

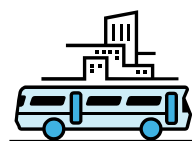
Chapter 7

運輸與供應鏈模擬

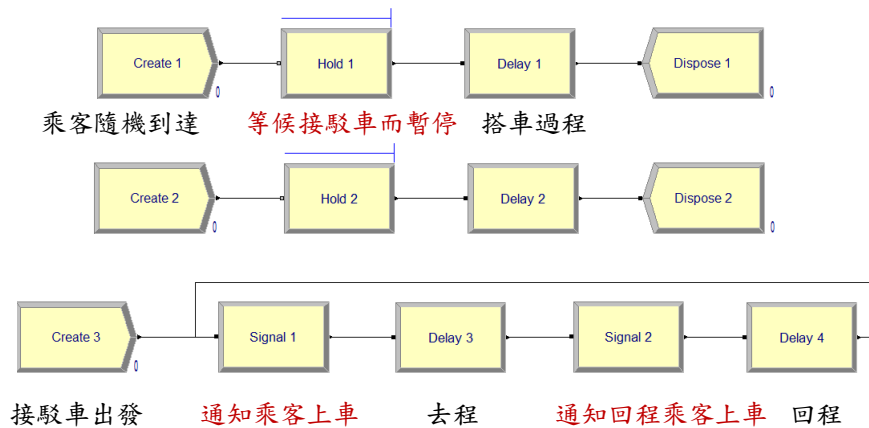
- 利用Hold與Signal模組控制流程暫停與繼續進行
- 使用Set資料模組，讓entity可以由不同類型的資源來處理
- 使用Variable與Set資料模組合併相似的流程，但是仍然保持各自運作

7.1 接駁巴士模擬

- 設某城市提供接駁車載運乘客往返於高鐵站與旅遊景點之間，中途沒有停靠站，管理者可以控制接駁車數量與發車間隔，希望以模擬評估對載客率與乘客等候時間的影響。
- 模擬包含三個同時進行的流程：
 - (1) 乘客在高鐵站等候接駁車，並搭車前往旅遊景點。
 - (2) 乘客在旅遊景點等候接駁車，並搭車前往高鐵站。
 - (3) 接駁車往返行駛於兩地之間的過程。

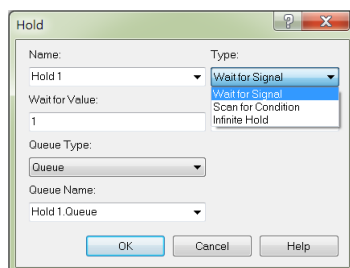


接駁巴士的初步流程設計



Question: 如何控制上車人數？如何讓人車同時到達？

乘客流程的Hold模組

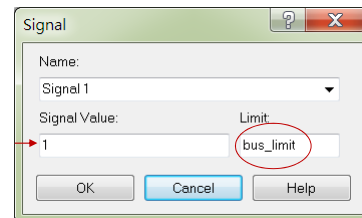
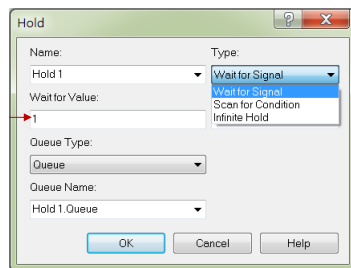


Hold模組內的Type選項有三個，代表流程恢復進行的方式。

- Wait for Signal是被動等待其他流程的Signal模組送出信號
- Scan for Condition是主動觀察某個條件是否成立。
- Infinite Hold終止流程，entity進入休眠狀態，必須要由其他流程的模組在其他位置取出。

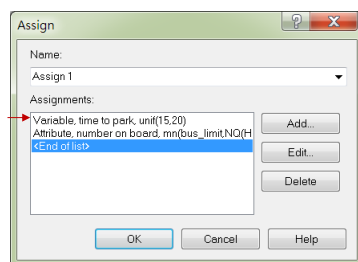
- Wait for Value選項用數值來區分不同信號
- Limit欄位空白，表示不限制可接收信號的entities數量
- Hold 1.queue儲存暫停的entities

接駁車流程的Signal模組



- Signal Value: 1代表通知高鐵站乘客上車的信號，2代表通知旅遊景點乘客上車的信號。
- Limit欄位設定發送信號的上限，代表接駁車載客上限，可輸入25，或輸入變數bus_limit，欄位空白則無上限。

