

eProxima RPCDDS C++ API

Version 0.4.0

Generated by Doxygen 1.8.6

Mon Dec 1 2014 17:19:28

Contents

1	eProsima RPC over DDS	1
2	Module Index	3
2.1	Modules	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Class Index	7
4.1	Class List	7
5	Module Documentation	11
5.1	eProsima RPCDDS API Reference	11
5.1.1	Detailed Description	11
5.2	Client Module	12
5.2.1	Detailed Description	12
5.3	Server Module	13
5.3.1	Detailed Description	13
5.4	Exceptions	14
5.4.1	Detailed Description	14
5.5	Strategies	15
5.5.1	Detailed Description	15
5.6	Transports	16
5.6.1	Detailed Description	17
5.7	Protocols	18
5.7.1	Detailed Description	18
5.8	Generated API example for eProsima RPCDDS	19
5.8.1	Detailed Description	19
6	Class Documentation	21
6.1	eprosima::rpc::transport::AsyncTask Class Reference	21
6.1.1	Detailed Description	21
6.2	eprosima::rpc::transport::dds::AsyncThread Class Reference	21

6.2.1	Detailed Description	22
6.2.2	Member Function Documentation	22
6.2.2.1	addTask	22
6.2.2.2	deleteAssociatedAsyncTasks	22
6.2.2.3	init	22
6.3	eprosima::rpc::exception::BadParamException Class Reference	22
6.3.1	Detailed Description	23
6.3.2	Constructor & Destructor Documentation	23
6.3.2.1	BadParamException	23
6.3.2.2	BadParamException	23
6.3.2.3	BadParamException	23
6.3.3	Member Function Documentation	24
6.3.3.1	operator=	24
6.3.3.2	operator=	24
6.4	eprosima::rpc::exception::ClientInternalException Class Reference	24
6.4.1	Detailed Description	25
6.4.2	Constructor & Destructor Documentation	25
6.4.2.1	ClientInternalException	25
6.4.2.2	ClientInternalException	25
6.4.2.3	ClientInternalException	25
6.4.3	Member Function Documentation	25
6.4.3.1	operator=	25
6.4.3.2	operator=	25
6.5	eprosima::rpc::transport::dds::DDSAsyncTask Class Reference	26
6.5.1	Detailed Description	26
6.5.2	Member Function Documentation	26
6.5.2.1	execute	26
6.5.2.2	getProcedureEndpoint	27
6.5.2.3	getReplyInstance	27
6.5.2.4	on_exception	27
6.5.2.5	setProcedureEndpoint	27
6.6	eprosima::rpc::transport::Endpoint Class Reference	27
6.6.1	Detailed Description	28
6.7	eprosima::rpc::exception::Exception Class Reference	28
6.7.1	Detailed Description	28
6.7.2	Constructor & Destructor Documentation	29
6.7.2.1	Exception	29
6.7.2.2	Exception	30
6.7.3	Member Function Documentation	30
6.7.3.1	operator=	30

6.7.3.2	operator=	30
6.8	FooDDS::Foo Interface Reference	30
6.8.1	Detailed Description	30
6.9	FooDDS::Foo_Call Class Reference	31
6.9.1	Detailed Description	32
6.9.2	Constructor & Destructor Documentation	32
6.9.2.1	Foo_Call	32
6.9.2.2	Foo_Call	32
6.9.3	Member Function Documentation	32
6.9.3.1	_d	32
6.9.3.2	_d	32
6.9.3.3	_d	32
6.9.3.4	deserialize	32
6.9.3.5	FooProcedure	33
6.9.3.6	FooProcedure	33
6.9.3.7	FooProcedure	33
6.9.3.8	FooProcedure	33
6.9.3.9	getMaxCdrSerializedSize	33
6.9.3.10	getSerializedSize	33
6.9.3.11	operator=	34
6.9.3.12	operator=	34
6.9.3.13	serialize	34
6.10	FooDDS::Foo_CallPlugin Class Reference	34
6.10.1	Detailed Description	34
6.11	FooDDS::Foo_FooProcedure_In Class Reference	35
6.11.1	Detailed Description	35
6.11.2	Constructor & Destructor Documentation	35
6.11.2.1	Foo_FooProcedure_In	35
6.11.3	Member Function Documentation	36
6.11.3.1	deserialize	36
6.11.3.2	getMaxCdrSerializedSize	36
6.11.3.3	getSerializedSize	36
6.11.3.4	operator=	36
6.11.3.5	operator=	36
6.11.3.6	serialize	37
6.12	FooDDS::Foo_FooProcedure_Out Class Reference	37
6.12.1	Detailed Description	38
6.12.2	Constructor & Destructor Documentation	38
6.12.2.1	Foo_FooProcedure_Out	38
6.12.2.2	Foo_FooProcedure_Out	38

6.12.3	Member Function Documentation	38
6.12.3.1	deserialize	38
6.12.3.2	getMaxCdrSerializedSize	38
6.12.3.3	getSerializedSize	38
6.12.3.4	operator=	39
6.12.3.5	operator=	39
6.12.3.6	serialize	39
6.13	FooDDS::Foo_FooProcedure_Result Class Reference	39
6.13.1	Detailed Description	40
6.13.2	Constructor & Destructor Documentation	40
6.13.2.1	Foo_FooProcedure_Result	40
6.13.2.2	Foo_FooProcedure_Result	41
6.13.3	Member Function Documentation	41
6.13.3.1	_d	41
6.13.3.2	_d	41
6.13.3.3	_d	41
6.13.3.4	deserialize	41
6.13.3.5	getMaxCdrSerializedSize	41
6.13.3.6	getSerializedSize	42
6.13.3.7	operator=	42
6.13.3.8	operator=	42
6.13.3.9	out_	42
6.13.3.10	out_	42
6.13.3.11	out_	43
6.13.3.12	out_	43
6.13.3.13	serialize	43
6.14	FooDDS::Foo_FooProcedureCallbackHandler Class Reference	43
6.14.1	Detailed Description	44
6.14.2	Member Function Documentation	44
6.14.2.1	FooProcedure	44
6.14.2.2	on_exception	44
6.15	FooDDS::Foo_FooProcedureTask Class Reference	44
6.15.1	Detailed Description	45
6.15.2	Constructor & Destructor Documentation	45
6.15.2.1	Foo_FooProcedureTask	45
6.15.3	Member Function Documentation	45
6.15.3.1	getObject	45
6.15.3.2	getReplyInstance	45
6.15.3.3	on_exception	45
6.16	FooDDS::Foo_Reply Class Reference	46

