

Generated by Doxygen 1.8.11

Contents

1	ReG	ienESyS	3		1					
2	Hier	archica	l Index	Index 3						
	2.1	Class	Hierarchy		3					
3	Clas	ss Index			7					
	3.1	Class	List		7					
4	File	Index			11					
	4.1	File Lis	st		11					
5	Clas	ss Docu	mentation	1	15					
	5.1	Assign	Class Re	ference	15					
		5.1.1	Member	Enumeration Documentation	17					
			5.1.1.1	DestinationType	17					
		5.1.2	Construc	ctor & 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	18					
			5.1.3.1	_check(std::string *errorMessage)	18					
			5.1.3.2	_execute(Entity *entity)	18					
			5.1.3.3	_initBetweenReplications()	19					
			5.1.3.4	_loadInstance(std::map< std::string, std::string > *fields)	20					
			5135	saveInstance()	20					

iv CONTENTS

		5.1.3.6	getAssignments() const	21
		5.1.3.7	GetPluginInformation()	21
		5.1.3.8	LoadInstance(Model *model, std::map< std::string, std::string > *fields)	22
		5.1.3.9	show()	22
5.2	Assign	:::Assignm	ent Class Reference	23
	5.2.1	Detailed	Description	23
	5.2.2	Construc	ctor & Destructor Documentation	23
		5.2.2.1	Assignment(DestinationType destinationType, std::string destination, std::string expression)	23
	5.2.3	Member	Function Documentation	23
		5.2.3.1	getDestination() const	23
		5.2.3.2	getDestinationType() const	24
		5.2.3.3	getExpression() const	24
		5.2.3.4	setDestination(std::string _destination)	25
		5.2.3.5	setDestinationType(DestinationType _destinationType)	25
		5.2.3.6	setExpression(std::string _expression)	25
5.3	Attribu	te Class R	deference	25
	5.3.1	Construc	ctor & Destructor Documentation	26
		5.3.1.1	Attribute()	26
		5.3.1.2	Attribute(std::string name)	27
		5.3.1.3	Attribute(const Attribute &orig)	27
		5.3.1.4	~Attribute()	27
	5.3.2	Member	Function Documentation	27
		5.3.2.1	_check(std::string *errorMessage)	27
		5.3.2.2	_loadInstance(std::map< std::string, std::string > *fields)	27
		5.3.2.3	_saveInstance()	27
		5.3.2.4	GetPluginInformation()	28
		5.3.2.5	LoadInstance(ElementManager *elems, std::map< std::string, std::string > *fields) 28
		5.3.2.6	show()	29
5.4	BuildS	imulationN	Model Class Reference	29
	5.4.1	Construc	ctor & Destructor Documentation	30

CONTENTS

		5.4.1.1	BuildSimulationModel()	30
	5.4.2	Member	Function Documentation	30
		5.4.2.1	main(int argc, char **argv)	30
5.5	Collect	tor_if Class	s Reference	32
	5.5.1	Detailed	Description	32
	5.5.2	Member	Function Documentation	32
		5.5.2.1	addValue(double value)=0	32
		5.5.2.2	clear()=0	33
		5.5.2.3	getLastValue()=0	33
		5.5.2.4	numElements()=0	33
		5.5.2.5	setAddValueHandler(CollectorAddValueHandler addValueHandler)=0	34
		5.5.2.6	setClearHandler(CollectorClearHandler clearHandler)=0	34
5.6	Collect	torDatafile	_if Class Reference	35
	5.6.1	Detailed	Description	36
	5.6.2	Member	Function Documentation	36
		5.6.2.1	getDataFilename()=0	36
		5.6.2.2	getNextValue()=0	36
		5.6.2.3	getValue(unsigned int rank)=0	37
		5.6.2.4	seekFirstValue()=0	37
		5.6.2.5	setDataFilename(std::string filename)=0	37
5.7	Collect	torDatafile	DefaultImpl1 Class Reference	37
	5.7.1	Construc	ctor & Destructor Documentation	38
		5.7.1.1	CollectorDatafileDefaultImpl1()	38
		5.7.1.2	CollectorDatafileDefaultImpl1(const CollectorDatafileDefaultImpl1 &orig)	38
		5.7.1.3	~CollectorDatafileDefaultImpl1()	38
	5.7.2	Member	Function Documentation	38
		5.7.2.1	addValue(double value)	38
		5.7.2.2	clear()	39
		5.7.2.3	getDataFilename()	39
		5.7.2.4	getLastValue()	39

vi

		5.7.2.5	getNextValue()	39
		5.7.2.6	getValue(unsigned int num)	39
		5.7.2.7	numElements()	39
		5.7.2.8	seekFirstValue()	39
		5.7.2.9	setAddValueHandler(CollectorAddValueHandler addValueHandler)	39
		5.7.2.10	setClearHandler(CollectorClearHandler clearHandler)	39
		5.7.2.11	setDataFilename(std::string filename)	40
5.8	Collect	torDatafileI	DummyImpl Class Reference	40
	5.8.1	Construc	tor & Destructor Documentation	41
		5.8.1.1	CollectorDatafileDummyImpl()	41
		5.8.1.2	CollectorDatafileDummyImpl(const CollectorDatafileDummyImpl &orig)	41
		5.8.1.3	~CollectorDatafileDummyImpl()	41
	5.8.2	Member	Function Documentation	41
		5.8.2.1	addValue(double value)	41
		5.8.2.2	clear()	41
		5.8.2.3	getDataFilename()	41
		5.8.2.4	getLastValue()	41
		5.8.2.5	getNextValue()	42
		5.8.2.6	getValue(unsigned int num)	42
		5.8.2.7	numElements()	42
		5.8.2.8	seekFirstValue()	42
		5.8.2.9	setAddValueHandler(CollectorAddValueHandler addValueHandler)	42
		5.8.2.10	setClearHandler(CollectorClearHandler clearHandler)	42
		5.8.2.11	setDataFilename(std::string filename)	42
5.9	Collect	torDefaultIr	mpl1 Class Reference	43
	5.9.1	Construc	tor & Destructor Documentation	44
		5.9.1.1	CollectorDefaultImpl1()	44
		5.9.1.2	CollectorDefaultImpl1(const CollectorDefaultImpl1 &orig)	44
		5.9.1.3	~CollectorDefaultImpl1()	44
	5.9.2	Member	Function Documentation	44

CONTENTS vii

		5.9.2.1	addValue(double value)	44
		5.9.2.2	clear()	44
		5.9.2.3	getLastValue()	44
		5.9.2.4	numElements()	44
		5.9.2.5	setAddValueHandler(CollectorAddValueHandler addValueHandler)	44
		5.9.2.6	setClearHandler(CollectorClearHandler clearHandler)	44
5.10	Collect	orDummyl	mpl Class Reference	45
	5.10.1	Construc	tor & Destructor Documentation	46
		5.10.1.1	CollectorDummyImpl()	46
		5.10.1.2	CollectorDummyImpl(const CollectorDummyImpl &orig)	46
		5.10.1.3	\sim CollectorDummyImpl()	46
	5.10.2	Member	Function Documentation	46
		5.10.2.1	addValue(double value)	46
		5.10.2.2	clear()	46
		5.10.2.3	getLastValue()	46
		5.10.2.4	numElements()	46
		5.10.2.5	setAddValueHandler(CollectorAddValueHandler addValueHandler)	46
		5.10.2.6	setClearHandler(CollectorClearHandler clearHandler)	46
5.11	Compo	nentMana	ger Class Reference	46
	5.11.1	Construc	tor & Destructor Documentation	47
		5.11.1.1	ComponentManager(Model *model)	47
		5.11.1.2	ComponentManager(const ComponentManager &orig)	47
		5.11.1.3	~ComponentManager()	47
	5.11.2	Member	Function Documentation	47
		5.11.2.1	begin()	47
		5.11.2.2	clear()	48
		5.11.2.3	end()	48
		5.11.2.4	getComponent(Util::identitifcation id)	49
		5.11.2.5	getComponent(std::string name)	49
		5.11.2.6	getNumberOfComponents()	49

viii CONTENTS

	5.11.2.7 insert(ModelComponent *comp)	49
	5.11.2.8 remove(ModelComponent *comp)	50
5.12 Count	ter Class Reference	50
5.12.1	Constructor & Destructor Documentation	51
	5.12.1.1 Counter()	51
	5.12.1.2 Counter(std::string name)	52
	5.12.1.3 Counter(std::string name, ModelElement *parent)	52
	5.12.1.4 Counter(const Counter &orig)	52
	5.12.1.5 ~Counter()	52
5.12.2	2 Member Function Documentation	52
	5.12.2.1 _check(std::string *errorMessage)	52
	5.12.2.2 _loadInstance(std::map< std::string, std::string > *fields)	52
	5.12.2.3 _saveInstance()	52
	5.12.2.4 clear()	53
	5.12.2.5 getCountValue()	53
	5.12.2.6 getParent() const	53
	5.12.2.7 GetPluginInformation()	54
	5.12.2.8 incCountValue(int value=1)	54
	5.12.2.9 LoadInstance(ElementManager *elems, std::map< std::string, std::string > *fields)	54
	5.12.2.10 show()	55
5.13 Creat	e Class Reference	55
5.13.1	Detailed Description	57
5.13.2	2 Constructor & Destructor Documentation	57
	5.13.2.1 Create(Model *model)	57
	5.13.2.2 Create(const Create &orig)	58
	5.13.2.3 ~Create()	58
5.13.3	Member Function Documentation	58
	5.13.3.1 _check(std::string *errorMessage)	58
	5.13.3.2 _execute(Entity *entity)	58
	5.13.3.3 _initBetweenReplications()	59

CONTENTS

		5.13.3.4	_loadInstance(std::map< std::string, std::string > *fields)	59
		5.13.3.5	_saveInstance()	60
		5.13.3.6	GetPluginInformation()	60
		5.13.3.7	LoadInstance(Model *model, std::map< std::string, std::string > *fields)	61
		5.13.3.8	show()	61
5.14	Decide	Class Re	erence	62
	5.14.1	Construc	tor & Destructor Documentation	63
		5.14.1.1	Decide(Model *model)	63
		5.14.1.2	Decide(const Decide &orig)	63
		5.14.1.3	~Decide()	63
	5.14.2	Member	Function Documentation	63
		5.14.2.1	_check(std::string *errorMessage)	63
		5.14.2.2	_execute(Entity *entity)	64
		5.14.2.3	_initBetweenReplications()	64
		5.14.2.4	_loadInstance(std::map< std::string, std::string > *fields)	64
		5.14.2.5	_saveInstance()	65
		5.14.2.6	getConditions() const	65
		5.14.2.7	GetPluginInformation()	65
		5.14.2.8	LoadInstance(Model *model, std::map< std::string, std::string > *fields)	66
		5.14.2.9	show()	66
5.15	Sample	erDefaultIn	npl1::DefaultImpl1RNG_Parameters Struct Reference	66
	5.15.1	Construc	tor & Destructor Documentation	67
		5.15.1.1	~DefaultImpl1RNG_Parameters()=default	67
	5.15.2	Member	Data Documentation	67
		5.15.2.1	module	68
		5.15.2.2	multiplier	68
		5.15.2.3	seed	68
5.16	Delay (Class Refe	rence	68
	5.16.1	Construc	tor & Destructor Documentation	69
		5.16.1.1	Delay(Model *model)	69

X CONTENTS

	5.16.1.2 Delay(const Delay &orig)	70
	5.16.1.3 ~Delay()	70
5.16.2	Member Function Documentation	70
	5.16.2.1 _check(std::string *errorMessage)	70
	5.16.2.2 _execute(Entity *entity)	70
	5.16.2.3 _initBetweenReplications()	71
	5.16.2.4 _loadInstance(std::map< std::string, std::string > *fields)	71
	5.16.2.5 _saveInstance()	71
	5.16.2.6 getDelayExpression() const	72
	5.16.2.7 getDelayTimeUnit() const	72
	5.16.2.8 GetPluginInformation()	72
	5.16.2.9 LoadInstance(Model *model, std::map< std::string, std::string > *fields)	72
	5.16.2.10 setDelayExpression(std::string _delayExpression)	73
	5.16.2.11 setDelayTimeUnit(Util::TimeUnit _delayTimeUnit)	73
	5.16.2.12 show()	73
5.17 Dispos	se Class Reference	74
5.17.1	Constructor & Destructor Documentation	75
	5.17.1.1 Dispose(Model *model)	75
	5.17.1.2 Dispose(const Dispose &orig)	76
	5.17.1.3 ~Dispose()	76
5.17.2	Member Function Documentation	76
	5.17.2.1 _check(std::string *errorMessage)	76
	5.17.2.2 _execute(Entity *entity)	76
	5.17.2.3 _initBetweenReplications()	76
	5.17.2.4 _loadInstance(std::map< std::string, std::string > *fields)	77
	5.17.2.5 _saveInstance()	77
	5.17.2.6 GetPluginInformation()	77
	5.17.2.7 LoadInstance(Model *model, std::map< std::string, std::string > *fields)	78
	5.17.2.8 show()	78

CONTENTS xi

	5.18.1	Construc	tor & Destructor Documentation	80
		5.18.1.1	Dummy(Model *model)	80
		5.18.1.2	Dummy(const Dummy &orig)	80
		5.18.1.3	\sim Dummy()	80
	5.18.2	Member	Function Documentation	80
		5.18.2.1	_check(std::string *errorMessage)	80
		5.18.2.2	_execute(Entity *entity)	81
		5.18.2.3	_initBetweenReplications()	81
		5.18.2.4	_loadInstance(std::map< std::string, std::string > *fields)	81
		5.18.2.5	_saveInstance()	82
		5.18.2.6	GetPluginInformation()	82
		5.18.2.7	LoadInstance(Model *model, std::map< std::string, std::string > *fields)	82
		5.18.2.8	show()	83
5.19	Elemer	ntManager	Class Reference	83
	5.19.1	Detailed	Description	84
	5.19.2	Construc	tor & Destructor Documentation	84
		5.19.2.1	ElementManager(Model *model)	84
		5.19.2.2	ElementManager(const ElementManager &orig)	84
		5.19.2.3	~ElementManager()	84
	5.19.3	Member	Function Documentation	84
		5.19.3.1	check(std::string infraTypename, ModelElement *infra, std::string expression ← Name, std::string *errorMessage)	84
		5.19.3.2	check(std::string infraTypename, std::string infraName, std::string expression → Name, bool mandatory, std::string *errorMessage)	85
		5.19.3.3	clear()	85
		5.19.3.4	getElement(std::string infraTypename, Util::identitifcation id)	85
		5.19.3.5	getElement(std::string infraTypename, std::string name)	86
		5.19.3.6	getElements(std::string infraTypename) const	86
		5.19.3.7	getElementTypenames() const	88
		5.19.3.8	getModel() const	88
		5.19.3.9	getNumberOfElements(std::string infraTypename)	88

xii CONTENTS

5.19.3.10 ge	etRankOf(std::string infraTypename, std::string name)	89
5.19.3.11 in:	sert(std::string infraTypename, ModelElement *infra)	89
5.19.3.12 re	move(std::string infraTypename, ModelElement *infra)	90
5.19.3.13 sh	now()	91
5.20 ElementManager_if	Class Reference	91
5.20.1 Constructor	& Destructor Documentation	92
5.20.1.1 El	ementManager_if()	92
5.20.1.2 El	ementManager_if(const ElementManager_if &orig)	92
5.20.1.3 ~	ElementManager_if()	92
5.21 Entity Class Referen	ice	92
5.21.1 Constructor	& Destructor Documentation	93
5.21.1.1 Er	ntity(ElementManager *elements)	93
5.21.1.2 Er	ntity(const Entity &orig)	93
5.21.1.3 ~	Entity()	93
5.21.2 Member Fur	nction Documentation	93
5.21.2.1 _c	check(std::string *errorMessage)	93
5.21.2.2 _	oadInstance(std::map< std::string, std::string > *fields)	94
5.21.2.3 _s	saveInstance()	94
5.21.2.4 ge	etAttributeValue(std::string attributeName)	94
5.21.2.5 ge	etEntityNumber() const	94
5.21.2.6 ge	etEntityType() const	95
5.21.2.7 ge	etEntityTypeName() const	95
5.21.2.8 se	etAttributeValue(std::string attributeName, double value)	95
5.21.2.9 se	etEntityType(EntityType *entityType)	96
5.21.2.10 se	etEntityTypeName(std::string entityTypeName)	96
5.21.2.11 sh	now()	96
5.22 EntityType Class Re	ference	97
5.22.1 Constructor	& Destructor Documentation	98
5.22.1.1 Er	ntityType(ElementManager *elemManager)	98
5.22.1.2 Er	ntityType(ElementManager *elemManager, std::string name)	99

CONTENTS xiii

		5.22.1.3	EntityType(const EntityType &orig)	99
		5.22.1.4	~EntityType()	99
	5.22.2	Member F	Function Documentation	99
		5.22.2.1	_check(std::string *errorMessage)	99
		5.22.2.2	_loadInstance(std::map< std::string, std::string > *fields)	99
		5.22.2.3	_saveInstance()	100
		5.22.2.4	getCstatNVATime() const	100
		5.22.2.5	getCstatOtherTime() const	100
		5.22.2.6	getCstatTotalTime() const	100
		5.22.2.7	getCstatTransferTime() const	100
		5.22.2.8	getCstatVATime() const	100
		5.22.2.9	getCstatWaitingTime() const	101
		5.22.2.10	getInitialNVACost() const	101
		5.22.2.11	getInitialOtherCost() const	101
		5.22.2.12	getInitialPicture() const	101
		5.22.2.13	getInitialVACost() const	101
		5.22.2.14	getInitialWaitingCost() const	101
		5.22.2.15	GetPluginInformation()	101
		5.22.2.16	LoadInstance(ElementManager *elems, std::map< std::string, std::string > *fields)	102
		5.22.2.17	setInitialNVACost(double _initialNVACost)	102
		5.22.2.18	setInitialOtherCost(double _initialOtherCost)	102
		5.22.2.19	setInitialPicture(std::string _initialPicture)	102
		5.22.2.20	setInitialVACost(double _initialVACost)	102
		5.22.2.21	setInitialWaitingCost(double _initialWaitingCost)	102
		5.22.2.22	show()	102
5.23	Event C	Class Refer	rence	103
	5.23.1	Construct	or & Destructor Documentation	103
		5.23.1.1	Event(double time, Entity *entity, ModelComponent *component)	103
		5.23.1.2	Event(const Event &orig)	103
		5.23.1.3	~Event()	103

xiv CONTENTS

	5.23.2	Member	Function Documentation	103
		5.23.2.1	getComponent() const	103
		5.23.2.2	getEntity() const	103
		5.23.2.3	getTime() const	103
		5.23.2.4	show()	104
5.24	Experir	mentDesig	n_if Class Reference	104
	5.24.1	Detailed	Description	105
	5.24.2	Member	Function Documentation	105
		5.24.2.1	calculateContributionAndCoefficients()=0	105
		5.24.2.2	generate2krScenarioExperiments()=0	105
		5.24.2.3	getContributions() const =0	105
		5.24.2.4	getProcessAnalyser() const =0	105
5.25	Experir	mentDesig	nDefaultImpl1 Class Reference	106
	5.25.1	Construc	tor & Destructor Documentation	107
		5.25.1.1	ExperimentDesignDefaultImpl1()	107
		5.25.1.2	ExperimentDesignDefaultImpl1(const ExperimentDesignDefaultImpl1 &orig)	107
		5.25.1.3	~ExperimentDesignDefaultImpl1()	107
	5.25.2	Member	Function Documentation	107
		5.25.2.1	calculateContributionAndCoefficients()	107
		5.25.2.2	generate2krScenarioExperiments()	107
		5.25.2.3	getContributions() const	107
		5.25.2.4	getProcessAnalyser() const	107
5.26	Experir	nentDesig	nDummyImpl Class Reference	108
	5.26.1	Construc	tor & Destructor Documentation	109
		5.26.1.1	ExperimentDesignDummyImpl()	109
		5.26.1.2	ExperimentDesignDummyImpl(const ExperimentDesignDummyImpl &orig)	109
		5.26.1.3	$\sim\!\!ExperimentDesignDummyImpI() \ldots \ldots \ldots \ldots \ldots \ldots \ldots$	109
	5.26.2	Member	Function Documentation	109
		5.26.2.1	calculateContributionAndCoefficients()	109
		5.26.2.2	generate2krScenarioExperiments()	109

CONTENTS xv

		5.26.2.3	getContributions() const	109
		5.26.2.4	getProcessAnalyser() const	109
5.27	Factor	OrInteraction	onContribution Class Reference	109
	5.27.1	Detailed I	Description	110
	5.27.2	Construct	tor & Destructor Documentation	110
		5.27.2.1	$\label{lem:contribution} FactorOrInteractionContribution(double contribution, double modelCoefficient, std::list< SimulationControl * > *controls)$	110
		5.27.2.2	FactorOrInteractionContribution(const FactorOrInteractionContribution &orig)	110
		5.27.2.3	$\sim\!\!\text{FactorOrInteractionContribution}() \dots \dots \dots \dots \dots \dots \dots \dots \dots $	110
	5.27.3	Member I	Function Documentation	110
		5.27.3.1	getContribution() const	110
		5.27.3.2	getControls() const	110
		5.27.3.3	getModelCoefficient() const	110
5.28	Fitter_i	f Class Re	ference	110
	5.28.1	Member I	Function Documentation	111
		5.28.1.1	fitAll(double *sqrerror, std::string *name)=0	111
		5.28.1.2	fitBeta(double *sqrerror, double *alpha, double *beta, double *infLimit, double *supLimit)=0	111
		5.28.1.3	fitErlang(double *sqrerror, double *avg, double *m)=0	111
		5.28.1.4	fitExpo(double *sqrerror, double *avg1)=0	111
		5.28.1.5	fitNormal(double *sqrerror, double *avg, double *stddev)=0	112
		5.28.1.6	fitTriangular(double *sqrerror, double *min, double *mo, double *max)=0	112
		5.28.1.7	fitUniform(double *sqrerror, double *min, double *max)=0	112
		5.28.1.8	fitWeibull(double *sqrerror, double *alpha, double *scale)=0	112
		5.28.1.9	getDataFilename()=0	113
		5.28.1.10	isNormalDistributed(double confidencelevel)=0	113
		5.28.1.11	setDataFilename(std::string dataFilename)=0	113
5.29	FitterDe	efaultImpl1	Class Reference	113
	5.29.1	Construct	tor & Destructor Documentation	114
		5.29.1.1	FitterDefaultImpl1()	114
		5.29.1.2	FitterDefaultImpl1(const FitterDefaultImpl1 &orig)	114

xvi CONTENTS

	5.29.1.3	~FitterDefaultImpl1()	114
5.29.2	Member F	Function Documentation	114
	5.29.2.1	fitAll(double *sqrerror, std::string *name)	114
	5.29.2.2	fitBeta(double *sqrerror, double *alpha, double *beta, double *infLimit, double *supLimit)	115
	5.29.2.3	fitErlang(double *sqrerror, double *avg, double *m)	115
	5.29.2.4	fitExpo(double *sqrerror, double *avg1)	115
	5.29.2.5	fitNormal(double *sqrerror, double *avg, double *stddev)	115
	5.29.2.6	fitTriangular(double *sqrerror, double *min, double *mo, double *max)	115
	5.29.2.7	fitUniform(double *sqrerror, double *min, double *max)	115
	5.29.2.8	fitWeibull(double *sqrerror, double *alpha, double *scale)	115
	5.29.2.9	getDataFilename()	115
	5.29.2.10	isNormalDistributed(double confidencelevel)	115
	5.29.2.11	setDataFilename(std::string dataFilename)	115
FitterDu	ummylmpl	Class Reference	116
5.30.1	Construct	or & Destructor Documentation	117
	5.30.1.1	FitterDummyImpl()	117
	5.30.1.2	FitterDummyImpl(const FitterDummyImpl &orig)	117
	5.30.1.3	$\sim\!\!FitterDummyImpI()\ldots\ldots\ldots\ldots\ldots\ldots\ldots$	117
5.30.2	Member F	Function Documentation	117
	5.30.2.1	fitAll(double *sqrerror, std::string *name)	117
	5.30.2.2	fitBeta(double *sqrerror, double *alpha, double *beta, double *infLimit, double *supLimit)	117
	5.30.2.3	fitErlang(double *sqrerror, double *avg, double *m)	117
	5.30.2.4	fitExpo(double *sqrerror, double *avg1)	117
	5.30.2.5	fitNormal(double *sqrerror, double *avg, double *stddev)	117
	5.30.2.6	fitTriangular(double *sqrerror, double *min, double *mo, double *max)	117
	5.30.2.7	fitUniform(double *sqrerror, double *min, double *max)	117
	5.30.2.8	fitWeibull(double *sqrerror, double *alpha, double *scale)	117
	5.30.2.9	getDataFilename()	118
	5.30.2.10	isNormalDistributed(double confidencelevel)	118
	FitterDt 5.30.1	5.29.2 Member F 5.29.2.1 5.29.2.3 5.29.2.4 5.29.2.5 5.29.2.6 5.29.2.7 5.29.2.8 5.29.2.10 5.29.2.11 FitterDummyImpl 5.30.1 Construct 5.30.1.1 5.30.1.2 5.30.1.3 5.30.2.1 5.30.2.1 5.30.2.2 5.30.2.3 5.30.2.4 5.30.2.5 5.30.2.6 5.30.2.7 5.30.2.8	5.29.2 Member Function Documentation 5.29.2.1 fitAll(double *sqrerror, std::string *name) 5.29.2.2 fitBeta(double *sqrerror, double *alpha, double *beta, double *infLimit, double *supLimit) 5.29.2.3 fitErlang(double *sqrerror, double *avg, double *m) 5.29.2.4 fitExpo(double *sqrerror, double *avg, double *stddev) 5.29.2.5 fitNormal(double *sqrerror, double *avg, double *stddev) 5.29.2.6 fitTriangular(double *sqrerror, double *min, double *mo, double *max) 5.29.2.7 fitUniform(double *sqrerror, double *min, double *max) 5.29.2.8 fitWeibull(double *sqrerror, double *alpha, double *scale) 5.29.2.9 getDataFilename() 5.29.2.10 isNormalDistributed(double confidencelevel) 5.29.2.11 setDataFilename(std::string dataFilename) FitterDummylmpl Class Reference 5.30.1. Constructor & Destructor Documentation 5.30.1.1 FitterDummylmpl() 5.30.1.2 FitterDummylmpl() 5.30.1.3 ~FitterDummylmpl() 5.30.2.1 fitAll(double *sqrerror, double *alpha, double *beta, double *infLimit, double *supLimit) 5.30.2.2 fitBeta(double *sqrerror, double *avg, double *m) 5.30.2.3 fitErlang(double *sqrerror, double *avg, double *m) 5.30.2.5 fitNormal(double *sqrerror, double *avg, double *mo, double *max) 5.30.2.6 fitTriangular(double *sqrerror, double *avg, double *mo, double *max) 5.30.2.7 fitUniform(double *sqrerror, double *avg, double *mo, double *max) 5.30.2.8 fitUniform(double *sqrerror, double *min, double *mo, double *max) 5.30.2.8 fitWeibull(double *sqrerror, double *min, double *max)

CONTENTS xvii

5.30.2.11 setDataFilename(std::string dataFilename)	118
5.31 GenesysApplication_if Class Reference	118
5.31.1 Member Function Documentation	119
5.31.1.1 main(int argc, char **argv)=0	119
5.32 GenesysConsole Class Reference	119
5.32.1 Constructor & Destructor Documentation	120
5.32.1.1 GenesysConsole()	121
5.32.1.2 GenesysConsole(const GenesysConsole &orig)	121
5.32.1.3 ~GenesysConsole()	121
5.32.2 Member Function Documentation	121
5.32.2.1 cmdHelp()	122
5.32.2.2 cmdModelCheck()	122
5.32.2.3 cmdModelLoad()	123
5.32.2.4 cmdModelSave()	123
5.32.2.5 cmdModelShow()	124
5.32.2.6 cmdQuit()	124
5.32.2.7 cmdScript()	125
5.32.2.8 cmdShowReport()	126
5.32.2.9 cmdStart()	127
5.32.2.10 cmdStep()	128
5.32.2.11 cmdStop()	129
5.32.2.12 cmdTraceLevel()	129
5.32.2.13 cmdVersion()	130
5.32.2.14 main(int argc, char **argv)	130
5.33 GenesysGUI Class Reference	131
5.33.1 Constructor & Destructor Documentation	131
5.33.1.1 GenesysGUI()	131
5.33.1.2 GenesysGUI(const GenesysGUI &orig)	131
5.33.1.3 ~GenesysGUI()	132
5.33.2 Member Function Documentation	132

xviii CONTENTS

		5.33.2.1	main(int argc, char **argv)	132
5.34	Genes	ysShell_if(Class Reference	132
	5.34.1	Member F	Function Documentation	133
		5.34.1.1	addFromFile(std::string filename)=0	133
		5.34.1.2	addPlugin(std::string filename)=0	133
		5.34.1.3	check()=0	133
		5.34.1.4	closeModel()=0	133
		5.34.1.5	createModel()=0	133
		5.34.1.6	deleteTraceFiles()=0	133
		5.34.1.7	execLinuxCommand(std::string command)=0	133
		5.34.1.8	getCommandLine()=0	133
		5.34.1.9	getGenesysInfo()=0	134
		5.34.1.10	listComponents()=0	134
		5.34.1.11	listElements()=0	134
		5.34.1.12	listHosts()=0	134
		5.34.1.13	listPlugins()=0	134
		5.34.1.14	openModel(std::string filename)=0	134
		5.34.1.15	readCommandsFromFile(std::string filename)=0	134
		5.34.1.16	receiveFile(std::string filename)=0	134
		5.34.1.17	redirectTrace(std::string trace, std::string dest, std::string filename)=0	134
		5.34.1.18	saveModel()=0	134
		5.34.1.19	saveModelAs(std::string filename)=0	134
		5.34.1.20	sendFile(std::string filename, std::string hostname, std::string portname)=0	134
		5.34.1.21	setActivationCode(std::string code)=0	134
		5.34.1.22	showHelp()=0	134
		5.34.1.23	showHostName()=0	134
		5.34.1.24	showInit()=0	134
		5.34.1.25	startSimulation()=0	134
		5.34.1.26	stepSimulation()=0	134
		5.34.1.27	stopSimulation()=0	134

CONTENTS xix

		5.34.1.28	traceLevel(Util::TraceLevel tracelevel)=0	134
		5.34.1.29	verboseMode(bool on)=0	134
5.35	Hypoth	esisTester	_if Class Reference	135
	5.35.1	Detailed	Description	135
	5.35.2	Member	Enumeration Documentation	135
		5.35.2.1	H1Comparition	135
	5.35.3	Member	Function Documentation	136
		5.35.3.1	getDataFilename()=0	136
		5.35.3.2	setDataFilename(std::string dataFilename)=0	136
		5.35.3.3	testAverage(double confidencelevel, double avg, H1Comparition comp)=0	136
		5.35.3.4	testAverage(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)=0	136
		5.35.3.5	$test Proportion (double\ confidence level,\ double\ prop,\ H1Comparition\ comp) = 0 . .$	136
		5.35.3.6	testProportion(double confidencelevel, std::string secondPopulationData← Filename, H1Comparition comp)=0	137
		5.35.3.7	testVariance(double confidencelevel, double var, H1Comparition comp)=0	137
		5.35.3.8	testVariance(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)=0	137
5.36	Hypoth	esisTester	DefaultImpl1 Class Reference	137
	5.36.1	Construc	tor & Destructor Documentation	138
		5.36.1.1	HypothesisTesterDefaultImpl1()	138
		5.36.1.2	HypothesisTesterDefaultImpl1(const HypothesisTesterDefaultImpl1 &orig)	138
		5.36.1.3	~HypothesisTesterDefaultImpl1()	138
	5.36.2	Member	Function Documentation	138
		5.36.2.1	getDataFilename()	138
		5.36.2.2	setDataFilename(std::string dataFilename)	139
		5.36.2.3	testAverage(double confidencelevel, double avg, H1Comparition comp)	139
		5.36.2.4	testAverage(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)	139
		5.36.2.5	testProportion(double confidencelevel, double prop, H1Comparition comp)	139
		5.36.2.6	testProportion(double confidencelevel, std::string secondPopulationData← Filename, H1Comparition comp)	139
		5.36.2.7	testVariance(double confidencelevel, double var, H1Comparition comp)	139

CONTENTS

		5.36.2.8	testVariance(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)	139
5.37	Hypoth	esisTester	DummyImpl Class Reference	140
	5.37.1	Construc	tor & Destructor Documentation	141
		5.37.1.1	HypothesisTesterDummyImpl()	141
		5.37.1.2	HypothesisTesterDummyImpl(const HypothesisTesterDummyImpl &orig)	141
		5.37.1.3	~HypothesisTesterDummyImpI()	141
	5.37.2	Member	Function Documentation	141
		5.37.2.1	getDataFilename()	141
		5.37.2.2	setDataFilename(std::string dataFilename)	141
		5.37.2.3	testAverage(double confidencelevel, double avg, H1Comparition comp)	141
		5.37.2.4	testAverage(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)	141
		5.37.2.5	testProportion(double confidencelevel, double prop, H1Comparition comp)	141
		5.37.2.6	testProportion(double confidencelevel, std::string secondPopulationData⇔ Filename, H1Comparition comp)	141
		5.37.2.7	testVariance(double confidencelevel, double var, H1Comparition comp)	141
		5.37.2.8	testVariance(double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp)	142
5.38	Integra	tor_if Clas	s Reference	142
	5.38.1	Detailed	Description	142
	5.38.2	Member	Function Documentation	143
		5.38.2.1	getPrecision()=0	143
		5.38.2.2	integrate(double min, double max, double(*f)(double, double), double p2)=0	143
		5.38.2.3	integrate(double min, double max, double(*f)(double, double, double), double p2, double p3)=0	143
		5.38.2.4	integrate(double min, double max, double(*f)(double, double, double, double), double p2, double p3, double p4)=0	143
		5.38.2.5	integrate(double min, double max, double(*f)(double, double, double, double, double, double p2, double p3, double p4, double p5)=0	143
		5.38.2.6	setPrecision(double e)=0	143
5.39	Integra	torDefaultI	mpl1 Class Reference	144
	5.39.1	Construc	tor & Destructor Documentation	145

CONTENTS xxi

		5.39.1.1	IntegratorDefaultImpl1()	145
		5.39.1.2	IntegratorDefaultImpl1(const IntegratorDefaultImpl1 &orig)	145
		5.39.1.3	~IntegratorDefaultImpl1()	145
	5.39.2	Member	Function Documentation	145
		5.39.2.1	getPrecision()	145
		5.39.2.2	integrate(double min, double max, double(*f)(double, double), double p2)	145
		5.39.2.3	integrate(double min, double max, double(*f)(double, double, double), double p2, double p3)	145
		5.39.2.4	$\label{eq:continuous} \begin{array}{llllllllllllllllllllllllllllllllllll$	145
		5.39.2.5	integrate(double min, double max, double(*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)	145
		5.39.2.6	setPrecision(double e)	145
5.40	Integra	torDummy	Impl Class Reference	146
	5.40.1	Construc	tor & Destructor Documentation	147
		5.40.1.1	IntegratorDummyImpl()	147
		5.40.1.2	IntegratorDummyImpl(const IntegratorDummyImpl &orig)	147
		5.40.1.3	~IntegratorDummyImpl()	147
	5.40.2	Member	Function Documentation	147
		5.40.2.1	getPrecision()	147
		5.40.2.2	$integrate(double\ min,\ double\ max,\ double(*f)(double,\ double),\ double\ p2)\ \ .\ \ .\ \ .$	147
		5.40.2.3	integrate(double min, double max, double(*f)(double, double, double), double p2, double p3)	147
		5.40.2.4	$integrate(double\ min,\ double\ max,\ double(*f)(double,\ double,\ double,\ double),\\ double\ p2,\ double\ p3,\ double\ p4) \ \ . \ . \ . \ . \ . \ . \ . \ . \ . $	147
		5.40.2.5	integrate(double min, double max, double(*f)(double, double, double, double, double), double p2, double p3, double p4, double p5)	147
		5.40.2.6	setPrecision(double e)	147
5.41	Licence	eManager	Class Reference	148
	5.41.1	Construc	tor & Destructor Documentation	148
		5.41.1.1	LicenceManager(Simulator *simulator)	148
		5.41.1.2	LicenceManager(const LicenceManager &orig)	148
		5.41.1.3	~LicenceManager()	148

xxii CONTENTS

5.4	41.2	Member I	Function Do	cumentati	ion .		 	 	٠.	 	 	 	148
		5.41.2.1	getEntityLir	nit()			 	 		 	 	 	148
		5.41.2.2	getHostsLin	nit()			 	 		 	 	 	148
		5.41.2.3	getModelC	omponen	tsLimit(()	 	 		 	 	 	148
		5.41.2.4	getModelEl	ementsLi	imit()		 	 		 	 	 	148
		5.41.2.5	getThreads	Limit() .			 	 		 	 	 	148
		5.41.2.6	insertActiva	ationCode	e()		 	 		 	 	 	148
		5.41.2.7	lookforActiv	ationCod	le() .		 	 		 	 	 	148
		5.41.2.8	removeActi	vationCo	de() .		 	 		 	 	 	148
		5.41.2.9	showActiva	tionCode	() cons	t	 	 		 	 	 	149
		5.41.2.10	showLicen	ce() const	i		 	 		 	 	 	149
		5.41.2.11	showLimits	() const			 	 		 	 	 	149
5.42 Lir	nkedE	By Class F	deference .				 	 		 	 	 	149
5.4	42.1	Construct	or & Destru	ctor Docu	ımentat	ion	 	 		 	 	 	150
		5.42.1.1	LinkedBy()				 	 		 	 	 	150
		5.42.1.2	LinkedBy(c	onst Link	edBy &	orig)	 	 		 	 	 	150
		5.42.1.3	\sim LinkedBy	()			 	 		 	 	 	150
5.4	42.2	Member I	Function Do	cumentati	ion .		 	 		 	 	 	150
		5.42.2.1	addLink() .				 	 		 	 	 	150
		5.42.2.2	isLinked() .				 	 		 	 	 	150
		5.42.2.3	removeLink	κ()			 	 		 	 	 	150
5.43 Lis	st< T	> Class	Гетрlate Re	ference .			 	 		 	 	 	150
5.4	43.1	Detailed I	Description				 	 		 	 	 	151
5.4	43.2	Member ⁻	Typedef Doc	umentatio	on		 	 		 	 	 	151
		5.43.2.1	CompFunc	t			 	 		 	 	 	151
5.4	43.3	Construct	or & Destru	ctor Docu	ımentat	ion	 	 		 	 	 	151
		5.43.3.1	List()				 	 		 	 	 	151
		5.43.3.2	List(const L	.ist &orig)			 	 		 	 	 	151
		5.43.3.3	\sim List()				 	 		 	 	 	151
5.4	43.4	Member I	Function Do	cumentati	ion .		 	 		 	 	 	151

CONTENTS xxiii

		5.43.4.1	actual()	152
		5.43.4.2	clear()	152
		5.43.4.3	create()	152
		5.43.4.4	create(U arg)	152
		5.43.4.5	empty()	152
		5.43.4.6	find(T element)	153
		5.43.4.7	front()	153
		5.43.4.8	getAtRank(unsigned int rank)	153
		5.43.4.9	getList() const	154
		5.43.4.10	insert(T element)	155
		5.43.4.11	last()	156
		5.43.4.12	? next()	156
		5.43.4.13	pop_front()	156
		5.43.4.14	previous()	157
		5.43.4.15	remove(T element)	157
		5.43.4.16	s setAtRank(unsigned int rank, T element)	157
		5.43.4.17	setSortFunc(CompFunct _sortFunc)	158
		5.43.4.18	show()	159
		5.43.4.19	size()	159
		5.43.4.20	sort(Compare comp)	159
5.44	Model (Class Refe	erence	159
	5.44.1	Detailed I	Description	160
	5.44.2	Construc	tor & Destructor Documentation	161
		5.44.2.1	Model(Simulator *simulator)	161
		5.44.2.2	Model(const Model &orig)	161
		5.44.2.3	~Model()	162
	5.44.3	Member I	Function Documentation	162
		5.44.3.1	checkExpression(const std::string expression, const std::string expressionName, std::string *errorMessage)	162
		5.44.3.2	checkModel()	163
		5.44.3.3	clear()	163

xxiv CONTENTS

	5.44.3.4	getComponentManager() const	164
	5.44.3.5	getControls() const	164
	5.44.3.6	getElementManager() const	164
	5.44.3.7	getEvents() const	166
	5.44.3.8	getId() const	166
	5.44.3.9	getInfos() const	166
	5.44.3.10	getOnEventManager() const	167
	5.44.3.11	getParent() const	167
	5.44.3.12	getResponses() const	167
	5.44.3.13	getSimulation() const	168
	5.44.3.14	getTraceManager() const	168
	5.44.3.15	loadModel(std::string filename)	169
	5.44.3.16	parseExpression(const std::string expression)	170
	5.44.3.17	parseExpression(const std::string expression, bool *success, std::string *error← Message)	171
	5.44.3.18	removeEntity(Entity *entity, bool collectStatistics)	171
	5.44.3.19	saveModel(std::string filename)	172
	5.44.3.20	sendEntityToComponent(Entity *entity, ModelComponent *component, double timeDelay)	172
	5.44.3.21	setTraceManager(TraceManager *_traceManager)	173
	5.44.3.22	show()	174
5.45 Model(Checker_if	Class Reference	175
5.45.1	Detailed I	Description	175
5.45.2	Member I	Function Documentation	175
	5.45.2.1	checkActivationCode()=0	175
	5.45.2.2	checkAII()=0	175
	5.45.2.3	checkConnected()=0	176
	5.45.2.4	checkSymbols()=0	176
5.46 Model	CheckerDe	faultImpl1 Class Reference	176
5.46.1	Construct	tor & Destructor Documentation	177
	5.46.1.1	ModelCheckerDefaultImpl1(Model *model)	177

CONTENTS xxv

		5.46.1.2	ModelCheckerDefaultImpl1(const ModelCheckerDefaultImpl1 &orig)	177
		5.46.1.3	~ModelCheckerDefaultImpl1()	177
	5.46.2	Member F	Function Documentation	177
		5.46.2.1	checkActivationCode()	177
		5.46.2.2	checkAll()	177
		5.46.2.3	checkConnected()	178
		5.46.2.4	checkSymbols()	179
5.47	ModelC	Component	Class Reference	181
	5.47.1	Detailed D	Description	182
	5.47.2	Constructo	or & Destructor Documentation	182
		5.47.2.1	ModelComponent(Model *model, std::string componentTypename)	182
		5.47.2.2	ModelComponent(const ModelComponent &orig)	182
		5.47.2.3	~ModelComponent()	182
	5.47.3	Member F	Function Documentation	182
		5.47.3.1	_execute(Entity *entity)=0	182
		5.47.3.2	_initBetweenReplications()=0	183
		5.47.3.3	_loadInstance(std::map< std::string, std::string > *fields)	183
		5.47.3.4	_saveInstance()	184
		5.47.3.5	Check(ModelComponent *component)	185
		5.47.3.6	Execute(Entity *entity, ModelComponent *component)	185
		5.47.3.7	getNextComponents() const	186
		5.47.3.8	InitBetweenReplications(ModelComponent *component)	187
		5.47.3.9	LoadInstance(Model *model, std::map< std::string, std::string > *fields)	187
		5.47.3.10	SaveInstance(ModelComponent *component)	187
		5.47.3.11	show()	188
	5.47.4	Member D	Data Documentation	188
		5.47.4.1	_model	188
5.48	ModelC	Component	Manager_if Class Reference	188
	5.48.1	Constructo	or & Destructor Documentation	189
		5.48.1.1	ModelComponentManager_if()	189

xxvi CONTENTS

		5.48.1.2 M	lodelComponentManager_if(const ModelComponentManager_if &orig)	189
		5.48.1.3 ~	ModelComponentManager_if()	189
5.49	ModelE	Element Class	s Reference	189
	5.49.1	Detailed De	scription	190
	5.49.2	Constructor	& Destructor Documentation	190
		5.49.2.1 M	lodelElement(std::string elementTypename)	190
		5.49.2.2 M	lodelElement(const ModelElement &orig)	191
		5.49.2.3 ~	ModelElement()	191
	5.49.3	Member Fur	nction Documentation	191
		5.49.3.1 _0	check(std::string *errorMessage)	191
		5.49.3.2 _l	loadInstance(std::map< std::string, std::string > *fields)	192
		5.49.3.3	saveInstance()	192
		5.49.3.4 C	heck(ModelElement *element, std::string *errorMessage)	193
		5.49.3.5 ge	etld() const	194
		5.49.3.6 ge	etName() const	195
		5.49.3.7 ge	etTypename() const	195
		5.49.3.8 Lo	oadInstance(std::map< std::string, std::string > *fields)	196
		5.49.3.9 S	aveInstance(ModelElement *element)	196
		5.49.3.10 se	etName(std::string _name)	196
		5.49.3.11 sh	now()	197
	5.49.4	Member Date	ta Documentation	197
		5.49.4.1 <u>i</u>	id	197
		5.49.4.2 _r	name	197
		5.49.4.3 _t	typename	197
5.50	Modelli	nfo Class Ref	ference	197
	5.50.1	Detailed De	scription	198
	5.50.2	Constructor	& Destructor Documentation	198
		5.50.2.1 M	lodelInfo()	198
		5.50.2.2 M	lodelInfo(const ModelInfo &orig)	198
		5.50.2.3 ~	ModelInfo()	198

CONTENTS xxvii

5.50.3	Member F	Function Documentation	198
	5.50.3.1	getAnalystName() const	199
	5.50.3.2	getDescription() const	199
	5.50.3.3	getName() const	199
	5.50.3.4	getNumberOfReplications() const	200
	5.50.3.5	getProjectTitle() const	200
	5.50.3.6	getReplicationLength() const	200
	5.50.3.7	getReplicationLengthTimeUnit() const	201
	5.50.3.8	getTerminatingCondition() const	201
	5.50.3.9	getVersion() const	201
	5.50.3.10	getWarmUpPeriod() const	202
	5.50.3.11	getWarmUpPeriodTimeUnit() const	202
	5.50.3.12	loadInstance(std::map< std::string, std::string > *fields)	202
	5.50.3.13	saveInstance()	203
	5.50.3.14	setAnalystName(std::string _analystName)	204
	5.50.3.15	setDescription(std::string _description)	204
	5.50.3.16	setName(std::string _name)	204
	5.50.3.17	setNumberOfReplications(unsigned int _numberOfReplications)	204
	5.50.3.18	setProjectTitle(std::string _projectTitle)	205
	5.50.3.19	setReplicationLength(double _replicationLength)	205
	5.50.3.20	setReplicationLengthTimeUnit(Util::TimeUnit _replicationLengthTimeUnit)	205
	5.50.3.21	setTerminatingCondition(std::string _terminatingCondition)	206
	5.50.3.22	setVersion(std::string _version)	206
	5.50.3.23	setWarmUpPeriod(double _warmUpPeriod)	206
	5.50.3.24	setWarmUpPeriodTimeUnit(Util::TimeUnit _warmUpPeriodTimeUnit)	206
	5.50.3.25	show()	206
ModelN	Manager C	lass Reference	207
5.51.1	Construct	tor & Destructor Documentation	207
	5.51.1.1	ModelManager(Simulator *simulator)	207
	5.51.1.2	ModelManager(const ModelManager &orig)	207

5.51

xxviii CONTENTS

		5.51.1.3	~ModelManager()	. 207
	5.51.2	Member	Function Documentation	. 207
		5.51.2.1	current()	. 207
		5.51.2.2	end()	. 208
		5.51.2.3	front()	. 208
		5.51.2.4	insert(Model *model)	. 208
		5.51.2.5	loadModel(std::string filename)	. 209
		5.51.2.6	next()	. 209
		5.51.2.7	remove(Model *model)	. 210
		5.51.2.8	saveModel(std::string filename)	. 210
		5.51.2.9	setCurrent(Model *model)	. 210
5.52	ModelF	Persistence	e_if Class Reference	. 210
	5.52.1	Detailed	Description	. 211
	5.52.2	Member	Function Documentation	. 211
		5.52.2.1	isSaved()=0	. 211
		5.52.2.2	load(std::string filename)=0	. 211
		5.52.2.3	save(std::string filename)=0	. 212
5.53	ModelF	ersistence	eDefaultImpl1 Class Reference	. 212
	5.53.1	Construc	tor & Destructor Documentation	. 213
		5.53.1.1	ModelPersistenceDefaultImpl1(Model *model)	. 213
		5.53.1.2	ModelPersistenceDefaultImpl1(const ModelPersistenceDefaultImpl1 &orig)	. 213
		5.53.1.3	~ModelPersistenceDefaultImpl1()	. 213
	5.53.2	Member	Function Documentation	. 213
		5.53.2.1	isSaved()	. 213
		5.53.2.2	load(std::string filename)	. 214
		5.53.2.3	save(std::string filename)	. 214
5.54	ModelS	Simulation	Class Reference	. 215
	5.54.1	Detailed	Description	. 216
	5.54.2	Construc	tor & Destructor Documentation	. 216
		5.54.2.1	ModelSimulation(Model *model)	. 216

CONTENTS xxix

		5.54.2.2	ModelSimulation(const ModelSimulation &orig)	 	 	 217
		5.54.2.3	\sim ModelSimulation()	 	 	 217
	5.54.3	Member F	Function Documentation	 	 	 217
		5.54.3.1	getCurrentComponent() const	 	 	 217
		5.54.3.2	getCurrentEntity() const	 	 	 217
		5.54.3.3	getCurrentReplicationNumber() const	 	 	 217
		5.54.3.4	getSimulatedTime() const	 	 	 218
		5.54.3.5	getSimulationReporter() const	 	 	 218
		5.54.3.6	isInitializeStatistics() const	 	 	 218
		5.54.3.7	isInitializeSystem() const	 	 	 218
		5.54.3.8	isPauseOnEvent() const	 	 	 218
		5.54.3.9	isPauseOnReplication() const	 	 	 218
		5.54.3.10	isRunning() const	 	 	 218
		5.54.3.11	isStepByStep() const	 	 	 219
		5.54.3.12	pauseSimulation()	 	 	 219
		5.54.3.13	restartSimulation()	 	 	 219
		5.54.3.14	setInitializeStatistics(bool _initializeStatistics)	 	 	 219
		5.54.3.15	setInitializeSystem(bool _initializeSystem)	 	 	 219
		5.54.3.16	setPauseOnEvent(bool _pauseOnEvent)	 	 	 219
		5.54.3.17	$set Pause On Replication (bool_pause Between Replications)$	 	 	 219
		5.54.3.18	setStepByStep(bool _stepByStep)	 	 	 219
		5.54.3.19	startSimulation()	 	 	 219
		5.54.3.20	stepSimulation()	 	 	 219
		5.54.3.21	stopSimulation()	 	 	 220
5.55	МуАрр	Class Refe	erence	 	 	 220
	5.55.1	Construct	or & Destructor Documentation	 	 	 221
		5.55.1.1	MyApp()	 	 	 221
		5.55.1.2	MyApp(const MyApp &orig)	 	 	 221
		5.55.1.3	~MyApp()	 	 	 222
	5.55.2	Member F	Function Documentation	 	 	 222

CONTENTS

		5.55.2.1	main(int argc, char **argv)	222
5.56	Sample	erDummylı	mpl::MyRNG_Parameters Struct Reference	223
	5.56.1	Construc	tor & Destructor Documentation	224
		5.56.1.1	~MyRNG_Parameters()=default	224
	5.56.2	Member	Data Documentation	224
		5.56.2.1	module	224
		5.56.2.2	multiplier	224
		5.56.2.3	seed	224
5.57	OnEve	ntManage	r Class Reference	224
	5.57.1	Detailed	Description	225
	5.57.2	Construc	tor & Destructor Documentation	225
		5.57.2.1	OnEventManager()	225
		5.57.2.2	OnEventManager(const OnEventManager &orig)	225
		5.57.2.3	~OnEventManager()	225
	5.57.3	Member	Function Documentation	225
		5.57.3.1	addOnEntityRemoveHandler(simulationEventHandler EventHandler)	225
		5.57.3.2	addOnProcessEventHandler(simulationEventHandler EventHandler)	225
		5.57.3.3	addOnReplicationEndHandler(simulationEventHandler EventHandler)	225
		5.57.3.4	addOnReplicationStartHandler(simulationEventHandler EventHandler)	225
		5.57.3.5	addOnReplicationStepHandler(simulationEventHandler EventHandler)	225
		5.57.3.6	addOnSimulationEndHandler(simulationEventHandler EventHandler)	225
		5.57.3.7	addOnSimulationStartHandler(simulationEventHandler EventHandler)	225
		5.57.3.8	NotifyProcessEventHandlers(SimulationEvent *se)	225
		5.57.3.9	NotifyReplicationEndHandlers(SimulationEvent *se)	226
		5.57.3.10	NotifyReplicationStartHandlers(SimulationEvent *se)	226
		5.57.3.11	NotifyReplicationStepHandlers(SimulationEvent *se)	226
		5.57.3.12	NotifySimulationEndHandlers(SimulationEvent *se)	226
		5.57.3.13	NotifySimulationStartHandlers(SimulationEvent *se)	227
5.58	Parser_	_if Class R	deference	227
	5.58.1	Member	Function Documentation	227

CONTENTS xxxi

		5.58.1.1	getErrorMessage()=0	227
		5.58.1.2	parse(const std::string expression)=0	228
		5.58.1.3	parse(const std::string expression, bool *success, std::string *errorMessage)=0	228
5.59	Parserl	DefaultImp	I1 Class Reference	228
	5.59.1	Construc	tor & Destructor Documentation	229
		5.59.1.1	ParserDefaultImpl1(Model *model)	229
		5.59.1.2	ParserDefaultImpl1(const ParserDefaultImpl1 &orig)	229
		5.59.1.3	~ParserDefaultImpl1()	229
	5.59.2	Member I	Function Documentation	229
		5.59.2.1	getErrorMessage()	229
		5.59.2.2	parse(const std::string expression)	229
		5.59.2.3	parse(const std::string expression, bool *success, std::string *errorMessage)	230
5.60	Parserl	Dummylmp	ol Class Reference	230
	5.60.1	Construc	tor & Destructor Documentation	231
		5.60.1.1	ParserDummyImpl(Model *model)	231
		5.60.1.2	ParserDummyImpl(const ParserDummyImpl &orig)	231
		5.60.1.3	~ParserDummyImpl()	231
	5.60.2	Member I	Function Documentation	231
		5.60.2.1	getErrorMessage()	231
		5.60.2.2	parse(const std::string expression)	231
		5.60.2.3	parse(const std::string expression, bool *success, std::string *errorMessage)	232
5.61	Plugin	Class Refe	erence	232
	5.61.1	Detailed I	Description	232
	5.61.2	Construc	tor & Destructor Documentation	233
		5.61.2.1	Plugin(StaticGetPluginInformation getInformation)	233
		5.61.2.2	Plugin(const Plugin &orig)	233
		5.61.2.3	\sim Plugin()	233
	5.61.3	Member I	Function Documentation	233
		5.61.3.1	getPluginInfo() const	233
		5.61.3.2	isIsValidPlugin() const	233

xxxii CONTENTS

		5.61.3.3	loadAndInsertNew(Model *model, std::map< std::string, std::string > *fields)	234
		5.61.3.4	$loadNew(Model*model, std::map{< std::string, std::string} > *fields) \ . \ . \ . \ . \ . \ .$	234
5.62	Pluginl	nformation	Class Reference	235
	5.62.1	Construc	tor & Destructor Documentation	236
		5.62.1.1	PluginInformation(std::string pluginTypename, StaticLoaderComponentInstance componentloader)	236
		5.62.1.2	PluginInformation(std::string pluginTypename, StaticLoaderElementInstance elementloader)	236
	5.62.2	Member	Data Documentation	236
		5.62.2.1	author	236
		5.62.2.2	componentloader	236
		5.62.2.3	date	236
		5.62.2.4	dependencies	236
		5.62.2.5	elementloader	236
		5.62.2.6	generateReport	236
		5.62.2.7	isComponent	236
		5.62.2.8	isSink	236
		5.62.2.9	isSource	236
		5.62.2.10	observation	236
		5.62.2.11	pluginTypename	236
5.63	PluginN	Manager C	lass Reference	236
	5.63.1	Construc	tor & Destructor Documentation	237
		5.63.1.1	PluginManager(Simulator *simulator)	237
		5.63.1.2	PluginManager(const PluginManager &orig)	237
		5.63.1.3	~PluginManager()	237
	5.63.2	Member	Function Documentation	237
		5.63.2.1	find(std::string pluginTypeName)	237
		5.63.2.2	front()	238
		5.63.2.3	insert(Plugin *plugin)	238
		5.63.2.4	last()	239
		5.63.2.5	next()	239

CONTENTS xxxiii

	5.63.2.6	remove(Plugin *plugin)	239
5.64 ProbDi	istrib Class	Reference	239
5.64.1	Member I	Function Documentation	240
	5.64.1.1	_gammaFunction(double z)	240
	5.64.1.2	beta(double x, double alpha, double beta)	240
	5.64.1.3	chi2(double x, double m)	241
	5.64.1.4	erlang(double x, double mean, double M)	241
	5.64.1.5	exponential(double x, double mean)	241
	5.64.1.6	$fFisher(double\ x,\ double\ k,\ double\ m) \ \ .$	241
	5.64.1.7	gamma(double x, unsigned int alpha, double beta)	241
	5.64.1.8	inverseChi2(double cumulativeProbability, double m)	242
	5.64.1.9	$inverse FS nedecor (double\ cumulative Probability,\ double\ u,\ double\ v)\ .\ .\ .\ .\ .$	242
	5.64.1.10	inverseNormal(double cumulativeProbability, double mean, double stddev)	242
	5.64.1.11	inverseTStudent(double cumulativeProbability, double mean, double stddev, double degreeFreedom)	242
	5.64.1.12	logNormal(double x, double mean, double stddev)	242
	5.64.1.13	normal(double x, double mean, double stddev)	242
	5.64.1.14	triangular(double x, double min, double mode, double max)	242
	5.64.1.15	$tStudent(double\ x,\ double\ mean,\ double\ stddev,\ unsigned\ int\ degreeFreedom) .$	242
	5.64.1.16	uniform(double x, double min, double max)	242
	5.64.1.17	weibull(double x, double alpha, double scale)	243
5.65 Proces	sAnalyser_	_if Class Reference	243
5.65.1	Detailed I	Description	243
5.65.2	Member I	Function Documentation	243
	5.65.2.1	$add Trace Simulation Handler (trace Simulation Process Listener \\ Process Listener) = 0 \\ \dots \\ \dots \\ \dots \\ \dots$	243
	5.65.2.2	extractControlsFromModel(std::string modelFilename) const =0	244
	5.65.2.3	extractResponsesFromModel(std::string modelFilename) const =0	244
	5.65.2.4	getControls() const =0	244
	5.65.2.5	getResponses() const =0	244
	5.65.2.6	getScenarios() const =0	244

CONTENTS

		5.65.2.7	startSimulation()=0	244
		5.65.2.8	startSimulationOfScenario(SimulationScenario *scenario)=0	244
		5.65.2.9	stopSimulation()=0	244
5.66	Proces	sAnalyserl	DefaultImpl1 Class Reference	245
	5.66.1	Construc	tor & Destructor Documentation	246
		5.66.1.1	ProcessAnalyserDefaultImpl1()	246
		5.66.1.2	ProcessAnalyserDefaultImpl1(const ProcessAnalyserDefaultImpl1 &orig)	246
		5.66.1.3	~ProcessAnalyserDefaultImpl1()	246
	5.66.2	Member I	Function Documentation	246
		5.66.2.1	$add Trace Simulation Handler (trace Simulation Process Listener \\ Process Listener) \\ \ldots \\ \ldots \\ \ldots \\ \ldots \\ \ldots$	246
		5.66.2.2	extractControlsFromModel(std::string modelFilename) const	246
		5.66.2.3	extractResponsesFromModel(std::string modelFilename) const	246
		5.66.2.4	getControls() const	246
		5.66.2.5	getResponses() const	246
		5.66.2.6	getScenarios() const	246
		5.66.2.7	startSimulation()	246
		5.66.2.8	startSimulationOfScenario(SimulationScenario *scenario)	247
		5.66.2.9	stopSimulation()	247
5.67	Proces	sAnalyserl	DummyImpl Class Reference	247
	5.67.1	Construc	tor & Destructor Documentation	248
		5.67.1.1	ProcessAnalyserDummyImpl()	248
		5.67.1.2	ProcessAnalyserDummyImpl(const ProcessAnalyserDummyImpl &orig)	248
		5.67.1.3	~ProcessAnalyserDummyImpl()	248
	5.67.2	Member I	Function Documentation	248
		5.67.2.1	$add Trace Simulation Handler (trace Simulation Process Listener \\ Process Listener) \\ \dots \\ $	248
		5.67.2.2	extractControlsFromModel(std::string modelFilename) const	248
		5.67.2.3	extractResponsesFromModel(std::string modelFilename) const	248
		5.67.2.4	getControls() const	248
		5.67.2.5	getResponses() const	248

CONTENTS XXXV

		5.67.2.6	getScenarios() const	249
		5.67.2.7	startSimulation()	249
		5.67.2.8	startSimulationOfScenario(SimulationScenario *scenario)	249
		5.67.2.9	stopSimulation()	249
5.68	Queue	Class Ref	erence	249
	5.68.1	Member	Enumeration Documentation	251
		5.68.1.1	OrderRule	251
	5.68.2	Construc	tor & Destructor Documentation	251
		5.68.2.1	Queue(ElementManager *elems)	251
		5.68.2.2	Queue(ElementManager *elems, std::string name)	251
		5.68.2.3	Queue(const Queue &orig)	252
		5.68.2.4	~Queue()	252
	5.68.3	Member	Function Documentation	252
		5.68.3.1	_check(std::string *errorMessage)	252
		5.68.3.2	_loadInstance(std::map< std::string, std::string > *fields)	252
		5.68.3.3	_saveInstance()	253
		5.68.3.4	first()	253
		5.68.3.5	getAttributeName() const	253
		5.68.3.6	getOrderRule() const	253
		5.68.3.7	GetPluginInformation()	254
		5.68.3.8	initBetweenReplication()	254
		5.68.3.9	insertElement(Waiting *element)	255
		5.68.3.10	LoadInstance(ElementManager *elems, std::map< std::string, std::string > *fields)	255
		5.68.3.11	removeElement(Waiting *element, double tnow)	256
		5.68.3.12	setAttributeName(std::string _attributeName)	257
		5.68.3.13	setOrderRule(OrderRule _orderRule)	257
		5.68.3.14	show()	257
		5.68.3.15	size()	257
5.69	Record	Class Re	ference	258
	5.69.1	Construc	tor & Destructor Documentation	260

xxxvi CONTENTS

		5.69.1.1 F	Record(Model *model)	 260
		5.69.1.2 F	Record(const Record &orig)	 260
		5.69.1.3 ~	~Record()	 260
5.6	69.2	Member Fu	nction Documentation	 261
		5.69.2.1	check(std::string *errorMessage)	 261
		5.69.2.2 _	execute(Entity *entity)	 261
		5.69.2.3	initBetweenReplications()	 261
		5.69.2.4 _	loadInstance(std::map< std::string, std::string > *fields)	 262
		5.69.2.5	saveInstance()	 262
		5.69.2.6 g	petCstatExpression() const	 262
		5.69.2.7 g	getExpression() const	 262
		5.69.2.8 g	getExpressionName() const	 262
		5.69.2.9 g	getFilename() const	 262
		5.69.2.10	GetPluginInformation()	 263
		5.69.2.11 L	oadInstance(Model *model, std::map< std::string, std::string > *fields)	 263
		5.69.2.12 s	etExpression(std::string expression)	 264
		5.69.2.13 s	etExpressionName(std::string expressionName)	 264
		5.69.2.14 s	etFilename(std::string filename)	 264
		5.69.2.15 s	how()	 265
5.70 Re	elease	Class Refe	erence	 265
5.7	70.1	Constructo	r & Destructor Documentation	 267
		5.70.1.1 F	Release(Model *model)	 267
		5.70.1.2 F	Release(const Release &orig)	 267
		5.70.1.3 ~	~Release()	 267
5.7	70.2	Member Fu	nction Documentation	 267
		5.70.2.1 _	check(std::string *errorMessage)	 267
		5.70.2.2 _	execute(Entity *entity)	 267
		5.70.2.3 _	initBetweenReplications()	 268
		5.70.2.4 _	loadInstance(std::map< std::string, std::string > *fields)	 268
		5.70.2.5 _	saveInstance()	 269

