

ReGenESyS - Reborned GENeric and Expansible SYstem Simulator  
2019.0509

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>GenESyS-Reborn</b>	<b>1</b>
<b>2</b>	<b>Hierarchical Index</b>	<b>3</b>
2.1	Class Hierarchy . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>7</b>
3.1	Class List . . . . .	7
<b>4</b>	<b>File Index</b>	<b>11</b>
4.1	File List . . . . .	11
<b>5</b>	<b>Class Documentation</b>	<b>15</b>
5.1	Assign Class Reference . . . . .	15
5.1.1	Member Enumeration Documentation . . . . .	16
5.1.1.1	DestinationType . . . . .	16
5.1.2	Constructor & Destructor Documentation . . . . .	17
5.1.2.1	Assign(Model *model) . . . . .	17
5.1.2.2	Assign(const Assign &orig) . . . . .	17
5.1.2.3	~Assign() . . . . .	17
5.1.3	Member Function Documentation . . . . .	17
5.1.3.1	_execute(Entity *entity) . . . . .	17
5.1.3.2	_loadInstance(std::list< std::string > words) . . . . .	18
5.1.3.3	_saveInstance() . . . . .	19
5.1.3.4	_verifySymbols(std::string *errorMessage) . . . . .	19
5.1.3.5	getAssignments() const . . . . .	19

5.1.3.6	<code>show()</code>	20
5.2	<code>Assign::Assignment</code> Class Reference	20
5.2.1	Detailed Description	20
5.2.2	Constructor & Destructor Documentation	21
5.2.2.1	<code>Assignment(DestinationType destinationType, std::string destination, std::string expression)</code>	21
5.2.3	Member Function Documentation	21
5.2.3.1	<code>getDestination() const</code>	21
5.2.3.2	<code>getDestinationType() const</code>	21
5.2.3.3	<code>getExpression() const</code>	22
5.2.3.4	<code>setDestination(std::string _destination)</code>	22
5.2.3.5	<code>setDestinationType(DestinationType _destinationType)</code>	22
5.2.3.6	<code>setExpression(std::string _expression)</code>	22
5.3	<code>Attribute</code> Class Reference	23
5.3.1	Constructor & Destructor Documentation	24
5.3.1.1	<code>Attribute()</code>	24
5.3.1.2	<code>Attribute(std::string name)</code>	24
5.3.1.3	<code>Attribute(const Attribute &amp;orig)</code>	24
5.3.1.4	<code>~Attribute()</code>	24
5.3.2	Member Function Documentation	24
5.3.2.1	<code>_loadInstance(std::list&lt; std::string &gt; words)</code>	24
5.3.2.2	<code>_saveInstance()</code>	24
5.3.2.3	<code>_verifySymbols(std::string *errorMessage)</code>	24
5.3.2.4	<code>show()</code>	24
5.4	<code>BuildSimulationModel</code> Class Reference	25
5.4.1	Constructor & Destructor Documentation	25
5.4.1.1	<code>BuildSimulationModel()</code>	25
5.4.2	Member Function Documentation	25
5.4.2.1	<code>main(int argc, char **argv)</code>	25
5.5	<code>Collector_if</code> Class Reference	27
5.5.1	Detailed Description	27

5.5.2	Member Function Documentation	27
5.5.2.1	addValue(double value)=0	27
5.5.2.2	clear()=0	28
5.5.2.3	getLastValue()=0	28
5.5.2.4	numElements()=0	28
5.5.2.5	setAddValueHandler(CollectorAddValueHandler addValueHandler)=0	29
5.5.2.6	setClearHandler(CollectorClearHandler clearHandler)=0	29
5.6	CollectorDatafile_if Class Reference	30
5.6.1	Detailed Description	31
5.6.2	Member Function Documentation	31
5.6.2.1	getDataFilename()=0	31
5.6.2.2	getNextValue()=0	31
5.6.2.3	getValue(unsigned int rank)=0	32
5.6.2.4	seekFirstValue()=0	32
5.6.2.5	setDataFilename(std::string filename)=0	32
5.7	CollectorDatafileDefaultImpl1 Class Reference	32
5.7.1	Constructor & Destructor Documentation	33
5.7.1.1	CollectorDatafileDefaultImpl1()	33
5.7.1.2	CollectorDatafileDefaultImpl1(const CollectorDatafileDefaultImpl1 &orig)	33
5.7.1.3	~CollectorDatafileDefaultImpl1()	33
5.7.2	Member Function Documentation	33
5.7.2.1	addValue(double value)	33
5.7.2.2	clear()	34
5.7.2.3	getDataFilename()	34
5.7.2.4	getLastValue()	34
5.7.2.5	getNextValue()	34
5.7.2.6	getValue(unsigned int num)	34
5.7.2.7	numElements()	34
5.7.2.8	seekFirstValue()	34
5.7.2.9	setAddValueHandler(CollectorAddValueHandler addValueHandler)	34

5.7.2.10	<code>setClearHandler(CollectorClearHandler clearHandler)</code>	34
5.7.2.11	<code>setDataFilename(std::string filename)</code>	35
5.8	CollectorDatafileDummyImpl Class Reference	35
5.8.1	Constructor & Destructor Documentation	36
5.8.1.1	<code>CollectorDatafileDummyImpl()</code>	36
5.8.1.2	<code>CollectorDatafileDummyImpl(const CollectorDatafileDummyImpl &amp;orig)</code>	36
5.8.1.3	<code>~CollectorDatafileDummyImpl()</code>	36
5.8.2	Member Function Documentation	36
5.8.2.1	<code>addValue(double value)</code>	36
5.8.2.2	<code>clear()</code>	36
5.8.2.3	<code>getDataFilename()</code>	36
5.8.2.4	<code>getLastValue()</code>	36
5.8.2.5	<code>getNextValue()</code>	37
5.8.2.6	<code>getValue(unsigned int num)</code>	37
5.8.2.7	<code>numElements()</code>	37
5.8.2.8	<code>seekFirstValue()</code>	37
5.8.2.9	<code>setAddValueHandler(CollectorAddValueHandler addValueHandler)</code>	37
5.8.2.10	<code>setClearHandler(CollectorClearHandler clearHandler)</code>	37
5.8.2.11	<code>setDataFilename(std::string filename)</code>	37
5.9	CollectorDefaultImpl1 Class Reference	38
5.9.1	Constructor & Destructor Documentation	39
5.9.1.1	<code>CollectorDefaultImpl1()</code>	39
5.9.1.2	<code>CollectorDefaultImpl1(const CollectorDefaultImpl1 &amp;orig)</code>	39
5.9.1.3	<code>~CollectorDefaultImpl1()</code>	39
5.9.2	Member Function Documentation	39
5.9.2.1	<code>addValue(double value)</code>	39
5.9.2.2	<code>clear()</code>	39
5.9.2.3	<code>getLastValue()</code>	39
5.9.2.4	<code>numElements()</code>	39
5.9.2.5	<code>setAddValueHandler(CollectorAddValueHandler addValueHandler)</code>	39

5.9.2.6	<a href="#">setClearHandler(CollectorClearHandler clearHandler)</a>	39
5.10	<a href="#">CollectorDummyImpl Class Reference</a>	40
5.10.1	<a href="#">Constructor &amp; Destructor Documentation</a>	41
5.10.1.1	<a href="#">CollectorDummyImpl()</a>	41
5.10.1.2	<a href="#">CollectorDummyImpl(const CollectorDummyImpl &amp;orig)</a>	41
5.10.1.3	<a href="#">~CollectorDummyImpl()</a>	41
5.10.2	<a href="#">Member Function Documentation</a>	41
5.10.2.1	<a href="#">addValue(double value)</a>	41
5.10.2.2	<a href="#">clear()</a>	41
5.10.2.3	<a href="#">getLastValue()</a>	41
5.10.2.4	<a href="#">numElements()</a>	41
5.10.2.5	<a href="#">setAddValueHandler(CollectorAddValueHandler addValueHandler)</a>	41
5.10.2.6	<a href="#">setClearHandler(CollectorClearHandler clearHandler)</a>	41
5.11	<a href="#">Create Class Reference</a>	42
5.11.1	<a href="#">Detailed Description</a>	43
5.11.2	<a href="#">Constructor &amp; Destructor Documentation</a>	43
5.11.2.1	<a href="#">Create(Model *model)</a>	43
5.11.2.2	<a href="#">Create(const Create &amp;orig)</a>	44
5.11.2.3	<a href="#">~Create()</a>	44
5.11.3	<a href="#">Member Function Documentation</a>	44
5.11.3.1	<a href="#">_execute(Entity *entity)</a>	44
5.11.3.2	<a href="#">_loadInstance(std::list&lt; std::string &gt; words)</a>	44
5.11.3.3	<a href="#">_saveInstance()</a>	45
5.11.3.4	<a href="#">_verifySymbols(std::string *errorMessage)</a>	45
5.11.3.5	<a href="#">show()</a>	45
5.12	<a href="#">Decide Class Reference</a>	46
5.12.1	<a href="#">Constructor &amp; Destructor Documentation</a>	47
5.12.1.1	<a href="#">Decide(Model *model)</a>	47
5.12.1.2	<a href="#">Decide(const Decide &amp;orig)</a>	47
5.12.1.3	<a href="#">~Decide()</a>	47

5.12.2	Member Function Documentation	47
5.12.2.1	<code>_execute(Entity *entity)</code>	47
5.12.2.2	<code>_loadInstance(std::list&lt; std::string &gt; words)</code>	47
5.12.2.3	<code>_saveInstance()</code>	47
5.12.2.4	<code>_verifySymbols(std::string *errorMessage)</code>	48
5.12.2.5	<code>getConditions() const</code>	48
5.12.2.6	<code>show()</code>	48
5.13	SamplerDefaultImpl1::DefaultImpl1RNG_Parameters Class Reference	48
5.13.1	Member Data Documentation	49
5.13.1.1	<code>module</code>	49
5.13.1.2	<code>multiplier</code>	49
5.13.1.3	<code>seed</code>	49
5.14	Delay Class Reference	49
5.14.1	Constructor & Destructor Documentation	51
5.14.1.1	<code>Delay(Model *model)</code>	51
5.14.1.2	<code>Delay(const Delay &amp;orig)</code>	51
5.14.1.3	<code>~Delay()</code>	51
5.14.2	Member Function Documentation	51
5.14.2.1	<code>_execute(Entity *entity)</code>	51
5.14.2.2	<code>_loadInstance(std::list&lt; std::string &gt; words)</code>	52
5.14.2.3	<code>_saveInstance()</code>	52
5.14.2.4	<code>_verifySymbols(std::string *errorMessage)</code>	53
5.14.2.5	<code>getDelayExpression() const</code>	53
5.14.2.6	<code>getDelayTimeUnit() const</code>	53
5.14.2.7	<code>setDelayExpression(std::string _delayExpression)</code>	54
5.14.2.8	<code>setDelayTimeUnit(Util::TimeUnit _delayTimeUnit)</code>	54
5.14.2.9	<code>show()</code>	54
5.15	Dispose Class Reference	55
5.15.1	Constructor & Destructor Documentation	56
5.15.1.1	<code>Dispose(Model *model)</code>	56



5.15.1.2	<a href="#">Dispose(const Dispose &amp;orig)</a>	57
5.15.1.3	<a href="#">~Dispose()</a>	57
5.15.2	<a href="#">Member Function Documentation</a>	57
5.15.2.1	<a href="#">_execute(Entity *entity)</a>	57
5.15.2.2	<a href="#">_loadInstance(std::list&lt; std::string &gt; words)</a>	57
5.15.2.3	<a href="#">_saveInstance()</a>	57
5.15.2.4	<a href="#">_verifySymbols(std::string *errorMessage)</a>	57
5.15.2.5	<a href="#">getNumberOut() const</a>	58
5.15.2.6	<a href="#">isCollectStatistics() const</a>	58
5.15.2.7	<a href="#">setCollectStatistics(bool _collectStatistics)</a>	58
5.15.2.8	<a href="#">show()</a>	58
5.16	<a href="#">ElementManager Class Reference</a>	59
5.16.1	<a href="#">Detailed Description</a>	59
5.16.2	<a href="#">Constructor &amp; Destructor Documentation</a>	59
5.16.2.1	<a href="#">ElementManager(Model *model)</a>	59
5.16.2.2	<a href="#">ElementManager(const ElementManager &amp;orig)</a>	59
5.16.2.3	<a href="#">~ElementManager()</a>	59
5.16.3	<a href="#">Member Function Documentation</a>	59
5.16.3.1	<a href="#">getElement(std::string infraTypename, Util::identification id)</a>	59
5.16.3.2	<a href="#">getElement(std::string infraTypename, std::string name)</a>	60
5.16.3.3	<a href="#">getElements(std::string infraTypename) const</a>	60
5.16.3.4	<a href="#">getElementTypenames() const</a>	61
5.16.3.5	<a href="#">getNumberOfElements(std::string infraTypename)</a>	62
5.16.3.6	<a href="#">getRankOf(std::string infraTypename, std::string name)</a>	62
5.16.3.7	<a href="#">insertElement(std::string infraTypename, ModelElement *infra)</a>	63
5.16.3.8	<a href="#">removeElement(std::string infraTypename, ModelElement *infra)</a>	63
5.16.3.9	<a href="#">show()</a>	64
5.17	<a href="#">ElementManager_if Class Reference</a>	64
5.17.1	<a href="#">Constructor &amp; Destructor Documentation</a>	65
5.17.1.1	<a href="#">ElementManager_if()</a>	65

5.17.1.2	<code>ElementManager_if(const ElementManager_if &amp;orig)</code>	65
5.17.1.3	<code>~ElementManager_if()</code>	65
5.18	Entity Class Reference	65
5.18.1	Constructor & Destructor Documentation	66
5.18.1.1	<code>Entity(ElementManager *elements)</code>	66
5.18.1.2	<code>Entity(const Entity &amp;orig)</code>	66
5.18.1.3	<code>~Entity()</code>	66
5.18.2	Member Function Documentation	66
5.18.2.1	<code>_loadInstance(std::list&lt; std::string &gt; words)</code>	66
5.18.2.2	<code>_saveInstance()</code>	67
5.18.2.3	<code>_verifySymbols(std::string *errorMessage)</code>	67
5.18.2.4	<code>getAttributeValue(std::string attributeName)</code>	67
5.18.2.5	<code>getEntityType() const</code>	68
5.18.2.6	<code>getEntityTypeName() const</code>	68
5.18.2.7	<code>setAttributeValue(std::string attributeName, double value)</code>	68
5.18.2.8	<code>setEntityType(EntityType *entityType)</code>	69
5.18.2.9	<code>setEntityTypeName(std::string entityTypeName)</code>	69
5.18.2.10	<code>show()</code>	69
5.19	EntityType Class Reference	70
5.19.1	Constructor & Destructor Documentation	71
5.19.1.1	<code>EntityType(ElementManager *elemManager)</code>	71
5.19.1.2	<code>EntityType(ElementManager *elemManager, std::string name)</code>	71
5.19.1.3	<code>EntityType(const EntityType &amp;orig)</code>	72
5.19.1.4	<code>~EntityType()</code>	72
5.19.2	Member Function Documentation	72
5.19.2.1	<code>_loadInstance(std::list&lt; std::string &gt; words)</code>	72
5.19.2.2	<code>_saveInstance()</code>	72
5.19.2.3	<code>_verifySymbols(std::string *errorMessage)</code>	72
5.19.2.4	<code>getCstatNVATime() const</code>	72
5.19.2.5	<code>getCstatOtherTime() const</code>	72

5.19.2.6	<a href="#">getCstatTotalTime() const</a>	72
5.19.2.7	<a href="#">getCstatTransferTime() const</a>	73
5.19.2.8	<a href="#">getCstatVATime() const</a>	73
5.19.2.9	<a href="#">getCstatWaitingTime() const</a>	73
5.19.2.10	<a href="#">getInitialNVACost() const</a>	73
5.19.2.11	<a href="#">getInitialOtherCost() const</a>	73
5.19.2.12	<a href="#">getInitialPicture() const</a>	73
5.19.2.13	<a href="#">getInitialVACost() const</a>	73
5.19.2.14	<a href="#">getInitialWaitingCost() const</a>	73
5.19.2.15	<a href="#">setInitialNVACost(double _initialNVACost)</a>	73
5.19.2.16	<a href="#">setInitialOtherCost(double _initialOtherCost)</a>	73
5.19.2.17	<a href="#">setInitialPicture(std::string _initialPicture)</a>	73
5.19.2.18	<a href="#">setInitialVACost(double _initialVACost)</a>	73
5.19.2.19	<a href="#">setInitialWaitingCost(double _initialWaitingCost)</a>	73
5.19.2.20	<a href="#">show()</a>	73
5.20	<a href="#">Event Class Reference</a>	74
5.20.1	<a href="#">Constructor &amp; Destructor Documentation</a>	74
5.20.1.1	<a href="#">Event(double time, Entity *entity, ModelComponent *component)</a>	74
5.20.1.2	<a href="#">Event(const Event &amp;orig)</a>	74
5.20.1.3	<a href="#">~Event()</a>	74
5.20.2	<a href="#">Member Function Documentation</a>	74
5.20.2.1	<a href="#">getComponent() const</a>	74
5.20.2.2	<a href="#">getEntity() const</a>	74
5.20.2.3	<a href="#">getTime() const</a>	74
5.20.2.4	<a href="#">show()</a>	75
5.21	<a href="#">ExperimentDesign_if Class Reference</a>	75
5.21.1	<a href="#">Detailed Description</a>	76
5.21.2	<a href="#">Member Function Documentation</a>	76
5.21.2.1	<a href="#">calculateContributionAndCoefficients()=0</a>	76
5.21.2.2	<a href="#">generate2krScenarioExperiments()=0</a>	76

5.21.2.3	<code>getContributions() const =0</code>	76
5.21.2.4	<code>getProcessAnalyser() const =0</code>	76
5.22	ExperimentDesignDummyImpl Class Reference	77
5.22.1	Constructor & Destructor Documentation	78
5.22.1.1	<code>ExperimentDesignDummyImpl()</code>	78
5.22.1.2	<code>ExperimentDesignDummyImpl(const ExperimentDesignDummyImpl &amp;orig)</code>	78
5.22.1.3	<code>~ExperimentDesignDummyImpl()</code>	78
5.22.2	Member Function Documentation	78
5.22.2.1	<code>calculateContributionAndCoefficients()</code>	78
5.22.2.2	<code>generate2krScenarioExperiments()</code>	78
5.22.2.3	<code>getContributions() const</code>	78
5.22.2.4	<code>getProcessAnalyser() const</code>	78
5.23	FactorOrInteractionContribution Class Reference	78
5.23.1	Detailed Description	79
5.23.2	Constructor & Destructor Documentation	79
5.23.2.1	<code>FactorOrInteractionContribution(double contribution, double modelCoefficient, std::list&lt; SimulationControl * &gt; *controls)</code>	79
5.23.2.2	<code>FactorOrInteractionContribution(const FactorOrInteractionContribution &amp;orig)</code>	79
5.23.2.3	<code>~FactorOrInteractionContribution()</code>	79
5.23.3	Member Function Documentation	79
5.23.3.1	<code>getContribution() const</code>	79
5.23.3.2	<code>getControls() const</code>	79
5.23.3.3	<code>getModelCoefficient() const</code>	79
5.24	Fitter_if Class Reference	79
5.24.1	Member Function Documentation	80
5.24.1.1	<code>fitAll(double *sqerror, std::string *name)=0</code>	80
5.24.1.2	<code>fitBeta(double *sqerror, double *alpha, double *beta, double *infLimit, double *supLimit)=0</code>	80
5.24.1.3	<code>fitErlang(double *sqerror, double *avg, double *m)=0</code>	80
5.24.1.4	<code>fitExpo(double *sqerror, double *avg1)=0</code>	80
5.24.1.5	<code>fitNormal(double *sqerror, double *avg, double *stddev)=0</code>	81

5.24.1.6	<code>fitTriangular(double *sqerror, double *min, double *mo, double *max)=0</code>	81
5.24.1.7	<code>fitUniform(double *sqerror, double *min, double *max)=0</code>	81
5.24.1.8	<code>fitWeibull(double *sqerror, double *alpha, double *scale)=0</code>	81
5.24.1.9	<code>getDataFilename()</code>	82
5.24.1.10	<code>isNormalDistributed(double confidencelevel)=0</code>	82
5.24.1.11	<code>setDataFilename(std::string dataFilename)=0</code>	82
5.25	<b>FitterDummyImpl Class Reference</b>	82
5.25.1	<b>Constructor &amp; Destructor Documentation</b>	83
5.25.1.1	<code>FitterDummyImpl()</code>	83
5.25.1.2	<code>FitterDummyImpl(const FitterDummyImpl &amp;orig)</code>	83
5.25.1.3	<code>~FitterDummyImpl()</code>	83
5.25.2	<b>Member Function Documentation</b>	83
5.25.2.1	<code>fitAll(double *sqerror, std::string *name)</code>	83
5.25.2.2	<code>fitBeta(double *sqerror, double *alpha, double *beta, double *infLimit, double *supLimit)</code>	84
5.25.2.3	<code>fitErlang(double *sqerror, double *avg, double *m)</code>	84
5.25.2.4	<code>fitExpo(double *sqerror, double *avg1)</code>	84
5.25.2.5	<code>fitNormal(double *sqerror, double *avg, double *stddev)</code>	84
5.25.2.6	<code>fitTriangular(double *sqerror, double *min, double *mo, double *max)</code>	84
5.25.2.7	<code>fitUniform(double *sqerror, double *min, double *max)</code>	84
5.25.2.8	<code>fitWeibull(double *sqerror, double *alpha, double *scale)</code>	84
5.25.2.9	<code>getDataFilename()</code>	84
5.25.2.10	<code>isNormalDistributed(double confidencelevel)</code>	84
5.25.2.11	<code>setDataFilename(std::string dataFilename)</code>	84
5.26	<b>GenesysApplication_if Class Reference</b>	85
5.26.1	<b>Member Function Documentation</b>	85
5.26.1.1	<code>main(int argc, char **argv)=0</code>	85
5.27	<b>HypothesisTester_if Class Reference</b>	86
5.27.1	<b>Detailed Description</b>	86
5.27.2	<b>Member Enumeration Documentation</b>	86
5.27.2.1	<b>H1Comparition</b>	86

5.27.3	Member Function Documentation	87
5.27.3.1	getDataFilename() <sub>0</sub>	87
5.27.3.2	setDataFilename(std::string dataFilename) <sub>0</sub>	87
5.27.3.3	testAverage(double confidencelevel, double avg, H1Comparition comp) <sub>0</sub>	87
5.27.3.4	testAverage(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp) <sub>0</sub>	87
5.27.3.5	testProportion(double confidencelevel, double prop, H1Comparition comp) <sub>0</sub>	87
5.27.3.6	testProportion(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp) <sub>0</sub>	88
5.27.3.7	testVariance(double confidencelevel, double var, H1Comparition comp) <sub>0</sub>	88
5.27.3.8	testVariance(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp) <sub>0</sub>	88
5.28	HypothesisTesterDummyImpl Class Reference	88
5.28.1	Constructor & Destructor Documentation	89
5.28.1.1	HypothesisTesterDummyImpl()	89
5.28.1.2	HypothesisTesterDummyImpl(const HypothesisTesterDummyImpl &orig)	89
5.28.1.3	~HypothesisTesterDummyImpl()	89
5.28.2	Member Function Documentation	89
5.28.2.1	getDataFilename()	89
5.28.2.2	setDataFilename(std::string dataFilename)	90
5.28.2.3	testAverage(double confidencelevel, double avg, H1Comparition comp)	90
5.28.2.4	testAverage(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)	90
5.28.2.5	testProportion(double confidencelevel, double prop, H1Comparition comp)	90
5.28.2.6	testProportion(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)	90
5.28.2.7	testVariance(double confidencelevel, double var, H1Comparition comp)	90
5.28.2.8	testVariance(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)	90
5.29	Integrator_if Class Reference	91
5.29.1	Detailed Description	91
5.29.2	Member Function Documentation	91
5.29.2.1	getPrecision() <sub>0</sub>	91

5.29.2.2	<a href="#">integrate(double min, double max, double(*f)(double, double), double p2)=0</a>	92
5.29.2.3	<a href="#">integrate(double min, double max, double(*f)(double, double, double), double p2, double p3)=0</a>	92
5.29.2.4	<a href="#">integrate(double min, double max, double(*f)(double, double, double, double), double p2, double p3, double p4)=0</a>	92
5.29.2.5	<a href="#">integrate(double min, double max, double(*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)=0</a>	92
5.29.2.6	<a href="#">setPrecision(double e)=0</a>	92
5.30	<a href="#">IntegratorDefaultImpl1 Class Reference</a>	93
5.30.1	<a href="#">Constructor &amp; Destructor Documentation</a>	94
5.30.1.1	<a href="#">IntegratorDefaultImpl1()</a>	94
5.30.1.2	<a href="#">IntegratorDefaultImpl1(const IntegratorDefaultImpl1 &amp;orig)</a>	94
5.30.1.3	<a href="#">~IntegratorDefaultImpl1()</a>	94
5.30.2	<a href="#">Member Function Documentation</a>	94
5.30.2.1	<a href="#">getPrecision()</a>	94
5.30.2.2	<a href="#">integrate(double min, double max, double(*f)(double, double), double p2)</a>	94
5.30.2.3	<a href="#">integrate(double min, double max, double(*f)(double, double, double), double p2, double p3)</a>	94
5.30.2.4	<a href="#">integrate(double min, double max, double(*f)(double, double, double, double), double p2, double p3, double p4)</a>	94
5.30.2.5	<a href="#">integrate(double min, double max, double(*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)</a>	94
5.30.2.6	<a href="#">setPrecision(double e)</a>	94
5.31	<a href="#">IntegratorDummyImpl Class Reference</a>	95
5.31.1	<a href="#">Constructor &amp; Destructor Documentation</a>	96
5.31.1.1	<a href="#">IntegratorDummyImpl()</a>	96
5.31.1.2	<a href="#">IntegratorDummyImpl(const IntegratorDummyImpl &amp;orig)</a>	96
5.31.1.3	<a href="#">~IntegratorDummyImpl()</a>	96
5.31.2	<a href="#">Member Function Documentation</a>	96
5.31.2.1	<a href="#">getPrecision()</a>	96
5.31.2.2	<a href="#">integrate(double min, double max, double(*f)(double, double), double p2)</a>	96
5.31.2.3	<a href="#">integrate(double min, double max, double(*f)(double, double, double), double p2, double p3)</a>	96

5.31.2.4	integrate(double min, double max, double(*f)(double, double, double, double), double p2, double p3, double p4)	96
5.31.2.5	integrate(double min, double max, double(*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)	96
5.31.2.6	setPrecision(double e)	96
5.32	LinkBy Class Reference	97
5.32.1	Constructor & Destructor Documentation	97
5.32.1.1	LinkBy()	97
5.32.1.2	LinkBy(const LinkBy &orig)	97
5.32.1.3	~LinkBy()	97
5.32.2	Member Function Documentation	97
5.32.2.1	addLink()	97
5.32.2.2	isLinked()	97
5.32.2.3	removeLink()	97
5.33	List< T > Class Template Reference	98
5.33.1	Detailed Description	98
5.33.2	Member Typedef Documentation	99
5.33.2.1	CompFunc	99
5.33.3	Constructor & Destructor Documentation	99
5.33.3.1	List()	99
5.33.3.2	List(const List &orig)	99
5.33.3.3	~List()	99
5.33.4	Member Function Documentation	99
5.33.4.1	actual()	99
5.33.4.2	clear()	99
5.33.4.3	create()	99
5.33.4.4	create(U arg)	99
5.33.4.5	empty()	99
5.33.4.6	find(T element)	100
5.33.4.7	first()	100
5.33.4.8	getAtRank(unsigned int rank)	100



5.33.4.9	<code>getList() const</code>	101
5.33.4.10	<code>insert(T element)</code>	102
5.33.4.11	<code>last()</code>	102
5.33.4.12	<code>next()</code>	102
5.33.4.13	<code>pop_front()</code>	103
5.33.4.14	<code>previous()</code>	103
5.33.4.15	<code>remove(T element)</code>	103
5.33.4.16	<code>setAtRank(unsigned int rank, T element)</code>	103
5.33.4.17	<code>setSortFunc(CompFunct _sortFunc)</code>	104
5.33.4.18	<code>show()</code>	104
5.33.4.19	<code>size()</code>	105
5.33.4.20	<code>sort(Compare comp)</code>	105
5.34	Model Class Reference	105
5.34.1	Detailed Description	106
5.34.2	Constructor & Destructor Documentation	106
5.34.2.1	<code>Model(Simulator *simulator)</code>	106
5.34.2.2	<code>Model(const Model &amp;orig)</code>	107
5.34.2.3	<code>~Model()</code>	107
5.34.3	Member Function Documentation	107
5.34.3.1	<code>checkModel()</code>	107
5.34.3.2	<code>getComponents() const</code>	108
5.34.3.3	<code>getControls() const</code>	109
5.34.3.4	<code>getElementManager() const</code>	109
5.34.3.5	<code>getEvents() const</code>	110
5.34.3.6	<code>getId() const</code>	110
5.34.3.7	<code>getInfos() const</code>	110
5.34.3.8	<code>getEventManager() const</code>	110
5.34.3.9	<code>getParent() const</code>	111
5.34.3.10	<code>getResponses() const</code>	111
5.34.3.11	<code>getSimulation() const</code>	111

5.34.3.12	<code>getTracer() const</code>	112
5.34.3.13	<code>loadModel(std::string filename)</code>	113
5.34.3.14	<code>parseExpression(const std::string expression)</code>	113
5.34.3.15	<code>parseExpression(const std::string expression, bool *success, std::string *error← Message)</code>	114
5.34.3.16	<code>removeEntity(Entity *entity, bool collectStatistics)</code>	114
5.34.3.17	<code>saveModel(std::string filename)</code>	115
5.34.3.18	<code>sendEntityToComponent(Entity *entity, ModelComponent *component, double timeDelay)</code>	115
5.34.3.19	<code>showReports()</code>	116
5.34.3.20	<code>verifySymbol(std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory)</code>	116
5.35	<code>ModelChecker_if</code> Class Reference	117
5.35.1	Detailed Description	117
5.35.2	Member Function Documentation	117
5.35.2.1	<code>checkActivationCode()=0</code>	117
5.35.2.2	<code>checkAll()=0</code>	118
5.35.2.3	<code>checkAndAddInternalLiterals()=0</code>	118
5.35.2.4	<code>checkConnected()=0</code>	118
5.35.2.5	<code>checkPathway()=0</code>	118
5.35.2.6	<code>checkSymbols()=0</code>	118
5.35.2.7	<code>verifySymbol(std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory)=0</code>	118
5.36	<code>ModelCheckerDefaultImpl1</code> Class Reference	119
5.36.1	Constructor & Destructor Documentation	120
5.36.1.1	<code>ModelCheckerDefaultImpl1(Model *model)</code>	120
5.36.1.2	<code>ModelCheckerDefaultImpl1(const ModelCheckerDefaultImpl1 &amp;orig)</code>	120
5.36.1.3	<code>~ModelCheckerDefaultImpl1()</code>	120
5.36.2	Member Function Documentation	120
5.36.2.1	<code>checkActivationCode()</code>	120
5.36.2.2	<code>checkAll()</code>	120
5.36.2.3	<code>checkAndAddInternalLiterals()</code>	120

5.36.2.4	<a href="#">checkConnected()</a>	121
5.36.2.5	<a href="#">checkPathway()</a>	121
5.36.2.6	<a href="#">checkSymbols()</a>	121
5.36.2.7	<a href="#">verifySymbol(std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory)</a>	122
5.37	<a href="#">ModelCheckerDummyImpl Class Reference</a>	123
5.37.1	<a href="#">Detailed Description</a>	124
5.37.2	<a href="#">Constructor &amp; Destructor Documentation</a>	124
5.37.2.1	<a href="#">ModelCheckerDummyImpl(Model *model)</a>	124
5.37.2.2	<a href="#">ModelCheckerDummyImpl(const ModelCheckerDummyImpl &amp;orig)</a>	124
5.37.2.3	<a href="#">~ModelCheckerDummyImpl()</a>	124
5.37.3	<a href="#">Member Function Documentation</a>	124
5.37.3.1	<a href="#">checkActivationCode()</a>	124
5.37.3.2	<a href="#">checkAll()</a>	125
5.37.3.3	<a href="#">checkAndAddInternalLiterals()</a>	125
5.37.3.4	<a href="#">checkConnected()</a>	126
5.37.3.5	<a href="#">checkPathway()</a>	126
5.37.3.6	<a href="#">checkSymbols()</a>	126
5.37.3.7	<a href="#">verifySymbol(std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory)</a>	127
5.38	<a href="#">ModelComponent Class Reference</a>	127
5.38.1	<a href="#">Detailed Description</a>	129
5.38.2	<a href="#">Constructor &amp; Destructor Documentation</a>	129
5.38.2.1	<a href="#">ModelComponent(Model *model, std::string componentName)</a>	129
5.38.2.2	<a href="#">ModelComponent(const ModelComponent &amp;orig)</a>	129
5.38.2.3	<a href="#">~ModelComponent()</a>	129
5.38.3	<a href="#">Member Function Documentation</a>	129
5.38.3.1	<a href="#">_execute(Entity *entity)=0</a>	129
5.38.3.2	<a href="#">_saveInstance()</a>	130
5.38.3.3	<a href="#">_saveInstance(std::string type)</a>	130
5.38.3.4	<a href="#">Execute(Entity *entity, ModelComponent *component)</a>	131

5.38.3.5	<a href="#">getNextComponents() const</a>	131
5.38.3.6	<a href="#">SaveInstance(ModelComponent *component)</a>	132
5.38.3.7	<a href="#">show()</a>	132
5.38.3.8	<a href="#">VerifySymbols(ModelComponent *component, std::string *errorMessage)</a>	133
5.38.4	<a href="#">Member Data Documentation</a>	133
5.38.4.1	<a href="#">_model</a>	133
5.39	<a href="#">ModelComponentManager_if Class Reference</a>	134
5.39.1	<a href="#">Constructor &amp; Destructor Documentation</a>	134
5.39.1.1	<a href="#">ModelComponentManager_if()</a>	134
5.39.1.2	<a href="#">ModelComponentManager_if(const ModelComponentManager_if &amp;orig)</a>	134
5.39.1.3	<a href="#">~ModelComponentManager_if()</a>	134
5.40	<a href="#">ModelElement Class Reference</a>	134
5.40.1	<a href="#">Detailed Description</a>	135
5.40.2	<a href="#">Constructor &amp; Destructor Documentation</a>	135
5.40.2.1	<a href="#">ModelElement(std::string elementTypename)</a>	135
5.40.2.2	<a href="#">ModelElement(const ModelElement &amp;orig)</a>	135
5.40.2.3	<a href="#">~ModelElement()</a>	135
5.40.3	<a href="#">Member Function Documentation</a>	136
5.40.3.1	<a href="#">_loadInstance(std::list&lt; std::string &gt; words)=0</a>	136
5.40.3.2	<a href="#">_saveInstance()</a>	136
5.40.3.3	<a href="#">_saveInstance(std::string type)</a>	136
5.40.3.4	<a href="#">_verifySymbols(std::string *errorMessage)=0</a>	137
5.40.3.5	<a href="#">getId() const</a>	137
5.40.3.6	<a href="#">getName() const</a>	138
5.40.3.7	<a href="#">getTypename() const</a>	138
5.40.3.8	<a href="#">LoadInstance(std::list&lt; std::string &gt; words)</a>	138
5.40.3.9	<a href="#">SaveInstance(ModelElement *element)</a>	138
5.40.3.10	<a href="#">setName(std::string _name)</a>	139
5.40.3.11	<a href="#">show()</a>	139
5.40.3.12	<a href="#">VerifySymbols(ModelElement *element, std::string *errorMessage)</a>	139

5.40.4	Member Data Documentation . . . . .	140
5.40.4.1	_id . . . . .	140
5.40.4.2	_name . . . . .	140
5.40.4.3	_typename . . . . .	140
5.41	ModelInfo Class Reference . . . . .	140
5.41.1	Detailed Description . . . . .	141
5.41.2	Constructor & Destructor Documentation . . . . .	141
5.41.2.1	ModelInfo() . . . . .	141
5.41.2.2	ModelInfo(const ModelInfo &orig) . . . . .	141
5.41.2.3	~ModelInfo() . . . . .	141
5.41.3	Member Function Documentation . . . . .	141
5.41.3.1	getAnalystName() const . . . . .	141
5.41.3.2	getDescription() const . . . . .	141
5.41.3.3	getName() const . . . . .	141
5.41.3.4	getNumberOfReplications() const . . . . .	141
5.41.3.5	getProjectTitle() const . . . . .	142
5.41.3.6	getReplicationLength() const . . . . .	142
5.41.3.7	getReplicationLengthTimeUnit() const . . . . .	142
5.41.3.8	getTerminatingCondition() const . . . . .	142
5.41.3.9	getVersion() const . . . . .	143
5.41.3.10	getWarmUpPeriod() const . . . . .	143
5.41.3.11	getWarmUpPeriodTimeUnit() const . . . . .	143
5.41.3.12	setAnalystName(std::string _analystName) . . . . .	143
5.41.3.13	setDescription(std::string _description) . . . . .	143
5.41.3.14	setName(std::string _name) . . . . .	143
5.41.3.15	setNumberOfReplications(unsigned int _numberOfReplications) . . . . .	143
5.41.3.16	setProjectTitle(std::string _projectTitle) . . . . .	144
5.41.3.17	setReplicationLength(double _replicationLength) . . . . .	144
5.41.3.18	setReplicationLengthTimeUnit(Util::TimeUnit _replicationLengthTimeUnit) . . . . .	144
5.41.3.19	setTerminatingCondition(std::string _terminatingCondition) . . . . .	144

5.41.3.20	setVersion(std::string _version)	144
5.41.3.21	setWarmUpPeriod(double _warmUpPeriod)	144
5.41.3.22	setWarmUpPeriodTimeUnit(Util::TimeUnit _warmUpPeriodTimeUnit)	145
5.42	ModelPersistence_if Class Reference	145
5.42.1	Detailed Description	145
5.42.2	Member Function Documentation	145
5.42.2.1	isSaved()=0	145
5.42.2.2	load(std::string filename)=0	146
5.42.2.3	loadAsTXT(std::string filename)=0	146
5.42.2.4	loadAsXML(std::string filename)=0	146
5.42.2.5	save(std::string filename)=0	146
5.42.2.6	saveAsTXT(std::string filename)=0	146
5.42.2.7	saveAsXML(std::string filename)=0	147
5.43	ModelPersistenceDummyImpl Class Reference	147
5.43.1	Constructor & Destructor Documentation	148
5.43.1.1	ModelPersistenceDummyImpl(Model *model)	148
5.43.1.2	ModelPersistenceDummyImpl(const ModelPersistenceDummyImpl &orig)	148
5.43.1.3	~ModelPersistenceDummyImpl()	148
5.43.2	Member Function Documentation	148
5.43.2.1	isSaved()	148
5.43.2.2	load(std::string filename)	148
5.43.2.3	loadAsTXT(std::string filename)	148
5.43.2.4	loadAsXML(std::string filename)	149
5.43.2.5	save(std::string filename)	149
5.43.2.6	saveAsTXT(std::string filename)	149
5.43.2.7	saveAsXML(std::string filename)	150
5.44	ModelSimulation Class Reference	150
5.44.1	Detailed Description	151
5.44.2	Constructor & Destructor Documentation	151
5.44.2.1	ModelSimulation(Model *model)	151

5.44.2.2	<a href="#">ModelSimulation(const ModelSimulation &amp;orig)</a>	152
5.44.2.3	<a href="#">~ModelSimulation()</a>	152
5.44.3	<a href="#">Member Function Documentation</a>	152
5.44.3.1	<a href="#">getCurrentComponent() const</a>	152
5.44.3.2	<a href="#">getCurrentEntity() const</a>	152
5.44.3.3	<a href="#">getCurrentReplicationNumber() const</a>	152
5.44.3.4	<a href="#">getSimulatedTime() const</a>	153
5.44.3.5	<a href="#">isInitializeStatistics() const</a>	153
5.44.3.6	<a href="#">isInitializeSystem() const</a>	153
5.44.3.7	<a href="#">isPauseOnEvent() const</a>	153
5.44.3.8	<a href="#">isPauseOnReplication() const</a>	153
5.44.3.9	<a href="#">isRunning() const</a>	153
5.44.3.10	<a href="#">isStepByStep() const</a>	153
5.44.3.11	<a href="#">pauseSimulation()</a>	153
5.44.3.12	<a href="#">restartSimulation()</a>	153
5.44.3.13	<a href="#">setInitializeStatistics(bool _initializeStatistics)</a>	153
5.44.3.14	<a href="#">setInitializeSystem(bool _initializeSystem)</a>	153
5.44.3.15	<a href="#">setPauseOnEvent(bool _pauseOnEvent)</a>	154
5.44.3.16	<a href="#">setPauseOnReplication(bool _pauseBetweenReplications)</a>	154
5.44.3.17	<a href="#">setStepByStep(bool _stepByStep)</a>	154
5.44.3.18	<a href="#">startSimulation()</a>	154
5.44.3.19	<a href="#">stepSimulation()</a>	156
5.44.3.20	<a href="#">stopSimulation()</a>	156
5.45	<a href="#">SamplerDummyImpl::MyRNG_Parameters Class Reference</a>	156
5.45.1	<a href="#">Member Data Documentation</a>	157
5.45.1.1	<a href="#">module</a>	157
5.45.1.2	<a href="#">multiplier</a>	157
5.45.1.3	<a href="#">seed</a>	157
5.46	<a href="#">OnEventManager Class Reference</a>	157
5.46.1	<a href="#">Detailed Description</a>	158

5.46.2	Constructor & Destructor Documentation . . . . .	158
5.46.2.1	OnEventManager() . . . . .	158
5.46.2.2	OnEventManager(const OnEventManager &orig) . . . . .	158
5.46.2.3	~OnEventManager() . . . . .	158
5.46.3	Member Function Documentation . . . . .	158
5.46.3.1	addOnEntityRemoveHandler(simulationEventHandler EventHandler) . . . . .	158
5.46.3.2	addOnProcessEventHandler(simulationEventHandler EventHandler) . . . . .	158
5.46.3.3	addOnReplicationEndHandler(simulationEventHandler EventHandler) . . . . .	159
5.46.3.4	addOnReplicationStartHandler(simulationEventHandler EventHandler) . . . . .	159
5.46.3.5	addOnReplicationStepHandler(simulationEventHandler EventHandler) . . . . .	159
5.46.3.6	addOnSimulationEndHandler(simulationEventHandler EventHandler) . . . . .	159
5.46.3.7	addOnSimulationStartHandler(simulationEventHandler EventHandler) . . . . .	159
5.46.3.8	NotifyProcessEventHandlers(SimulationEvent *se) . . . . .	159
5.46.3.9	NotifyReplicationEndHandlers(SimulationEvent *se) . . . . .	160
5.46.3.10	NotifyReplicationStartHandlers(SimulationEvent *se) . . . . .	160
5.46.3.11	NotifyReplicationStepHandlers(SimulationEvent *se) . . . . .	160
5.46.3.12	NotifySimulationEndHandlers(SimulationEvent *se) . . . . .	160
5.46.3.13	NotifySimulationStartHandlers(SimulationEvent *se) . . . . .	161
5.47	Parser_if Class Reference . . . . .	161
5.47.1	Member Function Documentation . . . . .	161
5.47.1.1	getErrorMessage()=0 . . . . .	161
5.47.1.2	parse(const std::string expression)=0 . . . . .	162
5.47.1.3	parse(const std::string expression, bool *success, std::string *errorMessage)=0 . . . . .	162
5.48	ParserDefaultImpl1 Class Reference . . . . .	162
5.48.1	Constructor & Destructor Documentation . . . . .	163
5.48.1.1	ParserDefaultImpl1(Model *model) . . . . .	163
5.48.1.2	ParserDefaultImpl1(const ParserDefaultImpl1 &orig) . . . . .	163
5.48.1.3	~ParserDefaultImpl1() . . . . .	163
5.48.2	Member Function Documentation . . . . .	163
5.48.2.1	getErrorMessage() . . . . .	163



5.48.2.2	<a href="#">parse(const std::string expression)</a>	163
5.48.2.3	<a href="#">parse(const std::string expression, bool *success, std::string *errorMessage)</a>	164
5.49	<a href="#">ParserDummyImpl Class Reference</a>	164
5.49.1	<a href="#">Constructor &amp; Destructor Documentation</a>	165
5.49.1.1	<a href="#">ParserDummyImpl(Model *model)</a>	165
5.49.1.2	<a href="#">ParserDummyImpl(const ParserDummyImpl &amp;orig)</a>	165
5.49.1.3	<a href="#">~ParserDummyImpl()</a>	165
5.49.2	<a href="#">Member Function Documentation</a>	165
5.49.2.1	<a href="#">getErrorMessage()</a>	165
5.49.2.2	<a href="#">parse(const std::string expression)</a>	165
5.49.2.3	<a href="#">parse(const std::string expression, bool *success, std::string *errorMessage)</a>	166
5.50	<a href="#">Plugin Class Reference</a>	166
5.50.1	<a href="#">Detailed Description</a>	166
5.50.2	<a href="#">Constructor &amp; Destructor Documentation</a>	167
5.50.2.1	<a href="#">Plugin(std::string name, bool source, bool drain)</a>	167
5.50.2.2	<a href="#">Plugin(const Plugin &amp;orig)</a>	167
5.50.2.3	<a href="#">~Plugin()</a>	167
5.50.3	<a href="#">Member Function Documentation</a>	167
5.50.3.1	<a href="#">isDrain() const</a>	167
5.50.3.2	<a href="#">isSource() const</a>	167
5.51	<a href="#">ProbDistrib Class Reference</a>	167
5.51.1	<a href="#">Member Function Documentation</a>	168
5.51.1.1	<a href="#">beta(double x, double alpha, double beta)</a>	168
5.51.1.2	<a href="#">erlang(double x, double mean, double M)</a>	168
5.51.1.3	<a href="#">exponential(double x, double mean)</a>	168
5.51.1.4	<a href="#">gamma(double x, double mean, double alpha)</a>	168
5.51.1.5	<a href="#">logNormal(double x, double mean, double stddev)</a>	168
5.51.1.6	<a href="#">normal(double x, double mean, double stddev)</a>	168
5.51.1.7	<a href="#">triangular(double x, double min, double mode, double max)</a>	168
5.51.1.8	<a href="#">uniform(double x, double min, double max)</a>	168

5.51.1.9	<code>weibull(double x, double alpha, double scale)</code>	168
5.52	ProcessAnalyser_if Class Reference	169
5.52.1	Detailed Description	169
5.52.2	Member Function Documentation	169
5.52.2.1	<code>addTraceSimulationHandler(traceSimulationProcessListener traceSimulation↔ ProcessListener)=0</code>	169
5.52.2.2	<code>extractControlsFromModel(std::string modelName) const =0</code>	169
5.52.2.3	<code>extractResponsesFromModel(std::string modelName) const =0</code>	170
5.52.2.4	<code>getControls() const =0</code>	170
5.52.2.5	<code>getResponses() const =0</code>	170
5.52.2.6	<code>getScenarios() const =0</code>	170
5.52.2.7	<code>startSimulation()=0</code>	170
5.52.2.8	<code>startSimulationOfScenario(SimulationScenario *scenario)=0</code>	170
5.52.2.9	<code>stopSimulation()=0</code>	170
5.53	ProcessAnalyserDummyImpl Class Reference	171
5.53.1	Constructor & Destructor Documentation	172
5.53.1.1	<code>ProcessAnalyserDummyImpl()</code>	172
5.53.1.2	<code>ProcessAnalyserDummyImpl(const ProcessAnalyserDummyImpl &amp;orig)</code>	172
5.53.1.3	<code>~ProcessAnalyserDummyImpl()</code>	172
5.53.2	Member Function Documentation	172
5.53.2.1	<code>addTraceSimulationHandler(traceSimulationProcessListener traceSimulation↔ ProcessListener)</code>	172
5.53.2.2	<code>extractControlsFromModel(std::string modelName) const</code>	172
5.53.2.3	<code>extractResponsesFromModel(std::string modelName) const</code>	172
5.53.2.4	<code>getControls() const</code>	172
5.53.2.5	<code>getResponses() const</code>	172
5.53.2.6	<code>getScenarios() const</code>	172
5.53.2.7	<code>startSimulation()</code>	172
5.53.2.8	<code>startSimulationOfScenario(SimulationScenario *scenario)</code>	173
5.53.2.9	<code>stopSimulation()</code>	173
5.54	Queue Class Reference	173

5.54.1	Member Enumeration Documentation	174
5.54.1.1	OrderRule	174
5.54.2	Constructor & Destructor Documentation	175
5.54.2.1	Queue(ElementManager *elems)	175
5.54.2.2	Queue(ElementManager *elems, std::string name)	175
5.54.2.3	Queue(const Queue &orig)	175
5.54.2.4	~Queue()	175
5.54.3	Member Function Documentation	175
5.54.3.1	_loadInstance(std::list< std::string > words)	175
5.54.3.2	_saveInstance()	176
5.54.3.3	_verifySymbols(std::string *errorMessage)	176
5.54.3.4	first()	176
5.54.3.5	getAttributeName() const	177
5.54.3.6	getOrderRule() const	177
5.54.3.7	insertElement(Waiting *element)	177
5.54.3.8	removeElement(Waiting *element, double tnow)	178
5.54.3.9	setAttributeName(std::string _attributeName)	178
5.54.3.10	setOrderRule(OrderRule _orderRule)	178
5.54.3.11	show()	179
5.54.3.12	size()	179
5.55	Record Class Reference	180
5.55.1	Constructor & Destructor Documentation	181
5.55.1.1	Record(Model *model)	181
5.55.1.2	Record(const Record &orig)	181
5.55.1.3	~Record()	181
5.55.2	Member Function Documentation	182
5.55.2.1	_execute(Entity *entity)	182
5.55.2.2	_loadInstance(std::list< std::string > words)	182
5.55.2.3	_saveInstance()	182
5.55.2.4	_verifySymbols(std::string *errorMessage)	182

5.55.2.5	<a href="#">getCstatExpression() const</a>	182
5.55.2.6	<a href="#">getExpression() const</a>	182
5.55.2.7	<a href="#">getExpressionName() const</a>	182
5.55.2.8	<a href="#">getFilename() const</a>	182
5.55.2.9	<a href="#">setExpression(std::string expression)</a>	183
5.55.2.10	<a href="#">setExpressionName(std::string expressionName)</a>	183
5.55.2.11	<a href="#">setFilename(std::string filename)</a>	183
5.55.2.12	<a href="#">show()</a>	184
5.56	<a href="#">Release Class Reference</a>	184
5.56.1	<a href="#">Constructor &amp; Destructor Documentation</a>	186
5.56.1.1	<a href="#">Release(Model *model)</a>	186
5.56.1.2	<a href="#">Release(const Release &amp;orig)</a>	186
5.56.1.3	<a href="#">~Release()</a>	186
5.56.2	<a href="#">Member Function Documentation</a>	186
5.56.2.1	<a href="#">_execute(Entity *entity)</a>	186
5.56.2.2	<a href="#">_loadInstance(std::list&lt; std::string &gt; words)</a>	186
5.56.2.3	<a href="#">_saveInstance()</a>	186
5.56.2.4	<a href="#">_verifySymbols(std::string *errorMessage)</a>	187
5.56.2.5	<a href="#">getPriority() const</a>	187
5.56.2.6	<a href="#">getQuantity() const</a>	187
5.56.2.7	<a href="#">getResource() const</a>	187
5.56.2.8	<a href="#">getResourceName() const</a>	187
5.56.2.9	<a href="#">getResourceType() const</a>	187
5.56.2.10	<a href="#">getRule() const</a>	187
5.56.2.11	<a href="#">getSaveAttribute() const</a>	187
5.56.2.12	<a href="#">setPriority(unsigned short _priority)</a>	187
5.56.2.13	<a href="#">setQuantity(std::string _quantity)</a>	187
5.56.2.14	<a href="#">setResource(Resource *_resource)</a>	187
5.56.2.15	<a href="#">setResourceName(std::string resourceName)</a>	188
5.56.2.16	<a href="#">setResourceType(Resource::ResourceType _resourceType)</a>	188

5.56.2.17	<a href="#">setRule(Resource::ResourceRule _rule)</a>	188
5.56.2.18	<a href="#">setSaveAttribute(std::string _saveAttribute)</a>	188
5.56.2.19	<a href="#">show()</a>	188
5.57	<a href="#">Resource Class Reference</a>	189
5.57.1	<a href="#">Member Typedef Documentation</a>	190
5.57.1.1	<a href="#">ResourceEventHandler</a>	190
5.57.2	<a href="#">Member Enumeration Documentation</a>	190
5.57.2.1	<a href="#">ResourceRule</a>	190
5.57.2.2	<a href="#">ResourceState</a>	191
5.57.2.3	<a href="#">ResourceType</a>	191
5.57.3	<a href="#">Constructor &amp; Destructor Documentation</a>	191
5.57.3.1	<a href="#">Resource(ElementManager *elems)</a>	191
5.57.3.2	<a href="#">Resource(ElementManager *elems, std::string name)</a>	191
5.57.3.3	<a href="#">Resource(const Resource &amp;orig)</a>	191
5.57.3.4	<a href="#">~Resource()</a>	191
5.57.4	<a href="#">Member Function Documentation</a>	192
5.57.4.1	<a href="#">_loadInstance(std::list&lt; std::string &gt; words)</a>	192
5.57.4.2	<a href="#">_saveInstance()</a>	192
5.57.4.3	<a href="#">_verifySymbols(std::string *errorMessage)</a>	192
5.57.4.4	<a href="#">addResourceEventHandler(ResourceEventHandler eventHandler)</a>	192
5.57.4.5	<a href="#">getCapacity() const</a>	193
5.57.4.6	<a href="#">getCostBusyHour() const</a>	193
5.57.4.7	<a href="#">getCostIdleHour() const</a>	193
5.57.4.8	<a href="#">getCostPerUse() const</a>	193
5.57.4.9	<a href="#">getNumberBusy() const</a>	193
5.57.4.10	<a href="#">getNumberOut() const</a>	193
5.57.4.11	<a href="#">getResourceState() const</a>	193
5.57.4.12	<a href="#">release(unsigned int quantity, double tnow)</a>	194
5.57.4.13	<a href="#">seize(unsigned int quantity, double tnow)</a>	194
5.57.4.14	<a href="#">setCapacity(unsigned int _capacity)</a>	195

5.57.4.15	setCostBusyHour(double _costBusyHour)	195
5.57.4.16	setCostIdleHour(double _costIdleHour)	195
5.57.4.17	setCostPerUse(double _costPerUse)	195
5.57.4.18	SetResourceEventHandler(void(Class::*function)(Resource *), Class *object)	195
5.57.4.19	setResourceState(ResourceState _resourceState)	195
5.57.4.20	show()	195
5.58	Sampler_if::RNG_Parameters Class Reference	196
5.58.1	Detailed Description	196
5.59	Sampler_if Class Reference	196
5.59.1	Detailed Description	197
5.59.2	Member Function Documentation	197
5.59.2.1	getRNGparameters() const =0	197
5.59.2.2	random()=0	197
5.59.2.3	sampleBeta(double alpha, double beta, double infLimit, double supLimit)=0	197
5.59.2.4	sampleDiscrete(double value, double acumProb,...)=0	198
5.59.2.5	sampleErlang(double mean, int M)=0	198
5.59.2.6	sampleExponential(double mean)=0	198
5.59.2.7	sampleGamma(double mean, double alpha)=0	198
5.59.2.8	sampleLogNormal(double mean, double stddev)=0	198
5.59.2.9	sampleNormal(double mean, double stddev)=0	198
5.59.2.10	sampleTriangular(double min, double mode, double max)=0	199
5.59.2.11	sampleUniform(double min, double max)=0	199
5.59.2.12	sampleWeibull(double alpha, double scale)=0	199
5.59.2.13	setRNGparameters(RNG_Parameters *param)=0	199
5.60	SamplerDefaultImpl1 Class Reference	200
5.60.1	Constructor & Destructor Documentation	201
5.60.1.1	SamplerDefaultImpl1()	201
5.60.1.2	SamplerDefaultImpl1(const SamplerDefaultImpl1 &orig)	201
5.60.1.3	~SamplerDefaultImpl1()	201
5.60.2	Member Function Documentation	201

5.60.2.1	<a href="#">getRNGparameters() const</a>	201
5.60.2.2	<a href="#">random()</a>	201
5.60.2.3	<a href="#">reset()</a>	202
5.60.2.4	<a href="#">sampleBeta(double alpha, double beta, double infLimit, double supLimit)</a>	202
5.60.2.5	<a href="#">sampleDiscrete(double value, double acumProb,...)</a>	202
5.60.2.6	<a href="#">sampleErlang(double mean, int M)</a>	202
5.60.2.7	<a href="#">sampleExponential(double mean)</a>	203
5.60.2.8	<a href="#">sampleGamma(double mean, double alpha)</a>	203
5.60.2.9	<a href="#">sampleLogNormal(double mean, double stddev)</a>	203
5.60.2.10	<a href="#">sampleNormal(double mean, double stddev)</a>	203
5.60.2.11	<a href="#">sampleTriangular(double min, double mode, double max)</a>	203
5.60.2.12	<a href="#">sampleUniform(double min, double max)</a>	204
5.60.2.13	<a href="#">sampleWeibull(double alpha, double scale)</a>	204
5.60.2.14	<a href="#">setRNGparameters(RNG_Parameters *param)</a>	204
5.61	<a href="#">SamplerDummyImpl Class Reference</a>	204
5.61.1	<a href="#">Constructor &amp; Destructor Documentation</a>	205
5.61.1.1	<a href="#">SamplerDummyImpl()</a>	205
5.61.1.2	<a href="#">SamplerDummyImpl(const SamplerDummyImpl &amp;orig)</a>	205
5.61.1.3	<a href="#">~SamplerDummyImpl()</a>	205
5.61.2	<a href="#">Member Function Documentation</a>	205
5.61.2.1	<a href="#">getRNGparameters() const</a>	205
5.61.2.2	<a href="#">random()</a>	206
5.61.2.3	<a href="#">sampleBeta(double alpha, double beta, double infLimit, double supLimit)</a>	206
5.61.2.4	<a href="#">sampleDiscrete(double value, double acumProb,...)</a>	206
5.61.2.5	<a href="#">sampleErlang(double mean, int M)</a>	206
5.61.2.6	<a href="#">sampleExponential(double mean)</a>	206
5.61.2.7	<a href="#">sampleGamma(double mean, double alpha)</a>	206
5.61.2.8	<a href="#">sampleLogNormal(double mean, double stddev)</a>	206
5.61.2.9	<a href="#">sampleNormal(double mean, double stddev)</a>	206
5.61.2.10	<a href="#">sampleTriangular(double min, double mode, double max)</a>	206

5.61.2.11 sampleUniform(double min, double max) . . . . .	206
5.61.2.12 sampleWeibull(double alpha, double scale) . . . . .	207
5.61.2.13 setRNGparameters(RNG_Parameters *param) . . . . .	207
5.62 ScenarioExperiment_if Class Reference . . . . .	207
5.63 Seize Class Reference . . . . .	207
5.63.1 Detailed Description . . . . .	209
5.63.2 Constructor & Destructor Documentation . . . . .	209
5.63.2.1 Seize(Model *model) . . . . .	209
5.63.2.2 Seize(const Seize &orig) . . . . .	209
5.63.2.3 ~Seize() . . . . .	209
5.63.3 Member Function Documentation . . . . .	209
5.63.3.1 _execute(Entity *entity) . . . . .	209
5.63.3.2 _loadInstance(std::list< std::string > words) . . . . .	210
5.63.3.3 _saveInstance() . . . . .	210
5.63.3.4 _verifySymbols(std::string *errorMessage) . . . . .	210
5.63.3.5 getAllocationType() const . . . . .	210
5.63.3.6 getLastMemberSeized() const . . . . .	210
5.63.3.7 getPriority() const . . . . .	210
5.63.3.8 getQuantity() const . . . . .	210
5.63.3.9 getQueue() const . . . . .	210
5.63.3.10 getQueueName() const . . . . .	210
5.63.3.11 getResource() const . . . . .	211
5.63.3.12 getResourceName() const . . . . .	211
5.63.3.13 getResourceType() const . . . . .	211
5.63.3.14 getRule() const . . . . .	211
5.63.3.15 getSaveAttribute() const . . . . .	211
5.63.3.16 setAllocationType(unsigned int _allocationType) . . . . .	211
5.63.3.17 setLastMemberSeized(unsigned int _lastMemberSeized) . . . . .	211
5.63.3.18 setPriority(unsigned short _priority) . . . . .	211
5.63.3.19 setQuantity(std::string _quantity) . . . . .	211



5.63.3.20	setQueue(Queue *queue)	211
5.63.3.21	setQueueName(std::string queueName)	212
5.63.3.22	setResource(Resource *resource)	212
5.63.3.23	setResourceName(std::string _resourceName)	213
5.63.3.24	setResourceType(Resource::ResourceType _resourceType)	213
5.63.3.25	setRule(Resource::ResourceRule _rule)	213
5.63.3.26	setSaveAttribute(std::string _saveAttribute)	213
5.63.3.27	show()	213
5.64	SimulationControl Class Reference	214
5.64.1	Detailed Description	214
5.64.2	Constructor & Destructor Documentation	215
5.64.2.1	SimulationControl(std::string type, std::string name, GetterMember getter↔ Member, SetterMember setterMember)	215
5.64.2.2	SimulationControl(const SimulationControl &orig)	215
5.64.2.3	~SimulationControl()	215
5.64.3	Member Function Documentation	215
5.64.3.1	setValue(double value)	215
5.65	SimulationEvent Class Reference	215
5.65.1	Constructor & Destructor Documentation	215
5.65.1.1	SimulationEvent(unsigned int replicationNumber, Event *event)	215
5.65.2	Member Function Documentation	215
5.65.2.1	getEventProcessed() const	215
5.65.2.2	getReplicationNumber() const	216
5.66	SimulationReporter_if Class Reference	216
5.66.1	Member Function Documentation	216
5.66.1.1	showReplicationStatistics()=0	216
5.66.1.2	showSimulationStatistics()=0	217
5.67	SimulationReporterDefaultImpl1 Class Reference	217
5.67.1	Constructor & Destructor Documentation	218
5.67.1.1	SimulationReporterDefaultImpl1(ModelSimulation *simulation, Model *model)	218
5.67.1.2	SimulationReporterDefaultImpl1(const SimulationReporterDefaultImpl1 &orig)	218

5.67.1.3	<a href="#">~SimulationReporterDefaultImpl1()</a>	218
5.67.2	Member Function Documentation	218
5.67.2.1	<a href="#">showReplicationStatistics()</a>	218
5.67.2.2	<a href="#">showSimulationStatistics()</a>	219
5.68	SimulationResponse Class Reference	220
5.68.1	Detailed Description	221
5.68.2	Constructor & Destructor Documentation	221
5.68.2.1	<a href="#">SimulationResponse(std::string type, std::string name, GetterMember getter↔Member)</a>	221
5.68.2.2	<a href="#">SimulationResponse(const SimulationResponse &amp;orig)</a>	221
5.68.2.3	<a href="#">~SimulationResponse()</a>	221
5.68.3	Member Function Documentation	221
5.68.3.1	<a href="#">getName() const</a>	221
5.68.3.2	<a href="#">getType() const</a>	221
5.68.3.3	<a href="#">getValue()</a>	221
5.68.4	Member Data Documentation	221
5.68.4.1	<a href="#">_getterMemberFunction</a>	221
5.68.4.2	<a href="#">_name</a>	221
5.68.4.3	<a href="#">_type</a>	221
5.69	SimulationScenario Class Reference	221
5.69.1	Detailed Description	222
5.69.2	Constructor & Destructor Documentation	222
5.69.2.1	<a href="#">SimulationScenario()</a>	222
5.69.2.2	<a href="#">SimulationScenario(const SimulationScenario &amp;orig)</a>	222
5.69.2.3	<a href="#">~SimulationScenario()</a>	222
5.69.3	Member Function Documentation	222
5.69.3.1	<a href="#">getControlValue(SimulationControl *control)</a>	222
5.69.3.2	<a href="#">getControlValues() const</a>	222
5.69.3.3	<a href="#">getModelFilename() const</a>	222
5.69.3.4	<a href="#">getName() const</a>	222
5.69.3.5	<a href="#">getResponseValue(SimulationResponse *value)</a>	222

5.69.3.6	<a href="#">getResponseValues() const</a>	222
5.69.3.7	<a href="#">setControlValue(SimulationControl *control, double value)</a>	222
5.69.3.8	<a href="#">setModelFilename(std::string _modelName)</a>	222
5.69.3.9	<a href="#">setName(std::string _name)</a>	222
5.70	<a href="#">Simulator Class Reference</a>	223
5.70.1	<a href="#">Detailed Description</a>	223
5.70.2	<a href="#">Constructor &amp; Destructor Documentation</a>	223
5.70.2.1	<a href="#">Simulator()</a>	223
5.70.2.2	<a href="#">Simulator(const Simulator &amp;orig)</a>	223
5.70.2.3	<a href="#">~Simulator()</a>	223
5.70.3	<a href="#">Member Function Documentation</a>	223
5.70.3.1	<a href="#">getFitter() const</a>	223
5.70.3.2	<a href="#">getLicense() const</a>	224
5.70.3.3	<a href="#">getModels() const</a>	224
5.70.3.4	<a href="#">getName() const</a>	224
5.70.3.5	<a href="#">getPlugins() const</a>	224
5.70.3.6	<a href="#">getSampler() const</a>	224
5.70.3.7	<a href="#">getVersion() const</a>	225
5.71	<a href="#">SinkModelComponent Class Reference</a>	225
5.71.1	<a href="#">Detailed Description</a>	226
5.71.2	<a href="#">Constructor &amp; Destructor Documentation</a>	226
5.71.2.1	<a href="#">SinkModelComponent(Model *model, std::string componentName)</a>	226
5.71.2.2	<a href="#">SinkModelComponent(const SinkModelComponent &amp;orig)</a>	226
5.71.2.3	<a href="#">~SinkModelComponent()</a>	226
5.71.3	<a href="#">Member Function Documentation</a>	226
5.71.3.1	<a href="#">isCollectStatistics() const</a>	226
5.71.3.2	<a href="#">setCollectStatistics(bool _collectStatistics)</a>	226
5.72	<a href="#">SourceModelComponent Class Reference</a>	227
5.72.1	<a href="#">Detailed Description</a>	228
5.72.2	<a href="#">Constructor &amp; Destructor Documentation</a>	229

5.72.2.1	SourceModelComponent(Model *model, std::string componentTypename)	229
5.72.2.2	SourceModelComponent(const SourceModelComponent &orig)	229
5.72.2.3	~SourceModelComponent()	229
5.72.3	Member Function Documentation	229
5.72.3.1	getEntitiesCreated() const	229
5.72.3.2	getEntitiesPerCreation() const	229
5.72.3.3	getEntityType() const	229
5.72.3.4	getFirstCreation() const	229
5.72.3.5	getMaxCreations() const	230
5.72.3.6	getTimeBetweenCreationsExpression() const	230
5.72.3.7	getTimeUnit() const	230
5.72.3.8	isCollectStatistics() const	230
5.72.3.9	setCollectStatistics(bool _collectStatistics)	230
5.72.3.10	setEntitiesCreated(unsigned int _entitiesCreated)	230
5.72.3.11	setEntitiesPerCreation(unsigned int _entitiesPerCreation)	230
5.72.3.12	setEntityType(EntityType *_entityType)	230
5.72.3.13	setFirstCreation(double _firstCreation)	230
5.72.3.14	setMaxCreations(unsigned int _maxCreations)	230
5.72.3.15	setTimeBetweenCreationsExpression(std::string _timeBetweenCreations)	230
5.72.3.16	setTimeUnit(Util::TimeUnit _timeUnit)	231
5.72.3.17	show()	231
5.72.4	Member Data Documentation	231
5.72.4.1	_collectStatistics	231
5.72.4.2	_entitiesCreatedSoFar	231
5.72.4.3	_entitiesPerCreation	231
5.72.4.4	_entityType	232
5.72.4.5	_firstCreation	232
5.72.4.6	_maxCreations	232
5.72.4.7	_timeBetweenCreationsExpression	232
5.72.4.8	_timeBetweenCreationsTimeUnit	232

5.73 Statistics_if Class Reference . . . . .	232
5.73.1 Detailed Description . . . . .	233
5.73.2 Member Function Documentation . . . . .	233
5.73.2.1 average()=0 . . . . .	233
5.73.2.2 getCollector()=0 . . . . .	233
5.73.2.3 halfWidthConfidenceInterval(double confidencelevel)=0 . . . . .	234
5.73.2.4 max()=0 . . . . .	234
5.73.2.5 min()=0 . . . . .	234
5.73.2.6 newSampleSize(double confidencelevel, double halfWidth)=0 . . . . .	235
5.73.2.7 numElements()=0 . . . . .	235
5.73.2.8 setCollector(Collector_if *collector)=0 . . . . .	235
5.73.2.9 stddeviation()=0 . . . . .	235
5.73.2.10 variance()=0 . . . . .	236
5.73.2.11 variationCoef()=0 . . . . .	236
5.74 StatisticsCollector Class Reference . . . . .	236
5.74.1 Constructor & Destructor Documentation . . . . .	238
5.74.1.1 StatisticsCollector() . . . . .	238
5.74.1.2 StatisticsCollector(std::string name) . . . . .	238
5.74.1.3 StatisticsCollector(std::string name, ModelElement *parent) . . . . .	238
5.74.1.4 StatisticsCollector(const StatisticsCollector &orig) . . . . .	238
5.74.1.5 ~StatisticsCollector() . . . . .	238
5.74.2 Member Function Documentation . . . . .	238
5.74.2.1 _loadInstance(std::list< std::string > words) . . . . .	238
5.74.2.2 _saveInstance() . . . . .	238
5.74.2.3 _verifySymbols(std::string *errorMessage) . . . . .	239
5.74.2.4 getParent() const . . . . .	239
5.74.2.5 getStatistics() const . . . . .	239
5.74.2.6 show() . . . . .	239
5.75 StatisticsDatafile_if Class Reference . . . . .	240
5.75.1 Member Function Documentation . . . . .	241

5.75.1.1	centil(unsigned short num)=0	241
5.75.1.2	decil(unsigned short num)=0	241
5.75.1.3	histogramClassFrequency(unsigned short classNum)=0	241
5.75.1.4	histogramClassLowerLimit(unsigned short classNum)=0	241
5.75.1.5	histogramNumClasses()=0	242
5.75.1.6	mediane()=0	242
5.75.1.7	mode()=0	242
5.75.1.8	quartil(unsigned short num)=0	242
5.75.1.9	setHistogramNumClasses(unsigned short num)=0	242
5.76	StatisticsDataFileDummyImpl Class Reference	242
5.76.1	Constructor & Destructor Documentation	243
5.76.1.1	StatisticsDataFileDummyImpl()	243
5.76.1.2	StatisticsDataFileDummyImpl(const StatisticsDataFileDummyImpl &orig)	243
5.76.1.3	~StatisticsDataFileDummyImpl()	244
5.76.2	Member Function Documentation	244
5.76.2.1	average()	244
5.76.2.2	centil(unsigned short num)	244
5.76.2.3	decil(unsigned short num)	244
5.76.2.4	getCollector()	244
5.76.2.5	halfWidthConfidenceInterval(double confidencelevel)	244
5.76.2.6	histogramClassFrequency(unsigned short classNum)	244
5.76.2.7	histogramClassLowerLimit(unsigned short classNum)	244
5.76.2.8	histogramNumClasses()	244
5.76.2.9	max()	244
5.76.2.10	mediane()	245
5.76.2.11	min()	245
5.76.2.12	mode()	245
5.76.2.13	newSampleSize(double confidencelevel, double halfWidth)	245
5.76.2.14	numElements()	245
5.76.2.15	quartil(unsigned short num)	245

5.76.2.16	setCollector(Collector_if *collector)	245
5.76.2.17	setHistogramNumClasses(unsigned short num)	245
5.76.2.18	stddeviation()	245
5.76.2.19	variance()	245
5.76.2.20	variationCoef()	246
5.77	StatisticsDefaultImpl1 Class Reference	246
5.77.1	Constructor & Destructor Documentation	247
5.77.1.1	StatisticsDefaultImpl1()	247
5.77.1.2	StatisticsDefaultImpl1(Collector_if *collector)	248
5.77.1.3	StatisticsDefaultImpl1(const StatisticsDefaultImpl1 &orig)	248
5.77.1.4	~StatisticsDefaultImpl1()	248
5.77.2	Member Function Documentation	248
5.77.2.1	average()	248
5.77.2.2	getCollector()	249
5.77.2.3	halfWidthConfidenceInterval(double confidencelevel)	249
5.77.2.4	max()	249
5.77.2.5	min()	249
5.77.2.6	newSampleSize(double confidencelevel, double halfWidth)	249
5.77.2.7	numElements()	249
5.77.2.8	setCollector(Collector_if *collector)	250
5.77.2.9	stddeviation()	250
5.77.2.10	variance()	250
5.77.2.11	variationCoef()	250
5.78	StatisticsDummyImpl Class Reference	250
5.78.1	Constructor & Destructor Documentation	251
5.78.1.1	StatisticsDummyImpl()	252
5.78.1.2	StatisticsDummyImpl(const StatisticsDummyImpl &orig)	252
5.78.1.3	~StatisticsDummyImpl()	252
5.78.2	Member Function Documentation	252
5.78.2.1	average()	252

5.78.2.2	<a href="#">getCollector()</a> . . . . .	252
5.78.2.3	<a href="#">halfWidthConfidenceInterval(double confidencelevel)</a> . . . . .	252
5.78.2.4	<a href="#">max()</a> . . . . .	252
5.78.2.5	<a href="#">min()</a> . . . . .	253
5.78.2.6	<a href="#">newSampleSize(double confidencelevel, double halfWidth)</a> . . . . .	253
5.78.2.7	<a href="#">numElements()</a> . . . . .	253
5.78.2.8	<a href="#">setCollector(Collector_if *collector)</a> . . . . .	253
5.78.2.9	<a href="#">stddeviation()</a> . . . . .	253
5.78.2.10	<a href="#">variance()</a> . . . . .	253
5.78.2.11	<a href="#">variationCoef()</a> . . . . .	253
5.79	<a href="#">TestInputAnalyserTools Class Reference</a> . . . . .	254
5.79.1	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	254
5.79.1.1	<a href="#">TestInputAnalyserTools()</a> . . . . .	254
5.79.2	<a href="#">Member Function Documentation</a> . . . . .	254
5.79.2.1	<a href="#">main(int argc, char **argv)</a> . . . . .	254
5.80	<a href="#">TestParser Class Reference</a> . . . . .	256
5.80.1	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	256
5.80.1.1	<a href="#">TestParser()</a> . . . . .	256
5.80.1.2	<a href="#">TestParser(const TestParser &amp;orig)</a> . . . . .	256
5.80.1.3	<a href="#">~TestParser()</a> . . . . .	257
5.80.2	<a href="#">Member Function Documentation</a> . . . . .	257
5.80.2.1	<a href="#">main(int argc, char **argv)</a> . . . . .	257
5.81	<a href="#">TestStatistics Class Reference</a> . . . . .	257
5.81.1	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	258
5.81.1.1	<a href="#">TestStatistics()</a> . . . . .	258
5.81.2	<a href="#">Member Function Documentation</a> . . . . .	258
5.81.2.1	<a href="#">main(int argc, char **argv)</a> . . . . .	258
5.82	<a href="#">TraceErrorEvent Class Reference</a> . . . . .	259
5.82.1	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	260
5.82.1.1	<a href="#">TraceErrorEvent(std::string text, std::exception e)</a> . . . . .	260



5.82.2	Member Function Documentation	260
5.82.2.1	getException() const	260
5.83	TraceEvent Class Reference	261
5.83.1	Constructor & Destructor Documentation	261
5.83.1.1	TraceEvent(Util::TraceLevel tracelevel, std::string text)	261
5.83.2	Member Function Documentation	261
5.83.2.1	getText() const	261
5.83.2.2	getTracelevel() const	261
5.84	TraceManager Class Reference	262
5.84.1	Detailed Description	262
5.84.2	Constructor & Destructor Documentation	262
5.84.2.1	TraceManager(Model *model)	262
5.84.2.2	TraceManager(const TraceManager &orig)	262
5.84.2.3	~TraceManager()	262
5.84.3	Member Function Documentation	262
5.84.3.1	addTraceErrorHandler(traceErrorListener traceErrorListener)	262
5.84.3.2	addTraceHandler(traceListener traceListener)	262
5.84.3.3	addTraceReportHandler(traceListener traceReportListener)	263
5.84.3.4	addTraceSimulationHandler(traceSimulationListener traceSimulationListener)	263
5.84.3.5	getErrorMessage() const	263
5.84.3.6	getTraceLevel() const	263
5.84.3.7	setTraceLevel(Util::TraceLevel _traceLevel)	263
5.84.3.8	trace(Util::TraceLevel tracelevel, std::string text)	264
5.84.3.9	traceError(std::exception e, std::string text)	264
5.84.3.10	traceReport(Util::TraceLevel tracelevel, std::string text)	265
5.84.3.11	traceSimulation(Util::TraceLevel tracelevel, double time, Entity *entity, ModelComponent *component, std::string text)	265
5.85	TraceSimulationEvent Class Reference	266
5.85.1	Constructor & Destructor Documentation	267
5.85.1.1	TraceSimulationEvent(Util::TraceLevel tracelevel, double time, Entity *entity, ModelComponent *component, std::string text)	267

5.85.2	Member Function Documentation	267
5.85.2.1	getComponent() const	267
5.85.2.2	getEntity() const	267
5.85.2.3	getTime() const	267
5.86	TraceSimulationProcess Class Reference	268
5.86.1	Detailed Description	268
5.86.2	Constructor & Destructor Documentation	268
5.86.2.1	TraceSimulationProcess(Util::TraceLevel tracelevel, std::string text)	268
5.87	Traits< T > Struct Template Reference	269
5.88	Traits< Collector_if > Struct Template Reference	269
5.88.1	Member Typedef Documentation	269
5.88.1.1	Implementation	269
5.89	Traits< ExperimentDesign_if > Struct Template Reference	269
5.89.1	Member Typedef Documentation	269
5.89.1.1	Implementation	269
5.90	Traits< Fitter_if > Struct Template Reference	270
5.90.1	Member Typedef Documentation	270
5.90.1.1	Implementation	270
5.91	Traits< GenesysApplication_if > Struct Template Reference	270
5.91.1	Member Typedef Documentation	270
5.91.1.1	Application	270
5.92	Traits< HypothesisTester_if > Struct Template Reference	270
5.92.1	Member Typedef Documentation	271
5.92.1.1	Implementation	271
5.93	Traits< Integrator_if > Struct Template Reference	271
5.93.1	Member Typedef Documentation	271
5.93.1.1	Implementation	271
5.93.2	Member Data Documentation	271
5.93.2.1	MaxIterations	271
5.93.2.2	Precision	271

5.94 Traits< Model > Struct Template Reference . . . . .	271
5.94.1 Member Data Documentation . . . . .	272
5.94.1.1 debugged . . . . .	272
5.94.1.2 traceLevel . . . . .	272
5.95 Traits< ModelChecker_if > Struct Template Reference . . . . .	272
5.95.1 Member Typedef Documentation . . . . .	272
5.95.1.1 Implementation . . . . .	272
5.96 Traits< ModelComponent > Struct Template Reference . . . . .	272
5.96.1 Member Typedef Documentation . . . . .	272
5.96.1.1 StatisticsCollector_CollectorImplementation . . . . .	272
5.96.1.2 StatisticsCollector_StatisticsImplementation . . . . .	272
5.97 Traits< ModelPersistence_if > Struct Template Reference . . . . .	273
5.97.1 Member Typedef Documentation . . . . .	273
5.97.1.1 Implementation . . . . .	273
5.98 Traits< Parser_if > Struct Template Reference . . . . .	273
5.98.1 Member Typedef Documentation . . . . .	273
5.98.1.1 Implementation . . . . .	273
5.99 Traits< ProcessAnalyser_if > Struct Template Reference . . . . .	273
5.99.1 Member Typedef Documentation . . . . .	274
5.99.1.1 Implementation . . . . .	274
5.100 Traits< Sampler_if > Struct Template Reference . . . . .	274
5.100.1 Member Typedef Documentation . . . . .	274
5.100.1.1 Implementation . . . . .	274
5.100.1.2 Parameters . . . . .	274
5.101 Traits< SimulationReporter_if > Struct Template Reference . . . . .	274
5.101.1 Member Typedef Documentation . . . . .	274
5.101.1.1 Implementation . . . . .	274
5.102 Traits< Statistics_if > Struct Template Reference . . . . .	275
5.102.1 Member Typedef Documentation . . . . .	275
5.102.1.1 CollectorImplementation . . . . .	275

5.102.1.2 Implementation . . . . .	275
5.103Util Class Reference . . . . .	275
5.103.1 Member Typedef Documentation . . . . .	276
5.103.1.1 identitification . . . . .	276
5.103.1.2 rank . . . . .	276
5.103.2 Member Enumeration Documentation . . . . .	276
5.103.2.1 TimeUnit . . . . .	276
5.103.2.2 TraceLevel . . . . .	276
5.103.3 Member Function Documentation . . . . .	276
5.103.3.1 ClearIndent() . . . . .	276
5.103.3.2 DeclIndent() . . . . .	277
5.103.3.3 GenerateNewId() . . . . .	277
5.103.3.4 GenerateNewIdOfType(std::string objtyp) . . . . .	277
5.103.3.5 GenerateNewIdOfType() . . . . .	277
5.103.3.6 IncIndent() . . . . .	278
5.103.3.7 Indent() . . . . .	278
5.103.3.8 SetW(std::string text, unsigned short width) . . . . .	279
5.103.3.9 TimeUnitConvert(Util::TimeUnit timeUnit1, Util::TimeUnit timeUnit2) . . . . .	279
5.103.3.10TypeOf() . . . . .	279
5.104Variable Class Reference . . . . .	280
5.104.1 Constructor & Destructor Documentation . . . . .	281
5.104.1.1 Variable() . . . . .	281
5.104.1.2 Variable(std::string name) . . . . .	281
5.104.1.3 Variable(const Variable &orig) . . . . .	281
5.104.1.4 ~Variable() . . . . .	281
5.104.2 Member Function Documentation . . . . .	281
5.104.2.1 _loadInstance(std::list< std::string > words) . . . . .	281
5.104.2.2 _saveInstance() . . . . .	281
5.104.2.3 _verifySymbols(std::string *errorMessage) . . . . .	281
5.104.2.4 getValue() . . . . .	281

5.104.2.5	<code>getValue(std::string index)</code>	281
5.104.2.6	<code>setValue(double value)</code>	281
5.104.2.7	<code>setValue(std::string index, double value)</code>	282
5.104.2.8	<code>show()</code>	282
5.105	<b>Waiting Class Reference</b>	282
5.105.1	<b>Constructor &amp; Destructor Documentation</b>	283
5.105.1.1	<code>Waiting(Entity *entity, ModelComponent *component, double timeStartedWaiting)</code>	283
5.105.1.2	<code>Waiting(const Waiting &amp;orig)</code>	283
5.105.1.3	<code>~Waiting()</code>	283
5.105.2	<b>Member Function Documentation</b>	283
5.105.2.1	<code>getComponent() const</code>	283
5.105.2.2	<code>getEntity() const</code>	283
5.105.2.3	<code>getTimeStartedWaiting() const</code>	283
5.105.2.4	<code>show()</code>	283
5.106	<b>WaitingResource Class Reference</b>	284
5.106.1	<b>Constructor &amp; Destructor Documentation</b>	285
5.106.1.1	<code>WaitingResource(Entity *entity, ModelComponent *component, double timeStartedWaiting, unsigned int quantity)</code>	285
5.106.1.2	<code>WaitingResource(const WaitingResource &amp;orig)</code>	285
5.106.1.3	<code>~WaitingResource()</code>	285
5.106.2	<b>Member Function Documentation</b>	285
5.106.2.1	<code>getQuantity() const</code>	285
5.106.2.2	<code>show()</code>	285

<b>6</b>	<b>File Documentation</b>	<b>287</b>
6.1	.dep.inc File Reference	287
6.2	Assign.cpp File Reference	287
6.3	Assign.h File Reference	287
6.4	Attribute.cpp File Reference	288
6.5	Attribute.h File Reference	289
6.6	BuildSimulationModel.cpp File Reference	290
6.6.1	Function Documentation	291
6.6.1.1	buildModel(Model *model)	291
6.6.1.2	buildSimulationSystem()	293
6.6.1.3	onEntityRemoveHandler(SimulationEvent *re)	295
6.6.1.4	onProcessEventHandler(SimulationEvent *re)	295
6.6.1.5	onReplicationEndHandler(SimulationEvent *re)	296
6.6.1.6	onReplicationStartHandler(SimulationEvent *re)	296
6.6.1.7	onSimulationStartHandler(SimulationEvent *re)	297
6.6.1.8	traceHandler(TraceEvent e)	297
6.6.1.9	traceSimulationHandler(TraceSimulationEvent e)	297
6.7	BuildSimulationModel.h File Reference	298
6.8	Collector_if.h File Reference	298
6.8.1	Typedef Documentation	300
6.8.1.1	CollectorAddValueHandler	300
6.8.1.2	CollectorClearHandler	300
6.8.2	Function Documentation	300
6.8.2.1	SetCollectorAddValueHandler(void(Class::*function)(double), Class *object)	300
6.8.2.2	SetCollectorClearHandler(void(Class::*function)(), Class *object)	300
6.9	CollectorDatafile_if.h File Reference	301
6.10	CollectorDatafileDefaultImpl1.cpp File Reference	301
6.11	CollectorDatafileDefaultImpl1.h File Reference	302
6.12	CollectorDatafileDummyImpl.cpp File Reference	303
6.13	CollectorDatafileDummyImpl.h File Reference	304

6.14	CollectorDefaultImpl1.cpp File Reference	305
6.15	CollectorDefaultImpl1.h File Reference	306
6.16	CollectorDummyImpl.cpp File Reference	307
6.17	CollectorDummyImpl.h File Reference	307
6.18	Create.cpp File Reference	308
6.19	Create.h File Reference	309
6.20	Decide.cpp File Reference	310
6.21	Decide.h File Reference	310
6.22	DefineGetterSetter.h File Reference	311
6.22.1	Typedef Documentation	313
6.22.1.1	GetterMember	313
6.22.1.2	SetterMember	313
6.22.2	Function Documentation	313
6.22.2.1	DefineGetterMember(Class *object, double(Class::*function)())	313
6.22.2.2	DefineGetterMember(Class *object, unsigned int(Class::*function)() const)	313
6.22.2.3	DefineGetterMember(Class *object, bool(Class::*function)() const)	313
6.22.2.4	DefineGetterMember(Class *object, std::string(Class::*function)() const)	313
6.22.2.5	DefineGetterMember(Class *object, Util::TimeUnit(Class::*function)() const)	313
6.22.2.6	DefineSetterMember(Class *object, void(Class::*function)(double))	313
6.22.2.7	DefineSetterMember(Class *object, void(Class::*function)(unsigned int))	313
6.22.2.8	DefineSetterMember(Class *object, void(Class::*function)(bool))	313
6.22.2.9	DefineSetterMember(Class *object, void(Class::*function)(std::string) const)	313
6.22.2.10	DefineSetterMember(Class *object, void(Class::*function)(Util::TimeUnit))	313
6.23	Delay.cpp File Reference	313
6.24	Delay.h File Reference	314
6.25	Dispose.cpp File Reference	314
6.26	Dispose.h File Reference	315
6.27	ElementManager.cpp File Reference	316
6.28	ElementManager.h File Reference	316
6.29	ElementManager_if.h File Reference	317

6.30 Entity.cpp File Reference . . . . .	317
6.31 Entity.h File Reference . . . . .	318
6.32 EntityType.cpp File Reference . . . . .	319
6.33 EntityType.h File Reference . . . . .	319
6.34 Event.cpp File Reference . . . . .	320
6.35 Event.h File Reference . . . . .	320
6.36 ExperimentDesign_if.h File Reference . . . . .	321
6.37 ExperimentDesignDummyImpl.cpp File Reference . . . . .	322
6.38 ExperimentDesignDummyImpl.h File Reference . . . . .	323
6.39 FactorOrInteractionContribution.cpp File Reference . . . . .	323
6.40 FactorOrInteractionContribution.h File Reference . . . . .	324
6.41 Fitter_if.h File Reference . . . . .	325
6.42 FitterDummyImpl.cpp File Reference . . . . .	326
6.43 FitterDummyImpl.h File Reference . . . . .	327
6.44 Functor.h File Reference . . . . .	328
6.45 GenesysApplication_if.h File Reference . . . . .	328
6.46 HypothesisTester_if.h File Reference . . . . .	329
6.47 HypothesisTesterDummyImpl.cpp File Reference . . . . .	329
6.48 HypothesisTesterDummyImpl.h File Reference . . . . .	330
6.49 Integrator_if.h File Reference . . . . .	331
6.50 IntegratorDefaultImpl1.cpp File Reference . . . . .	331
6.51 IntegratorDefaultImpl1.h File Reference . . . . .	332
6.52 IntegratorDummyImpl.cpp File Reference . . . . .	332
6.53 IntegratorDummyImpl.h File Reference . . . . .	333
6.54 LinkedBy.cpp File Reference . . . . .	333
6.55 LinkedBy.h File Reference . . . . .	334
6.56 List.h File Reference . . . . .	334
6.57 main.cpp File Reference . . . . .	335
6.57.1 Function Documentation . . . . .	336
6.57.1.1 main(int argc, char **argv) . . . . .	336