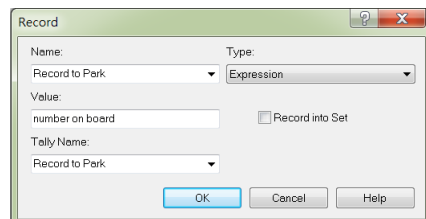
接駁車記錄載客人數



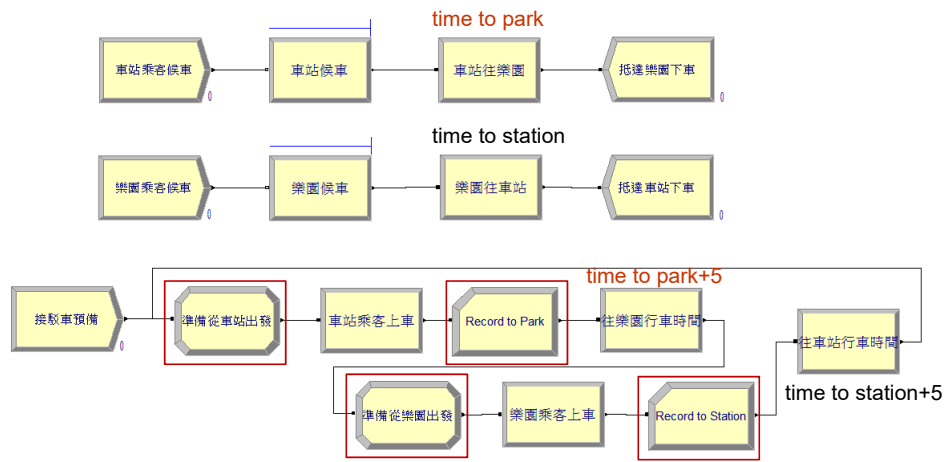
初步流程設計未記錄載客數

載客人數number on board等於
 $\min(\text{bus_limit}, \text{NQ}(\text{Hold 1.Queue}))$

變數time to park控制乘客流程與接駁車流程同步



接駁巴士的完整流程設計



Example 7-1 模擬結果

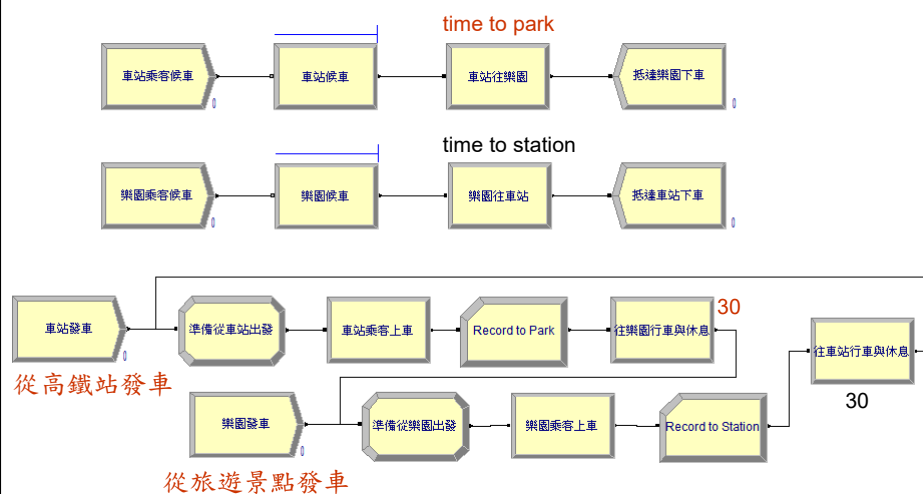
Replication Length = 18 hours

	高鐵站出發	旅遊景點出發
乘客平均等待時間	8.3852	8.2789
乘客等待時間最大值	25.4899	26.9825
排隊人數最大值	37	35
平均載客數	13.5368	14.4681

改變發車方式的影響

- 原有方式為所有接駁車都從高鐵站發車，起始的間隔為15分鐘，接下來都是到站後停留5分鐘再行駛回程。
- 由於行駛時間變化，容易造成發車間隔不穩定。
- 新的發車方式是各有兩部車從高鐵站與旅遊景點發車，起始的發車間隔仍為15分鐘。規定每班車從任何一站出發後半小時，就從另一站發車回來。
- 降低不確定性：對駕駛員而言，等於每趟的行駛時間與休息時間相加為30分鐘，對旅客而言，等於固定每隔15分鐘就發車。

改變發車方式後的流程



模擬18小時的結果比較

Example 7-1

	高鐵站出發	旅遊景點出發
乘客平均等待時間	8.3852	8.2789
乘客等待時間最大值	25.4899	26.9825
排隊人數最大值	37	35
平均載客數	13.5368	14.4681

Example 7-2

	高鐵站出發	旅遊景點出發
乘客平均等待時間	7.8621	7.5479
乘客等待時間最大值	19.764	15.000
排隊人數最大值	31	25
平均載客數	19.291	17.791



以上的討論僅根據一次replication的模擬結果 pickup, dropoff

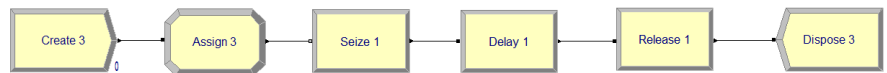
7.2 多廠區訂單分配問題

- 水果商不定期釋出product 1, product 2的緊急訂單，分給兩家代工夥伴。A夥伴的工廠為：Kaohsiung(2), Dalian(1), Shanghai(1, 2)，B夥伴的工廠：HsinChu(1), Shenzhen(2)
- 各廠平常處理來自一般客戶的訂單，當接到水果商緊急訂單時，會優先安排上線生產，一次只處理一張訂單。
- 緊急訂單的分配原則是有產能可立即生產者優先，如果有兩個廠可選擇時，指派的優先順序為：

priority	Product 1	Product 2
1	Dalian (A)	Shanghai (A)
2	Shanghai (A)	Kaohsiung (A)
3	HsinChu(B)	Shenzhen (B)

