



IF505  
Simulation par  
événements discrets



Louis Granger



ÉCOLE  
POLYTECHNIQUE  
MONTREAL

*Le génie  
sans frontières*


A-306.13  
340-4711-4782  
louis.granger@mail.polymtl.ca

IF505 Simulation par événements discrets

2-2

CHAPITRE

LANGAGE SIMSCRIPT  
II.5



IF505 Simulation par événements discret-- L.Granger--SIM2--2000-10-19-10:3

## SIMSCRIPT II.5 CARACTÉRISTIQUES GÉNÉRALES

- ☐ Langage de programmation algorithmique tout usage  
(General-Purpose Langage)  
(comme FORTRAN, PL/1, PASCAL, C ...)
- ☐ Mécanismes supplémentaires pour la simulation
- ☐ Généralités
  - ☐ langage similaire à l'anglais
  - ☐ format libre
  - ☐ modulaire
  - ☐ conceptuel
  - ☐ compilation séparée
  - ☐ allocation dynamique de mémoire

## SIMSCRIPT II.5 CARACTÉRISTIQUES GÉNÉRALES Langage similaire à l'anglais

- ☐ Chaque énoncé commence par un mot clé  
LET A = B  
ACTIVATE A Car NOW  
START SIMULATION  
SCHEDULE Stop.Simulation IN 10 DAYS  
FOR EACH Train IN Mouvement  
DO  
...  
LOOP
- ☐ Les mots-clés ne sont pas réservés
- ☐ Noms de variables : longueur illimitée (< longueur d'une ligne)
- ☐ Points à la fin : ignorés  
ENTITIES.....
- ☐ Plusieurs énoncés/lignes ou plusieurs lignes/énoncé
- ☐ Pas de colonne ou symbole de continuation

## SIMSCRIPT II.5 CARACTÉRISTIQUES GÉNÉRALES Variables & Constantes

### Variables de système

- Se termine par ".", "Lettre"

|         |                  |
|---------|------------------|
| PL.C    | (Constante)      |
| HOURS.V | (Variable)       |
| SQRT.F  | (Fonction)       |
| TIME.R  | (Routine)        |
| EV.S    | (Ensemble) (Set) |

### Variables définies par l'utilisateur

- Combinaison de lettre, chiffres, et point

Client  
Nb.Client  
X18  
X24.3

### Constantes Entières (Pas de point décimal)

|   |      |       |
|---|------|-------|
| 1 | 1378 | 12054 |
|---|------|-------|

### Constantes Réelles

|     |       |     |
|-----|-------|-----|
| 1.0 | 34.26 | 0.5 |
|-----|-------|-----|

### Constantes Scientifiques

|          |        |
|----------|--------|
| -2.87E-4 | 1.0E+6 |
|----------|--------|

La notation scientifique ne peut être utilisée dans les programmes  
mais seulement dans les fichiers de données



## CARACTÉRISTIQUES GÉNÉRALES Variables & Constantes

### Constante Alpha

"ABCD" (maximum de 4 caractères)

### Constante Texte

"Une chaîne de texte"

### Constante Sous-programme

'sin.f'



## SIMSCRIPT II.5

### STRUCTURE GÉNÉRALE D'UN PROGRAMME

#### PREAMBLE

- Définitions globales
- Variables globales
- Cueillette des statistiques
- Définitions des
  - entités permanentes
  - entités temporaires
  - processus
  - ressources
  - événements
  - routines
  - fonctions
  - ensemble

END " PREAMBLE

#### MAIN

- Variables locales
- Initialisation
- Contrôle des répétitions de simulation  
(START SIMULATION)
- Impression des résultats en fin de simulation

END " MAIN

## SIMSCRIPT II.5

### STRUCTURE GÉNÉRALE D'UN PROGRAMME

ROUTINE Sous.Programme [GIVING Parm.A, Parm.B  
YIELDING Parm.c]

Définition des paramètres et des variables locales

Énoncés exécutables

RETURN [WITH expression] ([] si c'est une fonction)

END " Sous.Programme

EVENT Evenement [ GIVING Parm.A, Parm.B  
YIELDING Parm.c]

Définition des paramètres et des variables locales

Énoncés exécutables

SCHEDULE AN Autre.Eventement IN expression MINUTES

END " Evenement

PROCESS Processus [ GIVING Parm.A, Parm.B  
YIELDING Parm.c]

Définition des paramètres et des variables locales

Énoncés exécutables

ACTIVATE AN Autre.Processus IN expression MINUTES

END " Evenement

## SIMSCRIPT II.5 DÉFINITION DES VARIABLES

```

DEFINE Variable AS [A]
    INTEGER
    REAL
    DOUBLE
    ALPHA
    TEXT
    SIGNED INTEGER
    POINTER
    [entier - DIM]

[SUBPROGRAM] [SAVED] [VARIABLES]
[RECURSIVE] [ARRAYS]
    
```

### Exemple:

```

DEFINE Nombre AS INTEGER VARIABLE
DEFINE No.Interstation
    Statut,
    Direction AS INTEGER VARIABLE
NORMALLY MODE IS UNDEFINED....
DEFINE Coeff AS A 1-DIM REAL ARRAY
DEFINE Fonct AS A SUBPROGRAM VARIABLE
DEFINE Nom.Station,
    Ville AS TEXT VARIABLE
DEFINE Resultat AS AN INTEGER 2-DIM ARRAY
    MONITORED ON THE LEFT AND RIGHT
DEFINE i, j, k AS INTEGER VARIABLE
DEFINE Fonction AS SUBPROGRAM VARIABLE
DEFINE Liste.Train AS A 1-DIM TEXT ARRAY
DEFINE Liste.Tmp AS A POINTER VARIABLE
DEFINE Alpha,
    Theta,
    Phi AS REAL VARIABLES
DEFINE Semence AS A 1-DIM REAL ARRAY
    
```

## SIMSCRIPT II.5 EXPRESSIONS ARITHMÉTIQUES

### Opérateurs

- + (Addition)
- (Soustraction)
- \* (Multiplication)
- / (Division)
- \*\* (Exponentiation)
- () (Evaluation des expressions entre () en premier)

### [LET] Variable = valeur

#### Exemple:

```

LET i = i + 1
LET D = (-B + SQRT.F(B * B - 4 * A * C)) / (2.0 * A)
LET Capacite(Vol) = Premiere.Classe ( Vol) +
    Classe.Touriste(Vol) +
    Classe.Econimique(Vol)
    
```

- Expressions mixtes : évaluées dans le mode le plus précis
- Division toujours réelle ( pas d'entier)
- Expressions réelles arrondies si résultat affecté à une variable entière

## SIMSCRIPT II.5 BOUCLE

```

DO
.....
LOOP
FOR Variable { = } quantié TO quantié [BY quantié][.]
FOR EACH { Entité _Permanente } [CALLED Variable_ Pointeur][.]
      Re ssource
FOR EACH Pointeur_ Entité { FROM } Pointeur_ Entité } OF Ensemble
      [AFTER]
      [IN REVERSEORDER][.]
    
```

```

Exemple:
FOR i = 1 TO n,
  FOR j=1 TO m
  DO
  .....
  LOOP

FOR x = -12.0 TO 23 BY 0.234
DO
....
LOOP
    
```

## SIMSCRIPT II.5 BOUCLE

### Exemple:

```


FOR EACH Train OF LIGNE(1) IN REVERSE ORDER
  WITH Status = .En.Feu
  DO
  REMOVE Train FROM LIGNE(1)
  DESTROY Train
  LOOP

FOR EACH Vol OF Depart
  DO
  REQUEST 1 Piste(Aeroport)
  WORK 1.2 MINUTES
  RELINQUISH 1 Piste(Aeroport)
  LOOP

FOR EACH Piste AFTER 6 OF Piste.Atterissage,
FOR EACH Aeroport
FOR EACH Vol OF Depart IN REVERSE ORDER
  UNLESS Destination(Vol) = "Mexico"
FOR i = 0 TO 2 * (N-1) BY 2 * Delta.i
    
```

## SIMSCRIPT II.5 BOUCLE UNTIL

[ALSO] UNTIL Expression\_logique <sup>{AND}</sup> OR {,}

 Exemple:

```
FOR i = 1 TO n, UNTIL x(i) = y(i)
DO
.....
LOOP


UNTIL (Vue NE Vue.Courante OR Contordre = .Oui)
DO
CALL Entrer.Point
CALL Afficher.Point
LOOP

UNTIL LIGNE(2) IS EMPTY
DO

LOOP
```

## SIMSCRIPT II.5 BOUCLE WHILE

[ALSO] WHILE Expression\_logique <sup>{AND}</sup> OR {,}


 Exemple:

```
FOR i = 1 TO n, WHILE x(i) >= 0.0
DO
.....
LOOP

WHILE (Service.Time > Quantum )
DO
REQUEST 1 CPU(1)
WORK (Quantum+Overhead) .Seconds
LET Service.Time = Service.Time - Quantum
RELINQUISH 1 CPU(1)
LOOP
```

## SIMSCRIPT II.5 Énoncé conditionnel IF

```
[THEN] IF Expression_logique[,
    [Énoncé1]
[ELSE
    [Énoncé2]
ENDIF
```

 **Exemple:**

```
IF X(i) < 0.0,
    i = i + 1
ENDIF
```

```
IF X(i) < 0.0,
    i = i + 1
    k = k + 1
    FOR j = i TO k,
        DO
        ....
    LOOP
ENDIF
```

## SIMSCRIPT II.5 Énoncé conditionnel IF

 Exemple

```
IF (Effet.Pente > Effet.Resistance),
    Call Freiner
ELSE
    Frein = Mecanique " Passer en freinage mecanique
    IF Acceleration.Moteur < Effet.Pente,
        Energie(Train) = Energie1 * Delta.T / 3600.0 " Kwh
    ELSE
        Energie(Train) = Energie1 * (Effet.Resistance +
            Effet.Pente) / (Acceleration.Moteur +
            Effet.Resistance) * Delta.T / 3600.0 " Kwh
    ENDIF
ENDIF
```

## SIMSCRIPT II.5 CASE

*SELECT CASE* *variable*  
[*CASE* *constante<sup>c</sup>*  
*énoncés*]

[*CASE* *constante<sup>c</sup>*  
*énoncés*]

[*DEFAULT*  
*énoncé*]

*ENDSELECT*

☐ Exemple

```
SELECT CASE Direction
CASE 1, 2, 3
.....
.....
CASE 4
.....
.....
ENDSELECT
```

## SIMSCRIPT II.5 PARTICULARITÉS POUR LA SIMULATION

- ☐ Modélisation du monde réel
  - PROCESS
  - RESOURCE
  - ENTITIES
  - EVENTS
  - SETS
- ☐ Mécanismes pour une horloge simulée
  - ☐ Représentation du temps
- ☐ Mécanismes pour mesurer les performances ou recueillir des statistiques
  - TALLY
  - ACCUMULATE
- ☐ Mécanismes pour structurer les données
- ☐ Mécanismes pour générer des rapports

## SIMSCRIPT II.5 MODÉLISATION DU MONDE RÉEL

- ☐ PROCESS  
Un processus contient la description d'un objet et la séquence de ses activités qu'il effectuera pendant sa "vie" dans une simulation.
- ☐ RESOURCE  
Une ressource décrit un objet qui donne un service à un autre objet (actuellement traité), retardant s'il y a lieu l'objet s'il est occupé à servir un autre objet.
- ☐ ENTITIES  
Une entité modélise un objet passif qui se déplace dans le système.
- ☐ EVENTS  
Processus instantané ou actions qui doivent être effectuées au début ou à la fin d'une activité
- ☐ SETS  
Liste ordonnée d'objets : entités, processus, ou ressources.

