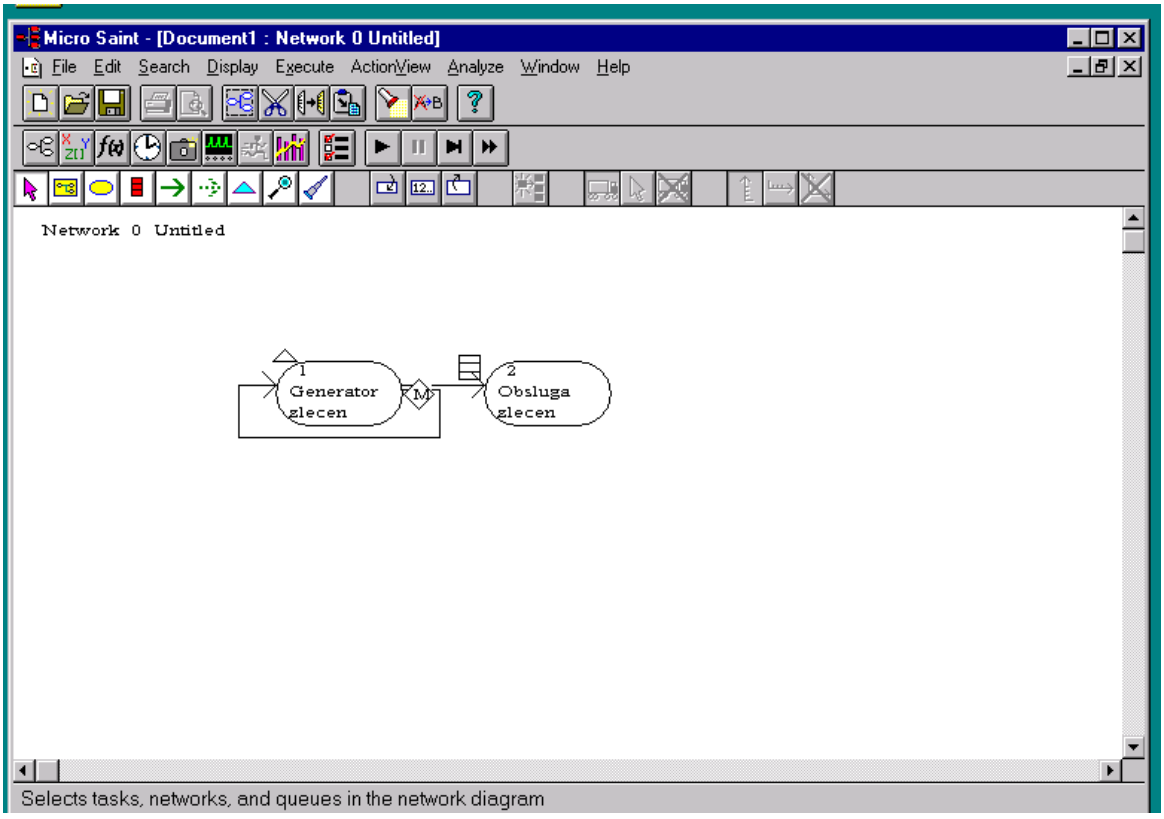
Skrócona instrukcja użytkowania programu Micro Saint

1. Uruchomienie programu

Program Micro Saint służy do symulacji procesów dyskretnych. Działa w środowisku systemu Windows i w związku z tym ogólny sposób posługiwania się programem jest zgodny ze standardami Windows. Program jest dostępny na dysku systemowym, w katalogu **..\util\Msaint**, skąd można go uruchomić wywołując plik **saint.exe**. W trakcie korzystania z programu dostępna jest (anglojęzyczna) pomoc dla użytkownika (polecenie *Help*).

2. Definiowanie modelu

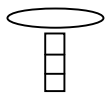
Po uruchomieniu programu Micro Saint zostaje otwarty nowy arkusz zatytułowany *untitled,* na którym można rozpocząć pracę nad badanym modelem. Definiowanie modelu przebiega w dwóch fazach. W fazie pierwszej należy zdefiniować topologię modelu, który chcemy symulować, a w fazie drugiej jego parametry.



Rys. 1 Schemat prostego systemu, zdefiniowanego w Micro Saint

2.1 Definiowanie topologii modelu

Przed rozpoczęciem wprowadzania modelu należy przeprowadzić analizę procesu, której wynikiem będzie rozbicie go na operacje i określenie wzajemnych kolejności wykonywania operacji.
Topologię modelu definiuje się budując schemat blokowy, składający się z następujących obiektów:



Task – zasób trwały, na którym są wykonywane operacje, np. maszyna lub procesor.

Queue - kolejka (bufor), umieszczana na wejściu obiektu *Task*.

Path - ścieżka określająca kolejność obsługi (przepływ produktów, klientów, zleceń itp.).

Decision Node - umieszczany automatycznie, gdy następuje rozgałęzienie ścieżki.

