# Utilisation d'AnyLogic

Auteurs : P. Lacomme (placomme@isima.fr)

D. Lamy (<u>lamy@isima.fr</u>)

Date de création : Janvier 2017

### 1) Installation

Le logiciel de simulation est disponible à l'adresse suivante : <u>http://www.anylogic.com/</u> Le logiciel se trouve dans **Download** et il faut choisir Free PLE (Free Personal Learning Edition).



Il faut choisir la version correspondante au système utilisé (ici Windows 64 bits)

😵 Downloads — AnyLogic 🗙			
← → C û www.anylogic.com/downloads	7	2 🕐 🐵	🛯 🔒 🍋 🖳 🖓 🛈 💔 (
👖 Applications 🌗 WebCast - Captation / 🦻 Introduction aux Rése 🕴	😓 www.kinecthacks.com 🗋 Composants du systèr 🎦 file	:///C:/Users/Laco	m » Autres favor
Evaluation: Use links below to download, and install AnyLogic. Run A Upgrade: All users under maintenance: please download and upgrac For information about the new AnyLogic 7.3 features, check out the An	AnyLogic and use Activation Wizard to request Evaluation Key, ie your AnyLogic to the latest version ( <u>changes history</u> ). <u>vil ogic 7.3</u> page.		maintenance renewal.
Free Persona AnyLogic is available for free for add Download the <u>free:</u>	Learning Edition ucational purposes and personal learning. simulation software below!		
AnyLogic Software Licensing and Model End User Agreeme 	<u>n</u>		
AnyLogic 7.3.6 FOR WINDOWS	AnyLogic 7.3.6		AnyLogic 7.3.6
PROFESSIONAL			
UNIVERSITY RESEARCHER X32   X64	UNIVERSITY RESEARCHER X64		University Researcher 🗴
Free PLE <u>x32</u>   <u>x64</u>	Free PLE X64		
Activation Guides »	Activation Guides »		Activation -
<ul> <li>&gt; USA: +1 312 635 3344</li> <li>&gt; Europe: +3 3(0) 160 71 60 58</li> <li>&gt; World-wide: +7 812 441 31 05</li> </ul>	search	۶ ا	DOWNLOAD FREE SIMULATION SOFTWARE

Il ne reste plus qu'à remplir le formulaire

😵 Download AnyLogic PLE - 🗙		(!) Philippe 📄	
$\leftarrow$ $\rightarrow$ C $\triangle$ $\textcircled{0}$ www.anylog	ic.cc Q 🖈 🕐 🐵 👪	R 8 🛛 🔇	2 🛈 🔒
🗰 Applications 🕨 WebCast - Captati	ion / 🦻 Introduction aux Rése	» 📃 A	utres favoris
Last Name:*			•
Organization:*			
Department:			
Website:			
Country:*	Select	-	
E-mail:*			
Phone:*			
What problem are you solving with simulation?*			
How did you hear about AnyLogic?*			
Software Version:*	Windows x64	-	
Sign up for monthly newsletter:	۲		
4	Download the software	»	+

### Remarque :

Contrairement à beaucoup d'autres environnements, Anylogic permet de réaliser plusieurs types de modèle de simulation. Il s'agit là du point fort de l'environnement.

# 2) Modèle de simulation à évènements discrets : simuler une file MM1

Le système à simuler se compose d'une source (entrée), d'une station (serveur + file d'attente) et d'une sortie.



\rm New Model	
New Model Create a new mo	del
Model name:	FILE_MM1
Location:	esktop\prins_novembre_2016\essai_simio\Anylogic\Philippe Browse
Java package:	file_mm1
Model time units:	seconds 👻
The following mo	del will be created:
C:\Users\lacomn \FILE_MM1\FILE_	ne\Desktop\prins_novembre_2016\essai_simio\Anylogic\Philippe MM1.alp
	Finish Cancel

Trois éléments doivent être utilisés :

- une source ;
- un service ;
- un puits.

Ces éléments se trouvent dans Process Modeling Library.



Pour la source, il faut spécifier que les arrivées sont définies par une durée interarrivée et que la durée interarrivée vaut 1.

			• source - Source		
source	service	sink	v Name: Arrivals defined by:	source	Show name 🔲 Ignore
+>+	О	×	Interarrival time:	7	seconds 🔻
			Set agent parameters from DB:	=, 🔳	
			Multiple agents per arrival:	=, 🔳	
			Limited number of arrivals:	=,	

Au niveau du service, on peut spécifier une longueur de file t'attente (par exemple 100) et un temps de traitement (Delay Time) de valeur 4.