6.58	Model.cpp File Reference . . . . .	336
6.58.1	Function Documentation . . . . .	337
6.58.1.1	EventCompare(const Event *a, const Event *b) . . . . .	337
6.58.1.2	getReplicationLengthNotMemberFunction() . . . . .	337
6.58.1.3	setReplicationLengthNotMemberFunction(double value) . . . . .	337
6.59	Model.h File Reference . . . . .	337
6.60	ModelChecker_if.h File Reference . . . . .	338
6.61	ModelCheckerDefaultImpl1.cpp File Reference . . . . .	339
6.62	ModelCheckerDefaultImpl1.h File Reference . . . . .	339
6.63	ModelCheckerDummyImpl.cpp File Reference . . . . .	340
6.64	ModelCheckerDummyImpl.h File Reference . . . . .	340
6.65	ModelComponent.cpp File Reference . . . . .	341
6.66	ModelComponent.h File Reference . . . . .	341
6.67	ModelComponentManager_if.h File Reference . . . . .	342
6.68	ModelElement.cpp File Reference . . . . .	342
6.69	ModelElement.h File Reference . . . . .	343
6.70	ModelInfo.cpp File Reference . . . . .	344
6.71	ModelInfo.h File Reference . . . . .	344
6.72	ModelPersistence_if.h File Reference . . . . .	345
6.73	ModelPersistenceDummyImpl.cpp File Reference . . . . .	346
6.74	ModelPersistenceDummyImpl.h File Reference . . . . .	346
6.75	ModelSimulation.cpp File Reference . . . . .	347
6.76	ModelSimulation.h File Reference . . . . .	347
6.77	OnEventManager.cpp File Reference . . . . .	348
6.78	OnEventManager.h File Reference . . . . .	349
6.78.1	Typedef Documentation . . . . .	350
6.78.1.1	simulationEventHandler . . . . .	350
6.79	Parser_if.h File Reference . . . . .	350
6.80	ParserDefaultImpl1.cpp File Reference . . . . .	351
6.81	ParserDefaultImpl1.h File Reference . . . . .	351

6.82 ParserDummyImpl.cpp File Reference . . . . .	352
6.83 ParserDummyImpl.h File Reference . . . . .	353
6.84 Plugin.cpp File Reference . . . . .	353
6.85 Plugin.h File Reference . . . . .	354
6.86 ProbDistrib.cpp File Reference . . . . .	355
6.87 ProbDistrib.h File Reference . . . . .	355
6.88 ProcessAnalyser_if.h File Reference . . . . .	356
6.89 ProcessAnalyserDummyImpl.cpp File Reference . . . . .	357
6.90 ProcessAnalyserDummyImpl.h File Reference . . . . .	357
6.91 Queue.cpp File Reference . . . . .	358
6.92 Queue.h File Reference . . . . .	359
6.93 README.md File Reference . . . . .	360
6.94 Record.cpp File Reference . . . . .	360
6.95 Record.h File Reference . . . . .	360
6.96 Release.cpp File Reference . . . . .	361
6.97 Release.h File Reference . . . . .	361
6.98 Resource.cpp File Reference . . . . .	362
6.99 Resource.h File Reference . . . . .	363
6.100Sampler_if.h File Reference . . . . .	364
6.101SamplerDefaultImpl1.cpp File Reference . . . . .	364
6.102SamplerDefaultImpl1.h File Reference . . . . .	365
6.103SamplerDummyImpl.cpp File Reference . . . . .	366
6.104SamplerDummyImpl.h File Reference . . . . .	366
6.105ScenarioExperiment_if.h File Reference . . . . .	367
6.106Seize.cpp File Reference . . . . .	367
6.107Seize.h File Reference . . . . .	368
6.108SimulationControl.cpp File Reference . . . . .	368
6.109SimulationControl.h File Reference . . . . .	369
6.110SimulationReporter_if.h File Reference . . . . .	370
6.111SimulationReporterDefaultImpl1.cpp File Reference . . . . .	371

6.112SimulationReporterDefaultImpl1.h File Reference . . . . .	371
6.113SimulationResponse.cpp File Reference . . . . .	372
6.114SimulationResponse.h File Reference . . . . .	372
6.115SimulationScenario.cpp File Reference . . . . .	373
6.116SimulationScenario.h File Reference . . . . .	374
6.117Simulator.cpp File Reference . . . . .	375
6.118Simulator.h File Reference . . . . .	376
6.119SinkModelComponent.cpp File Reference . . . . .	377
6.120SinkModelComponent.h File Reference . . . . .	377
6.121SourceModelComponent.cpp File Reference . . . . .	378
6.122SourceModelComponent.h File Reference . . . . .	378
6.123Statistics_if.h File Reference . . . . .	379
6.124StatisticsCollector.cpp File Reference . . . . .	380
6.125StatisticsCollector.h File Reference . . . . .	381
6.126StatisticsDataFile_if.h File Reference . . . . .	381
6.127StatisticsDataFileDummyImpl.cpp File Reference . . . . .	383
6.128StatisticsDataFileDummyImpl.h File Reference . . . . .	383
6.129StatisticsDefaultImpl1.cpp File Reference . . . . .	384
6.130StatisticsDefaultImpl1.h File Reference . . . . .	384
6.131StatisticsDummyImpl.cpp File Reference . . . . .	385
6.132StatisticsDummyImpl.h File Reference . . . . .	386
6.133TestInputAnalyserTools.cpp File Reference . . . . .	387
6.133.1 Function Documentation . . . . .	387
6.133.1.1 testStudentSoftwareDevelopments() . . . . .	388
6.134TestInputAnalyserTools.h File Reference . . . . .	389
6.135TestParser.cpp File Reference . . . . .	389
6.136TestParser.h File Reference . . . . .	390
6.137TestStatistics.cpp File Reference . . . . .	391
6.138TestStatistics.h File Reference . . . . .	391
6.139TraceManager.cpp File Reference . . . . .	392

6.140TraceManager.h File Reference . . . . .	392
6.140.1 Typedef Documentation . . . . .	393
6.140.1.1 traceErrorListener . . . . .	393
6.140.1.2 traceListener . . . . .	393
6.140.1.3 traceSimulationListener . . . . .	393
6.140.1.4 traceSimulationProcessListener . . . . .	393
6.141 Traits.h File Reference . . . . .	393
6.142Util.cpp File Reference . . . . .	394
6.143Util.h File Reference . . . . .	395
6.144Variable.cpp File Reference . . . . .	396
6.145Variable.h File Reference . . . . .	397
6.146Waiting.cpp File Reference . . . . .	398
6.147Waiting.h File Reference . . . . .	398
6.148WaitingResource.cpp File Reference . . . . .	399
6.149WaitingResource.h File Reference . . . . .	400

# Chapter 1

## GenESyS-Reborn

Generic and Expansible System [Simulator](#)

(Work in progress C++ port from the original in Pascal)

Developed by [rlcancian](#)



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Assign::Assignment . . . . .	20
Collector_if . . . . .	27
CollectorDatafile_if . . . . .	30
CollectorDatafileDefaultImpl1 . . . . .	32
CollectorDatafileDummyImpl . . . . .	35
CollectorDefaultImpl1 . . . . .	38
CollectorDummyImpl . . . . .	40
ElementManager . . . . .	59
ElementManager_if . . . . .	64
Event . . . . .	74
ExperimentDesign_if . . . . .	75
ExperimentDesignDummyImpl . . . . .	77
FactorOrInteractionContribution . . . . .	78
Fitter_if . . . . .	79
FitterDummyImpl . . . . .	82
GenesysApplication_if . . . . .	85
BuildSimulationModel . . . . .	25
TestInputAnalyserTools . . . . .	254
TestParser . . . . .	256
TestStatistics . . . . .	257
HypothesisTester_if . . . . .	86
HypothesisTesterDummyImpl . . . . .	88
Integrator_if . . . . .	91
IntegratorDefaultImpl1 . . . . .	93
IntegratorDummyImpl . . . . .	95
LinkedBy . . . . .	97
Queue . . . . .	173
Resource . . . . .	189
List< T > . . . . .	98
List< Assign::Assignment * > . . . . .	98
List< double > . . . . .	98
List< Event * > . . . . .	98
List< Model * > . . . . .	98

List< ModelComponent * > . . . . .	98
List< Plugin * > . . . . .	98
List< ResourceEventHandler > . . . . .	98
List< SimulationControl * > . . . . .	98
List< SimulationResponse * > . . . . .	98
List< std::string > . . . . .	98
List< Waiting * > . . . . .	98
Model . . . . .	105
ModelChecker_if . . . . .	117
ModelCheckerDefaultImpl1 . . . . .	119
ModelCheckerDummyImpl . . . . .	123
ModelComponentManager_if . . . . .	134
ModelElement . . . . .	134
Attribute . . . . .	23
Entity . . . . .	65
EntityType . . . . .	70
ModelComponent . . . . .	127
Assign . . . . .	15
Decide . . . . .	46
Delay . . . . .	49
Record . . . . .	180
Release . . . . .	184
Seize . . . . .	207
SinkModelComponent . . . . .	225
Dispose . . . . .	55
SourceModelComponent . . . . .	227
Create . . . . .	42
Queue . . . . .	173
Resource . . . . .	189
StatisticsCollector . . . . .	236
Variable . . . . .	280
ModelInfo . . . . .	140
ModelPersistence_if . . . . .	145
ModelPersistenceDummyImpl . . . . .	147
ModelSimulation . . . . .	150
OnEventManager . . . . .	157
Parser_if . . . . .	161
ParserDefaultImpl1 . . . . .	162
ParserDummyImpl . . . . .	164
Plugin . . . . .	166
ProbDistrib . . . . .	167
ProcessAnalyser_if . . . . .	169
ProcessAnalyserDummyImpl . . . . .	171
Sampler_if::RNG_Parameters . . . . .	196
SamplerDefaultImpl1::DefaultImpl1RNG_Parameters . . . . .	48
SamplerDummyImpl::MyRNG_Parameters . . . . .	156
Sampler_if . . . . .	196
SamplerDefaultImpl1 . . . . .	200
SamplerDummyImpl . . . . .	204
ScenarioExperiment_if . . . . .	207
SimulationEvent . . . . .	215
SimulationReporter_if . . . . .	216
SimulationReporterDefaultImpl1 . . . . .	217
SimulationResponse . . . . .	220
SimulationControl . . . . .	214



SimulationScenario . . . . .	221
Simulator . . . . .	223
Statistics_if . . . . .	232
StatisticsDatafile_if . . . . .	240
StatisticsDataFileDummyImpl . . . . .	242
StatisticsDefaultImpl1 . . . . .	246
StatisticsDummyImpl . . . . .	250
TraceEvent . . . . .	261
TraceErrorEvent . . . . .	259
TraceSimulationEvent . . . . .	266
TraceSimulationProcess . . . . .	268
TraceManager . . . . .	262
Traits< T > . . . . .	269
Traits< Collector_if > . . . . .	269
Traits< ExperimentDesign_if > . . . . .	269
Traits< Fitter_if > . . . . .	270
Traits< GenesysApplication_if > . . . . .	270
Traits< HypothesisTester_if > . . . . .	270
Traits< Integrator_if > . . . . .	271
Traits< Model > . . . . .	271
Traits< ModelChecker_if > . . . . .	272
Traits< ModelComponent > . . . . .	272
Traits< ModelPersistence_if > . . . . .	273
Traits< Parser_if > . . . . .	273
Traits< ProcessAnalyser_if > . . . . .	273
Traits< Sampler_if > . . . . .	274
Traits< SimulationReporter_if > . . . . .	274
Traits< Statistics_if > . . . . .	275
Util . . . . .	275
Waiting . . . . .	282
WaitingResource . . . . .	284



## Chapter 3

# Class Index

### 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Assign	15
Assign::Assignment	20
Attribute	23
BuildSimulationModel	25
Collector_if	27
CollectorDatafile_if	30
CollectorDatafileDefaultImpl1	32
CollectorDatafileDummyImpl	35
CollectorDefaultImpl1	38
CollectorDummyImpl	40
Create	42
Decide	46
SamplerDefaultImpl1::DefaultImpl1RNG_Parameters	48
Delay	49
Dispose	55
ElementManager	59
ElementManager_if	64
Entity	65
EntityType	70
Event	74
ExperimentDesign_if	75
ExperimentDesignDummyImpl	77
FactorOrInteractionContribution	78
Fitter_if	79
FitterDummyImpl	82
GenesysApplication_if	85
HypothesisTester_if	86
HypothesisTesterDummyImpl	88
Integrator_if	91
IntegratorDefaultImpl1	93
IntegratorDummyImpl	95
LinkedBy	97
List< T >	98
Model	105
ModelChecker_if	117

ModelCheckerDefaultImpl1	119
ModelCheckerDummyImpl	123
ModelComponent	127
ModelComponentManager_if	134
ModelElement	134
ModelInfo	140
ModelPersistence_if	145
ModelPersistenceDummyImpl	147
ModelSimulation	150
SamplerDummyImpl::MyRNG_Parameters	156
OnEventManager	157
Parser_if	161
ParserDefaultImpl1	162
ParserDummyImpl	164
Plugin	166
ProbDistrib	167
ProcessAnalyser_if	169
ProcessAnalyserDummyImpl	171
Queue	173
Record	180
Release	184
Resource	189
Sampler_if::RNG_Parameters	196
Sampler_if	196
SamplerDefaultImpl1	200
SamplerDummyImpl	204
ScenarioExperiment_if	207
Seize	207
SimulationControl	214
SimulationEvent	215
SimulationReporter_if	216
SimulationReporterDefaultImpl1	217
SimulationResponse	220
SimulationScenario	221
Simulator	223
SinkModelComponent	225
SourceModelComponent	227
Statistics_if	232
StatisticsCollector	236
StatisticsDatafile_if	240
StatisticsDataFileDummyImpl	242
StatisticsDefaultImpl1	246
StatisticsDummyImpl	250
TestInputAnalyserTools	254
TestParser	256
TestStatistics	257
TraceErrorEvent	259
TraceEvent	261
TraceManager	262
TraceSimulationEvent	266
TraceSimulationProcess	268
Traits< T >	269
Traits< Collector_if >	269
Traits< ExperimentDesign_if >	269
Traits< Fitter_if >	270
Traits< GenesysApplication_if >	270
Traits< HypothesisTester_if >	270
Traits< Integrator_if >	271

Traits< Model > . . . . .	271
Traits< ModelChecker_if > . . . . .	272
Traits< ModelComponent > . . . . .	272
Traits< ModelPersistence_if > . . . . .	273
Traits< Parser_if > . . . . .	273
Traits< ProcessAnalyser_if > . . . . .	273
Traits< Sampler_if > . . . . .	274
Traits< SimulationReporter_if > . . . . .	274
Traits< Statistics_if > . . . . .	275
Util . . . . .	275
Variable . . . . .	280
Waiting . . . . .	282
WaitingResource . . . . .	284



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

.dep.inc	287
Assign.cpp	287
Assign.h	287
Attribute.cpp	288
Attribute.h	289
BuildSimulationModel.cpp	290
BuildSimulationModel.h	298
Collector_if.h	298
CollectorDatafile_if.h	301
CollectorDatafileDefaultImpl1.cpp	301
CollectorDatafileDefaultImpl1.h	302
CollectorDatafileDummyImpl.cpp	303
CollectorDatafileDummyImpl.h	304
CollectorDefaultImpl1.cpp	305
CollectorDefaultImpl1.h	306
CollectorDummyImpl.cpp	307
CollectorDummyImpl.h	307
Create.cpp	308
Create.h	309
Decide.cpp	310
Decide.h	310
DefineGetterSetter.h	311
Delay.cpp	313
Delay.h	314
Dispose.cpp	314
Dispose.h	315
ElementManager.cpp	316
ElementManager.h	316
ElementManager_if.h	317
Entity.cpp	317
Entity.h	318
EntityType.cpp	319
EntityType.h	319
Event.cpp	320
Event.h	320

ExperimentDesign_if.h	321
ExperimentDesignDummyImpl.cpp	322
ExperimentDesignDummyImpl.h	323
FactorOrInteractionContribution.cpp	323
FactorOrInteractionContribution.h	324
Fitter_if.h	325
FitterDummyImpl.cpp	326
FitterDummyImpl.h	327
Functor.h	328
GenesysApplication_if.h	328
HypothesisTester_if.h	329
HypothesisTesterDummyImpl.cpp	329
HypothesisTesterDummyImpl.h	330
Integrator_if.h	331
IntegratorDefaultImpl1.cpp	331
IntegratorDefaultImpl1.h	332
IntegratorDummyImpl.cpp	332
IntegratorDummyImpl.h	333
LinkedBy.cpp	333
LinkedBy.h	334
List.h	334
main.cpp	335
Model.cpp	336
Model.h	337
ModelChecker_if.h	338
ModelCheckerDefaultImpl1.cpp	339
ModelCheckerDefaultImpl1.h	339
ModelCheckerDummyImpl.cpp	340
ModelCheckerDummyImpl.h	340
ModelComponent.cpp	341
ModelComponent.h	341
ModelComponentManager_if.h	342
ModelElement.cpp	342
ModelElement.h	343
ModelInfo.cpp	344
ModelInfo.h	344
ModelPersistence_if.h	345
ModelPersistenceDummyImpl.cpp	346
ModelPersistenceDummyImpl.h	346
ModelSimulation.cpp	347
ModelSimulation.h	347
OnEventManager.cpp	348
OnEventManager.h	349
Parser_if.h	350
ParserDefaultImpl1.cpp	351
ParserDefaultImpl1.h	351
ParserDummyImpl.cpp	352
ParserDummyImpl.h	353
Plugin.cpp	353
Plugin.h	354
ProbDistrib.cpp	355
ProbDistrib.h	355
ProcessAnalyser_if.h	356
ProcessAnalyserDummyImpl.cpp	357
ProcessAnalyserDummyImpl.h	357
Queue.cpp	358
Queue.h	359
Record.cpp	360



Record.h	360
Release.cpp	361
Release.h	361
Resource.cpp	362
Resource.h	363
Sampler_if.h	364
SamplerDefaultImpl1.cpp	364
SamplerDefaultImpl1.h	365
SamplerDummyImpl.cpp	366
SamplerDummyImpl.h	366
ScenarioExperiment_if.h	367
Seize.cpp	367
Seize.h	368
SimulationControl.cpp	368
SimulationControl.h	369
SimulationReporter_if.h	370
SimulationReporterDefaultImpl1.cpp	371
SimulationReporterDefaultImpl1.h	371
SimulationResponse.cpp	372
SimulationResponse.h	372
SimulationScenario.cpp	373
SimulationScenario.h	374
Simulator.cpp	375
Simulator.h	376
SinkModelComponent.cpp	377
SinkModelComponent.h	377
SourceModelComponent.cpp	378
SourceModelComponent.h	378
Statistics_if.h	379
StatisticsCollector.cpp	380
StatisticsCollector.h	381
StatisticsDataFile_if.h	381
StatisticsDataFileDummyImpl.cpp	383
StatisticsDataFileDummyImpl.h	383
StatisticsDefaultImpl1.cpp	384
StatisticsDefaultImpl1.h	384
StatisticsDummyImpl.cpp	385
StatisticsDummyImpl.h	386
TestInputAnalyserTools.cpp	387
TestInputAnalyserTools.h	389
TestParser.cpp	389
TestParser.h	390
TestStatistics.cpp	391
TestStatistics.h	391
TraceManager.cpp	392
TraceManager.h	392
Traits.h	393
Util.cpp	394
Util.h	395
Variable.cpp	396
Variable.h	397
Waiting.cpp	398
Waiting.h	398
WaitingResource.cpp	399
WaitingResource.h	400



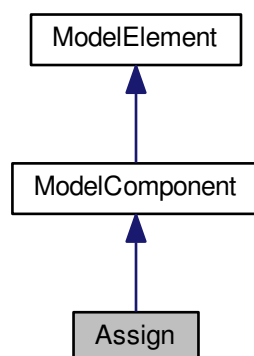
## Chapter 5

# Class Documentation

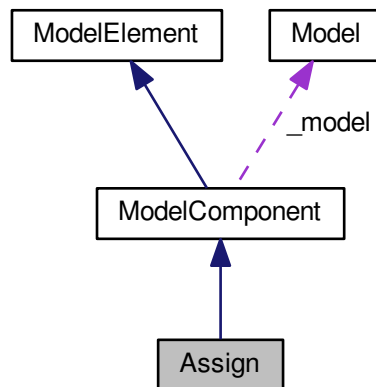
### 5.1 Assign Class Reference

```
#include <Assign.h>
```

Inheritance diagram for Assign:



Collaboration diagram for Assign:



## Classes

- class [Assignment](#)

## Public Types

- enum [DestinationType](#) : int { [DestinationType::Attribute](#), [DestinationType::Variable](#) }

## Public Member Functions

- [Assign](#) ([Model](#) \*model)
- [Assign](#) (const [Assign](#) &orig)
- virtual [~Assign](#) ()
- virtual std::string [show](#) ()
- [List](#)< [Assignment](#) \* > \* [getAssignments](#) () const

## Protected Member Functions

- virtual void [\\_execute](#) ([Entity](#) \*entity)
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.1.1 Member Enumeration Documentation

5.1.1.1 enum [Assign::DestinationType](#) : int [strong]

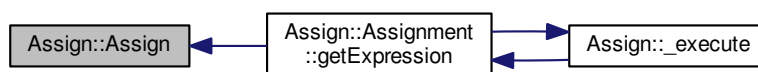
Enumerator

***Attribute***  
***Variable***

## 5.1.2 Constructor & Destructor Documentation

### 5.1.2.1 `Assign::Assign ( Model * model )`

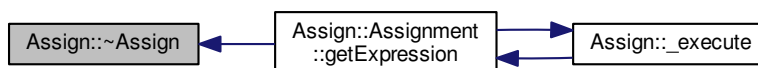
Here is the caller graph for this function:



### 5.1.2.2 `Assign::Assign ( const Assign & orig )`

### 5.1.2.3 `Assign::~~Assign ( )` [virtual]

Here is the caller graph for this function:

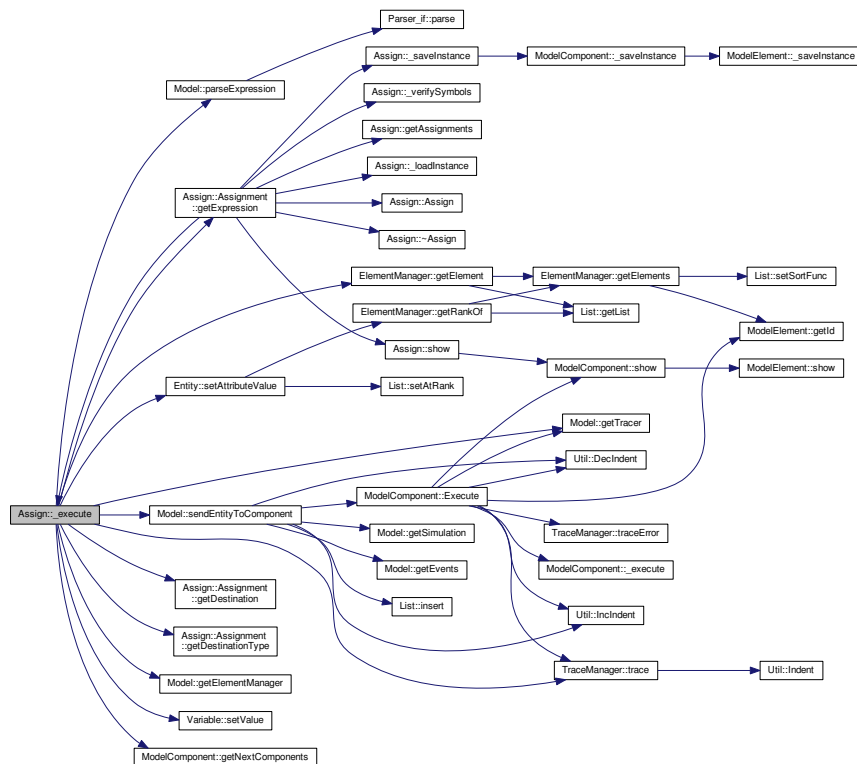


## 5.1.3 Member Function Documentation

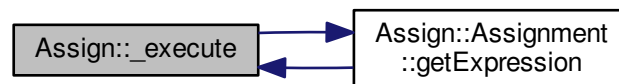
### 5.1.3.1 `void Assign::_execute ( Entity * entity )` [protected], [virtual]

Implements [ModelComponent](#).

Here is the call graph for this function:



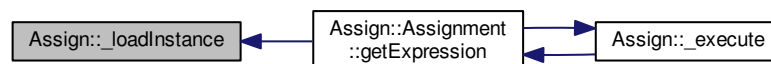
Here is the caller graph for this function:



**5.1.3.2** `void Assign::_loadInstance ( std::list< std::string > words )` `[protected], [virtual]`

Implements [ModelElement](#).

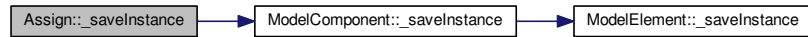
Here is the caller graph for this function:



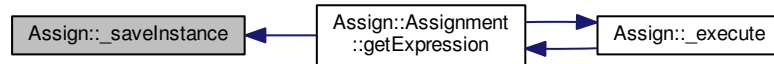
5.1.3.3 `std::list< std::string > * Assign::_saveInstance ( )` [protected],[virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



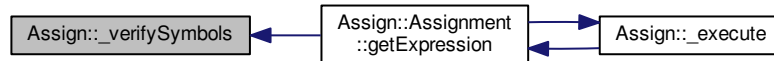
Here is the caller graph for this function:



5.1.3.4 `bool Assign::_verifySymbols ( std::string * errorMessage )` [protected],[virtual]

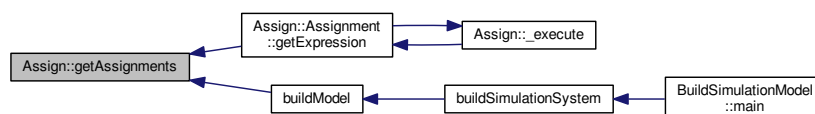
Implements [ModelElement](#).

Here is the caller graph for this function:



5.1.3.5 `List< Assign::Assignment * > * Assign::getAssignments ( )` const

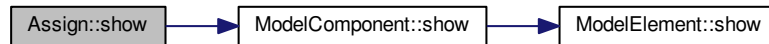
Here is the caller graph for this function:



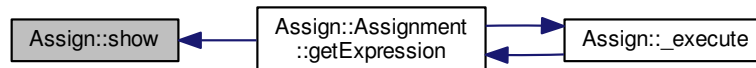
### 5.1.3.6 `std::string Assign::show ( )` [virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



Here is the caller graph for this function:



The documentation for this class was generated from the following files:

- [Assign.h](#)
- [Assign.cpp](#)

## 5.2 Assign::Assignment Class Reference

```
#include <Assign.h>
```

### Public Member Functions

- [Assignment](#) ([DestinationType](#) destinationType, std::string destination, std::string expression)
- void [setDestination](#) (std::string \_destination)
- std::string [getDestination](#) () const
- void [setDestinationType](#) ([DestinationType](#) \_destinationType)
- [DestinationType](#) [getDestinationType](#) () const
- void [setExpression](#) (std::string \_expression)
- std::string [getExpression](#) () const

### 5.2.1 Detailed Description

While the assign class allows you to perform multiple assignments, the assignment class defines an assignment itself.



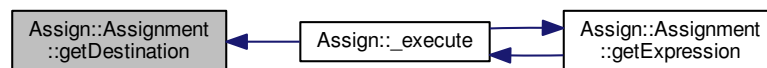
## 5.2.2 Constructor & Destructor Documentation

5.2.2.1 `Assign::Assignment::Assignment ( DestinationType destinationType, std::string destination, std::string expression ) [inline]`

## 5.2.3 Member Function Documentation

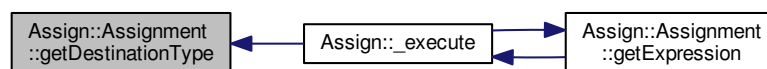
5.2.3.1 `std::string Assign::Assignment::getDestination ( ) const [inline]`

Here is the caller graph for this function:



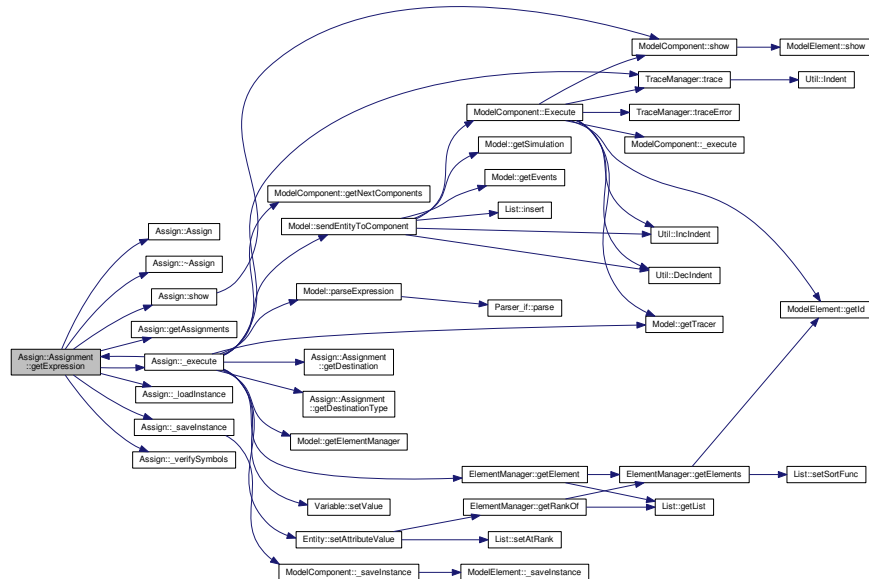
5.2.3.2 `DestinationType Assign::Assignment::getDestinationType ( ) const [inline]`

Here is the caller graph for this function:



### 5.2.3.3 `std::string Assign::Assignment::getExpression ( ) const [inline]`

Here is the call graph for this function:



Here is the caller graph for this function:



### 5.2.3.4 `void Assign::Assignment::setDestination ( std::string _destination ) [inline]`

### 5.2.3.5 `void Assign::Assignment::setDestinationType ( DestinationType _destinationType ) [inline]`

### 5.2.3.6 `void Assign::Assignment::setExpression ( std::string _expression ) [inline]`

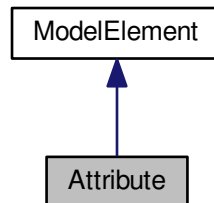
The documentation for this class was generated from the following file:

- [Assign.h](#)

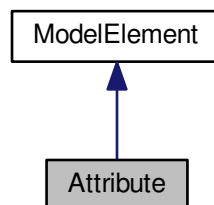
## 5.3 Attribute Class Reference

```
#include <Attribute.h>
```

Inheritance diagram for Attribute:



Collaboration diagram for Attribute:



### Public Member Functions

- [Attribute](#) ()
- [Attribute](#) (std::string name)
- [Attribute](#) (const [Attribute](#) &orig)
- virtual [~Attribute](#) ()
- virtual std::string [show](#) ()

### Protected Member Functions

- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.3.1 Constructor & Destructor Documentation

5.3.1.1 `Attribute::Attribute ( )`

5.3.1.2 `Attribute::Attribute ( std::string name )`

5.3.1.3 `Attribute::Attribute ( const Attribute & orig )`

5.3.1.4 `Attribute::~~Attribute ( )` `[virtual]`

### 5.3.2 Member Function Documentation

5.3.2.1 `void Attribute::_loadInstance ( std::list< std::string > words )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

5.3.2.2 `std::list< std::string > * Attribute::_saveInstance ( )` `[protected]`, `[virtual]`

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



5.3.2.3 `bool Attribute::_verifySymbols ( std::string * errorMessage )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

5.3.2.4 `std::string Attribute::show ( )` `[virtual]`

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



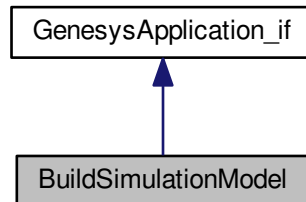
The documentation for this class was generated from the following files:

- [Attribute.h](#)
- [Attribute.cpp](#)

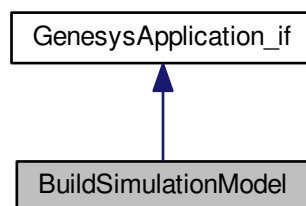
## 5.4 BuildSimulationModel Class Reference

```
#include <BuildSimulationModel.h>
```

Inheritance diagram for BuildSimulationModel:



Collaboration diagram for BuildSimulationModel:



### Public Member Functions

- [BuildSimulationModel](#) ()
- int [main](#) (int argc, char \*\*argv)

#### 5.4.1 Constructor & Destructor Documentation

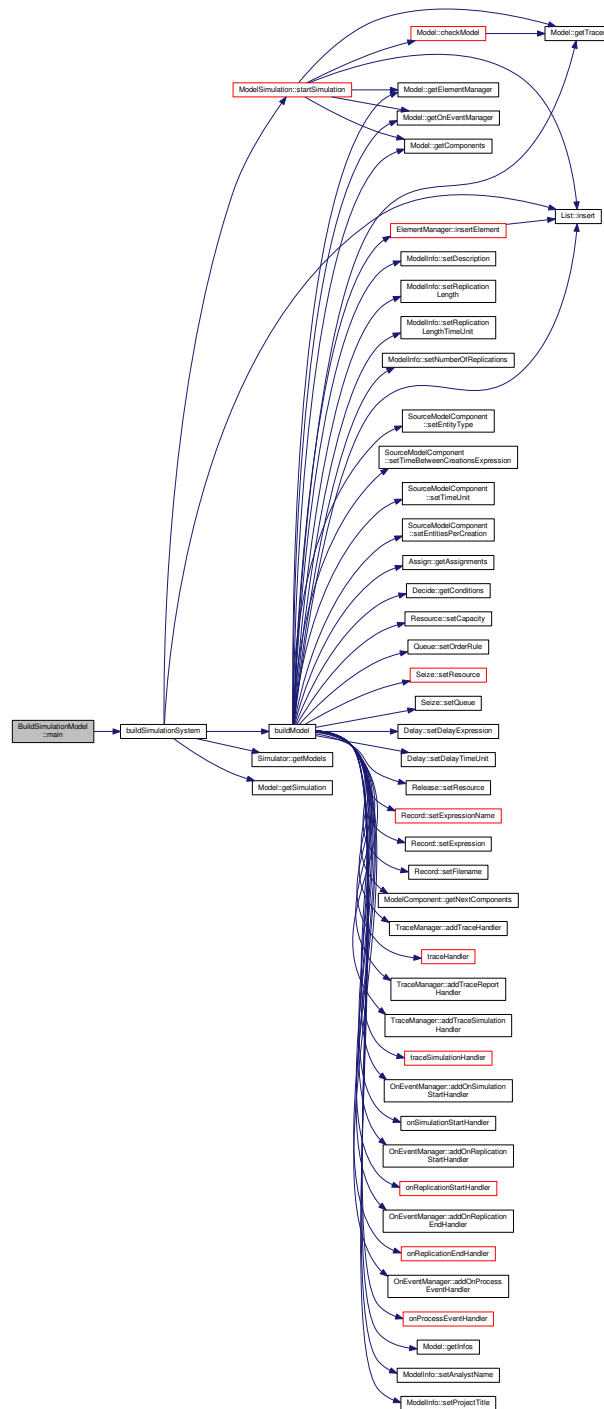
5.4.1.1 [BuildSimulationModel::BuildSimulationModel](#) ( )

#### 5.4.2 Member Function Documentation

5.4.2.1 int [BuildSimulationModel::main](#) ( int *argc*, char \*\* *argv* ) [virtual]

Implements [GenesysApplication\\_if](#).

Here is the call graph for this function:



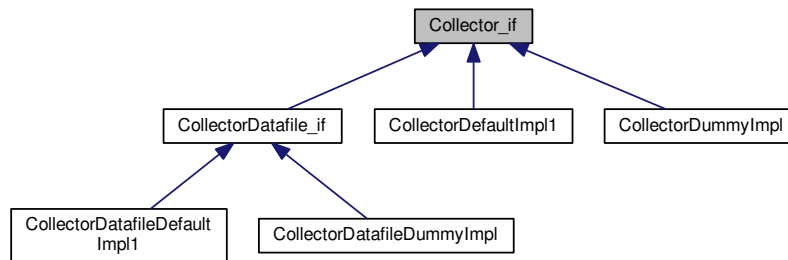
The documentation for this class was generated from the following files:

- [BuildSimulationModel.h](#)
- [BuildSimulationModel.cpp](#)

## 5.5 Collector\_if Class Reference

```
#include <Collector_if.h>
```

Inheritance diagram for Collector\_if:



### Public Member Functions

- virtual void [clear](#) ()=0
- virtual void [addValue](#) (double value)=0
- virtual double [getLastValue](#) ()=0
- virtual unsigned long [numElements](#) ()=0
- virtual void [setAddValueHandler](#) ([CollectorAddValueHandler](#) addValueHandler)=0
- virtual void [setClearHandler](#) ([CollectorClearHandler](#) clearHandler)=0

#### 5.5.1 Detailed Description

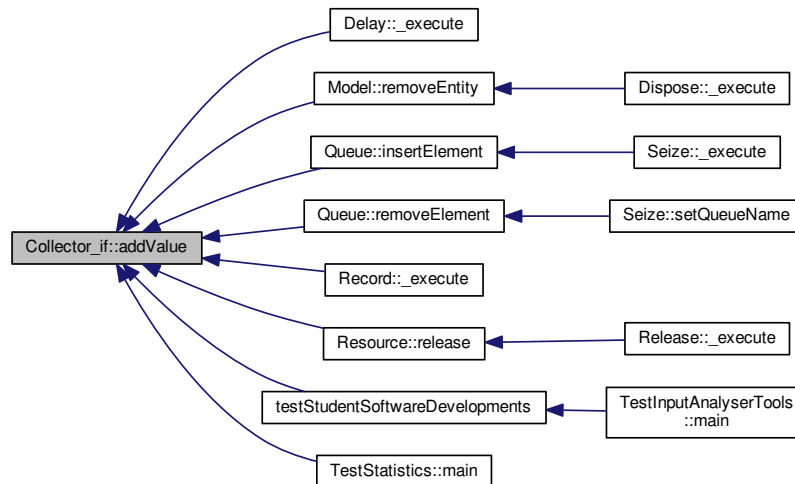
Interface for collecting values of a single stochastic variable. Values collected can be used as base for statistical analysis.

#### 5.5.2 Member Function Documentation

**5.5.2.1** virtual void [Collector\\_if::addValue](#) ( double *value* ) [pure virtual]

Implemented in [CollectorDatafileDefaultImpl1](#), [CollectorDummyImpl](#), [CollectorDatafileDummyImpl](#), and [CollectorDatafileDefaultImpl1](#).

Here is the caller graph for this function:



#### 5.5.2.2 virtual void Collector\_if::clear ( ) [pure virtual]

Implemented in [CollectorDatafileDefaultImpl1](#), [CollectorDummyImpl](#), [CollectorDatafileDummyImpl](#), and [CollectorDefaultImpl1](#).

Here is the caller graph for this function:



#### 5.5.2.3 virtual double Collector\_if::getLastValue ( ) [pure virtual]

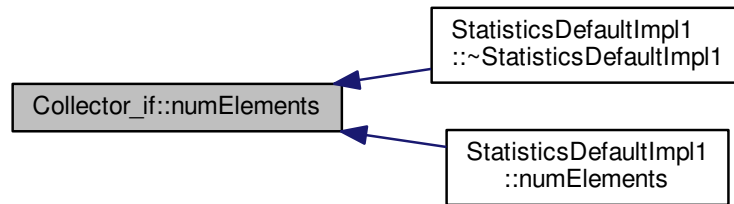
Implemented in [CollectorDatafileDefaultImpl1](#), [CollectorDummyImpl](#), [CollectorDatafileDummyImpl](#), and [CollectorDefaultImpl1](#).

#### 5.5.2.4 virtual unsigned long Collector\_if::numElements ( ) [pure virtual]

Implemented in [CollectorDatafileDefaultImpl1](#), [CollectorDummyImpl](#), [CollectorDatafileDummyImpl](#), and [CollectorDefaultImpl1](#).



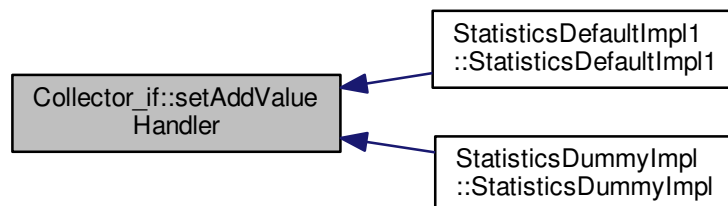
Here is the caller graph for this function:



5.5.2.5 `virtual void Collector_if::setAddValueHandler ( CollectorAddValueHandler addValueHandler )` [pure virtual]

Implemented in [CollectorDatafileDefaultImpl1](#), [CollectorDatafileDummyImpl](#), [CollectorDummyImpl](#), and [CollectorDatafileDefaultImpl1](#).

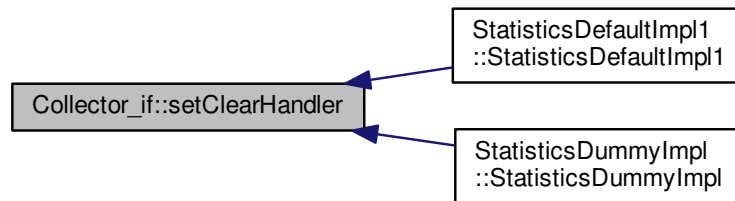
Here is the caller graph for this function:



5.5.2.6 `virtual void Collector_if::setClearHandler ( CollectorClearHandler clearHandler )` [pure virtual]

Implemented in [CollectorDatafileDefaultImpl1](#), [CollectorDatafileDummyImpl](#), [CollectorDummyImpl](#), and [CollectorDatafileDefaultImpl1](#).

Here is the caller graph for this function:



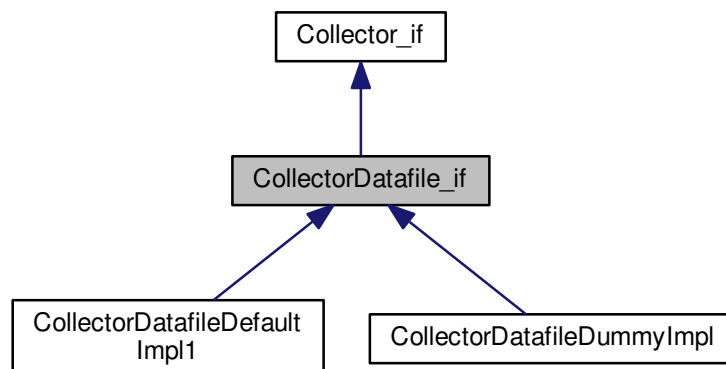
The documentation for this class was generated from the following file:

- [Collector\\_if.h](#)

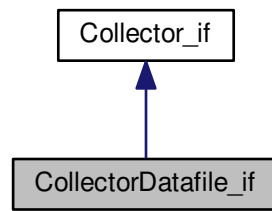
## 5.6 CollectorDatafile\_if Class Reference

```
#include <CollectorDatafile_if.h>
```

Inheritance diagram for CollectorDatafile\_if:



Collaboration diagram for CollectorDatafile\_if:



### Public Member Functions

- virtual double [getValue](#) (unsigned int rank)=0
- virtual void [seekFirstValue](#) ()=0
- virtual double [getNextValue](#) ()=0
- virtual std::string [getDataFilename](#) ()=0
- virtual void [setDataFilename](#) (std::string filename)=0

#### 5.6.1 Detailed Description

Interface for collecting values of a stochastic variable that will be stores in a datafile.

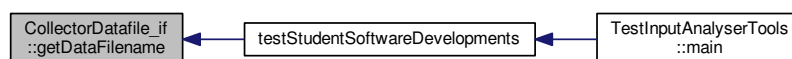
#### 5.6.2 Member Function Documentation

##### 5.6.2.1 virtual std::string CollectorDatafile\_if::getDataFilename ( ) [pure virtual]

Get the next value in the file and advances the pointer

Implemented in [CollectorDatafileDefaultImpl1](#), and [CollectorDatafileDummyImpl](#).

Here is the caller graph for this function:



##### 5.6.2.2 virtual double CollectorDatafile\_if::getNextValue ( ) [pure virtual]

Set the pointer to the first value in the file

Implemented in [CollectorDatafileDefaultImpl1](#), and [CollectorDatafileDummyImpl](#).

5.6.2.3 `virtual double CollectorDatafile_if::getValue ( unsigned int rank ) [pure virtual]`

Implemented in [CollectorDatafileDefaultImpl1](#), and [CollectorDatafileDummyImpl](#).

5.6.2.4 `virtual void CollectorDatafile_if::seekFirstValue ( ) [pure virtual]`

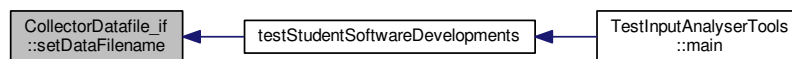
Get a value from a specific position

Implemented in [CollectorDatafileDefaultImpl1](#), and [CollectorDatafileDummyImpl](#).

5.6.2.5 `virtual void CollectorDatafile_if::setDataFilename ( std::string filename ) [pure virtual]`

Implemented in [CollectorDatafileDefaultImpl1](#), and [CollectorDatafileDummyImpl](#).

Here is the caller graph for this function:



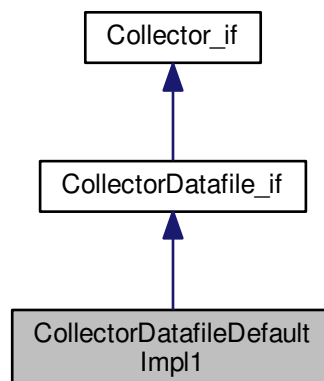
The documentation for this class was generated from the following file:

- [CollectorDatafile\\_if.h](#)

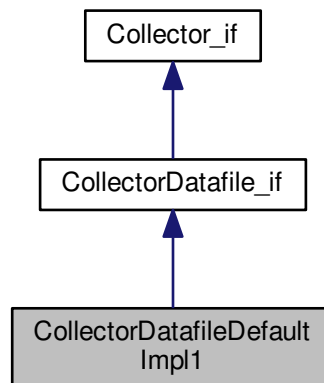
## 5.7 CollectorDatafileDefaultImpl1 Class Reference

```
#include <CollectorDatafileDefaultImpl1.h>
```

Inheritance diagram for `CollectorDatafileDefaultImpl1`:



Collaboration diagram for CollectorDatafileDefaultImpl1:



## Public Member Functions

- [CollectorDatafileDefaultImpl1](#) ()
- [CollectorDatafileDefaultImpl1](#) (const [CollectorDatafileDefaultImpl1](#) &orig)
- virtual [~CollectorDatafileDefaultImpl1](#) ()
- void [clear](#) ()
- void [addValue](#) (double value)
- double [getLastValue](#) ()
- unsigned long [numElements](#) ()
- double [getValue](#) (unsigned int num)
- double [getNextValue](#) ()
- void [seekFirstValue](#) ()
- std::string [getDataFilename](#) ()
- void [setDataFilename](#) (std::string filename)
- void [setAddValueHandler](#) ([CollectorAddValueHandler](#) addValueHandler)
- void [setClearHandler](#) ([CollectorClearHandler](#) clearHandler)

## 5.7.1 Constructor & Destructor Documentation

5.7.1.1 [CollectorDatafileDefaultImpl1::CollectorDatafileDefaultImpl1](#) ( )

5.7.1.2 [CollectorDatafileDefaultImpl1::CollectorDatafileDefaultImpl1](#) ( const [CollectorDatafileDefaultImpl1](#) & orig )

5.7.1.3 [CollectorDatafileDefaultImpl1::~~CollectorDatafileDefaultImpl1](#) ( ) [virtual]

## 5.7.2 Member Function Documentation

5.7.2.1 void [CollectorDatafileDefaultImpl1::addValue](#) ( double value ) [virtual]

Implements [Collector\\_if](#).

5.7.2.2 `void CollectorDatafileDefaultImpl1::clear ( )` [virtual]

Implements [Collector\\_if](#).

5.7.2.3 `std::string CollectorDatafileDefaultImpl1::getDataFilename ( )` [virtual]

Get the next value in the file and advances the pointer

Implements [CollectorDatafile\\_if](#).

5.7.2.4 `double CollectorDatafileDefaultImpl1::getLastValue ( )` [virtual]

Implements [Collector\\_if](#).

5.7.2.5 `double CollectorDatafileDefaultImpl1::getNextValue ( )` [virtual]

Set the pointer to the first value in the file

Implements [CollectorDatafile\\_if](#).

5.7.2.6 `double CollectorDatafileDefaultImpl1::getValue ( unsigned int num )` [virtual]

Implements [CollectorDatafile\\_if](#).

5.7.2.7 `unsigned long CollectorDatafileDefaultImpl1::numElements ( )` [virtual]

Implements [Collector\\_if](#).

5.7.2.8 `void CollectorDatafileDefaultImpl1::seekFirstValue ( )` [virtual]

Get a value from a specific position

Implements [CollectorDatafile\\_if](#).

5.7.2.9 `void CollectorDatafileDefaultImpl1::setAddValueHandler ( CollectorAddValueHandler addValueHandler )`  
[virtual]

Implements [Collector\\_if](#).

5.7.2.10 `void CollectorDatafileDefaultImpl1::setClearHandler ( CollectorClearHandler clearHandler )` [virtual]

Implements [Collector\\_if](#).

5.7.2.11 `void CollectorDatafileDefaultImpl1::setDataFilename ( std::string filename )` [virtual]

Implements [CollectorDatafile\\_if](#).

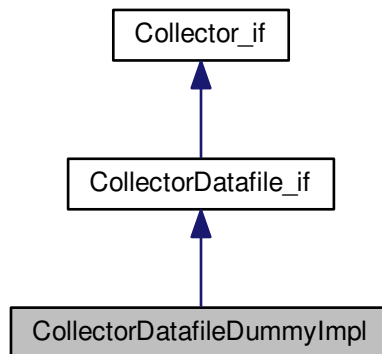
The documentation for this class was generated from the following files:

- [CollectorDatafileDefaultImpl1.h](#)
- [CollectorDatafileDefaultImpl1.cpp](#)

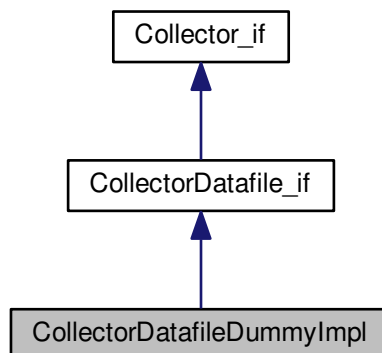
## 5.8 CollectorDatafileDummyImpl Class Reference

```
#include <CollectorDatafileDummyImpl.h>
```

Inheritance diagram for CollectorDatafileDummyImpl:



Collaboration diagram for CollectorDatafileDummyImpl:



## Public Member Functions

- [CollectorDatafileDummyImpl](#) ()
- [CollectorDatafileDummyImpl](#) (const [CollectorDatafileDummyImpl](#) &orig)
- [~CollectorDatafileDummyImpl](#) ()
- void [clear](#) ()
- void [addValue](#) (double value)
- double [getLastValue](#) ()
- unsigned long [numElements](#) ()
- double [getValue](#) (unsigned int num)
- double [getNextValue](#) ()
- void [seekFirstValue](#) ()
- std::string [getDataFilename](#) ()
- void [setDataFilename](#) (std::string filename)
- void [setAddValueHandler](#) ([CollectorAddValueHandler](#) addValueHandler)
- void [setClearHandler](#) ([CollectorClearHandler](#) clearHandler)

## 5.8.1 Constructor & Destructor Documentation

5.8.1.1 [CollectorDatafileDummyImpl::CollectorDatafileDummyImpl](#) ( )

5.8.1.2 [CollectorDatafileDummyImpl::CollectorDatafileDummyImpl](#) ( const [CollectorDatafileDummyImpl](#) & orig )

5.8.1.3 [CollectorDatafileDummyImpl::~~CollectorDatafileDummyImpl](#) ( )

## 5.8.2 Member Function Documentation

5.8.2.1 void [CollectorDatafileDummyImpl::addValue](#) ( double *value* ) [virtual]

Implements [Collector\\_if](#).

5.8.2.2 void [CollectorDatafileDummyImpl::clear](#) ( ) [virtual]

Implements [Collector\\_if](#).

5.8.2.3 std::string [CollectorDatafileDummyImpl::getDataFilename](#) ( ) [virtual]

Get the next value in the file and advances the pointer

Implements [CollectorDatafile\\_if](#).

5.8.2.4 double [CollectorDatafileDummyImpl::getLastValue](#) ( ) [virtual]

Implements [Collector\\_if](#).



5.8.2.5 `double CollectorDatafileDummyImpl::getNextValue ( )` [virtual]

Set the pointer to the first value in the file

Implements [CollectorDatafile\\_if](#).

5.8.2.6 `double CollectorDatafileDummyImpl::getValue ( unsigned int num )` [virtual]

Implements [CollectorDatafile\\_if](#).

5.8.2.7 `unsigned long CollectorDatafileDummyImpl::numElements ( )` [virtual]

Implements [Collector\\_if](#).

5.8.2.8 `void CollectorDatafileDummyImpl::seekFirstValue ( )` [virtual]

Get a value from a specific position

Implements [CollectorDatafile\\_if](#).

5.8.2.9 `void CollectorDatafileDummyImpl::setAddValueHandler ( CollectorAddValueHandler addValueHandler )`  
[virtual]

Implements [Collector\\_if](#).

5.8.2.10 `void CollectorDatafileDummyImpl::setClearHandler ( CollectorClearHandler clearHandler )` [virtual]

Implements [Collector\\_if](#).

5.8.2.11 `void CollectorDatafileDummyImpl::setDataFilename ( std::string filename )` [virtual]

Implements [CollectorDatafile\\_if](#).

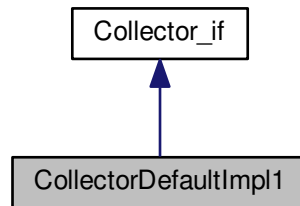
The documentation for this class was generated from the following files:

- [CollectorDatafileDummyImpl.h](#)
- [CollectorDatafileDummyImpl.cpp](#)

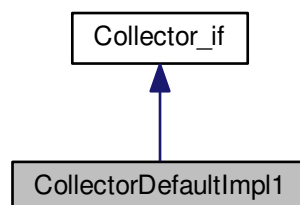
## 5.9 CollectorDefaultImpl1 Class Reference

```
#include <CollectorDefaultImpl1.h>
```

Inheritance diagram for CollectorDefaultImpl1:



Collaboration diagram for CollectorDefaultImpl1:



### Public Member Functions

- [CollectorDefaultImpl1](#) ()
- [CollectorDefaultImpl1](#) (const [CollectorDefaultImpl1](#) &orig)
- virtual [~CollectorDefaultImpl1](#) ()
- void [clear](#) ()
- void [addValue](#) (double value)
- double [getLastValue](#) ()
- unsigned long [numElements](#) ()
- void [setAddValueHandler](#) ([CollectorAddValueHandler](#) addValueHandler)
- void [setClearHandler](#) ([CollectorClearHandler](#) clearHandler)

## 5.9.1 Constructor & Destructor Documentation

5.9.1.1 `CollectorDefaultImpl1::CollectorDefaultImpl1 ( )`

5.9.1.2 `CollectorDefaultImpl1::CollectorDefaultImpl1 ( const CollectorDefaultImpl1 & orig )`

5.9.1.3 `CollectorDefaultImpl1::~~CollectorDefaultImpl1 ( )` [virtual]

## 5.9.2 Member Function Documentation

5.9.2.1 `void CollectorDefaultImpl1::addValue ( double value )` [virtual]

Implements [Collector\\_if](#).

5.9.2.2 `void CollectorDefaultImpl1::clear ( )` [virtual]

Implements [Collector\\_if](#).

5.9.2.3 `double CollectorDefaultImpl1::getLastValue ( )` [virtual]

Implements [Collector\\_if](#).

5.9.2.4 `unsigned long CollectorDefaultImpl1::numElements ( )` [virtual]

Implements [Collector\\_if](#).

5.9.2.5 `void CollectorDefaultImpl1::setAddValueHandler ( CollectorAddValueHandler addValueHandler )` [virtual]

Implements [Collector\\_if](#).

5.9.2.6 `void CollectorDefaultImpl1::setClearHandler ( CollectorClearHandler clearHandler )` [virtual]

Implements [Collector\\_if](#).

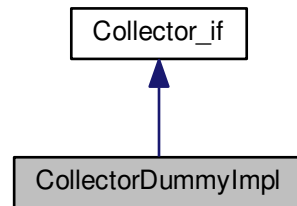
The documentation for this class was generated from the following files:

- [CollectorDefaultImpl1.h](#)
- [CollectorDefaultImpl1.cpp](#)

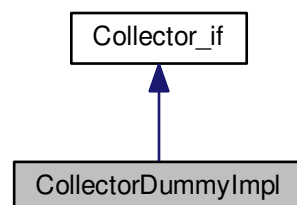
## 5.10 CollectorDummyImpl Class Reference

```
#include <CollectorDummyImpl.h>
```

Inheritance diagram for CollectorDummyImpl:



Collaboration diagram for CollectorDummyImpl:



### Public Member Functions

- [CollectorDummyImpl](#) ()
- [CollectorDummyImpl](#) (const [CollectorDummyImpl](#) &orig)
- [~CollectorDummyImpl](#) ()
- void [clear](#) ()
- void [addValue](#) (double value)
- double [getLastValue](#) ()
- unsigned long [numElements](#) ()
- void [setAddValueHandler](#) ([CollectorAddValueHandler](#) addValueHandler)
- void [setClearHandler](#) ([CollectorClearHandler](#) clearHandler)

### 5.10.1 Constructor & Destructor Documentation

5.10.1.1 `CollectorDummyImpl::CollectorDummyImpl ( )`

5.10.1.2 `CollectorDummyImpl::CollectorDummyImpl ( const CollectorDummyImpl & orig )`

5.10.1.3 `CollectorDummyImpl::~~CollectorDummyImpl ( )`

### 5.10.2 Member Function Documentation

5.10.2.1 `void CollectorDummyImpl::addValue ( double value )` [virtual]

Implements [Collector\\_if](#).

5.10.2.2 `void CollectorDummyImpl::clear ( )` [virtual]

Implements [Collector\\_if](#).

5.10.2.3 `double CollectorDummyImpl::getLastValue ( )` [virtual]

Implements [Collector\\_if](#).

5.10.2.4 `unsigned long CollectorDummyImpl::numElements ( )` [virtual]

Implements [Collector\\_if](#).

5.10.2.5 `void CollectorDummyImpl::setAddValueHandler ( CollectorAddValueHandler addValueHandler )` [virtual]

Implements [Collector\\_if](#).

5.10.2.6 `void CollectorDummyImpl::setClearHandler ( CollectorClearHandler clearHandler )` [virtual]

Implements [Collector\\_if](#).

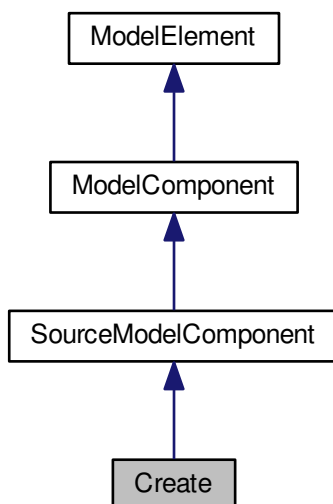
The documentation for this class was generated from the following files:

- [CollectorDummyImpl.h](#)
- [CollectorDummyImpl.cpp](#)

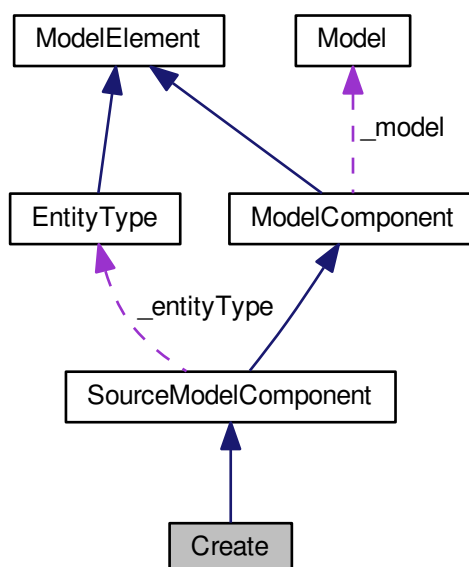
## 5.11 Create Class Reference

```
#include <Create.h>
```

Inheritance diagram for Create:



Collaboration diagram for Create:



## Public Member Functions

- [Create](#) ([Model](#) \*model)
- [Create](#) (const [Create](#) &orig)
- virtual [~Create](#) ()
- virtual std::string [show](#) ()

## Protected Member Functions

- virtual void [\\_execute](#) ([Entity](#) \*entity)
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

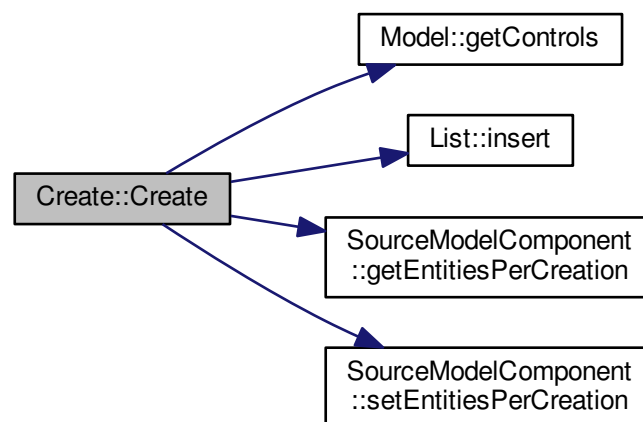
### 5.11.1 Detailed Description

[Create](#) is the most basic component to include the first entities into the model, and therefore is a source component (derived from [SourceModelComponent](#))

### 5.11.2 Constructor & Destructor Documentation

#### 5.11.2.1 [Create::Create](#) ( [Model](#) \* *model* )

Here is the call graph for this function:



5.11.2.2 `Create::Create ( const Create & orig )`

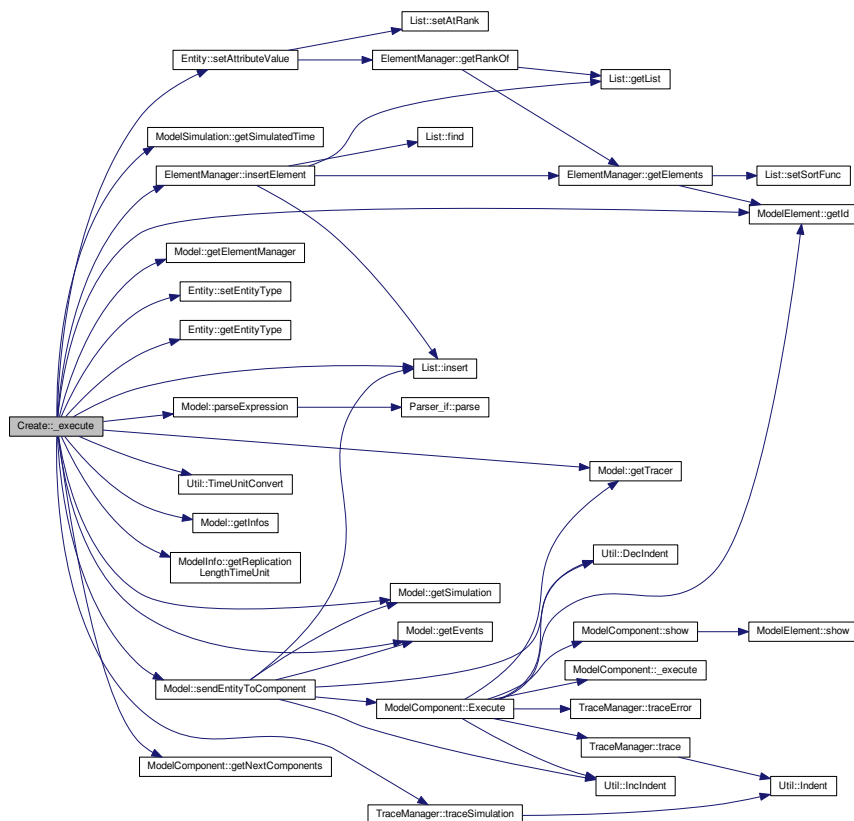
5.11.2.3 `Create::~~Create ( )` [virtual]

### 5.11.3 Member Function Documentation

5.11.3.1 `void Create::_execute ( Entity * entity )` [protected],[virtual]

Implements [ModelComponent](#).

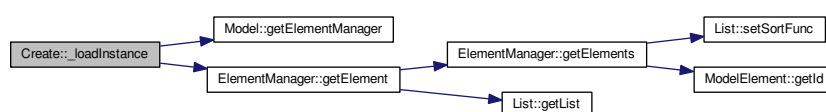
Here is the call graph for this function:



5.11.3.2 `void Create::_loadInstance ( std::list< std::string > words )` [protected],[virtual]

Implements [ModelElement](#).

Here is the call graph for this function:

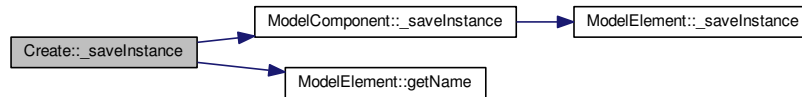




5.11.3.3 `std::list< std::string > * Create::_saveInstance ( )` [protected], [virtual]

Reimplemented from [ModelComponent](#).

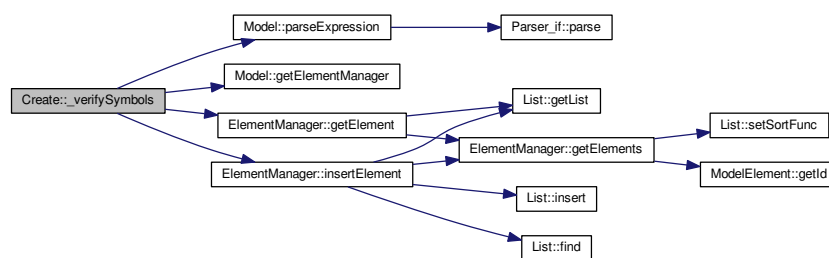
Here is the call graph for this function:



5.11.3.4 `bool Create::_verifySymbols ( std::string * errorMessage )` [protected], [virtual]

Implements [ModelElement](#).

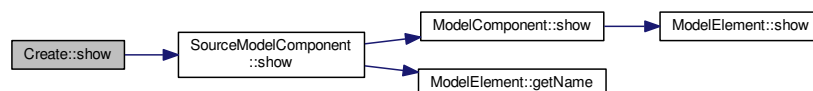
Here is the call graph for this function:



5.11.3.5 `std::string Create::show ( )` [virtual]

Reimplemented from [SourceModelComponent](#).

Here is the call graph for this function:



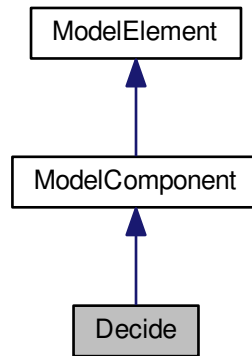
The documentation for this class was generated from the following files:

- [Create.h](#)
- [Create.cpp](#)

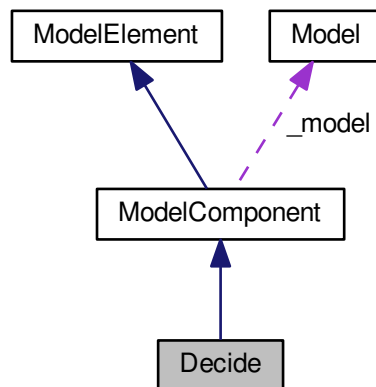
## 5.12 Decide Class Reference

```
#include <Decide.h>
```

Inheritance diagram for Decide:



Collaboration diagram for Decide:



### Public Member Functions

- [Decide](#) ([Model](#) \*model)
- [Decide](#) (const [Decide](#) &orig)
- virtual [~Decide](#) ()
- [List](#)< std::string > \* [getConditions](#) () const
- virtual std::string [show](#) ()

## Protected Member Functions

- virtual void [\\_execute](#) ([Entity](#) \*entity)
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.12.1 Constructor & Destructor Documentation

5.12.1.1 `Decide::Decide ( Model * model )`

5.12.1.2 `Decide::Decide ( const Decide & orig )`

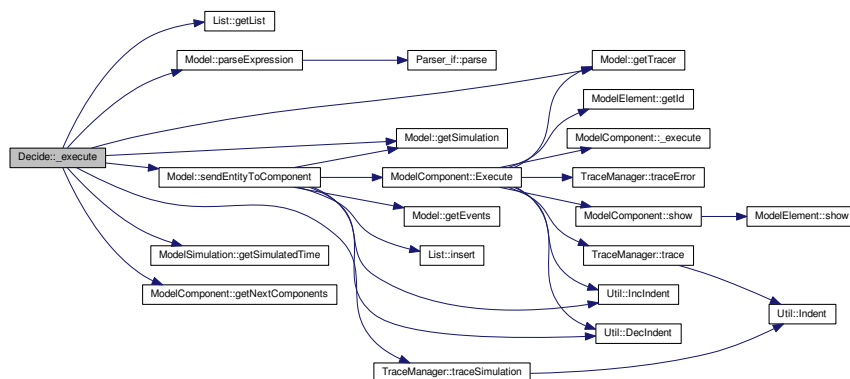
5.12.1.3 `Decide::~Decide ( )` `[virtual]`

### 5.12.2 Member Function Documentation

5.12.2.1 `void Decide::\_execute ( Entity * entity )` `[protected]`, `[virtual]`

Implements [ModelComponent](#).

Here is the call graph for this function:



5.12.2.2 `void Decide::\_loadInstance ( std::list< std::string > words )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

5.12.2.3 `std::list< std::string > * Decide::\_saveInstance ( )` `[protected]`, `[virtual]`

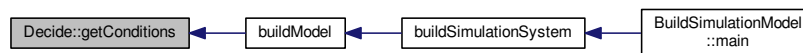
Reimplemented from [ModelComponent](#).

5.12.2.4 `bool Decide::_verifySymbols ( std::string * errorMessage )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

5.12.2.5 `List< std::string > * Decide::getConditions ( )` `const`

Here is the caller graph for this function:



5.12.2.6 `std::string Decide::show ( )` `[virtual]`

Reimplemented from [ModelComponent](#).

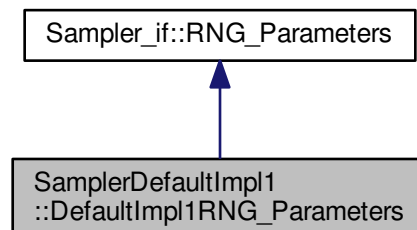
The documentation for this class was generated from the following files:

- [Decide.h](#)
- [Decide.cpp](#)

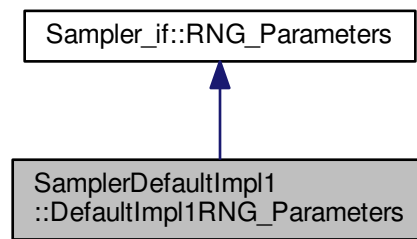
## 5.13 SamplerDefaultImpl1::DefaultImpl1RNG\_Parameters Class Reference

```
#include <SamplerDefaultImpl1.h>
```

Inheritance diagram for SamplerDefaultImpl1::DefaultImpl1RNG\_Parameters:



Collaboration diagram for `SamplerDefaultImpl1::DefaultImpl1RNG_Parameters`:



### Public Attributes

- unsigned int `seed` = 666
- unsigned int `module` = 2147483647
- unsigned int `multiplier` = 950706376

### 5.13.1 Member Data Documentation

5.13.1.1 unsigned int `SamplerDefaultImpl1::DefaultImpl1RNG_Parameters::module` = 2147483647

5.13.1.2 unsigned int `SamplerDefaultImpl1::DefaultImpl1RNG_Parameters::multiplier` = 950706376

5.13.1.3 unsigned int `SamplerDefaultImpl1::DefaultImpl1RNG_Parameters::seed` = 666

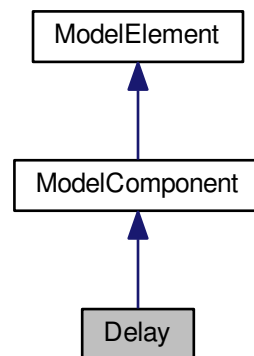
The documentation for this class was generated from the following file:

- [SamplerDefaultImpl1.h](#)

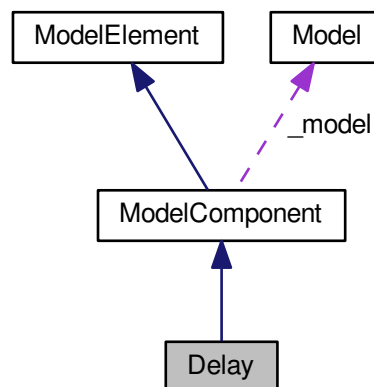
## 5.14 Delay Class Reference

```
#include <Delay.h>
```

Inheritance diagram for Delay:



Collaboration diagram for Delay:



## Public Member Functions

- [Delay](#) ([Model](#) \*model)
- [Delay](#) (const [Delay](#) &orig)
- virtual [~Delay](#) ()
- void [setDelayExpression](#) (std::string \_delayExpression)
- std::string [getDelayExpression](#) () const
- void [setDelayTimeUnit](#) ([Util::TimeUnit](#) \_delayTimeUnit)
- [Util::TimeUnit](#) [getDelayTimeUnit](#) () const
- virtual std::string [show](#) ()

## Protected Member Functions

- virtual void [\\_execute](#) ([Entity](#) \*entity)
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.14.1 Constructor & Destructor Documentation

5.14.1.1 [Delay::Delay](#) ( [Model](#) \* *model* )

5.14.1.2 [Delay::Delay](#) ( const [Delay](#) & *orig* )

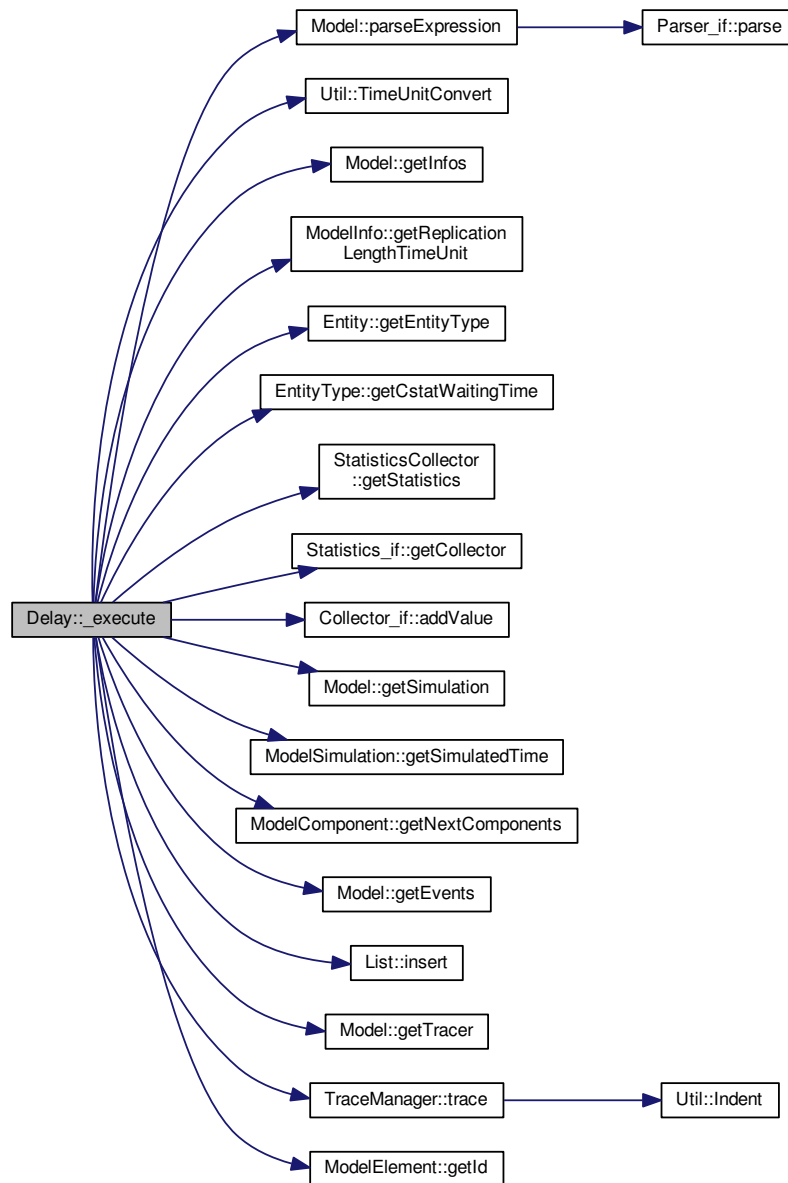
5.14.1.3 [Delay::~Delay](#) ( ) [virtual]

### 5.14.2 Member Function Documentation

5.14.2.1 void [Delay::\\_execute](#) ( [Entity](#) \* *entity* ) [protected],[virtual]

Implements [ModelComponent](#).

Here is the call graph for this function:



5.14.2.2 `void Delay::_loadInstance ( std::list< std::string > words )` [protected],[virtual]

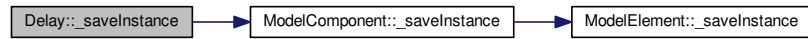
Implements [ModelElement](#).

5.14.2.3 `std::list< std::string > * Delay::_saveInstance ( )` [protected],[virtual]

Reimplemented from [ModelComponent](#).



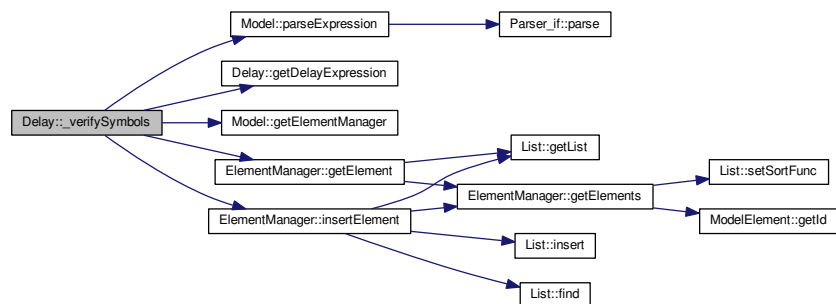
Here is the call graph for this function:



#### 5.14.2.4 `bool Delay::_verifySymbols ( std::string * errorMessage )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

Here is the call graph for this function:



#### 5.14.2.5 `std::string Delay::getDelayExpression ( )` `const`

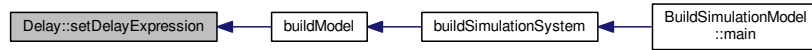
Here is the caller graph for this function:



#### 5.14.2.6 `Util::TimeUnit Delay::getDelayTimeUnit ( )` `const`

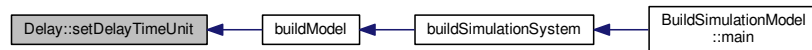
#### 5.14.2.7 void Delay::setDelayExpression ( std::string *\_delayExpression* )

Here is the caller graph for this function:



#### 5.14.2.8 void Delay::setDelayTimeUnit ( Util::TimeUnit *\_delayTimeUnit* )

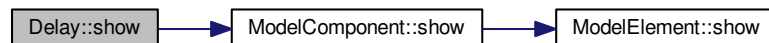
Here is the caller graph for this function:



#### 5.14.2.9 std::string Delay::show ( ) [virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



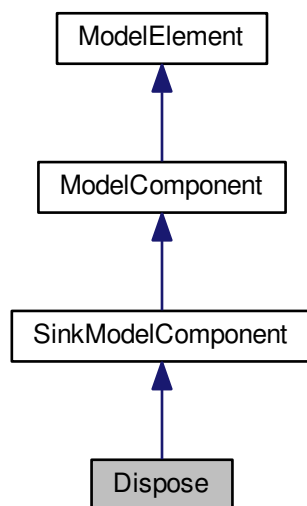
The documentation for this class was generated from the following files:

- [Delay.h](#)
- [Delay.cpp](#)

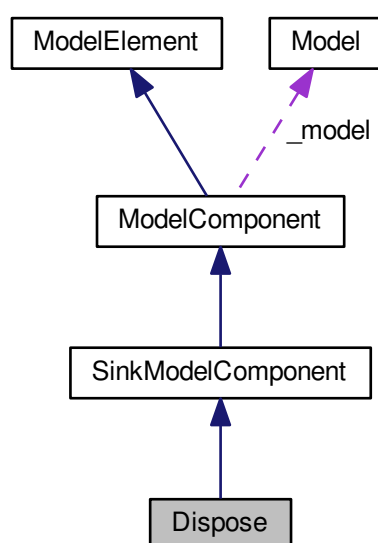
## 5.15 Dispose Class Reference

```
#include <Dispose.h>
```

Inheritance diagram for Dispose:



Collaboration diagram for Dispose:



## Public Member Functions

- [Dispose](#) ([Model](#) \*model)
- [Dispose](#) (const [Dispose](#) &orig)
- virtual [~Dispose](#) ()
- virtual std::string [show](#) ()
- unsigned int [getNumberOut](#) () const
- virtual void [setCollectStatistics](#) (bool \_collectStatistics)
- virtual bool [isCollectStatistics](#) () const

## Protected Member Functions

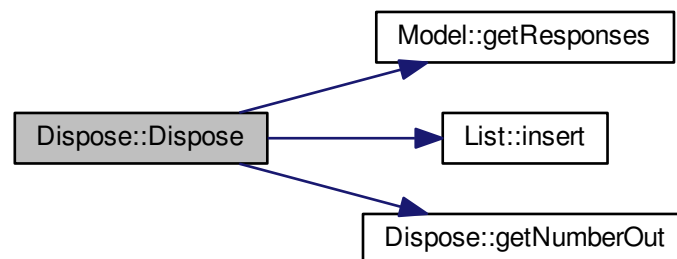
- virtual void [\\_\\_execute](#) ([Entity](#) \*entity)
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.15.1 Constructor & Destructor Documentation

#### 5.15.1.1 [Dispose::Dispose](#) ( [Model](#) \* *model* )

Here is the call graph for this function:



5.15.1.2 `Dispose::Dispose ( const Dispose & orig )`

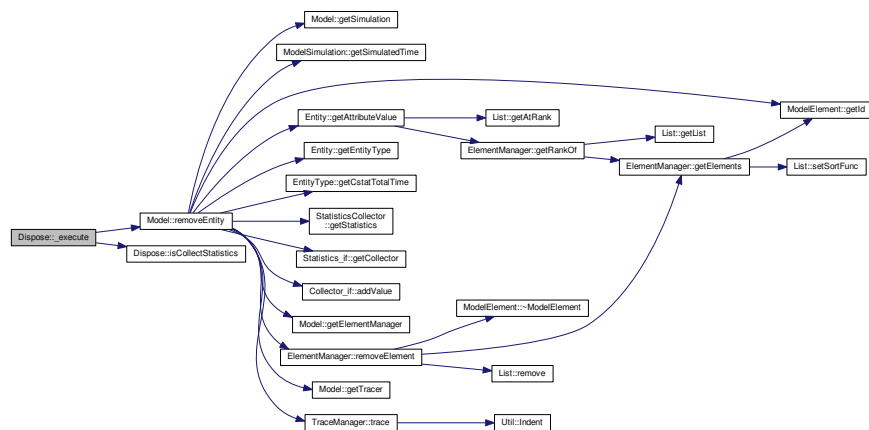
5.15.1.3 `Dispose::~~Dispose ( )` [virtual]

## 5.15.2 Member Function Documentation

5.15.2.1 `void Dispose::_execute ( Entity * entity )` [protected],[virtual]

Implements [ModelComponent](#).

Here is the call graph for this function:



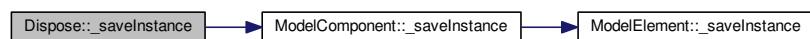
5.15.2.2 `void Dispose::_loadInstance ( std::list< std::string > words )` [protected],[virtual]

Implements [ModelElement](#).

5.15.2.3 `std::list< std::string > * Dispose::_saveInstance ( )` [protected],[virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



5.15.2.4 `bool Dispose::_verifySymbols ( std::string * errorMessage )` [protected],[virtual]

Implements [ModelElement](#).

