Risk-based Planning and Scheduling with Simio

Introduction

Risk-based Planning and Scheduling (RPS) is the dual use of a simulation model to generate both a detailed resource-constrained deterministic schedule as well as a probability-based risk analysis of that schedule to account for variation in the system. RPS is used to generate schedules that minimize risks and reduce costs in the presence of uncertainty.

Traditional Advanced Planning and Scheduling (APS) generates schedules by assuming there is no variation or uncertainty in the system. A deterministic APS schedule quickly becomes obsolete as machines break, processes vary in time, material arrives late, etc. APS schedules which appear initially feasible become infeasible over time as variation degrades performance. By ignoring variation APS schedules are optimistic by nature - they promise more than can be delivered. The user of traditional APS has no way to assess or mitigate the underlying risk inherent in the schedule. RPS augments the deterministic schedule with risk measures that allow the decision maker to properly account for the underlying variation and uncertainty in the system.

RPS uses a purpose-built simulation model of the system to fully capture both the detailed constraints and variations in the system. RPS then uses this model in two ways. The first is to generate a detailed schedule/plan. In this case the model is executed in a purely deterministic mode; machines do not break, process times are always constant, materials arrive on time, etc. This is the optimistic view assumed by all APS systems and produces a deterministic plan/schedule. Once the schedule has been generated RPS then replicates this same simulation model with variation added back into the system and performs a probabilistic analysis to estimate the underlying risks associated with the schedule. The risk measures generated by RPS include the probability of meeting user-defined targets, as well as expected, pessimistic, and optimistic schedule performance.

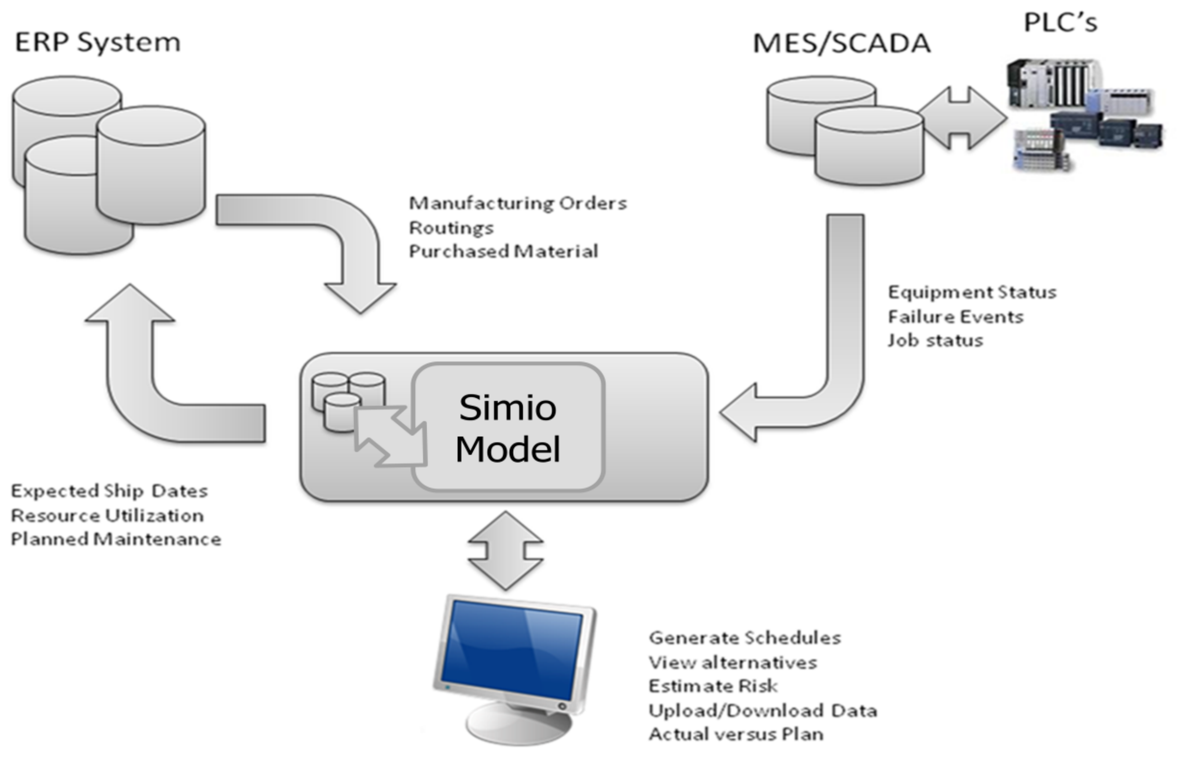
By providing up-front visibility into the inherent risk associated with a specific plan/schedule, RPS provides the necessary information to take early action in the operational plan to mitigate risks and reduce costs. RPS provides a realistic view of expected schedule performance. Specific alternatives such as overtime or expediting external material/components from suppliers can be compared in terms of their impact on both risks of meeting schedule targets, and costs of mitigating those risks, thereby providing a customer-satisfying operational strategy at a minimum cost.

RPS uses a purpose-built Simio model of the system to simulate the flow of a specific set of jobs. The orders to be scheduled are downloaded to the RPS model from the ERP/MRP system. The same Simio simulation model that is used by RPS can also be used to analyze and improve the design of the underlying production system by randomly generating orders over a long planning horizon. For example the model could be used to evaluate the long term impact on performance of capital equipment purchases, changes in process flows, or new product introductions. Hence the same simulation model can be used to both improve the facility design as well as provide the detailed model logic for planning/scheduling of day-to-day operations.

The Simio model that drives the RPS solution can be built using any of the Simio simulation products. However the Simio Enterprise Edition contains a number of unique features that are useful in building planning and scheduling solutions. The Simio Enterprise Edition encompasses all the functionality of the Simio Design/Team Edition plus additional scheduling functionality; it is the ideal product for the internal/external consultant that is building a scheduling solution. The Simio Scheduling Edition is a runtime deployment platform for planning and scheduling. This product has no model building functionality of its own, but uses a model built by the Simio Enterprise Edition. This is a streamlined product that is specifically designed for day to day use of the Simio model for planning and scheduling.

ERP Integration

The Simio RPS model is typically integrated with the existing ERP and MES/SCADA system to provide the data on the current status of the system and the actual jobs to be processed through the system. This data is provided to Simio in the form of relational data sets that are imported and held in memory by Simio for fast execution. These data sets typically contain data such as a list of jobs to be processed, a bill of material for each job, job routings including setup and processing times, purchased material, etc, along with the status of all jobs in the system at the start of the planning period. Simio does not have a fixed data schema – the date sets used by Simio can be configured as needed to match the form of the external data. The flow of jobs as defined by the data sets is then simulated using the Simio model to generate a detailed schedule. A schematic of a typical implementation is shown below:



The logic for the schedule generation is provided by a purpose-built Simio model that captures the critical constraints of the facility. This is a key advantage over traditional APS systems that rely on either a predefined or mathematical model of the system that cannot represent the critical constraints in many systems. The Simio model can be as simple or detailed as required, and may include complex processing and material handling constraints such as ovens, forklift trucks, conveyors, moving operators, etc., as well as complex work crews.

Once the data is imported from the ERP system, a schedule is generated by simulating the flow of jobs using the Simio model with no variation. The simulation is then replicated multiple times with variation incorporated to generate probabilistic risk measures for the associated schedule. The risk measures include the probability of meeting user-defined targets, as well as expected, pessimistic, and optimistic schedule performance. The resulting schedule and associated risk analysis may then be exported back to the ERP system.

Sample Application

We will now build a small example Simio model to illustrate the use of Simio for planning and scheduling. In this simple example we have a list of jobs that are processed across four workstations named Cut, Shape, Weld, and Finish. A single operator is shared between the Shape and Weld stations, but must be present during setup and processing. Each job produces a product of type A, B, or C. The routing sequences for each product type, along with the required bill of material (BOM) and setup and processing times at each step are summarized in the following table.

Part Type	Sequence	Required BOM	Setup Time (Days)	Process Time (Days)
A	Cut	MaterialCutA	.11	Random.Triangular(.1,.2,.3)
A	Weld		Sequence Dependent	Random.Triangular(.2,.3,.35)
A	Shape		.09	Random.Triangular(.13,.22,.4)
A	Finish		.1	Random.Triangular(.1,.2,.3)
A	Ship			
B	Cut	MaterialCutB		Random.Triangular(.15,.18,.37)
B	Weld		Sequence Dependent	Random.Triangular(.1,.15,.2)
B	Finish			Random.Triangular(.1,.2,.3)
B	Ship			
C	Cut	MaterialCutC		Random.Triangular(.16,.28,.44)
C	Finish			Random.Triangular(.1,.2,.3)
C	Ship			

The setup time at the welding station is sequence dependent based on the part size. Part type A is large, and types B and C are both small. The sequence dependent setup times in minutes are defined by the following changeover matrix.

From/To	Small	Large
Small	0	10

Large	30	0

The first step in each job routing is the Cut workstation which requires a bill of material comprised of MaterialX and MaterialY as defined in the following table.

Bill of Material Name	Component Material Name	Required Quantity
MaterialCutA	MaterialX	2
MaterialCutA	MaterialY	1
MaterialCutB	MaterialX	2
MaterialCutB	MaterialY	2
MaterialCutC	MaterialY	3

Each material has an initial stock of 6, and has purchased material expected to arrive at the following times. The actual arrival occurs on its scheduled date 25% of the time, one day early 25% of the time, and one day late 50% of the time. The material can also be expedited at an additional cost; this guarantees it will arrive one day early.

Material	Quantity	Expected Arrival Date
MaterialX	6	10/4/2011 12:00:00 AM
MaterialX	6	10/6/2011 12:00:00 AM
MaterialY	6	10/4/2011 12:00:00 AM
MaterialY	6	10/7/2011 12:00:00 AM

The jobs to be processed during this planning period include nine new orders, as well as seven orders that are in process in the system. The following table summarizes these sixteen orders.

Order ID	Part Type	Release Date	Due Date	Order Status
Order01	A	10/3/2011 12:00:00 AM	10/12/2011 11:00:00 PM	New
Order02	A	10/3/2011 12:00:00 AM	10/14/2011 9:00:00 PM	New
Order03	B	10/4/2011 12:00:00 AM	10/13/2011 4:00:00 PM	New
Order04	B	10/4/2011 12:00:00 AM	10/14/2011 2:00:00 PM	New
Order05	A	10/5/2011 12:00:00 AM	10/19/2011 9:00:00 PM	New
Order06	C	10/6/2011 12:00:00 AM	10/12/2011 12:00:00 AM	New
Order07	A	10/7/2011 12:00:00 AM	10/19/2011 4:00:00 PM	New
Order08	C	10/8/2011 12:00:00 AM	10/13/2011 10:00:00 PM	New
Order09	B	10/9/2011 12:00:00 AM	10/19/2011 4:00:00 PM	New
OrderWIP1	A	10/3/2011 12:00:00 AM	10/10/2011 4:00:00 PM	WIP
OrderWIP2	A	10/3/2011 12:00:00 AM	10/5/2011 4:00:00 PM	WIP
OrderWIP3	B	10/3/2011 12:00:00 AM	10/7/2011 4:00:00 PM	WIP
OrderWIP4	B	10/3/2011 12:00:00 AM	10/5/2011 4:00:00 PM	WIP
OrderWIP5	C	10/3/2011 12:00:00 AM	10/6/2011 4:00:00 PM	WIP
OrderWIP6	C	10/3/2011 12:00:00 AM	10/3/2011 4:00:00 PM	WIP
OrderWIP7	C	10/3/2011 12:00:00 AM	10/3/2011 4:00:00 PM	WIP

The work in process (WIP) is comprised of the last seven orders, and the status of these orders is summarized in the following table (typically imported from the SCADA system).