CONTENTS xxxvii

	5.70.2.6 GetPluginInformation()
	5.70.2.7 getPriority() const
	5.70.2.8 getQuantity() const
	5.70.2.9 getResource() const
	5.70.2.10 getResourceName() const
	5.70.2.11 getResourceType() const
	5.70.2.12 getRule() const
	5.70.2.13 getSaveAttribute() const
	5.70.2.14 LoadInstance(Model *model, std::map< std::string, std::string > *fields) 27
	5.70.2.15 setPriority(unsigned short _priority)
	5.70.2.16 setQuantity(std::string _quantity)
	5.70.2.17 setResource(Resource *_resource)
	5.70.2.18 setResourceName(std::string resourceName)
	5.70.2.19 setResourceType(Resource::ResourceType _resourceType)
	5.70.2.20 setRule(Resource::ResourceRule _rule)
	5.70.2.21 setSaveAttribute(std::string _saveAttribute)
	5.70.2.22 show()
Resour	rce Class Reference
5.71.1	Detailed Description
5.71.2	Member Typedef Documentation
	5.71.2.1 ResourceEventHandler
5.71.3	Member Enumeration Documentation
	5.71.3.1 ResourceRule
	5.71.3.2 ResourceState
	5.71.3.3 ResourceType
5.71.4	Constructor & Destructor Documentation
	5.71.4.1 Resource(ElementManager *elems)
	5.71.4.2 Resource(ElementManager *elems, std::string name)
	5.71.4.3 Resource(const Resource &orig)
	5.71.4.4 ~Resource()
	5.71.1 5.71.2 5.71.3

xxxviii CONTENTS

	5.71.5	Member Function Documentation	75
		5.71.5.1 _check(std::string *errorMessage)	⁷ 5
		5.71.5.2 _loadInstance(std::map< std::string, std::string > *fields)	⁷ 5
		5.71.5.3 _saveInstance()	⁷ 6
		5.71.5.4 addResourceEventHandler(ResourceEventHandler eventHandler) 27	⁷ 6
		5.71.5.5 getCapacity() const	77
		5.71.5.6 getCostBusyHour() const	77
		5.71.5.7 getCostIdleHour() const	77
		5.71.5.8 getCostPerUse() const	77
		5.71.5.9 getLastTimeSeized() const	77
		5.71.5.10 getNumberBusy() const	78
		5.71.5.11 GetPluginInformation()	78
		5.71.5.12 getResourceState() const	78
		5.71.5.13 initBetweenReplications()	79
		5.71.5.14 LoadInstance(ElementManager *elems, std::map< std::string, std::string > *fields)27	79
		5.71.5.15 release(unsigned int quantity, double tnow)	30
		5.71.5.16 seize(unsigned int quantity, double tnow)	30
		5.71.5.17 setCapacity(unsigned int _capacity)	31
		5.71.5.18 setCostBusyHour(double _costBusyHour)	31
		5.71.5.19 setCostIdleHour(double _costIdleHour)	31
		5.71.5.20 setCostPerUse(double _costPerUse)	31
		5.71.5.21 SetResourceEventHandler(void(Class::*function)(Resource *), Class *object) . 28	31
		5.71.5.22 setResourceState(ResourceState _resourceState)	31
		5.71.5.23 show()	31
5.72	Sample	r_if::RNG_Parameters Struct Reference	32
	5.72.1	Detailed Description	32
	5.72.2	Constructor & Destructor Documentation	32
		5.72.2.1 ~RNG_Parameters()=default	32
5.73	Sample	r_if Class Reference	33
	5.73.1	Detailed Description	33

CONTENTS xxxix

	5.73.2	Member I	Function Documentation	284
		5.73.2.1	getRNGparameters() const =0	284
		5.73.2.2	random()=0	284
		5.73.2.3	sampleBeta(double alpha, double beta, double infLimit, double supLimit)=0	284
		5.73.2.4	sampleDiscrete(double value, double acumProb,)=0	284
		5.73.2.5	sampleErlang(double mean, int M)=0	284
		5.73.2.6	sampleExponential(double mean)=0	285
		5.73.2.7	sampleGamma(double mean, double alpha)=0	285
		5.73.2.8	sampleLogNormal(double mean, double stddev)=0	285
		5.73.2.9	sampleNormal(double mean, double stddev)=0	285
		5.73.2.10	sampleTriangular(double min, double mode, double max)=0	285
		5.73.2.11	sampleUniform(double min, double max)=0	286
		5.73.2.12	sampleWeibull(double alpha, double scale)=0	286
		5.73.2.13	setRNGparameters(RNG_Parameters *param)=0	286
5.74	Sample	erDefaultIn	npl1 Class Reference	286
	5.74.1	Construc	tor & Destructor Documentation	287
		5.74.1.1	SamplerDefaultImpl1()	288
		5.74.1.2	SamplerDefaultImpl1(const SamplerDefaultImpl1 &orig)	288
		5.74.1.3	~SamplerDefaultImpl1()	288
	5.74.2	Member I	Function Documentation	288
		5.74.2.1	getRNGparameters() const	288
		5.74.2.2	random()	288
		5.74.2.3	reset()	289
		5.74.2.4	sampleBeta(double alpha, double beta, double infLimit, double supLimit)	289
		5.74.2.5	sampleDiscrete(double value, double acumProb,)	289
		5.74.2.6	sampleErlang(double mean, int M)	289
		5.74.2.7	sampleExponential(double mean)	289
		5.74.2.8	sampleGamma(double mean, double alpha)	290
		5.74.2.9	sampleLogNormal(double mean, double stddev)	290
		5.74.2.10	sampleNormal(double mean, double stddev)	290

xI CONTENTS

		5.74.2.11 sampleTriangular(double min, double mode, double max)	90
		5.74.2.12 sampleUniform(double min, double max)	90
		5.74.2.13 sampleWeibull(double alpha, double scale)	91
		5.74.2.14 setRNGparameters(RNG_Parameters *param)	91
5.75	Sample	erDummyImpl Class Reference	91
	5.75.1	Constructor & Destructor Documentation	92
		5.75.1.1 SamplerDummyImpl()	92
		5.75.1.2 SamplerDummyImpl(const SamplerDummyImpl &orig)	92
		5.75.1.3 ~SamplerDummyImpI()	92
	5.75.2	Member Function Documentation	92
		5.75.2.1 getRNGparameters() const	92
		5.75.2.2 random()	92
		5.75.2.3 sampleBeta(double alpha, double beta, double infLimit, double supLimit) 29	92
		5.75.2.4 sampleDiscrete(double value, double acumProb,)	93
		5.75.2.5 sampleErlang(double mean, int M)	93
		5.75.2.6 sampleExponential(double mean)	93
		5.75.2.7 sampleGamma(double mean, double alpha)	93
		5.75.2.8 sampleLogNormal(double mean, double stddev)	93
		5.75.2.9 sampleNormal(double mean, double stddev)	93
		5.75.2.10 sampleTriangular(double min, double mode, double max)	93
		5.75.2.11 sampleUniform(double min, double max)	93
		5.75.2.12 sampleWeibull(double alpha, double scale)	93
		5.75.2.13 setRNGparameters(RNG_Parameters *param)	93
5.76	Scenar	rioExperiment_if Class Reference	94
5.77	Seize C	Class Reference	94
	5.77.1	Detailed Description	95
	5.77.2	Constructor & Destructor Documentation	96
		5.77.2.1 Seize(Model *model)	96
		5.77.2.2 Seize(const Seize &orig)	96
		5.77.2.3 ~Seize()	96

CONTENTS xli

5.77.3	Member Function Documentation	296
	5.77.3.1 _check(std::string *errorMessage)	296
	5.77.3.2 _execute(Entity *entity)	296
	5.77.3.3 _initBetweenReplications()	297
	5.77.3.4 _loadInstance(std::map< std::string, std::string > *fields)	297
	5.77.3.5 _saveInstance()	298
	5.77.3.6 getAllocationType() const	298
	5.77.3.7 getLastMemberSeized() const	298
	5.77.3.8 GetPluginInformation()	299
	5.77.3.9 getPriority() const	299
	5.77.3.10 getQuantity() const	299
	5.77.3.11 getQueue() const	299
	5.77.3.12 getQueueName() const	299
	5.77.3.13 getResource() const	300
	5.77.3.14 getResourceName() const	300
	5.77.3.15 getResourceType() const	300
	5.77.3.16 getRule() const	300
	5.77.3.17 getSaveAttribute() const	300
	5.77.3.18 LoadInstance(Model *model, std::map< std::string, std::string > *fields)	300
	5.77.3.19 setAllocationType(unsigned int _allocationType)	301
	5.77.3.20 setLastMemberSeized(unsigned int _lastMemberSeized)	301
	5.77.3.21 setPriority(unsigned short _priority)	301
	5.77.3.22 setQuantity(std::string _quantity)	301
	5.77.3.23 setQueue(Queue *queue)	301
	5.77.3.24 setQueueName(std::string queueName)	301
	5.77.3.25 setResource(Resource *resource)	302
	5.77.3.26 setResourceName(std::string _resourceName)	302
	5.77.3.27 setResourceType(Resource::ResourceType _resourceType)	302
	5.77.3.28 setRule(Resource::ResourceRule _rule)	302
	5.77.3.29 setSaveAttribute(std::string _saveAttribute)	302

xlii CONTENTS

		5.77.3.30	show()	302
5.78	Simula	tionContro	I Class Reference	303
	5.78.1	Detailed I	Description	304
	5.78.2	Construc	tor & Destructor Documentation	304
		5.78.2.1	SimulationControl(std::string type, std::string name, GetterMember getter← Member, SetterMember setterMember)	304
		5.78.2.2	SimulationControl(const SimulationControl &orig)	304
		5.78.2.3	~SimulationControl()	304
	5.78.3	Member I	Function Documentation	304
		5.78.3.1	setValue(double value)	304
5.79	Simula	tionEvent (Class Reference	304
	5.79.1	Construc	tor & Destructor Documentation	305
		5.79.1.1	SimulationEvent(unsigned int replicationNumber, Event *event)	305
	5.79.2	Member I	Function Documentation	305
		5.79.2.1	getEventProcessed() const	305
		5.79.2.2	getReplicationNumber() const	305
5.80	Simula	tionReport	er_if Class Reference	306
	5.80.1	Member I	Function Documentation	306
		5.80.1.1	showReplicationStatistics()=0	306
		5.80.1.2	showSimulationStatistics()=0	307
5.81	Simula	tionReport	erDefaultImpl1 Class Reference	307
	5.81.1	Detailed I	Description	308
	5.81.2	Construc	tor & Destructor Documentation	308
		5.81.2.1	lem:lem:lem:lem:lem:lem:lem:lem:lem:lem:	308
		5.81.2.2	SimulationReporterDefaultImpl1(const SimulationReporterDefaultImpl1 &orig)	308
		5.81.2.3	~SimulationReporterDefaultImpl1()	308
	5.81.3	Member I	Function Documentation	308
		5.81.3.1	showReplicationStatistics()	308
		5.81.3.2	showSimulationStatistics()	309
5.82	Simula	tionRespo	nse Class Reference	311

CONTENTS xliii

	5.82.1	Detailed Description				
	5.82.2	Construc	tor & Destructor Documentation	312		
		5.82.2.1	SimulationResponse(std::string type, std::string name, GetterMember getter ← Member)	312		
		5.82.2.2	SimulationResponse(const SimulationResponse &orig)	312		
		5.82.2.3	~SimulationResponse()	312		
	5.82.3	Member	Function Documentation	312		
		5.82.3.1	getName() const	312		
		5.82.3.2	getType() const	312		
		5.82.3.3	getValue()	312		
	5.82.4	Member	Data Documentation	312		
		5.82.4.1	_getterMemberFunction	312		
		5.82.4.2	_name	312		
		5.82.4.3	_type	312		
5.83	Simula	tionScena	rio Class Reference	312		
	5.83.1	Detailed	Description	313		
	5.83.2	Construc	tor & Destructor Documentation	313		
		5.83.2.1	SimulationScenario()	313		
		5.83.2.2	SimulationScenario(const SimulationScenario &orig)	313		
		5.83.2.3	~SimulationScenario()	313		
	5.83.3	Member	Function Documentation	313		
		5.83.3.1	getControlValue(SimulationControl *control)	313		
		5.83.3.2	getControlValues() const	313		
		5.83.3.3	getModelFilename() const	313		
		5.83.3.4	getName() const	313		
		5.83.3.5	getResponseValue(SimulationResponse *value)	313		
		5.83.3.6	getResponseValues() const	313		
		5.83.3.7	setControlValue(SimulationControl *control, double value)	313		
		5.83.3.8	setModelFilename(std::string _modelFilename)	313		
		5.83.3.9	setName(std::string _name)	313		
5.84	Simula	tor Class F	Reference	313		

XIIV CONTENTS

	5.84.1	Detailed Description
	5.84.2	Constructor & Destructor Documentation
		5.84.2.1 Simulator()
		5.84.2.2 Simulator(const Simulator &orig)
		5.84.2.3 ~Simulator()
	5.84.3	Member Function Documentation
		5.84.3.1 getLicenceManager() const
		5.84.3.2 getModelManager() const
		5.84.3.3 getName() const
		5.84.3.4 getPluginManager() const
		5.84.3.5 getToolManager() const
		5.84.3.6 getTraceManager() const
		5.84.3.7 getVersion() const
5.85	SinkMo	delComponent Class Reference
	5.85.1	Detailed Description
	5.85.2	Constructor & Destructor Documentation
		5.85.2.1 SinkModelComponent(Model *model, std::string componentTypename) 31
		5.85.2.2 SinkModelComponent(const SinkModelComponent &orig)
		5.85.2.3 ~SinkModelComponent()
	5.85.3	Member Function Documentation
		5.85.3.1 _check(std::string *errorMessage)
		5.85.3.2 _initBetweenReplications()
		$5.85.3.3 \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ $
		5.85.3.4 _saveInstance()
		5.85.3.5 isCollectStatistics() const
		5.85.3.6 setCollectStatistics(bool _collectStatistics)
5.86	Source	ModelComponent Class Reference
	5.86.1	Detailed Description
	5.86.2	Constructor & Destructor Documentation
		5.86.2.1 SourceModelComponent(Model *model, std::string componentTypename) 32

CONTENTS xlv

	5.86.2.2 SourceModelComponent(const SourceModelComponent & orig)	323
	5.86.2.3 ~SourceModelComponent()	323
5.86.3	Member Function Documentation	323
	5.86.3.1 _check(std::string *errorMessage)	323
	5.86.3.2 _initBetweenReplications()	324
	5.86.3.3 _loadInstance(std::map< std::string, std::string > *fields)	324
	5.86.3.4 _saveInstance()	325
	5.86.3.5 getEntitiesCreated() const	325
	5.86.3.6 getEntitiesPerCreation() const	325
	5.86.3.7 getEntityType() const	326
	5.86.3.8 getFirstCreation() const	326
	5.86.3.9 getMaxCreations() const	326
	5.86.3.10 getTimeBetweenCreationsExpression() const	326
	5.86.3.11 getTimeUnit() const	326
	5.86.3.12 isCollectStatistics() const	326
	5.86.3.13 setCollectStatistics(bool _collectStatistics)	326
	5.86.3.14 setEntitiesCreated(unsigned int _entitiesCreated)	326
	5.86.3.15 setEntitiesPerCreation(unsigned int _entitiesPerCreation)	326
	5.86.3.16 setEntityType(EntityType *_entityType)	327
	5.86.3.17 setFirstCreation(double _firstCreation)	327
	5.86.3.18 setMaxCreations(std::string _maxCreationsExpression)	327
	5.86.3.19 setTimeBetweenCreationsExpression(std::string _timeBetweenCreations)	327
	5.86.3.20 setTimeUnit(Util::TimeUnit _timeUnit)	327
	5.86.3.21 show()	328
5.86.4	Member Data Documentation	328
	5.86.4.1 _entitiesCreatedSoFar	328
	5.86.4.2 _entitiesPerCreation	328
	5.86.4.3 _entityType	328
	5.86.4.4 _firstCreation	328
	5.86.4.5 _maxCreationsExpression	328

XIVI

		5.86.4.6	_timeBetweenCreationsExpression	328
		5.86.4.7	_timeBetweenCreationsTimeUnit	328
5.87	Statistic	cs_if Class	s Reference	329
	5.87.1	Detailed I	Description	329
	5.87.2	Member I	Function Documentation	330
		5.87.2.1	average()=0	330
		5.87.2.2	getCollector()=0	330
		5.87.2.3	getConfidenceLevel()=0	330
		5.87.2.4	halfWidthConfidenceInterval()=0	331
		5.87.2.5	max()=0	331
		5.87.2.6	min()=0	332
		5.87.2.7	newSampleSize(double halfWidth)=0	332
		5.87.2.8	numElements()=0	332
		5.87.2.9	setCollector(Collector_if *collector)=0	333
		5.87.2.10	setConfidenceLevel(double confidencelevel)=0	333
		5.87.2.11	stddeviation()=0	333
		5.87.2.12	? variance()=0	333
		5.87.2.13	3 variationCoef()=0	334
5.88	Statistic	csCollecto	r Class Reference	334
	5.88.1	Construc	tor & Destructor Documentation	335
		5.88.1.1	StatisticsCollector()	335
		5.88.1.2	StatisticsCollector(std::string name)	336
		5.88.1.3	StatisticsCollector(std::string name, ModelElement *parent)	336
		5.88.1.4	StatisticsCollector(const StatisticsCollector &orig)	336
		5.88.1.5	~StatisticsCollector()	336
	5.88.2	Member I	Function Documentation	336
		5.88.2.1	_check(std::string *errorMessage)	336
		5.88.2.2	_loadInstance(std::map< std::string, std::string > *fields)	336
		5.88.2.3	_saveInstance()	337
		5.88.2.4	getParent() const	337

CONTENTS xlvii

		5.88.2.5	GetPluginInformation()	337
		5.88.2.6	getStatistics() const	338
		5.88.2.7	LoadInstance(ElementManager *elems, std::map< std::string, std::string > *fields	s)338
		5.88.2.8	show()	339
5.89	Statistic	csDatafile_	_if Class Reference	339
	5.89.1	Member	Function Documentation	340
		5.89.1.1	centil(unsigned short num)=0	340
		5.89.1.2	decil(unsigned short num)=0	341
		5.89.1.3	histogramClassFrequency(unsigned short classNum)=0	341
		5.89.1.4	histogramClassLowerLimit(unsigned short classNum)=0	341
		5.89.1.5	histogramNumClasses()=0	341
		5.89.1.6	mediane()=0	341
		5.89.1.7	mode()=0	341
		5.89.1.8	quartil(unsigned short num)=0	341
		5.89.1.9	setHistogramNumClasses(unsigned short num)=0	341
5.90	Statistic	csDataFile	DummyImpl Class Reference	342
	5.90.1	Construc	tor & Destructor Documentation	343
		5.90.1.1	StatisticsDataFileDummyImpl()	343
		5.90.1.2	StatisticsDataFileDummyImpl(const StatisticsDataFileDummyImpl &orig)	343
		5.90.1.3	~StatisticsDataFileDummyImpl()	343
	5.90.2	Member	Function Documentation	343
		5.90.2.1	average()	343
		5.90.2.2	centil(unsigned short num)	343
		5.90.2.3	decil(unsigned short num)	343
		5.90.2.4	getCollector()	343
		5.90.2.5	halfWidthConfidenceInterval(double confidencelevel)	344
		5.90.2.6	histogramClassFrequency(unsigned short classNum)	344
		5.90.2.7	histogramClassLowerLimit(unsigned short classNum)	344
		5.90.2.8	histogramNumClasses()	344
		5.90.2.9	max()	344

xlviii CONTENTS

	5.90.2.10 media	ane()				 	 344
	5.90.2.11 min()					 	 344
	5.90.2.12 mode	()				 	 344
	5.90.2.13 newS	ampleSize(double con	fidencelevel, dou	uble halfWid	th)	 	 344
	5.90.2.14 numE	lements()				 	 344
	5.90.2.15 quarti	I(unsigned short num)				 	 344
	5.90.2.16 setCo	llector(Collector_if *co	ollector)			 	 345
	5.90.2.17 setHis	stogramNumClasses(u	ınsigned short กเ	um)		 	 345
	5.90.2.18 stdde	viation()				 	 345
	5.90.2.19 variar	nce()				 	 345
	5.90.2.20 variat	ionCoef()				 	 345
5.91 Statisti	csDefaultImpl1 C	lass Reference				 	 345
5.91.1	Constructor & D	estructor Documentati	on			 	 346
	5.91.1.1 Statis	ticsDefaultImpl1()				 	 346
	5.91.1.2 Statis	ticsDefaultImpl1(Collec	ctor_if *collector)		 	 347
	5.91.1.3 Statis	ticsDefaultImpl1(const	StatisticsDefault	tlmpl1 &orig)	 	 347
	5.91.1.4 ∼Stat	tisticsDefaultImpl1() .				 	 348
5.91.2	Member Function	on Documentation				 	 348
	5.91.2.1 avera	ge()				 	 348
	5.91.2.2 getCo	ollector()				 	 348
	5.91.2.3 getCo	onfidenceLevel()				 	 348
	5.91.2.4 halfW	idthConfidenceInterva	l()			 	 348
	5.91.2.5 max()					 	 349
	5.91.2.6 min()					 	 349
	5.91.2.7 newS	ampleSize(double half	Width)			 	 349
	5.91.2.8 numE	lements()				 	 349
	5.91.2.9 setCo	llector(Collector_if *co	ollector)			 	 349
	5.91.2.10 setCo	nfidenceLevel(double	confidencelevel)			 	 349
	5.91.2.11 stdde	viation()				 	 349
	5.91.2.12 variar	nce()				 	 350

CONTENTS xlix

		5.91.2.13 \	variationCoef()	 	 . 350
5.92	Statistic	csDummyIm	npl Class Reference	 	 . 350
	5.92.1	Constructo	r & Destructor Documentation	 	 . 351
		5.92.1.1	StatisticsDummyImpI()	 	 . 351
		5.92.1.2	StatisticsDummyImpl(const StatisticsDummyImpl &orig)	 	 . 351
		5.92.1.3	~StatisticsDummyImpl()	 	 . 351
	5.92.2	Member Fu	unction Documentation	 	 . 351
		5.92.2.1	average()	 	 . 351
		5.92.2.2	getCollector()	 	 . 352
		5.92.2.3	getConfidenceLevel()	 	 . 352
		5.92.2.4 h	nalfWidthConfidenceInterval()	 	 . 352
		5.92.2.5 r	max()	 	 . 352
		5.92.2.6 r	min()	 	 . 352
		5.92.2.7 r	newSampleSize(double halfWidth)	 	 . 352
		5.92.2.8 r	numElements()	 	 . 352
		5.92.2.9	setCollector(Collector_if *collector)	 	 . 352
		5.92.2.10	setConfidenceLevel(double confidencelevel)	 	 . 352
		5.92.2.11	stddeviation()	 	 . 352
		5.92.2.12 v	variance()	 	 . 353
		5.92.2.13 v	variationCoef()	 	 . 353
5.93	TestInp	utAnalyser1	Tools Class Reference	 	 . 353
	5.93.1	Constructo	r & Destructor Documentation	 	 . 354
		5.93.1.1	TestInputAnalyserTools()	 	 . 354
	5.93.2	Member Fu	unction Documentation	 	 . 354
		5.93.2.1 r	main(int argc, char **argv)	 	 . 354
5.94	TestPar	ser Class F	Reference	 	 . 356
	5.94.1	Constructo	r & Destructor Documentation	 	 . 356
		5.94.1.1	TestParser()	 	 . 356
			TestParser(const TestParser &orig)		
			∼TestParser()		
			· · · · · · · · · · · · · · · · · · ·		

I CONTENTS

	5.94.2	Member Function Documentation	357
		5.94.2.1 main(int argc, char **argv)	357
5.95	TestSta	atistics Class Reference	357
	5.95.1	Constructor & Destructor Documentation	358
		5.95.1.1 TestStatistics()	358
	5.95.2	Member Function Documentation	358
		5.95.2.1 main(int argc, char **argv)	358
5.96	ToolMa	nager Class Reference	359
	5.96.1	Constructor & Destructor Documentation	360
		5.96.1.1 ToolManager(Simulator *_simulator)	360
		5.96.1.2 ToolManager(const ToolManager & orig)	360
		5.96.1.3 ~ToolManager()	360
	5.96.2	Member Function Documentation	360
		5.96.2.1 getFitter() const	360
		5.96.2.2 getSampler() const	360
5.97	TraceE	rrorEvent Class Reference	361
	5.97.1	Constructor & Destructor Documentation	361
		5.97.1.1 TraceErrorEvent(std::string text, std::exception e)	361
	5.97.2	Member Function Documentation	361
		5.97.2.1 getException() const	361
5.98	TraceE	vent Class Reference	362
	5.98.1	Constructor & Destructor Documentation	362
		5.98.1.1 TraceEvent(Util::TraceLevel tracelevel, std::string text)	362
	5.98.2	Member Function Documentation	362
		5.98.2.1 getText() const	362
		5.98.2.2 getTracelevel() const	362
5.99	TraceM	lanager Class Reference	363
	5.99.1	Detailed Description	363
	5.99.2	Constructor & Destructor Documentation	363
		5.99.2.1 TraceManager(Simulator *simulator)	363

5	5.99.2.2	TraceManager(const TraceManager &orig)	363
Ę	5.99.2.3	~TraceManager()	363
5.99.3 N	Member F	Function Documentation	363
Ę	5.99.3.1	addTraceErrorHandler(traceErrorListener traceErrorListener)	363
Ę	5.99.3.2	addTraceHandler(traceListener traceListener)	363
Ę	5.99.3.3	addTraceReportHandler(traceListener traceReportListener)	364
Ę	5.99.3.4	$add Trace Simulation Handler (trace Simulation Listener\ trace Simulation Listener)\ .\ .$	364
Ę	5.99.3.5	getErrorMessages() const	364
Ę	5.99.3.6	getSimulator() const	364
Ę	5.99.3.7	getTraceLevel() const	364
Ę	5.99.3.8	setTraceLevel(Util::TraceLevel _traceLevel)	364
Ę	5.99.3.9	trace(Util::TraceLevel tracelevel, std::string text)	365
Ę	5.99.3.10	traceError(std::exception e, std::string text)	366
Ę	5.99.3.11	traceReport(Util::TraceLevel tracelevel, std::string text)	366
Ę		traceSimulation(Util::TraceLevel tracelevel, double time, Entity ∗entity, Model ← Component ∗component, std::string text)	367
5.100TraceSim	nulationEv	vent Class Reference	367
5.100.1	Constructo	or & Destructor Documentation	368
Ę		TraceSimulationEvent(Util::TraceLevel tracelevel, double time, Entity *entity, ModelComponent *component, std::string text)	368
5.100.2 N	Member F	Function Documentation	368
5	5.100.2.1	getComponent() const	368
Ę	5.100.2.2	getEntity() const	368
Ę	5.100.2.3	getTime() const	368
5.101TraceSim	nulationPr	rocess Class Reference	369
5.101.1 [Detailed D	Description	369
5.101.20	Construct	or & Destructor Documentation	369
Ę	5.101.2.1	TraceSimulationProcess(Util::TraceLevel tracelevel, std::string text)	369
5.102Traits < 1	Γ > Struct	t Template Reference	370
5.103Traits < 0	Collector_	if > Struct Template Reference	370
5.103.1 N	Member T	ypedef Documentation	370

lii CONTENTS

5.103.1.1 Implementation
5.104Traits < ExperimentDesign_if > Struct Template Reference
5.104.1 Member Typedef Documentation
5.104.1.1 Implementation
5.105Traits < Fitter_if > Struct Template Reference
5.105.1 Member Typedef Documentation
5.105.1.1 Implementation
5.106Traits < GenesysApplication_if > Struct Template Reference
5.106.1 Member Typedef Documentation
5.106.1.1 Application
5.107Traits < HypothesisTester_if > Struct Template Reference
5.107.1 Member Typedef Documentation
5.107.1.1 Implementation
5.107.1.2 IntegratorImplementation
5.108Traits < Integrator_if > Struct Template Reference
5.108.1 Member Typedef Documentation
5.108.1.1 Implementation
5.108.2 Member Data Documentation
5.108.2.1 MaxIterations
5.108.2.2 Precision
5.109Traits < Model > Struct Template Reference
5.109.1 Member Data Documentation
5.109.1.1 debugged
5.109.1.2 traceLevel
5.110Traits < ModelChecker_if > Struct Template Reference
5.110.1 Member Typedef Documentation
5.110.1.1 Implementation
5.111Traits < ModelComponent > Struct Template Reference
5.111.1 Member Typedef Documentation
5.111.1.1 StatisticsCollector_CollectorImplementation

5.111.1.2 StatisticsCollector_StatisticsImplementation
5.112Traits < ModelPersistence_if > Struct Template Reference
5.112.1 Member Typedef Documentation
5.112.1.1 Implementation
5.113Traits < Parser_if > Struct Template Reference
5.113.1 Member Typedef Documentation
5.113.1.1 Implementation
5.114Traits < ProcessAnalyser_if > Struct Template Reference
5.114.1 Member Typedef Documentation
5.114.1.1 Implementation
5.115Traits < Sampler_if > Struct Template Reference
5.115.1 Member Typedef Documentation
5.115.1.1 Implementation
5.115.1.2 Parameters
5.116Traits < SimulationReporter_if > Struct Template Reference
5.116.1 Member Typedef Documentation
5.116.1.1 Implementation
5.117Traits < Statistics_if > Struct Template Reference
5.117.1 Member Typedef Documentation
5.117.1.1 CollectorImplementation
5.117.1.2 Implementation
5.117.2 Member Data Documentation
5.117.2.1 SignificanceLevel
5.118Util Class Reference
5.118.1 Member Typedef Documentation
5.118.1.1 identitifcation
5.118.1.2 rank
5.118.2 Member Enumeration Documentation
5.118.2.1 TimeUnit
5.118.2.2 TraceLevel

liv CONTENTS

5.118.3 Member Function Documentation	8
5.118.3.1 DecIndent()	⁷ 8
5.118.3.2 GenerateNewId()	⁷ 8
5.118.3.3 GenerateNewIdOfType(std::string objtype)	79
5.118.3.4 GenerateNewIdOfType()	79
5.118.3.5 GetLastIdOfType(std::string objtype)	79
5.118.3.6 Inclndent()	79
5.118.3.7 Indent()	30
5.118.3.8 ResetIdOfType(std::string objtype)	30
5.118.3.9 SepKeyVal(std::string str, std::string *key, std::string *value)	30
5.118.3.10SetIndent(const unsigned short indent)	30
5.118.3.11SetW(std::string text, unsigned short width)	31
5.118.3.12StrTimeUnit(Util::TimeUnit timeUnit)	31
5.118.3.13TimeUnitConvert(Util::TimeUnit timeUnit1, Util::TimeUnit timeUnit2) 38	31
5.118.3.14TypeOf()	31
5.118.4 Member Data Documentation	32
5.118.4.1 _S_indentation	32
5.119 Variable Class Reference	32
5.119.1 Constructor & Destructor Documentation	33
5.119.1.1 Variable()	33
5.119.1.2 Variable(std::string name)	33
5.119.1.3 Variable(const Variable &orig)	33
5.119.1.4 ~Variable()	33
5.119.2 Member Function Documentation	33
5.119.2.1 _check(std::string *errorMessage)	33
5.119.2.2 _loadInstance(std::map< std::string, std::string > *fields)	34
5.119.2.3 _saveInstance()	34
5.119.2.4 GetPluginInformation()	34
5.119.2.5 getValue()	35
5.119.2.6 getValue(std::string index)	35

5.119.2.7 LoadInstance(ElementManager *elems, std::map< std::string, std::string > *fields)3	85
5.119.2.8 setValue(double value)	85
5.119.2.9 setValue(std::string index, double value)	86
5.119.2.10show()	86
5.120 Waiting Class Reference	86
5.120.1 Constructor & Destructor Documentation	87
5.120.1.1 Waiting(Entity *entity, ModelComponent *component, double timeStartedWaiting) 3	87
5.120.1.2 Waiting(const Waiting &orig)	87
5.120.1.3 ~Waiting()	87
5.120.2 Member Function Documentation	87
5.120.2.1 getComponent() const	87
5.120.2.2 getEntity() const	87
5.120.2.3 getTimeStartedWaiting() const	87
5.120.2.4 show()	87
5.121 Waiting Resource Class Reference	88
5.121.1 Constructor & Destructor Documentation	89
5.121.1.1 WaitingResource(Entity ∗entity, ModelComponent ∗component, double time⇔ StartedWaiting, unsigned int quantity)	89
5.121.1.2 WaitingResource(const WaitingResource &orig)	89
5.121.1.3 ~WaitingResource()	89
5.121.2 Member Function Documentation	89
5.121.2.1 getQuantity() const	89
5.121.2.2 show()	89

lvi CONTENTS

6	File I	Docum	entation												391
	6.1	.dep.in	c File Refe	rence											 391
	6.2	Assign	.cpp File R	eference											 391
	6.3	Assign	.h File Refe	erence											 391
	6.4	Attribu	te.cpp File	Reference											 392
	6.5	Attribu	te.h File Re	eference .											 393
	6.6	BuildS	imulationM	odel.cpp Fi	le Refere	ence									 394
	6.7	BuildS	imulationM	odel.h File	Reference	ce									 394
	6.8	Collect	or_if.h File	Reference											 395
		6.8.1	Typedef E	Oocumentat	ion										 397
			6.8.1.1	CollectorA	.ddValue	Handler	·								 397
			6.8.1.2	CollectorC	learHan	dler									 397
		6.8.2	Function	Documenta	ition										 397
			6.8.2.1	SetCollect	orAddVa	alueHand	dler(voi	d(Class	s::*func	tion)(d	ouble)	, Class	s *ob	ject)	 397
			6.8.2.2	SetCollect	orClearH	Handler(void(Cl	ass::*fı	unction)	(), Cla	ss *ob	ject)			 397
	6.9	Collect	orDatafile_	if.h File Re	ference										 398
	6.10	Collect	:orDatafileE	DefaultImpl1	Lcpp File	e Refere	ence								 398
	6.11	Collect	:orDatafileE	DefaultImpl1	I.h File F	Referenc	e								 399
	6.12	Collect	:orDatafile[DummyImpl	.cpp File	Referei	nce								 400
	6.13	Collect	:orDatafileE	DummyImpl	.h File R	leference	е								 401
	6.14	Collect	orDefaultIn	npl1.cpp Fi	le Refere	ence									 402
	6.15	Collect	orDefaultIn	npl1.h File	Reference	ce									 403
	6.16	Collect	orDummyl	mpl.cpp File	e Refere	nce									 404
	6.17	Collect	orDummyl	mpl.h File F	Referenc	e									 404
	6.18	Compo	onentMana	ger.cpp File	Referer	nce									 405
	6.19	Compo	onentMana	ger.h File R	eference	э									 406
	6.20	Counte	er.cpp File I	Reference											 407
	6.21	Counte	er.h File Re	ference .											 407
	6.22	Create	.cpp File R	eference											 408
	6.23	Create	.h File Refe	erence											 409

6.24	Decide	.cpp File F	Reference	0
6.25	Decide	.h File Ref	ference	0
6.26	Define	GetterSette	er.h File Reference	1
	6.26.1	Typedef [Documentation	3
		6.26.1.1	GetterMember	3
		6.26.1.2	SetterMember	3
	6.26.2	Function	Documentation	3
		6.26.2.1	DefineGetterMember(Class *object, double(Class::*function)()) 41	3
		6.26.2.2	DefineGetterMember(Class *object, unsigned int(Class::*function)() const) 41	3
		6.26.2.3	DefineGetterMember(Class *object, bool(Class::*function)() const) 41	3
		6.26.2.4	DefineGetterMember(Class *object, std::string(Class::*function)() const) 41	3
		6.26.2.5	DefineGetterMember(Class *object, Util::TimeUnit(Class::*function)() const) 41	3
		6.26.2.6	DefineSetterMember(Class *object, void(Class::*function)(double)) 41	3
		6.26.2.7	DefineSetterMember(Class *object, void(Class::*function)(unsigned int)) 41	3
		6.26.2.8	DefineSetterMember(Class *object, void(Class::*function)(bool)) 41	3
		6.26.2.9	DefineSetterMember(Class *object, void(Class::*function)(std::string) const) 41	3
		6.26.2.10	DefineSetterMember(Class *object, void(Class::*function)(Util::TimeUnit)) 41	3
6.27	Delay.c	pp File Re	eference	3
6.28	Delay.h	File Refe	rence 41	4
6.29	Dispos	e.cpp File	Reference	4
6.30	Dispos	e.h File Re	eference	5
6.31	Dummy	.cpp File I	Reference	6
6.32	Dummy	/.h File Re	ference	6
6.33	Elemer	ntManager	ccpp File Reference	7
6.34	Elemer	ntManager	th File Reference	8
6.35	Elemer	ntManager	_if.h File Reference	9
6.36	Entity.c	pp File Re	eference	9
6.37	Entity.h	File Refe	rence	9
6.38	EntityTy	ype.cpp Fi	le Reference	:0
6.39	EntityTy	ype.h File	Reference	:1

Iviii CONTENTS

6.40	Event.cpp File Reference	421
6.41	Event.h File Reference	422
6.42	ExperimentDesign_if.h File Reference	423
6.43	ExperimentDesignDefaultImpl1.cpp File Reference	424
6.44	ExperimentDesignDefaultImpl1.h File Reference	424
6.45	ExperimentDesignDummyImpl.cpp File Reference	425
6.46	ExperimentDesignDummyImpl.h File Reference	425
6.47	FactorOrInteractionContribution.cpp File Reference	426
6.48	FactorOrInteractionContribution.h File Reference	427
6.49	Fitter_if.h File Reference	428
6.50	FitterDefaultImpl1.cpp File Reference	429
6.51	FitterDefaultImpl1.h File Reference	430
6.52	FitterDummyImpl.cpp File Reference	431
6.53	FitterDummyImpl.h File Reference	431
6.54	Functor.h File Reference	432
6.55	GenesysApplication_if.h File Reference	432
6.56	GenesysConsole.cpp File Reference	433
6.57	GenesysConsole.h File Reference	433
6.58	GenesysGUI.cpp File Reference	434
6.59	GenesysGUI.h File Reference	434
6.60	GenesysShell_if.h File Reference	435
6.61	HypothesisTester_if.h File Reference	435
6.62	HypothesisTesterDefaultImpl1.cpp File Reference	436
6.63	HypothesisTesterDefaultImpl1.h File Reference	436
6.64	HypothesisTesterDummyImpl.cpp File Reference	437
6.65	HypothesisTesterDummyImpl.h File Reference	438
6.66	Integrator_if.h File Reference	439
6.67	IntegratorDefaultImpl1.cpp File Reference	439
6.68	IntegratorDefaultImpl1.h File Reference	440
6.69	IntegratorDummyImpl.cpp File Reference	440

6.70 IntegratorDummyImpl.h File Reference	41
6.71 LicenceManager.cpp File Reference	41
6.72 LicenceManager.h File Reference	42
6.73 LinkedBy.cpp File Reference	43
6.74 LinkedBy.h File Reference	43
6.75 List.h File Reference	43
6.76 main.cpp File Reference	44
6.76.1 Function Documentation	45
6.76.1.1 main(int argc, char **argv)	45
6.77 Model.cpp File Reference	45
6.77.1 Function Documentation	46
6.77.1.1 EventCompare(const Event *a, const Event *b)	46
6.77.1.2 getReplicationLengthNotMemberFunction()	46
6.77.1.3 setReplicationLengthNotMemberFunction(double value)	46
6.78 Model.h File Reference	46
6.79 ModelChecker_if.h File Reference	47
6.80 ModelCheckerDefaultImpl1.cpp File Reference	48
6.81 ModelCheckerDefaultImpl1.h File Reference	48
6.82 ModelComponent.cpp File Reference	49
6.83 ModelComponent.h File Reference	49
6.84 ModelComponentManager_if.h File Reference	50
6.85 ModelElement.cpp File Reference	50
6.86 ModelElement.h File Reference	50
6.87 ModelInfo.cpp File Reference	51
6.88 ModelInfo.h File Reference	52
6.89 ModelManager.cpp File Reference	53
6.90 ModelManager.h File Reference	53
6.91 ModelPersistence_if.h File Reference	54
6.92 ModelPersistenceDefaultImpl1.cpp File Reference	54
6.93 ModelPersistenceDefaultImpl1.h File Reference	55

IX

6.94 ModelSimulation.cpp File Reference
6.95 ModelSimulation.h File Reference
6.96 MyApp.cpp File Reference
6.96.1 Function Documentation
6.96.1.1 _buildModel01_CreDelDis(Model *model)
6.96.1.2 _buildModel02_CreDelDis(Model *model)
6.96.1.3 _buildModel03_CreSeiDelResDis(Model *model)
6.96.1.4 _buildMostCompleteModel(Model *model)
6.96.1.5 onEntityRemoveHandler(SimulationEvent *re)
6.96.1.6 onProcessEventHandler(SimulationEvent *re)
6.96.1.7 onReplicationEndHandler(SimulationEvent *re)
6.96.1.8 onReplicationStartHandler(SimulationEvent *re)
6.96.1.9 onSimulationStartHandler(SimulationEvent *re)
6.96.1.10 traceHandler(TraceEvent e)
6.96.1.11 traceSimulationHandler(TraceSimulationEvent e)
6.97 MyApp.h File Reference
6.98 OnEventManager.cpp File Reference
6.99 OnEventManager.h File Reference
6.99.1 Typedef Documentation
6.99.1.1 simulationEventHandler
6.100 Parser_if.h File Reference
6.101 Parser Default Impl1.cpp File Reference
6.102ParserDefaultImpl1.h File Reference
6.103ParserDummyImpl.cpp File Reference
6.104ParserDummyImpl.h File Reference
6.105 Plugin.cpp File Reference
6.106Plugin.h File Reference
6.106.1 Typedef Documentation
6.106.1.1 StaticGetPluginInformation
6.106.1.2 StaticLoaderComponentInstance

6.106.1.3 StaticLoaderElementInstance	72
6.107 PluginManager.cpp File Reference	72
6.108 Plugin Manager.h File Reference	72
6.109ProbDistrib.cpp File Reference	73
6.110ProbDistrib.h File Reference	73
6.111 Process Analyser_if.h File Reference	74
6.112ProcessAnalyserDefaultImpl1.cpp File Reference	74
6.113ProcessAnalyserDefaultImpl1.h File Reference	75
6.114ProcessAnalyserDummyImpl.cpp File Reference	76
6.115ProcessAnalyserDummyImpl.h File Reference	76
6.116Queue.cpp File Reference	77
6.117Queue.h File Reference	78
6.118README.md File Reference	79
6.119Record.cpp File Reference	79
6.120 Record.h File Reference	80
6.121 Release.cpp File Reference	80
6.122Release.h File Reference	81
6.123Resource.cpp File Reference	82
6.124Resource.h File Reference	83
6.125Sampler_if.h File Reference	84
6.126SamplerDefaultImpl1.cpp File Reference	84
6.127SamplerDefaultImpl1.h File Reference	84
6.128SamplerDummyImpl.cpp File Reference	85
6.129SamplerDummyImpl.h File Reference	86
6.130 Scenario Experiment_if.h File Reference	86
6.131 Seize.cpp File Reference	87
6.132Seize.h File Reference	87
6.133 Simulation Control.cpp File Reference	88
6.134SimulationControl.h File Reference	89
6.135SimulationReporter_if.h File Reference	89

lxii CONTENTS

6.136SimulationReporterDefaultImpl1.cpp File Reference
6.137SimulationReporterDefaultImpl1.h File Reference
6.138 Simulation Response.cpp File Reference
6.139 Simulation Response. h File Reference
6.140 Simulation Scenario.cpp File Reference
6.141 Simulation Scenario.h File Reference
6.142Simulator.cpp File Reference
6.143 Simulator.h File Reference
6.144SinkModelComponent.cpp File Reference
6.145SinkModelComponent.h File Reference
6.146SourceModelComponent.cpp File Reference
6.147 Source Model Component. h File Reference
6.148 Statistics_if.h File Reference
6.149 Statistics Collector.cpp File Reference
6.150 Statistics Collector.h File Reference
6.151 Statistics Data File_if.h File Reference
6.152StatisticsDataFileDummyImpl.cpp File Reference
6.153 Statistics Data File Dummy Impl. h File Reference
6.154StatisticsDefaultImpl1.cpp File Reference
6.155 Statistics Default Impl1.h File Reference
6.156StatisticsDummyImpl.cpp File Reference
6.157StatisticsDummyImpl.h File Reference
6.158TestInputAnalyserTools.cpp File Reference
6.158.1 Function Documentation
6.158.1.1 testStudentSoftwareDevelopments()
6.159TestInputAnalyserTools.h File Reference
6.160 TestParser.cpp File Reference
6.161 TestParser.h File Reference
6.162TestStatistics.cpp File Reference
6.163TestStatistics.h File Reference

6.164ToolManager.cpp File Reference
6.165ToolManager.h File Reference
6.166TraceManager.cpp File Reference
6.167TraceManager.h File Reference
6.167.1 Typedef Documentation
6.167.1.1 traceErrorListener
6.167.1.2 traceListener
6.167.1.3 traceSimulationListener
6.167.1.4 traceSimulationProcessListener
6.168Traits.h File Reference
6.169Util.cpp File Reference
6.170Util.h File Reference
6.171 Variable.cpp File Reference
6.172 Variable.h File Reference
6.173Waiting.cpp File Reference
6.174Waiting.h File Reference
6.175WaitingResource.cpp File Reference
6.176WaitingResource.h File Reference

Chapter 1

ReGenESyS

Reborn Generic and Expansible System Simulator

Genesys is a result of teaching and research activities of Professor Dr. Ing Rafael Luiz Cancian. It began in early 2002 as a way to teach students the basics and simulation techniques of systems implemented by other comercial simulation tools, such as Arena. In Genesys development he replicated all the SIMAN language, used by Arena software, and Genesys has become a clone of that tool, including its graphical interface. Genesys allowed the inclusion of new simulation components through dynamic link libraries and also the parallel execution of simulation models in a distributed environment. The development of Genesys continued until 2007, when the professor stopped teaching systems simulation classes. Ten years later the professor starts again to teach systems simulation classes and to carry out scientific research in the area. So in 2018 Genesys is reborn as ReGenesys, with new language and programming techniques, and even more ambitious goals.

Developed by rlcancian

2 ReGenESyS

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

	23
Collector Datafile if	
CollectorDatafileDefaultImpl1	
'	10
	13
CollectorDummyImpl	15
	16
	33
	91
Event 10 ExperimentDesign_if 10	
ExperimentDesignDefaultImpl1	
ExperimentDesignDummyImpl	
FactorOrInteractionContribution	
Fitter if	
- FitterDefaultImpl1	
FitterDummyImpl	
GenesysApplication_if	8
BuildSimulationModel	29
GenesysConsole	9
GenesysGUI	
GenesysShell_if	
MyApp </td <td></td>	
TestParser	
TestStatistics	
HypothesisTester if	
HypothesisTesterDefaultImpl1	
HypothesisTesterDummyImpl	
Integrator if	
IntegratorDefaultImpl1	
IntegratorDummyImpl	

4 Hierarchical Index

LicenceManager	
LinkedBy	 149
List< T >	
List< Assign::Assignment *>	150
List< double >	150
List< Event * >	150
List < Model * >	150
List < ModelComponent * >	150
List < ModelElement * >	150
List< Plugin * >	150 150
List < ShellCommand * >	
List< SimulationControl * >	
List< SimulationResponse * >	
List< std::string >	
List< Util::identitifcation >	
List< Waiting * >	
Model	
ModelChecker if	
ModelCheckerDefaultImpl1	
ModelComponentManager_if	
ModelElement	
Attribute	
Counter	
Entity	
EntityType	
ModelComponent	
Assign	
Decide	
Delay	
Dummy	
Record	
Release	 . 265
Seize	 . 294
SinkModelComponent	 . 317
Dispose	 . 74
SourceModelComponent	 . 321
Create	
Queue	 . 249
Resource	 . 272
StatisticsCollector	 . 334
Variable	 . 382
ModelInfo	 197
ModelManager	 207
ModelPersistence_if	 210
ModelPersistenceDefaultImpl1	 . 212
ModelSimulation	 215
OnEventManager	
Parser_if	
ParserDefaultImpl1	
ParserDummyImpl	
Plugin	
PluginInformation	
PluginManager	
ProbDistrib	
ProcessAnalyser_if	
ProcessAnalyserDefaultImpl1	

2.1 Class Hierarchy 5

ProcessAnalyserDummyImpl
Sampler_if::RNG_Parameters
SamplerDefaultImpl1::DefaultImpl1RNG_Parameters
SamplerDummyImpl::MyRNG_Parameters
Sampler_if
SamplerDefaultImpl1
SamplerDummyImpl
ScenarioExperiment_if
SimulationEvent
SimulationReporter_if
SimulationReporterDefaultImpl1
SimulationResponse
SimulationControl
SimulationScenario
Simulator
Statistics_if
StatisticsDatafile_if
StatisticsDataFileDummyImpl
StatisticsDefaultImpl1
StatisticsDummyImpl
ToolManager
TraceEvent
TraceErrorEvent
TraceSimulationEvent
TraceSimulationProcess
TraceManager
Traits < T >
Traits Collector_if 370 Traits ExperimentDesign_if 370
Traits < Fitter if >
Traits < GenesysApplication_if >
Traits< HypothesisTester_if >
Traits < Integrator if >
Traits < Model >
Traits < Model Checker_if >
Traits < ModelComponent >
Traits < ModelPersistence_if >
Traits < Parser_if >
Traits< ProcessAnalyser_if >
Traits < Sampler_if >
Traits SimulationReporter_if 375 Traits Statistics if 376
Util
Waiting
WaitingResource

6 Hierarchical Index

Chapter 3

Class Index

3.1 Class List

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

9	15
Assign::Assignment	23
Attribute	25
BuildSimulationModel	29
Collector_if	32
CollectorDatafile_if	35
CollectorDatafileDefaultImpl1	37
CollectorDatafileDummyImpl	40
CollectorDefaultImpl1	43
CollectorDummyImpl	45
ComponentManager	46
Counter	50
Create	55
Decide	62
SamplerDefaultImpl1::DefaultImpl1RNG_Parameters	66
Delay	68
Dispose	74
Dummy	79
ElementManager	83
ElementManager_if	91
Entity	92
EntityType	97
Event	03
ExperimentDesign if	04
ExperimentDesignDefaultImpl1	06
ExperimentDesignDummyImpl	08
	09
	10
FitterDefaultImpl1	13
FitterDummyImpl	16
	18
	19
GenesysGUI	31
GenesysShell_if	
•	35

8 Class Index

HypothesisTesterDefaultImpl1	37
HypothesisTesterDummyImpl	40
Integrator_if	42
IntegratorDefaultImpI1	44
IntegratorDummyImpl	46
LicenceManager	48
LinkedBy	49
List< T >	50
	59
ModelChecker if	75
ModelCheckerDefaultImpl1	76
•	81
·	88
•	89
	97
	207
	210
	212
·	215
	220
	223
OnEventManager	
Parser if	
ParserDefaultImpl1	
·	
ParserDummyImpl	
Plugin	
PluginInformation	
PluginManager	
ProbDistrib	
ProcessAnalyser_if	
ProcessAnalyserDefaultImpl1	
ProcessAnalyserDummyImpl	
Queue	
Record	
Release	
Resource	
Sampler_if::RNG_Parameters	282
Sampler_if	283
SamplerDefaultImpl1	286
	291
ScenarioExperiment_if	294
Seize	294
SimulationControl	303
SimulationEvent	304
SimulationReporter_if	306
SimulationReporterDefaultImpl1	307
SimulationResponse	311
SimulationScenario	312
Simulator	313
SinkModelComponent	317
·	321
·	329
	334
	339
	342
	345
•	350
	353
100th pair that you 10010	,00

3.1 Class List

TestParser	56
TestStatistics	57
ToolManager	59
TraceErrorEvent	61
TraceEvent	62
TraceManager	63
TraceSimulationEvent	67
TraceSimulationProcess	69
Traits < T >	70
Traits < Collector_if >	70
Traits < ExperimentDesign_if >	70
Traits < Fitter_if >	71
Traits < GenesysApplication_if >	71
Traits< HypothesisTester_if >	71
Traits < Integrator_if >	72
Traits < Model >	72
Traits < ModelChecker_if >	73
Traits < ModelComponent >	73
Traits < ModelPersistence_if >	74
	74
Traits < Process Analyser_if >	75
• –	75
Traits < SimulationReporter_if >	75
	76
Util	76
	82
	86
Waiting Resource	88

10 Class Index

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

.dep.inc) 1
Assign.cpp	91
Assign.h	91
Attribute.cpp	92
Attribute.h	93
BuildSimulationModel.cpp	94
BuildSimulationModel.h	94
Collector_if.h) 5
CollectorDatafile_if.h	98
CollectorDatafileDefaultImpl1.cpp	98
CollectorDatafileDefaultImpl1.h	99
CollectorDatafileDummyImpl.cpp)0
CollectorDatafileDummyImpl.h)1
CollectorDefaultImpl1.cpp)2
CollectorDefaultImpl1.h)3
CollectorDummyImpl.cpp)4
CollectorDummyImpl.h)4
ComponentManager.cpp)5
ComponentManager.h)6
Counter.cpp)7
Counter.h)7
Create.cpp)8
Create.h)9
Decide.cpp	10
Decide.h	10
DefineGetterSetter.h	11
Delay.cpp	13
Delay.h	
Dispose.cpp	14
Dispose.h	
Dummy.cpp	16
Dummy.h	16
ElementManager.cpp	17
ElementManager.h	18

12 File Index

Entity.cpp	419
Entity.h	419
EntityType.cpp	420
EntityType.h	421
Event.cpp	421
Event.h	422
ExperimentDesign_if.h	423
ExperimentDesignDefaultImpl1.cpp	424
ExperimentDesignDefaultImpl1.h	424
ExperimentDesignDummyImpl.cpp	425
ExperimentDesignDummyImpl.h	425
FactorOrInteractionContribution.cpp	426
FactorOrInteractionContribution.h	427
Fitter_if.h	428
FitterDefaultImpl1.cpp	429
FitterDefaultImpl1.h	430
FitterDummyImpl.cpp	431
FitterDummyImpl.h	431
Functor.h	
GenesysApplication_if.h	
GenesysConsole.cpp	
GenesysConsole.h	
GenesysGUI.cpp	
GenesysGUI.h	
GenesysShell_if.h	
HypothesisTester_if.h	
HypothesisTesterDefaultImpl1.cpp	
HypothesisTesterDefaultImpl1.h	
HypothesisTesterDummyImpl.cpp	
HypothesisTesterDummyImpl.h	
Integrator_if.h	
IntegratorDefaultImpl1.cpp	
IntegratorDefaultImpl1.h	
IntegratorDummyImpl.cpp	
IntegratorDummyImpl.h	
LicenceManager.cpp	
LicenceManager.h	
LinkedBy.cpp	443
LinkedBy.h	443
List.h	443
main.cpp	444
Model.cpp	445
Model.h	446
ModelChecker_if.h	447 448
ModelCheckerDefaultImpl1.cpp	448
ModelComponent.cpp	449
ModelComponent.h	449
ModelComponentManager_if.h	450
ModelElement.cpp	450
ModelElement.h	450
ModelInfo.cpp	450
ModelInfo.h	451
ModelManager.cpp	452
ModelManager.h	453
ModelPersistence if.h	454
ModelPersistenceDefaultImpl1.cpp	454
ModelPersistenceDefaultImph1.cpp	455
model contribution and the contribution of the	,00

4.1 File List

ModelSimulation.cpp	456
ModelSimulation.h	456
MyApp.cpp	457
MyApp.h	464
OnEventManager.cpp	464
OnEventManager.h	465
Parser if.h	466
ParserDefaultImpl1.cpp	
ParserDefaultImpl1.h	
·	
ParserDummyImpl.cpp	
ParserDummyImpl.h	469
Plugin.cpp	470
Plugin.h	470
PluginManager.cpp	472
PluginManager.h	472
ProbDistrib.cpp	473
ProbDistrib.h	473
ProcessAnalyser_if.h	474
ProcessAnalyserDefaultImpl1.cpp	474
ProcessAnalyserDefaultImpl1.h	475
ProcessAnalyserDummyImpl.cpp	476
ProcessAnalyserDummyImpl.h	
Queue.cpp	
Queue.h	
Record.cpp	
Record.h	
Release.cpp	480
Release.h	
Resource.cpp	482
Resource.h	483
Sampler_if.h	484
SamplerDefaultImpl1.cpp	484
SamplerDefaultImpl1.h	484
SamplerDummyImpl.cpp	485
SamplerDummyImpl.h	486
ScenarioExperiment if.h	486
Seize.cpp	487
Seize.h	487
SimulationControl.cpp	488
SimulationControl.h	489
SimulationReporter if.h	489
SimulationReporterDefaultImpl1.cpp	490
	490
SimulationReporterDefaultImpl1.h	
SimulationResponse.cpp	492
SimulationResponse.h	492
SimulationScenario.cpp	493
SimulationScenario.h	494
Simulator.cpp	494
Simulator.h	495
SinkModelComponent.cpp	496
SinkModelComponent.h	496
SourceModelComponent.cpp	497
SourceModelComponent.h	497
Statistics if.h	498
StatisticsCollector.cpp	499
StatisticsCollector.h	500
StatisticsDataFile if.h	500
StatisticsDataFileDummyImpl.cpp	502
otationopatai nepaininyimpi.opp	502

14 File Index

StatisticsDataFileDummyImpl.h	502
StatisticsDefaultImpl1.cpp	503
StatisticsDefaultImpl1.h	503
StatisticsDummyImpl.cpp	504
StatisticsDummyImpl.h	505
TestInputAnalyserTools.cpp	506
TestInputAnalyserTools.h	508
TestParser.cpp	509
TestParser.h	509
TestStatistics.cpp	510
TestStatistics.h	510
ToolManager.cpp	511
ToolManager.h	512
TraceManager.cpp	512
TraceManager.h	513
Traits.h	514
Util.cpp	515
Util.h	516
Variable.cpp	516
Variable.h	517
Waiting.cpp	518
Waiting.h	518
WaitingResource.cpp	519
	520

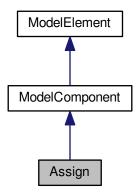
Chapter 5

Class Documentation

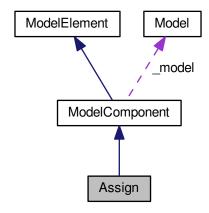
5.1 Assign Class Reference

#include <Assign.h>

Inheritance diagram for Assign:



Collaboration diagram for Assign:



Classes

· class Assignment

Public Types

• enum DestinationType : int { DestinationType::Attribute =0, DestinationType::Variable =1 }

Public Member Functions

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

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual void _execute (Entity *entity)
- virtual bool _loadInstance (std::map< std::string, std::string > *fields)
- virtual void _initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

5.1.1 Member Enumeration Documentation

5.1.1.1 enum Assign::DestinationType:int [strong]

Enumerator

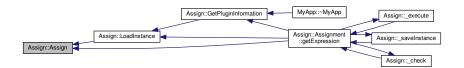
Attribute

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:

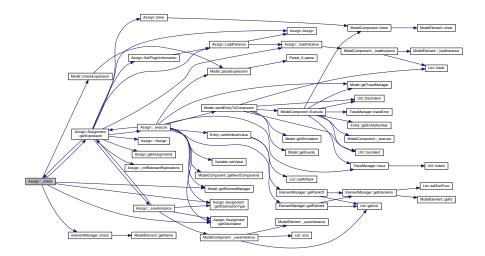


5.1.3 Member Function Documentation

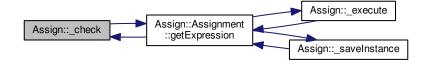
5.1.3.1 bool Assign::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



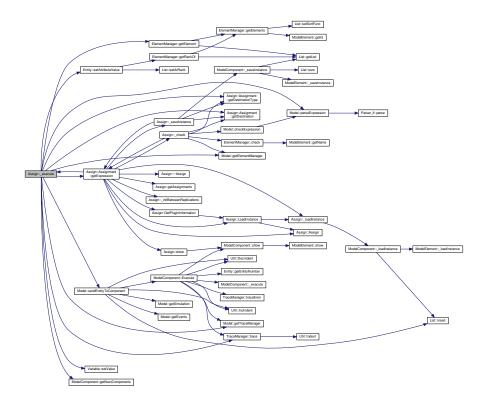
Here is the caller graph for this function:



5.1.3.2 void Assign::_execute(Entity * *entity* **)** [protected], [virtual]

Implements ModelComponent.

Here is the call graph for this function:



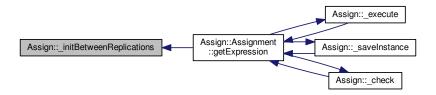
Here is the caller graph for this function:



5.1.3.3 void Assign::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

Here is the caller graph for this function:



5.1.3.4 bool Assign::_loadInstance(std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



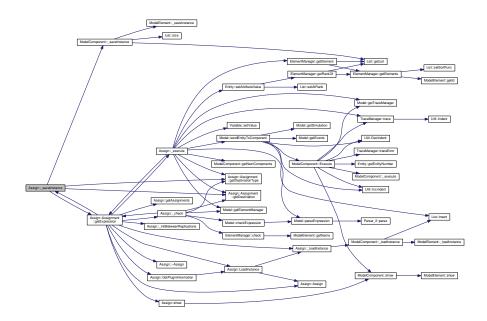
Here is the caller graph for this function:



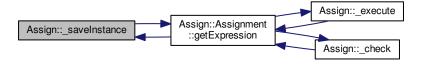
5.1.3.5 std::map< std::string, std::string > * Assign::_saveInstance() [protected], [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:

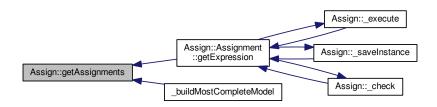


Here is the caller graph for this function:



5.1.3.6 List < Assign::Assignment * > * Assign::getAssignments () const

Here is the caller graph for this function:

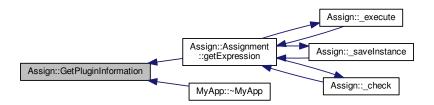


5.1.3.7 PluginInformation * Assign::GetPluginInformation () [static]

Here is the call graph for this function:

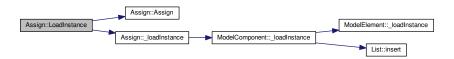


Here is the caller graph for this function:



5.1.3.8 ModelComponent * Assign::LoadInstance (Model * model, std::map < std::string, std::string > * fields) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



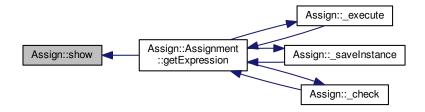
5.1.3.9 std::string Assign::show() [virtual]

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.

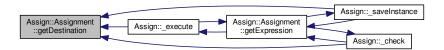
5.2.2 Constructor & Destructor Documentation

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

5.2.3 Member Function Documentation

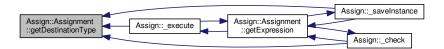
5.2.3.1 std::string Assign::Assignment::getDestination() const [inline]

Here is the caller graph for this function:



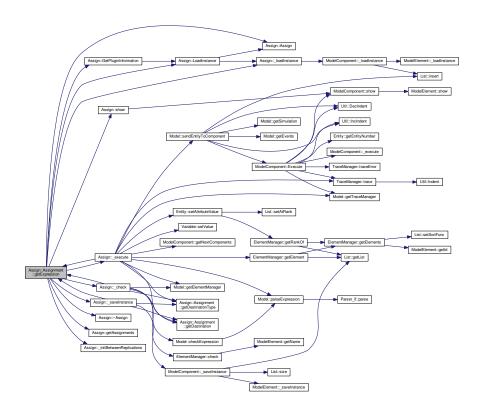
5.2.3.2 DestinationType Assign::Assignment::getDestinationType () const [inline]

Here is the caller graph for this function:

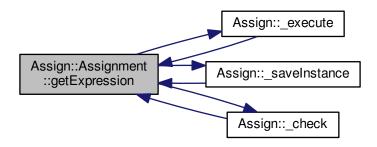


5.2.3.3 std::string Assign::Assignment::getExpression() const [inline]

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]

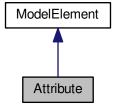
The documentation for this class was generated from the following file:

• Assign.h

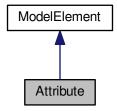
5.3 Attribute Class Reference

#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 ()

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelElement * LoadInstance (ElementManager *elems, std::map< std::string, std::string > *fields)

Protected Member Functions

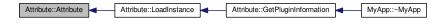
- virtual bool _loadInstance (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

5.3.1 Constructor & Destructor Documentation

5.3.1.1 Attribute::Attribute ()

Here is the caller graph for this function:



- 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** bool Attribute::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

5.3.2.2 bool Attribute::_loadInstance(std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



Here is the caller graph for this function:



 $\textbf{5.3.2.3} \quad \textbf{std::map} < \textbf{std::string}, \textbf{std::string} > * \textbf{Attribute::_saveInstance()}, \texttt{[protected]}, \texttt{[virtual]}$

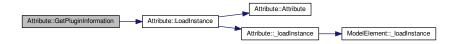
Reimplemented from ModelElement.