		* *	<sup>™</sup> service - Service	
source	service	sink	Name: Seize: Resource sets (alternatives):	service       Image: Show name       Ignore         =       (alternative) resource sets       units of the same pool         =       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name         Image: Show name       Image: Show name       Image: Show name
			Queue capacity: Maximum queue capacity: Delay time:	= 100 = 100 = 100 = 100 = 100 = 100 = 100

Le résultat d'exécution est le suivant :

FILE_MM1 : Simulation - AnyLogi	c PLE [PERSONAL LEARNING USE ONI	Y] 🗆 🗆 🗙
🔲 🕶 🕨 🔳 📔 🤷 🛛 🗴 🗎	📑 🚯 🛛 🕸 🕼 root:Mair	1
source	service	sink
35	35 31	31
Run: 0 🜔 Running   Time: 35.55	Simulation: Stop time not set Da	a <b>te:</b> Jan 5, 2017 12:00:35 AM │ Dू

On peut avoir accès aux informations sur le service (on dit couramment station dans la terminologie QNAP2) par un simple clic sur le service.



On peut limiter la taille de la file à 2, en modifiant les attributs du service.

Properties	<b>T</b>
<sup>∎</sup> Service - Service	
Name:	service Show name Ignore
Seize:	<ul> <li>alternative) resource sets</li> <li>units of the same pool</li> </ul>
Resource sets (alternatives):	=] ● ひ ひ 怒 覧 ● Add list
Queue capacity:	=2
Maximum queue capacity:	=, 🔲
Delay time:	a seconds 🔻

#### 3) Modèle de simulation « spatial »

(Ce modèle est décrit dans le document officiel anylogic-7-in-3-days.pdf)

#### 3.1. Création d'un modèle.

Il faut créer un projet nommé Market.

春 New Model	
New Model	
Create a new mod	lel
Model name:	market
Location:	esktop\prins novembre 2016\essai simio\Anylogic\Philippe Browse
Java package:	market
Model time units:	seconds 👻
The following mod	lel will be created:
C:\Users\lacomm \market.alp	e\Desktop\prins_novembre_2016\essai_simio\Anylogic\Philippe\market
	Finish Cancel

Ce projet va utiliser la notion d'agent. Un agent dans **Anylogic** peut correspondre à un élément de flux « classique » tel que nous l'avons dans un modèle de simulation à événements discrets.

Il faut créer un Agent et le « déposer » sur la feuille. Il faut choisir Population of agents.



On peut ensuite choisir le nom de la classe Java ici **Consumers** et l'environnement AnyLogic complète avec le nom **consumers** pour la population.

🗛 New agent	
Step 2. Creating new age	nt type
Agent type name:	Consumer
Agent population name:	consumers
<ul> <li>Create the agent type "for the open set of the op</li></ul>	from scratch" ters of agents from database pwcharts
< <u>B</u> ;	ack Next > Finish Cancel

Ces deux notions sont très différentes comme nous le verrons par la suite.

On peut choisir ensuite sa représentation graphique en 2D.

A. New agent			
Step 3. Agent animation			
Choose animation: 🔘 3D	2D (	🔿 None	
💌 General	<b>^</b>		
Person	=		
🏚 Nurse		•	
🛉 Doctor			
🖗 Patient		- <b>T</b>	
🔍 USA Map		-	
🗆 Lorry			
Dirry 2	~		
< <u>B</u> ack	<u>N</u> ext >	<u><u> </u></u>	Cancel

On va ajouter à la classe Consumer un attribut nommé AdEffectiveness avec une valeur par défaut de 0.01.

New agent	
ep 4. Agent parameters	
Please fix the parameters you want to	see in your Consumer:
Parameters	Parameter: AdEffectiveness
AdEffectiveness	Type: double 🔻
<add new=""></add>	.,,,
	Specify value or stochastic expression
	0.01
	Eollow empirical distribution
	Percentage distribution of the population:
	Interval start Interval end Number of o
X	
	< <u>B</u> ack <u>N</u> ext > <u>Finish</u> Cancel

La population est alors constituée de 5000 agents.

New agent	
Oreate population with	5000 agents
This is the initial populati	on size.
You will be able to add m	nore agents or delete any agent at runtime.
Create initially empty pop	pulation, I will add agents at the model runtime
	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Cancel

Comme on souhaite obtenir une représentation graphique, il faut définir un rectangle de par exemple 500 par 500.

春 New agent	
Step 6. Configure ne	w environment
This agent will live in	the 'Main' agent type.
The following are the	environment settings.
You can always chan	ge them from the properties of Main agent type (see Space and network section)
Space type:	Continuous      GIS      Discrete     Discrete
Size:	500 x 500
Apply random I	ayout
Network type:	No network/User-defined 🔻
	< <u>Back</u> <u>N</u> ext > <u>Finish</u> Cancel

Le modèle se présente alors comme suit :

0	2 4 6 8 10 1meter = 10px meter
	connections
	consumers []

Il faut sélectionner l'icône et cocher la case I Draw agent with offset to this position

0 2 4 6 8 10 	Rotation, *: =, 0.0 v degrees v
	Scale is:
connections	✓ Advanced
·····	Show in:
	On click:
(i) consumers []	☑ Draw agent with offset to this position
	Show name
	▼ Description

On peut ensuite vérifier le bon fonctionnement du modèle... en cliquant sur L'environnement génère le code java et lance l'application qui se présente comme ceci :



En utilisant le bouton **Run**, on obtient une première exécution.