Order ID	Current Step	Fraction Remaining
OrderWIP1	1	0.5
OrderWIP2	2	1
OrderWIP3	1	0.7
OrderWIP4	2	1
OrderWIP5	1	1
OrderWIP6	3	0
OrderWIP7	2	1

The current step value specifies the step in the sequence where the job is currently being processes, and the fraction remaining specifies the fraction of the total processing time that remains at this station. For example OrderWIP1 is at its first workstation (Cut) and has half of its processing time remaining.

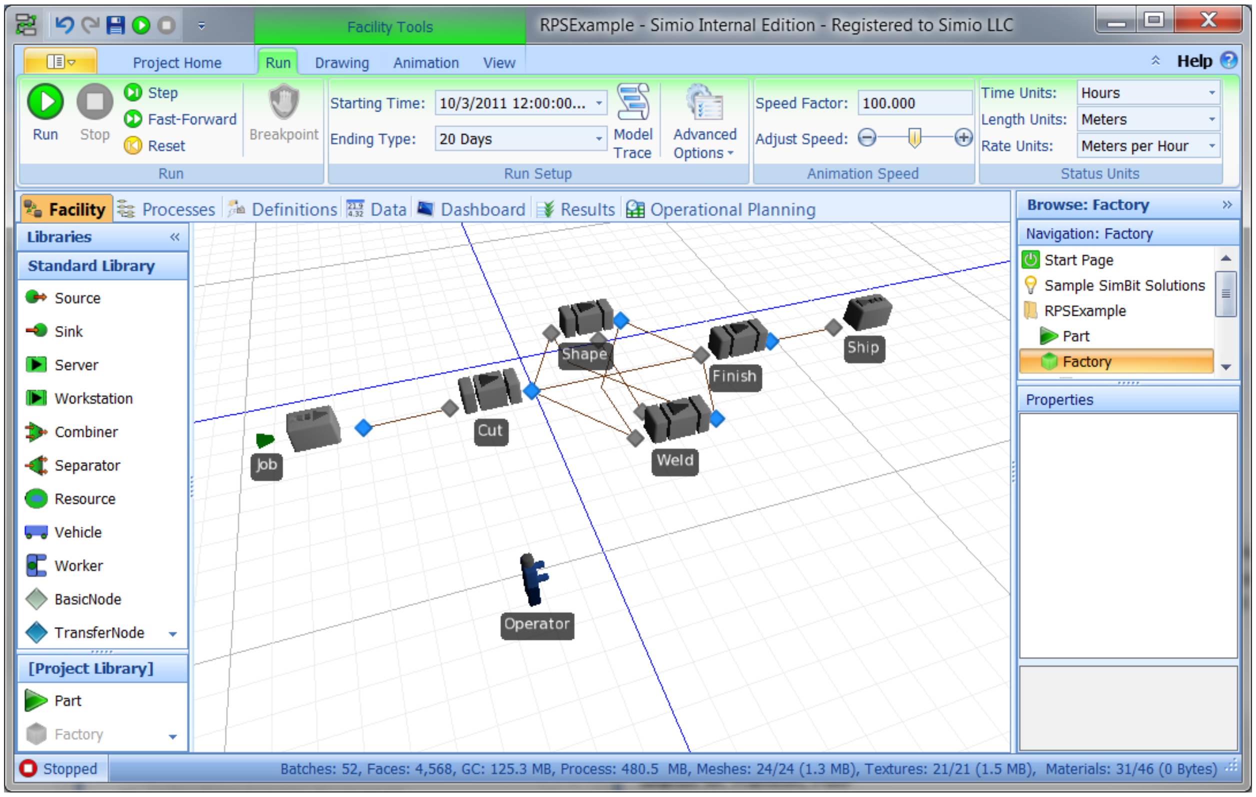
The Simio Model

This section assumes a basic understanding of Simio, including the use of relational data tables. The logic for the RPS solution is provided by a custom Simio model of the system. In this case we have a very simple system comprised of four workstations (*Cut*, *Shape*, *Weld*, and *Finish*) and an *Operator* that is shared between the *Shape* and *Weld* workstations. All workstations follow a standard work schedule Monday-Friday from 8-12 and 1-5.

We will model this system using the Workstation object in the Standard Library. The Workstation is useful for applications with complex setup and a bill of materials, both present in our simple example.

We will begin by placing our four Workstations named *Cut*, *Shape*, *Weld*, and *Finish* in our model named *Factory*. We will also place a Source named *PartArrivals*, and a Sink named *Ship*. We will draw Paths from *PartArrivals* to *Cut*, from *Cut* to *Shape* and *Weld*, from *Shape* to *Weld* and *Finish*, from *Weld* to *Shape* and *Finish*, and from *Finish* to *Ship*. We set the Entity Destination on the output nodes at the *PartArrivals*, *Cut*, *Shape*, *Weld*, and *Finish* Workstations to *By Sequence*; this is necessary to ensure that the parts flow by their sequence. We will model the operator by placing a Worker named *Operator*. We will place a Basic Node by the *Shape* and *Weld* Workstations (naming them *ShapeOperator* and *WeldOperator*), and connect them with a bidirectional Path for use by the *Operator*. We set the Operator's Initial Node to *ShapeOperator*, and specify the Idle Action and Off Shift Action to *Remain In Place*.

We will represent all three part types with a single entity. We will rename the ModelEntity to Part and place the default entity next to the PartArrivals source and rename it *Job*. We will add two additional symbols to the *Job* symbol list, and color the second one red and the third one blue. We will use these three symbols for the *Job* entity to represent the three different part types in our model. We will also add a real state named *RemainingWork* to the Part entity (with an initial value 1.0). This state will be used for initializing the model with partially completed jobs. The initial layout for are model is shown below:


RPSExample - Simio Internal Edition - Registered to Simio LLC

Starting Time: 10/3/2011 12:00:00...
Ending Type: 20 Days
Speed Factor: 100.000
Time Units: Hours
Length Units: Meters
Rate Units: Meters per Hour

Stopped   Batches: 52, Faces: 4,568, GC: 125.3 MB, Process: 480.5 MB, Meshes: 24/24 (1.3 MB), Textures: 21/21 (1.5 MB), Materials: 31/46 (0 Bytes)

Before completing our model logic we will define our data tables for interfacing the model to the ERP system. We will create a total of seven relational data tables as summarized below:

Materials: Materials that are used (consumed or produced) at this facility.

Bill Of Materials: Component materials and required quantities for each parent bill of material.

Purchased Materials: Material purchases that are scheduled to arrive during the planning period.

Manufactured Items: Characteristics of all parts that can be manufactured at this factory.

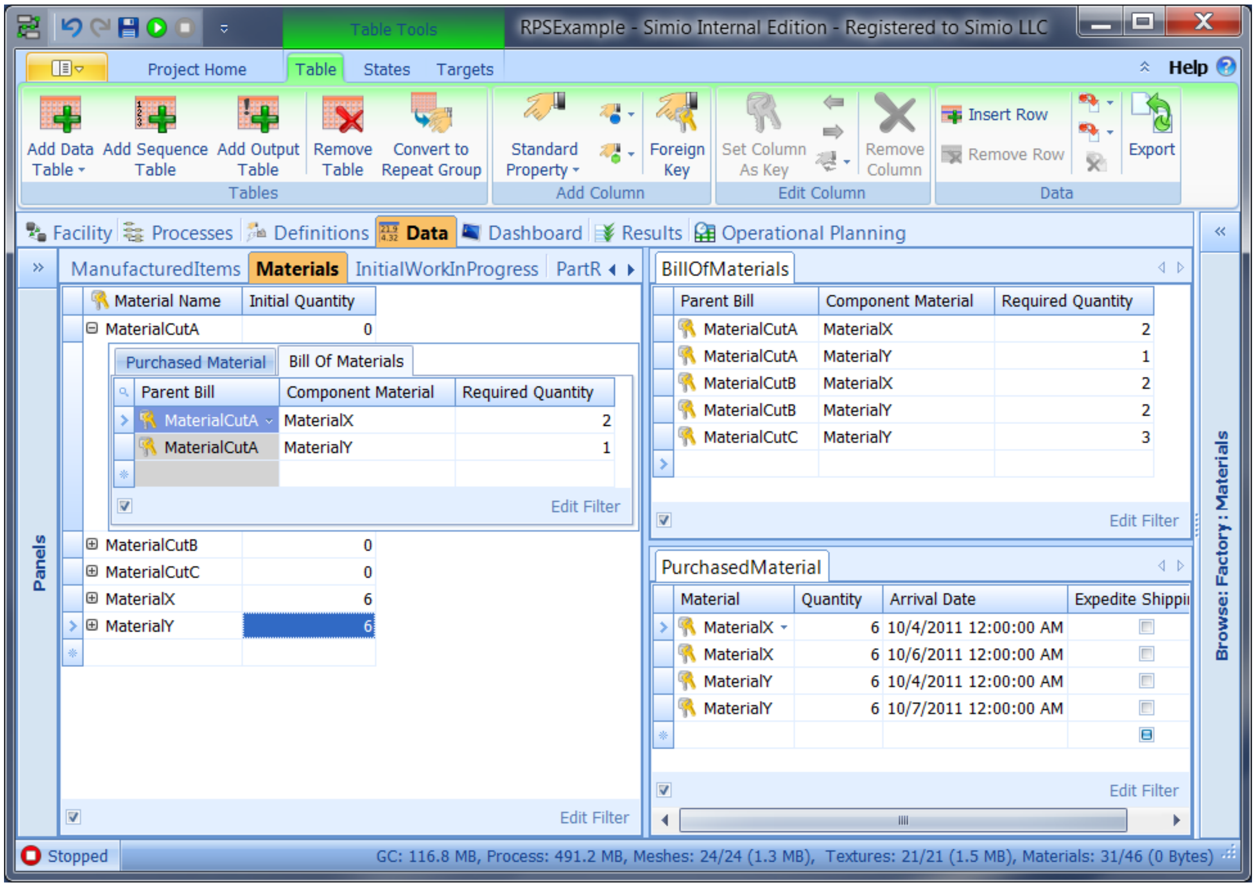
Part Routings: Routing sequence along with setup and processing times for each part type.

Manufacturing Orders: Jobs currently in or scheduled for release to the factory during this planning period.

InitialWorkInProgress: Work in process (WIP), along with the current step and fraction of work remaining at that step.