Wprowadzanie modelu odbywa się przy wykorzystaniu paska narzędzi znajdującego się w lewej części arkusza. Dostępnych jest 9 trybów, wybieranych za pomocą myszy. Poszczególne tryby pracy umożliwiają:

- tryb *Pointer* - edycję elementów modelu (definiowanie parametrów, kopiowanie, kasowanie, przenoszenie itp.),
- tryb *Network* - tworzenie hierarchicznych modeli symulacyjnych,
- tryb *Task* - umieszczenie obiektów *Task* (operacji) na arkuszu,
- tryb *Queue* – umieszczanie obiektów *Queue* (kolejek) na arkuszu,
- tryb *Path* – wprowadzanie połączeń (ścieżek przepływu) między obiektami,
- tryb *Undo Path* – usuwanie połączeń,
- tryb *Start Job* – oznaczenie operacji startowej,
- tryb *Zoom In* – powiększanie widoku modelu,
- tryb *Zoom Back* – pomniejszanie widoku modelu.

Przy wykorzystaniu poszczególnych trybów pracy możliwe jest wykonywanie następujących operacji:

- **Umieszczanie operacji na arkuszu** (tryb *Task*)

W tym trybie pracy każde wciśnięcie lewego przycisku myszki, gdy jej wskaźnik znajduje się w polu arkusza *Network*, powoduje umieszczenie obiektu *Task* (operacji). Kolejne operacje są numerowane automatycznie.

- **Wprowadzanie połączeń między obiektami** (tryb *Path*)

Aby połączyć dwa obiekty ścieżką, należy umieścić wskaźnik myszki w polu obiektu *Task*, od którego ścieżka ma się rozpocząć, przycisnąć lewy przycisk myszki i przesunąć wskaźnik myszki na obiekt *Task*, do którego ma być doprowadzona ścieżka. Zwolnienie lewego przycisku myszki spowoduje połączenie obiektów. Aby narysować ścieżkę łączącą wyjście z wejściem danego obiektu, należy nacisnąć lewy przycisk myszki, gdy jej wskaźnik znajduje się w polu obiektu. W przypadku gdy z obiektu wychodzi więcej niż jedna ścieżka, automatycznie wstawiany jest węzeł decyzyjny (*Decision Node*).

- **Usuwanie połączeń** (tryb *Undo Path*)

Aby usunąć ścieżkę, należy zaznaczyć ją w trybie pracy Undo Path.

- **Wprowadzanie kolejek** (tryb *Queue*)

W trybie pracy *Queue* kolejka zostaje umieszczona na wejściu obiektu *Task* po wciśnięciu lewego przycisku myszki (wskaźnik myszki powinien znajdować się w polu obiektu). Kolejne kolejki są numerowane automatycznie.

- **Usuwanie operacji i kolejek** (tryb *Pointer*)

Aby usunąć operację lub kolejkę z arkusza, należy ją zaznaczyć (przez wciśnięcie lewego przycisku myszki, gdy jej wskaźnik znajduje się w polu obiektu przeznaczonego do skasowania). Kiedy obiekt lub grupa obiektów jest zaznaczona (zaznaczone obiekty są pogrubione), można ją skasować wykorzystując polecenie *Clear* z menu *Edit* lub klawisz DEL.

- **Kopiowanie elementów modelu** (tryb *Pointer*)

Należy zaznaczyć elementy, które chcemy skopiować. Z menu *Edit* należy wybrać polecenie *Copy*, a następnie, w celu umieszczenia nowych elementów na arkuszu, wybrać polecenie *Paste* z menu *Edit*. W trakcie kopiowania program pyta się o nowe numery zadań (kolejek). Można podać dowolne numery, które jeszcze nie zostały wykorzystane w modelu (zalecane jest podawanie kolejnych numerów poczynając od ostatnio użytego).

- **Definiowanie operacji startowej** (tryb *Start Job*)

Aby zdefiniować operację, od której rozpocznie się symulacja, należy wskaźnik myszki umieścić w polu tej operacji, a następnie wcisnąć lewy przycisk myszki. Operacja startowa jest zaznaczana trójkątem w lewym górnym rogu. W chwili rozpoczęcia symulacji w operacji startowej generowany jest znacznik, któremu przypisana jest wartość zmiennej systemowej *tag*. Znaczniki reprezentują przemieszczające się obiekty identyfikowane wartością zmiennej *tag*. W zależności od symulowanego procesu mogą one odpowiadać detalom, produktom, zadaniom, klientom, zgłoszeniom itp. Możliwość definiowania operacji startowej pozwala na wykorzystanie jednego arkusza do symulacji kilku niezależnych modeli. Po uruchomieniu symulacji aktywny jest tylko ten model, który ma zdefiniowaną operację startową.

2.2 Definiowanie parametrów modelu

Po zdefiniowaniu topologii modelu należy określić parametry funkcjonalne poszczególnych obiektów oraz zdefiniować zmienne, które będą występować w wyrażeniach sterujących symulacją modelu.

Wprowadzenie parametrów danego obiektu (*Task, Queue* lub *Decision*) jest możliwe po wciśnięciu klawisza *Pointer* i podwójnym kliknięciu w polu obiektu.

