w dalszych badaniach.

Bibliografia

[1] Beaverstock M., Greenwood A., Nordgren W.: *Applied Simulation: Modeling and Analysis using FlexSim.* 5th Edition. FlexSim Software Products, Inc., Orem, USA 2017.
[2] Bhat U.N.: *An Introduction to Queueing Theory. Modeling and Analysis in Applications*. Springer, Birkhäuser Science, Berlin 2015.
[3] Filipowicz B.: *Modelowanie i optymalizacja systemów kolejkowych. Cześć I. Systemy markowskie*. Wydawnictwo POLDEX, Kraków 2006.
[4] Hanczewski S., Kaliszan A.: *Badania symulacyjne wielousługowych systemów kolejkowych*. Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjne, nr 8–9, 2016, s. 1019–1024.
[5] Hanczewski S., Kmiecik D., Weissenberg J.: *Modelowanie analitycznie wielousługowych, niepełnodostępowych systemów kolejkowych*. Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjne, nr 8–9, 2016, s. 874–878.
[6] Jurczyk K., Woźniak W.: *Metoda ustalania liczby powtórzeń eksperymentu symulacyjnego o skończonym horyzoncie czasowym*. Studies & Proceedings of Polish Association for Knowledge Management, tom 78, 2016, s. 78–87.
[7] Karkula M.: *Modelowanie i symulacja procesów logistycznych*. AGH, Kraków 2013.
[8] Kisielewski P., Sobota Ł.: *Zastosowanie teorii masowej obsługi do modelowania systemów transportowych*. Autobusy: technika, eksploatacja, systemy transportowe, nr 6, 2016, s. 600–604.
[9] Kwiecień J., Filipowicz B.: *Comparison of firefly and cockroach algorithms in selected discrete and combinatorial problems*. Bulletin of the Polish Academy of Sciences. Technical Sciences, Vol. 62, No. 4, 2014, s 797–804.
[10] Wu Y., Dong M.: *Combining multi-class queueing networks and inventory models for performance analysis of multi-product manufacturing logistics chains*. International Journal of Advanced Manufacturing Technology, Vol. 37, 2008, s. 564–575.

MODELING AND SIMULATION OF QUEUING SYSTEMS IN FLEXSIM SOFTWARE – A CASE STUDY

Summary

The paper proposes a simulation model of a transportation system of two different item types for two dedicated machines by one automated conveyor. The model was created in FexSim 3D Simulation Software. The following paragraphs of the article detail the principles of operation of the created model and the results of the numerical tests that have been carried out for identifying the system bottleneck.

Keywords: modelling and simulation, advanced technical computing environments, queueing systems

Praca realizowana w ramach grantu dziekańskiego nr 15.11.200.329

Monika Klaś
Katedra Badań Operacyjnych
Wydział Zarządzania
AGH Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie
Ul. Gramatyka 10, 30-067 Kraków
e-mail: monika.klas@flexsim.pl

Krzysztof Jurczyk
Katedra Inżynierii Zarządzania
Wydział Zarządzania
AGH Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie
Ul. Gramatyka 10, 30-067 Kraków
e-mail: kjurczyk@zarz.agh.edu.pl