### 3.2. Définir un comportement du « consommateur »

Il faut se placer dans l'onglet Consumer qui en réalité modélise la classe Consumer.



Définir le comportement du consommateur revient à définir un diagramme définissant les changements d'état : un **statechart**.

L'ensemble des objets nécessaires à la définition d'un diagramme d'état sont dans l'onglet :



Dans un premier temps, il faut définir le point d'entrée et un premier état (nommé **PotentialUser**) avec une couleur par défaut (lavender).

👸 Main 🕱 👸 Consumer 🕱			□ Properties 🛿
		^ 	O PotentialUser - State
			Name: PotentialUser 🔽 Show name 🔲 Ignore
			Fill color:
AdEffectiveness	statechart		Entry action: shapeBody.setFillColor(lavender);
			Exit action:
	PotentialUser		Description

On retrouve l'ensemble des éléments ajoutés dans le projet.



On peut ajouter un deuxième état "user" auquel on peut associer une couleur verte.

at state bart	Properties 🛛				
Statechart	OUser - State				
•	Name: User 🕼 Show name 🔲 Ignore				
PotentialUser	Fill color: vellowGreen v				
	Entry action: shapeBody.setFillColor(yellowGreen);				
¥ IIII	Exit action:				
9	▼ Description				
User g					

On sélectionne la relation entre PotentialUser et User pour définir les conditions de mise en œuvre.

Le trigger qui déclenche la mise à feu de la transition est donné par **AdEffectiveness**.

	Properties 🛛	▽ □
• statechart	🖌 Ad - Transition	
······	Name: O Ad Show name Ignore	
PotentialLIser	Triggered by: Rate -	
I Oteritidioser	Rate: Q AdEffectiveness per day	Ŧ
	Action:	
	Guard:	
User	▼ Description	

Il faut ensuite modifier le modèle pour choisir l'unité avec laquelle le modèle s'exécute.



#### **Résultats d'exécution**



#### Ajout d'outils de visualisation

Il faut sélectionner **consumers** à partir du panel **Main**.

On sélectionne ainsi la classe consumers.

👸 Main 🛛 👸 Consumer 📃 🗖				Properties 🛛		
		^	🙆 consumers - Consumer			
				Name:	consumers	📝 Show nar
0 2 4 6 8 10				Single agent Oppulation	of agents	
1meter = 10px meter				Population is:	Initially empty	
					Contains a given number Loaded from database	of agents
connections				Initial number of agents:	=_ 5000	
				AdEffectiveness: = 0.0	91	
			►	Movement		
		E	2	Initial location		
consumers []			-	Statistics		
		->		No data items defined yet. Press	"+" to add a new data item.	
				<b>♀</b> 🖹 ≍ ि ८		
				Advanced		
			-	Description		

Il faut s'intéresser à l'onglet Statistics.

En utilisant <sup>(C)</sup> on peut définir des nouveaux attributs (qui sont en réalité des méthodes) pour la classe **Consumer** qui représente la liste de tous les **consumers**.

On compte l'ensemble de tous les consumer qui sont dans l'état PotentialUser.

Name:	NPotential
Туре:	Ocunt
Condition:	item.inState(Consumer.PotentialUser)

On peut définir de la même manière **NUser** pour compter le nombre de consumer dans l'état **User**.

Name:	NUser
Туре:	🖲 Count 🔘 Sum 🔘 Average 🔘 Min 🔘 Max
Condition:	<pre>v item.inState(Consumer.User)</pre>

L'ensemble des outils pour créer les rapports sont dans l'onglet :



Il faut déposer un Time Stack Chart sur le page principale.



Il faut modifier la partie Data.



On va représenter sur le schéma deux courbes montrant l'évolution du nombre de consommateurs dans chacun des deux états.

Utilisons le bouton Add data item pour ajouter des courbes....

La méthode recherche appartient à **consumers** et on peut utiliser le système de complétion automatique via CTRL + ESPACE.

▼ Data	TITR	ES	
💿 Value 💿 Data set	Title: Dataset Title	بدام میں ہ	
Value: consumers.	devo	tre doçi	
Color: chocolati	equals(Object obj) : boolean - Object		
	forEach(Consumer super Consumer action) : void	- Ite	
🖶 Add data item	getClass() : Class - Object		
	hashCode(): int - Object		
Data update	notify(): void - Object		
♦ Scale	notifyAll() : void - Object		
Appearance	NPotential() : intconsumers_Population		
Position and size	NUser() : intconsumers_Population		
▶ Legend	spliterator() : Spliterator <consumer> - Iterable</consumer>		
Chart area	wait(): void - Object	- 1	
Advanced	<	► a	
Description			

On peut alors choisir une couleur pour la visualisation et comme on a sélectionné le vers pour diagramme d'état, on peut faire le même choix pour être cohérent.

💿 Value 🔘 Data set 🛛 Tit	le: Dataset Title	
Value: consumers.NUser	()	」 ひ
Color: <u>yellowGreen</u>	•	x

On peut recommencer pour le nombre de consommateurs dans l'état Potential...