Here is the call graph for this function:



5.3.2.4 PluginInformation * Attribute::GetPluginInformation () [static]

Here is the call graph for this function:

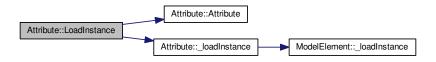


Here is the caller graph for this function:



5.3.2.5 ModelElement * Attribute::LoadInstance (ElementManager * elems, std::map < std::string, std::string > * fields) [static]

Here is the call graph for this function:



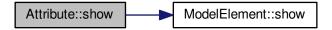
Here is the caller graph for this function:



5.3.2.6 std::string Attribute::show() [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



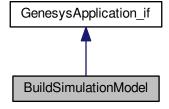
The documentation for this class was generated from the following files:

- · Attribute.h
- Attribute.cpp

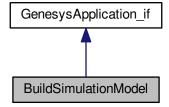
5.4 BuildSimulationModel Class Reference

#include <BuildSimulationModel.h>

Inheritance diagram for BuildSimulationModel:



 $Collaboration\ diagram\ for\ BuildSimulation Model:$

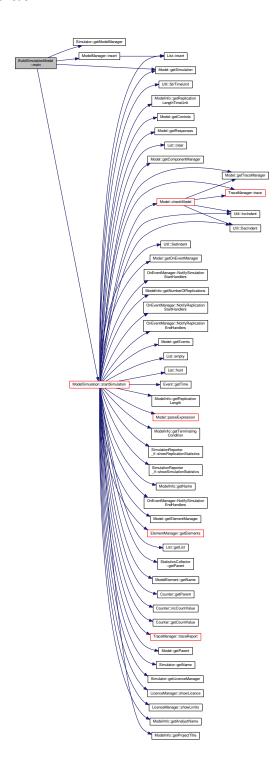


Public N	<i>l</i> lember	Func	tions
----------	-----------------	------	-------

Implements GenesysApplication_if.

	BuildSimulationModel () virtual 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	<pre>int BuildSimulationModel::main(int argc, char ** argv) [virtual]</pre>
	the main function of the BuildSimulationModel application. It instanciates the simulator, builds a simulation and then simulate that model.

Here is the call graph for this function:



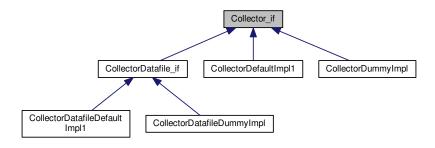
The documentation for this class was generated from the following files:

- 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

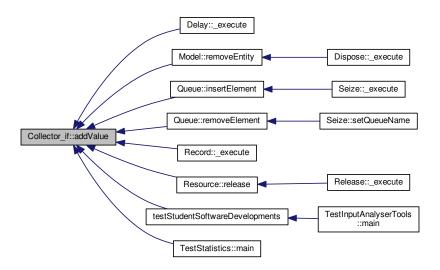
Interface for collecting values of a single stochastic variable. Values collected can be used as base for statistical analysis.

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 Collector → DefaultImpl1.

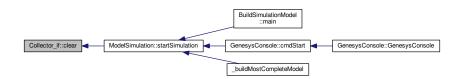
Here is the caller graph for this function:



5.5.2.2 virtual void Collector_if::clear() [pure virtual]

Implemented in CollectorDatafileDefaultImpl1, CollectorDummyImpl, CollectorDatafileDummyImpl, and Collector → DefaultImpl1.

Here is the caller graph for this function:



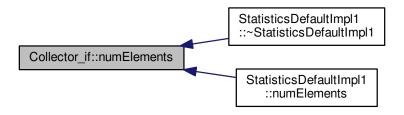
 $\textbf{5.5.2.3} \quad \textbf{virtual double Collector_if::getLastValue()} \quad [\texttt{pure virtual}]$

 $Implemented \ in \ Collector Data file Default Impl 1, \ Collector Dummy Impl, \ Collector Data file Dummy Impl, \ and \ Collector \\ Default Impl 1.$

5.5.2.4 virtual unsigned long Collector_if::numElements () [pure virtual]

Implemented in CollectorDatafileDefaultImpl1, CollectorDummyImpl, CollectorDatafileDummyImpl, and Collector → DefaultImpl1.

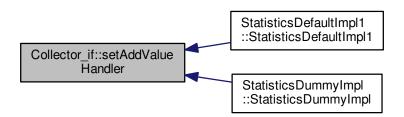
Here is the caller graph for this function:



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

Implemented in CollectorDatafileDefaultImpl1, CollectorDatafileDummyImpl, CollectorDummyImpl, and CollectorDefaultImpl1.

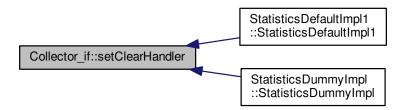
Here is the caller graph for this function:



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

 $Implemented \ in \ Collector Data file Default Impl 1, \ Collector Data file Dummy Impl, \ Collector Dummy Impl, \ and \ Collector Default Impl 1.$

Here is the caller graph for this function:



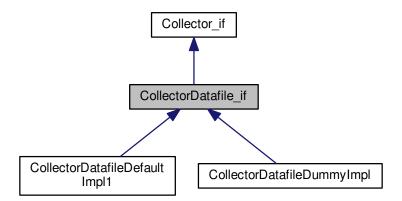
The documentation for this class was generated from the following file:

· Collector_if.h

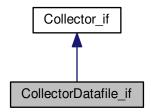
5.6 CollectorDatafile_if Class Reference

#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 Collector Datafile_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:



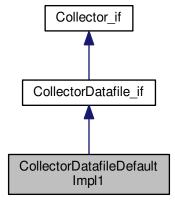
The documentation for this class was generated from the following file:

· CollectorDatafile_if.h

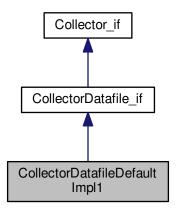
5.7 CollectorDatafileDefaultImpl1 Class Reference

#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 Collector Datafile Default Impl1::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.
\textbf{5.7.2.10} \quad \textbf{void CollectorDatafileDefaultImpl1::setClearHandler ( \ \textbf{CollectorClearHandler clearHandler }) \quad [\texttt{virtual}]
Implements Collector_if.
```

5.7.2.11 void Collector Datafile Default Impl1::set DataFilename (std::string filename) [virtual]

Implements CollectorDatafile_if.

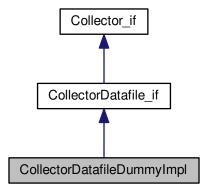
The documentation for this class was generated from the following files:

- CollectorDatafileDefaultImpl1.h
- CollectorDatafileDefaultImpl1.cpp

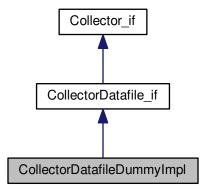
5.8 CollectorDatafileDummyImpl Class Reference

#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 Collector Datafile 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 Collector Datafile Dummylmpl::set Clear Handler ( Collector Clear Handler clear Handler ) [virtual]
Implements Collector_if.
5.8.2.11 void CollectorDatafileDummyImpl::setDataFilename ( std::string filename ) [virtual]
Implements CollectorDatafile_if.
```

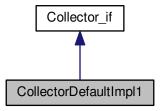
- · CollectorDatafileDummyImpl.h
- CollectorDatafileDummyImpl.cpp

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

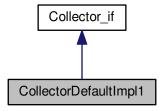
5.9 Collector Default Impl 1 Class Reference

#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
       CollectorDefaultImpl1::CollectorDefaultImpl1 ( )
5.9.1.1
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.
```

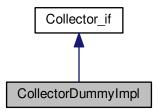
The documentation for this class was generated from the following files:

- CollectorDefaultImpl1.h
- CollectorDefaultImpl1.cpp

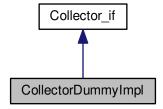
5.10 CollectorDummyImpl Class Reference

#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 CollectorDummylmpl::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 ComponentManager Class Reference

#include <ComponentManager.h>

Public Member Functions

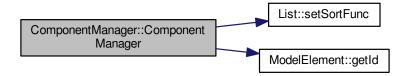
- ComponentManager (Model *model)
- ComponentManager (const ComponentManager & orig)
- virtual ∼ComponentManager ()
- bool insert (ModelComponent *comp)
- void remove (ModelComponent *comp)
- void clear ()
- ModelComponent * getComponent (Util::identitifcation id)
- ModelComponent * getComponent (std::string name)
- unsigned int getNumberOfComponents ()
- std::list< ModelComponent * >::iterator begin ()
- std::list< ModelComponent * >::iterator end ()

5.11.1 Constructor & Destructor Documentation

5.11.1.1 ComponentManager::ComponentManager (Model * model)

Components are sorted by ID

Here is the call graph for this function:



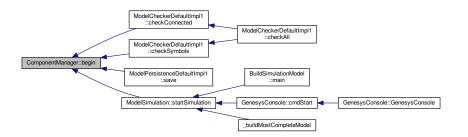
- 5.11.1.2 ComponentManager::ComponentManager (const ComponentManager & orig)
- **5.11.1.3 ComponentManager::**~ComponentManager() [virtual]

5.11.2 Member Function Documentation

 $\textbf{5.11.2.1} \quad \textbf{std::list} < \textbf{ModelComponent} * > :: \textbf{iterator ComponentManager::begin (} \quad \textbf{)}$



Here is the caller graph for this function:

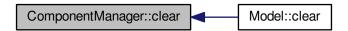


5.11.2.2 void ComponentManager::clear ()

Here is the call graph for this function:



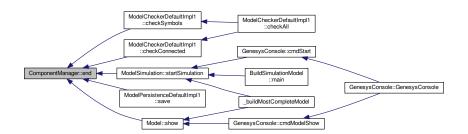
Here is the caller graph for this function:



5.11.2.3 std::list < ModelComponent * >::iterator ComponentManager::end ()

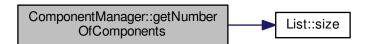


Here is the caller graph for this function:



- 5.11.2.4 ModelComponent * ComponentManager::getComponent (Util::identitifcation id)
- 5.11.2.5 ModelComponent * ComponentManager::getComponent (std::string name)
- 5.11.2.6 unsigned int ComponentManager::getNumberOfComponents ()

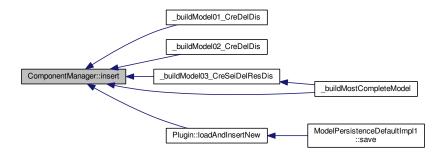
Here is the call graph for this function:



5.11.2.7 bool ComponentManager::insert (ModelComponent * comp)



Here is the caller graph for this function:



5.11.2.8 void ComponentManager::remove (ModelComponent * comp)

Here is the call graph for this function:



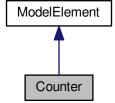
The documentation for this class was generated from the following files:

- ComponentManager.h
- ComponentManager.cpp

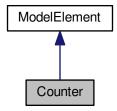
5.12 Counter Class Reference

#include <Counter.h>

Inheritance diagram for Counter:



Collaboration diagram for Counter:



Public Member Functions

- Counter ()
- Counter (std::string name)
- Counter (std::string name, ModelElement *parent)
- Counter (const Counter &orig)
- virtual ∼Counter ()
- virtual std::string show ()
- void clear ()
- void incCountValue (int value=1)
- unsigned long getCountValue ()
- ModelElement * getParent () const

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelElement * LoadInstance (ElementManager *elems, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

5.12.1 Constructor & Destructor Documentation

5.12.1.1 Counter::Counter()



```
5.12.1.2 Counter::Counter ( std::string name )
```

5.12.1.3 Counter::Counter (std::string name, ModelElement * parent)

```
5.12.1.4 Counter::Counter ( const Counter & orig )
```

```
5.12.1.5 Counter::~Counter() [virtual]
```

5.12.2 Member Function Documentation

```
5.12.2.1 bool Counter::_check( std::string * errorMessage ) [protected], [virtual]
```

Reimplemented from ModelElement.

```
5.12.2.2 bool Counter::_loadInstance ( std::map < std::string, std::string > * fields ) [protected], [virtual]
```

Reimplemented from ModelElement.

Here is the call graph for this function:



Here is the caller graph for this function:



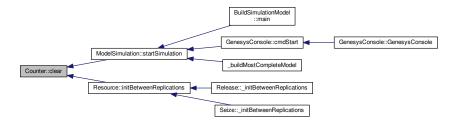
5.12.2.3 std::map< std::string, std::string > * Counter::_saveInstance() [protected], [virtual]

Reimplemented from ModelElement.



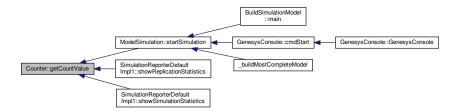
5.12.2.4 void Counter::clear ()

Here is the caller graph for this function:

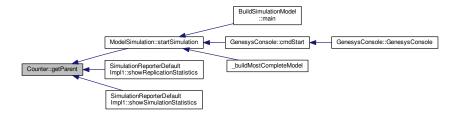


5.12.2.5 unsigned long Counter::getCountValue ()

Here is the caller graph for this function:

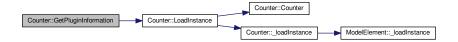


5.12.2.6 ModelElement * Counter::getParent () const



5.12.2.7 PluginInformation * Counter::GetPluginInformation () [static]

Here is the call graph for this function:

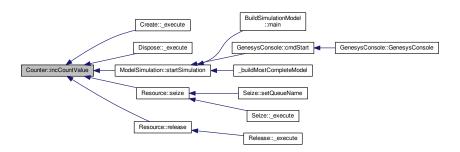


Here is the caller graph for this function:

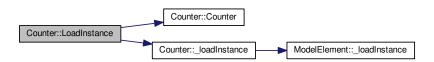


5.12.2.8 void Counter::incCountValue (int value = 1)

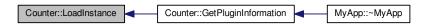
Here is the caller graph for this function:



5.12.2.9 ModelElement * Counter::LoadInstance (ElementManager * elems, std::map< std::string, std::string > * fields) [static]



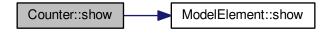
Here is the caller graph for this function:



5.12.2.10 std::string Counter::show() [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



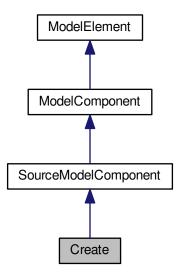
The documentation for this class was generated from the following files:

- · Counter.h
- · Counter.cpp

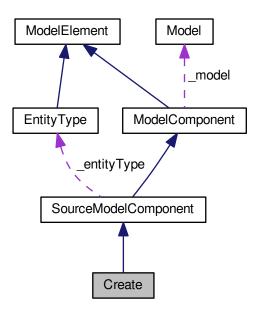
5.13 Create Class Reference

#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 ()

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string) > *fields)

Protected Member Functions

- virtual void _execute (Entity *entity)
- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual void _initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

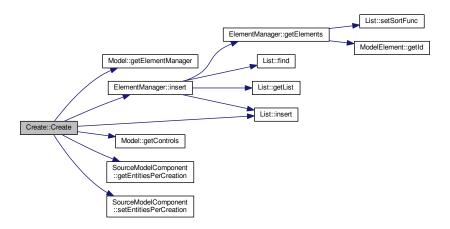
5.13.1 Detailed Description

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

5.13.2 Constructor & Destructor Documentation

5.13.2.1 Create::Create (Model * model)

Here is the call graph for this function:





5.13.2.2 Create::Create (const Create & orig)

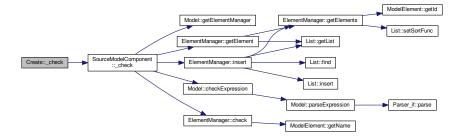
5.13.2.3 Create::~Create() [virtual]

5.13.3 Member Function Documentation

5.13.3.1 bool Create::_check(std::string * errorMessage) [protected], [virtual]

Reimplemented from SourceModelComponent.

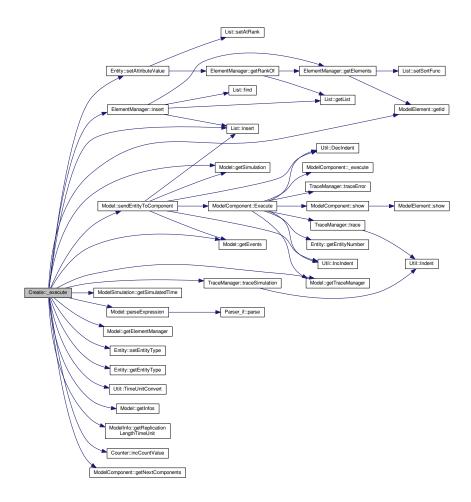
Here is the call graph for this function:



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

Implements ModelComponent.

Here is the call graph for this function:



5.13.3.3 void Create::_initBetweenReplications() [protected], [virtual]

Reimplemented from SourceModelComponent.

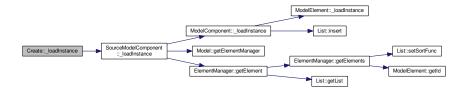
Here is the call graph for this function:



 $\textbf{5.13.3.4} \quad \textbf{bool Create::_loadInstance (std::map} < \textbf{std::string}, \textbf{std::string} > * \textit{fields} \textbf{)} \quad \texttt{[protected], [virtual]}$

Reimplemented from SourceModelComponent.

Here is the call graph for this function:



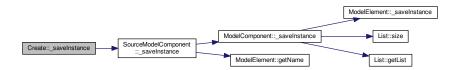
Here is the caller graph for this function:



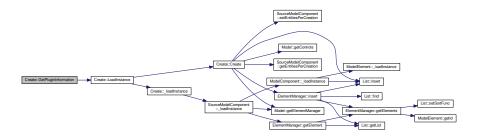
5.13.3.5 std::map < std::string, std::string > * Create::_saveInstance() [protected], [virtual]

Reimplemented from SourceModelComponent.

Here is the call graph for this function:



5.13.3.6 PluginInformation * Create::GetPluginInformation () [static]

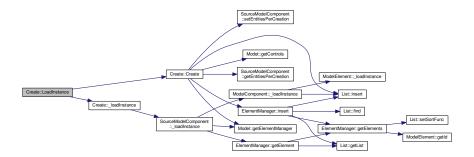


Here is the caller graph for this function:

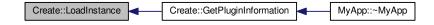


5.13.3.7 ModelComponent * Create::LoadInstance (Model * model, std::map< std::string, std::string > * fields) [static]

Here is the call graph for this function:

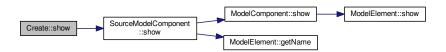


Here is the caller graph for this function:



5.13.3.8 std::string Create::show() [virtual]

 $\label{lem:control_control} \textbf{Reimplemented from } \textbf{SourceModelComponent}.$



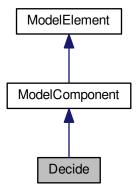
The documentation for this class was generated from the following files:

- Create.h
- Create.cpp

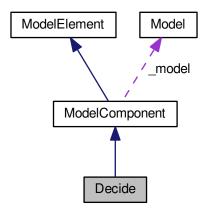
5.14 Decide Class Reference

#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 ()

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual void execute (Entity *entity)
- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual void _initBetweenReplications ()
- virtual std::map< std::string, std::string > * saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

5.14.1 Constructor & Destructor Documentation

5.14.1.1 Decide::Decide (Model * model)

Here is the caller graph for this function:



- 5.14.1.2 Decide::Decide (const Decide & orig)
- **5.14.1.3 Decide::**~Decide() [virtual]

5.14.2 Member Function Documentation

5.14.2.1 bool Decide::_check(std::string * errorMessage) [protected], [virtual]

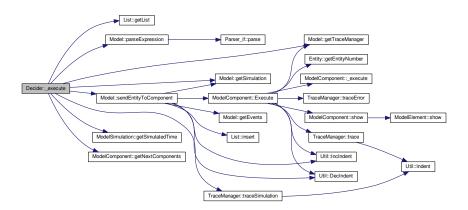
Reimplemented from ModelElement.



5.14.2.2 void Decide::_execute(Entity * *entity*) [protected], [virtual]

Implements ModelComponent.

Here is the call graph for this function:



5.14.2.3 void Decide::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

5.14.2.4 bool Decide::_loadInstance (std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:

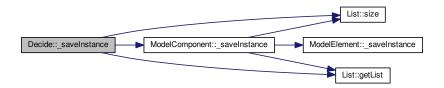




5.14.2.5 std::map< std::string, std::string > * Decide::_saveInstance() [protected], [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



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

Here is the caller graph for this function:



 $\textbf{5.14.2.7} \quad \textbf{PluginInformation} * \textbf{Decide::GetPluginInformation()} \quad [\, \texttt{static} \,]$

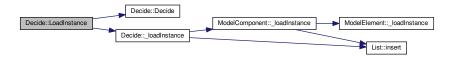
Here is the call graph for this function:





5.14.2.8 ModelComponent * Decide::LoadInstance (Model * model, std::map< std::string, std::string > * fields) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.2.9 std::string Decide::show() [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



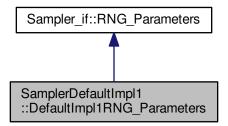
The documentation for this class was generated from the following files:

- Decide.h
- · Decide.cpp

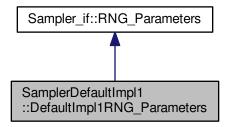
5.15 SamplerDefaultImpl1::DefaultImpl1RNG_Parameters Struct Reference

#include <SamplerDefaultImpl1.h>

Inheritance diagram for SamplerDefaultImpl1::DefaultImpl1RNG_Parameters:



Collaboration diagram for SamplerDefaultImpl1::DefaultImpl1RNG_Parameters:



Public Member Functions

 $\bullet \ {\sim} \mathsf{DefaultImpl1RNG_Parameters} \ () \texttt{=} \mathsf{default}$

Public Attributes

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

5.15.1 Constructor & Destructor Documentation

5.15.1.1 SamplerDefaultImpl1::DefaultImpl1RNG_Parameters:: DefaultImpl1RNG_Parameters() [default]

5.15.2 Member Data Documentation

5.15.2.1 unsigned int SamplerDefaultImpl1::DefaultImpl1RNG_Parameters::module = 2147483647

 $5.15.2.2 \quad unsigned \ int \ Sampler Default Impl 1:: Default Impl 1RNG_Parameters:: multiplier = 950706376$

5.15.2.3 unsigned int SamplerDefaultImpl1::DefaultImpl1RNG_Parameters::seed = 666

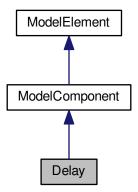
The documentation for this struct was generated from the following file:

· SamplerDefaultImpl1.h

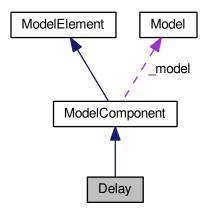
5.16 Delay Class Reference

#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 ()

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual void _execute (Entity *entity)
- virtual void initBetweenReplications ()
- virtual bool _loadInstance (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

5.16.1 Constructor & Destructor Documentation

5.16.1.1 Delay::Delay (Model * model)



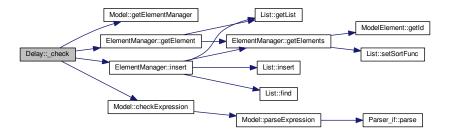
- 5.16.1.2 Delay::Delay (const Delay & orig)
- 5.16.1.3 Delay:: \sim Delay() [virtual]

5.16.2 Member Function Documentation

5.16.2.1 bool Delay::_check (std::string * *errorMessage* **)** [protected], [virtual]

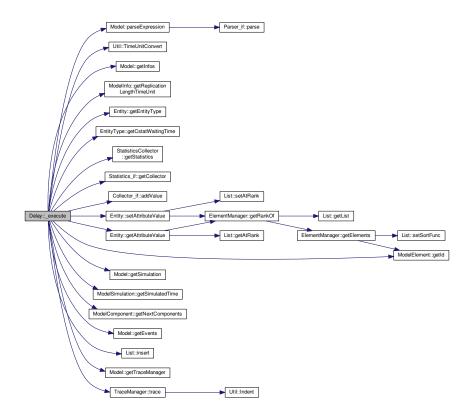
Reimplemented from ModelElement.

Here is the call graph for this function:



5.16.2.2 void Delay::_execute (Entity * *entity* **)** [protected], [virtual]

Implements ModelComponent.



5.16.2.3 void Delay::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

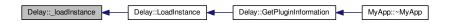
5.16.2.4 bool Delay::_loadInstance (std::map < std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:

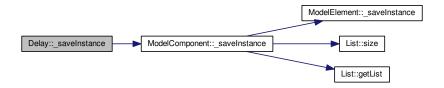


Here is the caller graph for this function:



 $\textbf{5.16.2.5} \quad \textbf{std::map} < \textbf{std::string}, \textbf{std::string} > * \textbf{Delay::_saveInstance()}, \texttt{[protected], [virtual]}$

Reimplemented from ModelComponent.



- 5.16.2.6 std::string Delay::getDelayExpression () const
- 5.16.2.7 Util::TimeUnit Delay::getDelayTimeUnit () const
- **5.16.2.8 PluginInformation * Delay::GetPluginInformation()** [static]

Here is the call graph for this function:

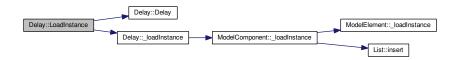


Here is the caller graph for this function:



5.16.2.9 ModelComponent * Delay::LoadInstance (Model * model, std::map< std::string, std::string > * fields) [static]

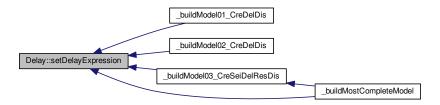
Here is the call graph for this function:





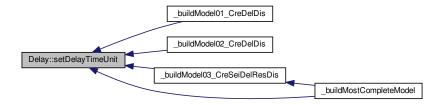
$5.16.2.10 \quad \text{void Delay::setDelayExpression (std::string $_$ delayExpression)}$

Here is the caller graph for this function:



5.16.2.11 void Delay::setDelayTimeUnit (Util::TimeUnit _delayTimeUnit)

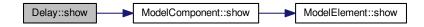
Here is the caller graph for this function:



5.16.2.12 std::string Delay::show() [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



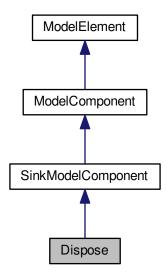
The documentation for this class was generated from the following files:

- · Delay.h
- Delay.cpp

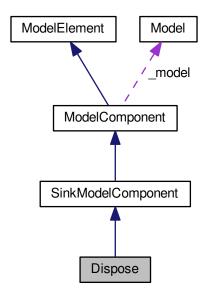
5.17 Dispose Class Reference

#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 ()

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

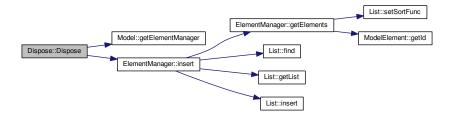
- virtual void _execute (Entity *entity)
- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual void _initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

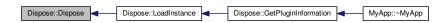
Additional Inherited Members

5.17.1 Constructor & Destructor Documentation

5.17.1.1 Dispose::Dispose (Model * model)

Here is the call graph for this function:





5.17.1.2 Dispose::Dispose (const Dispose & orig)

5.17.1.3 Dispose::∼**Dispose()** [virtual]

5.17.2 Member Function Documentation

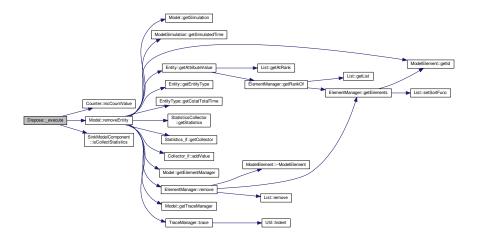
5.17.2.1 bool Dispose::_check (std::string * *errorMessage* **)** [protected], [virtual]

Reimplemented from SinkModelComponent.

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

Implements ModelComponent.

Here is the call graph for this function:



5.17.2.3 void Dispose::_initBetweenReplications() [protected], [virtual]

Reimplemented from SinkModelComponent.



5.17.2.4 bool Dispose::_loadInstance (std::map < std::string, std::string > * fields) [protected], [virtual]

Reimplemented from SinkModelComponent.

Here is the call graph for this function:



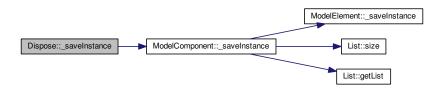
Here is the caller graph for this function:



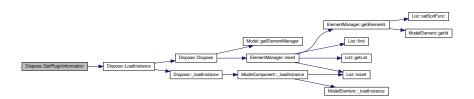
 $\textbf{5.17.2.5} \quad \textbf{std::map} < \textbf{std::string}, \textbf{std::string} > * \textbf{Dispose::_saveInstance()}, \texttt{[protected], [virtual]}$

Reimplemented from SinkModelComponent.

Here is the call graph for this function:



5.17.2.6 PluginInformation * Dispose::GetPluginInformation () [static]

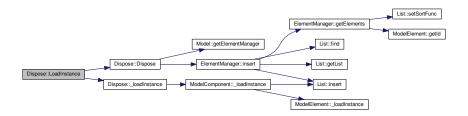


Here is the caller graph for this function:



5.17.2.7 ModelComponent * Dispose::LoadInstance (Model * model, std::map < std::string, std::string > * fields) [static]

Here is the call graph for this function:



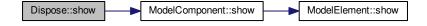
Here is the caller graph for this function:



5.17.2.8 std::string Dispose::show() [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



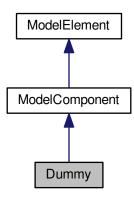
The documentation for this class was generated from the following files:

- · Dispose.h
- Dispose.cpp

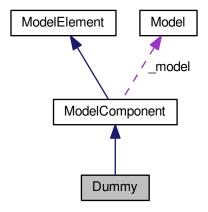
5.18 Dummy Class Reference

#include <Dummy.h>

Inheritance diagram for Dummy:



Collaboration diagram for Dummy:



Public Member Functions

- Dummy (Model *model)
- Dummy (const Dummy &orig)
- virtual ~Dummy ()
- virtual std::string show ()

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual void _execute (Entity *entity)
- virtual void _initBetweenReplications ()
- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

5.18.1 Constructor & Destructor Documentation

```
5.18.1.1 Dummy::Dummy ( Model * model )
```

Here is the caller graph for this function:



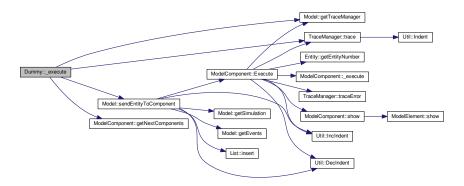
- 5.18.1.2 Dummy::Dummy (const Dummy & orig)
- **5.18.1.3 Dummy::**∼**Dummy()** [virtual]
- 5.18.2 Member Function Documentation
- **5.18.2.1** bool Dummy::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

5.18.2.2 void Dummy::_execute(Entity * *entity*) [protected], [virtual]

Implements ModelComponent.

Here is the call graph for this function:



5.18.2.3 void Dummy::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

 $\textbf{5.18.2.4} \quad \textbf{bool Dummy::_loadInstance (std::map} < \textbf{std::string}, \textbf{std::string} > * \textit{fields} \textbf{)} \quad \texttt{[protected], [virtual]}$

Reimplemented from ModelComponent.

Here is the call graph for this function:

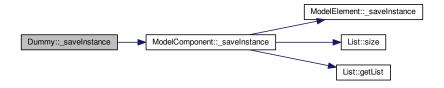




5.18.2.5 std::map < std::string, std::string > * Dummy::_saveInstance() [protected], [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



5.18.2.6 PluginInformation * Dummy::GetPluginInformation () [static]

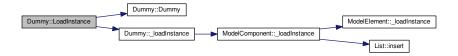
Here is the call graph for this function:



Here is the caller graph for this function:



5.18.2.7 ModelComponent * Dummy::LoadInstance (Model * model, std::map< std::string, std::string > * fields) [static]



Here is the caller graph for this function:



5.18.2.8 std::string Dummy::show() [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



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

- Dummy.h
- Dummy.cpp

5.19 ElementManager Class Reference

#include <ElementManager.h>

Public Member Functions

- ElementManager (Model *model)
- ElementManager (const ElementManager &orig)
- virtual ∼ElementManager ()
- bool insert (std::string infraTypename, ModelElement *infra)
- void remove (std::string infraTypename, ModelElement *infra)
- bool check (std::string infraTypename, ModelElement *infra, std::string expressionName, std::string *error←
 Message)
- bool check (std::string infraTypename, std::string infraName, std::string expressionName, bool mandatory, std::string *errorMessage)
- void clear ()
- ModelElement * getElement (std::string infraTypename, Util::identitifcation 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 ()
- Model * getModel () const

5.19.1 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

5.19.2 Constructor & Destructor Documentation

5.19.2.1 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

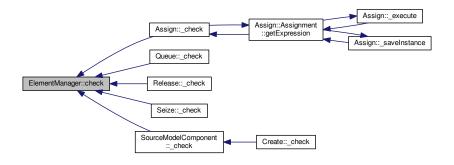
- 5.19.2.2 ElementManager::ElementManager (const ElementManager & orig)
- **5.19.2.3 ElementManager::**~ElementManager() [virtual]

5.19.3 Member Function Documentation

5.19.3.1 bool ElementManager::check (std::string infraTypename, ModelElement * infra, std::string expressionName, std::string * errorMessage)

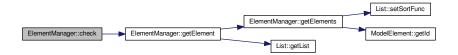
Here is the call graph for this function:





5.19.3.2 bool ElementManager::check (std::string *infraTypename*, std::string *infraName*, std::string *expressionName*, bool *mandatory*, std::string * *errorMessage*)

Here is the call graph for this function:



5.19.3.3 void ElementManager::clear ()

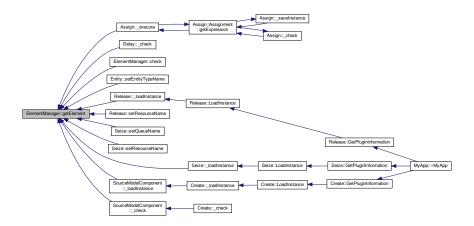
Here is the caller graph for this function:



5.19.3.4 ModelElement * ElementManager::getElement (std::string infraTypename, Util::identitifcation id)



Here is the caller graph for this function:

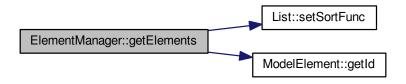


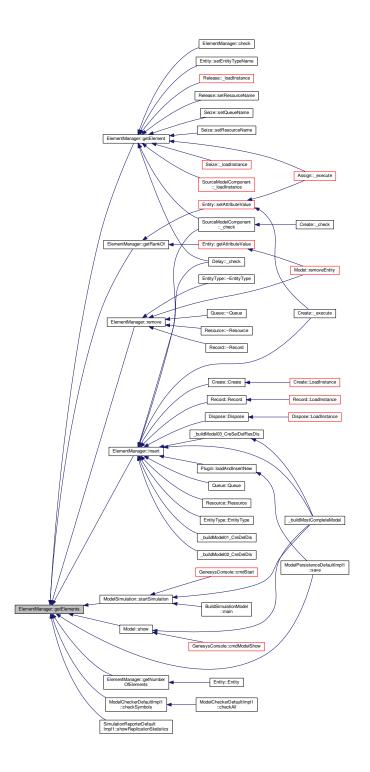
5.19.3.5 ModelElement * ElementManager::getElement (std::string infraTypename, std::string name)

Here is the call graph for this function:



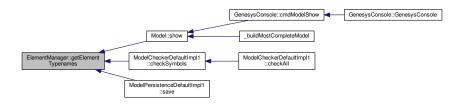
 $\textbf{5.19.3.6} \quad \textbf{List} < \textbf{ModelElement} * > * \textbf{ElementManager::getElements} \text{ (std::string } \textit{infraTypename } \text{) const}$





5.19.3.7 std::list < std::string > * ElementManager::getElementTypenames () const

Here is the caller graph for this function:



5.19.3.8 **Model** * ElementManager::getModel () const

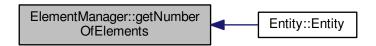
Here is the caller graph for this function:



5.19.3.9 unsigned int ElementManager::getNumberOfElements (std::string infraTypename)

Here is the call graph for this function:





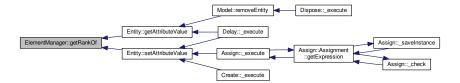
5.19.3.10 int ElementManager::getRankOf (std::string infraTypename, std::string name)

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

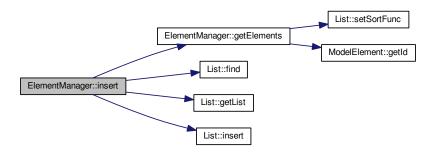
Here is the call graph for this function:



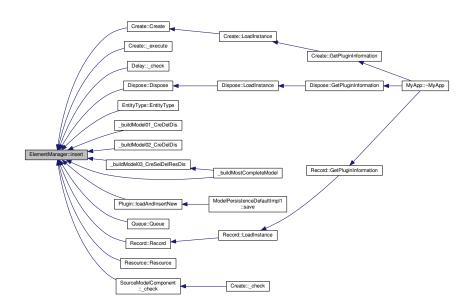
Here is the caller graph for this function:



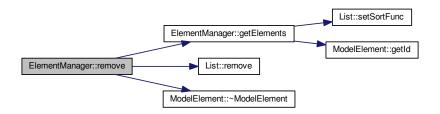
5.19.3.11 bool ElementManager::insert (std::string infraTypename, ModelElement *infra)



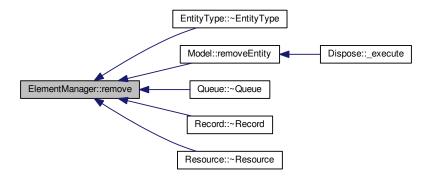
Here is the caller graph for this function:



5.19.3.12 void ElementManager::remove (std::string infraTypename, ModelElement * infra)

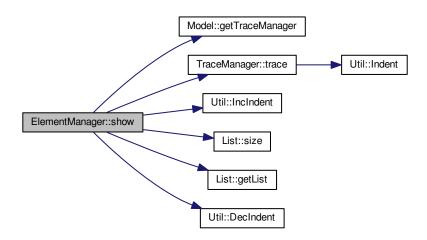


Here is the caller graph for this function:



5.19.3.13 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

5.20 ElementManager_if Class Reference

#include <ElementManager_if.h>

Public Member Functions

- ElementManager_if ()
- ElementManager_if (const ElementManager_if &orig)
- virtual ~ElementManager_if ()

5.20.1 Constructor & Destructor Documentation

```
5.20.1.1 ElementManager_if::ElementManager_if ( )
```

5.20.1.2 ElementManager_if::ElementManager_if (const ElementManager_if & orig)

```
5.20.1.3 virtual ElementManager_if::~ElementManager_if( ) [virtual]
```

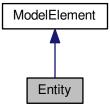
The documentation for this class was generated from the following file:

• ElementManager_if.h

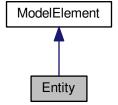
5.21 Entity Class Reference

```
#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 setEntityTypeName (std::string entityTypeName) throw ()
- std::string getEntityTypeName () const
- void setEntityType (EntityType *entityType)
- EntityType * getEntityType () const
- double getAttributeValue (std::string attributeName)
- void setAttributeValue (std::string attributeName, double value)
- Util::identitifcation getEntityNumber () const

Protected Member Functions

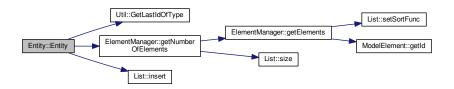
- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

5.21.1 Constructor & Destructor Documentation

5.21.1.1 Entity::Entity (ElementManager * elements)

Here is the call graph for this function:



5.21.1.2 Entity::Entity (const Entity & orig)

5.21.1.3 Entity::~Entity() [virtual]

5.21.2 Member Function Documentation

5.21.2.1 bool Entity::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

5.21.2.2 bool Entity::_loadInstance(std::map<std::string, std::string>* fields) [protected], [virtual]

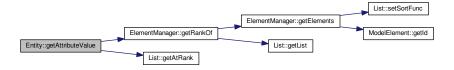
Reimplemented from ModelElement.

5.21.2.3 std::map< std::string, std::string > * Entity::_saveInstance() [protected], [virtual]

Reimplemented from ModelElement.

5.21.2.4 double Entity::getAttributeValue (std::string attributeName)

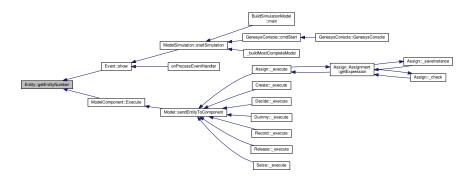
Here is the call graph for this function:



Here is the caller graph for this function:

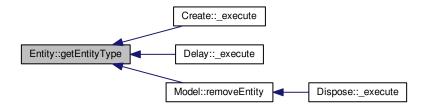


5.21.2.5 Util::identitifcation Entity::getEntityNumber () const



5.21.2.6 EntityType * Entity::getEntityType () const

Here is the caller graph for this function:



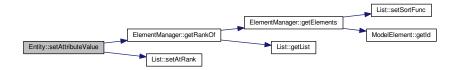
5.21.2.7 std::string Entity::getEntityTypeName () const

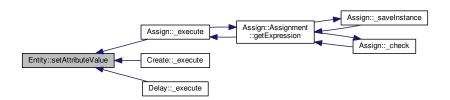
Here is the call graph for this function:



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

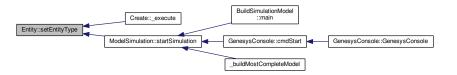
Here is the call graph for this function:





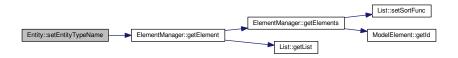
5.21.2.9 void Entity::setEntityType (EntityType * entityType)

Here is the caller graph for this function:



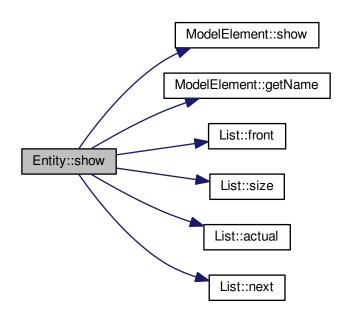
5.21.2.10 void Entity::setEntityTypeName (std::string entityTypeName) throw)

Here is the call graph for this function:

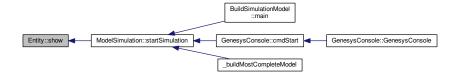


5.21.2.11 std::string Entity::show() [virtual]

Reimplemented from ModelElement.



Here is the caller graph for this function:



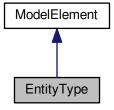
The documentation for this class was generated from the following files:

- Entity.h
- Entity.cpp

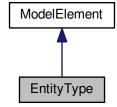
5.22 EntityType Class Reference

#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

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelElement * LoadInstance (ElementManager *elems, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

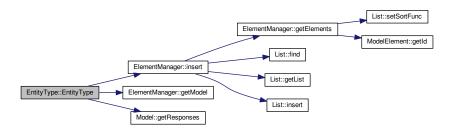
5.22.1 Constructor & Destructor Documentation

5.22.1.1 EntityType::EntityType (ElementManager * elemManager)



5.22.1.2 EntityType::EntityType (ElementManager * elemManager, std::string name)

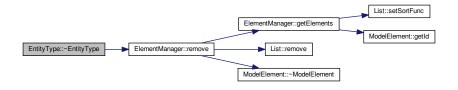
Here is the call graph for this function:



5.22.1.3 EntityType::EntityType (const EntityType & orig)

5.22.1.4 EntityType::~EntityType() [virtual]

Here is the call graph for this function:



5.22.2 Member Function Documentation

5.22.2.1 bool EntityType::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

 $\textbf{5.22.2.2} \quad \textbf{bool EntityType::_loadInstance (std::map} < \textbf{std::string}, \textbf{std::string} > * \textit{fields} \textbf{)} \quad \texttt{[protected], [virtual]}$

Reimplemented from ModelElement.



Here is the caller graph for this function:



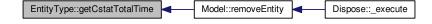
5.22.2.3 std::map< std::string, std::string > * EntityType::_saveInstance() [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



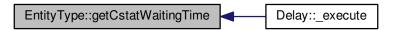
- 5.22.2.4 StatisticsCollector * EntityType::getCstatNVATime () const
- $5.22.2.5 \quad \textbf{StatisticsCollector} * \textbf{EntityType::getCstatOtherTime} \ (\ \) \ \textbf{const}$
- $5.22.2.6 \quad \textbf{StatisticsCollector} * \textbf{EntityType::getCstatTotalTime (} \quad \textbf{) const}$



- 5.22.2.7 StatisticsCollector * EntityType::getCstatTransferTime () const
- 5.22.2.8 StatisticsCollector * EntityType::getCstatVATime () const

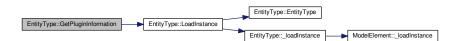
5.22.2.9 StatisticsCollector * EntityType::getCstatWaitingTime () const

Here is the caller graph for this function:



- 5.22.2.10 double EntityType::getInitialNVACost () const
- 5.22.2.11 double EntityType::getInitialOtherCost () const
- 5.22.2.12 std::string EntityType::getInitialPicture () const
- 5.22.2.13 double EntityType::getInitialVACost () const
- 5.22.2.14 double EntityType::getInitialWaitingCost () const
- **5.22.2.15** PluginInformation * EntityType::GetPluginInformation() [static]

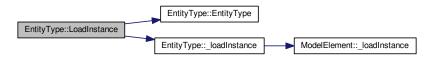
Here is the call graph for this function:





5.22.2.16 ModelElement * EntityType::LoadInstance (ElementManager * elems, std::map < std::string, std::string > * fields) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



- 5.22.2.17 void EntityType::setInitialNVACost (double _initialNVACost)
- 5.22.2.18 void EntityType::setInitialOtherCost (double _initialOtherCost)
- 5.22.2.19 void EntityType::setInitialPicture (std::string _initialPicture)
- 5.22.2.20 void EntityType::setInitialVACost (double _initialVACost)
- $5.22.2.21 \quad \text{void EntityType::setInitialWaitingCost (} \ \ \text{double} \ _\textit{initialWaitingCost} \ \ \text{)}$
- 5.22.2.22 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

5.23 Event Class Reference 103

5.23 Event Class Reference

#include <Event.h>

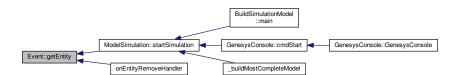
Public Member Functions

- 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 ()

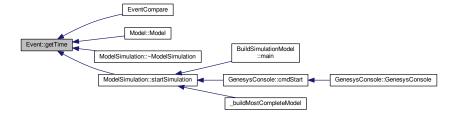
5.23.1 Constructor & Destructor Documentation

- 5.23.1.1 Event::Event (double time, Entity * entity, ModelComponent * component)
- 5.23.1.2 Event::Event (const Event & orig)
- 5.23.1.3 Event::~Event() [virtual]
- 5.23.2 Member Function Documentation
- 5.23.2.1 ModelComponent * Event::getComponent () const
- 5.23.2.2 Entity * Event::getEntity () const

Here is the caller graph for this function:

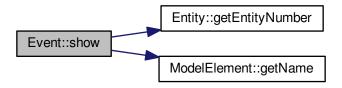


5.23.2.3 double Event::getTime () const

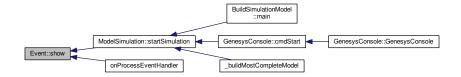


5.23.2.4 std::string Event::show()

Here is the call graph for this function:



Here is the caller graph for this function:



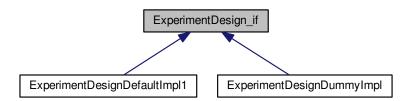
The documentation for this class was generated from the following files:

- Event.h
- Event.cpp

5.24 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

5.24.1 Detailed Description

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

5.24.2 Member Function Documentation

```
5.24.2.1 virtual bool ExperimentDesign_if::calculateContributionAndCoefficients() [pure virtual]
```

Implemented in ExperimentDesignDummyImpl, and ExperimentDesignDefaultImpl1.

5.24.2.2 virtual bool ExperimentDesign_if::generate2krScenarioExperiments() [pure virtual]

Implemented in ExperimentDesignDummyImpl, and ExperimentDesignDefaultImpl1.

5.24.2.3 virtual std::list<FactorOrInteractionContribution*>* ExperimentDesign_if::getContributions () const [pure virtual]

Implemented in ExperimentDesignDummyImpl, and ExperimentDesignDefaultImpl1.

5.24.2.4 virtual ProcessAnalyser_if* ExperimentDesign_if::getProcessAnalyser() const [pure virtual]

Implemented in ExperimentDesignDummyImpl, and ExperimentDesignDefaultImpl1.

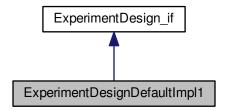
The documentation for this class was generated from the following file:

• ExperimentDesign_if.h

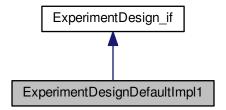
5.25 ExperimentDesignDefaultImpl1 Class Reference

#include <ExperimentDesignDefaultImpl1.h>

Inheritance diagram for ExperimentDesignDefaultImpl1:



Collaboration diagram for ExperimentDesignDefaultImpl1:



- ExperimentDesignDefaultImpl1 ()
- ExperimentDesignDefaultImpl1 (const ExperimentDesignDefaultImpl1 &orig)
- virtual ~ExperimentDesignDefaultImpl1 ()
- virtual ProcessAnalyser_if * getProcessAnalyser () const
- virtual bool generate2krScenarioExperiments ()
- virtual bool calculateContributionAndCoefficients ()
- virtual std::list< FactorOrInteractionContribution * > * getContributions () const

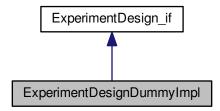
```
5.25.1
        Constructor & Destructor Documentation
5.25.1.1 ExperimentDesignDefaultImpl1::ExperimentDesignDefaultImpl1 ( )
5.25.1.2 ExperimentDesignDefaultImpl1::ExperimentDesignDefaultImpl1 ( const ExperimentDesignDefaultImpl1 & orig )
5.25.1.3 ExperimentDesignDefaultImpl1::~ExperimentDesignDefaultImpl1() [virtual]
5.25.2 Member Function Documentation
5.25.2.1 bool ExperimentDesignDefaultImpl1::calculateContributionAndCoefficients ( ) [virtual]
Implements ExperimentDesign if.
5.25.2.2 bool ExperimentDesignDefaultImpl1::generate2krScenarioExperiments ( ) [virtual]
Implements ExperimentDesign_if.
5.25.2.3 std::list < FactorOrInteractionContribution *>* ExperimentDesignDefaultImpl1::getContributions ( ) const
         [virtual]
Implements ExperimentDesign_if.
5.25.2.4 ProcessAnalyser_if * ExperimentDesignDefaultImpl1::getProcessAnalyser( ) const [virtual]
Implements ExperimentDesign_if.
The documentation for this class was generated from the following files:
```

- ExperimentDesignDefaultImpl1.h
- ExperimentDesignDefaultImpl1.cpp

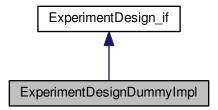
5.26 ExperimentDesignDummyImpl Class Reference

#include <ExperimentDesignDummyImpl.h>

Inheritance diagram for ExperimentDesignDummyImpl:



Collaboration diagram for ExperimentDesignDummyImpl:



- ExperimentDesignDummyImpl ()
- ExperimentDesignDummyImpl (const ExperimentDesignDummyImpl &orig)
- virtual ~ExperimentDesignDummyImpl ()
- virtual ProcessAnalyser_if * getProcessAnalyser () const
- virtual bool generate2krScenarioExperiments ()
- virtual bool calculateContributionAndCoefficients ()
- virtual std::list< FactorOrInteractionContribution * > * getContributions () const

```
5.26.1 Constructor & Destructor Documentation
5.26.1.1 ExperimentDesignDummyImpl::ExperimentDesignDummyImpl ( )
5.26.1.2 ExperimentDesignDummyImpl::ExperimentDesignDummyImpl & orig )
5.26.1.3 ExperimentDesignDummyImpl::~ExperimentDesignDummyImpl() [virtual]
5.26.2 Member Function Documentation
5.26.2.1 bool ExperimentDesignDummyImpl::calculateContributionAndCoefficients ( ) [virtual]
Implements ExperimentDesign_if.
5.26.2.2 bool ExperimentDesignDummyImpl::generate2krScenarioExperiments ( ) [virtual]
Implements ExperimentDesign_if.
5.26.2.3 std::list < FactorOrInteractionContribution *>* ExperimentDesignDummyImpl::getContributions ( ) const
        [virtual]
Implements ExperimentDesign if.
5.26.2.4 ProcessAnalyser_if * ExperimentDesignDummyImpl::getProcessAnalyser( ) const [virtual]
Implements ExperimentDesign_if.
```

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

- ExperimentDesignDummyImpl.h
- ExperimentDesignDummyImpl.cpp

5.27 FactorOrInteractionContribution Class Reference

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

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

5.27.1 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.

5.27.2 Constructor & Destructor Documentation

- 5.27.2.1 FactorOrInteractionContribution::FactorOrInteractionContribution (double *contribution*, double *modelCoefficient*, std::list< SimulationControl * > * controls)
- 5.27.2.2 FactorOrInteractionContribution::FactorOrInteractionContribution (const FactorOrInteractionContribution & orig)
- 5.27.2.3 FactorOrInteractionContribution::~FactorOrInteractionContribution()

5.27.3 Member Function Documentation

- 5.27.3.1 double FactorOrInteractionContribution::getContribution () const
- 5.27.3.2 std::list < SimulationControl * > * FactorOrInteractionContribution::getControls () const
- 5.27.3.3 double FactorOrInteractionContribution::getModelCoefficient () const

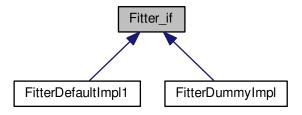
The documentation for this class was generated from the following files:

- FactorOrInteractionContribution.h
- FactorOrInteractionContribution.cpp

5.28 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 *sqrerror, double *min, double *max)=0
- virtual void fitTriangular (double *sqrerror, double *min, double *mo, double *max)=0
- virtual void fitNormal (double *sqrerror, double *avg, double *stddev)=0
- virtual void fitExpo (double *sqrerror, double *avg1)=0
- virtual void fitErlang (double *sqrerror, double *avg, double *m)=0
- virtual void fitBeta (double *sqrerror, double *alpha, double *beta, double *infLimit, double *supLimit)=0
- virtual void fitWeibull (double *sgrerror, double *alpha, double *scale)=0
- virtual void fitAll (double *sqrerror, std::string *name)=0
- virtual void setDataFilename (std::string dataFilename)=0
- virtual std::string getDataFilename ()=0

5.28.1 Member Function Documentation

5.28.1.1 virtual void Fitter_if::fitAll (double * sqrerror, std::string * name) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

Here is the caller graph for this function:



5.28.1.2 virtual void Fitter_if::fitBeta (double * sqrerror, double * alpha, double * beta, double * infLimit, double * supLimit) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

5.28.1.3 virtual void Fitter_if::fitErlang (double * sqrerror, double * avg, double * m) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

5.28.1.4 virtual void Fitter_if::fitExpo (double * *sqrerror*, double * *avg1*) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

5.28.1.5 virtual void Fitter_if::fitNormal (double * sqrerror, double * avg, double * stddev) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

Here is the caller graph for this function:



5.28.1.6 virtual void Fitter_if::fitTriangular (double * sqrerror, double * min, double * mo, double * max) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

Here is the caller graph for this function:



5.28.1.7 virtual void Fitter_if::fitUniform (double * sqrerror, double * min, double * max) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

Here is the caller graph for this function:



5.28.1.8 virtual void Fitter_if::fitWeibull (double * sqrerror, double * alpha, double * scale) [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

5.28.1.9 virtual std::string Fitter_if::getDataFilename() [pure virtual]

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

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

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

Here is the caller graph for this function:



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

Implemented in FitterDefaultImpl1, and FitterDummyImpl.

Here is the caller graph for this function:



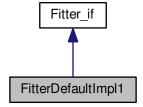
The documentation for this class was generated from the following file:

• Fitter_if.h

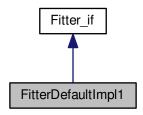
5.29 FitterDefaultImpl1 Class Reference

#include <FitterDefaultImpl1.h>

Inheritance diagram for FitterDefaultImpl1:



Collaboration diagram for FitterDefaultImpl1:



Public Member Functions

- FitterDefaultImpl1 ()
- FitterDefaultImpl1 (const FitterDefaultImpl1 &orig)
- virtual ∼FitterDefaultImpl1 ()
- bool isNormalDistributed (double confidencelevel)
- void fitUniform (double *sqrerror, double *min, double *max)
- void fitTriangular (double *sgrerror, double *min, double *mo, double *max)
- void fitNormal (double *sqrerror, double *avg, double *stddev)
- void fitExpo (double *sqrerror, double *avg1)
- void fitErlang (double *sqrerror, double *avg, double *m)
- void fitBeta (double *sqrerror, double *alpha, double *beta, double *infLimit, double *supLimit)
- void fitWeibull (double *sqrerror, double *alpha, double *scale)
- void fitAll (double *sqrerror, std::string *name)
- void setDataFilename (std::string dataFilename)
- std::string getDataFilename ()

5.29.1 Constructor & Destructor Documentation

- 5.29.1.1 FitterDefaultImpl1::FitterDefaultImpl1 ()
- 5.29.1.2 FitterDefaultImpl1::FitterDefaultImpl1 (const FitterDefaultImpl1 & orig)
- 5.29.1.3 FitterDefaultImpl1:: \sim FitterDefaultImpl1() [virtual]
- 5.29.2 Member Function Documentation
- **5.29.2.1** void FitterDefaultImpl1::fitAll (double * sqrerror, std::string * name) [virtual]

Implements Fitter_if.

```
5.29.2.2 void FitterDefaultImpl1::fitBeta ( double * sqrerror, double * alpha, double * beta, double * infLimit, double *
         supLimit ) [virtual]
Implements Fitter_if.
5.29.2.3 void FitterDefaultImpl1::fitErlang ( double * sqrerror, double * avg, double * m ) [virtual]
Implements Fitter_if.
5.29.2.4 void FitterDefaultImpl1::fitExpo ( double * sqrerror, double * avg1 ) [virtual]
Implements Fitter_if.
5.29.2.5 void FitterDefaultImpl1::fitNormal ( double * sqrerror, double * avg, double * stddev ) [virtual]
Implements Fitter if.
5.29.2.6 void FitterDefaultImpl1::fitTriangular ( double * sqrerror, double * min, double * mo, double * max ) [virtual]
Implements Fitter_if.
5.29.2.7 void FitterDefaultImpl1::fitUniform ( double * sqrerror, double * min, double * max ) [virtual]
Implements Fitter_if.
5.29.2.8 void FitterDefaultImpl1::fitWeibull ( double * sqrerror, double * alpha, double * scale ) [virtual]
Implements Fitter_if.
5.29.2.9 std::string FitterDefaultImpl1::getDataFilename() [virtual]
Implements Fitter if.
5.29.2.10 bool FitterDefaultImpl1::isNormalDistributed ( double confidencelevel ) [virtual]
Implements Fitter_if.
5.29.2.11 void FitterDefaultImpl1::setDataFilename ( std::string dataFilename ) [virtual]
Implements Fitter_if.
```

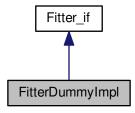
The documentation for this class was generated from the following files:

- FitterDefaultImpl1.h
- FitterDefaultImpl1.cpp

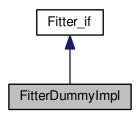
5.30 FitterDummyImpl Class Reference

#include <FitterDummyImpl.h>

Inheritance diagram for FitterDummyImpl:



Collaboration diagram for FitterDummyImpl:



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

```
5.30.1 Constructor & Destructor Documentation
5.30.1.1 FitterDummyImpl::FitterDummyImpl ( )
5.30.1.2 FitterDummyImpl::FitterDummyImpl ( const FitterDummyImpl & orig )
5.30.1.3 FitterDummyImpl::~FitterDummyImpl ( )
5.30.2 Member Function Documentation
5.30.2.1 void FitterDummyImpl::fitAll ( double * sqrerror, std::string * name ) [virtual]
Implements Fitter_if.
5.30.2.2 void FitterDummyImpl::fitBeta ( double * sqrerror, double * alpha, double * beta, double * infLimit, double *
         supLimit ) [virtual]
Implements Fitter if.
5.30.2.3 void FitterDummyImpl::fitErlang ( double * sqrerror, double * avg, double * m ) [virtual]
Implements Fitter_if.
5.30.2.4 void FitterDummyImpl::fitExpo ( double * sqrerror, double * avg1 ) [virtual]
Implements Fitter if.
5.30.2.5 void FitterDummyImpl::fitNormal ( double * sqrerror, double * avg, double * stddev ) [virtual]
Implements Fitter_if.
5.30.2.6 void FitterDummyImpl::fitTriangular ( double * sqrerror, double * min, double * mo, double * max ) [virtual]
Implements Fitter_if.
5.30.2.7 void FitterDummylmpl::fitUniform ( double * sqrerror, double * min, double * max ) [virtual]
Implements Fitter_if.
5.30.2.8 void FitterDummylmpl::fitWeibull ( double * sqrerror, double * alpha, double * scale ) [virtual]
Implements Fitter_if.
```

```
5.30.2.9 std::string FitterDummyImpl::getDataFilename() [virtual]
Implements Fitter_if.
5.30.2.10 bool FitterDummyImpl::isNormalDistributed( double confidencelevel ) [virtual]
Implements Fitter_if.
5.30.2.11 void FitterDummyImpl::setDataFilename( std::string dataFilename) [virtual]
Implements Fitter_if.
```

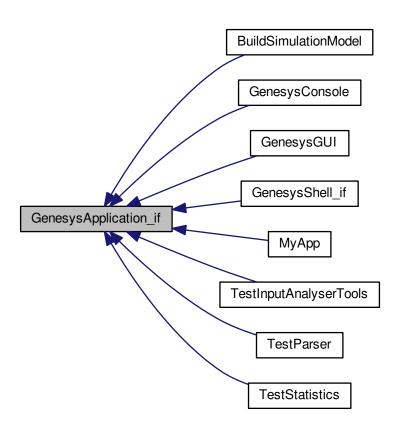
The documentation for this class was generated from the following files:

- · FitterDummyImpl.h
- FitterDummyImpl.cpp

5.31 GenesysApplication_if Class Reference

#include <GenesysApplication_if.h>

Inheritance diagram for GenesysApplication_if:



Public Member Functions

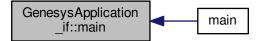
• virtual int main (int argc, char **argv)=0

5.31.1 Member Function Documentation

5.31.1.1 virtual int GenesysApplication_if::main (int argc, char ** argv) [pure virtual]

Implemented in GenesysConsole, MyApp, GenesysGUI, TestInputAnalyserTools, TestParser, BuildSimulation \leftarrow Model, and TestStatistics.

