

Manufacturing Simulation with Plant Simulation and SimTalk

Steffen Bangsow

Manufacturing Simulation with Plant Simulation and SimTalk

Usage and Programming with
Examples and Solutions

 Springer

Steffen Bangsow
Freiligrathstraße 23
08058 Zwickau
Germany
E-mail: steffen@bangsow.de

ISBN 978-3-642-05073-2

e-ISBN 978-3-642-05074-9

DOI 10.1007/978-3-642-05074-9

Library of Congress Control Number: 2010923701

© 2010 Springer-Verlag Berlin Heidelberg

Originally published in German “Fertigungssimulationen mit Plant Simulation und SimTalk” with Carl Hanser Verlag, Munich. © 2008

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Camera-ready by author, data conversion by Markus Richter, Heidelberg

Cover Design: Scientific Publishing Services Pvt. Ltd., Chennai, India

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

Based on the competition of international production networks, the pressure to increase the efficiency of production systems has increased significantly. In addition, the number of technical components in many products and as a consequence also the requirements for corresponding assembly processes and logistics processes increases. International logistics networks require corresponding logistics concepts.

These requirements can be managed only by using appropriate Digital Factory tools in the context of a product lifecycle management environment, which allows reusing data, supports an effective cooperation between different departments, and provides up-to-date and relevant data to every user who needs it.

Simulating the complete material flow including all relevant production, storage, and transport activities is recognized as a key component of the Digital Factory in the industry and as of today widely used and accepted. Cutting inventory and throughput time by 20–60% and enhancing the productivity of existing production facilities by 15–20% can be achieved in real-life projects.

The purpose of running simulations varies from strategic to tactical up to operational goals. From a strategic point of view, users answer questions like which factory in which country suits best to produce the next generation product taking into account factors like consequences for logistics, worker efficiency, downtimes, flexibility, storage costs, etc., looking at production strategies for the next years. In this context, users also evaluate the flexibility of the production system, e.g., for significant changes of production numbers — a topic which becomes more and more important. On a tactical level, simulation is executed for a time frame of 1–3 months in average to analyze required resources, optimize the sequence of orders, and lot sizes. For simulation on an operational level, data are imported about the current status of production equipment and the status of work in progress to execute a forward simulation till the end of the current shift. In this case, the purpose is to check if the target output for the shift will be reached and to evaluate emergency strategies in case of disruptions or capacities being not available unexpectedly.

In any case, users run simulation to take a decision about a new production system or evaluate an existing production system. Usually, the value of those systems is a significant factor for the company, so the users have to be sure that they take the right decision based on accurate numbers. There are several random processes in real production systems like technical availabilities, arrival times of assembly parts, process times of human activities, etc., so stochastic processes play an important role for throughput simulation. Therefore, Plant Simulation provides a whole range of easy-to-use tools to analyze models with stochastic processes, to

calculate distributions for sample values, to manage simulation experiments, and to determine optimized system parameters.

Besides that, results of a simulation model depend on the quality of the input data and the accuracy of the model compared to the behavior of the real production system. As soon as assembly processes are involved, several transport systems with their transport controls, workers with multiple qualification profiles or storage logic, production processes become highly complex. Plant Simulation provides all necessary functionality to model, analyze, and maintain large and complex systems in an efficient way. Key features like object orientation and inheritance allow users to develop, exchange/reuse, and maintain their own objects and libraries to increase modeling efficiency. The unique Plant Simulation optimization capabilities support users to optimize multiple system parameters at once like the number of transporters, monorail carriers, buffer/storage capacities, etc., taking into account multiple evaluation criteria like reduced stock, increased utilization, increased throughput, etc.

Based on these accurate modeling capabilities and statistic analysis capabilities, typically an accuracy of at least 99% of the throughput values is achieved with Plant Simulation models in real-life projects depending on the level of detail. Based on the price of production equipment, a return on investment of the costs to introduce simulation is quite often already achieved after the first simulation project.

Visualizing the complete model in the Plant Simulation 3D environment allows an impressive 3D presentation of the system behavior. Logfiles can be used to visualize the simulation in a Virtual Reality (VR) environment. The support of a Siemens PLM Software unified 3D graphics engine and unified graphics format allows a common look-and-feel and easy access to 3D graphics which were created in other tools like digital product design or 3D factory layout design tools.

The modeling of complex logic always requires the usage of a programming language. Plant Simulation simplifies the need to work with programming language tremendously by supporting the user with templates, with an extensive examples collection and a professional debugging environment.

Compared to other simulation tools in the market, Plant Simulation supports a very flexible way of working with the model, e.g., by changing system parameters while the simulation is running.

This book provides the first comprehensive introduction to Plant Simulation. It supports new users of the software to get started quickly, provides an excellent introduction how to work with the embedded programming language SimTalk, and even helps advanced users with examples of typical modeling tasks. The book focuses on the basic knowledge required to execute simulation projects with Plant Simulation, which is an excellent starting point for real-life projects.

We wish you a lot of success with Tecnomatix Plant Simulation.