Modeling Issues

- Entity: 五個廠的一般訂單流程，兩種緊急訂單的流程
- Resource: 五個廠的生產線
- 各廠的一般訂單流程相似，可以使用共通的流程模組，以簡化模式架構。



Q1: 如何讓相似流程共用模組？

Q2: 如何分配緊急訂單給各廠？

Q3: 如何讓各廠同時處理一般訂單與緊急訂單？

以集合來處理多類型與多資源的流程

- 先在Resource資料模組定義五個資源，名稱為Dalian, HsinChu, Kaohsiung, Shanghai, Shenzhen，然後在Set資料模組定義一個集合 **factory set**，類型為Resource。

Set - Basic Process			
	Name	Type	Members
1	Set P1	Resource	3 rows
2	Set P2	Resource	3 rows
3	factory set	Resource	5 rows
4	Set Entity Type	Entity Type	5 rows

Double-click here to add a new row.

Members	
	Resource Name
1	Dalian
2	Hsinchu
3	Kaohsiung
4	Shanghai
5	Shenzhen

Set - Basic Process			
	Name	Type	Members
1	Set P1	Resource	3 rows
2	Set P2	Resource	3 rows
3	factory set	Resource	5 rows
4	Set Entity Type	Entity Type	5 rows

Double-click here to add a new row.

Members	
	Entity Type
1	Order 1
2	Order 2
3	Order 3
4	Order 4
5	Order 5

Set Entity Type

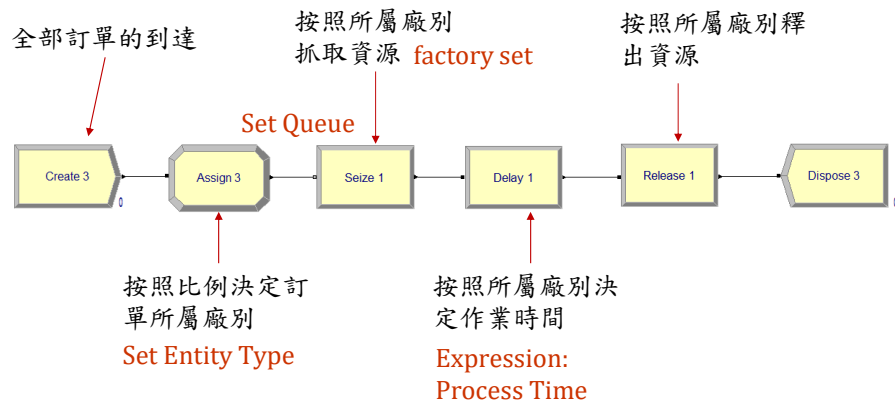
Advanced Set - Advanced Process			
	Name	Set Type	Members
1	Set Queue	Queue	5 rows

Double-click here to add a new row.

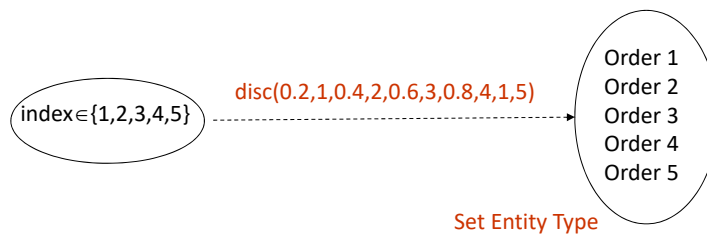
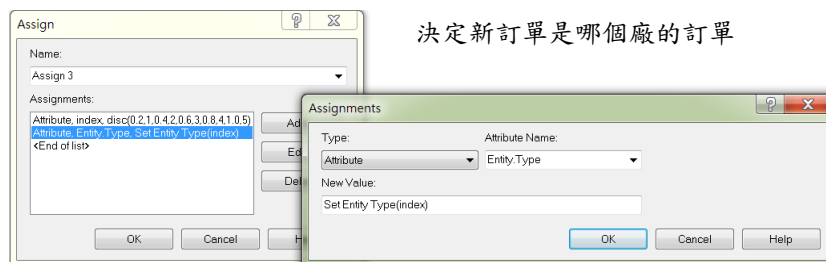
Members	
	Queue Name
1	Queue 1
2	Queue 2
3	Queue 3
4	Queue 4
5	Queue 5