Here is the caller graph for this function:



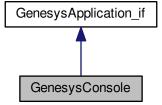
The documentation for this class was generated from the following file:

· GenesysApplication_if.h

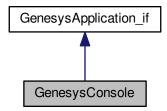
5.32 GenesysConsole Class Reference

#include <GenesysConsole.h>

Inheritance diagram for GenesysConsole:



Collaboration diagram for GenesysConsole:



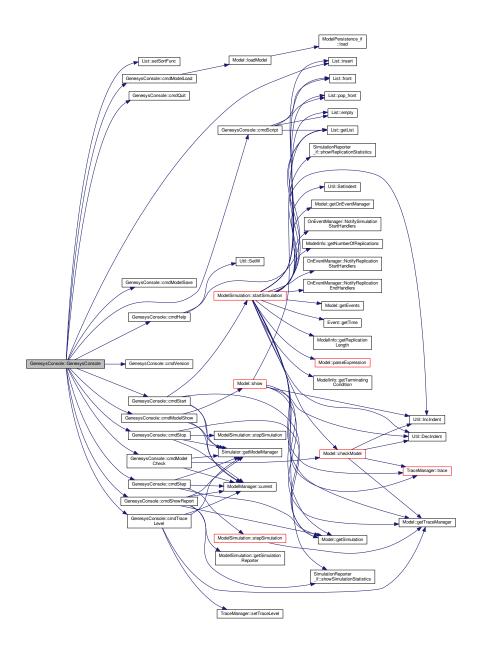
Public Member Functions

- GenesysConsole ()
- GenesysConsole (const GenesysConsole &orig)
- virtual ∼GenesysConsole ()
- virtual int main (int argc, char **argv)
- void cmdScript ()
- void cmdHelp ()
- void cmdQuit ()
- void cmdModelLoad ()
- void cmdModelCheck ()
- void cmdStart ()
- void cmdStep ()
- void cmdStop ()
- void cmdShowReport ()
- void cmdModelSave ()
- void cmdModelShow ()
- void cmdVersion ()
- void cmdTraceLevel ()

5.32.1 Constructor & Destructor Documentation

5.32.1.1 GenesysConsole::GenesysConsole ()

Here is the call graph for this function:

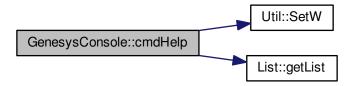


- 5.32.1.2 GenesysConsole::GenesysConsole (const GenesysConsole & orig)
- **5.32.1.3** GenesysConsole::~GenesysConsole() [virtual]

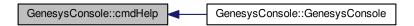
5.32.2 Member Function Documentation

5.32.2.1 void GenesysConsole::cmdHelp()

Here is the call graph for this function:

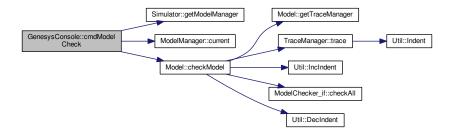


Here is the caller graph for this function:



5.32.2.2 void GenesysConsole::cmdModelCheck ()

Here is the call graph for this function:





5.32.2.3 void GenesysConsole::cmdModelLoad ()

Here is the call graph for this function:



Here is the caller graph for this function:



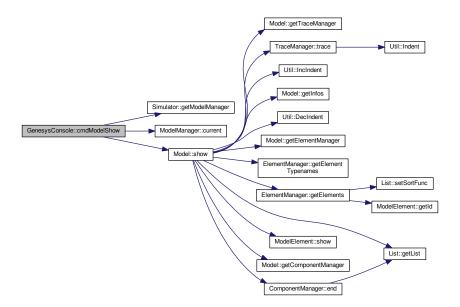
5.32.2.4 void GenesysConsole::cmdModelSave ()

Here is the caller graph for this function:

GenesysConsole::GenesysConsole

5.32.2.5 void GenesysConsole::cmdModelShow()

Here is the call graph for this function:



Here is the caller graph for this function:

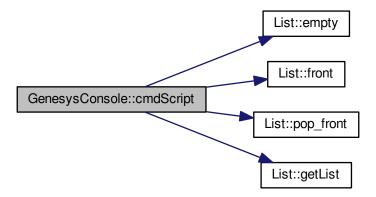


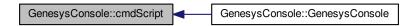
5.32.2.6 void GenesysConsole::cmdQuit ()



5.32.2.7 void GenesysConsole::cmdScript()

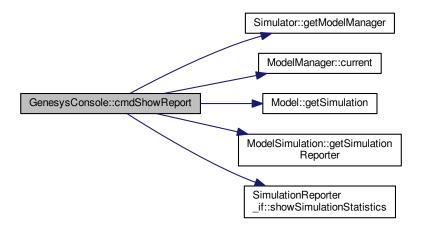
Here is the call graph for this function:





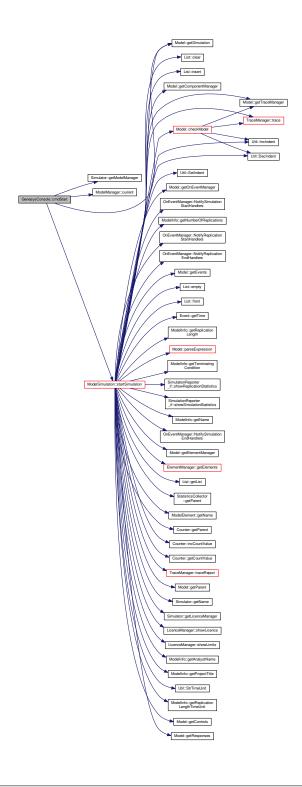
5.32.2.8 void GenesysConsole::cmdShowReport ()

Here is the call graph for this function:





5.32.2.9 void GenesysConsole::cmdStart ()

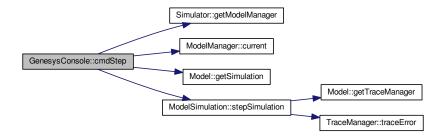


Here is the caller graph for this function:



5.32.2.10 void GenesysConsole::cmdStep ()

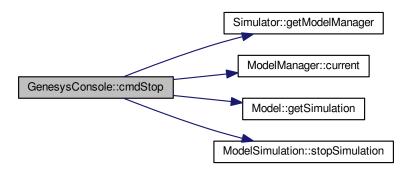
Here is the call graph for this function:





5.32.2.11 void GenesysConsole::cmdStop()

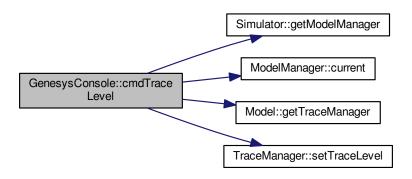
Here is the call graph for this function:



Here is the caller graph for this function:



5.32.2.12 void GenesysConsole::cmdTraceLevel ()



Here is the caller graph for this function:



5.32.2.13 void GenesysConsole::cmdVersion ()

Here is the caller graph for this function:



5.32.2.14 int GenesysConsole::main (int argc, char ** argv) [virtual]

Implements GenesysApplication_if.

Here is the call graph for this function:



Here is the caller graph for this function:



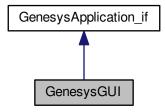
The documentation for this class was generated from the following files:

- GenesysConsole.h
- GenesysConsole.cpp

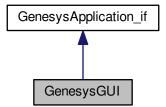
5.33 GenesysGUI Class Reference

#include <GenesysGUI.h>

Inheritance diagram for GenesysGUI:



Collaboration diagram for GenesysGUI:



Public Member Functions

- GenesysGUI ()
- GenesysGUI (const GenesysGUI &orig)
- virtual ∼GenesysGUI ()
- virtual int main (int argc, char **argv)

5.33.1 Constructor & Destructor Documentation

- 5.33.1.1 GenesysGUI::GenesysGUI()
- 5.33.1.2 GenesysGUI::GenesysGUI (const GenesysGUI & orig)

```
5.33.1.3 GenesysGUI::~GenesysGUI( ) [virtual]
```

5.33.2 Member Function Documentation

```
5.33.2.1 int GenesysGUI::main ( int argc, char ** argv ) [virtual]
```

Implements GenesysApplication_if.

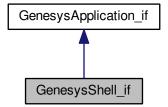
The documentation for this class was generated from the following files:

- · GenesysGUI.h
- GenesysGUI.cpp

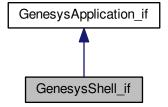
5.34 GenesysShell_if Class Reference

```
#include <GenesysShell_if.h>
```

Inheritance diagram for GenesysShell_if:



Collaboration diagram for GenesysShell_if:



Public Member Functions

- virtual void openModel (std::string filename)=0
- virtual void saveModelAs (std::string filename)=0
- virtual void saveModel ()=0
- virtual void listElements ()=0
- virtual void listComponents ()=0
- virtual void listHosts ()=0
- virtual void listPlugins ()=0
- virtual void deleteTraceFiles ()=0
- virtual void traceLevel (Util::TraceLevel tracelevel)=0
- virtual void addPlugin (std::string filename)=0
- virtual void addFromFile (std::string filename)=0
- virtual void readCommandsFromFile (std::string filename)=0
- virtual void redirectTrace (std::string trace, std::string dest, std::string filename)=0
- virtual void closeModel ()=0
- virtual void createModel ()=0
- virtual void execLinuxCommand (std::string command)=0
- virtual void verboseMode (bool on)=0
- virtual void check ()=0
- virtual void getGenesysInfo ()=0
- virtual void getCommandLine ()=0
- virtual void sendFile (std::string filename, std::string hostname, std::string portname)=0
- virtual void setActivationCode (std::string code)=0
- virtual void receiveFile (std::string filename)=0
- virtual void startSimulation ()=0
- virtual void stepSimulation ()=0
- virtual void stopSimulation ()=0
- virtual void showInit ()=0
- virtual void showHelp ()=0
- virtual void showHostName ()=0

5.34.1 Member Function Documentation

```
5.34.1.1 virtual void GenesysShell_if::addFromFile ( std::string filename ) [pure virtual]
5.34.1.2 virtual void GenesysShell_if::addPlugin ( std::string filename ) [pure virtual]
5.34.1.3 virtual void GenesysShell_if::check( ) [pure virtual]
5.34.1.4 virtual void GenesysShell_if::closeModel( ) [pure virtual]
5.34.1.5 virtual void GenesysShell_if::createModel( ) [pure virtual]
5.34.1.6 virtual void GenesysShell_if::deleteTraceFiles( ) [pure virtual]
5.34.1.7 virtual void GenesysShell_if::execLinuxCommand ( std::string command ) [pure virtual]
5.34.1.8 virtual void GenesysShell_if::getCommandLine( ) [pure virtual]
```

```
5.34.1.9 virtual void GenesysShell_if::getGenesysInfo() [pure virtual]
5.34.1.10 virtual void GenesysShell_if::listComponents() [pure virtual]
5.34.1.11 virtual void GenesysShell_if::listElements() [pure virtual]
5.34.1.12 virtual void GenesysShell_if::listHosts() [pure virtual]
5.34.1.13 virtual void GenesysShell_if::listPlugins() [pure virtual]
5.34.1.14 virtual void GenesysShell_if::openModel( std::string filename ) [pure virtual]
5.34.1.15 virtual void GenesysShell_if::readCommandsFromFile( std::string filename ) [pure virtual]
5.34.1.16 virtual void GenesysShell_if::receiveFile ( std::string filename ) [pure virtual]
5.34.1.17 virtual void GenesysShell_if::redirectTrace ( std::string trace, std::string dest, std::string filename ) [pure
         virtual]
5.34.1.18 virtual void GenesysShell_if::saveModel() [pure virtual]
5.34.1.19 virtual void GenesysShell_if::saveModelAs ( std::string filename ) [pure virtual]
5.34.1.20 virtual void GenesysShell if::sendFile ( std::string filename, std::string hostname, std::string portname ) [pure
         virtual]
5.34.1.21 virtual void GenesysShell_if::setActivationCode ( std::string code ) [pure virtual]
5.34.1.22 virtual void GenesysShell_if::showHelp() [pure virtual]
5.34.1.23 virtual void GenesysShell_if::showHostName() [pure virtual]
5.34.1.24 virtual void GenesysShell_if::showlnit() [pure virtual]
5.34.1.25 virtual void GenesysShell_if::startSimulation() [pure virtual]
5.34.1.26 virtual void GenesysShell_if::stepSimulation() [pure virtual]
5.34.1.27 virtual void GenesysShell_if::stopSimulation() [pure virtual]
5.34.1.28 virtual void GenesysShell if::traceLevel ( Util::TraceLevel tracelevel ) [pure virtual]
5.34.1.29 virtual void GenesysShell_if::verboseMode ( bool on ) [pure virtual]
```

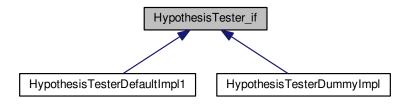
The documentation for this class was generated from the following file:

GenesysShell_if.h

5.35 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, H1← Comparition comp)=0
- virtual double testProportion (double confidencelevel, std::string secondPopulationDataFilename, H1← Comparition comp)=0
- virtual double testVariance (double confidencelevel, std::string secondPopulationDataFilename, H1← Comparition comp)=0
- virtual void setDataFilename (std::string dataFilename)=0
- virtual std::string getDataFilename ()=0

5.35.1 Detailed Description

Interface for parametric hypothesis tests based on a datafile.

5.35.2 Member Enumeration Documentation

5.35.2.1 enum HypothesisTester_if::H1Comparition

Enumerator

DIFFERENT LESS_THAN GREATER_THAN

5.35.3 Member Function Documentation

5.35.3.1 virtual std::string HypothesisTester_if::getDataFilename() [pure virtual]

 $Implemented \ in \ Hypothesis Tester Default Impl 1, \ and \ Hypothesis Tester Dummy Impl.$

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

Implemented in HypothesisTesterDefaultImpl1, and HypothesisTesterDummyImpl.

Here is the caller graph for this function:



5.35.3.3 virtual double HypothesisTester_if::testAverage (double *confidencelevel*, double *avg*, H1Comparition *comp*) [pure virtual]

Implemented in HypothesisTesterDefaultImpl1, and HypothesisTesterDummyImpl.

Here is the caller graph for this function:



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

Implemented in HypothesisTesterDefaultImpl1, and HypothesisTesterDummyImpl.

5.35.3.5 virtual double HypothesisTester_if::testProportion (double *confidencelevel*, double *prop*, H1Comparition *comp*)

[pure virtual]

Implemented in HypothesisTesterDefaultImpl1, and HypothesisTesterDummyImpl.

5.35.3.6 virtual double HypothesisTester_if::testProportion (double *confidencelevel*, std::string secondPopulationDataFilename, H1Comparition comp) [pure virtual]

Implemented in HypothesisTesterDefaultImpl1, and HypothesisTesterDummyImpl.

5.35.3.7 virtual double HypothesisTester_if::testVariance (double *confidencelevel*, double *var*, H1Comparition *comp*)

[pure virtual]

Implemented in HypothesisTesterDefaultImpl1, and HypothesisTesterDummyImpl.

Here is the caller graph for this function:



5.35.3.8 virtual double HypothesisTester_if::testVariance (double confidencelevel, std::string secondPopulationDataFilename, H1Comparition comp) [pure virtual]

Implemented in HypothesisTesterDefaultImpl1, and HypothesisTesterDummyImpl.

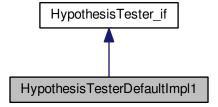
The documentation for this class was generated from the following file:

• HypothesisTester_if.h

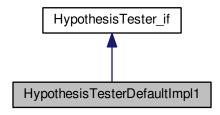
5.36 HypothesisTesterDefaultImpl1 Class Reference

#include <HypothesisTesterDefaultImpl1.h>

Inheritance diagram for HypothesisTesterDefaultImpl1:



Collaboration diagram for HypothesisTesterDefaultImpl1:



Public Member Functions

- HypothesisTesterDefaultImpl1 ()
- HypothesisTesterDefaultImpl1 (const HypothesisTesterDefaultImpl1 &orig)
- virtual ∼HypothesisTesterDefaultImpl1 ()
- virtual double testAverage (double confidencelevel, double avg, H1Comparition comp)
- virtual double testProportion (double confidencelevel, double prop, H1Comparition comp)
- virtual double testVariance (double confidencelevel, double var, H1Comparition comp)
- virtual double testAverage (double confidencelevel, std::string secondPopulationDataFilename, H1← Comparition comp)
- virtual double testProportion (double confidencelevel, std::string secondPopulationDataFilename, H1← Comparition comp)
- virtual double testVariance (double confidencelevel, std::string secondPopulationDataFilename, H1← Comparition comp)
- virtual void setDataFilename (std::string dataFilename)
- virtual std::string getDataFilename ()

Additional Inherited Members

5.36.1 Constructor & Destructor Documentation

- 5.36.1.1 HypothesisTesterDefaultImpl1::HypothesisTesterDefaultImpl1 ()
- $5.36.1.2 \quad \text{HypothesisTesterDefaultImpl1}: \text{HypothesisTesterDefaultImpl1} \ (\ const \ \text{HypothesisTesterDefaultImpl1} \ \& \ orig \)$
- **5.36.1.3** HypothesisTesterDefaultImpl1::~HypothesisTesterDefaultImpl1() [virtual]
- 5.36.2 Member Function Documentation
- **5.36.2.1** std::string HypothesisTesterDefaultImpl1::getDataFilename() [virtual]

Implements HypothesisTester_if.

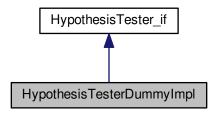
```
5.36.2.2 void HypothesisTesterDefaultImpl1::setDataFilename ( std::string dataFilename ) [virtual]
Implements HypothesisTester_if.
5.36.2.3 double HypothesisTesterDefaultImpl1::testAverage ( double confidencelevel, double avg, H1Comparition comp )
         [virtual]
Implements HypothesisTester_if.
5.36.2.4 double HypothesisTesterDefaultImpl1::testAverage ( double confidencelevel, std::string
         secondPopulationDataFilename, H1Comparition comp ) [virtual]
Implements HypothesisTester_if.
5.36.2.5 double HypothesisTesterDefaultImpl1::testProportion ( double confidencelevel, double prop, H1Comparition comp
         ) [virtual]
Implements HypothesisTester if.
5.36.2.6 double HypothesisTesterDefaultImpl1::testProportion ( double confidencelevel, std::string
         secondPopulationDataFilename, H1Comparition comp ) [virtual]
Implements HypothesisTester_if.
5.36.2.7 double HypothesisTesterDefaultImpl1::testVariance ( double confidencelevel, double var, H1Comparition comp )
         [virtual]
Implements HypothesisTester_if.
5.36.2.8 double HypothesisTesterDefaultImpl1::testVariance ( double confidencelevel, std::string
         secondPopulationDataFilename, H1Comparition comp ) [virtual]
Implements HypothesisTester_if.
The documentation for this class was generated from the following files:
```

- · HypothesisTesterDefaultImpl1.h
- HypothesisTesterDefaultImpl1.cpp

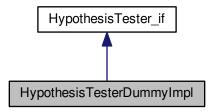
5.37 HypothesisTesterDummyImpl Class Reference

#include <HypothesisTesterDummyImpl.h>

Inheritance diagram for HypothesisTesterDummyImpl:



Collaboration diagram for HypothesisTesterDummyImpl:



Public Member Functions

- HypothesisTesterDummyImpl ()
- HypothesisTesterDummyImpl (const HypothesisTesterDummyImpl &orig)
- ∼HypothesisTesterDummyImpl ()
- virtual double testAverage (double confidencelevel, double avg, H1Comparition comp)
- virtual double testProportion (double confidencelevel, double prop, H1Comparition comp)
- virtual double testVariance (double confidencelevel, double var, H1Comparition comp)
- virtual double testAverage (double confidencelevel, std::string secondPopulationDataFilename, H1← Comparition comp)
- virtual double testProportion (double confidencelevel, std::string secondPopulationDataFilename, H1←
 Comparition comp)
- virtual double testVariance (double confidencelevel, std::string secondPopulationDataFilename, H1←
 Comparition comp)
- virtual void setDataFilename (std::string dataFilename)
- virtual std::string getDataFilename ()

Additional Inherited Members

```
5.37.1 Constructor & Destructor Documentation
5.37.1.1 HypothesisTesterDummyImpl::HypothesisTesterDummyImpl ( )
5.37.1.2 HypothesisTesterDummylmpl::HypothesisTesterDummylmpl ( const HypothesisTesterDummylmpl & orig )
5.37.1.3 HypothesisTesterDummyImpl::~HypothesisTesterDummyImpl ( )
5.37.2 Member Function Documentation
5.37.2.1 std::string HypothesisTesterDummyImpl::getDataFilename() [virtual]
Implements HypothesisTester_if.
5.37.2.2 void HypothesisTesterDummyImpl::setDataFilename ( std::string dataFilename ) [virtual]
Implements HypothesisTester_if.
5.37.2.3 double HypothesisTesterDummyImpl::testAverage ( double confidencelevel, double avg, H1Comparition comp )
         [virtual]
Implements HypothesisTester_if.
5.37.2.4 double HypothesisTesterDummyImpl::testAverage ( double confidencelevel, std::string
        secondPopulationDataFilename, H1Comparition comp ) [virtual]
Implements HypothesisTester_if.
5.37.2.5 double HypothesisTesterDummyImpl::testProportion ( double confidencelevel, double prop, H1Comparition comp
        ) [virtual]
Implements HypothesisTester_if.
5.37.2.6 double HypothesisTesterDummyImpl::testProportion ( double confidencelevel, std::string
         secondPopulationDataFilename, H1Comparition comp ) [virtual]
Implements HypothesisTester if.
5.37.2.7 double HypothesisTesterDummyImpl::testVariance ( double confidencelevel, double var, H1Comparition comp )
         [virtual]
Implements HypothesisTester_if.
```

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

Implements HypothesisTester_if.

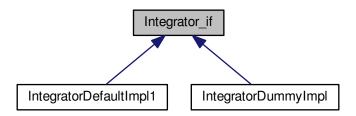
The documentation for this class was generated from the following files:

- · HypothesisTesterDummyImpl.h
- HypothesisTesterDummyImpl.cpp

5.38 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, double, double p2, double p3, double p4, double p5)=0

5.38.1 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.

5.38.2 Member Function Documentation

5.38.2.1 virtual double Integrator_if::getPrecision() [pure virtual]

Implemented in IntegratorDefaultImpl1, and IntegratorDummyImpl.

5.38.2.2 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:



5.38.2.3 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.

5.38.2.4 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.

5.38.2.5 virtual double Integrator_if::integrate (double *min*, double *max*, double(*)(double, double, double, double, double, double *p4*, double *p4*, double *p5*) [pure virtual]

Implemented in IntegratorDefaultImpl1, and IntegratorDummyImpl.

5.38.2.6 virtual void Integrator_if::setPrecision (double *e* **)** [pure virtual]

Implemented in IntegratorDefaultImpl1, and IntegratorDummyImpl.

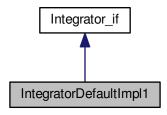
The documentation for this class was generated from the following file:

· Integrator_if.h

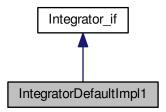
5.39 Integrator Default Impl 1 Class Reference

#include <IntegratorDefaultImpl1.h>

Inheritance diagram for IntegratorDefaultImpl1:



Collaboration diagram for IntegratorDefaultImpl1:



Public Member Functions

- IntegratorDefaultImpl1 ()
- IntegratorDefaultImpl1 (const IntegratorDefaultImpl1 &orig)
- virtual \sim 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)

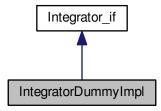
```
5.39.1
        Constructor & Destructor Documentation
5.39.1.1 IntegratorDefaultImpl1::IntegratorDefaultImpl1 ( )
https://codereview.stackexchange.com/questions/200289/implementing-numerical-integration
5.39.1.2 IntegratorDefaultImpl1::IntegratorDefaultImpl1 ( const IntegratorDefaultImpl1 & orig )
5.39.1.3 IntegratorDefaultImpl1::~IntegratorDefaultImpl1() [virtual]
5.39.2
        Member Function Documentation
5.39.2.1 double IntegratorDefaultImpl1::getPrecision() [virtual]
Implements Integrator if.
5.39.2.2 double IntegratorDefaultImpl1::integrate ( double min, double max, double(*)(double, double) f, double p2 )
         [virtual]
Implements Integrator_if.
5.39.2.3 double IntegratorDefaultImpl1::integrate ( double min, double max, double(*)(double, double, double) f, double p2,
        double p3 ) [virtual]
Implements Integrator_if.
5.39.2.4 double IntegratorDefaultImpl1::integrate ( double min, double max, double(*)(double, double, double) f,
        double p2, double p3, double p4 ) [virtual]
Implements Integrator_if.
5.39.2.5 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.
5.39.2.6 void Integrator DefaultImpl1::setPrecision ( double e ) [virtual]
Implements Integrator_if.
The documentation for this class was generated from the following files:
```

- IntegratorDefaultImpl1.h IntegratorDefaultImpl1.cpp

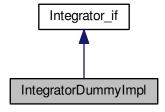
5.40 Integrator Dummy Impl Class Reference

#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), double p2, double p3, double p4)
- double integrate (double min, double max, double(*f)(double, double, double, double, double, double, double p2, double p3, double p4, double p5)

```
5.40.1
        Constructor & Destructor Documentation
        IntegratorDummyImpl::IntegratorDummyImpl ( )
5.40.1.1
5.40.1.2 IntegratorDummyImpl::IntegratorDummyImpl ( const IntegratorDummyImpl & orig )
5.40.1.3 IntegratorDummyImpl::~IntegratorDummyImpl ( )
5.40.2
        Member Function Documentation
5.40.2.1 double IntegratorDummyImpl::getPrecision() [virtual]
Implements Integrator_if.
5.40.2.2 double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double) f, double p2 )
         [virtual]
Implements Integrator_if.
5.40.2.3 double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double, double) f, double p2,
         double p3 ) [virtual]
Implements Integrator_if.
5.40.2.4 double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double, double, double) f,
         double p2, double p3, double p4 ) [virtual]
Implements Integrator_if.
5.40.2.5 double IntegratorDummyImpl::integrate ( double min, double max, double(*)(double, double, double, double)
         f, double p2, double p3, double p4, double p5 ) [virtual]
Implements Integrator_if.
5.40.2.6 void IntegratorDummyImpl::setPrecision ( double e ) [virtual]
Implements Integrator if.
```

- IntegratorDummyImpl.h IntegratorDummyImpl.cpp

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

5.41 LicenceManager Class Reference

```
#include <LicenceManager.h>
```

Public Member Functions

- LicenceManager (Simulator *simulator)
- LicenceManager (const LicenceManager &orig)
- virtual ~LicenceManager ()
- · const std::string showLicence () const
- · const std::string showLimits () const
- const std::string showActivationCode () const
- bool lookforActivationCode ()
- bool insertActivationCode ()
- void removeActivationCode ()
- unsigned int getModelComponentsLimit ()
- unsigned int getModelElementsLimit ()
- unsigned int getEntityLimit ()
- unsigned int getHostsLimit ()
- unsigned int getThreadsLimit ()

5.41.1 Constructor & Destructor Documentation

```
5.41.1.1 LicenceManager::LicenceManager ( Simulator * simulator )

5.41.1.2 LicenceManager::LicenceManager ( const LicenceManager & orig )

5.41.1.3 LicenceManager::~LicenceManager ( ) [virtual]

5.41.2 Member Function Documentation

5.41.2.1 unsigned int LicenceManager::getEntityLimit ( )

5.41.2.2 unsigned int LicenceManager::getHostsLimit ( )

5.41.2.3 unsigned int LicenceManager::getModelComponentsLimit ( )

5.41.2.4 unsigned int LicenceManager::getModelElementsLimit ( )

5.41.2.5 unsigned int LicenceManager::getThreadsLimit ( )

5.41.2.6 bool LicenceManager::insertActivationCode ( )

5.41.2.7 bool LicenceManager::lookforActivationCode ( )
```

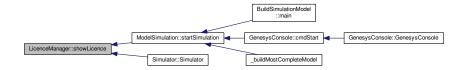
5.41.2.9 const std::string LicenceManager::showActivationCode () const

Here is the caller graph for this function:



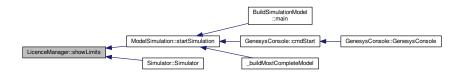
5.41.2.10 const std::string LicenceManager::showLicence () const

Here is the caller graph for this function:



5.41.2.11 const std::string LicenceManager::showLimits () const

Here is the caller graph for this function:



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

- LicenceManager.h
- · LicenceManager.cpp

5.42 LinkedBy Class Reference

#include <LinkedBy.h>

Public Member Functions

- LinkedBy ()
- LinkedBy (const LinkedBy &orig)
- virtual ∼LinkedBy ()
- void addLink ()
- void removeLink ()
- bool isLinked ()

5.42.1 Constructor & Destructor Documentation

```
5.42.1.1 LinkedBy::LinkedBy ( )

5.42.1.2 LinkedBy::LinkedBy ( const LinkedBy & orig )

5.42.1.3 LinkedBy::~LinkedBy ( ) [virtual]

5.42.2 Member Function Documentation

5.42.2.1 void LinkedBy::addLink ( )

5.42.2.2 bool LinkedBy::isLinked ( )
```

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

- · LinkedBy.h
- · LinkedBy.cpp

5.42.2.3 void LinkedBy::removeLink()

5.43 List < T > Class Template Reference

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

Public Types

• using CompFunct = 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 front ()
- T last ()
- T previous ()
- T actual ()
- void setSortFunc (CompFunct _sortFunc)

5.43.1 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.

5.43.2 Member Typedef Documentation

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

5.43.3 Constructor & Destructor Documentation

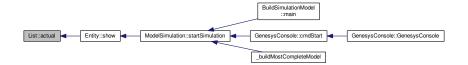
```
5.43.3.1 template < typename T > List < T >::List ( )
```

- 5.43.3.2 template < typename T > List < T >::List (const List < T > & orig)
- 5.43.3.3 template<typename $T > List < T > :: \sim List()$ [virtual]

5.43.4 Member Function Documentation

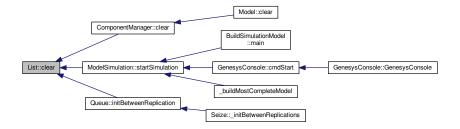
```
5.43.4.1 template<typename T > T List< T >::actual ( )
```

Here is the caller graph for this function:

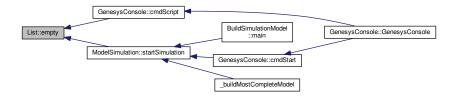


5.43.4.2 template < typename T > void List < T >::clear ()

Here is the caller graph for this function:

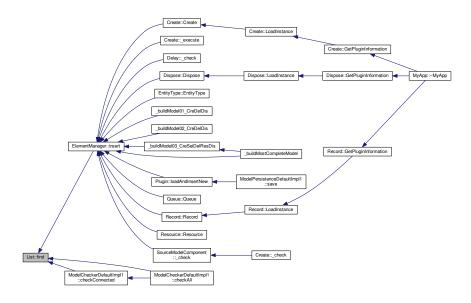


- 5.43.4.3 template<typename T > T List< T >::create ()
- 5.43.4.4 template<typename T > template < typename U > T List < T >::create (U arg)
- 5.43.4.5 template < typename T > bool List < T >::empty ()



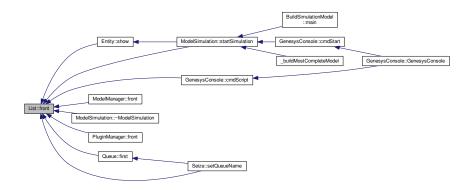
5.43.4.6 template < typename T> std::list < T>::find (T element)

Here is the caller graph for this function:

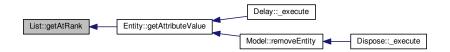


5.43.4.7 template < typename T > T List < T >::front ()

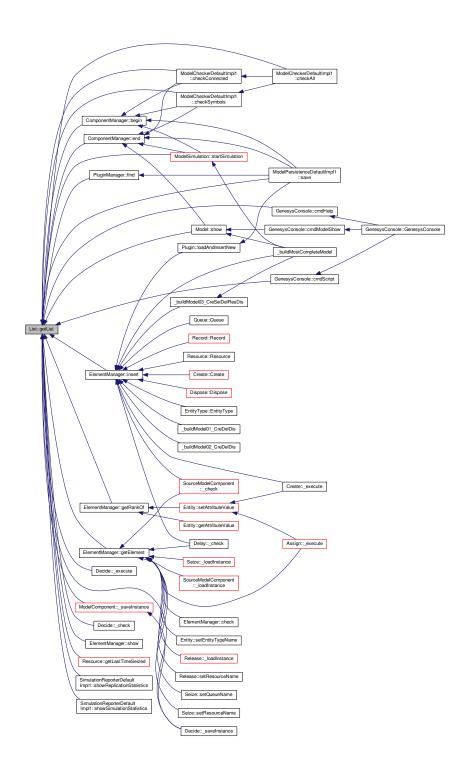
Here is the caller graph for this function:



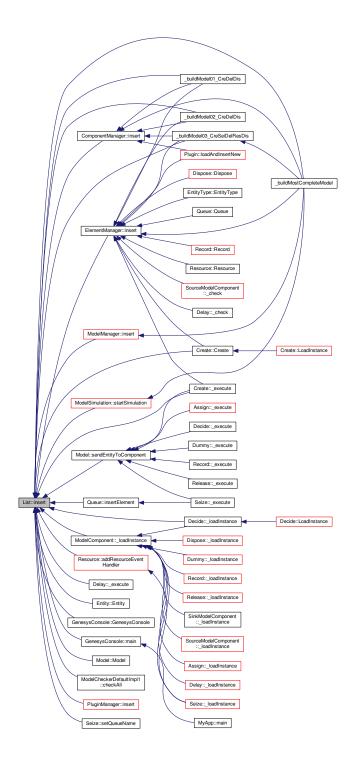
5.43.4.8 template < typename T > T List < T >::getAtRank (unsigned int rank)



5.43.4.9 template < typename T > std::list < T > * List < T >::getList () const

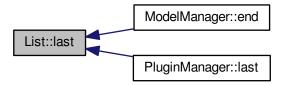


5.43.4.10 template < typename T> void List < T >::insert (T element)



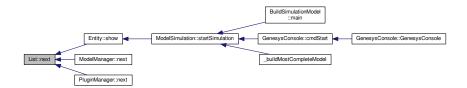
5.43.4.11 template < typename T > T List < T >::last ()

Here is the caller graph for this function:

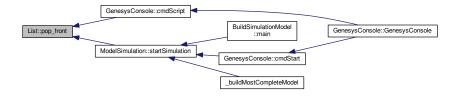


5.43.4.12 template<typename T > T List < T >::next ()

Here is the caller graph for this function:



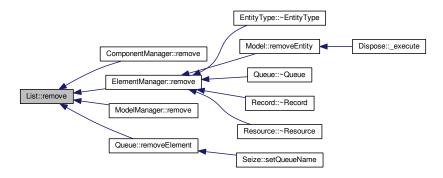
5.43.4.13 template<typename T > void List< T >::pop_front()



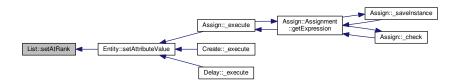
5.43.4.14 template < typename T > T List < T >::previous ()

5.43.4.15 template<typename T> void List< T>::remove (T element)

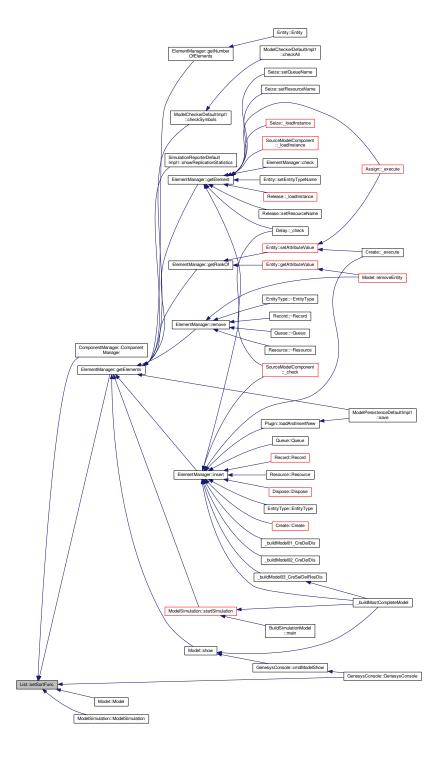
Here is the caller graph for this function:



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



 $5.43.4.17 \quad template < typename \ T > void \ List < T > ::setSortFunc \ (\ CompFunct _ sortFunc \)$



5.44 Model Class Reference 159

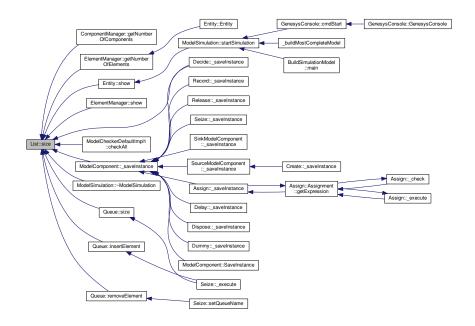
5.43.4.18 template<typename T > std::string List< T >::show ()

Here is the caller graph for this function:



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

Here is the caller graph for this function:



5.43.4.20 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

5.44 Model Class Reference

#include <Model.h>

Public Member Functions

- Model (Simulator *simulator)
- · Model (const Model &orig)
- virtual ∼Model ()
- 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.

- void clear ()
- · void show ()
- 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)
- bool checkExpression (const std::string expression, const std::string expressionName, std::string *error←
 Message)
- · Util::identitifcation 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.

- 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.).

ComponentManager * getComponentManager () 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).

- ModelInfo * getInfos () const
- Simulator * getParent () const
- ModelSimulation * getSimulation () const

Provides access to the class that manages the model simulation.

- List< Event * > * getEvents () const
- void setTraceManager (TraceManager * traceManager)
- TraceManager * getTraceManager () const

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

5.44.1 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 configurated and connected, and a collection of under layered element (ModelElement).

5.44 Model Class Reference 161

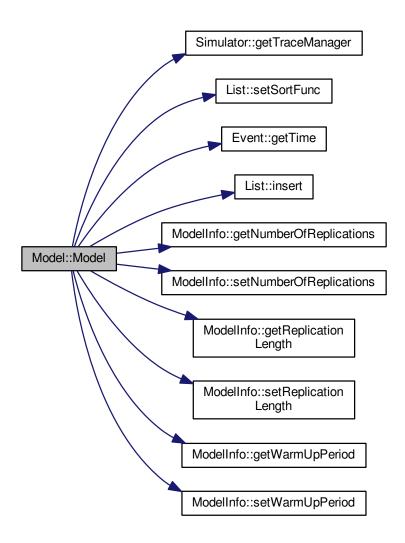
5.44.2 Constructor & Destructor Documentation

5.44.2.1 Model::Model (Simulator * simulator)

The future events list must be chronologicaly sorted

Events are sorted chronologically

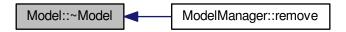
Here is the call graph for this function:



5.44.2.2 Model::Model (const Model & orig)

```
5.44.2.3 Model::∼Model() [virtual]
```

Here is the caller graph for this function:

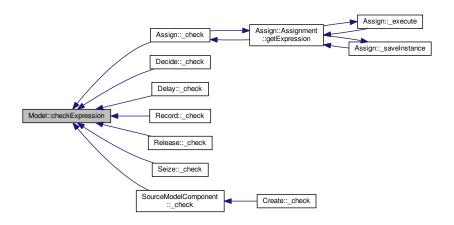


5.44.3 Member Function Documentation

5.44.3.1 bool Model::checkExpression (const std::string *expression*, const std::string *expressionName*, std::string * errorMessage)

Here is the call graph for this function:



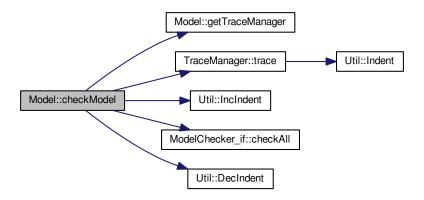


5.44 Model Class Reference 163

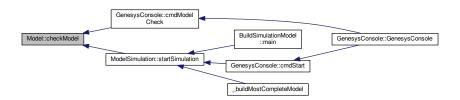
5.44.3.2 bool Model::checkModel()

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

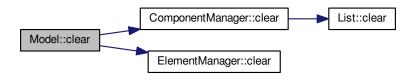
Here is the call graph for this function:



Here is the caller graph for this function:



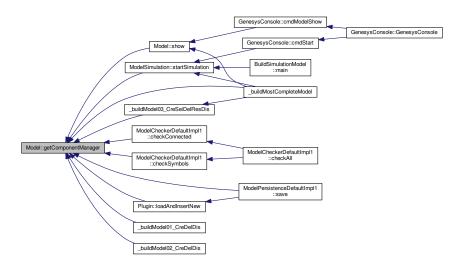
5.44.3.3 void Model::clear ()



5.44.3.4 ComponentManager * Model::getComponentManager () 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).

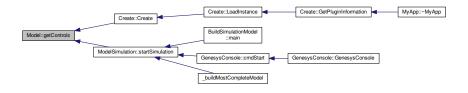
Here is the caller graph for this function:



$\textbf{5.44.3.5} \quad \textbf{List} < \textbf{SimulationControl} * > * \texttt{Model::getControls} \text{ (\ \) const}$

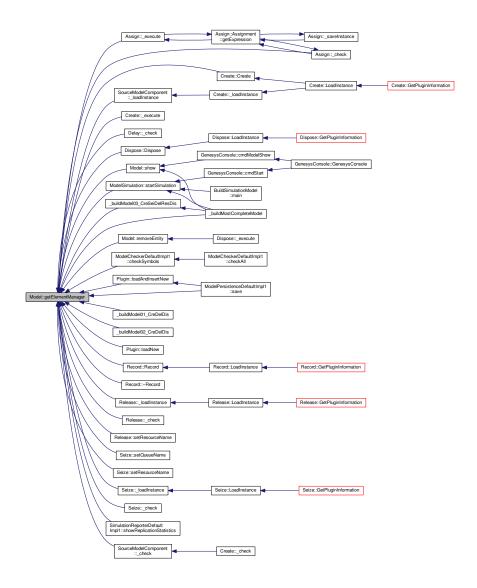
Returns a list of values that can be externally controlled (changed). They usually correspond to input parameters in the simulation model that must be changed for an experimental design.

Here is the caller graph for this function:



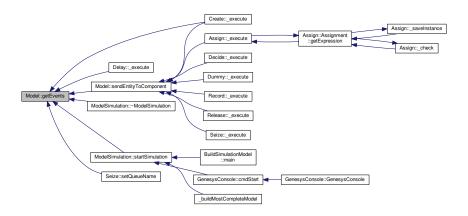
$5.44.3.6 \quad \textbf{ElementManager} * \textbf{Model::getElementManager} (\quad) \ \textbf{const}$

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



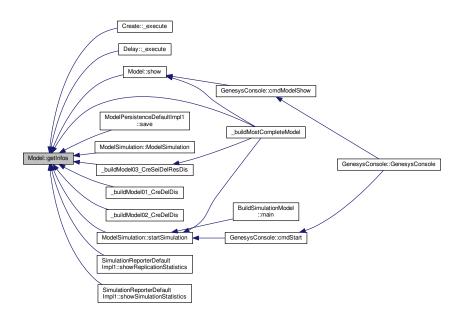
5.44.3.7 List < Event *>* Model::getEvents () const

Here is the caller graph for this function:



5.44.3.8 Util::identitifcation Model::getld () const

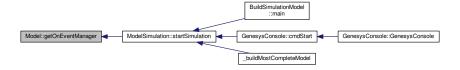
5.44.3.9 ModelInfo * Model::getInfos () const



5.44 Model Class Reference 167

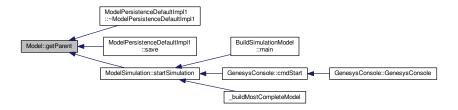
5.44.3.10 OnEventManager * Model::getOnEventManager () const

Here is the caller graph for this function:



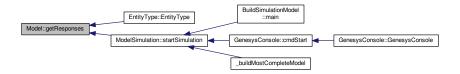
5.44.3.11 Simulator * Model::getParent () const

Here is the caller graph for this function:



$\textbf{5.44.3.12} \quad \textbf{List} < \textbf{SimulationResponse} * > * \, \textbf{Model::getResponses} \, (\quad) \, \textbf{const}$

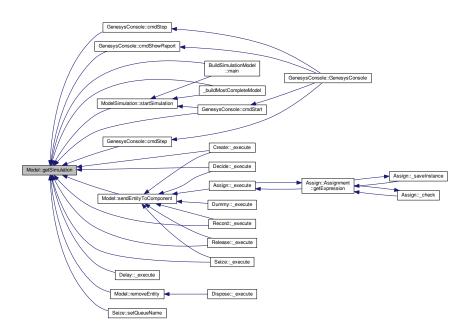
Returns a list of exits or simulation results that can be read externally. They usually correspond to statistics resulting from the simulation that must be read for an experiment design.



5.44.3.13 ModelSimulation * Model::getSimulation () const

Provides access to the class that manages the model simulation.

Here is the caller graph for this function:

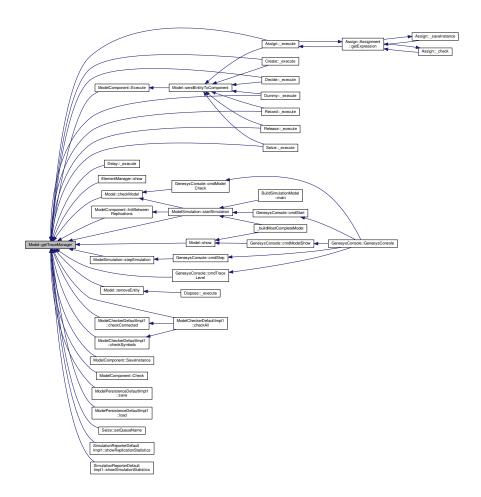


 $5.44.3.14 \quad \textbf{TraceManager} * \textbf{Model::getTraceManager} \left(\right. \right) \textbf{const}$

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

5.44 Model Class Reference 169

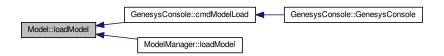
Here is the caller graph for this function:



5.44.3.15 bool Model::loadModel (std::string filename)



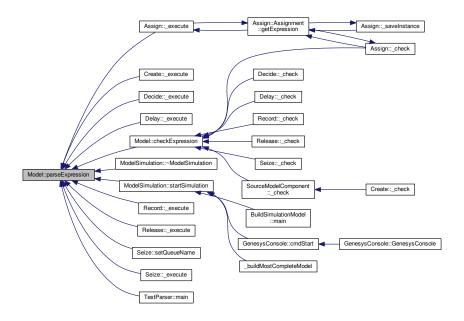
Here is the caller graph for this function:



5.44.3.16 double Model::parseExpression (const std::string expression)

Here is the call graph for this function:





5.44 Model Class Reference 171

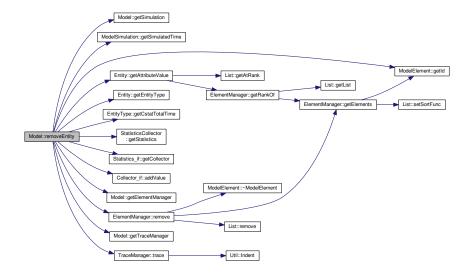
 $5.44.3.17 \quad \text{double Model::parseExpression (const std::string \textit{expression, bool} * \textit{success, std::string} * \textit{errorMessage)}$

Here is the call graph for this function:



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

Here is the call graph for this function:



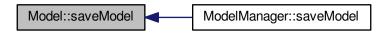


5.44.3.19 bool Model::saveModel (std::string filename)

Here is the call graph for this function:

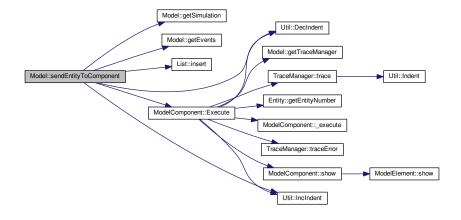


Here is the caller graph for this function:

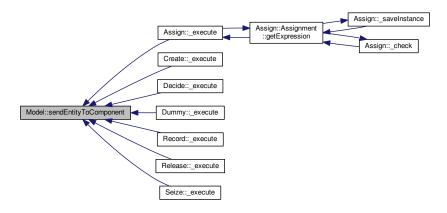


5.44.3.20 void Model::sendEntityToComponent (Entity * entity, ModelComponent * component, double timeDelay)

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).



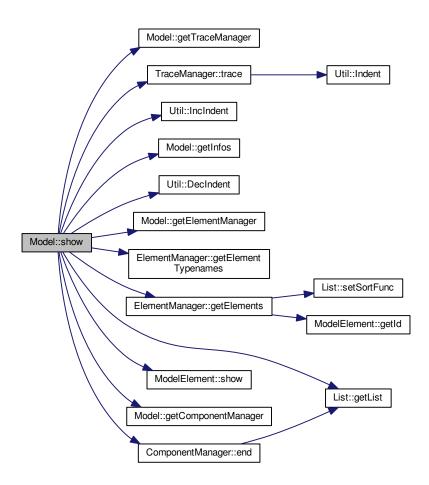
Here is the caller graph for this function:



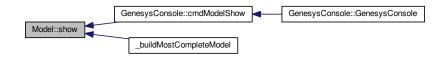
 $5.44.3.21 \quad \text{void Model} :: set Trace Manager (\ \, \text{TraceManager} * \, \underline{\quad} trace Manager)$

5.44.3.22 void Model::show ()

Here is the call graph for this function:



Here is the caller graph for this function:



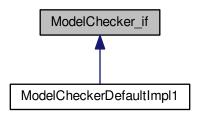
The documentation for this class was generated from the following files:

- Model.h
- Model.cpp

5.45 ModelChecker_if Class Reference

#include <ModelChecker_if.h>

Inheritance diagram for ModelChecker_if:



Public Member Functions

- virtual bool checkAll ()=0
- virtual bool checkConnected ()=0
- virtual bool checkSymbols ()=0
- virtual bool checkActivationCode ()=0

5.45.1 Detailed Description

The ModelChecker is responsable for verifying the model consistency, fixing inconsistencies wheneaver possible

5.45.2 Member Function Documentation

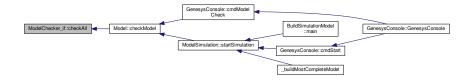
5.45.2.1 virtual bool ModelChecker_if::checkActivationCode() [pure virtual]

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

Implemented in ModelCheckerDefaultImpl1.

5.45.2.2 virtual bool ModelChecker_if::checkAll() [pure virtual]

Implemented in ModelCheckerDefaultImpl1.



5.45.2.3 virtual bool ModelChecker_if::checkConnected() [pure virtual]

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

Implemented in ModelCheckerDefaultImpl1.

5.45.2.4 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 ModelCheckerDefaultImpl1.

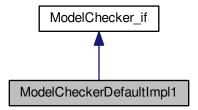
The documentation for this class was generated from the following file:

· ModelChecker_if.h

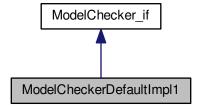
5.46 ModelCheckerDefaultImpl1 Class Reference

#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 checkConnected ()
- virtual bool checkSymbols ()
- · virtual bool checkActivationCode ()

5.46.1 Constructor & Destructor Documentation

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

- 5.46.1.2 ModelCheckerDefaultImpl1::ModelCheckerDefaultImpl1 (const ModelCheckerDefaultImpl1 & orig)
- $\textbf{5.46.1.3} \quad \textbf{ModelCheckerDefaultImpl1::} \sim \textbf{ModelCheckerDefaultImpl1} \ \, (\ \,) \quad [\texttt{virtual}]$
- 5.46.2 Member Function Documentation
- **5.46.2.1** bool ModelCheckerDefaultImpl1::checkActivationCode() [virtual]

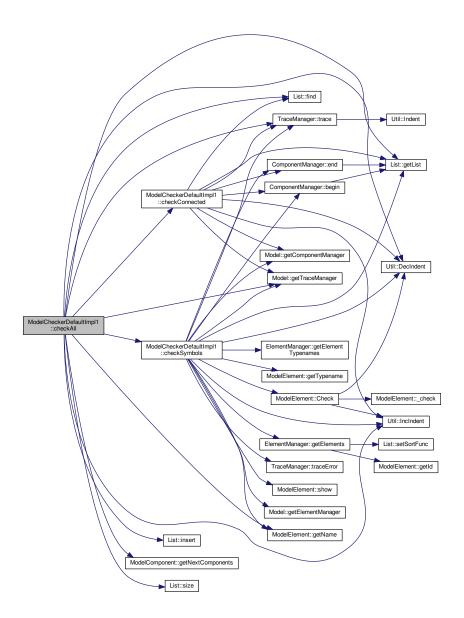
Checks if user-defined strings for symbols required by components, usually expressions or functions, are valid or references existing and valid elements.

Implements ModelChecker_if.

5.46.2.2 bool ModelCheckerDefaultImpl1::checkAll() [virtual]

Implements ModelChecker_if.

Here is the call graph for this function:

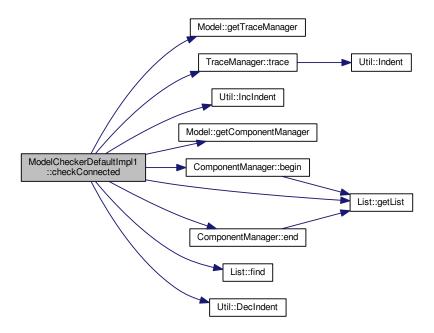


5.46.2.3 bool ModelCheckerDefaultImpl1::checkConnected() [virtual]

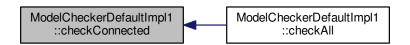
Invoques all other checks and returns true only if all of them returned true

Implements ModelChecker_if.

Here is the call graph for this function:



Here is the caller graph for this function:

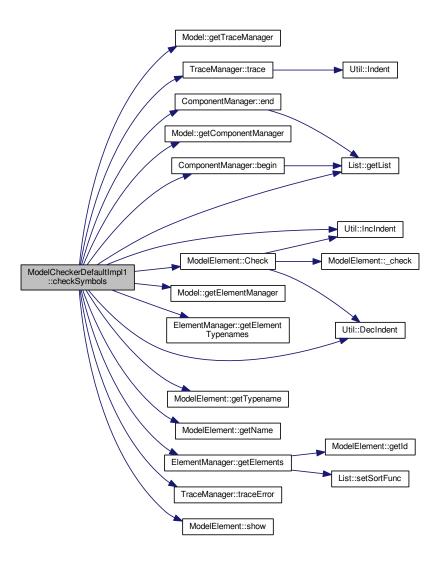


5.46.2.4 bool ModelCheckerDefaultImpl1::checkSymbols() [virtual]

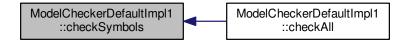
Checks if components are consistently connected to other to form a valid process-oriented model, describing how entities proceed to the flow

Implements ModelChecker_if.

Here is the call graph for this function:



Here is the caller graph for this function:



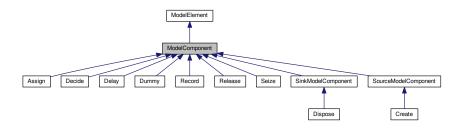
The documentation for this class was generated from the following files:

- ModelCheckerDefaultImpl1.h
- ModelCheckerDefaultImpl1.cpp

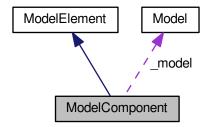
5.47 ModelComponent Class Reference

#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 void InitBetweenReplications (ModelComponent *component)
- static bool Check (ModelComponent *component)
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)
- static std::map< std::string, std::string > * SaveInstance (ModelComponent *component)

Protected Member Functions

- virtual void <u>execute</u> (Entity *entity)=0
- virtual void _initBetweenReplications ()=0
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)

Protected Attributes

· Model * model

5.47.1 Detailed Description

A component of the model is a block that represents a specific behavior to be simulated. The behavior is triggered when an entity arrives at the component, which corresponds to the occurrence of an event. A simulation model corresponds to a set of interconnected components to form the process by which the entity is submitted.

Parameters

-		
ſ	model	The model this component belongs to

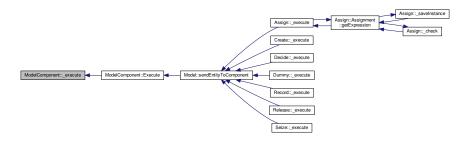
5.47.2 Constructor & Destructor Documentation

- $5.47.2.1 \quad \text{ModelComponent::} \\ \text{Model } * \textit{model}, \textit{ std::} \\ \text{string } \textit{componentTypename })$
- 5.47.2.2 ModelComponent::ModelComponent (const ModelComponent & orig)
- **5.47.2.3** ModelComponent::~ModelComponent() [virtual]

5.47.3 Member Function Documentation

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

Implemented in Assign, Seize, Release, Create, Record, Delay, Decide, Dispose, and Dummy.



5.47.3.2 virtual void ModelComponent::_initBetweenReplications() [protected], [pure virtual]

Implemented in Assign, Seize, Release, SourceModelComponent, Create, Record, Delay, Decide, Dispose, Sink ModelComponent, and Dummy.

Here is the caller graph for this function:

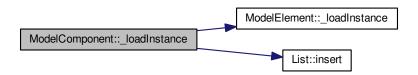


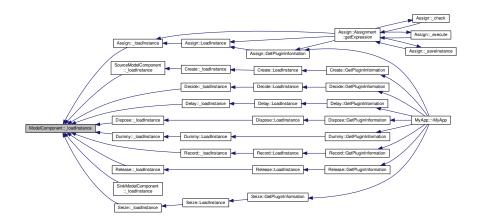
5.47.3.3 bool ModelComponent::_loadInstance (std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelElement.

Reimplemented in Assign, Seize, Release, SourceModelComponent, Create, Delay, Record, Decide, Dispose, SinkModelComponent, and Dummy.

Here is the call graph for this function:



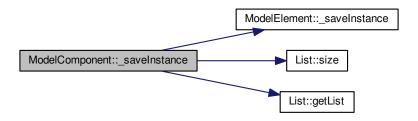


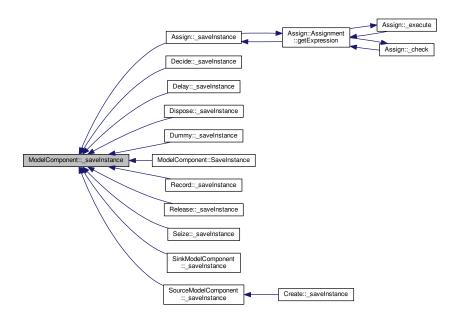
5.47.3.4 std::map< std::string, std::string > * ModelComponent::_saveInstance() [protected], [virtual]

Reimplemented from ModelElement.

Reimplemented in Assign, Seize, Release, SourceModelComponent, Create, Record, Delay, Decide, Dispose, SinkModelComponent, and Dummy.

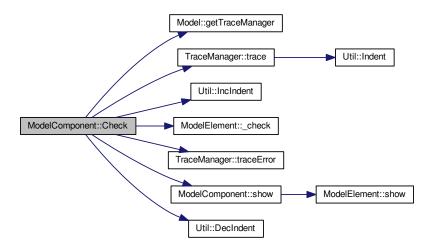
Here is the call graph for this function:





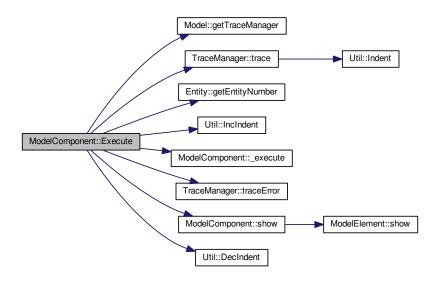
5.47.3.5 bool ModelComponent::Check (ModelComponent * component) [static]

Here is the call graph for this function:

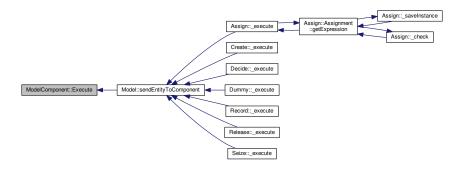


5.47.3.6 void ModelComponent::Execute (Entity * entity, ModelComponent * component) [static]

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.

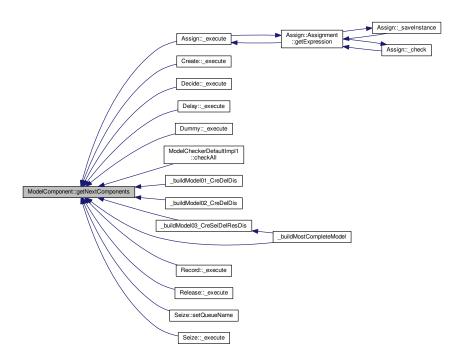


Here is the caller graph for this function:



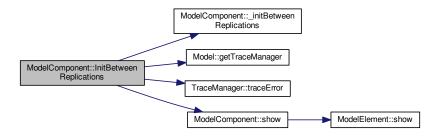
5.47.3.7 List < ModelComponent *>* ModelComponent::getNextComponents () const

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).

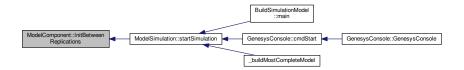


5.47.3.8 void ModelComponent::InitBetweenReplications (ModelComponent * component) [static]

Here is the call graph for this function:

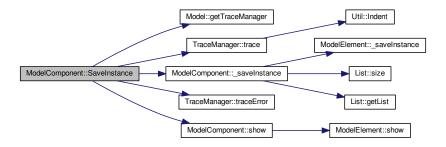


Here is the caller graph for this function:



5.47.3.9 static ModelComponent* ModelComponent::LoadInstance (Model * model, std::map< std::string, std::string > * fields) [static]

5.47.3.10 std::map < std::string, std::string > * ModelComponent::SaveInstance (ModelComponent * component) [static]



5.47.3.11 std::string ModelComponent::show() [virtual]

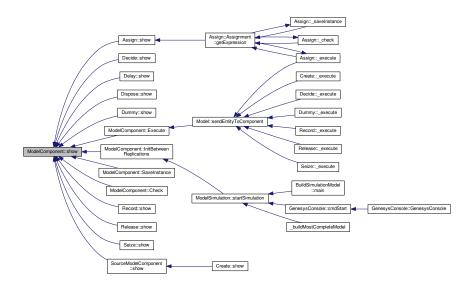
Reimplemented from ModelElement.

Reimplemented in Assign, SourceModelComponent, Create, Record, Seize, Delay, Release, Decide, Dispose, and Dummy.

Here is the call graph for this function:



Here is the caller graph for this function:



5.47.4 Member Data Documentation

5.47.4.1 Model* ModelComponent::_model [protected]

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

- · ModelComponent.h
- ModelComponent.cpp

5.48 ModelComponentManager_if Class Reference

#include <ModelComponentManager_if.h>

Public Member Functions

- ModelComponentManager if ()
- ModelComponentManager_if (const ModelComponentManager_if &orig)
- virtual ~ModelComponentManager_if ()

5.48.1 Constructor & Destructor Documentation

- 5.48.1.1 ModelComponentManager_if::ModelComponentManager_if ()
- 5.48.1.2 ModelComponentManager_if::ModelComponentManager_if (const ModelComponentManager_if & orig)
- 5.48.1.3 virtual ModelComponentManager_if::~ModelComponentManager_if() [virtual]

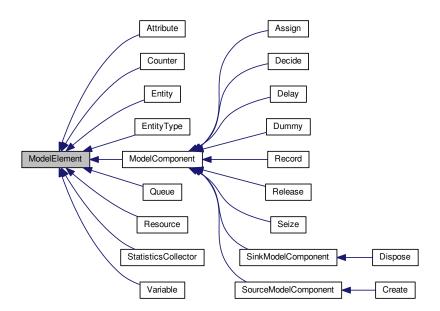
The documentation for this class was generated from the following file:

· ModelComponentManager_if.h

5.49 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::identitifcation getId () const
- void setName (std::string name)
- std::string getName () const
- std::string getTypename () const

Static Public Member Functions

- static ModelElement * LoadInstance (std::map< std::string, std::string > *fields)
- static std::map< std::string, std::string > * SaveInstance (ModelElement *element)
- static bool Check (ModelElement *element, std::string *errorMessage)

Protected Member Functions

- virtual bool loadInstance (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Protected Attributes

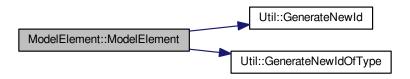
- · Util::identitifcation id
- · std::string _name
- std::string _typename

5.49.1 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.