O Value	
Value: consumers.NPotential()	ъ
Color: lavender 🔻	x

Sur la page en cours de construction, les courbes se présentent comme suit :



Dans l'onglet Scale, on peut ensuite contrôler l'amplitude de la courbe. Ici on choisit une période de 1 an avec une amplitude en ordonnées de 5 000.

Properties		$\bigtriangledown$	
🕍 chart - Time Stack Chart			
Name: chart Ignore			
▶ Data			
► Data update			
Scale			
Time window: 1 years	-	-	
Vertical scale: 🔘 Auto 💿 Fixed 🔘 100%			
From: 0 To: 5000			

Le rafraichissement de la courbe peut lui aussi être paramétré via l'onglet **Data update** et **Appearance**.

▼ Data update				
Opdate data automatica	illy			
🔘 Do not update data auto	omatically			
Use model time OUse	e calendar dates			
First update time:	ays 🔻			
Update date:	05/01/2017			
Recurrence time:	ays 🔻			
Display up to 1365	latest samples (applies to "Value" data)			
▼ Scale				
Time window: 1	years 👻			
Vertical scale: 🔘 Auto 🧕	Fixed			
From: 0 To: 5000				
▼ Appearance				
Horizontal axis labels: Bel	low 🔻			
Vertical axis labels: Lef	ft 🖛			
Time axis format: 🔹 Model date (date only) 🖘				
Labels color:	darkGray 🔻			
Background color:	No fill 👻			
Border color:	No line 🔻			
Grid color:	darkGray 🔫			

Il faut penser à positionner les courbes à côté et pas sur le modèle... par exemple, en décalant les courbes en haut à droite.



### **Résultat d'exécution**



#### 3.3. Définir un comportement plus complexe du « consommateur » : notion d'interaction

On va ajouter dans le modèle des interactions entre les agents. Par rapport à ce qu'on fait classiquement en simulation à évènements discrets, on peut considérer qu'il s'agit d'envoi de messages entre des éléments de flux.

٠	
	AdEffectiveness
	ContactRate
	AdoptionFraction

Etape 1. Ajout de deux nouvelles variables globales On ajoute **ContactRate** et **AdoptionFraction**.

On peut maintenant attribuer des valeurs par défaut. Par exemple un **ContactRate** de 1 contact par jour.

🔲 Properties 🔀		1	$\bigtriangledown$	Ē
ContactRat	te - Parameter			
Name:	ContactRate	Show name		
Ignore				
Visible:	() yes			
Туре:	double 👻			
Default value:	=, 1			
🔲 System dyna	amics array			
Value editor				
Advanced				
Description				

Pour le taux de passage dans l'état User sur la réception d'un message, on définir 0.01 comme valeur (1% de chance).



#### Etape 2. Modification du schéma

Pour cela on va ajouter une transition interne à un état : ici l'état USER.

Pour cela il faut utiliser :

🔪 Transition

Ce type de transition définit une opération qui est effectuée de manière répétitive. Visuellement, le schéma se présente comme ceci :



Ceci signifie que périodiquement, selon le taux **ContactRate**, un agent émet un message **Buy** à destination d'un agent choisit aléatoirement.

<b>^</b>	Contact - Transition	
• statechart	Name: Contact Show name	*
PotentialUser	Triggered by:  Rate    Rate:  ContactRate    Action:  sendToRandom("Buy");	
User	Guard:  Description	

Un agent recevant le message « **buy** » va passer, dans certaines conditions dans l'état **User**. Il faut dont créer :

- Une transition déclenchée sur la réception d'un Message ;
- Spécifier le type de message, ici String ;
- Spécifier le contenu « buy » ;
- et définir une condition supplémentaire (ici on accepte la transition avec une certaine probabilité).

	🔲 Properties 🛛	
· · · · · · · · · · · · · · · · · · ·	🔾 transition -	Transition
• statechart	Name:	transition Show name
•	Triggered by:	Message 👻
PotentialUser	Message type:	String 👻
	Fire transition:	O Unconditionally
		On particular message
		If expression is true
User	Message:	"buy"
	Action:	
	Guard:	randomTrue(AdoptionFraction)

Etape 3. Vérifier l'exécution du modèle



**3.4.** Définir un comportement plus complexe du « consommateur » : les produits sont périssables et doivent être renouvelés.



La variable a une valeur de 6 mois.

👩 Main 🛛 👩 Consumer 🛛	- 8	Properties 🛛				~ -	
	· · · · · · · · · · · · · · · · · · ·	C DiscardTim	ne - Parameter				
AdEffectiveness	statechart	Name: Visible:	DiscardTime	Show name	🔲 Ignore		*
ContactRate		Туре:	Time 🔻				
AdoptionFraction	PotentialUser	Unit:	months -				
DiscardTime		Default value:	amics array				
		► Value editor					
	User	Advanced					
		<ul> <li>Description</li> </ul>					

Ce qu'on va exprimer c'est qu'au bout de 6 mois, le consommateur repasse dans l'état **PotentialUser**.