Table of Contents

1	Introducing Factory Simulation	1
1.1	Uses.....	1
1.2	Definitions.....	2
1.3	Procedure of Simulation.....	2
1.3.1	Formulation of Problems	2
1.3.2	Test of the Simulation-Worthiness	3
1.3.3	Formulation of Targets	3
1.3.4	Data Collection	3
1.3.5	Modeling.....	4
1.3.5.1	First Modeling Stage.....	4
1.3.5.2	Second Modeling Stage	5
1.3.6	Executing Simulation Runs	5
1.3.7	Result Analysis and Result Interpretation.....	5
1.3.8	Documentation.....	5
2	Plant Simulation	7
2.1	First Steps.....	7
2.1.1	Online Tutorial	7
2.1.2	Examples	7
2.1.3	Help	7
2.1.4	Website.....	8
2.2	Introductory Example	8
2.2.1	The Program	8
2.2.1.1	The Program Window	8
2.2.1.2	The Class Library.....	8
2.2.1.3	The Console	9
2.2.1.4	The Toolbox.....	9
2.2.2	First Simulation Example	9
2.2.2.1	Design of the Model.....	9
2.2.2.2	Insert Objects into the Frame	10
2.2.2.3	Connect the Objects	10
2.2.2.4	Define the Settings of the Objects.....	10
2.2.2.5	Run the Simulation.....	11
2.3	Modeling.....	12
2.3.1	Object-Related Modeling.....	12

2.3.2	Object-Oriented Modeling.....	12
2.3.2.1	Objects and Properties	12
2.3.2.2	Classes and Instances	13
2.3.2.3	Inheritance.....	13
2.3.2.4	Duplication and Derivation	13
3	Standard Classes in PLANT SIMULATION	17
3.1	Overview.....	17
3.2	Material Flow Objects.....	17
3.2.1	General Behavior of the Material Flow Objects.....	17
3.2.1.1	Time Consumption.....	18
3.2.1.2	Capacity	20
3.2.1.3	Blocking.....	20
3.2.1.4	Failures.....	21
3.2.2	The Source.....	25
3.2.2.1	Basic Behavior	25
3.2.2.2	Settings.....	25
3.2.3	The Drain.....	29
3.2.4	The SingleProc	29
3.2.5	The ParallelProc	29
3.2.5.1	Basic Behavior and Use	29
3.2.5.2	Settings.....	30
3.2.6	The AssemblyStation.....	32
3.2.7	The Buffer.....	34
3.2.8	The DismantleStation	35
3.2.8.1	Basic Behavior	35
3.2.8.2	Cycle	38
3.2.9	The Store	39
3.2.10	The Line.....	40
3.2.10.1	Behavior of the Line	40
3.2.10.2	Attributes of the Line	40
3.2.10.3	Curves and Corners.....	43
3.2.11	AngularConverter and Turntable	44
3.2.11.1	Settings of the AngularConverter.....	45
3.2.11.2	Settings of the Turntable	46
3.2.12	The PickAndPlace Robot.....	46
3.2.12.1	Basic Behavior	46
3.2.12.2	Attributes.....	47
3.2.13	The Track.....	50
3.2.14	The Sorter	51
3.2.14.1	Basic Behavior	51
3.2.14.2	Attributes of the Sorter	52
3.2.15	The FlowControl.....	55

3.2.15.1	Basic Behavior	55
3.2.15.2	Attributes.....	55
3.3	Resource Objects.....	60
3.3.1	Usage and Example	60
3.3.2	The Worker-WorkerPool-Workplace-FootPath Concept	61
3.3.3	The Broker.....	61
3.3.4	The WorkerPool	62
3.3.5	The Worker.....	63
3.3.6	The Footpath.....	63
3.3.7	The Workplace	64
3.3.8	Worker Transporting Parts.....	65
3.4	General Objects.....	66
3.4.1	The Frame.....	66
3.4.1.1	General.....	66
3.4.1.2	The Frame Window	67
3.4.2	The Connector	68
3.4.2.1	Basic Behavior	68
3.4.2.2	Attributes.....	69
3.4.3	The EventController	69
3.4.3.1	Basic Behavior	69
3.4.4	The Interface.....	71
3.4.4.1	Basic Behavior	71
3.4.4.2	Attributes of the Interface	74
4	Icons	75
4.1	Basics	75
4.2	The Icon Editor	75
4.3	Drawing Icons.....	76
4.4	Inserting Images.....	76
4.4.1	Insert Images from the Clipboard	76
4.4.2	Inserting Images from a File.....	77
4.5	Changing the Background Color of the Frame	78
4.6	Animation Structures and Reference Points.....	78
4.6.1	Basics.....	78
4.6.2	Set Reference Points	79
4.6.3	Animation Structures.....	80
4.7	Animating Frames.....	81
5	Programming with SimTalk.....	85
5.1	The Object Method	85
5.1.1	Introductory Example	85

5.2	The Method Editor	87
5.2.1	Line Numbers, Entering Text	87
5.2.2	Bookmarks	87
5.2.3	Code Completion	88
5.2.4	Information About Attributes and Methods	88
5.2.5	Templates	89
5.2.6	The Debugger	90
5.3	SimTalk	90
5.3.1	Names	91
5.3.2	Anonymous Identifiers	91
5.3.3	Paths	92
5.3.3.1	Absolute Path	93
5.3.3.2	Relative Path	93
5.3.3.3	Name Scope	93
5.3.4	Comments	94
5.4	Variables and Data Types	95
5.4.1	Variables	95
5.4.1.1	Local Variables	95
5.5	Operators	99
5.5.1	Mathematical Operators	99
5.5.2	Logical (Relational) Operators	99
5.5.3	Assignments	100
5.6	Branching	102
5.7	Case Differentiation	104
5.8	Loops	105
5.8.1	Conditional Loops	105
5.8.1.1	Header-Controlled Loops	105
5.8.1.2	Footer-Controlled Loops	106
5.8.2	For-Loop	107
5.9	Methods and Functions	108
5.9.1	Passing Arguments	108
5.9.2	Passing Several Arguments at the Same Time	109
5.9.3	Result of a Function	110
5.9.4	Predefined SimTalk Functions	111
5.9.4.1	Functions for Manipulating Strings	111
5.9.4.2	Mathematical Functions	112
5.9.5	Method Call	113
5.9.5.1	Sensors	113
5.9.5.2	Other Events for Calling Methods	114
5.9.5.3	Method Call After a Certain Timeout	115