We will begin by building the first three interrelated material tables. However before adding these tables we must first manually create the corresponding materials as elements in our model. We do this manual step to add material elements named MaterialCutA, MaterialCutB, MaterialCutC, MaterialX, and MaterialY, and add rows to the Bill Of Materials for the first three materials corresponding to the specified bill for those materials. (NOTE: This step will be unnecessary in the near future).

Once we have manually defined our materials, we next create our material related tables. We add a table named Materials, and add a column of type material element reference named *MaterialName* and set it as a Key column, and add a second column named *InitialQuantity* of type real. We then populate the table with the same information used to manually create the material elements. Next we create a new table named *BillOfMaterials* and add a column named *ParentBill* that is a foreign key reference to *Materials.MaterialName*, a column named *ComponentMaterial* of type material element reference, and a column named *RequiredQuantity* of type real. We then populate this table to define the components of the top level material bills. Finally we add a table named PurchasedMaterial and add a column to this table named Material that is a foreign key reference to Materials.MaterialName. We also add columns for Quantity (real), ArrivalDate (date time), and ExpediteShipping (Boolean). We also set the Editable flag to true for this column to allow values in this column to be edited by the end-user of the model. We then populate this table using the data provided. The following shows the three completed material-related tables. Note that each material in the Materials table automatically has tabs for Purchased Material and Bill Of Materials based on the relations established by the foreign key references.



Materials

Material Name	Initial Quantity
MaterialCutA	0
MaterialCutB	0
MaterialCutC	0
MaterialX	6
MaterialY	6

MaterialCutA — Purchased Material | Bill Of Materials

Parent Bill	Component Material	Required Quantity
MaterialCutA	MaterialX	2
MaterialCutA	MaterialY	1

BillOfMaterials

Parent Bill	Component Material	Required Quantity
MaterialCutA	MaterialX	2
MaterialCutA	MaterialY	1
MaterialCutB	MaterialX	2
MaterialCutB	MaterialY	2
MaterialCutC	MaterialY	3

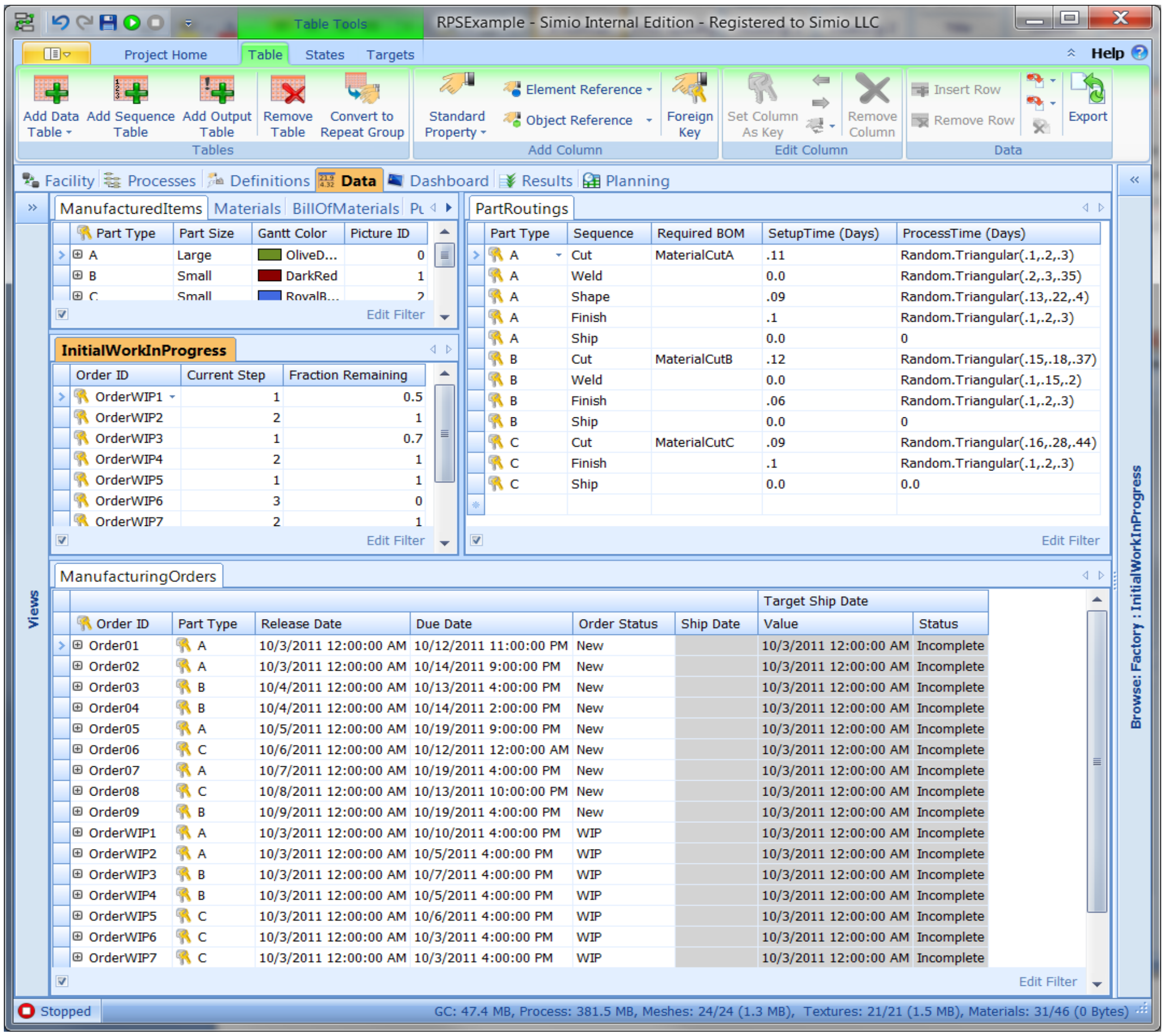
PurchasedMaterial

Material	Quantity	Arrival Date	Expedite Shippi
MaterialX	6	10/4/2011 12:00:00 AM	☐
MaterialX	6	10/6/2011 12:00:00 AM	☐
MaterialY	6	10/4/2011 12:00:00 AM	☐
MaterialY	6	10/7/2011 12:00:00 AM	☐

Although we have entered this data manually, in a typical application the data in these tables would be imported at the beginning of each planning period from the ERP system.

Next we will create the last four interrelated tables for defining the jobs and parts. These tables will make use of three custom Simio string lists. We will define a custom Simio string list named *OrderStatus* with values *New* and *WIP*, and a string list named *PartSize* with values *Small* and *Large*.

A screen shot of these four tables is shown below. We begin by creating the *ManufacturedItems* table (upper left hand corner) that defines the characteristics of the three part types. This table has a column named *PartType* of type string. This column is set as a Key column. The next column is named *PartSize* and is type List and is bound to the list named *PartSize*. The third column added to this table is named *GanttColor* and is type Color. The last column is *PictureID* and is the index of the picture to use when animating this part. The values for the three part types are then added to the table.



ManufacturedItems

Part Type	Part Size	Gantt Color	Picture ID
A	Large	🟩 OliveD...	0
B	Small	🟥 DarkRed	1
C	Small	🟦 RoyalB...	2

PartRoutings

Part Type	Sequence	Required BOM	SetupTime (Days)	ProcessTime (Days)
A	Cut	MaterialCutA	.11	Random.Triangular(.1,.2,.3)
A	Weld		0.0	Random.Triangular(.2,.3,.35)
A	Shape		.09	Random.Triangular(.13,.22,.4)
A	Finish		.1	Random.Triangular(.1,.2,.3)
A	Ship		0.0	0
B	Cut	MaterialCutB	.12	Random.Triangular(.15,.18,.37)
B	Weld		0.0	Random.Triangular(.1,.15,.2)
B	Finish		.06	Random.Triangular(.1,.2,.3)
B	Ship		0.0	0
C	Cut	MaterialCutC	.09	Random.Triangular(.16,.28,.44)
C	Finish		.1	Random.Triangular(.1,.2,.3)
C	Ship		0.0	0.0

InitialWorkInProgress

Order ID	Current Step	Fraction Remaining
OrderWIP1	1	0.5
OrderWIP2	2	1
OrderWIP3	1	0.7
OrderWIP4	2	1
OrderWIP5	1	1
OrderWIP6	3	0
OrderWIP7	2	1

ManufacturingOrders

						Target Ship Date	
Order ID	Part Type	Release Date	Due Date	Order Status	Ship Date	Value	Status
Order01	A	10/3/2011 12:00:00 AM	10/12/2011 11:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
Order02	A	10/3/2011 12:00:00 AM	10/14/2011 9:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
Order03	B	10/4/2011 12:00:00 AM	10/13/2011 4:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
Order04	B	10/4/2011 12:00:00 AM	10/14/2011 2:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
Order05	A	10/5/2011 12:00:00 AM	10/19/2011 9:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
Order06	C	10/6/2011 12:00:00 AM	10/12/2011 12:00:00 AM	New		10/3/2011 12:00:00 AM	Incomplete
Order07	A	10/7/2011 12:00:00 AM	10/19/2011 4:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
Order08	C	10/8/2011 12:00:00 AM	10/13/2011 10:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
Order09	B	10/9/2011 12:00:00 AM	10/19/2011 4:00:00 PM	New		10/3/2011 12:00:00 AM	Incomplete
OrderWIP1	A	10/3/2011 12:00:00 AM	10/10/2011 4:00:00 PM	WIP		10/3/2011 12:00:00 AM	Incomplete
OrderWIP2	A	10/3/2011 12:00:00 AM	10/5/2011 4:00:00 PM	WIP		10/3/2011 12:00:00 AM	Incomplete
OrderWIP3	B	10/3/2011 12:00:00 AM	10/7/2011 4:00:00 PM	WIP		10/3/2011 12:00:00 AM	Incomplete
OrderWIP4	B	10/3/2011 12:00:00 AM	10/5/2011 4:00:00 PM	WIP		10/3/2011 12:00:00 AM	Incomplete
OrderWIP5	C	10/3/2011 12:00:00 AM	10/6/2011 4:00:00 PM	WIP		10/3/2011 12:00:00 AM	Incomplete
OrderWIP6	C	10/3/2011 12:00:00 AM	10/3/2011 4:00:00 PM	WIP		10/3/2011 12:00:00 AM	Incomplete
OrderWIP7	C	10/3/2011 12:00:00 AM	10/3/2011 4:00:00 PM	WIP		10/3/2011 12:00:00 AM	Incomplete

GC: 47.4 MB, Process: 381.5 MB, Meshes: 24/24 (1.3 MB), Textures: 21/21 (1.5 MB), Materials: 31/46 (0 Bytes)

Next we define the PartRoutings sequence table (upper right). This table has a column named *PartType* that is a foreign key reference to *ManufacturingItems.PartType*. The second column is the Sequence column and its property "Accepts Any Node" is set to false to allow the sequence to be specified by object name rather than the fully qualified name (NodeName@ObjectName). Next we add a column named *RequiredBOM* of type material reference that specifies the bill of material that is required by this processing step. Finally we add expression properties to define the *SetupTime* and *ProcessTime* specified as a time unit in Days. We then populate this table with the specified data.