2.2.1 Parametry obiektu *Task*

Na Rys. 2 przedstawiono przykładowe okno do definicji obiektu *Task*, poniżej opisano najważniejsze parametry obiektu:

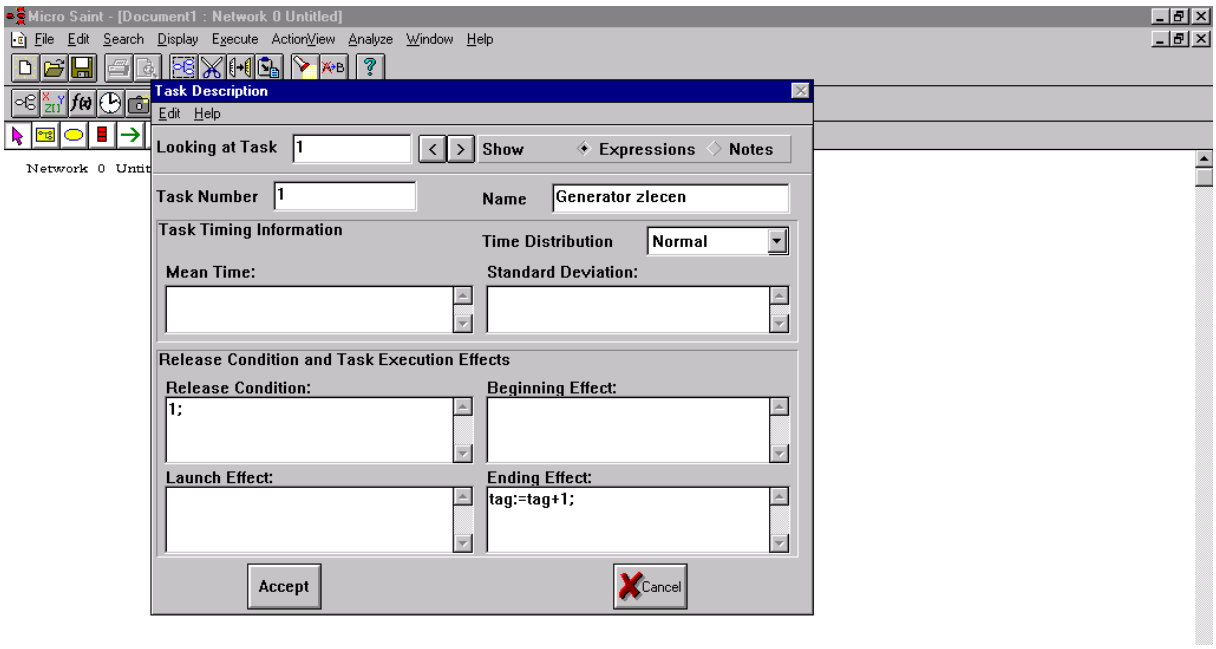
- *Task Number* – pole z numerem operacji. Operacje numerowane są automatycznie w momencie tworzenia.
- *Name* – opisowa nazwa operacji (stanowiska)**,**
- *Show Expressions/Notes* – pole wyboru wyglądu ekranu – wybór *Expressions* wyświetla wyrażenia zdefiniowane w dolnej części ekranu; wybór *Notes* zamiast wyrażeń wyświetla pole z notatkami
- *Task Timing Information* - parametry określające czas wykonywania operacji w tym:
  - *Time Distribution* - rozkład prawdopodobieństwa, według którego jest wyznaczany czas wykonywania zadania. Do wyboru są rozkłady: *Normal* (normalny)*, Exponential* (wykładniczy)*, Gamma, Rectangular* (jednostajny)*, Lognormal* (logrytmiczno-normalny). Szczegółowa charakterystyka rozkładów dostępna jest w pomocy (opcja *Help*). Wartość inicjująca generatora (*Random Number Seed*) ustawia się wywołując z menu opcję *Execute/Settings*.
  - *Mean Time* - wartość średnia rozkładu; można wprowadzić liczbę lub wyrażenie (w jednostkach czasu).
  - *Standard Deviation* - odchylenie standardowe. Chcąc określić czas operacji w sposób deterministyczny, należy zadać zerową wartość odchylenia standardowego (taka wartość jest przyjmowana domyślnie, gdy nic się nie wpisze).
- *Release Condition* - warunek rozpoczęcia operacji. W polu wpisywane jest wyrażenie, które rozstrzyga o wykonaniu operacji. Wykonywanie operacji zostanie rozpoczęte, jeżeli dotrze do niej *znacznik* (*tag*) i wyrażenie wpisane w pole *Release Condition* ma wartość niezerową. W przypadku zerowej wartości wyrażenia rozpoczęcie operacji będzie wstrzymane. (Jeżeli nic nie zostanie wpisane, to domyślna wartość pola jest równa jeden).
- *Beginning Effect* - w tym polu wpisuje się wyrażenia (instrukcje), które mają być wykonane (wyliczone) w chwili rozpoczęcia wykonywania operacji. Przeważnie są to instrukcje zmieniające wartości zmiennych stanu lub parametrów sterujących. Czas wykonywania operacji i zmienna *duration* (opisana dalej) są obliczane dopiero po wykonaniu instrukcji z pola *Beginnig Effect*. Instrukcje mogą być wielolinijkowe – wystarczy wcisnąć ENTER na koniec każdej linii.
- *Launch Effect* - pole to pełni prawie taką samą rolę jak pole *Beginning Effect* z tą różnicą, że jest ono wykonywane (wyliczane) po obliczeniu zmiennej *duration* (pole przydatne, gdy chcemy korzystać z opcji ActionView – czyli animacji symulowanych zdarzeń).
- *Ending Effect* - pole dla instrukcji wykonywanych w chwili zakończenia operacji.