6	Simtalk and Material Flow Objects.....	117
6.1	Attributes of the Material Flow Objects.....	117
6.2	State of Material Flow Objects	119
6.2.1	Operational, Failed, Pause	119
6.2.2	Ready	121
6.2.3	Empty	122
6.2.4	Occupied.....	123
6.2.5	Full.....	123
6.2.6	Capacity	124
6.3	Suspending Methods	126
6.4	Observer.....	127
6.5	Content of the Objects.....	129
6.6	Sensors	132
6.7	User-Defined Attributes	134
7	Mobile Units.....	139
7.1	Standard Methods of Mobile Units	139
7.1.1	Create.....	139
7.1.2	MU-Related Attributes and Methods.....	140
7.2	Length, Width, and Booking Point.....	141
7.3	The Entity.....	142
7.4	The Container.....	143
7.4.1	Attributes of the Container	143
7.4.2	Loading Containers.....	143
7.4.3	Unloading Containers	145
7.5	The Transporter.....	158
7.5.1	Basic Behavior.....	158
7.5.2	Attributes of the Transporter.....	158
7.5.3	Routing	160
7.5.3.1	Automatic Routing.....	160
7.5.3.2	Driving Control	164
7.5.4	Methods and Attributes of the Transporter	167
7.5.4.1	Creating a Transporter.....	167
7.5.4.2	Unloading a Transporter	167
7.5.4.3	Driving Forward and Backward.....	167
7.5.4.4	Stopping and Continuing.....	168
7.5.4.5	Drive after a Certain Time	169
7.5.4.6	Start Delay Duration	171
7.5.4.7	Important Methods and Attributes of the Transporter	176

8	Information Flow Objects	183
8.1	The List Editor	183
8.2	The CardFile	184
8.3	StackFile and QueueFile	195
8.4	The TableFile	200
8.4.1	Basic Behavior	200
8.4.2	Methods and Attributes of the TableFile	202
8.4.3	Calculating within Tables	204
8.5	The TimeSequence	208
8.5.1	Basic Behavior	208
8.5.2	Settings	208
8.6	The Trigger	212
8.6.1	Basic Behavior	212
8.7	The ShiftCalendar	215
8.8	The Generator	217
8.9	The AttributeExplorer	218
8.10	The EventController	221
9	Statistics	223
9.1	Basics	223
9.1.1	Statistics Collection Period	223
9.1.2	Activating Statistics Collection	224
9.2	Statistics – Methods and Attributes	224
9.3	User Interface Objects	230
9.3.1	Chart	230
9.3.1.1	Plotter	230
9.3.1.2	Chart Types	233
9.3.1.3	Statistics Wizard	236
9.3.1.4	Histograms	237
9.3.2	The Sankey Diagram	238
9.3.3	The Bottleneck Analyzer	241
9.3.4	The Display	242
9.3.4.1	Behavior	242
9.3.4.2	Attributes of the Display	243
9.3.5	The Comment	245
9.3.6	The Report	246
9.3.6.1	Automatic Resource Report (Statistics Report)	246
9.3.6.2	Report Header	246
9.3.6.3	Report Data	247

9.3.6.4	Texts in Reports	249
9.3.6.5	Show Objects in Reports	250
9.3.6.6	Show Images in Reports.....	252
10	User Interface Objects	253
10.1	General	253
10.2	Elements of the Dialog.....	253
10.2.1	The Dialog Object	254
10.2.2	Insert Elements	254
10.2.3	Callback Function.....	256
10.2.4	The Static Text Box.....	257
10.2.5	The Edit Text Box	257
10.2.6	Images in Dialogs	258
10.2.7	Buttons.....	260
10.2.8	Radio Buttons	261
10.2.9	Checkbox.....	263
10.2.10	Drop-Down List Box and List Box.....	263
10.2.11	List View	265
10.2.12	Tab Control.....	267
10.2.13	Group Box	267
10.2.14	Menu and Menu Item.....	267
10.3	Accessing Dialogs.....	268
10.4	Protection of Methods and Objects	269
10.5	Validation User Input.....	270
10.5.1	Type Validation and Plausibility Check	270
10.5.2	Message Box.....	271
10.6	HTML-Help.....	272
11	Data Exchange.....	273
11.1	DDE with Plant Simulation.....	273
11.1.1	Read Plant Simulation Data in Microsoft Excel	273
11.1.2	Excel Data Import in Plant Simulation	274
11.1.3	Plant Simulation Remote Control.....	276
11.1.4	DDE Hotlinks	277
11.2	The File Interface	278
11.3	The ODBC Interface	279
11.3.1	Setup an ODBC Data Source.....	280
11.3.2	Read Data from a Database	282
11.3.3	Write Data in a Database	283
11.3.4	Delete Data in a Database Table.....	284

11.3.5 SQL Commands	285
11.3.5.1 SELECT	285
11.3.5.2 INSERT (Insert New Records)	286
11.3.5.3 UPDATE (Change Data).....	286
11.3.5.4 DELETE	287
12 Plant Simulation 3D	289
12.1 Sample Project	289
12.2 Views and Move in Plant Simulation 3D.....	290
12.3 Control the Simulation in Plant Simulation 3D.....	290
Index	293