# Etape 2. Ajout d'une nouvelle transition

	□ Properties 🛛 📑 🖓 🖓 🗖
	🖌 transition1 - Transition
	Name: transition1
• statechart	Show name 🔲 Ignore
•	Triggered by: Timeout 👻
PotentialUser	Timeout: 😡 DiscardTime months 🔻
	Action:
<b>₽ ↓ ↓</b>	Guard:
User	Description

La transition est validée sur une condition temporelle de valeur **DiscardTime**.

### Etape 3. Résultat d'exécution



# **3.4.** Définir un comportement plus complexe du « consommateur » : prise en compte d'un délai de livraison

# Etape 1. Ajout d'un nouvel état

Il faut modifier le schéma et inclure un nouvel état WantsToBuy.



🔲 Properties 🛛				
WantsToE	Buy - State			
Name:	WantsToBuy	Show	name	*
Ignore				
Fill color:	gold			
Entry action:	shapeBody.setFillCol	.or(gold)	);	
Exit action:				

Il faut ajouter un passage de l'état **WantsToBuy** à celui de **Users** comme ci-dessous.

	Properties 🛿 📑 🔽 🗖
● statechart	<b>Q</b> Purchase - Transition
	Name: Purchase
Patantial Isar	📝 Show name 🛛 Ignore
n	Triggered by: Timeout 👻
	Timeout: 2 days 🔻
	Action:
	Guard:
	Description
WantsToBuy	
Discard	
<u>Purchase</u>	
User	

# Etape 2. Ajout de nouvelles statistiques

Il faut sélectionner

0	consumers []	

pour ajouter une nouvelle méthode.

Actuellement 2 méthodes figurent dans la partie Statistics.

(Oconsumers [_]	▼ Statistics	
	Name:	NPotential
	Туре:	O Count
	Condition:	item.inState(Consumer.PotentialUser)
	Name:	NUser
	Туре:	🖲 Count 🔘 Sum 🔘 Average 🔘 Min 🔘 Max
E	Condition:	item.inState(Consumer.User)
	¢ 🗎 ک	s 🔂 🕹

La méthode est nommée **NWantToBuy**.

Name:	WantToBuy
Туре:	💿 Count 🔘 Sum 🔘 Average 🔘 Min 🔘 Max
Condition: item.inState(Consumer.WantsToBuy)	

Seules deux courbes sont utilisées sur le schéma. Pour les consulter, il faut « retrouver » le dessin et consulter l'onglet **Data**.

👸 Consumer	🗖 🗖 Properties 🛛 📃	¶ ▽ □ □
	🔶 chart - Time Stack Chart	
	Name: chart Ignore	^
	Visible on upper level	
- <b>Ç</b>	□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	
4,000 -		
	Value: consumers.NUser()	
3,000	Color: yellowGreen 🔻	
2,000	● Value ○ Data set Title: Dataset Title	
1,000	Value: consumers.NPotential()	
0	May 1, 2017 Color: lavender -	-
Dataset Title     Dataset Title	🖨 Add data item	

On ajoute une courbe représentant le nombre de consommateurs... « en attente » d'un achat....

O Value      O Data set Title: Want to buy	
Value: consumers.NWantToBuy()	ப் த
Color: gold -	x

Pour augmenter la lisibilité, on peut en profiter pour modifier le champ **Title** des différentes courbes.

Name:	chart Ignore Visible on upper level	
• Data		
🔘 Valu	e 🔘 Data set 🛛 Title: Users	
Value:	consumers.NUser()	
Color:	yellowGreen 🔻	
Value:	e O Data set (itle: Users Potential	÷
Color:	lavender -	₽ ¤
Valu	e 💿 Data set 🛛 fitle: 🛛 Want to buy	
Value:	consumers.NWantToBuy()	
Color:	gold -	
🖶 Add	data item	

# Etape 3. Résultat d'exécution



# 4) Modèle de simulation dynamique : machine à état

# 4.1. Création d'un modèle.

Il faut créer un projet nommé SEIR qui va simuler une épidémie.

ſ	春 New Model				
	New Model				
	Create a new mod	el			
	<u>M</u> odel name:	SEIR			
	Location:	sktop\prins_novembre_2016\essai_simio\Anylogic\Philippe\ Browse			
ľ	Java package:	seir			
	Model time units:	days 👻			
	The following mod	lel will be created:			
	C:\Users\lacomme\Desktop\prins_novembre_2016\essai_simio\Anylogic\Philippe\SEIR \SEIR.alp				
		Finish Cancel			

L'ensemble des composants dont nous allons avoir besoin sont dans l'onglet *i.e.* System Dynamics.

# 4.2. Modélisation

### Etape 1. Définition des états

On introduit 4 états à partir du composant Stock de l'onglet System Dynamics.



# Il faut ajouter ensuite les flots.



Susceptible Exposed Infectious Recovered

On peut ensuite donner un nom aux flots décrivant les changements d'état.