## SIMSCRIPT II.5 MODÉLISATION DU MONDE RÉEL

- ☐ Un PROCESS modélise un objet (statique) et représente sa "vie" dans le système.
- ☐ Une simulation peut contenir:
  - ☐ plusieurs copies du même PROCESS
  - ☐ plusieurs PROCESS différents
- ☐ Les PROCESS interagissent en :
  - ☐ changeant l'état du système
  - ☐ compétitionnant pour les ressources

☐ Exemple:

```
PROCESS Generateur
FOR i = 1 to 100
DO
ACTIVATE 1 Appel NOW
WAIT UNIFORM.F ( 2.0, 6.0, 1) MINUTES
LOOP
END ** Generateur
```

## SIMSCRIPT II.5 MODÉLISATION DU MONDE RÉEL

ACTIVATE { THE [ABOVE] } { Processus } { CALLED Variable\_pointeur }

A

{ Evé nement }

[ GIVEN Valeur<sup>c</sup> ]

( valeur )<sup>c</sup>

AT quantité

NOW

IN quantité { DAY[S] }

{ HOUR[S] }

{ MINUTE[S] }

{ WAIT } quantité { DAY[S] }

{ WORK } { HOUR[S] }

{ MINUTE[S] }

☐ Exemple:

```
ACTIVATE A Job NOW
ACTIVATE A Changement.Releve IN 8 HOURS
ACTIVATE A Client IN 5 MINUTES
WAIT 0.5 MINUTES
WORK UNIFORM.F(A, B, 1) HOURS
```

## SIMSCRIPT II.5 HORLOGE SIMULÉE REPRÉSENTATION DU TEMPS

☐ Unités de temps

DAYS (UNITS)  
HOURS (HOURS.V = 24)  
MINUTES (MINUTES.V = 60)

☐ De façon interne, le temps est conservé en jour dans une variable

TIME.V (réelle) (horloge du système)

☐ Redéfinition des unités de temps

☐ Exemple :

```
PREAMBLE
DEFINE .Secondes TO MEAN DAYS
DEFINE .MiliSecondes TO MEAN HOURS
DEFINE .MicroSecondes TO MEAN MINUTES
```

END " PREAMBLE

MAIN

```
LET HOURS.V = 1000
LET MINUTES.V = 1000
```

END " MAIN

## SIMSCRIPT II.5 EXEMPLE COMPLET SYSTÈME TÉLÉPHONIQUE

- ☐ Supposons que l'on veut évaluer les pertes d'une compagnie à cause de son système téléphonique inadéquat. Les appels arrivent aléatoirement. Si une ligne est libre, on répond à l'appel, sinon l'appel est perdu.

Les inter-arrivées des appels et la durée du service suivent une distribution uniforme avec les paramètres suivants:

Arrivée: (4 +- 2 minutes)

i.e. un minimum de 2 minutes entre chaque appel et un maximum de 6 minutes.

Durée de (8 +- 2 minutes)

l'appel:

- ☐ On veut mesurer :
  - ☐ le nombre d'appel servis (répondus)
  - ☐ le nombre d'appel perdus
  - ☐ l'utilisation du système téléphonique

- ☐ Expérimentation :

- ☐ optimiser le nombre de lignes
- ☐ réduire le temps de service

- ☐ **FORMULATION DU PROBLÈME**

- ☐ Mécanismes pour recevoir un appel
- ☐ Mécanismes pour le comportement d'un appel dans le système
- ☐ Mécanismes pour le terminer la simulation



## SIMSCRIPT II.5 EXEMPLE COMPLET SYSTÈME TÉLÉPHONIQUE

- ☐ Mécanismes pour recevoir un appel

PROCESS Generateur

FOR i = 1 to 100

DO

    ACTIVATE 1 Appel NOW

    WAIT UNIFORM.F ( 2.0, 6.0, 1) MINUTES

LOOP

END " Generateur

OU

PROCESS Ceduleur

UNTIL TIME.V >= Heure.Fermeture

DO

    ACTIVATE AN Appel NOW

    WAIT UNIFORM.F ( 2.0, 6.0, 1) MINUTES

LOOP

END " Ceduleur



## SIMSCRIPT II.5 EXEMPLE COMPLET SYSTÈME TÉLÉPHONIQUE

☞ Mécanismes pour le comportement d'un appel dans le système

```
PROCESS Appel
  IF Nb.Occupe < 2
    ADD 1 TO Nb.Occupe
    WORK UNIFORM.F ( 6.0, 10.0, 2) MINUTES
    SUBTRACT 1 FROM Nb.Occupe
  ELSE...
    ADD 1 TO Clients.Perdus
  ENDIF
END " Appel
```

## SIMSCRIPT II.5 EXEMPLE COMPLET SYSTÈME TÉLÉPHONIQUE

☞ Préambule et programme principal

```
PREAMBLE
  PROCESS INCLUDE...
    Generateur AND
    Appel

  DEFINE Nb.Occupe
    Clients.Perdus AS INTEGER VARIABLES

END "Preamble

MAIN
  ACTIVATE A Generateur NOW
  START SIMULATION
END " Main
```

## SIMSCRIPT II.5 EXEMPLE COMPLET SYSTÈME TÉLÉPHONIQUE

- ☑ Mécanismes pour le terminer la simulation
- ☆ Il n'y a plus rien à faire.  
Aucun événements ou processus à traiter.

```

PROCESS Generateur
  FOR i = 1 to 100
  DO
    ACTIVATE 1 Appel NOW
    WAIT UNIFORM.F ( 2.0, 6.0, 1) MINUTES
  LOOP
END
    
```

- ⌚ Arrêter à un temps prédéfini

```

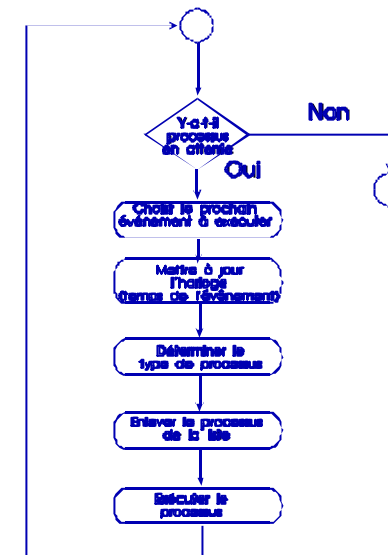
PROCESS Ceduleur
  UNTIL TIME.V >= Heure.Fermeture
  DO
    ACTIVATE AN Appel NOW
    WAIT UNIFORM.F ( 2.0, 6.0, 1) MINUTES
  LOOP
END
    
```

- ⌚ Une combinaison des deux (Arrêter les arrivées et servir ceux qui reste)

## SIMSCRIPT II.5 EXÉCUTION

- ☑ La TIMING routine et la liste des processus sont le coeur d'une exécution SIMSCRIPT

### SIMULATION



## SIMSCRIPT II.5 LISTE DES PROCESSUS

- ☐ L'activation d'un processus crée une "notice" pour ce processus. Il s'agit d'un bloc de données décrivant l'instance du processus.
- ☐ Il y a plusieurs notices de processus dans le système en même temps. Tous sont dans une liste.
- ☐ Dans les diagrammes qui suivent, chaque rectangle représente une notice d'un processus avec l'information suivante:

|  |
|--|
|  |
|  |
|  |

Nom du processus  
Temps où il doit être activé  
Prochaine ligne à exécuter  
Information de système



## SIMSCRIPT II.5 LISTE DES PROCESSUS

|                 |                      |                      |                      |
|-----------------|----------------------|----------------------|----------------------|
| Début           | GEN1<br>0.0<br>1     |                      |                      |
| Premier appel   | APPEL1<br>0.0<br>1   | GEN1<br>2.547<br>7   |                      |
| Deuxième appel  | GEN1<br>2.547<br>7   | APPEL1<br>7.142<br>5 |                      |
| Troisième appel | APPEL2<br>2.547<br>1 | GEN1<br>5.5<br>7     | APPEL1<br>7.142<br>5 |
| Quatrième appel | APPEL1<br>5.5<br>7   | APPEL1<br>7.142<br>5 | APPEL2<br>8.547<br>5 |



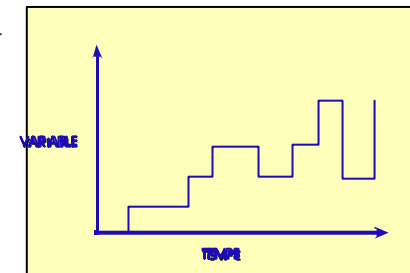
## SIMSCRIPT II.5 LISTE DES PROCESSUS

|                        |                     |                      |                      |                      |
|------------------------|---------------------|----------------------|----------------------|----------------------|
| <b>Quatrième appel</b> | APPEL3<br>5.5<br>1  | APPEL1<br>7.142<br>5 | APPEL1<br>8.25<br>7  | APPEL2<br>8.547<br>5 |
| <b>Sixième appel</b>   | GEN1<br>7.142<br>5  | GEN1<br>8.25<br>7    | APPEL2<br>8.547<br>7 |                      |
| <b>Septième appel</b>  | GEN1<br>8.25<br>7   | APPEL2<br>8.547<br>5 |                      |                      |
| <b>Huitième appel</b>  | APPEL4<br>8.25<br>7 | APPEL2<br>8.547<br>5 | GEN1<br>13.5<br>7    |                      |

## SIMSCRIPT II.5 STATISTIQUES (MESURES DU SYSTÈME)

ACCUMULATE } { Nom { AS [THE][nom]not - clé - statistique } }<sup>e</sup>  
 TALLY } { (quantité TO quantité BY quantité) AS [THE] HISTOGRAM }