Table of Examples

Example 1: Properties of the SingleProc	11
Example 2: Inheritance 1	13
Example 3: Inheritance 2	14
Example 4: Inheritance 3	14
Example 5: Material Flow – Time Consumption.....	18
Example 6: Blocking	20
Example 7: Failure 1.....	21
Example 8: Failure 2.....	22
Example 9: Multiple Failures.....	24
Example 10: Source Delivery Table	26
Example 11: Randomly Produce MUs.....	28
Example 12: ParallelProc.....	29
Example 13: ParallelProc; Different Processing Times	31
Example 14: Assembly	32
Example 15: DismantleStation	35
Example 16: Dismantle Station, Exit Sequence.....	37
Example 17: Cycle.....	39
Example 18: Line 1	41
Example 19: Line 2.....	42
Example 20: Turntable and AngularConverter	45
Example 21: PickAndPlace 1.....	47
Example 22: PickAndPlace 2.....	48
Example 23: Track.....	50
Example 24: Sorter	52
Example 25: FlowControl 1.....	55
Example 26: FlowControl 2.....	57
Example 27: Just in Sequence.....	59
Example 28: Resources for Repairs.....	60
Example 29: Resources Exit Strategy Carry Part Away	65
Example 30: Frame and Interface	71
Example 31: Icon Editor	75
Example 32: Reference Points and Animation Structures	79
Example 33: Animation Structures	80
Example 34: Machine with a Ringloader, Animation on a Frame	81
Example 35: Stock Removal.....	85
Example 36: root.....	91
Example 37: Anonymous Identifier self.....	92

Example 38: Declarating Variables	95
Example 39: Global Variables	97
Example 40: Global Variable 2.....	98
Example 41: Logical Operators	100
Example 42: Variable – Value Assignment.....	100
Example 43: Type Conversion 1.....	101
Example 44: Branch 1.....	103
Example 45: Branch 2.....	103
Example 46: Case Differentiation.....	104
Example 47: while-Loop.....	105
Example 48: repeat-Loop.....	106
Example 49: from-Loop.....	107
Example 50: for-Loop.....	107
Example 51: for-Loop with downto.....	108
Example 52: Passing Arguments 1	109
Example 53: Passing Arguments 2	110
Example 54: Results of a Function	110
Example 51: Functions for Manipulating Strings	112
Example 56: Methodcalls by Sensors	113
Example 57: Fail Control.....	114
Example 58: Ref-Call	116
Example 59: Basic Settings	117
Example 60: Status Display of Material Flow Objects.....	119
Example 61: Replacement Machine	120
Example 62: Object State Empty	122
Example 63: Method full	123
Example 64: Machine with Parallel Processing Stations.....	124
Example 65: Observer	127
Example 66: deleteMovables.....	129
Example 67: Method cont.....	129
Example 68: Surface Treatment.....	130
Example 69: Method MU	131
Example 70: Sensors, Color Sorting.....	132
Example 71: Production Costs and Working Assets.....	135
Example 72: Create MUs.....	139
Example 73: Change MU Length	141
Example 74: Loading Containers.....	143
Example 75: Batch Production	145
Example 76: Saw	151
Example 77: Kanban Control.....	153
Example 78: Automatic Routing	159
Example 79: Automatic Routing	160
Example 80: Driving Control.....	164
Example 81: Unloading a Transporter	167

Example 82: Stopping Transporters.....	168
Example 83: Start Delay Duration, Crossroads	172
Example 84: Portal Loader Parallel Processing	176
Example 85: Materials List.....	184
Example 86: Handling by a Robot.....	186
Example 87: Queuing	195
Example 88: Determining Sensor Positions.....	198
Example 89: Lot Change	200
Example 90: Calculating Machine-Hour Rates.....	204
Example 91: TimeSequence	209
Example 92: Trigger	212
Example 93: ShiftCalendar	215
Example 94: Generator, Outward Stock Movement	217
Example 95: AttributeExplorer.....	218
Example 96: Statistics.....	225
Example 97: Plotter	230
Example 98: Chart from a TableFile.....	234
Example 99: Histogram	237
Example 100: Sankey Diagram	238
Example 101: Display.....	243
Example 102: Report	246
Example 103: Dialog	253
Example 104: Error Dialog.....	260
Example 105: Statistics Dialog.....	263
Example 106: Dialog Product Mix with Listview	265
Example 107: Dialog Menu	268
Example 108: Protection of Frames.....	269
Example 109: Type Validation	270
Example 110: Data Exchange DDE Excel.....	273
Example 111: Data Exchange, Importing a Working Plan from Excel	274
Example 112: DDE Remote Control	276
Example 113: DDE Hotlinks	277
Example 114: File Interface.....	278
Example 115: ODBC	280
Example 116: ODBC – Write Simulation Results into a Database	283
Example 117: ODBC – Delete Data	284
Example 118: Plant Simulation 3D.....	289

1 Introducing Factory Simulation

Simulation technology is an important tool for planning, implementing, and operating complex technical systems.

Several trends in the economy such as

- increasing product complexity and variety
- increasing quality demands in connection with high cost pressure
- increasing demands regarding flexibility
- shorter product life cycles
- shrinking lot sizes
- increasing competitive pressure

lead to shorter planning cycles. Simulation has found its place where simpler methods no longer provide useful results.