Set Queue

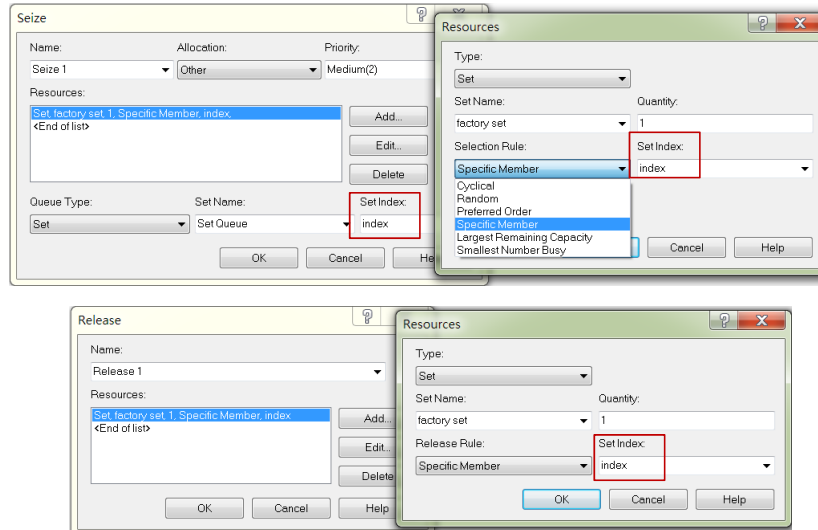
各廠一般訂單的共用流程



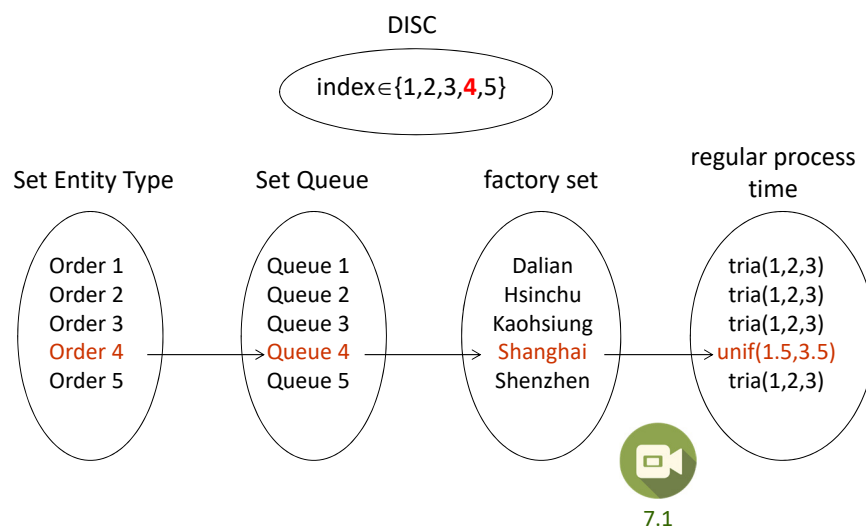
屬性index與entity type的對應



以index屬性區分不同廠的生產流程

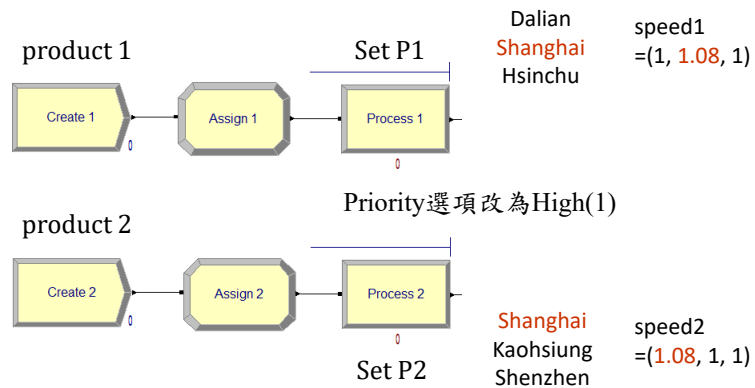


集合之間的對應關係



緊急訂單流程

Shanghai廠因管理問題，需要比其他廠多花費8%的時間來處理緊急訂單



緊急訂單按照優先順序分配

Set - Basic Process

Name	Type	Members
1 Set P1	Resource	3 rows
2 Set P2	Resource	3 rows
3 factory set	Resource	5 rows
4 Set Entity Type	Entity Type	5 rows

Members

Resource Name
1 Dalian
2 Shanghai
3 Hsinchu

優先順序

Process

Name: Process 1

Type: Standard

Logic

Action: Seize Delay Release

Priority: High(1)

Resources:

Set SetP1.1 Preferred Order index

Delay Type: Expression

Units: Days

Expression: order_size*speed 1(index)

Report Statistics

Resources

Type: Set

Set Name: Set P1

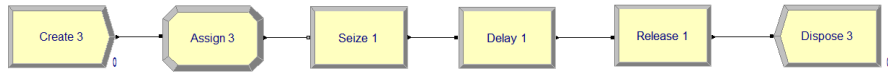
Quantity: 1

Selection Rule: Preferred Order

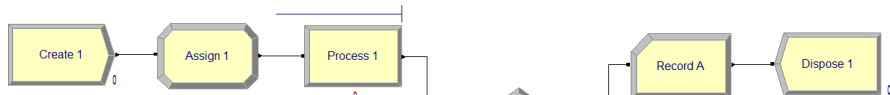
Save Attribute: Index

完整的流程模組設計

共用的一般訂單流程



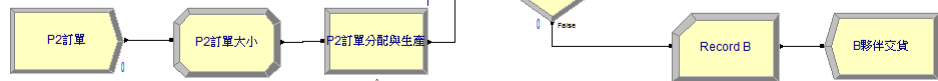
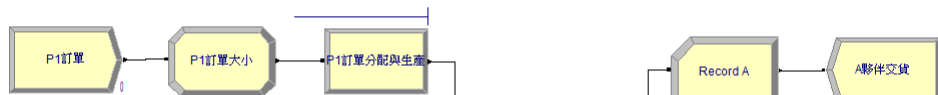
Product 1 緊急訂單流程