#### Ajouter ensuite 5 variables globales



Avec comme valeur :

- TotalPopulation = 10 000
- Infectivity = 0.6 avec le jour comme unité (days)
- ContactRateInfectious = 1.25 avec le jour comme unité (days)
- AverageIncubationTime = 10 avec le jour comme unité (days)
- AverageIllnessDuration = 15 avec le jour comme unité (days)

### Etape 2. Définition du stock initial (état : Susceptible)

On définit ensuite une valeur pour le nombre initial de personnes infectées. Ici 1 personne, ce qui signifie que le nombre de personnes non infectées mais susceptible de l'être est : TotalPopulation-1.



Il faut noter le symbole route à côté de Initial Value qui montre qu'on ne peut pas utiliser la variable TotalPopulation car celle-ci est inconnue au niveau du stock. Il faut ajouter un lien entre la variable TotalPopulation et Susceptible.



On suppose qu'un seul individu est dans l'état Infectious et on définit 1 comme valeur initiale.



Etape 3. Passage de Suceptible à exposer.

Le nombre de personnes exposées est défini par :

Infectious\*ContactRateInfectious\*Infectivity\*Susceptible/TotalPopulation II faut saisir cette information sur le flow.

	A SexposedRate - Flow
Susceptible Exposed	Name:       ExposedRate       Show name       Ignore         Visible on upper level         Visible:       yes         Color:       Default          Array       Dependent       Constant         ExposedRate=       Infectious*ContactRateInfectious*Infectivity*Susceptible/TotalPopulation
	Array dimensions
	▼ Advanced
Total Population	System dynamics units:

Pour les mêmes raisons que précédemment, il faut relier les variables globales au flot pour déclarer leur utilisation.

Susceptible	e Exposed	Infectious	Recovered
	ExposedRate	InfectiousRate	RecoveryRate
TotalPopulation		ContactRateInfectious	AverageIIInessDuration
	Infectivity		AverageIncubationTime

#### Etape 4. Passage d'Exposed à Infectious

Le nombre de personnes infectées est donné par le nombre de personnes exposés divisé par le temps moyen d'incubation : Exposed/AverageIncubationTime



#### Etape 5. Passage de Infectious à Recovered

Le nombre de personne ayant retrouvé la santé, est défini par le nombre de personne malade/la durée moyen de guérison : Infectious/AverageIllnessDuration



# 4.3. Exécution du modèle



#### 4.4. Visualisation de la dynamique du système



Pour mieux appréhender la dynamique, on ajouter une boucle de renforcement nommée **Contagion**.

Le type de renforcement doit être R pour « Renforcement » et dépendant de la date (Clockwise).

🔲 Properties 🛛	🛃 🗸 🗖 🖬	3
🕞 loop - Loo	р	
Ignore		*
Direction ()	Clockwise 🕓 Counterclockwise	
Туре: 🔘	B 🔍 R 🔘	
Color:	Default 👻	
Text: C	ontagion	

Il faut poser sur la feuille un élément graphique de type Time Plot.



Il faut ajouter 4 courbes.

▼ Data	
🖲 Value 🔘	Data set
Title:	Susceptible People
Value:	Susceptible
Point style:	
Line width:	T pt
Color:	dodgerBlue 🔻
🔘 Value 🔘	Data set
Title:	Exposed People
Value:	Exposed
Point style:	
Line width:	
Color:	darkOrange 🔹

	Data set
Title:	Infectious People
Value:	Infectious
Point style:	
Line width:	<b>v</b> 1 pt
Color:	magenta 🔹
◉ Value	Data set Recovered People
<ul> <li>● Value</li> <li>●</li> <li>Title:</li> <li>Value:</li> </ul>	Data set Recovered People Recovered
<ul> <li>Value</li> <li>Title:</li> <li>Value:</li> <li>Point style:</li> </ul>	Data set       Recovered People       Recovered
<ul> <li>Value</li> <li>Title:</li> <li>Value:</li> <li>Point style:</li> <li>Line width:</li> </ul>	Data set       Recovered People       Recovered



#### 5) Modèle de simulation spatial et discrets

Le modèle est nommé JobShop pour respecter le choix réalisé dans le document pdf original mais le nom est un peu mal choisi.

#### 4.1. Création d'un modèle.

A New Model □ ■ X		
New Model		
Create a new mod	lel	
<u>M</u> odel name:	JobShop	
Location:	esktop\prins_novembre_2016\essai_simio\Anylogic\Philippe Browse	
Java package:	jobshop	
Model time units:	minutes 👻	
The following mod	del will be created:	
C:\Users\lacomme\Desktop\prins_novembre_2016\essai_simio\Anylogic\Philippe \JobShop\JobShop.alp		
	Finish Cancel	

Il faut insérer une image et choisir l'image qui se trouve dans le répertoire suivant :

C:\Program Files\AnyLogic 7 Personal Learning Edition\resources\AnyLogic in 3 days\Job Shop Cette image représente la vue aérienne des salles d'un atelier de production.



Une fois choisie la disposition de l'image, il est recommandé de passer l'attribut Lock à true, ce qui évitera des déplacements intempestifs de l'image.