1.1 Uses

You can use simulation during planning, implementation, and operation of equipment. Possible questions can be:

- **Planning phase**
 - Identification of bottlenecks in derivation of potential improvement
 - Uncover hidden, unused potentials
 - Minimum and maximum of utilization
 - Juxtaposition of different planning alternatives
 - Test of arguments regarding capacity, effectiveness of control, performance limits, bottlenecks, throughput speed, and volume of stocks
 - Visualization of planning alternatives for decision making
- **Implementation phase**
 - Performance tests
 - Problem analysis, performance test on future requirements
 - Simulation of exceptional system conditions and accidents
 - Training new employees (e.g., incident management)
 - Simulation of ramp up and cool-down behaviors
- **Operational phase**
 - Testing of control alternatives
 - Review of emergency strategies and accident programs
 - Proof of quality assurance and fault management
 - Dispatching of orders and determination of the probable delivery dates

1.2 Definitions

Simulation (source: VDI 3633)

Simulation is the reproduction of a real system with its dynamic processes in a model. The aim is to reach transferable findings for the reality. In a wider sense, simulation means preparing, implementing, and evaluating specific experiments with a simulation model.

System: (VDI 3633)

A system is defined as a separate set of components which are related to each other.

Model: A model is a simplified replica of a planned or real system with its processes in another system. It differs in important properties only within specified tolerance from the original.

Simulation run: (source: VDI 3633)

A simulation run is the image of the behavior of the system in the simulation model within a specified period.

Experiment: (source: VDI 3633)

An experiment is a targeted empirical study of the behavior of a model by repeated simulation runs with systematic variation of arguments.

1.3 Procedure of Simulation

According to VDI guideline 3633, the following approach is recommended:

1. Formulation of problems
2. Test of the simulation-worthiness
3. Formulation of targets
4. Data collection and data analysis
5. Modeling
6. Execute simulation runs
7. Result analysis and result interpretation
8. Documentation

1.3.1 Formulation of Problems

Together with the customer of the simulation, the simulation expert must formulate the requirements for the simulation. The result of the formulated problem should be a written agreement (e.g., a technical specification), which contains concrete problems which will be studied using simulation.

1.3.2 Test of the Simulation-Worthiness

To assess the simulation-worthiness you can, for example, examine:

- The lack of analytical mathematical models (for instance, many variables)
- High complexity, many factors to be considered
- Inaccurate data
- Gradual exploration of system limits
- Repeated use of the simulation model

1.3.3 Formulation of Targets

Each company aims at a system of targets. It usually consists of a top target (such as profitability), that splits into a variety of subtargets, which interact with each other. The definition of the target system is an important preparatory step. Frequent targets for simulations are for example:

- Minimize processing time
- Maximize utilization
- Minimize inventory
- Increase in-time delivery

All defined targets must be collected and analyzed statistically at the end of the simulation runs, which implies a certain required level of detail for the simulation model. As a result, they determinate the range of the simulation study.

1.3.4 Data Collection

The data required for the simulation study can be structured as follows:

- System load data
- Organizational data
- Technical data

The following overview is a small selection of data to be collected:

Technical data	
Factory structural data	Layout Means of production Transport functions Transport routes Areas Restrictions
Manufacturing data	Use time Performance data Capacity

Material flow data	Topology
	Conveyors
	Capacities
Accident data	Functional accidents
	Availability
Organizational data	
Working time organization	Break scheme
	Shift scheme
Resource allocation	Worker
	Machines
	Conveyors
Organization	Strategy
	Restrictions
	Incident management
System load data	
Product data	Working plans
	BOMs
Job data	Production orders
	Transportation orders
	Volumes
	Dates

1.3.5 Modeling

The modeling phase includes building and testing the simulation model. Modeling usually consists of two stages:

- 1 Derive an iconic model from the conceptual model.
- 2 Transfer the model into a software model.

1.3.5.1 First Modeling Stage

First, you have to develop a general understanding of the simulated system. Based on the objectives to be tested, you have to make decisions about the accuracy of the simulation. Based on the accuracy of the simulation, necessary decisions are taken about which aspects you want to simplify. The first modeling stage covers two activities:

- Analysis (breakdown)
- Abstraction (generalization)

Using the system analysis, the complexity of the system in accordance with the original investigation targets will be dissolved by meaningful dissection of the system into its elements. By abstraction, the amount of the specific system attributes will be decreased as far as it is practical to form an essential limited image of

the original system. Typical methods of abstraction are reduction (elimination of not relevant details) and generalization (simplification of the essential details).

1.3.5.2 Second Modeling Stage

A simulation model will be built and tested. The result of modeling has to be included in the model documentation to make further changes of the simulation model possible. In practice, this step is often neglected, so that models due to the lack of documentation of functionality cannot be used. Therefore, there is a need for commenting the models and the source code during programming. In this way the explanation of the functionality is still available after programming is finished.

1.3.6 Executing Simulation Runs

Depending on the objectives of the simulation study, the experiments based on a test plan will be realized. In the test plan, the individual experiments output data, arguments of the model, objectives, and expected results are determined. It is also important to define a time span for the simulation experiments, based on the findings of the test runs. Computer runs spanning several hours or frequent repetitive experiments for the statistical coverage are not uncommon. In these cases it is helpful to check if it is possible to control the experiments by a separate programmed object (batch runs). The realization times for the experiments can be relocated partly in the night hours, so the available computing capacity can be utilized optimally. Input and output data and the underlying parameters of the simulation model must be documented for each experiment.

1.3.7 Result Analysis and Result Interpretation

The values, which will change in the modeled system, are derived from the simulation results. The correct interpretation of the simulation results significantly influences the success of a simulation study. If the results contradict the assumptions made, it is necessary to analyze what influences are responsible for the unexpected results. It is also important to realize that complex systems often have a ramp up phase. This phase may run differently in reality and in the simulation. Therefore, the results obtained during the ramp up phase are often not transferable to the modeled system and may have no influence for the evaluation (Exception: the ramp up phase of the original system has to be fully modeled).