6.16.1 Detailed Description	47
6.16.2 Constructor & Destructor Documentation	47
6.16.2.1 Foo_Reply	47
6.16.3 Member Function Documentation	47
6.16.3.1 deserialize	47
6.16.3.2 getMaxCdrSerializedSize	47
6.16.3.3 getSerializedSize	47
6.16.3.4 header	47
6.16.3.5 header	48
6.16.3.6 header	48
6.16.3.7 header	48
6.16.3.8 operator=	48
6.16.3.9 operator=	48
6.16.3.10 reply	48
6.16.3.11 reply	49
6.16.3.12 reply	49
6.16.3.13 reply	49
6.16.3.14 serialize	49
6.17 FooDDS::Foo_ReplyDataReader Class Reference	49
6.17.1 Detailed Description	50
6.18 FooDDS::Foo_ReplyDataWriter Class Reference	50
6.18.1 Detailed Description	50
6.19 FooDDS::Foo_ReplyPlugin Class Reference	50
6.19.1 Detailed Description	51
6.19.2 Member Function Documentation	51
6.19.2.1 register_type	51
6.20 FooDDS::Foo_Request Class Reference	51
6.20.1 Detailed Description	52
6.20.2 Constructor & Destructor Documentation	52
6.20.2.1 Foo_Request	52
6.20.2.2 Foo_Request	53
6.20.3 Member Function Documentation	53
6.20.3.1 deserialize	53
6.20.3.2 getMaxCdrSerializedSize	53
6.20.3.3 getSerializedSize	53
6.20.3.4 header	53
6.20.3.5 header	54
6.20.3.6 header	54
6.20.3.7 header	54
6.20.3.8 operator=	54

6.20.3.9	<code>operator=</code>	54
6.20.3.10	<code>request</code>	54
6.20.3.11	<code>request</code>	55
6.20.3.12	<code>request</code>	55
6.20.3.13	<code>request</code>	55
6.20.3.14	<code>serialize</code>	55
6.21	<code>FooDDS::Foo_RequestDataReader</code> Class Reference	55
6.21.1	Detailed Description	56
6.22	<code>FooDDS::Foo_RequestDataWriter</code> Class Reference	56
6.22.1	Detailed Description	56
6.23	<code>FooDDS::Foo_RequestPlugin</code> Class Reference	56
6.23.1	Detailed Description	57
6.23.2	Member Function Documentation	57
6.23.2.1	<code>register_type</code>	57
6.24	<code>FooDDS::Foo_Return</code> Class Reference	57
6.24.1	Detailed Description	58
6.24.2	Constructor & Destructor Documentation	58
6.24.2.1	<code>Foo_Return</code>	58
6.24.2.2	<code>Foo_Return</code>	59
6.24.3	Member Function Documentation	59
6.24.3.1	<code>_d</code>	59
6.24.3.2	<code>_d</code>	59
6.24.3.3	<code>_d</code>	59
6.24.3.4	<code>deserialize</code>	59
6.24.3.5	<code>FooProcedure</code>	59
6.24.3.6	<code>FooProcedure</code>	60
6.24.3.7	<code>FooProcedure</code>	60
6.24.3.8	<code>FooProcedure</code>	60
6.24.3.9	<code>getMaxCdrSerializedSize</code>	60
6.24.3.10	<code>getSerializedSize</code>	60
6.24.3.11	<code>operator=</code>	60
6.24.3.12	<code>operator=</code>	61
6.24.3.13	<code>serialize</code>	61
6.25	<code>FooDDS::Foo_ReturnPlugin</code> Class Reference	61
6.25.1	Detailed Description	61
6.26	<code>eprosima::rpc::protocol::dds::FooDDSProtocol</code> Class Reference	61
6.26.1	Detailed Description	62
6.26.2	Member Function Documentation	62
6.26.2.1	<code>activateInterface</code>	62
6.26.2.2	<code>FooDDS_Foo_serve</code>	62

6.26.2.3	setTransport	63
6.27	eprosima::rpc::protocol::FooDDSProtocol Class Reference	63
6.27.1	Detailed Description	64
6.27.2	Member Function Documentation	64
6.27.2.1	activateInterface	64
6.27.2.2	linkFooDDS_FoolImpl	64
6.27.2.3	setTransport	64
6.28	FooDDS::FooPlugin Class Reference	64
6.28.1	Detailed Description	65
6.29	FooDDS::FooPlugin::FooProcedure_InPlugin Class Reference	65
6.29.1	Detailed Description	65
6.30	FooDDS::FooPlugin::FooProcedure_OutPlugin Class Reference	65
6.30.1	Detailed Description	65
6.31	FooDDS::FooPlugin::FooProcedure_ResultPlugin Class Reference	66
6.31.1	Detailed Description	66
6.32	FooDDS::FooProxy Class Reference	66
6.32.1	Detailed Description	66
6.32.2	Constructor & Destructor Documentation	67
6.32.2.1	FooProxy	67
6.33	FooDDS::FooServer Class Reference	68
6.33.1	Detailed Description	68
6.33.2	Constructor & Destructor Documentation	68
6.33.2.1	FooServer	68
6.34	FooDDS::FooServerImpl Class Reference	69
6.34.1	Detailed Description	69
6.35	eprosima::rpc::protocol::dds::GUID_t Class Reference	69
6.35.1	Member Function Documentation	70
6.35.1.1	deserialize	70
6.35.1.2	getMaxCdrSerializedSize	70
6.35.1.3	serialize	70
6.36	eprosima::rpc::protocol::dds::GUID_tPlugin Class Reference	70
6.36.1	Member Function Documentation	70
6.36.1.1	get_typecode	70
6.37	eprosima::rpc::exception::IncompatibleException Class Reference	71
6.37.1	Detailed Description	71
6.37.2	Constructor & Destructor Documentation	71
6.37.2.1	IncompatibleException	71
6.37.2.2	IncompatibleException	72
6.37.2.3	IncompatibleException	72
6.37.3	Member Function Documentation	72