OF { variable globale non - indicé e  
 attribut d'une entité  
 attribut du système (non - indicé) }



Exemple

ACCUMULATE Nb.Auto AS THE NUMBER,  
 Max.Auto AS THE MAXIMUM OF Nb.AUTO.Stationnees

ACCUMULATE Max.Auto.Total AS THE MAXIMUM,  
 Max.Auto AS THE WEEKLY MAXIMUM,  
 (50 TO 1000 By 50) AS THE HISTOGRAM  
 OF Nb.Auto.Stationnees

## SIMSCRIPT II.5 STATISTIQUES (MESURES DU SYSTÈME)

☞ Exemple

```

ACCUMULATE V.Moy      AS THE MEAN,
V.Ecart              AS THE STD.DEV,
V.Min               AS THE MINIMUM,
V.Max               AS THE MAXIMUM,
V.Nombre            AS THE NUMBER,
V.Frein (Lim.Inf.VFrein TO Lim.Sup.VFrein
BY Incr.Frein)
                    AS THE HISTOGRAM OF
V.Debut.Freinage
ACCUMULATE Longueur.File AS THE MEAN,
Fil.Max             AS THE MAXIMUM OF
N.Q.Autobus
ACCUMULATE Utilisation AS THE AVERAGE OF
N.X.Autobus
    
```



## SIMSCRIPT II.5 STATISTIQUES (MESURES DU SYSTÈME)

| Statistiques   | Calcul avec                             | Calcul avec                   |
|----------------|---|-------------------------------|
| Mot-clé        | ACCUMULATE                              | TALLY                         |
| NUMBER         | N=nombre de changement de X             | N=Nombre d'échantillon de x   |
| SUM            | $\sum_{i=1}^n x_i (TIME.V - T_i)$       | $\sum_{i=1}^n x_i$            |
| MEAN AVERAGE   | $\frac{SUM}{(TIME.V - T_0)}$            | $\frac{SUM}{N}$               |
| SUM.OF.SQUARES | $\sum_{i=1}^n x_i^2 (TIME.V - T_i)$     | $\sum_{i=1}^n x_i^2$          |
| MEAN.SQUARE    | $\frac{SUM.OF.SQUARES}{(TIME.V - T_0)}$ | $\frac{SUM.OF.SQUARES}{N}$    |
| VARIANCE       | MEAN.SQUARE-MEAN <sup>2</sup>           | MEAN.SQUARE-MEAN <sup>2</sup> |
| STD.DEV        | $\sqrt{VARIANCE}$                       | $\sqrt{VARIANCE}$             |
| MAXIMUM        | M=Maximum (x) pour tous les x           | M=Maximum (x) pour tous les x |
| MINIMUM        | M=Minimum (x) pour tous les x           | M=Minimum (x) pour tous les x |

$T_L$  = temps où la variable a pris sa valeur courante

$T_0$  = dernier temps où la variable a changé de valeur



## SIMSCRIPT II.5 AJOUT DE SORTIES

*PRINT entier [DOUBLE] LINE[S] WITH  $\left. \begin{array}{l} \text{variable} \\ \text{A GROUP OF valeur - entiere}^c \text{ FIELDS} \end{array} \right\}^c$*   
*[SUPPRESSING FROM COLUMNentier] THUS*

☐ Exemple

PRINT 1 LINE THUS

Rapport hebdomadaire pour les vols

PRINT 3 LINES AS FOLLOWS

Rapport hebdomadaire pour les vols

|           |     |      |           |
|-----------|-----|------|-----------|
| Compagnie | Vol | Date | Passagers |
|-----------|-----|------|-----------|

PRINT 1 LINE WITH Annees, Mois, Jour THUS

19\*\*\_\*\*\_\*\*

PRINT 3 LINE WITH Tarif, Distance, Distance/HOURS

THUS

Tarif est \$\*\*\*\*.\*\*

Distance est \*\*\*\* km

Moyenne est \*\*\*\* km par heure

PRINT 1 DOUBLE LINE THUS

FOR EACH Ville, PRINT 1 LINE WITH Nom(Ville),

Population(Ville) AND Superficie (Ville) THUS

\*\*\*\*\*                      \*\*\*\*\*                      \*\*\*\*\*

## SIMSCRIPT II.5 AJOUT DE SORTIES

☐ Correspondance une à une entre les variables et les formats

☐ Les formats sont constitués de \* et de .

☐ Le texte est copié tel quel.

☐ Le nombre de lignes imprimées est fonction du compteur de lignes.

☐ On ajoute des interlignes par

SKIP n LINES

☐ Deux lignes de format avec l'option DOUBLE

PRINT 1 DOUBLE LINE THUS

## SIMSCRIPT II.5 RESSOURCE

- ☐ Une ressource modélise un objet passif qui:
  - ☐ Fournit un service aux objets traités,
  - ☐ Retarde l'objet traité lorsque la ressource est prise par un autre objet,
  - ☐ Automatiquement redémarre les processus lorsque des ressources deviennent disponibles.
- ☐ Une simulation peut contenir
  - ☐ Plusieurs ressources différentes
  - ☐ Plusieurs unités de chaque ressource pour une seule file d'attente de processus.
- ☐ Traitement des ressources
  - REQUEST 1 Pompe (1)
  - REQUEST 5 Travailleurs
  - RELINQUISH 1 Piste.Atterissage

## SIMSCRIPT II.5 RESSOURCE

```
REQUEST Valeur_entiere [UNIT[S] OF] ressource [(valeur_entiere)]  
[.] [[WITH PRIORITY valeur_entiere]  
RELINQUISH Valeur_entiere [UNIT[S] OF] ressource [(valeur_entiere)]
```

## SIMSCRIPT II.5 ALLOCATION DES RESSOURCES

- 1 Doivent être déclarées dans le PREAMBLE  
PREAMBLE

```
RESSOURCE...
INCLUDE CPU AND Memory
END " PREAMBLE
```

- 2 Les ressources doivent être créées avant leur utilisation  
ROUTINE Initialisation

```
.
CREATE EVERY CPU(1) ou LET N.CPU = 1
CREATE EVERY CPU
```

```
LET U.CPU(1) = 1
```

```
LET N.Memory = 2
CREATE ALL Memory
```

```
LET U.Memory (1) = 12000 " Mémoire rapide
LET U.Memory (2) = 320000 "Mémoire lente
```

```
ACTIVATE A Job NOW
```

```
.
PROCESS Job
REQUEST 1 CPU (1)
REQUEST 5000 UNITS OF Memory (1)
WORK...
```

```
END"Job
```



## SIMSCRIPT II.5 ALLOCATION DES RESSOURCES

- 1 Doivent être déclarées dans le PREAMBLE
- 2 Les ressources doivent être créées avant leur utilisation

Automatiquement, un certain nombre de variables sont attachées à une ressource:

N.Resource: variable globale contenant le nombre de "sortes" de ressources,

U.Resource(i): variable globale contenant le nombre d'unités disponibles de la Ressource(i),  
REQUEST/RELINQUISH affecte la valeur de U.Ressource(i),

N.X.Resource(i): variable globale contenant le nombre de requêtes satisfaites pour la Ressource(i),

N.Q.Resource(i): variable globale contenant le nombre de requêtes en attente pour la Ressource(i).  
(non-remplies).



## SIMSCRIPT II.5 PLUS SUR LES RESSOURCES

☞ A la création d'une ressource, deux ensembles sont créés:

- X.Resource --processus utilisant la ressource
- Q.Resource --processus en attente de la ressource ordonné selon PTY.A (Priorité)

+ une notice contenant:

|                               |   |
|-------------------------------|---|
| WHO.A                         | Pointeur au processus exécutant ou en attente de la ressource.  |
| QTY.A                         | Nombre d'unités de la ressource utilisée par le processus.  |
| PTY.A                         | Priorité de la requête  |
| P.RS.S                        | Prédécesseur (pointeur) dans l'ensemble des processus (RS.S)  |
| S.RS.S                        | Successeur (pointeur) dans l'ensemble des processus (RS.S)  |
| P.Q. Resource ou P.X Resource | Prédécesseur (pointeur) dans l'ensemble des processus en attente de satisfaction d'une requête ou utilisant la ressource (Q.Resource ou X.Resource) |
| S.Q Resource ou S.X Resource  | Successeur (pointeur) dans l'ensemble des processus en attente de satisfaction d'une requête ou utilisant la ressource (Q.Resource ou X.Resource)   |

## SIMSCRIPT II.5 ALLOCATION DES RESSOURCES

- 1 Doivent être déclarées dans le PREAMBLE  
 PREAMBLE  
 RESOURCES...  
     EVERY Caissier HAS A Temps.Service RANDOM LINEAR VARIABLE  
     DEFINE Temps.Service AS A REAL STREAM 6 VARIABLE  
 END "PREAMBLE"
  - 2 Les ressources doivent être créées avant leur utilisation ROUTINE Initialisation  
 :  
     READ N.Caissier  
     CREATE EVERY Caissier  
     FOR EACH Caissier,  
         READ U.Caissier(Caissier), Temps.Service(Caissier)
- ```

PROCESS Client
    DEFINE i AS AN INTEGER VARIABLE
    ..
    REQUEST 1 Caissier (i)
    WORK...
    RELINQUISH 1 Caissier (i)
END "Client"
    
```

## SIMSCRIPT II.5 EXEMPLE COMPLET STATION SERVICE

### Cas 2

Des clients arrivent aléatoirement à une station d'essence. Si les préposés sont occupés, les clients attendent jusqu'à ce qu'un préposé soit disponible. Une fois servi, le client quitte la station. Supposons que les arrivées sont distribuées uniformément avec un minimum de 2 minutes et un maximum de 8 ( $5 \pm 3$ ). Le temps de service est aussi uniforme ( $10 \pm 5$ ).

Mesures d'intérêt:

- longueur moyenne de la file d'attente,
- longueur maximum de la file d'attente,
- utilisation des préposés.

Expérimentation devraient conduire à:

- optimiser le nombre de préposés,
- varier le patron d'arrivée,
- réduire le temps de service.



## SIMSCRIPT II.5 EXEMPLE COMPLET STATION SERVICE

### FORMULATION DU MODÈLE

Dans ce système, les clients sont considérés comme des éléments dynamiques et les préposés comme des éléments passifs. Il est naturel de représenter les clients comme des PROCESS qui demandent un service effectué par un préposé (RESOURCE). La file d'attente pour les ressources est automatiquement créée. Les mesures ou statistiques désirées sont obtenues des variables du système -- le nombre dans la file d'attente et le nombre dans les ressources.