1.3.8 Documentation

For the documentation of a simulation study, the form of a project report is recommended. The documentation should provide an overview of the timing of the study and document the work carried out. Of interest in this context is the documentation of failed system variants and constellations. The core of the project report should be a presentation of the simulation results based on the customer re-

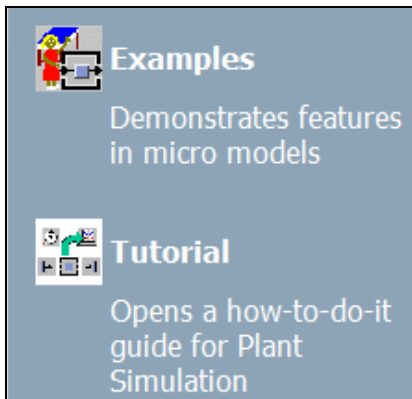
quirement specification. Resulting from the simulation study it makes sense to include proposals for actions in the documentation. Finally, we recommend describing the simulation model in its structure and its functionality.

2 Plant Simulation

2.1 First Steps

2.1.1 Online Tutorial

The online tutorial offers a quick start and guides you systematically in creating a simple simulation model. To start the tutorial, start Plant Simulation and left-click the tab **INFO PAGES**, then **EXAMPLES**, and **TUTORIAL** in the Explorer window.



2.1.2 Examples

The sample model includes a variety of examples of small models that are thematically ordered and show how and with which settings you can use the components and functions. To open the models after starting Plant Simulation, click on the tab **INFO PAGES**, then **EXAMPLES** and **EXAMPLES**.

2.1.3 Help

The step by step help provides descriptions of steps, which are necessary to model several tasks. The step by step help is part of the online help, chapter “Step-by-step help”.

The full functionality of version 9 is part of the online documentation. The manuals are available as Adobe Acrobat ® *. pdf files on the Plant Simulation installation CD and can be printed if required. The context-sensitive help in the dialogs of objects provides additional explanations of the dialog elements. To show context-sensitive help, click on the question mark on the top right corner of the dialog, and then click on the dialog element. The window of the context-sensitive help shows a reference to the corresponding SimTalk attribute at its end.

2.1.4 Website

Current information about Tecnomatix and Plant Simulation is available on the website <http://www.plm.automation.siemens.com>, or you use the direct link to the description of Plant Simulation:

http://www.plm.automation.siemens.com/en_us/products/tecnomatix/plant_design/plant_simulation.shtml.

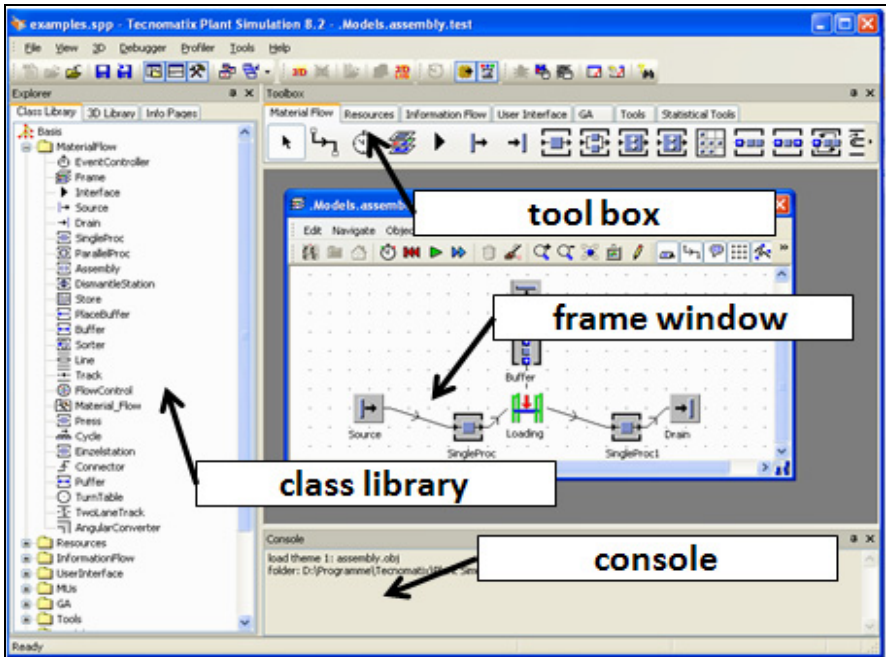
2.2 Introductory Example

2.2.1 The Program

Start Plant Simulation by clicking on the icon in the program group or the desktop icon.

2.2.1.1 The Program Window


To define the layout, you can use the menu item: **VIEW – TOOLBOX**. Here you set what you see on the screen. A standard Plant Simulation window can, for example, contain the following elements:



2.2.1.2 The Class Library

In the class library, you find all objects required for the simulation. You can create your own folders, derive and duplicate classes, create frames, or load objects from other simulation models.


To show the class library, you can use the command:

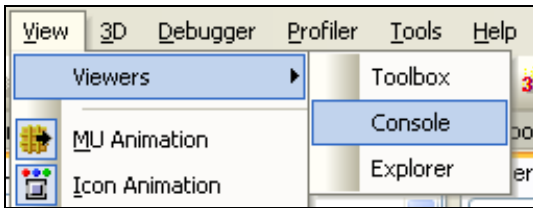
VIEW – VIEWERS – EXPLORER or the icon .

You can hide the class library by clicking the X in the title bar.