6.37.3.1	<code>operator=</code>	72
6.37.3.2	<code>operator=</code>	72
6.38	<code>eprosima::rpc::exception::InitializeException</code> Class Reference	72
6.38.1	Detailed Description	73
6.38.2	Constructor & Destructor Documentation	73
6.38.2.1	<code>InitializeException</code>	73
6.38.2.2	<code>InitializeException</code>	73
6.38.2.3	<code>InitializeException</code>	73
6.38.3	Member Function Documentation	74
6.38.3.1	<code>operator=</code>	74
6.38.3.2	<code>operator=</code>	74
6.39	<code>eprosima::rpc::protocol::Protocol</code> Class Reference	74
6.39.1	Detailed Description	75
6.39.2	Member Function Documentation	75
6.39.2.1	<code>_setTransport</code>	75
6.39.2.2	<code>getTransport</code>	75
6.39.2.3	<code>setTransport</code>	75
6.40	<code>eprosima::rpc::proxy::Proxy</code> Class Reference	75
6.40.1	Detailed Description	76
6.40.2	Constructor & Destructor Documentation	76
6.40.2.1	<code>Proxy</code>	76
6.40.3	Member Function Documentation	76
6.40.3.1	<code>getProtocol</code>	76
6.40.3.2	<code>getTransport</code>	76
6.41	<code>eprosima::rpc::transport::dds::ProxyProcedureEndpoint</code> Class Reference	76
6.41.1	Detailed Description	77
6.41.2	Constructor & Destructor Documentation	77
6.41.2.1	<code>ProxyProcedureEndpoint</code>	77
6.41.3	Member Function Documentation	77
6.41.3.1	<code>freeQuery</code>	77
6.41.3.2	<code>initialize</code>	78
6.41.3.3	<code>send</code>	78
6.41.3.4	<code>send_async</code>	78
6.42	<code>eprosima::rpc::transport::dds::ProxyTransport</code> Class Reference	79
6.42.1	Detailed Description	80
6.42.2	Constructor & Destructor Documentation	80
6.42.2.1	<code>ProxyTransport</code>	80
6.42.3	Member Function Documentation	80
6.42.3.1	<code>addAsyncTask</code>	80
6.42.3.2	<code>createProcedureEndpoint</code>	80

6.42.3.3	deleteAssociatedAsyncTasks	81
6.42.3.4	getRemoteServiceName	81
6.42.3.5	getTimeout	81
6.42.3.6	setTransport	81
6.43	eprosima::rpc::transport::ProxyTransport Class Reference	82
6.43.1	Detailed Description	82
6.43.2	Member Function Documentation	82
6.43.2.1	connect	82
6.43.2.2	getBehaviour	82
6.43.2.3	receive	82
6.43.2.4	send	83
6.44	eprosima::rpc::protocol::dds::ReplyHeader Class Reference	83
6.44.1	Detailed Description	84
6.44.2	Constructor & Destructor Documentation	84
6.44.2.1	ReplyHeader	84
6.44.2.2	ReplyHeader	84
6.44.3	Member Function Documentation	84
6.44.3.1	deserialize	84
6.44.3.2	getMaxCdrSerializedSize	84
6.44.3.3	operator=	85
6.44.3.4	operator=	85
6.44.3.5	request_id	85
6.44.3.6	request_id	85
6.44.3.7	request_id	85
6.44.3.8	request_id	85
6.44.3.9	serialize	86
6.45	eprosima::rpc::protocol::dds::ReplyHeaderPlugin Class Reference	86
6.45.1	Detailed Description	86
6.45.2	Member Function Documentation	86
6.45.2.1	get_typecode	86
6.46	eprosima::rpc::protocol::dds::RequestHeader Class Reference	86
6.46.1	Detailed Description	87
6.46.2	Constructor & Destructor Documentation	87
6.46.2.1	RequestHeader	87
6.46.2.2	RequestHeader	88
6.46.3	Member Function Documentation	88
6.46.3.1	deserialize	88
6.46.3.2	getMaxCdrSerializedSize	88
6.46.3.3	instance_name	88
6.46.3.4	instance_name	88