Product 2 緊急訂單流程



$index \leq 2 \Rightarrow A$ 代工



Record A → A夥伴交貨

Record B → B夥伴交貨

Statistics模組定義績效指標

Statistic - Advanced Process					
	Name	Type	Expression	Report Label	Output File
1	product 1 time	Output	TAVG(product 1.TotalTime)	product 1 time	
2	product 2 time	Output	TAVG(product 2.TotalTime)	product 2 time	
3	ratio A	Output	$NC(\text{Record A}) / (NC(\text{Record A}) + NC(\text{Record B}))$	ratio A	
4	avg order 1 time	Output	TAVG(Order 1.TotalTime)	avg order 1 time	
5	avg order 2 time	Output	TAVG(Order 2.TotalTime)	avg order 2 time	
6	avg order 3 time	Output	TAVG(Order 3.TotalTime)	avg order 3 time	
7	avg order 4 time	Output	TAVG(Order 4.TotalTime)	avg order 4 time	
8	avg order 5 time	Output	TAVG(Order 5.TotalTime)	avg order 5 time	

Example 7-3 模擬結果

- replication length: 300 days
- warm up period: 30 days
- number of replications: 30

	Average	Half-Width
Product 1 Time	2.4684	0.02890
Product 2 Time	3.5267	0.04009
Dalian Utilization	0.92475	0.01699
Hsinchu Utilization	0.92870	0.01628
Kaohsiung Utilization	0.94065	0.01837
Shanghai Utilization	0.99569	0.00402
Shenzhen Utilization	0.90453	0.02060
Order Ratio for A	0.62343	0.01709

ShenZhen廠產能變動的影響

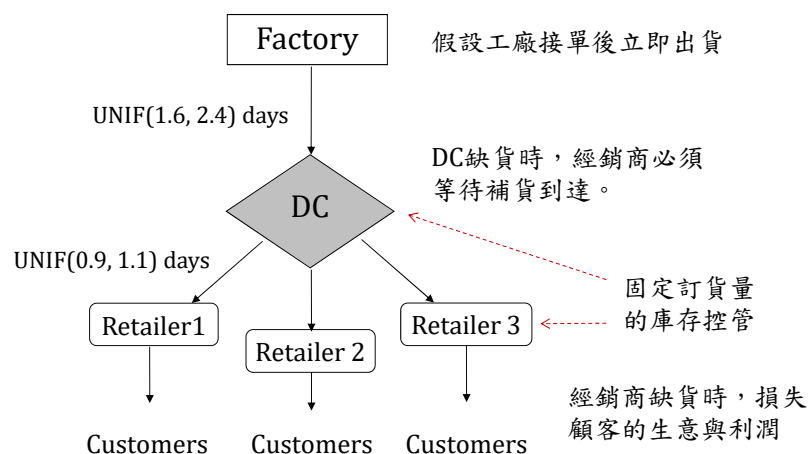
ShenZhen廠將可接受product 1的緊急訂單，優先順序排在Hsinchu廠之前。

- 更改Set P1的成員架構
- 修改speed1為(1, 1.08, 1, 1)

Dalian
Shanghai
ShenZhen
Hsinchu

	Average	Half-Width
Product 1 Time	2.3103	0.03582
Product 2 Time	3.5595	0.05383
Dalian Utilization	0.89510	0.01860
Hsinchu Utilization	0.90336	0.01871
Kaohsiung Utilization	0.93132	0.02083
Shanghai Utilization	0.99539	0.00505
Shenzhen Utilization	0.96193	0.01679
Order Ratio for A	0.57466	0.01824

7.3 三階層供應鏈庫存管理



供應鏈庫存管理流程

- 供應鏈的流程可分為三部份：經銷商銷售流程、經銷商向DC進貨的流程、以及DC向工廠進貨的流程。
- 三個流程相互關聯：經銷商銷售後可能因庫存降至訂貨點而啟動向DC補貨的流程，DC出貨給經銷商後可能啟動向工廠訂貨的流程，**工廠補貨到達DC後可能啟動出貨給欠貨待補經銷商的流程。**
- 模擬的目的是調整各成員的庫存控管參數，期望以較低的整體成本達到較高的供貨水準(fill rate)。

績效定義

平均單日的庫存相關成本包含：

- 全體成員的庫存持有成本，包含在途庫存，單位持有成本為\$0.3
- 經銷商的缺貨損失，單位缺貨損失為\$3
- 經銷商的訂貨成本，單次成本為\$64
- DC的訂貨成本，單次成本為\$200

供貨水準 (Fill Rate)