#### 5.15.2.5 unsigned int Dispose::getNumberOut ( ) const

Here is the caller graph for this function:



#### 5.15.2.6 bool Dispose::isCollectStatistics ( ) const [virtual]

Here is the caller graph for this function:

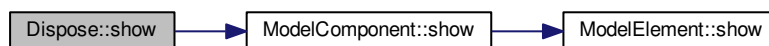


#### 5.15.2.7 void Dispose::setCollectStatistics ( bool *\_collectStatistics* ) [virtual]

#### 5.15.2.8 std::string Dispose::show ( ) [virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



The documentation for this class was generated from the following files:

- [Dispose.h](#)
- [Dispose.cpp](#)

## 5.16 ElementManager Class Reference

```
#include <ElementManager.h>
```

### Public Member Functions

- [ElementManager](#) ([Model](#) \*model)
- [ElementManager](#) (const [ElementManager](#) &orig)
- virtual [~ElementManager](#) ()
- bool [insertElement](#) (std::string infraTypename, [ModelElement](#) \*infra)
- void [removeElement](#) (std::string infraTypename, [ModelElement](#) \*infra)
- [ModelElement](#) \* [getElement](#) (std::string infraTypename, [Util::identification](#) id)
- [ModelElement](#) \* [getElement](#) (std::string infraTypename, std::string name)
- unsigned int [getNumberOfElements](#) (std::string infraTypename)
- int [getRankOf](#) (std::string infraTypename, std::string name)  
*returns the position (1st position=0) of the element if found, or negative value if not found*
- std::list< std::string > \* [getElementTypenames](#) () const
- [List](#)< [ModelElement](#) \* > \* [getElements](#) (std::string infraTypename) const
- void [show](#) ()

### 5.16.1 Detailed Description

The [ElementManager](#) is responsible for inserting and removing elements ([ModelElement](#)) used by components, in a consistent way. TO FIX: No direct access for insertion or deletion should be allow

### 5.16.2 Constructor & Destructor Documentation

#### 5.16.2.1 [ElementManager::ElementManager](#) ( [Model](#) \* model )

Elements are organized as a map from a string (key), the type of an element, and a list of elements of that type

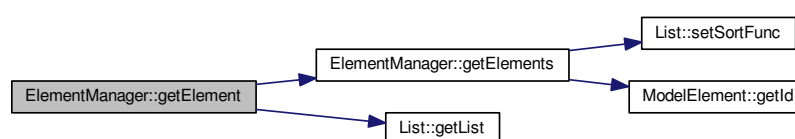
#### 5.16.2.2 [ElementManager::ElementManager](#) ( const [ElementManager](#) & orig )

#### 5.16.2.3 [ElementManager::~~ElementManager](#) ( ) [virtual]

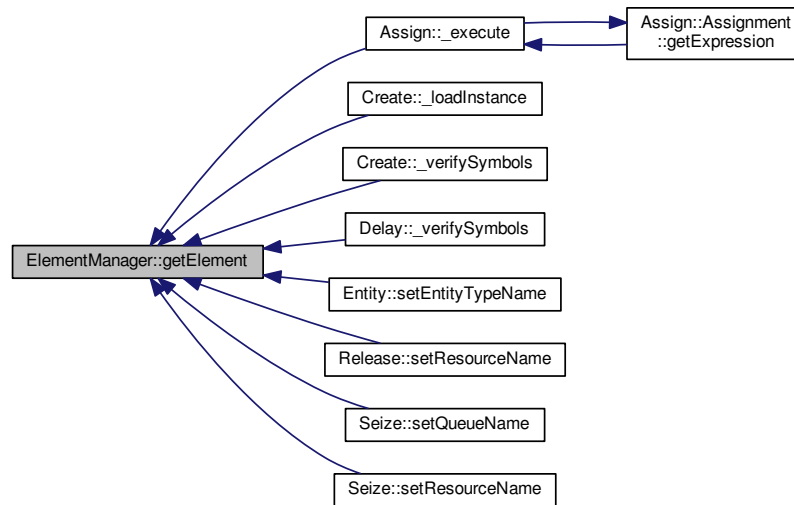
### 5.16.3 Member Function Documentation

#### 5.16.3.1 [ModelElement](#) \* [ElementManager::getElement](#) ( std::string infraTypename, [Util::identification](#) id )

Here is the call graph for this function:

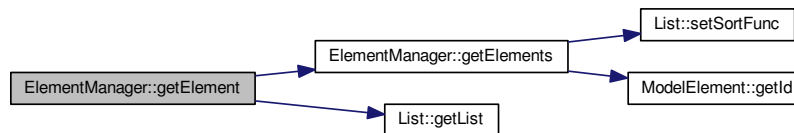


Here is the caller graph for this function:



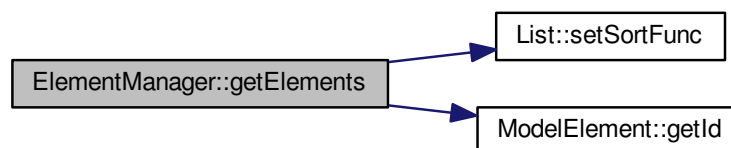
#### 5.16.3.2 `ModelElement * ElementManager::getElement ( std::string infraTypename, std::string name )`

Here is the call graph for this function:



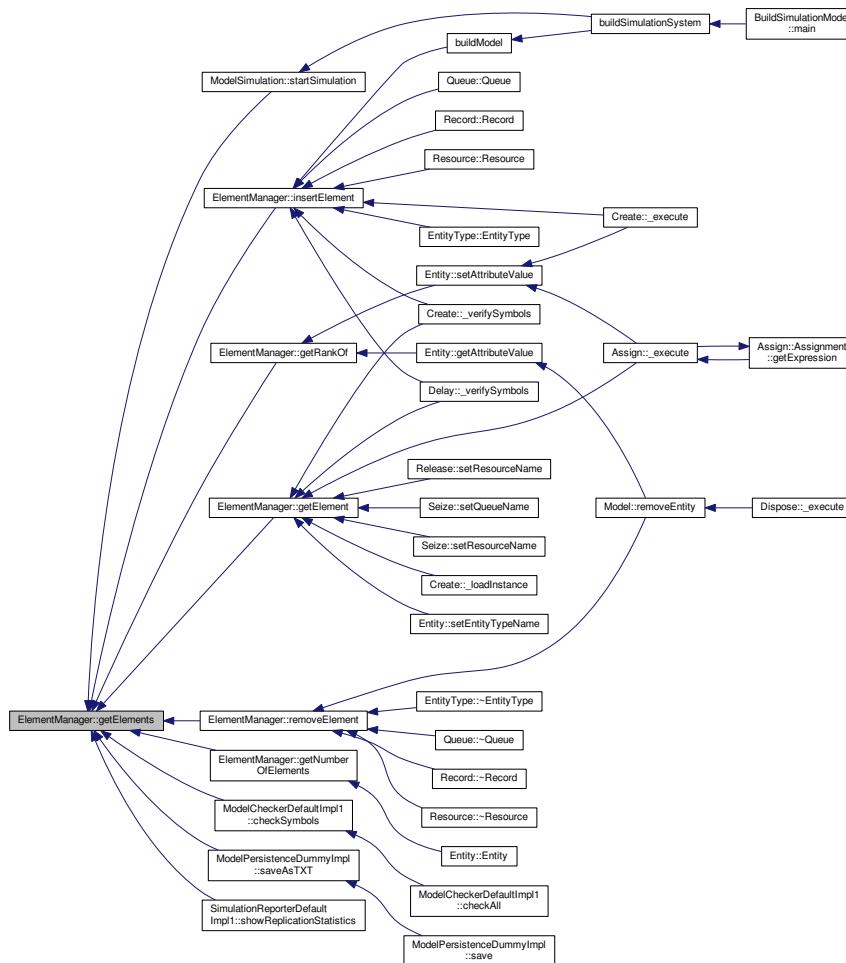
#### 5.16.3.3 `List< ModelElement * > * ElementManager::getElements ( std::string infraTypename ) const`

Here is the call graph for this function:



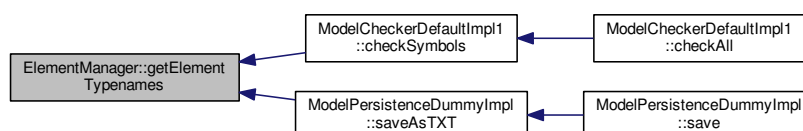


Here is the caller graph for this function:



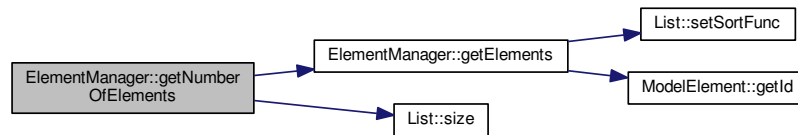
#### 5.16.3.4 `std::list< std::string > * ElementManager::getElementTypenames ( ) const`

Here is the caller graph for this function:

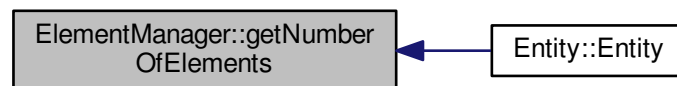


#### 5.16.3.5 unsigned int ElementManager::getNumberOfElements ( std::string *infraTypename* )

Here is the call graph for this function:



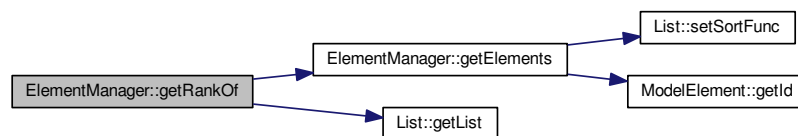
Here is the caller graph for this function:



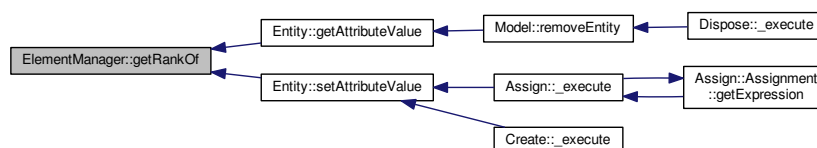
#### 5.16.3.6 int ElementManager::getRankOf ( std::string *infraTypename*, std::string *name* )

returns the position (1st position=0) of the element if found, or negative value if not found

Here is the call graph for this function:

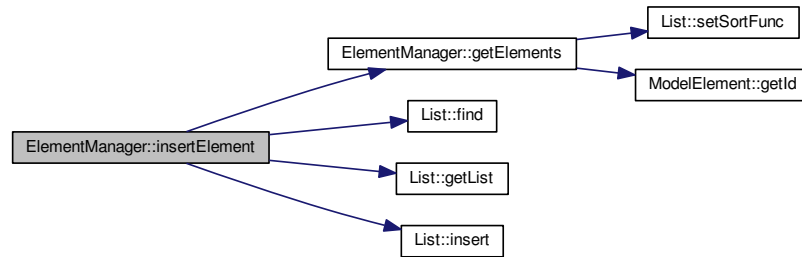


Here is the caller graph for this function:

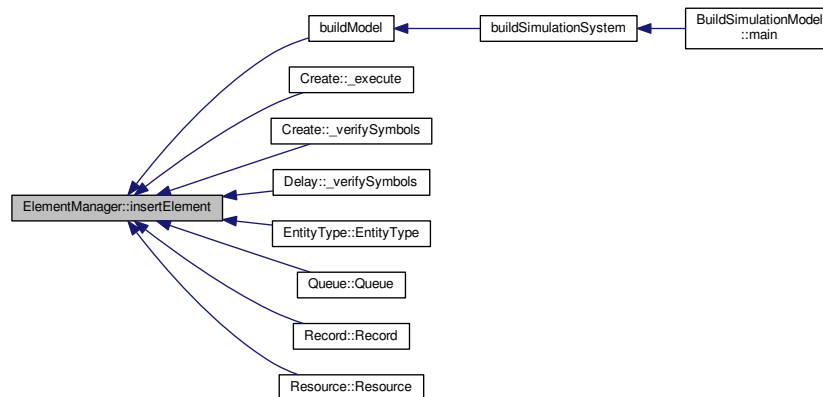


#### 5.16.3.7 bool ElementManager::insertElement ( std::string *infraTypename*, ModelElement \* *infra* )

Here is the call graph for this function:

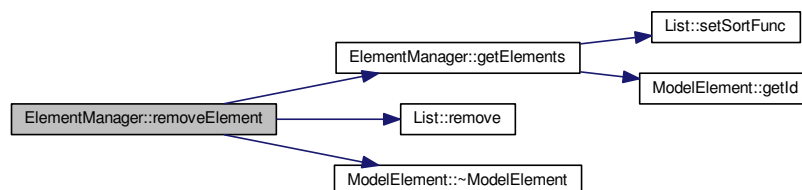


Here is the caller graph for this function:

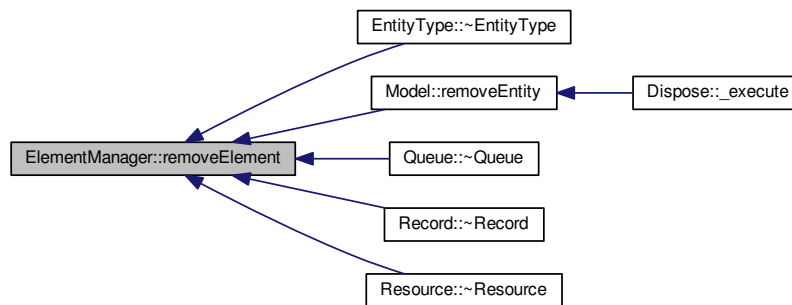


#### 5.16.3.8 void ElementManager::removeElement ( std::string *infraTypename*, ModelElement \* *infra* )

Here is the call graph for this function:

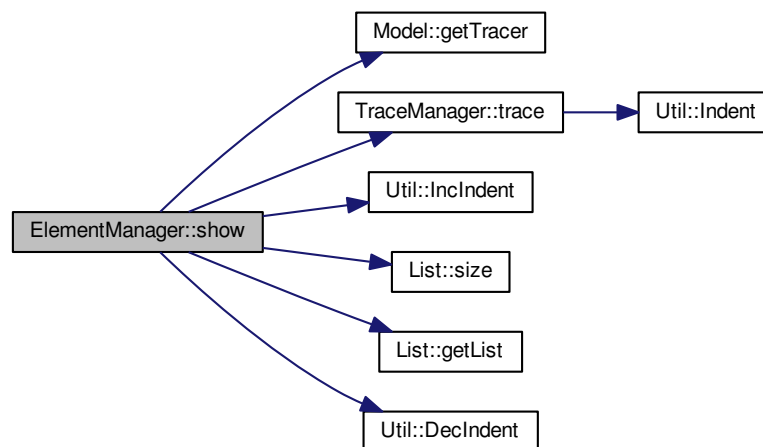


Here is the caller graph for this function:



#### 5.16.3.9 void ElementManager::show ( )

Here is the call graph for this function:



The documentation for this class was generated from the following files:

- [ElementManager.h](#)
- [ElementManager.cpp](#)

## 5.17 ElementManager\_if Class Reference

```
#include <ElementManager_if.h>
```

## Public Member Functions

- [ElementManager\\_if](#) ()
- [ElementManager\\_if](#) (const [ElementManager\\_if](#) &orig)
- virtual [~ElementManager\\_if](#) ()

## 5.17.1 Constructor & Destructor Documentation

5.17.1.1 [ElementManager\\_if::ElementManager\\_if](#) ( )

5.17.1.2 [ElementManager\\_if::ElementManager\\_if](#) ( const [ElementManager\\_if](#) & orig )

5.17.1.3 virtual [ElementManager\\_if::~~ElementManager\\_if](#) ( ) [virtual]

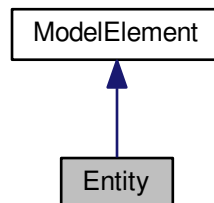
The documentation for this class was generated from the following file:

- [ElementManager\\_if.h](#)

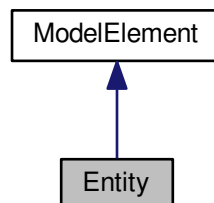
## 5.18 Entity Class Reference

```
#include <Entity.h>
```

Inheritance diagram for Entity:



Collaboration diagram for Entity:



## Public Member Functions

- [Entity](#) ([ElementManager](#) \*elements)
- [Entity](#) (const [Entity](#) &orig)
- virtual [~Entity](#) ()
- virtual std::string [show](#) ()
- void [setEntityType](#) (std::string entityType) throw ()
- std::string [getEntityType](#) () const
- void [setEntityType](#) ([EntityType](#) \*entityType)
- [EntityType](#) \* [getEntityType](#) () const
- double [getAttributeValue](#) (std::string attributeName)
- void [setAttributeValue](#) (std::string attributeName, double value)

## Protected Member Functions

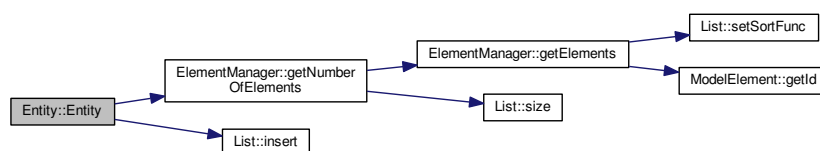
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.18.1 Constructor & Destructor Documentation

#### 5.18.1.1 Entity::Entity ( [ElementManager](#) \* *elements* )

Here is the call graph for this function:



#### 5.18.1.2 Entity::Entity ( const [Entity](#) & *orig* )

#### 5.18.1.3 Entity::~~Entity ( ) [virtual]

### 5.18.2 Member Function Documentation

#### 5.18.2.1 void Entity::\_loadInstance ( std::list< std::string > *words* ) [protected], [virtual]

Implements [ModelElement](#).

5.18.2.2 `std::list< std::string > * Entity::_saveInstance ( )` [protected],[virtual]

Reimplemented from [ModelElement](#).

Here is the call graph for this function:

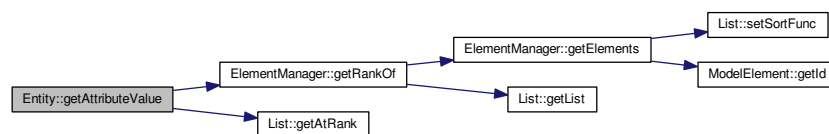


5.18.2.3 `bool Entity::_verifySymbols ( std::string * errorMessage )` [protected],[virtual]

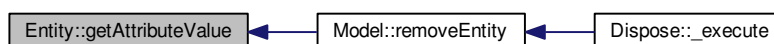
Implements [ModelElement](#).

5.18.2.4 `double Entity::getAttributeValue ( std::string attributeName )`

Here is the call graph for this function:

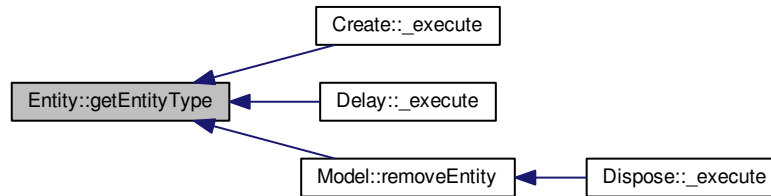


Here is the caller graph for this function:



#### 5.18.2.5 Entity \* Entity::getEntityType ( ) const

Here is the caller graph for this function:



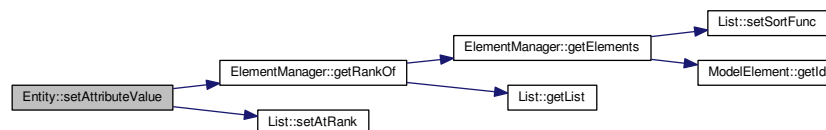
#### 5.18.2.6 std::string Entity::getEntityTypeName ( ) const

Here is the call graph for this function:

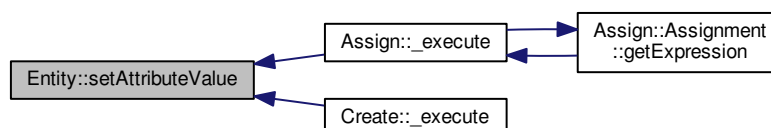


#### 5.18.2.7 void Entity::setAttributeValue ( std::string attributeName, double value )

Here is the call graph for this function:



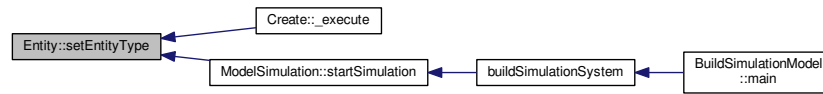
Here is the caller graph for this function:





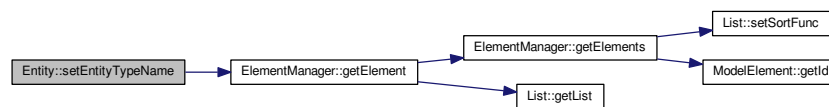
#### 5.18.2.8 void Entity::setEntityType ( EntityType \* entityType )

Here is the caller graph for this function:



#### 5.18.2.9 void Entity::setEntityTypeName ( std::string entityTypeName ) throw )

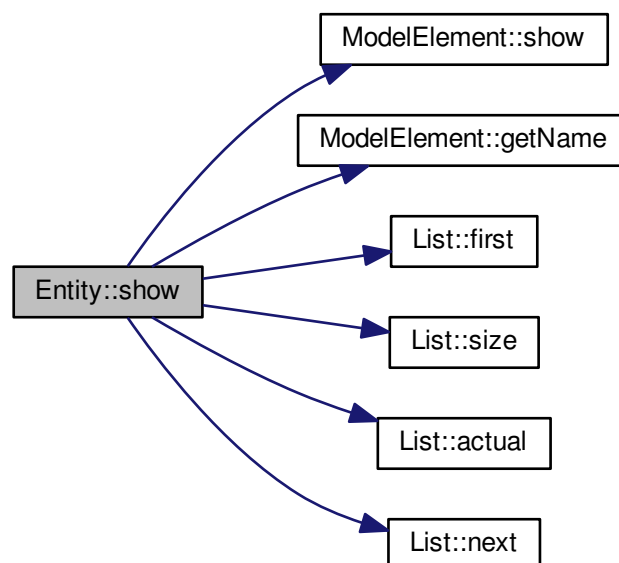
Here is the call graph for this function:



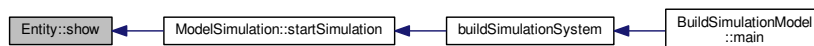
#### 5.18.2.10 std::string Entity::show ( ) [virtual]

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



Here is the caller graph for this function:



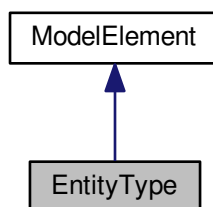
The documentation for this class was generated from the following files:

- [Entity.h](#)
- [Entity.cpp](#)

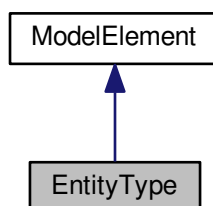
## 5.19 EntityType Class Reference

```
#include <EntityType.h>
```

Inheritance diagram for EntityType:



Collaboration diagram for EntityType:



## Public Member Functions

- [EntityType](#) ([ElementManager](#) \*elemManager)
- [EntityType](#) ([ElementManager](#) \*elemManager, std::string name)
- [EntityType](#) (const [EntityType](#) &orig)
- virtual [~EntityType](#) ()
- virtual std::string [show](#) ()
- void [setInitialWaitingCost](#) (double \_initialWaitingCost)
- double [getInitialWaitingCost](#) () const
- void [setInitialOtherCost](#) (double \_initialOtherCost)
- double [getInitialOtherCost](#) () const
- void [setInitialNVACost](#) (double \_initialNVACost)
- double [getInitialNVACost](#) () const
- void [setInitialVACost](#) (double \_initialVACost)
- double [getInitialVACost](#) () const
- void [setInitialPicture](#) (std::string \_initialPicture)
- std::string [getInitialPicture](#) () const
- [StatisticsCollector](#) \* [getCstatTotalTime](#) () const
- [StatisticsCollector](#) \* [getCstatNVATime](#) () const
- [StatisticsCollector](#) \* [getCstatVATime](#) () const
- [StatisticsCollector](#) \* [getCstatOtherTime](#) () const
- [StatisticsCollector](#) \* [getCstatTransferTime](#) () const
- [StatisticsCollector](#) \* [getCstatWaitingTime](#) () const

## Protected Member Functions

- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

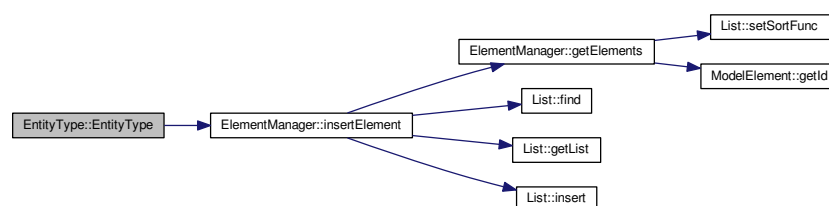
## Additional Inherited Members

### 5.19.1 Constructor & Destructor Documentation

#### 5.19.1.1 [EntityType::EntityType](#) ( [ElementManager](#) \* *elemManager* )

#### 5.19.1.2 [EntityType::EntityType](#) ( [ElementManager](#) \* *elemManager*, std::string *name* )

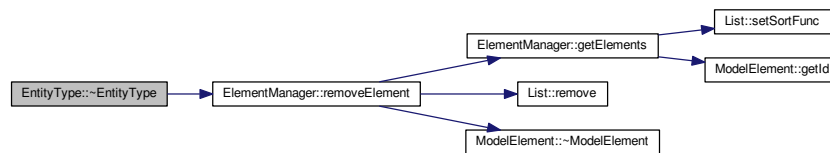
Here is the call graph for this function:



5.19.1.3 `EntityType::EntityType ( const EntityType & orig )`

5.19.1.4 `EntityType::~EntityType ( ) [virtual]`

Here is the call graph for this function:



## 5.19.2 Member Function Documentation

5.19.2.1 `void EntityType::_loadInstance ( std::list< std::string > words ) [protected],[virtual]`

Implements [ModelElement](#).

5.19.2.2 `std::list< std::string > * EntityType::_saveInstance ( ) [protected],[virtual]`

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



5.19.2.3 `bool EntityType::_verifySymbols ( std::string * errorMessage ) [protected],[virtual]`

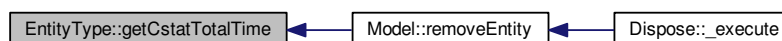
Implements [ModelElement](#).

5.19.2.4 `StatisticsCollector * EntityType::getCstatNVATime ( ) const`

5.19.2.5 `StatisticsCollector * EntityType::getCstatOtherTime ( ) const`

5.19.2.6 `StatisticsCollector * EntityType::getCstatTotalTime ( ) const`

Here is the caller graph for this function:

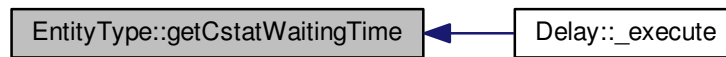


5.19.2.7 **StatisticsCollector** \* EntityType::getCstatTransferTime ( ) const

5.19.2.8 **StatisticsCollector** \* EntityType::getCstatVATime ( ) const

5.19.2.9 **StatisticsCollector** \* EntityType::getCstatWaitingTime ( ) const

Here is the caller graph for this function:



5.19.2.10 double EntityType::getInitialNVACost ( ) const

5.19.2.11 double EntityType::getInitialOtherCost ( ) const

5.19.2.12 std::string EntityType::getInitialPicture ( ) const

5.19.2.13 double EntityType::getInitialVACost ( ) const

5.19.2.14 double EntityType::getInitialWaitingCost ( ) const

5.19.2.15 void EntityType::setInitialNVACost ( double \_initialNVACost )

5.19.2.16 void EntityType::setInitialOtherCost ( double \_initialOtherCost )

5.19.2.17 void EntityType::setInitialPicture ( std::string \_initialPicture )

5.19.2.18 void EntityType::setInitialVACost ( double \_initialVACost )

5.19.2.19 void EntityType::setInitialWaitingCost ( double \_initialWaitingCost )

5.19.2.20 std::string EntityType::show ( ) [virtual]

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



The documentation for this class was generated from the following files:

- [EntityType.h](#)
- [EntityType.cpp](#)

## 5.20 Event Class Reference

```
#include <Event.h>
```

### Public Member Functions

- [Event](#) (double *time*, [Entity](#) \**entity*, [ModelComponent](#) \**component*)
- [Event](#) (const [Event](#) &*orig*)
- virtual [~Event](#) ()
- double [getTime](#) () const
- [ModelComponent](#) \* [getComponent](#) () const
- [Entity](#) \* [getEntity](#) () const
- std::string [show](#) ()

### 5.20.1 Constructor & Destructor Documentation

5.20.1.1 [Event::Event](#) ( double *time*, [Entity](#) \* *entity*, [ModelComponent](#) \* *component* )

5.20.1.2 [Event::Event](#) ( const [Event](#) & *orig* )

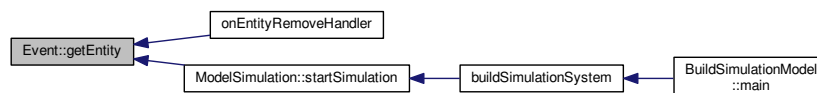
5.20.1.3 [Event::~~Event](#) ( ) [virtual]

### 5.20.2 Member Function Documentation

5.20.2.1 [ModelComponent](#) \* [Event::getComponent](#) ( ) const

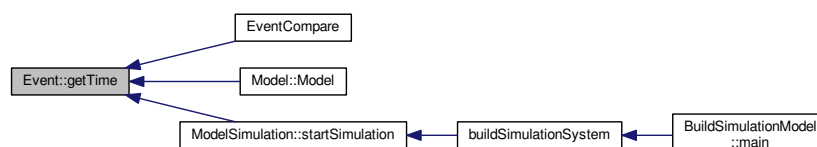
5.20.2.2 [Entity](#) \* [Event::getEntity](#) ( ) const

Here is the caller graph for this function:



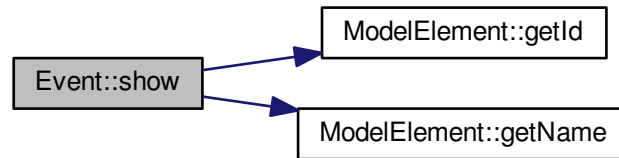
5.20.2.3 double [Event::getTime](#) ( ) const

Here is the caller graph for this function:



## 5.20.2.4 std::string Event::show ( )

Here is the call graph for this function:



Here is the caller graph for this function:



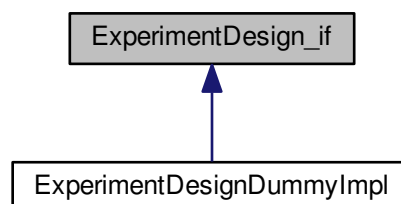
The documentation for this class was generated from the following files:

- [Event.h](#)
- [Event.cpp](#)

## 5.21 ExperimentDesign\_if Class Reference

```
#include <ExperimentDesign_if.h>
```

Inheritance diagram for ExperimentDesign\_if:



## Public Member Functions

- virtual [ProcessAnalyser\\_if](#) \* [getProcessAnalyser](#) () const =0
- virtual bool [generate2krScenarioExperiments](#) ()=0
- virtual bool [calculateContributionAndCoefficients](#) ()=0
- virtual std::list< [FactorOrInteractionContribution](#) \* > \* [getContributions](#) () const =0

### 5.21.1 Detailed Description

It designs a set of experiments ([SimulationScenario](#)) where que level of factors ([SimulationControl](#)) are set automatically to create a  $2^k$  experiment design, and where the contributions of the factors and their interactions (just a set of [SimulationControl](#)) can be obtained.

### 5.21.2 Member Function Documentation

5.21.2.1 virtual bool [ExperimentDesign\\_if::calculateContributionAndCoefficients](#) ( ) [pure virtual]

Implemented in [ExperimentDesignDummyImpl](#).

5.21.2.2 virtual bool [ExperimentDesign\\_if::generate2krScenarioExperiments](#) ( ) [pure virtual]

Implemented in [ExperimentDesignDummyImpl](#).

5.21.2.3 virtual std::list<[FactorOrInteractionContribution](#)\*>\* [ExperimentDesign\\_if::getContributions](#) ( ) const [pure virtual]

Implemented in [ExperimentDesignDummyImpl](#).

5.21.2.4 virtual [ProcessAnalyser\\_if](#)\* [ExperimentDesign\\_if::getProcessAnalyser](#) ( ) const [pure virtual]

Implemented in [ExperimentDesignDummyImpl](#).

The documentation for this class was generated from the following file:

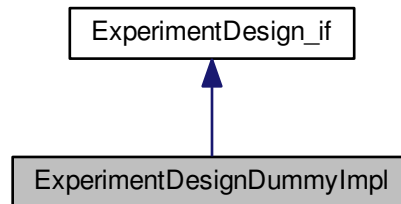
- [ExperimentDesign\\_if.h](#)



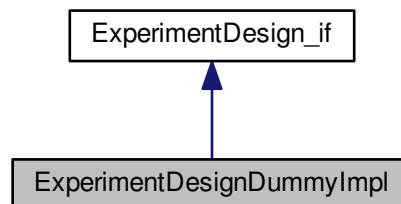
## 5.22 ExperimentDesignDummyImpl Class Reference

```
#include <ExperimentDesignDummyImpl.h>
```

Inheritance diagram for ExperimentDesignDummyImpl:



Collaboration diagram for ExperimentDesignDummyImpl:



### Public Member Functions

- [ExperimentDesignDummyImpl \(\)](#)
- [ExperimentDesignDummyImpl \(const ExperimentDesignDummyImpl &orig\)](#)
- [virtual ~ExperimentDesignDummyImpl \(\)](#)
- [ProcessAnalyser\\_if \\* getProcessAnalyser \(\) const](#)
- [bool generate2krScenarioExperiments \(\)](#)
- [bool calculateContributionAndCoefficients \(\)](#)
- [std::list< FactorOrInteractionContribution \\* > \\* getContributions \(\) const](#)

## 5.22.1 Constructor & Destructor Documentation

5.22.1.1 `ExperimentDesignDummyImpl::ExperimentDesignDummyImpl ( )`

5.22.1.2 `ExperimentDesignDummyImpl::ExperimentDesignDummyImpl ( const ExperimentDesignDummyImpl & orig )`

5.22.1.3 `ExperimentDesignDummyImpl::~~ExperimentDesignDummyImpl ( )` [virtual]

## 5.22.2 Member Function Documentation

5.22.2.1 `bool ExperimentDesignDummyImpl::calculateContributionAndCoefficients ( )` [virtual]

Implements [ExperimentDesign\\_if](#).

5.22.2.2 `bool ExperimentDesignDummyImpl::generate2krScenarioExperiments ( )` [virtual]

Implements [ExperimentDesign\\_if](#).

5.22.2.3 `std::list< FactorOrInteractionContribution * > * ExperimentDesignDummyImpl::getContributions ( ) const` [virtual]

Implements [ExperimentDesign\\_if](#).

5.22.2.4 `ProcessAnalyser_if * ExperimentDesignDummyImpl::getProcessAnalyser ( ) const` [virtual]

Implements [ExperimentDesign\\_if](#).

The documentation for this class was generated from the following files:

- [ExperimentDesignDummyImpl.h](#)
- [ExperimentDesignDummyImpl.cpp](#)

## 5.23 FactorOrInteractionContribution Class Reference

```
#include <FactorOrInteractionContribution.h>
```

### Public Member Functions

- [FactorOrInteractionContribution](#) (double contribution, double modelCoefficient, std::list< [SimulationControl](#) \* > \*controls)
- [FactorOrInteractionContribution](#) (const [FactorOrInteractionContribution](#) &orig)
- [~FactorOrInteractionContribution](#) ()
- double [getModelCoefficient](#) () const
- std::list< [SimulationControl](#) \* > \* [getControls](#) () const
- double [getContribution](#) () const

### 5.23.1 Detailed Description

This simple class corresponds to a factor when it refers to just one [SimulationControl](#), or to the interaction between two or more factors when it refers to more [SimulationControl](#). It also encapsulates the contribution of the factor or interaction and its coefficient in the full model that estimates one specific [SimulationResponse](#).

### 5.23.2 Constructor & Destructor Documentation

5.23.2.1 `FactorOrInteractionContribution::FactorOrInteractionContribution ( double contribution, double modelCoefficient, std::list< SimulationControl * > * controls )`

5.23.2.2 `FactorOrInteractionContribution::FactorOrInteractionContribution ( const FactorOrInteractionContribution & orig )`

5.23.2.3 `FactorOrInteractionContribution::~~FactorOrInteractionContribution ( )`

### 5.23.3 Member Function Documentation

5.23.3.1 `double FactorOrInteractionContribution::getContribution ( ) const`

5.23.3.2 `std::list< SimulationControl * > * FactorOrInteractionContribution::getControls ( ) const`

5.23.3.3 `double FactorOrInteractionContribution::getModelCoefficient ( ) const`

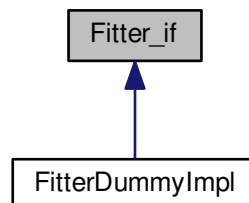
The documentation for this class was generated from the following files:

- [FactorOrInteractionContribution.h](#)
- [FactorOrInteractionContribution.cpp](#)

## 5.24 Fitter\_if Class Reference

```
#include <Fitter_if.h>
```

Inheritance diagram for `Fitter_if`:



## Public Member Functions

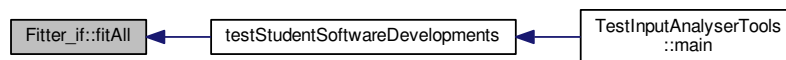
- virtual bool [isNormalDistributed](#) (double confidencelevel)=0
- virtual void [fitUniform](#) (double \*sqerror, double \*min, double \*max)=0
- virtual void [fitTriangular](#) (double \*sqerror, double \*min, double \*mo, double \*max)=0
- virtual void [fitNormal](#) (double \*sqerror, double \*avg, double \*stddev)=0
- virtual void [fitExpo](#) (double \*sqerror, double \*avg1)=0
- virtual void [fitErlang](#) (double \*sqerror, double \*avg, double \*m)=0
- virtual void [fitBeta](#) (double \*sqerror, double \*alpha, double \*beta, double \*infLimit, double \*supLimit)=0
- virtual void [fitWeibull](#) (double \*sqerror, double \*alpha, double \*scale)=0
- virtual void [fitAll](#) (double \*sqerror, std::string \*name)=0
- virtual void [setDataFilename](#) (std::string dataFilename)=0
- virtual std::string [getDataFilename](#) ()=0

### 5.24.1 Member Function Documentation

5.24.1.1 virtual void Fitter\_if::fitAll ( double \* *sqerror*, std::string \* *name* ) [pure virtual]

Implemented in [FitterDummyImpl](#).

Here is the caller graph for this function:



5.24.1.2 virtual void Fitter\_if::fitBeta ( double \* *sqerror*, double \* *alpha*, double \* *beta*, double \* *infLimit*, double \* *supLimit* ) [pure virtual]

Implemented in [FitterDummyImpl](#).

5.24.1.3 virtual void Fitter\_if::fitErlang ( double \* *sqerror*, double \* *avg*, double \* *m* ) [pure virtual]

Implemented in [FitterDummyImpl](#).

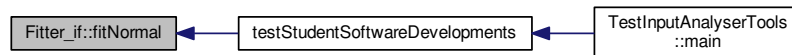
5.24.1.4 virtual void Fitter\_if::fitExpo ( double \* *sqerror*, double \* *avg1* ) [pure virtual]

Implemented in [FitterDummyImpl](#).

5.24.1.5 `virtual void Fitter_if::fitNormal ( double * sqerror, double * avg, double * stddev )` [pure virtual]

Implemented in [FitterDummyImpl](#).

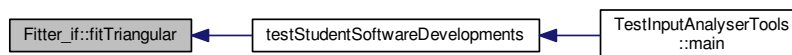
Here is the caller graph for this function:



5.24.1.6 `virtual void Fitter_if::fitTriangular ( double * sqerror, double * min, double * mo, double * max )` [pure virtual]

Implemented in [FitterDummyImpl](#).

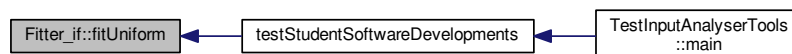
Here is the caller graph for this function:



5.24.1.7 `virtual void Fitter_if::fitUniform ( double * sqerror, double * min, double * max )` [pure virtual]

Implemented in [FitterDummyImpl](#).

Here is the caller graph for this function:



5.24.1.8 `virtual void Fitter_if::fitWeibull ( double * sqerror, double * alpha, double * scale )` [pure virtual]

Implemented in [FitterDummyImpl](#).

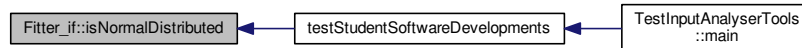
5.24.1.9 `virtual std::string Fitter_if::getDataFilename ( )` [pure virtual]

Implemented in [FitterDummyImpl](#).

5.24.1.10 `virtual bool Fitter_if::isNormalDistributed ( double confidencelevel )` [pure virtual]

Implemented in [FitterDummyImpl](#).

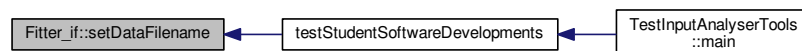
Here is the caller graph for this function:



5.24.1.11 `virtual void Fitter_if::setDataFilename ( std::string dataFilename )` [pure virtual]

Implemented in [FitterDummyImpl](#).

Here is the caller graph for this function:



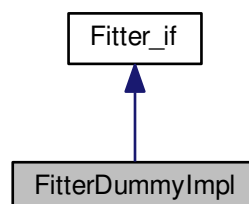
The documentation for this class was generated from the following file:

- [Fitter\\_if.h](#)

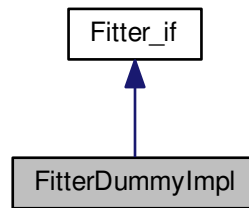
## 5.25 FitterDummyImpl Class Reference

```
#include <FitterDummyImpl.h>
```

Inheritance diagram for FitterDummyImpl:



Collaboration diagram for FitterDummyImpl:



## Public Member Functions

- [FitterDummyImpl](#) ()
- [FitterDummyImpl](#) (const [FitterDummyImpl](#) &orig)
- [~FitterDummyImpl](#) ()
- bool [isNormalDistributed](#) (double confidencelevel)
- void [fitUniform](#) (double \*sqerror, double \*min, double \*max)
- void [fitTriangular](#) (double \*sqerror, double \*min, double \*mo, double \*max)
- void [fitNormal](#) (double \*sqerror, double \*avg, double \*stddev)
- void [fitExpo](#) (double \*sqerror, double \*avg1)
- void [fitErlang](#) (double \*sqerror, double \*avg, double \*m)
- void [fitBeta](#) (double \*sqerror, double \*alpha, double \*beta, double \*infLimit, double \*supLimit)
- void [fitWeibull](#) (double \*sqerror, double \*alpha, double \*scale)
- void [fitAll](#) (double \*sqerror, std::string \*name)
- void [setDataFilename](#) (std::string dataFilename)
- std::string [getDataFilename](#) ()

## 5.25.1 Constructor & Destructor Documentation

5.25.1.1 `FitterDummyImpl::FitterDummyImpl ( )`

5.25.1.2 `FitterDummyImpl::FitterDummyImpl ( const FitterDummyImpl & orig )`

5.25.1.3 `FitterDummyImpl::~~FitterDummyImpl ( )`

## 5.25.2 Member Function Documentation

5.25.2.1 `void FitterDummyImpl::fitAll ( double * sqerror, std::string * name ) [virtual]`

Implements [Fitter\\_if](#).

5.25.2.2 void FitterDummyImpl::fitBeta ( double \* *sqrrerror*, double \* *alpha*, double \* *beta*, double \* *infLimit*, double \* *supLimit* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.3 void FitterDummyImpl::fitErlang ( double \* *sqrrerror*, double \* *avg*, double \* *m* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.4 void FitterDummyImpl::fitExpo ( double \* *sqrrerror*, double \* *avg1* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.5 void FitterDummyImpl::fitNormal ( double \* *sqrrerror*, double \* *avg*, double \* *stddev* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.6 void FitterDummyImpl::fitTriangular ( double \* *sqrrerror*, double \* *min*, double \* *mo*, double \* *max* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.7 void FitterDummyImpl::fitUniform ( double \* *sqrrerror*, double \* *min*, double \* *max* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.8 void FitterDummyImpl::fitWeibull ( double \* *sqrrerror*, double \* *alpha*, double \* *scale* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.9 std::string FitterDummyImpl::getDataFilename ( ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.10 bool FitterDummyImpl::isNormalDistributed ( double *confidencelevel* ) [virtual]

Implements [Fitter\\_if](#).

5.25.2.11 void FitterDummyImpl::setDataFilename ( std::string *dataFilename* ) [virtual]

Implements [Fitter\\_if](#).

The documentation for this class was generated from the following files:

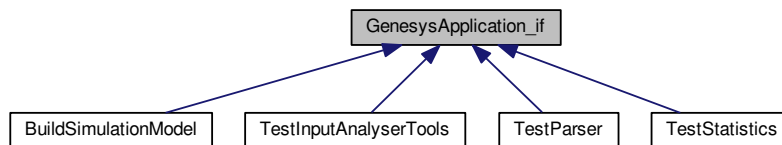
- [FitterDummyImpl.h](#)
- [FitterDummyImpl.cpp](#)



## 5.26 GenesysApplication\_if Class Reference

```
#include <GenesysApplication_if.h>
```

Inheritance diagram for GenesysApplication\_if:



### Public Member Functions

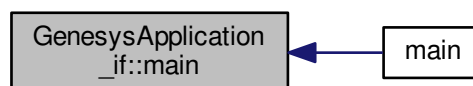
- virtual int [main](#) (int argc, char \*\*argv)=0

#### 5.26.1 Member Function Documentation

5.26.1.1 virtual int GenesysApplication\_if::main ( int *argc*, char \*\* *argv* ) [pure virtual]

Implemented in [TestInputAnalyserTools](#), [TestParser](#), [BuildSimulationModel](#), and [TestStatistics](#).

Here is the caller graph for this function:



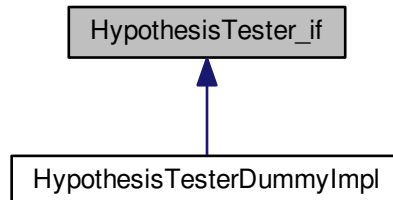
The documentation for this class was generated from the following file:

- [GenesysApplication\\_if.h](#)

## 5.27 HypothesisTester\_if Class Reference

```
#include <HypothesisTester_if.h>
```

Inheritance diagram for HypothesisTester\_if:



### Public Types

- enum [H1Comparition](#) { [DIFFERENT](#) = 1, [LESS\\_THAN](#) = 2, [GREATER\\_THAN](#) = 3 }

### Public Member Functions

- virtual double [testAverage](#) (double confidencelevel, double avg, [H1Comparition](#) comp)=0
- virtual double [testProportion](#) (double confidencelevel, double prop, [H1Comparition](#) comp)=0
- virtual double [testVariance](#) (double confidencelevel, double var, [H1Comparition](#) comp)=0
- virtual double [testAverage](#) (double confidencelevel, std::string secondPopulationDataFilename, [H1Comparition](#) comp)=0
- virtual double [testProportion](#) (double confidencelevel, std::string secondPopulationDataFilename, [H1Comparition](#) comp)=0
- virtual double [testVariance](#) (double confidencelevel, std::string secondPopulationDataFilename, [H1Comparition](#) comp)=0
- virtual void [setDataFilename](#) (std::string dataFilename)=0
- virtual std::string [getDataFilename](#) ()=0

### 5.27.1 Detailed Description

Interface for parametric hypothesis tests based on a datafile.

### 5.27.2 Member Enumeration Documentation

#### 5.27.2.1 enum HypothesisTester\_if::H1Comparition

Enumerator

***DIFFERENT***  
***LESS\_THAN***  
***GREATER\_THAN***

### 5.27.3 Member Function Documentation

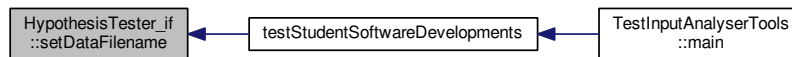
5.27.3.1 `virtual std::string HypothesisTester_if::getDataFilename ( ) [pure virtual]`

Implemented in [HypothesisTesterDummyImpl](#).

5.27.3.2 `virtual void HypothesisTester_if::setDataFilename ( std::string dataFilename ) [pure virtual]`

Implemented in [HypothesisTesterDummyImpl](#).

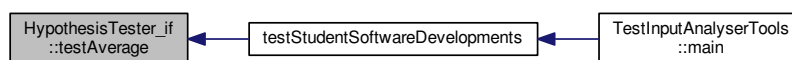
Here is the caller graph for this function:



5.27.3.3 `virtual double HypothesisTester_if::testAverage ( double confidencelevel, double avg, H1Comparition comp ) [pure virtual]`

Implemented in [HypothesisTesterDummyImpl](#).

Here is the caller graph for this function:



5.27.3.4 `virtual double HypothesisTester_if::testAverage ( double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp ) [pure virtual]`

Implemented in [HypothesisTesterDummyImpl](#).

5.27.3.5 `virtual double HypothesisTester_if::testProportion ( double confidencelevel, double prop, H1Comparition comp ) [pure virtual]`

Implemented in [HypothesisTesterDummyImpl](#).

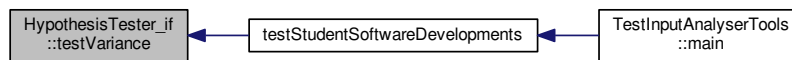
5.27.3.6 virtual double HypothesisTester\_if::testProportion ( double *confidencelevel*, std::string *secondPopulationDataFilename*, H1Comparition *comp* ) [pure virtual]

Implemented in [HypothesisTesterDummyImpl](#).

5.27.3.7 virtual double HypothesisTester\_if::testVariance ( double *confidencelevel*, double *var*, H1Comparition *comp* ) [pure virtual]

Implemented in [HypothesisTesterDummyImpl](#).

Here is the caller graph for this function:



5.27.3.8 virtual double HypothesisTester\_if::testVariance ( double *confidencelevel*, std::string *secondPopulationDataFilename*, H1Comparition *comp* ) [pure virtual]

Implemented in [HypothesisTesterDummyImpl](#).

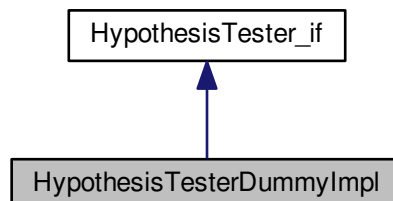
The documentation for this class was generated from the following file:

- [HypothesisTester\\_if.h](#)

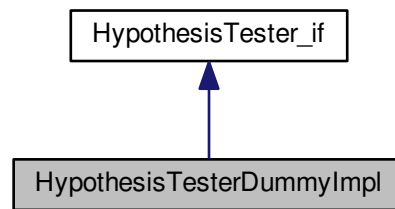
## 5.28 HypothesisTesterDummyImpl Class Reference

```
#include <HypothesisTesterDummyImpl.h>
```

Inheritance diagram for HypothesisTesterDummyImpl:



Collaboration diagram for HypothesisTesterDummyImpl:



## Public Member Functions

- [HypothesisTesterDummyImpl](#) ( )
- [HypothesisTesterDummyImpl](#) (const [HypothesisTesterDummyImpl](#) &orig)
- [~HypothesisTesterDummyImpl](#) ( )
- double [testAverage](#) (double confidencelevel, double avg, [H1Comparison](#) comp)
- double [testProportion](#) (double confidencelevel, double prop, [H1Comparison](#) comp)
- double [testVariance](#) (double confidencelevel, double var, [H1Comparison](#) comp)
- double [testAverage](#) (double confidencelevel, std::string secondPopulationDataFilename, [H1Comparison](#) comp)
- double [testProportion](#) (double confidencelevel, std::string secondPopulationDataFilename, [H1Comparison](#) comp)
- double [testVariance](#) (double confidencelevel, std::string secondPopulationDataFilename, [H1Comparison](#) comp)
- void [setDataFilename](#) (std::string dataFilename)
- std::string [getDataFilename](#) ( )

## Additional Inherited Members

### 5.28.1 Constructor & Destructor Documentation

5.28.1.1 [HypothesisTesterDummyImpl::HypothesisTesterDummyImpl](#) ( )

5.28.1.2 [HypothesisTesterDummyImpl::HypothesisTesterDummyImpl](#) ( const [HypothesisTesterDummyImpl](#) & orig )

5.28.1.3 [HypothesisTesterDummyImpl::~~HypothesisTesterDummyImpl](#) ( )

### 5.28.2 Member Function Documentation

5.28.2.1 std::string [HypothesisTesterDummyImpl::getDataFilename](#) ( ) [virtual]

Implements [HypothesisTester\\_if](#).

5.28.2.2 `void HypothesisTesterDummyImpl::setDataFilename ( std::string dataFilename ) [virtual]`

Implements [HypothesisTester\\_if](#).

5.28.2.3 `double HypothesisTesterDummyImpl::testAverage ( double confidencelevel, double avg, H1Comparition comp ) [virtual]`

Implements [HypothesisTester\\_if](#).

5.28.2.4 `double HypothesisTesterDummyImpl::testAverage ( double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp ) [virtual]`

Implements [HypothesisTester\\_if](#).

5.28.2.5 `double HypothesisTesterDummyImpl::testProportion ( double confidencelevel, double prop, H1Comparition comp ) [virtual]`

Implements [HypothesisTester\\_if](#).

5.28.2.6 `double HypothesisTesterDummyImpl::testProportion ( double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp ) [virtual]`

Implements [HypothesisTester\\_if](#).

5.28.2.7 `double HypothesisTesterDummyImpl::testVariance ( double confidencelevel, double var, H1Comparition comp ) [virtual]`

Implements [HypothesisTester\\_if](#).

5.28.2.8 `double HypothesisTesterDummyImpl::testVariance ( double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp ) [virtual]`

Implements [HypothesisTester\\_if](#).

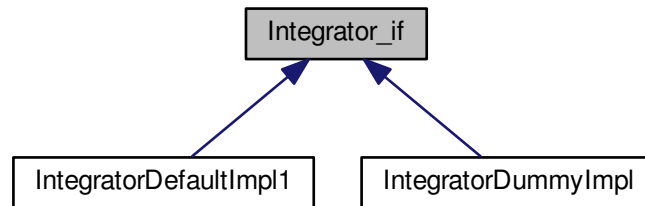
The documentation for this class was generated from the following files:

- [HypothesisTesterDummyImpl.h](#)
- [HypothesisTesterDummyImpl.cpp](#)

## 5.29 Integrator\_if Class Reference

```
#include <Integrator_if.h>
```

Inheritance diagram for Integrator\_if:



### Public Member Functions

- virtual void [setPrecision](#) (double e)=0
- virtual double [getPrecision](#) ()=0
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double), double p2)=0
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double, double), double p2, double p3)=0
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double, double, double), double p2, double p3, double p4)=0
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)=0

### 5.29.1 Detailed Description

Interface used by classes that perform the numerical integration of functions with one to four parameters. It is mainly used for calculating the probability of theoretical distributions, from its probability distribution functions.

### 5.29.2 Member Function Documentation

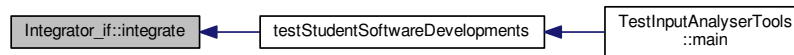
#### 5.29.2.1 virtual double Integrator\_if::getPrecision ( ) [pure virtual]

Implemented in [IntegratorDefaultImpl1](#), and [IntegratorDummyImpl](#).

5.29.2.2 `virtual double Integrator_if::integrate ( double min, double max, double(*) (double, double) f, double p2 )` [pure virtual]

Implemented in [IntegratorDefaultImpl1](#), and [IntegratorDummyImpl](#).

Here is the caller graph for this function:



5.29.2.3 `virtual double Integrator_if::integrate ( double min, double max, double(*) (double, double, double) f, double p2, double p3 )` [pure virtual]

Implemented in [IntegratorDefaultImpl1](#), and [IntegratorDummyImpl](#).

5.29.2.4 `virtual double Integrator_if::integrate ( double min, double max, double(*) (double, double, double, double) f, double p2, double p3, double p4 )` [pure virtual]

Implemented in [IntegratorDefaultImpl1](#), and [IntegratorDummyImpl](#).

5.29.2.5 `virtual double Integrator_if::integrate ( double min, double max, double(*) (double, double, double, double, double) f, double p2, double p3, double p4, double p5 )` [pure virtual]

Implemented in [IntegratorDefaultImpl1](#), and [IntegratorDummyImpl](#).

5.29.2.6 `virtual void Integrator_if::setPrecision ( double e )` [pure virtual]

Implemented in [IntegratorDefaultImpl1](#), and [IntegratorDummyImpl](#).

The documentation for this class was generated from the following file:

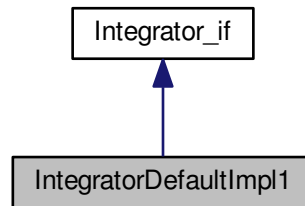
- [Integrator\\_if.h](#)



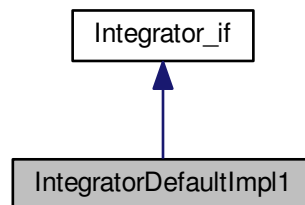
## 5.30 IntegratorDefaultImpl1 Class Reference

```
#include <IntegratorDefaultImpl1.h>
```

Inheritance diagram for IntegratorDefaultImpl1:



Collaboration diagram for IntegratorDefaultImpl1:



### Public Member Functions

- [IntegratorDefaultImpl1](#) ()
- [IntegratorDefaultImpl1](#) (const [IntegratorDefaultImpl1](#) &orig)
- virtual [~IntegratorDefaultImpl1](#) ()
- virtual void [setPrecision](#) (double e)
- virtual double [getPrecision](#) ()
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double), double p2)
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double, double), double p2, double p3)
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double, double, double), double p2, double p3, double p4)
- virtual double [integrate](#) (double min, double max, double(\*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)

### 5.30.1 Constructor & Destructor Documentation

5.30.1.1 `IntegratorDefaultImpl1::IntegratorDefaultImpl1 ( )`

5.30.1.2 `IntegratorDefaultImpl1::IntegratorDefaultImpl1 ( const IntegratorDefaultImpl1 & orig )`

5.30.1.3 `IntegratorDefaultImpl1::~~IntegratorDefaultImpl1 ( )` [virtual]

### 5.30.2 Member Function Documentation

5.30.2.1 `double IntegratorDefaultImpl1::getPrecision ( )` [virtual]

Implements [Integrator\\_if](#).

5.30.2.2 `double IntegratorDefaultImpl1::integrate ( double min, double max, double (*)(double, double) f, double p2 )`  
[virtual]

Implements [Integrator\\_if](#).

5.30.2.3 `double IntegratorDefaultImpl1::integrate ( double min, double max, double (*)(double, double, double) f, double p2, double p3 )` [virtual]

Implements [Integrator\\_if](#).

5.30.2.4 `double IntegratorDefaultImpl1::integrate ( double min, double max, double (*)(double, double, double, double) f, double p2, double p3, double p4 )` [virtual]

Implements [Integrator\\_if](#).

5.30.2.5 `double IntegratorDefaultImpl1::integrate ( double min, double max, double (*)(double, double, double, double, double) f, double p2, double p3, double p4, double p5 )` [virtual]

Implements [Integrator\\_if](#).

5.30.2.6 `void IntegratorDefaultImpl1::setPrecision ( double e )` [virtual]

Implements [Integrator\\_if](#).

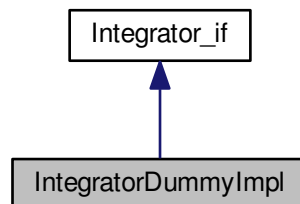
The documentation for this class was generated from the following files:

- [IntegratorDefaultImpl1.h](#)
- [IntegratorDefaultImpl1.cpp](#)

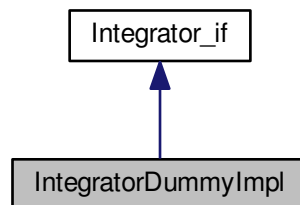
## 5.31 IntegratorDummyImpl Class Reference

```
#include <IntegratorDummyImpl.h>
```

Inheritance diagram for IntegratorDummyImpl:



Collaboration diagram for IntegratorDummyImpl:



### Public Member Functions

- [IntegratorDummyImpl](#) ()
- [IntegratorDummyImpl](#) (const [IntegratorDummyImpl](#) &orig)
- [~IntegratorDummyImpl](#) ()
- void [setPrecision](#) (double e)
- double [getPrecision](#) ()
- double [integrate](#) (double min, double max, double(\*f)(double, double), double p2)
- double [integrate](#) (double min, double max, double(\*f)(double, double, double), double p2, double p3)
- double [integrate](#) (double min, double max, double(\*f)(double, double, double, double), double p2, double p3, double p4)
- double [integrate](#) (double min, double max, double(\*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)

### 5.31.1 Constructor & Destructor Documentation

5.31.1.1 `IntegratorDummyImpl::IntegratorDummyImpl ( )`

5.31.1.2 `IntegratorDummyImpl::IntegratorDummyImpl ( const IntegratorDummyImpl & orig )`

5.31.1.3 `IntegratorDummyImpl::~~IntegratorDummyImpl ( )`

### 5.31.2 Member Function Documentation

5.31.2.1 `double IntegratorDummyImpl::getPrecision ( ) [virtual]`

Implements [Integrator\\_if](#).

5.31.2.2 `double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double) f, double p2 ) [virtual]`

Implements [Integrator\\_if](#).

5.31.2.3 `double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double, double) f, double p2, double p3 ) [virtual]`

Implements [Integrator\\_if](#).

5.31.2.4 `double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double, double, double) f, double p2, double p3, double p4 ) [virtual]`

Implements [Integrator\\_if](#).

5.31.2.5 `double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double, double, double, double) f, double p2, double p3, double p4, double p5 ) [virtual]`

Implements [Integrator\\_if](#).

5.31.2.6 `void IntegratorDummyImpl::setPrecision ( double e ) [virtual]`

Implements [Integrator\\_if](#).

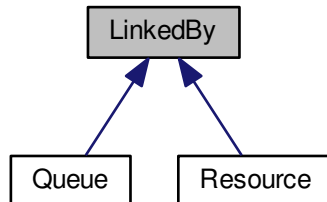
The documentation for this class was generated from the following files:

- [IntegratorDummyImpl.h](#)
- [IntegratorDummyImpl.cpp](#)

## 5.32 LinkBy Class Reference

```
#include <LinkBy.h>
```

Inheritance diagram for LinkBy:



### Public Member Functions

- [LinkBy](#) ()
- [LinkBy](#) (const [LinkBy](#) &orig)
- virtual [~LinkBy](#) ()
- void [addLink](#) ()
- void [removeLink](#) ()
- bool [isLinked](#) ()

### 5.32.1 Constructor & Destructor Documentation

5.32.1.1 [LinkBy::LinkBy](#) ( )

5.32.1.2 [LinkBy::LinkBy](#) ( const [LinkBy](#) & orig )

5.32.1.3 [LinkBy::~~LinkBy](#) ( ) [virtual]

### 5.32.2 Member Function Documentation

5.32.2.1 void [LinkBy::addLink](#) ( )

5.32.2.2 bool [LinkBy::isLinked](#) ( )

5.32.2.3 void [LinkBy::removeLink](#) ( )

The documentation for this class was generated from the following files:

- [LinkBy.h](#)
- [LinkBy.cpp](#)

## 5.33 List< T > Class Template Reference

```
#include <List.h>
```

### Public Types

- using [CompFunc](#) = std::function< bool(const T, const T) >

### Public Member Functions

- [List](#) ()
- [List](#) (const [List](#) &orig)
- virtual [~List](#) ()
- unsigned int [size](#) ()
- bool [empty](#) ()
- void [clear](#) ()
- void [pop\\_front](#) ()
- template<class Compare >  
void [sort](#) (Compare comp)
- std::list< T > \* [getList](#) () const
- T [create](#) ()
- template<typename U >  
T [create](#) (U arg)
- std::string [show](#) ()
- std::list< T >::iterator [find](#) (T element)
- void [insert](#) (T element)
- void [remove](#) (T element)
- void [setAtRank](#) (unsigned int rank, T element)
- T [getAtRank](#) (unsigned int rank)
- T [next](#) ()
- T [first](#) ()
- T [last](#) ()
- T [previous](#) ()
- T [actual](#) ()
- void [setSortFunc](#) ([CompFunc](#) \_sortFunc)

#### 5.33.1 Detailed Description

```
template<typename T>
class List< T >
```

[List](#) corresponds to an extended version of the list that must guarantee the consistency of the elements that make up the simulation model.

### 5.33.2 Member Typedef Documentation

5.33.2.1 `template<typename T> using List< T >::CompFunct = std::function<bool(const T, const T) >`

### 5.33.3 Constructor & Destructor Documentation

5.33.3.1 `template<typename T> List< T >::List ( )`

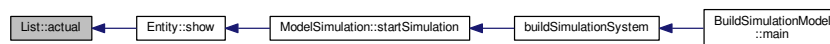
5.33.3.2 `template<typename T> List< T >::List ( const List< T > & orig )`

5.33.3.3 `template<typename T> List< T >::~~List ( ) [virtual]`

### 5.33.4 Member Function Documentation

5.33.4.1 `template<typename T> T List< T >::actual ( )`

Here is the caller graph for this function:



5.33.4.2 `template<typename T> void List< T >::clear ( )`

Here is the caller graph for this function:



5.33.4.3 `template<typename T> T List< T >::create ( )`

5.33.4.4 `template<typename T> template<typename U> T List< T >::create ( U arg )`

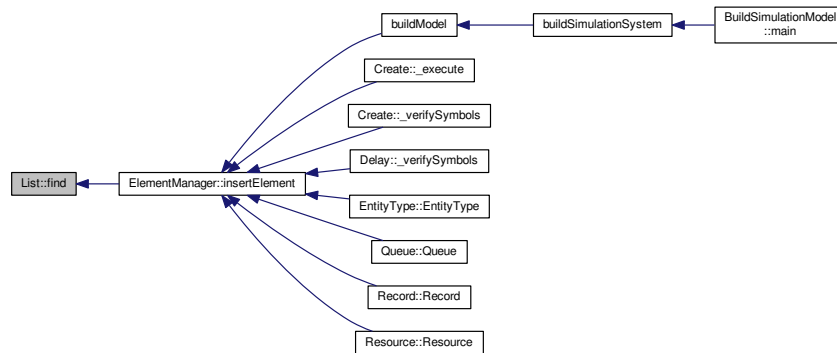
5.33.4.5 `template<typename T> bool List< T >::empty ( )`

Here is the caller graph for this function:



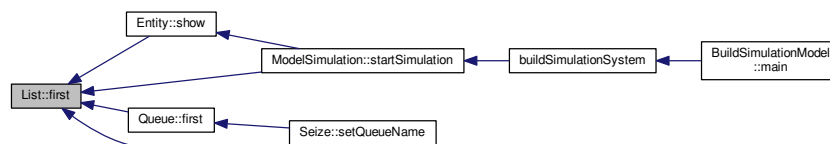
#### 5.33.4.6 `template<typename T> std::list< T >::iterator List< T >::find ( T element )`

Here is the caller graph for this function:



#### 5.33.4.7 `template<typename T> T List< T >::first ( )`

Here is the caller graph for this function:



#### 5.33.4.8 `template<typename T> T List< T >::getAtRank ( unsigned int rank )`

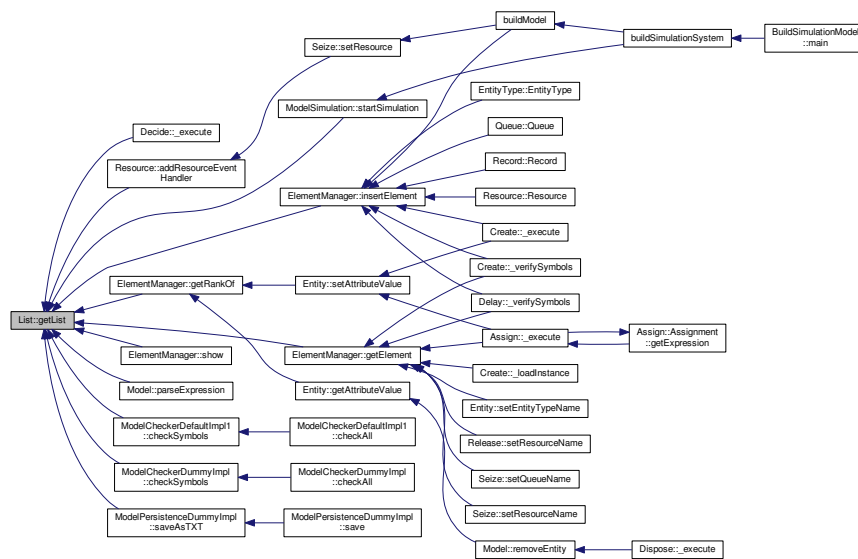
Here is the caller graph for this function:





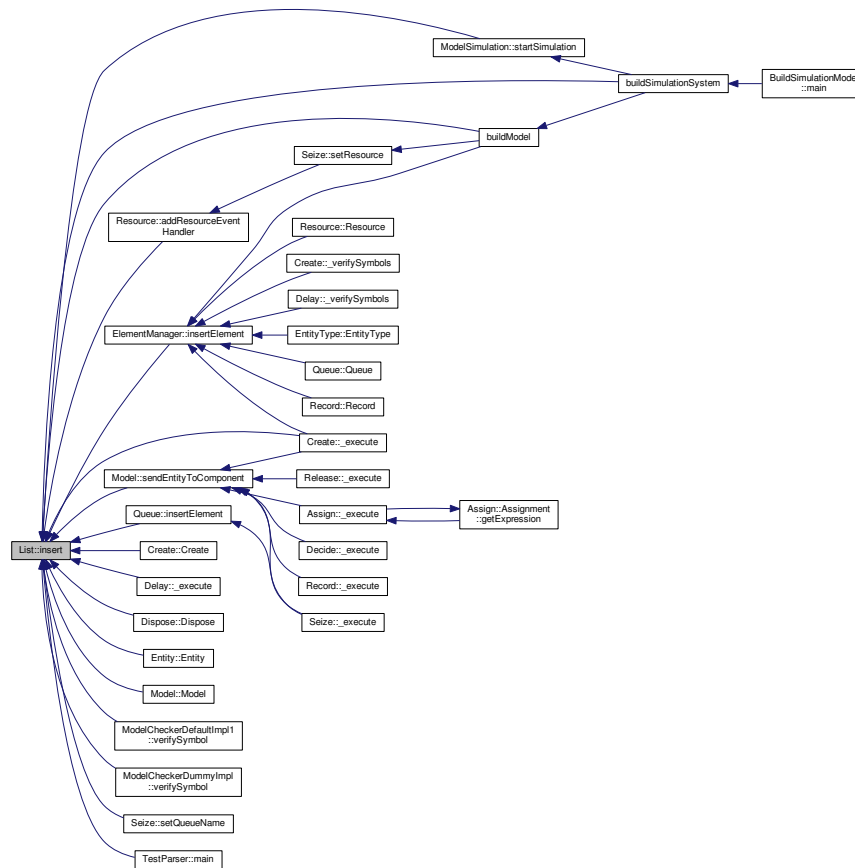
## 5.33.4.9 template&lt;typename T&gt; std::list&lt; T &gt; \* List&lt; T &gt;::getList ( ) const

Here is the caller graph for this function:



#### 5.33.4.10 `template<typename T> void List< T >::insert ( T element )`

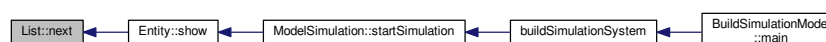
Here is the caller graph for this function:



#### 5.33.4.11 `template<typename T> T List< T >::last ( )`

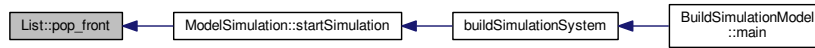
#### 5.33.4.12 `template<typename T> T List< T >::next ( )`

Here is the caller graph for this function:

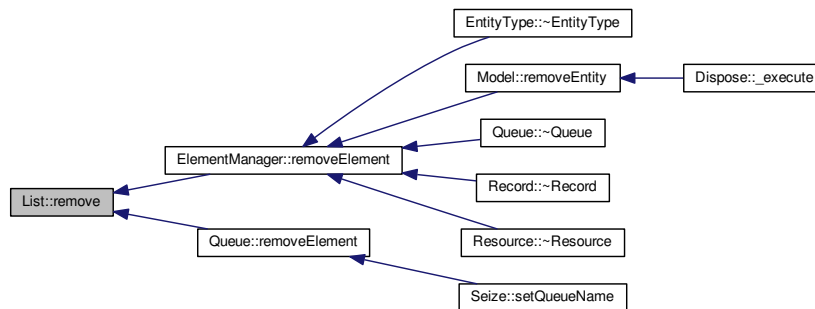


5.33.4.13 `template<typename T> void List< T >::pop_front ( )`

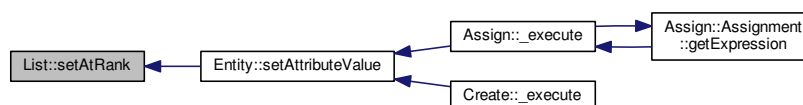
Here is the caller graph for this function:

5.33.4.14 `template<typename T> T List< T >::previous ( )`5.33.4.15 `template<typename T> void List< T >::remove ( T element )`

Here is the caller graph for this function:

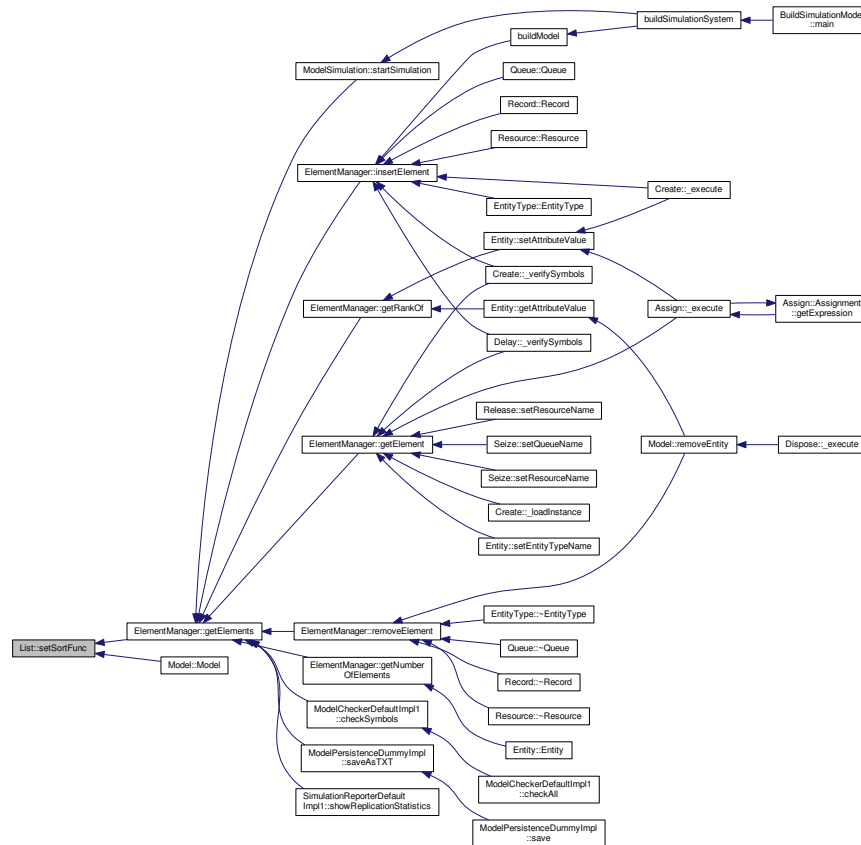
5.33.4.16 `template<typename T> void List< T >::setAtRank ( unsigned int rank, T element )`

Here is the caller graph for this function:



#### 5.33.4.17 `template<typename T> void List< T >::setSortFunc ( CompFuncT _sortFunc )`

Here is the caller graph for this function:



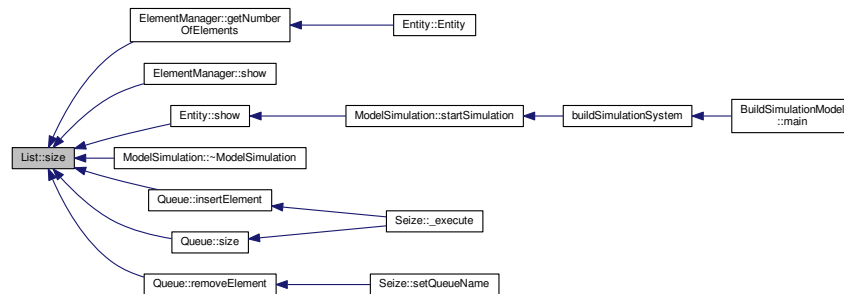
#### 5.33.4.18 `template<typename T> std::string List< T >::show ( )`

Here is the caller graph for this function:



5.33.4.19 `template<typename T> unsigned int List< T >::size ( )`

Here is the caller graph for this function:

5.33.4.20 `template<typename T> template<class Compare> void List< T >::sort ( Compare comp )`

The documentation for this class was generated from the following file:

- [List.h](#)

## 5.34 Model Class Reference

```
#include <Model.h>
```

### Public Member Functions

- `Model ( Simulator *simulator )`
- `Model ( const Model &orig )`
- `virtual ~Model ()`
- `void showReports ()`
- `bool saveModel ( std::string filename )`
- `bool loadModel ( std::string filename )`
- `bool checkModel ()`

*Checks the integrity and consistency of the model, possibly corrects some inconsistencies, and returns if the model is in position to the simulated.*

- `bool verifySymbol ( std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory )`

*Verifies if a symbol defined in a component ([ModelComponent](#)) or element is syntactically valid and addresses existing components or elements. It's used only by and directed by the component that defines the symbol.*

- `void removeEntity ( Entity *entity, bool collectStatistics )`
- `void sendEntityToComponent ( Entity *entity, ModelComponent *component, double timeDelay )`

*Used by components ([ModelComponent](#)) to send entities to another specific component, usually the next one connected to it, or used by the model itself, when processing an event ([Event](#)).*

- `double parseExpression ( const std::string expression )`
- `double parseExpression ( const std::string expression, bool *success, std::string *errorMessage )`

- [Util::identification getId](#) () const
- [List< SimulationControl \\* > \\* getControls](#) () const  
*Returns a list of values that can be externally controlled (changed). They usually correspond to input parameters in the simulation model that must be changed for an experimental design.*
- [List< SimulationResponse \\* > \\* getResponses](#) () const  
*Returns a list of exits or simulation results that can be read externally. They usually correspond to statistics resulting from the simulation that must be read for an experiment design.*
- [TraceManager \\* getTracer](#) () const  
*Provides access to the class that performs the trace of simulation and replications.*
- [OnEventManager \\* getOnEventManager](#) () const
- [ElementManager \\* getElementManager](#) () const  
*Provides access to the class that manages the most basic elements of the simulation model (such as queues, resources, variables, etc.).*
- [ModellInfo \\* getInfos](#) () const
- [Simulator \\* getParent](#) () const
- [ModelSimulation \\* getSimulation](#) () const  
*Provides access to the class that manages the model simulation.*
- [List< ModelComponent \\* > \\* getComponents](#) () const  
*Returns the list of components (such as [Create](#), [Delay](#), [Dispose](#), etc.) that make up the simulation model.*
- [List< Event \\* > \\* getEvents](#) () const  
*The future events list chronologically sorted; Events are scheduled by components when processing other events, and a replication evolves over time by sequentially processing the very first event in this list. It's initialized with events first described by source components ([SourceComponentModel](#)).*

### 5.34.1 Detailed Description

[Model](#) is probably the most important class of Genesys kernel. It represents a discrete event-driven simulation model. Each model is responsible for controlling its own simulation, ie, for sequentially processing events and collecting statistical results. A model is mainly represented by a collection of components ([ModelComponent](#)), adequately configured and connected, and a collection of under layered element ([ModelElement](#)).

### 5.34.2 Constructor & Destructor Documentation

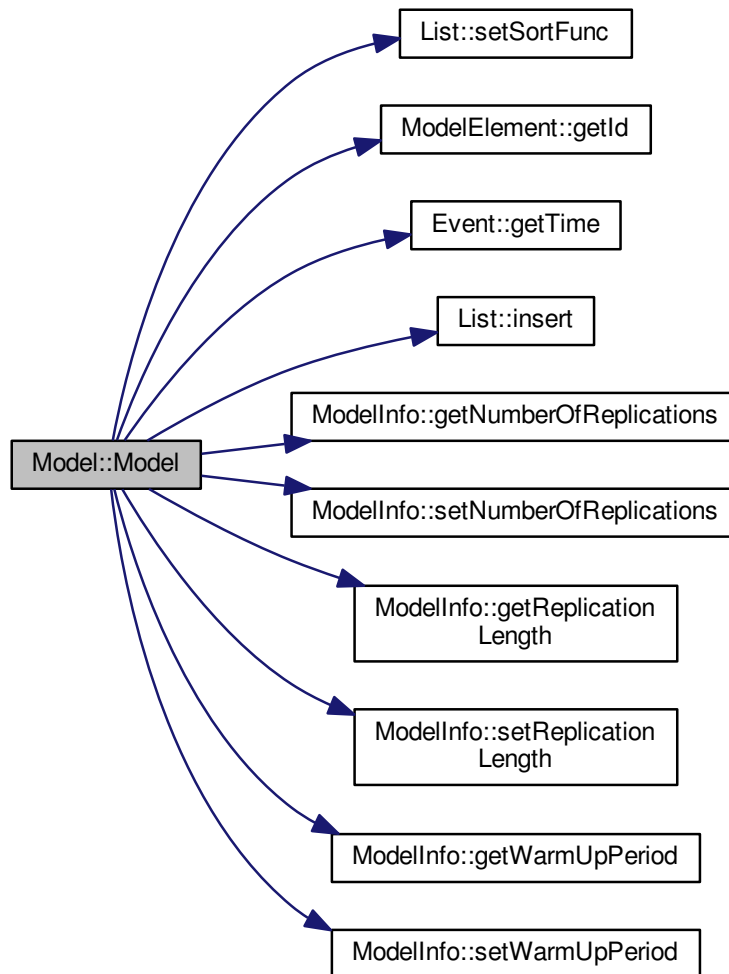
#### 5.34.2.1 [Model::Model](#) ( [Simulator](#) \* *simulator* )

Components are sorted by ID

The future events list must be chronologically sorted

Events are sorted chronologically

Here is the call graph for this function:



5.34.2.2 `Model::Model ( const Model & orig )`

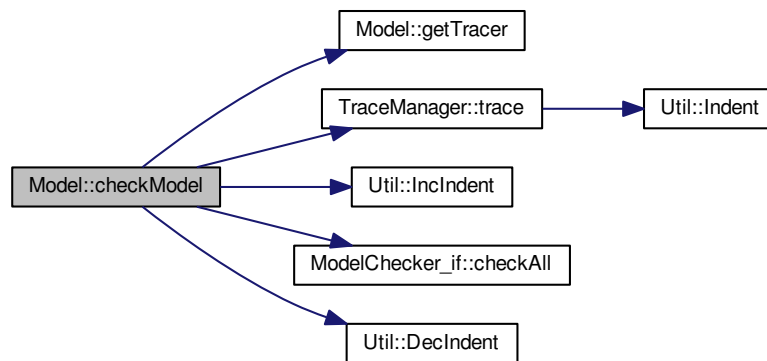
5.34.2.3 `Model::~~Model ( ) [virtual]`

### 5.34.3 Member Function Documentation

5.34.3.1 `bool Model::checkModel ( )`

Checks the integrity and consistency of the model, possibly corrects some inconsistencies, and returns if the model is in position to be simulated.

Here is the call graph for this function:



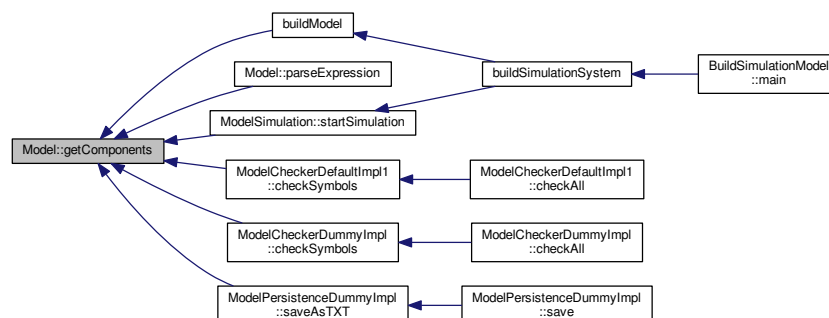
Here is the caller graph for this function:



### 5.34.3.2 List< ModelComponent \* > \* Model::getComponents ( ) const

Returns the list of components (such as [Create](#), [Delay](#), [Dispose](#), etc.) that make up the simulation model.

Here is the caller graph for this function:

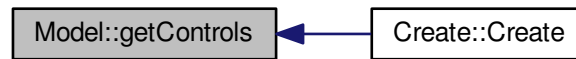




### 5.34.3.3 List< SimulationControl \* > \* Model::getControls ( ) const

Returns a list of values that can be externally controlled (changed). They usually correspond to input parameters in the simulation model that must be changed for an experimental design.

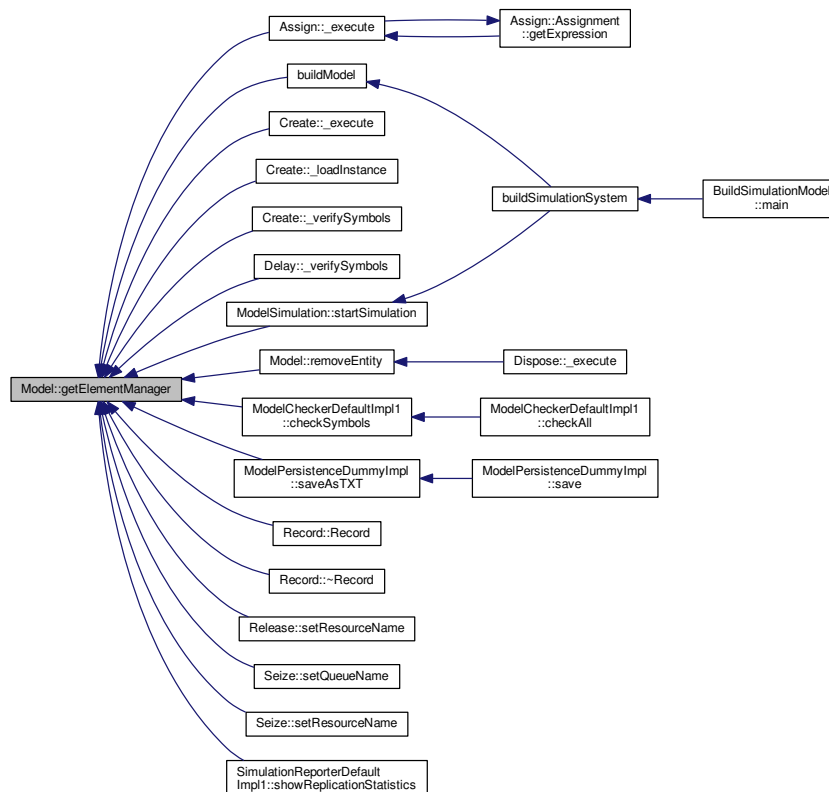
Here is the caller graph for this function:



### 5.34.3.4 ElementManager \* Model::getElementManager ( ) const

Provides access to the class that manages the most basic elements of the simulation model (such as queues, resources, variables, etc.).

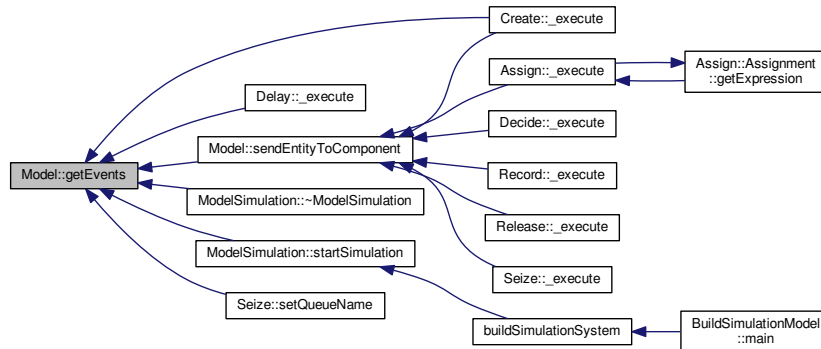
Here is the caller graph for this function:



#### 5.34.3.5 List< Event \* > \* Model::getEvents ( ) const

The future events list chronologically sorted; Events are scheduled by components when processing other events, and a replication evolves over time by sequentially processing the very first event in this list. It's initialized with events first described by source components (SourceComponentModel).

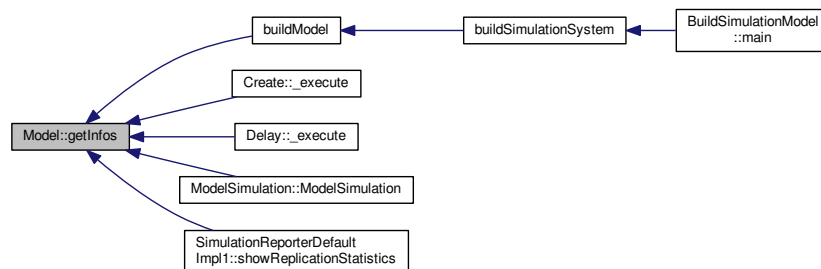
Here is the caller graph for this function:



#### 5.34.3.6 Util::identification Model::getId ( ) const

#### 5.34.3.7 ModelInfo \* Model::getInfos ( ) const

Here is the caller graph for this function:



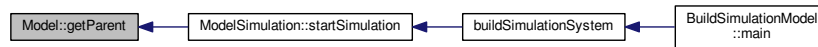
#### 5.34.3.8 OnEventManager \* Model::getOnEventManager ( ) const

Here is the caller graph for this function:



#### 5.34.3.9 Simulator \* Model::getParent ( ) const

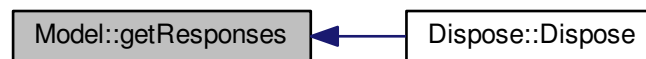
Here is the caller graph for this function:



#### 5.34.3.10 List< SimulationResponse \* > \* Model::getResponses ( ) const

Returns a list of exits or simulation results that can be read externally. They usually correspond to statistics resulting from the simulation that must be read for an experiment design.

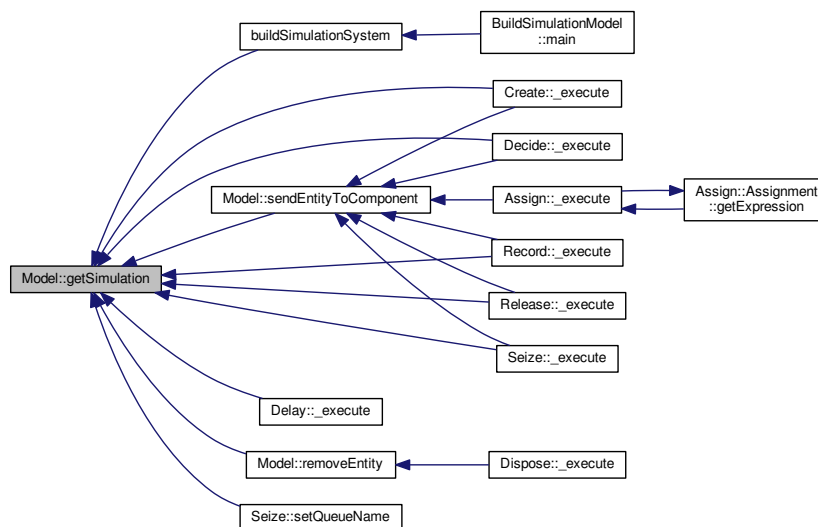
Here is the caller graph for this function:



#### 5.34.3.11 ModelSimulation \* Model::getSimulation ( ) const

Provides access to the class that manages the model simulation.

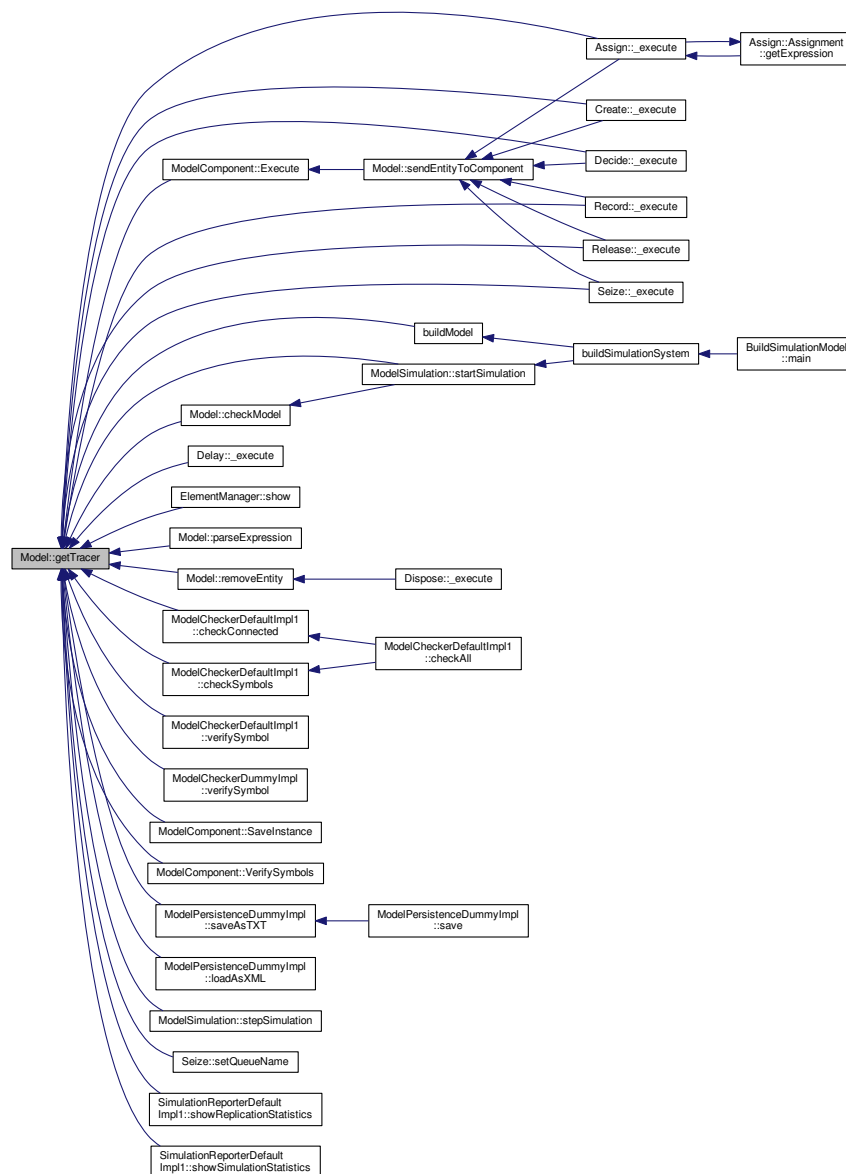
Here is the caller graph for this function:



5.34.3.12 `TraceManager * Model::getTracer ( ) const`

Provides access to the class that performs the trace of simulation and replications.