6.46.3.5	operator=	88
6.46.3.6	operator=	89
6.46.3.7	remote_service_name	89
6.46.3.8	remote_service_name	89
6.46.3.9	request_id	89
6.46.3.10	request_id	89
6.46.3.11	request_id	89
6.46.3.12	request_id	90
6.46.3.13	serialize	90
6.47	eprosima::rpc::protocol::dds::RequestHeaderPlugin Class Reference	90
6.47.1	Detailed Description	90
6.47.2	Member Function Documentation	90
6.47.2.1	get_typecode	90
6.48	eprosima::rpc::protocol::dds::SampleIdentity_t Class Reference	90
6.48.1	Detailed Description	91
6.48.2	Constructor & Destructor Documentation	91
6.48.2.1	SampleIdentity_t	91
6.48.2.2	SampleIdentity_t	92
6.48.3	Member Function Documentation	92
6.48.3.1	deserialize	92
6.48.3.2	getMaxCdrSerializedSize	92
6.48.3.3	guid	92
6.48.3.4	guid	92
6.48.3.5	operator=	92
6.48.3.6	operator=	93
6.48.3.7	sequence_number	93
6.48.3.8	sequence_number	93
6.48.3.9	sequence_number	93
6.48.3.10	serialize	93
6.49	eprosima::rpc::protocol::dds::SampleIdentity_tPlugin Class Reference	93
6.49.1	Member Function Documentation	94
6.49.1.1	get_typecode	94
6.50	eprosima::rpc::server::Server Class Reference	94
6.50.1	Detailed Description	94
6.50.2	Constructor & Destructor Documentation	95
6.50.2.1	Server	95
6.50.3	Member Function Documentation	96
6.50.3.1	process	96
6.51	eprosima::rpc::exception::ServerInternalException Class Reference	96
6.51.1	Detailed Description	97

6.51.2	Constructor & Destructor Documentation	97
6.51.2.1	ServerInternalException	97
6.51.2.2	ServerInternalException	97
6.51.2.3	ServerInternalException	97
6.51.3	Member Function Documentation	97
6.51.3.1	operator=	97
6.51.3.2	operator=	97
6.52	eprosima::rpc::exception::ServerNotFoundException Class Reference	98
6.52.1	Detailed Description	98
6.52.2	Constructor & Destructor Documentation	98
6.52.2.1	ServerNotFoundException	98
6.52.2.2	ServerNotFoundException	99
6.52.2.3	ServerNotFoundException	99
6.52.3	Member Function Documentation	99
6.52.3.1	operator=	99
6.52.3.2	operator=	99
6.53	eprosima::rpc::transport::dds::ServerProcedureEndpoint Class Reference	99
6.53.1	Detailed Description	100
6.53.2	Constructor & Destructor Documentation	100
6.53.2.1	ServerProcedureEndpoint	100
6.53.3	Member Function Documentation	101
6.53.3.1	getProcessFunc	101
6.53.3.2	initialize	101
6.53.3.3	sendReply	101
6.53.3.4	start	101
6.53.3.5	stop	101
6.54	eprosima::rpc::strategy::ServerStrategy Class Reference	102
6.54.1	Detailed Description	102
6.54.2	Member Function Documentation	102
6.54.2.1	getImpl	102
6.55	eprosima::rpc::strategy::ServerStrategyImpl Class Reference	102
6.55.1	Detailed Description	103
6.55.2	Member Function Documentation	103
6.55.2.1	schedule	103
6.56	eprosima::rpc::exception::ServerTimeoutException Class Reference	103
6.56.1	Detailed Description	104
6.56.2	Constructor & Destructor Documentation	104
6.56.2.1	ServerTimeoutException	104
6.56.2.2	ServerTimeoutException	104
6.56.2.3	ServerTimeoutException	104

6.56.3	Member Function Documentation	104
6.56.3.1	operator=	104
6.56.3.2	operator=	104
6.57	eprosima::rpc::transport::ServerTransport Class Reference	105
6.57.1	Detailed Description	105
6.57.2	Member Function Documentation	106
6.57.2.1	getBehaviour	106
6.57.2.2	getCallback	106
6.57.2.3	getLinkedProtocol	106
6.57.2.4	getStrategy	106
6.57.2.5	linkProtocol	106
6.57.2.6	receive	106
6.57.2.7	sendReply	107
6.57.2.8	setCallback	107
6.57.2.9	setStrategy	107
6.58	eprosima::rpc::transport::dds::ServerTransport Class Reference	107
6.58.1	Detailed Description	108
6.58.2	Constructor & Destructor Documentation	108
6.58.2.1	ServerTransport	108
6.58.3	Member Function Documentation	108
6.58.3.1	createProcedureEndpoint	109
6.58.3.2	process	109
6.58.3.3	sendReply	109
6.58.3.4	setTransport	109
6.59	eprosima::rpc::strategy::SingleThreadStrategy Class Reference	110
6.59.1	Detailed Description	110
6.59.2	Member Function Documentation	110
6.59.2.1	getImpl	110
6.60	eprosima::rpc::exception::SystemException Class Reference	111
6.60.1	Detailed Description	111
6.60.2	Constructor & Destructor Documentation	111
6.60.2.1	SystemException	111
6.60.2.2	SystemException	112
6.60.2.3	SystemException	112
6.60.2.4	SystemException	112
6.60.3	Member Function Documentation	112
6.60.3.1	minor	112
6.60.3.2	minor	112
6.60.3.3	operator=	112
6.60.3.4	operator=	113