5.49.2 Constructor & Destructor Documentation

5.49.2.1 ModelElement::ModelElement (std::string elementTypename)



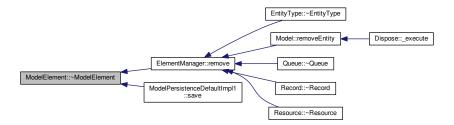
Here is the caller graph for this function:



5.49.2.2 ModelElement::ModelElement (const ModelElement & orig)

5.49.2.3 ModelElement::~ModelElement() [virtual]

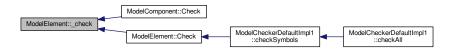
Here is the caller graph for this function:



5.49.3 Member Function Documentation

5.49.3.1 bool ModelElement::_check (std::string * errorMessage) [protected], [virtual]

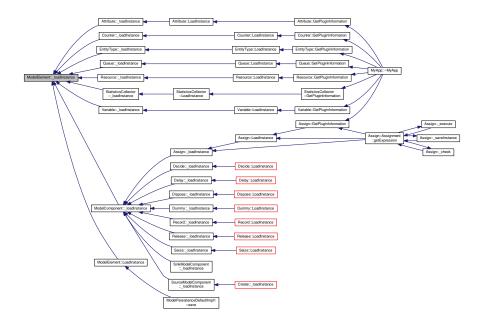
Reimplemented in Assign, Resource, Seize, Queue, EntityType, Release, SourceModelComponent, Entity, Counter, Create, Record, Delay, StatisticsCollector, Variable, Attribute, Decide, Dispose, SinkModelComponent, and Dummy.



5.49.3.2 bool ModelElement::_loadInstance (std::map < std::string, std::string > * fields) [protected], [virtual]

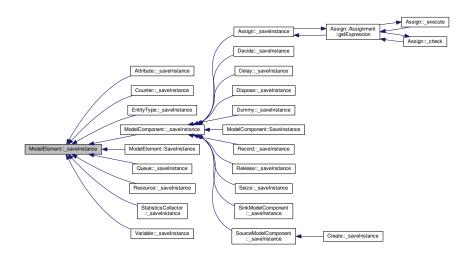
Reimplemented in Assign, Resource, Seize, Queue, EntityType, ModelComponent, Release, SourceModel ← Component, Entity, Counter, Create, Delay, Record, StatisticsCollector, Variable, Attribute, Decide, Dispose, Sink ← ModelComponent, and Dummy.

Here is the caller graph for this function:



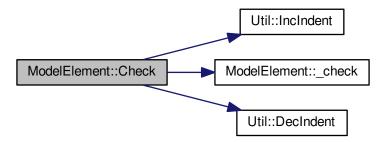
 $\textbf{5.49.3.3} \quad \textbf{std::map} < \textbf{std::string}, \textbf{std::string} > * \textbf{ModelElement::_saveInstance()}, \texttt{[virtual]}$

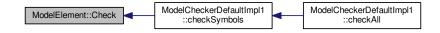
Reimplemented in Assign, Resource, Seize, Queue, EntityType, Release, ModelComponent, SourceModel ← Component, Entity, Create, Record, Counter, Delay, StatisticsCollector, Variable, Attribute, Decide, Dispose, Sink ← ModelComponent, and Dummy.



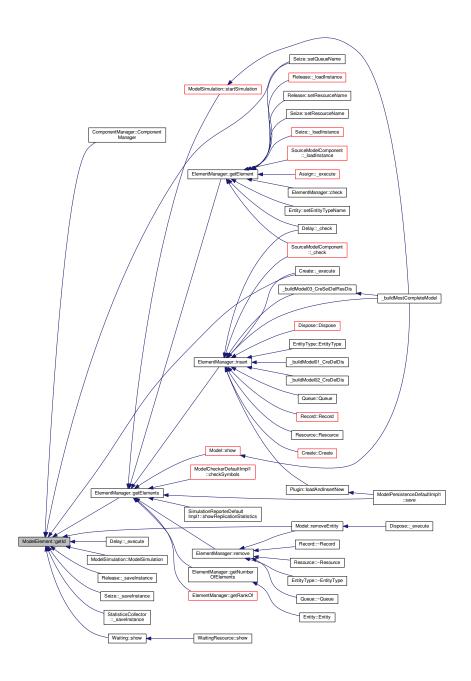
5.49.3.4 bool ModelElement::Check (ModelElement * element, std::string * errorMessage) [static]

Here is the call graph for this function:



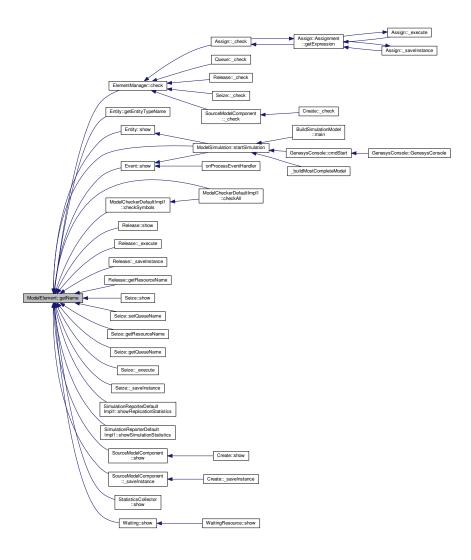


5.49.3.5 Util::identitifcation ModelElement::getId () const

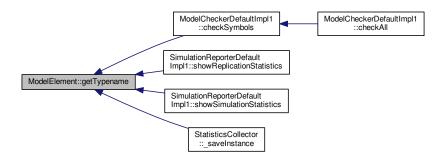


5.49.3.6 std::string ModelElement::getName () const

Here is the caller graph for this function:

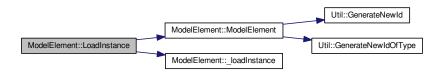


5.49.3.7 std::string ModelElement::getTypename () const

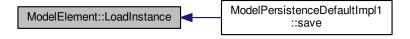


5.49.3.8 ModelElement * ModelElement::LoadInstance (std::map < std::string, std::string > * fields) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



5.49.3.9 std::map < std::string, std::string > * ModelElement::SaveInstance (ModelElement * element) [static]

Here is the call graph for this function:



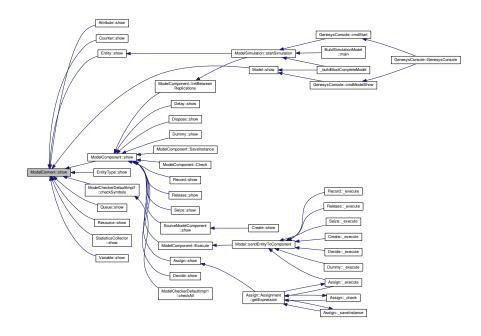
5.49.3.10 void ModelElement::setName (std::string _name)



5.49.3.11 std::string ModelElement::show() [virtual]

Reimplemented in Assign, Resource, SourceModelComponent, Queue, ModelComponent, Create, Record, Seize, Delay, EntityType, Attribute, Entity, StatisticsCollector, Counter, Release, Variable, Decide, Dispose, and Dummy.

Here is the caller graph for this function:



5.49.4 Member Data Documentation

5.49.4.1 Util::identitifcation ModelElement::_id [protected]

5.49.4.2 std::string ModelElement::_name [protected]

5.49.4.3 std::string ModelElement::_typename [protected]

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

- · ModelElement.h
- ModelElement.cpp

5.50 ModelInfo Class Reference

#include <ModelInfo.h>

Public Member Functions

- ModelInfo ()
- ModelInfo (const ModelInfo &orig)
- virtual ∼ModelInfo ()
- std::string show ()
- 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
- void loadInstance (std::map< std::string, std::string > *fields)
- std::map< std::string, std::string > * saveInstance ()

5.50.1 Detailed Description

ModelInfo stores basic model project information.

5.50.2 Constructor & Destructor Documentation

```
5.50.2.1 ModelInfo::ModelInfo ( )
```

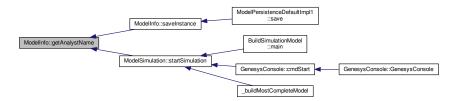
5.50.2.2 ModelInfo::ModelInfo (const ModelInfo & orig)

5.50.2.3 ModelInfo::∼ModelInfo() [virtual]

5.50.3 Member Function Documentation

5.50.3.1 std::string ModelInfo::getAnalystName () const

Here is the caller graph for this function:

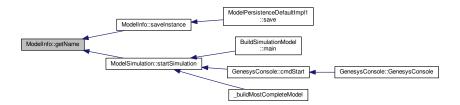


5.50.3.2 std::string ModelInfo::getDescription () const

Here is the caller graph for this function:

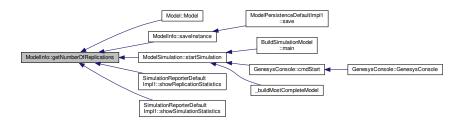


5.50.3.3 std::string ModelInfo::getName () const



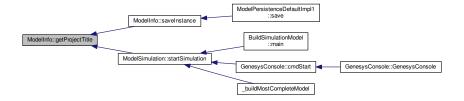
5.50.3.4 unsigned int ModelInfo::getNumberOfReplications () const

Here is the caller graph for this function:

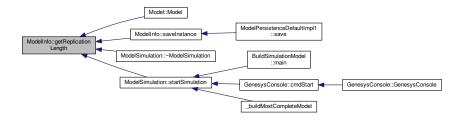


5.50.3.5 std::string ModelInfo::getProjectTitle () const

Here is the caller graph for this function:

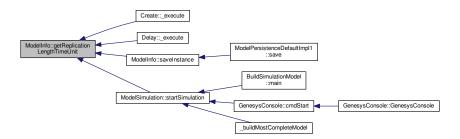


5.50.3.6 double ModelInfo::getReplicationLength () const



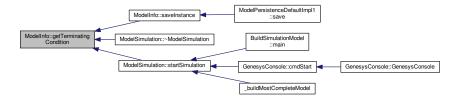
5.50.3.7 Util::TimeUnit ModelInfo::getReplicationLengthTimeUnit () const

Here is the caller graph for this function:



5.50.3.8 std::string ModelInfo::getTerminatingCondition () const

Here is the caller graph for this function:

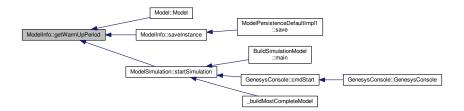


5.50.3.9 std::string ModelInfo::getVersion () const



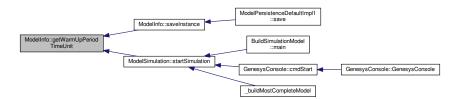
5.50.3.10 double ModelInfo::getWarmUpPeriod () const

Here is the caller graph for this function:



5.50.3.11 Util::TimeUnit ModelInfo::getWarmUpPeriodTimeUnit () const

Here is the caller graph for this function:

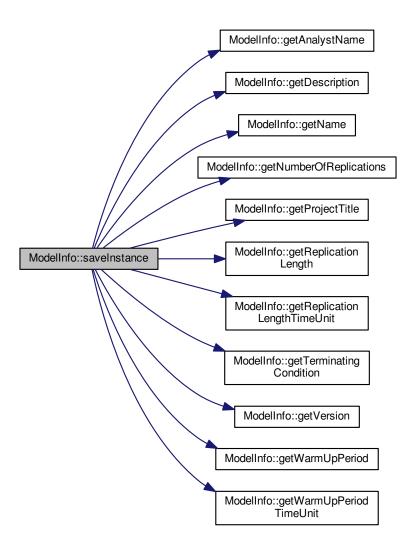


5.50.3.12 void ModelInfo::loadInstance (std::map < std::string, std::string > * fields)



5.50.3.13 std::map < std::string, std::string > * ModelInfo::saveInstance ()

Here is the call graph for this function:





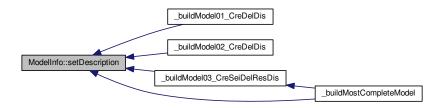
5.50.3.14 void ModelInfo::setAnalystName (std::string _analystName)

Here is the caller graph for this function:



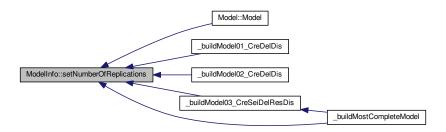
5.50.3.15 void ModelInfo::setDescription (std::string _description)

Here is the caller graph for this function:



5.50.3.16 void ModelInfo::setName (std::string _name)

5.50.3.17 void ModelInfo::setNumberOfReplications (unsigned int _numberOfReplications)



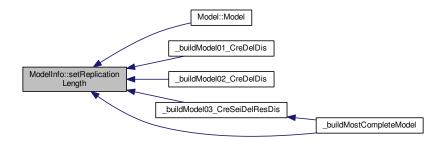
5.50.3.18 void ModelInfo::setProjectTitle (std::string _projectTitle)

Here is the caller graph for this function:

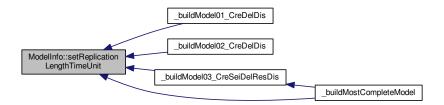


5.50.3.19 void ModelInfo::setReplicationLength (double _replicationLength)

Here is the caller graph for this function:



5.50.3.20 void ModelInfo::setReplicationLengthTimeUnit (Util::TimeUnit _replicationLengthTimeUnit)

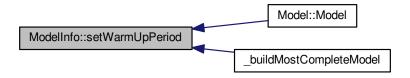


5.50.3.21 void ModelInfo::setTerminatingCondition (std::string _terminatingCondition)

5.50.3.22 void ModelInfo::setVersion (std::string _version)

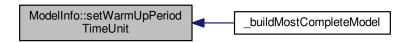
5.50.3.23 void ModelInfo::setWarmUpPeriod (double _warmUpPeriod)

Here is the caller graph for this function:



5.50.3.24 void ModelInfo::setWarmUpPeriodTimeUnit (Util::TimeUnit _warmUpPeriodTimeUnit)

Here is the caller graph for this function:



5.50.3.25 std::string ModelInfo::show ()

Here is the call graph for this function:



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

- ModelInfo.h
- ModelInfo.cpp

5.51 ModelManager Class Reference

#include <ModelManager.h>

Public Member Functions

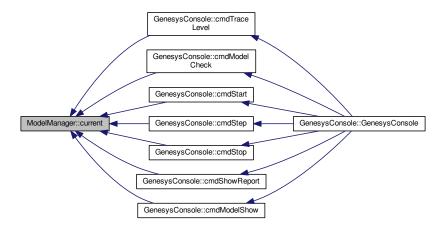
- ModelManager (Simulator *simulator)
- ModelManager (const ModelManager &orig)
- virtual ∼ModelManager ()
- void insert (Model *model)
- void remove (Model *model)
- void setCurrent (Model *model)
- bool saveModel (std::string filename)
- bool loadModel (std::string filename)
- Model * front ()
- Model * current ()
- Model * next ()
- Model * end ()

5.51.1 Constructor & Destructor Documentation

- 5.51.1.1 ModelManager::ModelManager (Simulator * simulator)
- 5.51.1.2 ModelManager::ModelManager (const ModelManager & orig)
- 5.51.1.3 ModelManager:: \sim ModelManager() [virtual]

5.51.2 Member Function Documentation

5.51.2.1 Model * ModelManager::current ()



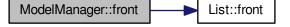
5.51.2.2 Model * ModelManager::end ()

Here is the call graph for this function:



5.51.2.3 Model * ModelManager::front ()

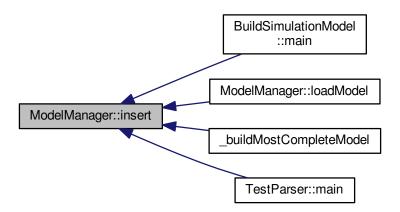
Here is the call graph for this function:



5.51.2.4 void ModelManager::insert (Model * model)

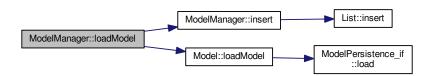


Here is the caller graph for this function:



5.51.2.5 bool ModelManager::loadModel (std::string filename)

Here is the call graph for this function:

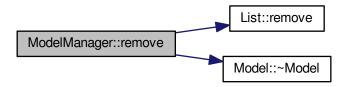


5.51.2.6 Model * ModelManager::next ()



5.51.2.7 void ModelManager::remove (Model * model)

Here is the call graph for this function:



5.51.2.8 bool ModelManager::saveModel (std::string filename)

Here is the call graph for this function:



5.51.2.9 void ModelManager::setCurrent (Model * model)

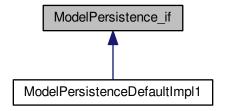
The documentation for this class was generated from the following files:

- · ModelManager.h
- ModelManager.cpp

5.52 ModelPersistence_if Class Reference

#include <ModelPersistence_if.h>

Inheritance diagram for ModelPersistence_if:



Public Member Functions

- virtual bool save (std::string filename)=0
- virtual bool load (std::string filename)=0
- virtual bool isSaved ()=0

5.52.1 Detailed Description

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

5.52.2 Member Function Documentation

5.52.2.1 virtual bool ModelPersistence_if::isSaved() [pure virtual]

Implemented in ModelPersistenceDefaultImpl1.

5.52.2.2 virtual bool ModelPersistence_if::load (std::string *filename* **)** [pure virtual]

Implemented in ModelPersistenceDefaultImpl1.



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

Implemented in ModelPersistenceDefaultImpl1.

Here is the caller graph for this function:



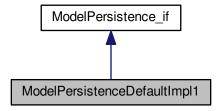
The documentation for this class was generated from the following file:

• ModelPersistence_if.h

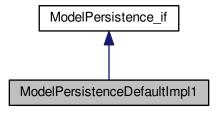
5.53 ModelPersistenceDefaultImpl1 Class Reference

#include <ModelPersistenceDefaultImpl1.h>

Inheritance diagram for ModelPersistenceDefaultImpl1:



Collaboration diagram for ModelPersistenceDefaultImpl1:



Public Member Functions

- ModelPersistenceDefaultImpl1 (Model *model)
- ModelPersistenceDefaultImpl1 (const ModelPersistenceDefaultImpl1 &orig)
- virtual ~ModelPersistenceDefaultImpl1 ()
- virtual bool save (std::string filename)
- · virtual bool load (std::string filename)
- virtual bool isSaved ()

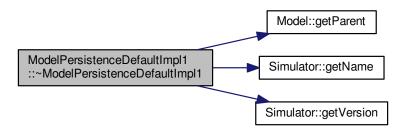
5.53.1 Constructor & Destructor Documentation

5.53.1.1 ModelPersistenceDefaultImpl1::ModelPersistenceDefaultImpl1 (Model * model)

5.53.1.2 ModelPersistenceDefaultImpl1::ModelPersistenceDefaultImpl1 & orig)

5.53.1.3 ModelPersistenceDefaultImpl1::~ModelPersistenceDefaultImpl1() [virtual]

Here is the call graph for this function:



5.53.2 Member Function Documentation

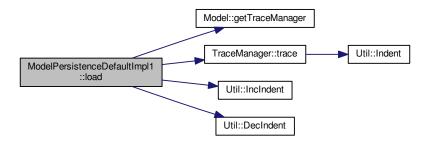
5.53.2.1 bool ModelPersistenceDefaultImpl1::isSaved() [virtual]

Implements ModelPersistence_if.

5.53.2.2 bool ModelPersistenceDefaultImpl1::load (std::string filename) [virtual]

Implements ModelPersistence_if.

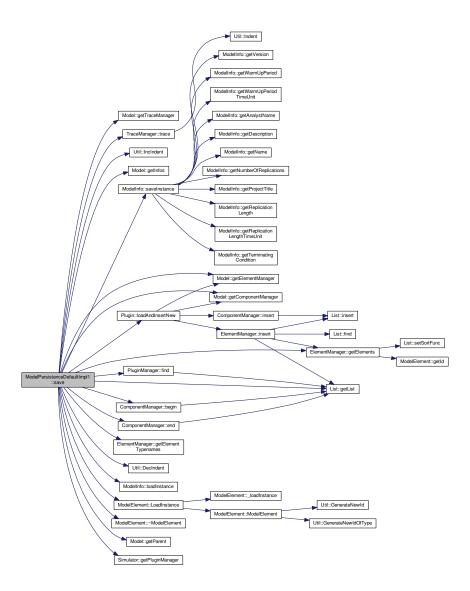
Here is the call graph for this function:



5.53.2.3 bool ModelPersistenceDefaultImpl1::save (std::string filename) [virtual]

Implements ModelPersistence_if.

Here is the call graph for this function:



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

- ModelPersistenceDefaultImpl1.h
- ModelPersistenceDefaultImpl1.cpp

5.54 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.

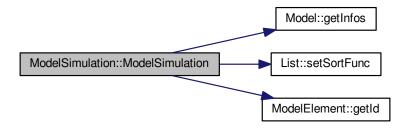
- void stopSimulation ()
- void restartSimulation ()
- void setPauseOnEvent (bool _pauseOnEvent)
- bool isPauseOnEvent () const
- void setStepByStep (bool _stepByStep)
- bool isStepByStep () const
- void setInitializeStatistics (bool _initializeStatistics)
- bool isInitializeStatistics () const
- void 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
- SimulationReporter_if * getSimulationReporter () const

5.54.1 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.

5.54.2 Constructor & Destructor Documentation

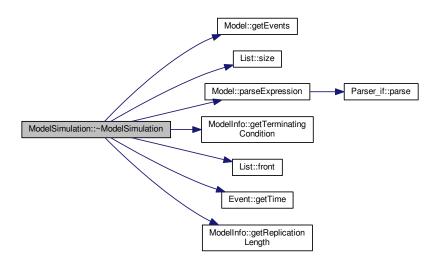
5.54.2.1 ModelSimulation::ModelSimulation (Model*model)



5.54.2.2 ModelSimulation::ModelSimulation (const ModelSimulation & orig)

5.54.2.3 ModelSimulation::~ModelSimulation() [virtual]

Here is the call graph for this function:

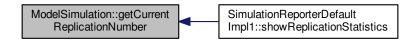


5.54.3 Member Function Documentation

5.54.3.1 ModelComponent * ModelSimulation::getCurrentComponent () const

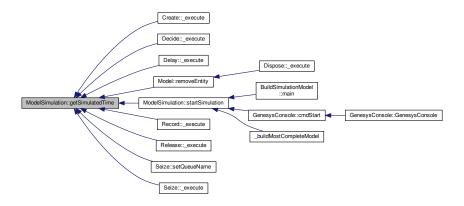
5.54.3.2 Entity * ModelSimulation::getCurrentEntity () const

5.54.3.3 unsigned int ModelSimulation::getCurrentReplicationNumber () const



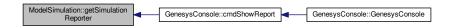
5.54.3.4 double ModelSimulation::getSimulatedTime () const

Here is the caller graph for this function:



5.54.3.5 SimulationReporter_if * ModelSimulation::getSimulationReporter () const

Here is the caller graph for this function:



- 5.54.3.6 bool ModelSimulation::isInitializeStatistics () const
- 5.54.3.7 bool ModelSimulation::isInitializeSystem () const
- 5.54.3.8 bool ModelSimulation::isPauseOnEvent () const
- 5.54.3.9 bool ModelSimulation::isPauseOnReplication () const
- 5.54.3.10 bool ModelSimulation::isRunning () const

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

```
5.54.3.11 bool ModelSimulation::isStepByStep ( ) const

5.54.3.12 void ModelSimulation::pauseSimulation ( )

5.54.3.13 void ModelSimulation::restartSimulation ( )

5.54.3.14 void ModelSimulation::setInitializeStatistics ( bool _initializeStatistics )

5.54.3.15 void ModelSimulation::setInitializeSystem ( bool _initializeSystem )

5.54.3.16 void ModelSimulation::setPauseOnEvent ( bool _pauseOnEvent )

5.54.3.17 void ModelSimulation::setPauseOnReplication ( bool _pauseBetweenReplications )

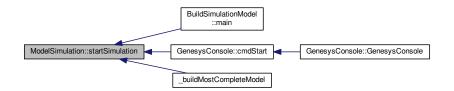
5.54.3.18 void ModelSimulation::setStepByStep ( bool _stepByStep )

5.54.3.19 void ModelSimulation::startSimulation ( )
```

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

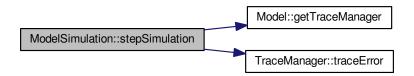
Checks the model and if ok then initialize the simulation, execute repeatedly each replication and then show simulation statistics

Here is the caller graph for this function:



5.54.3.20 void ModelSimulation::stepSimulation()

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



Here is the caller graph for this function:



5.54.3.21 void ModelSimulation::stopSimulation()

Here is the caller graph for this function:



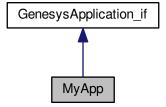
The documentation for this class was generated from the following files:

- ModelSimulation.h
- ModelSimulation.cpp

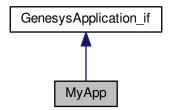
5.55 MyApp Class Reference

#include <MyApp.h>

Inheritance diagram for MyApp:



Collaboration diagram for MyApp:



Public Member Functions

- MyApp ()
- MyApp (const MyApp &orig)
- virtual ∼MyApp ()
- virtual int main (int argc, char **argv)

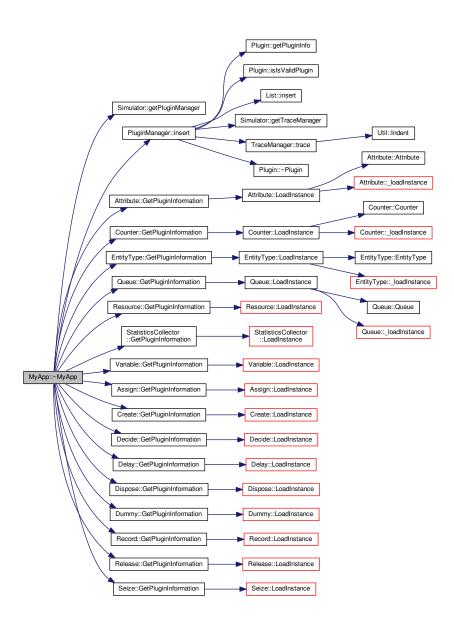
5.55.1 Constructor & Destructor Documentation

5.55.1.1 MyApp::MyApp ()

5.55.1.2 MyApp::MyApp (const MyApp & orig)

5.55.1.3 MyApp::~MyApp() [virtual]

Here is the call graph for this function:



5.55.2 Member Function Documentation

5.55.2.1 int MyApp::main (int argc, char ** argv) [virtual]

Implements GenesysApplication_if.

Here is the call graph for this function:



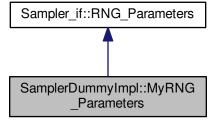
The documentation for this class was generated from the following files:

- MyApp.h
- MyApp.cpp

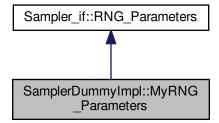
5.56 SamplerDummyImpl::MyRNG_Parameters Struct Reference

#include <SamplerDummyImpl.h>

Inheritance diagram for SamplerDummyImpl::MyRNG_Parameters:



Collaboration diagram for SamplerDummyImpl::MyRNG_Parameters:



Public Member Functions

~MyRNG_Parameters ()=default

Public Attributes

- · unsigned int seed
- · unsigned int module
- · unsigned int multiplier

5.56.1 Constructor & Destructor Documentation

```
5.56.1.1 SamplerDummyImpl::MyRNG_Parameters::~MyRNG_Parameters() [default]
```

5.56.2 Member Data Documentation

- 5.56.2.1 unsigned int SamplerDummyImpl::MyRNG_Parameters::module
- 5.56.2.2 unsigned int SamplerDummyImpl::MyRNG_Parameters::multiplier
- 5.56.2.3 unsigned int SamplerDummyImpl::MyRNG_Parameters::seed

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

· SamplerDummyImpl.h

5.57 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)

5.57.1 Detailed Description

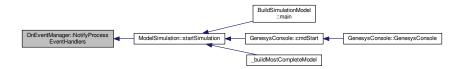
OnEventManager allows external methods to hook interval simulation events as listeners (or observers) of pecific events. All methods added as listeners of an event will be invovked when that event is triggered.

5.57.2 Constructor & Destructor Documentation

- 5.57.2.1 OnEventManager::OnEventManager()
- 5.57.2.2 OnEventManager::OnEventManager (const OnEventManager & orig)
- 5.57.2.3 OnEventManager::~OnEventManager() [virtual]

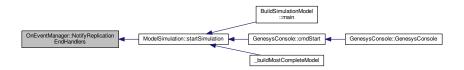
5.57.3 Member Function Documentation

- 5.57.3.1 void OnEventManager::addOnEntityRemoveHandler (simulationEventHandler EventHandler)
- 5.57.3.2 void OnEventManager::addOnProcessEventHandler (simulationEventHandler EventHandler)
- 5.57.3.3 void OnEventManager::addOnReplicationEndHandler (simulationEventHandler EventHandler)
- 5.57.3.4 void OnEventManager::addOnReplicationStartHandler (simulationEventHandler EventHandler)
- 5.57.3.5 void OnEventManager::addOnReplicationStepHandler (simulationEventHandler EventHandler)
- 5.57.3.6 void OnEventManager::addOnSimulationEndHandler (simulationEventHandler EventHandler)
- 5.57.3.7 void OnEventManager::addOnSimulationStartHandler (simulationEventHandler EventHandler)
- 5.57.3.8 void OnEventManager::NotifyProcessEventHandlers (SimulationEvent * se)



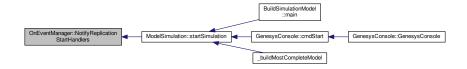
5.57.3.9 void OnEventManager::NotifyReplicationEndHandlers (SimulationEvent * se)

Here is the caller graph for this function:



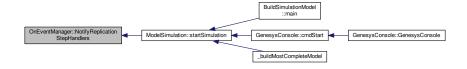
5.57.3.10 void OnEventManager::NotifyReplicationStartHandlers (SimulationEvent * se)

Here is the caller graph for this function:

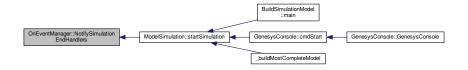


$5.57.3.11 \quad \text{void OnEventManager::NotifyReplicationStepHandlers (\ SimulationEvent} * \textit{se} \)$

Here is the caller graph for this function:

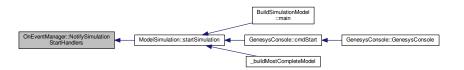


5.57.3.12 void OnEventManager::NotifySimulationEndHandlers (SimulationEvent * se)



5.57.3.13 void OnEventManager::NotifySimulationStartHandlers (SimulationEvent * se)

Here is the caller graph for this function:



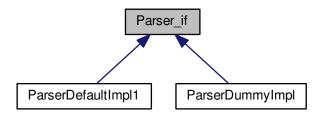
The documentation for this class was generated from the following files:

- OnEventManager.h
- OnEventManager.cpp

5.58 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

5.58.1 Member Function Documentation

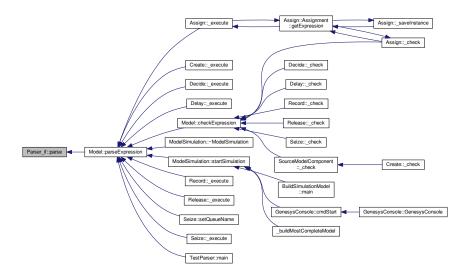
5.58.1.1 virtual std::string* Parser_if::getErrorMessage() [pure virtual]

Implemented in ParserDummyImpl, and ParserDefaultImpl1.

5.58.1.2 virtual double Parser_if::parse (const std::string expression) [pure virtual]

Implemented in ParserDummyImpl, and ParserDefaultImpl1.

Here is the caller graph for this function:



5.58.1.3 virtual double Parser_if::parse (const std::string *expression*, bool * *success*, std::string * *errorMessage*) [pure virtual]

Implemented in ParserDummyImpl, and ParserDefaultImpl1.

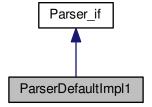
The documentation for this class was generated from the following file:

· Parser_if.h

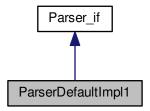
5.59 ParserDefaultImpl1 Class Reference

#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 ()

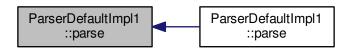
5.59.1 Constructor & Destructor Documentation

- 5.59.1.1 ParserDefaultImpl1::ParserDefaultImpl1 (Model*model)
- 5.59.1.2 ParserDefaultImpl1::ParserDefaultImpl1 (const ParserDefaultImpl1 & orig)
- 5.59.1.3 ParserDefaultImpl1:: \sim ParserDefaultImpl1() [virtual]
- 5.59.2 Member Function Documentation
- 5.59.2.1 std::string * ParserDefaultImpl1::getErrorMessage() [virtual]

Implements Parser_if.

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

Implements Parser_if.



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

Implements Parser_if.

Here is the call graph for this function:



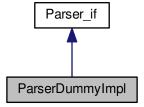
The documentation for this class was generated from the following files:

- ParserDefaultImpl1.h
- ParserDefaultImpl1.cpp

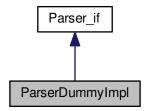
5.60 ParserDummyImpl Class Reference

#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 ()

5.60.1 Constructor & Destructor Documentation

- $5.60.1.1 \quad \textbf{ParserDummyImpl::ParserDummyImpl (} \quad \textbf{Model} * \textit{model} \text{)}$
- $5.60.1.2 \quad {\sf ParserDummyImpl::ParserDummyImpl~(~const~ParserDummyImpl~\&~orig~)}$
- **5.60.1.3** ParserDummylmpl::~ParserDummylmpl() [virtual]

5.60.2 Member Function Documentation

5.60.2.1 std::string * ParserDummyImpl::getErrorMessage() [virtual]

Implements Parser_if.

5.60.2.2 double ParserDummyImpl::parse (const std::string expression) [virtual]

Implements Parser_if.



5.60.2.3 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

5.61 Plugin Class Reference

```
#include <Plugin.h>
```

Public Member Functions

- Plugin (StaticGetPluginInformation getInformation)
- Plugin (const Plugin &orig)
- virtual ∼Plugin ()
- · bool islsValidPlugin () const
- PluginInformation * getPluginInfo () const
- ModelElement * loadNew (Model *model, std::map< std::string, std::string > *fields)
- bool loadAndInsertNew (Model *model, std::map< std::string, std::string > *fields)

5.61.1 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

5.61.2 Constructor & Destructor Documentation

5.61.2.1 Plugin::Plugin (StaticGetPluginInformation getInformation)

5.61.2.2 Plugin::Plugin (const Plugin & orig)

5.61.2.3 Plugin::∼Plugin() [virtual]

Here is the caller graph for this function:



5.61.3 Member Function Documentation

5.61.3.1 PluginInformation * Plugin::getPluginInfo () const

Here is the caller graph for this function:

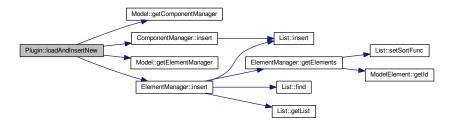


5.61.3.2 bool Plugin::islsValidPlugin () const



5.61.3.3 bool Plugin::loadAndInsertNew (Model * model, std::map < std::string, std::string > * fields)

Here is the call graph for this function:



Here is the caller graph for this function:



5.61.3.4 ModelElement * Plugin::loadNew (Model * model, std::map < std::string, std::string > * fields)

Here is the call graph for this function:



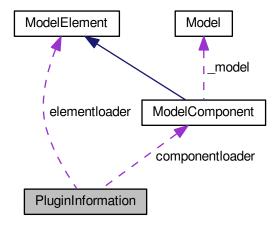
The documentation for this class was generated from the following files:

- Plugin.h
- Plugin.cpp

5.62 PluginInformation Class Reference

#include <Plugin.h>

Collaboration diagram for PluginInformation:



Public Member Functions

- · PluginInformation (std::string pluginTypename, StaticLoaderComponentInstance componentloader)
- PluginInformation (std::string pluginTypename, StaticLoaderElementInstance elementloader)

Public Attributes

- std::string pluginTypename
- std::string author = ""
- std::string date = ""
- std::string observation = ""
- · bool isSource
- bool isSink
- bool isComponent
- bool generateReport = false
- StaticLoaderComponentInstance componentloader
- · StaticLoaderElementInstance elementloader
- std::list< PluginInformation > dependencies

5.62.1 Constructor & Destructor Documentation

- 5.62.1.1 PluginInformation::PluginInformation (std::string pluginTypename, StaticLoaderComponentInstance componentloader) [inline]
- 5.62.1.2 PluginInformation::PluginInformation (std::string *pluginTypename*, StaticLoaderElementInstance *elementloader*) [inline]
- 5.62.2 Member Data Documentation
- 5.62.2.1 std::string PluginInformation::author = ""
- 5.62.2.2 StaticLoaderComponentInstance PluginInformation::componentloader
- 5.62.2.3 std::string PluginInformation::date = ""
- 5.62.2.4 std::list<PluginInformation> PluginInformation::dependencies
- 5.62.2.5 StaticLoaderElementInstance PluginInformation::elementloader
- 5.62.2.6 bool PluginInformation::generateReport = false
- 5.62.2.7 bool PluginInformation::isComponent
- 5.62.2.8 bool PluginInformation::isSink
- 5.62.2.9 bool PluginInformation::isSource
- 5.62.2.10 std::string PluginInformation::observation = ""
- 5.62.2.11 std::string PluginInformation::pluginTypename

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

• Plugin.h

5.63 PluginManager Class Reference

#include <PluginManager.h>

Public Member Functions

- PluginManager (Simulator *simulator)
- PluginManager (const PluginManager &orig)
- virtual ∼PluginManager ()
- bool insert (Plugin *plugin)
- void remove (Plugin *plugin)
- Plugin * find (std::string pluginTypeName)
- Plugin * front ()
- Plugin * next ()
- Plugin * last ()

5.63.1 Constructor & Destructor Documentation

```
5.63.1.1 PluginManager::PluginManager ( Simulator * simulator )
```

5.63.1.2 PluginManager::PluginManager (const PluginManager & orig)

5.63.1.3 PluginManager::~PluginManager() [virtual]

5.63.2 Member Function Documentation

5.63.2.1 Plugin * PluginManager::find (std::string pluginTypeName)

Here is the call graph for this function:





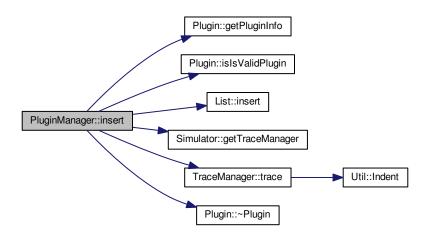
5.63.2.2 Plugin * PluginManager::front ()

Here is the call graph for this function:



5.63.2.3 bool PluginManager::insert (Plugin * plugin)

Here is the call graph for this function:





5.63.2.4 Plugin * PluginManager::last ()

Here is the call graph for this function:



5.63.2.5 Plugin * PluginManager::next ()

Here is the call graph for this function:



5.63.2.6 void PluginManager::remove (Plugin * plugin)

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

- PluginManager.h
- PluginManager.cpp

5.64 ProbDistrib Class Reference

#include <ProbDistrib.h>

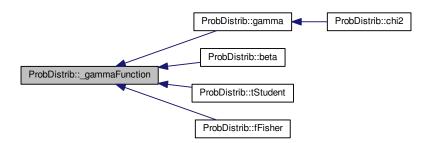
Static Public Member Functions

- 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, unsigned int alpha, double beta)
- 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)
- static double tStudent (double x, double mean, double stddev, unsigned int degreeFreedom)
- static double fFisher (double x, double k, double m)
- static double chi2 (double x, double m)
- static double inverseNormal (double cumulativeProbability, double mean, double stddev)
- static double inverseTStudent (double cumulativeProbability, double mean, double stddev, double degree ←
 Freedom)
- static double inverseFSnedecor (double cumulativeProbability, double u, double v)
- static double inverseChi2 (double cumulativeProbability, double m)
- static long double <u>gammaFunction</u> (double z)

5.64.1 Member Function Documentation

5.64.1.1 long double ProbDistrib::_gammaFunction (double z) [static]

Here is the caller graph for this function:



5.64.1.2 double ProbDistrib::beta (double x, double alpha, double beta) [static]



5.64.1.3 double ProbDistrib::chi2 (double x, double m) [static]

Here is the call graph for this function:



5.64.1.4 double ProbDistrib::erlang (double x, double mean, double M) [static]

5.64.1.5 double ProbDistrib::exponential (double x, double mean) [static]

5.64.1.6 double ProbDistrib::fFisher (double x, double k, double m) [static]

Here is the call graph for this function:



5.64.1.7 double ProbDistrib::gamma (double x, unsigned int alpha, double beta) [static]

Here is the call graph for this function:





5.64.1.8 double ProbDistrib::inverseChi2 (double cumulativeProbability, double m) [static]
5.64.1.9 double ProbDistrib::inverseFSnedecor (double cumulativeProbability, double u, double v) [static]
5.64.1.10 double ProbDistrib::inverseNormal (double cumulativeProbability, double mean, double stddev) [static]
5.64.1.11 double ProbDistrib::inverseTStudent (double cumulativeProbability, double mean, double stddev, double degreeFreedom) [static]
5.64.1.12 double ProbDistrib::logNormal (double x, double mean, double stddev) [static]
5.64.1.13 double ProbDistrib::normal (double x, double mean, double stddev) [static]
Here is the caller graph for this function:

ProbDistrib::normal testStudentSoftwareDevelopments TestInputAnalyserTools ::main

5.64.1.14 double ProbDistrib::triangular (double x, double min, double mode, double max) [static]

5.64.1.15 double ProbDistrib::tStudent (double x, double mean, double stddev, unsigned int degreeFreedom) [static]

Here is the call graph for this function:



5.64.1.16 double ProbDistrib::uniform (double x, double min, double max) [static]



5.64.1.17 double ProbDistrib::weibull (double x, double alpha, double scale) [static]

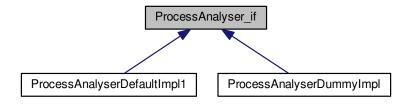
The documentation for this class was generated from the following files:

- · ProbDistrib.h
- ProbDistrib.cpp

5.65 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 modelFilename) const =0
- virtual List< SimulationResponse * > * extractResponsesFromModel (std::string modelFilename) const =0
- virtual void startSimulationOfScenario (SimulationScenario *scenario)=0
- virtual void startSimulation ()=0
- virtual void stopSimulation ()=0
- virtual void addTraceSimulationHandler (traceSimulationProcessListener traceSimulationProcessListener)=0

5.65.1 Detailed Description

The process analyser allows to extract controls and responses from a model, incluse some of then as controls and responses for a set of scenarios to be simulated

5.65.2 Member Function Documentation

5.65.2.1 virtual void ProcessAnalyser_if::addTraceSimulationHandler (traceSimulationProcessListener traceSimulationProcessListener) [pure virtual]

Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.

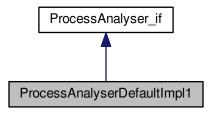
```
5.65.2.2 virtual List<SimulationControl*>* ProcessAnalyser_if::extractControlsFromModel ( std::string modelFilename )
        const [pure virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
5.65.2.3 virtual List<SimulationResponse*>* ProcessAnalyser_if::extractResponsesFromModel ( std::string
        modelFilename ) const [pure virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
5.65.2.4 virtual List < Simulation Control *> * Process Analyser_if::getControls() const [pure virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
5.65.2.5 virtual List<SimulationResponse*>* ProcessAnalyser_if::getResponses( ) const [pure virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
5.65.2.6 virtual List<SimulationScenario*>* ProcessAnalyser_if::getScenarios() const [pure virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
5.65.2.7 virtual void ProcessAnalyser_if::startSimulation() [pure virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
5.65.2.8 virtual void ProcessAnalyser if::startSimulationOfScenario ( SimulationScenario * scenario ) [pure
        virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
5.65.2.9 virtual void ProcessAnalyser_if::stopSimulation() [pure virtual]
Implemented in ProcessAnalyserDefaultImpl1, and ProcessAnalyserDummyImpl.
The documentation for this class was generated from the following file:
```

ProcessAnalyser_if.h

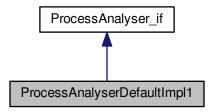
5.66 ProcessAnalyserDefaultImpl1 Class Reference

#include <ProcessAnalyserDefaultImpl1.h>

Inheritance diagram for ProcessAnalyserDefaultImpl1:



Collaboration diagram for ProcessAnalyserDefaultImpl1:



Public Member Functions

- ProcessAnalyserDefaultImpl1 ()
- ProcessAnalyserDefaultImpl1 (const ProcessAnalyserDefaultImpl1 &orig)
- virtual ~ProcessAnalyserDefaultImpl1 ()
- virtual List< SimulationScenario * > * getScenarios () const
- virtual List< SimulationControl * > * getControls () const
- virtual List< SimulationResponse * > * getResponses () const
- virtual List< SimulationControl * > * extractControlsFromModel (std::string modelFilename) const
- virtual List< SimulationResponse * > * extractResponsesFromModel (std::string modelFilename) const
- virtual void startSimulationOfScenario (SimulationScenario *scenario)
- · virtual void startSimulation ()
- virtual void stopSimulation ()
- virtual void addTraceSimulationHandler (traceSimulationProcessListener traceSimulationProcessListener)

```
5.66.1 Constructor & Destructor Documentation
5.66.1.1 ProcessAnalyserDefaultImpl1::ProcessAnalyserDefaultImpl1 ( )
5.66.1.2 ProcessAnalyserDefaultImpl1::ProcessAnalyserDefaultImpl1 ( const ProcessAnalyserDefaultImpl1 & orig )
5.66.1.3 ProcessAnalyserDefaultImpl1::~ProcessAnalyserDefaultImpl1() [virtual]
5.66.2 Member Function Documentation
5.66.2.1 void ProcessAnalyserDefaultImpl1::addTraceSimulationHandler ( traceSimulationProcessListener
         traceSimulationProcessListener ) [virtual]
Implements ProcessAnalyser if.
5.66.2.2 List < SimulationControl * > * ProcessAnalyserDefaultImpl1::extractControlsFromModel ( std::string
         modelFilename ) const [virtual]
Implements ProcessAnalyser if.
5.66.2.3 List < SimulationResponse * > * ProcessAnalyserDefaultImpl1::extractResponsesFromModel ( std::string
         modelFilename ) const [virtual]
Implements ProcessAnalyser if.
\textbf{5.66.2.4} \quad \textbf{List} < \textbf{SimulationControl} *> * \textbf{ProcessAnalyserDefaultImpl1::getControls() const} \quad [\texttt{virtual}]
Implements ProcessAnalyser_if.
5.66.2.5 List < SimulationResponse * > * ProcessAnalyserDefaultImpl1::getResponses( ) const [virtual]
Implements ProcessAnalyser_if.
5.66.2.6 List < SimulationScenario * > * ProcessAnalyserDefaultImpl1::getScenarios() const [virtual]
Implements ProcessAnalyser_if.
5.66.2.7 void ProcessAnalyserDefaultImpl1::startSimulation() [virtual]
Implements ProcessAnalyser_if.
```

5.66.2.8 void ProcessAnalyserDefaultImpl1::startSimulationOfScenario (SimulationScenario * scenario) [virtual]

Implements ProcessAnalyser_if.

5.66.2.9 void ProcessAnalyserDefaultImpl1::stopSimulation() [virtual]

Implements ProcessAnalyser_if.

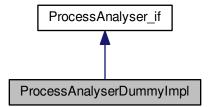
The documentation for this class was generated from the following files:

- ProcessAnalyserDefaultImpl1.h
- · ProcessAnalyserDefaultImpl1.cpp

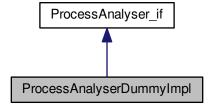
5.67 ProcessAnalyserDummyImpl Class Reference

#include <ProcessAnalyserDummyImpl.h>

Inheritance diagram for ProcessAnalyserDummyImpl:



Collaboration diagram for ProcessAnalyserDummyImpl:



Public Member Functions

- ProcessAnalyserDummyImpl ()
- ProcessAnalyserDummyImpl (const ProcessAnalyserDummyImpl &orig)
- ∼ProcessAnalyserDummyImpl ()
- virtual List< SimulationScenario * > * getScenarios () const
- virtual List< SimulationControl * > * getControls () const
- virtual List< SimulationResponse * > * getResponses () const
- virtual List< SimulationControl * > * extractControlsFromModel (std::string modelFilename) const
- virtual List< SimulationResponse * > * extractResponsesFromModel (std::string modelFilename) const
- virtual void startSimulationOfScenario (SimulationScenario *scenario)
- virtual void startSimulation ()
- virtual void stopSimulation ()
- virtual void addTraceSimulationHandler (traceSimulationProcessListener traceSimulationProcessListener)
- 5.67.1 Constructor & Destructor Documentation
- 5.67.1.1 ProcessAnalyserDummyImpl::ProcessAnalyserDummyImpl ()
- 5.67.1.2 ProcessAnalyserDummyImpl & orig) rocessAnalyserDummyImpl & orig)
- 5.67.1.3 ProcessAnalyserDummylmpl::∼ProcessAnalyserDummylmpl ()
- 5.67.2 Member Function Documentation
- 5.67.2.1 void ProcessAnalyserDummylmpl::addTraceSimulationHandler (traceSimulationProcessListener traceSimulationProcessListener) [virtual]

Implements ProcessAnalyser_if.

5.67.2.2 List < SimulationControl * > * ProcessAnalyserDummyImpl::extractControlsFromModel (std::string modelFilename) const [virtual]

Implements ProcessAnalyser if.

5.67.2.3 List< SimulationResponse * > * ProcessAnalyserDummyImpl::extractResponsesFromModel (std::string modelFilename) const [virtual]

Implements ProcessAnalyser_if.

5.67.2.4 List < SimulationControl * > * ProcessAnalyserDummylmpl::getControls() const [virtual]

Implements ProcessAnalyser_if.

 $\textbf{5.67.2.5} \quad \textbf{List} < \textbf{SimulationResponse} * > * \textbf{ProcessAnalyserDummyImpl::getResponses () const} \quad [\texttt{virtual}]$

Implements ProcessAnalyser_if.

5.68 Queue Class Reference 249

```
5.67.2.6 List< SimulationScenario * > * ProcessAnalyserDummyImpl::getScenarios() const [virtual]

Implements ProcessAnalyser_if.

5.67.2.7 void ProcessAnalyser_if.

5.67.2.8 void ProcessAnalyser_if.

5.67.2.8 void ProcessAnalyser_if.

5.67.2.9 void ProcessAnalyser_if.

5.67.2.9 void ProcessAnalyser_if.
```

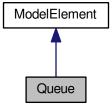
The documentation for this class was generated from the following files:

- · ProcessAnalyserDummyImpl.h
- ProcessAnalyserDummyImpl.cpp

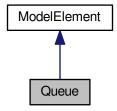
5.68 Queue Class Reference

#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, Order
 Rule::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)
- void initBetweenReplication ()
- unsigned int size ()
- Waiting * first ()
- void setAttributeName (std::string _attributeName)
- std::string getAttributeName () const
- void setOrderRule (OrderRule _orderRule)
- · Queue::OrderRule getOrderRule () const

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelElement * LoadInstance (ElementManager *elems, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual bool _loadInstance (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

5.68.1 Member Enumeration Documentation

5.68.1.1 enum Queue::OrderRule:int [strong]

Enumerator

FIFO

LIFO

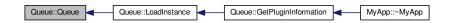
HIGHESTVALUE

SMALLESTVALUE

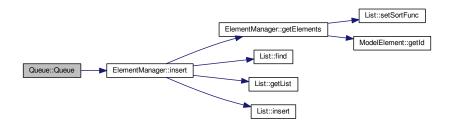
5.68.2 Constructor & Destructor Documentation

5.68.2.1 Queue::Queue (ElementManager * elems)

Here is the caller graph for this function:



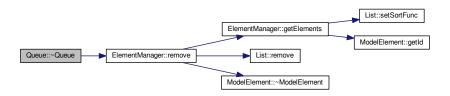
5.68.2.2 Queue::Queue (ElementManager * elems, std::string name)



5.68.2.3 Queue::Queue (const Queue & orig)

5.68.2.4 Queue::∼Queue() [virtual]

Here is the call graph for this function:

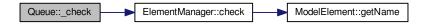


5.68.3 Member Function Documentation

5.68.3.1 bool Queue::_check (std::string * *errorMessage* **)** [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



5.68.3.2 bool Queue::_loadInstance (std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:





5.68 Queue Class Reference 253

```
5.68.3.3 std::map< std::string, std::string > * Queue::_saveInstance( ) [protected], [virtual]
```

Reimplemented from ModelElement.

Here is the call graph for this function:



```
5.68.3.4 Waiting * Queue::first()
```

Here is the call graph for this function:



Here is the caller graph for this function:

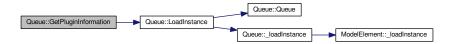


5.68.3.5 std::string Queue::getAttributeName () const

5.68.3.6 Queue::OrderRule Queue::getOrderRule () const

5.68.3.7 PluginInformation * Queue::GetPluginInformation () [static]

Here is the call graph for this function:



Here is the caller graph for this function:



5.68.3.8 void Queue::initBetweenReplication ()

Here is the call graph for this function:

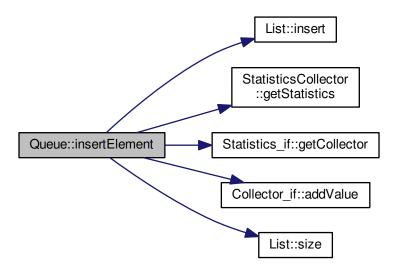




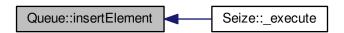
5.68 Queue Class Reference 255

5.68.3.9 void Queue::insertElement (Waiting * element)

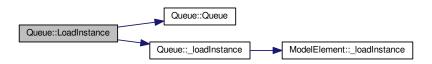
Here is the call graph for this function:



Here is the caller graph for this function:



5.68.3.10 ModelElement * Queue::LoadInstance (ElementManager * elems, std::map< std::string, std::string > * fields) [static]

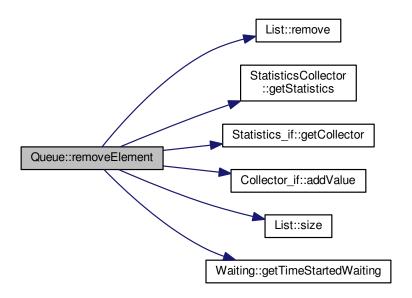


Here is the caller graph for this function:



5.68.3.11 void Queue::removeElement (Waiting * element, double tnow)

Here is the call graph for this function:



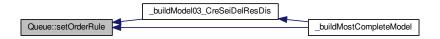


5.68 Queue Class Reference 257

5.68.3.12 void Queue::setAttributeName (std::string _attributeName)

5.68.3.13 void Queue::setOrderRule (OrderRule _orderRule)

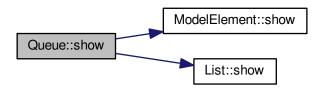
Here is the caller graph for this function:



5.68.3.14 std::string Queue::show() [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



5.68.3.15 unsigned int Queue::size ()



Here is the caller graph for this function:



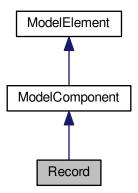
The documentation for this class was generated from the following files:

- Queue.h
- Queue.cpp

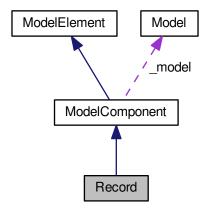
5.69 Record Class Reference

#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 ()

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

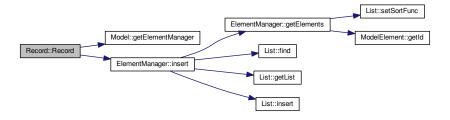
- virtual void _execute (Entity *entity)
- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual void initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

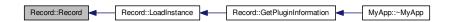
5.69.1 Constructor & Destructor Documentation

5.69.1.1 Record::Record (Model * model)

Here is the call graph for this function:

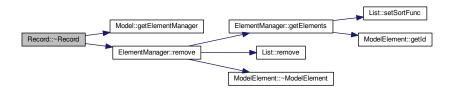


Here is the caller graph for this function:



5.69.1.2 Record::Record (const Record & orig)

```
5.69.1.3 Record::\simRecord( ) [virtual]
```



5.69.2 Member Function Documentation

5.69.2.1 bool Record::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

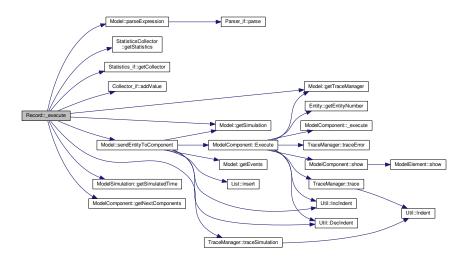
Here is the call graph for this function:



5.69.2.2 void Record::_execute (Entity * *entity*) [protected], [virtual]

Implements ModelComponent.

Here is the call graph for this function:



5.69.2.3 void Record::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

```
5.69.2.4 bool Record::_loadInstance( std::map < std::string, std::string > * fields ) [protected], [virtual]
```

Reimplemented from ModelComponent.

Here is the call graph for this function:



Here is the caller graph for this function:

```
Record::_loadInstance Record::GetPluginInformation MyApp::~MyApp
```

```
5.69.2.5 std::map < std::string, std::string > * Record::_saveInstance( ) [protected], [virtual]
```

Reimplemented from ModelComponent.

Here is the call graph for this function:



5.69.2.6 StatisticsCollector * Record::getCstatExpression () const

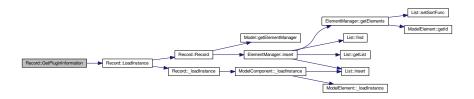
5.69.2.7 std::string Record::getExpression () const

5.69.2.8 std::string Record::getExpressionName () const

5.69.2.9 std::string Record::getFilename () const

5.69.2.10 PluginInformation * Record::GetPluginInformation() [static]

Here is the call graph for this function:

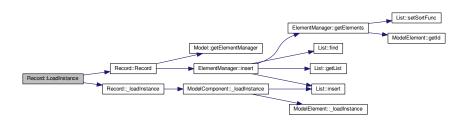


Here is the caller graph for this function:



$\textbf{5.69.2.11} \quad \textbf{ModelComponent} * \textbf{Record::LoadInstance (Model} * \textit{model}, \ \textit{std::map} < \textit{std::string, std::string} > * \textit{fields)} \\ [\textit{static}]$

Here is the call graph for this function:





5.69.2.12 void Record::setExpression (std::string expression)

Here is the caller graph for this function:



5.69.2.13 void Record::setExpressionName (std::string expressionName)

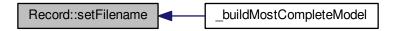
Here is the call graph for this function:



Here is the caller graph for this function:



5.69.2.14 void Record::setFilename (std::string filename)



```
5.69.2.15 std::string Record::show( ) [virtual]
```

Reimplemented from ModelComponent.