2.2.1.3 The Console

The Console provides information during the simulation (e.g., error messages). You can use the command Print to output messages to the console. If you do not need the console, you can hide it by clicking the X in the title bar.

Show the console with the icon:  in the toolbar or in the menu with **VIEW – VIEWERS – CONSOLE**:



2.2.1.4 The Toolbox

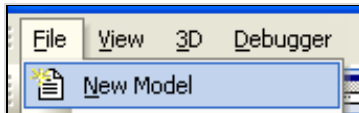
The Toolbox provides quick access to the classes in the class library. You can easily create your own tabs in the toolbox and fill it with your own objects. The best way to show the toolbox is a click on the button .

2.2.2 First Simulation Example

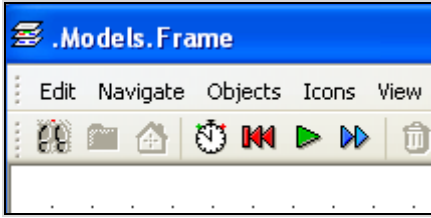
2.2.2.1 Design of the Model

As a first example, a simple production line is to be build with a source (material producer), two workstations, and a drain (material consumer). Start Plant Simulation, and select the menu command:

FILE – NEW MODEL



It opens the Plant Simulation class library with the basic objects and a Frame (window). Simulation models are created in the object Frame.

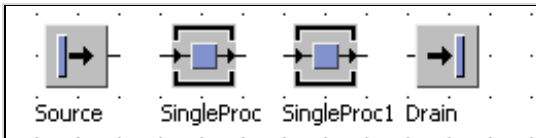


2.2.2.2 Insert Objects into the Frame

To insert objects into the Frame, you have two options:

- Click on the object icon in the toolbox, and then click in the Frame. The object will be inserted into the Frame at the position, at which you clicked.
- Another way: Drag the object from the class library to the Frame and drop it there (drag and drop).

Insert the following objects in the Frame:

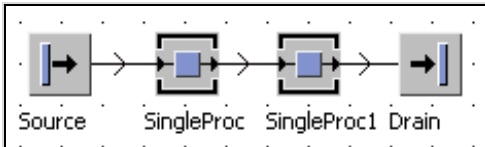


2.2.2.3 Connect the Objects

You have to connect the objects along the material flow, so that the different parts can be transported from one object to the next. This is what the object Connector does. The Connector has the following icon in the toolbar:



Click the Connector in the toolbar, then the object in the Frame, which you want to connect (the cursor changed its icon on Connector) – click the next object ... If you want to insert several Connectors successively, hold down the CTRL key. Result:

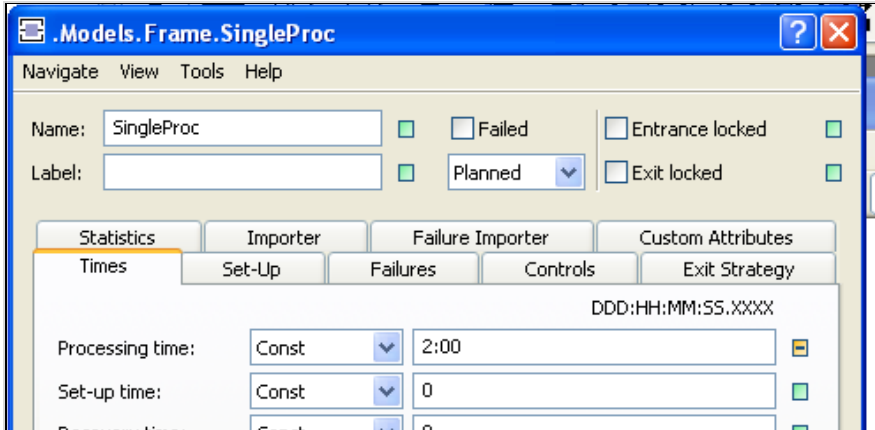


2.2.2.4 Define the Settings of the Objects

You have to define some settings in the objects such as processing times, capacity, information for setup, failures, breaks, etc. Properties can be easily set in the dialogs of the objects. You can open its dialog by double-clicking an object.

Example 1: Properties of the SingleProc

Set the following values: *SingleProc* stations: processing time: 2 minutes, *Drain*: Processing time: zero seconds, *Source*: 2-minute interval



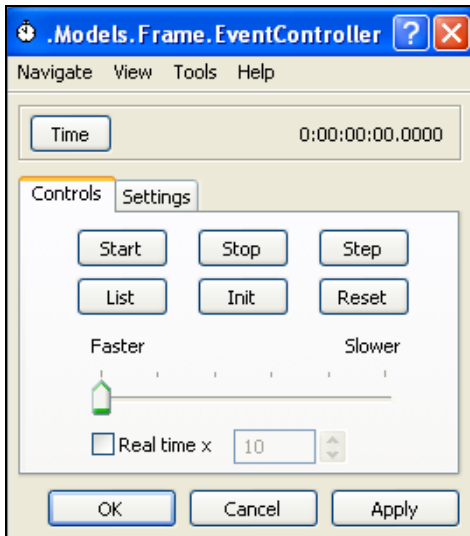
The **APPLY** button saves the values, but the dialog remains open. **OK** saves the values and closes the dialog. Finally, you need to insert an event controller. It coordinates the processes that run during a simulation.



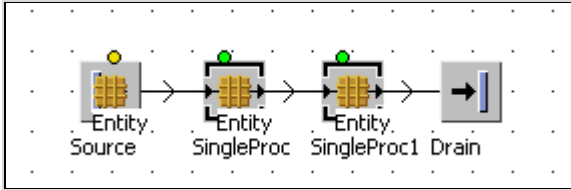
Click on the event controller in the toolbox, then in the Frame.