6.60.3.5	what	113
6.61	eprosima::rpc::protocol::dds::SystemExceptionCodePlugin Class Reference	113
6.61.1	Member Function Documentation	113
6.61.1.1	get_typecode	113
6.62	eprosima::rpc::transport::dds::TCPProxyTransport Class Reference	113
6.62.1	Detailed Description	114
6.62.2	Constructor & Destructor Documentation	114
6.62.2.1	TCPProxyTransport	114
6.62.3	Member Function Documentation	114
6.62.3.1	setTransport	114
6.63	eprosima::rpc::transport::dds::TCPServerTransport Class Reference	114
6.63.1	Detailed Description	115
6.63.2	Constructor & Destructor Documentation	115
6.63.2.1	TCPServerTransport	115
6.63.3	Member Function Documentation	115
6.63.3.1	setTransport	115
6.64	eprosima::rpc::strategy::ThreadPerRequestStrategy Class Reference	116
6.64.1	Detailed Description	116
6.64.2	Member Function Documentation	116
6.64.2.1	getImpl	116
6.65	eprosima::rpc::strategy::ThreadPoolStrategy Class Reference	116
6.65.1	Detailed Description	117
6.65.2	Constructor & Destructor Documentation	117
6.65.2.1	ThreadPoolStrategy	117
6.65.3	Member Function Documentation	117
6.65.3.1	getImpl	117
6.66	eprosima::rpc::transport::dds::Transport Class Reference	117
6.66.1	Detailed Description	118
6.66.2	Constructor & Destructor Documentation	118
6.66.2.1	Transport	118
6.66.3	Member Function Documentation	118
6.66.3.1	createProcedureEndpoint	118
6.66.3.2	getParticipant	119
6.66.3.3	getPublisher	119
6.66.3.4	getSubscriber	119
6.66.3.5	setTransport	119
6.67	eprosima::rpc::transport::Transport Class Reference	120
6.67.1	Detailed Description	120
6.67.2	Member Function Documentation	120
6.67.2.1	getBehaviour	120

6.68	eprosima::rpc::transport::dds::UDPProxyTransport Class Reference	120
6.68.1	Detailed Description	121
6.68.2	Constructor & Destructor Documentation	121
6.68.2.1	UDPProxyTransport	121
6.68.2.2	UDPProxyTransport	121
6.68.3	Member Function Documentation	121
6.68.3.1	setTransport	121
6.69	eprosima::rpc::transport::dds::UDPServerTransport Class Reference	122
6.69.1	Detailed Description	122
6.69.2	Constructor & Destructor Documentation	122
6.69.2.1	UDPServerTransport	122
6.69.3	Member Function Documentation	122
6.69.3.1	setTransport	123
6.70	eprosima::rpc::protocol::dds::UnknownExceptionPlugin Class Reference	124
6.70.1	Member Function Documentation	124
6.70.1.1	get_typecode	124
6.71	eprosima::rpc::protocol::dds::UnknownOperationPlugin Class Reference	124
6.71.1	Member Function Documentation	124
6.71.1.1	get_typecode	124
6.72	eprosima::rpc::exception::UserException Class Reference	125
6.72.1	Detailed Description	125
6.72.2	Constructor & Destructor Documentation	125
6.72.2.1	UserException	125
6.72.2.2	UserException	125
6.72.3	Member Function Documentation	126
6.72.3.1	operator=	126
6.72.3.2	operator=	126
	Index	127

Chapter 1

eProsimas RPC over DDS

eProsimas RPCDDS Library



eProsimas

eProsimas RPC over DDS is a service invocation framework that enables to build distributed applications with minimal effort using the client/server paradigm. It makes transparent the remote procedure call to developer without the programmer explicitly coding the details for this remote interaction and allows developers to focus his efforts on their application logic.

eProsimas RPC over DDS provides an easy way to invoke remote procedures using DDS standard as communication middleware. DDS (Data Distribution Service for Real-Time Systems) is an OMG specification of a data centric publish/subscribe communication model among real time software applications. eProsimas RPC over DDS comes with all benefits that DDS standard provides as reliable and efficient communications for distributed real time systems.

eProsimas RPC over DDS also brings other features:

- Synchronous, asynchronous and one-way invocations. The synchronous invocation is the common invocation and it blocks the client's thread until the reply is received from the server. The asynchronous invocation sends the request to the server but it doesn't blocks the client's thread. In the asynchronous invocation the developer provides a callback object that will be invoked when the reply is received from the server. The one-way invocation is a fire-and-forget invocation where the client does not care about the success or failure of the invocation. The one-way invocation does not expect any reply from the server.
- eProsimas RPC over DDS provides several strategies for the server. These strategies define how the server acts when a new request is received. Current supported strategies are: single-thread strategy, thread-pool strategy and thread-per-request strategy. Single-thread strategy uses one thread for all incoming requests. Thread-pool strategy uses thread-pool's threads to process the incoming requests. Thread-per-request strategy creates a new thread for each new incoming request and this new thread will process the request.
- eProsimas RPC over DDS supports several transports that DDS will use in the communications. There are two available transports. An UDP transport that brings the powerful benefit of DDS discovery in a local network or a TCP transport that allows connections with public servers located in internet.
- For DDS developers, eProsimas RPC over DDS allows enhancing DDS with client/service communications. A developer that uses DDS in its distributed application will be able to use a service-oriented interaction too.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

eProsima RPCDDS API Reference	11
Client Module	12
Server Module	13
Strategies	15
Exceptions	14
Transports	16
Protocols	18
Generated API example for eProsima RPCDDS	19

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

eprosima::rpc::transport::AsyncTask	21
eprosima::rpc::transport::dds::DDSAsyncTask	26
FooDDS::Foo_FooProcedureTask	44
eprosima::rpc::transport::dds::AsyncThread	21
DDSDDataReader	
FooDDS::Foo_ReplyDataReader	49
FooDDS::Foo_RequestDataReader	55
DDSDDataReaderListener	
eprosima::rpc::transport::dds::ServerProcedureEndpoint	99
DDSDDataWriter	
FooDDS::Foo_ReplyDataWriter	50
FooDDS::Foo_RequestDataWriter	56
DDSTypeSupport	
FooDDS::Foo_ReplyPlugin	50
FooDDS::Foo_RequestPlugin	56
FooDDS::FooPlugin::FooProcedure_ResultPlugin	66
eprosima::rpc::transport::Endpoint	27
eprosima::rpc::transport::dds::ProxyProcedureEndpoint	76
eprosima::rpc::transport::dds::ServerProcedureEndpoint	99
exception	
eprosima::rpc::exception::Exception	28
eprosima::rpc::exception::SystemException	111
eprosima::rpc::exception::BadParamException	22
eprosima::rpc::exception::ClientInternalException	24
eprosima::rpc::exception::IncompatibleException	71
eprosima::rpc::exception::InitializeException	72
eprosima::rpc::exception::ServerInternalException	96
eprosima::rpc::exception::ServerNotFoundException	98
eprosima::rpc::exception::ServerTimeoutException	103
eprosima::rpc::exception::UserException	125
FooDDS::Foo	30
FooDDS::FooProxy	66
FooDDS::FooServerImpl	69
FooDDS::Foo_Call	31
FooDDS::Foo_CallPlugin	34
FooDDS::Foo_FooProcedure_In	35
FooDDS::Foo_FooProcedure_Out	37