Kolejność obliczania wyrażeń opisujących operacje:

1. Release condition
2. Beginning effect
3. Mean time
4. Standard deviation
5. Launch effect
6. Ending effect

Po wpisaniu zmian przyciśnij *Accept*.

Uwaga: Obiekt *Task* nie ma ograniczeń na liczbę przyjmowanych znaczników do przetwarzania. Efekt ograniczenia ilości przetwarzanych jednocześnie znaczników można uzyskać definiując odpowiednio zmienne.



Rys. 2 Przykładowe okno parametrów *Task*

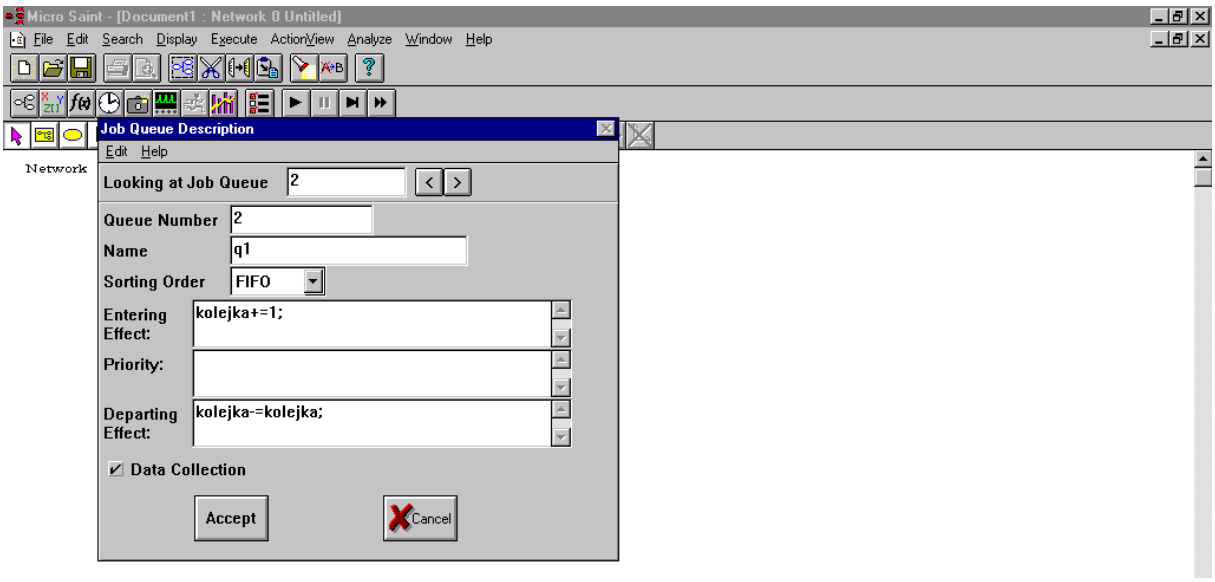
2.2.2Parametry obiektu *Queue*

Na Rys. 3 przedstawiono przykładowe okno do definicji obiektu *Queue*, poniżej opisano najważniejsze parametry obiektu:

- *Name* - opisowa nazwa kolejki
- *Sorting Order* – wybór reguły sortowania znaczników (zadań) w kolejce. W rozwijanym menu dostępne są reguły FIFO, LIFO lub *Sorted* (na podstawie priorytetów zadanych w polu *Priority*).
- *Entering Effect* - w polu wpisuje się wyrażenia (instrukcje), które mają być wykonane (wyliczone) w chwili wejścia znacznika do kolejki. Przeważnie są to instrukcje zmieniające wartości zmiennych stanu lub parametrów sterujących. Należy zwrócić uwagę, że reguła sortowania jest wyznaczana dopiero po wykonaniu instrukcji z pola *Entering Effect*. Instrukcje mogą być wielolinijkowe – wystarczy wcisnąć ENTER na koniec każdej linii.
- *Priority* - pole istotne tylko przy wyborze reguły *Sorted*. Micro Saint oblicza wartości priorytetów dla wszystkich znaczników znajdujących się w kolejce, zgodnie z wyrażeniem podanym w polu. Należy zwrócić uwagę, że aby otrzymać różne priorytety dla różnych znaczników, wpisane wyrażenie powinno być zależne od wartości znacznika (np. tablica *p[tag]*). Znacznik z największym priorytetem (wartością wyrażenia) będzie wybierany z kolejki jako pierwszy.
- *Departing Effect* - pole dla instrukcji wykonywanych w chwili, gdy znacznik opuszcza kolejkę.
- *Data Collection* – pole wyboru (domyślnie jest włączone); pozwala na automatyczne gromadzenie wszelkich danych nt. kolejki (np. zajętość w funkcji czasu). Dane są gromadzone, jeżeli dodatkowo zostało wybrane pole *Queue Data Collection* w oknie *Execute/Settings* wywołanym z menu.

Po wpisaniu zmian przyciśnij *Accept*.

Uwaga: Obiekt kolejka nie ma pola definiującego długość bufora. Efekt ograniczenia długości bufora można uzyskać definiując odpowiednio zmienne.