Next we create the *ManufacturingOrders* table. The first column of this table is a Key column named *OrderID* of type string. The second column is a foreign key reference to *ManufacturedItems.PartType*, and specifies the type of part required for this order. The next two columns are type date time and define the *ReleaseDate* and *DueDate* for the order. The next column is named *OrderStatus* and is a List column bound to the list named *OrderStatus*.

We next add a state variable named *ShipDate* of type date time to the table. You add this by selecting the States ribbon from the Data window in Simio Enterprise Edition. We will write the ship date to this column each time an order completes processing through the model.

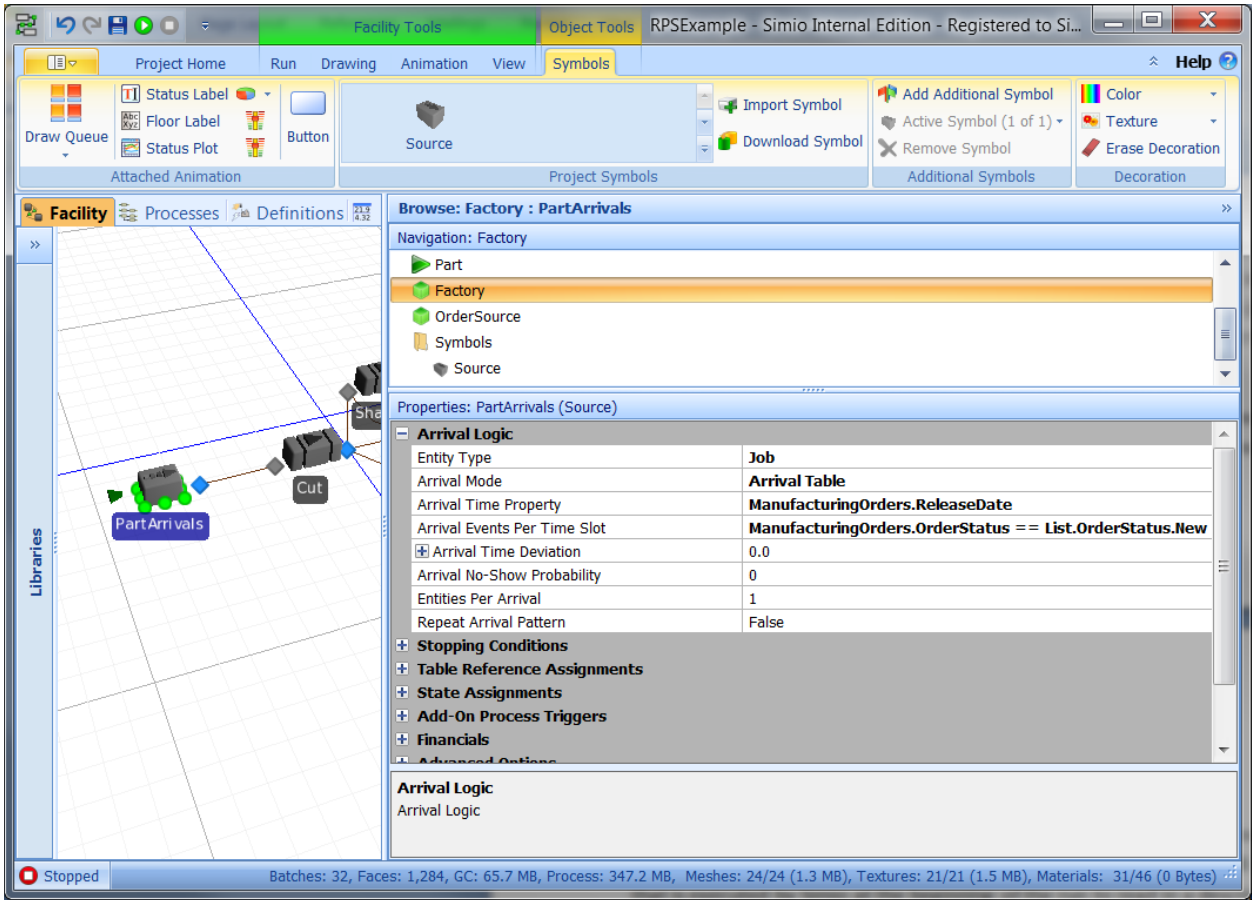
We now add a target to our table. A target is defined by a range of desired values, along with labels for displaying the status of the target based on its actual value. A target has an upper bound, lower bound, and associated strings for displaying the status of no value recorded, within bounds, below lower bound, above the upper bound. A table can have any number of targets, and targets can be different data types such as a real or date time. Targets of type date time are referred to as milestone targets. A common milestone target is a ship date for the product. As we will see in a moment milestone targets may be displayed directly on our Gantt chart.

In this example we add a milestone target named *TargetShipDate* from the Target ribbon within the Data window. We then specify the target expression as: *ManufacturingOrders.ShipDate*, set the data format to DateTime (making it a milestone), and specify the upper bound on the target as *ManufacturingOrders.DueDate*. We then specify the within bounds string as *OnTime*, the above upper bound string as *Tardy*, and the no value string as *Incomplete*. We populate all the columns except for the ShipDate and TargetShipDate columns with the specified data. When we run the simulation the model will write the ship date to the ShipDate column and the target value will be updated and status will be automatically updated from *Incomplete* to either *OnTime* or *Tardy* based on the ship date relative to the upper bound (i.e. the due date).

Our last table is the *InitialWorkInProgress* table (center left). This table has a column named *OrderID* that is a foreign reference to *ManufacturingOrders.OrderID*. The next column is named *CurrentStep* and is type integer. This column specifies the current step number for this job. The last column is named *FractionRemaining* and is the fraction of processing work remaining at the current workstation. We then populate this table with the specified data.

Now that we have our data tables fully defined we can complete our modeling logic and set the necessary references from our modeling objects to these table columns.

We will begin by setting the properties for the PartArrivals Souce. We set the Arrival Mode to Arrival Table, and specify the Arrival Time Property as ManufacturingOrders.ReleaseDate. We set the Arrival Events Per Time Slot to the Boolean expression ManufacturingOrders.OrderStatus == List.OrderStatus.New, which returns 1 if true, and 0 if false. This causes arrivals to only occur for New orders, and the existing WIP orders. The property settings for the PartsArrivals Source is shown below:



Properties: PartArrivals (Source)

Arrival Logic	
Entity Type	Job
Arrival Mode	Arrival Table
Arrival Time Property	ManufacturingOrders.ReleaseDate
Arrival Events Per Time Slot	ManufacturingOrders.OrderStatus == List.OrderStatus.New
Arrival Time Deviation	0.0
Arrival No-Show Probability	0
Entities Per Arrival	1
Repeat Arrival Pattern	False

Stopping Conditions
Table Reference Assignments
State Assignments
Add-On Process Triggers
Financials

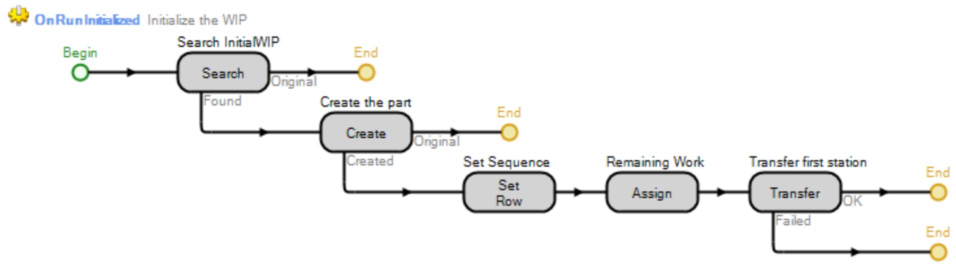
Batches: 32, Faces: 1,284, GC: 65.7 MB, Process: 347.2 MB, Meshes: 24/24 (1.3 MB), Textures: 21/21 (1.5 MB), Materials: 31/46 (0 Bytes)

We will also set the Display Name for the Job entity to ManufacturingOrders.OrderID. This will display the Order ID for the entity in the Gantt chart and Logs. We also set the Current Symbol Index to ManufacturedItems.PartType. This will cause the correct symbol to be displayed in the facility animation for each of the three party types.

Next we add the necessary logic to our model to initialize our system with the WIP defined in our Manufacturing Orders and Initial Work in Progress tables. We will add a OnRunInitialized process that is executed by Simio at the beginning of the run to read in a description of the WIP and send the jobs to their appropriate locations. We begin by searching all rows of the *InitialWorkInProgress* table, and for each row we create a new Job entity, set the row in the *PartRoutings* table to the current step number specified by *InitialWorkInProgress.CurrentStep*, assign the *Part.RemainingWork* to

InitialWorkInProgress.FractionRemaining, and then transfer from the Job from free space to the node specified by *PartRoutings.Sequence*.

The process for this is shown below:

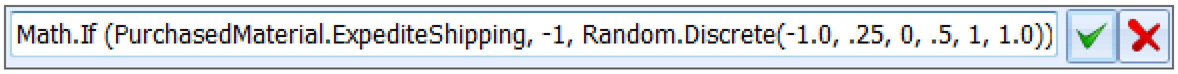


The first operation for all three Parts begins at the *Cut* station. The *Capacity Type* is specified as *WorkSchedule*, and the *Work Schedule* is specified as *StandardWeek*. The *Processing Time* is specified as *PartRoutings.ProcessTime * Part.RemainingWork.* Note that RemainingWork is 1 for all new jobs, but may be less than one for WIP jobs. Finally a Bill Of Materials consumption is specified as *PartRoutings.RequiredBOM*. This causes the specified BOM to be consumed before the cut operation can be begin.

The *Weld*, *Shape*, and *Finish* Workstations are specified in a similar way, except they do not require material. In addition both the *Shape* and *Weld* Workstation require the *Operator* as a secondary resource, with a visit request to the appropriate Node (*ShapeOperator* or *WeldOperator*). Finally the *Weld* Workstation specifies the *Setup Time Type* as *Sequence Dependent*, with the *Operation Attribute* specified as *ManufacturedItems.PartSize*. We then create a Changeover matrix named *WeldSetup* based on the List name *PartSize*, and specify the changeover times in this matrix. We then specify the *Changeover Matrix* on the *Weld* Workstation as *WeldSetup*.

Next we assign *ManufacturingOrders.ShipDate* to *TimeNow* as state assignment at the Ship Sink. This causes the target value to be automatically evaluated.