Here is the call graph for this function:



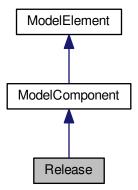
The documentation for this class was generated from the following files:

- · Record.h
- Record.cpp

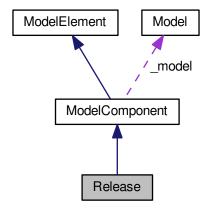
5.70 Release Class Reference

```
#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

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

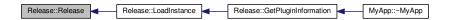
- virtual void execute (Entity *entity)
- virtual bool _loadInstance (std::map< std::string, std::string > *fields)
- virtual void initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

5.70.1 Constructor & Destructor Documentation

5.70.1.1 Release::Release (Model * model)

Here is the caller graph for this function:



5.70.1.2 Release::Release (const Release & orig)

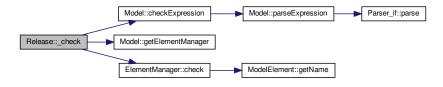
```
5.70.1.3 Release::~Release() [virtual]
```

5.70.2 Member Function Documentation

```
5.70.2.1 bool Release::_check( std::string * errorMessage ) [protected], [virtual]
```

Reimplemented from ModelElement.

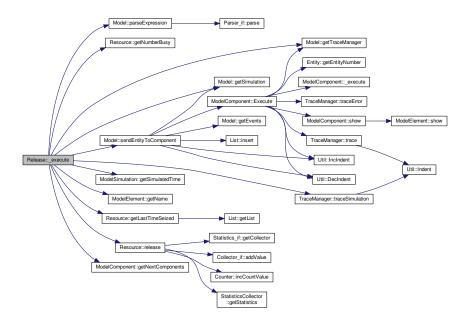
Here is the call graph for this function:



5.70.2.2 void Release::_execute(Entity * *entity* **)** [protected], [virtual]

Implements ModelComponent.

Here is the call graph for this function:



5.70.2.3 void Release::_initBetweenReplications() [protected], [virtual]

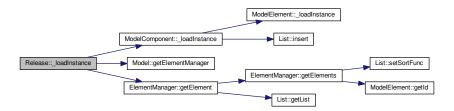
Implements ModelComponent.

Here is the call graph for this function:



5.70.2.4 bool Release::_loadInstance(std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelComponent.



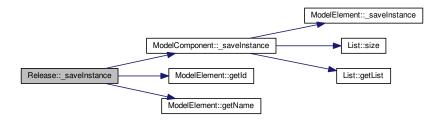
Here is the caller graph for this function:



5.70.2.5 std::map< std::string, std::string > * Release::_saveInstance() [protected], [virtual]

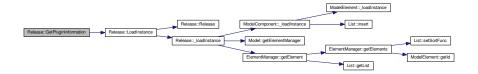
Reimplemented from ModelComponent.

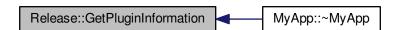
Here is the call graph for this function:



5.70.2.6 PluginInformation * Release::GetPluginInformation () [static]

Here is the call graph for this function:





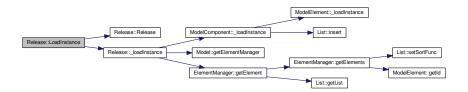
- 5.70.2.7 unsigned short Release::getPriority () const
- 5.70.2.8 std::string Release::getQuantity () const
- 5.70.2.9 Resource * Release::getResource () const
- 5.70.2.10 std::string Release::getResourceName () const

Here is the call graph for this function:



- 5.70.2.11 Resource::ResourceType Release::getResourceType () const
- 5.70.2.12 Resource::ResourceRule Release::getRule () const
- 5.70.2.13 std::string Release::getSaveAttribute () const
- $\textbf{5.70.2.14} \quad \textbf{ModelComponent} * \textbf{Release::LoadInstance (Model} * \textit{model, std::map} < \textbf{std::string, std::string} > * \textit{fields)} \\ [\, \texttt{static} \,]$

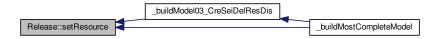
Here is the call graph for this function:





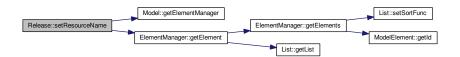
- 5.70.2.15 void Release::setPriority (unsigned short _priority)
- 5.70.2.16 void Release::setQuantity (std::string _quantity)
- 5.70.2.17 void Release::setResource (Resource * _resource)

Here is the caller graph for this function:



5.70.2.18 void Release::setResourceName (std::string resourceName) throw)

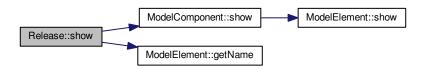
Here is the call graph for this function:



- 5.70.2.19 void Release::setResourceType (Resource::ResourceType _resourceType)
- 5.70.2.20 void Release::setRule (Resource::ResourceRule _rule)
- 5.70.2.21 void Release::setSaveAttribute (std::string _saveAttribute)
- 5.70.2.22 std::string Release::show() [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



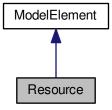
The documentation for this class was generated from the following files:

- Release.h
- Release.cpp

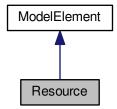
5.71 Resource Class Reference

#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, Resource
 Rule::SMALLESTBUSY = 4,
 ResourceRule::LARGESTREMAININGCAPACITY = 5 }
- enum ResourceState: int {
 ResourceState::IDLE = 1, ResourceState::BUSY = 2, ResourceState::FAILED = 3, ResourceState::INACT
 IVE = 4,
 ResourceState::OTHER = 5 }
- $\bullet \ \ typedef \ std:: function < void (Resource *) > Resource Event Handler \\$

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 initBetweenReplications ()
- 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
- void addResourceEventHandler (ResourceEventHandler eventHandler)
- double getLastTimeSeized () const

Static Public Member Functions

- template<typename Class >
 static ResourceEventHandler SetResourceEventHandler (void(Class::*function)(Resource*), Class *object)
- static PluginInformation * GetPluginInformation ()
- static ModelElement * LoadInstance (ElementManager *elems, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

5.71.1 Detailed Description

Resource represents a facility that...

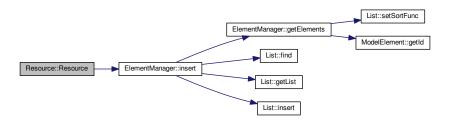
5.71.2 Member Typedef Documentation 5.71.2.1 typedef std::function<void(Resource*) > Resource::ResourceEventHandler 5.71.3 Member Enumeration Documentation **5.71.3.1 enum Resource::ResourceRule:int** [strong] Enumerator **RANDOM** CICLICAL **ESPECIFIC SMALLESTBUSY LARGESTREMAININGCAPACITY 5.71.3.2 enum Resource::ResourceState:int** [strong] Enumerator IDLE BUSY **FAILED INACTIVE** OTHER **5.71.3.3 enum Resource::ResourceType:int** [strong] Enumerator SET RESOURCE 5.71.4 Constructor & Destructor Documentation 5.71.4.1 Resource::Resource (ElementManager * elems) Here is the caller graph for this function:

Resource::LoadInstance

Resource::GetPluginInformation

5.71.4.2 Resource::Resource (ElementManager * elems, std::string name)

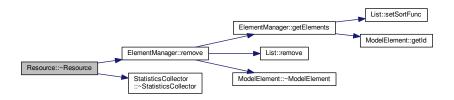
Here is the call graph for this function:



5.71.4.3 Resource::Resource (const Resource & orig)

5.71.4.4 Resource::~Resource() [virtual]

Here is the call graph for this function:



5.71.5 Member Function Documentation

5.71.5.1 bool Resource::_check (std::string * *errorMessage* **)** [protected], [virtual]

Reimplemented from ModelElement.

5.71.5.2 bool Resource::_loadInstance(std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelElement.



Here is the caller graph for this function:



5.71.5.3 std::map< **std::string**, **std::string** > * **Resource::_saveInstance()** [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



5.71.5.4 void Resource::addResourceEventHandler (ResourceEventHandler eventHandler)

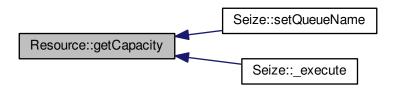
Here is the call graph for this function:





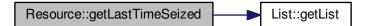
5.71.5.5 unsigned int Resource::getCapacity () const

Here is the caller graph for this function:



- $5.71.5.6 \quad double \ Resource:: get Cost Busy Hour \ (\quad) \ const$
- 5.71.5.7 double Resource::getCostIdleHour () const
- 5.71.5.8 double Resource::getCostPerUse () const
- 5.71.5.9 double Resource::getLastTimeSeized () const

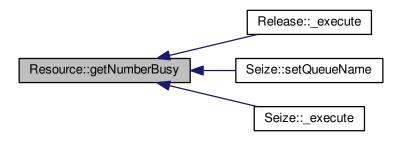
Here is the call graph for this function:





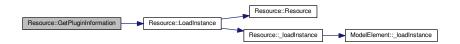
5.71.5.10 unsigned int Resource::getNumberBusy () const

Here is the caller graph for this function:



5.71.5.11 PluginInformation * Resource::GetPluginInformation () [static]

Here is the call graph for this function:



Here is the caller graph for this function:



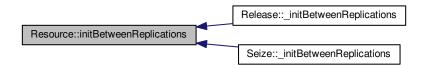
5.71.5.12 Resource::ResourceState Resource::getResourceState () const

5.71.5.13 void Resource::initBetweenReplications ()

Here is the call graph for this function:

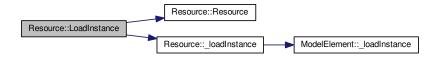


Here is the caller graph for this function:



5.71.5.14 ModelElement * Resource::LoadInstance (ElementManager * elems, std::map < std::string, std::string > * fields) [static]

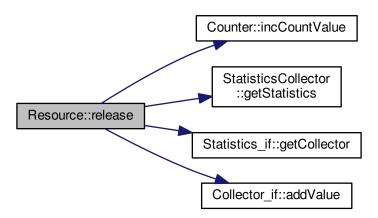
Here is the call graph for this function:





5.71.5.15 void Resource::release (unsigned int quantity, double tnow)

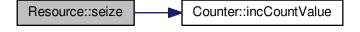
Here is the call graph for this function:



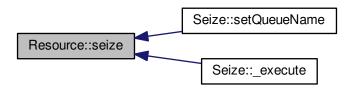
Here is the caller graph for this function:



5.71.5.16 void Resource::seize (unsigned int quantity, double tnow)



Here is the caller graph for this function:



5.71.5.17 void Resource::setCapacity (unsigned int _capacity)

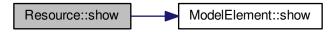
Here is the caller graph for this function:



- 5.71.5.18 void Resource::setCostBusyHour (double _costBusyHour)
- 5.71.5.19 void Resource::setCostIdleHour (double $_costIdleHour$)
- $5.71.5.20 \quad \text{void Resource::setCostPerUse (double } _\textit{costPerUse} \)$
- 5.71.5.21 template < typename Class > static Resource Event Handler Resource :: Set Resource Event Handler (void (Class::*)(Resource*) function, Class*object) [inline], [static]
- 5.71.5.22 void Resource::setResourceState (ResourceState _resourceState)
- 5.71.5.23 std::string Resource::show() [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



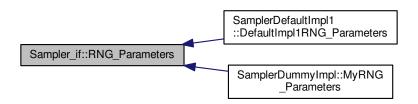
The documentation for this class was generated from the following files:

- · Resource.h
- · Resource.cpp

5.72 Sampler_if::RNG_Parameters Struct Reference

```
#include <Sampler_if.h>
```

Inheritance diagram for Sampler_if::RNG_Parameters:



Public Member Functions

virtual ∼RNG_Parameters ()=default

5.72.1 Detailed Description

class that encapsulates attributes required to generate random numbers, which depends on the generation method used.

5.72.2 Constructor & Destructor Documentation

5.72.2.1 virtual Sampler_if::RNG_Parameters::~RNG_Parameters() [virtual], [default]

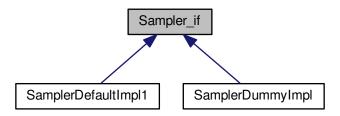
The documentation for this struct was generated from the following file:

Sampler_if.h

5.73 Sampler_if Class Reference

```
#include <Sampler_if.h>
```

Inheritance diagram for Sampler if:



Classes

• struct 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

5.73.1 Detailed Description

Interface that describes the methods to be implemented by classes that generate random values that follow a specific probability distribution.

5.73.2 Member Function Documentation

5.73.2.1 virtual RNG_Parameters* Sampler_if::getRNGparameters()const [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

Here is the caller graph for this function:



5.73.2.2 virtual double Sampler_if::random() [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

5.73.2.3 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:

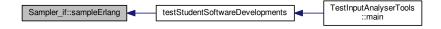


5.73.2.4 virtual double Sampler if::sampleDiscrete (double value, double acumProb, ...) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

5.73.2.5 virtual double Sampler_if::sampleErlang (double mean, int M) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.



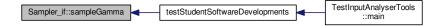
5.73.2.6 virtual double Sampler_if::sampleExponential (double *mean* **)** [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

5.73.2.7 virtual double Sampler_if::sampleGamma (double mean, double alpha) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

Here is the caller graph for this function:



5.73.2.8 virtual double Sampler_if::sampleLogNormal (double mean, double stddev) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

5.73.2.9 virtual double Sampler_if::sampleNormal (double mean, double stddev) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

Here is the caller graph for this function:



5.73.2.10 virtual double Sampler_if::sampleTriangular (double min, double mode, double max) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.



5.73.2.11 virtual double Sampler_if::sampleUniform (double *min*, double *max*) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

5.73.2.12 virtual double Sampler_if::sampleWeibull (double *alpha*, double *scale*) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

Here is the caller graph for this function:



5.73.2.13 virtual void Sampler_if::setRNGparameters (RNG_Parameters * param) [pure virtual]

Implemented in SamplerDefaultImpl1, and SamplerDummyImpl.

Here is the caller graph for this function:



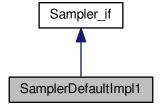
The documentation for this class was generated from the following file:

· Sampler_if.h

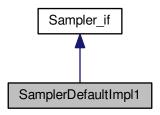
5.74 SamplerDefaultImpl1 Class Reference

#include <SamplerDefaultImpl1.h>

Inheritance diagram for SamplerDefaultImpl1:



Collaboration diagram for SamplerDefaultImpl1:



Classes

• struct DefaultImpl1RNG_Parameters

Public Member Functions

- SamplerDefaultImpl1 ()
- SamplerDefaultImpl1 (const SamplerDefaultImpl1 &orig)
- virtual \sim SamplerDefaultImpl1 ()
- virtual double random ()
- virtual double sampleUniform (double min, double max)
- virtual double sampleExponential (double mean)
- virtual double sampleErlang (double mean, int M)
- virtual double sampleNormal (double mean, double stddev)
- virtual double sampleGamma (double mean, double alpha)
- · virtual double sampleBeta (double alpha, double beta, double infLimit, double supLimit)
- virtual double sampleWeibull (double alpha, double scale)
- virtual double sampleLogNormal (double mean, double stddev)
- virtual double sampleTriangular (double min, double mode, double max)
- virtual double sampleDiscrete (double value, double acumProb,...)
- void reset ()

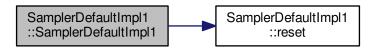
reinitialize seed and other parameters so (pseudo) random number sequence will be generated again.

- virtual void setRNGparameters (RNG Parameters *param)
- virtual RNG_Parameters * getRNGparameters () const

5.74.1 Constructor & Destructor Documentation

5.74.1.1 SamplerDefaultImpl1::SamplerDefaultImpl1 ()

Here is the call graph for this function:

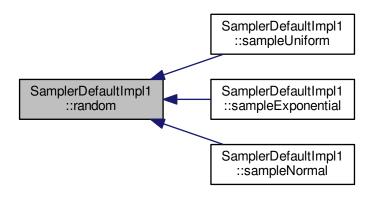


- 5.74.1.2 SamplerDefaultImpl1::SamplerDefaultImpl1 (const SamplerDefaultImpl1 & orig)
- **5.74.1.3 SamplerDefaultImpl1::**~SamplerDefaultImpl1() [virtual]
- 5.74.2 Member Function Documentation
- 5.74.2.1 Sampler_if::RNG_Parameters * SamplerDefaultImpl1::getRNGparameters() const [virtual]

Implements Sampler_if.

5.74.2.2 double SamplerDefaultImpl1::random() [virtual]

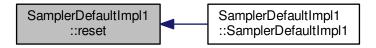
Implements Sampler_if.



```
5.74.2.3 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:



5.74.2.4 double SamplerDefaultImpl1::sampleBeta (double *alpha*, double *beta*, double *infLimit*, double *supLimit*) [virtual]

Implements Sampler_if.

5.74.2.5 double SamplerDefaultImpl1::sampleDiscrete (double value, double acumProb, ...) [virtual]

Implements Sampler_if.

5.74.2.6 double SamplerDefaultImpl1::sampleErlang (double *mean*, int *M*) [virtual]

Implements Sampler_if.

5.74.2.7 double SamplerDefaultImpl1::sampleExponential (double mean) [virtual]

Implements Sampler_if.



```
double SamplerDefaultImpl1::sampleGamma ( double mean, double alpha ) [virtual]
Implements Sampler_if.
5.74.2.9 double SamplerDefaultImpl1::sampleLogNormal ( double mean, double stddev ) [virtual]
Implements Sampler_if.
5.74.2.10 double SamplerDefaultImpl1::sampleNormal ( double mean, double stddev ) [virtual]
Implements Sampler_if.
Here is the call graph for this function:
                          SamplerDefaultImpl1
                                                             SamplerDefaultImpl1
                             ::sampleNormal
                                                                   ::random
5.74.2.11 double SamplerDefaultImpl1::sampleTriangular ( double min, double mode, double max ) [virtual]
Implements Sampler_if.
5.74.2.12 double SamplerDefaultImpl1::sampleUniform ( double min, double max ) [virtual]
Implements Sampler_if.
Here is the call graph for this function:
                          SamplerDefaultImpl1
                                                             SamplerDefaultImpl1
                            ::sampleUniform
                                                                   ::random
```

5.74.2.13 double SamplerDefaultImpl1::sampleWeibull (double alpha, double scale) [virtual]

Implements Sampler_if.

5.74.2.14 void SamplerDefaultImpl1::setRNGparameters (Sampler_if::RNG_Parameters * param) [virtual]

Implements Sampler_if.

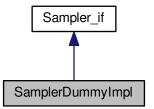
The documentation for this class was generated from the following files:

- SamplerDefaultImpl1.h
- · SamplerDefaultImpl1.cpp

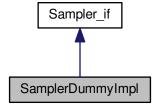
5.75 SamplerDummyImpl Class Reference

#include <SamplerDummyImpl.h>

Inheritance diagram for SamplerDummyImpl:



Collaboration diagram for SamplerDummyImpl:



Classes

• struct 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

5.75.1 Constructor & Destructor Documentation

- 5.75.1.1 SamplerDummyImpl::SamplerDummyImpl ()
- 5.75.1.2 SamplerDummyImpl::SamplerDummyImpl (const SamplerDummyImpl & orig)
- 5.75.1.3 SamplerDummyImpl::~SamplerDummyImpl ()
- 5.75.2 Member Function Documentation
- 5.75.2.1 Sampler if::RNG Parameters * SamplerDummyImpl::getRNGparameters() const [virtual]

Implements Sampler if.

5.75.2.2 double SamplerDummyImpl::random() [virtual]

Implements Sampler_if.

5.75.2.3 double SamplerDummyImpl::sampleBeta (double *alpha*, double *beta*, double *infLimit*, double *supLimit*) [virtual]

Implements Sampler_if.

```
5.75.2.4 double SamplerDummyImpl::sampleDiscrete ( double value, double acumProb, ... ) [virtual]
Implements Sampler if.
5.75.2.5 double SamplerDummyImpl::sampleErlang ( double mean, int M ) [virtual]
Implements Sampler if.
5.75.2.6 double SamplerDummyImpl::sampleExponential ( double mean ) [virtual]
Implements Sampler_if.
5.75.2.7 double SamplerDummyImpl::sampleGamma ( double mean, double alpha ) [virtual]
Implements Sampler if.
5.75.2.8 double SamplerDummyImpl::sampleLogNormal ( double mean, double stddev ) [virtual]
Implements Sampler_if.
5.75.2.9 double SamplerDummyImpl::sampleNormal ( double mean, double stddev ) [virtual]
Implements Sampler_if.
5.75.2.10 double SamplerDummyImpl::sampleTriangular ( double min, double mode, double max ) [virtual]
Implements Sampler_if.
5.75.2.11 double SamplerDummyImpl::sampleUniform ( double min, double max ) [virtual]
Implements Sampler_if.
5.75.2.12 double SamplerDummyImpl::sampleWeibull ( double alpha, double scale ) [virtual]
Implements Sampler_if.
5.75.2.13 void Sampler Dummylmpl::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

5.76 ScenarioExperiment_if Class Reference

#include <ScenarioExperiment_if.h>

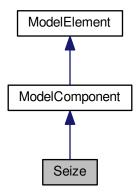
The documentation for this class was generated from the following file:

• ScenarioExperiment_if.h

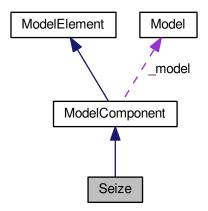
5.77 Seize Class Reference

#include <Seize.h>

Inheritance diagram for Seize:



Collaboration diagram for Seize:



5.77 Seize Class Reference 295

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

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelComponent * LoadInstance (Model *model, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual void <u>execute</u> (Entity *entity)
- virtual bool loadInstance (std::map< std::string, std::string > *fields)
- virtual void initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

5.77.1 Detailed Description

Seize tries to allocate a certain amount of a resource

5.77.2 Constructor & Destructor Documentation

5.77.2.1 Seize::Seize (Model * model)

Here is the caller graph for this function:



5.77.2.2 Seize::Seize (const Seize & orig)

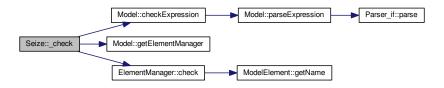
5.77.2.3 Seize::∼Seize() [virtual]

5.77.3 Member Function Documentation

5.77.3.1 bool Seize::_check (std::string * *errorMessage* **)** [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:

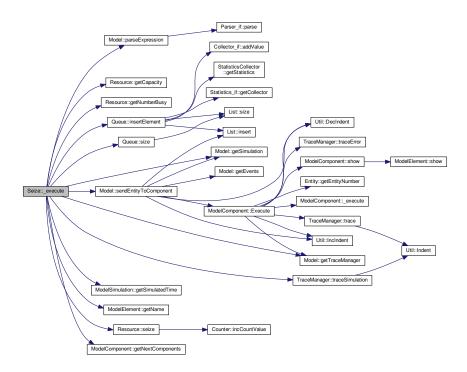


5.77.3.2 void Seize::_execute(Entity * *entity* **)** [protected], [virtual]

Implements ModelComponent.

5.77 Seize Class Reference 297

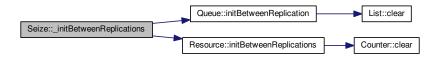
Here is the call graph for this function:



5.77.3.3 void Seize::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

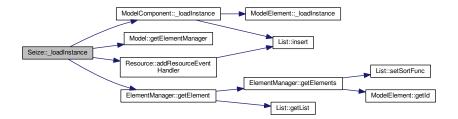
Here is the call graph for this function:



5.77.3.4 bool Seize::_loadInstance(std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



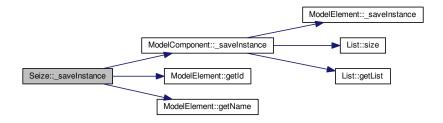
Here is the caller graph for this function:



5.77.3.5 $std::map < std::string, std::string > * Seize::_saveInstance() [protected], [virtual]$

Reimplemented from ModelComponent.

Here is the call graph for this function:



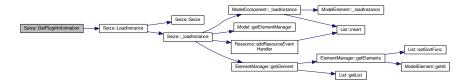
5.77.3.6 unsigned int Seize::getAllocationType () const

5.77.3.7 unsigned int Seize::getLastMemberSeized () const

5.77 Seize Class Reference 299

5.77.3.8 PluginInformation * Seize::GetPluginInformation () [static]

Here is the call graph for this function:



Here is the caller graph for this function:



- 5.77.3.9 unsigned short Seize::getPriority () const
- 5.77.3.10 std::string Seize::getQuantity () const
- 5.77.3.11 Queue * Seize::getQueue () const
- 5.77.3.12 std::string Seize::getQueueName () const



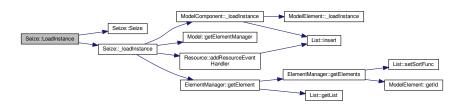
- 5.77.3.13 Resource * Seize::getResource () const
- 5.77.3.14 std::string Seize::getResourceName () const

Here is the call graph for this function:



- 5.77.3.15 Resource::ResourceType Seize::getResourceType () const
- 5.77.3.16 Resource::ResourceRule Seize::getRule () const
- 5.77.3.17 std::string Seize::getSaveAttribute () const
- 5.77.3.18 ModelComponent * Seize::LoadInstance (Model * model, std::map< std::string, std::string > * fields) [static]

Here is the call graph for this function:

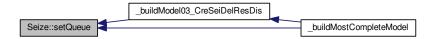




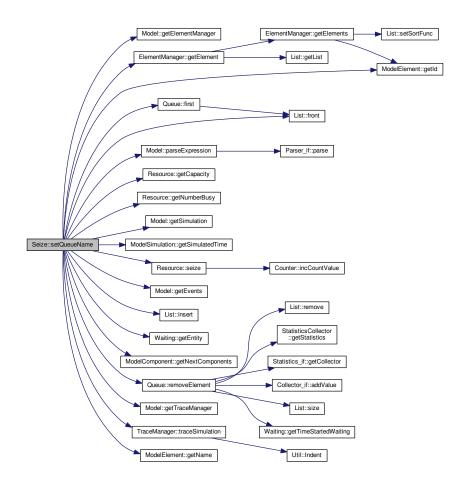
5.77 Seize Class Reference 301

- 5.77.3.19 void Seize::setAllocationType (unsigned int _allocationType)
- 5.77.3.20 void Seize::setLastMemberSeized (unsigned int _lastMemberSeized)
- 5.77.3.21 void Seize::setPriority (unsigned short _priority)
- 5.77.3.22 void Seize::setQuantity (std::string _quantity)
- 5.77.3.23 void Seize::setQueue (Queue * queue)

Here is the caller graph for this function:

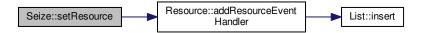


5.77.3.24 void Seize::setQueueName (std::string queueName) throw)



```
5.77.3.25 void Seize::setResource ( Resource * resource )
```

Here is the call graph for this function:

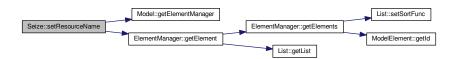


Here is the caller graph for this function:



$5.77.3.26 \quad \text{void Seize::setResourceName (} \text{std::string} _\textit{resourceName } \text{) } \text{throw)}$

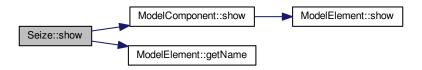
Here is the call graph for this function:



- $5.77.3.27 \quad \text{void Seize::setResourceType (} \textbf{Resource::ResourceType} \ _\textbf{resourceType} \)$
- 5.77.3.28 void Seize::setRule (Resource::ResourceRule _rule)
- 5.77.3.29 void Seize::setSaveAttribute (std::string _saveAttribute)
- 5.77.3.30 std::string Seize::show() [virtual]

Reimplemented from ModelComponent.

Here is the call graph for this function:



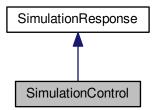
The documentation for this class was generated from the following files:

- · Seize.h
- · Seize.cpp

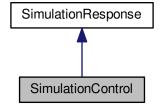
5.78 SimulationControl Class Reference

#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

5.78.1 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

5.78.2 Constructor & Destructor Documentation

- 5.78.2.1 SimulationControl::SimulationControl (std::string *type*, std::string *name*, GetterMember *getterMember*, SetterMember setterMember)
- 5.78.2.2 SimulationControl::SimulationControl (const SimulationControl & orig)
- 5.78.2.3 SimulationControl:: \sim SimulationControl() [virtual]

5.78.3 Member Function Documentation

5.78.3.1 void SimulationControl::setValue (double value)

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

- SimulationControl.h
- SimulationControl.cpp

5.79 SimulationEvent Class Reference

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

Public Member Functions

- SimulationEvent (unsigned int replicationNumber, Event *event)
- unsigned int getReplicationNumber () const
- Event * getEventProcessed () const

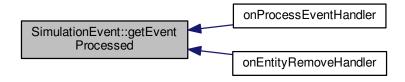
5.79.1 Constructor & Destructor Documentation

5.79.1.1 SimulationEvent::SimulationEvent (unsigned int replicationNumber, Event * event) [inline]

5.79.2 Member Function Documentation

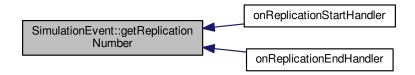
5.79.2.1 Event* SimulationEvent::getEventProcessed() const [inline]

Here is the caller graph for this function:



5.79.2.2 unsigned int SimulationEvent::getReplicationNumber () const [inline]

Here is the caller graph for this function:



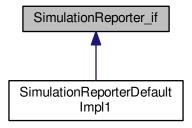
The documentation for this class was generated from the following file:

OnEventManager.h

5.80 SimulationReporter_if Class Reference

#include <SimulationReporter_if.h>

Inheritance diagram for SimulationReporter_if:



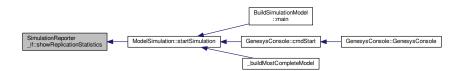
Public Member Functions

- virtual void showReplicationStatistics ()=0
- virtual void showSimulationStatistics ()=0

5.80.1 Member Function Documentation

5.80.1.1 virtual void SimulationReporter_if::showReplicationStatistics() [pure virtual]

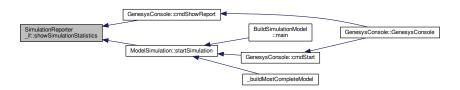
Implemented in SimulationReporterDefaultImpl1.



5.80.1.2 virtual void SimulationReporter_if::showSimulationStatistics() [pure virtual]

Implemented in SimulationReporterDefaultImpl1.

Here is the caller graph for this function:



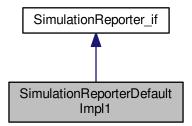
The documentation for this class was generated from the following file:

· SimulationReporter_if.h

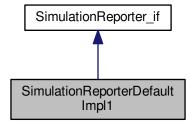
5.81 SimulationReporterDefaultImpl1 Class Reference

#include <SimulationReporterDefaultImpl1.h>

Inheritance diagram for SimulationReporterDefaultImpl1:



Collaboration diagram for SimulationReporterDefaultImpl1:



Public Member Functions

- SimulationReporterDefaultImpl1 (ModelSimulation *simulation, Model *model, List< ModelElement * > *statsCountersSimulation)
- SimulationReporterDefaultImpl1 (const SimulationReporterDefaultImpl1 &orig)
- virtual ~SimulationReporterDefaultImpl1 ()
- · virtual void showReplicationStatistics ()
- · virtual void showSimulationStatistics ()

5.81.1 Detailed Description

Class that implements SimulationReporter_if interface and is responsible for building and showing replication and simulation reports

5.81.2 Constructor & Destructor Documentation

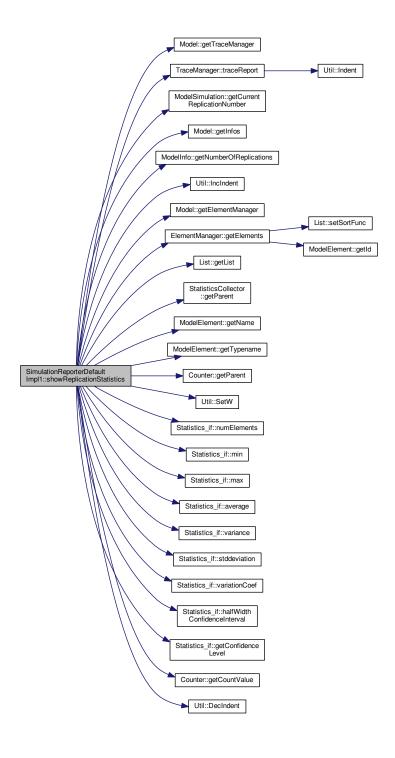
- 5.81.2.1 SimulationReporterDefaultImpl1::SimulationReporterDefaultImpl1 (ModelSimulation * simulation, Model * model, List< ModelElement * > * statsCountersSimulation)
- 5.81.2.2 SimulationReporterDefaultImpl1::SimulationReporterDefaultImpl1 (const SimulationReporterDefaultImpl1 & orig)
- 5.81.2.3 SimulationReporterDefaultImpl1::~SimulationReporterDefaultImpl1() [virtual]

5.81.3 Member Function Documentation

5.81.3.1 void SimulationReporterDefaultImpl1::showReplicationStatistics() [virtual]

Implements SimulationReporter_if.

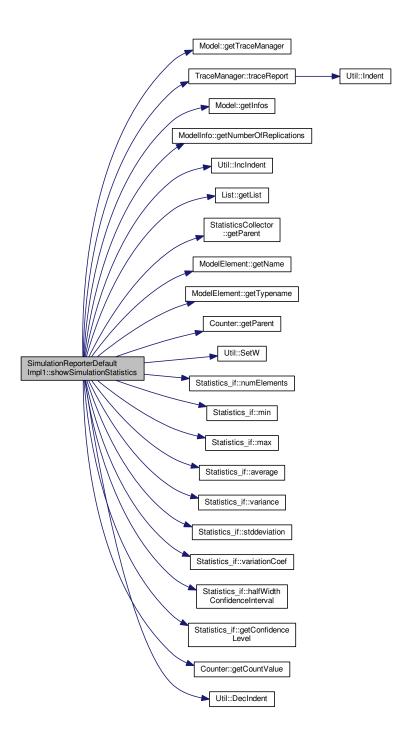
Here is the call graph for this function:



5.81.3.2 void SimulationReporterDefaultImpl1::showSimulationStatistics() [virtual]

Implements SimulationReporter_if.

Here is the call graph for this function:



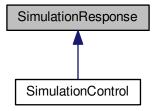
The documentation for this class was generated from the following files:

- SimulationReporterDefaultImpl1.h
- SimulationReporterDefaultImpl1.cpp

5.82 SimulationResponse Class Reference

#include <SimulationResponse.h>

Inheritance diagram for SimulationResponse:



Public Member Functions

- SimulationResponse (std::string type, std::string name, GetterMember)
- SimulationResponse (const SimulationResponse &orig)
- virtual \sim SimulationResponse ()
- double getValue ()
- std::string getName () const
- std::string getType () const

Protected Attributes

- std::string _type
- std::string _name
- GetterMember _getterMemberFunction

5.82.1 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

```
5.82.2 Constructor & Destructor Documentation

5.82.2.1 SimulationResponse::SimulationResponse ( std::string type, std::string name, GetterMember getterMember )

5.82.2.2 SimulationResponse::SimulationResponse ( const SimulationResponse & orig )

5.82.2.3 SimulationResponse::~SimulationResponse ( ) [virtual]

5.82.3 Member Function Documentation

5.82.3.1 std::string SimulationResponse::getName ( ) const

5.82.3.2 std::string SimulationResponse::getType ( ) const

5.82.3.3 double SimulationResponse::getValue ( )

5.82.4 Member Data Documentation

5.82.4.1 GetterMember SimulationResponse::_getterMemberFunction [protected]

5.82.4.2 std::string SimulationResponse::_name [protected]

5.82.4.3 std::string SimulationResponse::_type [protected]
```

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

- SimulationResponse.h
- SimulationResponse.cpp

5.83 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)

5.83.1 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.

```
5.83.2 Constructor & Destructor Documentation
5.83.2.1 SimulationScenario::SimulationScenario ( )
5.83.2.2 SimulationScenario::SimulationScenario ( const SimulationScenario & orig )
5.83.2.3 SimulationScenario::~SimulationScenario() [virtual]
5.83.3 Member Function Documentation
5.83.3.1 double SimulationScenario::getControlValue ( SimulationControl * control )
5.83.3.2 std::list < double > * SimulationScenario::getControlValues ( ) const
5.83.3.3 std::string SimulationScenario::getModelFilename ( ) const
5.83.3.4 std::string SimulationScenario::getName ( ) const
5.83.3.5 double SimulationScenario::getResponseValue ( SimulationResponse * value )
5.83.3.6 std::list< double > * SimulationScenario::getResponseValues ( ) const
5.83.3.7 void SimulationScenario::setControlValue ( SimulationControl * control, double value )
5.83.3.8 void SimulationScenario::setModelFilename ( std::string _modelFilename )
5.83.3.9 void SimulationScenario::setName ( std::string _name )
```

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

- SimulationScenario.h
- SimulationScenario.cpp

5.84 Simulator Class Reference

```
#include <Simulator.h>
```

Public Member Functions

- Simulator ()
- Simulator (const Simulator &orig)
- virtual ∼Simulator ()
- std::string getVersion () const
- std::string getName () const
- LicenceManager * getLicenceManager () const
- PluginManager * getPluginManager () const
- ModelManager * getModelManager () const
- ToolManager * getToolManager () const
- TraceManager * getTraceManager () const

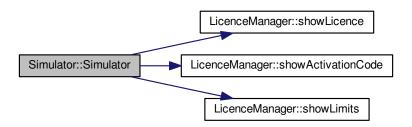
5.84.1 Detailed Description

The main class of the ReGenesys KERNEL simulation. It gives access to simulation models and tools.

5.84.2 Constructor & Destructor Documentation

5.84.2.1 Simulator::Simulator()

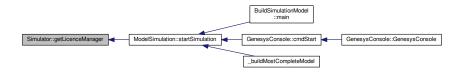
Here is the call graph for this function:



- 5.84.2.2 Simulator::Simulator (const Simulator & orig)
- 5.84.2.3 Simulator::~Simulator() [virtual]

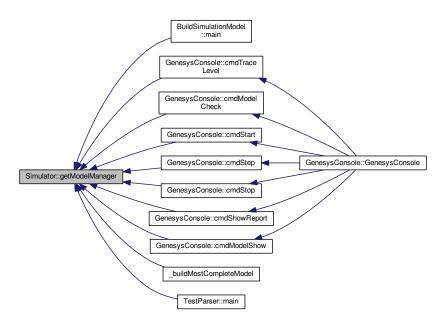
5.84.3 Member Function Documentation

5.84.3.1 LicenceManager * Simulator::getLicenceManager () const

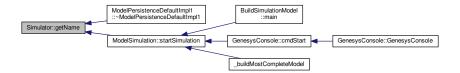


5.84.3.2 ModelManager * Simulator::getModelManager () const

Here is the caller graph for this function:

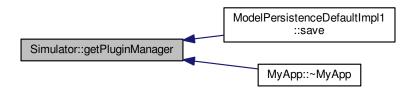


5.84.3.3 std::string Simulator::getName () const



5.84.3.4 PluginManager * Simulator::getPluginManager () const

Here is the caller graph for this function:

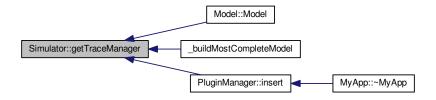


5.84.3.5 ToolManager * Simulator::getToolManager () const

Here is the caller graph for this function:

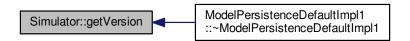


5.84.3.6 TraceManager * Simulator::getTraceManager () const



5.84.3.7 std::string Simulator::getVersion () const

Here is the caller graph for this function:



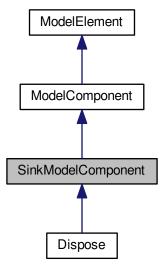
The documentation for this class was generated from the following files:

- · Simulator.h
- Simulator.cpp

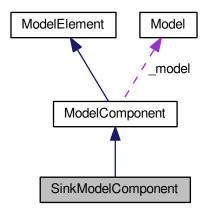
5.85 SinkModelComponent Class Reference

#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

Protected Member Functions

- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual void _initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

5.85.1 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.

5.85.2 Constructor & Destructor Documentation

5.85.2.1 SinkModelComponent::SinkModelComponent (Model * model, std::string componentTypename)

5.85.2.2 SinkModelComponent::SinkModelComponent (const SinkModelComponent & orig)

5.85.2.3 SinkModelComponent::~SinkModelComponent() [virtual]

5.85.3 Member Function Documentation

5.85.3.1 bool SinkModelComponent::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

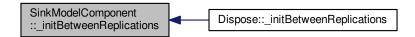
Reimplemented in Dispose.

5.85.3.2 void SinkModelComponent::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

Reimplemented in Dispose.

Here is the caller graph for this function:



5.85.3.3 bool SinkModelComponent::_loadInstance (std::map < std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelComponent.

Reimplemented in Dispose.



5.85.3.4 std::map< std::string, std::string > * SinkModelComponent::_saveInstance() [protected], [virtual]

Reimplemented from ModelComponent.

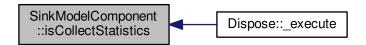
Reimplemented in Dispose.

Here is the call graph for this function:



5.85.3.5 bool SinkModelComponent::isCollectStatistics () const

Here is the caller graph for this function:



5.85.3.6 void SinkModelComponent::setCollectStatistics (bool _collectStatistics)

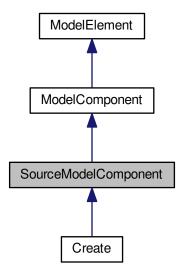
The documentation for this class was generated from the following files:

- SinkModelComponent.h
- SinkModelComponent.cpp

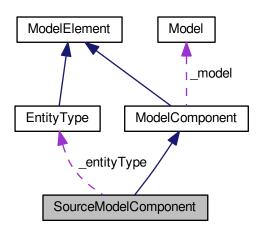
5.86 SourceModelComponent Class Reference

#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 (std::string maxCreationsExpression)
- std::string 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 Member Functions

- virtual bool <u>loadInstance</u> (std::map< std::string, std::string > *fields)
- virtual void _initBetweenReplications ()
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool check (std::string *errorMessage)

Protected Attributes

- EntityType * entityType
- double _firstCreation = 0.0
- unsigned int _entitiesPerCreation = 1
- std::string maxCreationsExpression = std::to string(std::numeric limits<unsigned int>::max())
- std::string timeBetweenCreationsExpression = "EXPO(1)"
- Util::TimeUnit _timeBetweenCreationsTimeUnit = Util::TimeUnit::second
- unsigned int _entitiesCreatedSoFar = 0

Additional Inherited Members

5.86.1 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

5.86.2 Constructor & Destructor Documentation

5.86.2.1 SourceModelComponent::SourceModelComponent (Model * model, std::string componentTypename)

5.86.2.2 SourceModelComponent::SourceModelComponent (const SourceModelComponent & orig)

5.86.2.3 SourceModelComponent::∼SourceModelComponent() [virtual]

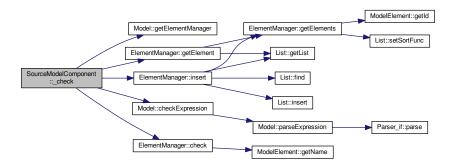
5.86.3 Member Function Documentation

5.86.3.1 bool SourceModelComponent::_check(std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

Reimplemented in Create.

Here is the call graph for this function:



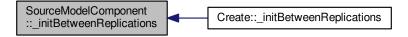


5.86.3.2 void SourceModelComponent::_initBetweenReplications() [protected], [virtual]

Implements ModelComponent.

Reimplemented in Create.

Here is the caller graph for this function:

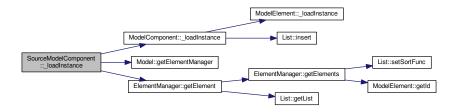


5.86.3.3 bool SourceModelComponent::_loadInstance (std::map < std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelComponent.

Reimplemented in Create.

Here is the call graph for this function:



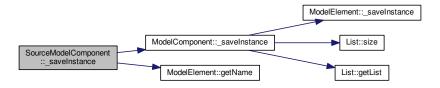


5.86.3.4 std::map < std::string, std::string > * SourceModelComponent::_saveInstance () [protected], [virtual]

Reimplemented from ModelComponent.

Reimplemented in Create.

Here is the call graph for this function:



Here is the caller graph for this function:



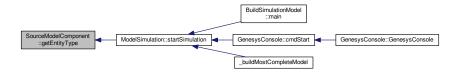
5.86.3.5 unsigned int SourceModelComponent::getEntitiesCreated () const

5.86.3.6 unsigned int SourceModelComponent::getEntitiesPerCreation () const



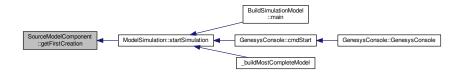
5.86.3.7 EntityType * SourceModelComponent::getEntityType () const

Here is the caller graph for this function:

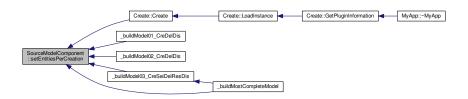


5.86.3.8 double SourceModelComponent::getFirstCreation () const

Here is the caller graph for this function:

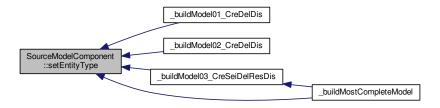


- 5.86.3.9 std::string SourceModelComponent::getMaxCreations () const
- $5.86.3.10 \quad std:: string \ Source Model Component:: get Time Between Creations Expression \ (\quad) \ const$
- 5.86.3.11 Util::TimeUnit SourceModelComponent::getTimeUnit () const
- 5.86.3.12 bool SourceModelComponent::isCollectStatistics () const
- 5.86.3.13 void SourceModelComponent::setCollectStatistics (bool _collectStatistics)
- 5.86.3.14 void SourceModelComponent::setEntitiesCreated (unsigned int _entitiesCreated)
- 5.86.3.15 void SourceModelComponent::setEntitiesPerCreation (unsigned int _entitiesPerCreation)



5.86.3.16 void SourceModelComponent::setEntityType (EntityType * _entityType)

Here is the caller graph for this function:



5.86.3.17 void SourceModelComponent::setFirstCreation (double _firstCreation)

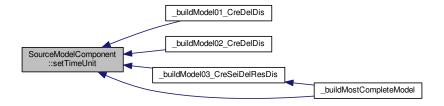
5.86.3.18 void SourceModelComponent::setMaxCreations (std::string _maxCreationsExpression)

5.86.3.19 void SourceModelComponent::setTimeBetweenCreationsExpression (std::string_timeBetweenCreations)

Here is the caller graph for this function:



5.86.3.20 void SourceModelComponent::setTimeUnit (Util::TimeUnit _timeUnit)

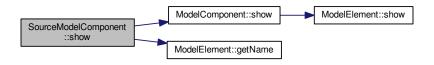


5.86.3.21 std::string SourceModelComponent::show() [virtual]

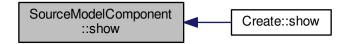
Reimplemented from ModelComponent.

Reimplemented in Create.

Here is the call graph for this function:



Here is the caller graph for this function:



5.86.4 Member Data Documentation

- **5.86.4.1** unsigned int SourceModelComponent::_entitiesCreatedSoFar = 0 [protected]
- **5.86.4.2 unsigned int SourceModelComponent::_entitiesPerCreation = 1** [protected]
- **5.86.4.3 EntityType* SourceModelComponent::_entityType** [protected]
- **5.86.4.4** double SourceModelComponent::_firstCreation = 0.0 [protected]
- 5.86.4.5 std::string SourceModelComponent::_maxCreationsExpression = std::to_string(std::numeric_limits<unsigned int>::max()) [protected]
- 5.86.4.6 std::string SourceModelComponent::_timeBetweenCreationsExpression = "EXPO(1)" [protected]
- 5.86.4.7 Util::TimeUnit SourceModelComponent::_timeBetweenCreationsTimeUnit = Util::TimeUnit::second [protected]

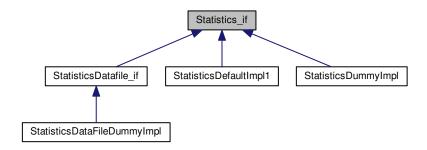
The documentation for this class was generated from the following files:

- SourceModelComponent.h
- SourceModelComponent.cpp

5.87 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 ()=0
- virtual unsigned int newSampleSize (double halfWidth)=0
- virtual double getConfidenceLevel ()=0
- virtual void setConfidenceLevel (double confidencelevel)=0

5.87.1 Detailed Description

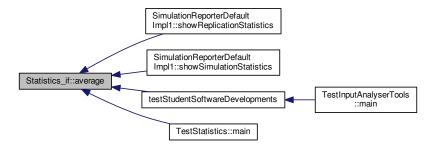
Interface for statisct synthesis of a stochastic variable collected by a Collector_if. The statistics generated may be updated based only on the previous statistics and the single newest added value or they may be updated based on a datafile, depending on the Collector implementation.

5.87.2 Member Function Documentation

5.87.2.1 virtual double Statistics_if::average () [pure virtual]

Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl.

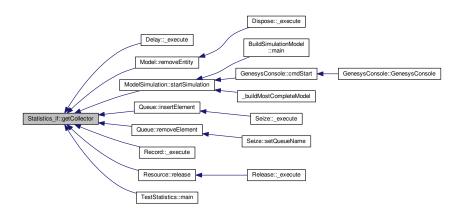
Here is the caller graph for this function:



5.87.2.2 virtual Collector_if* Statistics_if::getCollector() [pure virtual]

Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl.

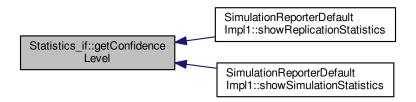
Here is the caller graph for this function:



5.87.2.3 virtual double Statistics_if::getConfidenceLevel() [pure virtual]

Implemented in StatisticsDefaultImpl1, and StatisticsDummyImpl.

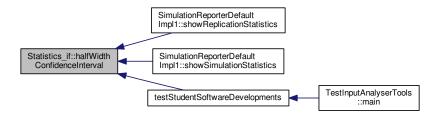
Here is the caller graph for this function:



5.87.2.4 virtual double Statistics_if::halfWidthConfidenceInterval() [pure virtual]

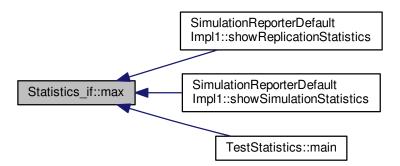
Implemented in StatisticsDefaultImpl1, and StatisticsDummyImpl.

Here is the caller graph for this function:



5.87.2.5 virtual double Statistics_if::max() [pure virtual]

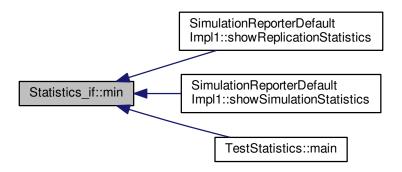
Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl. Here is the caller graph for this function:



```
5.87.2.6 virtual double Statistics_if::min() [pure virtual]
```

Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl.

Here is the caller graph for this function:

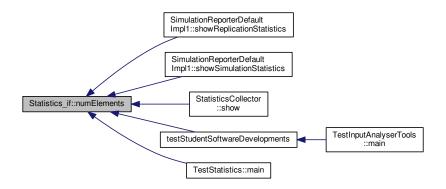


5.87.2.7 virtual unsigned int Statistics_if::newSampleSize (double halfWidth) [pure virtual]

Implemented in StatisticsDefaultImpl1, and StatisticsDummyImpl.

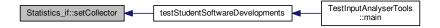
5.87.2.8 virtual unsigned int Statistics_if::numElements() [pure virtual]

Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl.



5.87.2.9 virtual void Statistics_if::setCollector (Collector_if * collector) [pure virtual]

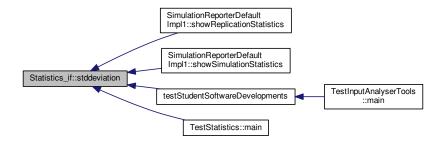
Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl. Here is the caller graph for this function:



5.87.2.10 virtual void Statistics_if::setConfidenceLevel (double confidencelevel) [pure virtual] Implemented in StatisticsDefaultImpl1, and StatisticsDummyImpl.

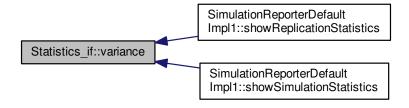
5.87.2.11 virtual double Statistics_if::stddeviation() [pure virtual]

Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl. Here is the caller graph for this function:



5.87.2.12 virtual double Statistics_if::variance() [pure virtual]

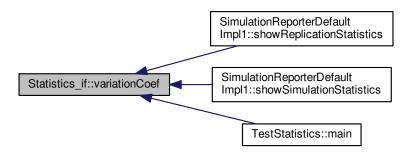
Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl. Here is the caller graph for this function:



```
5.87.2.13 virtual double Statistics_if::variationCoef() [pure virtual]
```

Implemented in StatisticsDefaultImpl1, StatisticsDummyImpl, and StatisticsDataFileDummyImpl.

Here is the caller graph for this function:



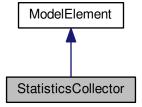
The documentation for this class was generated from the following file:

· Statistics_if.h

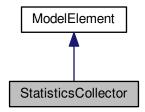
5.88 StatisticsCollector Class Reference

```
#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

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelElement * LoadInstance (ElementManager *elems, std::map< std::string, std::string > *fields)

Protected Member Functions

- virtual bool loadInstance (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>check</u> (std::string *errorMessage)

Additional Inherited Members

5.88.1 Constructor & Destructor Documentation

5.88.1.1 StatisticsCollector::StatisticsCollector()



- 5.88.1.2 StatisticsCollector::StatisticsCollector (std::string name)
- 5.88.1.3 StatisticsCollector::StatisticsCollector (std::string name, ModelElement * parent)
- 5.88.1.4 StatisticsCollector::StatisticsCollector (const StatisticsCollector & orig)
- **5.88.1.5 StatisticsCollector::**~StatisticsCollector() [virtual]

Here is the caller graph for this function:



5.88.2 Member Function Documentation

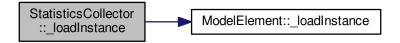
5.88.2.1 bool StatisticsCollector::_check(std::string * errorMessage) [protected], [virtual]

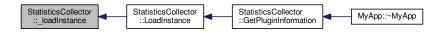
Reimplemented from ModelElement.

5.88.2.2 bool StatisticsCollector::_loadInstance (std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:

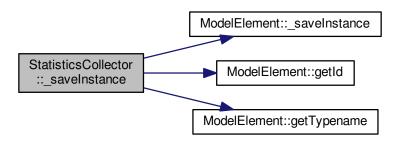




5.88.2.3 std::map< std::string, std::string > * StatisticsCollector::_saveInstance() [protected], [virtual]

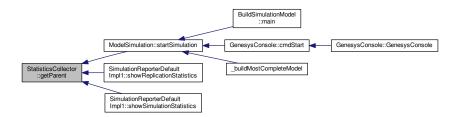
Reimplemented from ModelElement.

Here is the call graph for this function:

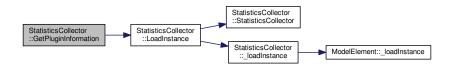


5.88.2.4 ModelElement * StatisticsCollector::getParent () const

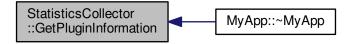
Here is the caller graph for this function:



5.88.2.5 PluginInformation * StatisticsCollector::GetPluginInformation() [static]

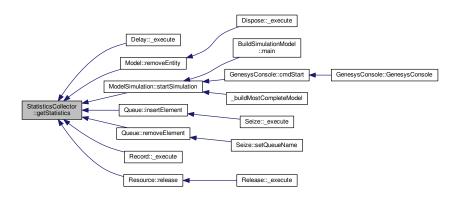


Here is the caller graph for this function:



5.88.2.6 Statistics if * StatisticsCollector::getStatistics () const

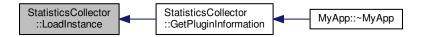
Here is the caller graph for this function:



5.88.2.7 ModelElement * StatisticsCollector::LoadInstance (ElementManager * elems, std::map< std::string, std::string > * fields) [static]



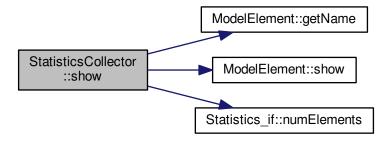
Here is the caller graph for this function:



5.88.2.8 std::string StatisticsCollector::show() [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



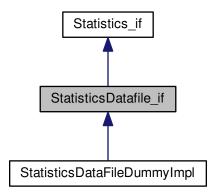
The documentation for this class was generated from the following files:

- StatisticsCollector.h
- StatisticsCollector.cpp

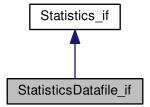
5.89 StatisticsDatafile_if Class Reference

#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

5.89.1 Member Function Documentation

5.89.1.1 virtual double StatisticsDatafile_if::centil (unsigned short num) [pure virtual]

Implemented in StatisticsDataFileDummyImpl.

```
5.89.1.2 virtual double StatisticsDatafile_if::decil(unsigned short num) [pure virtual]
Implemented in StatisticsDataFileDummyImpl.
5.89.1.3 virtual unsigned int StatisticsDatafile_if::histogramClassFrequency (unsigned short classNum) [pure
        virtuall
Implemented in StatisticsDataFileDummyImpl.
5.89.1.4 virtual double StatisticsDatafile_if::histogramClassLowerLimit (unsigned short classNum) [pure virtual]
Implemented in StatisticsDataFileDummyImpl.
5.89.1.5 virtual unsigned short StatisticsDatafile if::histogramNumClasses() [pure virtual]
Implemented in StatisticsDataFileDummyImpl.
5.89.1.6 virtual double StatisticsDatafile_if::mediane() [pure virtual]
Implemented in StatisticsDataFileDummyImpl.
5.89.1.7 virtual double StatisticsDatafile_if::mode( ) [pure virtual]
Implemented in StatisticsDataFileDummyImpl.
5.89.1.8 virtual double StatisticsDatafile_if::quartil ( unsigned short num ) [pure virtual]
Implemented in StatisticsDataFileDummyImpl.
5.89.1.9 virtual void StatisticsDatafile_if::setHistogramNumClasses (unsigned short num) [pure virtual]
Implemented in StatisticsDataFileDummyImpl.
The documentation for this class was generated from the following file:
```

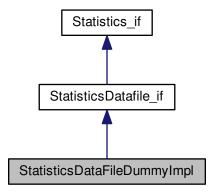
Generated by Doxygen

StatisticsDataFile_if.h

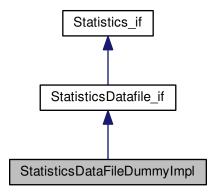
5.90 StatisticsDataFileDummyImpl Class Reference

#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)

5.90.1 Constructor & Destructor Documentation

```
5.90.1.1 StatisticsDataFileDummyImpl::StatisticsDataFileDummyImpl ( )
5.90.1.2 StatisticsDataFileDummyImpl::StatisticsDataFileDummyImpl ( const StatisticsDataFileDummyImpl & orig )
5.90.1.3 StatisticsDataFileDummyImpl::~StatisticsDataFileDummyImpl ( ) [virtual]
5.90.2 Member Function Documentation
5.90.2.1 double StatisticsDataFileDummyImpl::average ( ) [virtual]
Implements Statistics_if.
5.90.2.2 double StatisticsDataFileDummyImpl::centil ( unsigned short num ) [virtual]
Implements StatisticsDataFileDummyImpl::decil ( unsigned short num ) [virtual]
Implements StatisticsDataFileDummyImpl::decil ( unsigned short num ) [virtual]
Implements StatisticsDataFileDummyImpl::getCollector( ) [virtual]
Implements Statistics_if.
```

```
5.90.2.5 double StatisticsDataFileDummyImpl::halfWidthConfidenceInterval ( double confidenceIevel ) [virtual]
5.90.2.6 unsigned int StatisticsDataFileDummyImpl::histogramClassFrequency (unsigned short classNum ) [virtual]
Implements StatisticsDatafile_if.
5.90.2.7 double StatisticsDataFileDummyImpl::histogramClassLowerLimit (unsigned short classNum ) [virtual]
Implements Statistics Datafile if.
5.90.2.8 unsigned short StatisticsDataFileDummyImpl::histogramNumClasses() [virtual]
Implements StatisticsDatafile_if.
5.90.2.9 double StatisticsDataFileDummyImpl::max() [virtual]
Implements Statistics_if.
5.90.2.10 double StatisticsDataFileDummyImpl::mediane() [virtual]
Implements StatisticsDatafile if.
5.90.2.11 double StatisticsDataFileDummyImpl::min() [virtual]
Implements Statistics_if.
5.90.2.12 double StatisticsDataFileDummyImpl::mode( ) [virtual]
Implements Statistics Datafile if.
5.90.2.13 unsigned int StatisticsDataFileDummyImpl::newSampleSize ( double confidencelevel, double halfWidth )
          [virtual]
5.90.2.14 unsigned int StatisticsDataFileDummyImpl::numElements() [virtual]
Implements Statistics if.
5.90.2.15 double StatisticsDataFileDummyImpl::quartil (unsigned short num) [virtual]
Implements StatisticsDatafile_if.
```

```
5.90.2.16 void StatisticsDataFileDummyImpl::setCollector( Collector_if * collector) [virtual]
Implements Statistics_if.
5.90.2.17 void StatisticsDataFileDummyImpl::setHistogramNumClasses( unsigned short num ) [virtual]
Implements StatisticsDataFileDummyImpl::stddeviation() [virtual]
Implements Statistics_if.
5.90.2.19 double StatisticsDataFileDummyImpl::variance() [virtual]
Implements Statistics_if.
5.90.2.20 double StatisticsDataFileDummyImpl::variationCoef() [virtual]
Implements Statistics_if.
```

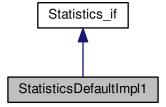
The documentation for this class was generated from the following files:

- StatisticsDataFileDummyImpl.h
- StatisticsDataFileDummyImpl.cpp

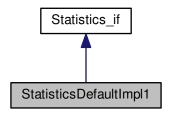
5.91 StatisticsDefaultImpl1 Class Reference

```
#include <StatisticsDefaultImpl1.h>
```

Inheritance diagram for StatisticsDefaultImpl1:



Collaboration diagram for StatisticsDefaultImpl1:



Public Member Functions

• StatisticsDefaultImpl1 ()

When constructor is invoked without a Collector, it is taken from Traits < Statistics_if>::CollectorImplementation configuration.

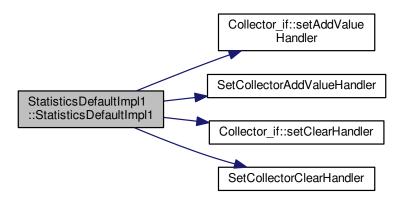
- StatisticsDefaultImpl1 (Collector_if *collector)
- StatisticsDefaultImpl1 (const StatisticsDefaultImpl1 &orig)
- virtual ~StatisticsDefaultImpl1 ()
- virtual Collector_if * getCollector ()
- virtual void setCollector (Collector_if *collector)
- virtual unsigned int numElements ()
- virtual double min ()
- · virtual double max ()
- virtual double average ()
- virtual double variance ()
- · virtual double stddeviation ()
- virtual double variationCoef ()
- · virtual double halfWidthConfidenceInterval ()
- virtual unsigned int newSampleSize (double halfWidth)
- · virtual double getConfidenceLevel ()
- virtual void setConfidenceLevel (double confidencelevel)

5.91.1 Constructor & Destructor Documentation

5.91.1.1 StatisticsDefaultImpl1::StatisticsDefaultImpl1 ()

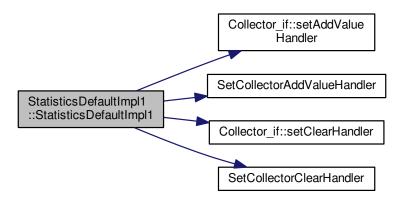
When constructor is invoked without a Collector, it is taken from Traits < Statistics_if>::CollectorImplementation configuration.

Here is the call graph for this function:



5.91.1.2 StatisticsDefaultImpl1::StatisticsDefaultImpl1 (Collector_if * collector)

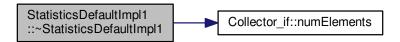
Here is the call graph for this function:



5.91.1.3 StatisticsDefaultImpl1::StatisticsDefaultImpl1 (const StatisticsDefaultImpl1 & orig)

```
5.91.1.4 StatisticsDefaultImpl1::~StatisticsDefaultImpl1() [virtual]
```

Here is the call graph for this function:



5.91.2 Member Function Documentation

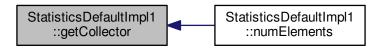
```
5.91.2.1 double StatisticsDefaultImpl1::average() [virtual]
```

Implements Statistics_if.

```
5.91.2.2 Collector_if * StatisticsDefaultImpl1::getCollector( ) [virtual]
```

Implements Statistics_if.

Here is the caller graph for this function:



```
\textbf{5.91.2.3} \quad \textbf{double StatisticsDefaultImpl1::getConfidenceLevel ( )} \quad [\texttt{virtual}]
```

Implements Statistics_if.

5.91.2.4 double StatisticsDefaultImpl1::halfWidthConfidenceInterval() [virtual]

Implements Statistics_if.

```
Implements Statistics_if.

5.91.2.6 double StatisticsDefaultImpl1::min() [virtual]

Implements Statistics_if.

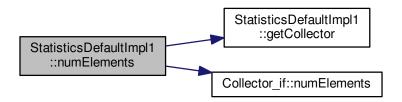
5.91.2.7 unsigned int StatisticsDefaultImpl1::newSampleSize(double halfWidth) [virtual]

Implements Statistics_if.

5.91.2.8 unsigned int StatisticsDefaultImpl1::numElements() [virtual]

Implements Statistics_if.

Here is the call graph for this function:
```



```
5.91.2.9 void StatisticsDefaultImpl1::setCollector ( Collector_if * collector ) [virtual]
Implements Statistics_if.

5.91.2.10 void StatisticsDefaultImpl1::setConfidenceLevel ( double confidencelevel ) [virtual]
Implements Statistics_if.

5.91.2.11 double StatisticsDefaultImpl1::stddeviation ( ) [virtual]
Implements Statistics_if.
```

5.91.2.12 double StatisticsDefaultImpl1::variance() [virtual]

Implements Statistics_if.

5.91.2.13 double StatisticsDefaultImpl1::variationCoef() [virtual]

Implements Statistics_if.

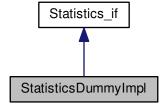
The documentation for this class was generated from the following files:

- StatisticsDefaultImpl1.h
- StatisticsDefaultImpl1.cpp

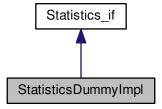
5.92 StatisticsDummyImpl Class Reference

#include <StatisticsDummyImpl.h>

Inheritance diagram for StatisticsDummyImpl:



Collaboration diagram for StatisticsDummyImpl:



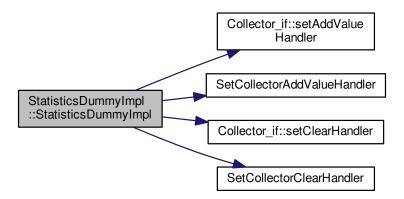
Public Member Functions

- 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 ()
- virtual unsigned int newSampleSize (double halfWidth)
- virtual double getConfidenceLevel ()
- virtual void setConfidenceLevel (double confidencelevel)

5.92.1 Constructor & Destructor Documentation

5.92.1.1 StatisticsDummyImpl::StatisticsDummyImpl ()

Here is the call graph for this function:



- $5.92.1.2 \quad \textbf{StatisticsDummyImpl}:: \textbf{StatisticsDummyImpl} \ (\ \textbf{const} \ \textbf{StatisticsDummyImpl} \ \& \ \textbf{\textit{orig}} \)$
- **5.92.1.3 StatisticsDummyImpl::**~StatisticsDummyImpl() [virtual]

5.92.2 Member Function Documentation

5.92.2.1 double StatisticsDummyImpl::average() [virtual]

Implements Statistics_if.

```
5.92.2.2 Collector_if * StatisticsDummyImpl::getCollector() [virtual]
Implements Statistics_if.
5.92.2.3 double StatisticsDummyImpl::getConfidenceLevel( ) [virtual]
Implements Statistics_if.
5.92.2.4 double StatisticsDummyImpl::halfWidthConfidenceInterval() [virtual]
Implements Statistics_if.
5.92.2.5 double StatisticsDummyImpl::max() [virtual]
Implements Statistics_if.
5.92.2.6 double StatisticsDummyImpl::min() [virtual]
Implements Statistics_if.
5.92.2.7 unsigned int StatisticsDummyImpl::newSampleSize ( double halfWidth ) [virtual]
Implements Statistics_if.
5.92.2.8 unsigned int StatisticsDummyImpl::numElements ( ) [virtual]
Implements Statistics_if.
5.92.2.9 void StatisticsDummyImpl::setCollector ( Collector_if * collector ) [virtual]
Implements Statistics_if.
5.92.2.10 void StatisticsDummyImpl::setConfidenceLevel ( double confidencelevel ) [virtual]
Implements Statistics_if.
5.92.2.11 double StatisticsDummyImpl::stddeviation() [virtual]
Implements Statistics_if.
```

5.92.2.12 double StatisticsDummyImpl::variance() [virtual]

Implements Statistics_if.

5.92.2.13 double StatisticsDummyImpl::variationCoef() [virtual]

Implements Statistics_if.

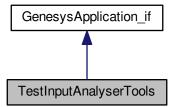
The documentation for this class was generated from the following files:

- StatisticsDummyImpl.h
- StatisticsDummyImpl.cpp

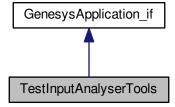
5.93 TestInputAnalyserTools Class Reference

#include <TestInputAnalyserTools.h>

Inheritance diagram for TestInputAnalyserTools:



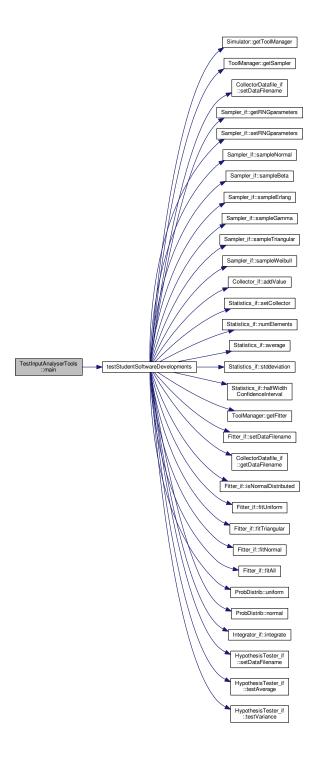
Collaboration diagram for TestInputAnalyserTools:



55 1		Class Boodinentati
Public Member Functi		
TestInputAnalyseint main (int argc		
5.93.1 Constructor 8	& Destructor Documentation	
5.93.1.1 TestInputAnalys	serTools::TestInputAnalyserTools()	
5.93.2 Member Fund	etion Documentation	
5.93.2.1 int TestInputAna	alyserTools::main (int <i>argc,</i> char ** <i>argv</i>) [virtual]	

Implements GenesysApplication_if.