Rys. 3 Przykładowe okno parametrów *Queue*

2.2.3Parametry obiektu *Decision Node* (rozgałęzienie)

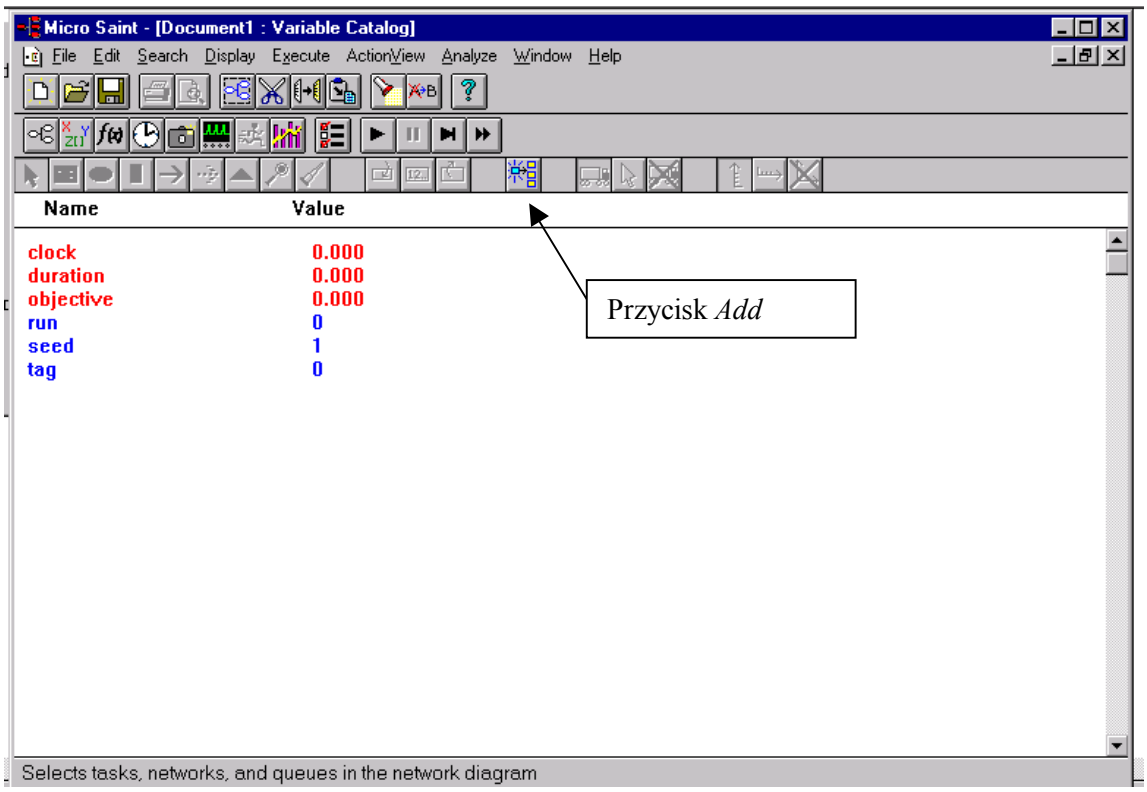
Poniżej opisano najważniejsze parametry obiektu:

- *Task Name* – zawiera nazwę operacji, z której wychodzi rozgałęzienie.
- *Decision Type* – typ decyzji wyboru ścieżki. Dostępne są następujące trzy typy decyzji:
  - *Multiple* – znaczniki ulegają „powieleniu" (z zachowaniem tej samej wartości *tag*), tak że każdy „klon" jest rozsyłany dalej po tych rozgałęziających się ścieżkach, które mają niezerową wartość wyrażenia zdefiniowanego w polu *Routing Condition*.
  - *Probabilistic* – znacznik wybiera losowo jedną spośród rozgałęziających się ścieżek z prawdopodobieństwem wyboru danej ścieżki określonym wartością wyrażenia wpisanego w pole *Routing Condition* (po podzieleniu przez sumę wartości wyrażeń wszystkich pól (ścieżek)). Uwaga: domyślnie decyzja wyboru ścieżki jest typu probabilistycznego, z jednakowymi prawdopodobieństwami wyboru.
  - *Tactical* - znacznik wybiera tę ścieżkę, dla której wartość wyrażenia w polu *Routing Condition* jest największa.
- *Next Task* – zawiera numer oraz nazwę następnej operacji, która będzie wykonywana jeśli spełnione będą warunki w polu *Routing Condition*.
- *Routing Condition* – w polu wpisuje się warunki wyboru danej ścieżki.

Po wpisaniu zmian przyciśnij *Accept*.

3.Zmienne i wyrażenia

Okno *Variable Catalog* (wywołane z menu *Display* lub przez wciśnięcie odpowiedniej ikonki) zawiera listę wszystkich zmiennych zdefiniowanych dla modelu. Lista zawiera ich nazwy oraz aktualne lub inicjujące wartości. Zmienne są używane w programie do sterowania procesem symulacji lub przechowywania informacji o jego przebiegu; można je dodawać i usuwać.