FooDDS::Foo_FooProcedure_Result	39
FooDDS::Foo_FooProcedureCallbackHandler	43
FooDDS::Foo_Reply	46
FooDDS::Foo_Request	51
FooDDS::Foo_Return	57
FooDDS::Foo_ReturnPlugin	61
FooDDS::FooPlugin	64
FooDDS::FooPlugin::FooProcedure_InPlugin	65
FooDDS::FooPlugin::FooProcedure_OutPlugin	65
eprosima::rpc::protocol::dds::GUID_t	69
eprosima::rpc::protocol::dds::GUID_tPlugin	70
eprosima::rpc::protocol::Protocol	74
eprosima::rpc::protocol::FooDDSProtocol	63
eprosima::rpc::protocol::dds::FooDDSProtocol	61
eprosima::rpc::proxy::Proxy	75
FooDDS::FooProxy	66
eprosima::rpc::protocol::dds::ReplyHeader	83
eprosima::rpc::protocol::dds::ReplyHeaderPlugin	86
eprosima::rpc::protocol::dds::RequestHeader	86
eprosima::rpc::protocol::dds::RequestHeaderPlugin	90
eprosima::rpc::protocol::dds::SampleIdentity_t	90
eprosima::rpc::protocol::dds::SampleIdentity_tPlugin	93
eprosima::rpc::server::Server	94
FooDDS::FooServer	68
eprosima::rpc::strategy::ServerStrategy	102
eprosima::rpc::strategy::SingleThreadStrategy	110
eprosima::rpc::strategy::ThreadPerRequestStrategy	116
eprosima::rpc::strategy::ThreadPoolStrategy	116
eprosima::rpc::strategy::ServerStrategyImpl	102
eprosima::rpc::protocol::dds::SystemExceptionCodePlugin	113
eprosima::rpc::transport::dds::Transport	117
eprosima::rpc::transport::dds::ProxyTransport	79
eprosima::rpc::transport::dds::TCPProxyTransport	113
eprosima::rpc::transport::dds::UDPPProxyTransport	120
eprosima::rpc::transport::dds::ServerTransport	107
eprosima::rpc::transport::dds::TCPServerTransport	114
eprosima::rpc::transport::dds::UDPServerTransport	122
eprosima::rpc::transport::Transport	120
eprosima::rpc::transport::ProxyTransport	82
eprosima::rpc::transport::dds::ProxyTransport	79
eprosima::rpc::transport::ServerTransport	105
eprosima::rpc::transport::dds::ServerTransport	107
eprosima::rpc::protocol::dds::UnknownExceptionPlugin	124
eprosima::rpc::protocol::dds::UnknownOperationPlugin	124

Chapter 4

Class Index

4.1 Class List

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

eprosima::rpc::transport::AsyncTask	This class represents a asynchronous task created to wait the reply from the server in an asynchronous call	21
eprosima::rpc::transport::dds::AsyncThread	This class is a separated thread used to manage asynchronous tasks	21
eprosima::rpc::exception::BadParamException	This class is thrown as an exception when there is some bad parameter in a object	22
eprosima::rpc::exception::ClientInternalException	This class is thrown as an exception when there is an error in the proxy side	24
eprosima::rpc::transport::dds::DDSAsyncTask	This class represents a asynchronous task created to wait the reply from the server in an asynchronous call	26
eprosima::rpc::transport::Endpoint	This class represents an endpoint	27
eprosima::rpc::exception::Exception	This abstract class is used to create exceptions	28
FooDDS::Foo	This class represents the interface Foo defined by the user in the IDL file	30
FooDDS::Foo_Call	This class represents the union used in the DDS topic to encapsulate the operations in request samples	31
FooDDS::Foo_CallPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	34
FooDDS::Foo_FooProcedure_In	This class represents the structure Foo_FooProcedure_In that can be used to send/receive requests for the operation Foo::FooProcedure	35
FooDDS::Foo_FooProcedure_Out	This class encapsulates output parameters for operation Foo::FooProcedure	37
FooDDS::Foo_FooProcedure_Result	This class represents the structure Foo_FooProcedure_Result that can be used to send/receive replies for the operation Foo::FooProcedure	39
FooDDS::Foo_FooProcedureCallbackHandler	This abstract class defines the callbacks that eProsima RPC will call in an asynchronous call. These callback has to be implemented in a derived class	43
FooDDS::Foo_FooProcedureTask	This class represents a asynchronous task created to wait the reply of the procedure Foo::FooProcedure from the server in an asynchronous call	44