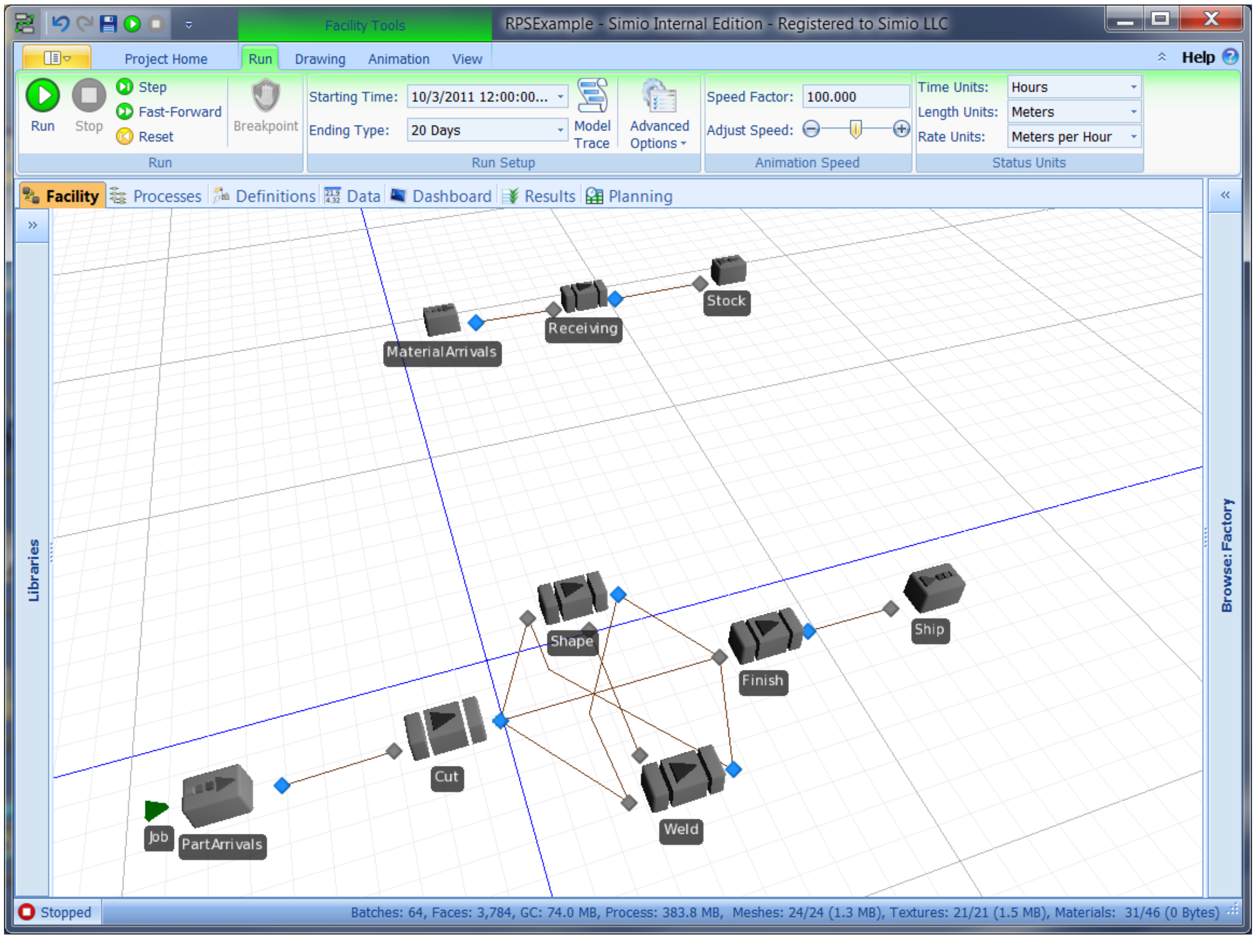
Our last remaining task is to process the purchased material arrivals. We do this by placing a Source named *MaterialArrivals*, a Workstation named *Receiving*, and a Sink named *Stock,* with Connectors in between. The *MaterialArrivals* generates arrivals from the *PurchasedMaterial.ArrivalDate* column, and deviates these arrivals by up to one day using the following expression with Day time units:



```
Math.If (PurchasedMaterial.ExpediteShipping, -1, Random.Discrete(-1.0, .25, 0, .5, 1, 1.0))
```

Note that is *ExpediteShipping* is true the material arrives 1 day early, otherwise it arrives 1 day early, on time, or 1 day late with probability of .25, .25, and .5 (recall that the discrete distribution specifies cumulative probabilities, hence .25, .5, 1.0).

The following shows our completed model. We can run this same model in 3D animated mode or in planning mode. Next we will look at using this model within Simio Scheduling Edition for operation planning and scheduling.


RPSExample - Simio Internal Edition - Registered to Simio LLC

Starting Time: 10/3/2011 12:00:00...
Ending Type: 20 Days
Speed Factor: 100.000
Time Units: Hours
Length Units: Meters
Rate Units: Meters per Hour

Stopped

Batches: 64, Faces: 3,784, GC: 74.0 MB, Process: 383.8 MB, Meshes: 24/24 (1.3 MB), Textures: 21/21 (1.5 MB), Materials: 31/46 (0 Bytes)

Simio Scheduling Edition

The Simio Scheduling Edition is used to deploy a Simio model in an RPS application. This version of Simio is specifically designed for use in day-to-day RPS applications; all of the functionality related to model building is removed to simplify the interface for the user. The Simio model is built using the Simio Enterprise Edition, and then deployed to end-users with the Simio Scheduling Edition.

The basic functionality provided by the Simio Scheduling Edition is the Planning tab in the Enterprise Edition and includes the ability to import/export ERP data, view and edit selected portions of the table data (as specified by the model builder), generate schedules and risk measures, and view those schedules in the form of or a resource plan and detailed entity workflow.

Simio Scheduling Edition provides six views for use by the scheduler. The view is selected by clicking on the icon in the Views panel on the left. The basic functionality provided by each of these six views is summarized below:

Facility: Provides a 3D animated view of facility that is useful for running and viewing the animated construction of the schedule. The user can jump to a specific resource and time in the animation by right clicking on a resource bar in the Resource Plan Gantt.

Tables: Provides a view of all table data used by the scheduling model. Selected columns (as specified by the modeler) may also be editable by the scheduler. This view also provides selective import/export functionality.

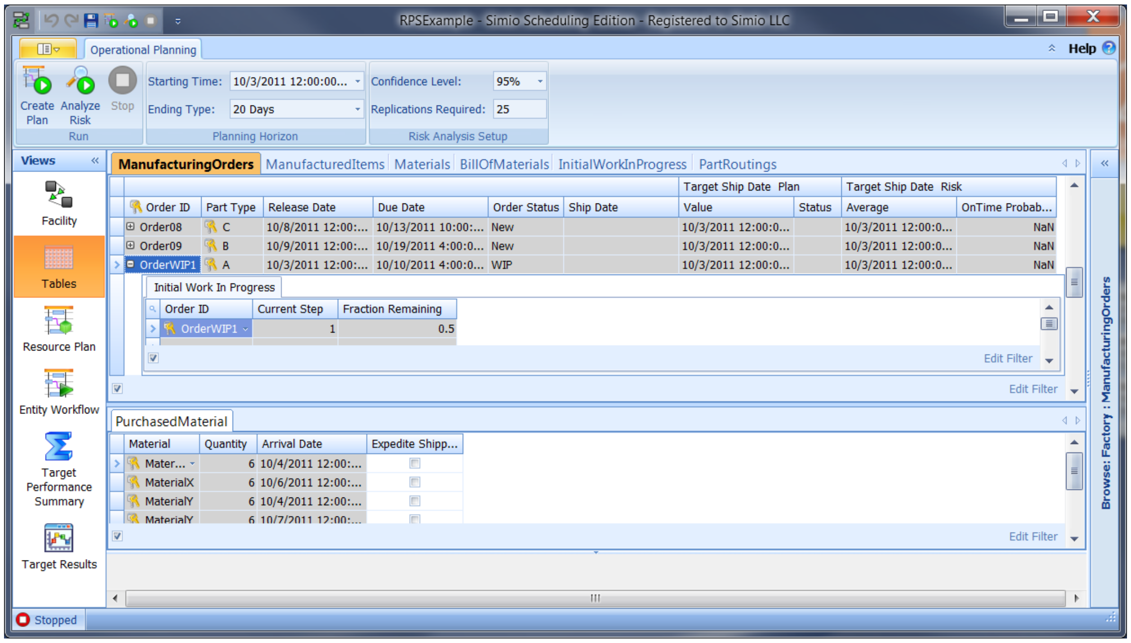
Resource Plan: Provides a Gantt chart that displays each resource along with time bars that show each entity (e.g. a job) that utilizes the resource, or waits for the resource as a constraint on its progress. This Gantt also allows direct editing of the resource capacity. The Resource Plan includes detailed logs for the resource that entity usage and state changes in the resource. These logs may be filtered and sorted by the user.

Entity Workflow: Provides a Gantt chart that displays each entity along with time bars that shows each resource that the entity utilizes or waits for as a constraint on its progress. The Gantt also displays date-time based milestones and targets for each entity, along with associated risk measures.

Target Performance: Provides a high level summary of target performance including the number and percent of entities that fall within target bounds (e.g. Tardy, TooEarly, OnTime, and Incomplete).

Target Results: Provides a detailed summary of target performance including confidence intervals and percentiles for each target.

The following shows the Tables views for our simple example. Note that in the *ManufacturingOrders* table the details for *OrderWIP1* have been expanded to show that this order has .5 remaining processing time in its first step of its routing. Note also that in the *PurchasedMaterial* table the *Expedite Shipping* column is editable by the scheduler; however the remaining columns are read-only. This is based on the *Editable* Boolean value specified for each column of the table by the model builder.



Starting Time: 10/3/2011 12:00:00...
Ending Type: 20 Days
Confidence Level: 95%
Replications Required: 25

						Target Ship Date Plan		Target Ship Date Risk	
Order ID	Part Type	Release Date	Due Date	Order Status	Ship Date	Value	Status	Average	OnTime Probab...
Order08	C	10/8/2011 12:00:...	10/13/2011 10:00:...	New		10/3/2011 12:00:0...		10/3/2011 12:00:0...	NaN
Order09	B	10/9/2011 12:00:...	10/19/2011 4:00:0...	New		10/3/2011 12:00:0...		10/3/2011 12:00:0...	NaN
OrderWIP1	A	10/3/2011 12:00:...	10/10/2011 4:00:0...	WIP		10/3/2011 12:00:0...		10/3/2011 12:00:0...	NaN