Rys. 4 Tryb pracy *Variable Catalog*

Każda używana zmienna niesystemowa musi być zadeklarowana przez wywołanie opcji *Display/Variable Catalog*, naciśnięcie przycisku *Add* (w standardowym ustawieniu na pasku narzędzi przedostatni przycisk w środkowym rzędzie, oznaczony symbolem graficznym – patrz Rys. 4) i podanie jej nazwy (*Name*), typu (*Type*) i wartości początkowej (*Initial Value*). Zmienna może być typu *Integer* (całkowitego), *Real* (rzeczywistego) lub tablicą liczb całkowitych (*Array of Integers*) lub rzeczywistych (*Array of Reals*). W przypadku zmiennych typu tablicowego trzeba dodatkowo podać zakres tablicy wpisując w polu *First index numered 0 through* odpowiednią liczbę. Dodatkowo jako komentarz można wprowadzić opis zmiennej (*Purpose*).
Przypisanie wartości początkowych w przypadku tablic powoduje, że elementy tablicy otrzymują wówczas **tę samą wartość**. Aby przypisać w chwili początkowej (zerowej) różne wartości elementom tablicy, należy wywołać z menu opcję *Display/Event Queue*, nacisnąć przycisk *Add* i wpisać w pole *Perform At Time* wartość 0, a w pole *Expressions* — wyrażenia przypisujące poszczególnym elementom tablicy odpowiednie wartości (np. tab[3]:=12, tab[2]=5;). W ten sposób wymusza się wykonanie (wyliczenie) wyrażeń w chwili uruchomienia symulacji.

Pewne zmienne są w symulatorze predefiniowane jako **zmienne systemowe**. Są to:

- clock - przechowuje aktualny czas symulacji, od chwili jej rozpoczęcia. Można używać jej w dowolnych wyrażeniach w modelu, zwracając jednak uwagę na to, by jej nie modyfikować. Można np. zapisywać czas, w którym wydarzenie miało miejsce, definiując inną zmienną równoważną zmiennej *clock*, np. finishtime:=clock;
- duration – przechowuje czas, jaki każdy znacznik spędza w bieżącej operacji (*Task*) lub kolejce. Na przykład jeśli znacznik opuszcza kolejkę, Micro Saint ustawia *duration* na wartość równą czasowi przebywania znacznika w tej kolejce. Jeśli znacznik zaczyna być poddawany danej operacji (przybywa do *Task*), Micro Saint ustawia *duration* na wartość równą czasowi przetwarzania znacznika. Używanie zmiennej *duration* ma sens jedynie w polach *Launch Effect* i *Departing Effect*.
- run – przechowuje numer przebiegu symulacji, gdy wykonujemy wiele przebiegów. Przydatna, gdy w części przebiegów chcemy np. zdefiniować inne parametry (if run < 50 then dl:=45;). Liczbę przebiegów symulacji ustawia się wywołując z menu opcję *Execute/Settings*.

- seed – losowy numer ziarna generatora pseudolosowego; można ustawić go ręcznie w Execution Settings.
- tag – przechowuje identyfikator (numer) każdego znacznika poruszającego się w modelu. Pierwszy znacznik numerowany jest od zera. Przypisana do znacznika wartość *tag* pozostaje niezmieniona do końca symulacji (chyba, że zostanie zmieniona w wyrażeniu – co nie jest zalecane).
  Zmienna *tag* standardowo może być wykorzystywana do modelowania następujących procesów:
  - generacja znaczników: generowanie znaczników reprezentujących produkty, klientów, zlecenia, zgłoszenia itp. można uzyskać definiując obiekt *Task* z pętlą generującą zadaną ilość znaczników – patrz przykład.
  - wybór jednej z równoległych operacji,
  - wybór zadania z kolejki wg. priorytetu,
  - przypisanie specyficznych wartości do poszczególnych znaczników: przykładowo, czas wykonywania operacji może zależeć od rodzaju produktu, zlecenia czy klienta. W takiej sytuacji najlepiej użyć tablicy (wcześniej ją deklarując i nadając jej wartości zależne od numeru znacznika), a w wyrażeniach indeksować tablicę zmienną *tag*. Na przykład wielkość_zlecenia[tag].

Uwaga. Nie zaleca się modyfikowania zmiennych systemowych!
Na zmiennych można wykonywać wszystkie podstawowe operacje arytmetyczne (+, , *, /, ^, %) i logiczne (==- , <= , >=, <, >, &, |), tworząc złożone wyrażenia. Dostępne też są różnego rodzaju funkcje (np. trygonometryczne, matematyczne itp.), których szczegółowy opis można znaleźć wywołując z menu opcję *Help/Functions*. Wartości zmiennych można zmieniać stosując instrukcje podstawienia (:=) lub modyfikacji (+=, –= ,*=, /=). Dostępne są też instrukcje warunkowe *if ... then ... else* oraz pętle *while ... do ...* patrz *Help*.
Koniec każdego wyrażenia i instrukcji musi być zaznaczony średnikiem (;). Komentarze umieszcza się w nawiasach klamrowych ({...}).

4.Przykład

Rozpatrzmy system z jednym stanowiskiem obsługi, w którym pojawia się 10 zgłoszeń różnego typu. Stanowisko może obsłużyć co najwyżej jedno zgłoszenie na raz, a bufor przed stanowiskiem ma nieograniczoną pojemność. Czas obsługi zgłoszeń jest zależny od typu zgłoszenia.
Model symulacyjny powyższego procesu jest przedstawiony na rysunku na stronie 1.
Użyte zmienne:

- *wolne* - określa, czy stanowisko obsługujące zlecenie jest wolne (wartość początkowa = 1),
- *czas[]* – tablica czasów wykonywania poszczególnych zleceń; zakres od 0 do 10 (wartości czasów zdefiniowano w *Event Queue*).