FooDDS::Foo_Reply	This class represents the structure Foo_Reply that can be used to send/receive replies for the interface Foo	46
FooDDS::Foo_ReplyDataReader	Reply DataReader	49
FooDDS::Foo_ReplyDataWriter	Reply DataWriter	50
FooDDS::Foo_ReplyPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	50
FooDDS::Foo_Request	This class represents the structure Foo_Request that can be used to send/receive requests for the interface Foo	51
FooDDS::Foo_RequestDataReader	Request DataReader	55
FooDDS::Foo_RequestDataWriter	Request DataWriter	56
FooDDS::Foo_RequestPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	56
FooDDS::Foo_Return	This class represents the union used in the DDS topic to encapsulate the operations in reply samples	57
FooDDS::Foo_ReturnPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	61
eprosima::rpc::protocol::dds::FooDDSProtocol	This class is responsible for serializing and deserializing the requests and responses of this application. It uses DDS	61
eprosima::rpc::protocol::FooDDSProtocol	Protocol base class for the specific application	63
FooDDS::FooPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	64
FooDDS::FooPlugin::FooProcedure_InPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	65
FooDDS::FooPlugin::FooProcedure_OutPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	65
FooDDS::FooPlugin::FooProcedure_ResultPlugin	This class encapsulates the methods used on DDS topics by DDS middleware	66
FooDDS::FooProxy	This class implements a specific server's proxy for the defined interface Foo	66
FooDDS::FooServer	This class implements a specific server for the defined interface Foo by user	68
FooDDS::FooServerImpl	This class is the skeleton of the servant and its remote procedures has to be implemented	69
eprosima::rpc::protocol::dds::GUID_t	69
eprosima::rpc::protocol::dds::GUID_tPlugin	70
eprosima::rpc::exception::IncompatibleException	This class is thrown as an exception when a selected protocol and transport are incompatible	71
eprosima::rpc::exception::InitializeException	This class is thrown as an exception when there is an error initializing an object	72
eprosima::rpc::protocol::Protocol	This abstract class represents the protocol used by the RPCs. It serializes and deserializes the information and uses a eprosima::rpc::transport::Transport to send it and receive it	74
eprosima::rpc::proxy::Proxy	This class implements the common functionalities that all server's proxies have	75
eprosima::rpc::transport::dds::ProxyProcedureEndpoint	This class represents a remote endpoint used by a proxy. It also encapsulates the DDS datawriter and the DDS datareader	76
eprosima::rpc::transport::dds::ProxyTransport	This class is the base of all proxies that implement a transport using DDS	79

eprosima::rpc::transport::ProxyTransport	
This interface is the base of all classes that implement a transport that can be used by the proxy	82
eprosima::rpc::protocol::dds::ReplyHeader	
Header information used in all generated reply topics	83
eprosima::rpc::protocol::dds::ReplyHeaderPlugin	
This class offers the functions needed by DDS middleware to use the class ReplyHeaderPlugin	86
eprosima::rpc::protocol::dds::RequestHeader	
Header information used in all generated request topics	86
eprosima::rpc::protocol::dds::RequestHeaderPlugin	
This class offers the functions needed by DDS middleware to use the class RequestHeader-Plugin	90
eprosima::rpc::protocol::dds::SampleIdentity_t	
This class is used to identify clients	90
eprosima::rpc::protocol::dds::SampleIdentity_tPlugin	93
eprosima::rpc::server::Server	
This class implements the common functionalities that any server has	94
eprosima::rpc::exception::ServerInternalException	
This class is thrown as an exception when there is an error in the server side	96
eprosima::rpc::exception::ServerNotFoundException	
This class is thrown as an exception when the server is not found	98
eprosima::rpc::transport::dds::ServerProcedureEndpoint	
This class represents a remote endpoint used by a proxy. Also this class encapsulate the DDS datawriter and the DDS datareader	99
eprosima::rpc::strategy::ServerStrategy	
This class is the base of all classes that implement a server strategy. that could be used by the server	102
eprosima::rpc::strategy::ServerStrategyImpl	
This class is the base of all classes that implement a server strategy. that could be used by the server	102
eprosima::rpc::exception::ServerTimeoutException	
This class is thrown as an exception when the remote procedure call exceeds the maximum time	103
eprosima::rpc::transport::ServerTransport	
This interface is the base of all classes that implement a transport that can be used by the server	105
eprosima::rpc::transport::dds::ServerTransport	
This class is the base of all classes that implement a transport using DDS. This transport can be used by the servers	107
eprosima::rpc::strategy::SingleThreadStrategy	
This class implements the sigle thread strategy. The server uses a reception thread to execute all the requests	110
eprosima::rpc::exception::SystemException	
This abstract class is used to create internal FASTRPC exceptions	111
eprosima::rpc::protocol::dds::SystemExceptionCodePlugin	113
eprosima::rpc::transport::dds::TCPProxyTransport	
This class implements a transport using DDS over TCPv4. This transport can only be used by a server proxy	113
eprosima::rpc::transport::dds::TCPServerTransport	
This class implements a transport using DDS over TCPv4. This transport can only be used by a server	114
eprosima::rpc::strategy::ThreadPerRequestStrategy	
This class implements the thread per request strategy. The server creates a new thread for every new incoming request	116
eprosima::rpc::strategy::ThreadPoolStrategy	
This class implements a thread pool strategy. The server schedules the incoming requests in a free thread of the thread pool	116
eprosima::rpc::transport::dds::Transport	
This class is the base of all classes that implement a transport using DDS. This transport could be used by both proxies and servers	117

eprosima::rpc::transport::Transport	
This class is the base of all classes that implement a transport that could be used by the proxy or the server	120
eprosima::rpc::transport::dds::UDPProxyTransport	
This class implements a transport using DDS over UDPv4. This transport only can be used by a server's proxy	120
eprosima::rpc::transport::dds::UDPServerTransport	
This class implements transport using DDS over UDPv4. This transport can only be used by a server	122
eprosima::rpc::protocol::dds::UnknownExceptionPlugin	124
eprosima::rpc::protocol::dds::UnknownOperationPlugin	124
eprosima::rpc::exception::UserException	
This abstract class is used to create user exceptions	125

Chapter 5

Module Documentation

5.1 eProsima RPCDDS API Reference

eProsima RPC over DDS internal API grouped in modules.