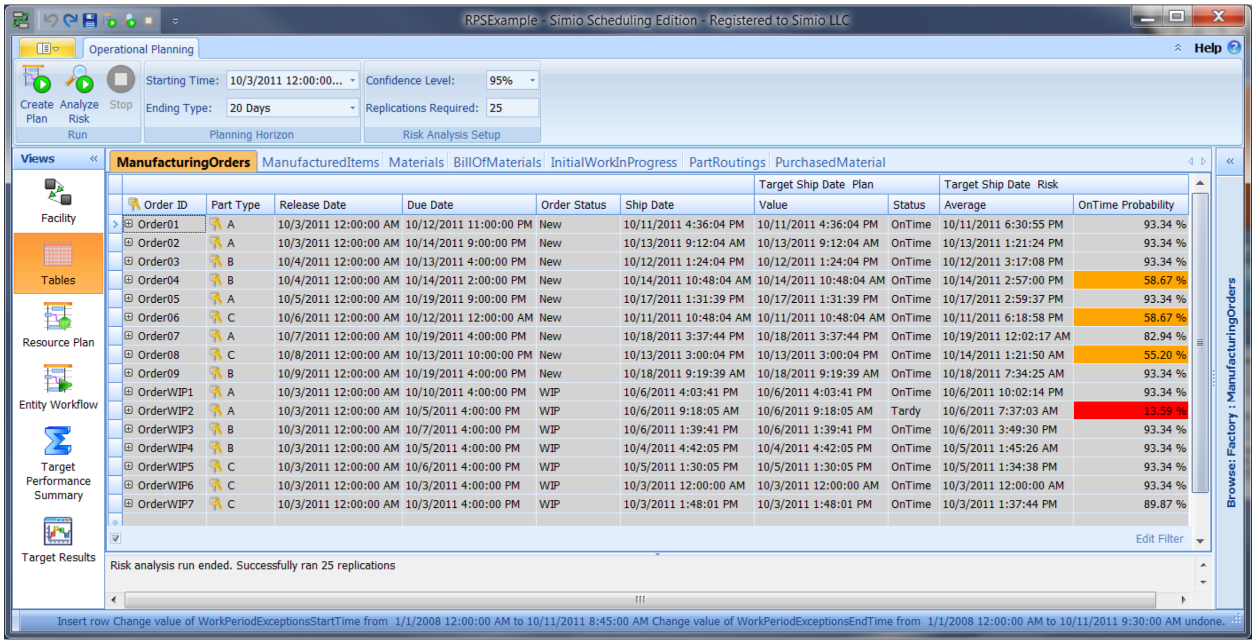
Initial Work In Progress

Order ID	Current Step	Fraction Remaining
OrderWIP1	1	0.5

PurchasedMaterial

Material	Quantity	Arrival Date	Expedite Shipp...
Mater...	6	10/4/2011 12:00:...	☐
MaterialX	6	10/6/2011 12:00:...	☐
MaterialY	6	10/4/2011 12:00:...	☐
MaterialY	6	10/7/2011 12:00:...	☐

The Operational Planning ribbon can be used to both create a deterministic plan, and also analyze the risk associate with the plan. In this example clicking on the Create Plan button causes the flow of entities defined in the ManufacturingOrders table to be simulated using the facility model. This simulation is run with all variation removed, and causes the target values in the ManufacturingOrders table to be recorded, and the Status set accordingly. Clicking on the Analyze Risk button then runs 25 replications of this same model with variation added in, and computes the average ship date and OnTime probability. The results for this example are shown below:



Starting Time: 10/3/2011 12:00:00...
Ending Type: 20 Days
Confidence Level: 95%
Replications Required: 25

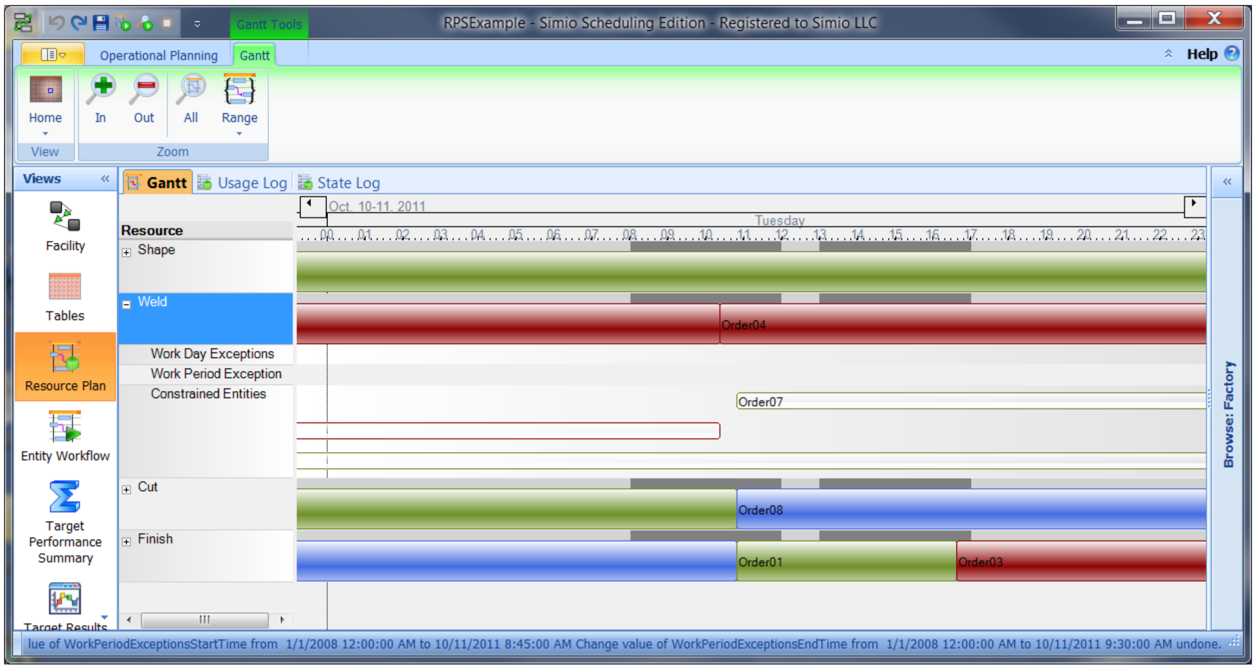
						Target Ship Date Plan		Target Ship Date Risk	
Order ID	Part Type	Release Date	Due Date	Order Status	Ship Date	Value	Status	Average	OnTime Probability
Order01	A	10/3/2011 12:00:00 AM	10/12/2011 11:00:00 PM	New	10/11/2011 4:36:04 PM	10/11/2011 4:36:04 PM	OnTime	10/11/2011 6:30:55 PM	93.34 %
Order02	A	10/3/2011 12:00:00 AM	10/14/2011 9:00:00 PM	New	10/13/2011 9:12:04 AM	10/13/2011 9:12:04 AM	OnTime	10/13/2011 1:21:24 PM	93.34 %
Order03	B	10/4/2011 12:00:00 AM	10/13/2011 4:00:00 PM	New	10/12/2011 1:24:04 PM	10/12/2011 1:24:04 PM	OnTime	10/12/2011 3:17:08 PM	93.34 %
Order04	B	10/4/2011 12:00:00 AM	10/14/2011 2:00:00 PM	New	10/14/2011 10:48:04 AM	10/14/2011 10:48:04 AM	OnTime	10/14/2011 2:57:00 PM	58.67 %
Order05	A	10/5/2011 12:00:00 AM	10/19/2011 9:00:00 PM	New	10/17/2011 1:31:39 PM	10/17/2011 1:31:39 PM	OnTime	10/17/2011 2:59:37 PM	93.34 %
Order06	C	10/6/2011 12:00:00 AM	10/12/2011 12:00:00 AM	New	10/11/2011 10:48:04 AM	10/11/2011 10:48:04 AM	OnTime	10/11/2011 6:18:58 PM	58.67 %
Order07	A	10/7/2011 12:00:00 AM	10/19/2011 4:00:00 PM	New	10/18/2011 3:37:44 PM	10/18/2011 3:37:44 PM	OnTime	10/19/2011 12:02:17 AM	82.94 %
Order08	C	10/8/2011 12:00:00 AM	10/13/2011 10:00:00 PM	New	10/13/2011 3:00:04 PM	10/13/2011 3:00:04 PM	OnTime	10/14/2011 1:21:50 AM	55.20 %
Order09	B	10/9/2011 12:00:00 AM	10/19/2011 4:00:00 PM	New	10/18/2011 9:19:39 AM	10/18/2011 9:19:39 AM	OnTime	10/18/2011 7:34:25 AM	93.34 %
OrderWIP1	A	10/3/2011 12:00:00 AM	10/10/2011 4:00:00 PM	WIP	10/6/2011 4:03:41 PM	10/6/2011 4:03:41 PM	OnTime	10/6/2011 10:02:14 PM	93.34 %
OrderWIP2	A	10/3/2011 12:00:00 AM	10/5/2011 4:00:00 PM	WIP	10/6/2011 9:18:05 AM	10/6/2011 9:18:05 AM	Tardy	10/6/2011 7:37:03 AM	13.59 %
OrderWIP3	B	10/3/2011 12:00:00 AM	10/7/2011 4:00:00 PM	WIP	10/6/2011 1:39:41 PM	10/6/2011 1:39:41 PM	OnTime	10/6/2011 3:49:30 PM	93.34 %
OrderWIP4	B	10/3/2011 12:00:00 AM	10/5/2011 4:00:00 PM	WIP	10/4/2011 4:42:05 PM	10/4/2011 4:42:05 PM	OnTime	10/5/2011 1:45:26 AM	93.34 %
OrderWIP5	C	10/3/2011 12:00:00 AM	10/6/2011 4:00:00 PM	WIP	10/5/2011 1:30:05 PM	10/5/2011 1:30:05 PM	OnTime	10/5/2011 1:34:38 PM	93.34 %
OrderWIP6	C	10/3/2011 12:00:00 AM	10/3/2011 4:00:00 PM	WIP	10/3/2011 12:00:00 AM	10/3/2011 12:00:00 AM	OnTime	10/3/2011 12:00:00 AM	93.34 %
OrderWIP7	C	10/3/2011 12:00:00 AM	10/3/2011 4:00:00 PM	WIP	10/3/2011 1:48:01 PM	10/3/2011 1:48:01 PM	OnTime	10/3/2011 1:37:44 PM	89.87 %

Risk analysis run ended. Successfully ran 25 replications

Insert row Change value of WorkPeriodExceptionsStartTime from 1/1/2008 12:00:00 AM to 10/11/2011 8:45:00 AM Change value of WorkPeriodExceptionsEndTime from 1/1/2008 12:00:00 AM to 10/11/2011 9:30:00 AM undone.

The Resource Plan provides a summary of how each resource in the system is utilized by each entity that is processed by that resource. The Resource Plan presents this information in both a graphical view

(Gantt) and tabular view (Usage and State Logs). The resource Gantt displays each entity that utilizes the resource as a time bar drawn from the starting to ending time of the usage period. In this example the color for this bar is specified in the *GanttColor* column in the *ManufacturedItems* table. The time bar is down over a background that depicts the changing state of the resource (e.g. Starved, Processing, Blocked, Offshift, and Failed).