Here is the caller graph for this function:



5.34.3.13 `bool Model::loadModel ( std::string filename )`

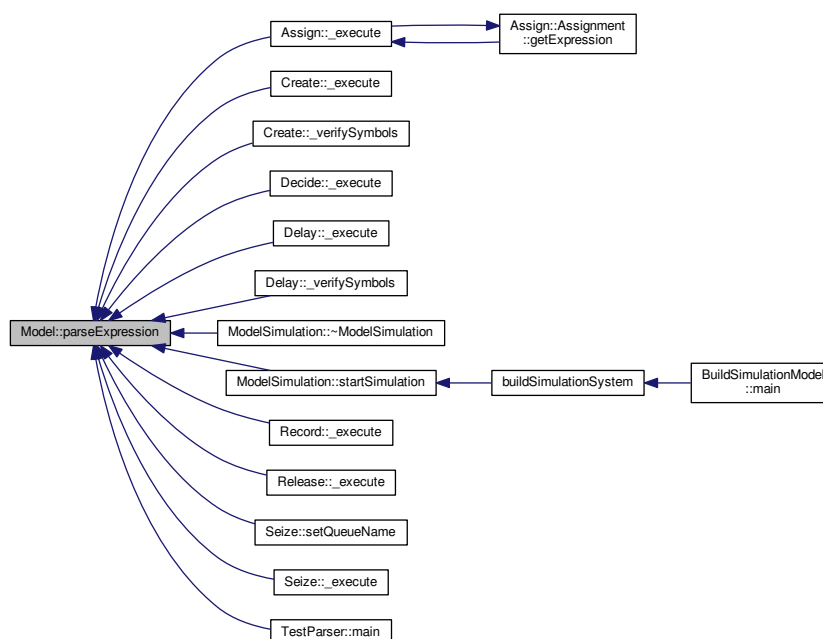
Here is the call graph for this function:

5.34.3.14 `double Model::parseExpression ( const std::string expression )`

Here is the call graph for this function:

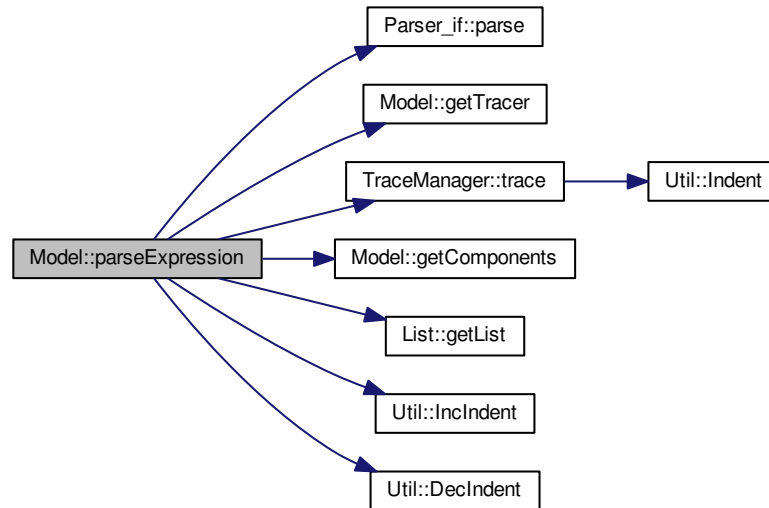


Here is the caller graph for this function:



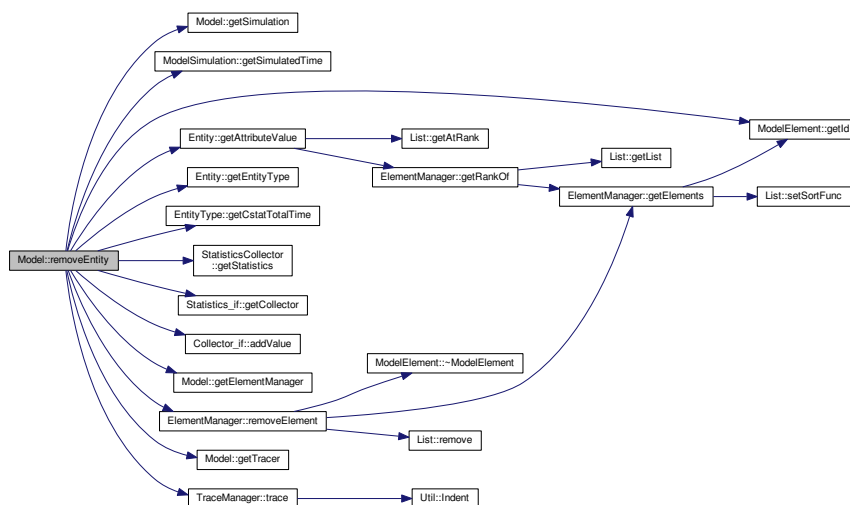
5.34.3.15 `double Model::parseExpression ( const std::string expression, bool * success, std::string * errorMessage )`

Here is the call graph for this function:



5.34.3.16 `void Model::removeEntity ( Entity * entity, bool collectStatistics )`

Here is the call graph for this function:



Here is the caller graph for this function:



#### 5.34.3.17 `bool Model::saveModel ( std::string filename )`

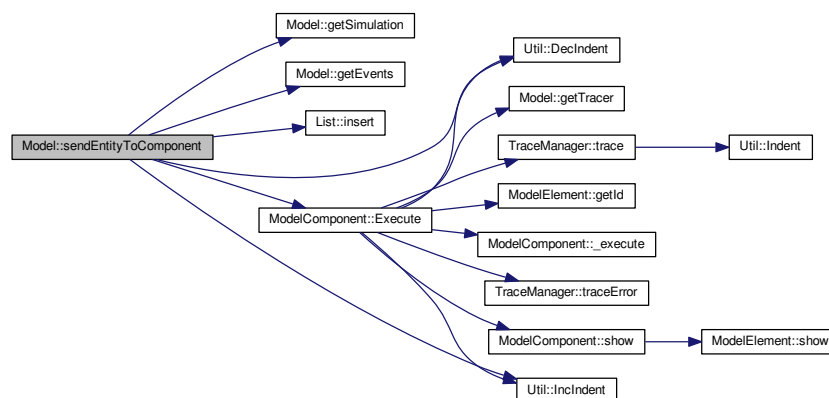
Here is the call graph for this function:



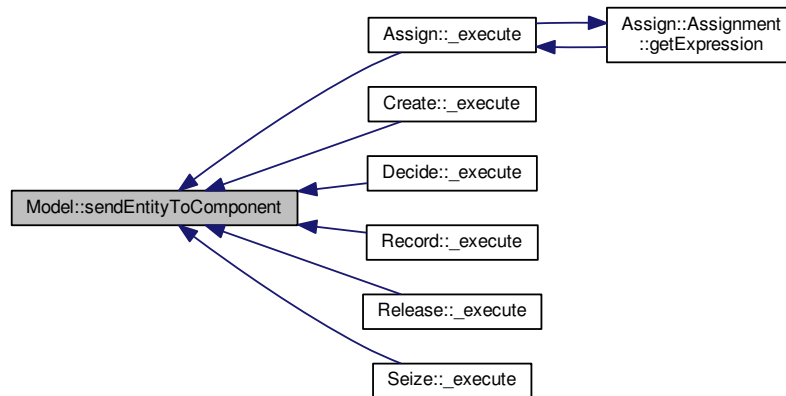
#### 5.34.3.18 `void Model::sendEntityToComponent ( Entity * entity, ModelComponent * component, double timeDelay )`

Used by components ([ModelComponent](#)) to send entities to another specific component, usually the next one connected to it, or used by the model itself, when processing an event ([Event](#)).

Here is the call graph for this function:



Here is the caller graph for this function:



5.34.3.19 `void Model::showReports ( )`

5.34.3.20 `bool Model::verifySymbol ( std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory )`

Verifies if a symbol defined in a component ([ModelComponent](#)) or element is syntactically valid and addresses existing components or elements. It's used only by and directed by the component that defines the symbol.

Here is the call graph for this function:



The documentation for this class was generated from the following files:

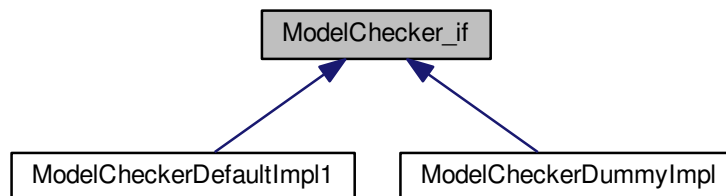
- [Model.h](#)
- [Model.cpp](#)



## 5.35 ModelChecker\_if Class Reference

```
#include <ModelChecker_if.h>
```

Inheritance diagram for ModelChecker\_if:



### Public Member Functions

- virtual bool [checkAll](#) ()=0
- virtual bool [checkAndAddInternalLiterals](#) ()=0
- virtual bool [checkConnected](#) ()=0
- virtual bool [checkSymbols](#) ()=0
- virtual bool [checkPathway](#) ()=0
- virtual bool [checkActivationCode](#) ()=0
- virtual bool [verifySymbol](#) (std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory)=0

### 5.35.1 Detailed Description

The ModelChecker is responsible for verifying the model consistency, fixing inconsistencies whenever possible

### 5.35.2 Member Function Documentation

#### 5.35.2.1 virtual bool ModelChecker\_if::checkActivationCode ( ) [pure virtual]

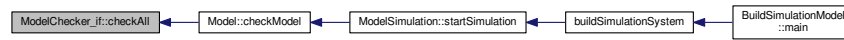
Check if components forms a valid pathway, including logical connections, such as routes, statios and pickups, for example

Implemented in [ModelCheckerDummyImpl](#), and [ModelCheckerDefaultImpl1](#).

#### 5.35.2.2 `virtual bool ModelChecker_if::checkAll ( ) [pure virtual]`

Implemented in [ModelCheckerDummyImpl](#), and [ModelCheckerDefaultImpl1](#).

Here is the caller graph for this function:



#### 5.35.2.3 `virtual bool ModelChecker_if::checkAndAddInternalLiterals ( ) [pure virtual]`

Invokes all other checks and returns true only if all of them returned true

Implemented in [ModelCheckerDummyImpl](#), and [ModelCheckerDefaultImpl1](#).

#### 5.35.2.4 `virtual bool ModelChecker_if::checkConnected ( ) [pure virtual]`

Implemented in [ModelCheckerDummyImpl](#), and [ModelCheckerDefaultImpl1](#).

#### 5.35.2.5 `virtual bool ModelChecker_if::checkPathway ( ) [pure virtual]`

Checks if user-defined strings for symbols required by components, usually expressions or functions, are valid or references existing and valid elements.

Implemented in [ModelCheckerDummyImpl](#), and [ModelCheckerDefaultImpl1](#).

#### 5.35.2.6 `virtual bool ModelChecker_if::checkSymbols ( ) [pure virtual]`

Checks if components are consistently connected to other to form a valid process-oriented model, describing how entities proceed to the flow

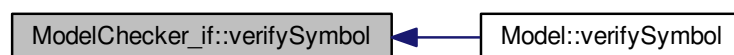
Implemented in [ModelCheckerDummyImpl](#), and [ModelCheckerDefaultImpl1](#).

#### 5.35.2.7 `virtual bool ModelChecker_if::verifySymbol ( std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory ) [pure virtual]`

unnecessary

Implemented in [ModelCheckerDummyImpl](#), and [ModelCheckerDefaultImpl1](#).

Here is the caller graph for this function:



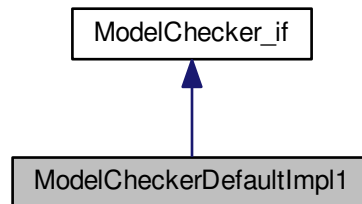
The documentation for this class was generated from the following file:

- [ModelChecker\\_if.h](#)

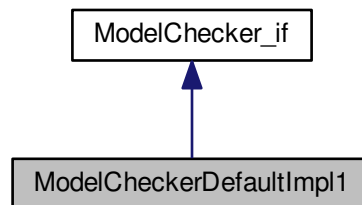
## 5.36 ModelCheckerDefaultImpl1 Class Reference

```
#include <ModelCheckerDefaultImpl1.h>
```

Inheritance diagram for ModelCheckerDefaultImpl1:



Collaboration diagram for ModelCheckerDefaultImpl1:



### Public Member Functions

- [ModelCheckerDefaultImpl1](#) ([Model](#) \*model)
- [ModelCheckerDefaultImpl1](#) (const [ModelCheckerDefaultImpl1](#) &orig)
- virtual [~ModelCheckerDefaultImpl1](#) ()
- virtual bool [checkAll](#) ()
- virtual bool [checkAndAddInternalLiterals](#) ()
- virtual bool [checkConnected](#) ()
- virtual bool [checkSymbols](#) ()
- virtual bool [checkPathway](#) ()
- virtual bool [checkActivationCode](#) ()
- virtual bool [verifySymbol](#) (std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory)

### 5.36.1 Constructor & Destructor Documentation

5.36.1.1 `ModelCheckerDefaultImpl1::ModelCheckerDefaultImpl1 ( Model * model )`

5.36.1.2 `ModelCheckerDefaultImpl1::ModelCheckerDefaultImpl1 ( const ModelCheckerDefaultImpl1 & orig )`

5.36.1.3 `ModelCheckerDefaultImpl1::~~ModelCheckerDefaultImpl1 ( )` [virtual]

### 5.36.2 Member Function Documentation

5.36.2.1 `bool ModelCheckerDefaultImpl1::checkActivationCode ( )` [virtual]

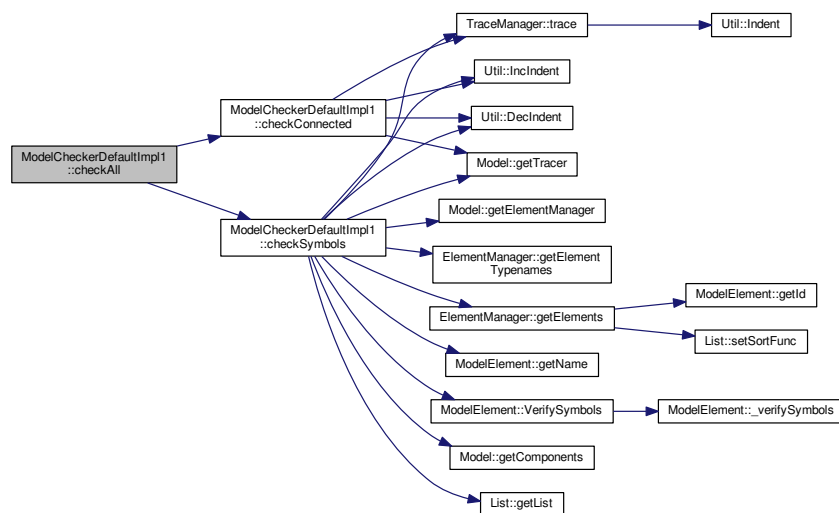
Check if components forms a valid pathway, including logical connections, such as routes, stations and pickups, for example

Implements [ModelChecker\\_if](#).

5.36.2.2 `bool ModelCheckerDefaultImpl1::checkAll ( )` [virtual]

Implements [ModelChecker\\_if](#).

Here is the call graph for this function:



5.36.2.3 `bool ModelCheckerDefaultImpl1::checkAndAddInternalLiterals ( )` [virtual]

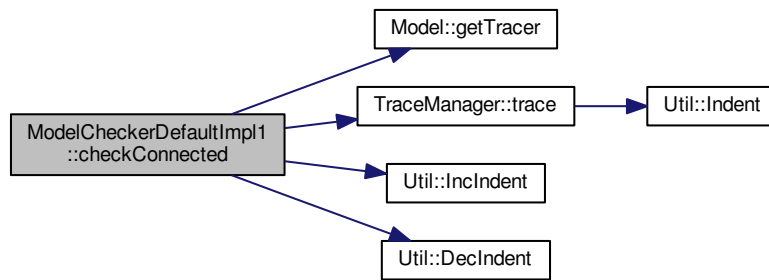
Invokes all other checks and returns true only if all of them returned true

Implements [ModelChecker\\_if](#).

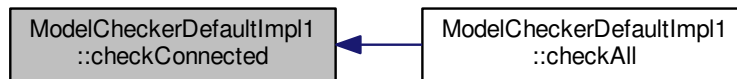
#### 5.36.2.4 bool ModelCheckerDefaultImpl1::checkConnected ( ) [virtual]

Implements [ModelChecker\\_if](#).

Here is the call graph for this function:



Here is the caller graph for this function:



#### 5.36.2.5 bool ModelCheckerDefaultImpl1::checkPathway ( ) [virtual]

Checks if user-defined strings for symbols required by components, usually expressions or functions, are valid or references existing and valid elements.

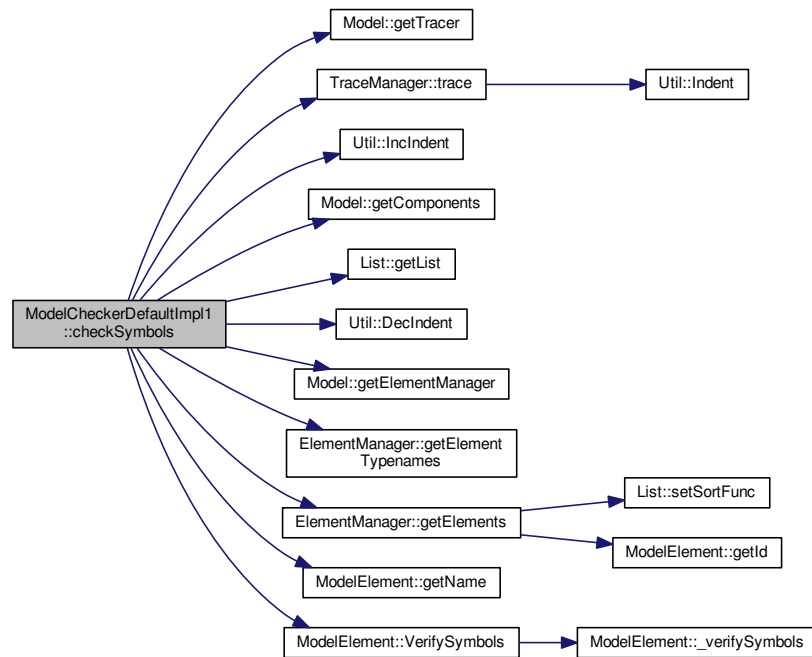
Implements [ModelChecker\\_if](#).

#### 5.36.2.6 bool ModelCheckerDefaultImpl1::checkSymbols ( ) [virtual]

Checks if components are consistently connected to other to form a valid process-oriented model, describing how entities proceed to the flow

Implements [ModelChecker\\_if](#).

Here is the call graph for this function:



Here is the caller graph for this function:

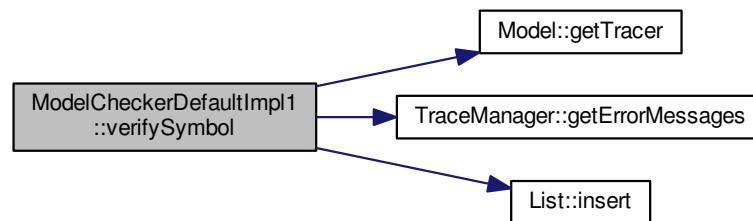


5.36.2.7 `bool ModelCheckerDefaultImpl1::verifySymbol ( std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory ) [virtual]`

unnecessary

Implements [ModelChecker\\_if](#).

Here is the call graph for this function:



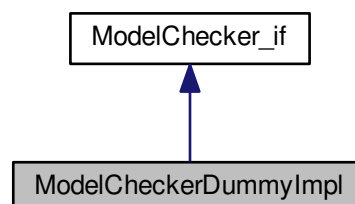
The documentation for this class was generated from the following files:

- [ModelCheckerDefaultImpl1.h](#)
- [ModelCheckerDefaultImpl1.cpp](#)

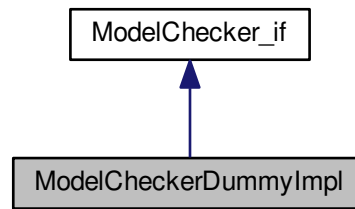
## 5.37 ModelCheckerDummyImpl Class Reference

```
#include <ModelCheckerDummyImpl.h>
```

Inheritance diagram for ModelCheckerDummyImpl:



Collaboration diagram for ModelCheckerDummyImpl:



## Public Member Functions

- [ModelCheckerDummyImpl \(Model \\*model\)](#)
- [ModelCheckerDummyImpl \(const ModelCheckerDummyImpl &orig\)](#)
- [~ModelCheckerDummyImpl \(\)](#)
- virtual bool [checkAll \(\)](#)
- virtual bool [checkAndAddInternalLiterals \(\)](#)
- virtual bool [checkConnected \(\)](#)
- virtual bool [checkSymbols \(\)](#)
- virtual bool [checkPathway \(\)](#)
- virtual bool [checkActivationCode \(\)](#)
- virtual bool [verifySymbol](#) (std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory)

### 5.37.1 Detailed Description

Just an example of possible implementation of the ModelChecker interface. Developers can implement their own class

### 5.37.2 Constructor & Destructor Documentation

5.37.2.1 `ModelCheckerDummyImpl::ModelCheckerDummyImpl ( Model * model )`

5.37.2.2 `ModelCheckerDummyImpl::ModelCheckerDummyImpl ( const ModelCheckerDummyImpl & orig )`

5.37.2.3 `ModelCheckerDummyImpl::~~ModelCheckerDummyImpl ( )`

### 5.37.3 Member Function Documentation

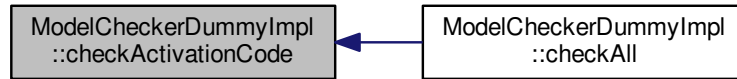
5.37.3.1 `bool ModelCheckerDummyImpl::checkActivationCode ( )` [virtual]

Check if components forms a valid pathway, including logical connections, such as routes, statios and pickups, for example



Implements [ModelChecker\\_if](#).

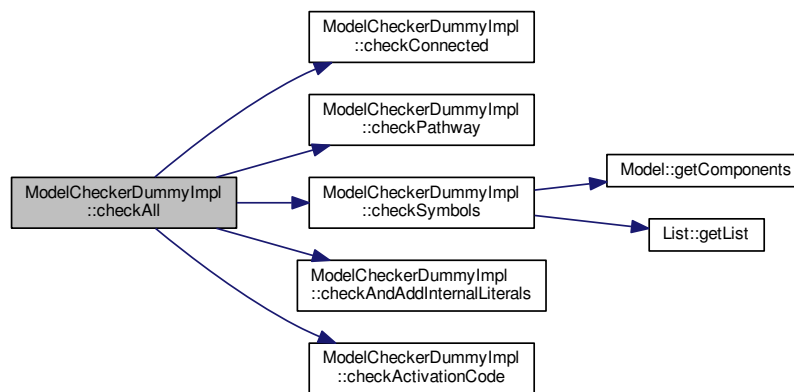
Here is the caller graph for this function:



### 5.37.3.2 `bool ModelCheckerDummyImpl::checkAll ( ) [virtual]`

Implements [ModelChecker\\_if](#).

Here is the call graph for this function:

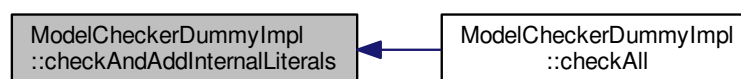


### 5.37.3.3 `bool ModelCheckerDummyImpl::checkAndAddInternalLiterals ( ) [virtual]`

Invokes all other checks and returns true only if all of them returned true

Implements [ModelChecker\\_if](#).

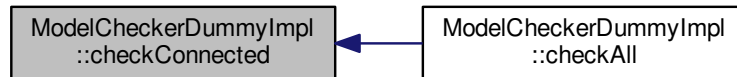
Here is the caller graph for this function:



#### 5.37.3.4 `bool ModelCheckerDummyImpl::checkConnected ( ) [virtual]`

Implements [ModelChecker\\_if](#).

Here is the caller graph for this function:



#### 5.37.3.5 `bool ModelCheckerDummyImpl::checkPathway ( ) [virtual]`

Checks if user-defined strings for symbols required by components, usually expressions or functions, are valid or references existing and valid elements.

Implements [ModelChecker\\_if](#).

Here is the caller graph for this function:

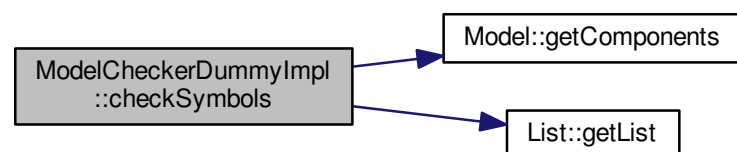


#### 5.37.3.6 `bool ModelCheckerDummyImpl::checkSymbols ( ) [virtual]`

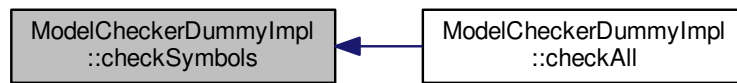
Checks if components are consistently connected to other to form a valid process-oriented model, describing how entities proceed to the flow

Implements [ModelChecker\\_if](#).

Here is the call graph for this function:



Here is the caller graph for this function:

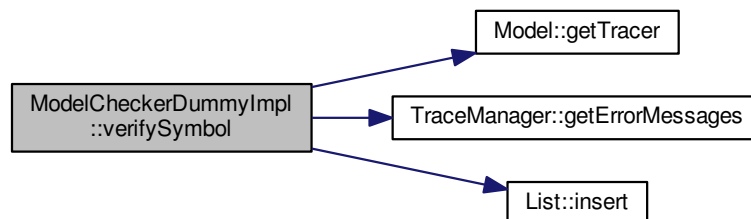


5.37.3.7 `bool ModelCheckerDummyImpl::verifySymbol ( std::string componentName, std::string expressionName, std::string expression, std::string expressionResult, bool mandatory )` [virtual]

unnecessary

Implements [ModelChecker\\_if](#).

Here is the call graph for this function:



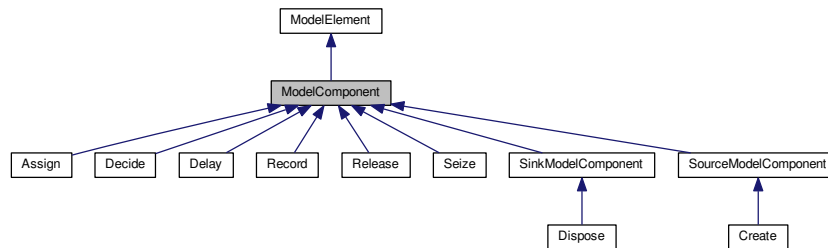
The documentation for this class was generated from the following files:

- [ModelCheckerDummyImpl.h](#)
- [ModelCheckerDummyImpl.cpp](#)

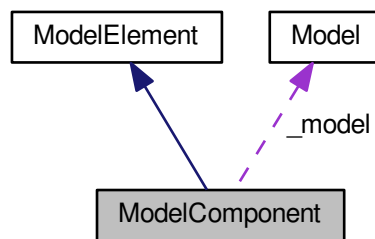
## 5.38 ModelComponent Class Reference

```
#include <ModelComponent.h>
```

Inheritance diagram for ModelComponent:



Collaboration diagram for ModelComponent:



## Public Member Functions

- [ModelComponent](#) ([Model](#) \*model, std::string componentTypename)
- [ModelComponent](#) (const [ModelComponent](#) &orig)
- virtual [~ModelComponent](#) ()
- virtual std::string [show](#) ()
- [List](#)< [ModelComponent](#) \* > \* [getNextComponents](#) () const

Returns a list of components directly connected to the output. Usually the components have a single output, but they may have none (such as [Dispose](#)) or more than one (as [Decide](#)).

## Static Public Member Functions

- static void [Execute](#) ([Entity](#) \*entity, [ModelComponent](#) \*component)
- This method triggers the simulation of the behavior of the component. It is invoked when an event (corresponding to this component) is taken from the list of future events or when an entity arrives at this component by connection.
- static bool [VerifySymbols](#) ([ModelComponent](#) \*component, std::string \*errorMessage)
  - static std::list< std::string > \* [SaveInstance](#) ([ModelComponent](#) \*component)

## Protected Member Functions

- virtual void `_execute` (`Entity *entity`)=0
- virtual std::list< std::string > \* `_saveInstance` ()
- virtual std::list< std::string > \* `_saveInstance` (std::string type)

## Protected Attributes

- `Model * _model`

### 5.38.1 Detailed Description

Um componente do modelo é um bloco que representa um comportamento específico a ser simulado. O comportamento é disparado quando uma entidade chega ao componente, o que corresponde à ocorrência de um evento. Um modelo de simulação corresponde a um conjunto de componentes interconectados para formar o processo pelo qual a entidade é submetida.

#### Parameters

<code>model</code>	The model this component belongs to
--------------------	-------------------------------------

### 5.38.2 Constructor & Destructor Documentation

5.38.2.1 `ModelComponent::ModelComponent ( Model * model, std::string componentType )`

5.38.2.2 `ModelComponent::ModelComponent ( const ModelComponent & orig )`

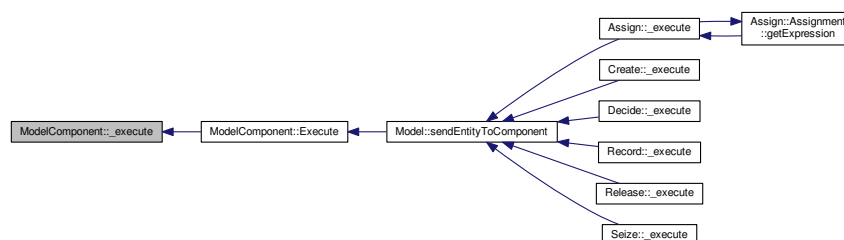
5.38.2.3 `ModelComponent::~ModelComponent ( )` [virtual]

### 5.38.3 Member Function Documentation

5.38.3.1 `virtual void ModelComponent::_execute ( Entity * entity )` [protected],[pure virtual]

Implemented in [Assign](#), [Seize](#), [Release](#), [Record](#), [Create](#), [Delay](#), [Dispose](#), and [Decide](#).

Here is the caller graph for this function:



5.38.3.2 `std::list< std::string > * ModelComponent::_saveInstance ( )` [protected],[virtual]

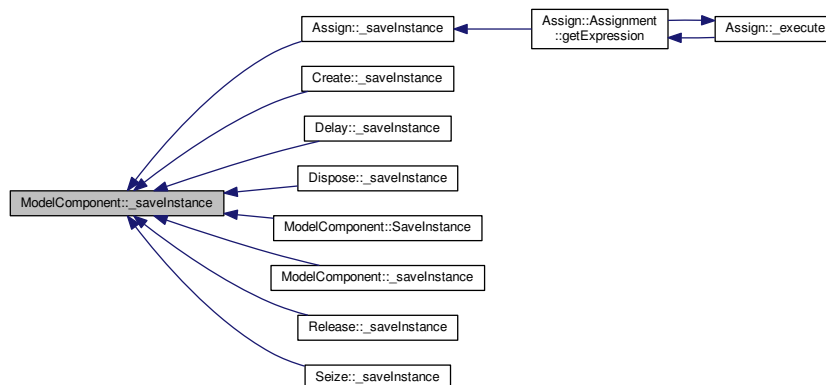
Reimplemented from [ModelElement](#).

Reimplemented in [Assign](#), [Seize](#), [Release](#), [Record](#), [Create](#), [Delay](#), [Dispose](#), and [Decide](#).

Here is the call graph for this function:



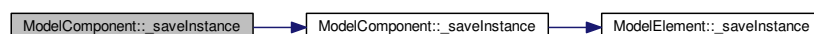
Here is the caller graph for this function:



5.38.3.3 `std::list< std::string > * ModelComponent::_saveInstance ( std::string type )` [protected],[virtual]

Reimplemented from [ModelElement](#).

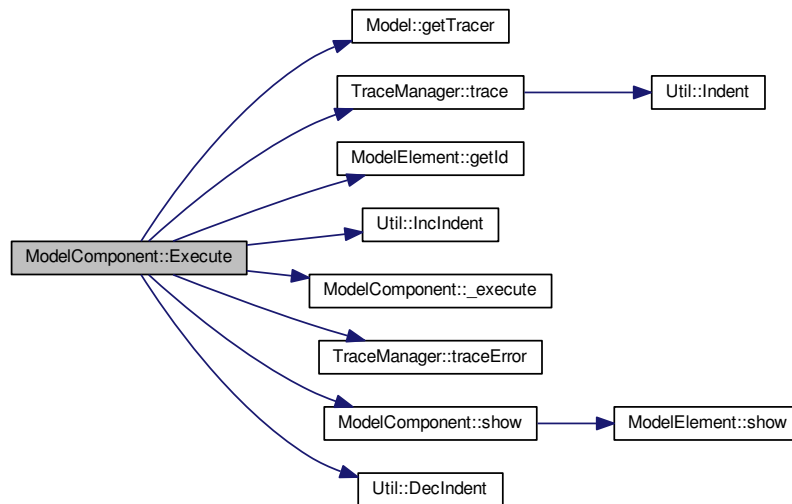
Here is the call graph for this function:



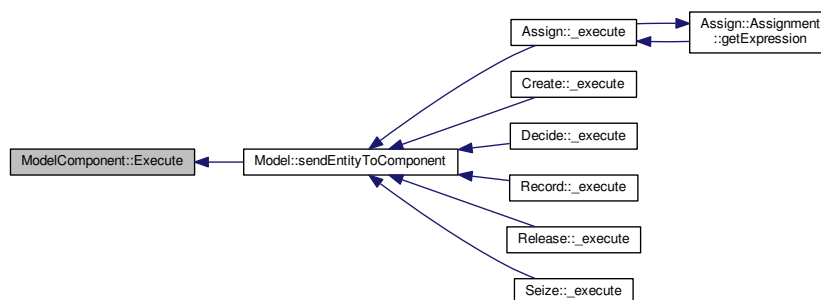
#### 5.38.3.4 void ModelComponent::Execute ( Entity \* *entity*, ModelComponent \* *component* ) [static]

This method triggers the simulation of the behavior of the component. It is invoked when an event (corresponding to this component) is taken from the list of future events or when an entity arrives at this component by connection.

Here is the call graph for this function:



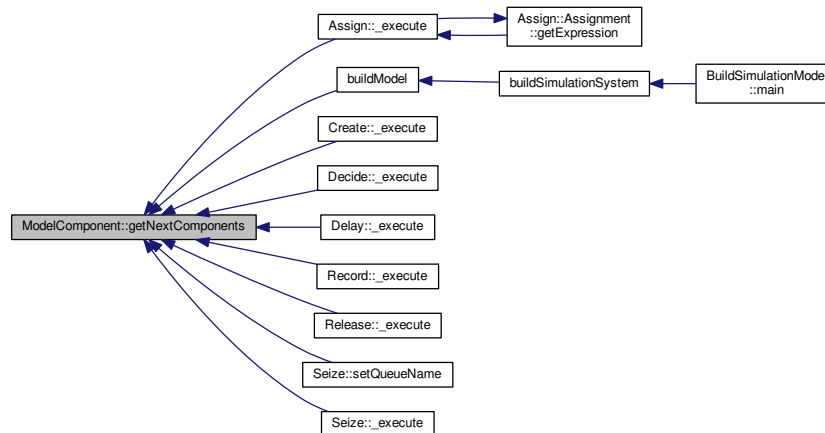
Here is the caller graph for this function:



#### 5.38.3.5 List< ModelComponent \* > \* ModelComponent::getNextComponents ( ) const

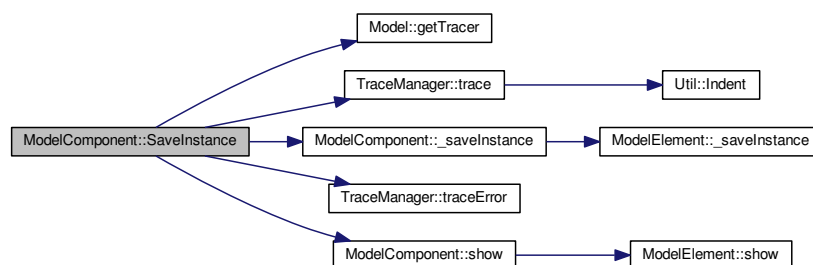
Returns a list of components directly connected to the output. Usually the components have a single output, but they may have none (such as [Dispose](#)) or more than one (as [Decide](#)).

Here is the caller graph for this function:



#### 5.38.3.6 `std::list< std::string > * ModelComponent::SaveInstance ( ModelComponent * component ) [static]`

Here is the call graph for this function:



#### 5.38.3.7 `std::string ModelComponent::show ( ) [virtual]`

Reimplemented from [ModelElement](#).

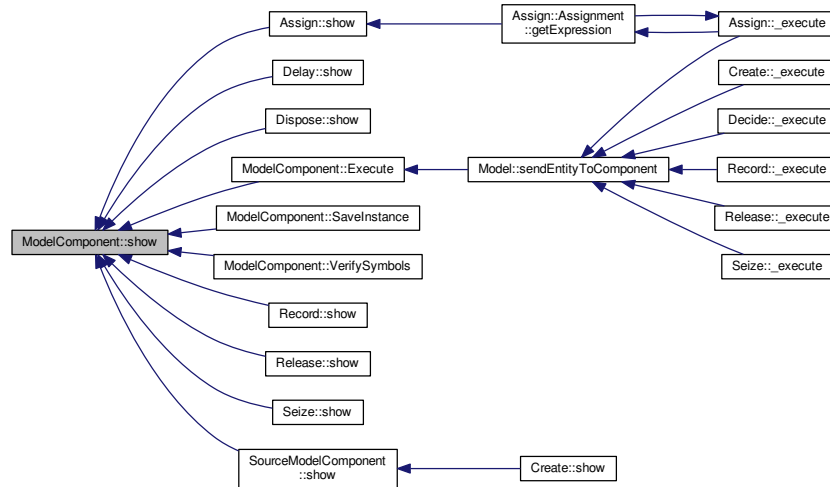
Reimplemented in [Assign](#), [SourceModelComponent](#), [Record](#), [Seize](#), [Create](#), [Delay](#), [Decide](#), [Release](#), and [Dispose](#).

Here is the call graph for this function:



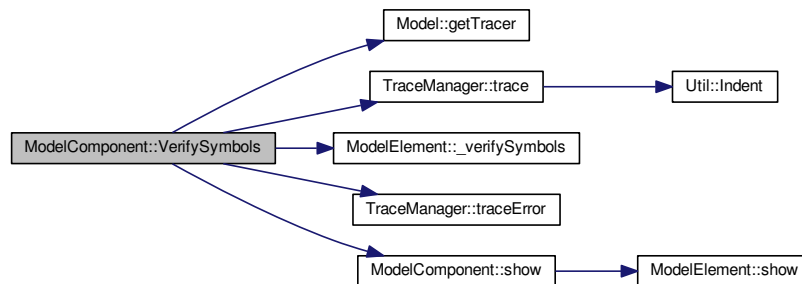


Here is the caller graph for this function:



**5.38.3.8** `bool ModelComponent::VerifySymbols ( ModelComponent * component, std::string * errorMessage )`  
`[static]`

Here is the call graph for this function:



## 5.38.4 Member Data Documentation

**5.38.4.1** `Model* ModelComponent::_model` `[protected]`

The documentation for this class was generated from the following files:

- [ModelComponent.h](#)
- [ModelComponent.cpp](#)

## 5.39 ModelComponentManager\_if Class Reference

```
#include <ModelComponentManager_if.h>
```

### Public Member Functions

- [ModelComponentManager\\_if](#) ()
- [ModelComponentManager\\_if](#) (const [ModelComponentManager\\_if](#) &orig)
- virtual [~ModelComponentManager\\_if](#) ()

### 5.39.1 Constructor & Destructor Documentation

5.39.1.1 [ModelComponentManager\\_if::ModelComponentManager\\_if](#) ( )

5.39.1.2 [ModelComponentManager\\_if::ModelComponentManager\\_if](#) ( const [ModelComponentManager\\_if](#) & orig )

5.39.1.3 virtual [ModelComponentManager\\_if::~ModelComponentManager\\_if](#) ( ) [virtual]

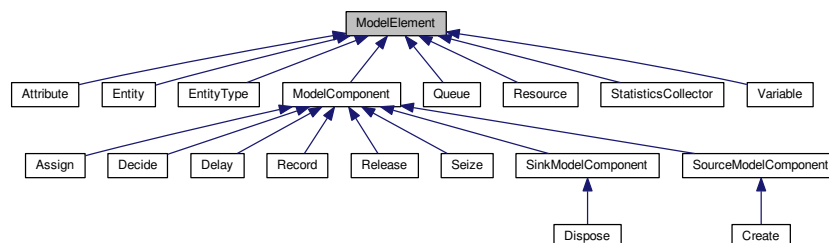
The documentation for this class was generated from the following file:

- [ModelComponentManager\\_if.h](#)

## 5.40 ModelElement Class Reference

```
#include <ModelElement.h>
```

Inheritance diagram for ModelElement:



### Public Member Functions

- [ModelElement](#) (std::string elementTypename)
- [ModelElement](#) (const [ModelElement](#) &orig)
- virtual [~ModelElement](#) ()
- virtual std::string [show](#) ()
- [Util::identification getId](#) () const
- void [setName](#) (std::string \_name)
- std::string [getName](#) () const
- std::string [getTypename](#) () const

### Static Public Member Functions

- static void [LoadInstance](#) (std::list< std::string > words)
- static std::list< std::string > \* [SaveInstance](#) ([ModelElement](#) \*element)
- static bool [VerifySymbols](#) ([ModelElement](#) \*element, std::string \*errorMessage)

### Protected Member Functions

- virtual void [\\_loadInstance](#) (std::list< std::string > words)=0
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual std::list< std::string > \* [\\_saveInstance](#) (std::string type)
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)=0

### Protected Attributes

- [Util::identification\\_id](#)
- std::string [\\_name](#)
- std::string [\\_typename](#)

#### 5.40.1 Detailed Description

This class is the basis for any element of the model (such as [Queue](#), [Resource](#), [Variable](#), etc.) and also for any component of the model. It has the infrastructure to read and write on file and to verify symbols.

#### 5.40.2 Constructor & Destructor Documentation

##### 5.40.2.1 [ModelElement::ModelElement](#) ( std::string *elementTypename* )

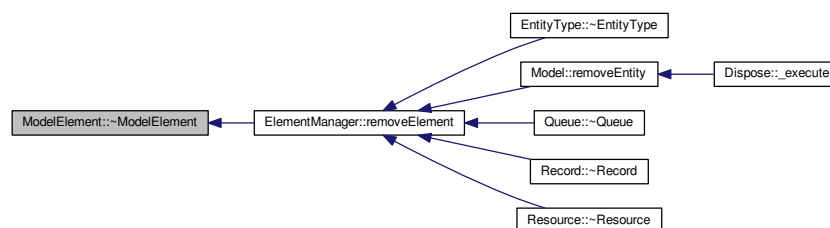
Here is the call graph for this function:



##### 5.40.2.2 [ModelElement::ModelElement](#) ( const [ModelElement](#) & *orig* )

##### 5.40.2.3 [ModelElement::~~ModelElement](#) ( ) [virtual]

Here is the caller graph for this function:



### 5.40.3 Member Function Documentation

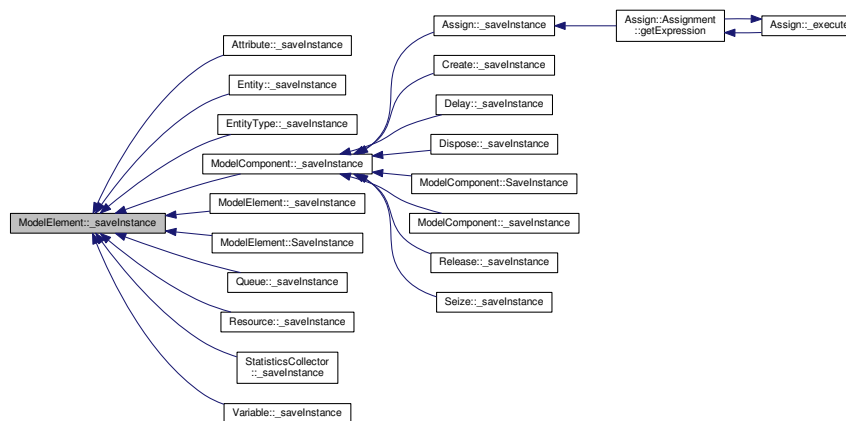
5.40.3.1 `virtual void ModelElement::_loadInstance ( std::list< std::string > words )` [protected], [pure virtual]

Implemented in [Assign](#), [Resource](#), [Seize](#), [Queue](#), [EntityType](#), [Release](#), [Entity](#), [Record](#), [Create](#), [Delay](#), [Variable](#), [Dispose](#), [StatisticsCollector](#), [Attribute](#), and [Decide](#).

5.40.3.2 `std::list< std::string > * ModelElement::_saveInstance ( )` [protected], [virtual]

Reimplemented in [Assign](#), [Resource](#), [Seize](#), [Queue](#), [EntityType](#), [ModelComponent](#), [Release](#), [Entity](#), [Record](#), [Create](#), [Delay](#), [Variable](#), [Dispose](#), [StatisticsCollector](#), [Attribute](#), and [Decide](#).

Here is the caller graph for this function:



5.40.3.3 `std::list< std::string > * ModelElement::_saveInstance ( std::string type )` [protected], [virtual]

Reimplemented in [ModelComponent](#).

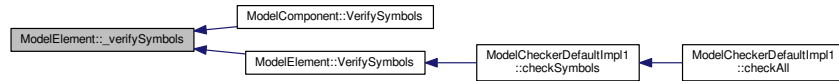
Here is the call graph for this function:



```
5.40.3.4 virtual bool ModelElement::_verifySymbols ( std::string * errorMessage ) [protected], [pure virtual]
```

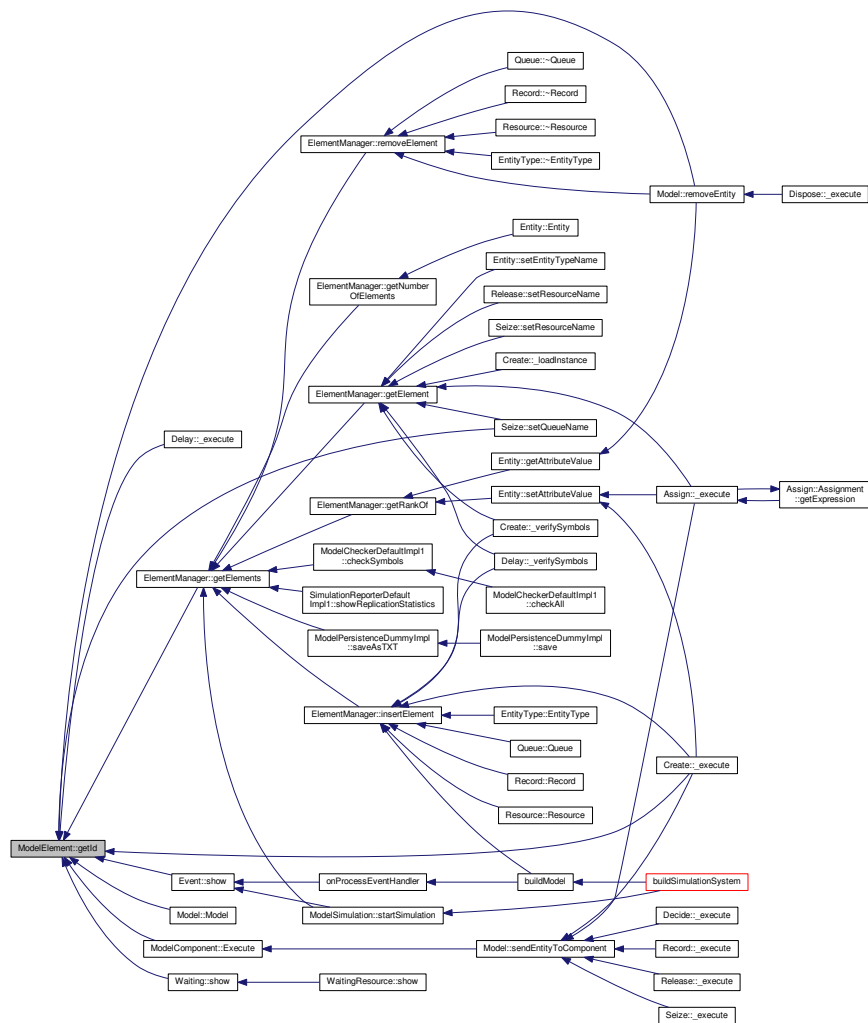
Implemented in [Assign](#), [Resource](#), [Seize](#), [Queue](#), [EntityType](#), [Release](#), [Entity](#), [Record](#), [Create](#), [Delay](#), [Variable](#), [Dispose](#), [StatisticsCollector](#), [Attribute](#), and [Decide](#).

Here is the caller graph for this function:



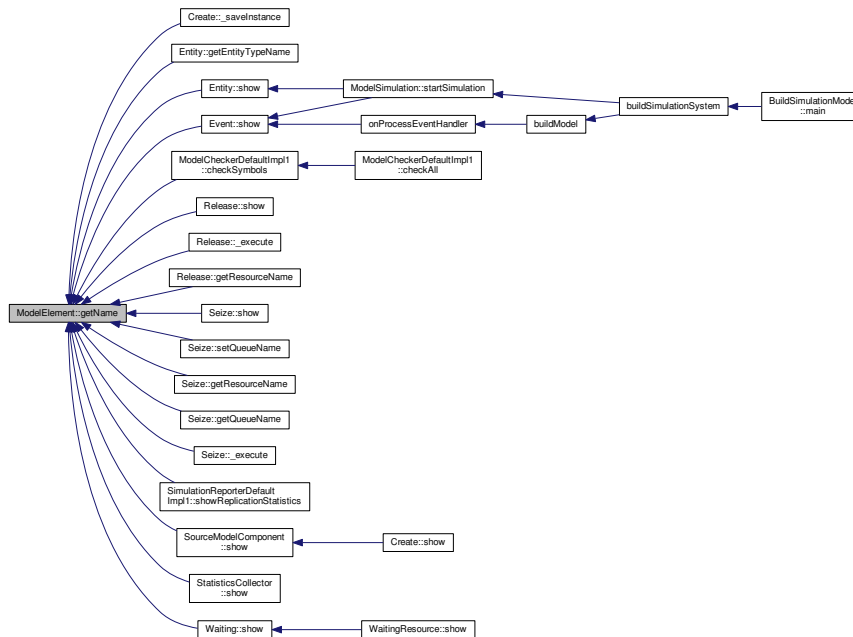
#### 5.40.3.5 Util::identitification ModelElement::getId ( ) const

Here is the caller graph for this function:



#### 5.40.3.6 `std::string ModelElement::getName ( ) const`

Here is the caller graph for this function:



#### 5.40.3.7 `std::string ModelElement::getTypeName ( ) const`

Here is the caller graph for this function:



#### 5.40.3.8 `static void ModelElement::LoadInstance ( std::list< std::string > words ) [static]`

#### 5.40.3.9 `std::list< std::string > * ModelElement::SaveInstance ( ModelElement * element ) [static]`

Here is the call graph for this function:



## 5.40.3.10 void ModelElement::setName ( std::string \_name )

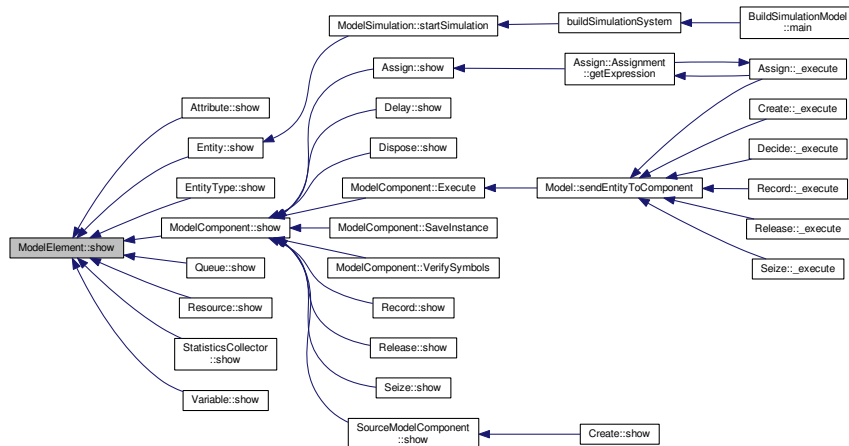
Here is the caller graph for this function:



## 5.40.3.11 std::string ModelElement::show ( ) [virtual]

Reimplemented in [Assign](#), [SourceModelComponent](#), [Resource](#), [Queue](#), [ModelComponent](#), [Record](#), [Seize](#), [Create](#), [Delay](#), [Entity](#), [EntityType](#), [Attribute](#), [Decide](#), [Release](#), [StatisticsCollector](#), [Variable](#), and [Dispose](#).

Here is the caller graph for this function:

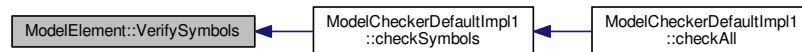


## 5.40.3.12 bool ModelElement::VerifySymbols ( ModelElement \* element, std::string \* errorMessage ) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



## 5.40.4 Member Data Documentation

5.40.4.1 `Util::identification ModelElement::_id` [protected]

5.40.4.2 `std::string ModelElement::_name` [protected]

5.40.4.3 `std::string ModelElement::_typename` [protected]

The documentation for this class was generated from the following files:

- [ModelElement.h](#)
- [ModelElement.cpp](#)

## 5.41 ModelInfo Class Reference

```
#include <ModelInfo.h>
```

### Public Member Functions

- [ModelInfo](#) ()
- [ModelInfo](#) (const [ModelInfo](#) &orig)
- virtual [~ModelInfo](#) ()
- void [setName](#) (std::string \_name)
- std::string [getName](#) () const
- void [setAnalystName](#) (std::string \_analystName)
- std::string [getAnalystName](#) () const
- void [setDescription](#) (std::string \_description)
- std::string [getDescription](#) () const
- void [setProjectTitle](#) (std::string \_projectTitle)
- std::string [getProjectTitle](#) () const
- void [setVersion](#) (std::string \_version)
- std::string [getVersion](#) () const
- void [setNumberOfReplications](#) (unsigned int \_numberOfReplications)
- unsigned int [getNumberOfReplications](#) () const
- void [setReplicationLength](#) (double \_replicationLength)
- double [getReplicationLength](#) () const
- void [setReplicationLengthTimeUnit](#) ([Util::TimeUnit](#) \_replicationLengthTimeUnit)
- [Util::TimeUnit](#) [getReplicationLengthTimeUnit](#) () const
- void [setWarmUpPeriod](#) (double \_warmUpPeriod)
- double [getWarmUpPeriod](#) () const
- void [setWarmUpPeriodTimeUnit](#) ([Util::TimeUnit](#) \_warmUpPeriodTimeUnit)
- [Util::TimeUnit](#) [getWarmUpPeriodTimeUnit](#) () const
- void [setTerminatingCondition](#) (std::string \_terminatingCondition)
- std::string [getTerminatingCondition](#) () const



### 5.41.1 Detailed Description

[ModellInfo](#) stores basic model project information.

### 5.41.2 Constructor & Destructor Documentation

5.41.2.1 `ModellInfo::ModellInfo ( )`

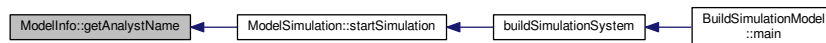
5.41.2.2 `ModellInfo::ModellInfo ( const ModellInfo & orig )`

5.41.2.3 `ModellInfo::~~ModellInfo ( )` [virtual]

### 5.41.3 Member Function Documentation

5.41.3.1 `std::string ModellInfo::getAnalystName ( ) const`

Here is the caller graph for this function:



5.41.3.2 `std::string ModellInfo::getDescription ( ) const`

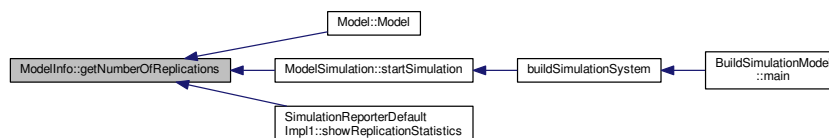
5.41.3.3 `std::string ModellInfo::getName ( ) const`

Here is the caller graph for this function:



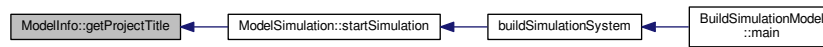
5.41.3.4 `unsigned int ModellInfo::getNumberOfReplications ( ) const`

Here is the caller graph for this function:



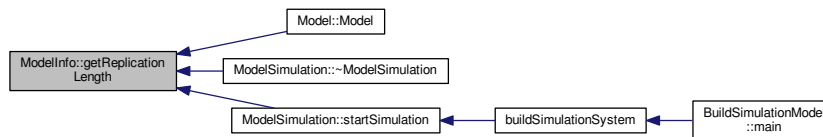
#### 5.41.3.5 `std::string ModelInfo::getProjectTitle ( ) const`

Here is the caller graph for this function:



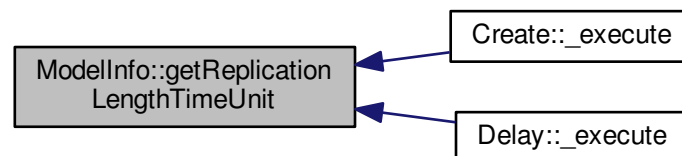
#### 5.41.3.6 `double ModelInfo::getReplicationLength ( ) const`

Here is the caller graph for this function:



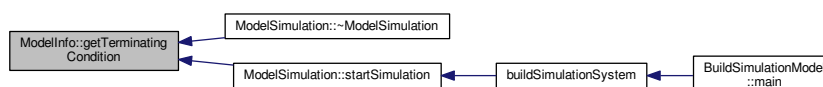
#### 5.41.3.7 `Util::TimeUnit ModelInfo::getReplicationLengthTimeUnit ( ) const`

Here is the caller graph for this function:



#### 5.41.3.8 `std::string ModelInfo::getTerminatingCondition ( ) const`

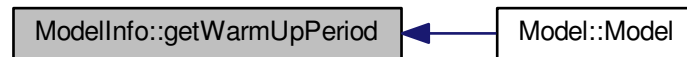
Here is the caller graph for this function:



5.41.3.9 `std::string ModelInfo::getVersion ( ) const`

5.41.3.10 `double ModelInfo::getWarmUpPeriod ( ) const`

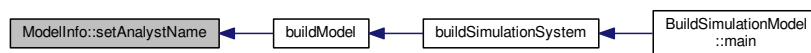
Here is the caller graph for this function:



5.41.3.11 `Util::TimeUnit ModelInfo::getWarmUpPeriodTimeUnit ( ) const`

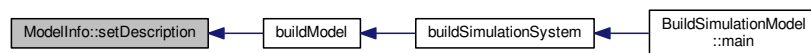
5.41.3.12 `void ModelInfo::setAnalystName ( std::string _analystName )`

Here is the caller graph for this function:



5.41.3.13 `void ModelInfo::setDescription ( std::string _description )`

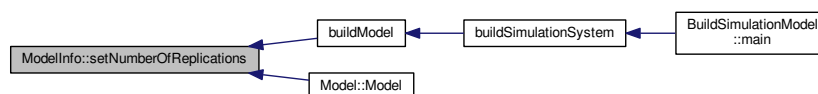
Here is the caller graph for this function:



5.41.3.14 `void ModelInfo::setName ( std::string _name )`

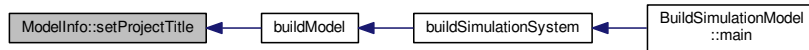
5.41.3.15 `void ModelInfo::setNumberOfReplications ( unsigned int _numberOfReplications )`

Here is the caller graph for this function:



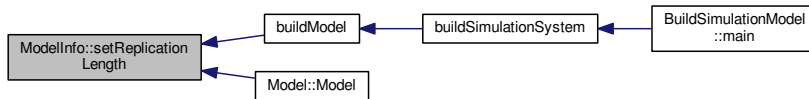
#### 5.41.3.16 void ModellInfo::setProjectTitle ( std::string *\_projectTitle* )

Here is the caller graph for this function:



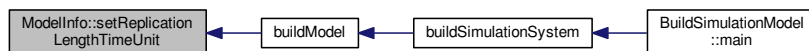
#### 5.41.3.17 void ModellInfo::setReplicationLength ( double *\_replicationLength* )

Here is the caller graph for this function:



#### 5.41.3.18 void ModellInfo::setReplicationLengthTimeUnit ( Util::TimeUnit *\_replicationLengthTimeUnit* )

Here is the caller graph for this function:

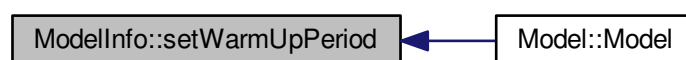


#### 5.41.3.19 void ModellInfo::setTerminatingCondition ( std::string *\_terminatingCondition* )

#### 5.41.3.20 void ModellInfo::setVersion ( std::string *\_version* )

#### 5.41.3.21 void ModellInfo::setWarmUpPeriod ( double *\_warmUpPeriod* )

Here is the caller graph for this function:



5.41.3.22 void ModelInfo::setWarmUpPeriodTimeUnit ( Util::TimeUnit *\_warmUpPeriodTimeUnit* )

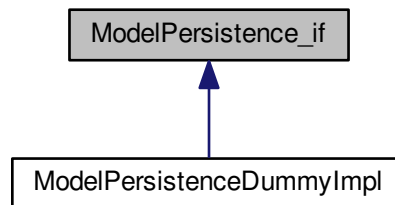
The documentation for this class was generated from the following files:

- [ModelInfo.h](#)
- [ModelInfo.cpp](#)

## 5.42 ModelPersistence\_if Class Reference

```
#include <ModelPersistence_if.h>
```

Inheritance diagram for ModelPersistence\_if:



### Public Member Functions

- virtual bool [saveAsTXT](#) (std::string filename)=0
- virtual bool [loadAsTXT](#) (std::string filename)=0
- virtual bool [saveAsXML](#) (std::string filename)=0
- virtual bool [loadAsXML](#) (std::string filename)=0
- virtual bool [save](#) (std::string filename)=0
- virtual bool [load](#) (std::string filename)=0
- virtual bool [isSaved](#) ()=0

### 5.42.1 Detailed Description

First and inadequate interface for model persistence. It should use the best pattern for the DAO approach

### 5.42.2 Member Function Documentation

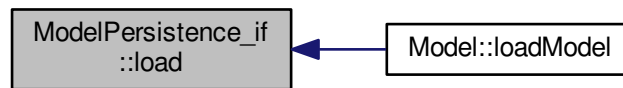
5.42.2.1 virtual bool ModelPersistence\_if::isSaved ( ) [pure virtual]

Implemented in [ModelPersistenceDummyImpl](#).

5.42.2.2 `virtual bool ModelPersistence_if::load ( std::string filename ) [pure virtual]`

Implemented in [ModelPersistenceDummyImpl](#).

Here is the caller graph for this function:



5.42.2.3 `virtual bool ModelPersistence_if::loadAsTXT ( std::string filename ) [pure virtual]`

Implemented in [ModelPersistenceDummyImpl](#).

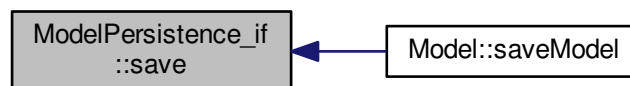
5.42.2.4 `virtual bool ModelPersistence_if::loadAsXML ( std::string filename ) [pure virtual]`

Implemented in [ModelPersistenceDummyImpl](#).

5.42.2.5 `virtual bool ModelPersistence_if::save ( std::string filename ) [pure virtual]`

Implemented in [ModelPersistenceDummyImpl](#).

Here is the caller graph for this function:



5.42.2.6 `virtual bool ModelPersistence_if::saveAsTXT ( std::string filename ) [pure virtual]`

Implemented in [ModelPersistenceDummyImpl](#).

5.42.2.7 `virtual bool ModelPersistence_if::saveAsXML ( std::string filename ) [pure virtual]`

Implemented in [ModelPersistenceDummyImpl](#).

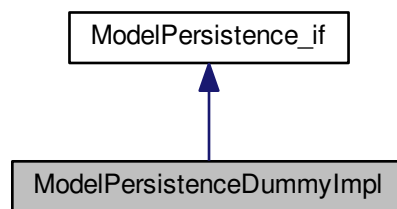
The documentation for this class was generated from the following file:

- [ModelPersistence\\_if.h](#)

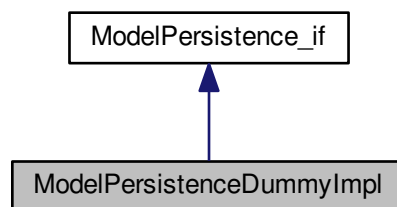
## 5.43 ModelPersistenceDummyImpl Class Reference

```
#include <ModelPersistenceDummyImpl.h>
```

Inheritance diagram for ModelPersistenceDummyImpl:



Collaboration diagram for ModelPersistenceDummyImpl:



### Public Member Functions

- [ModelPersistenceDummyImpl](#) ([Model](#) \*model)
- [ModelPersistenceDummyImpl](#) (const [ModelPersistenceDummyImpl](#) &orig)
- [~ModelPersistenceDummyImpl](#) ()
- virtual bool [saveAsTXT](#) (std::string filename)
- virtual bool [loadAsTXT](#) (std::string filename)
- virtual bool [saveAsXML](#) (std::string filename)
- virtual bool [loadAsXML](#) (std::string filename)
- virtual bool [save](#) (std::string filename)
- virtual bool [load](#) (std::string filename)
- virtual bool [isSaved](#) ()

### 5.43.1 Constructor & Destructor Documentation

5.43.1.1 `ModelPersistenceDummyImpl::ModelPersistenceDummyImpl ( Model * model )`

5.43.1.2 `ModelPersistenceDummyImpl::ModelPersistenceDummyImpl ( const ModelPersistenceDummyImpl & orig )`

5.43.1.3 `ModelPersistenceDummyImpl::~ModelPersistenceDummyImpl ( )`

### 5.43.2 Member Function Documentation

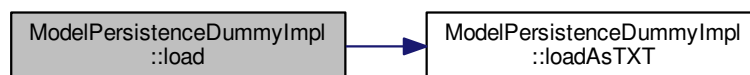
5.43.2.1 `bool ModelPersistenceDummyImpl::isSaved ( )` [virtual]

Implements [ModelPersistence\\_if](#).

5.43.2.2 `bool ModelPersistenceDummyImpl::load ( std::string filename )` [virtual]

Implements [ModelPersistence\\_if](#).

Here is the call graph for this function:



5.43.2.3 `bool ModelPersistenceDummyImpl::loadAsTXT ( std::string filename )` [virtual]

Implements [ModelPersistence\\_if](#).

Here is the caller graph for this function:

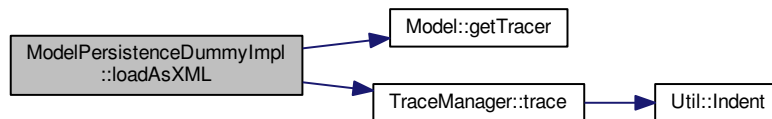




#### 5.43.2.4 bool ModelPersistenceDummyImpl::loadAsXML ( std::string filename ) [virtual]

Implements [ModelPersistence\\_if](#).

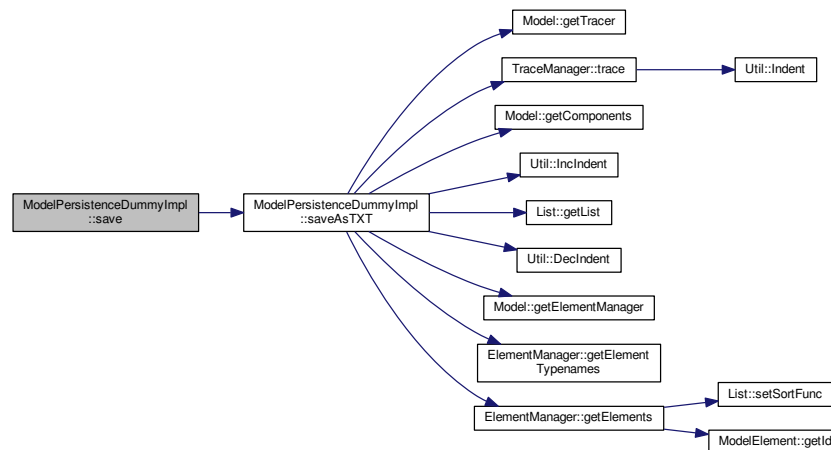
Here is the call graph for this function:



#### 5.43.2.5 bool ModelPersistenceDummyImpl::save ( std::string filename ) [virtual]

Implements [ModelPersistence\\_if](#).

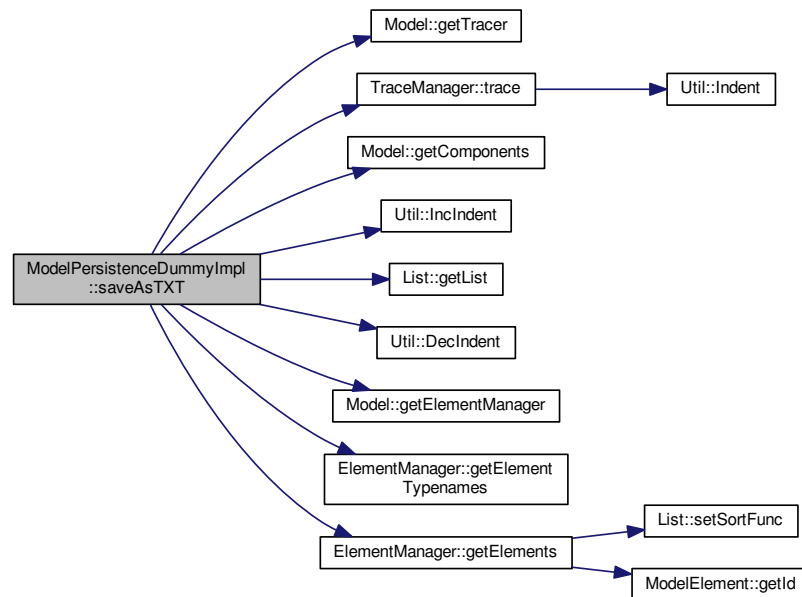
Here is the call graph for this function:



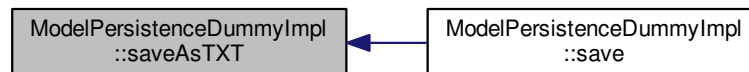
#### 5.43.2.6 bool ModelPersistenceDummyImpl::saveAsTXT ( std::string filename ) [virtual]

Implements [ModelPersistence\\_if](#).

Here is the call graph for this function:



Here is the caller graph for this function:



5.43.2.7 `bool ModelPersistenceDummyImpl::saveAsXML ( std::string filename ) [virtual]`

Implements [ModelPersistence\\_if](#).

The documentation for this class was generated from the following files:

- [ModelPersistenceDummyImpl.h](#)
- [ModelPersistenceDummyImpl.cpp](#)

## 5.44 ModelSimulation Class Reference

```
#include <ModelSimulation.h>
```

## Public Member Functions

- [ModelSimulation](#) ([Model](#) \*model)
- [ModelSimulation](#) (const [ModelSimulation](#) &orig)
- virtual [~ModelSimulation](#) ()
- void [startSimulation](#) ()
  - Starts a sequential execution of a simulation, ie, a set of replications of this model.*
- void [pauseSimulation](#) ()
- void [stepSimulation](#) ()
  - Executes the processing of a single event, the next one in the future events list.*
- void [stopSimulation](#) ()
- void [restartSimulation](#) ()
- void [setPauseOnEvent](#) (bool \_pauseOnEvent)
- bool [isPauseOnEvent](#) () const
- void [setStepByStep](#) (bool \_stepByStep)
- bool [isStepByStep](#) () const
- void [setInitializeStatistics](#) (bool \_initializeStatistics)
- bool [isInitializeStatistics](#) () const
- void [setInitializeSystem](#) (bool \_initializeSystem)
- bool [isInitializeSystem](#) () const
- void [setPauseOnReplication](#) (bool \_pauseBetweenReplications)
- bool [isPauseOnReplication](#) () const
- double [getSimulatedTime](#) () const
- bool [isRunning](#) () const
- unsigned int [getCurrentReplicationNumber](#) () const
- [ModelComponent](#) \* [getCurrentComponent](#) () const
- [Entity](#) \* [getCurrentEntity](#) () const

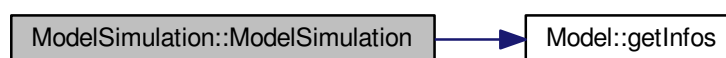
### 5.44.1 Detailed Description

The [ModelSimulation](#) controls the simulation of a model, allowing to start, pause, resume e stop a simulation, composed by a set of replications.

### 5.44.2 Constructor & Destructor Documentation

#### 5.44.2.1 [ModelSimulation::ModelSimulation](#) ( [Model](#) \* model )

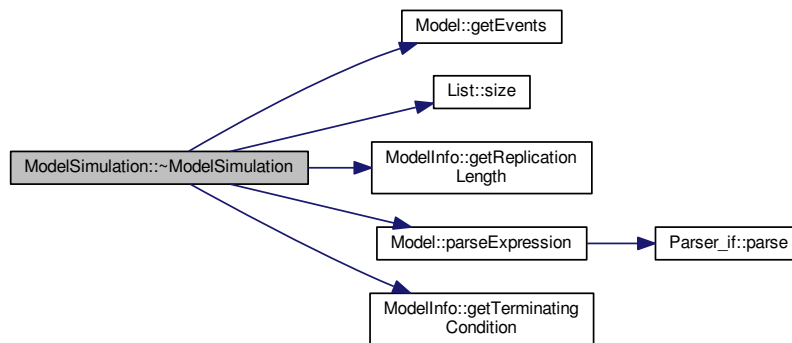
Here is the call graph for this function:



5.44.2.2 **ModelSimulation::ModelSimulation ( const ModelSimulation & orig )**

5.44.2.3 **ModelSimulation::~~ModelSimulation ( )** [virtual]

Here is the call graph for this function:



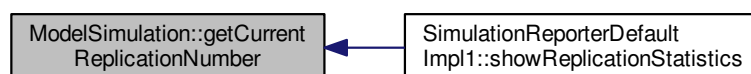
### 5.44.3 Member Function Documentation

5.44.3.1 **ModelComponent \* ModelSimulation::getCurrentComponent ( )** const

5.44.3.2 **Entity \* ModelSimulation::getCurrentEntity ( )** const

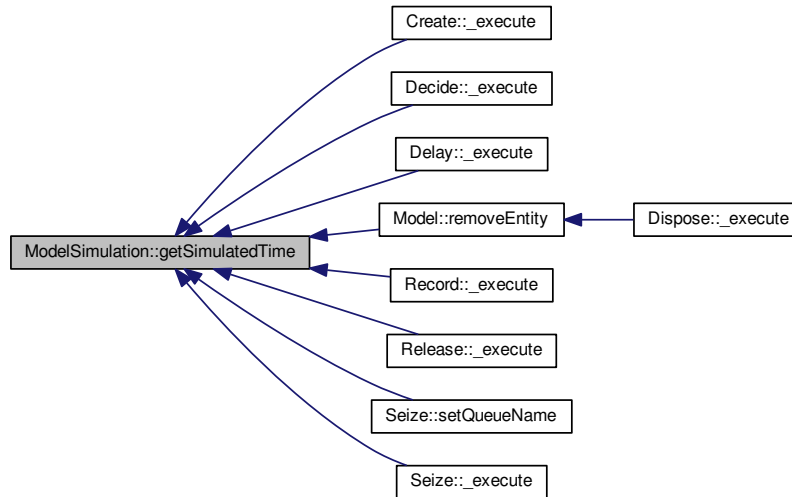
5.44.3.3 **unsigned int ModelSimulation::getCurrentReplicationNumber ( )** const

Here is the caller graph for this function:



## 5.44.3.4 double ModelSimulation::getSimulatedTime ( ) const

Here is the caller graph for this function:



## 5.44.3.5 bool ModelSimulation::isInitializeStatistics ( ) const

## 5.44.3.6 bool ModelSimulation::isInitializeSystem ( ) const

## 5.44.3.7 bool ModelSimulation::isPauseOnEvent ( ) const

## 5.44.3.8 bool ModelSimulation::isPauseOnReplication ( ) const

## 5.44.3.9 bool ModelSimulation::isRunning ( ) const

The current time in the model being simulated, i.e., the instant when the current event was triggered

## 5.44.3.10 bool ModelSimulation::isStepByStep ( ) const

## 5.44.3.11 void ModelSimulation::pauseSimulation ( )

## 5.44.3.12 void ModelSimulation::restartSimulation ( )

5.44.3.13 void ModelSimulation::setInitializeStatistics ( bool *\_initializeStatistics* )5.44.3.14 void ModelSimulation::setInitializeSystem ( bool *\_initializeSystem* )

5.44.3.15 void ModelSimulation::setPauseOnEvent ( bool *\_pauseOnEvent* )

5.44.3.16 void ModelSimulation::setPauseOnReplication ( bool *\_pauseBetweenReplications* )

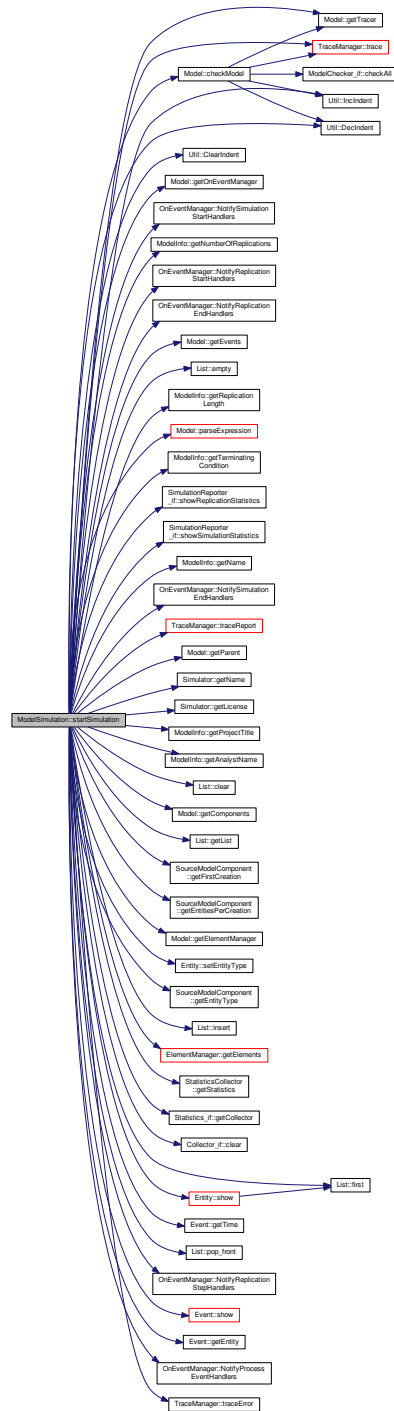
5.44.3.17 void ModelSimulation::setStepByStep ( bool *\_stepByStep* )

5.44.3.18 void ModelSimulation::startSimulation ( )

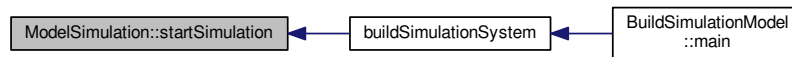
Starts a sequential execution of a simulation, ie, a set of replications of this model.

Checks the model and if ok then initialize the simulation, execute repeatedly each replication and then show simulation statistics

Here is the call graph for this function:



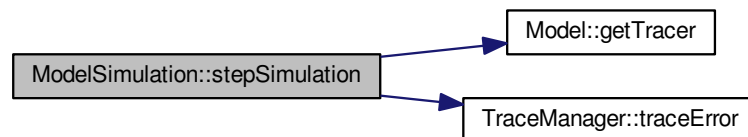
Here is the caller graph for this function:



#### 5.44.3.19 void ModelSimulation::stepSimulation ( )

Executes the processing of a single event, the next one in the future events list.

Here is the call graph for this function:



#### 5.44.3.20 void ModelSimulation::stopSimulation ( )

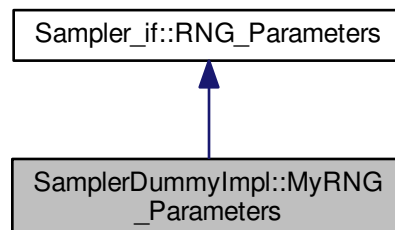
The documentation for this class was generated from the following files:

- [ModelSimulation.h](#)
- [ModelSimulation.cpp](#)

## 5.45 SamplerDummyImpl::MyRNG\_Parameters Class Reference

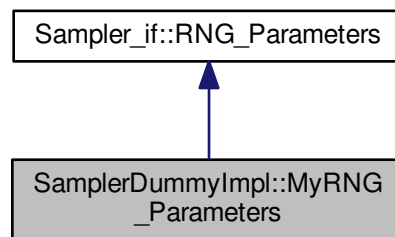
```
#include <SamplerDummyImpl.h>
```

Inheritance diagram for `SamplerDummyImpl::MyRNG_Parameters`:





Collaboration diagram for SamplerDummyImpl::MyRNG\_Parameters:



### Public Attributes

- unsigned int [seed](#)
- unsigned int [module](#)
- unsigned int [multiplier](#)

### 5.45.1 Member Data Documentation

5.45.1.1 unsigned int SamplerDummyImpl::MyRNG\_Parameters::module

5.45.1.2 unsigned int SamplerDummyImpl::MyRNG\_Parameters::multiplier

5.45.1.3 unsigned int SamplerDummyImpl::MyRNG\_Parameters::seed

The documentation for this class was generated from the following file:

- [SamplerDummyImpl.h](#)

## 5.46 OnEventManager Class Reference

```
#include <OnEventManager.h>
```

## Public Member Functions

- [OnEventManager](#) ()
- [OnEventManager](#) (const [OnEventManager](#) &orig)
- virtual [~OnEventManager](#) ()
- void [addOnReplicationStartHandler](#) ([simulationEventHandler](#) EventHandler)
- void [addOnReplicationStepHandler](#) ([simulationEventHandler](#) EventHandler)
- void [addOnReplicationEndHandler](#) ([simulationEventHandler](#) EventHandler)
- void [addOnProcessEventHandler](#) ([simulationEventHandler](#) EventHandler)
- void [addOnSimulationStartHandler](#) ([simulationEventHandler](#) EventHandler)
- void [addOnSimulationEndHandler](#) ([simulationEventHandler](#) EventHandler)
- void [addOnEntityRemoveHandler](#) ([simulationEventHandler](#) EventHandler)
- void [NotifyReplicationStartHandlers](#) ([SimulationEvent](#) \*se)
- void [NotifyReplicationStepHandlers](#) ([SimulationEvent](#) \*se)
- void [NotifyReplicationEndHandlers](#) ([SimulationEvent](#) \*se)
- void [NotifyProcessEventHandlers](#) ([SimulationEvent](#) \*se)
- void [NotifySimulationStartHandlers](#) ([SimulationEvent](#) \*se)
- void [NotifySimulationEndHandlers](#) ([SimulationEvent](#) \*se)

### 5.46.1 Detailed Description

[OnEventManager](#) allows external methods to hook interval simulation events as listeners (or observers) of pecific events. All methods added as listeners of an event will be invokved when that event is triggered.

### 5.46.2 Constructor & Destructor Documentation

5.46.2.1 [OnEventManager::OnEventManager](#) ( )

5.46.2.2 [OnEventManager::OnEventManager](#) ( const [OnEventManager](#) & orig )

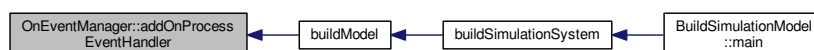
5.46.2.3 [OnEventManager::~~OnEventManager](#) ( ) [virtual]

### 5.46.3 Member Function Documentation

5.46.3.1 void [OnEventManager::addOnEntityRemoveHandler](#) ( [simulationEventHandler](#) EventHandler )

5.46.3.2 void [OnEventManager::addOnProcessEventHandler](#) ( [simulationEventHandler](#) EventHandler )

Here is the caller graph for this function:



#### 5.46.3.3 void OnEventManager::addOnReplicationEndHandler ( simulationEventHandler *EventHandler* )

Here is the caller graph for this function:



#### 5.46.3.4 void OnEventManager::addOnReplicationStartHandler ( simulationEventHandler *EventHandler* )

Here is the caller graph for this function:



#### 5.46.3.5 void OnEventManager::addOnReplicationStepHandler ( simulationEventHandler *EventHandler* )

#### 5.46.3.6 void OnEventManager::addOnSimulationEndHandler ( simulationEventHandler *EventHandler* )

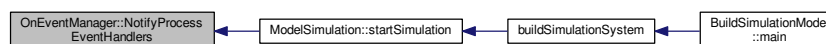
#### 5.46.3.7 void OnEventManager::addOnSimulationStartHandler ( simulationEventHandler *EventHandler* )

Here is the caller graph for this function:



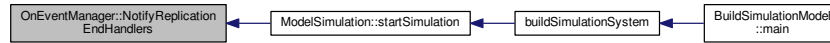
#### 5.46.3.8 void OnEventManager::NotifyProcessEventHandlers ( SimulationEvent \* *se* )

Here is the caller graph for this function:



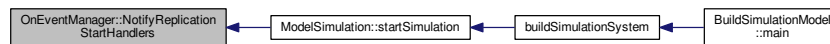
#### 5.46.3.9 void OnEventManager::NotifyReplicationEndHandlers ( SimulationEvent \* se )

Here is the caller graph for this function:



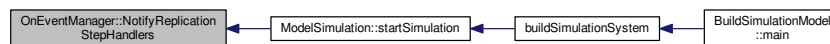
#### 5.46.3.10 void OnEventManager::NotifyReplicationStartHandlers ( SimulationEvent \* se )

Here is the caller graph for this function:



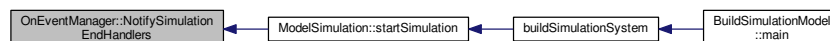
#### 5.46.3.11 void OnEventManager::NotifyReplicationStepHandlers ( SimulationEvent \* se )

Here is the caller graph for this function:



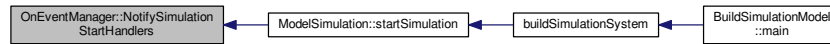
#### 5.46.3.12 void OnEventManager::NotifySimulationEndHandlers ( SimulationEvent \* se )

Here is the caller graph for this function:



## 5.46.3.13 void OnEventManager::NotifySimulationStartHandlers ( SimulationEvent \* se )

Here is the caller graph for this function:



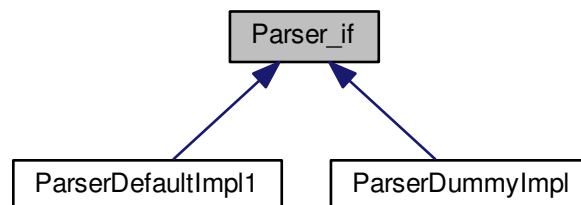
The documentation for this class was generated from the following files:

- [OnEventManager.h](#)
- [OnEventManager.cpp](#)

## 5.47 Parser\_if Class Reference

```
#include <Parser_if.h>
```

Inheritance diagram for Parser\_if:



### Public Member Functions

- virtual double [parse](#) (const std::string expression)=0
- virtual double [parse](#) (const std::string expression, bool \*success, std::string \*errorMessage)=0
- virtual std::string \* [getErrorMessage](#) ()=0

### 5.47.1 Member Function Documentation

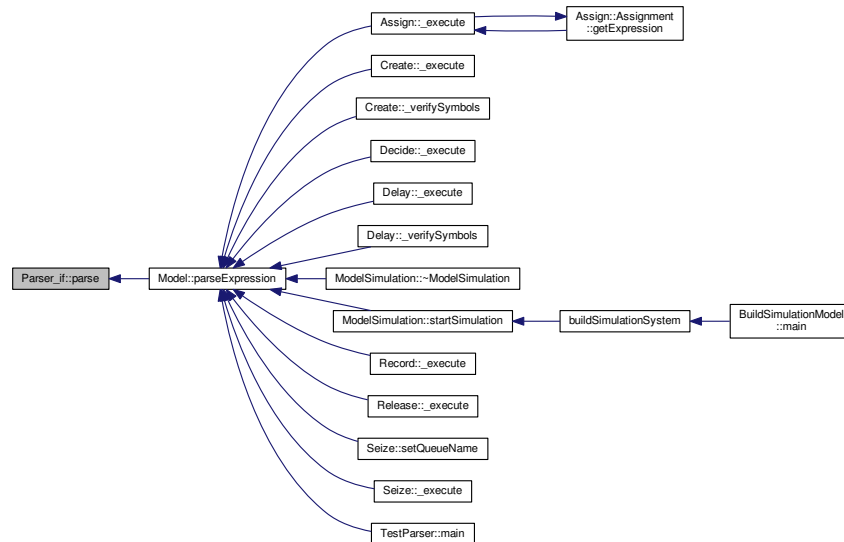
#### 5.47.1.1 virtual std::string\* Parser\_if::getErrorMessage ( ) [pure virtual]

Implemented in [ParserDummyImpl](#), and [ParserDefaultImpl1](#).

5.47.1.2 virtual double Parser\_if::parse ( const std::string *expression* ) [pure virtual]

Implemented in [ParserDummyImpl](#), and [ParserDefaultImpl1](#).

Here is the caller graph for this function:



5.47.1.3 virtual double Parser\_if::parse ( const std::string *expression*, bool \* *success*, std::string \* *errorMessage* ) [pure virtual]

Implemented in [ParserDummyImpl](#), and [ParserDefaultImpl1](#).

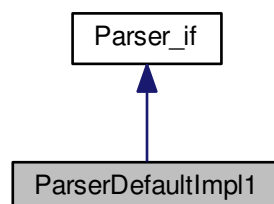
The documentation for this class was generated from the following file:

- [Parser\\_if.h](#)

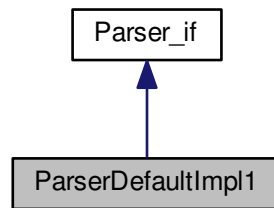
## 5.48 ParserDefaultImpl1 Class Reference

```
#include <ParserDefaultImpl1.h>
```

Inheritance diagram for ParserDefaultImpl1:



Collaboration diagram for ParserDefaultImpl1:



### Public Member Functions

- [ParserDefaultImpl1](#) ([Model](#) \*model)
- [ParserDefaultImpl1](#) (const [ParserDefaultImpl1](#) &orig)
- virtual [~ParserDefaultImpl1](#) ()
- double [parse](#) (const std::string expression)
- double [parse](#) (const std::string expression, bool \*success, std::string \*errorMessage)
- std::string \* [getErrorMessage](#) ()

#### 5.48.1 Constructor & Destructor Documentation

5.48.1.1 `ParserDefaultImpl1::ParserDefaultImpl1 ( Model * model )`

5.48.1.2 `ParserDefaultImpl1::ParserDefaultImpl1 ( const ParserDefaultImpl1 & orig )`

5.48.1.3 `ParserDefaultImpl1::~~ParserDefaultImpl1 ( )` [virtual]

#### 5.48.2 Member Function Documentation

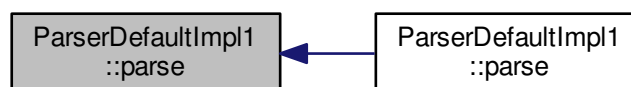
5.48.2.1 `std::string * ParserDefaultImpl1::getErrorMessage ( )` [virtual]

Implements [Parser\\_if](#).