- 8	Properties 🛛 📑 🍷 🗖
<b>^</b>	😼 image - Image
	Name: image Ignore
	🖉 Visible on upper level 🛛 Icon 📝 Lock
	Visible: 😑 💿 yes
	Reset to original size
	Images: layout.png 🚺 🕨 🖶 Add image 🖾
	Position and size
	Advanced
	Description

#### 4.2. Création d'un réseau de transport



Un Rectangular Node (section Space Markup) est déposé sur le dessin.

Ce nœud est nommé receivingDock.



Un deuxième nœud est ajouté et nommé forkliftParking



On définir ensuite un chemin <sup>2</sup> Path entre ces nœuds, chemin qui servira à animer le déplacement des objets et des véhicules.

Space Markup	88 88 B	and shares	and the second	No. of Contraction		and the second second	
B 2 Path							
🛧 🛴 Rectangular Node		102					
📊 🛅 Polygonal Node			No. of the Party o	Contract of the second			
🗮 🔌 Point Node		and the second	C. C	10000			
-ộ- Attractor				1200			
Pallet Rack		A PROPERTY	A STATE OF	States a			-
🕫 🔻 GIS				121223			
GIS Map						10000	
🔮 💡 GIS Point		and the second second	C. S. S. S. S.	1.2.22 Files		Constant	-
CIS Route		and the state of the					
🚺 😋 GIS Region				Design of the			
🕜 🔞 Route Provider				The second			
<ul> <li>Pedestrian</li> </ul>	5						-
🖹 🗖 Wall	🖉 = 🕌	-					
Rectangular Wall		A STATE OF THE STATE OF	No. of Concession, Name	C. C	Contraction of the local data	Contraction of the second	100
🛓 🔘 Circular Wall							
<b>x</b>					A CONTRACTOR OF A CONTRACTOR		

Il faut déposer une Pallet Rack sur le schéma en intersection avec le chemin. Attention, lors d'un positionnement correct (le rack est situé sur le chemin) une partie rectangulaire verte doit apparaître.



On définit une zone de stockage avec deux éléments et donc de type **Two racks and one aisles** configuré comme suit :

palletRack - Pallet Rack				
Name:	palletRack 🔲 Ignore			
Visible on upper level	Lock			
Visible:	yes			
Туре:	Two racks, one aisle 🔻			
Number of cells is:	Oefined explicitly			
	Calculated based on the cell width			
Number of cells:	10			
Number of deep positions:	1			
Number of levels	1			
Number of levels.				
Level height:	10			

On peut ensuite modifier les paramètres de **Position and size** comme suit :

<ul> <li>Position and size</li> </ul>		
X:	520	
Y:	290	
Z:	0	
Length:	160	
Left pallet rack depth:	14	
Right pallet rack depth:	14	
Aisle width:	11	
Rotation, °:	• 0.0	

# 4.3. Définition d'un routage des pièces

On utilise une source qu'on configure comme indiqué ci-dessous.



#### On ajout un rackStore à la suite.



On le relie ensuite à la source.

source	storeRawMaterial
+>•	• <sup>-</sup> •

Il faut mettre à jour les attributs qui font le lien avec les objets graphiques...

🔲 Properties 🛛	<b>1</b> - V
r 🗄 storeRawMaterial -	RackStore
Name:	storeRawMaterial 🕼 Show name 🔲 Ignore
Pallet rack / Rack system:	and palletRack 🔽 😰 ট
The cell is:	=_ Chosen automatically 👻
Choose cell closest to:	Front of storage/zone 👻
Move agent to:	=_ cell (with elevation) 👻
Elevation time per level:	☐ 10 seconds ▼
Use delay:	=, 🔲
Agent location (queue):	🗧 🔁 receivingDock 🔹 🖓

Les liens sont les suivants :



On ajoute ensuite un **délais**.

source storeRawMaterial		raw Material In Storage		
+>+	┉ッ╴			

On le configure pour simuler une durée qui suit une loi triangulaire.

🔲 Properties 🛛		1 2 1				
🕓 rawMaterialInStorage - Delay						
Name:	rawMaterialInStorage	V Show name	^			
🔲 Ignore						
Туре:	<ul> <li>Specified time</li> <li>Until stopDelay() is called</li> </ul>					
Delay time:	🖓 🗌 triangular( 15, 20, 30)	minutes 🔻				
Maximum capacity:	=, 🖉					

On ajoute un rackPick.

source	storeRawMaterial	rawMaterialInStorage	rackPick
+>>			

Et on relie cette opération aux différents objets dont le **palletRack**...