2.2.2.5 Run the Simulation

Open the control panel by double-clicking on the icon of the event controller.



Click the **START** button to start the simulation, and **STOP** to stop the simulation. In the Frame, the material movements are graphically displayed. You can now change the model to see what happens...

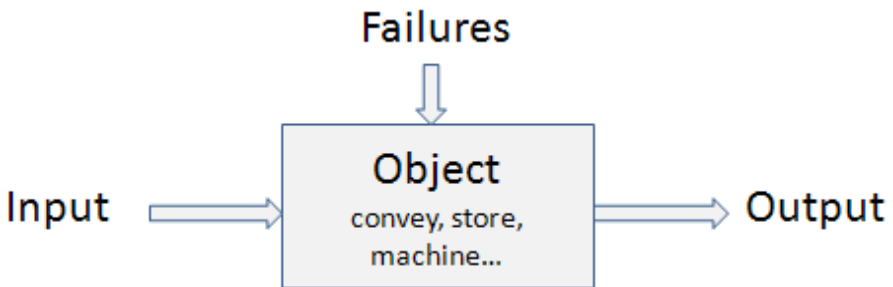


2.3 Modeling

2.3.1 Object-Related Modeling

In general, only a limited selection of objects is available for representing the real installation. They talk, e.g., of model objects, which show the real system with all the properties to be investigated. Hierarchically structured system models are best designed top-down. In this way, the real system will be decomposed into separate functional units (subsystems). If you are not able to model sufficiently precise with the available model objects, you should continue to decompose, etc.

Each object must be described precisely:



The individual objects and the operations within the objects are linked to an overall process. This creates a Frame. With the objects and the Frame various logistical systems can be modeled.

2.3.2 Object-Oriented Modeling

2.3.2.1 Objects and Properties

The hierarchical structure of an object allows to be exactly addressed (analogous to a file path). A robot Rob1 may be addressed in the hierarchy (the levels are separated by a period) as follows:

```
production1.press_hall.section1.cell1.Rob1
```

Rob1 itself is described by a number of properties such as: type of handling, speed, capacity, lead times, etc. All properties, which describe Rob1, are called “object”. An object is identified by its name (Rob1) and its path:

```
(production1.press_hall.section1.cell1.Rob1)
```

The properties are called attributes. They consist of a property description (attribute type), for example Engine type and a property value (attribute value) for instance: HANUK-ZsR1234578.

2.3.2.2 Classes and Instances

In object-oriented programming, a class is defined as follows:

A class is a user-defined data type. It designs a new data type to create a definition of a concept that has no direct counterpart in the fundamental data types.

Example: You want to create a new type of transport unit that cannot be defined by standard types. All definitions (properties, methods, behavior), required for creating a new type, are called a “class”.

The individual manifestation of the class is called an instance of the class (e.g., Transport – Forklift (general); Instance: Forklift 12/345 (concrete)). The instance has the same basic properties as the class and some special characteristics (such as a specific name).

2.3.2.3 Inheritance

In Plant Simulation, you can create a new class based on an existing class (derive of class, create a subclass). The original class is called base class. The derived class is called subclass. You can expand a data type through the derivation of a class without having to redefine it. You can use the basic objects of the class by employing inheritance.

Example: You have several machines of the same type; most of the properties are the same. Instead of defining each machine individually, you can define a basic machine. All other machines are derived from this basic machine. The subclasses inherited the properties of the base class they apply to these classes as if they were defined there.

2.3.2.4 Duplication and Derivation

Example 2: Inheritance 1

Select the *SingleProc* in the class library. Click the right mouse button to open the context menu. Select **DUPLICATE** from the context menu.

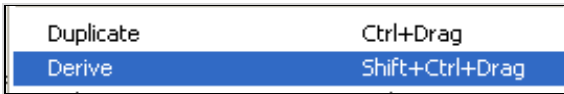


Plant Simulation names the duplicate SingleProc1. Change the processing time in the class SingleProc to 2 minutes. Open the dialog of the SingleProc1. The processing time has not changed.

The duplicate contains all the attributes of the original, but there is no connection between the original and the duplicate (there is no inheritance). You can also create duplicates using the mouse: Press the Control key and drag the object to its destination, then drop it.

Example 3: Inheritance 2

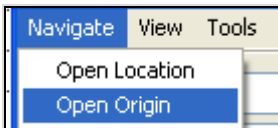
*Now do the same with **DERIVE**. Select the SingleProc again, click the right mouse button and select **DERIVE** from the context menu.*



*Plant Simulation names the new class SingleProc2. With **DERIVE** you created an instance of the class. This instance can either be a new class (in the library) or an object (e.g., in a Frame object). Initially, the instance inherits all the characteristics of the original class. Now, change the processing time of the SingleProc-class to 10 minutes (10:00). Save the changes in the SingleProc, and open the dialog of SingleProc2. SingleProc2 has applied the change of the processing time of the SingleProc-class.*

You can also derive in the class-library with CTRL + SHIFT and dragging the mouse. You can navigate to the original class from an object or from a derived class. Double-click the class/the object then select:

NAVIGATE – OPEN ORIGIN



It will open the dialog of the original class. If you drag a class from the class library into a Frame object, the new object is derived.

Example 4: Inheritance 3

Open a Frame object. Add a SingleProc to the Frame object (via drag and drop from the library). Change the processing time of the SingleProc in the library and check the processing time in the Frame. The values are inherited from the object in the library.