## SIMSCRIPT II.5 EXEMPLE COMPLET STATION SERVICE

Cas 2

### PREAMBLE

PROCESS INCLUDE...

    Generateur AND

    Client

RESOURCE INCLUDE Prepose

ACCUMULATE   Longueur.moy.file AS THE AVERAGE,

                  Longueur.max.file AS THE MAXIMUM

                                  OF N.Q.Prepose

ACCUMULATE Utilisation AS THE AVERAGE

                                  OF N.X.Prepose

### MAIN

CREATE EVERY Prepose (1) \*\* Créer le serveur

LET U.Preposé = 2           \*\* constitué de 2 préposés

ACTIVATE A Generateur NOW \*\* Première arrivée

START SIMULATION           \*\* On simule

PRINT 4 LINES WITH Longueur.moy.file (1),

                                  Longueur.max.file (1), Utilisation (1) / 2.0

                                  THUS

Modèle d'une station service avec 2 préposés

La longueur moyenne de la file est \*.\* \*\*

La longueur maximum de la file est \*\*\*\*

Les préposés sont occupés \*.\* \*\*% du temps

END \*\* Main

## SIMSCRIPT II.5 EXEMPLE COMPLET STATION SERVICE

PROCESS Generateur

    FOR i = 1 to 100

    DO

        ACTIVATE 1 Client NOW

        WAIT UNIFORM.F (2.0, 8.0, 1) MINUTES

    LOOP

    END

PROCESS Client

    REQUEST 1 Prepose (1)

    WORK UNIFORM.F (5.0, 15.0, 2) MINUTES

    RELINQUISH 1 Prepose (1)

END

Modèle d'une station service avec 2 préposés

La longueur moyenne de la file est 7.809

La longueur maximum de la file est 21

Les préposés sont occupés 0.99% du temps.

## SIMSCRIPT II.5 STRUCTURE DE DONNÉES

☰ Les éléments de base sont:

Entités temporaires

(semblables aux processus)

Entités permanentes

(semblables aux ressources)

Ensembles (Sets)

(collection d'entités)



## SIMSCRIPT II.5 STRUCTURE DE DONNÉES ENTITÉS

☰ Les entités incluent:

Entités temporaires

Processus

Entités permanentes

Ressources

☰ Une entité peut:

Avoir des attributs

Appartenir à un ensemble

Posséder un ensemble



## SIMSCRIPT II.5 NOM DES ENTITÉS

- ☐ Chaque entité a un "Nom" et une variable du même nom.
- ☐ La variable "pointe" vers l'entité.
- ☐ Dans certains cas, le nom générique doit être utilisé.
- ☐ Dans d'autres cas, n'importe quelle variable peut être utilisée pour référencer l'entité.
- ☐ Exemple:

```
TEMPORARY ENTITIES.....  
EVERY Bateau HAS A Equipage,  
A Tonnage  
EVERY Homme HAS A Voiture, AND  
A Maison  
EVERY Tache HAS A No. Machine,  
A Temps.Usinage,  
AND BELONGS TO A  
Liste.Taches
```



## SIMSCRIPT II.5 ENTITÉS

- ☐ Exemple:

```
PERMANENT ENTITIES.....  
EVERY Train HAS A Nb.Passagers,  
A Vitesse,  
A No.Interstation,  
AND BELONGS TO Mouvement,  
Arret.En.Station,  
Garage  
EVERY Avion HAS A Capitaine  
A No.Vol,  
A Origine,  
A Destination  
AND OWNS A Liste.Passagers  
MAIN  
READ N.Train  
CREATE ALL Train  
FOR EACH Train  
DO  
LET Vitesse(Train) = Calculer.Vitesse (TIME.V)  
LET No.Interstation = ...  
LOOP
```



## SIMSCRIPT II.5 RESSOURCES

```

Exemple:
PREAMBLE
  RESOURCE.....
    EVERY Pompe HAS A Type.Essence,
      A Reserve
    EVERY Pompiste HAS A Experience,
      A Sexe

  DEFINE .Sans.Plomb      TO MEAN 1
  DEFINE .Regulier       TO MEAN 2
  DEFINE .Super          TO MEAN 3

END "Preamble
MAIN
  LET N.Pompe = 4
  CREATE EVERY Pompe
  FOR EACH Pompe
  DO
    READ U.Pompe (Pompe), Type.Essence (Pompe),
      Reserve (Pompe)

  LOOP
  :
  FOR EACH Pompe
    WITH Type.Essence (Pompe) = .Sans.Plomb
    AND Reserve (Pompe) >= 100.0

  DO
  :
  LOOP
  
```

## SIMSCRIPT II.5 ENTITÉS PROCESSUS

```

Exemple:

PREAMBLE

PROCESSES
  EVERY Message HAS A Origine,
    AND A Destination

END "Preamble

MAIN

  ACTIVATE A Message GIVING MTL, LA NOW

END "Main

PROCESS Message GIVEN Origine, Destination
  DEFINE Origine, Destination AS INTEGER VARIABLE

END " Message
  
```

## SIMSCRIPT II.5 ENTITÉS PROCESSUS INITIALISATION DES ATTRIBUTS

☞ Lors de la création d'entités (activation d'un processus), les attributs sont initialisés à zéro.

☞ Il y a deux façons d'affecter des valeurs à des attributs:

MAIN

```
    ACTIVATE A Message GIVEN MTL, LA NOW
END " Main
```

ou

MAIN

```
    ACTIVATE A Message NOW
    LET Origine (Message) = MTL
    LET Destination (Message) = LA
```

END " Main

☞ (Assigner les valeurs dans l'ordre ou elles sont déclarées dans le PREAMBLE)

## SIMSCRIPT II.5 ENTITÉS PROCESSUS RÉFÉRENCE DE PLUSIEURS INSTANCES

☞ On peut référencer plusieurs instances d'un processus en les identifiant par un pointeur.

☞ Exemple:

```
    ACTIVATE A Message GIVING MTL, LA NOW
    ACTIVATE A Message CALLED Premier
        GIVING DRUMMOND, QUE NOW
```

```
    ACTIVATE A Message CALLED Dernier
        GIVING DRUMMOND, QUE NOW
```

☞ Référence des attributs

```
    LET Origine (Dernier) = .....
```

## SIMSCRIPT II.5 MÉCANISME DE RECHERCHE

```

FIND {
    THE FIRST CASE
    {variable = [THE][FIRST] valeur}c }{.}
    [THEN] IF {FOUND} {[.]}
            [ELSE
            é noncé }
    ENDIF
    
```

Exemple:

```

FOR EACH Pompe
    WITH Type.Essence (Pompe) = .Sans.Plomb
        AND Reserve (Pompe) > = 100.0
    FIND THE FIRST CASE
    IF FOUND,
        LET Reserve (Pompe) = Reserve (Pompe) - LE.Plein(Auto)
    ELSE
        :
    ENDIF
FOR EACH Ville
    WHEN Population (Ville) > 1000000,
    FIND Grande.Ville = Ville
    IF FOUND,
        LET Subvention(Grande.Ville) = ...
    ENDIF
    
```

## SIMSCRIPT II.5 MÉCANISME DE CALCUL

COMPUTE {nom AS mot\_clé\_stastique}<sup>c</sup> OF variable

mot-clé-stastiques: même que dans les énoncés TALLY et ACCUMULATE

Exemple:

□ Trouver le plus courte file d'attente:

```

FOR EACH Pompe,
    COMPUTE Plus.COURTE.File AS THE MINIMUM OF
        N.Q.Pompe (Pompe)
    Après la boucle, Plus.Courte.File contient la longueur de
    la file minimum
    
```

□ Trouver le numéro de la plus courte file d'attente:

```

FOR EACH Pompiste,
    COMPUTE Meuilleur.Pompiste AS THE MINIMUM
        (Pompiste) OF N.Q.Pompiste (Pompiste)
    Après la boucle, Meilleur.Pompiste contient le numéro de
    la plus courte file d'attente de pompiste
    REQUEST 1 UNIT OF Pompiste (Meilleur.Pompiste)
    
```

## SIMSCRIPT II.5 MÉCANISME DE RECHERCHE

☞ Exemple:

- ☐ Trouver le minimum et maximum ainsi que leurs numéros

```
FOR EACH Vol IN Départ,
  WITH Temps.Départ < 1200,
    COMPUTER Passenger.Max AS THE MAXIMUM,
      Passenger.Min AS THE MINIMUM,
      Vol.MAx AS THE MAXIMUM (Vol)
      Vol.Min AS THE MINIMUM (Vol) OF
      Nb.Passager (Vol))
```

Après la boucle,

|              |                                                                  |
|--------------|------------------------------------------------------------------|
| Vol.Max      | contient le numéro de vol avec le plus grand nombre de passagers |
| Passager.Max | contient le nombre de passagers maximum dans le vol Vol.Max      |
| Vol.Min      | contient le numéro de vol avec le plus petit nombre de passagers |
| Passager.Min | contient le nombre de passagers minimum dans le vol Vol.Min      |



## SIMSCRIPT II.5 CPU

```
"      cpu.sim
"
" Soit une compagnie possédant un ordinateur central (CPU)
" avec n terminaux. L'utilisateur du terminal réfléchit pendant en
" moyenne 25 secondes, durée distribuée selon une
" distribution exponentielle, et envoie une tâche à l'ordinateur
" qui l'exécute en un temps moyen de 0.8 secondes selon une
" distribution exponentielle. Les tâches entrent dans une file
" d'attente et sont servies selon la méthode du round-robin
" plutôt que FIFO. Le CPU alloue à chaque tâche une tranche
" de 0.1 secondes en plus d'un overhead de 0.015 secondes.
" Cette méthode permet de servir plus rapidement les plus
" petites tâches que les tâches demandant des ressources
" importantes. Si le temps de réponse est défini comme le
" temps ^Bcoul^B entre le moment où la tâche a été soumise
" et le moment où elle se termine, pour n=30, 35, ..., simuler
" le système pour 1000 tâches et recueillir les statistiques sur
" le nombre moyen de tâches en attente, et l'utilisation du
" CPU. On assume que tous les opérateurs réfléchissent au
" temps 0.0. La compagnie veut savoir combien utiliser de
" terminaux pour maintenir un temps de réponse inférieur à 30
" secondes.
"
```