🔲 Properties 🔀		₫ ▽□□	
By rackPick - RackPick			
Name: Ignore	rackPick	Show name	
Pallet rack / Rack system:	=_ palletRack	- R 🗘	
Destination is:	= Network node -		
Node:	= 🔁 forkliftParking	🗘 🎜 👻	

Il faut ajouter ensuite un puits.

source	storeRawMaterial	rawMaterialInStorage	rackPick	sink
• <del>&gt;-</del>	• ^>릴 •		<b>[</b> ]``	

#### Résultat d'exécution.

JobShop : Simulation - AnyLogic PLE [PERSONAL LEARNING USE ONLY]	
00 🕶 🕨 🔳   💁 💽 x5 💽 🥵 🚱 🚳 🚳 root:Main 🚽 🐚   D	💥 AnyLogic
source storeRawMaterial rawMaterialInStorage rackPick sink の 15 15 15 15 15 15 10 10 日か 10 日か 10 日の	
Run: 0 💽 Running   Time: 78.96   Simulation: Stop time not set   Date: Jan 5, 2017 1:18:57 AM   📐   Memory:	32M.of.228M

# 4.3. Ajout de ressources

Il faut ajouter un pool de ressources au modèle.



On nomme ce pool de ressources **forklift** en modifiant l'attribut **name**.

Properties 🔀	<b>⊡</b> ⊽	
** forklift - ResourcePo	bl	
Name:	forklift 🛛 🐼 Show name	e
Ignore		
Resource type:	= Moving -	
Capacity defined:	Tirectly	
Capacity:	=, 1	
When capacity decreases:	units are preserved ('End of shift') 🔻	
New resource unit:	=_ 🚯 Agent 🔻	=
	create a custom type	
Speed:	= 10 meters per second	-
Home location (nodes):	=,	
	🚽 X 🖓 X 🙀	

On va créer un type d'agent particulier pour modéliser les mouvements des chariots transportant les pièces.

On nomme ces nouveaux agents : ForkliftTruck.

A New agent	
Step 1. Creating new agent type	
Agent type name: ForliftTruck	
Oreate the agent type "from scratch"	
Use database table	
Thave agent data stored in a database	
< <u>B</u> ack <u>N</u> ext > Finish	Cancel

On choisit un objet dans la librairie d'objets 3D qui modélise au mieux un chariot.

<mark>₄</mark> New agent	
Step 2. Agent animation	
Choose animation: 💿 3D 💿 2D 💿 None	
Maritime Transport	
Military	
✓ Warehouses and Container Ter	la
🕴 Worker	
🔰 Fork Lift Truck 🥏	7
w Pallet	
< Back Next > Finish	Cancel

Une fois crée, le chariot apparaît dans un onglet nommé ForkliftTruck.



L'objet chariot peut être sélectionné à la souris et on peut accéder à ses propriétés.



On revient au forklif

	New resource unit:	= O ForliftTruck -
and the second	Speed:	=_ 10 meters per second v
	Home location (nodes):	ੂ ਡਾ ¤ ♡ ∿ ⊕
source storeRawMaterial ra	Shifts, breaks, failures, ma	intenance
·····································	▼ Advanced	
	Add units to:	=_ @ default population
<u>forklift</u> ≡		custom population
<u> </u>	Force statistics collection:	=, 🗉
	A	1

On spécifie alors la position initiale (dans le forkliftParking) et la vitesse en mètre par secondes.

New resource unit:	= 6 ForliftTruck -	
Speed:	=, 1	meters per second 🔻
Home location (nodes):	n 🔁 forkliftParking	
	📮 🕁 🗶 🕱	

On sélectionne le **storeRawMaterial** et on définit les ressources à déplacer en cochant la case dans l'onglet **Resources**.

storeRawMaterial ra	▼ Resources Use resources to move: Resource sets (alternatives): ■ 企 ひ 次 定 ● Add list
	Move at the speed of resource:

On définit ensuite les règles gérant les déplacements des chariots.

▼ Resources	
Use resources to move:	
Resource sets (alternatives):	= frit forklift
	🖶 슈 문 XX 🖡
	🖶 Add list
Move at the speed of resource:	=, 🔲
Task priority:	0
Task may preempt:	
Task preemption policy:	No preemption
After release resources:	<ul> <li>Return to home location</li> <li>Stay where they are</li> </ul>
Return home:	=_ () each time
	if no other tasks
	Custom

#### 4.5. Exécution du modèle

Properties		🛃 🗢 🗖
铈 forklift - ResourcePo	loo	
Name:		forklift
🔽 Show name 📃 Ignore		
Resource type:	=,	Moving -
Capacity defined:	=,	Directly -
Capacity:	=,	5
When capacity decreases:	=,	units are preserved ('End of shift') 👻

Afin d'obtenir un modèle intéressant on peut fixer à 5 le nombre de chariots.

Le résultat est celui fournit ci-dessous :



# 4.6. Ajout d'une animation 3D

Il faut poser un objet **3D Window** sur le modèle de sorte qu'il se présente comme ceci :

			o			
			<b></b>			
			0			
	source	storeRawMaterial		rackPick	sink	
	source	storeRawMaterial	rawMaterialInStorage	rackPick	sink	
forklift	source	storeRawMaterial	rawMaterialInStorage	rackPick 	sink	
forklift	source	storeRawMaterial	rawMaterialInStorage	rackPick	sink	

Il faut ensuite poser et orienter une caméra.



On fait le lien entre la scène et la caméra : on modifier les attributs de windows3d comme suit :



On obtient à l'exécution une vision 3D de la scène.