Here is the call graph for this function:



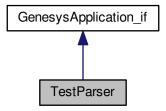
The documentation for this class was generated from the following files:

- TestInputAnalyserTools.h
- TestInputAnalyserTools.cpp

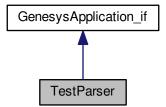
5.94 TestParser Class Reference

#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)

5.94.1 Constructor & Destructor Documentation

5.94.1.1 TestParser::TestParser()

5.94.1.2 TestParser::TestParser (const TestParser & orig)

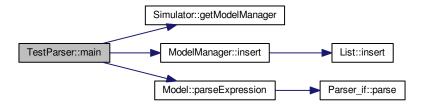
5.94.1.3 TestParser::∼**TestParser()** [virtual]

5.94.2 Member Function Documentation

5.94.2.1 int TestParser::main (int argc, char ** argv) [virtual]

Implements GenesysApplication_if.

Here is the call graph for this function:



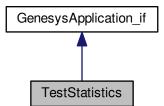
The documentation for this class was generated from the following files:

- · TestParser.h
- TestParser.cpp

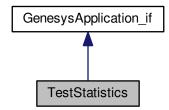
5.95 TestStatistics Class Reference

#include <TestStatistics.h>

Inheritance diagram for TestStatistics:



Collaboration diagram for TestStatistics:



Public Member Functions

- TestStatistics ()
- int main (int argc, char **argv)

5.95.1 Constructor & Destructor Documentation

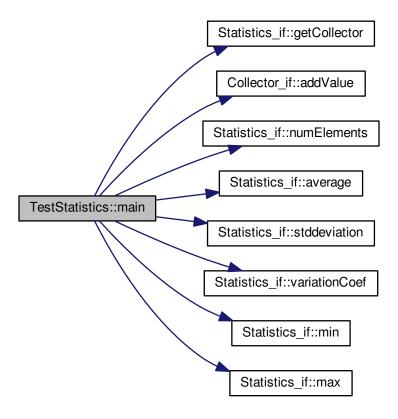
5.95.1.1 TestStatistics::TestStatistics ()

5.95.2 Member Function Documentation

5.95.2.1 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

5.96 ToolManager Class Reference

#include <ToolManager.h>

Public Member Functions

- ToolManager (Simulator *_simulator)
- ToolManager (const ToolManager &orig)
- virtual ∼ToolManager ()
- 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.

5.96.1 Constructor & Destructor Documentation

```
5.96.1.1 ToolManager::ToolManager ( Simulator * _simulator )
```

5.96.1.2 ToolManager::ToolManager (const ToolManager & orig)

```
5.96.1.3 ToolManager::~ToolManager( ) [virtual]
```

5.96.2 Member Function Documentation

```
5.96.2.1 Fitter if * ToolManager::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:



5.96.2.2 Sampler_if * ToolManager::getSampler () const

Returns the Sampler, used to generate samples accordingly to a probability distribution.

Here is the caller graph for this function:



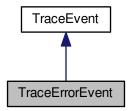
The documentation for this class was generated from the following files:

- · ToolManager.h
- ToolManager.cpp

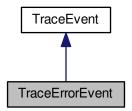
5.97 TraceErrorEvent Class Reference

#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

5.97.1 Constructor & Destructor Documentation

5.97.1.1 TraceErrorEvent::TraceErrorEvent (std::string *text*, std::exception *e*) [inline]

5.97.2 Member Function Documentation

5.97.2.1 std::exception TraceErrorEvent::getException () const [inline]

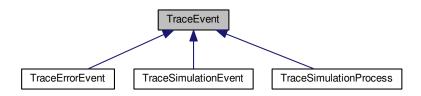
The documentation for this class was generated from the following file:

TraceManager.h

5.98 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

5.98.1 Constructor & Destructor Documentation

5.98.1.1 TraceEvent::TraceEvent (Util::TraceLevel tracelevel, std::string text) [inline]

5.98.2 Member Function Documentation

5.98.2.1 std::string TraceEvent::getText() const [inline]

Here is the caller graph for this function:



5.98.2.2 Util::TraceLevel TraceEvent::getTracelevel() const [inline]

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

TraceManager.h

5.99 TraceManager Class Reference

#include <TraceManager.h>

Public Member Functions

- TraceManager (Simulator *simulator)
- 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
- Simulator * getSimulator () const

5.99.1 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.

5.99.2 Constructor & Destructor Documentation

- 5.99.2.1 TraceManager::TraceManager (Simulator * simulator)
- 5.99.2.2 TraceManager::TraceManager (const TraceManager & orig)
- **5.99.2.3** TraceManager::~TraceManager() [virtual]

5.99.3 Member Function Documentation

- 5.99.3.1 void TraceManager::addTraceErrorHandler (traceErrorListener traceErrorListener)
- 5.99.3.2 void TraceManager::addTraceHandler (traceListener traceListener)



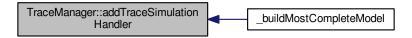
5.99.3.3 void TraceManager::addTraceReportHandler (traceListener traceReportListener)

Here is the caller graph for this function:



5.99.3.4 void TraceManager::addTraceSimulationHandler (traceSimulationListener traceSimulationListener)

Here is the caller graph for this function:



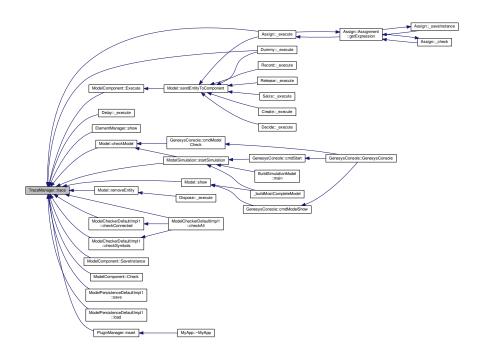
- $\textbf{5.99.3.5} \quad \textbf{List}{<} \, \textbf{std::string} > * \, \textbf{TraceManager::getErrorMessages} \, (\ \) \, \textbf{const}$
- 5.99.3.6 Simulator * TraceManager::getSimulator () const
- 5.99.3.7 Util::TraceLevel TraceManager::getTraceLevel () const
- 5.99.3.8 void TraceManager::setTraceLevel (Util::TraceLevel _traceLevel)



5.99.3.9 void TraceManager::trace (Util::TraceLevel tracelevel, std::string text)

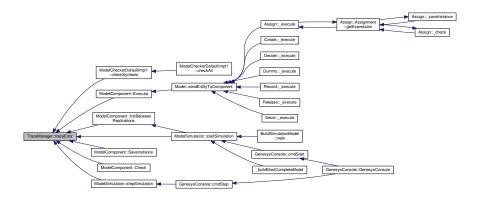
Here is the call graph for this function:





5.99.3.10 void TraceManager::traceError (std::exception e, std::string text)

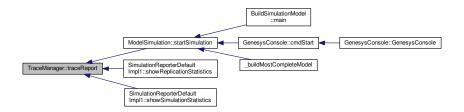
Here is the caller graph for this function:



5.99.3.11 void TraceManager::traceReport (Util::TraceLevel tracelevel, std::string text)

Here is the call graph for this function:



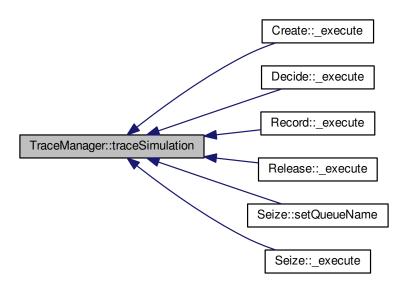


5.99.3.12 void TraceManager::traceSimulation (Util::TraceLevel tracelevel, double time, Entity * entity, ModelComponent * component, std::string text)

Here is the call graph for this function:



Here is the caller graph for this function:



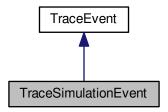
The documentation for this class was generated from the following files:

- TraceManager.h
- TraceManager.cpp

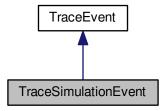
5.100 TraceSimulationEvent Class Reference

#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)

5.100.1 Constructor & Destructor Documentation

- 5.100.1.1 TraceSimulationEvent::TraceSimulationEvent (Util::TraceLevel tracelevel, double time, Entity * entity, ModelComponent * component, std::string text) [inline]
- 5.100.2 Member Function Documentation
- **5.100.2.1 ModelComponent*** TraceSimulationEvent::getComponent() const [inline]
- 5.100.2.2 Entity* TraceSimulationEvent::getEntity () const [inline]
- **5.100.2.3** double TraceSimulationEvent::getTime () const [inline]

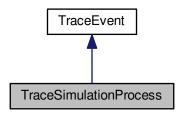
The documentation for this class was generated from the following file:

• TraceManager.h

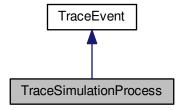
5.101 TraceSimulationProcess Class Reference

#include <TraceManager.h>

Inheritance diagram for TraceSimulationProcess:



Collaboration diagram for TraceSimulationProcess:



Public Member Functions

• TraceSimulationProcess (Util::TraceLevel tracelevel, std::string text)

5.101.1 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

5.101.2 Constructor & Destructor Documentation

5.101.2.1 TraceSimulationProcess::TraceSimulationProcess (Util::TraceLevel tracelevel, std::string text) [inline]

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

• TraceManager.h

5.102 Traits < T > Struct Template Reference

```
#include <Traits.h>
```

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

· Traits.h

5.103 Traits < Collector_if > Struct Template Reference

```
#include <Traits.h>
```

Public Types

• typedef CollectorDatafileDefaultImpl1 Implementation

5.103.1 Member Typedef Documentation

5.103.1.1 typedef Collector Datafile Default Impl Traits < Collector if >:: Implementation

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

• Traits.h

5.104 Traits < ExperimentDesign_if > Struct Template Reference

```
#include <Traits.h>
```

Public Types

• typedef ExperimentDesignDefaultImpl1 Implementation

5.104.1 Member Typedef Documentation

5.104.1.1 typedef ExperimentDesignDefaultImpl1 Traits < ExperimentDesign_if >::Implementation

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

Traits.h

5.105 Traits < Fitter_if > Struct Template Reference

#include <Traits.h>

Public Types

• typedef FitterDefaultImpl1 Implementation

5.105.1 Member Typedef Documentation

5.105.1.1 typedef FitterDefaultImpl1 Traits< Fitter_if >::Implementation

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

· Traits.h

5.106 Traits < GenesysApplication_if > Struct Template Reference

#include <Traits.h>

Public Types

· typedef MyApp Application

5.106.1 Member Typedef Documentation

 $5.106.1.1 \quad type def \ My App \ Traits < Genesys Application _if > :: Application$

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

· Traits.h

5.107 Traits < HypothesisTester_if > Struct Template Reference

#include <Traits.h>

Public Types

- · typedef IntegratorDefaultImpl1 IntegratorImplementation
- typedef HypothesisTesterDefaultImpl1 Implementation

5.107.1 Member Typedef Documentation

5.107.1.1 typedef HypothesisTesterDefaultImpl1 Traits < HypothesisTester_if >::Implementation

5.107.1.2 typedef IntegratorDefaultImpl1 Traits < HypothesisTester_if >::IntegratorImplementation

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

· Traits.h

5.108 Traits < Integrator_if > Struct Template Reference

```
#include <Traits.h>
```

Public Types

• typedef IntegratorDefaultImpl1 Implementation

Static Public Attributes

- static constexpr unsigned int MaxIterations = 1e3
- static constexpr double Precision = 1e-9

5.108.1 Member Typedef Documentation

 $5.108.1.1 \quad type def \ Integrator Default Impl 1 \ Traits < Integrator _if > :: Implementation$

5.108.2 Member Data Documentation

5.108.2.1 constexpr unsigned int Traits< Integrator_if >::MaxIterations = 1e3 [static]

5.108.2.2 constexpr double Traits < Integrator_if >::Precision = 1e-9 [static]

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

· Traits.h

5.109 Traits < Model > Struct Template Reference

#include <Traits.h>

Static Public Attributes

- static const bool debugged = true
- static const Util::TraceLevel traceLevel = Util::TraceLevel::blockArrival

5.109.1 Member Data Documentation

```
5.109.1.1 const bool Traits < Model >::debugged = true [static]
```

5.109.1.2 const Util::TraceLevel Traits < Model >::traceLevel = Util::TraceLevel::blockArrival [static]

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

· Traits.h

5.110 Traits < ModelChecker_if > Struct Template Reference

```
#include <Traits.h>
```

Public Types

• typedef ModelCheckerDefaultImpl1 Implementation

5.110.1 Member Typedef Documentation

5.110.1.1 typedef ModelCheckerDefaultImpl1 Traits < ModelChecker_if >::Implementation

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

Traits.h

5.111 Traits < ModelComponent > Struct Template Reference

```
#include <Traits.h>
```

Public Types

- typedef StatisticsDefaultImpl1 StatisticsCollector_StatisticsImplementation
- typedef CollectorDefaultImpl1 StatisticsCollector_CollectorImplementation

5.111.1 Member Typedef Documentation

5.111.1.1 typedef CollectorDefaultImpl1 Traits < ModelComponent >::StatisticsCollector_Collector ← Implementation

5.111.1.2 typedef StatisticsDefaultImpl1 Traits < ModelComponent >::StatisticsCollector_Statistics ← Implementation

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

· Traits.h

5.112 Traits < ModelPersistence_if > Struct Template Reference

```
#include <Traits.h>
```

Public Types

typedef ModelPersistenceDefaultImpl1 Implementation

5.112.1 Member Typedef Documentation

5.112.1.1 typedef ModelPersistenceDefaultImpl1 Traits < ModelPersistence_if >::Implementation

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

• Traits.h

5.113 Traits < Parser_if > Struct Template Reference

```
#include <Traits.h>
```

Public Types

• typedef ParserDefaultImpl2 Implementation

5.113.1 Member Typedef Documentation

5.113.1.1 typedef ParserDefaultImpl2 Traits < Parser if >::Implementation

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

· Traits.h

5.114 Traits < ProcessAnalyser_if > Struct Template Reference

#include <Traits.h>

Public Types

- typedef ProcessAnalyserDefaultImpl1 Implementation
- 5.114.1 Member Typedef Documentation
- 5.114.1.1 typedef ProcessAnalyserDefaultImpl1 Traits < ProcessAnalyser if >::Implementation

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

· Traits.h

5.115 Traits < Sampler_if > Struct Template Reference

#include <Traits.h>

Public Types

- typedef SamplerDefaultImpl1 Implementation
- typedef SamplerDefaultImpl1::DefaultImpl1RNG_Parameters Parameters
- 5.115.1 Member Typedef Documentation
- 5.115.1.1 typedef SamplerDefaultImpl1 Traits < Sampler_if >::Implementation
- $5.115.1.2 \quad type def \ Sampler Default Impl 1:: Default Impl 1RNG_Parameters \ Traits < Sampler_if >:: Parameters \ Traits < Sampler_if >: Parameters \ Traits < Sam$

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

· Traits.h

5.116 Traits < SimulationReporter_if > Struct Template Reference

#include <Traits.h>

Public Types

• typedef SimulationReporterDefaultImpl1 Implementation

5.116.1 Member Typedef Documentation

5.116.1.1 typedef SimulationReporterDefaultImpl1 Traits < SimulationReporter_if >::Implementation

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

· Traits.h

5.117 Traits < Statistics_if > Struct Template Reference

```
#include <Traits.h>
```

Public Types

- · typedef StatisticsDefaultImpl1 Implementation
- typedef CollectorDefaultImpl1 CollectorImplementation

Static Public Attributes

• static constexpr double SignificanceLevel = 0.05

5.117.1 Member Typedef Documentation

- 5.117.1.1 typedef CollectorDefaultImpl1 Traits < Statistics_if >::CollectorImplementation
- 5.117.1.2 typedef StatisticsDefaultImpl1 Traits < Statistics_if >::Implementation
- 5.117.2 Member Data Documentation
- **5.117.2.1** constexpr double Traits < Statistics_if >::SignificanceLevel = 0.05 [static]

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

· Traits.h

5.118 Util Class Reference

#include <Util.h>

5.118 Util Class Reference 377

Public Types

```
enum TimeUnit : int {
  TimeUnit::picosecond = 1, TimeUnit::nanosecond = 2, TimeUnit::microsecond = 3, TimeUnit::milisecond = 4,
  TimeUnit::second = 5, TimeUnit::minute = 6, TimeUnit::hour = 7, TimeUnit::day = 8,
  TimeUnit::week = 9 }
enum TraceLevel : int {
  TraceLevel::noTraces = 0, TraceLevel::errors = 1, TraceLevel::report = 2, TraceLevel::simulation = 3,
  TraceLevel::transferOnly = 4, TraceLevel::blockArrival = 5, TraceLevel::blockInternal = 6, TraceLevel::most ←
  Detailed = 7 }
· typedef unsigned long identitifcation
```

- typedef unsigned int rank

Static Public Member Functions

- static void SetIndent (const unsigned short indent)
- static void IncIndent ()
- static void DecIndent ()
- static void SepKeyVal (std::string str, std::string *key, std::string *value)
- static std::string Indent ()
- static std::string SetW (std::string text, unsigned short width)
- static std::string StrTimeUnit (Util::TimeUnit timeUnit)
- static Util::identitifcation GenerateNewld ()
- static Util::identitifcation GenerateNewIdOfType (std::string objtype)
- static Util::identitifcation GetLastIdOfType (std::string objtype)
- static void ResetIdOfType (std::string objtype)
- static double TimeUnitConvert (Util::TimeUnit timeUnit1, Util::TimeUnit timeUnit2)
- template < class T > static std::string TypeOf ()
- template<class T >

static Util::identitifcation GenerateNewIdOfType ()

Static Public Attributes

· static unsigned int _S_indentation

5.118.1 **Member Typedef Documentation**

5.118.1.1 typedef unsigned long Util::identitifcation

5.118.1.2 typedef unsigned int Util::rank

5.118.2 Member Enumeration Documentation

5.118.2.1 enum Util::TimeUnit:int [strong]

Enumerator

picosecond nanosecond microsecond milisecond second minute hour day

week

5.118.2.2 enum Util::TraceLevel:int [strong]

Enumerator

noTraces

errors

report

simulation

transferOnly

blockArrival

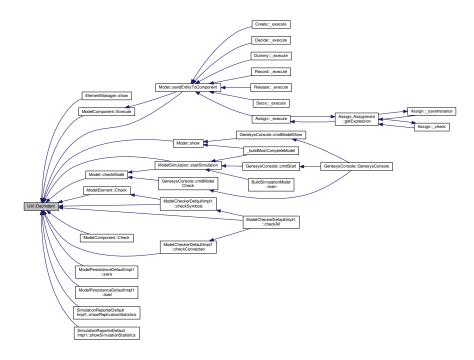
blockInternal

mostDetailed

5.118.3 Member Function Documentation

```
5.118.3.1 void Util::DecIndent() [static]
```

Here is the caller graph for this function:



5.118.3.2 Util::identitifcation Util::GenerateNewld() [static]



5.118 Util Class Reference 379

5.118.3.3 Util::identitifcation Util::GenerateNewIdOfType(std::string objtype) [static]

 $\textbf{5.118.3.4} \quad \textbf{template} < \textbf{class T} > \textbf{static Util::} \textbf{dentitifcation Util::} \textbf{GenerateNewldOfType ()} \quad \texttt{[inline], [static]}$

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:

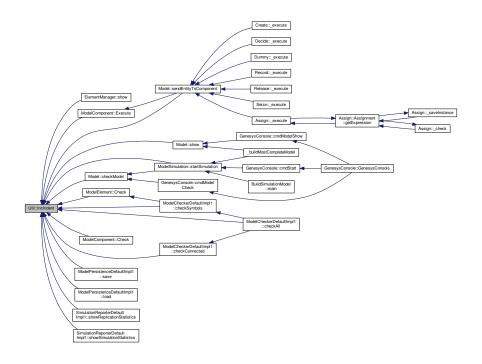


5.118.3.5 Util::identitifcation Util::GetLastIdOfType (std::string *objtype*) [static]

Here is the caller graph for this function:

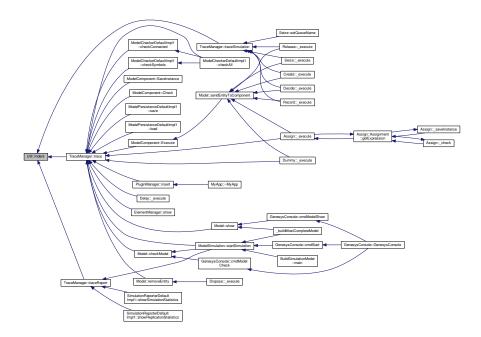


5.118.3.6 void Util::Inclndent() [static]



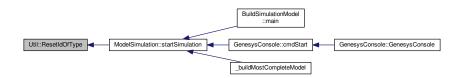
5.118.3.7 std::string Util::Indent() [static]

Here is the caller graph for this function:



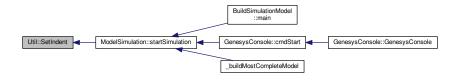
5.118.3.8 void Util::ResetIdOfType (std::string *objtype* **)** [static]

Here is the caller graph for this function:



5.118.3.9 void Util::SepKeyVal (std::string * key, std::string * value) [static]

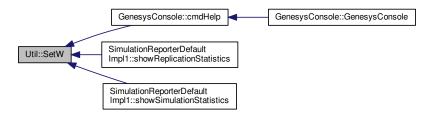
5.118.3.10 void Util::SetIndent (const unsigned short indent) [static]



5.118 Util Class Reference 381

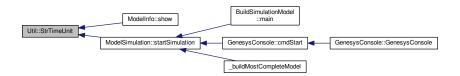
5.118.3.11 std::string Util::SetW (std::string text, unsigned short width) [static]

Here is the caller graph for this function:



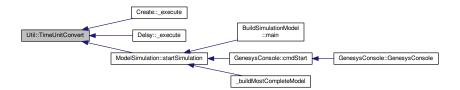
5.118.3.12 std::string Util::StrTimeUnit (Util::TimeUnit timeUnit) [static]

Here is the caller graph for this function:



5.118.3.13 double Util::TimeUnitConvert(Util::TimeUnit timeUnit1, Util::TimeUnit timeUnit2) [static]

Here is the caller graph for this function:



5.118.3.14 template < class T > static std::string Util::TypeOf() [inline], [static]

Return the name of the class used as T.

5.118.4 Member Data Documentation

5.118.4.1 unsigned int Util::_S_indentation [static]

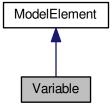
The documentation for this class was generated from the following files:

- Util.h
- Util.cpp

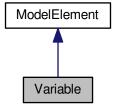
5.119 Variable Class Reference

#include <Variable.h>

Inheritance diagram for Variable:



Collaboration diagram for Variable:



Public Member Functions

- Variable ()
- Variable (std::string name)
- Variable (const Variable &orig)
- virtual \sim Variable ()
- virtual std::string show ()
- double getValue ()
- double getValue (std::string index)
- void setValue (double value)
- void setValue (std::string index, double value)

Static Public Member Functions

- static PluginInformation * GetPluginInformation ()
- static ModelElement * LoadInstance (ElementManager *elems, std::map< std::string, std::string > *fields)

Protected Member Functions

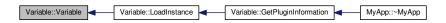
- virtual bool _loadInstance (std::map< std::string, std::string > *fields)
- virtual std::map< std::string, std::string > * _saveInstance ()
- virtual bool <u>_check</u> (std::string *errorMessage)

Additional Inherited Members

5.119.1 Constructor & Destructor Documentation

```
5.119.1.1 Variable::Variable ( )
```

Here is the caller graph for this function:



- 5.119.1.2 Variable::Variable (std::string name)
- 5.119.1.3 Variable::Variable (const Variable & orig)
- 5.119.1.4 Variable:: \sim Variable() [virtual]

5.119.2 Member Function Documentation

5.119.2.1 bool Variable::_check (std::string * errorMessage) [protected], [virtual]

Reimplemented from ModelElement.

5.119.2.2 bool Variable::_loadInstance (std::map< std::string, std::string > * fields) [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



Here is the caller graph for this function:



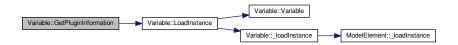
5.119.2.3 std::map< std::string, std::string > * Variable::_saveInstance() [protected], [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



5.119.2.4 PluginInformation * Variable::GetPluginInformation () [static]

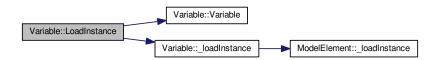


Here is the caller graph for this function:



- 5.119.2.5 double Variable::getValue ()
- 5.119.2.6 double Variable::getValue (std::string index)
- 5.119.2.7 ModelElement * Variable::LoadInstance (ElementManager * elems, std::map< std::string, std::string > * fields) [static]

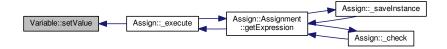
Here is the call graph for this function:



Here is the caller graph for this function:



5.119.2.8 void Variable::setValue (double value)



5.119.2.9 void Variable::setValue (std::string index, double value)

5.119.2.10 std::string Variable::show() [virtual]

Reimplemented from ModelElement.

Here is the call graph for this function:



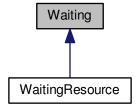
The documentation for this class was generated from the following files:

- · Variable.h
- · Variable.cpp

5.120 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

5.120.1 Constructor & Destructor Documentation

5.120.1.1 Waiting::Waiting (Entity * entity, ModelComponent * component, double timeStartedWaiting)

5.120.1.2 Waiting::Waiting (const Waiting & orig)

5.120.1.3 Waiting::~Waiting() [virtual]

5.120.2 Member Function Documentation

5.120.2.1 ModelComponent * Waiting::getComponent () const

5.120.2.2 Entity * Waiting::getEntity () const

Here is the caller graph for this function:



5.120.2.3 double Waiting::getTimeStartedWaiting () const

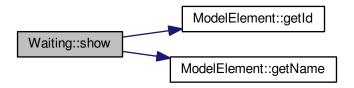
Here is the caller graph for this function:



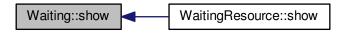
5.120.2.4 std::string Waiting::show() [virtual]

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

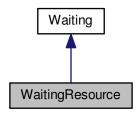
5.121 WaitingResource Class Reference

#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

5.121.1 Constructor & Destructor Documentation

- 5.121.1.1 WaitingResource::WaitingResource (Entity * entity, ModelComponent * component, double timeStartedWaiting, unsigned int quantity)
- 5.121.1.2 WaitingResource::WaitingResource (const WaitingResource & orig)
- **5.121.1.3** WaitingResource::~WaitingResource() [virtual]

5.121.2 Member Function Documentation

- 5.121.2.1 unsigned int WaitingResource::getQuantity () const
- **5.121.2.2** 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

Chapter 6

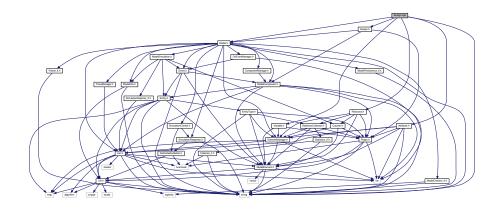
File Documentation

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:

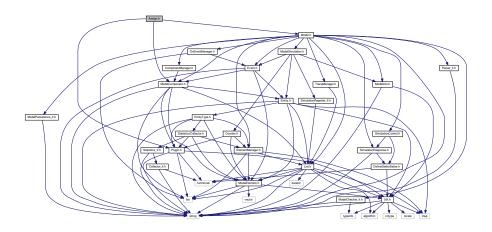


6.3 Assign.h File Reference

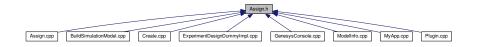
```
#include "ModelComponent.h"
#include "Model.h"
#include "Plugin.h"
```

392 File Documentation

Include dependency graph for Assign.h:



This graph shows which files directly or indirectly include this file:

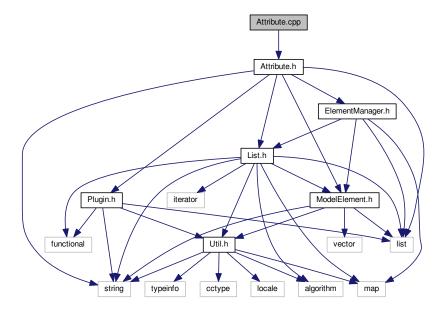


Classes

- class Assign
- class Assign::Assignment

6.4 Attribute.cpp File Reference

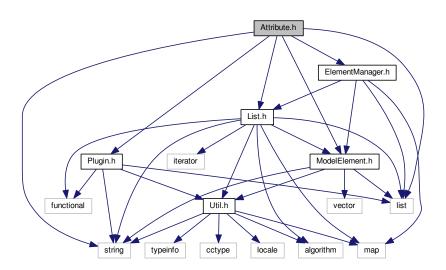
Include dependency graph for Attribute.cpp:



6.5 Attribute.h File Reference

```
#include <string>
#include <list>
#include "List.h"
#include "ModelElement.h"
#include "ElementManager.h"
#include "Plugin.h"
```

Include dependency graph for Attribute.h:



394 File Documentation

This graph shows which files directly or indirectly include this file:



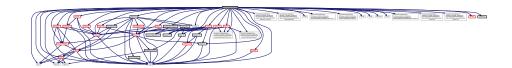
Classes

· 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 "ProbDistrib.h"
```

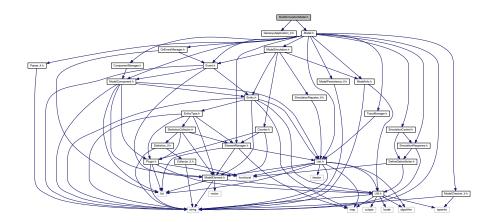
Include dependency graph for BuildSimulationModel.cpp:



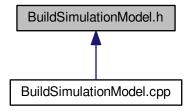
6.7 BuildSimulationModel.h File Reference

```
#include "GenesysApplication_if.h"
#include "Model.h"
```

Include dependency graph for BuildSimulationModel.h:



This graph shows which files directly or indirectly include this file:



Classes

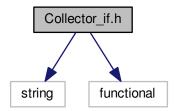
• class BuildSimulationModel

6.8 Collector_if.h File Reference

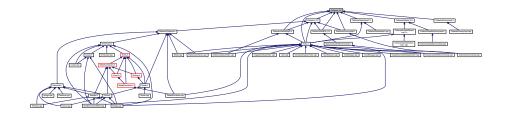
#include <string>
#include <functional>

396 File Documentation

Include dependency graph for Collector_if.h:



This graph shows which files directly or indirectly include this file:



Classes

· class Collector_if

Typedefs

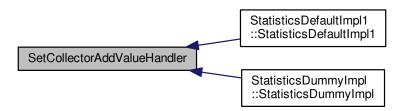
- $\hbox{ typedef std::} function < void (double) > Collector Add Value Handler \\$
- typedef std::function < void() > CollectorClearHandler

Functions

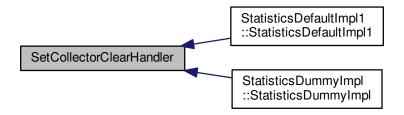
- template<typename Class >
 CollectorAddValueHandler SetCollectorAddValueHandler (void(Class::*function)(double), Class *object)
- template < typename Class >
 Collector Clear Handler Set Collector Clear Handler (void (Class::*function)(), Class *object)

- 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:



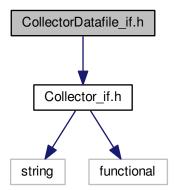
6.8.2.2 template<typename Class > CollectorClearHandler SetCollectorClearHandler (void(Class::*)() function, Class * object)



398 File Documentation

6.9 CollectorDatafile_if.h File Reference

#include "Collector_if.h"
Include dependency graph for CollectorDatafile_if.h:



This graph shows which files directly or indirectly include this file:



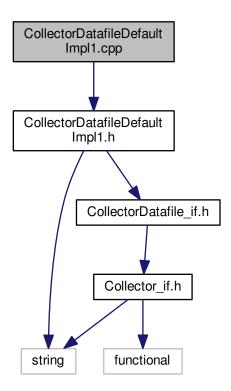
Classes

• class CollectorDatafile_if

6.10 CollectorDatafileDefaultImpl1.cpp File Reference

#include "CollectorDatafileDefaultImpl1.h"

Include dependency graph for CollectorDatafileDefaultImpl1.cpp:

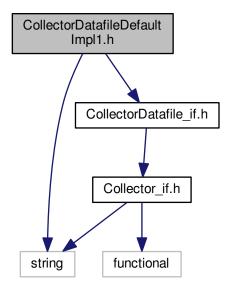


6.11 CollectorDatafileDefaultImpl1.h File Reference

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

400 File Documentation

Include dependency graph for CollectorDatafileDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



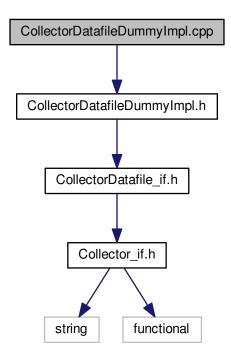
Classes

• class CollectorDatafileDefaultImpl1

6.12 CollectorDatafileDummyImpl.cpp File Reference

#include "CollectorDatafileDummyImpl.h"

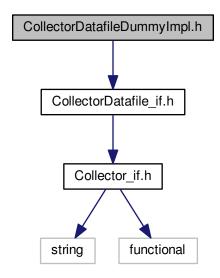
Include dependency graph for CollectorDatafileDummyImpl.cpp:



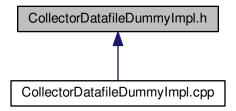
6.13 CollectorDatafileDummyImpl.h File Reference

402 File Documentation

Include dependency graph for CollectorDatafileDummyImpl.h:



This graph shows which files directly or indirectly include this file:



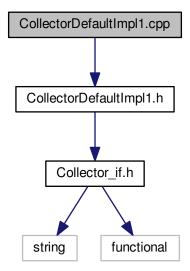
Classes

· class CollectorDatafileDummyImpl

6.14 CollectorDefaultImpl1.cpp File Reference

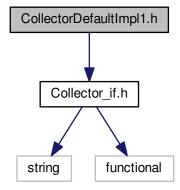
#include "CollectorDefaultImpl1.h"

Include dependency graph for CollectorDefaultImpl1.cpp:

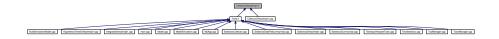


6.15 CollectorDefaultImpl1.h File Reference

#include "Collector_if.h"
Include dependency graph for CollectorDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



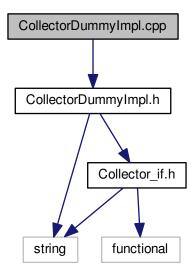
404 File Documentation

Classes

• class CollectorDefaultImpl1

6.16 CollectorDummyImpl.cpp File Reference

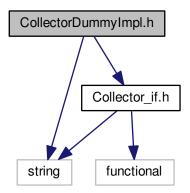
#include "CollectorDummyImpl.h"
Include dependency graph for CollectorDummyImpl.cpp:



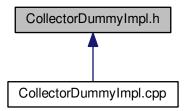
6.17 CollectorDummyImpl.h File Reference

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

Include dependency graph for CollectorDummyImpl.h:



This graph shows which files directly or indirectly include this file:



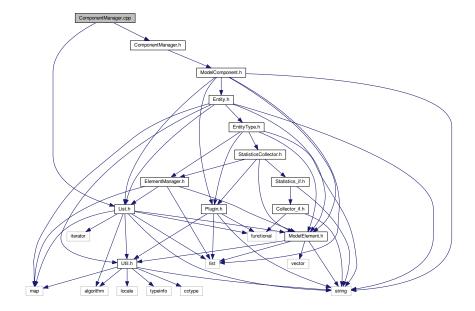
Classes

• class CollectorDummyImpl

6.18 ComponentManager.cpp File Reference

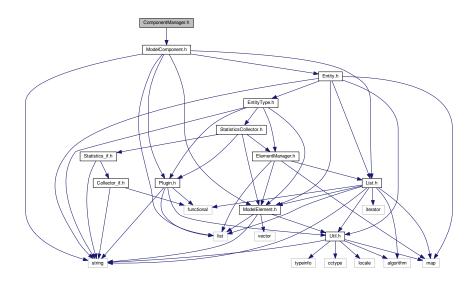
```
#include "ComponentManager.h"
#include "List.h"
```

Include dependency graph for ComponentManager.cpp:

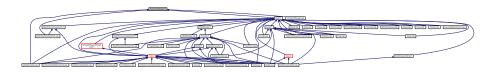


6.19 ComponentManager.h File Reference

#include "ModelComponent.h"
Include dependency graph for ComponentManager.h:



This graph shows which files directly or indirectly include this file:

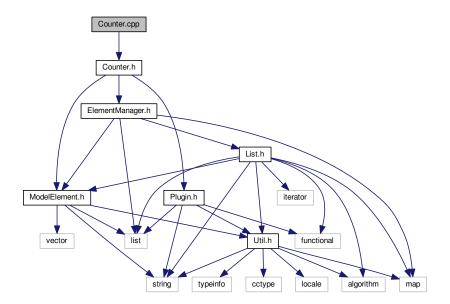


Classes

• class ComponentManager

6.20 Counter.cpp File Reference

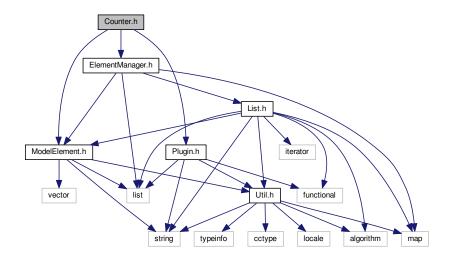
#include "Counter.h"
Include dependency graph for Counter.cpp:



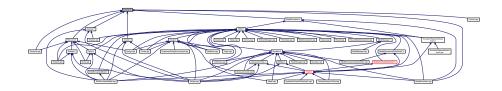
6.21 Counter.h File Reference

```
#include "ModelElement.h"
#include "ElementManager.h"
#include "Plugin.h"
```

Include dependency graph for Counter.h:



This graph shows which files directly or indirectly include this file:



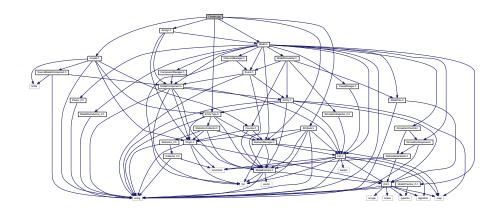
Classes

class Counter

6.22 Create.cpp File Reference

```
#include "Create.h"
#include "Model.h"
#include "EntityType.h"
#include "ElementManager.h"
#include "Attribute.h"
#include "Assign.h"
```

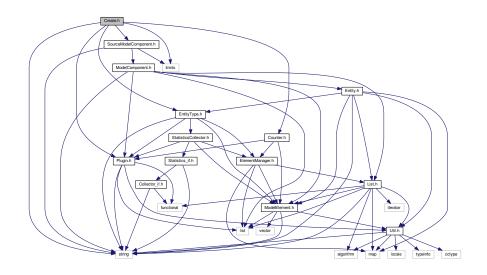
Include dependency graph for Create.cpp:



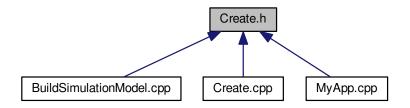
6.23 Create.h File Reference

```
#include <string>
#include <limits>
#include "SourceModelComponent.h"
#include "EntityType.h"
#include "Counter.h"
#include "Plugin.h"
```

Include dependency graph for Create.h:



This graph shows which files directly or indirectly include this file:



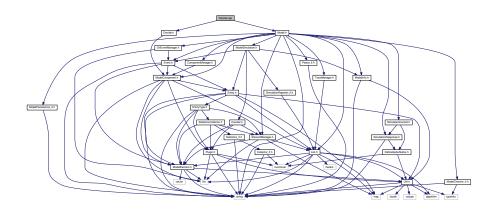
Classes

· class Create

6.24 Decide.cpp File Reference

#include "Decide.h"
#include "Model.h"

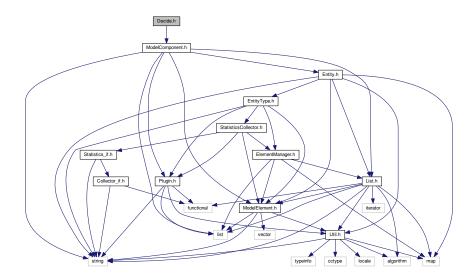
Include dependency graph for Decide.cpp:



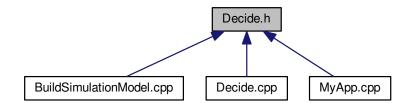
6.25 Decide.h File Reference

#include "ModelComponent.h"

Include dependency graph for Decide.h:



This graph shows which files directly or indirectly include this file:



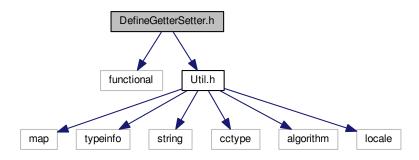
Classes

• class Decide

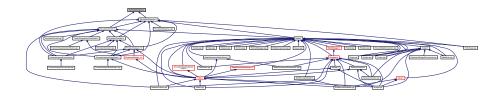
6.26 DefineGetterSetter.h File Reference

```
#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))

6.26.1 Typedef Documentation

- 6.26.1.1 typedef std::function < double() > GetterMember
- 6.26.1.2 typedef std::function<void(double) > SetterMember

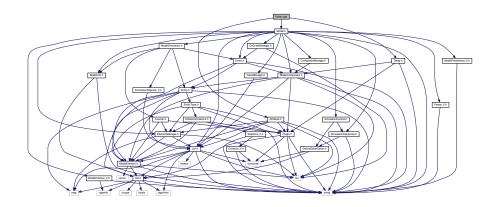
6.26.2 Function Documentation

- 6.26.2.1 template < typename Class > GetterMember DefineGetterMember (Class * object, double(Class::*)() function)
- 6.26.2.2 template < typename Class > GetterMember DefineGetterMember (Class * object, unsigned int(Class::*)() const function)
- 6.26.2.3 template < typename Class > GetterMember DefineGetterMember (Class * object, bool(Class::*)() const function)
- 6.26.2.4 template < typename Class > GetterMember DefineGetterMember (Class * object, std::string(Class::*)() const function)
- 6.26.2.5 template < typename Class > GetterMember DefineGetterMember (Class * object, Util::TimeUnit(Class::*)() const function)
- 6.26.2.6 template < typename Class > SetterMember DefineSetterMember (Class * object, void(Class::*)(double) function)
- 6.26.2.7 template < typename Class > SetterMember DefineSetterMember (Class * object, void(Class::*)(unsigned int) function)
- 6.26.2.8 template < typename Class > SetterMember DefineSetterMember (Class * object, void(Class::*)(bool) function)
- 6.26.2.9 template < typename Class > SetterMember DefineSetterMember (Class * object, void(Class::*)(std::string) const function)
- 6.26.2.10 template < typename Class > SetterMember DefineSetterMember (Class * object, void(Class::*)(Util::TimeUnit) function)

6.27 Delay.cpp File Reference

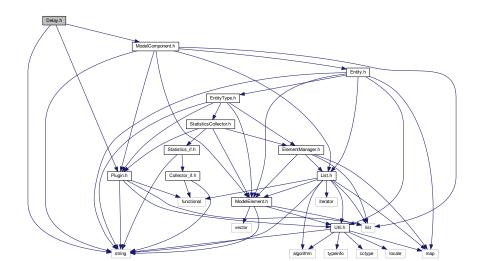
```
#include "Delay.h"
#include "Model.h"
#include "Attribute.h"
```

Include dependency graph for Delay.cpp:

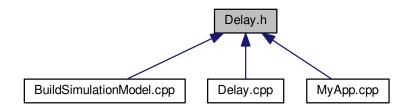


6.28 Delay.h File Reference

```
#include <string>
#include "ModelComponent.h"
#include "Plugin.h"
Include dependency graph for Delay.h:
```



This graph shows which files directly or indirectly include this file:



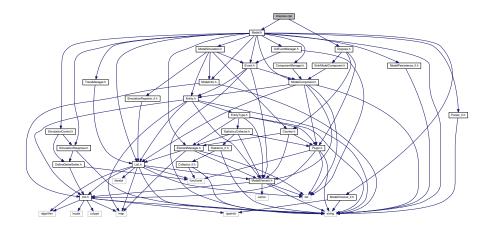
Classes

· class Delay

6.29 Dispose.cpp File Reference

```
#include "Dispose.h"
#include "Model.h"
```

Include dependency graph for Dispose.cpp:

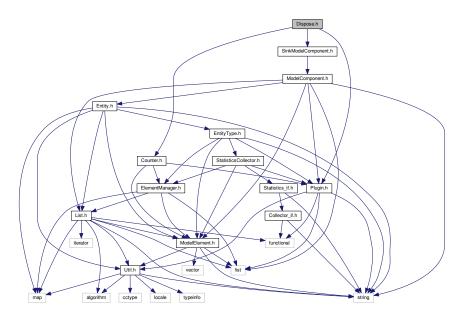


6.30 Dispose.h File Reference

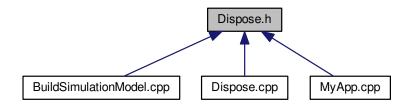
```
#include "SinkModelComponent.h"
```

#include "Counter.h"
#include "Plugin.h"

Include dependency graph for Dispose.h:



This graph shows which files directly or indirectly include this file:



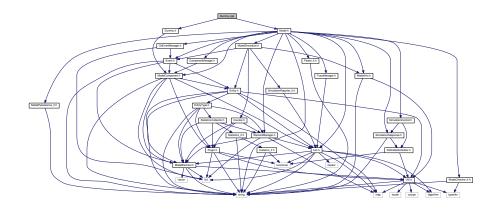
Classes

• class Dispose

6.31 Dummy.cpp File Reference

```
#include "Dummy.h"
#include "Model.h"
```

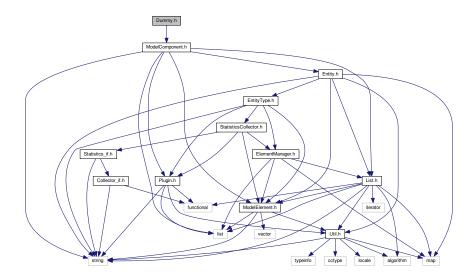
Include dependency graph for Dummy.cpp:



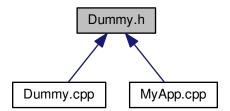
6.32 Dummy.h File Reference

#include "ModelComponent.h"

Include dependency graph for Dummy.h:



This graph shows which files directly or indirectly include this file:



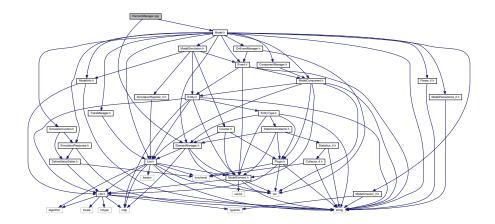
Classes

• class Dummy

6.33 ElementManager.cpp File Reference

```
#include "ElementManager.h"
#include "Model.h"
```

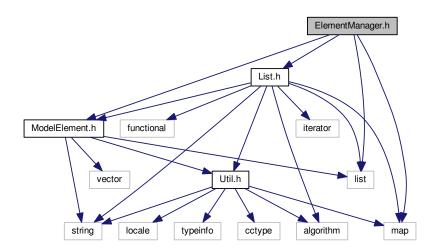
Include dependency graph for ElementManager.cpp:



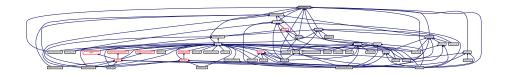
6.34 ElementManager.h File Reference

```
#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

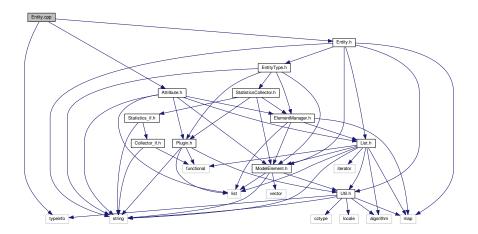
6.35 ElementManager_if.h File Reference

Classes

• class ElementManager_if

6.36 Entity.cpp File Reference

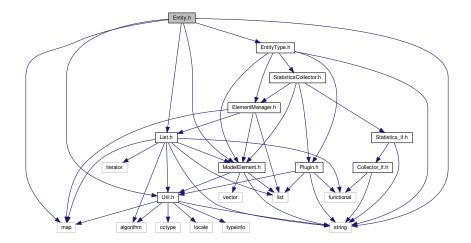
```
#include <typeinfo>
#include "Entity.h"
#include "Attribute.h"
Include dependency graph for Entity.cpp:
```



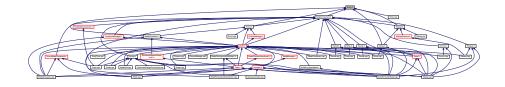
6.37 Entity.h File Reference

```
#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:

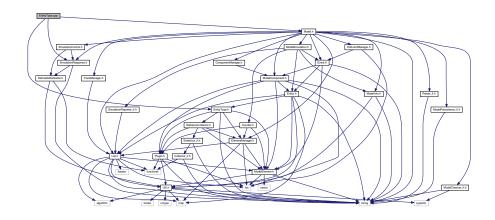


Classes

class Entity

6.38 EntityType.cpp File Reference

```
#include "EntityType.h"
#include "Model.h"
#include "SimulationResponse.h"
Include dependency graph for EntityType.cpp:
```



6.39 EntityType.h File Reference

```
#include <string>
#include "ModelElement.h"
#include "StatisticsCollector.h"
#include "ElementManager.h"
#include "Plugin.h"
Include dependency graph for EntityType.h:
```

ElementManager.h

StatisticsCollector.h

List.h

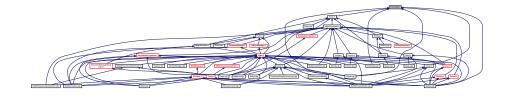
Statistics_if.h

Plugin.h

Collector_if.h

This graph shows which files directly or indirectly include this file:

algorithm typeinfo



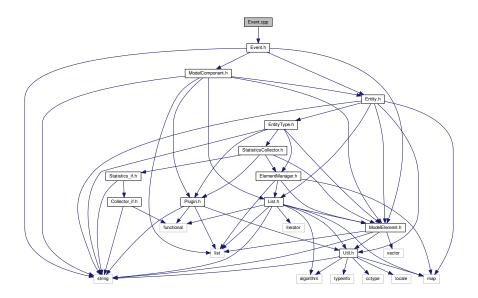
Classes

class EntityType

6.40 Event.cpp File Reference

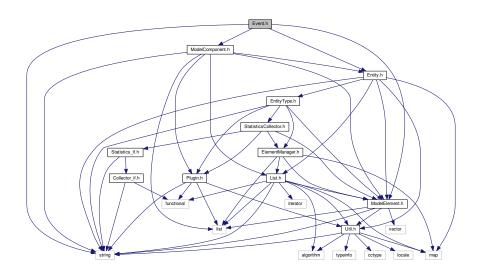
#include "Event.h"

Include dependency graph for Event.cpp:

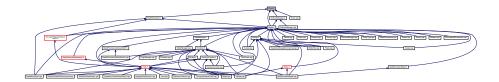


6.41 Event.h File Reference

```
#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:

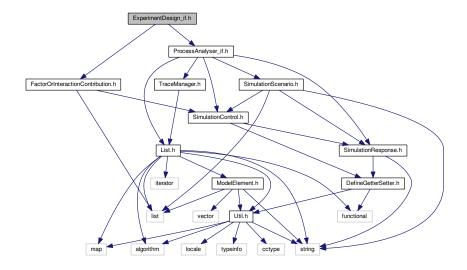


Classes

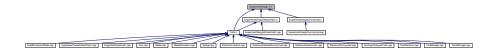
class Event

6.42 ExperimentDesign_if.h File Reference

#include "FactorOrInteractionContribution.h"
#include "ProcessAnalyser_if.h"
Include dependency graph for ExperimentDesign_if.h:



This graph shows which files directly or indirectly include this file:

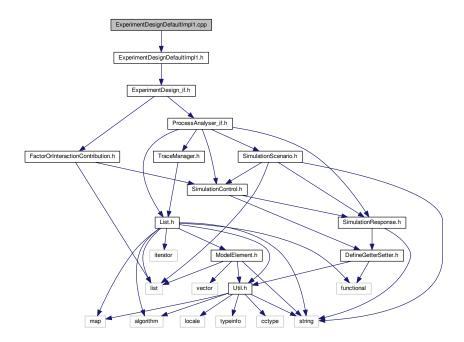


Classes

• class ExperimentDesign_if

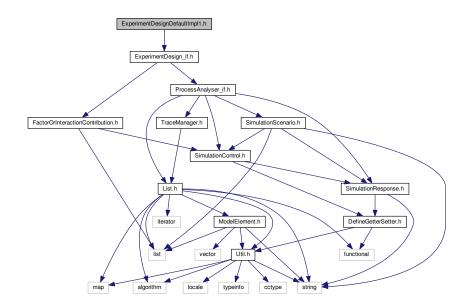
6.43 ExperimentDesignDefaultImpl1.cpp File Reference

#include "ExperimentDesignDefaultImpl1.h"
Include dependency graph for ExperimentDesignDefaultImpl1.cpp:



6.44 ExperimentDesignDefaultImpl1.h File Reference

#include "ExperimentDesign_if.h"
Include dependency graph for ExperimentDesignDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:

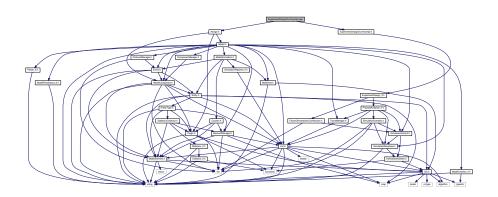


Classes

• class ExperimentDesignDefaultImpl1

6.45 ExperimentDesignDummyImpl.cpp File Reference

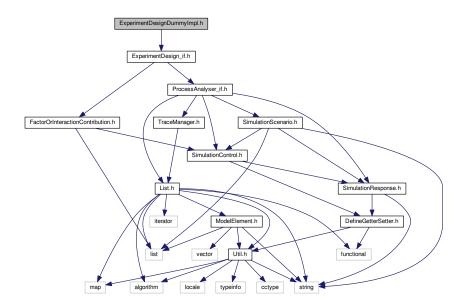
```
#include "ExperimentDesignDummyImpl.h"
#include "Assign.h"
Include dependency graph for ExperimentDesignDummyImpl.cpp:
```



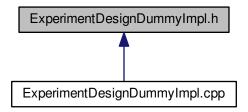
6.46 ExperimentDesignDummyImpl.h File Reference

#include "ExperimentDesign_if.h"

Include dependency graph for ExperimentDesignDummyImpl.h:



This graph shows which files directly or indirectly include this file:



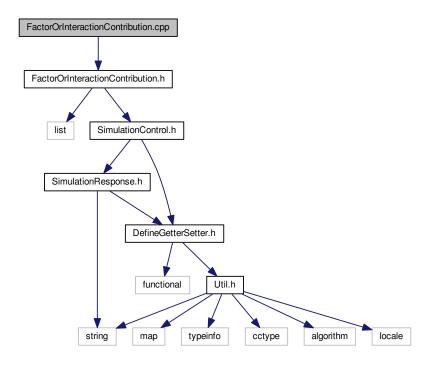
Classes

• class ExperimentDesignDummyImpl

6.47 FactorOrInteractionContribution.cpp File Reference

#include "FactorOrInteractionContribution.h"

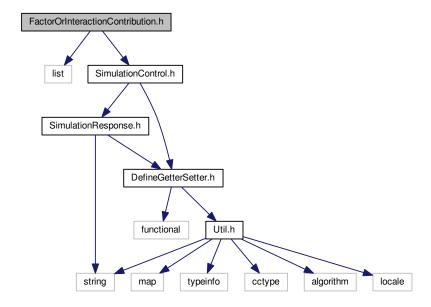
Include dependency graph for FactorOrInteractionContribution.cpp:



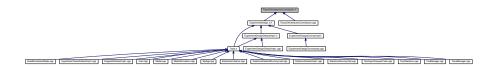
6.48 FactorOrInteractionContribution.h File Reference

#include <list>
#include "SimulationControl.h"

Include dependency graph for FactorOrInteractionContribution.h:



This graph shows which files directly or indirectly include this file:

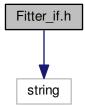


Classes

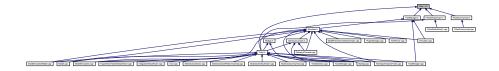
• class FactorOrInteractionContribution

6.49 Fitter_if.h File Reference

Include dependency graph for Fitter_if.h:



This graph shows which files directly or indirectly include this file:



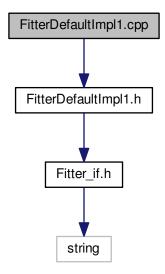
Classes

class Fitter_if

6.50 FitterDefaultImpl1.cpp File Reference

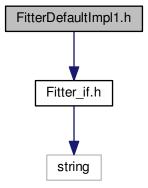
#include "FitterDefaultImpl1.h"

Include dependency graph for FitterDefaultImpl1.cpp:



6.51 FitterDefaultImpl1.h File Reference

#include "Fitter_if.h"
Include dependency graph for FitterDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:

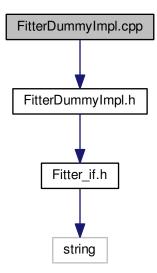


Classes

• class FitterDefaultImpl1

6.52 FitterDummyImpl.cpp File Reference

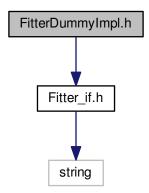
#include "FitterDummyImpl.h"
Include dependency graph for FitterDummyImpl.cpp:



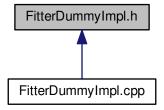
6.53 FitterDummylmpl.h File Reference

#include "Fitter_if.h"

Include dependency graph for FitterDummyImpl.h:



This graph shows which files directly or indirectly include this file:



Classes

class FitterDummyImpl

6.54 Functor.h File Reference

6.55 GenesysApplication_if.h File Reference

This graph shows which files directly or indirectly include this file:



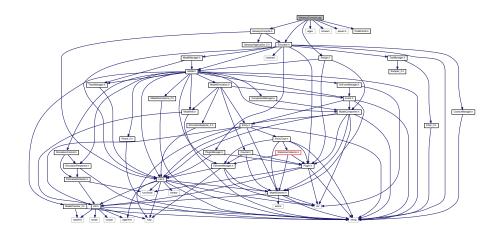
Classes

· class GenesysApplication_if

6.56 GenesysConsole.cpp File Reference

```
#include "GenesysConsole.h"
#include "Simulator.h"
#include "Assign.h"
#include <regex>
#include <fstream>
#include <assert.h>
#include "ProbDistrib.h"
```

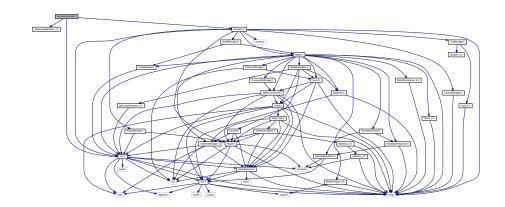
Include dependency graph for GenesysConsole.cpp:



6.57 GenesysConsole.h File Reference

```
#include "GenesysApplication_if.h"
#include "Simulator.h"
#include "List.h"
```

Include dependency graph for GenesysConsole.h:



This graph shows which files directly or indirectly include this file:

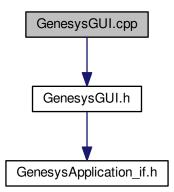


Classes

• class GenesysConsole

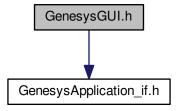
6.58 GenesysGUI.cpp File Reference

#include "GenesysGUI.h"
Include dependency graph for GenesysGUI.cpp:

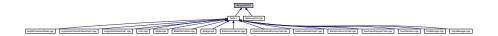


6.59 GenesysGUI.h File Reference

#include "GenesysApplication_if.h"
Include dependency graph for GenesysGUI.h:



This graph shows which files directly or indirectly include this file:

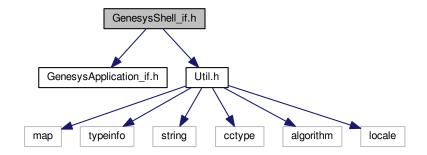


Classes

· class GenesysGUI

6.60 GenesysShell_if.h File Reference

```
#include "GenesysApplication_if.h"
#include "Util.h"
Include dependency graph for GenesysShell_if.h:
```



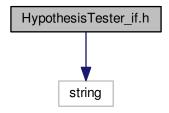
Classes

· class GenesysShell_if

6.61 HypothesisTester_if.h File Reference

#include <string>

Include dependency graph for HypothesisTester_if.h:



This graph shows which files directly or indirectly include this file:



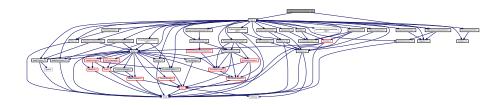
Classes

• class HypothesisTester_if

6.62 HypothesisTesterDefaultImpl1.cpp File Reference

```
#include "HypothesisTesterDefaultImpl1.h"
#include "Traits.h"
```

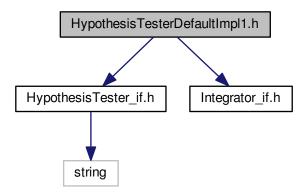
Include dependency graph for HypothesisTesterDefaultImpl1.cpp:



6.63 HypothesisTesterDefaultImpl1.h File Reference

```
#include "HypothesisTester_if.h"
#include "Integrator_if.h"
```

Include dependency graph for HypothesisTesterDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



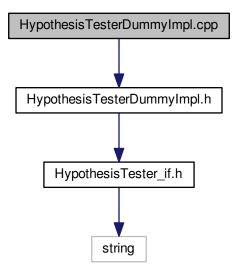
Classes

• class HypothesisTesterDefaultImpl1

6.64 HypothesisTesterDummyImpl.cpp File Reference

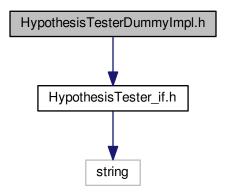
#include "HypothesisTesterDummyImpl.h"

Include dependency graph for HypothesisTesterDummyImpl.cpp:

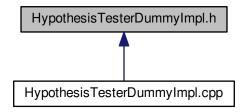


6.65 HypothesisTesterDummylmpl.h File Reference

#include "HypothesisTester_if.h"
Include dependency graph for HypothesisTesterDummyImpl.h:



This graph shows which files directly or indirectly include this file:



Classes

• class HypothesisTesterDummyImpl

6.66 Integrator_if.h File Reference

This graph shows which files directly or indirectly include this file:

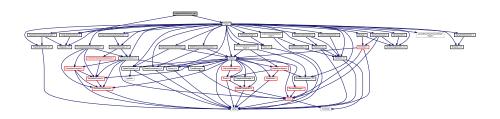


Classes

· class Integrator_if

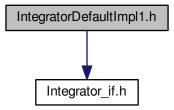
6.67 IntegratorDefaultImpl1.cpp File Reference

```
#include "IntegratorDefaultImpl1.h"
#include "Traits.h"
Include dependency graph for IntegratorDefaultImpl1.cpp:
```



6.68 Integrator Default Impl1.h File Reference

#include "Integrator_if.h"
Include dependency graph for IntegratorDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:

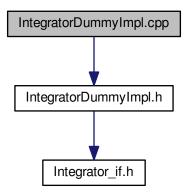


Classes

• class IntegratorDefaultImpl1

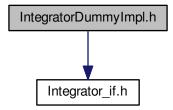
6.69 IntegratorDummyImpl.cpp File Reference

#include "IntegratorDummyImpl.h"
Include dependency graph for IntegratorDummyImpl.cpp:

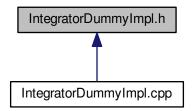


6.70 IntegratorDummyImpl.h File Reference

#include "Integrator_if.h"
Include dependency graph for IntegratorDummyImpl.h:



This graph shows which files directly or indirectly include this file:



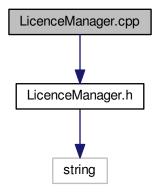
Classes

• class IntegratorDummyImpl

6.71 LicenceManager.cpp File Reference

#include "LicenceManager.h"

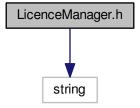
Include dependency graph for LicenceManager.cpp:



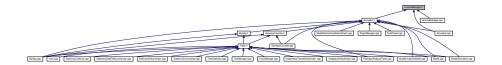
6.72 LicenceManager.h File Reference

#include <string>

Include dependency graph for LicenceManager.h:



This graph shows which files directly or indirectly include this file:

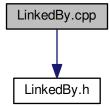


Classes

• class LicenceManager

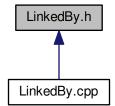
6.73 LinkedBy.cpp File Reference

```
#include "LinkedBy.h"
Include dependency graph for LinkedBy.cpp:
```



6.74 LinkedBy.h File Reference

This graph shows which files directly or indirectly include this file:



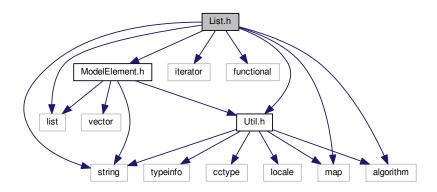
Classes

• class LinkedBy

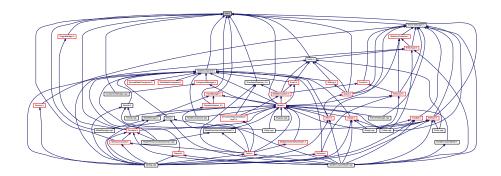
6.75 List.h File Reference

```
#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:



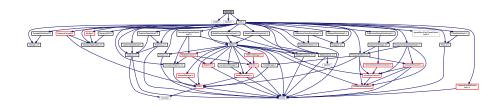
Classes

class List< T >

6.76 main.cpp File Reference

#include <cstdlib>
#include <iostream>
#include "Traits.h"

Include dependency graph for main.cpp:



Functions

• int main (int argc, char **argv)

6.76.1 Function Documentation

```
6.76.1.1 int main ( int argc, char ** argv )
```

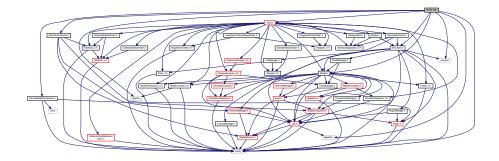
Here is the call graph for this function:



6.77 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)

6.77.1 Function Documentation

6.77.1.1 bool EventCompare (const Event *a, const Event *b)

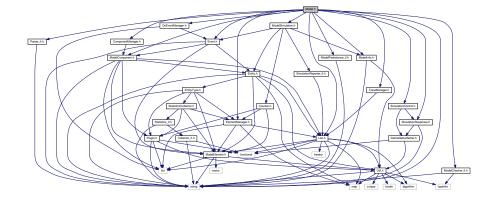
Here is the call graph for this function:



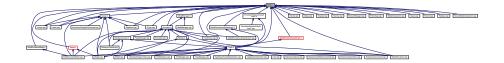
- 6.77.1.2 double getReplicationLengthNotMemberFunction ()
- 6.77.1.3 void setReplicationLengthNotMemberFunction (double value)

6.78 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 "ComponentManager.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:

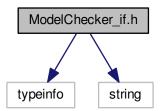


Classes

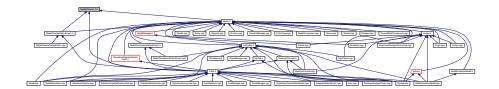
· class Model

6.79 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:



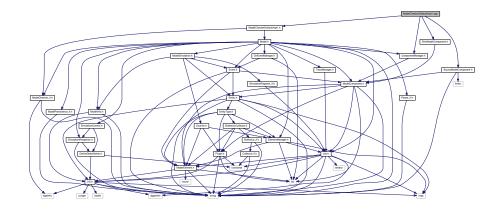
Classes

• class ModelChecker_if

6.80 ModelCheckerDefaultImpl1.cpp File Reference

```
#include "ModelCheckerDefaultImpl1.h"
#include "SourceModelComponent.h"
#include "SinkModelComponent.h"
#include "ComponentManager.h"
```

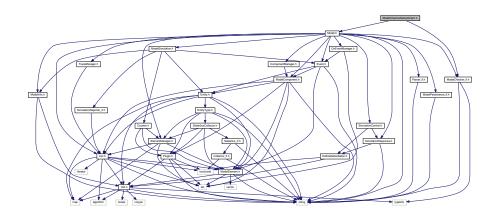
Include dependency graph for ModelCheckerDefaultImpl1.cpp:



6.81 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:



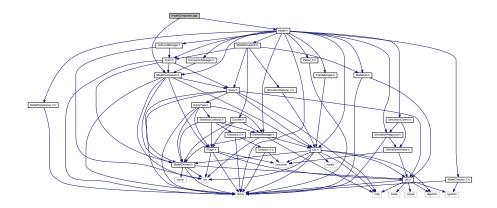
Classes

• class ModelCheckerDefaultImpl1

6.82 ModelComponent.cpp File Reference

```
#include "ModelComponent.h"
#include "Model.h"
```

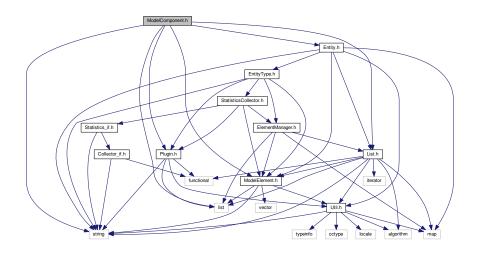
Include dependency graph for ModelComponent.cpp:



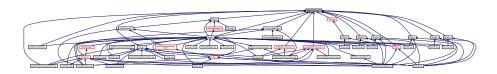
6.83 ModelComponent.h File Reference

```
#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

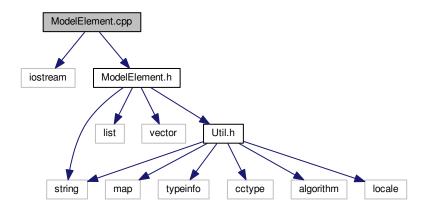
6.84 ModelComponentManager_if.h File Reference

Classes

· class ModelComponentManager_if

6.85 ModelElement.cpp File Reference

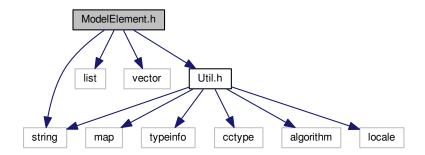
```
#include <iostream>
#include "ModelElement.h"
Include dependency graph for ModelElement.cpp:
```



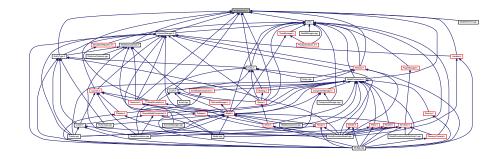
6.86 ModelElement.h File Reference

```
#include <string>
#include <list>
#include <vector>
#include "Util.h"
```

Include dependency graph for ModelElement.h:



This graph shows which files directly or indirectly include this file:



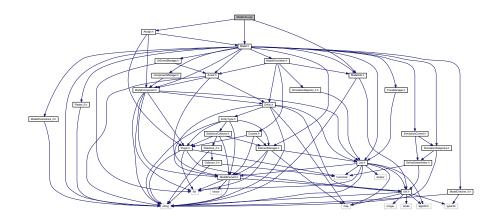
Classes

class ModelElement

6.87 ModelInfo.cpp File Reference

```
#include "ModelInfo.h"
#include "Model.h"
#include "Assign.h"
```

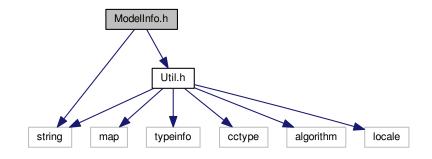
Include dependency graph for ModelInfo.cpp:



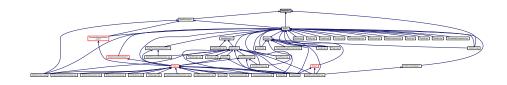
6.88 ModelInfo.h File Reference

#include <string>
#include "Util.h"

Include dependency graph for ModelInfo.h:



This graph shows which files directly or indirectly include this file:



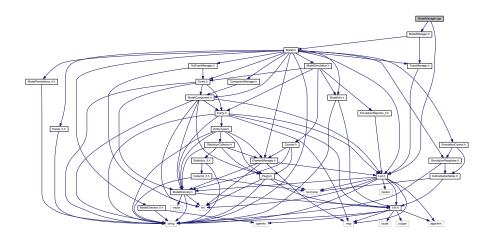
Classes

• class ModelInfo

6.89 ModelManager.cpp File Reference

#include "ModelManager.h"
#include "List.h"

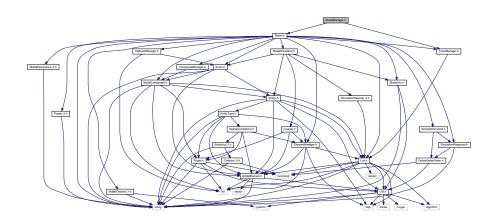
Include dependency graph for ModelManager.cpp:



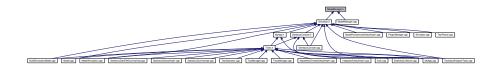
6.90 ModelManager.h File Reference

#include "Model.h"
#include "TraceManager.h"

Include dependency graph for ModelManager.h:



This graph shows which files directly or indirectly include this file:

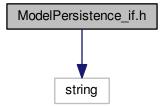


Classes

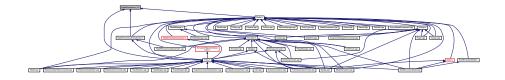
· class ModelManager

6.91 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:



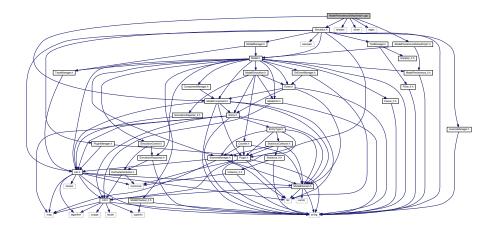
Classes

• class ModelPersistence_if

6.92 ModelPersistenceDefaultImpl1.cpp File Reference

```
#include "ModelPersistenceDefaultImpl1.h"
#include <fstream>
#include <ctime>
#include <regex>
#include "ModelComponent.h"
#include "Simulator.h"
```

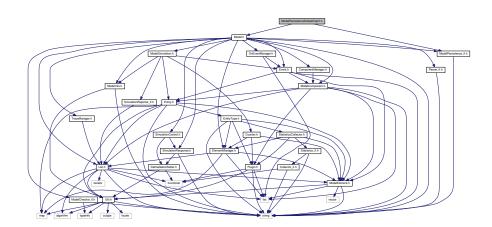
Include dependency graph for ModelPersistenceDefaultImpl1.cpp:



6.93 ModelPersistenceDefaultImpl1.h File Reference

#include "ModelPersistence_if.h"
#include "Model.h"

Include dependency graph for ModelPersistenceDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



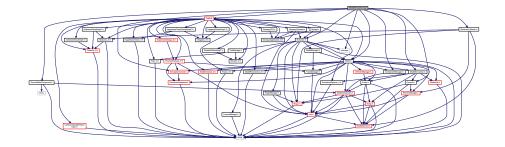
Classes

• class ModelPersistenceDefaultImpl1

6.94 ModelSimulation.cpp File Reference

```
#include "ModelSimulation.h"
#include <iostream>
#include "Model.h"
#include "Simulator.h"
#include "SourceModelComponent.h"
#include "StatisticsCollector.h"
#include "Counter.h"
#include "Traits.h"
#include "SimulationControl.h"
#include "ComponentManager.h"
```

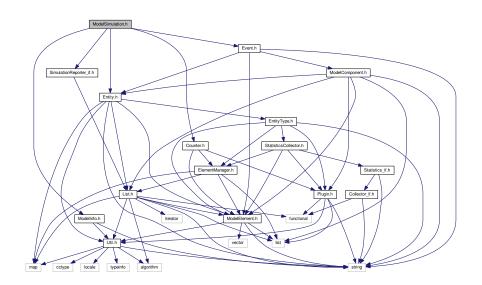
Include dependency graph for ModelSimulation.cpp:



6.95 ModelSimulation.h File Reference

```
#include "Event.h"
#include "Entity.h"
#include "ModelInfo.h"
#include "SimulationReporter_if.h"
#include "Counter.h"
```

Include dependency graph for ModelSimulation.h:



This graph shows which files directly or indirectly include this file:

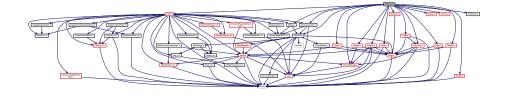


Classes

· class ModelSimulation

6.96 MyApp.cpp File Reference

```
#include "MyApp.h"
#include "GenesysConsole.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 "Dummy.h"
#include "ElementManager.h"
#include "EntityType.h"
#include "Attribute.h"
#include "Variable.h"
#include "ProbDistrib.h"
Include dependency graph for MyApp.cpp:
```



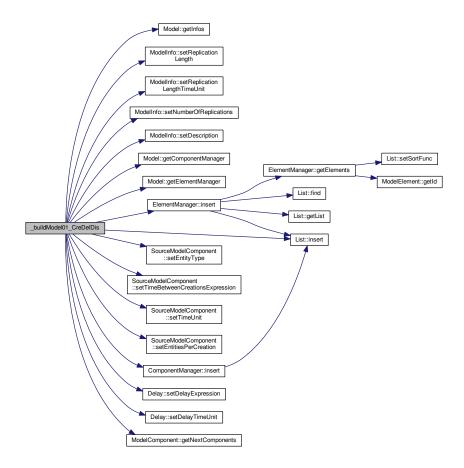
Functions

- void traceHandler (TraceEvent e)
- void traceSimulationHandler (TraceSimulationEvent e)
- void onSimulationStartHandler (SimulationEvent *re)
- void onReplicationStartHandler (SimulationEvent *re)
- void onProcessEventHandler (SimulationEvent *re)
- void onReplicationEndHandler (SimulationEvent *re)

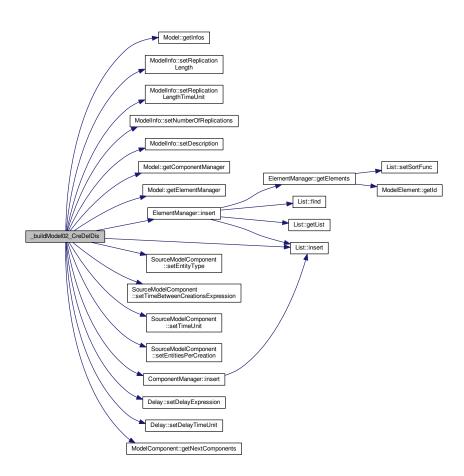
- void onEntityRemoveHandler (SimulationEvent *re)
- void _buildModel01_CreDelDis (Model *model)
- void _buildModel02_CreDelDis (Model *model)
- void _buildModel03_CreSeiDelResDis (Model *model)
- void _buildMostCompleteModel (Model *model)

6.96.1 Function Documentation

6.96.1.1 void _buildModel01_CreDelDis (Model * model)

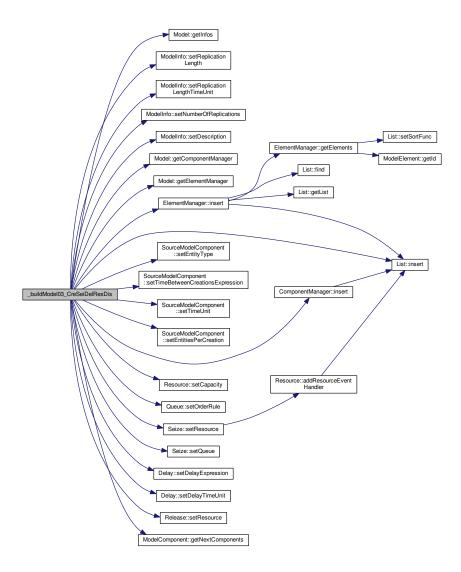


6.96.1.2 void _buildModel02_CreDelDis (Model*model)



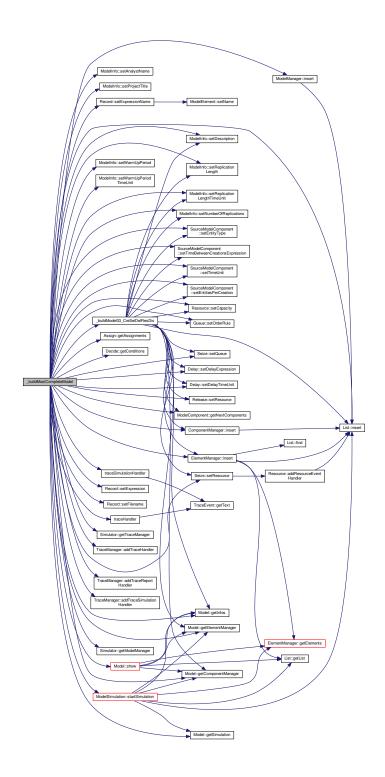
6.96.1.3 void _buildModel03_CreSeiDelResDis (Model * model)

Here is the call graph for this function:



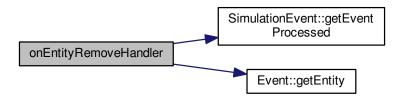


6.96.1.4 void _buildMostCompleteModel (Model*model)



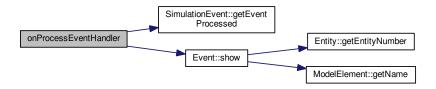
6.96.1.5 void on Entity Remove Handler (Simulation Event *re)

Here is the call graph for this function:

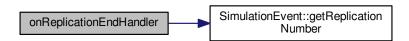


6.96.1.6 void on Process Event Handler (Simulation Event *re)

Here is the call graph for this function:

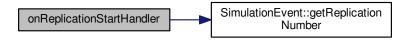


6.96.1.7 void on Replication End Handler (Simulation Event *re)



6.96.1.8 void on Replication Start Handler (Simulation Event *re)

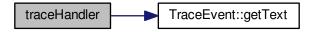
Here is the call graph for this function:



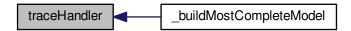
6.96.1.9 void on Simulation Start Handler (Simulation Event *re)

6.96.1.10 void traceHandler (TraceEvent e)

Here is the call graph for this function:



Here is the caller graph for this function:



6.96.1.11 void traceSimulationHandler (TraceSimulationEvent e)

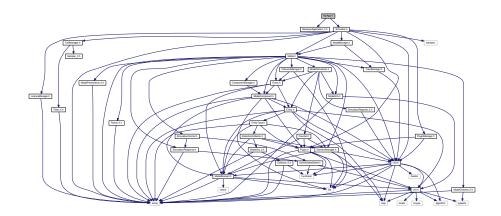


Here is the caller graph for this function:



6.97 MyApp.h File Reference

```
#include "GenesysApplication_if.h"
#include "Simulator.h"
Include dependency graph for MyApp.h:
```



This graph shows which files directly or indirectly include this file:



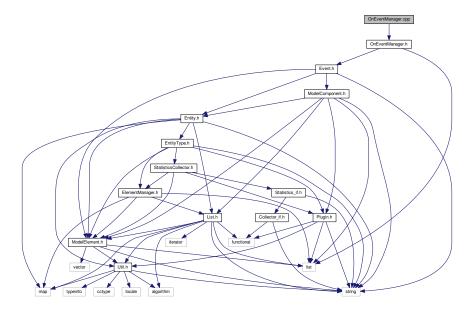
Classes

class MyApp

6.98 OnEventManager.cpp File Reference

#include "OnEventManager.h"

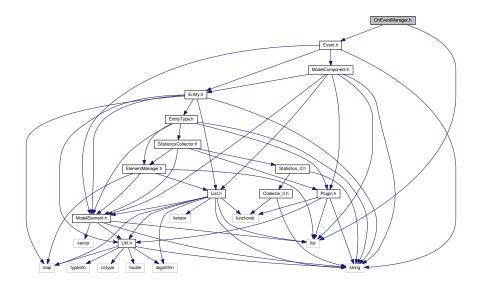
Include dependency graph for OnEventManager.cpp:



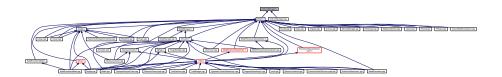
6.99 OnEventManager.h File Reference

#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

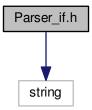
• typedef void(* simulationEventHandler) (SimulationEvent *)

6.99.1 Typedef Documentation

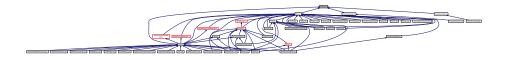
6.99.1.1 typedef void(* simulationEventHandler) (SimulationEvent *)

6.100 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:

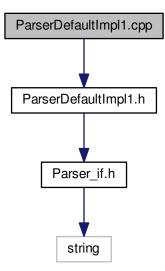


Classes

class Parser_if

6.101 ParserDefaultImpl1.cpp File Reference

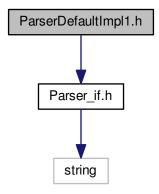
#include "ParserDefaultImpl1.h"
Include dependency graph for ParserDefaultImpl1.cpp:



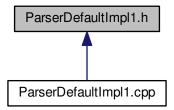
6.102 ParserDefaultImpl1.h File Reference

#include "Parser_if.h"

Include dependency graph for ParserDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



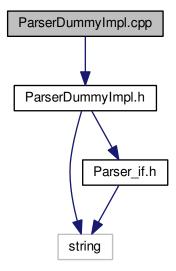
Classes

• class ParserDefaultImpl1

6.103 ParserDummyImpl.cpp File Reference

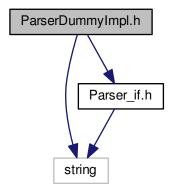
#include "ParserDummyImpl.h"

Include dependency graph for ParserDummyImpl.cpp:

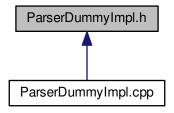


6.104 ParserDummyImpl.h File Reference

```
#include <string>
#include "Parser_if.h"
Include dependency graph for ParserDummyImpl.h:
```



This graph shows which files directly or indirectly include this file:



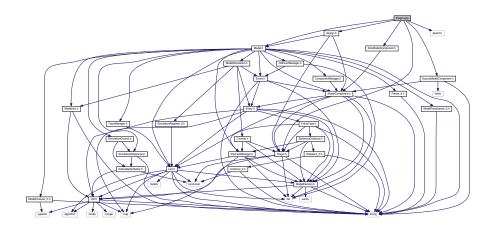
Classes

• class ParserDummyImpl

6.105 Plugin.cpp File Reference

```
#include "Plugin.h"
#include "Model.h"
#include "SourceModelComponent.h"
#include "SinkModelComponent.h"
#include "Assign.h"
#include <assert.h>
```

Include dependency graph for Plugin.cpp:

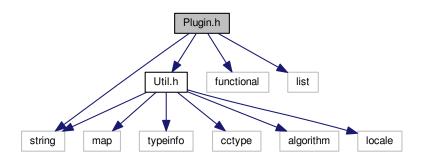


6.106 Plugin.h File Reference

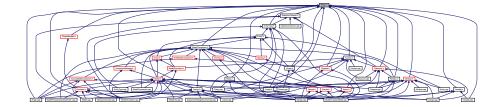
#include "Util.h"

```
#include <string>
#include <functional>
#include <list>
```

Include dependency graph for Plugin.h:



This graph shows which files directly or indirectly include this file:



Classes

- · class PluginInformation
- class Plugin

Typedefs

- typedef ModelComponent *(* StaticLoaderComponentInstance) (Model *, std::map< std::string, std::string > *)
- typedef ModelElement *(* StaticLoaderElementInstance) (ElementManager *, std::map< std::string, std
 ::string > *)
- typedef PluginInformation *(* StaticGetPluginInformation) ()

6.106.1 Typedef Documentation

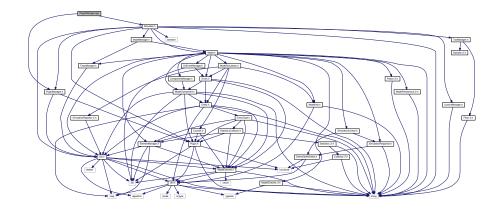
- 6.106.1.1 typedef PluginInformation*(* StaticGetPluginInformation) ()
- $\textbf{6.106.1.2} \quad \textbf{typedef ModelComponent}* (* \ \textbf{StaticLoaderComponentInstance}) \ (\textbf{Model} \ *, \ \textbf{std::string} \ < \ *)$

6.106.1.3 typedef ModelElement*(* StaticLoaderElementInstance) (ElementManager *, std::map< std::string, std::string > *)

6.107 PluginManager.cpp File Reference

#include "PluginManager.h"
#include "Simulator.h"

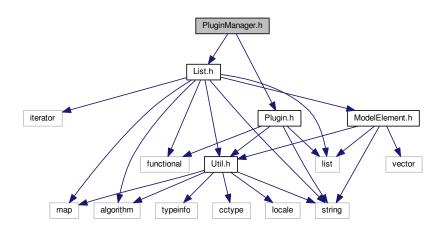
Include dependency graph for PluginManager.cpp:



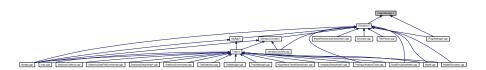
6.108 PluginManager.h File Reference

#include "List.h"
#include "Plugin.h"

Include dependency graph for PluginManager.h:



This graph shows which files directly or indirectly include this file:



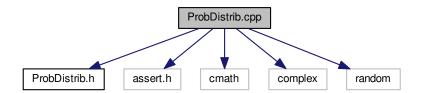
Classes

• class PluginManager

6.109 ProbDistrib.cpp File Reference

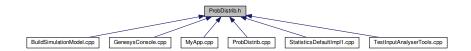
```
#include "ProbDistrib.h"
#include <assert.h>
#include <cmath>
#include <complex>
#include <random>
```

Include dependency graph for ProbDistrib.cpp:



6.110 ProbDistrib.h File Reference

This graph shows which files directly or indirectly include this file:



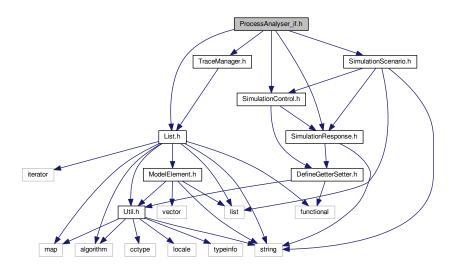
Classes

· class ProbDistrib

6.111 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:



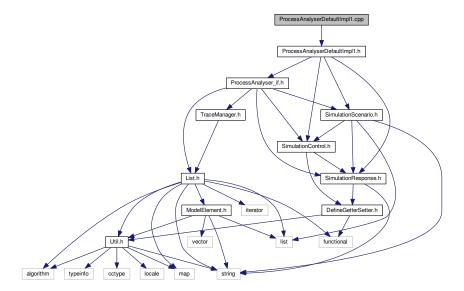
Classes

• class ProcessAnalyser_if

6.112 ProcessAnalyserDefaultImpl1.cpp File Reference

#include "ProcessAnalyserDefaultImpl1.h"

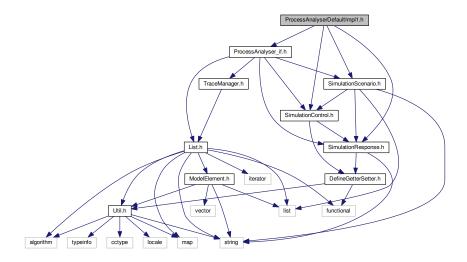
Include dependency graph for ProcessAnalyserDefaultImpl1.cpp:



6.113 ProcessAnalyserDefaultImpl1.h File Reference

```
#include "ProcessAnalyser_if.h"
#include "SimulationScenario.h"
#include "SimulationResponse.h"
#include "SimulationControl.h"
```

Include dependency graph for ProcessAnalyserDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:

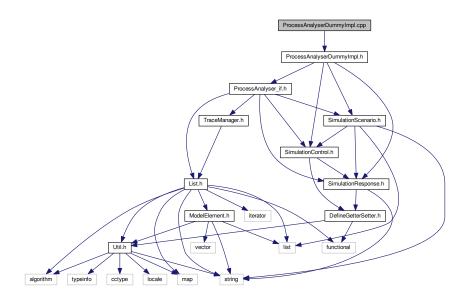


Classes

• class ProcessAnalyserDefaultImpl1

6.114 ProcessAnalyserDummyImpl.cpp File Reference

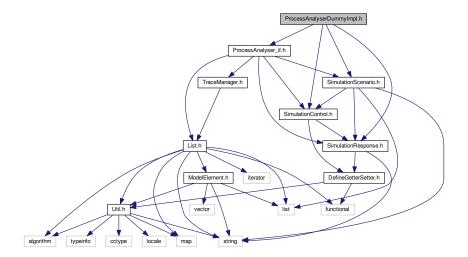
#include "ProcessAnalyserDummyImpl.h"
Include dependency graph for ProcessAnalyserDummyImpl.cpp:



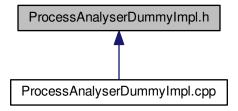
6.115 ProcessAnalyserDummyImpl.h File Reference

```
#include "ProcessAnalyser_if.h"
#include "SimulationScenario.h"
#include "SimulationResponse.h"
#include "SimulationControl.h"
```

Include dependency graph for ProcessAnalyserDummyImpl.h:



This graph shows which files directly or indirectly include this file:



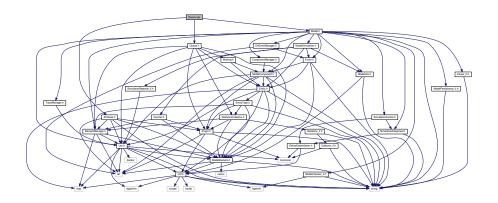
Classes

• class ProcessAnalyserDummyImpl

6.116 Queue.cpp File Reference

```
#include "Queue.h"
#include "Model.h"
#include "Attribute.h"
```

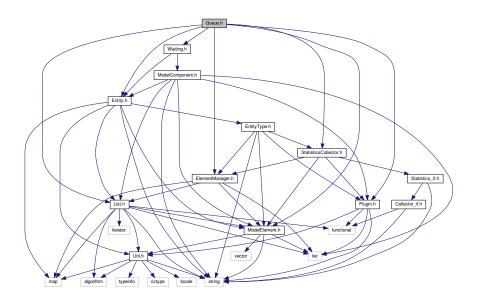
Include dependency graph for Queue.cpp:



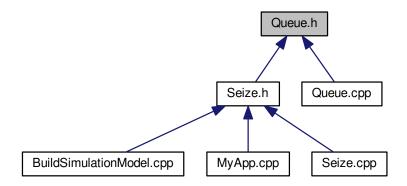
6.117 Queue.h File Reference

```
#include "ModelElement.h"
#include "List.h"
#include "Entity.h"
#include "Waiting.h"
#include "ElementManager.h"
#include "StatisticsCollector.h"
#include "Plugin.h"
```

Include dependency graph for Queue.h:



This graph shows which files directly or indirectly include this file:



Classes

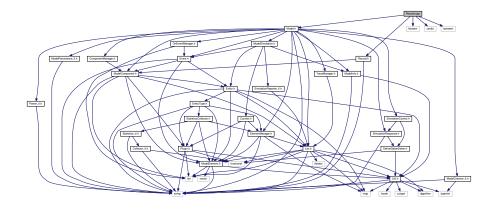
• class Queue

6.118 README.md File Reference

6.119 Record.cpp File Reference

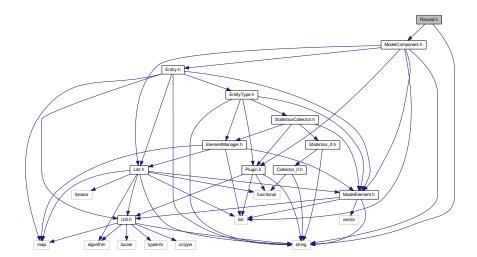
```
#include "Record.h"
#include "Model.h"
#include <fstream>
#include <cstdio>
#include <iostream>
```

Include dependency graph for Record.cpp:

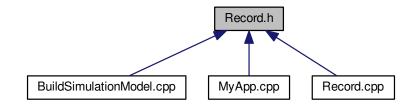


6.120 Record.h File Reference

#include "ModelComponent.h"
#include <string>
Include dependency graph for Record.h:



This graph shows which files directly or indirectly include this file:



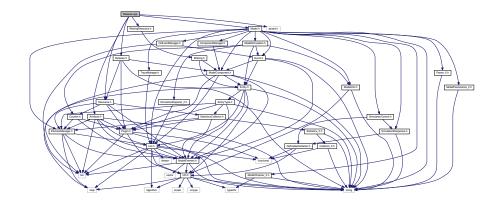
Classes

class Record

6.121 Release.cpp File Reference

```
#include "Release.h"
#include "Model.h"
#include "WaitingResource.h"
#include "Resource.h"
#include "Attribute.h"
#include <assert.h>
```

Include dependency graph for Release.cpp:



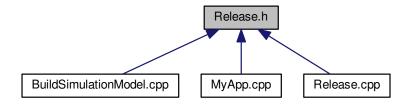
6.122 Release.h File Reference

```
#include <string>
#include "ModelComponent.h"
#include "Resource.h"
#include "Plugin.h"
Include dependency graph for Release.h:
```

Resource It

| Batalatics Collector A | Counter It
| Collector #It
| Collector

This graph shows which files directly or indirectly include this file:



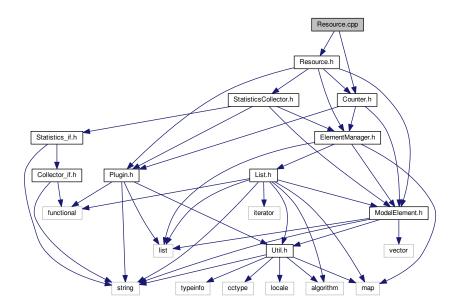
Classes

• class Release

6.123 Resource.cpp File Reference

#include "Resource.h"
#include "Counter.h"

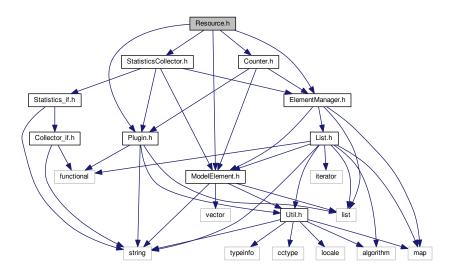
Include dependency graph for Resource.cpp:



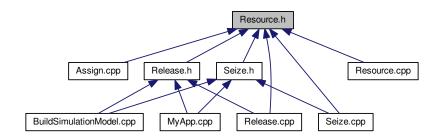
6.124 Resource.h File Reference

```
#include "ModelElement.h"
#include "StatisticsCollector.h"
#include "ElementManager.h"
#include "Counter.h"
#include "Plugin.h"
```

Include dependency graph for Resource.h:



This graph shows which files directly or indirectly include this file:

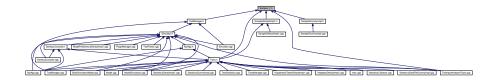


Classes

• class Resource

6.125 Sampler_if.h File Reference

This graph shows which files directly or indirectly include this file:

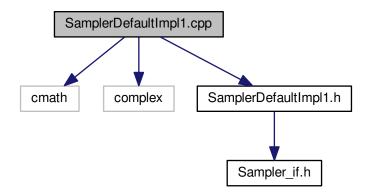


Classes

- · class Sampler_if
- struct Sampler_if::RNG_Parameters

6.126 SamplerDefaultImpl1.cpp File Reference

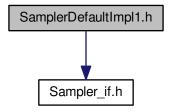
```
#include <cmath>
#include <complex>
#include "SamplerDefaultImpl1.h"
Include dependency graph for SamplerDefaultImpl1.cpp:
```



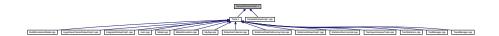
6.127 SamplerDefaultImpl1.h File Reference

#include "Sampler_if.h"

Include dependency graph for SamplerDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:



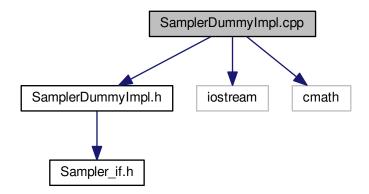
Classes

- class SamplerDefaultImpl1
- struct SamplerDefaultImpl1::DefaultImpl1RNG_Parameters

6.128 SamplerDummyImpl.cpp File Reference

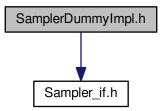
```
#include "SamplerDummyImpl.h"
#include <iostream>
#include <cmath>
```

Include dependency graph for SamplerDummyImpl.cpp:

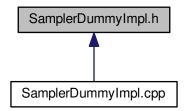


6.129 SamplerDummyImpl.h File Reference

#include "Sampler_if.h"
Include dependency graph for SamplerDummyImpl.h:



This graph shows which files directly or indirectly include this file:



Classes

- class SamplerDummyImpl
- struct SamplerDummyImpl::MyRNG_Parameters

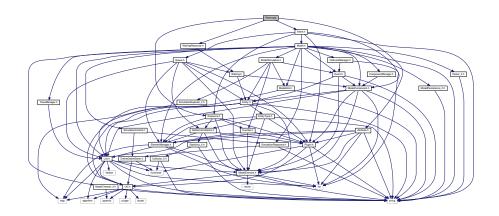
6.130 ScenarioExperiment_if.h File Reference

Classes

class ScenarioExperiment_if

6.131 Seize.cpp File Reference

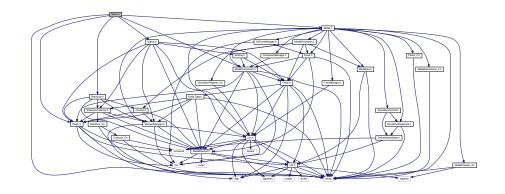
```
#include "Seize.h"
#include "WaitingResource.h"
#include "Resource.h"
#include "Attribute.h"
Include dependency graph for Seize.cpp:
```



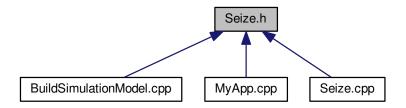
6.132 Seize.h File Reference

```
#include <string>
#include "ModelComponent.h"
#include "Model.h"
#include "Resource.h"
#include "Queue.h"
#include "Plugin.h"
```

Include dependency graph for Seize.h:



This graph shows which files directly or indirectly include this file:

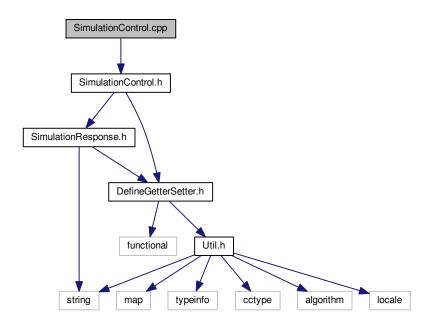


Classes

• class Seize

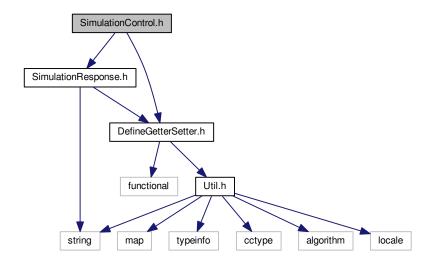
6.133 SimulationControl.cpp File Reference

#include "SimulationControl.h"
Include dependency graph for SimulationControl.cpp:

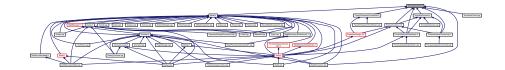


6.134 SimulationControl.h File Reference

```
#include "SimulationResponse.h"
#include "DefineGetterSetter.h"
Include dependency graph for SimulationControl.h:
```



This graph shows which files directly or indirectly include this file:



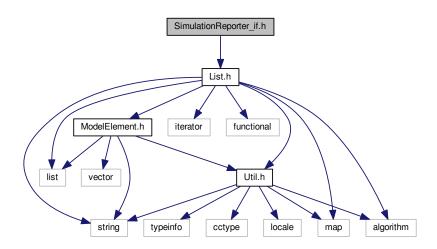
Classes

· class SimulationControl

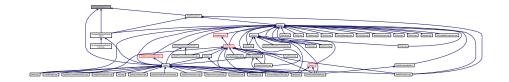
6.135 SimulationReporter_if.h File Reference

#include "List.h"

Include dependency graph for SimulationReporter_if.h:



This graph shows which files directly or indirectly include this file:



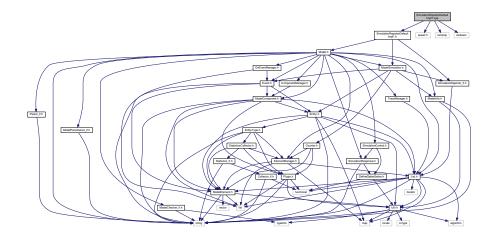
Classes

• class SimulationReporter_if

6.136 SimulationReporterDefaultImpl1.cpp File Reference

```
#include "SimulationReporterDefaultImpl1.h"
#include <assert.h>
#include <iomanip>
#include <iostream>
```

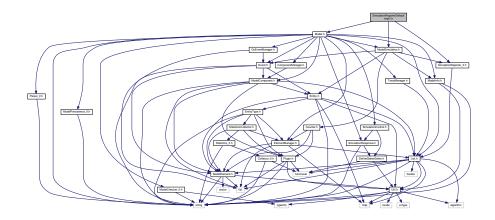
Include dependency graph for SimulationReporterDefaultImpl1.cpp:



6.137 SimulationReporterDefaultImpl1.h File Reference

```
#include "SimulationReporter_if.h"
#include "ModelSimulation.h"
#include "Model.h"
```

Include dependency graph for SimulationReporterDefaultImpl1.h:



This graph shows which files directly or indirectly include this file:

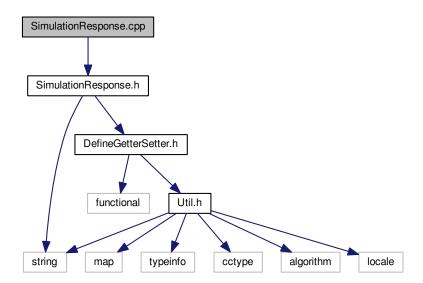


Classes

• class SimulationReporterDefaultImpl1

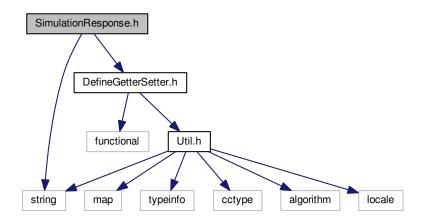
6.138 SimulationResponse.cpp File Reference

#include "SimulationResponse.h"
Include dependency graph for SimulationResponse.cpp:

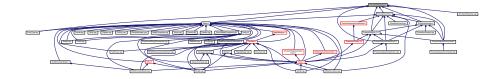


6.139 SimulationResponse.h File Reference

#include <string>
#include "DefineGetterSetter.h"
Include dependency graph for SimulationResponse.h:



This graph shows which files directly or indirectly include this file:

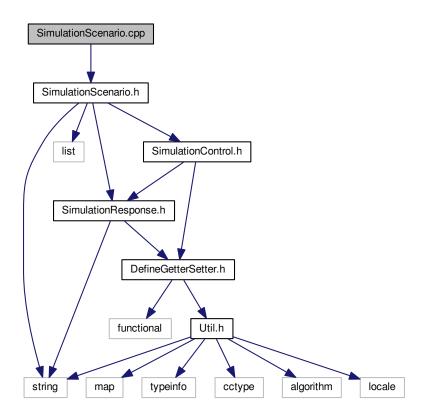


Classes

• class SimulationResponse

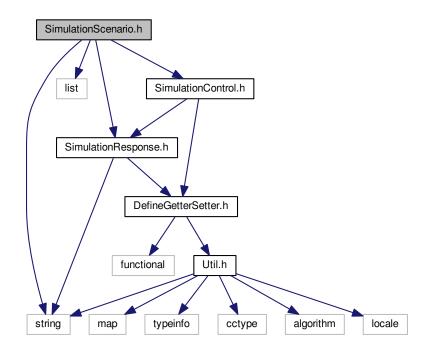
6.140 SimulationScenario.cpp File Reference

#include "SimulationScenario.h"
Include dependency graph for SimulationScenario.cpp:

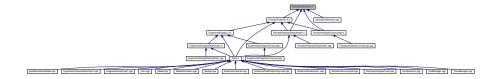


6.141 SimulationScenario.h File Reference

```
#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:



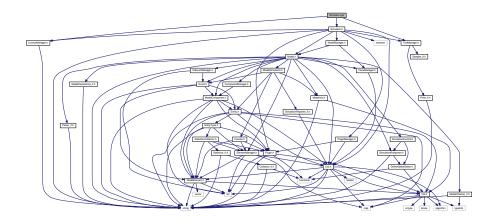
Classes

• class SimulationScenario

6.142 Simulator.cpp File Reference

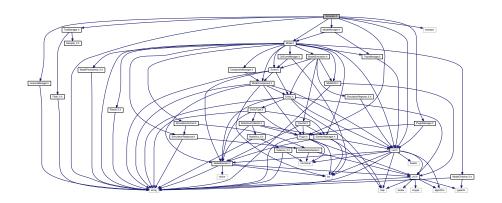
#include "Simulator.h"

```
#include "LicenceManager.h"
#include "ToolManager.h"
Include dependency graph for Simulator.cpp:
```

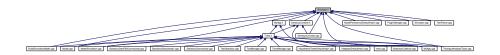


6.143 Simulator.h File Reference

```
#include <string>
#include <iostream>
#include "Model.h"
#include "Plugin.h"
#include "List.h"
#include "LicenceManager.h"
#include "PluginManager.h"
#include "ModelManager.h"
#include "ToolManager.h"
Include dependency graph for Simulator.h:
```



This graph shows which files directly or indirectly include this file:

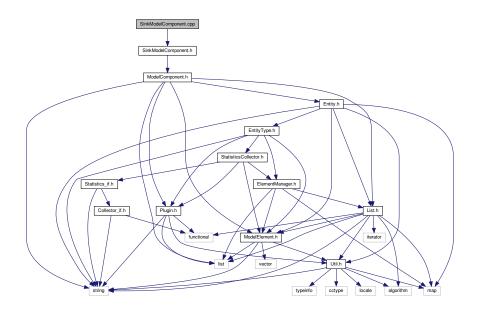


Classes

· class Simulator

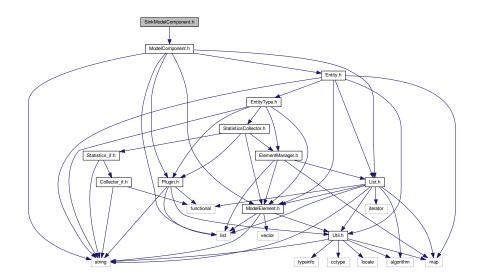
6.144 SinkModelComponent.cpp File Reference

#include "SinkModelComponent.h"
Include dependency graph for SinkModelComponent.cpp:

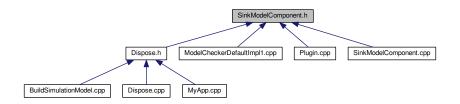


6.145 SinkModelComponent.h File Reference

#include "ModelComponent.h"
Include dependency graph for SinkModelComponent.h:



This graph shows which files directly or indirectly include this file:



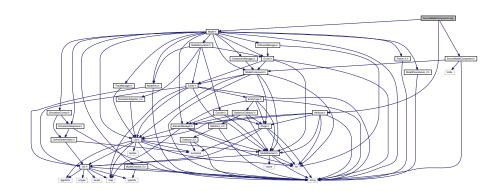
Classes

• class SinkModelComponent

6.146 SourceModelComponent.cpp File Reference

```
#include "SourceModelComponent.h"
#include "Model.h"
#include "Attribute.h"
```

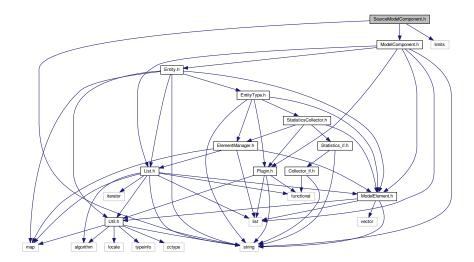
 $Include\ dependency\ graph\ for\ SourceModelComponent.cpp:$



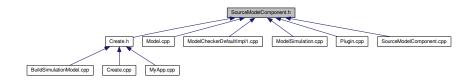
6.147 SourceModelComponent.h File Reference

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

Include dependency graph for SourceModelComponent.h:



This graph shows which files directly or indirectly include this file:



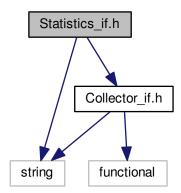
Classes

• class SourceModelComponent

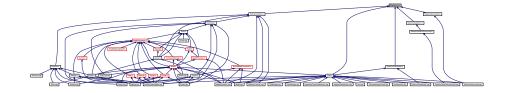
6.148 Statistics_if.h File Reference

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

Include dependency graph for Statistics_if.h:



This graph shows which files directly or indirectly include this file:

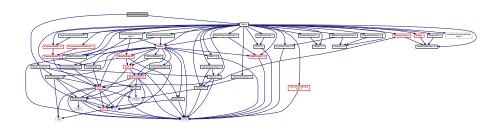


Classes

· class Statistics_if

6.149 StatisticsCollector.cpp File Reference

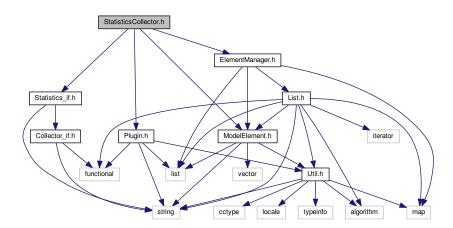
```
#include "StatisticsCollector.h"
#include "Traits.h"
Include dependency graph for StatisticsCollector.cpp:
```



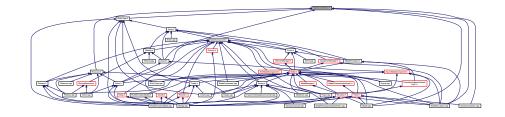
6.150 StatisticsCollector.h File Reference

```
#include "ModelElement.h"
#include "Statistics_if.h"
#include "ElementManager.h"
#include "Plugin.h"
```

Include dependency graph for StatisticsCollector.h:



This graph shows which files directly or indirectly include this file:



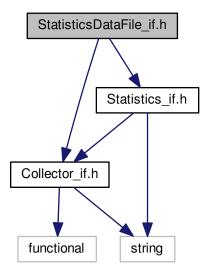
Classes

· class StatisticsCollector

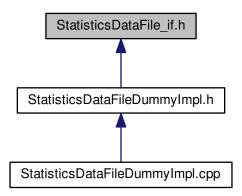
6.151 StatisticsDataFile_if.h File Reference

```
#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

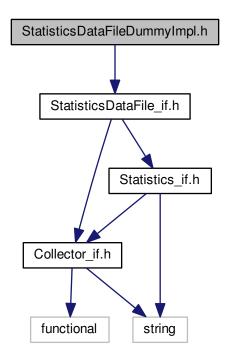
class StatisticsDatafile_if

6.152 StatisticsDataFileDummyImpl.cpp File Reference

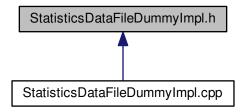
#include "StatisticsDataFileDummyImpl.h"
#include "Traits.h"
Include dependency graph for StatisticsDataFileDummyImpl.cpp:

6.153 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:

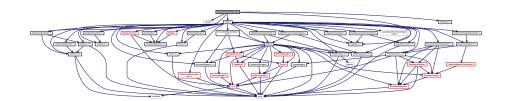


Classes

class StatisticsDataFileDummyImpl

6.154 StatisticsDefaultImpl1.cpp File Reference

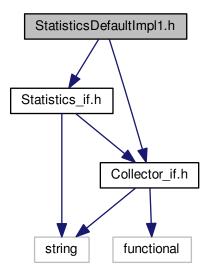
```
#include <complex>
#include "StatisticsDefaultImpl1.h"
#include "Traits.h"
#include "Integrator_if.h"
#include "ProbDistrib.h"
Include dependency graph for StatisticsDefaultImpl1.cpp:
```



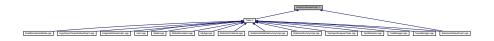
6.155 StatisticsDefaultImpl1.h File Reference

```
#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:

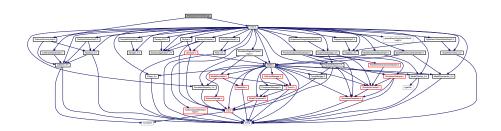


Classes

• class StatisticsDefaultImpl1

6.156 StatisticsDummyImpl.cpp File Reference

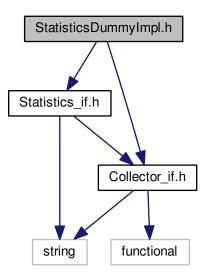
#include "StatisticsDummyImpl.h"
#include "Traits.h"
Include dependency graph for StatisticsDummyImpl.cpp:



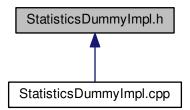
6.157 StatisticsDummyImpl.h File Reference

```
#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:



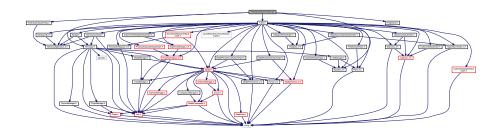
Classes

· class StatisticsDummyImpl

6.158 TestInputAnalyserTools.cpp File Reference

```
#include "TestInputAnalyserTools.h"
#include "Simulator.h"
#include "Sampler_if.h"
#include "ProbDistrib.h"
#include "Traits.h"
```

Include dependency graph for TestInputAnalyserTools.cpp:



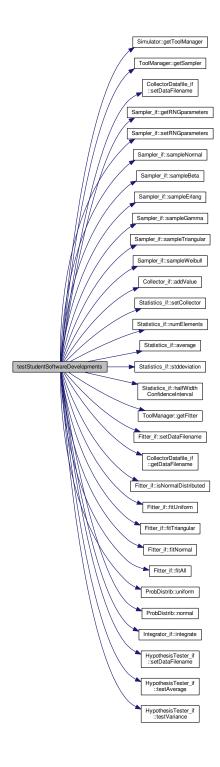
Functions

• void testStudentSoftwareDevelopments ()

6.158.1 Function Documentation

6.158.1.1 void testStudentSoftwareDevelopments ()

Here is the call graph for this function:

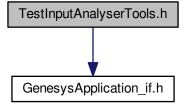


Here is the caller graph for this function:

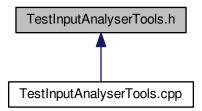


6.159 TestInputAnalyserTools.h File Reference

#include "GenesysApplication_if.h"
Include dependency graph for TestInputAnalyserTools.h:



This graph shows which files directly or indirectly include this file:



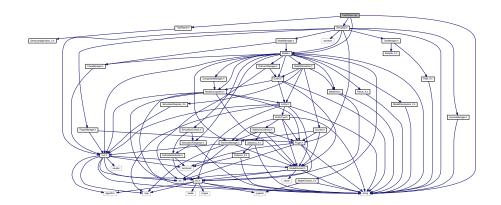
Classes

class TestInputAnalyserTools

6.160 TestParser.cpp File Reference

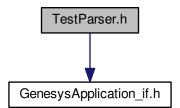
```
#include <string>
#include "TestParser.h"
#include "Model.h"
#include "Simulator.h"
```

Include dependency graph for TestParser.cpp:

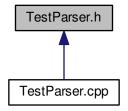


6.161 TestParser.h File Reference

#include "GenesysApplication_if.h"
Include dependency graph for TestParser.h:



This graph shows which files directly or indirectly include this file:

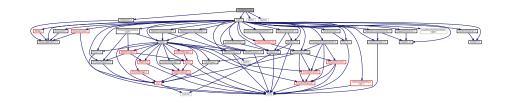


Classes

class TestParser

6.162 TestStatistics.cpp File Reference

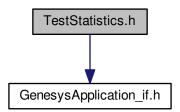
```
#include "TestStatistics.h"
#include <fstream>
#include <iostream>
#include "Traits.h"
Include dependency graph for TestStatistics.cpp:
```



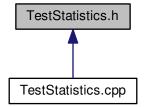
6.163 TestStatistics.h File Reference

#include "GenesysApplication_if.h"

Include dependency graph for TestStatistics.h:



This graph shows which files directly or indirectly include this file:

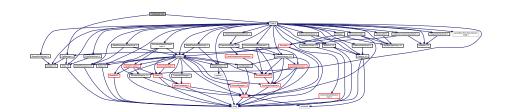


Classes

• class TestStatistics

6.164 ToolManager.cpp File Reference

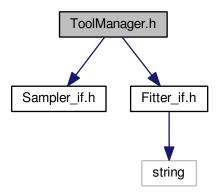
```
#include "ToolManager.h"
#include "Traits.h"
Include dependency graph for ToolManager.cpp:
```



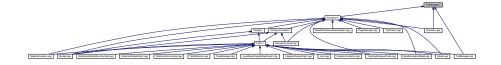
6.165 ToolManager.h File Reference

```
#include "Sampler_if.h"
#include "Fitter_if.h"
```

Include dependency graph for ToolManager.h:



This graph shows which files directly or indirectly include this file:

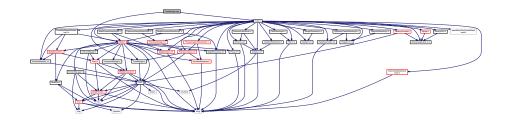


Classes

class ToolManager

TraceManager.cpp File Reference 6.166

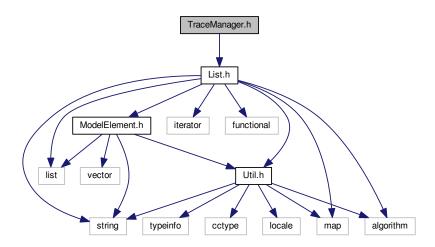
```
#include "TraceManager.h"
#include "Traits.h"
Include dependency graph for TraceManager.cpp:
```



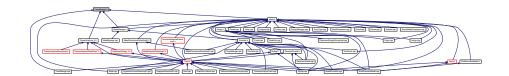
6.167 TraceManager.h File Reference

#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)

6.167.1 Typedef Documentation

```
6.167.1.1 typedef void(* traceErrorListener) (TraceErrorEvent)
```

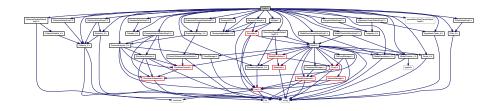
6.167.1.2 typedef void(* traceListener) (TraceEvent)

6.167.1.3 typedef void(* traceSimulationListener) (TraceSimulationEvent)

6.167.1.4 typedef void(* traceSimulationProcessListener) (TraceSimulationProcess)

6.168 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 "MyApp.h"
#include "GenesysGUI.h"
#include "GenesysConsole.h"
#include "CollectorDefaultImpl1.h"
#include "CollectorDatafileDefaultImpl1.h"
#include "StatisticsDefaultImpl1.h"
#include "IntegratorDefaultImpl1.h"
#include "HypothesisTesterDefaultImpl1.h"
#include "SamplerDefaultImpl1.h"
#include "SimulationReporterDefaultImpl1.h"
#include "ModelCheckerDefaultImpl1.h"
#include "ModelPersistenceDefaultImpl1.h"
#include "parserBisonFlex/ParserDefaultImpl2.h"
#include "ExperimentDesignDefaultImpl1.h"
#include "ProcessAnalyserDefaultImpl1.h"
#include "FitterDefaultImpl1.h"
Include dependency graph for Traits.h:
```



This graph shows which files directly or indirectly include this file:



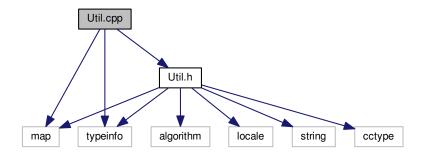
Classes

- 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 >

6.169 Util.cpp File Reference

```
#include "Util.h"
#include <typeinfo>
#include <map>
```

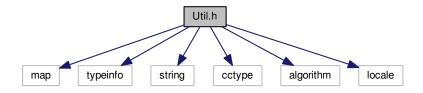
Include dependency graph for Util.cpp:



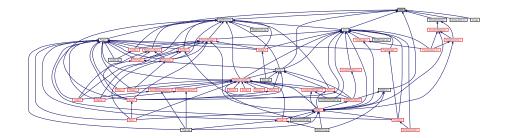
6.170 Util.h File Reference

```
#include <map>
#include <typeinfo>
#include <string>
#include <cctype>
#include <algorithm>
#include <locale>
```

Include dependency graph for Util.h:



This graph shows which files directly or indirectly include this file:



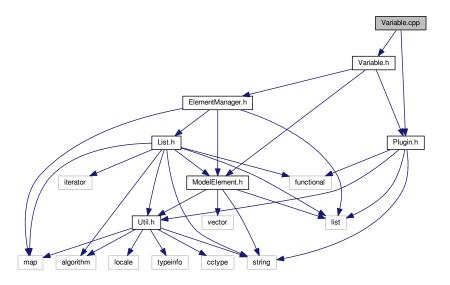
Classes

• class Util

6.171 Variable.cpp File Reference

```
#include "Variable.h"
#include "Plugin.h"
```

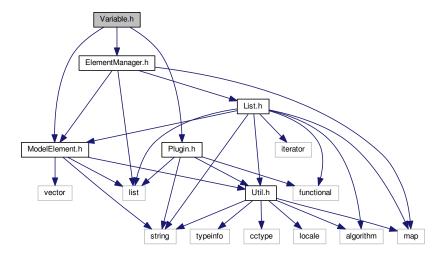
Include dependency graph for Variable.cpp:



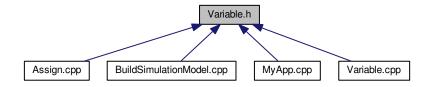
6.172 Variable.h File Reference

```
#include "ModelElement.h"
#include "ElementManager.h"
#include "Plugin.h"
```

Include dependency graph for Variable.h:



This graph shows which files directly or indirectly include this file:

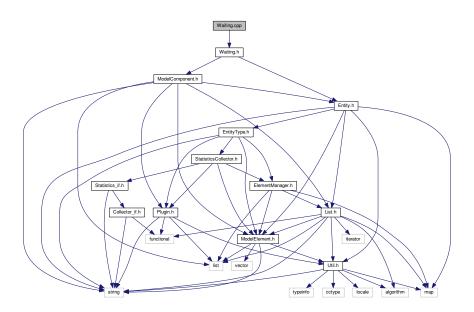


Classes

• class Variable

6.173 Waiting.cpp File Reference

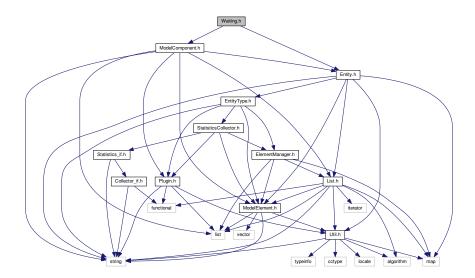
#include "Waiting.h"
Include dependency graph for Waiting.cpp:



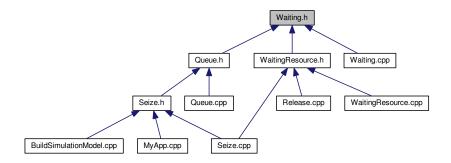
6.174 Waiting.h File Reference

#include "Entity.h"
#include "ModelComponent.h"

Include dependency graph for Waiting.h:



This graph shows which files directly or indirectly include this file:



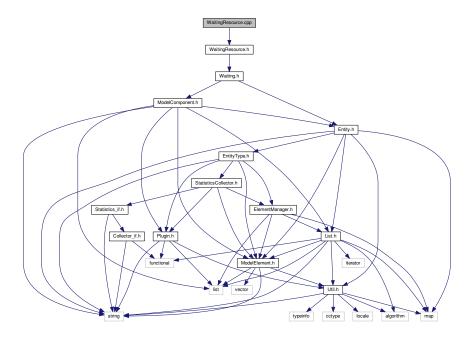
Classes

· class Waiting

6.175 WaitingResource.cpp File Reference

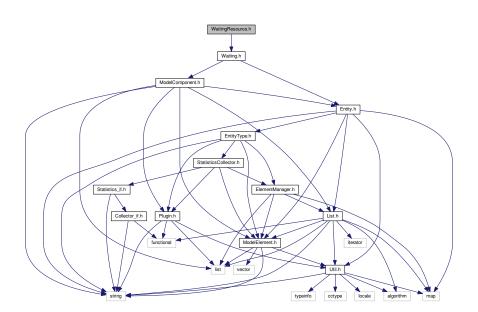
#include "WaitingResource.h"

Include dependency graph for WaitingResource.cpp:

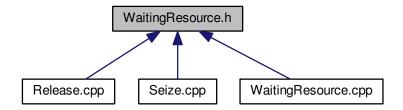


6.176 WaitingResource.h File Reference

#include "Waiting.h"
Include dependency graph for WaitingResource.h:



This graph shows which files directly or indirectly include this file:



Classes

• class WaitingResource