## SIMSCRIPT II.5 EXEMPLE COMPLET

```
"          cpu1.sim
"
PREAMBLE
  NORMALLY MODE IS UNDEFINED...
  LAST COLUMN IS 80
  RESOURCES INCLUDE CPU AND Memory
  PROCESSES INCLUDE Generateur AND Fin.Simulation
  EVERY Job HAS A Jb.Priorite
    A Jb.Besoin.Mémoire
  DEFINE Jb.Priorite
    Jb.Besoin.Mémoire AS AN INTEGER VARIABLE
  DEFINE Job.Delay.Time AS A REAL VARIABLE
  EXTERNAL PROCESS IS Job
  EXTERNAL PROCESS UNIT IS 7
  DEFINE Small.Job.Interarrival.Time,
    Mean.Small.Job.Processing.Time,
    Run.Length,
    Stop.Time          AS REAL VARIABLE
  DEFINE Max.CPU,
    Max.Memory        AS INTEGER VARIABLES
  DEFINE Max.Memory.Queue TO MEAN 1Max.Memory.Queue
```

## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

```
ACCUMULATE CPU.Utilisation AS THE AVERAGE OF N.X.CPU
ACCUMULATE Memory.Utilisation AS THE AVERAGE OF N.X.Memory
ACCUMULATE Avg.CPU.Queue AS THE AVERAGE,
  Max.CPU.Queue AS THE MAXIMUM OF N.Q.CPU
ACCUMULATE Avg.Memory.Queue AS THE AVERAGE
  Max.Memory.Queue AS THE MAXIMUM OF N.Q.Memoru
TALLY Avg.Job.Time AS THE AVERAGE,
  No.Jobs.Processed AS THE NUMBER OF Job.Delay.Time
  DEFINE HOURS TO MEAN UNITS
END " PREAMBLE
```

## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

MAIN

```

*****
*****
“ Fonction : Fait lire les données, sauvegarde les semences des
“ générateurs de nombres aléatoires, et déclenche les répétitions
“ des simulations pour un certain nombre de terminaux en
“ réinitialisant les variables tout en s'assurant que les mêmes
“ suites de nombres aléatoires sont utiliser pour les différentes
“ répétitions. *
"* Usage : Appel de main *
"* Entrée : Num : Nombre de suite de nombres aléatoires *
"* Retour : *
"* E/S : Aucune *
"* Portable: SIMSCRIPT II.5 (IBM 370 & IBM PC (Dos) & SUN) *
"* Notes : *
"* Statut : Production *
"* Creation: 06-92 Granger Louis & CACI *

*****
*****
CALL Lire.Donnees
Call ConserverSemence(2)
FOR Nb.Terminaux = Min.Terminals TO Max.Terminals
  BY Inc.Nb.Terminaux
  DO

```

CALL Initialiser(2) " Initialiser variables

START SIMULATION " Simuler ÉCOLE POLYTECHNIQUE MONTREAL  
IF505 Simulation par événements discret-- L.Granger--SIM61--2000-10-19-10:11

## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

```

Routine ConserverSemence (Num)
  Define Num as an integer variable
  DEFINE i AS AN INTEGER VARIABLE

  RESERVE SaveSeed(*) AS Num

  FOR i = 1 TO Num,
  DO
    SaveSeed(i) = SEED.V(i)
  LOOP
END " ConserverSemence

```

ÉCOLE POLYTECHNIQUE MONTREAL  
IF505 Simulation par événements discret-- L.Granger--SIM62--2000-10-19-10:11

### SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

ROUTINE Initialiser(Num)

```
*****
*****
" Fonction: Réinitialisation des variables et des totaux des "statistiques
recueillies après chaque répétition, et réactives les "terminaux pour la
prochaine.
"Usage : Appel de main *
"Entrée : Num : Nombre de suite de nombres aléatoires
"Retour :
"E/S : Aucune
"Portable: SIMSCRIPT II.5 (IBM 370 & IBM PC (Dos) & SUN)
"Notes :
"Statut : Production
"Creation: 06-92 Granger Louis & CACI
```

Define Num as an integer variable

Define i as an integer variable

```
LET TIME.V = 0.0          " Remettre l'heure a zero
LET Nb.Taches. Completees = 0      " ainsi que le nombre de
                                     " travaux completes
```

RESET TOTALS OF Temps.Reponse

for each CPU

RESET TOTALS OF N.O.CPU(CPU), AND N.X.CPU(CPU)



### SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

For i = 1 to Num,

do

SEED.V(i) = SaveSeed(i)

loop

FOR i = 1 TO Nb.Terminaux

DO

ACTIVATE A Terminal NOW

LOOP " au prochain terminals

END " Initialiser



## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

```

PROCESS Terminal
UNTIL Nb.Taches.Completees >= Nb.Taches.Desirees
DO
  WAIT EXPONENTIAL.F ( Temps.Reflexion.Moyen, 1 ) .Seconds "
  Reflechir
  CREATE A Job          " Soumettre une tâche
  LET Jb.Terminal ( Job ) = Terminal " pour ce terminal
  ACTIVATE THIS Job NOW          " immediatement
  SUSPEND " le Terminal
  ADD 1 TO Nb.Taches.Completees " une tâche de plus de
                                " completer
  IF Nb.Taches.Completees = Nb.Taches.Desirees
    CALL Report
  ENDIF
  LOOP " jusqu'a la fin
END " Terminal
    
```

## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

```

PROCESS Job

DEFINE Temps.Service AND Temps.Debut AS REAL VARIABLES

LET Temps.Debut = TIME.V " Debut de la job
LET Temps.Service = EXPONENTIAL.F ( Temps.Service.Moyen, 2 ) " Duree
                                " de la job

WHILE Temps.Service > Quantum
DO
  REQUEST 1 CPU (1)          " Demander le CPU
  WORK ( Quantum + Overhead ) .Seconds " pour une certaine duree
  LET Temps.Service = Temps.Service - Quantum " ce qui'il reste a executer
  RELINQUISH 1 CPU (1)      " Libere le CPU
LOOP

REQUEST 1 CPU (1)          " Pour la derniere tranche
WORK ( Temps.Service + Overhead ) .Seconds
RELINQUISH 1 CPU (1)
LET Temps.Reponse = TIME.V - Temps.Debut
REACTIVATE THE Terminal CALLED Jb.Terminal(Job) NOW

END " Job
    
```

## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

ROUTINE Report

PRINT 3 LINES WITH Nb.Terminaux, Temps.Reponse.Moyen,  
Max.Temps. Reponse,  
Nb.Moyen.Dans.File (1), Utilisation.Cpu (1) THUS

\*\*\* \*\*\*\*\*  
\*\*\*\*\*  
\*\*\* \*\*

END "Report

## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

ROUTINE Lire.Donnees  
CREATE EVERY CPU (1) " Un ordinateur a 1 processeur  
LET U.CPU(1) = 1 " Un seul processeur

READ Quantum, Overhead " Lire certains parametres  
READ Min.Terminals, Max. Terminals, Inc.Nb.Terminaux,  
Temps.Reflexion.Moyen  
READ Temps.Service.Moyen, Nb.Taches.Desirees  
PRINT 6 LINES THUS

Modele d'ordinateur en temps partage -- Parametre d'entree

SKIP 3 OUTPUT LINES  
PRINT 11 LINES WITH  
Temps.Reflexion.Moyen, Temps.Service.Moyen,  
Quantum, Overhead, Nb.Taches.Desirees THUS

Temps de reflexion sont exponentiels avec moyenne \*.\* secondes

Temps service sont exponentiels avec moyenne \*.\* secondes

Quantum \*.\* secondes

Overhead \*.\* secondes

Nb. taches traitees \*\*\*\*

## SIMSCRIPT II.5 EXEMPLE COMPLET (suite)

START NEW PAGE  
PRINT 6 LINES THUS

Resultats de simulation (Tous les temps en secondes)

SKIP 3 OUTPUT LINES  
PRINT 5 LINES THUS

| Nombre de terminaux | Moyenne temps reponse | Maximum temps reponse | Nombre moyen dans la file | Utilisatioin CPU |
|---------------------|-----------------------|-----------------------|---------------------------|------------------|
| -----               | -----                 | -----                 | -----                     | -----            |

END " Lire.Donnees



## GÉNÉRATEURS DE DISTRIBUTIONS UNIFORMES SIMSCRIPT II.5

URN30 SIMSCRIPT

$$x_{i+1} = 660,360,016 x_i \pmod{2^{31} - 1}$$

À noter  $660,360,016 = 14^{29} = (2^{29})(7^{29}) \pmod{2^{31} - 1}$   
ou

$$660,360,016 = (2^4)(11)(13)(137)(2011)$$

Congruence multiplicative

Période  $2^{31} - 1$  sur IBM

Implantation SIMSCRIPT

RANDOM.F(i) où  $i=1, \dots, 10$  (Suites différentes)

Chaque fois que RANDOM.F(i) est exécuté:

- production d'un nombre aléatoire (NA) dans l'intervalle  $0 \leq x_n \leq 1$ ,
- SEED.V(i) est mise à jour

|                         |                         |
|-------------------------|-------------------------|
| SEED.V(1)=2,116,429,302 | SEED.V(6)=1,157,240,309 |
| SEED.V(2)= 683,743,814  | SEED.V(7)= 17,726,055   |
| SEED.V(3)= 964,393,174  | SEED.V(8)= 48,108,509   |
| SEED.V(4)=1,217,426,631 | SEED.V(9)=1,797,920,909 |
| SEED.V(5)= 618,433,579  | SEED.V(10)= 477,424,540 |



### GÉNÉRATEURS DE DISTRIBUTIONS UNIFORMES SIMSCRIPT II.5

- ☐ Changer le nombre de suites ( par exemple 20)

```
RELEASE SEED.V  ** Supprimer le tableau standard
RESERVE SEED.V(*) AS 20 ** Définir un tableau 20
FOR i = 1 TO 20
  READ SEED.V(i)
```

- ☐ Pour générer un variable antithétique (technique de réduction de variance) -- utiliser un numéro de suite négatif  
EXPONENTIAL.F(Moyenne, -2)



### GÉNÉRATEURS DE DISTRIBUTIONS UNIFORMES SIMSCRIPT II.5

- ☐ La distribution uniforme

Intervalle :  $-\infty \leq x \leq \infty$ , paramètre  $a, b; b > a$

Fonction de densité :  $f(x) = \frac{1}{(b-a)}$

Fonction de distribution  $F(x) = \begin{cases} 0 & \text{si } x < a \\ \frac{x-a}{(b-a)} & \text{si } a \leq x \leq b \\ 1 & \text{si } x > b \end{cases}$

Moyenne  $\frac{(b+a)}{2}$

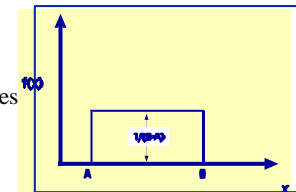
Variance  $\frac{(b-a)^2}{12}$

UNIFORM.F( $q_1, q_2, \text{Suite NA}$ )