Parametry:

- *Task* nr 1 (Generator zleceń)
  - *Mean Time, Standard Deviation*: 0;
  - *Release Condition*: 1;
  - *Ending Effect*: tag:=tag +1;
- *Decision* (*Rozgałęzienie)*
  - *Decision Type*: Multiple
  - *Routing Condition* z 1 do 1: tag<10;
  - *Routing Condition* z 1 do 2: 1;
- *Task* nr 2 (Obsługa zleceń)
  - *Time Distribution*: Normal
  - *Mean Time*: czas[tag];
  - *Release Condition*: wolne;
  - *Beginning Effect*: wolne:=0;
  - *Ending Effect*: wolne:=1;
- *Queue* (Kolejka zleceń)
  - *Sorting Order:* FIFO

W powyższym przykładzie obiekt *Task* nr 1 wraz z rozgałęzieniem pełni rolę generatora zleceń. Obiekt ten generuje pierwszy znacznik (*tag*=0), który na wyjściu przyjmuje wartość równą *tag*+1 czyli 1. W węźle *Decision Node* znacznik ten jest powielany. Jedna kopia trafia do bloku *Task* nr 2, natomiast druga kopia

generuje następny znacznik. Proces się powtarza aż do momentu pojawienia się znacznika nr 10, kiedy ścieżka zwrotna przestaje być aktywna. W rezultacie w chwili zerowej są wygenerowane znaczniki o numerach od 1 do 10, reprezentujące zlecenia. W momencie rozpoczęcia obsługi zlecenia na stanowisku (*Task* nr 2) zmienna *wolne* przyjmuje aż do zakończenia obsługi wartość 0, co blokuje możliwość wprowadzenia następnego zlecenia w tym czasie.

Rozważenie problemu z ograniczoną pojemnością bufora

Załóżmy, że w zdefiniowanym powyżej zadaniu, zlecenia przechodzą po kolei przez dwa stanowiska obsługi, przy czym bufory przed stanowiskami mają ograniczoną pojemność (np. równą jeden – dopóki stanowisko obsługi jest zajęte, co najwyżej jedno zlecenie może czekać w kolejce).
Sytuację taką można zamodelować na kilka sposobów:
- najprostsze podejście polega na zdefiniowaniu zmiennych określających aktualną liczbę zadań w kolejce: dopóki kolejka przed następnym stanowiskiem obsługi jest pełna, nie można rozpocząć obsługi zlecenia na poprzedzającym kolejkę stanowisku. Podejście takie jest nieefektywne, gdyż może powodować niepotrzebne przestoje (jeśli np. czas wykonywania zlecenia na stanowisku poprzedzającym jest o wiele dłuższy niż czas wykonywania zleceń obsługiwanych na kolejnym stanowisku), ponadto nie może być stosowane w bardziej złożonych systemach, składających się również ze stanowisk równoległych.
- w bardziej zaawansowanym podejściu, można wykorzystać zmienną systemową *clock* do porównywania czasów obsługi zadań na poszczególnych stanowiskach i w ten sposób uzależnić rozpoczęcie obsługi zlecenia na danym stanowisku od czasu zakończenia zleceń na kolejnym stanowisku. Podejście takie może być utrudnione w przypadku złożonych systemów.
- modelowanie efektywnego przepływu zleceń przez stanowiska obsługi z ograniczonymi kolejkami można również osiągnąć dokonując odmiennej interpretacji bufora znajdującego się za danym systemem obsługi – można przyjąć, że zlecenie obsłużone na danym stanowisku jest na nim przetrzymywane, dopóki nie opróżni się kolejne stanowisko. Przyjmiemy więc, że stanowisko jest zajęte, dopóki zlecenie, które zostało na nim obsłużone nie opuści kolejki znajdującej się za tym stanowiskiem. Kolejka jest więc interpretowana jako część poprzedzającego ją stanowiska obsługi. Takie podejście zapewnia efektywność i prostotę modelowania nawet złożonych systemów.

5.Uruchomienie symulacji

Uruchomienie symulacji następuje po wywołaniu polecenia *Execute/Go* (klawiszami Ctrl+G albo wciśnięciem przycisku oznaczonego ikonką). Następujące polecenia dostępne w menu *Execute* umożliwiają sterowanie symulacją.

Settings - ustawienie parametrów symulacji.

Pause (Ctrl+P)- wstrzymanie symulacji.

Single-step (Ctrl+T) - pojedynczy krok symulacji (przesunięcie do następnego zdarzenia, a nie o jednostkę czasu).

Top speed - symulacja z maksymalną szybkością (dotyczy polecenia GO).

Faster (F2) - zwiększenie szybkości symulacji.

Normal Speed - ustawienie standardowej szybkości symulacji.

Slower (F1) - zmniejszenie szybkości symulacji.

Halt - zatrzymanie symulacji.

6. Rejestracja i wyświetlanie wyników symulacji

Wyniki symulacji można obserwować na bieżąco w trakcie symulacji (szczególnie w trybie pracy krokowej) lub/i rejestrować je w celu archiwizacji bądź sporządzania wykresów. Najprostszy podgląd aktualnych wartości zmiennych możliwy jest w oknie *Variable Catalog*.