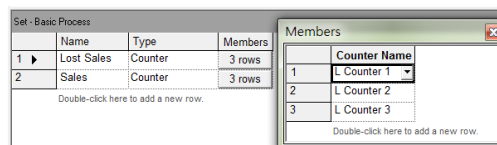
- 全體經銷商可滿足的需求比例

供應鏈模擬的變數設定

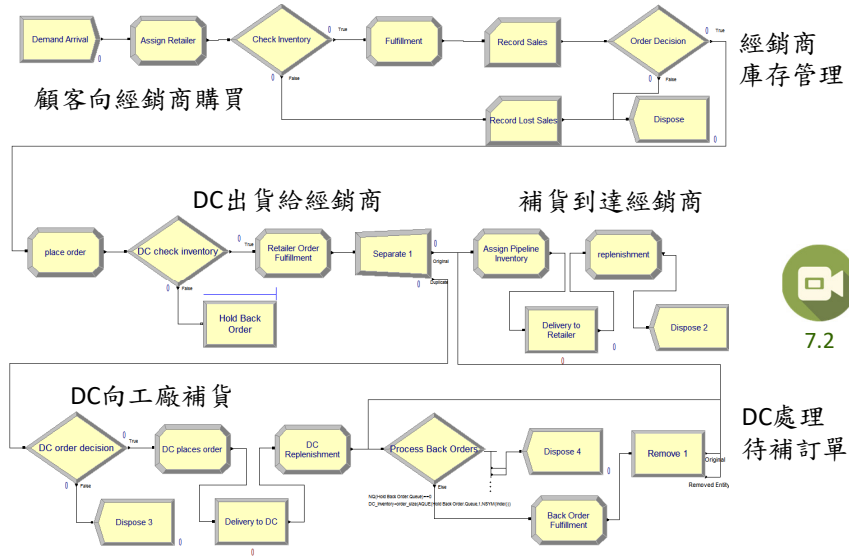
Variable	定義	預設值
inventory(index)	經銷商現有庫存量	30, 25, 25
order_size(index)	經銷商下單的固定訂貨量	60, 50, 50
reorder_pt(index)	經銷商訂貨點	20, 20, 20
on_order(index)	尚未送達經銷商之訂購量	
order_count(index)	經銷商的訂單次數	
DC_inventory	DC現有庫存量	80
DC_order_size	DC每次下單的訂貨量	200
DC_reorder_pt	DC訂貨點	60
DC_on_order	尚未送達DC之訂購量	
DC_order_count	DC的訂單次數	
DC_backorder	DC欠經銷商的總數量	
pipeline	DC運往經銷商的在途庫存	