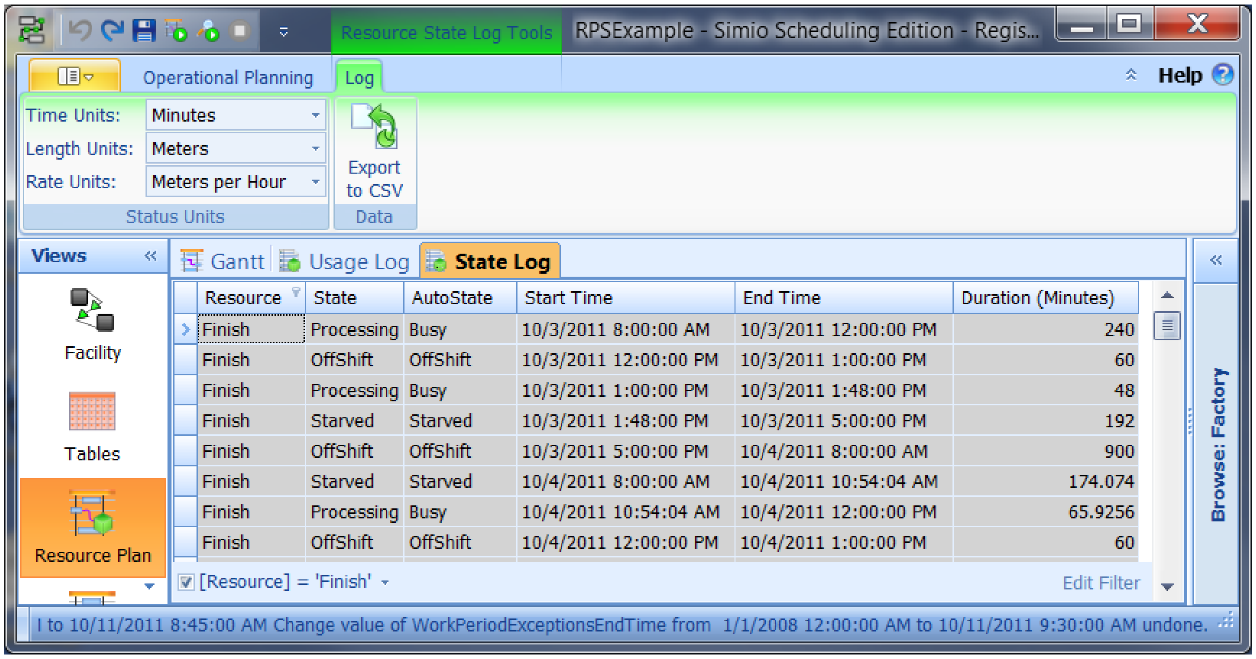
lue of WorkPeriodExceptionsStartTime from 1/1/2008 12:00:00 AM to 10/11/2011 8:45:00 AM Change value of WorkPeriodExceptionsEndTime from 1/1/2008 12:00:00 AM to 10/11/2011 9:30:00 AM undone.

The timeline for the Gantt is shown across the top. The Gantt can be scrolled left and right by clicking and dragging using the mouse, and up and down using either the vertical scroll bar or the mouse wheel. The time scale can be continuously zoomed by pressing the Ctrl key and dragging the timeline left and right using the mouse. The size of a single resource row can be changed by dragging and dropping the horizontal separator between the resource names. All resource rows can be simultaneously resized by dragging and dropping any resource separator while holding down the Ctrk key. The Gantt ribbon may also be used to zoom in/out, zoom to all, zoom to a specific range (e.g. a week), or scroll to a specific starting day (e.g. first day of the current week).

Each resource row also has a + symbol that can be opened to expose work day and period exception editing as well as the constrained entities. The scheduler may use the exception editing to assign a specific day exception (e.g. double shift) that replaces the normal work schedule for that resource on that day, or assign a specific period exception (from 10 am to 11 am) for that day to define a non-standard change in capacity – such as a breakdown.

The constrained entities section depicts each entity that waits for this resource. The waiting entity is outlined using the entity color assigned to the time bar. Note that these bars can be reduced to lines by reducing the size of the constraint area.

The Usage and State logs provide this same information in a convenient grid format that can be filtered and exported. The Usage log has a row for each entity that utilizes the resource, and displays the resource name, entity name (owner), and the start time, end time, and duration for the usage. The State log has a row for each change of state by each resource and along with the start time, end time, and duration of each state. For example the following shows the state of the *Finish* workstation; the *Resource* column filter has been changed from *All* to *Finish*.



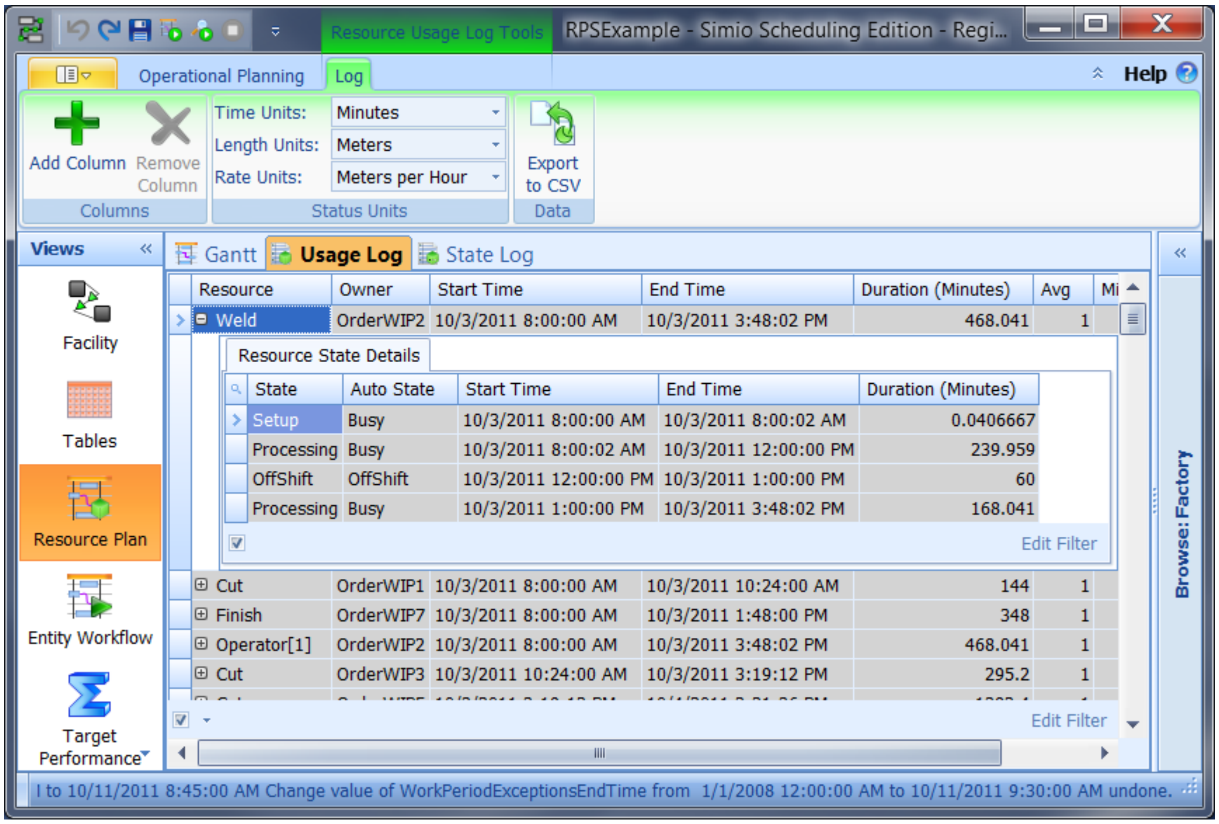
Time Units: Minutes
Length Units: Meters
Rate Units: Meters per Hour

Resource	State	AutoState	Start Time	End Time	Duration (Minutes)
Finish	Processing	Busy	10/3/2011 8:00:00 AM	10/3/2011 12:00:00 PM	240
Finish	OffShift	OffShift	10/3/2011 12:00:00 PM	10/3/2011 1:00:00 PM	60
Finish	Processing	Busy	10/3/2011 1:00:00 PM	10/3/2011 1:48:00 PM	48
Finish	Starved	Starved	10/3/2011 1:48:00 PM	10/3/2011 5:00:00 PM	192
Finish	OffShift	OffShift	10/3/2011 5:00:00 PM	10/4/2011 8:00:00 AM	900
Finish	Starved	Starved	10/4/2011 8:00:00 AM	10/4/2011 10:54:04 AM	174.074
Finish	Processing	Busy	10/4/2011 10:54:04 AM	10/4/2011 12:00:00 PM	65.9256
Finish	OffShift	OffShift	10/4/2011 12:00:00 PM	10/4/2011 1:00:00 PM	60

[Resource] = 'Finish'

l to 10/11/2011 8:45:00 AM Change value of WorkPeriodExceptionsEndTime from 1/1/2008 12:00:00 AM to 10/11/2011 9:30:00 AM undone.

The Usage and State logs are also linked together as a master-detail view. Each row in the Usage log has a + sign that can be expanded to display the states for that resource for a specific usage. For example the following shows the states for the *Weld* resource during the processing of *OrderWIP2*.



Time Units: Minutes
Length Units: Meters
Rate Units: Meters per Hour

Resource	Owner	Start Time	End Time	Duration (Minutes)	Avg	Mi
Weld	OrderWIP2	10/3/2011 8:00:00 AM	10/3/2011 3:48:02 PM	468.041	1	

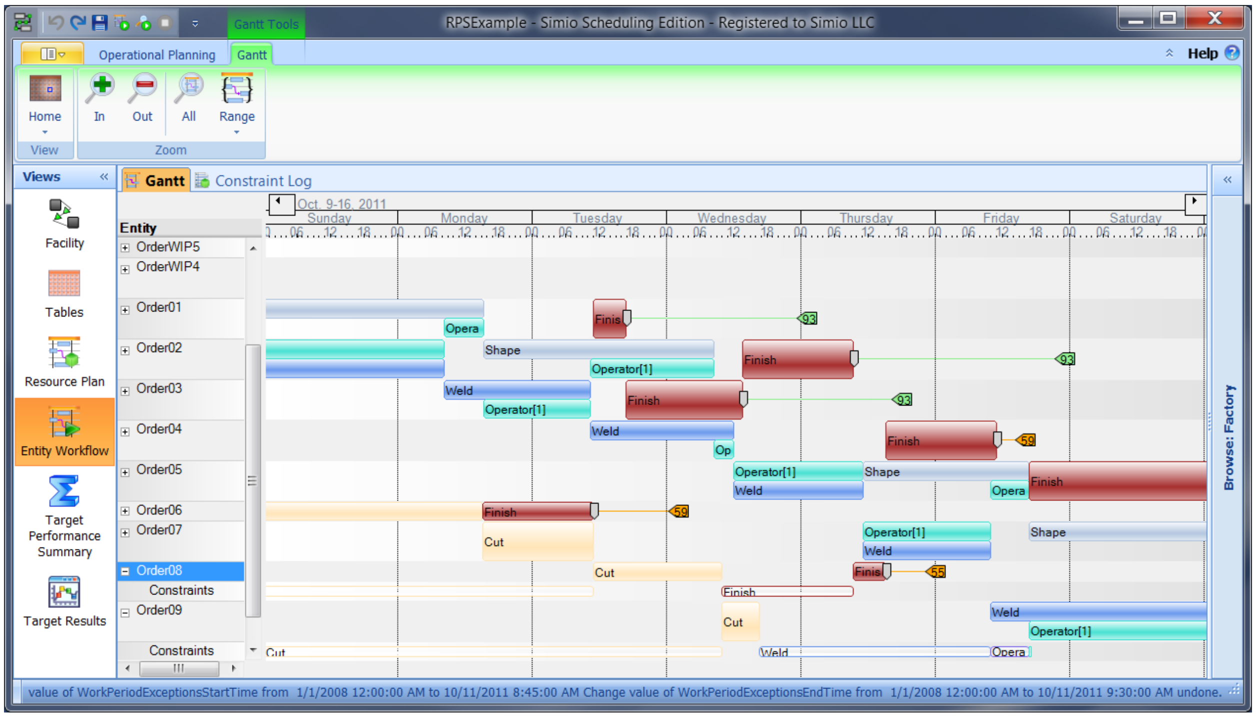
Resource State Details

State	Auto State	Start Time	End Time	Duration (Minutes)
Setup	Busy	10/3/2011 8:00:00 AM	10/3/2011 8:00:02 AM	0.0406667
Processing	Busy	10/3/2011 8:00:02 AM	10/3/2011 12:00:00 PM	239.959
OffShift	OffShift	10/3/2011 12:00:00 PM	10/3/2011 1:00:00 PM	60
Processing	Busy	10/3/2011 1:00:00 PM	10/3/2011 3:48:02 PM	168.041