5.48.2.2 `double ParserDefaultImpl1::parse ( const std::string expression )` [virtual]

Implements [Parser\\_if](#).

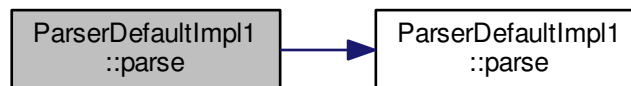
Here is the caller graph for this function:



5.48.2.3 `double ParserDefaultImpl1::parse ( const std::string expression, bool * success, std::string * errorMessage )`  
[virtual]

Implements [Parser\\_if](#).

Here is the call graph for this function:



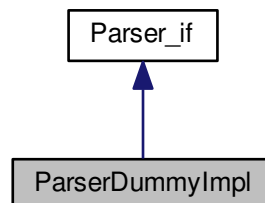
The documentation for this class was generated from the following files:

- [ParserDefaultImpl1.h](#)
- [ParserDefaultImpl1.cpp](#)

## 5.49 ParserDummyImpl Class Reference

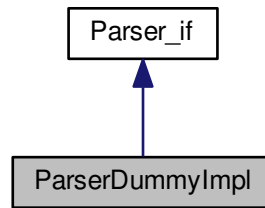
```
#include <ParserDummyImpl.h>
```

Inheritance diagram for ParserDummyImpl:





Collaboration diagram for ParserDummyImpl:



### Public Member Functions

- [ParserDummyImpl](#) ([Model](#) \*model)
- [ParserDummyImpl](#) (const [ParserDummyImpl](#) &orig)
- virtual [~ParserDummyImpl](#) ()
- double [parse](#) (const std::string expression)
- double [parse](#) (const std::string expression, bool \*success, std::string \*errorMessage)
- std::string \* [getErrorMessage](#) ()

#### 5.49.1 Constructor & Destructor Documentation

5.49.1.1 `ParserDummyImpl::ParserDummyImpl ( Model * model )`

5.49.1.2 `ParserDummyImpl::ParserDummyImpl ( const ParserDummyImpl & orig )`

5.49.1.3 `ParserDummyImpl::~~ParserDummyImpl ( )` [virtual]

#### 5.49.2 Member Function Documentation

5.49.2.1 `std::string * ParserDummyImpl::getErrorMessage ( )` [virtual]

Implements [Parser\\_if](#).

5.49.2.2 `double ParserDummyImpl::parse ( const std::string expression )` [virtual]

Implements [Parser\\_if](#).

Here is the caller graph for this function:



5.49.2.3 `double ParserDummyImpl::parse ( const std::string expression, bool * success, std::string * errorMessage )`  
`[virtual]`

Implements [Parser\\_if](#).

Here is the call graph for this function:



The documentation for this class was generated from the following files:

- [ParserDummyImpl.h](#)
- [ParserDummyImpl.cpp](#)

## 5.50 Plugin Class Reference

```
#include <Plugin.h>
```

### Public Member Functions

- [Plugin](#) (std::string name, bool source, bool drain)
- [Plugin](#) (const [Plugin](#) &orig)
- virtual [~Plugin](#) ()
- bool [isDrain](#) () const
- bool [isSource](#) () const

#### 5.50.1 Detailed Description

A [Plugin](#) represents a dynamically linked component class ([ModelComponent](#)) or element class ([ModelElement](#)); It gives access to a [ModelComponent](#) so it can be used by the model. Classes like [Create](#), [Delay](#), and [Dispose](#) are examples of Plugins. It corresponds directly to the "Expansible" part (the capitalized 'E') of the GenESyS acronymous Plugins are NOT implemented yet

## 5.50.2 Constructor & Destructor Documentation

5.50.2.1 `Plugin::Plugin ( std::string name, bool source, bool drain )`

5.50.2.2 `Plugin::Plugin ( const Plugin & orig )`

5.50.2.3 `Plugin::~~Plugin ( )` `[virtual]`

## 5.50.3 Member Function Documentation

5.50.3.1 `bool Plugin::isDrain ( ) const`

5.50.3.2 `bool Plugin::isSource ( ) const`

The documentation for this class was generated from the following files:

- [Plugin.h](#)
- [Plugin.cpp](#)

## 5.51 ProbDistrib Class Reference

```
#include <ProbDistrib.h>
```

### Static Public Member Functions

- static double [uniform](#) (double x, double min, double max)
- static double [exponential](#) (double x, double mean)
- static double [erlang](#) (double x, double mean, double M)
- static double [normal](#) (double x, double mean, double stddev)
- static double [gamma](#) (double x, double mean, double alpha)
- static double [beta](#) (double x, double alpha, double beta)
- static double [weibull](#) (double x, double alpha, double scale)
- static double [logNormal](#) (double x, double mean, double stddev)
- static double [triangular](#) (double x, double min, double mode, double max)

### 5.51.1 Member Function Documentation

5.51.1.1 `double ProbDistrib::beta ( double x, double alpha, double beta )` [static]

5.51.1.2 `double ProbDistrib::erlang ( double x, double mean, double M )` [static]

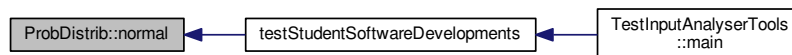
5.51.1.3 `double ProbDistrib::exponential ( double x, double mean )` [static]

5.51.1.4 `double ProbDistrib::gamma ( double x, double mean, double alpha )` [static]

5.51.1.5 `double ProbDistrib::logNormal ( double x, double mean, double stddev )` [static]

5.51.1.6 `double ProbDistrib::normal ( double x, double mean, double stddev )` [static]

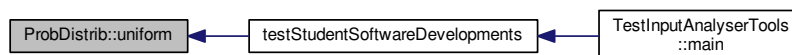
Here is the caller graph for this function:



5.51.1.7 `double ProbDistrib::triangular ( double x, double min, double mode, double max )` [static]

5.51.1.8 `double ProbDistrib::uniform ( double x, double min, double max )` [static]

Here is the caller graph for this function:



5.51.1.9 `double ProbDistrib::weibull ( double x, double alpha, double scale )` [static]

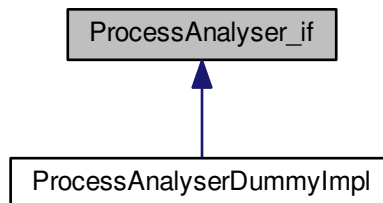
The documentation for this class was generated from the following files:

- [ProbDistrib.h](#)
- [ProbDistrib.cpp](#)

## 5.52 ProcessAnalyser\_if Class Reference

```
#include <ProcessAnalyser_if.h>
```

Inheritance diagram for ProcessAnalyser\_if:



### Public Member Functions

- virtual [List< SimulationScenario \\* >](#) \* [getScenarios](#) () const =0
- virtual [List< SimulationControl \\* >](#) \* [getControls](#) () const =0
- virtual [List< SimulationResponse \\* >](#) \* [getResponses](#) () const =0
- virtual [List< SimulationControl \\* >](#) \* [extractControlsFromModel](#) (std::string modelName) const =0
- virtual [List< SimulationResponse \\* >](#) \* [extractResponsesFromModel](#) (std::string modelName) const =0
- virtual void [startSimulationOfScenario](#) (SimulationScenario \*scenario)=0
- virtual void [startSimulation](#) ()=0
- virtual void [stopSimulation](#) ()=0
- virtual void [addTraceSimulationHandler](#) (traceSimulationProcessListener traceSimulationProcessListener)=0

### 5.52.1 Detailed Description

The process analyser allows to extract controls and responses from a model, include some of them as controls and responses for a set of scenarios to be simulated

### 5.52.2 Member Function Documentation

**5.52.2.1** virtual void `ProcessAnalyser_if::addTraceSimulationHandler` ( `traceSimulationProcessListener` `traceSimulationProcessListener` ) `[pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

**5.52.2.2** virtual `List<SimulationControl*>` \* `ProcessAnalyser_if::extractControlsFromModel` ( `std::string modelName` ) `const` `[pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

5.52.2.3 `virtual List<SimulationResponse*>* ProcessAnalyser_if::extractResponsesFromModel ( std::string  
modelFilename ) const [pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

5.52.2.4 `virtual List<SimulationControl*>* ProcessAnalyser_if::getControls ( ) const [pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

5.52.2.5 `virtual List<SimulationResponse*>* ProcessAnalyser_if::getResponses ( ) const [pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

5.52.2.6 `virtual List<SimulationScenario*>* ProcessAnalyser_if::getScenarios ( ) const [pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

5.52.2.7 `virtual void ProcessAnalyser_if::startSimulation ( ) [pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

5.52.2.8 `virtual void ProcessAnalyser_if::startSimulationOfScenario ( SimulationScenario * scenario ) [pure  
virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

5.52.2.9 `virtual void ProcessAnalyser_if::stopSimulation ( ) [pure virtual]`

Implemented in [ProcessAnalyserDummyImpl](#).

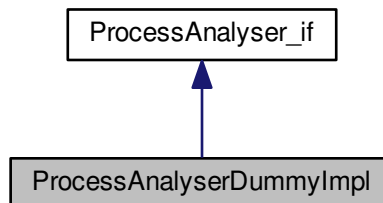
The documentation for this class was generated from the following file:

- [ProcessAnalyser\\_if.h](#)

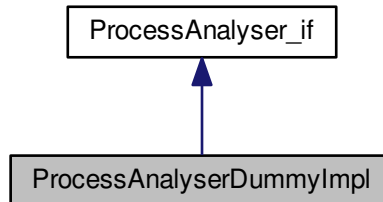
## 5.53 ProcessAnalyserDummyImpl Class Reference

```
#include <ProcessAnalyserDummyImpl.h>
```

Inheritance diagram for ProcessAnalyserDummyImpl:



Collaboration diagram for ProcessAnalyserDummyImpl:



### Public Member Functions

- [ProcessAnalyserDummyImpl](#) ()
- [ProcessAnalyserDummyImpl](#) (const [ProcessAnalyserDummyImpl](#) &orig)
- [~ProcessAnalyserDummyImpl](#) ()
- [List< SimulationScenario \\* > \\* getScenarios](#) () const
- [List< SimulationControl \\* > \\* getControls](#) () const
- [List< SimulationResponse \\* > \\* getResponses](#) () const
- [List< SimulationControl \\* > \\* extractControlsFromModel](#) (std::string modelName) const
- [List< SimulationResponse \\* > \\* extractResponsesFromModel](#) (std::string modelName) const
- void [startSimulationOfScenario](#) ([SimulationScenario](#) \*scenario)
- void [startSimulation](#) ()
- void [stopSimulation](#) ()
- void [addTraceSimulationHandler](#) ([traceSimulationProcessListener](#) [traceSimulationProcessListener](#))

### 5.53.1 Constructor & Destructor Documentation

5.53.1.1 `ProcessAnalyserDummyImpl::ProcessAnalyserDummyImpl ( )`

5.53.1.2 `ProcessAnalyserDummyImpl::ProcessAnalyserDummyImpl ( const ProcessAnalyserDummyImpl & orig )`

5.53.1.3 `ProcessAnalyserDummyImpl::~~ProcessAnalyserDummyImpl ( )`

### 5.53.2 Member Function Documentation

5.53.2.1 `void ProcessAnalyserDummyImpl::addTraceSimulationHandler ( traceSimulationProcessListener  
traceSimulationProcessListener ) [virtual]`

Implements [ProcessAnalyser\\_if](#).

5.53.2.2 `List< SimulationControl * > * ProcessAnalyserDummyImpl::extractControlsFromModel ( std::string  
modelName ) const [virtual]`

Implements [ProcessAnalyser\\_if](#).

5.53.2.3 `List< SimulationResponse * > * ProcessAnalyserDummyImpl::extractResponsesFromModel ( std::string  
modelName ) const [virtual]`

Implements [ProcessAnalyser\\_if](#).

5.53.2.4 `List< SimulationControl * > * ProcessAnalyserDummyImpl::getControls ( ) const [virtual]`

Implements [ProcessAnalyser\\_if](#).

5.53.2.5 `List< SimulationResponse * > * ProcessAnalyserDummyImpl::getResponses ( ) const [virtual]`

Implements [ProcessAnalyser\\_if](#).

5.53.2.6 `List< SimulationScenario * > * ProcessAnalyserDummyImpl::getScenarios ( ) const [virtual]`

Implements [ProcessAnalyser\\_if](#).

5.53.2.7 `void ProcessAnalyserDummyImpl::startSimulation ( ) [virtual]`

Implements [ProcessAnalyser\\_if](#).



5.53.2.8 `void ProcessAnalyserDummyImpl::startSimulationOfScenario ( SimulationScenario * scenario )` [virtual]

Implements [ProcessAnalyser\\_if](#).

5.53.2.9 `void ProcessAnalyserDummyImpl::stopSimulation ( )` [virtual]

Implements [ProcessAnalyser\\_if](#).

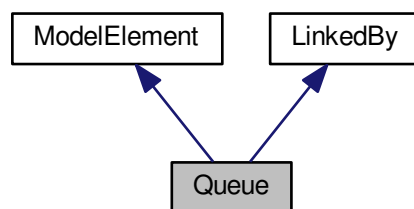
The documentation for this class was generated from the following files:

- [ProcessAnalyserDummyImpl.h](#)
- [ProcessAnalyserDummyImpl.cpp](#)

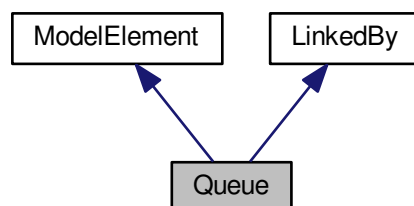
## 5.54 Queue Class Reference

```
#include <Queue.h>
```

Inheritance diagram for Queue:



Collaboration diagram for Queue:



## Public Types

- enum `OrderRule` : int { `OrderRule::FIFO` = 1, `OrderRule::LIFO` = 2, `OrderRule::HIGHESTVALUE` = 3, `OrderRule::SMALLESTVALUE` = 4 }

## Public Member Functions

- `Queue` (`ElementManager` \*elems)
- `Queue` (`ElementManager` \*elems, std::string name)
- `Queue` (const `Queue` &orig)
- virtual `~Queue` ()
- virtual std::string `show` ()
- void `insertElement` (`Waiting` \*element)
- void `removeElement` (`Waiting` \*element, double tnow)
- unsigned int `size` ()
- `Waiting` \* `first` ()
- void `setAttributeName` (std::string \_attributeName)
- std::string `getAttributeName` () const
- void `setOrderRule` (`OrderRule` \_orderRule)
- `Queue::OrderRule` `getOrderRule` () const

## Protected Member Functions

- virtual void `_loadInstance` (std::list< std::string > words)
- virtual std::list< std::string > \* `_saveInstance` ()
- virtual bool `_verifySymbols` (std::string \*errorMessage)

## Additional Inherited Members

### 5.54.1 Member Enumeration Documentation

5.54.1.1 enum `Queue::OrderRule` : int [strong]

Enumerator

***FIFO***

***LIFO***

***HIGHESTVALUE***

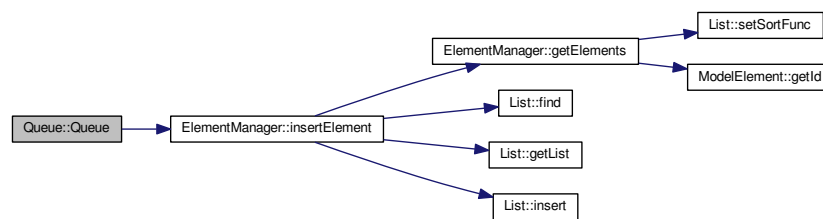
***SMALLESTVALUE***

## 5.54.2 Constructor & Destructor Documentation

### 5.54.2.1 Queue::Queue ( ElementManager \* *elems* )

### 5.54.2.2 Queue::Queue ( ElementManager \* *elems*, std::string *name* )

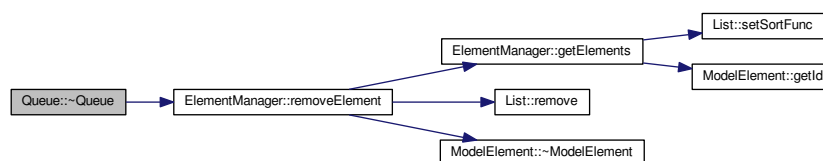
Here is the call graph for this function:



### 5.54.2.3 Queue::Queue ( const Queue & *orig* )

### 5.54.2.4 Queue::~~Queue ( ) [virtual]

Here is the call graph for this function:



## 5.54.3 Member Function Documentation

### 5.54.3.1 void Queue::\_loadInstance ( std::list< std::string > *words* ) [protected], [virtual]

Implements [ModelElement](#).

5.54.3.2 `std::list< std::string > * Queue::_saveInstance ( )` [protected],[virtual]

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



5.54.3.3 `bool Queue::_verifySymbols ( std::string * errorMessage )` [protected],[virtual]

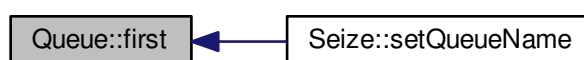
Implements [ModelElement](#).

5.54.3.4 `Waiting * Queue::first ( )`

Here is the call graph for this function:



Here is the caller graph for this function:

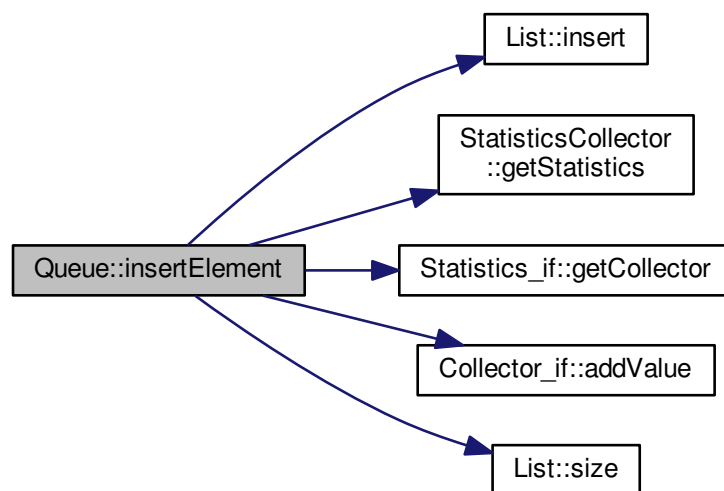


5.54.3.5 `std::string Queue::getAttributeName ( ) const`

5.54.3.6 `Queue::OrderRule Queue::getOrderRule ( ) const`

5.54.3.7 `void Queue::insertElement ( Waiting * element )`

Here is the call graph for this function:

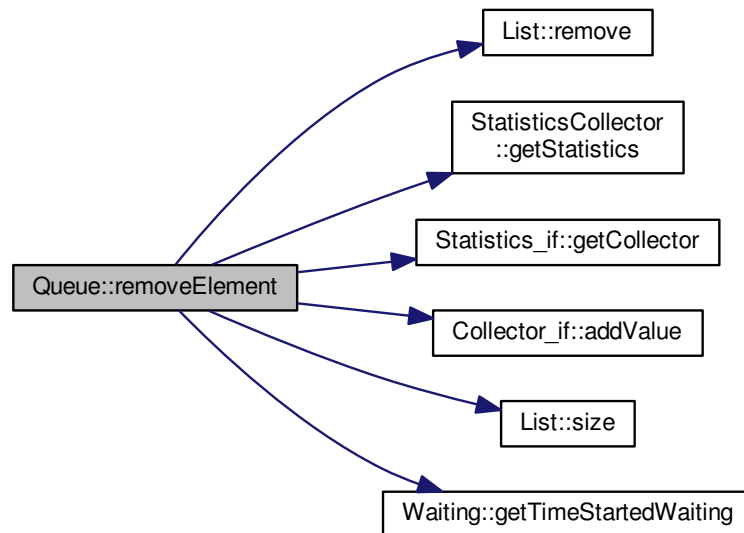


Here is the caller graph for this function:



#### 5.54.3.8 void Queue::removeElement ( Waiting \* *element*, double *tnow* )

Here is the call graph for this function:



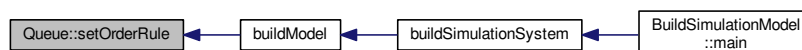
Here is the caller graph for this function:



#### 5.54.3.9 void Queue::setAttributeName ( std::string *\_attributeName* )

#### 5.54.3.10 void Queue::setOrderRule ( OrderRule *\_orderRule* )

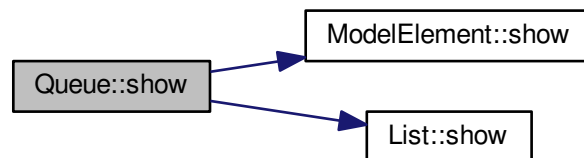
Here is the caller graph for this function:



#### 5.54.3.11 `std::string Queue::show ( )` [virtual]

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



#### 5.54.3.12 `unsigned int Queue::size ( )`

Here is the call graph for this function:



Here is the caller graph for this function:



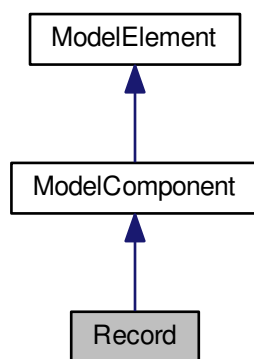
The documentation for this class was generated from the following files:

- [Queue.h](#)
- [Queue.cpp](#)

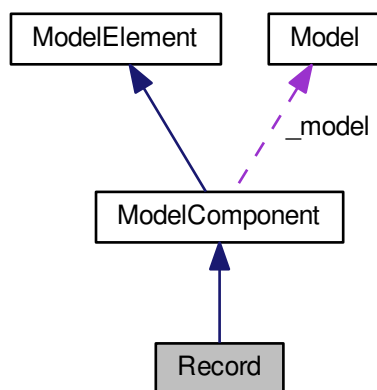
## 5.55 Record Class Reference

```
#include <Record.h>
```

Inheritance diagram for Record:



Collaboration diagram for Record:



### Public Member Functions

- [Record](#) ([Model](#) \*model)
- [Record](#) (const [Record](#) &orig)
- virtual [~Record](#) ()
- void [setFilename](#) (std::string filename)



- std::string [getFilename](#) () const
- void [setExpression](#) (std::string expression)
- std::string [getExpression](#) () const
- void [setExpressionName](#) (std::string expressionName)
- std::string [getExpressionName](#) () const
- [StatisticsCollector](#) \* [getCstatExpression](#) () const
- virtual std::string [show](#) ()

### Protected Member Functions

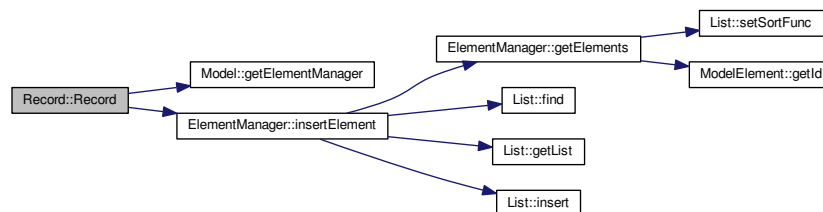
- virtual void [\\_execute](#) ([Entity](#) \*entity)
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

### Additional Inherited Members

#### 5.55.1 Constructor & Destructor Documentation

##### 5.55.1.1 `Record::Record ( Model * model )`

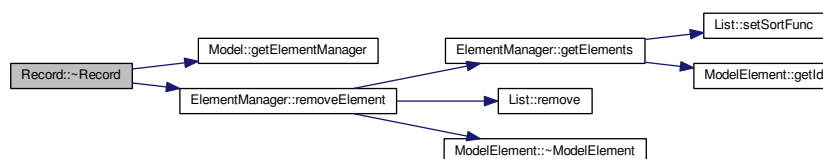
Here is the call graph for this function:



##### 5.55.1.2 `Record::Record ( const Record & orig )`

##### 5.55.1.3 `Record::~Record ( ) [virtual]`

Here is the call graph for this function:

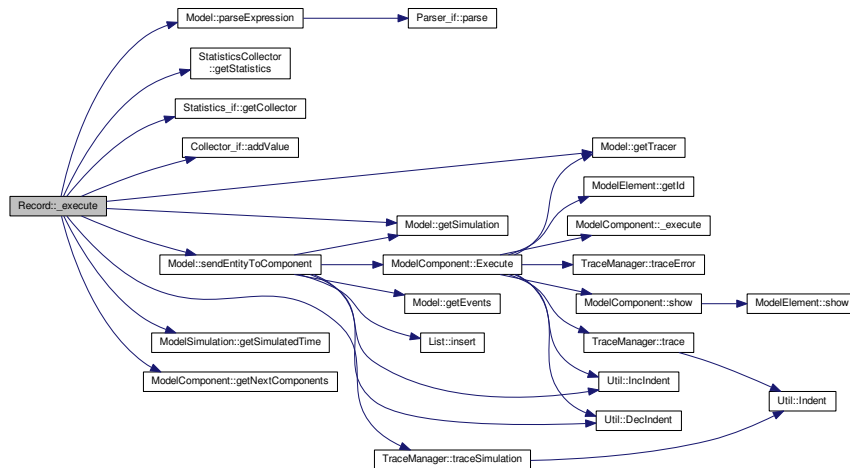


## 5.55.2 Member Function Documentation

### 5.55.2.1 void Record::\_execute ( Entity \* entity ) [protected], [virtual]

Implements [ModelComponent](#).

Here is the call graph for this function:



### 5.55.2.2 void Record::\_loadInstance ( std::list< std::string > words ) [protected], [virtual]

Implements [ModelElement](#).

### 5.55.2.3 std::list< std::string > \* Record::\_saveInstance ( ) [protected], [virtual]

Reimplemented from [ModelComponent](#).

### 5.55.2.4 bool Record::\_verifySymbols ( std::string \* errorMessage ) [protected], [virtual]

Implements [ModelElement](#).

### 5.55.2.5 StatisticsCollector \* Record::getCstatExpression ( ) const

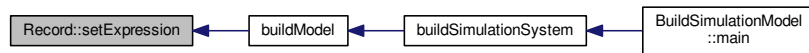
### 5.55.2.6 std::string Record::getExpression ( ) const

### 5.55.2.7 std::string Record::getExpressionName ( ) const

### 5.55.2.8 std::string Record::getFilename ( ) const

#### 5.55.2.9 void Record::setExpression ( std::string *expression* )

Here is the caller graph for this function:



#### 5.55.2.10 void Record::setExpressionName ( std::string *expressionName* )

Here is the call graph for this function:

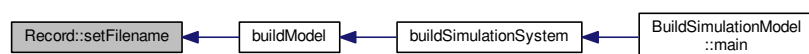


Here is the caller graph for this function:



#### 5.55.2.11 void Record::setFilename ( std::string *filename* )

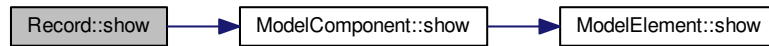
Here is the caller graph for this function:



#### 5.55.2.12 `std::string Record::show ( )` [virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



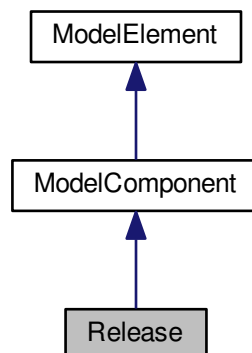
The documentation for this class was generated from the following files:

- [Record.h](#)
- [Record.cpp](#)

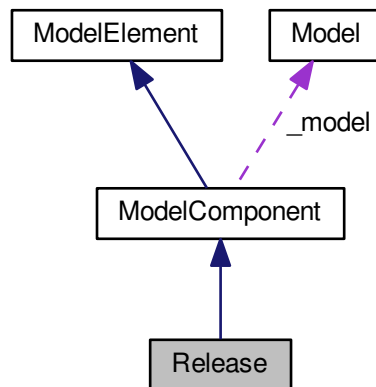
## 5.56 Release Class Reference

```
#include <Release.h>
```

Inheritance diagram for Release:



Collaboration diagram for Release:



## Public Member Functions

- [Release](#) ([Model](#) \*model)
- [Release](#) (const [Release](#) &orig)
- virtual [~Release](#) ()
- virtual std::string [show](#) ()
- void [setPriority](#) (unsigned short \_priority)
- unsigned short [getPriority](#) () const
- void [setResourceType](#) ([Resource::ResourceType](#) \_resourceType)
- [Resource::ResourceType](#) [getResourceType](#) () const
- void [setQuantity](#) (std::string \_quantity)
- std::string [getQuantity](#) () const
- void [setRule](#) ([Resource::ResourceRule](#) \_rule)
- [Resource::ResourceRule](#) [getRule](#) () const
- void [setSaveAttribute](#) (std::string \_saveAttribute)
- std::string [getSaveAttribute](#) () const
- void [setResource](#) ([Resource](#) \*\_resource)
- [Resource](#) \* [getResource](#) () const
- void [setResourceName](#) (std::string resourceName) throw ()
- std::string [getResourceName](#) () const

## Protected Member Functions

- virtual void [\\_execute](#) ([Entity](#) \*entity)
- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.56.1 Constructor & Destructor Documentation

5.56.1.1 `Release::Release ( Model * model )`

5.56.1.2 `Release::Release ( const Release & orig )`

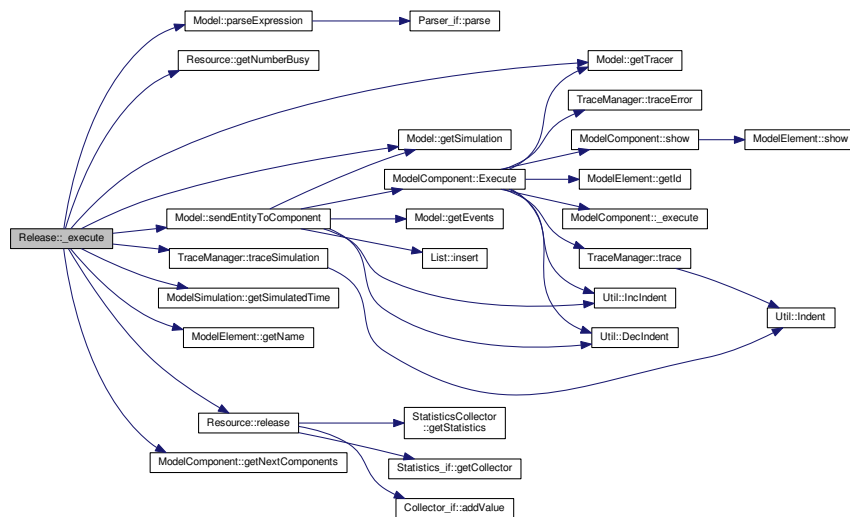
5.56.1.3 `Release::~~Release ( )` `[virtual]`

### 5.56.2 Member Function Documentation

5.56.2.1 `void Release::_execute ( Entity * entity )` `[protected]`, `[virtual]`

Implements [ModelComponent](#).

Here is the call graph for this function:



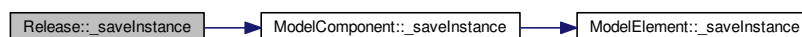
5.56.2.2 `void Release::_loadInstance ( std::list< std::string > words )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

5.56.2.3 `std::list< std::string > * Release::_saveInstance ( )` `[protected]`, `[virtual]`

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



5.56.2.4 `bool Release::_verifySymbols ( std::string * errorMessage )` `[protected], [virtual]`

Implements [ModelElement](#).

5.56.2.5 `unsigned short Release::getPriority ( ) const`

5.56.2.6 `std::string Release::getQuantity ( ) const`

5.56.2.7 `Resource * Release::getResource ( ) const`

5.56.2.8 `std::string Release::getResourceName ( ) const`

Here is the call graph for this function:



5.56.2.9 `Resource::ResourceType Release::getResourceType ( ) const`

5.56.2.10 `Resource::ResourceRule Release::getRule ( ) const`

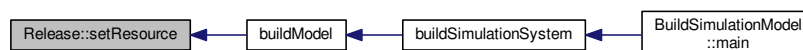
5.56.2.11 `std::string Release::getSaveAttribute ( ) const`

5.56.2.12 `void Release::setPriority ( unsigned short _priority )`

5.56.2.13 `void Release::setQuantity ( std::string _quantity )`

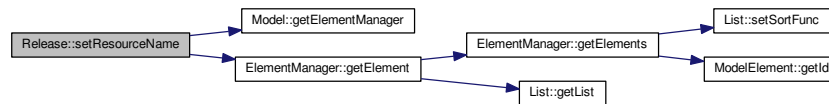
5.56.2.14 `void Release::setResource ( Resource * _resource )`

Here is the caller graph for this function:



5.56.2.15 `void Release::setResourceName ( std::string resourceName ) throw (`

Here is the call graph for this function:



5.56.2.16 `void Release::setResourceType ( Resource::ResourceType _resourceType )`

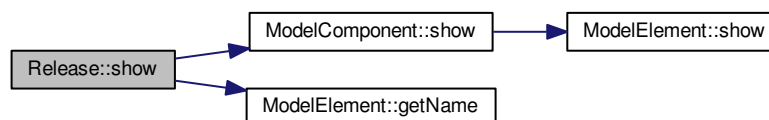
5.56.2.17 `void Release::setRule ( Resource::ResourceRule _rule )`

5.56.2.18 `void Release::setSaveAttribute ( std::string _saveAttribute )`

5.56.2.19 `std::string Release::show ( ) [virtual]`

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



The documentation for this class was generated from the following files:

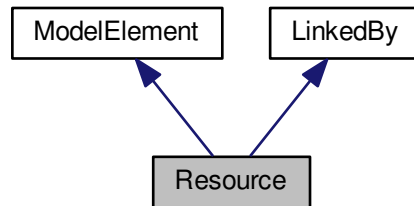
- [Release.h](#)
- [Release.cpp](#)



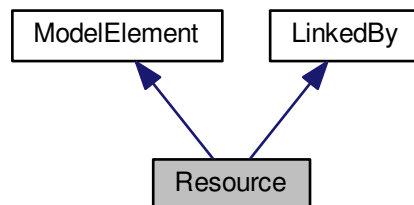
## 5.57 Resource Class Reference

```
#include <Resource.h>
```

Inheritance diagram for Resource:



Collaboration diagram for Resource:



### Public Types

- enum **ResourceType** : int { **ResourceType::SET** = 1, **ResourceType::RESOURCE** = 2 }
- enum **ResourceRule** : int { **ResourceRule::RANDOM** = 1, **ResourceRule::CICLICAL** = 2, **ResourceRule::ESPECIFIC** = 3, **ResourceRule::SMALLESTBUSY** = 4, **ResourceRule::LARGESTREMAININGCAPACITY** = 5 }
- enum **ResourceState** : int { **ResourceState::IDLE** = 1, **ResourceState::BUSY** = 2, **ResourceState::FAILED** = 3, **ResourceState::INACTIVE** = 4, **ResourceState::OTHER** = 5 }
- typedef std::function< void(**Resource** \*) > **ResourceEventHandler**

## Public Member Functions

- [Resource](#) ([ElementManager](#) \*elems)
- [Resource](#) ([ElementManager](#) \*elems, std::string name)
- [Resource](#) (const [Resource](#) &orig)
- virtual [~Resource](#) ()
- virtual std::string [show](#) ()
- void [seize](#) (unsigned int quantity, double tnow)
- void [release](#) (unsigned int quantity, double tnow)
- void [setResourceState](#) ([ResourceState](#) \_resourceState)
- [Resource::ResourceState](#) [getResourceState](#) () const
- void [setCapacity](#) (unsigned int \_capacity)
- unsigned int [getCapacity](#) () const
- void [setCostBusyHour](#) (double \_costBusyHour)
- double [getCostBusyHour](#) () const
- void [setCostIdleHour](#) (double \_costIdleHour)
- double [getCostIdleHour](#) () const
- void [setCostPerUse](#) (double \_costPerUse)
- double [getCostPerUse](#) () const
- unsigned int [getNumberBusy](#) () const
- unsigned int [getNumberOut](#) () const
- void [addResourceEventHandler](#) ([ResourceEventHandler](#) eventHandler)

## Static Public Member Functions

- template<typename Class >  
static [ResourceEventHandler](#) [SetResourceEventHandler](#) (void(Class::\*function)([Resource](#) \*), Class \*object)

## Protected Member Functions

- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.57.1 Member Typedef Documentation

5.57.1.1 typedef std::function<void([Resource](#)\*) > [Resource::ResourceEventHandler](#)

### 5.57.2 Member Enumeration Documentation

5.57.2.1 enum [Resource::ResourceRule](#) : int [strong]

Enumerator

***RANDOM***

***CICLICAL***

***ESPECIFIC***

***SMALLESTBUSY***

***LARGESTREMAININGCAPACITY***

5.57.2.2 `enum Resource::ResourceState : int [strong]`

Enumerator

**IDLE**  
**BUSY**  
**FAILED**  
**INACTIVE**  
**OTHER**

5.57.2.3 `enum Resource::ResourceType : int [strong]`

Enumerator

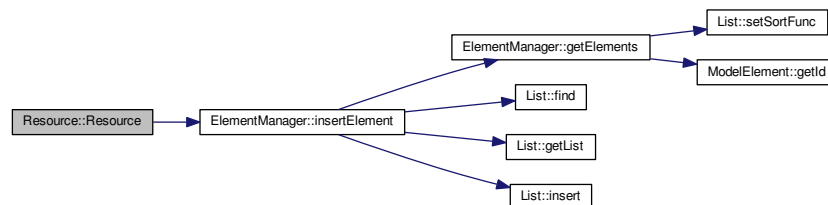
**SET**  
**RESOURCE**

### 5.57.3 Constructor & Destructor Documentation

5.57.3.1 `Resource::Resource ( ElementManager * elems )`

5.57.3.2 `Resource::Resource ( ElementManager * elems, std::string name )`

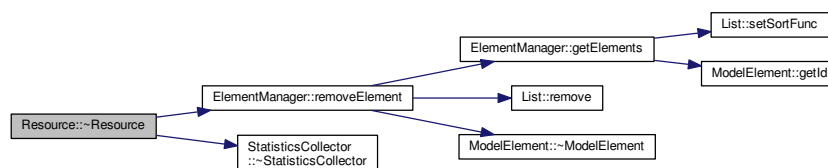
Here is the call graph for this function:



5.57.3.3 `Resource::Resource ( const Resource & orig )`

5.57.3.4 `Resource::~~Resource ( ) [virtual]`

Here is the call graph for this function:



### 5.57.4 Member Function Documentation

5.57.4.1 `void Resource::_loadInstance ( std::list< std::string > words )` [protected],[virtual]

Implements [ModelElement](#).

5.57.4.2 `std::list< std::string > * Resource::_saveInstance ( )` [protected],[virtual]

Reimplemented from [ModelElement](#).

Here is the call graph for this function:

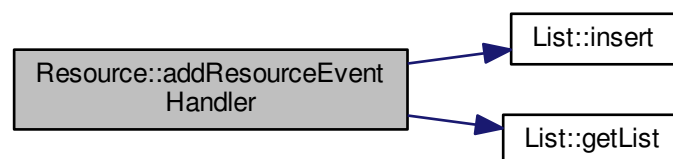


5.57.4.3 `bool Resource::_verifySymbols ( std::string * errorMessage )` [protected],[virtual]

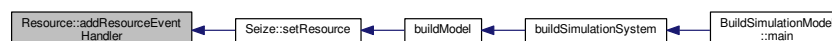
Implements [ModelElement](#).

5.57.4.4 `void Resource::addResourceEventHandler ( ResourceEventHandler eventHandler )`

Here is the call graph for this function:

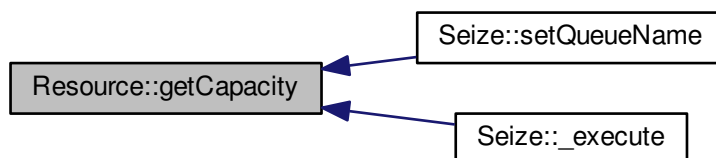


Here is the caller graph for this function:



## 5.57.4.5 unsigned int Resource::getCapacity ( ) const

Here is the caller graph for this function:



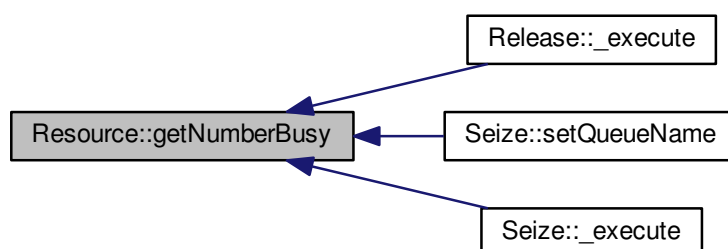
## 5.57.4.6 double Resource::getCostBusyHour ( ) const

## 5.57.4.7 double Resource::getCostIdleHour ( ) const

## 5.57.4.8 double Resource::getCostPerUse ( ) const

## 5.57.4.9 unsigned int Resource::getNumberBusy ( ) const

Here is the caller graph for this function:

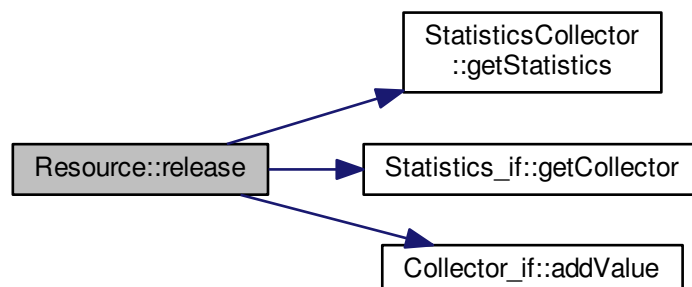


## 5.57.4.10 unsigned int Resource::getNumberOut ( ) const

## 5.57.4.11 Resource::ResourceState Resource::getResourceState ( ) const

#### 5.57.4.12 void Resource::release ( unsigned int *quantity*, double *tnow* )

Here is the call graph for this function:

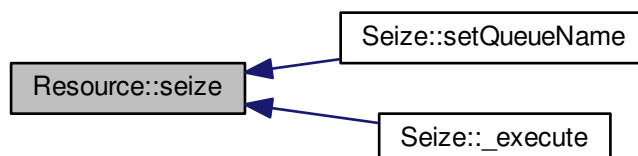


Here is the caller graph for this function:



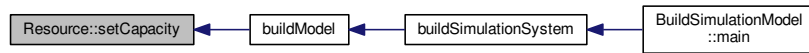
#### 5.57.4.13 void Resource::seize ( unsigned int *quantity*, double *tnow* )

Here is the caller graph for this function:



5.57.4.14 `void Resource::setCapacity ( unsigned int _capacity )`

Here is the caller graph for this function:



5.57.4.15 `void Resource::setCostBusyHour ( double _costBusyHour )`

5.57.4.16 `void Resource::setCostIdleHour ( double _costIdleHour )`

5.57.4.17 `void Resource::setCostPerUse ( double _costPerUse )`

5.57.4.18 `template<typename Class > static ResourceEventHandler Resource::SetResourceEventHandler ( void(Class::*)(Resource *) function, Class * object ) [inline], [static]`

5.57.4.19 `void Resource::setResourceState ( ResourceState _resourceState )`

5.57.4.20 `std::string Resource::show ( ) [virtual]`

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



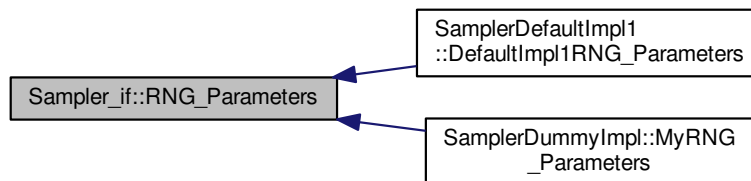
The documentation for this class was generated from the following files:

- [Resource.h](#)
- [Resource.cpp](#)

## 5.58 `Sampler_if::RNG_Parameters` Class Reference

```
#include <Sampler_if.h>
```

Inheritance diagram for `Sampler_if::RNG_Parameters`:



### 5.58.1 Detailed Description

class that encapsulates attributes required to generate random numbers, which depends on the generation method used.

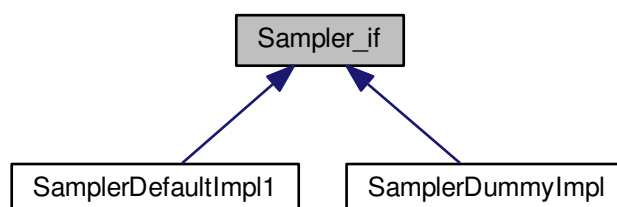
The documentation for this class was generated from the following file:

- [Sampler\\_if.h](#)

## 5.59 `Sampler_if` Class Reference

```
#include <Sampler_if.h>
```

Inheritance diagram for `Sampler_if`:



### Classes

- class [RNG\\_Parameters](#)



## Public Member Functions

- virtual double [random](#) ()=0
- virtual double [sampleUniform](#) (double min, double max)=0
- virtual double [sampleExponential](#) (double mean)=0
- virtual double [sampleErlang](#) (double mean, int M)=0
- virtual double [sampleNormal](#) (double mean, double stddev)=0
- virtual double [sampleGamma](#) (double mean, double alpha)=0
- virtual double [sampleBeta](#) (double alpha, double beta, double infLimit, double supLimit)=0
- virtual double [sampleWeibull](#) (double alpha, double scale)=0
- virtual double [sampleLogNormal](#) (double mean, double stddev)=0
- virtual double [sampleTriangular](#) (double min, double mode, double max)=0
- virtual double [sampleDiscrete](#) (double value, double acumProb,...)=0
- virtual void [setRNGparameters](#) ([RNG\\_Parameters](#) \*param)=0
- virtual [RNG\\_Parameters](#) \* [getRNGparameters](#) () const =0

### 5.59.1 Detailed Description

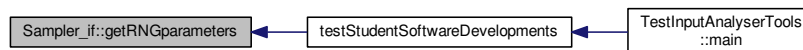
Interface that describes the methods to be implemented by classes that generate random values that follow a specific probability distribution.

### 5.59.2 Member Function Documentation

#### 5.59.2.1 virtual [RNG\\_Parameters](#)\* [Sampler\\_if::getRNGparameters](#) ( ) const [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

Here is the caller graph for this function:



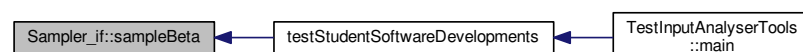
#### 5.59.2.2 virtual double [Sampler\\_if::random](#) ( ) [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

#### 5.59.2.3 virtual double [Sampler\\_if::sampleBeta](#) ( double *alpha*, double *beta*, double *infLimit*, double *supLimit* ) [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

Here is the caller graph for this function:



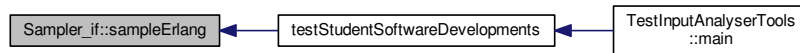
5.59.2.4 virtual double `Sampler_if::sampleDiscrete ( double value, double acumProb, ... )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

5.59.2.5 virtual double `Sampler_if::sampleErlang ( double mean, int M )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

Here is the caller graph for this function:



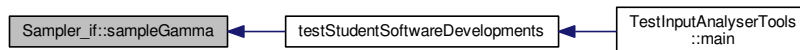
5.59.2.6 virtual double `Sampler_if::sampleExponential ( double mean )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

5.59.2.7 virtual double `Sampler_if::sampleGamma ( double mean, double alpha )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

Here is the caller graph for this function:



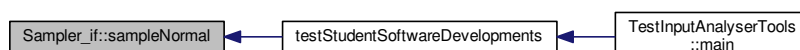
5.59.2.8 virtual double `Sampler_if::sampleLogNormal ( double mean, double stddev )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

5.59.2.9 virtual double `Sampler_if::sampleNormal ( double mean, double stddev )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

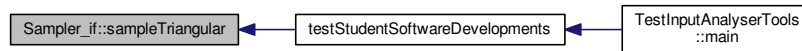
Here is the caller graph for this function:



5.59.2.10 `virtual double Sampler_if::sampleTriangular ( double min, double mode, double max )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

Here is the caller graph for this function:



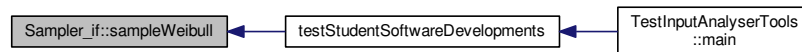
5.59.2.11 `virtual double Sampler_if::sampleUniform ( double min, double max )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

5.59.2.12 `virtual double Sampler_if::sampleWeibull ( double alpha, double scale )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

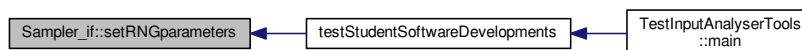
Here is the caller graph for this function:



5.59.2.13 `virtual void Sampler_if::setRNGparameters ( RNG_Parameters * param )` [pure virtual]

Implemented in [SamplerDefaultImpl1](#), and [SamplerDummyImpl](#).

Here is the caller graph for this function:



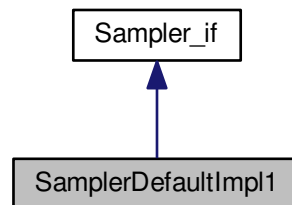
The documentation for this class was generated from the following file:

- [Sampler\\_if.h](#)

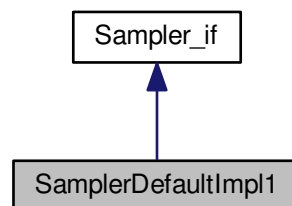
## 5.60 `SamplerDefaultImpl1` Class Reference

```
#include <SamplerDefaultImpl1.h>
```

Inheritance diagram for `SamplerDefaultImpl1`:



Collaboration diagram for `SamplerDefaultImpl1`:



### Classes

- class [DefaultImpl1RNG\\_Parameters](#)

### Public Member Functions

- [SamplerDefaultImpl1](#) ()
- [SamplerDefaultImpl1](#) (const [SamplerDefaultImpl1](#) &orig)
- virtual [~SamplerDefaultImpl1](#) ()
- double [random](#) ()
- double [sampleUniform](#) (double min, double max)
- double [sampleExponential](#) (double mean)
- double [sampleErlang](#) (double mean, int M)
- double [sampleNormal](#) (double mean, double stddev)
- double [sampleGamma](#) (double mean, double alpha)

- double [sampleBeta](#) (double alpha, double beta, double infLimit, double supLimit)
- double [sampleWeibull](#) (double alpha, double scale)
- double [sampleLogNormal](#) (double mean, double stddev)
- double [sampleTriangular](#) (double min, double mode, double max)
- double [sampleDiscrete](#) (double value, double acumProb,...)
- void [reset](#) ()  
*reinitialize seed and other parameters so (pseudo) random number sequence will be generated again.*
- void [setRNGparameters](#) (RNG\_Parameters \*param)
- RNG\_Parameters \* [getRNGparameters](#) () const

## 5.60.1 Constructor & Destructor Documentation

### 5.60.1.1 SamplerDefaultImpl1::SamplerDefaultImpl1 ( )

Here is the call graph for this function:



### 5.60.1.2 SamplerDefaultImpl1::SamplerDefaultImpl1 ( const SamplerDefaultImpl1 & orig )

### 5.60.1.3 SamplerDefaultImpl1::~SamplerDefaultImpl1 ( ) [virtual]

## 5.60.2 Member Function Documentation

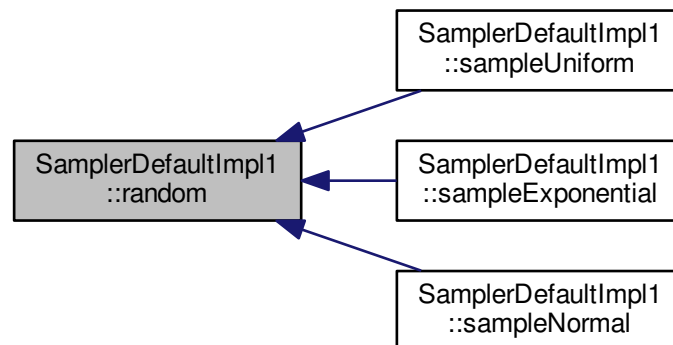
### 5.60.2.1 Sampler\_if::RNG\_Parameters \* SamplerDefaultImpl1::getRNGparameters ( ) const [virtual]

Implements [Sampler\\_if](#).

### 5.60.2.2 double SamplerDefaultImpl1::random ( ) [virtual]

Implements [Sampler\\_if](#).

Here is the caller graph for this function:



#### 5.60.2.3 void SamplerDefaultImpl1::reset ( )

reinitialize seed and other parameters so (pseudo) random number sequence will be generated again.

Here is the caller graph for this function:



#### 5.60.2.4 double SamplerDefaultImpl1::sampleBeta ( double *alpha*, double *beta*, double *infLimit*, double *supLimit* ) [virtual]

Implements [Sampler\\_if](#).

#### 5.60.2.5 double SamplerDefaultImpl1::sampleDiscrete ( double *value*, double *acumProb*, ... ) [virtual]

Implements [Sampler\\_if](#).

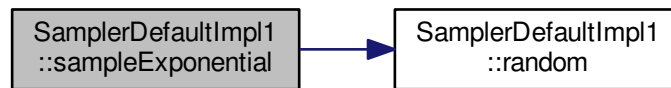
#### 5.60.2.6 double SamplerDefaultImpl1::sampleErlang ( double *mean*, int *M* ) [virtual]

Implements [Sampler\\_if](#).

5.60.2.7 `double SamplerDefaultImpl1::sampleExponential ( double mean )` `[virtual]`

Implements [Sampler\\_if](#).

Here is the call graph for this function:



5.60.2.8 `double SamplerDefaultImpl1::sampleGamma ( double mean, double alpha )` `[virtual]`

Implements [Sampler\\_if](#).

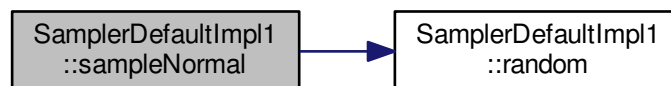
5.60.2.9 `double SamplerDefaultImpl1::sampleLogNormal ( double mean, double stddev )` `[virtual]`

Implements [Sampler\\_if](#).

5.60.2.10 `double SamplerDefaultImpl1::sampleNormal ( double mean, double stddev )` `[virtual]`

Implements [Sampler\\_if](#).

Here is the call graph for this function:



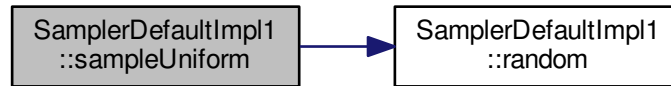
5.60.2.11 `double SamplerDefaultImpl1::sampleTriangular ( double min, double mode, double max )` `[virtual]`

Implements [Sampler\\_if](#).

5.60.2.12 `double SamplerDefaultImpl1::sampleUniform ( double min, double max )` [virtual]

Implements [Sampler\\_if](#).

Here is the call graph for this function:



5.60.2.13 `double SamplerDefaultImpl1::sampleWeibull ( double alpha, double scale )` [virtual]

Implements [Sampler\\_if](#).

5.60.2.14 `void SamplerDefaultImpl1::setRNGparameters ( Sampler\_if::RNG\_Parameters * param )` [virtual]

Implements [Sampler\\_if](#).

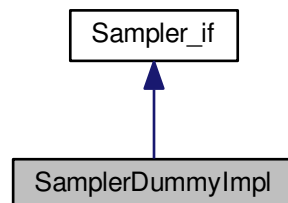
The documentation for this class was generated from the following files:

- [SamplerDefaultImpl1.h](#)
- [SamplerDefaultImpl1.cpp](#)

## 5.61 [SamplerDummyImpl](#) Class Reference

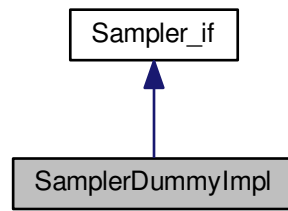
```
#include <SamplerDummyImpl.h>
```

Inheritance diagram for [SamplerDummyImpl](#):





Collaboration diagram for SamplerDummyImpl:



## Classes

- class [MyRNG\\_Parameters](#)

## Public Member Functions

- [SamplerDummyImpl](#) ()
- [SamplerDummyImpl](#) (const [SamplerDummyImpl](#) &orig)
- [~SamplerDummyImpl](#) ()
- double [random](#) ()
- double [sampleUniform](#) (double min, double max)
- double [sampleExponential](#) (double mean)
- double [sampleErlang](#) (double mean, int M)
- double [sampleNormal](#) (double mean, double stddev)
- double [sampleGamma](#) (double mean, double alpha)
- double [sampleBeta](#) (double alpha, double beta, double infLimit, double supLimit)
- double [sampleWeibull](#) (double alpha, double scale)
- double [sampleLogNormal](#) (double mean, double stddev)
- double [sampleTriangular](#) (double min, double mode, double max)
- double [sampleDiscrete](#) (double value, double acumProb,...)
- void [setRNGparameters](#) ([RNG\\_Parameters](#) \*param)
- [RNG\\_Parameters](#) \* [getRNGparameters](#) () const

### 5.61.1 Constructor & Destructor Documentation

5.61.1.1 `SamplerDummyImpl::SamplerDummyImpl ( )`

5.61.1.2 `SamplerDummyImpl::SamplerDummyImpl ( const SamplerDummyImpl & orig )`

5.61.1.3 `SamplerDummyImpl::~~SamplerDummyImpl ( )`

### 5.61.2 Member Function Documentation

5.61.2.1 `Sampler\_if::RNG\_Parameters * SamplerDummyImpl::getRNGparameters ( ) const` `[virtual]`

Implements [Sampler\\_if](#).

5.61.2.2 `double SamplerDummyImpl::random ( )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.3 `double SamplerDummyImpl::sampleBeta ( double alpha, double beta, double infLimit, double supLimit )`  
[virtual]

Implements [Sampler\\_if](#).

5.61.2.4 `double SamplerDummyImpl::sampleDiscrete ( double value, double acumProb, ... )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.5 `double SamplerDummyImpl::sampleErlang ( double mean, int M )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.6 `double SamplerDummyImpl::sampleExponential ( double mean )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.7 `double SamplerDummyImpl::sampleGamma ( double mean, double alpha )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.8 `double SamplerDummyImpl::sampleLogNormal ( double mean, double stddev )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.9 `double SamplerDummyImpl::sampleNormal ( double mean, double stddev )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.10 `double SamplerDummyImpl::sampleTriangular ( double min, double mode, double max )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.11 `double SamplerDummyImpl::sampleUniform ( double min, double max )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.12 `double SamplerDummyImpl::sampleWeibull ( double alpha, double scale )` [virtual]

Implements [Sampler\\_if](#).

5.61.2.13 `void SamplerDummyImpl::setRNGparameters ( Sampler_if::RNG_Parameters * param )` [virtual]

Implements [Sampler\\_if](#).

The documentation for this class was generated from the following files:

- [SamplerDummyImpl.h](#)
- [SamplerDummyImpl.cpp](#)

## 5.62 ScenarioExperiment\_if Class Reference

```
#include <ScenarioExperiment_if.h>
```

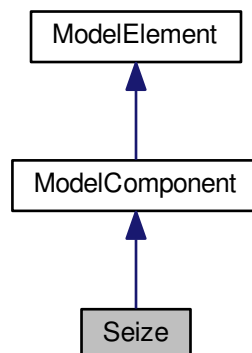
The documentation for this class was generated from the following file:

- [ScenarioExperiment\\_if.h](#)

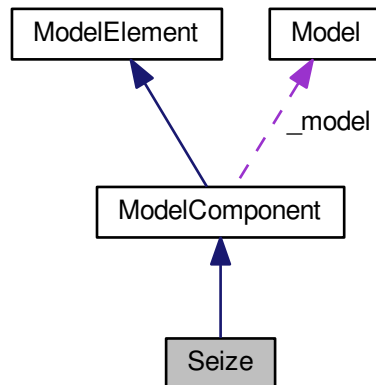
## 5.63 Seize Class Reference

```
#include <Seize.h>
```

Inheritance diagram for Seize:



Collaboration diagram for Seize:



## Public Member Functions

- [Seize](#) ([Model](#) \*model)
- [Seize](#) (const [Seize](#) &orig)
- virtual [~Seize](#) ()
- virtual std::string [show](#) ()
- void [setLastMemberSeized](#) (unsigned int \_lastMemberSeized)
- unsigned int [getLastMemberSeized](#) () const
- void [setSaveAttribute](#) (std::string \_saveAttribute)
- std::string [getSaveAttribute](#) () const
- void [setRule](#) ([Resource::ResourceRule](#) \_rule)
- [Resource::ResourceRule](#) [getRule](#) () const
- void [setQuantity](#) (std::string \_quantity)
- std::string [getQuantity](#) () const
- void [setResourceType](#) ([Resource::ResourceType](#) \_resourceType)
- [Resource::ResourceType](#) [getResourceType](#) () const
- void [setPriority](#) (unsigned short \_priority)
- unsigned short [getPriority](#) () const
- void [setAllocationType](#) (unsigned int \_allocationType)
- unsigned int [getAllocationType](#) () const
- void [setResourceName](#) (std::string \_resourceName) throw ()
- std::string [getResourceName](#) () const
- void [setQueueName](#) (std::string queueName) throw ()
- std::string [getQueueName](#) () const
- void [setResource](#) ([Resource](#) \*resource)
- [Resource](#) \* [getResource](#) () const
- void [setQueue](#) ([Queue](#) \*queue)
- [Queue](#) \* [getQueue](#) () const

## Protected Member Functions

- virtual void `_execute` (`Entity *entity`)
- virtual void `_loadInstance` (`std::list< std::string > words`)
- virtual `std::list< std::string > * _saveInstance` ()
- virtual bool `_verifySymbols` (`std::string *errorMessage`)

## Additional Inherited Members

### 5.63.1 Detailed Description

`Seize` tries to allocate a certain amount of a resource

### 5.63.2 Constructor & Destructor Documentation

5.63.2.1 `Seize::Seize ( Model * model )`

5.63.2.2 `Seize::Seize ( const Seize & orig )`

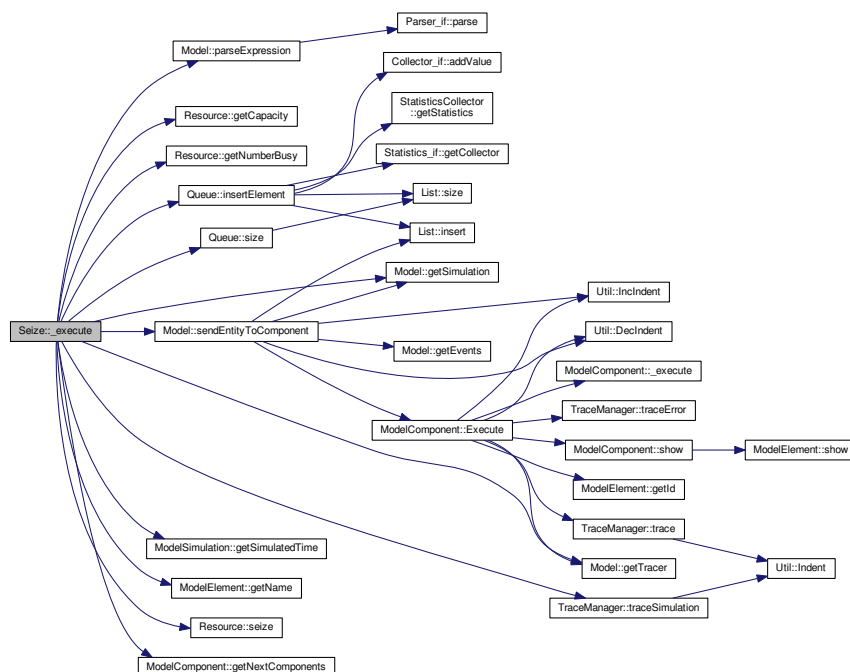
5.63.2.3 `Seize::~Seize ( )` [virtual]

### 5.63.3 Member Function Documentation

5.63.3.1 `void Seize::_execute ( Entity * entity )` [protected], [virtual]

Implements `ModelComponent`.

Here is the call graph for this function:



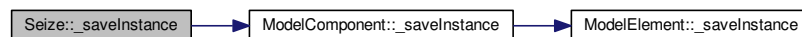
5.63.3.2 `void Seize::_loadInstance ( std::list< std::string > words )` [protected],[virtual]

Implements [ModelElement](#).

5.63.3.3 `std::list< std::string > * Seize::_saveInstance ( )` [protected],[virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



5.63.3.4 `bool Seize::_verifySymbols ( std::string * errorMessage )` [protected],[virtual]

Implements [ModelElement](#).

5.63.3.5 `unsigned int Seize::getAllocationType ( ) const`

5.63.3.6 `unsigned int Seize::getLastMemberSeized ( ) const`

5.63.3.7 `unsigned short Seize::getPriority ( ) const`

5.63.3.8 `std::string Seize::getQuantity ( ) const`

5.63.3.9 `Queue * Seize::getQueue ( ) const`

5.63.3.10 `std::string Seize::getQueueName ( ) const`

Here is the call graph for this function:



5.63.3.11 **Resource \*** Seize::getResource ( ) const

5.63.3.12 **std::string** Seize::getResourceName ( ) const

Here is the call graph for this function:



5.63.3.13 **Resource::ResourceType** Seize::getResourceType ( ) const

5.63.3.14 **Resource::ResourceRule** Seize::getRule ( ) const

5.63.3.15 **std::string** Seize::getSaveAttribute ( ) const

5.63.3.16 **void** Seize::setAllocationType ( unsigned int *\_allocationType* )

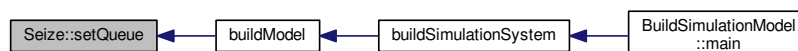
5.63.3.17 **void** Seize::setLastMemberSeized ( unsigned int *\_lastMemberSeized* )

5.63.3.18 **void** Seize::setPriority ( unsigned short *\_priority* )

5.63.3.19 **void** Seize::setQuantity ( **std::string** *\_quantity* )

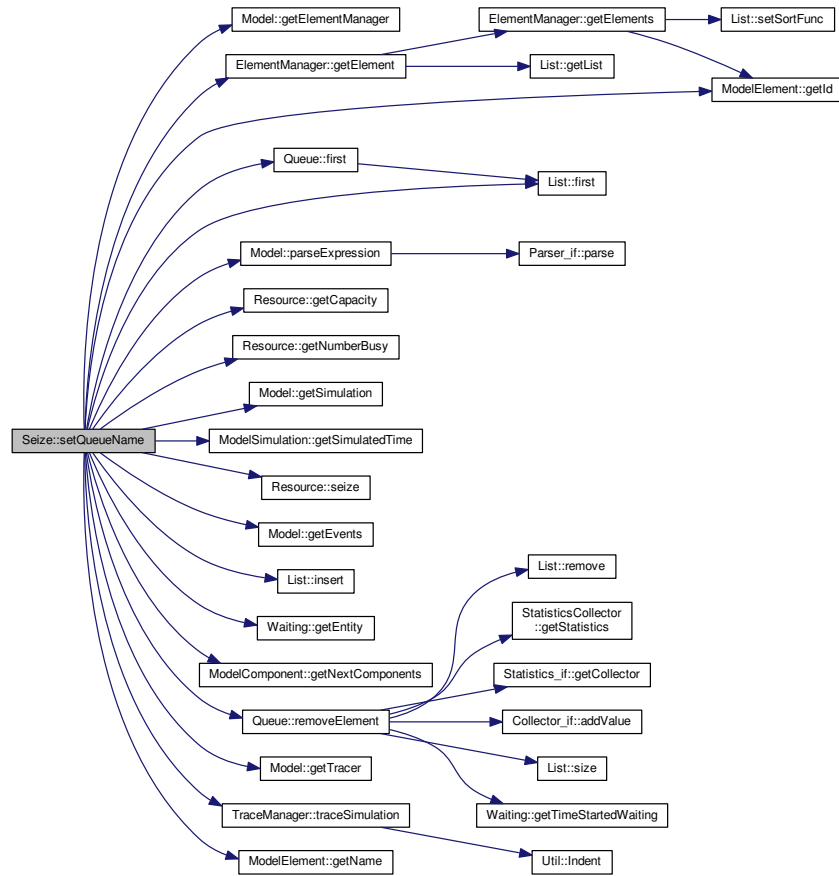
5.63.3.20 **void** Seize::setQueue ( **Queue \*** *queue* )

Here is the caller graph for this function:



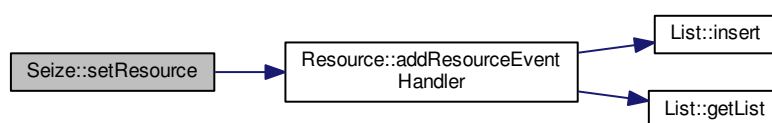
### 5.63.3.21 void Seize::setQueueName ( std::string *queueName* ) throw )

Here is the call graph for this function:



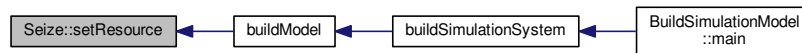
### 5.63.3.22 void Seize::setResource ( Resource \* *resource* )

Here is the call graph for this function:



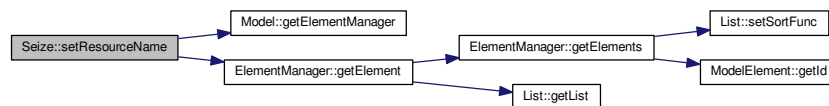


Here is the caller graph for this function:



#### 5.63.3.23 void Seize::setResourceName ( std::string \_resourceName ) throw )

Here is the call graph for this function:



#### 5.63.3.24 void Seize::setResourceType ( Resource::ResourceType \_resourceType )

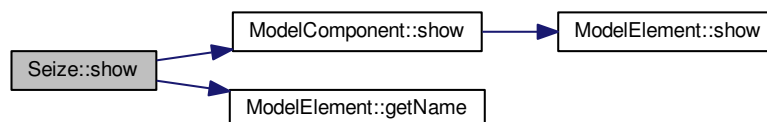
#### 5.63.3.25 void Seize::setRule ( Resource::ResourceRule \_rule )

#### 5.63.3.26 void Seize::setSaveAttribute ( std::string \_saveAttribute )

#### 5.63.3.27 std::string Seize::show ( ) [virtual]

Reimplemented from [ModelComponent](#).

Here is the call graph for this function:



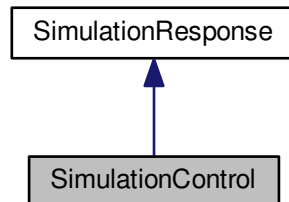
The documentation for this class was generated from the following files:

- [Seize.h](#)
- [Seize.cpp](#)

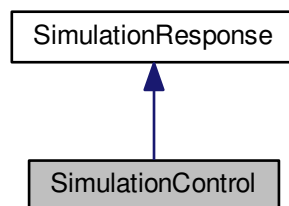
## 5.64 SimulationControl Class Reference

```
#include <SimulationControl.h>
```

Inheritance diagram for SimulationControl:



Collaboration diagram for SimulationControl:



### Public Member Functions

- [SimulationControl](#) (std::string type, std::string name, [GetterMember](#) getterMember, [SetterMember](#) setter↔Member)
- [SimulationControl](#) (const [SimulationControl](#) &orig)
- virtual [~SimulationControl](#) ()
- void [setValue](#) (double value)

### Additional Inherited Members

#### 5.64.1 Detailed Description

Represents any possible parameter or control for a simulation. Any element or event the model can declare one of its own attribute as a simulation control. It just have to create a [SimulationControl](#) object, passing the access to the methods that gets and sets the control value and including this [SimulationControl](#) in the corresponding list of the model

### 5.64.2 Constructor & Destructor Documentation

5.64.2.1 `SimulationControl::SimulationControl ( std::string type, std::string name, GetterMember getterMember, SetterMember setterMember )`

5.64.2.2 `SimulationControl::SimulationControl ( const SimulationControl & orig )`

5.64.2.3 `SimulationControl::~SimulationControl ( ) [virtual]`

### 5.64.3 Member Function Documentation

5.64.3.1 `void SimulationControl::setValue ( double value )`

The documentation for this class was generated from the following files:

- [SimulationControl.h](#)
- [SimulationControl.cpp](#)

## 5.65 SimulationEvent Class Reference

```
#include <OnEventManager.h>
```

### Public Member Functions

- [SimulationEvent](#) (unsigned int *replicationNumber*, [Event](#) \**event*)
- unsigned int [getReplicationNumber](#) () const
- [Event](#) \* [getEventProcessed](#) () const

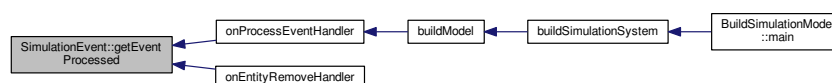
### 5.65.1 Constructor & Destructor Documentation

5.65.1.1 `SimulationEvent::SimulationEvent ( unsigned int replicationNumber, Event * event ) [inline]`

### 5.65.2 Member Function Documentation

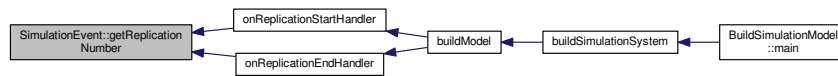
5.65.2.1 `Event* SimulationEvent::getEventProcessed ( ) const [inline]`

Here is the caller graph for this function:



### 5.65.2.2 unsigned int SimulationEvent::getReplicationNumber ( ) const [inline]

Here is the caller graph for this function:



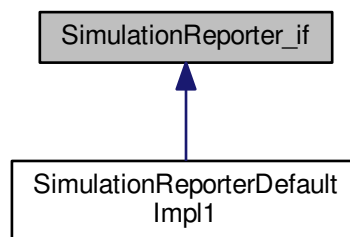
The documentation for this class was generated from the following file:

- [OnEventManager.h](#)

## 5.66 SimulationReporter\_if Class Reference

```
#include <SimulationReporter_if.h>
```

Inheritance diagram for SimulationReporter\_if:



### Public Member Functions

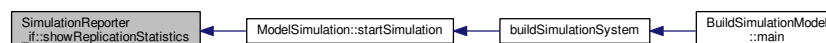
- virtual void [showReplicationStatistics](#) ()=0
- virtual void [showSimulationStatistics](#) ()=0

### 5.66.1 Member Function Documentation

#### 5.66.1.1 virtual void SimulationReporter\_if::showReplicationStatistics ( ) [pure virtual]

Implemented in [SimulationReporterDefaultImpl1](#).

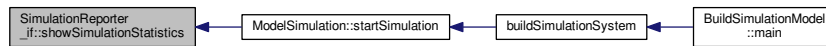
Here is the caller graph for this function:



5.66.1.2 `virtual void SimulationReporter_if::showSimulationStatistics ( ) [pure virtual]`

Implemented in [SimulationReporterDefaultImpl1](#).

Here is the caller graph for this function:



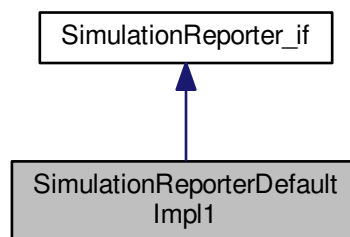
The documentation for this class was generated from the following file:

- [SimulationReporter\\_if.h](#)

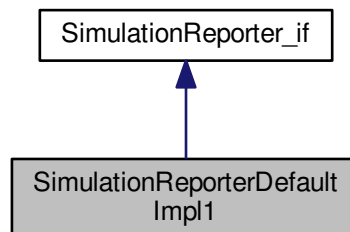
## 5.67 SimulationReporterDefaultImpl1 Class Reference

```
#include <SimulationReporterDefaultImpl1.h>
```

Inheritance diagram for `SimulationReporterDefaultImpl1`:



Collaboration diagram for `SimulationReporterDefaultImpl1`:



## Public Member Functions

- [SimulationReporterDefaultImpl1](#) ([ModelSimulation](#) \*simulation, [Model](#) \*model)
- [SimulationReporterDefaultImpl1](#) (const [SimulationReporterDefaultImpl1](#) &orig)
- virtual [~SimulationReporterDefaultImpl1](#) ()
- virtual void [showReplicationStatistics](#) ()
- virtual void [showSimulationStatistics](#) ()

### 5.67.1 Constructor & Destructor Documentation

5.67.1.1 [SimulationReporterDefaultImpl1::SimulationReporterDefaultImpl1](#) ( [ModelSimulation](#) \* *simulation*, [Model](#) \* *model* )

5.67.1.2 [SimulationReporterDefaultImpl1::SimulationReporterDefaultImpl1](#) ( const [SimulationReporterDefaultImpl1](#) & *orig* )

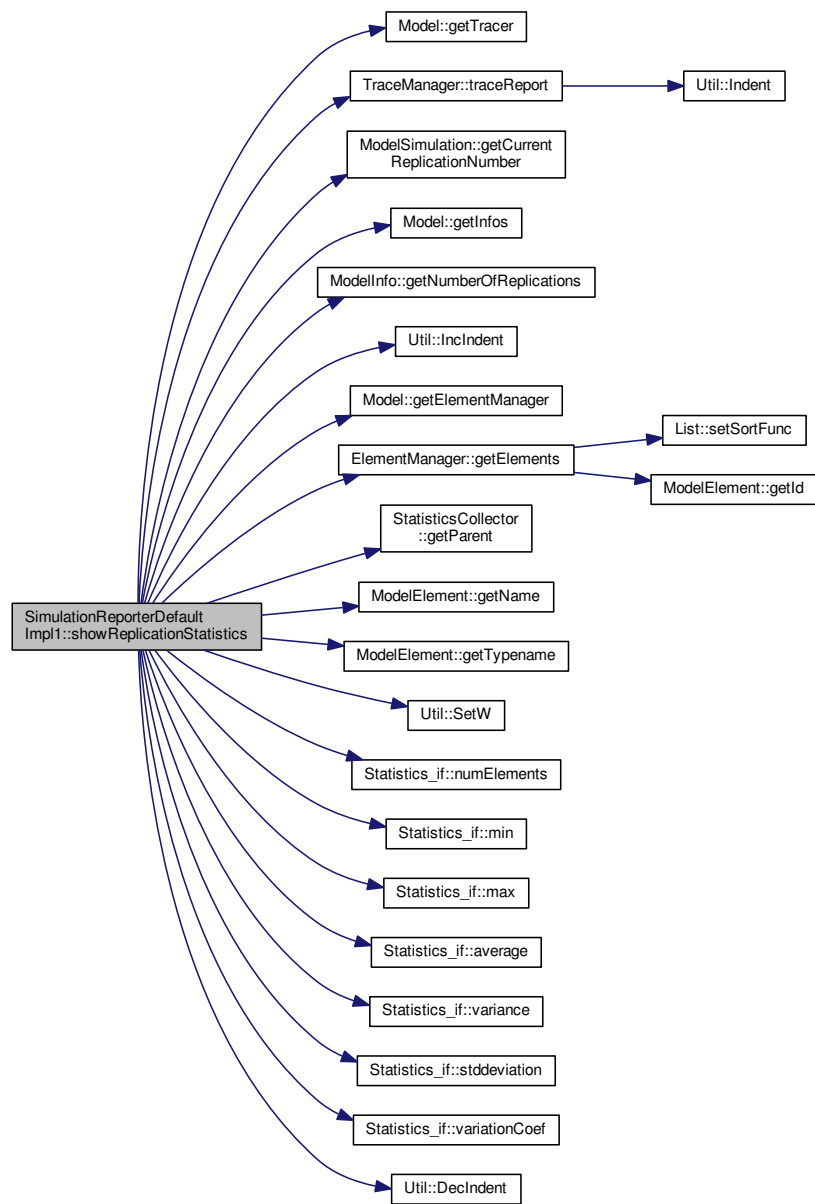
5.67.1.3 [SimulationReporterDefaultImpl1::~~SimulationReporterDefaultImpl1](#) ( ) [virtual]

### 5.67.2 Member Function Documentation

5.67.2.1 void [SimulationReporterDefaultImpl1::showReplicationStatistics](#) ( ) [virtual]

Implements [SimulationReporter\\_if](#).

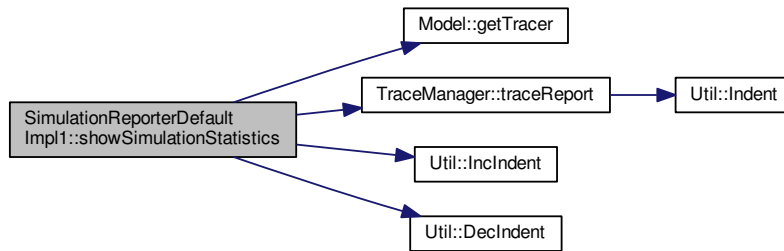
Here is the call graph for this function:



5.67.2.2 `void SimulationReporterDefaultImpl1::showSimulationStatistics ( ) [virtual]`

Implements [SimulationReporter\\_if](#).

Here is the call graph for this function:



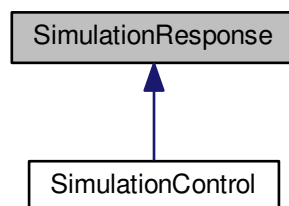
The documentation for this class was generated from the following files:

- [SimulationReporterDefaultImpl1.h](#)
- [SimulationReporterDefaultImpl1.cpp](#)

## 5.68 SimulationResponse Class Reference

```
#include <SimulationResponse.h>
```

Inheritance diagram for SimulationResponse:



### Public Member Functions

- [SimulationResponse](#) (std::string type, std::string name, [GetterMember](#) getterMember)
- [SimulationResponse](#) (const [SimulationResponse](#) &orig)
- virtual [~SimulationResponse](#) ()
- double [getValue](#) ()
- std::string [getName](#) () const
- std::string [getType](#) () const



## Protected Attributes

- `std::string _type`
- `std::string _name`
- `GetterMember _getterMemberFunction`

### 5.68.1 Detailed Description

Represents any possible response of a simulation. Any element or event the model can declare one of its own attribute as a simulation response. It just have to create a [SimulationResponse](#) object, passing the access to the method that gets the response value and including this [SimulationResponse](#) in the corresponding list of the model

### 5.68.2 Constructor & Destructor Documentation

5.68.2.1 `SimulationResponse::SimulationResponse ( std::string type, std::string name, GetterMember getterMember )`

5.68.2.2 `SimulationResponse::SimulationResponse ( const SimulationResponse & orig )`

5.68.2.3 `SimulationResponse::~SimulationResponse ( ) [virtual]`

### 5.68.3 Member Function Documentation

5.68.3.1 `std::string SimulationResponse::getName ( ) const`

5.68.3.2 `std::string SimulationResponse::getType ( ) const`

5.68.3.3 `double SimulationResponse::getValue ( )`

### 5.68.4 Member Data Documentation

5.68.4.1 `GetterMember SimulationResponse::_getterMemberFunction [protected]`

5.68.4.2 `std::string SimulationResponse::_name [protected]`

5.68.4.3 `std::string SimulationResponse::_type [protected]`

The documentation for this class was generated from the following files:

- [SimulationResponse.h](#)
- [SimulationResponse.cpp](#)

## 5.69 SimulationScenario Class Reference

```
#include <SimulationScenario.h>
```

## Public Member Functions

- [SimulationScenario](#) ()
- [SimulationScenario](#) (const [SimulationScenario](#) &orig)
- virtual [~SimulationScenario](#) ()
- void [setName](#) (std::string \_name)
- std::string [getName](#) () const
- std::list< double > \* [getResponseValues](#) () const
- std::list< double > \* [getControlValues](#) () const
- void [setModelFilename](#) (std::string \_modelFilename)
- std::string [getModelFilename](#) () const
- double [getResponseValue](#) ([SimulationResponse](#) \*value)
- double [getControlValue](#) ([SimulationControl](#) \*control)
- void [setControlValue](#) ([SimulationControl](#) \*control, double value)

### 5.69.1 Detailed Description

Represents a scenario where a specific model (defined my ModelFilename) will be simulated. To each scenario will be associated a set of [SimulationControl](#) and [SimulationResponse](#), and their values are set to the scenario by the ProcessAnalyser.

### 5.69.2 Constructor & Destructor Documentation

5.69.2.1 [SimulationScenario::SimulationScenario](#) ( )

5.69.2.2 [SimulationScenario::SimulationScenario](#) ( const [SimulationScenario](#) & orig )

5.69.2.3 [SimulationScenario::~~SimulationScenario](#) ( ) [virtual]

### 5.69.3 Member Function Documentation

5.69.3.1 double [SimulationScenario::getControlValue](#) ( [SimulationControl](#) \* control )

5.69.3.2 std::list< double > \* [SimulationScenario::getControlValues](#) ( ) const

5.69.3.3 std::string [SimulationScenario::getModelFilename](#) ( ) const

5.69.3.4 std::string [SimulationScenario::getName](#) ( ) const

5.69.3.5 double [SimulationScenario::getResponseValue](#) ( [SimulationResponse](#) \* value )

5.69.3.6 std::list< double > \* [SimulationScenario::getResponseValues](#) ( ) const

5.69.3.7 void [SimulationScenario::setControlValue](#) ( [SimulationControl](#) \* control, double value )

5.69.3.8 void [SimulationScenario::setModelFilename](#) ( std::string \_modelFilename )

5.69.3.9 void [SimulationScenario::setName](#) ( std::string \_name )

The documentation for this class was generated from the following files:

- [SimulationScenario.h](#)
- [SimulationScenario.cpp](#)

## 5.70 Simulator Class Reference

```
#include <Simulator.h>
```

### Public Member Functions

- [Simulator](#) ()
- [Simulator](#) (const [Simulator](#) &orig)
- virtual [~Simulator](#) ()
- [List< Model \\* > \\* getModels](#) () const  
*Returns the list of open models in the simulator.*
- [List< Plugin \\* > \\* getPlugins](#) () const
- std::string [getVersion](#) () const
- std::string [getLicense](#) () const
- std::string [getName](#) () const
- [Sampler\\_if \\* getSampler](#) () const  
*Returns the Sampler, used to generate samples accordingly to a probability distribution.*
- [Fitter\\_if \\* getFitter](#) () const  
*Returns the fitter, responsible for carrying out tests of adherence of theoretical distributions of probability with sampled data.*

### 5.70.1 Detailed Description

The main class of the ReGenesys KERNEL simulation. It gives access to simulation models and tools.

### 5.70.2 Constructor & Destructor Documentation

5.70.2.1 [Simulator::Simulator \( \)](#)

5.70.2.2 [Simulator::Simulator \( const Simulator & orig \)](#)

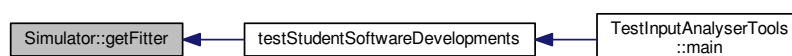
5.70.2.3 [Simulator::~~Simulator \( \)](#) [virtual]

### 5.70.3 Member Function Documentation

5.70.3.1 [Fitter\\_if \\* Simulator::getFitter \( \)](#) const

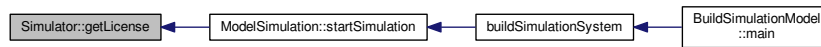
Returns the fitter, responsible for carrying out tests of adherence of theoretical distributions of probability with sampled data.

Here is the caller graph for this function:



### 5.70.3.2 `std::string Simulator::getLicense ( ) const`

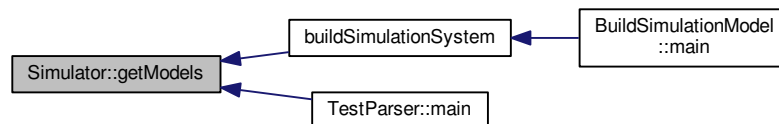
Here is the caller graph for this function:



### 5.70.3.3 `List< Model * > * Simulator::getModels ( ) const`

Returns the list of open models in the simulator.

Here is the caller graph for this function:



### 5.70.3.4 `std::string Simulator::getName ( ) const`

Here is the caller graph for this function:

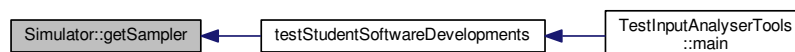


### 5.70.3.5 `List< Plugin * > * Simulator::getPlugins ( ) const`

### 5.70.3.6 `Sampler_if * Simulator::getSampler ( ) const`

Returns the Sampler, used to generate samples accordingly to a probability distribution.

Here is the caller graph for this function:



5.70.3.7 `std::string Simulator::getVersion ( ) const`

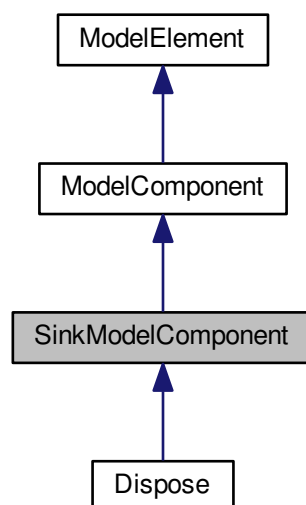
The documentation for this class was generated from the following files:

- [Simulator.h](#)
- [Simulator.cpp](#)

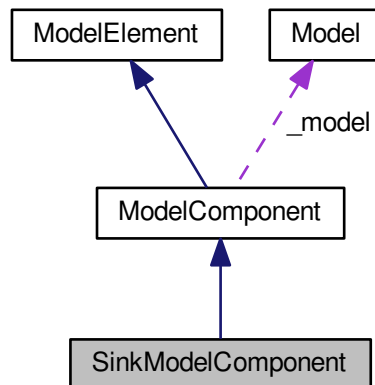
## 5.71 SinkModelComponent Class Reference

```
#include <SinkModelComponent.h>
```

Inheritance diagram for SinkModelComponent:



Collaboration diagram for SinkModelComponent:



## Public Member Functions

- [SinkModelComponent](#) ([Model](#) \*model, std::string componentTypename)
- [SinkModelComponent](#) (const [SinkModelComponent](#) &orig)
- virtual [~SinkModelComponent](#) ()
- void [setCollectStatistics](#) (bool \_collectStatistics)
- bool [isCollectStatistics](#) () const

## Additional Inherited Members

### 5.71.1 Detailed Description

This class is the basis for any component representing the end of a process flow, such as a [Dispose](#). It can remove entities from the system and collect statistics.