以集合方式記錄經銷商的銷售與損失

Counter set	定義
Lost Sales(index)	累計經銷商的生意損失
Sales(index)	累計經銷商的銷售量



三階層供應鏈的流程模組



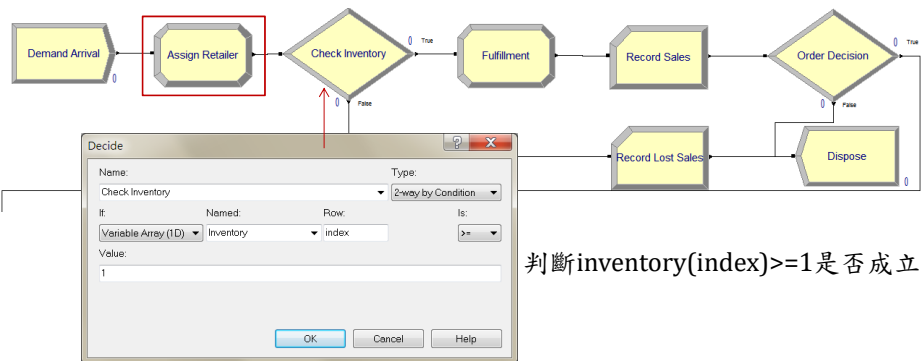
NKFUST Simulation

Chapter 7 – Transportation and Supply Chain Operation

Slide 31

顧客需求到達經銷商

經銷商的銷售比例為40% : 35% : 25%，因此在此Assign模組以DISC(0.4,1,0.75,2,1.0,3)隨機決定顧客屬性index。

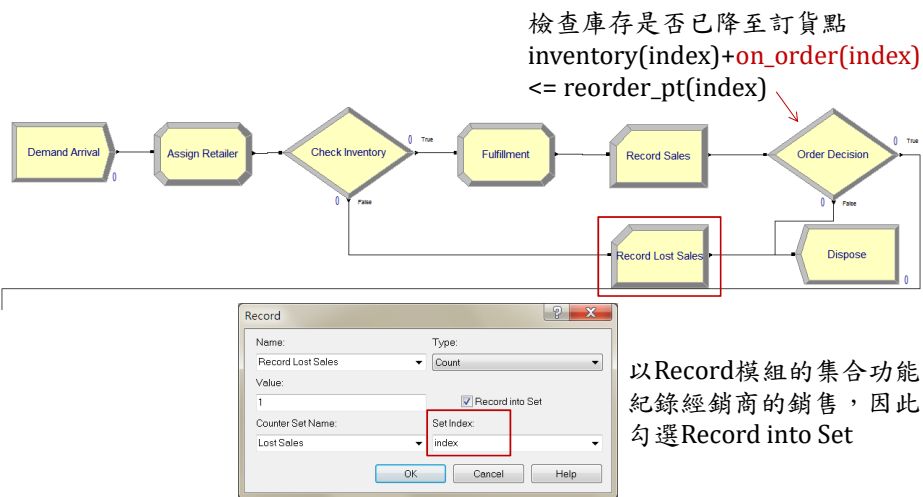


NKFUST Simulation

Chapter 7 – Transportation and Supply Chain Operation

Slide 32

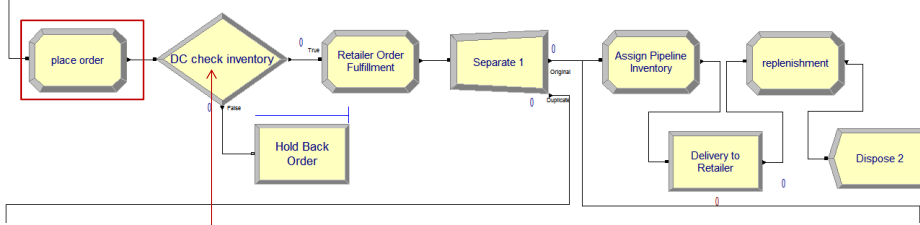
經銷商紀錄銷售損失與檢查庫存



經銷商向DC發出訂單

entity type=order

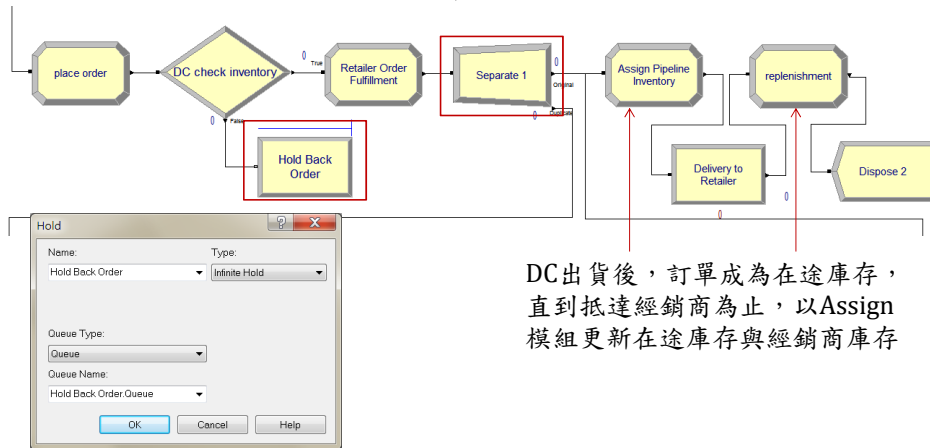
更新訂購次數 $\text{order_count}(\text{index})$ 、
 尚未到的已訂購量 $\text{on_order}(\text{index})$ 、
 紀錄訂單產生的時間 order time



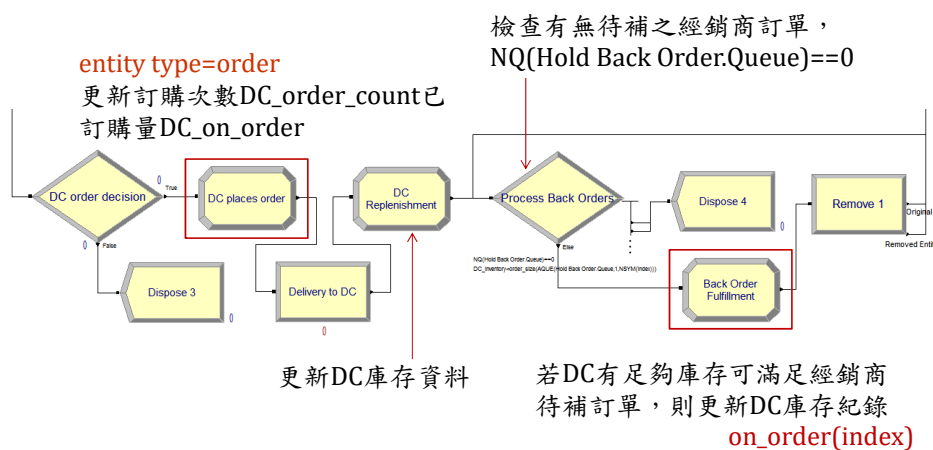
DC檢查是否有足夠庫存可出貨給經銷商，
 $\text{DC_inventory}(\text{index}) \geq \text{order_size}(\text{index})$