Snapshots

Aby wyniki symulacji zostały zapamiętane lub wyświetlone w postaci wykresów, muszą zostać zarejestrowane w pliku dyskowym. Wymaga to wywołania polecenia *Display/Snapshots* i przyciśnięcia *Add*. Pojawia się okno parametrów danej „migawki”, gdzie należy podać:

- nazwę pliku, w którym będą przechowywane migawki (*Document Name*),
- chwile rejestracji (*Trigger Type*). Można wybrać tryb *Clock*, wtedy rejestracja zaczyna się w chwili podanej w *Trigger at Time* co *Repeating* chwil, aż do chwili *Stop*. Można też wybrać chwile zdarzeń typu: rozpoczęcie operacji (tryb *Begin Task*), zakończenie operacji (tryb *End Task*), wejście znacznika do kolejki (tryb *Enter Queue*), wyjście z kolejki (tryb *Depart Queue),* zakończenie przebiegu symulacji (tryb *End of Run*) – wtedy w polu *Trigger on* trzeba podać numer obiektu, na który ustawiamy migawkę.
- nazwy zmiennych, których wartości mają być zarejestrowane (*Variables to Store*)

Przykład działania migawki, ustawionej dla zmiennych *clock*, *rozp* (rozpoczęto), *zak* (zakończono):
clock = 60, rozp = 2, zak = 0; clock = 120, rozp = 5, zak = 0; clock = 180, rozp = 9, zak = 2

Execution Monitor

Wartości badanych wyrażeń można śledzić używając opcji menu *Display/Execution Monitor*; wyrażenia, które chcemy śledzić, definiuje się po wciśnięciu przycisku *Add*.

Wyświetlanie wyników symulacji

Przed każdą symulacją można decydować o uruchamianiu bądź zaniechaniu rejestracji wyników wywołując z menu polecenie *Execute/Settings* i ustawiając bądź kasując opcje:

- *Trace of Tasks* - zapisuje w pliku o takiej samej nazwie jak model i rozszerzeniu .res: czas początkowy, końcowy i numer znacznika, dla każdej operacji *Task*,
- *Snapshots of Variables*. - zdefiniowane wcześniej „migawki" zapisuje we wcześniej zdefiniowanych plikach, z rozszerzeniem .res (patrz wyżej),
- *Queue Data Collection* - zapisuje wszystkie dane dotyczące kolejek zdefiniowanych w modelu, w pliku o takiej samej nazwie jak model i rozszerzeniu .que

Aby wyświetlić zarejestrowane dane, należy użyć polecenia *File/Open Results*.

Po podaniu nazwy pliku zdefiniowanego przy tworzeniu *Snapshots* mamy następujące możliwości:
Ukazuje się tabelka z wartościami zmiennych zdefiniowanymi wcześniej (patrz wyżej). Dane z tabeli można wyświetlić w postaci wykresu po określeniu myszą, jakie zmienne mają być wyświetlane na osi poziomej i pionowej, a następnie wybraniu rodzaju wykresu poleceniem *Analyze*, czyli: *Scatter Plot* (wykres punktowy), *Step Graph* (punkty x i y połączone pionowo i poziomo), *Line Graph* (połączone linią punkty x i y), *Bar Chart* (wykres słupkowy), *Frequency Distribution* (na osi y zaznaczana jest liczba wystąpień każdej wartości x).
Micro Saint tworzy automatycznie tabelę ze statystykami wartości minimalnych, maksymalnych, średnich i odchylenia standardowego zmiennych zapisanych w plikach wynikowych *Snapshots*. Statystyki wybiera się poleceniem *Analyze/ Statistics*
Po podaniu nazwy pliku z rozszerzeniem .que MicroSaint może analizować standardowe dane dotyczące kolejek:
Ukazuje się tabelka z automatycznie rejestrowanymi zmiennymi: *clock*, *tag*, *run*, *length* (bieżąca liczba znaczników w kolejce), *wait* (okres czasu, jaki dany znacznik spędził w danej kolejce), *trigger* (wskazuje co się wydarzyło). Dane z tabeli można wyświetlić w postaci wykresu po określeniu myszą, jakie zmienne mają być wyświetlane na osi poziomej i pionowej, a następnie wybraniu rodzaju wykresu poleceniem *Analyze* (tak jak w przypadku „migawek"). Można też wykorzystać gotowe wykresy dla kolejek, dostępne po wybraniu polecenia *Analyze/Queue Graphs*.
Micro Saint tworzy też automatycznie tabelę ze statystykami kolejek dla wartości minimalnych, maksymalnych, średnich i odchylenia standardowego zmiennych *lenght* i *wait*. Statystyki wybiera się poleceniem *Analyze/Queue Statistics*.

7. Najczęściej powtarzające się błędy

Micro Saint sprawdza poprawność zbudowanego modelu przed rozpoczęciem symulacji oraz podczas jej trwania. Poniżej przedstawiono najczęstsze błędy użytkowników. Listą tą można się kierować przy samodzielnym usuwaniu błędów w swoich programach (nie wszystkie błędy są wykrywane program!):

- brak praw zapisu – model należy zapisać w swoim katalogu (../users/students/Xnumer_indeksu),
- brak średnika na końcu wyrażenia,

- brak definicji zmiennej użytej w wyrażeniu,
- nieprawidłowy typ decyzji w rozgałęzieniu,
- definicja wartości początkowej zmiennej (np. ustawienie zmiennej definiującej zajętość procesora od początku na 0).