### 5.71.2 Constructor & Destructor Documentation

5.71.2.1 `SinkModelComponent::SinkModelComponent ( Model * model, std::string componentTypename )`

5.71.2.2 `SinkModelComponent::SinkModelComponent ( const SinkModelComponent & orig )`

5.71.2.3 `SinkModelComponent::~~SinkModelComponent ( ) [virtual]`

### 5.71.3 Member Function Documentation

5.71.3.1 `bool SinkModelComponent::isCollectStatistics ( ) const`

5.71.3.2 `void SinkModelComponent::setCollectStatistics ( bool _collectStatistics )`

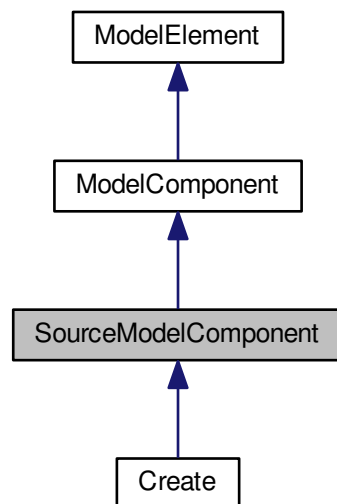
The documentation for this class was generated from the following files:

- [SinkModelComponent.h](#)
- [SinkModelComponent.cpp](#)

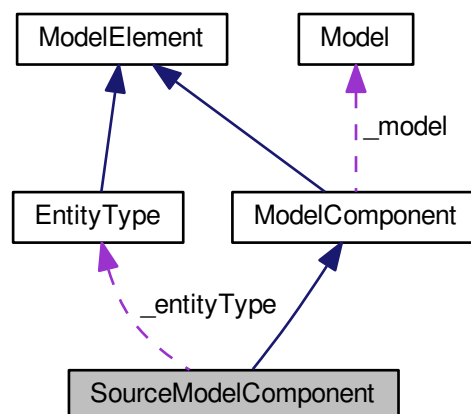
## 5.72 SourceModelComponent Class Reference

```
#include <SourceModelComponent.h>
```

Inheritance diagram for SourceModelComponent:



Collaboration diagram for SourceModelComponent:



## Public Member Functions

- [SourceModelComponent](#) ([Model](#) \*model, std::string componentTypename)
- [SourceModelComponent](#) (const [SourceModelComponent](#) &orig)
- virtual [~SourceModelComponent](#) ()
- void [setFirstCreation](#) (double [\\_firstCreation](#))
- double [getFirstCreation](#) () const
- void [setCollectStatistics](#) (bool [\\_collectStatistics](#))
- bool [isCollectStatistics](#) () const
- void [setEntityType](#) ([EntityType](#) \*[\\_entityType](#))
- [EntityType](#) \* [getEntityType](#) () const
- void [setTimeUnit](#) ([Util::TimeUnit](#) [\\_timeUnit](#))
- [Util::TimeUnit](#) [getTimeUnit](#) () const
- void [setTimeBetweenCreationsExpression](#) (std::string [\\_timeBetweenCreations](#))
- std::string [getTimeBetweenCreationsExpression](#) () const
- void [setMaxCreations](#) (unsigned int [\\_maxCreations](#))
- unsigned int [getMaxCreations](#) () const
- unsigned int [getEntitiesCreated](#) () const
- void [setEntitiesCreated](#) (unsigned int [\\_entitiesCreated](#))
- void [setEntitiesPerCreation](#) (unsigned int [\\_entitiesPerCreation](#))
- unsigned int [getEntitiesPerCreation](#) () const
- virtual std::string [show](#) ()

## Protected Attributes

- [EntityType](#) \* [\\_entityType](#)
- double [\\_firstCreation](#) = 0.0
- unsigned int [\\_entitiesPerCreation](#) = 1
- unsigned int [\\_maxCreations](#) = std::numeric\_limits<unsigned int>::max()
- std::string [\\_timeBetweenCreationsExpression](#) = "10"
- [Util::TimeUnit](#) [\\_timeBetweenCreationsTimeUnit](#) = [Util::TimeUnit::second](#)
- bool [\\_collectStatistics](#) = true
- unsigned int [\\_entitiesCreatedSoFar](#) = 0

## Additional Inherited Members

### 5.72.1 Detailed Description

A source component implements the base for inserting entities into the model when its simulation is initialized. During the initialization, the new and empty future events list is populated by events of creating entities and sending them to the source components existing in the model



## 5.72.2 Constructor & Destructor Documentation

5.72.2.1 `SourceModelComponent::SourceModelComponent ( Model * model, std::string componentTypename )`

5.72.2.2 `SourceModelComponent::SourceModelComponent ( const SourceModelComponent & orig )`

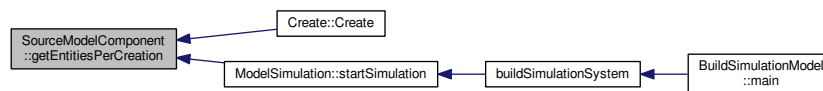
5.72.2.3 `SourceModelComponent::~~SourceModelComponent ( )` [virtual]

## 5.72.3 Member Function Documentation

5.72.3.1 `unsigned int SourceModelComponent::getEntitiesCreated ( )` const

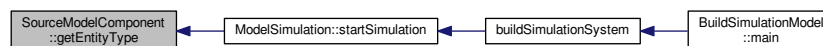
5.72.3.2 `unsigned int SourceModelComponent::getEntitiesPerCreation ( )` const

Here is the caller graph for this function:



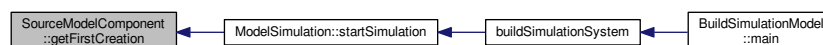
5.72.3.3 `EntityType * SourceModelComponent::getEntityType ( )` const

Here is the caller graph for this function:



5.72.3.4 `double SourceModelComponent::getFirstCreation ( )` const

Here is the caller graph for this function:



5.72.3.5 unsigned int SourceModelComponent::getMaxCreations ( ) const

5.72.3.6 std::string SourceModelComponent::getTimeBetweenCreationsExpression ( ) const

5.72.3.7 Util::TimeUnit SourceModelComponent::getTimeUnit ( ) const

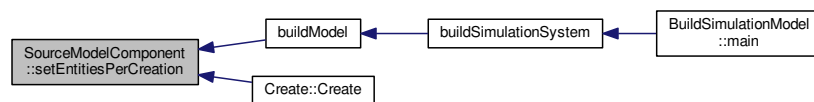
5.72.3.8 bool SourceModelComponent::isCollectStatistics ( ) const

5.72.3.9 void SourceModelComponent::setCollectStatistics ( bool *\_collectStatistics* )

5.72.3.10 void SourceModelComponent::setEntitiesCreated ( unsigned int *\_entitiesCreated* )

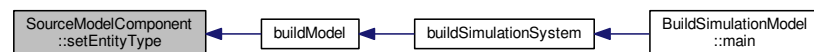
5.72.3.11 void SourceModelComponent::setEntitiesPerCreation ( unsigned int *\_entitiesPerCreation* )

Here is the caller graph for this function:



5.72.3.12 void SourceModelComponent::setEntityType ( EntityType \* *\_entityType* )

Here is the caller graph for this function:



5.72.3.13 void SourceModelComponent::setFirstCreation ( double *\_firstCreation* )

5.72.3.14 void SourceModelComponent::setMaxCreations ( unsigned int *\_maxCreations* )

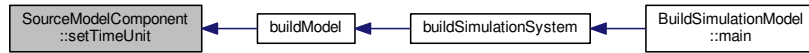
5.72.3.15 void SourceModelComponent::setTimeBetweenCreationsExpression ( std::string *\_timeBetweenCreations* )

Here is the caller graph for this function:



5.72.3.16 void SourceModelComponent::setTimeUnit ( Util::TimeUnit *timeUnit* )

Here is the caller graph for this function:

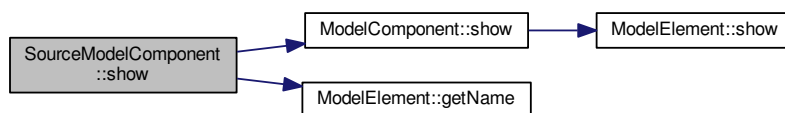


## 5.72.3.17 std::string SourceModelComponent::show ( ) [virtual]

Reimplemented from [ModelComponent](#).

Reimplemented in [Create](#).

Here is the call graph for this function:



Here is the caller graph for this function:



## 5.72.4 Member Data Documentation

## 5.72.4.1 bool SourceModelComponent::\_collectStatistics = true [protected]

## 5.72.4.2 unsigned int SourceModelComponent::\_entitiesCreatedSoFar = 0 [protected]

## 5.72.4.3 unsigned int SourceModelComponent::\_entitiesPerCreation = 1 [protected]

5.72.4.4 **EntityType\*** **SourceModelComponent::\_entityType** [protected]

5.72.4.5 **double** **SourceModelComponent::\_firstCreation = 0.0** [protected]

5.72.4.6 **unsigned int** **SourceModelComponent::\_maxCreations = std::numeric\_limits<unsigned int>::max()** [protected]

5.72.4.7 **std::string** **SourceModelComponent::\_timeBetweenCreationsExpression = "10"** [protected]

5.72.4.8 **Util::TimeUnit** **SourceModelComponent::\_timeBetweenCreationsTimeUnit = Util::TimeUnit::second** [protected]

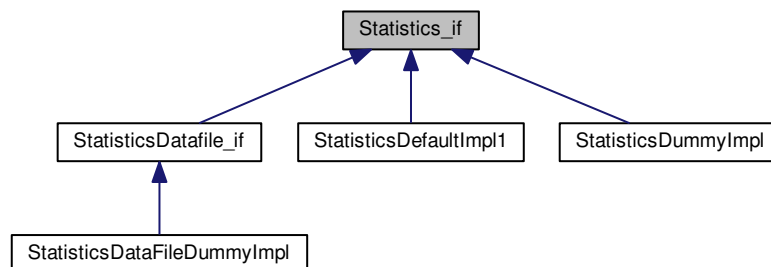
The documentation for this class was generated from the following files:

- [SourceModelComponent.h](#)
- [SourceModelComponent.cpp](#)

## 5.73 Statistics\_if Class Reference

```
#include <Statistics_if.h>
```

Inheritance diagram for Statistics\_if:



### Public Member Functions

- virtual [Collector\\_if](#) \* [getCollector](#) ()=0
- virtual void [setCollector](#) ([Collector\\_if](#) \*collector)=0
- virtual unsigned int [numElements](#) ()=0
- virtual double [min](#) ()=0
- virtual double [max](#) ()=0
- virtual double [average](#) ()=0
- virtual double [variance](#) ()=0
- virtual double [stddeviation](#) ()=0
- virtual double [variationCoef](#) ()=0
- virtual double [halfWidthConfidenceInterval](#) (double confidencelevel)=0
- virtual unsigned int [newSampleSize](#) (double confidencelevel, double halfWidth)=0

### 5.73.1 Detailed Description

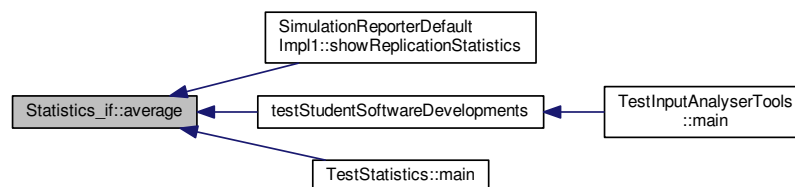
Interface for statistct synthesis of a stochastic variable collected by a [Collector\\_if](#). The statistics generated may be updated based only on the previous statistics and the single newest added value or they may be updated based on a datafile, depending on the Collector implementation.

### 5.73.2 Member Function Documentation

#### 5.73.2.1 `virtual double Statistics_if::average ( ) [pure virtual]`

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

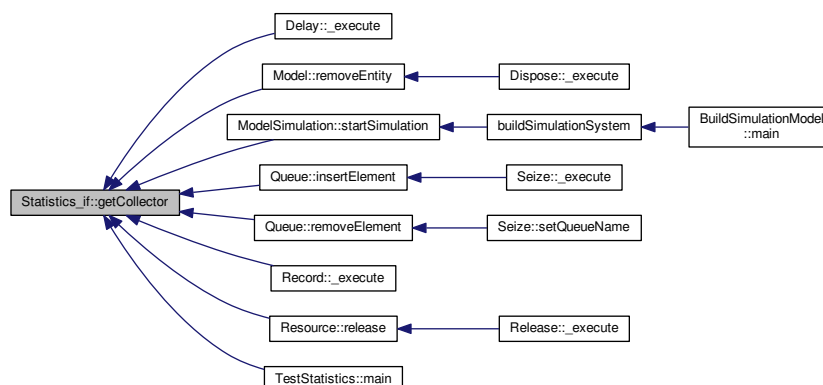
Here is the caller graph for this function:



#### 5.73.2.2 `virtual Collector_if* Statistics_if::getCollector ( ) [pure virtual]`

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

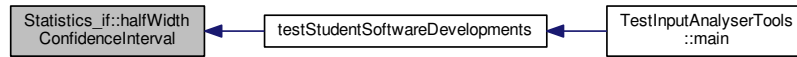
Here is the caller graph for this function:



### 5.73.2.3 virtual double Statistics\_if::halfWidthConfidenceInterval ( double *confidencelevel* ) [pure virtual]

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

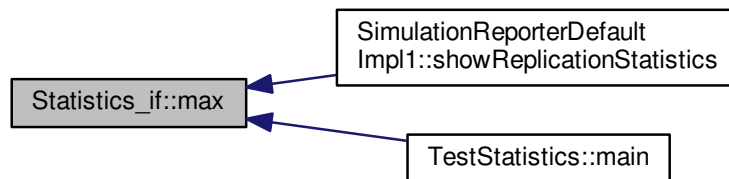
Here is the caller graph for this function:



### 5.73.2.4 virtual double Statistics\_if::max ( ) [pure virtual]

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

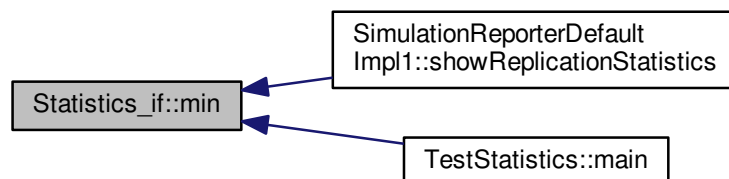
Here is the caller graph for this function:



### 5.73.2.5 virtual double Statistics\_if::min ( ) [pure virtual]

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

Here is the caller graph for this function:



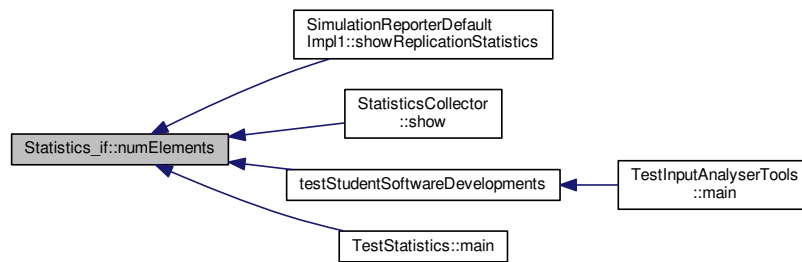
5.73.2.6 `virtual unsigned int Statistics_if::newSampleSize ( double confidencelevel, double halfWidth )` [pure virtual]

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

5.73.2.7 `virtual unsigned int Statistics_if::numElements ( )` [pure virtual]

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

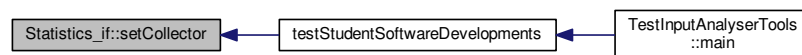
Here is the caller graph for this function:



5.73.2.8 `virtual void Statistics_if::setCollector ( Collector_if * collector )` [pure virtual]

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

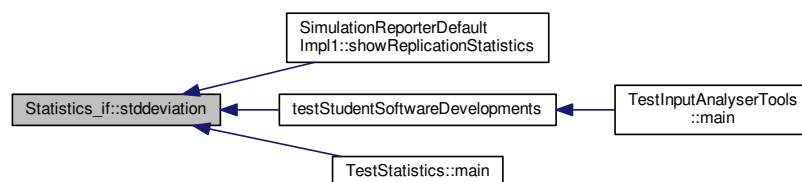
Here is the caller graph for this function:



5.73.2.9 `virtual double Statistics_if::stddeviation ( )` [pure virtual]

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

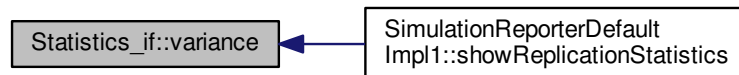
Here is the caller graph for this function:



5.73.2.10 `virtual double Statistics_if::variance ( ) [pure virtual]`

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

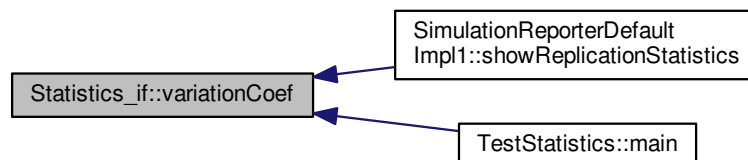
Here is the caller graph for this function:



5.73.2.11 `virtual double Statistics_if::variationCoef ( ) [pure virtual]`

Implemented in [StatisticsDefaultImpl1](#), [StatisticsDummyImpl](#), and [StatisticsDataFileDummyImpl](#).

Here is the caller graph for this function:



The documentation for this class was generated from the following file:

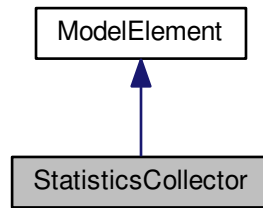
- [Statistics\\_if.h](#)

## 5.74 StatisticsCollector Class Reference

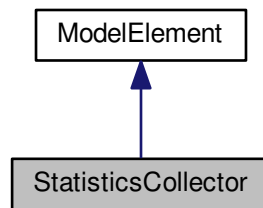
```
#include <StatisticsCollector.h>
```



Inheritance diagram for StatisticsCollector:



Collaboration diagram for StatisticsCollector:



## Public Member Functions

- [StatisticsCollector](#) ()
- [StatisticsCollector](#) (std::string name)
- [StatisticsCollector](#) (std::string name, [ModelElement](#) \*parent)
- [StatisticsCollector](#) (const [StatisticsCollector](#) &orig)
- virtual [~StatisticsCollector](#) ()
- virtual std::string [show](#) ()
- [ModelElement](#) \* [getParent](#) () const
- [Statistics\\_if](#) \* [getStatistics](#) () const

## Protected Member Functions

- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.74.1 Constructor & Destructor Documentation

5.74.1.1 `StatisticsCollector::StatisticsCollector ( )`

5.74.1.2 `StatisticsCollector::StatisticsCollector ( std::string name )`

5.74.1.3 `StatisticsCollector::StatisticsCollector ( std::string name, ModelElement * parent )`

5.74.1.4 `StatisticsCollector::StatisticsCollector ( const StatisticsCollector & orig )`

5.74.1.5 `StatisticsCollector::~~StatisticsCollector ( )` `[virtual]`

Here is the caller graph for this function:



### 5.74.2 Member Function Documentation

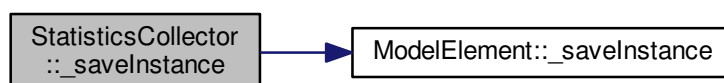
5.74.2.1 `void StatisticsCollector::_loadInstance ( std::list< std::string > words )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

5.74.2.2 `std::list< std::string > * StatisticsCollector::_saveInstance ( )` `[protected]`, `[virtual]`

Reimplemented from [ModelElement](#).

Here is the call graph for this function:

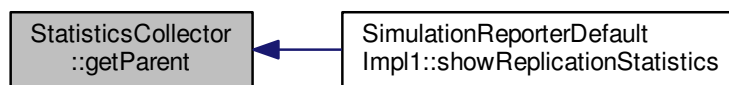


5.74.2.3 `bool StatisticsCollector::_verifySymbols ( std::string * errorMessage )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

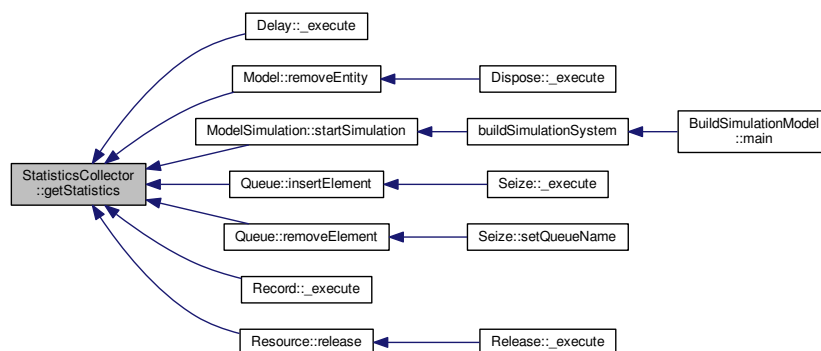
5.74.2.4 `ModelElement * StatisticsCollector::getParent ( ) const`

Here is the caller graph for this function:



5.74.2.5 `Statistics_if * StatisticsCollector::getStatistics ( ) const`

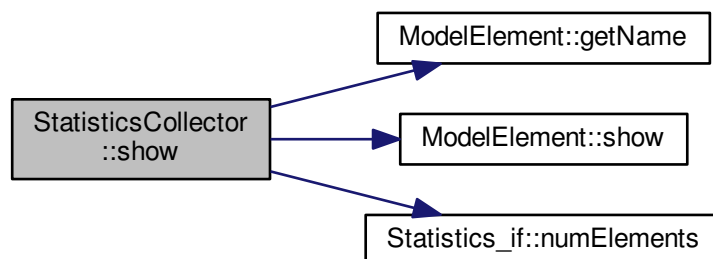
Here is the caller graph for this function:



5.74.2.6 `std::string StatisticsCollector::show ( )` `[virtual]`

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



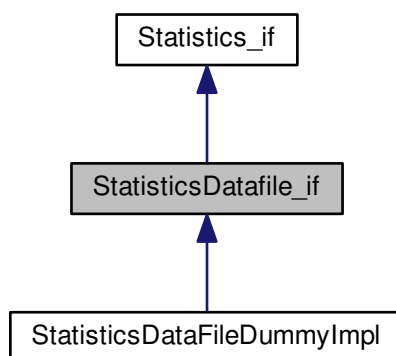
The documentation for this class was generated from the following files:

- [StatisticsCollector.h](#)
- [StatisticsCollector.cpp](#)

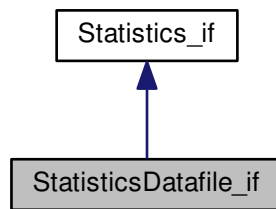
## 5.75 StatisticsDatafile\_if Class Reference

```
#include <StatisticsDataFile_if.h>
```

Inheritance diagram for StatisticsDatafile\_if:



Collaboration diagram for StatisticsDatafile\_if:



## Public Member Functions

- virtual double [mode](#) ()=0
- virtual double [mediane](#) ()=0
- virtual double [quartil](#) (unsigned short num)=0
- virtual double [decil](#) (unsigned short num)=0
- virtual double [centil](#) (unsigned short num)=0
- virtual void [setHistogramNumClasses](#) (unsigned short num)=0
- virtual unsigned short [histogramNumClasses](#) ()=0
- virtual double [histogramClassLowerLimit](#) (unsigned short classNum)=0
- virtual unsigned int [histogramClassFrequency](#) (unsigned short classNum)=0

### 5.75.1 Member Function Documentation

**5.75.1.1** virtual double `StatisticsDatafile_if::centil ( unsigned short num )` [pure virtual]

Implemented in [StatisticsDataFileDummyImpl](#).

**5.75.1.2** virtual double `StatisticsDatafile_if::decil ( unsigned short num )` [pure virtual]

Implemented in [StatisticsDataFileDummyImpl](#).

**5.75.1.3** virtual unsigned int `StatisticsDatafile_if::histogramClassFrequency ( unsigned short classNum )` [pure virtual]

Implemented in [StatisticsDataFileDummyImpl](#).

**5.75.1.4** virtual double `StatisticsDatafile_if::histogramClassLowerLimit ( unsigned short classNum )` [pure virtual]

Implemented in [StatisticsDataFileDummyImpl](#).

5.75.1.5 `virtual unsigned short StatisticsDatafile_if::histogramNumClasses ( ) [pure virtual]`

Implemented in [StatisticsDataFileDummyImpl](#).

5.75.1.6 `virtual double StatisticsDatafile_if::mediane ( ) [pure virtual]`

Implemented in [StatisticsDataFileDummyImpl](#).

5.75.1.7 `virtual double StatisticsDatafile_if::mode ( ) [pure virtual]`

Implemented in [StatisticsDataFileDummyImpl](#).

5.75.1.8 `virtual double StatisticsDatafile_if::quartil ( unsigned short num ) [pure virtual]`

Implemented in [StatisticsDataFileDummyImpl](#).

5.75.1.9 `virtual void StatisticsDatafile_if::setHistogramNumClasses ( unsigned short num ) [pure virtual]`

Implemented in [StatisticsDataFileDummyImpl](#).

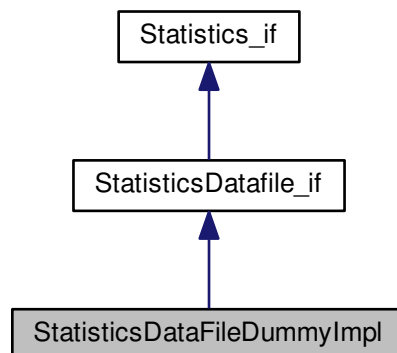
The documentation for this class was generated from the following file:

- [StatisticsDataFile\\_if.h](#)

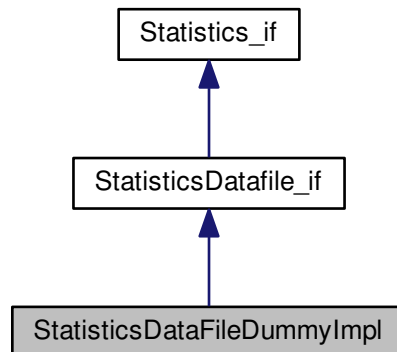
## 5.76 StatisticsDataFileDummyImpl Class Reference

```
#include <StatisticsDataFileDummyImpl.h>
```

Inheritance diagram for StatisticsDataFileDummyImpl:



Collaboration diagram for StatisticsDataFileDummyImpl:



## Public Member Functions

- [StatisticsDataFileDummyImpl](#) ()
- [StatisticsDataFileDummyImpl](#) (const [StatisticsDataFileDummyImpl](#) &orig)
- virtual [~StatisticsDataFileDummyImpl](#) ()
- virtual [Collector\\_if](#) \* [getCollector](#) ()
- void [setCollector](#) ([Collector\\_if](#) \*collector)
- virtual unsigned int [numElements](#) ()
- virtual double [min](#) ()
- virtual double [max](#) ()
- virtual double [average](#) ()
- virtual double [variance](#) ()
- virtual double [stddeviation](#) ()
- virtual double [variationCoef](#) ()
- virtual double [halfWidthConfidenceInterval](#) (double confidencelevel)
- virtual unsigned int [newSampleSize](#) (double confidencelevel, double halfWidth)
- virtual double [mode](#) ()
- virtual double [mediane](#) ()
- virtual double [quartil](#) (unsigned short num)
- virtual double [decil](#) (unsigned short num)
- virtual double [centil](#) (unsigned short num)
- virtual void [setHistogramNumClasses](#) (unsigned short num)
- virtual unsigned short [histogramNumClasses](#) ()
- virtual double [histogramClassLowerLimit](#) (unsigned short classNum)
- virtual unsigned int [histogramClassFrequency](#) (unsigned short classNum)

## 5.76.1 Constructor & Destructor Documentation

5.76.1.1 [StatisticsDataFileDummyImpl::StatisticsDataFileDummyImpl](#) ( )

5.76.1.2 [StatisticsDataFileDummyImpl::StatisticsDataFileDummyImpl](#) ( const [StatisticsDataFileDummyImpl](#) & orig )

5.76.1.3 `StatisticsDataFileDummyImpl::~StatisticsDataFileDummyImpl ( )` [virtual]

## 5.76.2 Member Function Documentation

5.76.2.1 `double StatisticsDataFileDummyImpl::average ( )` [virtual]

Implements [Statistics\\_if](#).

5.76.2.2 `double StatisticsDataFileDummyImpl::centil ( unsigned short num )` [virtual]

Implements [StatisticsDatafile\\_if](#).

5.76.2.3 `double StatisticsDataFileDummyImpl::decil ( unsigned short num )` [virtual]

Implements [StatisticsDatafile\\_if](#).

5.76.2.4 `Collector_if * StatisticsDataFileDummyImpl::getCollector ( )` [virtual]

Implements [Statistics\\_if](#).

5.76.2.5 `double StatisticsDataFileDummyImpl::halfWidthConfidenceInterval ( double confidencelevel )` [virtual]

Implements [Statistics\\_if](#).

5.76.2.6 `unsigned int StatisticsDataFileDummyImpl::histogramClassFrequency ( unsigned short classNum )` [virtual]

Implements [StatisticsDatafile\\_if](#).

5.76.2.7 `double StatisticsDataFileDummyImpl::histogramClassLowerLimit ( unsigned short classNum )` [virtual]

Implements [StatisticsDatafile\\_if](#).

5.76.2.8 `unsigned short StatisticsDataFileDummyImpl::histogramNumClasses ( )` [virtual]

Implements [StatisticsDatafile\\_if](#).

5.76.2.9 `double StatisticsDataFileDummyImpl::max ( )` [virtual]

Implements [Statistics\\_if](#).



5.76.2.10 `double StatisticsDataFileDummyImpl::median ( ) [virtual]`

Implements [StatisticsDatafile\\_if](#).

5.76.2.11 `double StatisticsDataFileDummyImpl::min ( ) [virtual]`

Implements [Statistics\\_if](#).

5.76.2.12 `double StatisticsDataFileDummyImpl::mode ( ) [virtual]`

Implements [StatisticsDatafile\\_if](#).

5.76.2.13 `unsigned int StatisticsDataFileDummyImpl::newSampleSize ( double confidencelevel, double halfWidth ) [virtual]`

Implements [Statistics\\_if](#).

5.76.2.14 `unsigned int StatisticsDataFileDummyImpl::numElements ( ) [virtual]`

Implements [Statistics\\_if](#).

5.76.2.15 `double StatisticsDataFileDummyImpl::quartil ( unsigned short num ) [virtual]`

Implements [StatisticsDatafile\\_if](#).

5.76.2.16 `void StatisticsDataFileDummyImpl::setCollector ( Collector_if * collector ) [virtual]`

Implements [Statistics\\_if](#).

5.76.2.17 `void StatisticsDataFileDummyImpl::setHistogramNumClasses ( unsigned short num ) [virtual]`

Implements [StatisticsDatafile\\_if](#).

5.76.2.18 `double StatisticsDataFileDummyImpl::stddeviation ( ) [virtual]`

Implements [Statistics\\_if](#).

5.76.2.19 `double StatisticsDataFileDummyImpl::variance ( ) [virtual]`

Implements [Statistics\\_if](#).

5.76.2.20 `double StatisticsDataFileDummyImpl::variationCoef ( ) [virtual]`

Implements [Statistics\\_if](#).

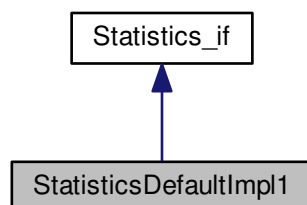
The documentation for this class was generated from the following files:

- [StatisticsDataFileDummyImpl.h](#)
- [StatisticsDataFileDummyImpl.cpp](#)

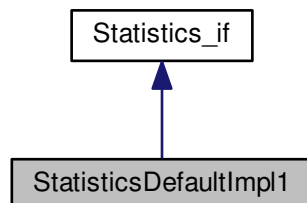
## 5.77 StatisticsDefaultImpl1 Class Reference

```
#include <StatisticsDefaultImpl1.h>
```

Inheritance diagram for StatisticsDefaultImpl1:



Collaboration diagram for StatisticsDefaultImpl1:



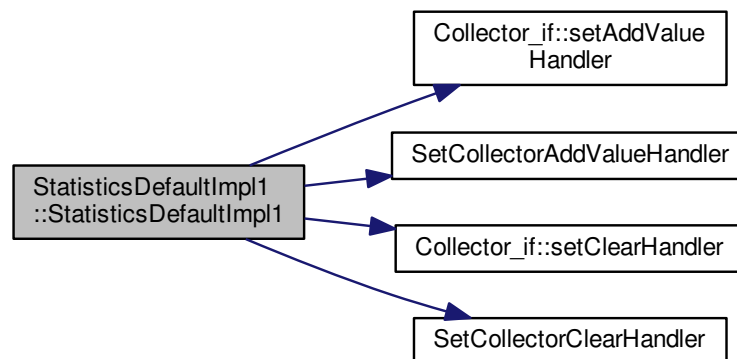
## Public Member Functions

- [StatisticsDefaultImpl1](#) ()
- [StatisticsDefaultImpl1](#) ([Collector\\_if](#) \*collector)
- [StatisticsDefaultImpl1](#) (const [StatisticsDefaultImpl1](#) &orig)
- virtual [~StatisticsDefaultImpl1](#) ()
- virtual [Collector\\_if](#) \* [getCollector](#) ()
- void [setCollector](#) ([Collector\\_if](#) \*collector)
- virtual unsigned int [numElements](#) ()
- virtual double [min](#) ()
- virtual double [max](#) ()
- virtual double [average](#) ()
- virtual double [variance](#) ()
- virtual double [stddeviation](#) ()
- virtual double [variationCoef](#) ()
- virtual double [halfWidthConfidenceInterval](#) (double confidencelevel)
- virtual unsigned int [newSampleSize](#) (double confidencelevel, double halfWidth)

## 5.77.1 Constructor &amp; Destructor Documentation

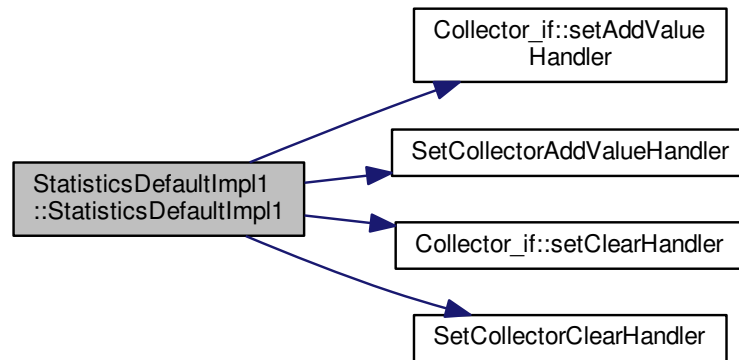
## 5.77.1.1 StatisticsDefaultImpl1::StatisticsDefaultImpl1 ( )

Here is the call graph for this function:



### 5.77.1.2 `StatisticsDefaultImpl1::StatisticsDefaultImpl1 ( Collector_if * collector )`

Here is the call graph for this function:



### 5.77.1.3 `StatisticsDefaultImpl1::StatisticsDefaultImpl1 ( const StatisticsDefaultImpl1 & orig )`

### 5.77.1.4 `StatisticsDefaultImpl1::~StatisticsDefaultImpl1 ( ) [virtual]`

Here is the call graph for this function:



## 5.77.2 Member Function Documentation

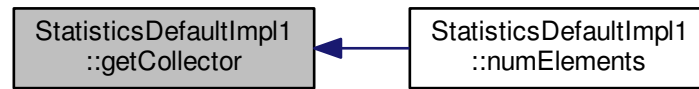
### 5.77.2.1 `double StatisticsDefaultImpl1::average ( ) [virtual]`

Implements [Statistics\\_if](#).

### 5.77.2.2 `Collector_if * StatisticsDefaultImpl1::getCollector ( )` [virtual]

Implements [Statistics\\_if](#).

Here is the caller graph for this function:



### 5.77.2.3 `double StatisticsDefaultImpl1::halfWidthConfidenceInterval ( double confidencelevel )` [virtual]

Implements [Statistics\\_if](#).

### 5.77.2.4 `double StatisticsDefaultImpl1::max ( )` [virtual]

Implements [Statistics\\_if](#).

### 5.77.2.5 `double StatisticsDefaultImpl1::min ( )` [virtual]

Implements [Statistics\\_if](#).

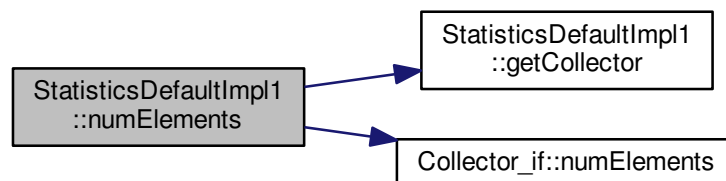
### 5.77.2.6 `unsigned int StatisticsDefaultImpl1::newSampleSize ( double confidencelevel, double halfWidth )` [virtual]

Implements [Statistics\\_if](#).

### 5.77.2.7 `unsigned int StatisticsDefaultImpl1::numElements ( )` [virtual]

Implements [Statistics\\_if](#).

Here is the call graph for this function:



**5.77.2.8** void StatisticsDefaultImpl1::setCollector ( Collector\_if \* collector ) [virtual]

Implements [Statistics\\_if](#).

**5.77.2.9** double StatisticsDefaultImpl1::stddeviation ( ) [virtual]

Implements [Statistics\\_if](#).

**5.77.2.10** double StatisticsDefaultImpl1::variance ( ) [virtual]

Implements [Statistics\\_if](#).

**5.77.2.11** double StatisticsDefaultImpl1::variationCoef ( ) [virtual]

Implements [Statistics\\_if](#).

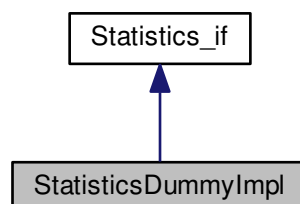
The documentation for this class was generated from the following files:

- [StatisticsDefaultImpl1.h](#)
- [StatisticsDefaultImpl1.cpp](#)

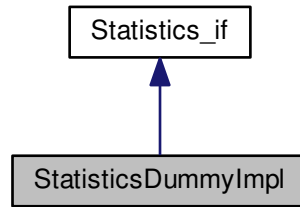
## 5.78 StatisticsDummyImpl Class Reference

```
#include <StatisticsDummyImpl.h>
```

Inheritance diagram for StatisticsDummyImpl:



Collaboration diagram for StatisticsDummyImpl:



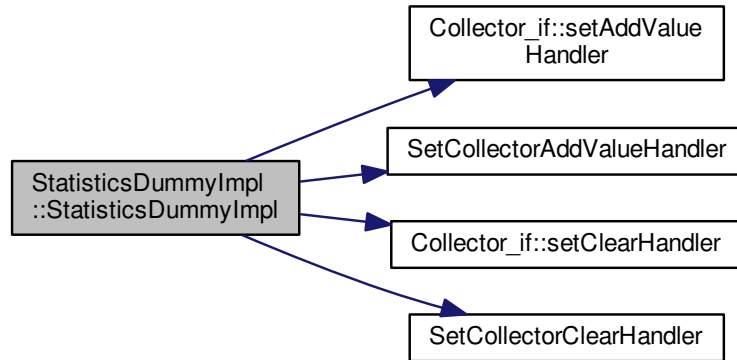
## Public Member Functions

- [StatisticsDummyImpl](#) ()
- [StatisticsDummyImpl](#) (const [StatisticsDummyImpl](#) &orig)
- virtual [~StatisticsDummyImpl](#) ()
- virtual [Collector\\_if](#) \* [getCollector](#) ()
- void [setCollector](#) ([Collector\\_if](#) \*collector)
- virtual unsigned int [numElements](#) ()
- virtual double [min](#) ()
- virtual double [max](#) ()
- virtual double [average](#) ()
- virtual double [variance](#) ()
- virtual double [stddeviation](#) ()
- virtual double [variationCoef](#) ()
- virtual double [halfWidthConfidenceInterval](#) (double confidencelevel)
- virtual unsigned int [newSampleSize](#) (double confidencelevel, double halfWidth)

### 5.78.1 Constructor & Destructor Documentation

#### 5.78.1.1 `StatisticsDummyImpl::StatisticsDummyImpl ( )`

Here is the call graph for this function:



#### 5.78.1.2 `StatisticsDummyImpl::StatisticsDummyImpl ( const StatisticsDummyImpl & orig )`

#### 5.78.1.3 `StatisticsDummyImpl::~StatisticsDummyImpl ( )` [virtual]

### 5.78.2 Member Function Documentation

#### 5.78.2.1 `double StatisticsDummyImpl::average ( )` [virtual]

Implements [Statistics\\_if](#).

#### 5.78.2.2 `Collector_if* StatisticsDummyImpl::getCollector ( )` [virtual]

Implements [Statistics\\_if](#).

#### 5.78.2.3 `double StatisticsDummyImpl::halfWidthConfidenceInterval ( double confidencelevel )` [virtual]

Implements [Statistics\\_if](#).

#### 5.78.2.4 `double StatisticsDummyImpl::max ( )` [virtual]

Implements [Statistics\\_if](#).



5.78.2.5 `double StatisticsDummyImpl::min ( )` [virtual]

Implements [Statistics\\_if](#).

5.78.2.6 `unsigned int StatisticsDummyImpl::newSampleSize ( double confidencelevel, double halfWidth )` [virtual]

Implements [Statistics\\_if](#).

5.78.2.7 `unsigned int StatisticsDummyImpl::numElements ( )` [virtual]

Implements [Statistics\\_if](#).

5.78.2.8 `void StatisticsDummyImpl::setCollector ( Collector_if * collector )` [virtual]

Implements [Statistics\\_if](#).

5.78.2.9 `double StatisticsDummyImpl::stddeviation ( )` [virtual]

Implements [Statistics\\_if](#).

5.78.2.10 `double StatisticsDummyImpl::variance ( )` [virtual]

Implements [Statistics\\_if](#).

5.78.2.11 `double StatisticsDummyImpl::variationCoef ( )` [virtual]

Implements [Statistics\\_if](#).

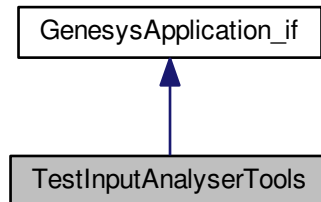
The documentation for this class was generated from the following files:

- [StatisticsDummyImpl.h](#)
- [StatisticsDummyImpl.cpp](#)

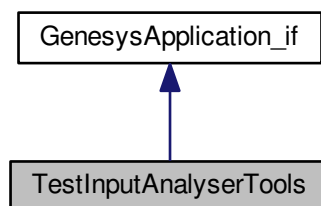
## 5.79 TestInputAnalyserTools Class Reference

```
#include <TestInputAnalyserTools.h>
```

Inheritance diagram for TestInputAnalyserTools:



Collaboration diagram for TestInputAnalyserTools:



### Public Member Functions

- [TestInputAnalyserTools](#) ()
- [main](#) (int argc, char \*\*argv)

### 5.79.1 Constructor & Destructor Documentation

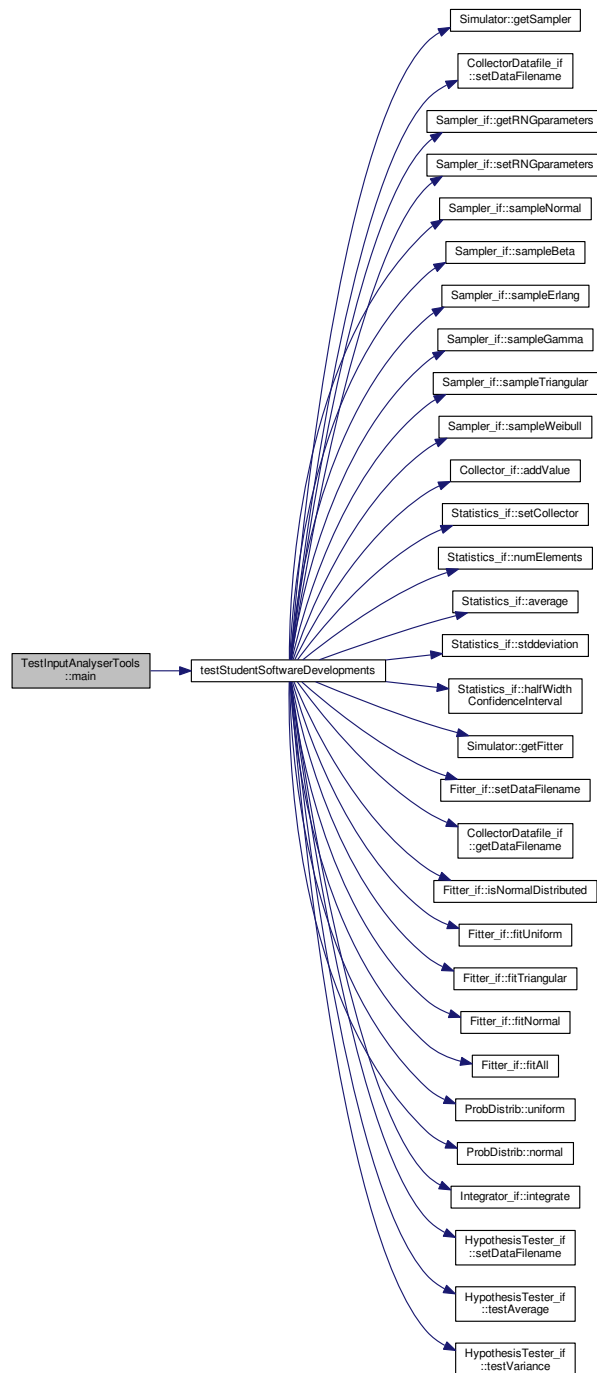
5.79.1.1 [TestInputAnalyserTools::TestInputAnalyserTools](#) ( )

### 5.79.2 Member Function Documentation

5.79.2.1 [int TestInputAnalyserTools::main](#) ( int *argc*, char \*\* *argv* ) [virtual]

Implements [GenesysApplication\\_if](#).

Here is the call graph for this function:



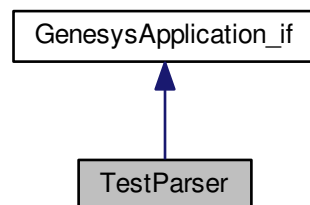
The documentation for this class was generated from the following files:

- [TestInputAnalyserTools.h](#)
- [TestInputAnalyserTools.cpp](#)

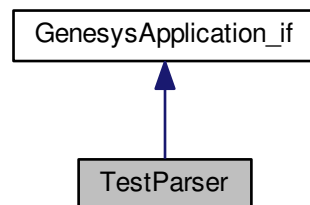
## 5.80 TestParser Class Reference

```
#include <TestParser.h>
```

Inheritance diagram for TestParser:



Collaboration diagram for TestParser:



### Public Member Functions

- [TestParser](#) ()
- [TestParser](#) (const [TestParser](#) &orig)
- virtual [~TestParser](#) ()
- virtual int [main](#) (int argc, char \*\*argv)

### 5.80.1 Constructor & Destructor Documentation

5.80.1.1 [TestParser::TestParser](#) ( )

5.80.1.2 [TestParser::TestParser](#) ( const [TestParser](#) & orig )

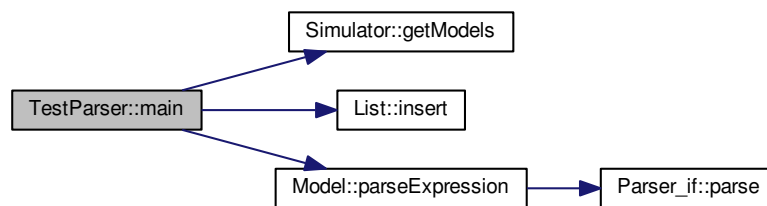
5.80.1.3 `TestParser::~TestParser ( )` `[virtual]`

## 5.80.2 Member Function Documentation

5.80.2.1 `int TestParser::main ( int argc, char ** argv )` `[virtual]`

Implements [GenesysApplication\\_if](#).

Here is the call graph for this function:



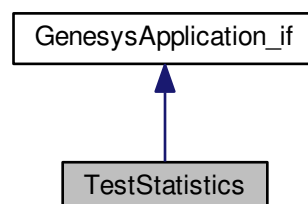
The documentation for this class was generated from the following files:

- [TestParser.h](#)
- [TestParser.cpp](#)

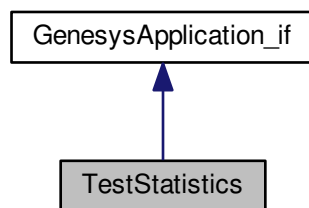
## 5.81 TestStatistics Class Reference

```
#include <TestStatistics.h>
```

Inheritance diagram for `TestStatistics`:



Collaboration diagram for TestStatistics:



## Public Member Functions

- [TestStatistics](#) ()
- int [main](#) (int argc, char \*\*argv)

### 5.81.1 Constructor & Destructor Documentation

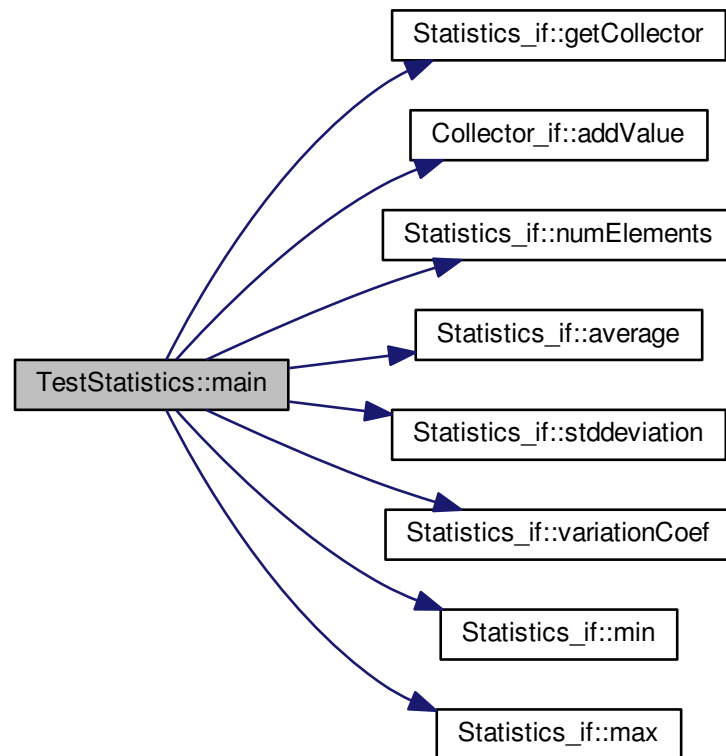
#### 5.81.1.1 TestStatistics::TestStatistics ( )

### 5.81.2 Member Function Documentation

#### 5.81.2.1 int TestStatistics::main ( int *argc*, char \*\* *argv* ) [virtual]

Implements [GenesysApplication\\_if](#).

Here is the call graph for this function:



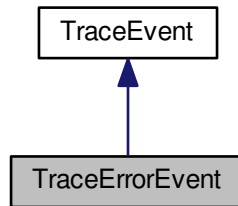
The documentation for this class was generated from the following files:

- [TestStatistics.h](#)
- [TestStatistics.cpp](#)

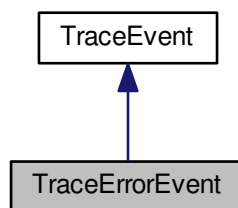
## 5.82 TraceErrorEvent Class Reference

```
#include <TraceManager.h>
```

Inheritance diagram for TraceErrorEvent:



Collaboration diagram for TraceErrorEvent:



## Public Member Functions

- [TraceErrorEvent](#) (std::string text, std::exception e)
- std::exception [getException](#) () const

### 5.82.1 Constructor & Destructor Documentation

5.82.1.1 `TraceErrorEvent::TraceErrorEvent ( std::string text, std::exception e ) [inline]`

### 5.82.2 Member Function Documentation

5.82.2.1 `std::exception TraceErrorEvent::getException ( ) const [inline]`

The documentation for this class was generated from the following file:

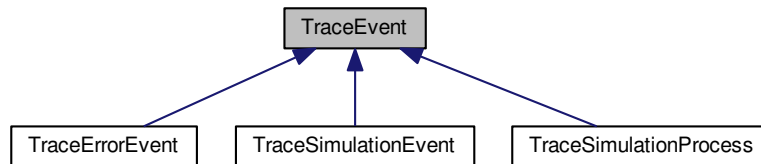
- [TraceManager.h](#)



## 5.83 TraceEvent Class Reference

```
#include <TraceManager.h>
```

Inheritance diagram for TraceEvent:



### Public Member Functions

- [TraceEvent](#) ([Util::TraceLevel](#) tracelevel, [std::string](#) text)
- [Util::TraceLevel](#) [getTracelevel](#) () const
- [std::string](#) [getText](#) () const

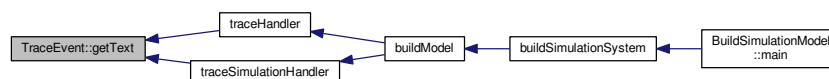
### 5.83.1 Constructor & Destructor Documentation

5.83.1.1 [TraceEvent::TraceEvent \( Util::TraceLevel tracelevel, std::string text \)](#) [\[inline\]](#)

### 5.83.2 Member Function Documentation

5.83.2.1 [std::string](#) [TraceEvent::getText \( \)](#) const [\[inline\]](#)

Here is the caller graph for this function:



5.83.2.2 [Util::TraceLevel](#) [TraceEvent::getTracelevel \( \)](#) const [\[inline\]](#)

The documentation for this class was generated from the following file:

- [TraceManager.h](#)

## 5.84 TraceManager Class Reference

```
#include <TraceManager.h>
```

### Public Member Functions

- [TraceManager](#) ([Model](#) \*model)
- [TraceManager](#) (const [TraceManager](#) &orig)
- virtual [~TraceManager](#) ()
- void [addTraceHandler](#) ([traceListener](#) traceListener)
- void [addTraceErrorHandler](#) ([traceErrorListener](#) traceErrorListener)
- void [addTraceReportHandler](#) ([traceListener](#) traceReportListener)
- void [addTraceSimulationHandler](#) ([traceSimulationListener](#) traceSimulationListener)
- void [trace](#) ([Util::TraceLevel](#) tracelevel, std::string text)
- void [traceError](#) (std::exception e, std::string text)
- void [traceSimulation](#) ([Util::TraceLevel](#) tracelevel, double time, [Entity](#) \*entity, [ModelComponent](#) \*component, std::string text)
- void [traceReport](#) ([Util::TraceLevel](#) tracelevel, std::string text)
- [List](#)< std::string > \* [getErrorMessages](#) () const
- void [setTraceLevel](#) ([Util::TraceLevel](#) \_traceLevel)
- [Util::TraceLevel](#) [getTraceLevel](#) () const

### 5.84.1 Detailed Description

The [TraceManager](#) is used to trace back model simulation information and track/debug the simulation. It works as the model simulation output (cout) and allows external methods to hook up such output as listeners.

### 5.84.2 Constructor & Destructor Documentation

5.84.2.1 [TraceManager::TraceManager](#) ( [Model](#) \* *model* )

5.84.2.2 [TraceManager::TraceManager](#) ( const [TraceManager](#) & *orig* )

5.84.2.3 [TraceManager::~~TraceManager](#) ( ) [virtual]

### 5.84.3 Member Function Documentation

5.84.3.1 void [TraceManager::addTraceErrorHandler](#) ( [traceErrorListener](#) *traceErrorListener* )

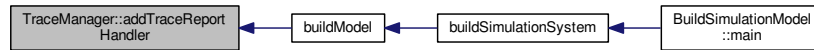
5.84.3.2 void [TraceManager::addTraceHandler](#) ( [traceListener](#) *traceListener* )

Here is the caller graph for this function:



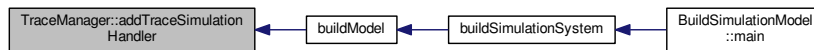
#### 5.84.3.3 void TraceManager::addTraceReportHandler ( *traceListener traceReportListener* )

Here is the caller graph for this function:



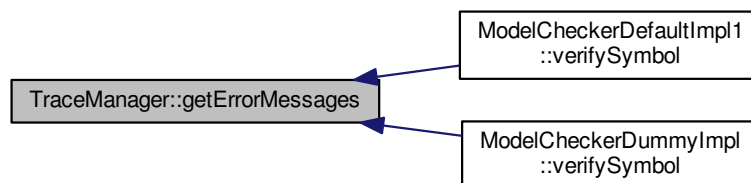
#### 5.84.3.4 void TraceManager::addTraceSimulationHandler ( *traceSimulationListener traceSimulationListener* )

Here is the caller graph for this function:



#### 5.84.3.5 List< std::string > \* TraceManager::getErrorMessage ( ) const

Here is the caller graph for this function:

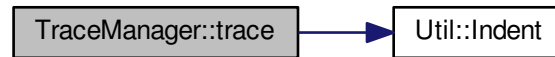


#### 5.84.3.6 Util::TraceLevel TraceManager::getTraceLevel ( ) const

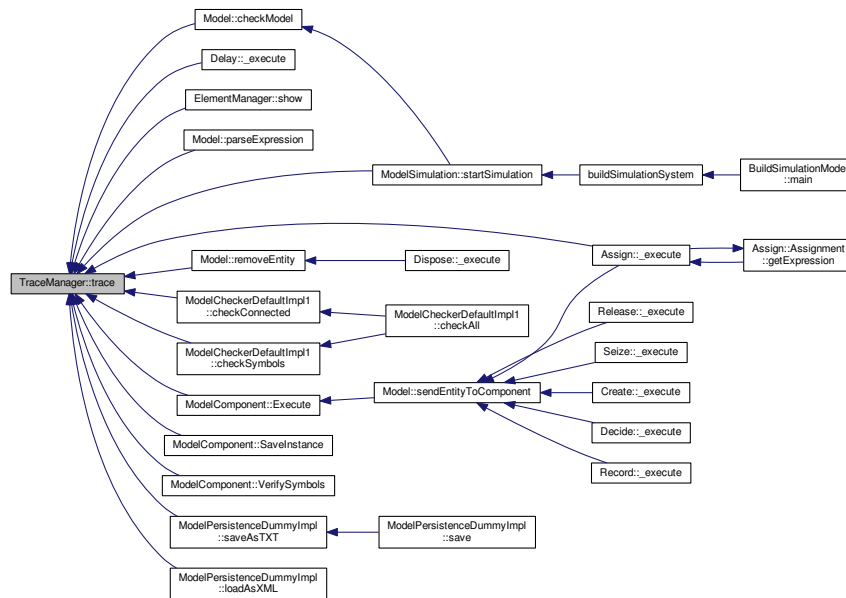
#### 5.84.3.7 void TraceManager::setTraceLevel ( Util::TraceLevel *\_traceLevel* )

#### 5.84.3.8 void TraceManager::trace ( Util::TraceLevel *tracelevel*, std::string *text* )

Here is the call graph for this function:

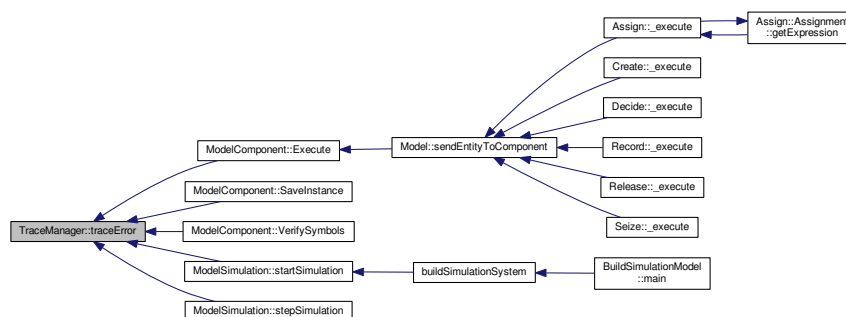


Here is the caller graph for this function:



#### 5.84.3.9 void TraceManager::traceError ( std::exception *e*, std::string *text* )

Here is the caller graph for this function:

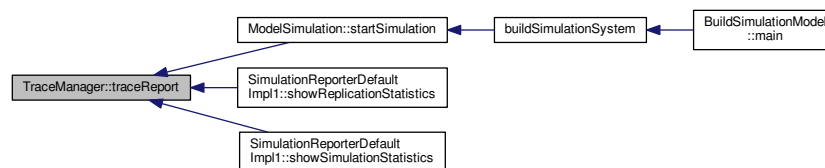


5.84.3.10 void TraceManager::traceReport ( Util::TraceLevel *tracelevel*, std::string *text* )

Here is the call graph for this function:

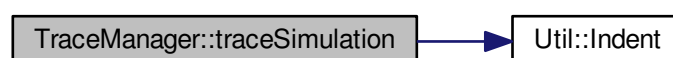


Here is the caller graph for this function:

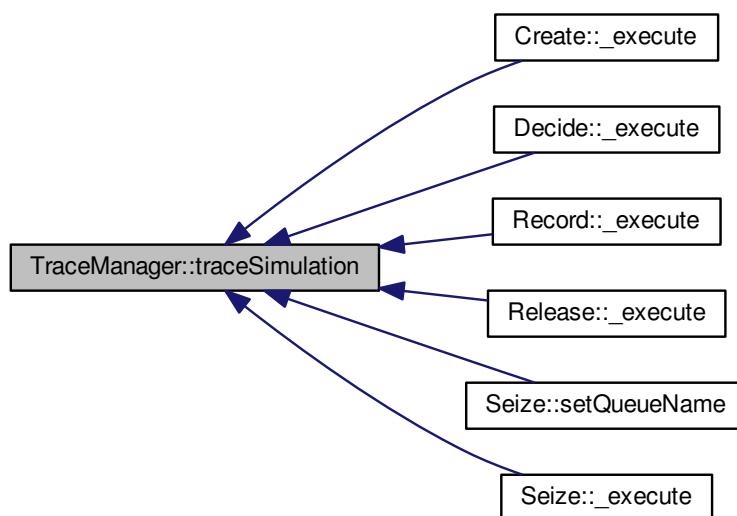


5.84.3.11 void TraceManager::traceSimulation ( Util::TraceLevel *tracelevel*, double *time*, Entity \* *entity*, ModelComponent \* *component*, std::string *text* )

Here is the call graph for this function:



Here is the caller graph for this function:



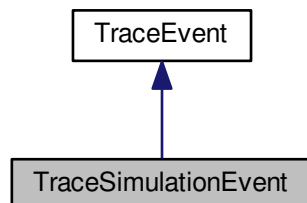
The documentation for this class was generated from the following files:

- [TraceManager.h](#)
- [TraceManager.cpp](#)

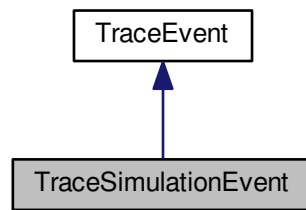
## 5.85 TraceSimulationEvent Class Reference

```
#include <TraceManager.h>
```

Inheritance diagram for `TraceSimulationEvent`:



Collaboration diagram for TraceSimulationEvent:



## Public Member Functions

- [ModelComponent](#) \* [getComponent](#) () const
- [Entity](#) \* [getEntity](#) () const
- double [getTime](#) () const
- [TraceSimulationEvent](#) ([Util::TraceLevel](#) tracelevel, double time, [Entity](#) \*entity, [ModelComponent](#) \*component, std::string text)

## 5.85.1 Constructor & Destructor Documentation

5.85.1.1 `TraceSimulationEvent::TraceSimulationEvent ( Util::TraceLevel tracelevel, double time, Entity * entity, ModelComponent * component, std::string text )` `[inline]`

## 5.85.2 Member Function Documentation

5.85.2.1 `ModelComponent* TraceSimulationEvent::getComponent ( )` const `[inline]`

5.85.2.2 `Entity* TraceSimulationEvent::getEntity ( )` const `[inline]`

5.85.2.3 `double TraceSimulationEvent::getTime ( )` const `[inline]`

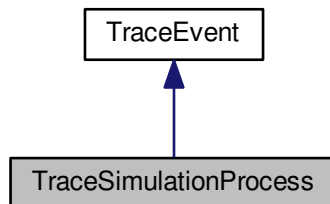
The documentation for this class was generated from the following file:

- [TraceManager.h](#)

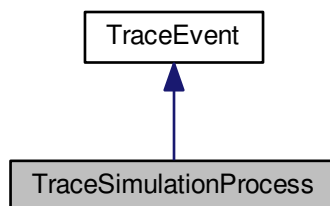
## 5.86 TraceSimulationProcess Class Reference

```
#include <TraceManager.h>
```

Inheritance diagram for TraceSimulationProcess:



Collaboration diagram for TraceSimulationProcess:



### Public Member Functions

- [TraceSimulationProcess](#) ([Util::TraceLevel](#) tracelevel, std::string text)

#### 5.86.1 Detailed Description

Events related to simulation "process" (usually process analyser), associated to entire replication or simulation events (begin/end/pause of replication/simulation) TODO: CLASS NOT COMPLETE

#### 5.86.2 Constructor & Destructor Documentation

5.86.2.1 `TraceSimulationProcess::TraceSimulationProcess ( Util::TraceLevel tracelevel, std::string text )` `[inline]`

The documentation for this class was generated from the following file:

- [TraceManager.h](#)



## 5.87 Traits< T > Struct Template Reference

```
#include <Traits.h>
```

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.88 Traits< Collector\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [CollectorDatafileDefaultImpl1 Implementation](#)

### 5.88.1 Member Typedef Documentation

5.88.1.1 typedef [CollectorDatafileDefaultImpl1 Traits< Collector\\_if >::Implementation](#)

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.89 Traits< ExperimentDesign\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [ExperimentDesignDummyImpl Implementation](#)

### 5.89.1 Member Typedef Documentation

5.89.1.1 typedef [ExperimentDesignDummyImpl Traits< ExperimentDesign\\_if >::Implementation](#)

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.90 Traits< Fitter\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [FitterDummyImpl Implementation](#)

### 5.90.1 Member Typedef Documentation

#### 5.90.1.1 typedef FitterDummyImpl Traits< Fitter\_if >::Implementation

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.91 Traits< GenesysApplication\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [BuildSimulationModel Application](#)

### 5.91.1 Member Typedef Documentation

#### 5.91.1.1 typedef BuildSimulationModel Traits< GenesysApplication\_if >::Application

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.92 Traits< HypothesisTester\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [HypothesisTesterDummyImpl Implementation](#)

### 5.92.1 Member Typedef Documentation

#### 5.92.1.1 typedef HypothesisTesterDummyImpl Traits< HypothesisTester\_if >::Implementation

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.93 Traits< Integrator\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [IntegratorDefaultImpl1](#) Implementation

### Static Public Attributes

- static constexpr unsigned int [MaxIterations](#) = 1000
- static constexpr double [Precision](#) = 1e-9

### 5.93.1 Member Typedef Documentation

#### 5.93.1.1 typedef IntegratorDefaultImpl1 Traits< Integrator\_if >::Implementation

### 5.93.2 Member Data Documentation

#### 5.93.2.1 constexpr unsigned int Traits< Integrator\_if >::MaxIterations = 1000 [static]

#### 5.93.2.2 constexpr double Traits< Integrator\_if >::Precision = 1e-9 [static]

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.94 Traits< Model > Struct Template Reference

```
#include <Traits.h>
```

### Static Public Attributes

- static const bool [debugged](#) = true
- static const [Util::TraceLevel](#) traceLevel = [Util::TraceLevel::mostDetailed](#)

### 5.94.1 Member Data Documentation

5.94.1.1 `const bool Traits< Model >::debugged = true` `[static]`

5.94.1.2 `const Util::TraceLevel Traits< Model >::traceLevel = Util::TraceLevel::mostDetailed` `[static]`

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.95 Traits< ModelChecker\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [ModelCheckerDefaultImpl1](#) [Implementation](#)

### 5.95.1 Member Typedef Documentation

5.95.1.1 `typedef ModelCheckerDefaultImpl1 Traits< ModelChecker_if >::Implementation`

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.96 Traits< ModelComponent > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [StatisticsDefaultImpl1](#) [StatisticsCollector\\_StatisticsImplementation](#)
- typedef [CollectorDefaultImpl1](#) [StatisticsCollector\\_CollectorImplementation](#)

### 5.96.1 Member Typedef Documentation

5.96.1.1 `typedef CollectorDefaultImpl1 Traits< ModelComponent >::StatisticsCollector_Collector↔  
Implementation`

5.96.1.2 `typedef StatisticsDefaultImpl1 Traits< ModelComponent >::StatisticsCollector_Statistics↔  
Implementation`

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.97 Traits< ModelPersistence\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [ModelPersistenceDummyImpl](#) Implementation

### 5.97.1 Member Typedef Documentation

#### 5.97.1.1 typedef ModelPersistenceDummyImpl Traits< ModelPersistence\_if >::Implementation

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.98 Traits< Parser\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef ParserFlexBisonImpl [Implementation](#)

### 5.98.1 Member Typedef Documentation

#### 5.98.1.1 typedef ParserFlexBisonImpl Traits< Parser\_if >::Implementation

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.99 Traits< ProcessAnalyser\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [ProcessAnalyserDummyImpl](#) Implementation

### 5.99.1 Member Typedef Documentation

#### 5.99.1.1 typedef ProcessAnalyserDummyImpl Traits< ProcessAnalyser\_if >::Implementation

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.100 Traits< Sampler\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [SamplerDefaultImpl1 Implementation](#)
- typedef [SamplerDefaultImpl1::DefaultImpl1RNG\\_Parameters Parameters](#)

### 5.100.1 Member Typedef Documentation

#### 5.100.1.1 typedef SamplerDefaultImpl1 Traits< Sampler\_if >::Implementation

#### 5.100.1.2 typedef SamplerDefaultImpl1::DefaultImpl1RNG\_Parameters Traits< Sampler\_if >::Parameters

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.101 Traits< SimulationReporter\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [SimulationReporterDefaultImpl1 Implementation](#)

### 5.101.1 Member Typedef Documentation

#### 5.101.1.1 typedef SimulationReporterDefaultImpl1 Traits< SimulationReporter\_if >::Implementation

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.102 Traits< Statistics\_if > Struct Template Reference

```
#include <Traits.h>
```

### Public Types

- typedef [StatisticsDefaultImpl1](#) [Implementation](#)
- typedef [CollectorDefaultImpl1](#) [CollectorImplementation](#)

### 5.102.1 Member Typedef Documentation

5.102.1.1 typedef [CollectorDefaultImpl1](#) Traits< [Statistics\\_if](#) >::[CollectorImplementation](#)

5.102.1.2 typedef [StatisticsDefaultImpl1](#) Traits< [Statistics\\_if](#) >::[Implementation](#)

The documentation for this struct was generated from the following file:

- [Traits.h](#)

## 5.103 Util Class Reference

```
#include <Util.h>
```

### Public Types

- enum [TimeUnit](#) : int {  
[TimeUnit::picosecond](#) = 1, [TimeUnit::nanosecond](#) = 2, [TimeUnit::microsecond](#) = 3, [TimeUnit::millisecond](#) = 4,  
[TimeUnit::second](#) = 5, [TimeUnit::minute](#) = 6, [TimeUnit::hour](#) = 7, [TimeUnit::day](#) = 8,  
[TimeUnit::week](#) = 9 }
- enum [TraceLevel](#) : int {  
[TraceLevel::noTraces](#) = 0, [TraceLevel::errors](#) = 10, [TraceLevel::report](#) = 20, [TraceLevel::simulation](#) = 30,  
[TraceLevel::transferOnly](#) = 40, [TraceLevel::blockArrival](#) = 50, [TraceLevel::blockInternal](#) = 60, [TraceLevel::mostDetailed](#) = 70 }
- typedef unsigned long [identification](#)
- typedef unsigned int [rank](#)

### Static Public Member Functions

- static void [ClearIndent](#) ()
- static void [IncIndent](#) ()
- static void [DecIndent](#) ()
- static std::string [Indent](#) ()
- static std::string [SetW](#) (std::string text, unsigned short width)
- static [Util::identification](#) [GenerateNewId](#) ()
- static [Util::identification](#) [GenerateNewIdOfType](#) (std::string objtyp)
- static double [TimeUnitConvert](#) ([Util::TimeUnit](#) timeUnit1, [Util::TimeUnit](#) timeUnit2)
- template<class T >  
static std::string [TypeOf](#) ()
- template<class T >  
static [Util::identification](#) [GenerateNewIdOfType](#) ()

### 5.103.1 Member Typedef Documentation

5.103.1.1 typedef unsigned long Util::identitification

5.103.1.2 typedef unsigned int Util::rank

### 5.103.2 Member Enumeration Documentation

5.103.2.1 enum Util::TimeUnit : int [strong]

Enumerator

*picosecond*  
*nanosecond*  
*microsecond*  
*millisecond*  
*second*  
*minute*  
*hour*  
*day*  
*week*

5.103.2.2 enum Util::TraceLevel : int [strong]

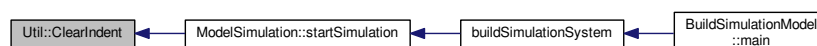
Enumerator

*noTraces*  
*errors*  
*report*  
*simulation*  
*transferOnly*  
*blockArrival*  
*blockInternal*  
*mostDetailed*

### 5.103.3 Member Function Documentation

5.103.3.1 void Util::ClearIndent ( ) [static]

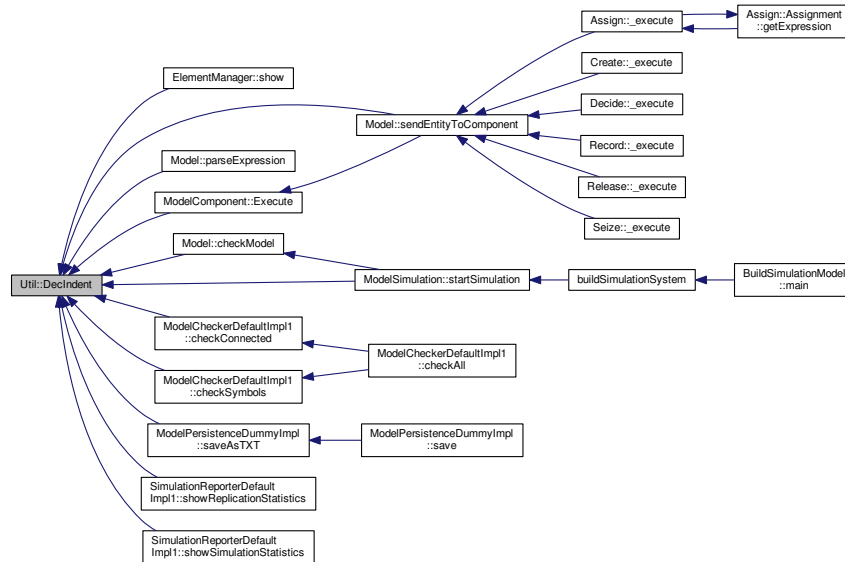
Here is the caller graph for this function:





## 5.103.3.2 void Util::DecIdent ( ) [static]

Here is the caller graph for this function:



## 5.103.3.3 Util::identification Util::GenerateNewId ( ) [static]

## 5.103.3.4 Util::identification Util::GenerateNewIdOfType ( std::string objtyp ) [static]

## 5.103.3.5 template&lt;class T&gt; static Util::identification Util::GenerateNewIdOfType ( ) [inline],[static]

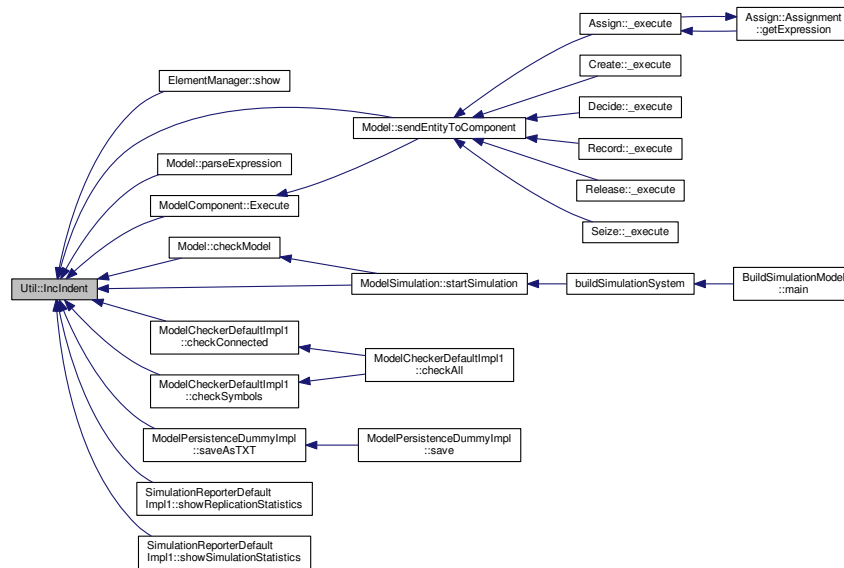
Every component or element has a unique ID for its class, but not unique for other classes. IDs are generated sequentially for each class.

Here is the caller graph for this function:



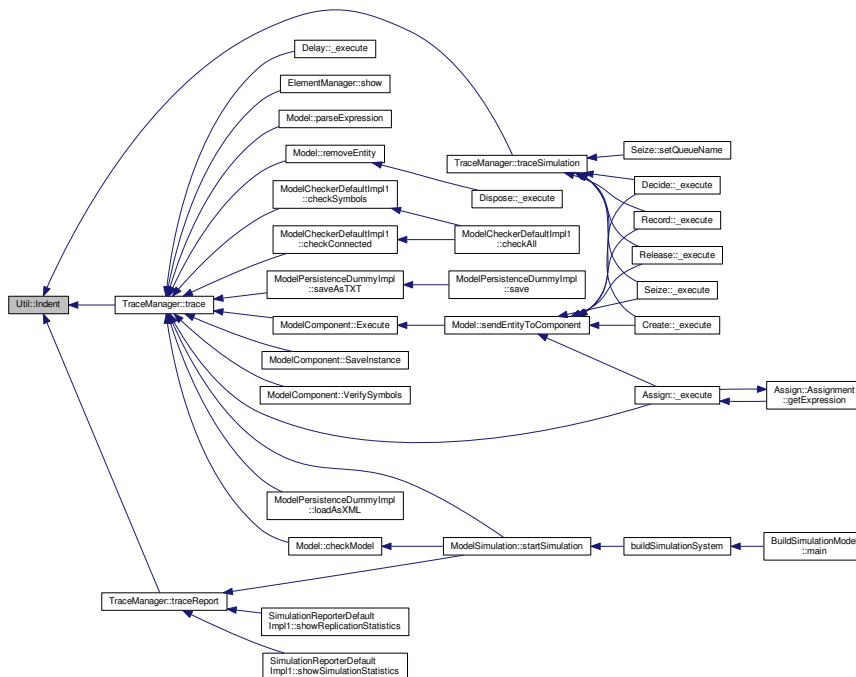
### 5.103.3.6 void Util::IncIndent ( ) [static]

Here is the caller graph for this function:



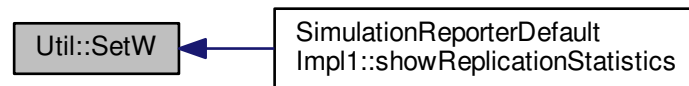
### 5.103.3.7 std::string Util::Indent ( ) [static]

Here is the caller graph for this function:



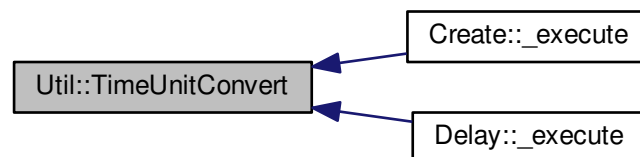
5.103.3.8 `static std::string Util::SetW ( std::string text, unsigned short width )` `[static]`

Here is the caller graph for this function:



5.103.3.9 `double Util::TimeUnitConvert ( Util::TimeUnit timeUnit1, Util::TimeUnit timeUnit2 )` `[static]`

Here is the caller graph for this function:



5.103.3.10 `template<class T> static std::string Util::TypeOf ( )` `[inline],[static]`

Return the name of the class used as T.

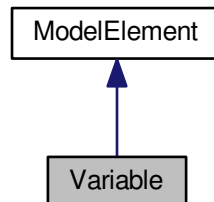
The documentation for this class was generated from the following files:

- [Util.h](#)
- [Util.cpp](#)

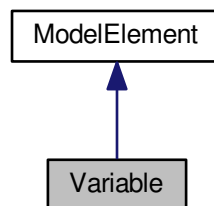
## 5.104 Variable Class Reference

```
#include <Variable.h>
```

Inheritance diagram for Variable:



Collaboration diagram for Variable:



### Public Member Functions

- [Variable](#) ()
- [Variable](#) (std::string name)
- [Variable](#) (const [Variable](#) &orig)
- virtual [~Variable](#) ()
- virtual std::string [show](#) ()
- double [getValue](#) ()
- double [getValue](#) (std::string index)
- void [setValue](#) (double value)
- void [setValue](#) (std::string index, double value)

### Protected Member Functions

- virtual void [\\_loadInstance](#) (std::list< std::string > words)
- virtual std::list< std::string > \* [\\_saveInstance](#) ()
- virtual bool [\\_verifySymbols](#) (std::string \*errorMessage)

## Additional Inherited Members

### 5.104.1 Constructor & Destructor Documentation

5.104.1.1 `Variable::Variable ( )`

5.104.1.2 `Variable::Variable ( std::string name )`

5.104.1.3 `Variable::Variable ( const Variable & orig )`

5.104.1.4 `Variable::~~Variable ( )` `[virtual]`

### 5.104.2 Member Function Documentation

5.104.2.1 `void Variable::_loadInstance ( std::list< std::string > words )` `[protected]`, `[virtual]`

Implements [ModelElement](#).

5.104.2.2 `std::list< std::string > * Variable::_saveInstance ( )` `[protected]`, `[virtual]`

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



5.104.2.3 `bool Variable::_verifySymbols ( std::string * errorMessage )` `[protected]`, `[virtual]`

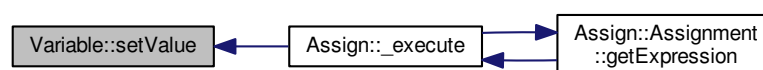
Implements [ModelElement](#).

5.104.2.4 `double Variable::getValue ( )`

5.104.2.5 `double Variable::getValue ( std::string index )`

5.104.2.6 `void Variable::setValue ( double value )`

Here is the caller graph for this function:



5.104.2.7 `void Variable::setValue ( std::string index, double value )`

5.104.2.8 `std::string Variable::show ( )` [virtual]

Reimplemented from [ModelElement](#).

Here is the call graph for this function:



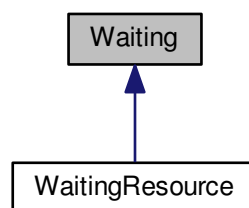
The documentation for this class was generated from the following files:

- [Variable.h](#)
- [Variable.cpp](#)

## 5.105 Waiting Class Reference

```
#include <Waiting.h>
```

Inheritance diagram for Waiting:



### Public Member Functions

- [Waiting](#) ([Entity](#) \*entity, [ModelComponent](#) \*component, double timeStartedWaiting)
- [Waiting](#) (const [Waiting](#) &orig)
- virtual [~Waiting](#) ()
- virtual std::string [show](#) ()
- double [getTimeStartedWaiting](#) () const
- [ModelComponent](#) \* [getComponent](#) () const
- [Entity](#) \* [getEntity](#) () const

## 5.105.1 Constructor & Destructor Documentation

5.105.1.1 `Waiting::Waiting ( Entity * entity, ModelComponent * component, double timeStartedWaiting )`

5.105.1.2 `Waiting::Waiting ( const Waiting & orig )`

5.105.1.3 `Waiting::~Waiting ( ) [virtual]`

## 5.105.2 Member Function Documentation

5.105.2.1 `ModelComponent * Waiting::getComponent ( ) const`

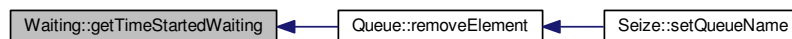
5.105.2.2 `Entity * Waiting::getEntity ( ) const`

Here is the caller graph for this function:



5.105.2.3 `double Waiting::getTimeStartedWaiting ( ) const`

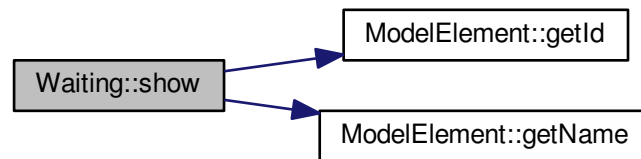
Here is the caller graph for this function:



5.105.2.4 `std::string Waiting::show ( ) [virtual]`

Reimplemented in [WaitingResource](#).

Here is the call graph for this function:



Here is the caller graph for this function:



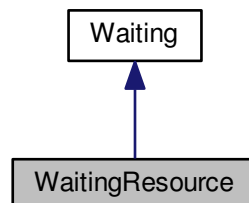
The documentation for this class was generated from the following files:

- [Waiting.h](#)
- [Waiting.cpp](#)

## 5.106 WaitingResource Class Reference

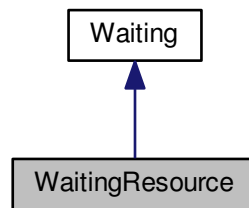
```
#include <WaitingResource.h>
```

Inheritance diagram for `WaitingResource`:





Collaboration diagram for WaitingResource:



### Public Member Functions

- [WaitingResource](#) ([Entity](#) \*entity, [ModelComponent](#) \*component, double timeStartedWaiting, unsigned int quantity)
- [WaitingResource](#) (const [WaitingResource](#) &orig)
- virtual [~WaitingResource](#) ()
- virtual std::string [show](#) ()
- unsigned int [getQuantity](#) () const

#### 5.106.1 Constructor & Destructor Documentation

5.106.1.1 [WaitingResource::WaitingResource](#) ( [Entity](#) \* *entity*, [ModelComponent](#) \* *component*, double *timeStartedWaiting*, unsigned int *quantity* )

5.106.1.2 [WaitingResource::WaitingResource](#) ( const [WaitingResource](#) & *orig* )

5.106.1.3 [WaitingResource::~~WaitingResource](#) ( ) [virtual]

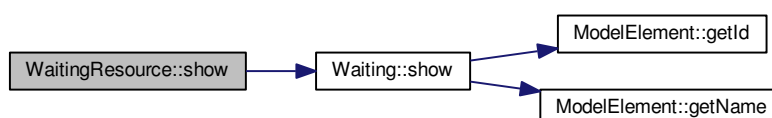
#### 5.106.2 Member Function Documentation

5.106.2.1 unsigned int [WaitingResource::getQuantity](#) ( ) const

5.106.2.2 std::string [WaitingResource::show](#) ( ) [virtual]

Reimplemented from [Waiting](#).