$q_1$ : début de l'intervalle a

$q_2$ : fin de l'intervalle b

Suite NA: # de la suite de nombres aléatoire



GÉNÉRATEURS  
DE DISTRIBUTIONS NORMALES  
SIMSCRIPT II.5

☰ La distribution normale

Intervalle :  $-\infty \leq x \leq \infty$

Paramètre  $m, s > 0$

Fonction de densité  $f(x) = \frac{1}{\sqrt{2\pi}} e^{-0.5 \frac{(x-m)^2}{s^2}}$

Moyenne  $m$

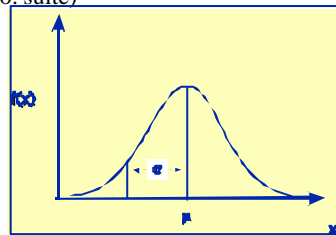
Variance  $s^2$

NORMAL.F, Mu, Sigma, No. suite)

Mu: moyenne  $m$

Sigma: écart-type  $s$

No.suite: # de la suite de  
nombres  
aléatoire



GÉNÉRATEURS  
DE DISTRIBUTIONS DE POISSON  
SIMSCRIPT II.5

☰ La loi de Poisson

Intervalle :  $n = 0,1,2,3,4 \dots N$

Paramètre  $IT$

Fonction de densité :  $f(x) = \Pr(X = n)$   
 $= \frac{(IT)^n}{n!} e^{-IT}$

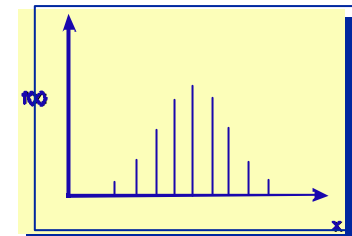
Moyenne  $IT$

Variance  $IT$

POISSON.F( $q_1$ , No. suite)

$q_1$  : moyenne

No. suite: # de la suite  
de nombres  
aléatoires



GÉNÉRATEURS  
DISTRIBUTION EXPONENTIELLE  
SIMSCRIPT II.5

☰ La loi exponentielle

Intervalle :  $0 \leq x < \infty$ , paramètre  $l$

Fonction de densité :  $f(x) = \begin{cases} l e^{-lx} & \text{pour } 0 \leq x < \infty \\ 0 & \text{ailleurs} \end{cases}$

Fonction de distribution  $F(x) = \int_0^x l e^{-lx} dx$   
 $= \frac{l e^{-lx}}{-l} \Big|_0^x$   
 $= 1 - e^{-lx}$

Moyenne  $\frac{1}{l}$

Variance  $\frac{1}{l^2}$

EXPONENTIAL.F( $q_1$ , No. suite)

$q_1$ : moyenne  
 no. suite: # de la suite  
 de nombres aléatoires



GÉNÉRATEURS  
DISTRIBUTION GAMMA  
SIMSCRIPT II.5

☰ La distribution Gamma

Intervalle :  $0 \leq x < \infty$

Paramètre paramètre d'échelle  $a > 0$   
 paramètre de forme  $b > 0$

Fonction de densité  $f(x) = \frac{\left(\frac{x}{a}\right)^{b-1} e^{-\left(\frac{x}{a}\right)}}{a \Gamma(b)}$

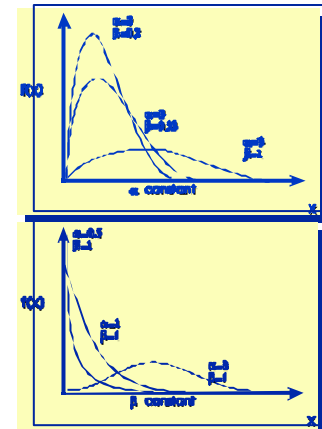
Moyenne  $ab$

Variance  $a^2 b$

BETA.F( $q_1, q_2$ , No. suite)

$q_1$ : moyenne  $\left(\frac{x}{a}\right)$   
 $q_2$ : puissance de

No. suite: # de la suite  
 de nombres  
 aléatoires



GÉNÉRATEURS  
DISTRIBUTION GAMMA  
SIMSCRIPT II.5

☰ La distribution beta

Intervalle :  $0 \leq x < 1$

Paramètre  $a > 0, b > 0$

Fonction de densité :  $f(x) = \frac{x^{a-1}(1-x)^{b-1}}{B(a,b)}$

ou

$$B(a,b) = \int_0^1 x^{a-1}(1-x)^{b-1} dx$$

Moyenne  $\frac{a}{(a+b)}$

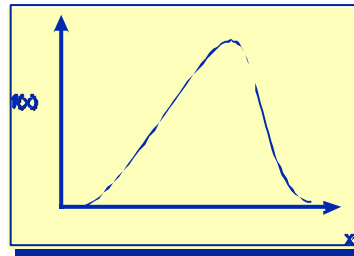
Variance  $\frac{ab}{(a+b)^2(a+b+1)}$

BETA.F( $q_1, q_2, No. suite$ )

$q_1$  : puissance de  $x$

$q_2$  : puissance de  $(1-x)$

No. suite : # de la suite de nombres aléatoires



GÉNÉRATEURS  
DISTRIBUTION LOGNORMALE  
SIMSCRIPT II.5

☰ La distribution lognormale

Intervalle :  $0 \leq x \leq \infty$

Paramètre  $m, s > 0$

Fonction de densité  $f(x) = \frac{1}{\sqrt{2\pi}sx} e^{-\frac{(\ln x - m)^2}{2s^2}}$

Moyenne  $e^{\frac{m+s^2}{2}}$

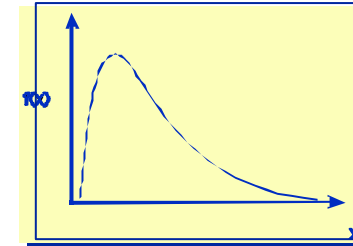
Variance  $e^{2ms^2}(e^{s^2} - 1)$

LOG.NORMAL.F( Mu, Sigma, No. suite)

Mu: moyenne  $m$

Sigma: écart-type  $s$

No.suite: # de la suite de nombres aléatoires



GÉNÉRATEURS  
DISTRIBUTION ERLANG-k  
SIMSCRIPT II.5

☰ La distribution Erlang-k

Intervalle :  $x \geq 0$

Paramètre  $a > 0, k > 0$

Fonction de densité  $f(x) = \frac{a}{\Gamma(k)} (ax)^{k-1} e^{-ax}$

Fonction cumulative  $F(x) = 1 - e^{-ax} \sum_{i=0}^{k-1} \frac{(ax)^i}{i!}$

Moyenne  $\frac{1}{a}k$

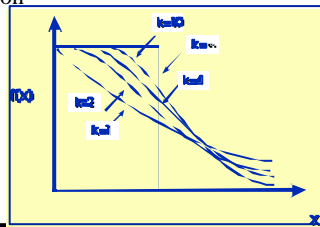
Variance  $\left(\frac{1}{a}\right)^2 k$

ERLANG.F ( $q_1, q_2, No\ suite$ )

$q_1$  : moyenne de la distribution

$q_2$  : paramètre  $k$

No. suite: # de la suite  
nombres aléatoires



GÉNÉRATEURS  
DISTRIBUTION WEIBULL  
SIMSCRIPT II.5

☰ La distribution Weibull

Intervalle :  $0 \leq x < \infty$

Paramètre paramètre d'échelle  $a > 0$   
paramètre de forme  $b > 0$

Fonction de densité  $f(x) = \frac{bx^{b-1}}{a^b} e^{-\left(\frac{x}{a}\right)^b}$

Fonction cumulative  $F(x) = 1 - e^{-\left(\frac{x}{a}\right)^b}$

Moyenne  $\frac{a}{b} \Gamma\left(\frac{1}{b}\right)$

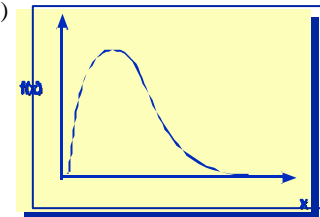
Variance  $\frac{a^2}{b^2} \left\{ 2b \Gamma\left(\frac{2}{b}\right) - \left[ \Gamma\left(\frac{1}{b}\right) \right]^2 \right\}$

WEIBULL.F ( $q_1, q_2, No\ suite$ )

$q_1$  : paramètre d'échelle

$q_2$  : paramètre de forme

No. suite: # de la suite  
nombres aléatoires



GÉNÉRATEURS  
DISTRIBUTION BINOMIALE  
SIMSCRIPT II.5

☰ La distribution Binomiale

Intervalle :  $x=0,1,2,\dots,n.$

Paramètre  $p$ =probabilité d'un succès dans un essai,  $0 \leq p < 1$ ,  
 $n$ = nombre d'essais;  $n > 0$

Fonction de densité  $f(x) = \binom{n}{x} p^x (1-p)^{n-x}$

Moyenne  $np$

Variance  $np(1-p)$

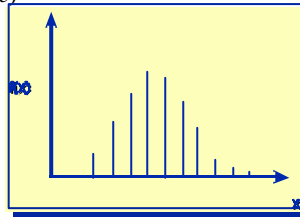
BINOMIAL.F ( $q_1, q_2, No\ suite$ )

$q_1$  :  $n$  nombre d'essais,

$q_2$  :  $p$  probabilité d'un succès

No. suite: # de la suite  
nombres aléatoires

:



GÉNÉRATEURS  
DISTRIBUTION BINOMIALE  
SIMSCRIPT II.5

☰ La distribution Binomiale

Intervalle :  $x=0,1,2,\dots,n.$

Paramètre  $p$ =probabilité d'un succès dans un essai,  $0 \leq p < 1$ ,  
 $n$ = nombre d'essais;  $n > 0$

Fonction de densité  $f(x) = \binom{n}{x} p^x (1-p)^{n-x}$

Moyenne  $np$

Variance  $np(1-p)$

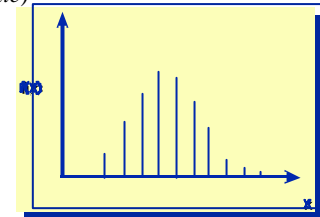
BINOMIAL.F ( $q_1, q_2, No\ suite$ )

$q_1$  :  $n$  nombre d'essais,

$q_2$  :  $p$  probabilité d'un succès

No. suite: # de la suite  
nombres aléatoires

:



GÉNÉRATEURS  
DISTRIBUTION EMPIRIQUE DISCRÈTE  
SIMSCRIPT II.5

**Implantation**

☞ Dans le PREAMBLE

```
{ THE SYSTEM }
{ EVERY ENTITY } HAS A identificateur
RANDOM { STEP }
        { LINEAR } VARIABLE
```

```
DEFINE identificateur AS A { REAL }
                          { INTEGER } STREAM
entier VARIABLE
```

GÉNÉRATEURS  
DISTRIBUTION EMPIRIQUE DISCRÈTE  
SIMSCRIPT II.5

**Implantation**

☞ ans le PREAMBLE

```
THE SYSTEM HAS A Temps.Chargement.F
RANDOM STEP VARIABLE
DEFINE Temps.Chargement.F AS A REAL VARIABLE
```

☞ Dans le programme d'initialisation

```
READ Temps.Chargement.F
```

Dans les données

```
0.25 18 0.55 24 0.2 36 *
```

ou

```
0.0 0 0.25 18 0.80 24 1.0 36 *
```

☞ Durant la simulation

```
WAIT Temps.Chargement.F HOURS
```

ou

```
WORK EXPONENTIAL.F (Temps.Chargement.F, 1) HOURS
```

ou bâtie dans le programme

```
WRITE AS "0.0 0 0.25 18 0.80 24 1.0 36 *" USING THE
BUFFER
```

```
READ Temps.Chargement.F USING THE BUFFER
```

GÉNÉRATEURS  
DISTRIBUTION EMPIRIQUE CONTINUE  
SIMSCRIPT II.5

PREAMBLE

THE SYSTEM OWNS A Ext1.Train,

A Ext2.Train,

A Mouvement,

A Res.Train

AND HAS A Lumiere,

A Fct.Distribution.F RANDOM LINEAR  
VARIABLE

DEFINE Fct.Distribution.F AS A REAL STREAM 3  
VARIABLE

END " Preamble



GÉNÉRATEURS  
DISTRIBUTION EMPIRIQUE DISCRÈTE  
SIMSCRIPT II.5

Implantation

☞ Dans le PREAMBLE

THE SYSTEM HAS A Temps.Chargement.F RANDOM  
STEP VARIABLE

DEFINE Temps.Chargement.F AS A REAL VARIABLE

☞ Dans le programme d'initialisation

READ Temps.Chargement.F

Dans les données  
0.25 18 0.55 24 0.2 36 \*

ou

0.0 0 0.25 18 0.80 24 1.0 36 \*

☞ Durant la simulation

WAIT Temps.Chargement.F HOURS

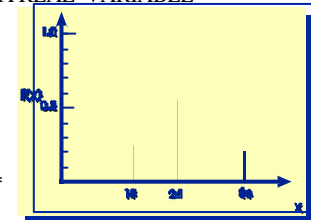
ou

WORK EXPONENTIAL.F (Temps.Chargement.F, 1) HOURS

ou bâtie dans le programme

WRITE AS "0.0 0 0.25 18 0.80 24 1.0 36 \* " USING THE  
BUFFER

READ Temps.Chargement.F USING THE BUFFER



GÉNÉRATEURS  
DISTRIBUTION EMPIRIQUE CONTINUE  
SIMSCRIPT II.5

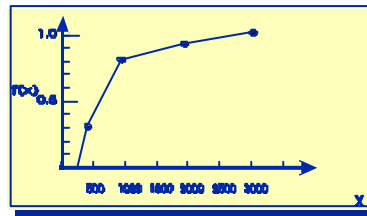
Exemple

- ☐ Poids des paquets de linge arrivant dans un lavoir
  - 30% entre 300 et 500 grammes
  - 50% entre 500 et 1000 grammes
  - 15% entre 1000 et 2000 grammes
  - 5% entre 2000 et 3000 grammes

PERMANENT ENTITIES..... ou RESOURCE....  
EVERY Lavoir HAS A Poids.F RANDOM LINEAR VARIABLE  
DEFINE Poids.F AS A REAL VARIABLE

N.Lavoir = 10  
CREATE ALL Lavoir

FOR EACH Lavoir,  
DO  
  READ Poids.F  
LOOP



GÉNÉRATEURS  
DISTRIBUTION EMPIRIQUE CONTINUE  
SIMSCRIPT II.5

ROUTINE Lire.Distribution " du temps d'arrêt en station

\*\*\*\*\*Lire.Distribution\*\*\*\*\*  
\*\*\*\*\*

DEFINE I AS INTEGER VARIABLE

USE UNIT .Fich.Distribution FOR INPUT  
READ Fct.Distribution

START NEW PAGE  
PRINT 2 LINES AS FOLLOWS

Fonction de distribution du temps d'arrêt  
Prob.Cum Var.Ind

I = F.Fct.Distribution

WHILE (I NE 0)

DO

  PRINT 1 LINE WITH PROB.A(I), RVALUE.A(I) AS FOLLOWS

    \*\*\*\*\* \*\*\*)\*\*

  I = S.Fct.Distribution(I)

LOOP

USE UNIT .FichParametre FOR INPUT  
RETURN

END "Lire.Distribution du temps d'arrêt en station

Concept d'ensemble [SET]  
(File d'attente)  
SIMSCRIPT II.5

Concept d'ensemble [SET]  
(File d'attente)

Les ensembles sont:

- listes ordonnées d'entités
- ordonnées au moment de s'insertion d'un nouveau membre
- propriétaire (OWNED par une entité, ou par THE SYSTEM)
- appartenance (BELONGS TO )
- efficace en temps et espace

Concept d'ensemble [SET]  
(File d'attente)  
SIMSCRIPT II.5

**Définition d'un ensemble:**

1. Déclaration des membres potentiels  
EVERY Client MAY BELONG TO THE FileAttente
2. Déclaration des propriétaires  
THE SYSTEM OWNS THE FileAttente  
ou  
EVERY Pbx HAS A NombreLignes,  
.....  
AND OWNS A FileAppel
3. Déclaration de la méthode d'ordonnement  
DEFINE FileAppel AS A FIFO SET  
DEFINE FileAttente AS A SET RANKED BY LOW Taille

Ensemble [SET]  
(File d'attente)  
SIMSCRIPT II.5

```
DEFINE ensemblec AS [A] [FIFO] SET[S] [RANKED {BY [HIGH] attribut}c then]
[WITHOUT attribut_ensemblec ATTRIBUTE[S]]
[.,]WITHOUT routin_ensemblec ROUTINE[S]
```

Exemple:

```
DEFINE FileJob AS A SET RANKED BY HIGH Priorite
```

```
DEFINE File AS A FIFO SET
```

```
DEFINE LigneAviation AS A SET RANKED BY LOW Nom
WITHOUT L AND P ATTRIBUTES
WITHOUT FL, FB, FA AND R ROUTINES
```



Ensemble [SET]  
(File d'attente)  
SIMSCRIPT II.5

☰ Entrée dans un ensemble

```
FILE Re pere {FIRST
LAST
BEFORE p} IN Queue
AFTER p}
```

Exemple:

```
FILE Tache IN Liste_Tache
```



Ensemble [SET]  
(File d'attente)  
SIMSCRIPT II.5

Sortie d'un ensemble

```
REMOVE { FIRST
        LAST } Re pere FROM Queue
        THE
```

Exemple:

```
REMOVE FIRST Tache FROM Liste_Tache(Machine)
```

```
REMOVE LAST Tache FROM Liste_Tache(Machine)
```

```
REMOVE THIS Tache FROM Liste_Tache(Machine)
```

```
FOR EACH Tache IN Liste_Tache (Machine),
```

```
WITH .....
```

```
DO
```

```
REMOVE THIS Tache FROM Liste_Tache (Machine)
```

```
LOOP
```

Ensemble [SET]  
(File d'attente)  
SIMSCRIPT II.5

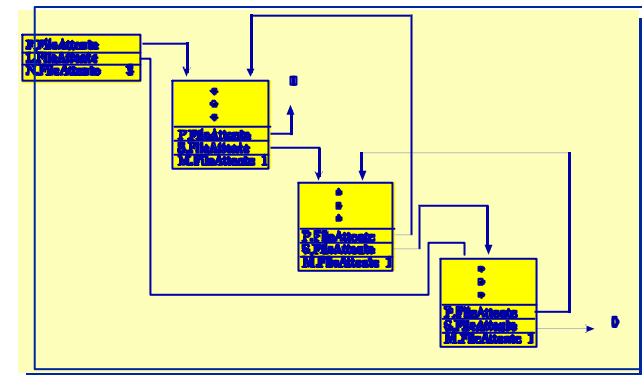
Ensemble décrit par ses attributs

Exemple:

```
EVERY Client BELONGS TO A FileAttente
```

```
THE SYSTEM OWNS THE FileAttente
```

```
DEFINE FileAttente AS A FIFO SET
```



## SIMSCRIPT II.5 Processus Commandes additionnelles

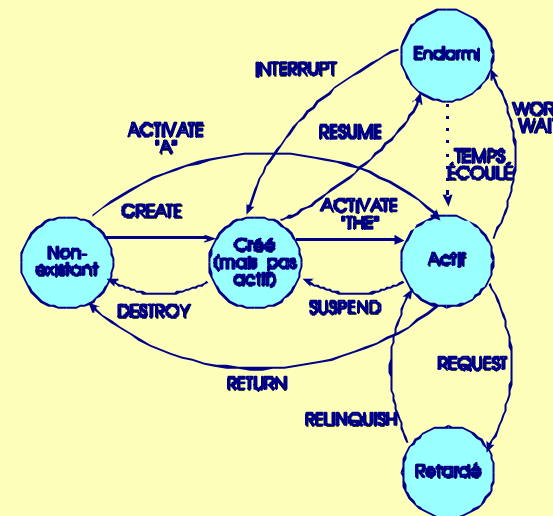
### Processus interagissent entre eux

Un processus peut être:

- INTERROMPTU par un autre processus  
(On assume qu'il est dans un WORK/WAIT)  
Le temps qu'il lui restait est conservé, dans un attribut du processus (TIME.A).
- REDEVENIR ACTIF  
RESUME un autre processus
- SUSPENDU par lui-même  
SUSPEND
- REACTIVE  
REACTIVATE

## SIMSCRIPT II.5 Processus Commandes additionnelles

Processus interagissent entre eux



## SIMSCRIPT II.5 Processus Arguments

- ☐ On peut référencer les attributs d'un processus l'intérieur de ce processus.
- ☐ La variables du nom du processus est automatiquement définie.
- ☐ L'indice peut être omis

Exemple:

LET a = TypeEssence (Auto)

est équivalent à

LET a = TypeEssence

si

1) TypeEssence est un attribut de l'entité

Auto ou processus Auto

2) La variable Auto a une bonne valeur



## SIMSCRIPT II.5 Processus Arguments

- ☐ Les arguments (Variables locales ) peuvent être déclarés.
- ☐ Les valeurs sont attribuées aux arguments au moment de l'entrée dans le processus dans l'ordre de déclaration dans le PREAMBLE

Exemple:

**PREAMBLE**

**PROCESSES.....**

**EVERY Auto HAS A TypeEssence,**

**A TailleReservoir**

**:**

**END "PREAMBLE**

**:**

**ACTIVATE AN Auto GIVING "Régulier" AND 23.5 NOW**

ou

**ACTIVATE AN Auto NOW**

**LET TypeEssence = "Régulier"**

**LET TailleReservoir= 23.5**

**PROCESS Auto GIVEN TypeEssence , Capacite**

**DEFINE TypeEssence AS A TEXT VARIABLE**

**DEFINE Capacite AS A REAL VARIABLE**



## SIMSCRIPT II.5 Structure de l'échéancier

Echéancier est un ensemble ordonné appelé EV.S

☞ Attributs de l'ensemble

- F.EV.S
- L.EV.S tableaux de système
- N.EV.S(i) fonction de système

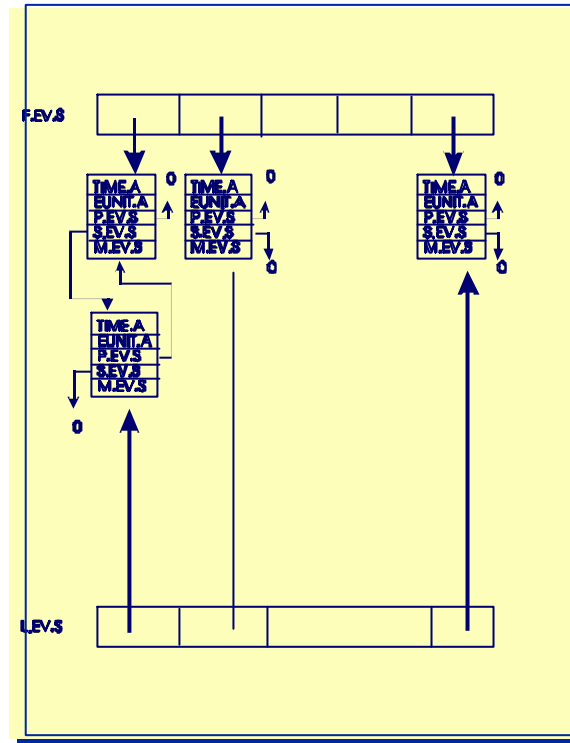
☞ Attributs de la notice

- TIME.A temps de l'occurrence de l'évènement
- P.EV.S prédécesseur
- S.EV.S successeur
- M.EV.S indicateur d'appartenance

☞ Variables globales

- EVENTS.V le nombre des différents types de processus et d'évènement
- EVENT.V la classe de prochain évènement à exécuter
- INom\_Evenement numéro du processus (en fonction de l'énoncé PRIORITY ORDER dans le préambule)

## SIMSCRIPT II.5 Structure de l'échéancier



## SIMSCRIPT II.5

### Structure de l'échéancier

### Notice d'un processus

|                |                                                                                                                           |
|----------------|---------------------------------------------------------------------------------------------------------------------------|
| <b>TIME.A</b>  | heure d'activation du processus, ou le temps restant à simuler dans un WORK/WAIT ou nombre de l'itération                 |
| <b>EUNIT.A</b> | numéro de l'unité pédagogique et le processus est activé (ordinateur)                                                     |
| <b>P.EV.S</b>  | pointeur à l'événement suivant dans la chaîne des événements futurs (échéancier)                                          |
| <b>S.EV.S</b>  | pointeur à l'événement précédent dans la chaîne des événements futurs (échéancier)                                        |
| <b>M.EV.S</b>  | indicateur si le processus fait partie ou non de l'échéancier                                                             |
| <b>STA.A</b>   | état du processus: 0 : prêt (mais à un ACTIVATE) mais non encore actif<br>1 : WORK/WAIT<br>2 : suspendu<br>3 : interrompu |
| <b>IPC.A</b>   | rates de accès du processus                                                                                               |
| <b>RSA.A</b>   | pointeur à une région de mémoire de sauvegarde pour ce processus                                                          |
| <b>F.R.S.S</b> | attribut de propriété d'une ressource                                                                                     |

## SIMSCRIPT II.5

### Priorité des événements et processus

- Les processus/événements les plus prioritaires sont exécutés en premier.

Dans le préambule

PRIORITY ORDER  $\left\{ \begin{array}{l} \text{Evenement} \\ \text{Processus} \end{array} \right\}^c$

Exemple:

PRIORITY ORDER IS Fin.Simulation, Arrivee, Client

Pour des processus/événements de même classe, la priorité est établie par

BREAK  $\left\{ \begin{array}{l} \text{Evenement} \\ \text{Processus} \end{array} \right\}$  TIES BY  $\left[ \begin{array}{l} \text{HIGH} \\ \text{LOW} \end{array} \right]$   $\left\{ \begin{array}{l} \text{Attributs} \end{array} \right\}^{c\text{THEN}}$

Exemple:

BREAK ReservationSiege TIES BY HIGH CodePrivilege,  
THEN BY HIGH Tarif,  
THEN BY LOW MembreClub

## SIMSCRIPT II.5 Évènements

☞ Historique: orienté évènements

☞ Relation avec les processus

- Processus instantané
- Processus composé d'une suite d'évènements et d'activités.
- Évènements utilisent des entités temporaires pour la communication
- Les processus et les évènements peuvent coexister dans un même modèle,
- Évènements ne peuvent utiliser des ressources.



## SIMSCRIPT II.5 Utilisation des évènements

Déclarés dans le PREAMBLE

PREAMBLE

```
EVENT NOTICES INCLUDE Arrivee, Depart, FinLot  
EVERY IntervalleInjection HAS A Delta_T
```

```
END "PREAMBLE
```

Appel à un évènement

```
SCHEDULE A FinLot IN 7 DAYS
```

Sous-programme EVENT

```
EVENT FinLot
```

```
CALL ImprimerResultats
```

```
RESETS TOTALS OF N.X.Serveur
```

```
SCHEDULE A FinLot IN 7 DAYS
```

```
END "FinLOT
```



## SIMSCRIPT II.5 Évènements et Processus externes

Données (EXTERNAL UNIT)

NomÉvenement Temps paramètres\_a\_lire \*

Temps

unités TEMPETE 14536.8 \*

jour-heure-minute TEMPETE 23 12 40 \*

Calendrier mm/jj/aa TEMPETE 2/23/89 8 30 \*

- Passer des données (paramètres à des évènements/processus externes.
- Lire les données dans le fichier externe et les stocker dans les attributs.

```
PROCESS Tempete GIVEN Location, Duree
DEFINE Location AS AN INTEGER VARIABLE
DEFINE Duree AS A DOUBLE VARIABLE
```

```
IF PROCESS IS EXTERNAL,
  READ Location, Duree
END IF
```

```
END"Tempete
```

## SIMSCRIPT II.5 Temps

Fonctions relatives à l'horloge

**ORIGIN.R** doit être appelée avant d'utiliser l'heure avec le format calendrier

**HOURS.V** Nombre d'heures par jour

**MINUTES.V** Nombre de minutes dans une heure

**DATE.F** convertit (mm-jj-aa) en format de temps interne

**DAY.F** extrait le jour

**MONTH.F** extrait le mois

**YEAR.F** extrait l'année

**HOURL.F** extrait l'heure du jour

**MINUTE.F** extrait la minute

Exemple

```
PRINT ! LINE WITH MONTH.F(TIME.V), DAY.F(TIME.V)
YEAR.F(TIME.V), HOURL.F(TIME.V),
MINUTE.F(TIME.V) THUS
```

L'heure est : \*\*/\*\*/\*\*\*\* \*\*:\*\*

qui donne

L'heure est : 8/19/1992 22:34

## SIMSCRIPT II.5

### Concept d'ensemble [SET] (File d'attente)

Les ensembles sont:

- listes ordonnées d'entités
- ordonnées au moment d'insertion d'un nouveau membre
- propriétaire (OWNED par une entité ou par THE SYSTEM)
- appartenance (BELONGS TO)
- efficace en temps et espace

## SIMSCRIPT II.5

### Gérer sa propre file d'attente

Preamble

Resources include CPU

....

Process include Job

The System owns a File.CPU

Define File.cpu as a fifo set

end " Preamble

Process Job

Define Temps.Debut AS a double variable

Let Temps.Debut = TIME.V

if U.CPU < 1

file Job in File.CPU

suspend

endif

request 1 unit of CPU

let Temps.Attente = TIME.V - Temps.Debut

work .....

relinquish 1 CPU

if File.CPU is not EMPTY

remove first job.suspendue from File.CPU

reactivate THE Job called job.suspendue NOW

end if

end " Job

SIMSCRIPT II.5

Gérer sa propre file d'attente  
Ressources simultanées

Preamble

```
Resources include CPU and Memoire
....
Process Job
    Every Job has Jb.Unite .Memoire
TheSystem owns a File.CPU
Define File.cpu as a fifo set
end " Preamble
```

Routine Initialiser

```
Let N.CPU = 1
Create all CPU
let U.cpu = 1.0
Let N.Memoire = 1 " Une memoire de 16 Mo
Create all Memoire
Let U.memoire = 16000000.0
```

```
.....
end "Initialiser
```



SIMSCRIPT II.5

Gérer sa propre file d'attente  
Ressources simultanées

Process Job (Memoire.Requise)

```
Define Temps.Debut AS a double variable
Define Memoire.Requise as an integer variable
```

```
Let Temps.Debut = TIME.V
if U.CPU < 1 OR Memoire.Requise > U.Memoire
    " Les deux ressources ne sont pas disponibles simultanément
    file Job in File.CPU " Mettre dans la Q
    suspend " Suspendre le processus
endif
```

```
" Les ressources étaient disponibles simultanément
request 1 unit of CPU
request Memoire.Requise units of Memoire
let Temps.Attente = TIME.V - Temps.Debut
work .....
relinquish 1 CPU
relinquish Memoire.Requise units of Memoire
```

```
if File.CPU is not EMPTY " Y-a-t-il des jobs en attente
    " Sortir de la Q la première
    Let I = F.File.CPU
    while (I NE 0)
        do
            if ( Memoire.Requise > U.Memoire)
                I = S.File.CPU
            else
                leave
            endif
        loop
    if ( I NE 0)
        remove I from File.CPU
        reactivate THE Job called I NOW
    endif
endif
```

```
end " Job
```