Resource	Owner	Start Time	End Time	Duration (Minutes)	Avg	Mi
Cut	OrderWIP1	10/3/2011 8:00:00 AM	10/3/2011 10:24:00 AM	144	1	
Finish	OrderWIP7	10/3/2011 8:00:00 AM	10/3/2011 1:48:00 PM	348	1	
Operator[1]	OrderWIP2	10/3/2011 8:00:00 AM	10/3/2011 3:48:02 PM	468.041	1	
Cut	OrderWIP3	10/3/2011 10:24:00 AM	10/3/2011 3:19:12 PM	295.2	1	

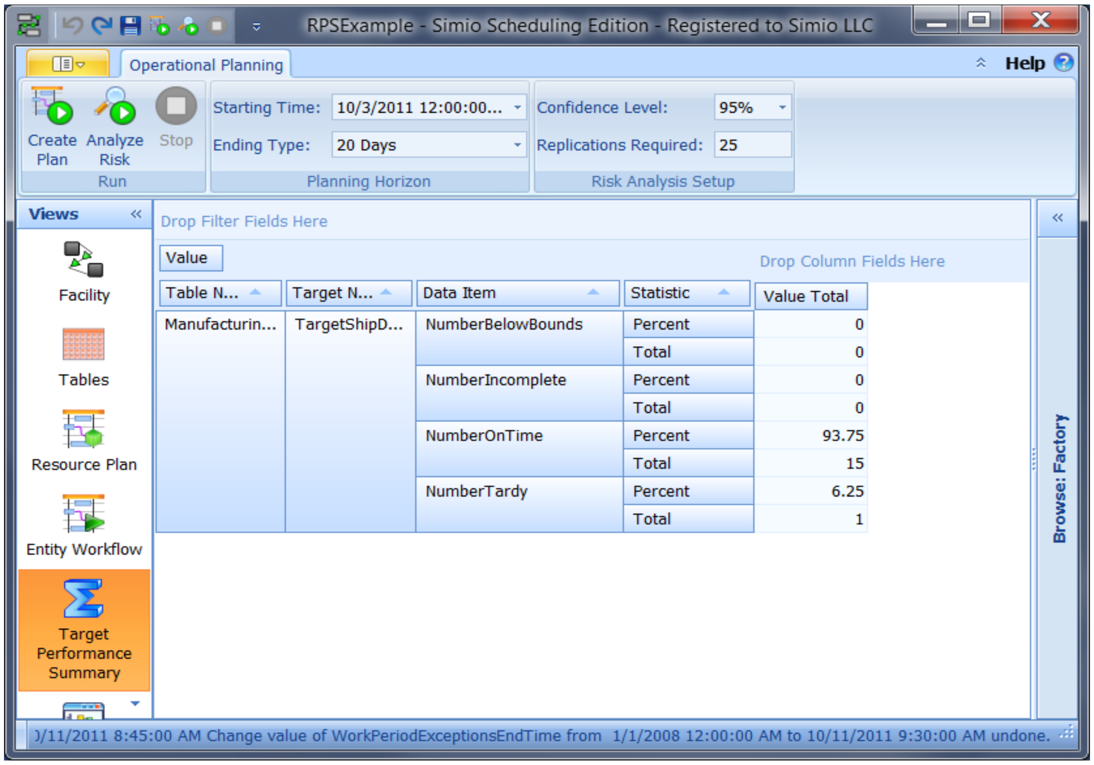
l to 10/11/2011 8:45:00 AM Change value of WorkPeriodExceptionsEndTime from 1/1/2008 12:00:00 AM to 10/11/2011 9:30:00 AM undone.

The Entity Workflow view provides a detailed view of the progress of each entity through the facility. In this case the rows correspond to entities (e.g. jobs), and the time bars going across represent resource usages by that entity. Note that in some cases multiple resources are used at the same time. This view also shows any date-time based milestones and upper/lower target values for those milestones, along with the color code probabilities of meeting each target. Note that by clicking on the + symbol for each entity the non-value added (NVA) constraints are displayed. In the entity Gantt below we see that Order08 is risky (55% chance of OnTime), and spends nearly a day waiting for access to the Finish workstation. This suggests that adding overtime to the Finish station might be a good risk mitigation strategy for testing. We can do this by adding either a work day or work period exception on the resource Gantt.



value of WorkPeriodExceptionsStartTime from 1/1/2008 12:00:00 AM to 10/11/2011 8:45:00 AM Change value of WorkPeriodExceptionsEndTime from 1/1/2008 12:00:00 AM to 10/11/2011 9:30:00 AM undone.

The Target Performance Summary provides a quick overview of the schedule performance relative to the upper and lower bounds for each target value. The results show that 6.25% of our orders are tardy.



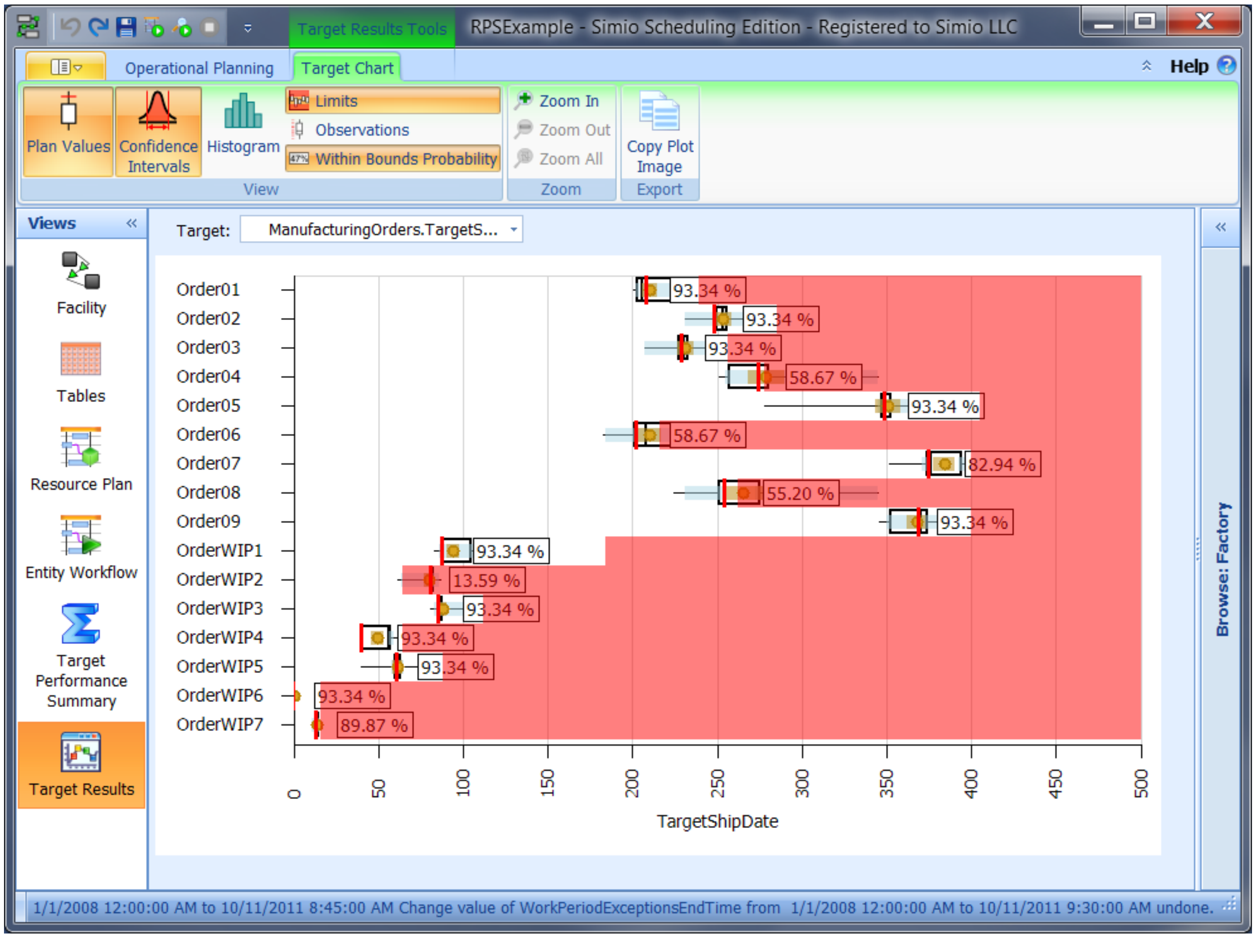
Starting Time: 10/3/2011 12:00:00...
Ending Type: 20 Days
Confidence Level: 95%
Replications Required: 25

Value

Table N...	Target N...	Data Item	Statistic	Value Total
Manufacturin...	TargetShipD...	NumberBelowBounds	Percent	0
			Total	0
		NumberIncomplete	Percent	0
			Total	0
		NumberOnTime	Percent	93.75
			Total	15
		NumberTardy	Percent	6.25
			Total	1

)/11/2011 8:45:00 AM Change value of WorkPeriodExceptionsEndTime from 1/1/2008 12:00:00 AM to 10/11/2011 9:30:00 AM undone.

The Target Results view provides a detailed summary of each our schedule performance relative to each target that we have defined for the model.   This view is useful for analyzing the risk associated with an entity at a detailed statistical view. This view displays a SMORE plot for the selected target for each entity. The SMORE plot consist of the mean value (gold circle), confidence intervals (brown/blue bands), range (horizontal line), and median and upper/lower percentiles (black vertical lines).   The upper and lower bounds are shaded in red.



Summary

Risk-based Planning and Scheduling provides detailed and realistic plans/schedules along with the associated risks. With Risk-based Planning and Scheduling a user can quickly glance at the output in the orders table or the entity workflow Gantt to access the quality of the resulting schedule.   A user can also view the NVA time spent by entities that are constrained by resources, operators, or materials. This small example has highlighted by basic functionality in Simio to support Risk-based Planning and Scheduling.