Here is the call graph for this function:



The documentation for this class was generated from the following files:

- [WaitingResource.h](#)
- [WaitingResource.cpp](#)

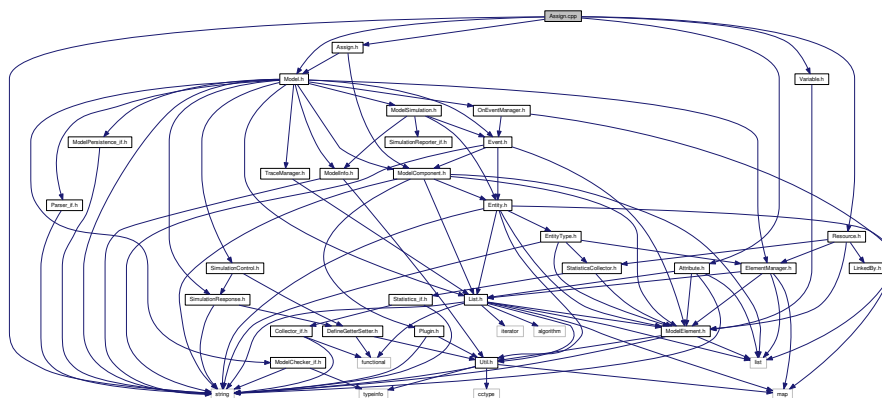
# File Documentation

## 6.1 .dep.inc File Reference

## 6.2 Assign.cpp File Reference

```
#include "Assign.h"
#include <string>
#include "Model.h"
#include "Variable.h"
#include "Attribute.h"
#include "Resource.h"
```

Include dependency graph for Assign.cpp:

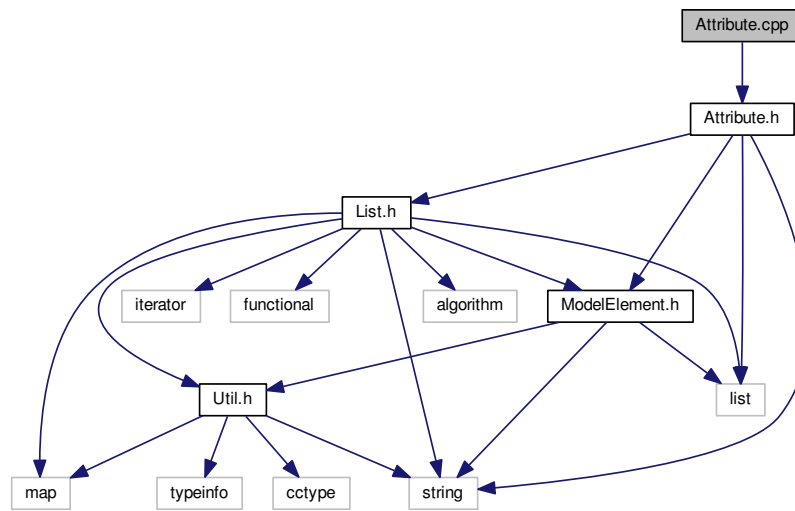


### 6.3 Assign.h File Reference

```
#include "ModelComponent.h"
#include "Model.h"
```



Include dependency graph for Attribute.cpp:



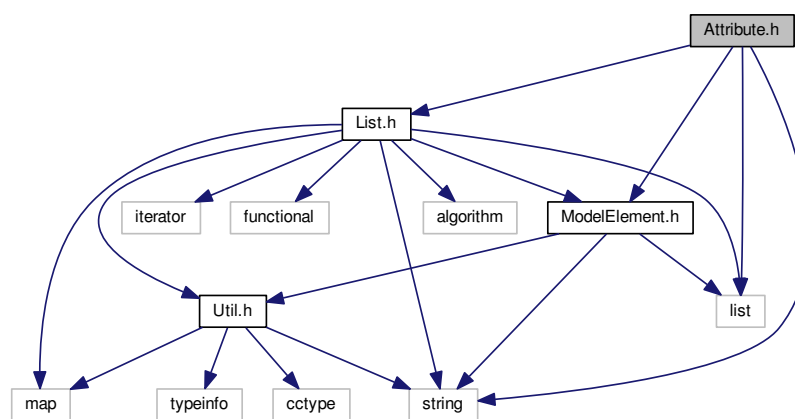
## 6.5 Attribute.h File Reference

```

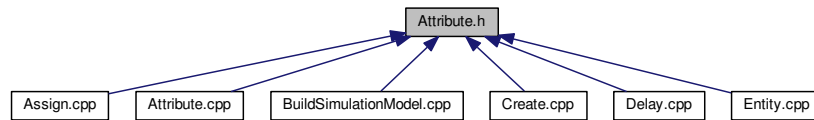
#include <string>
#include <list>
#include "List.h"
#include "ModelElement.h"

```

Include dependency graph for Attribute.h:



This graph shows which files directly or indirectly include this file:



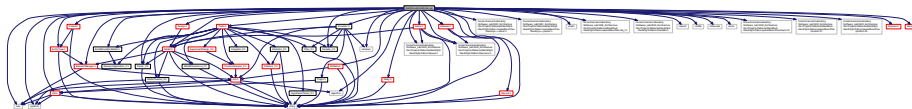
## Classes

- class [Attribute](#)

## 6.6 BuildSimulationModel.cpp File Reference

```
#include "BuildSimulationModel.h"
#include "Simulator.h"
#include "Traits.h"
#include "Create.h"
#include "Delay.h"
#include "Dispose.h"
#include "Seize.h"
#include "Release.h"
#include "Assign.h"
#include "Record.h"
#include "Decide.h"
#include "ElementManager.h"
#include "EntityType.h"
#include "Attribute.h"
```

Include dependency graph for BuildSimulationModel.cpp:



## Functions

- void [traceHandler](#) ([TraceEvent](#) e)
- void [traceSimulationHandler](#) ([TraceSimulationEvent](#) e)
- void [onSimulationStartHandler](#) ([SimulationEvent](#) \*re)
- void [onReplicationStartHandler](#) ([SimulationEvent](#) \*re)
- void [onProcessEventHandler](#) ([SimulationEvent](#) \*re)
- void [onReplicationEndHandler](#) ([SimulationEvent](#) \*re)
- void [onEntityRemoveHandler](#) ([SimulationEvent](#) \*re)
- void [buildModel](#) ([Model](#) \*model)
- void [buildSimulationSystem](#) ()

## 6.6.1 Function Documentation

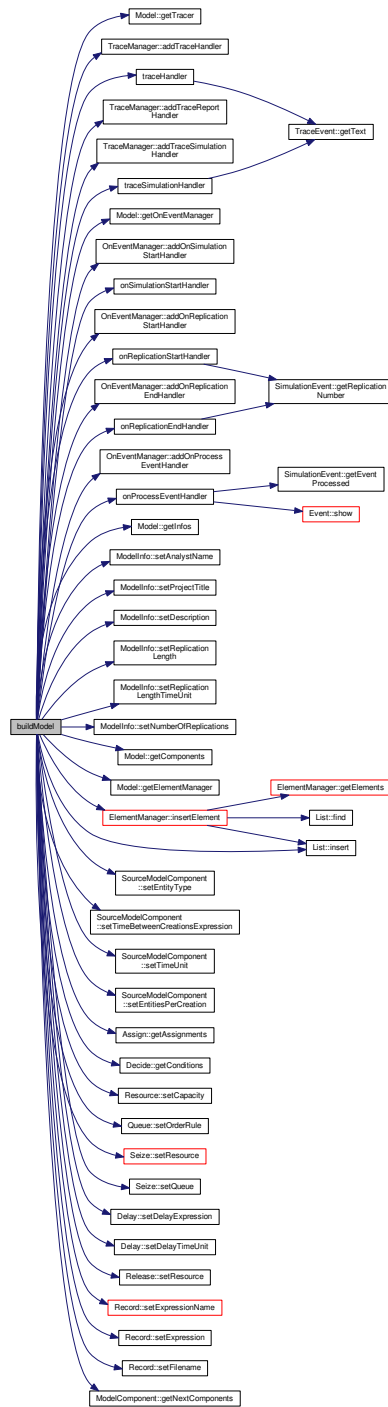
### 6.6.1.1 void buildModel ( Model \* *model* )

This function shows an example of how to create a simulation model. It creates some handlers for tracing (debug) and for events, set model infos and than creates the model itself. The model is a composition of components (and elements that they use), connected to form a process/fluxogram

#### Parameters

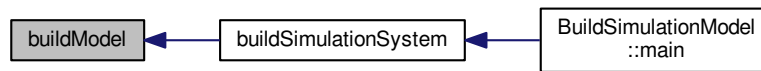
<i>model</i>	- The instance returned that will contains the built model
--------------	--

Here is the call graph for this function:





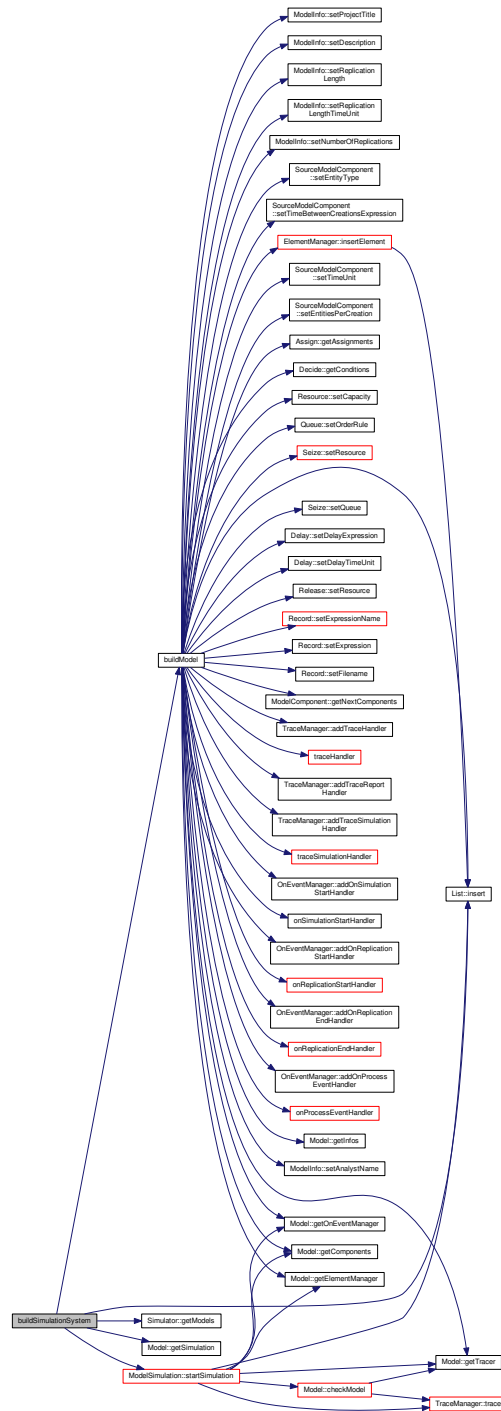
Here is the caller graph for this function:



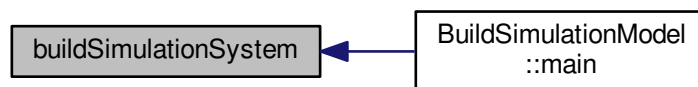
#### 6.6.1.2 void buildSimulationSystem ( )

This is the main function of the [BuildSimulationModel](#) application. It instantiates the simulator, builds a simulation model and then simulate that model.

Here is the call graph for this function:

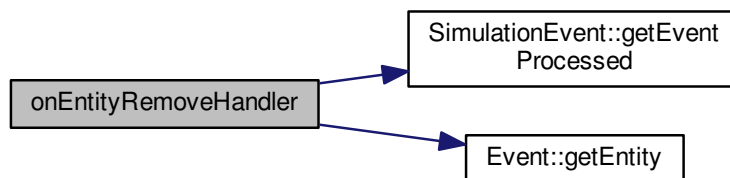


Here is the caller graph for this function:



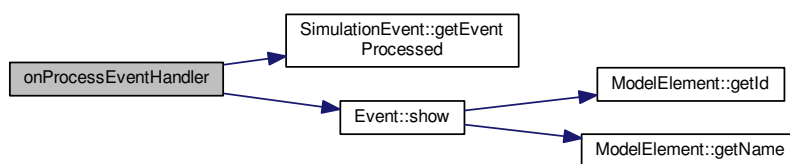
#### 6.6.1.3 void onEntityRemoveHandler ( SimulationEvent \* re )

Here is the call graph for this function:



#### 6.6.1.4 void onProcessEventHandler ( SimulationEvent \* re )

Here is the call graph for this function:

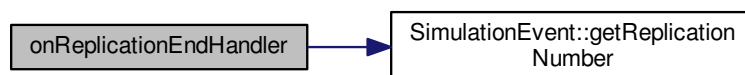


Here is the caller graph for this function:



#### 6.6.1.5 void onReplicationEndHandler ( SimulationEvent \* re )

Here is the call graph for this function:

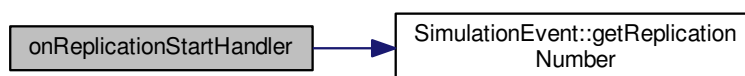


Here is the caller graph for this function:



#### 6.6.1.6 void onReplicationStartHandler ( SimulationEvent \* re )

Here is the call graph for this function:

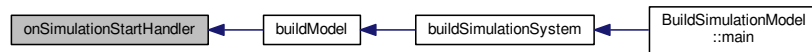


Here is the caller graph for this function:



#### 6.6.1.7 void onSimulationStartHandler ( SimulationEvent \* *re* )

Here is the caller graph for this function:

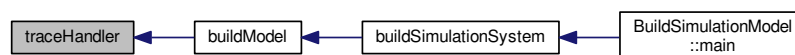


#### 6.6.1.8 void traceHandler ( TraceEvent *e* )

Here is the call graph for this function:



Here is the caller graph for this function:

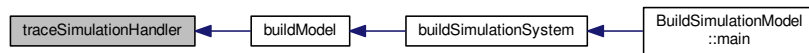


#### 6.6.1.9 void traceSimulationHandler ( TraceSimulationEvent *e* )

Here is the call graph for this function:

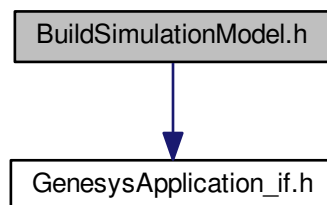


Here is the caller graph for this function:

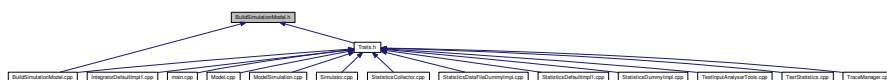


## 6.7 BuildSimulationModel.h File Reference

```
#include "GenesysApplication_if.h"
Include dependency graph for BuildSimulationModel.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [BuildSimulationModel](#)

## 6.8 Collector\_if.h File Reference

```
#include <string>
#include <functional>
```



## 6.8.1 Typedef Documentation

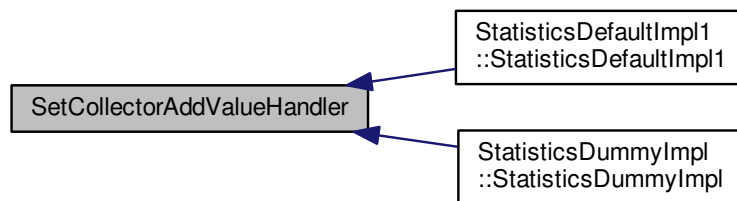
6.8.1.1 `typedef std::function<void(double)> CollectorAddValueHandler`

6.8.1.2 `typedef std::function<void()> CollectorClearHandler`

## 6.8.2 Function Documentation

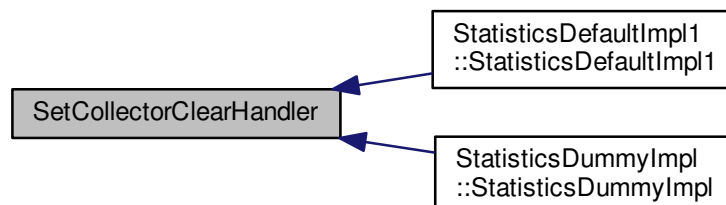
6.8.2.1 `template<typename Class> CollectorAddValueHandler SetCollectorAddValueHandler ( void(Class::*)(double) function, Class * object )`

Here is the caller graph for this function:



6.8.2.2 `template<typename Class> CollectorClearHandler SetCollectorClearHandler ( void(Class::*)() function, Class * object )`

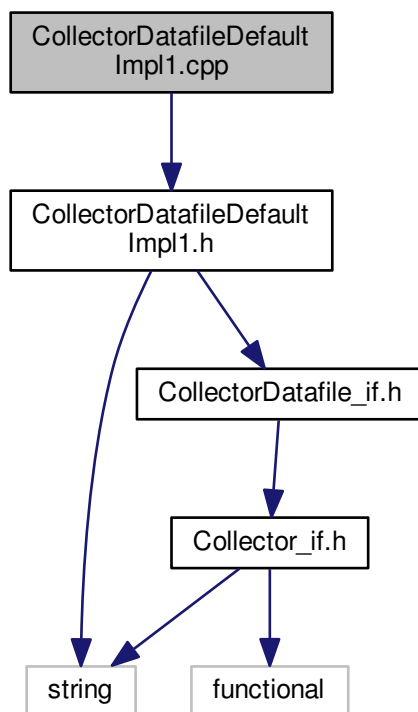
Here is the caller graph for this function:







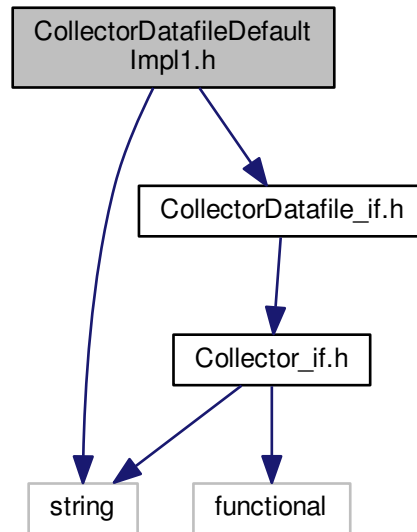
Include dependency graph for CollectorDatafileDefaultImpl1.cpp:



## 6.11 CollectorDatafileDefaultImpl1.h File Reference

```
#include <string>
#include "CollectorDatafile_if.h"
```

Include dependency graph for CollectorDatafileDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



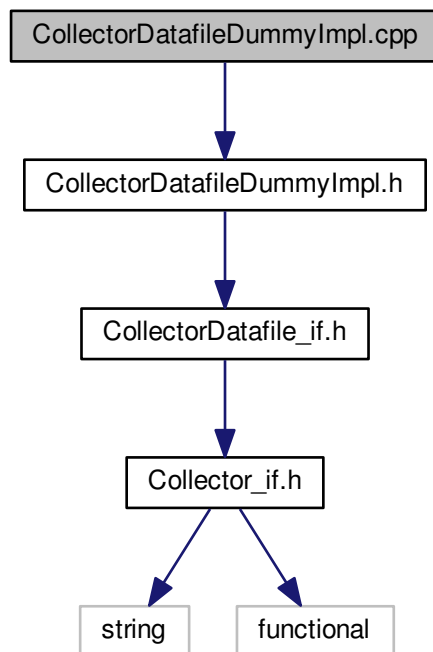
## Classes

- class [CollectorDatafileDefaultImpl1](#)

## 6.12 CollectorDatafileDummyImpl.cpp File Reference

```
#include "CollectorDatafileDummyImpl.h"
```

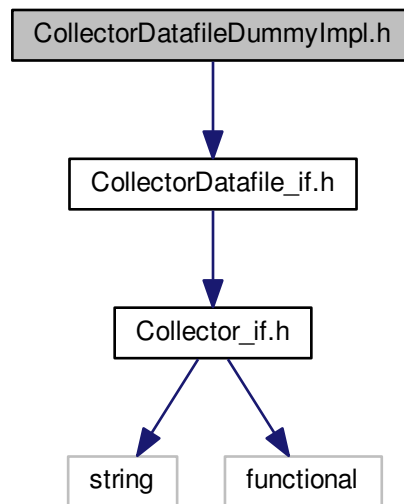
Include dependency graph for CollectorDatafileDummyImpl.cpp:



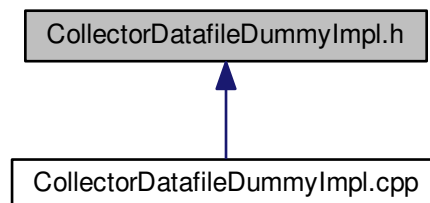
### 6.13 CollectorDatafileDummyImpl.h File Reference

```
#include "CollectorDatafile_if.h"
```

Include dependency graph for CollectorDatafileDummyImpl.h:



This graph shows which files directly or indirectly include this file:



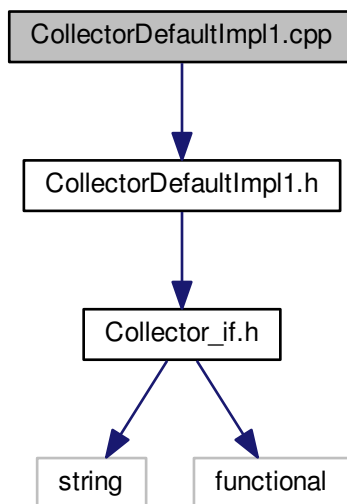
## Classes

- class [CollectorDatafileDummyImpl](#)

## 6.14 CollectorDefaultImpl1.cpp File Reference

```
#include "CollectorDefaultImpl1.h"
```

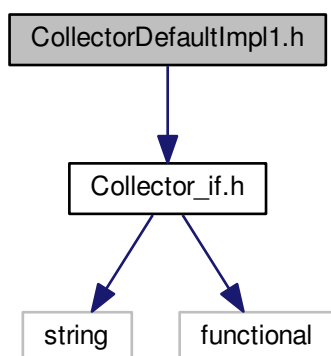
Include dependency graph for CollectorDefaultImpl1.cpp:



## 6.15 CollectorDefaultImpl1.h File Reference

```
#include "Collector_if.h"
```

Include dependency graph for CollectorDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



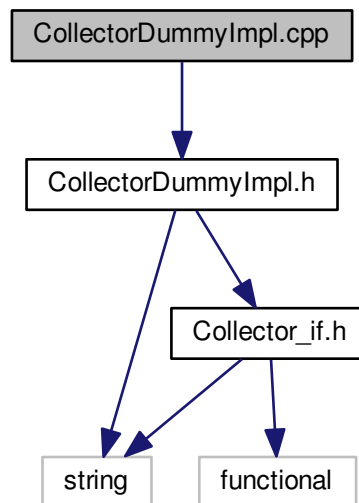
## Classes

- class [CollectorDefaultImpl1](#)

## 6.16 CollectorDummyImpl.cpp File Reference

```
#include "CollectorDummyImpl.h"
```

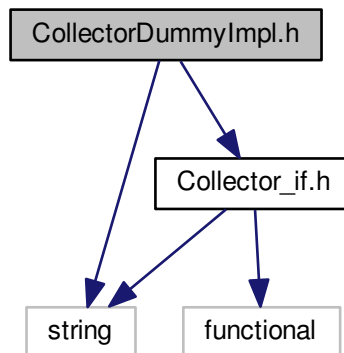
Include dependency graph for CollectorDummyImpl.cpp:



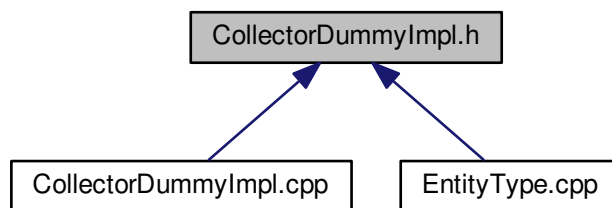
## 6.17 CollectorDummyImpl.h File Reference

```
#include <string>
#include "Collector_if.h"
```

Include dependency graph for CollectorDummyImpl.h:



This graph shows which files directly or indirectly include this file:



## Classes

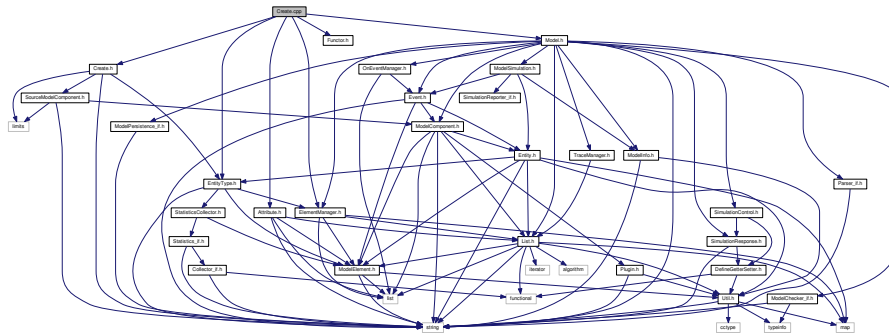
- class [CollectorDummyImpl](#)

## 6.18 Create.cpp File Reference

```
#include "Create.h"
#include "Model.h"
#include "EntityType.h"
#include "Functor.h"
#include "ElementManager.h"
#include "Attribute.h"
```

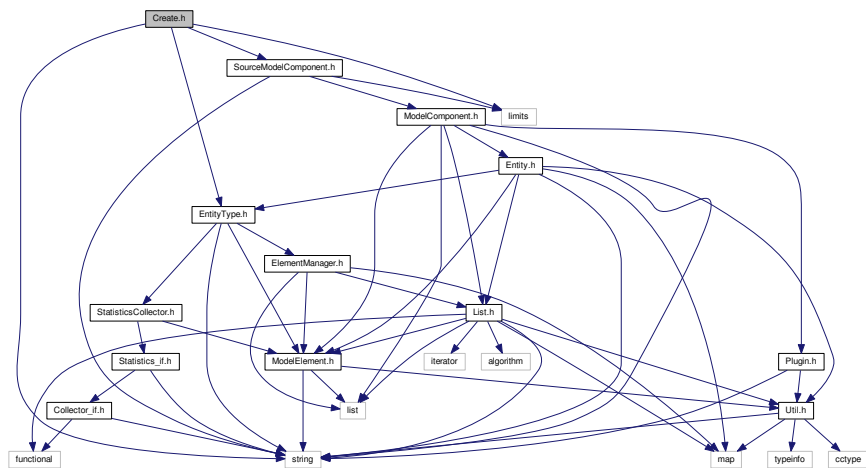


Include dependency graph for Create.cpp:

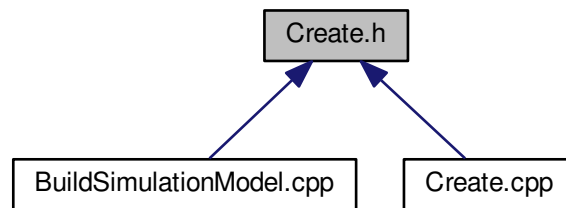


## 6.19 Create.h File Reference

```
#include <string>
#include <limits>
#include "SourceModelComponent.h"
#include "EntityType.h"
Include dependency graph for Create.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

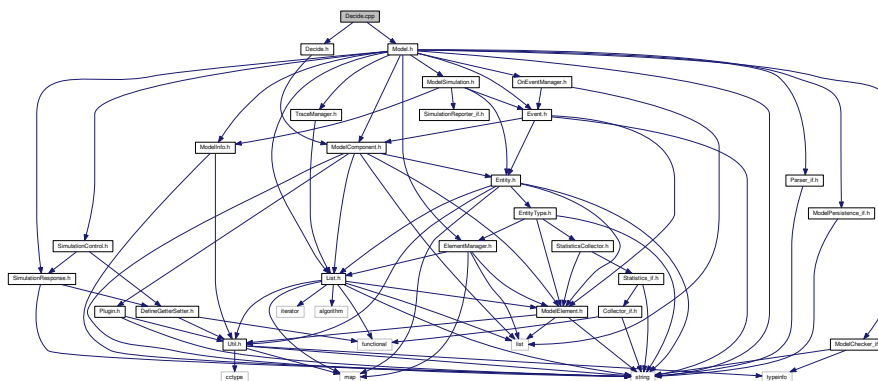
- class Create

## 6.20 Decide.cpp File Reference

```
#include "Decide.h"
```

```
#include "Model.h"
```

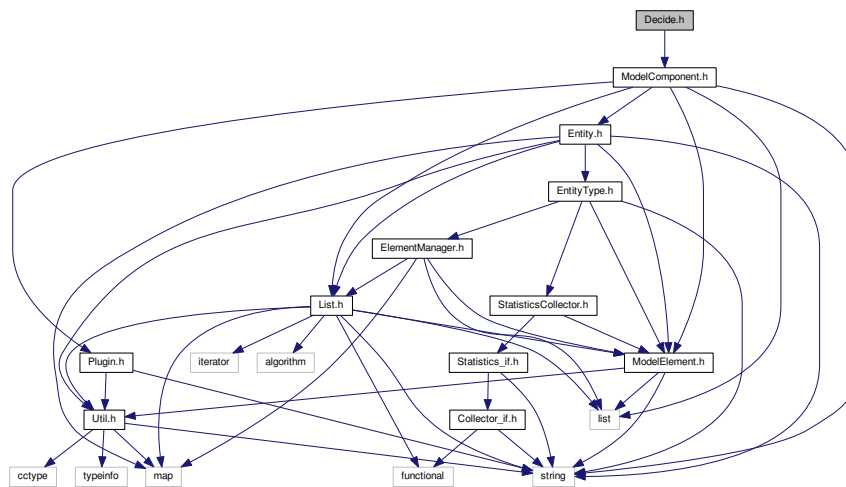
Include dependency graph for Decide.cpp:



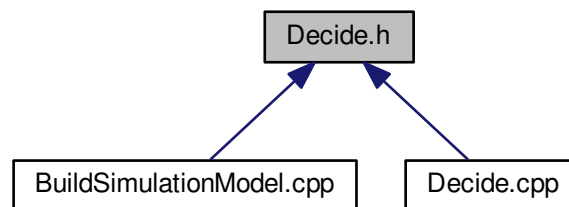
## 6.21 Decide.h File Reference

```
#include "ModelComponent.h"
```

Include dependency graph for Decide.h:



This graph shows which files directly or indirectly include this file:



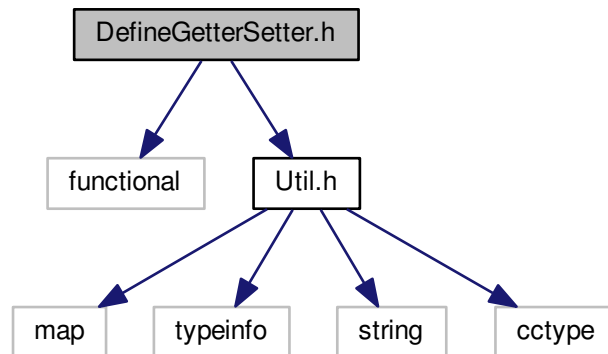
## Classes

- class [Decide](#)

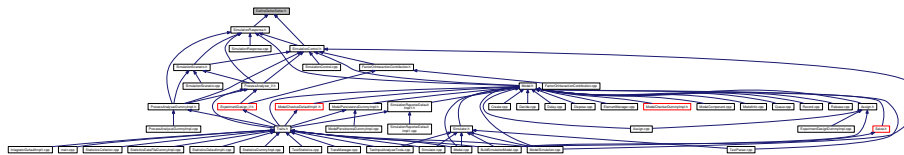
## 6.22 DefineGetterSetter.h File Reference

```
#include <functional>
#include "Util.h"
```

Include dependency graph for DefineGetterSetter.h:



This graph shows which files directly or indirectly include this file:



## Typedefs

- `typedef std::function< double() > GetterMember`
- `typedef std::function< void(double) > SetterMember`

## Functions

- `template<typename Class > GetterMember DefineGetterMember (Class *object, double(Class::*function)())`
- `template<typename Class > SetterMember DefineSetterMember (Class *object, void(Class::*function)(double))`
- `template<typename Class > GetterMember DefineGetterMember (Class *object, unsigned int(Class::*function)() const)`
- `template<typename Class > SetterMember DefineSetterMember (Class *object, void(Class::*function)(unsigned int))`
- `template<typename Class > GetterMember DefineGetterMember (Class *object, bool(Class::*function)() const)`
- `template<typename Class > SetterMember DefineSetterMember (Class *object, void(Class::*function)(bool))`
- `template<typename Class > GetterMember DefineGetterMember (Class *object, std::string(Class::*function)() const)`
- `template<typename Class > SetterMember DefineSetterMember (Class *object, void(Class::*function)(std::string) const)`
- `template<typename Class > GetterMember DefineGetterMember (Class *object, Util::TimeUnit(Class::*function)() const)`
- `template<typename Class > SetterMember DefineSetterMember (Class *object, void(Class::*function)(Util::TimeUnit))`

### 6.22.1 Typedef Documentation

### 6.22.1.1 typedef std::function<double()> GetterMember

### 6.22.1.2 `typedef std::function<void(double)> SetterMember`

### 6.22.2 Function Documentation

### 6.22.2.1 `template<typename Class> GetterMember DefineGetterMember ( Class * object, double(Class::*)( ) function )`

```
6.22.2.2 template<typename Class > GetterMember DefineGetterMember ( Class * object, unsigned int(Class::*)() const
function )
```

### 6.22.2.3 `template<typename Class> GetterMember DefineGetterMember ( Class * object, bool(Class::*)() const function )`

```
6.22.2.4 template<typename Class > GetterMember DefineGetterMember ( Class * object, std::string(Class::*)() const
function )
```

```
6.22.2.5 template<typename Class > GetterMember DefineGetterMember ( Class * object, Util::TimeUnit(Class::*)()
const function )
```

### 6.22.2.6 `template<typename Class> SetterMember DefineSetterMember ( Class * object, void(Class::*)(double) function )`

```
6.22.2.7 template<typename Class > SetterMember DefineSetterMember ( Class * object, void(Class::*)(unsigned int)
function )
```

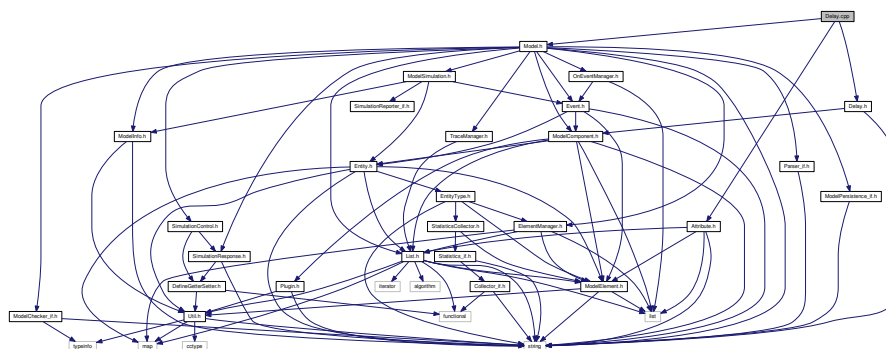
### 6.22.2.8 `template<typename Class> SetterMember DefineSetterMember ( Class * object, void(Class::*)(bool) function )`

```
6.22.2.9 template<typename Class > SetterMember DefineSetterMember ( Class * object, void(Class::*)(std::string) const
function )
```

```
6.22.2.10 template<typename Class > SetterMember DefineSetterMember ( Class * object, void(Class::*)(Util::TimeUnit)
function )
```

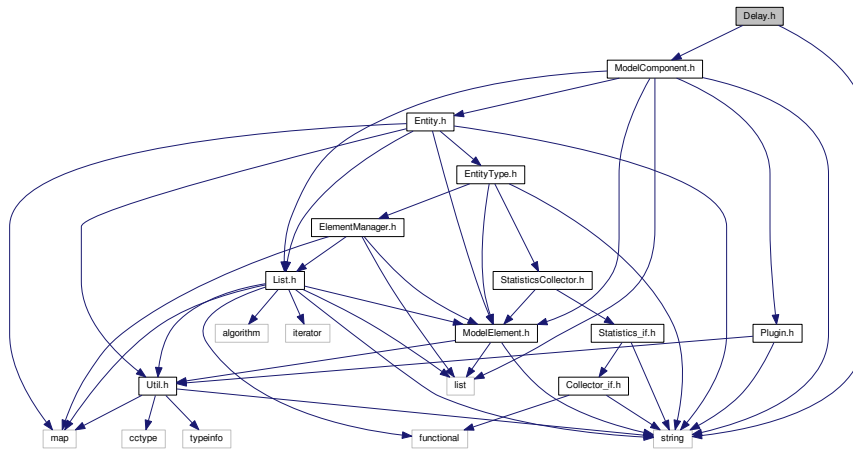
### 6.23 Delay.cpp File Reference

```
#include "Delay.h"
#include "Model.h"
#include "Attribute.h"
Include dependency graph for Delay.cpp:
```

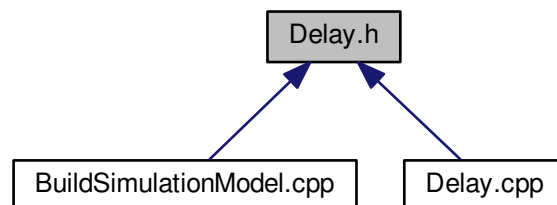


## 6.24 Delay.h File Reference

```
#include <string>
#include "ModelComponent.h"
Include dependency graph for Delay.h:
```



This graph shows which files directly or indirectly include this file:



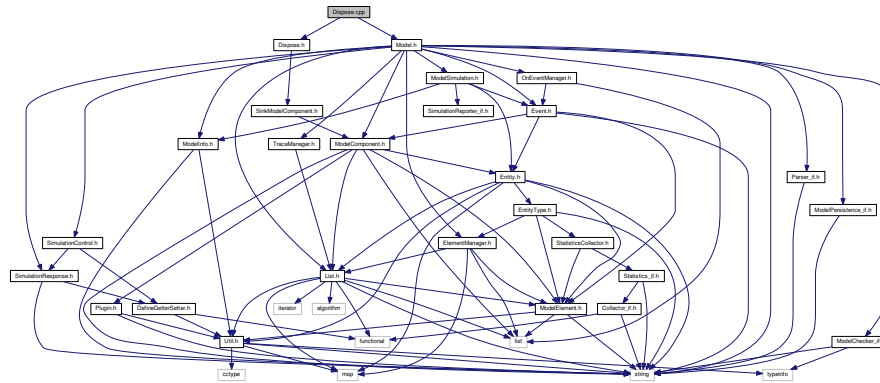
### Classes

- class [Delay](#)

## 6.25 Dispose.cpp File Reference

```
#include "Dispose.h"
#include "Model.h"
```

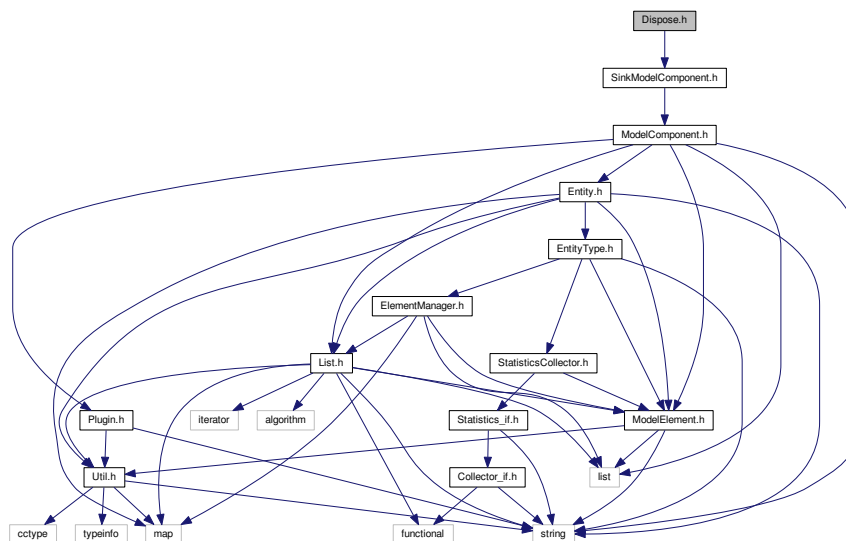
Include dependency graph for Dispose.cpp:



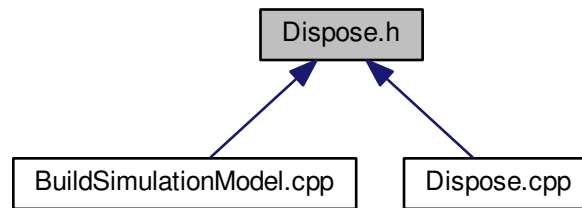
## 6.26 Dispose.h File Reference

```
#include "SinkModelComponent.h"
```

Include dependency graph for Dispose.h:



This graph shows which files directly or indirectly include this file:



## Classes

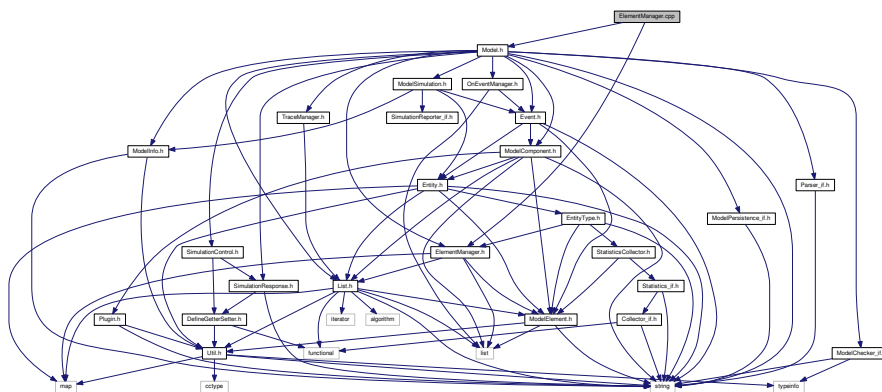
- class [Dispose](#)

## 6.27 ElementManager.cpp File Reference

```
#include "ElementManager.h"
```

```
#include "Model.h"
```

Include dependency graph for ElementManager.cpp:



## 6.28 ElementManager.h File Reference

```
#include <list>
```

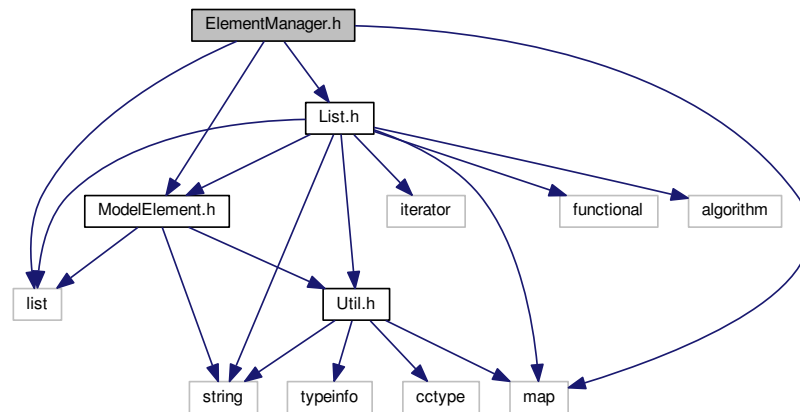
```
#include <map>
```

```
#include "List.h"
```

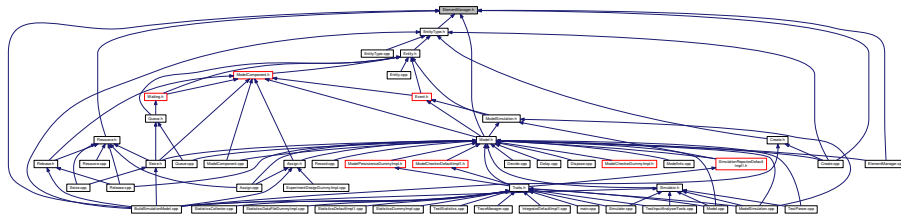
```
#include "ModelElement.h"
```



Include dependency graph for ElementManager.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [ElementManager](#)

## 6.29 ElementManager\_if.h File Reference

### Classes

- class [ElementManager\\_if](#)

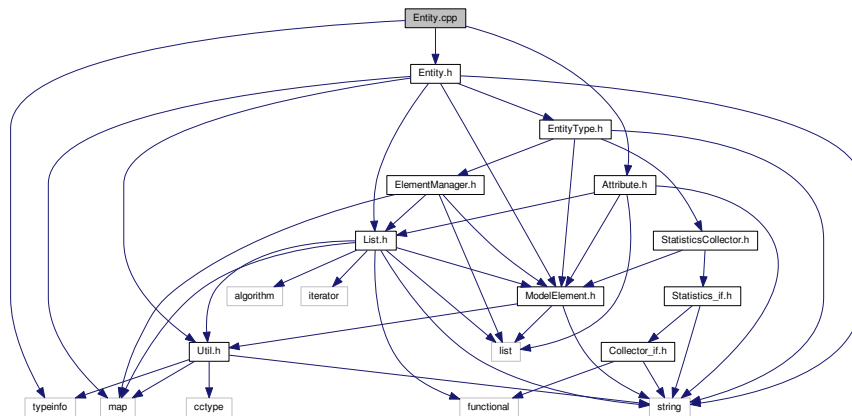
## 6.30 Entity.cpp File Reference

```

#include <typeinfo>
#include "Entity.h"
#include "Attribute.h"

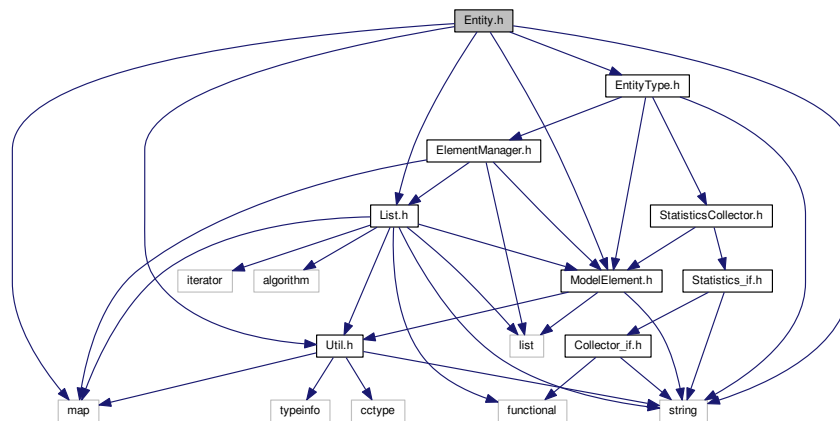
```

Include dependency graph for Entity.cpp:

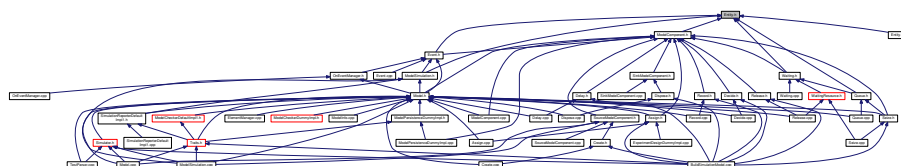


## 6.31 Entity.h File Reference

```
#include <string>
#include <map>
#include "Util.h"
#include "List.h"
#include "ModelElement.h"
#include "EntityType.h"
Include dependency graph for Entity.h:
```



This graph shows which files directly or indirectly include this file:

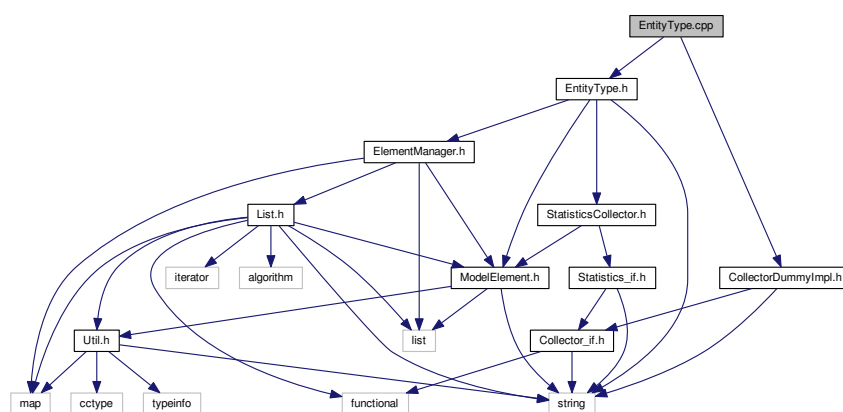


## Classes

- class Entity

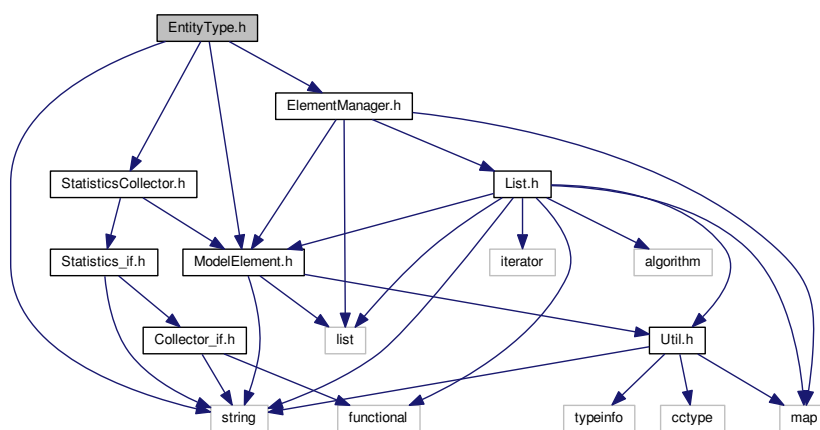
### 6.32 EntityType.cpp File Reference

```
#include "EntityType.h"
#include "CollectorDummyImpl.h"
Include dependency graph for EntityType.cpp:
```

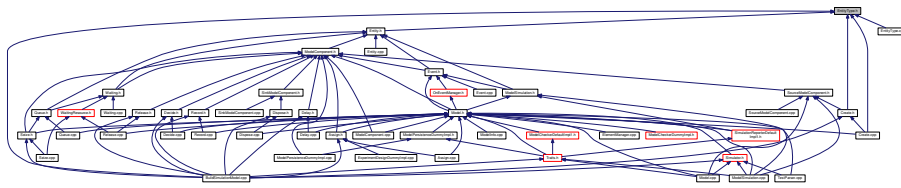


### 6.33 EntityType.h File Reference

```
#include <string>
#include "ModelElement.h"
#include "StatisticsCollector.h"
#include "ElementManager.h"
Include dependency graph for EntityType.h:
```



This graph shows which files directly or indirectly include this file:



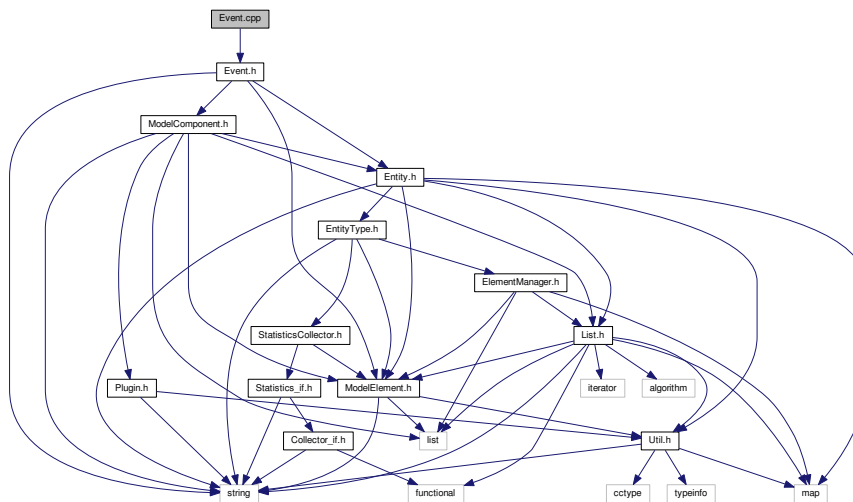
## Classes

- class [EntityType](#)

## 6.34 Event.cpp File Reference

```
#include "Event.h"
```

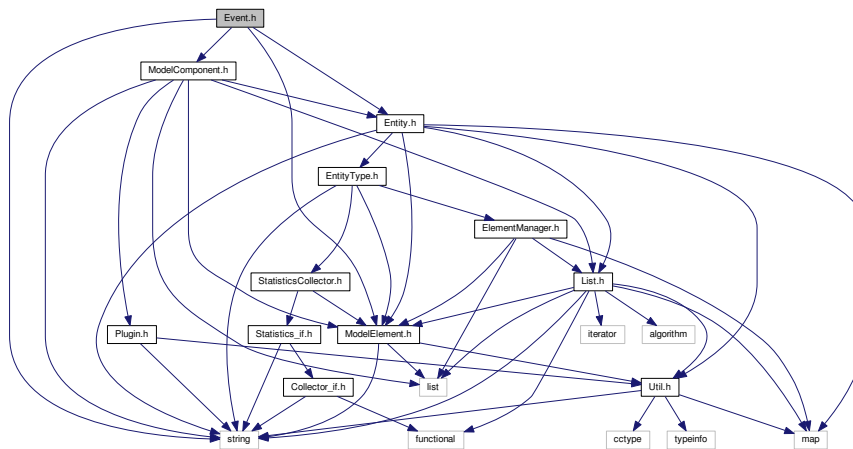
Include dependency graph for Event.cpp:



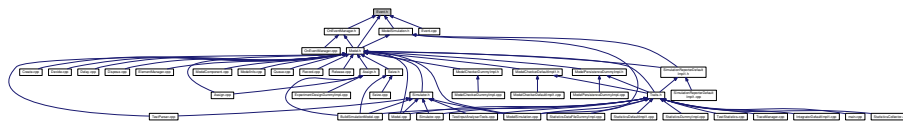
## 6.35 Event.h File Reference

```
#include <string>
#include "ModelElement.h"
#include "Entity.h"
#include "ModelComponent.h"
```

Include dependency graph for Event.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Event](#)

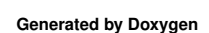
## 6.36 ExperimentDesign\_if.h File Reference

```
#include "FactorOrInteractionContribution.h"
#include "ProcessAnalyser_if.h"
```



- ### 6.37 ExperimentDesignDummyImpl.cpp File Reference

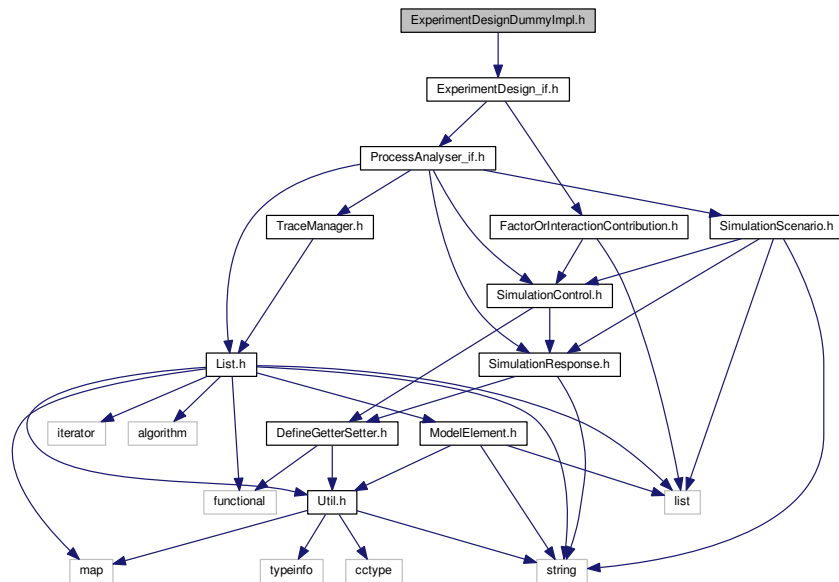
Include dependency graph for ExperimentDesignDummyImpl.cpp:



## 6.38 ExperimentDesignDummyImpl.h File Reference

```
#include "ExperimentDesign_if.h"
```

Include dependency graph for ExperimentDesignDummyImpl.h:



This graph shows which files directly or indirectly include this file:



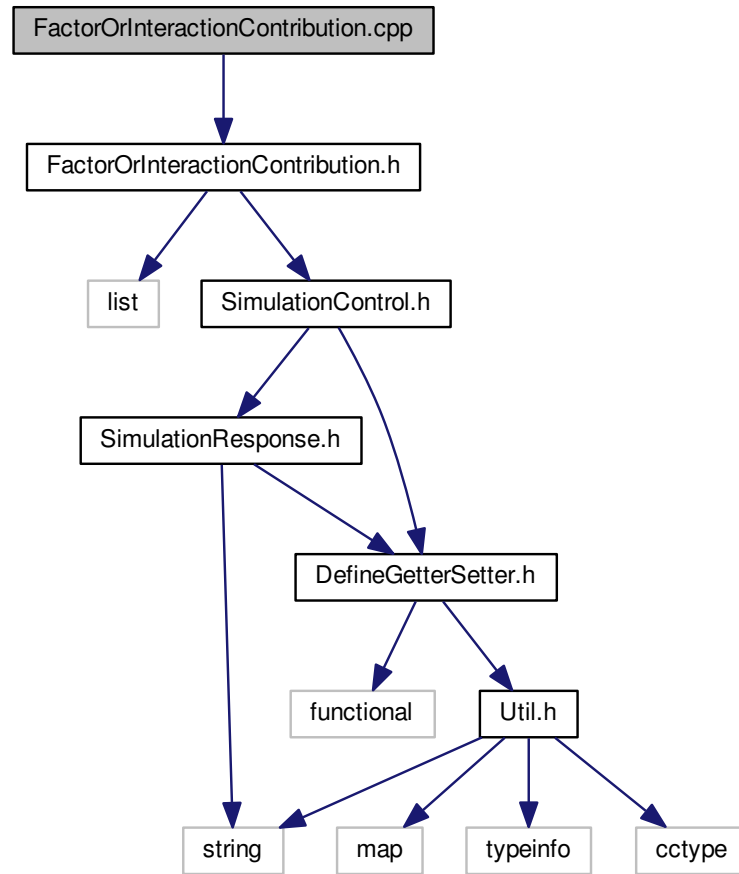
### Classes

- class [ExperimentDesignDummyImpl](#)

## 6.39 FactorOrInteractionContribution.cpp File Reference

```
#include "FactorOrInteractionContribution.h"
```

Include dependency graph for FactorOrInteractionContribution.cpp:

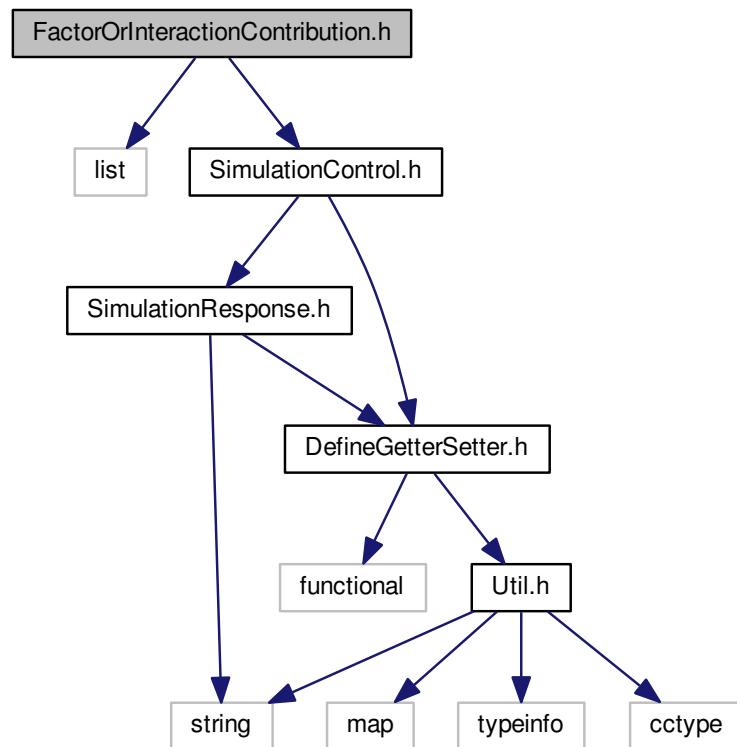


## 6.40 FactorOrInteractionContribution.h File Reference

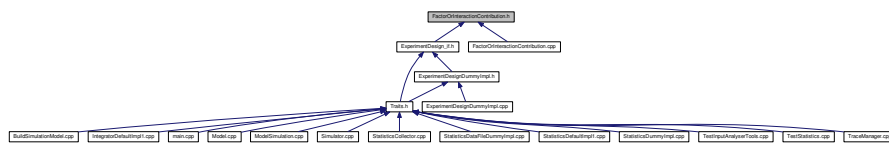
```
#include <list>
#include "SimulationControl.h"
```



Include dependency graph for FactorOrInteractionContribution.h:



This graph shows which files directly or indirectly include this file:



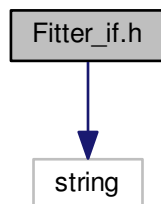
## Classes

- class [FactorOrInteractionContribution](#)

## 6.41 Fitter\_if.h File Reference

```
#include <string>
```

Include dependency graph for Fitter\_if.h:



This graph shows which files directly or indirectly include this file:



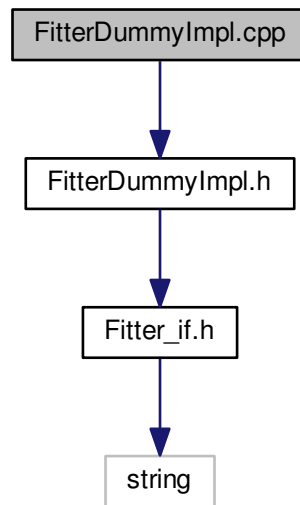
## Classes

- class `Fitter_if`

## 6.42 FitterDummyImpl.cpp File Reference

```
#include "FitterDummyImpl.h"
```

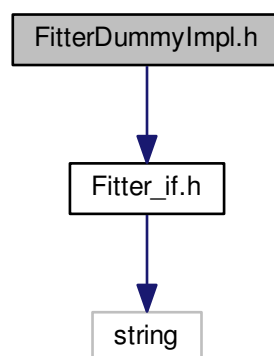
Include dependency graph for FitterDummyImpl.cpp:



## 6.43 FitterDummyImpl.h File Reference

```
#include "Fitter_if.h"
```

Include dependency graph for FitterDummyImpl.h:



This graph shows which files directly or indirectly include this file:

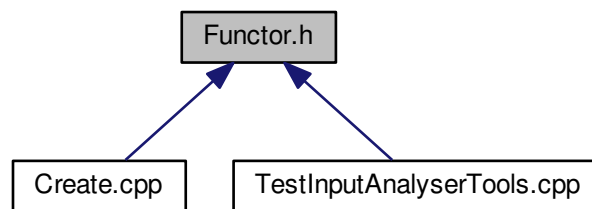


## Classes

- class [FitterDummyImpl](#)

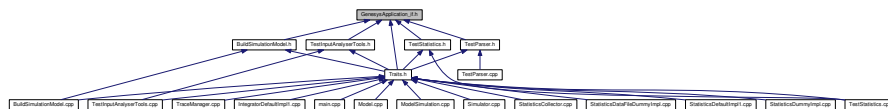
## 6.44 Functor.h File Reference

This graph shows which files directly or indirectly include this file:



## 6.45 GenesysApplication\_if.h File Reference

This graph shows which files directly or indirectly include this file:

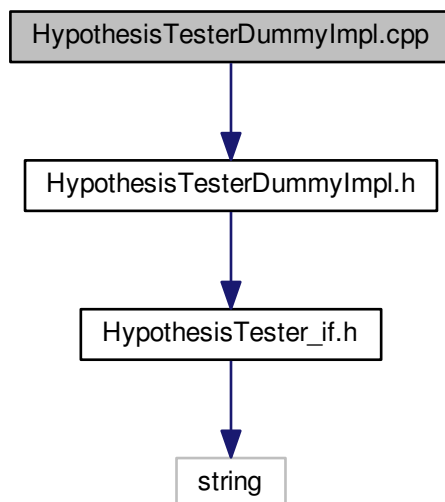


## Classes

- class [GenesysApplication\\_if](#)



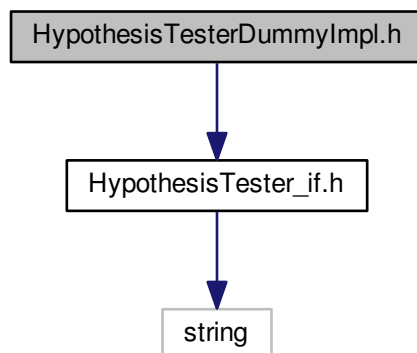
Include dependency graph for HypothesisTesterDummyImpl.cpp:



## 6.48 HypothesisTesterDummyImpl.h File Reference

```
#include "HypothesisTester_if.h"
```

Include dependency graph for HypothesisTesterDummyImpl.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [HypothesisTesterDummyImpl](#)

## 6.49 Integrator\_if.h File Reference

This graph shows which files directly or indirectly include this file:



## Classes

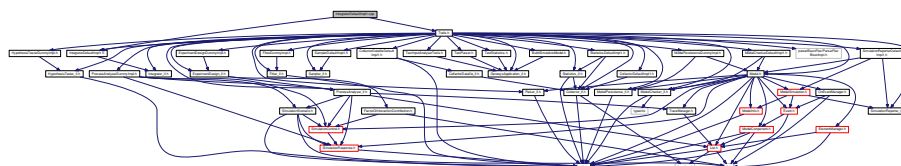
- class [Integrator\\_if](#)

## 6.50 IntegratorDefaultImpl1.cpp File Reference

```
#include "IntegratorDefaultImpl1.h"
```

```
#include "Traits.h"
```

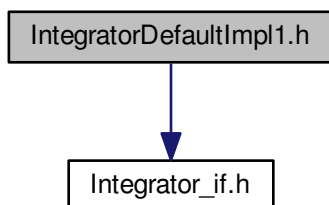
Include dependency graph for IntegratorDefaultImpl1.cpp:



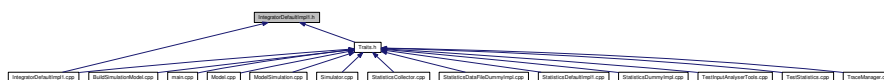
## 6.51 IntegratorDefaultImpl1.h File Reference

```
#include "Integrator_if.h"
```

Include dependency graph for IntegratorDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



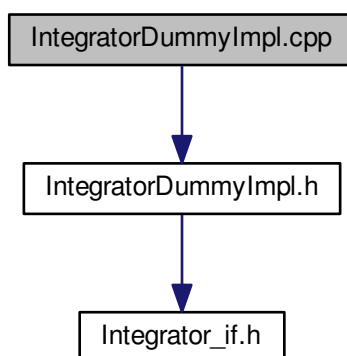
### Classes

- class [IntegratorDefaultImpl1](#)

## 6.52 IntegratorDummyImpl.cpp File Reference

```
#include "IntegratorDummyImpl.h"
```

Include dependency graph for IntegratorDummyImpl.cpp:

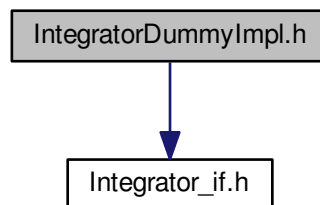




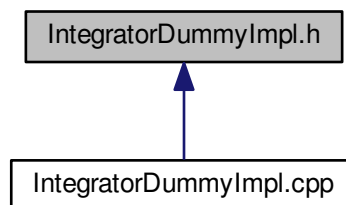
## 6.53 IntegratorDummyImpl.h File Reference

```
#include "Integrator_if.h"
```

Include dependency graph for IntegratorDummyImpl.h:



This graph shows which files directly or indirectly include this file:



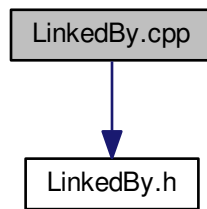
### Classes

- class [IntegratorDummyImpl](#)

## 6.54 LinkedBy.cpp File Reference

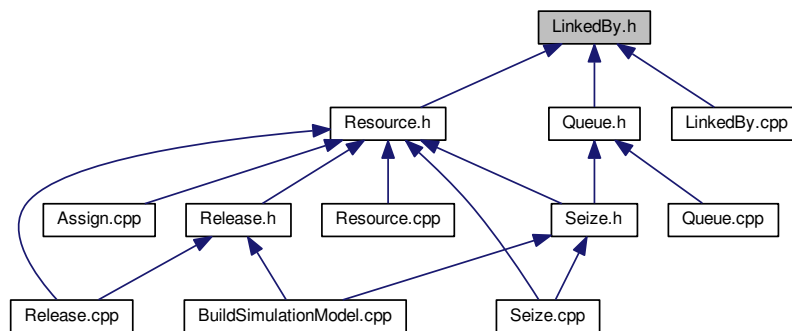
```
#include "LinkedBy.h"
```

Include dependency graph for `LinkedBy.cpp`:



## 6.55 LinkedBy.h File Reference

This graph shows which files directly or indirectly include this file:



### Classes

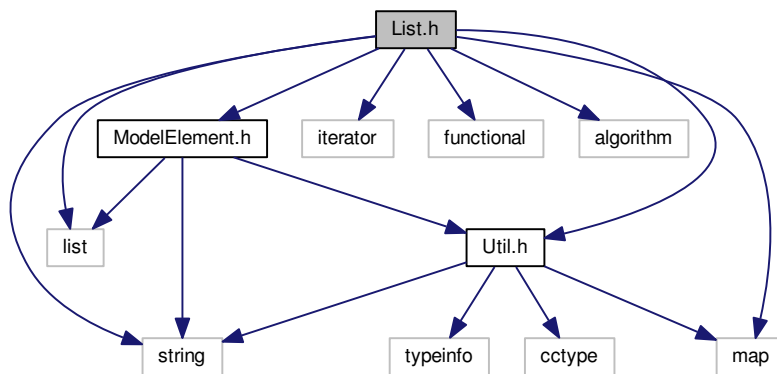
- class [LinkedBy](#)

## 6.56 List.h File Reference

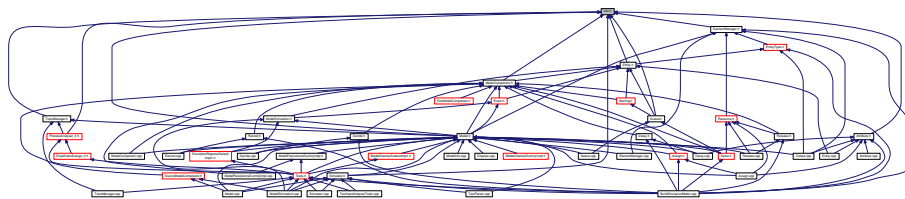
```

#include <string>
#include <list>
#include <map>
#include <iterator>
#include <functional>
#include <algorithm>
#include "Util.h"
#include "ModelElement.h"
  
```

Include dependency graph for List.h:



This graph shows which files directly or indirectly include this file:



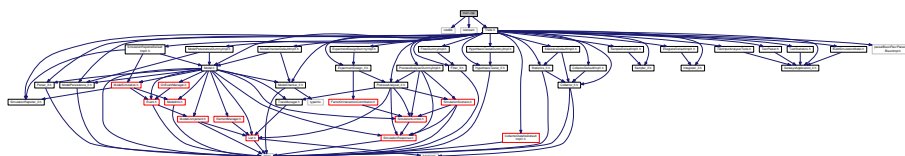
## Classes

- class [List< T >](#)

## 6.57 main.cpp File Reference

```
#include <cstdlib>
#include <iostream>
#include "Traits.h"
```

Include dependency graph for main.cpp:



## Functions

- int [main](#) (int argc, char \*\*argv)

## 6.57.1 Function Documentation

### 6.57.1.1 `int main ( int argc, char ** argv )`

Here is the call graph for this function:



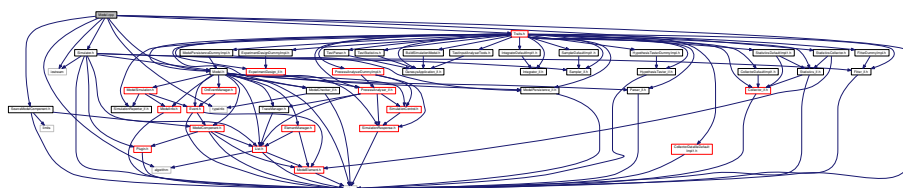
## 6.58 Model.cpp File Reference

```

#include <typeinfo>
#include <iostream>
#include <algorithm>
#include <string>
#include "Model.h"
#include "SourceModelComponent.h"
#include "Simulator.h"
#include "StatisticsCollector.h"
#include "Traits.h"

```

Include dependency graph for Model.cpp:



## Functions

- `bool EventCompare (const Event *a, const Event *b)`
- `double getReplicationLengthNotMemberFunction ()`
- `void setReplicationLengthNotMemberFunction (double value)`

## 6.58.1 Function Documentation

### 6.58.1.1 `bool EventCompare ( const Event * a, const Event * b )`

Here is the call graph for this function:



### 6.58.1.2 `double getReplicationLengthNotMemberFunction ( )`

### 6.58.1.3 `void setReplicationLengthNotMemberFunction ( double value )`

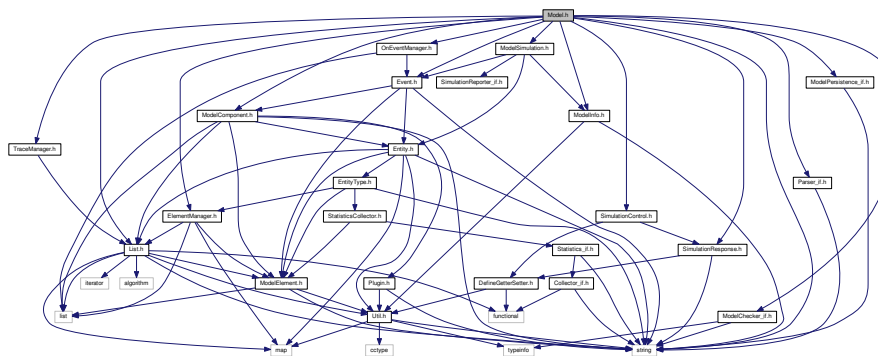
## 6.59 Model.h File Reference

```

#include <string>
#include "List.h"
#include "ModelComponent.h"
#include "Event.h"
#include "ModelChecker_if.h"
#include "Parser_if.h"
#include "ModelPersistence_if.h"
#include "ElementManager.h"
#include "TraceManager.h"
#include "OnEventManager.h"
#include "ModelInfo.h"
#include "ModelSimulation.h"
#include "SimulationResponse.h"
#include "SimulationControl.h"

```

Include dependency graph for Model.h:



This graph shows which files directly or indirectly include this file:

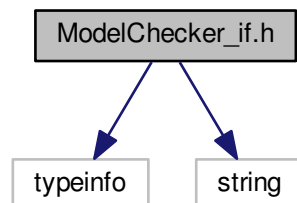


## Classes

- class [Model](#)

## 6.60 ModelChecker\_if.h File Reference

```
#include <typeinfo>
#include <string>
Include dependency graph for ModelChecker_if.h:
```



This graph shows which files directly or indirectly include this file:



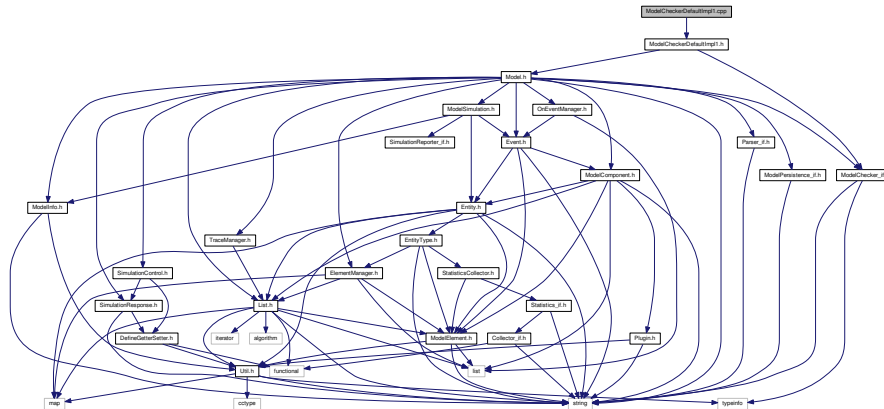
## Classes

- class [ModelChecker\\_if](#)

## 6.61 ModelCheckerDefaultImpl1.cpp File Reference

```
#include "ModelCheckerDefaultImpl1.h"
```

Include dependency graph for ModelCheckerDefaultImpl1.cpp:

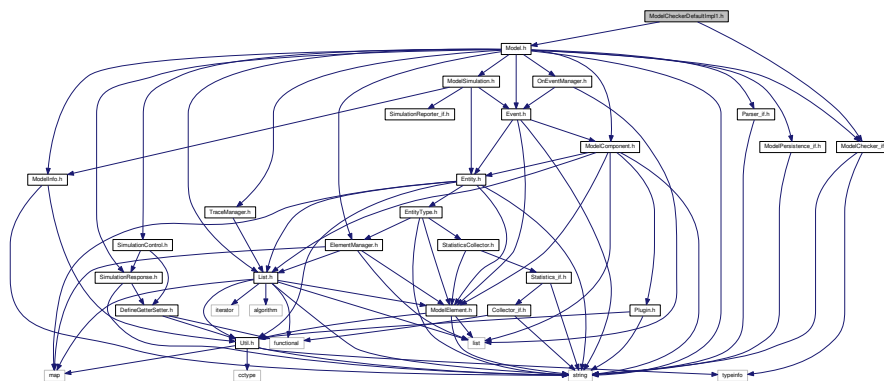


## 6.62 ModelCheckerDefaultImpl1.h File Reference

```
#include "ModelChecker_if.h"
```

```
#include "Model.h"
```

Include dependency graph for ModelCheckerDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:

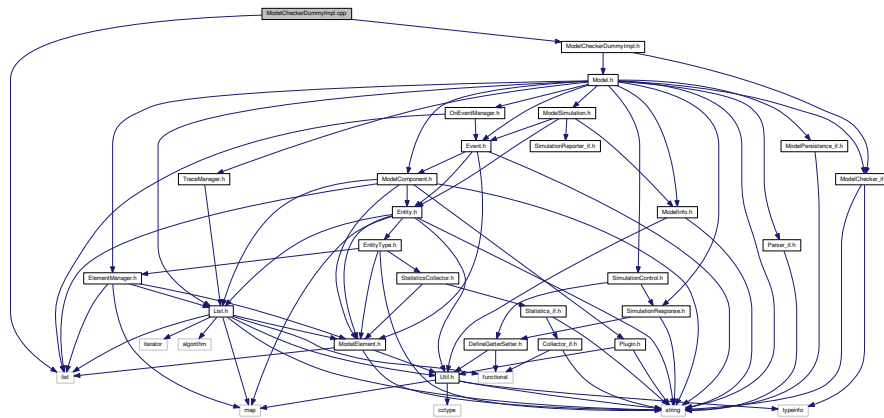


## Classes

- class [ModelCheckerDefaultImpl1](#)

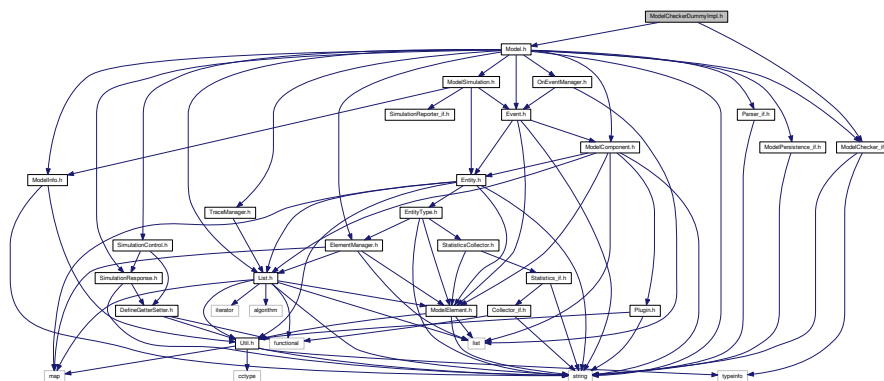
## 6.63 ModelCheckerDummyImpl.cpp File Reference

```
#include <list>
#include "ModelCheckerDummyImpl.h"
Include dependency graph for ModelCheckerDummyImpl.cpp:
```



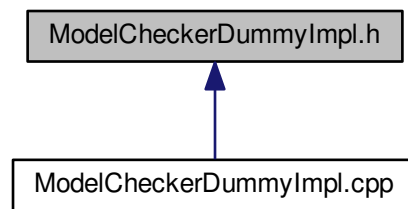
## 6.64 ModelCheckerDummyImpl.h File Reference

```
#include "ModelChecker_if.h"
#include "Model.h"
Include dependency graph for ModelCheckerDummyImpl.h:
```





This graph shows which files directly or indirectly include this file:



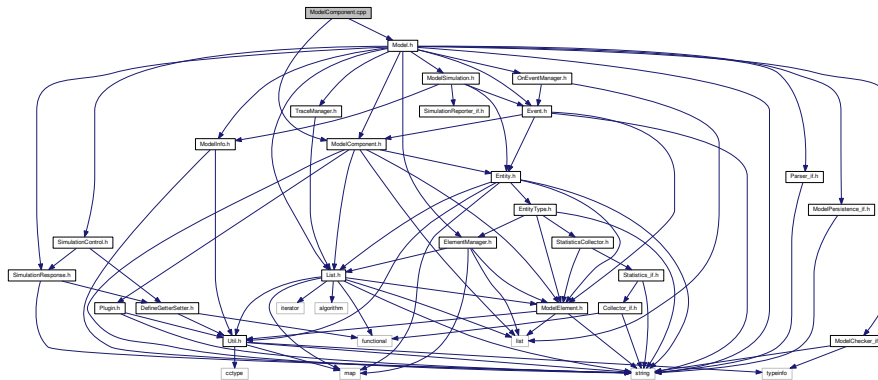
## Classes

- class [ModelCheckerDummyImpl](#)

## 6.65 ModelComponent.cpp File Reference

```
#include "ModelComponent.h"
#include "Model.h"
```

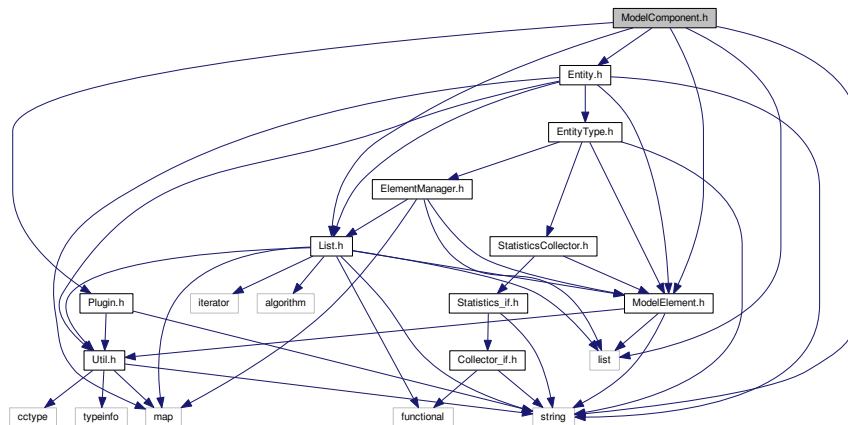
Include dependency graph for ModelComponent.cpp:



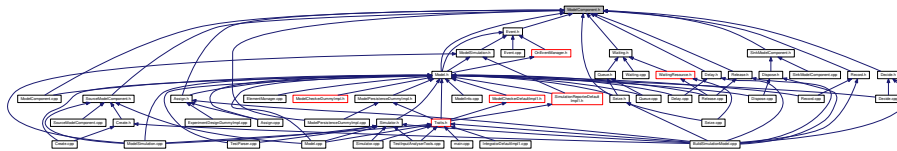
## 6.66 ModelComponent.h File Reference

```
#include <string>
#include <list>
#include "Plugin.h"
#include "List.h"
#include "Entity.h"
#include "ModelElement.h"
```

Include dependency graph for ModelComponent.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [ModelComponent](#)

## 6.67 ModelComponentManager\_if.h File Reference

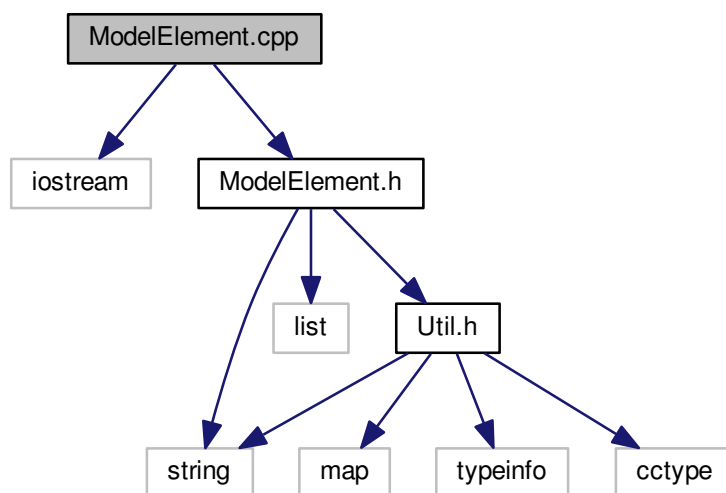
### Classes

- class [ModelComponentManager\\_if](#)

## 6.68 ModelElement.cpp File Reference

```
#include <iostream>
#include "ModelElement.h"
```

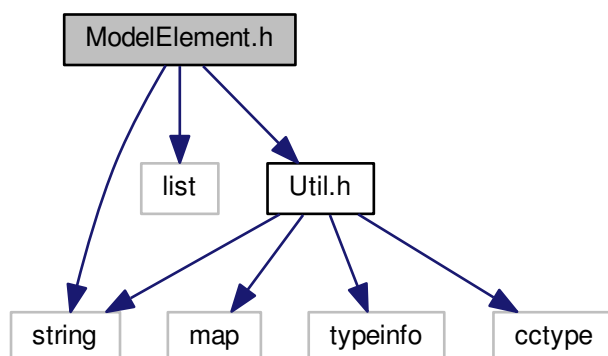
Include dependency graph for ModelElement.cpp:



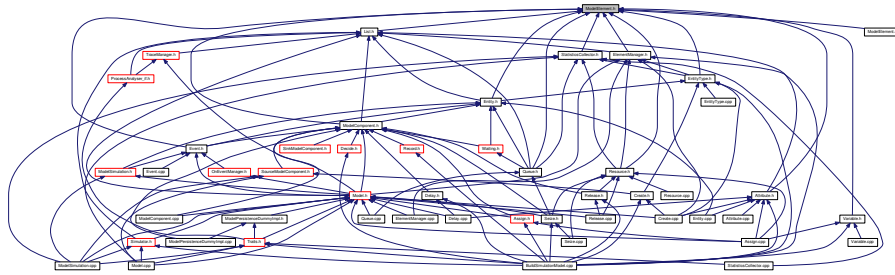
## 6.69 ModelElement.h File Reference

```
#include <string>
#include <list>
#include "Util.h"
```

Include dependency graph for ModelElement.h:



This graph shows which files directly or indirectly include this file:



## Classes

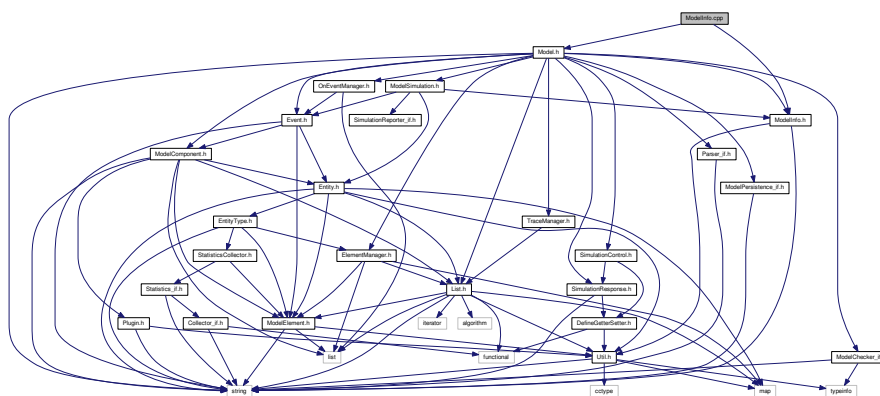
- class [ModelElement](#)

## 6.70 ModelInfo.cpp File Reference

```
#include "ModelInfo.h"
```

```
#include "Model.h"
```

Include dependency graph for ModelInfo.cpp:

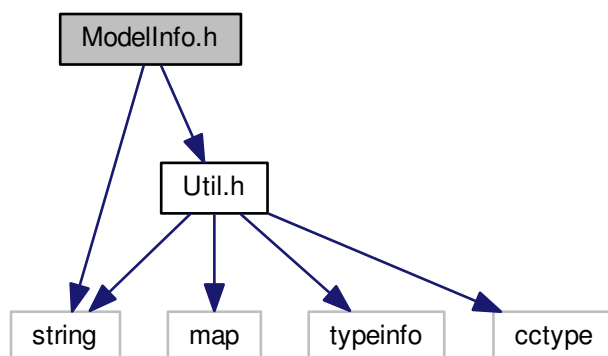


## 6.71 ModelInfo.h File Reference

```
#include <string>
```

```
#include "Util.h"
```

Include dependency graph for ModelInfo.h:



This graph shows which files directly or indirectly include this file:



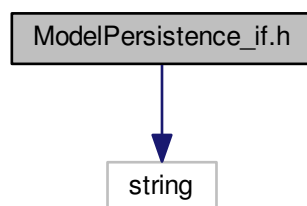
## Classes

- class [ModelInfo](#)

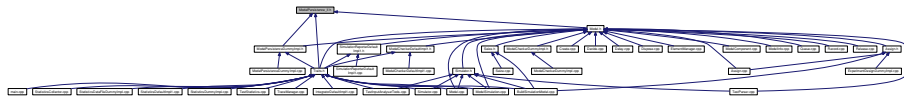
## 6.72 ModelPersistence\_if.h File Reference

```
#include <string>
```

Include dependency graph for ModelPersistence\_if.h:



This graph shows which files directly or indirectly include this file:

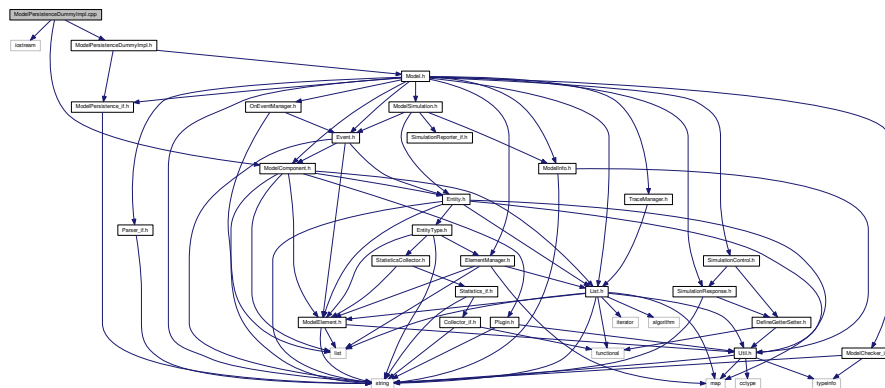


## Classes

- class [ModelPersistence\\_if](#)

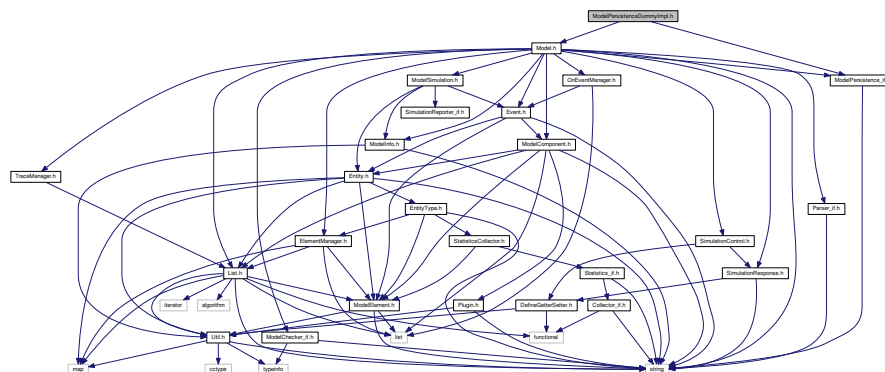
## 6.73 ModelPersistenceDummyImpl.cpp File Reference

```
#include <iostream>
#include "ModelPersistenceDummyImpl.h"
#include "ModelComponent.h"
Include dependency graph for ModelPersistenceDummyImpl.cpp:
```



## 6.74 ModelPersistenceDummyImpl.h File Reference

```
#include "ModelPersistence_if.h"
#include "Model.h"
Include dependency graph for ModelPersistenceDummyImpl.h:
```



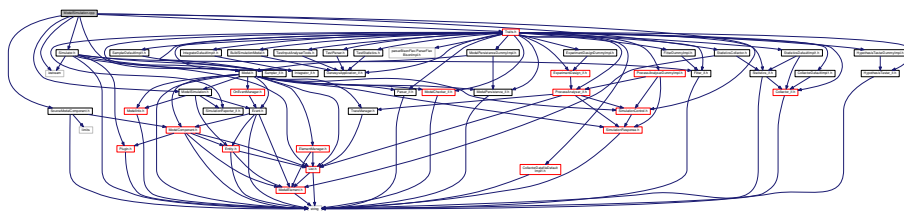
```

graph BT
    Stats_h[Stats.h] --> StatsResourceCummyImpl_h[StatsResourceCummyImpl.h]
    Stats_h --> Stats_h
    StatsResourceCummyImpl_h --> BuildEstimatorModel_cpp[BuildEstimatorModel.cpp]
    StatsResourceCummyImpl_h --> InferenceDefaultImpl_cpp[InferenceDefaultImpl.cpp]
    StatsResourceCummyImpl_h --> main_cpp[main.cpp]
    StatsResourceCummyImpl_h --> Model_cpp[Model.cpp]
    StatsResourceCummyImpl_h --> ModelFormation_cpp[ModelFormation.cpp]
    StatsResourceCummyImpl_h --> Simulator_cpp[Simulator.cpp]
    StatsResourceCummyImpl_h --> StatisticsCollector_cpp[StatisticsCollector.cpp]
    StatsResourceCummyImpl_h --> StatisticsCudaCummyImpl_cpp[StatisticsCudaCummyImpl.cpp]
    StatsResourceCummyImpl_h --> StatisticsDefaultImpl_cpp[StatisticsDefaultImpl.cpp]
    StatsResourceCummyImpl_h --> StatisticsCummyImpl_cpp[StatisticsCummyImpl.cpp]
    StatsResourceCummyImpl_h --> TestGeneratorTest_cpp[TestGeneratorTest.cpp]
    StatsResourceCummyImpl_h --> TestStatistics_cpp[TestStatistics.cpp]
    StatsResourceCummyImpl_h --> TestTrainer_cpp[TestTrainer.cpp]