DC處理訂單的流程

Separate模組將流程分為兩邊，一邊處理訂單，一邊檢查是否要補貨



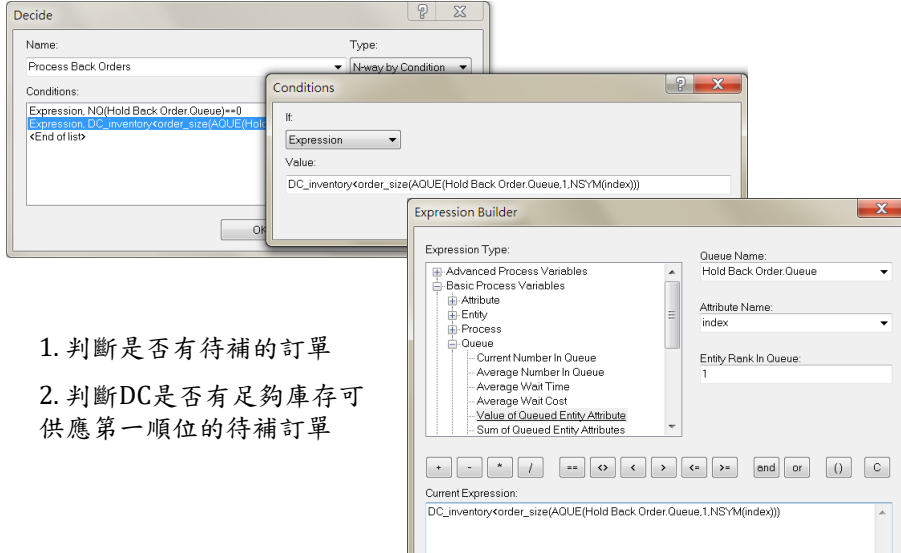
DC向工廠補貨的流程



DC處理經銷商之欠補訂單

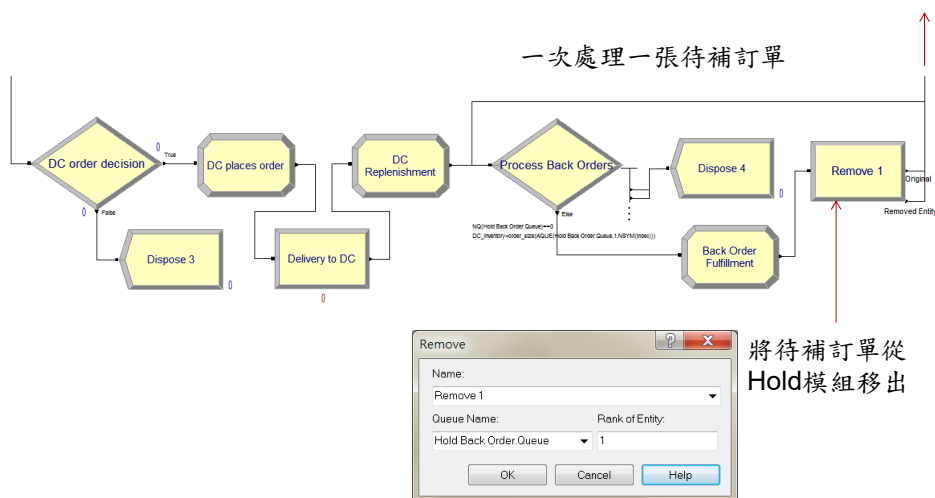
1. 判斷是否有待補的訂單

2. 判斷DC是否有足夠庫存可供應第一順位的待補訂單



The image shows three overlapping windows from a simulation software interface. The top window is titled 'Decide' and shows a process 'Process Back Orders' with a condition 'Expression: NQ(Hold Back Order Queue) >= 0'. The middle window is titled 'Conditions' and shows a condition 'It: Expression' with a value 'DC_inventory_order_size(AQOE(Hold Back Order Queue, 1, NSYM(index)))'. The bottom window is titled 'Expression Builder' and shows the same expression being built, with the 'Queue Name' set to 'Hold Back Order Queue' and 'Attribute Name' set to 'index'.

Remove模組讓待補訂單繼續出貨流程



Statistics模組設定系統績效

	Expression 定義計算方式
Total Inventory 供應鏈總庫存	Delivery to DC.WIP*DC_order_size + pipeline + DC_Inventory+Inventory(1)+Inventory(2)+Inventory(3)

	Expression 定義計算方式
avg fill rate 平均供貨水準	$\frac{(NC(S \text{ Counter } 1)+NC(S \text{ Counter } 2)+NC(S \text{ Counter } 3))}{(NC(S \text{ Counter } 1)+NC(S \text{ Counter } 2)+NC(S \text{ Counter } 3)+NC(L \text{ Counter } 1)+NC(L \text{ Counter } 2)+NC(L \text{ Counter } 3))}$
avg unit cost 平均單日成本	$((NC(L \text{ Counter } 1)+NC(L \text{ Counter } 2)+NC(L \text{ Counter } 3))*3 + (order_count(1)+order_count(2)+order_count(3))*64 + DC_order_count*200)/2970 + DAVG(Total Inventory)*0.3$

三層供應鏈模擬結果

使用Batch Means的實驗方式：設定warm up period=30 days，replication length=3000 days。

	Average	Half-width
avg fill rate	.9979	
avg unit cost	145.14	
Total Inventory	306.61	2.9893
dc inv	119.28	2.7886
pipeline inv	72.110	1.1674
inv1	40.044	0.3621
inv2	36.490	0.3164
inv3	38.681	0.4004

Example 7-5

調整庫存控管

- 經銷商的訂貨點都從20降為12
- DC的訂貨量從200降為160
- DC的訂貨點從60降為50

	Average	Half-width
avg fill rate	.97754	
avg unit cost	134.38	
Total Inventory	250.23	6.6199
dc inv	89.458	6.0513
pipeline inv	70.350	1.1674
inv1	31.912	0.4939
inv2	28.145	0.3823
inv3	30.365	0.3778

供應鏈流程設計的漏洞

- 將DC_order_size降為100。
- 三層供應鏈的流程有個漏洞需要補正，當DC沒有足夠庫存可滿足經銷商訂單時，流程應該一分為二，一方面讓訂單停留在Hold模組，另一方面讓DC檢查是否需要向工廠下單補貨。
- 如何修正此缺失？並在相同實驗條件下重新模擬。