```

- class `ModelPersistenceDummyImpl`

```
#include <iostream>
#include "ModelSimulation.h"
#include "Model.h"
#include "Simulator.h"
#include "SourceModelComponent.h"
#include "StatisticsCollector.h"
#include "Traits.h"
#include "SimulationControl.h"
Include dependency graph for ModelSimulation.cpp:
```

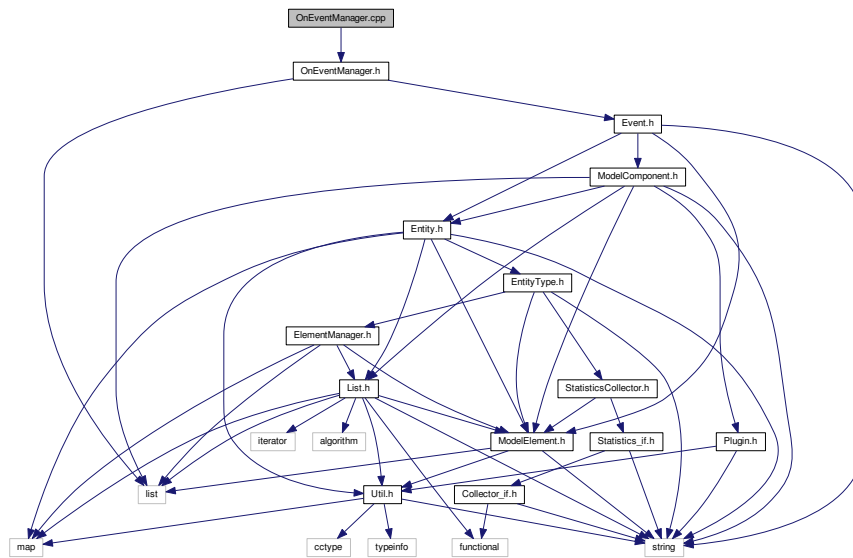


```
#include "Event.h"
#include "Entity.h"
#include "ModelInfo.h"
#include "SimulationReporter_if.h"
```





Include dependency graph for OnEventManager.cpp:

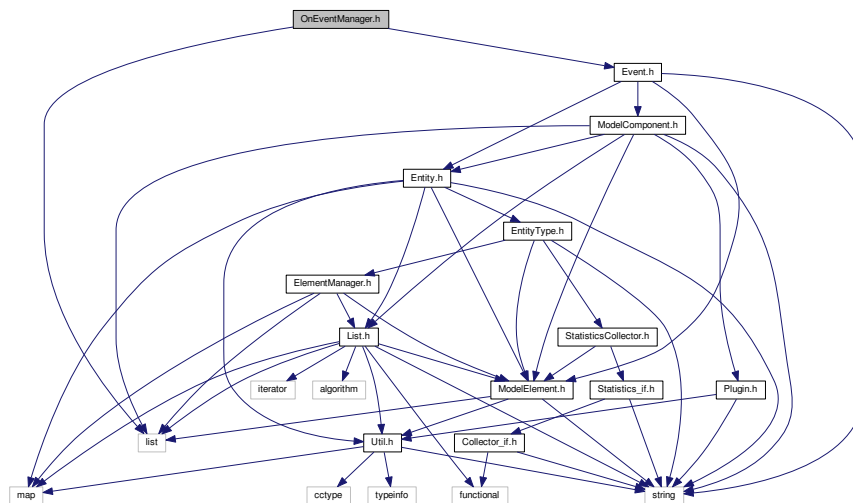


## 6.78 OnEventManager.h File Reference

```
#include <list>
```

```
#include "Event.h"
```

Include dependency graph for OnEventManager.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [SimulationEvent](#)
- class [OnEventManager](#)

## Typedefs

- typedef void(\* [simulationEventHandler](#)) ([SimulationEvent](#) \*)

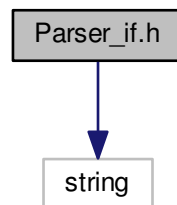
### 6.78.1 Typedef Documentation

#### 6.78.1.1 typedef void(\* simulationEventHandler) (SimulationEvent \*)

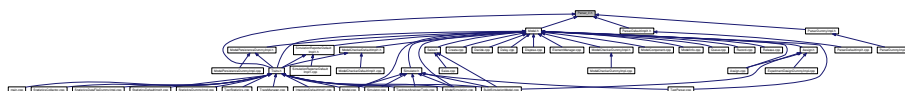
## 6.79 Parser\_if.h File Reference

```
#include <string>
```

Include dependency graph for Parser\_if.h:



This graph shows which files directly or indirectly include this file:



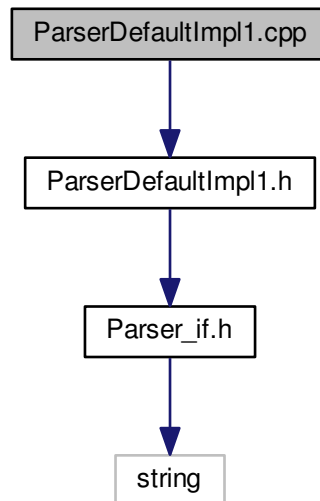
## Classes

- class [Parser\\_if](#)

## 6.80 ParserDefaultImpl1.cpp File Reference

```
#include "ParserDefaultImpl1.h"
```

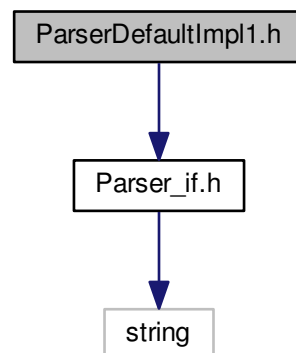
Include dependency graph for ParserDefaultImpl1.cpp:



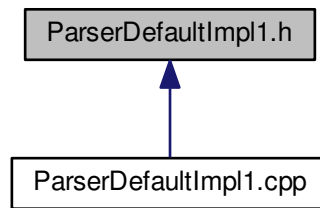
## 6.81 ParserDefaultImpl1.h File Reference

```
#include "Parser_if.h"
```

Include dependency graph for ParserDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



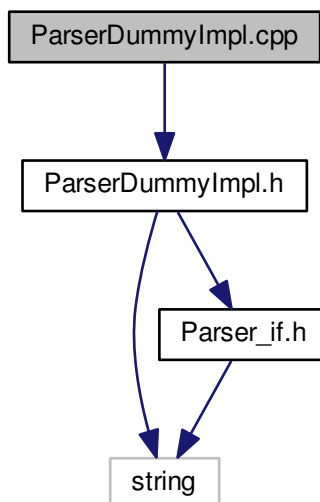
## Classes

- class [ParserDefaultImpl1](#)

## 6.82 ParserDummyImpl.cpp File Reference

```
#include "ParserDummyImpl.h"
```

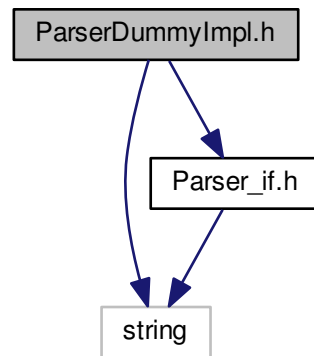
Include dependency graph for `ParserDummyImpl.cpp`:



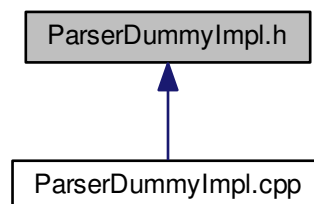
## 6.83 ParserDummyImpl.h File Reference

```
#include <string>
#include "Parser_if.h"
```

Include dependency graph for ParserDummyImpl.h:



This graph shows which files directly or indirectly include this file:



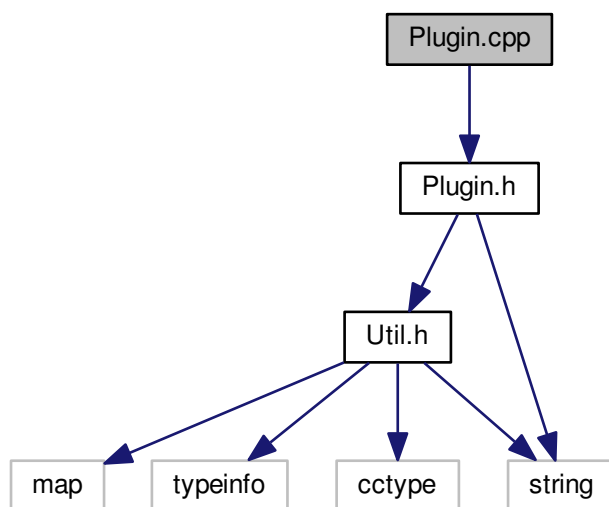
### Classes

- class [ParserDummyImpl](#)

## 6.84 Plugin.cpp File Reference

```
#include "Plugin.h"
```

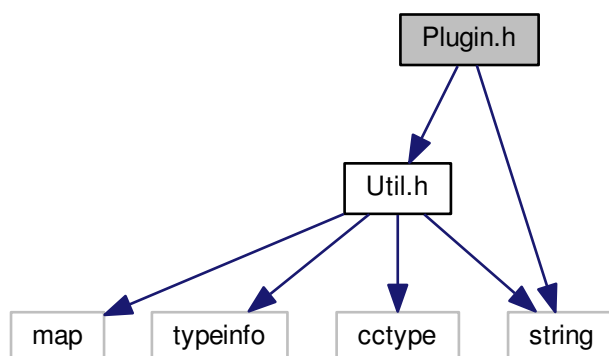
Include dependency graph for Plugin.cpp:



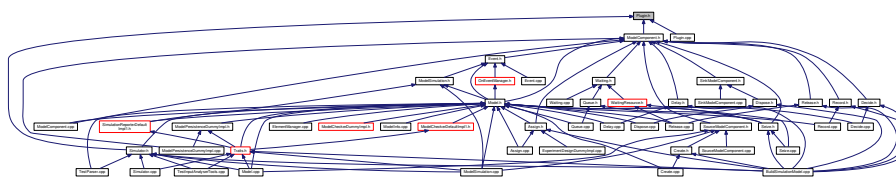
## 6.85 Plugin.h File Reference

```
#include "Util.h"
#include <string>
```

Include dependency graph for Plugin.h:



This graph shows which files directly or indirectly include this file:



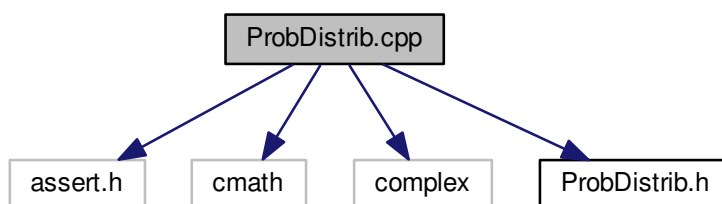
## Classes

- class [Plugin](#)

## 6.86 ProbDistrib.cpp File Reference

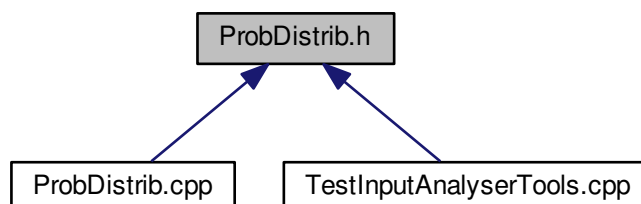
```
#include <assert.h>
#include <cmath>
#include <complex>
#include "ProbDistrib.h"
```

Include dependency graph for ProbDistrib.cpp:



## 6.87 ProbDistrib.h File Reference

This graph shows which files directly or indirectly include this file:



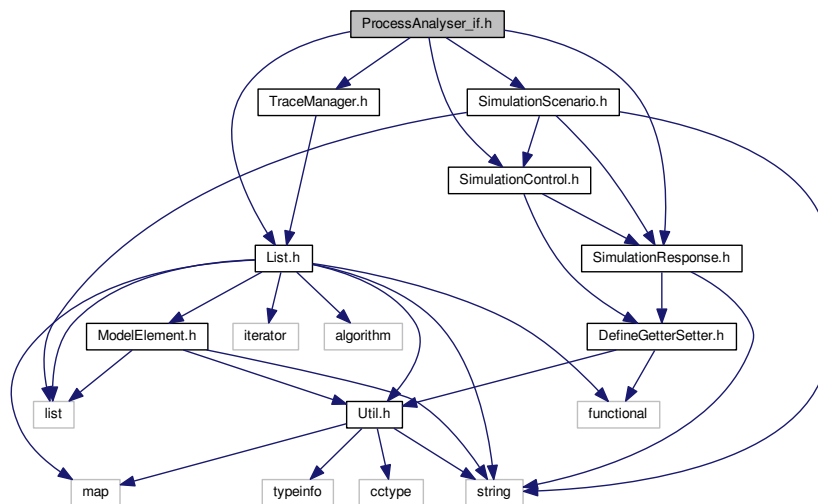
## Classes

- class [ProbDistrib](#)

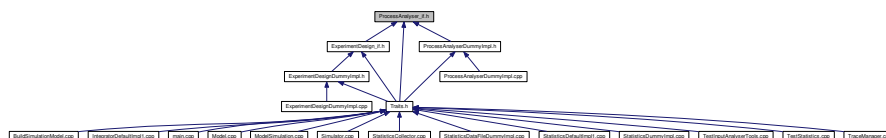
## 6.88 ProcessAnalyser\_if.h File Reference

```
#include "List.h"
#include "SimulationScenario.h"
#include "SimulationControl.h"
#include "SimulationResponse.h"
#include "TraceManager.h"
```

Include dependency graph for ProcessAnalyser\_if.h:



This graph shows which files directly or indirectly include this file:



## Classes

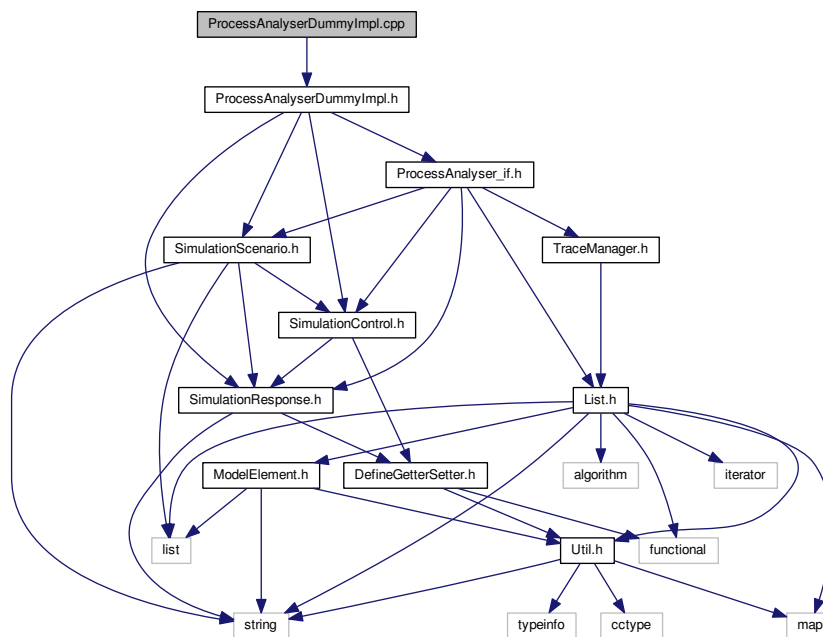
- class [ProcessAnalyser\\_if](#)



## 6.89 ProcessAnalyserDummyImpl.cpp File Reference

```
#include "ProcessAnalyserDummyImpl.h"
```

Include dependency graph for ProcessAnalyserDummyImpl.cpp:



## 6.90 ProcessAnalyserDummyImpl.h File Reference

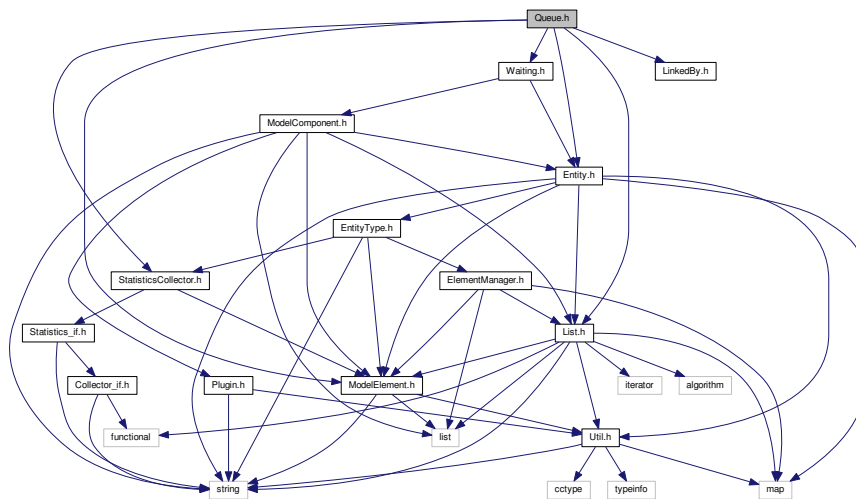
```
#include "ProcessAnalyser_if.h"
#include "SimulationScenario.h"
#include "SimulationResponse.h"
#include "SimulationControl.h"
```



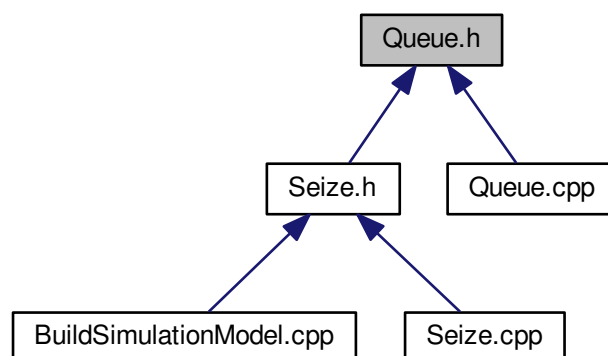
## 6.92 Queue.h File Reference

```
#include "ModelElement.h"
#include "LinkedBy.h"
#include "List.h"
#include "Entity.h"
#include "Waiting.h"
#include "StatisticsCollector.h"
```

Include dependency graph for Queue.h:



This graph shows which files directly or indirectly include this file:



### Classes

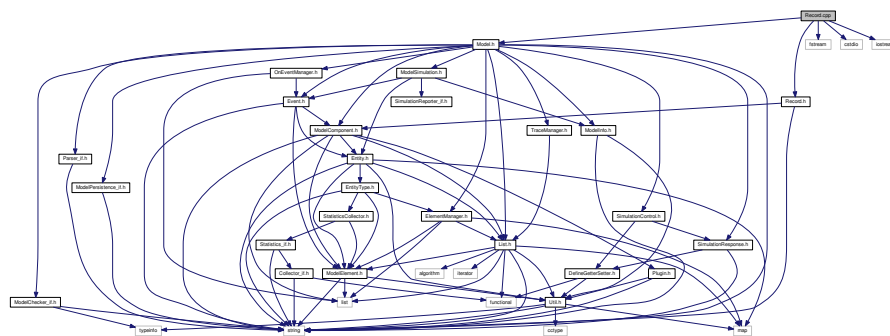
- class [Queue](#)

## 6.93 README.md File Reference

## 6.94 Record.cpp File Reference

```
#include "Record.h"
#include "Model.h"
#include <fstream>
#include <cstdio>
#include <iostream>
```

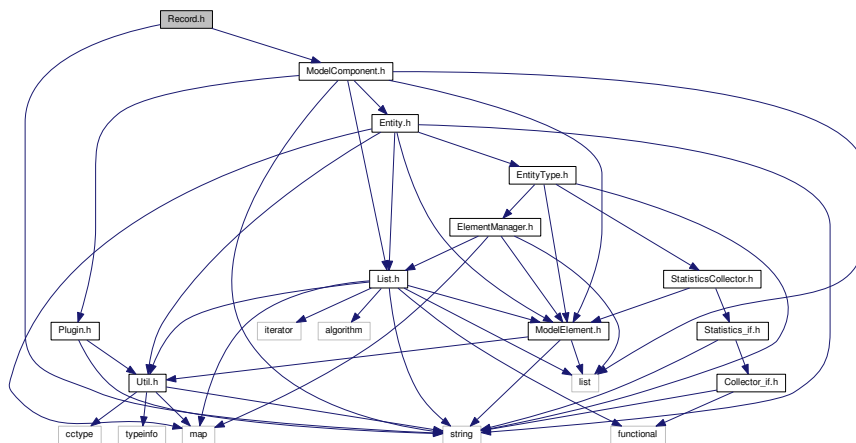
Include dependency graph for Record.cpp:



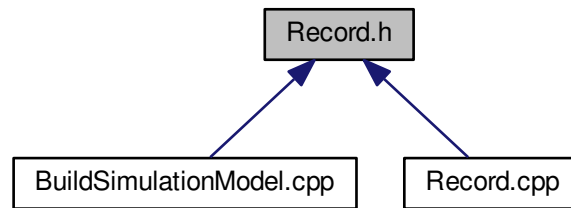
## 6.95 Record.h File Reference

```
#include "ModelComponent.h"
#include <string>
```

Include dependency graph for Record.h:



This graph shows which files directly or indirectly include this file:

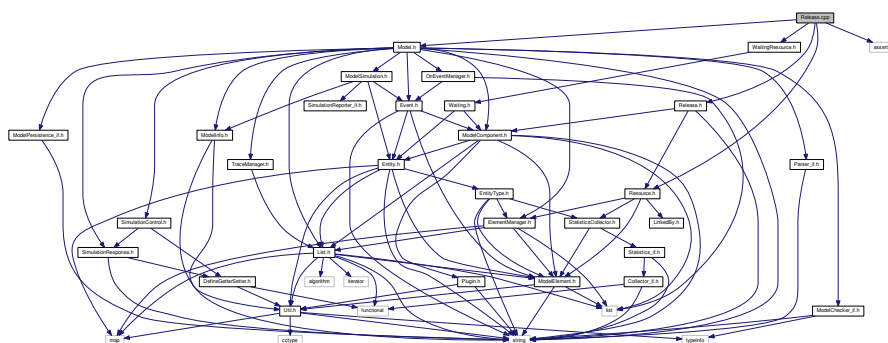


## Classes

- class [Record](#)

## 6.96 Release.cpp File Reference

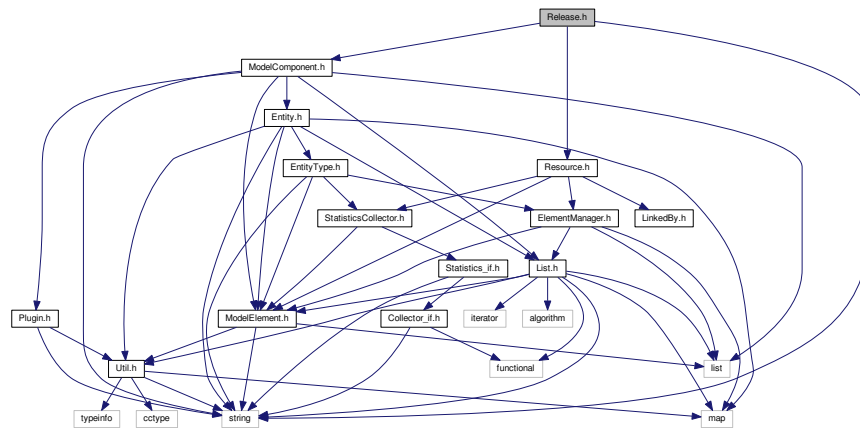
```
#include "Release.h"
#include "Model.h"
#include "WaitingResource.h"
#include "Resource.h"
#include <assert.h>
Include dependency graph for Release.cpp:
```



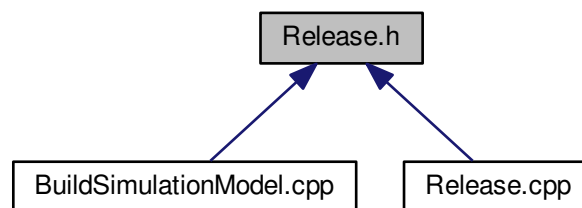
## 6.97 Release.h File Reference

```
#include <string>
#include "ModelComponent.h"
#include "Resource.h"
```

Include dependency graph for Release.h:



This graph shows which files directly or indirectly include this file:



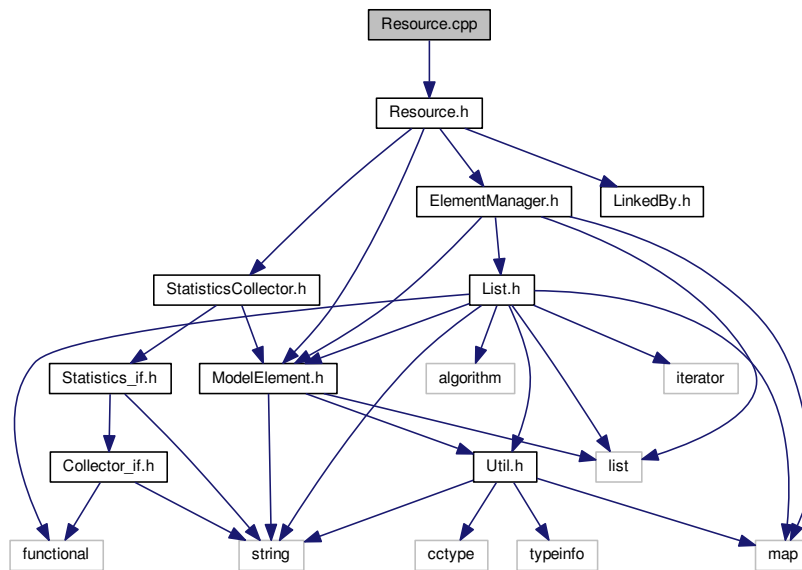
## Classes

- class [Release](#)

## 6.98 Resource.cpp File Reference

```
#include "Resource.h"
```

Include dependency graph for Resource.cpp:



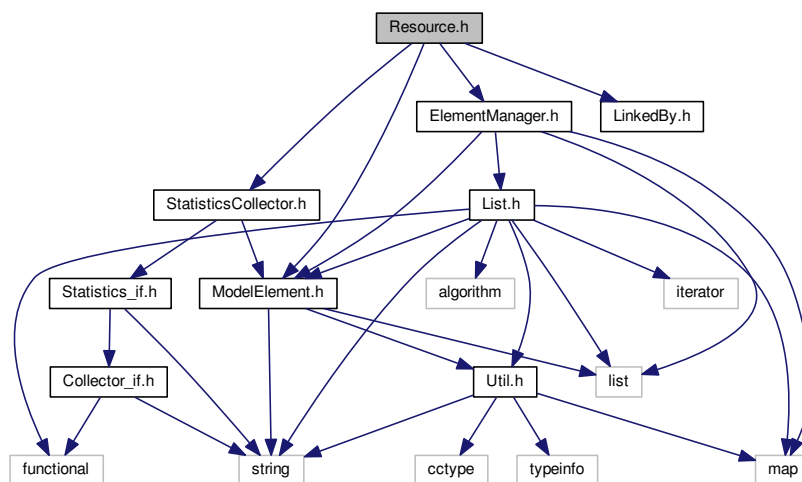
## 6.99 Resource.h File Reference

```

#include "ModelElement.h"
#include "LinkedBy.h"
#include "StatisticsCollector.h"
#include "ElementManager.h"

```

Include dependency graph for Resource.h:





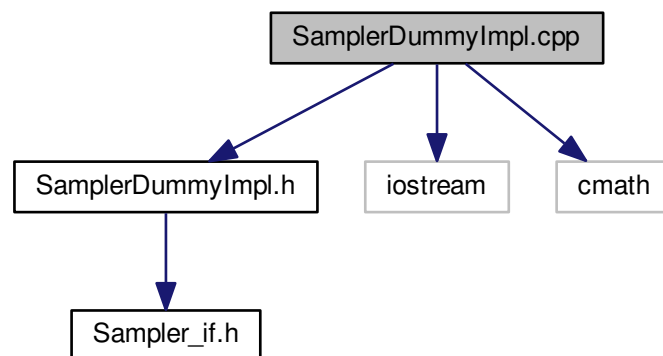




### 6.103 SamplerDummyImpl.cpp File Reference

```
#include "SamplerDummyImpl.h"  
#include <iostream>  
#include <cmath>
```

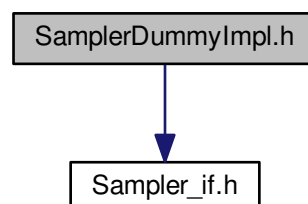
Include dependency graph for SamplerDummyImpl.cpp:



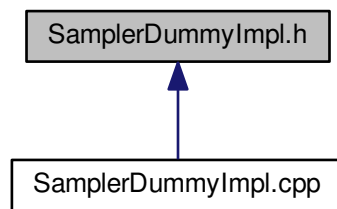
### 6.104 SamplerDummyImpl.h File Reference

```
#include "Sampler_if.h"
```

Include dependency graph for SamplerDummyImpl.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [SamplerDummyImpl](#)
- class [SamplerDummyImpl::MyRNG\\_Parameters](#)

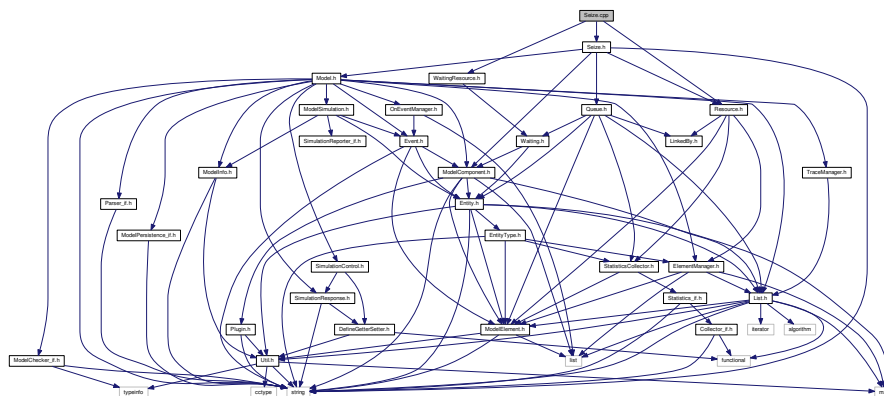
## 6.105 ScenarioExperiment\_if.h File Reference

### Classes

- class [ScenarioExperiment\\_if](#)

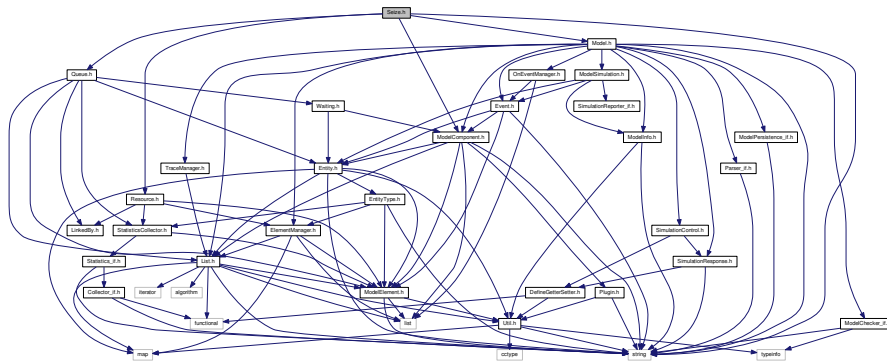
## 6.106 Seize.cpp File Reference

```
#include "Seize.h"
#include "WaitingResource.h"
#include "Resource.h"
Include dependency graph for Seize.cpp:
```

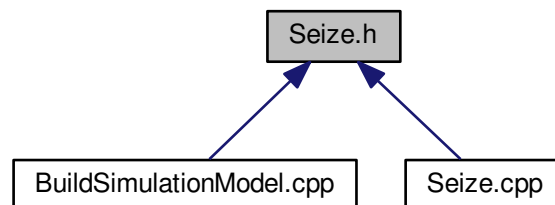


## 6.107 Seize.h File Reference

```
#include <string>
#include "ModelComponent.h"
#include "Model.h"
#include "Resource.h"
#include "Queue.h"
Include dependency graph for Seize.h:
```



This graph shows which files directly or indirectly include this file:



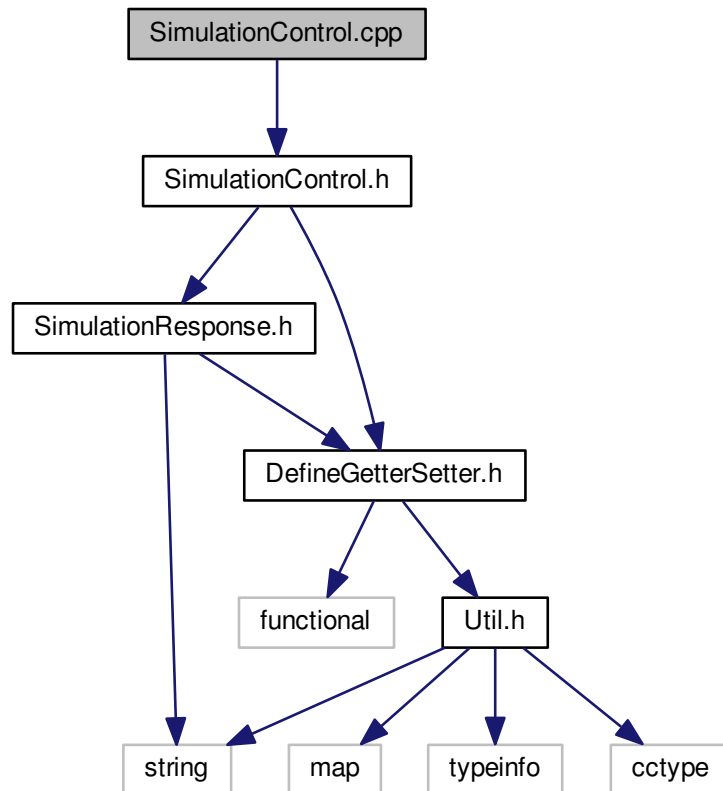
### Classes

- class [Seize](#)

## 6.108 SimulationControl.cpp File Reference

```
#include "SimulationControl.h"
```

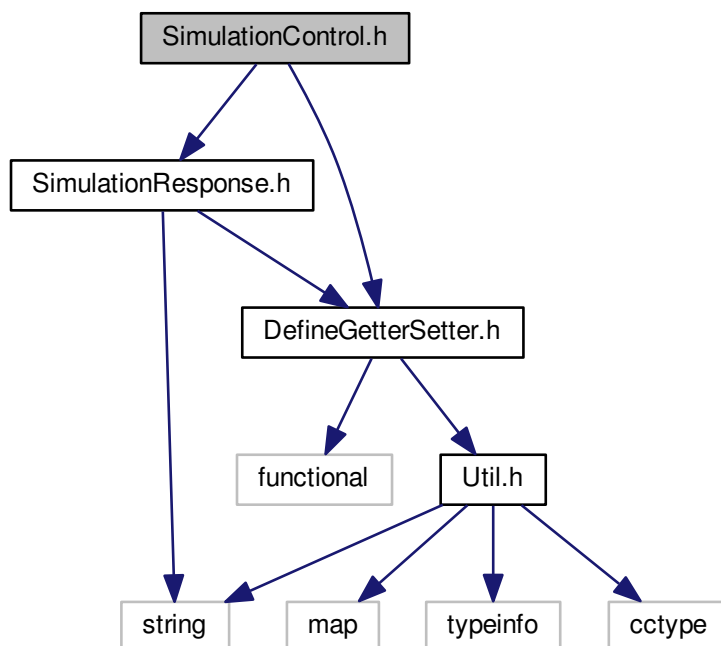
Include dependency graph for SimulationControl.cpp:



## 6.109 SimulationControl.h File Reference

```
#include "SimulationResponse.h"
#include "DefineGetterSetter.h"
```

Include dependency graph for SimulationControl.h:



This graph shows which files directly or indirectly include this file:

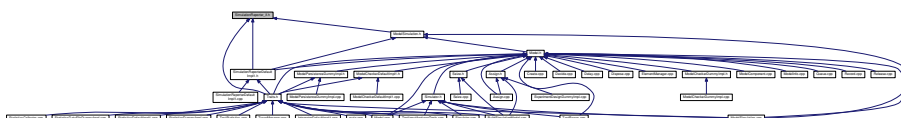


## Classes

- class [SimulationControl](#)

## 6.110 SimulationReporter\_if.h File Reference

This graph shows which files directly or indirectly include this file:



## Classes

- class [SimulationReporter\\_if](#)

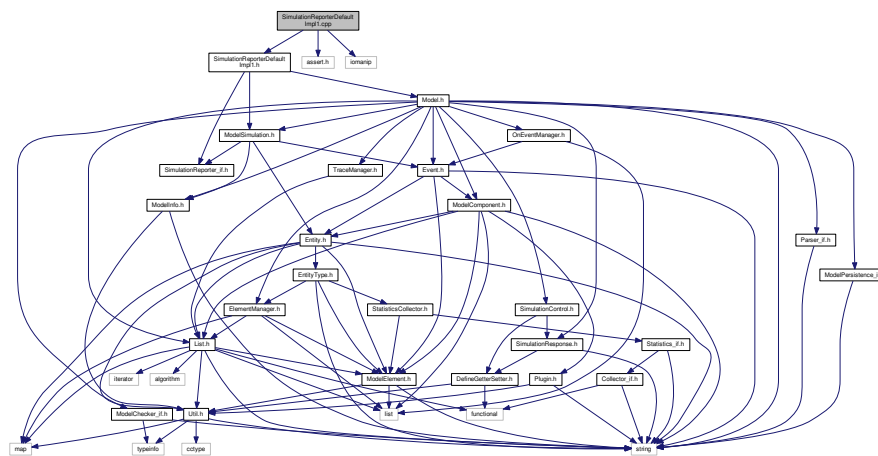
## 6.111 SimulationReporterDefaultImpl1.cpp File Reference

```
#include "SimulationReporterDefaultImpl1.h"
```

```
#include <assert.h>
```

```
#include <iomanip>
```

Include dependency graph for SimulationReporterDefaultImpl1.cpp:



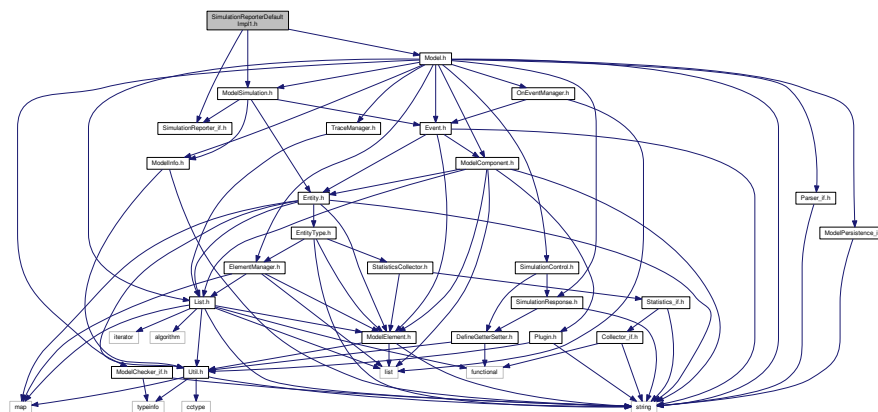
## 6.112 SimulationReporterDefaultImpl1.h File Reference

```
#include "SimulationReporter_if.h"
```

```
#include "ModelSimulation.h"
```

```
#include "Model.h"
```

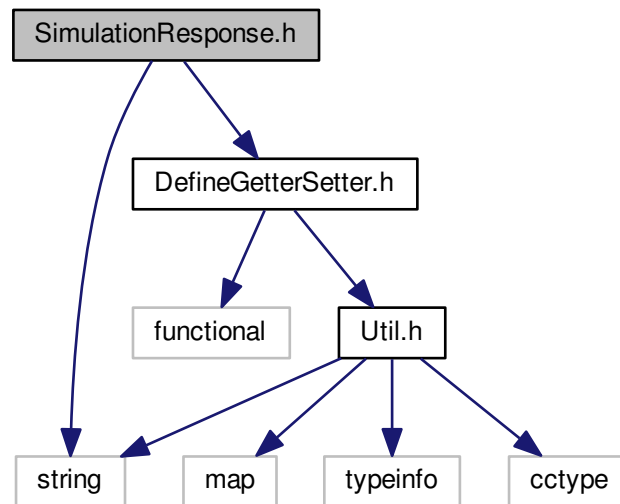
Include dependency graph for SimulationReporterDefaultImpl1.h:



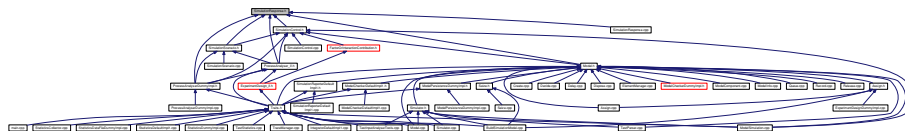




Include dependency graph for SimulationResponse.h:



This graph shows which files directly or indirectly include this file:



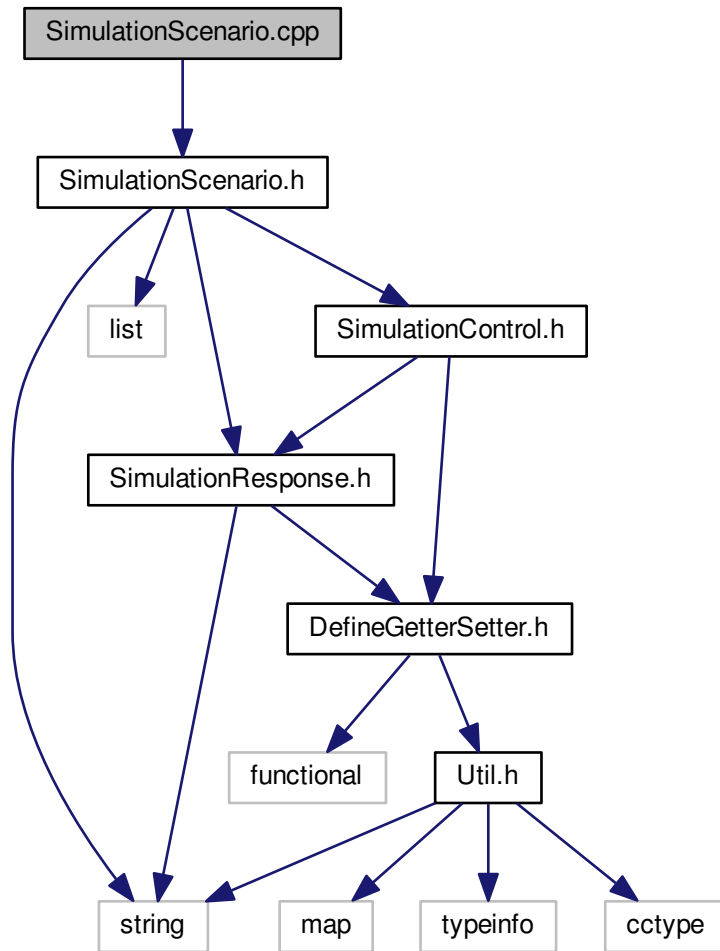
## Classes

- class [SimulationResponse](#)

## 6.115 SimulationScenario.cpp File Reference

```
#include "SimulationScenario.h"
```

Include dependency graph for SimulationScenario.cpp:



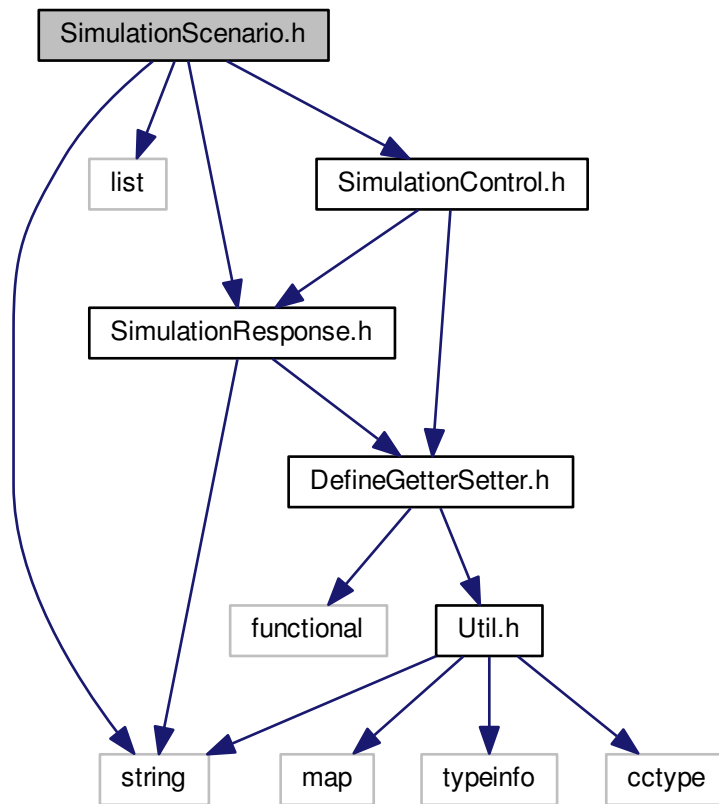
## 6.116 SimulationScenario.h File Reference

```

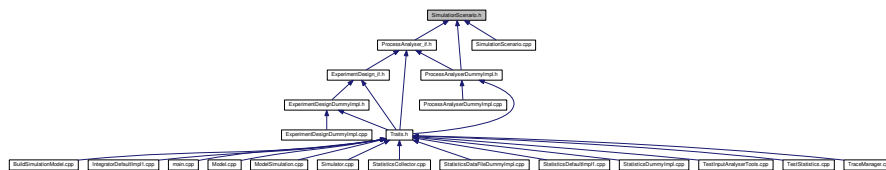
#include <string>
#include <list>
#include "SimulationResponse.h"
#include "SimulationControl.h"

```

Include dependency graph for SimulationScenario.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [SimulationScenario](#)

## 6.117 Simulator.cpp File Reference

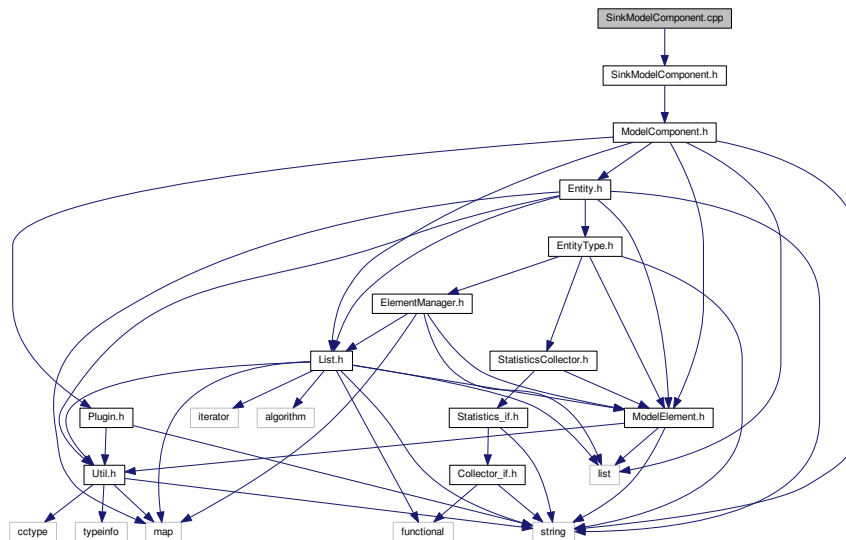
```
#include "Simulator.h"
```



## 6.119 SinkModelComponent.cpp File Reference

```
#include "SinkModelComponent.h"
```

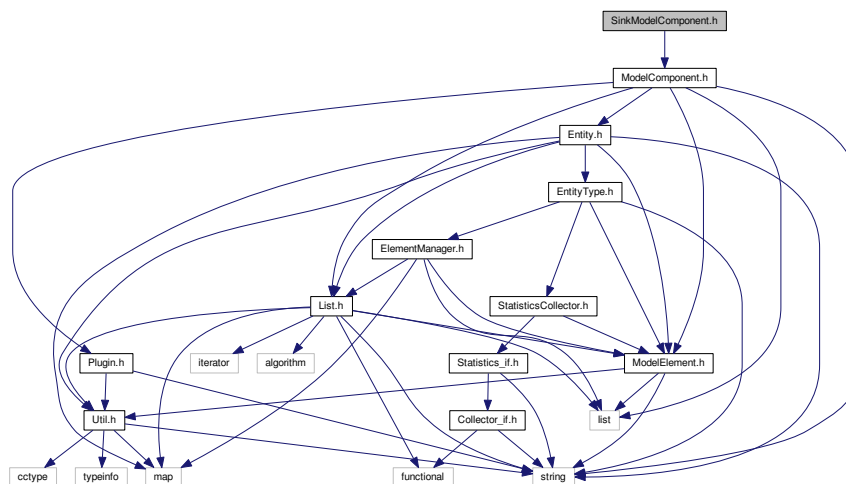
Include dependency graph for SinkModelComponent.cpp:



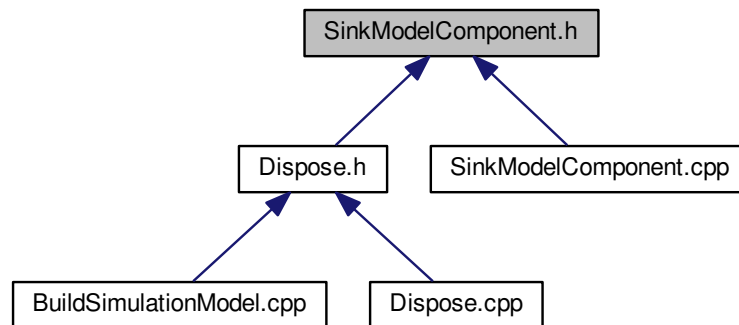
## 6.120 SinkModelComponent.h File Reference

```
#include "ModelComponent.h"
```

Include dependency graph for SinkModelComponent.h:



This graph shows which files directly or indirectly include this file:



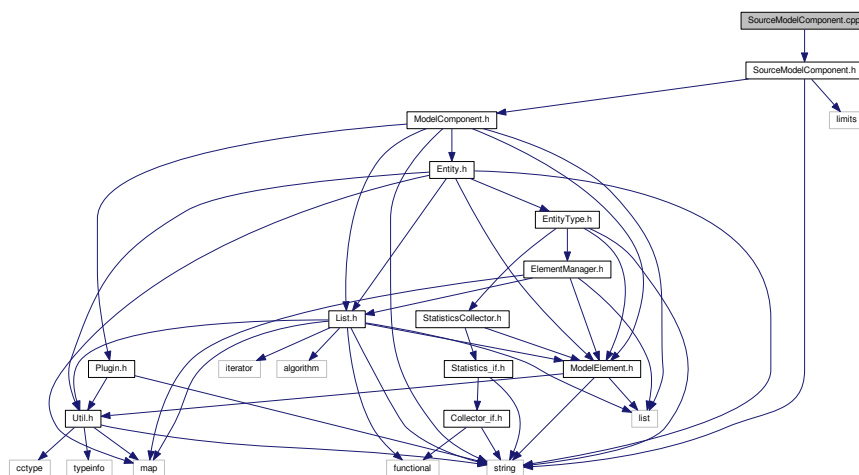
## Classes

- class [SinkModelComponent](#)

## 6.121 SourceModelComponent.cpp File Reference

```
#include "SourceModelComponent.h"
```

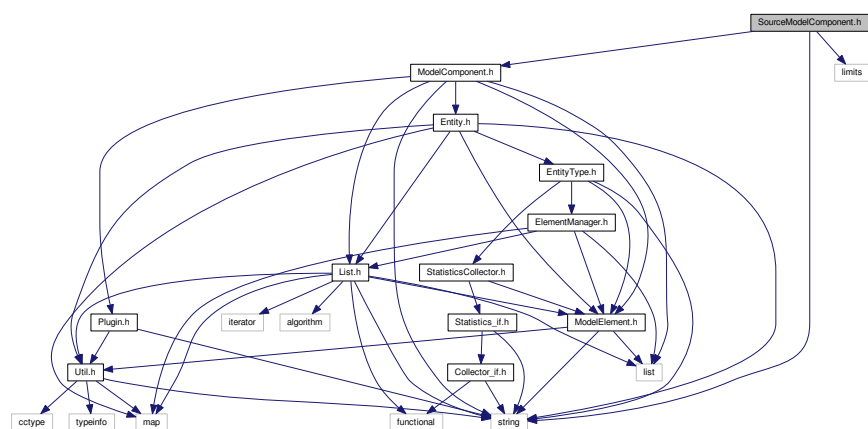
Include dependency graph for `SourceModelComponent.cpp`:



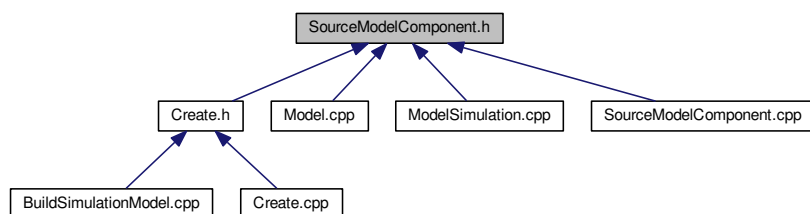
## 6.122 SourceModelComponent.h File Reference

```
#include "ModelComponent.h"
```

```
#include <string>
#include <limits>
Include dependency graph for SourceModelComponent.h:
```



This graph shows which files directly or indirectly include this file:



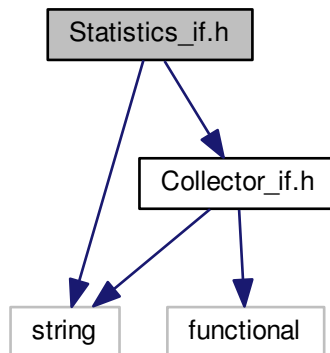
## Classes

- class **SourceModelComponent**

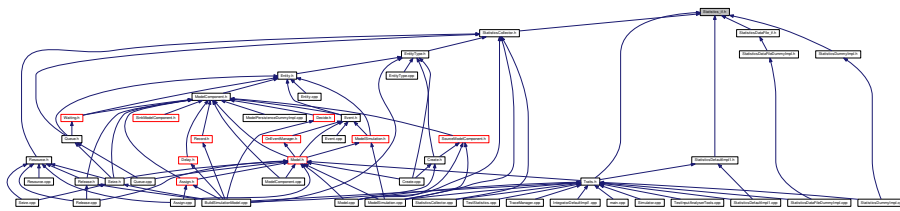
### 6.123 Statistics\_if.h File Reference

```
#include <string>
#include "Collector_if.h"
```

Include dependency graph for `Statistics_if.h`:



This graph shows which files directly or indirectly include this file:



## Classes

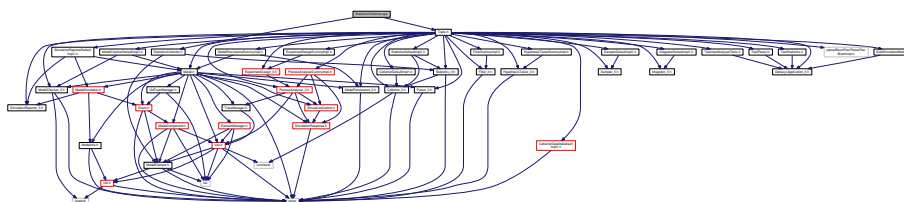
- class [Statistics\\_if](#)

## 6.124 StatisticsCollector.cpp File Reference

```
#include "StatisticsCollector.h"
```

```
#include "Traits.h"
```

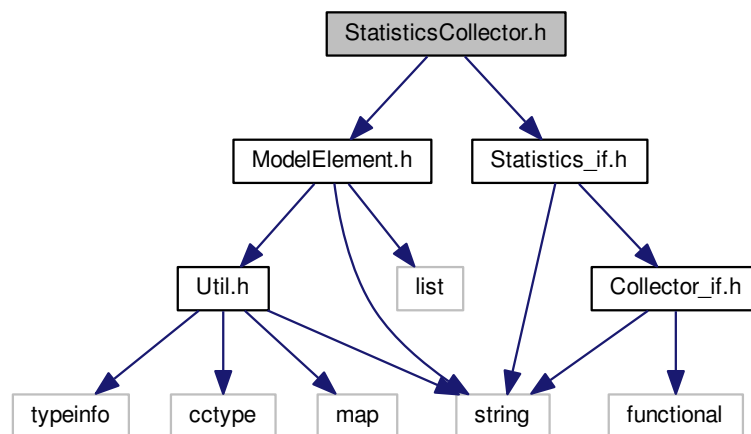
Include dependency graph for `StatisticsCollector.cpp`:



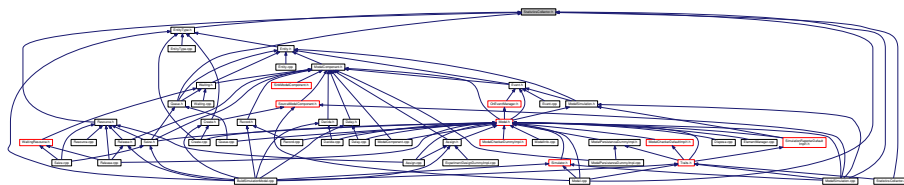


## 6.125 StatisticsCollector.h File Reference

```
#include "ModelElement.h"  
#include "Statistics_if.h"  
Include dependency graph for StatisticsCollector.h:
```



This graph shows which files directly or indirectly include this file:



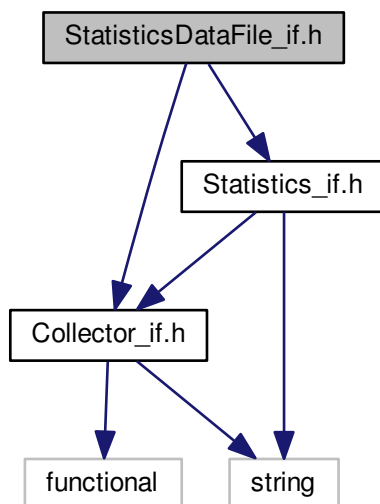
### Classes

- class [StatisticsCollector](#)

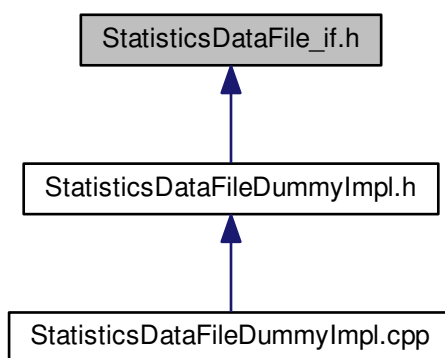
## 6.126 StatisticsDataFile\_if.h File Reference

```
#include "Collector_if.h"  
#include "Statistics_if.h"
```

Include dependency graph for StatisticsDataFile\_if.h:



This graph shows which files directly or indirectly include this file:



## Classes

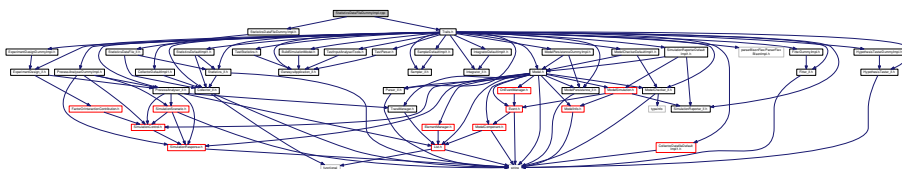
- class [StatisticsDatafile\\_if](#)

## 6.127 StatisticsDataFileDummyImpl.cpp File Reference

```
#include "StatisticsDataFileDummyImpl.h"
```

```
#include "Traits.h"
```

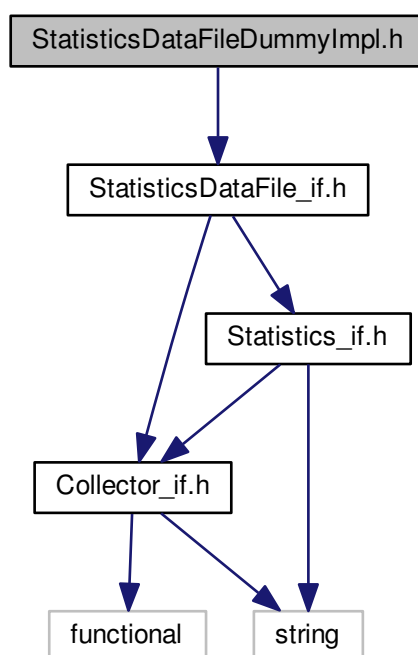
Include dependency graph for StatisticsDataFileDummyImpl.cpp:



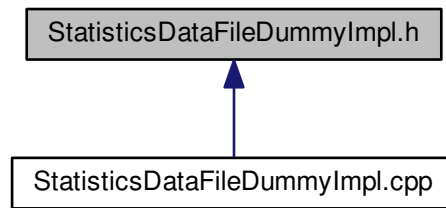
## 6.128 StatisticsDataFileDummyImpl.h File Reference

```
#include "StatisticsDataFile_if.h"
```

Include dependency graph for StatisticsDataFileDummyImpl.h:



This graph shows which files directly or indirectly include this file:



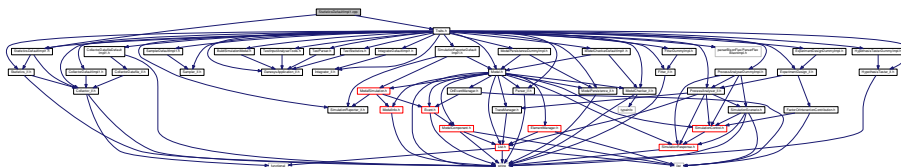
## Classes

- class [StatisticsDataFileDummyImpl](#)

## 6.129 StatisticsDefaultImpl1.cpp File Reference

```
#include "StatisticsDefaultImpl1.h"
#include "Traits.h"
```

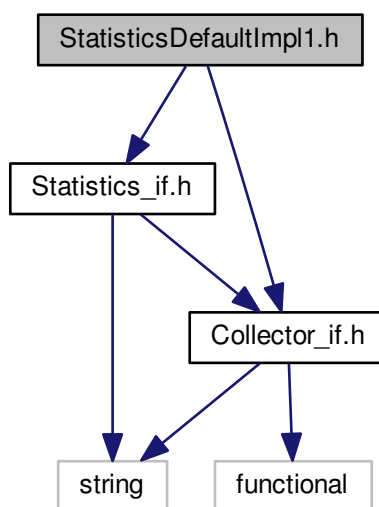
Include dependency graph for StatisticsDefaultImpl1.cpp:



## 6.130 StatisticsDefaultImpl1.h File Reference

```
#include "Statistics_if.h"
#include "Collector_if.h"
```

Include dependency graph for StatisticsDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:

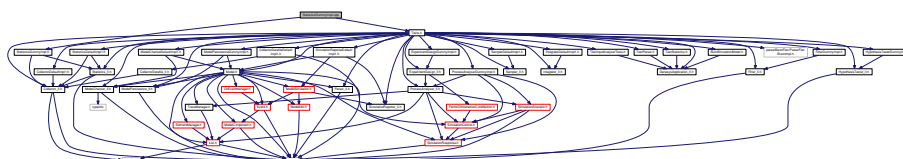


## Classes

- class `StatisticsDefaultImpl1`

## 6.131 StatisticsDummyImpl.cpp File Reference

```
#include "StatisticsDummyImpl.h"
#include "Traits.h"
Include dependency graph for StatisticsDummyImpl.cpp:
```

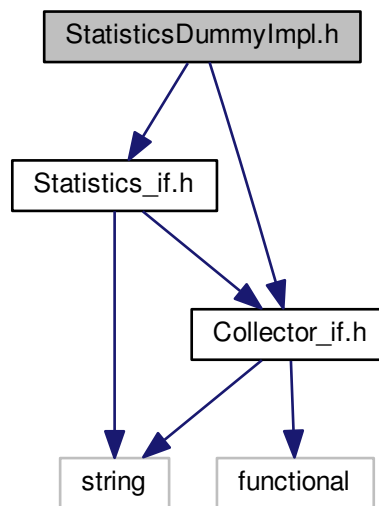


## 6.132 StatisticsDummyImpl.h File Reference

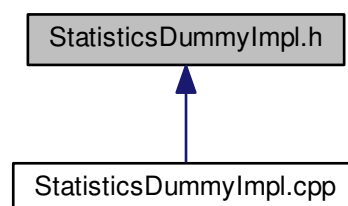
```
#include "Statistics_if.h"
```

```
#include "Collector_if.h"
```

Include dependency graph for StatisticsDummyImpl.h:



This graph shows which files directly or indirectly include this file:



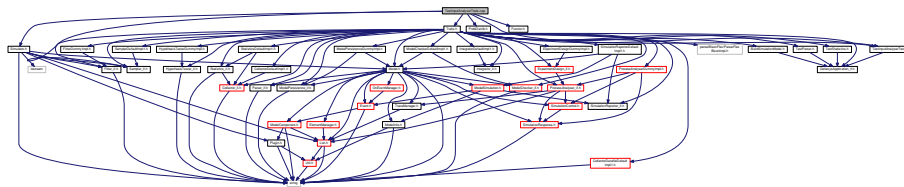
## Classes

- class [StatisticsDummyImpl](#)

## 6.133 TestInputAnalyserTools.cpp File Reference

```
#include "TestInputAnalyserTools.h"  
#include "Simulator.h"  
#include "Sampler_if.h"  
#include "ProbDistrib.h"  
#include "Traits.h"  
#include "Functor.h"
```

Include dependency graph for TestInputAnalyserTools.cpp:



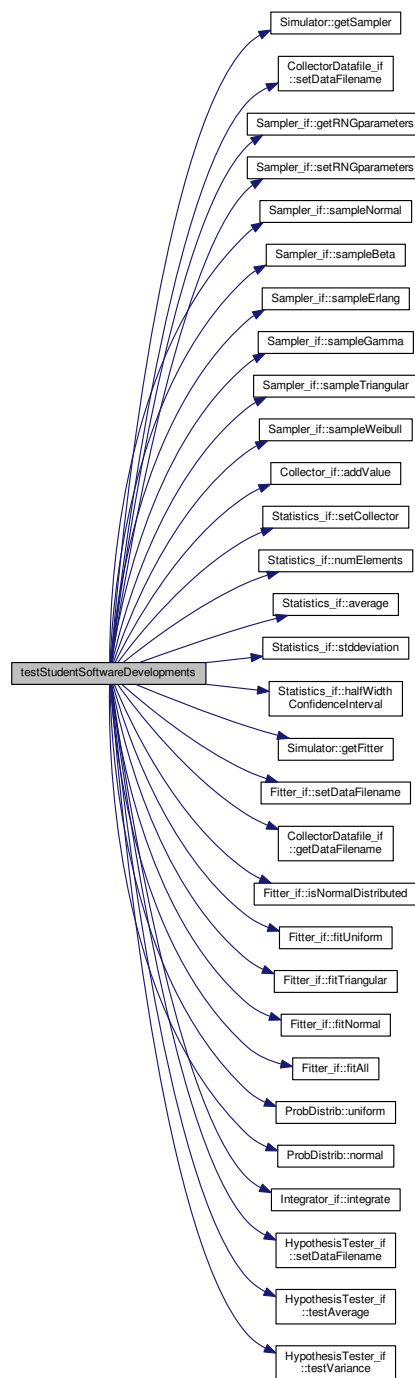
### Functions

- void [testStudentSoftwareDevelopments](#) ()

#### 6.133.1 Function Documentation

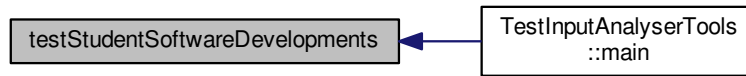
## 6.133.1.1 void testStudentSoftwareDevelopments ( )

Here is the call graph for this function:





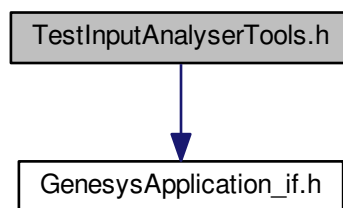
Here is the caller graph for this function:



## 6.134 TestInputAnalyserTools.h File Reference

```
#include "GenesysApplication_if.h"
```

Include dependency graph for TestInputAnalyserTools.h:



This graph shows which files directly or indirectly include this file:



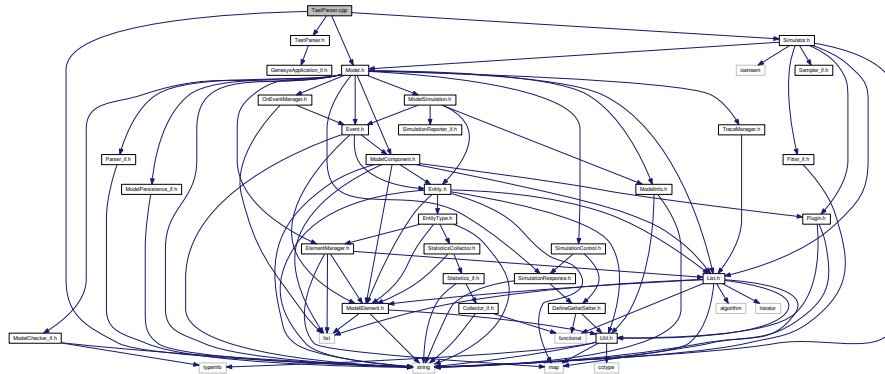
### Classes

- class [TestInputAnalyserTools](#)

## 6.135 TestParser.cpp File Reference

```
#include <string>
#include "TestParser.h"
#include "Model.h"
#include "Simulator.h"
```

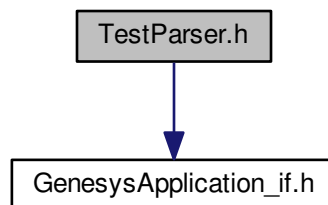
Include dependency graph for TestParser.cpp:



## 6.136 TestParser.h File Reference

```
#include "GenesysApplication_if.h"
```

Include dependency graph for TestParser.h:



This graph shows which files directly or indirectly include this file:

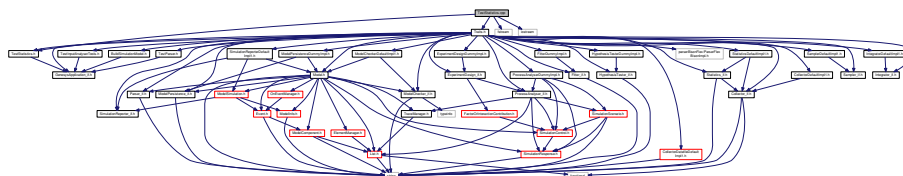


## Classes

- class [TestParser](#)

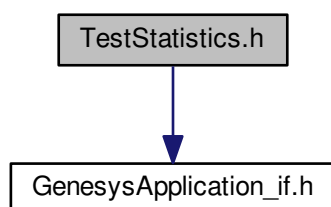
### 6.137 TestStatistics.cpp File Reference

```
#include "TestStatistics.h"
#include <fstream>
#include <iostream>
#include "Traits.h"
Include dependency graph for TestStatistics.cpp:
```



## 6.138 TestStatistics.h File Reference

```
#include "GenesysApplication_if.h"
Include dependency graph for TestStatistics.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

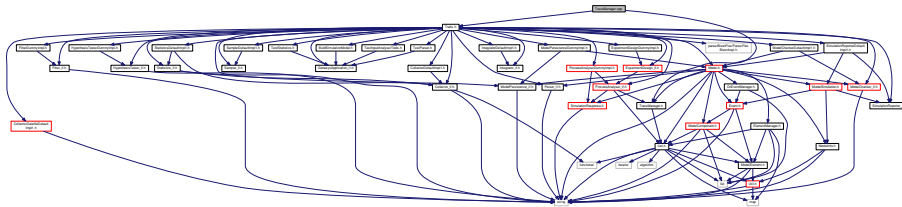
- class TestStatistics

## 6.139 TraceManager.cpp File Reference

```
#include "TraceManager.h"
```

```
#include "Traits.h"
```

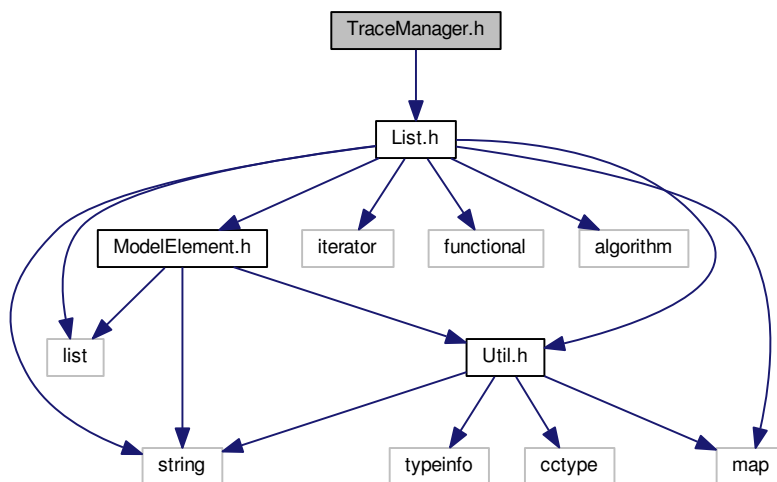
Include dependency graph for TraceManager.cpp:



## 6.140 TraceManager.h File Reference

```
#include "List.h"
```

Include dependency graph for TraceManager.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [TraceEvent](#)
- class [TraceErrorEvent](#)
- class [TraceSimulationEvent](#)
- class [TraceSimulationProcess](#)
- class [TraceManager](#)

## Typedefs

- typedef void(\* [traceListener](#)) ([TraceEvent](#))
- typedef void(\* [traceErrorListener](#)) ([TraceErrorEvent](#))
- typedef void(\* [traceSimulationListener](#)) ([TraceSimulationEvent](#))
- typedef void(\* [traceSimulationProcessListener](#)) ([TraceSimulationProcess](#))

### 6.140.1 Typedef Documentation

6.140.1.1 [typedef void\(\\* traceErrorListener\) \(TraceErrorEvent\)](#)

6.140.1.2 [typedef void\(\\* traceListener\) \(TraceEvent\)](#)

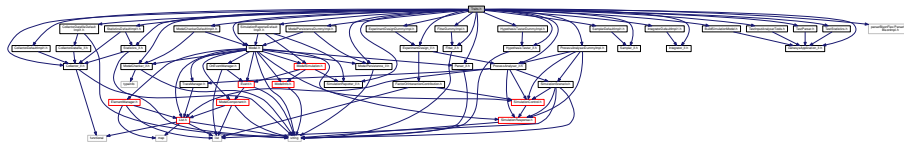
6.140.1.3 [typedef void\(\\* traceSimulationListener\) \(TraceSimulationEvent\)](#)

6.140.1.4 [typedef void\(\\* traceSimulationProcessListener\) \(TraceSimulationProcess\)](#)

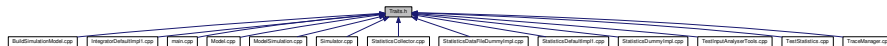
## 6.141 Traits.h File Reference

```
#include "Model.h"
#include "Collector_if.h"
#include "Sampler_if.h"
#include "Fitter_if.h"
#include "ModelChecker_if.h"
#include "Parser_if.h"
#include "Statistics_if.h"
#include "Integrator_if.h"
#include "HypothesisTester_if.h"
#include "ModelPersistence_if.h"
#include "GenesysApplication_if.h"
#include "ProcessAnalyser_if.h"
#include "ExperimentDesign_if.h"
#include "SimulationReporter_if.h"
#include "BuildSimulationModel.h"
#include "TestInputAnalyserTools.h"
#include "TestParser.h"
#include "TestStatistics.h"
#include "FitterDummyImpl.h"
#include "ExperimentDesignDummyImpl.h"
#include "ProcessAnalyserDummyImpl.h"
#include "HypothesisTesterDummyImpl.h"
#include "ModelPersistenceDummyImpl.h"
#include "CollectorDefaultImpl1.h"
#include "CollectorDatafileDefaultImpl1.h"
#include "StatisticsDefaultImpl1.h"
#include "IntegratorDefaultImpl1.h"
#include "SamplerDefaultImpl1.h"
#include "parserBisonFlex/ParserFlexBisonImpl.h"
#include "SimulationReporterDefaultImpl1.h"
#include "ModelCheckerDefaultImpl1.h"
```

Include dependency graph for Traits.h:



This graph shows which files directly or indirectly include this file:



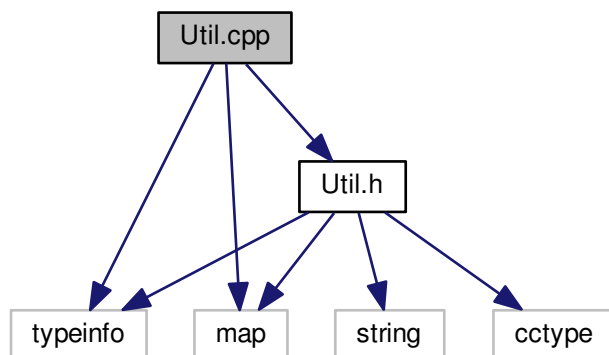
## Classes

- struct [Traits< T >](#)
- struct [Traits< GenesysApplication\\_if >](#)
- struct [Traits< Model >](#)
- struct [Traits< ModelPersistence\\_if >](#)
- struct [Traits< SimulationReporter\\_if >](#)
- struct [Traits< ModelComponent >](#)
- struct [Traits< ModelChecker\\_if >](#)
- struct [Traits< Parser\\_if >](#)
- struct [Traits< Collector\\_if >](#)
- struct [Traits< Statistics\\_if >](#)
- struct [Traits< Integrator\\_if >](#)
- struct [Traits< Sampler\\_if >](#)
- struct [Traits< Fitter\\_if >](#)
- struct [Traits< HypothesisTester\\_if >](#)
- struct [Traits< ExperimentDesign\\_if >](#)
- struct [Traits< ProcessAnalyser\\_if >](#)

## 6.142 Util.cpp File Reference

```
#include <typeinfo>
#include <map>
#include "Util.h"
```

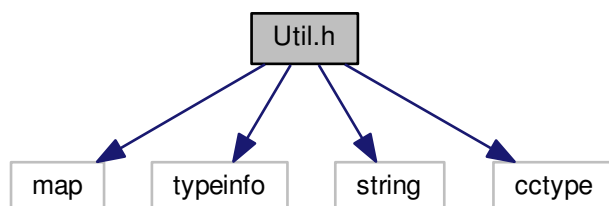
Include dependency graph for Util.cpp:



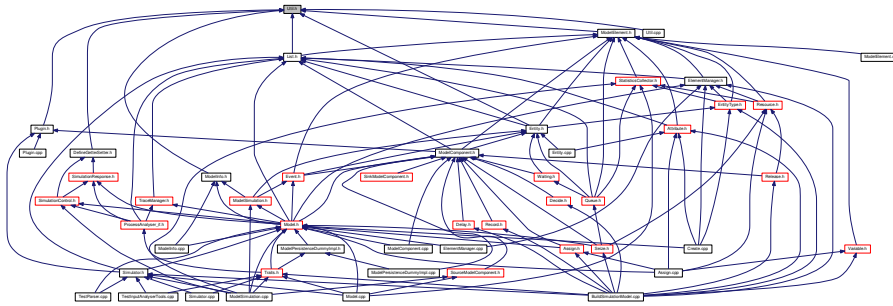
## 6.143 Util.h File Reference

```
#include <map>
#include <typeinfo>
#include <string>
#include <cctype>
```

Include dependency graph for Util.h:



This graph shows which files directly or indirectly include this file:



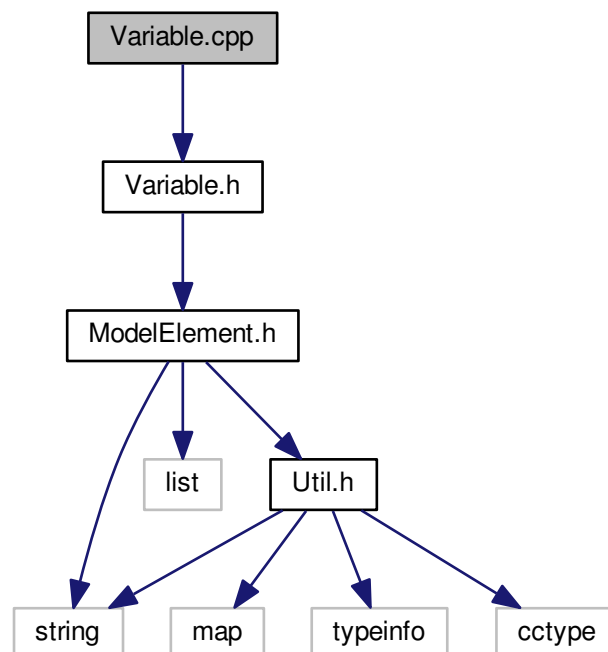
## Classes

- class [Util](#)

## 6.144 Variable.cpp File Reference

```
#include "Variable.h"
```

Include dependency graph for Variable.cpp:

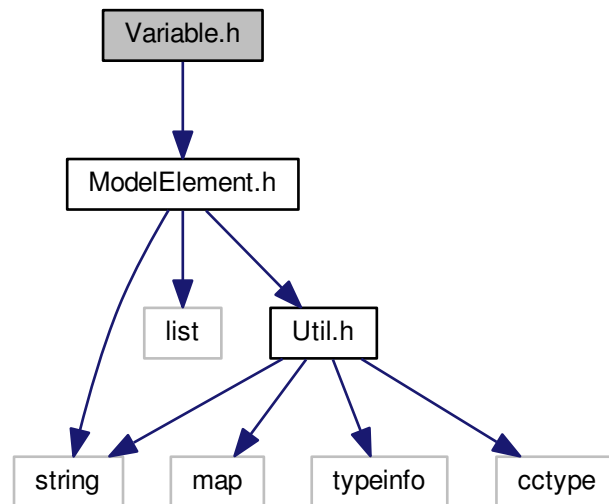




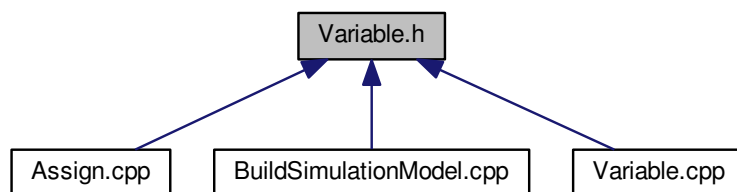
## 6.145 Variable.h File Reference

```
#include "ModelElement.h"
```

Include dependency graph for Variable.h:



This graph shows which files directly or indirectly include this file:



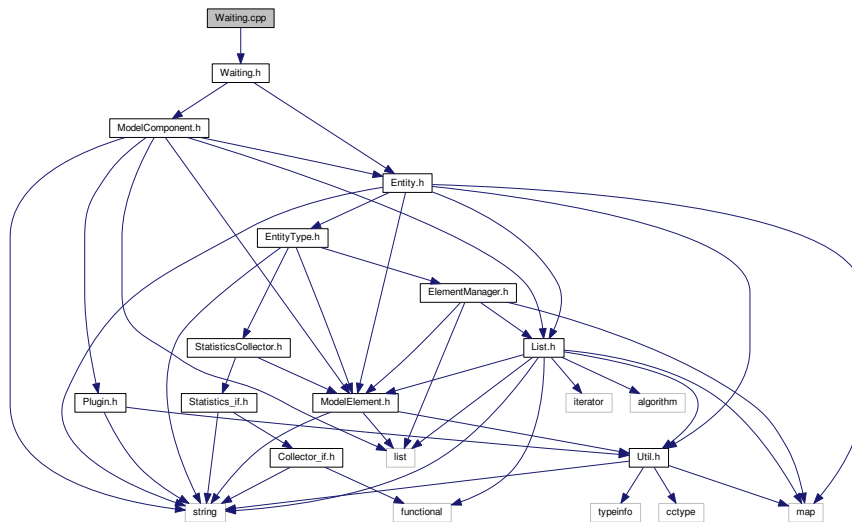
### Classes

- class [Variable](#)

## 6.146 Waiting.cpp File Reference

```
#include "Waiting.h"
```

Include dependency graph for Waiting.cpp:

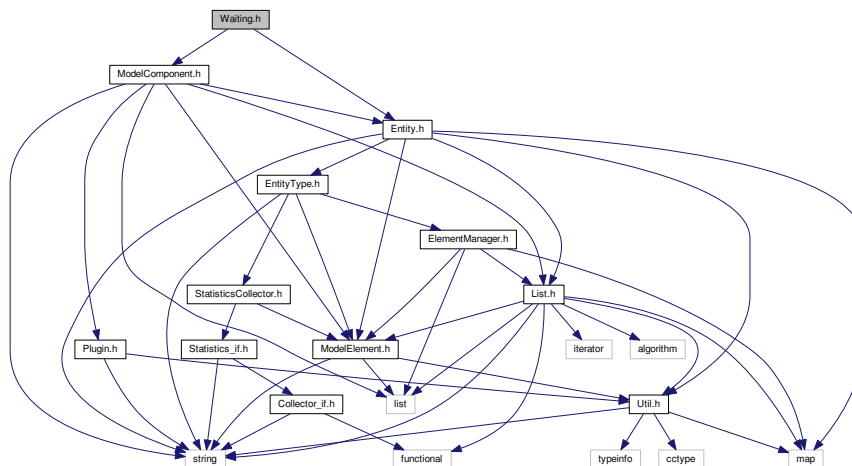


## 6.147 Waiting.h File Reference

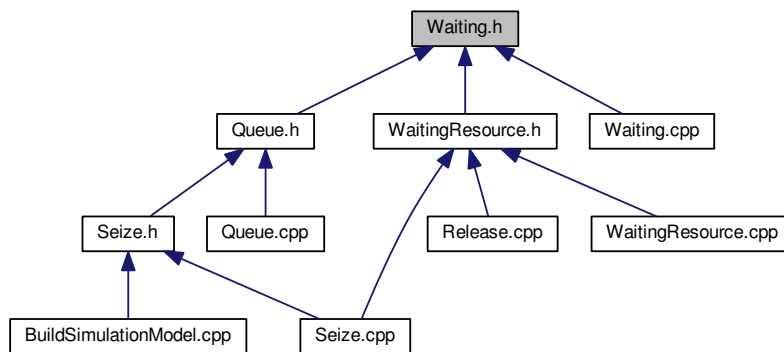
```
#include "Entity.h"
```

```
#include "ModelComponent.h"
```

Include dependency graph for Waiting.h:



This graph shows which files directly or indirectly include this file:



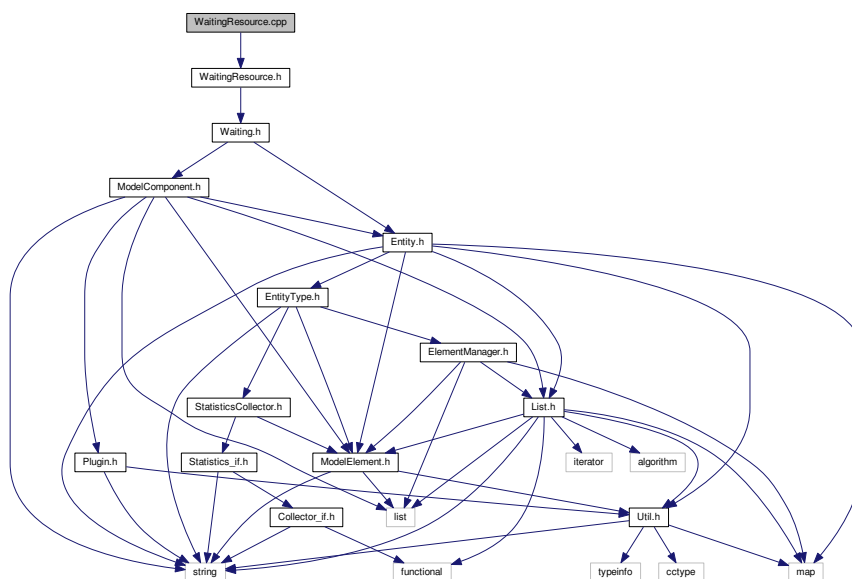
## Classes

- class [Waiting](#)

## 6.148 WaitingResource.cpp File Reference

```
#include "WaitingResource.h"
```

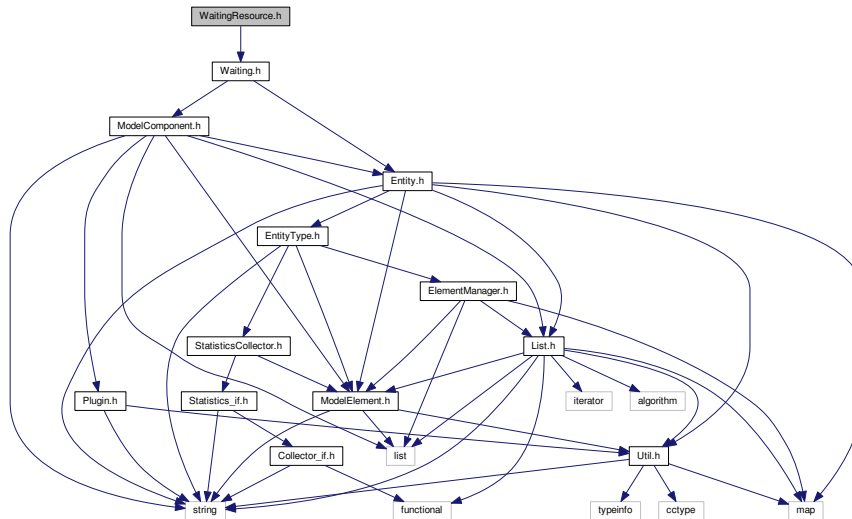
Include dependency graph for `WaitingResource.cpp`:



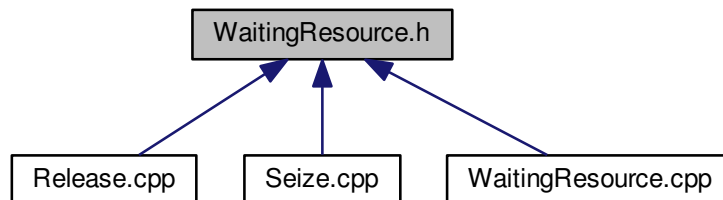
## 6.149 WaitingResource.h File Reference

```
#include "Waiting.h"
```

Include dependency graph for WaitingResource.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [WaitingResource](#)