Collaboration diagram for eProsima RPCDDS API Reference:

Modules

- [Client Module](#)

This group contains related API to create a client application. This API is used by the tool `rpcddsgengen` to generate custom code based in a defined interface. That interface and its remote procedure have to be defined in IDL language.

- [Server Module](#)

This group contains related API to create a server application. Except the custom server's strategies, this API is used by the tool `rpcddsgen` to generate custom code based in a defined interface. That interface and its remote procedure have to be defined in IDL language.

- [Exceptions](#)

Exceptions used by the eProsima RPCDDS API. All exceptions defined in this module are thrown by the eProsima RPCDDS library and the code generated by the tool `rpcddsgen`.

- [Transports](#)

Network transports that eProsima RPCDDS library offers. These transports define how a connection is established between a proxy and a server.

- [Protocols](#)

Protocols used by the RPCs. They define how to serialize and deserialize the information and use a `eprosima::rpc::transport::Transport` to send it and receive it.

5.1.1 Detailed Description

eProsima RPC over DDS internal API grouped in modules.

5.2 Client Module

This group contains related API to create a client application. This API is used by the tool *rpcddsgengen* to generate custom code based in a defined interface. That interface and its remote procedure have to be defined in IDL language.

Collaboration diagram for Client Module:

Classes

- class [eprosima::rpc::proxy::Proxy](#)

This class implements the common functionalities that all server's proxies have.

5.2.1 Detailed Description

This group contains related API to create a client application. This API is used by the tool *rpcddsgengen* to generate custom code based in a defined interface. That interface and its remote procedure have to be defined in IDL language.

5.3 Server Module

This group contains related API to create a server application. Except the custom server's strategies, this API is used by the tool *rpcddsgen* to generate custom code based in a defined interface. That interface and its remote procedure have to be defined in IDL language.

Collaboration diagram for Server Module:

Modules

- [Strategies](#)

Server's strategies that can be used in the server application. These strategies define how the server schedules a incoming request.

Classes

- class [eprosima::rpc::server::Server](#)

This class implements the common functionalities that any server has.

- class [eprosima::rpc::strategy::ServerStrategy](#)

This class is the base of all classes that implement a server strategy. that could be used by the server.

- class [eprosima::rpc::strategy::ServerStrategyImpl](#)

This class is the base of all classes that implement a server strategy. that could be used by the server.

5.3.1 Detailed Description

This group contains related API to create a server application. Except the custom server's strategies, this API is used by the tool *rpcddsgen* to generate custom code based in a defined interface. That interface and its remote procedure have to be defined in IDL language.

5.4 Exceptions

Exceptions used by the eProsima RPCDDS API. All exceptions defined in this module are thrown by the eProsima RPCDDS library and the code generated by the tool *rpcddsgen*.

Collaboration diagram for Exceptions:

Classes

- class [eprosima::rpc::exception::BadParamException](#)
This class is thrown as an exception when there is some bad parameter in a object.
- class [eprosima::rpc::exception::ClientInternalException](#)
This class is thrown as an exception when there is an error in the proxy side.
- class [eprosima::rpc::exception::Exception](#)
This abstract class is used to create exceptions.
- class [eprosima::rpc::exception::IncompatibleException](#)
This class is thrown as an exception when a selected protocol and transport are incompatible.
- class [eprosima::rpc::exception::InitializeException](#)
This class is thrown as an exception when there is an error initializing an object.
- class [eprosima::rpc::exception::ServerInternalException](#)
This class is thrown as an exception when there is an error in the server side.
- class [eprosima::rpc::exception::ServerNotFoundException](#)
This class is thrown as an exception when the server is not found.
- class [eprosima::rpc::exception::ServerTimeoutException](#)
This class is thrown as an exception when the remote procedure call exceeds the maximum time.
- class [eprosima::rpc::exception::SystemException](#)
This abstract class is used to create internal FASTRPC exceptions.
- class [eprosima::rpc::exception::UserException](#)
This abstract class is used to create user exceptions.

5.4.1 Detailed Description

Exceptions used by the eProsima RPCDDS API. All exceptions defined in this module are thrown by the eProsima RPCDDS library and the code generated by the tool *rpcddsgen*.

5.5 Strategies

Server's strategies that can be used in the server application. These strategies define how the server schedules a incoming request.

Collaboration diagram for Strategies:

Classes

- class [eprosima::rpc::strategy::SingleThreadStrategy](#)
This class implements the sigle thread strategy. The server uses a reception thread to execute all the requests.
- class [eprosima::rpc::strategy::ThreadPerRequestStrategy](#)
This class implements the thread per request strategy. The server creates a new thread for every new incoming request.
- class [eprosima::rpc::strategy::ThreadPoolStrategy](#)
This class implements a thread pool strategy. The server schedules the incoming requests in a free thread of the thread pool.

5.5.1 Detailed Description

Server's strategies that can be used in the server application. These strategies define how the server schedules a incoming request.

5.6 Transports

Network transports that eProsima RPCDDS library offers. These transports define how a connection is established between a proxy and a server.

Collaboration diagram for Transports:

Classes

- class [eprosima::rpc::transport::AsyncTask](#)
This class represents a asynchronous task created to wait the reply from the server in an asynchronous call.
- class [eprosima::rpc::transport::Endpoint](#)
This class represents an endpoint.
- class [eprosima::rpc::transport::dds::AsyncThread](#)
This class is a separated thread used to manage asynchronous tasks.
- class [eprosima::rpc::transport::dds::ProxyProcedureEndpoint](#)
This class represents a remote endpoint used by a proxy. It also encapsulates the DDS datawriter and the DDS datareader.
- class [eprosima::rpc::transport::dds::ServerProcedureEndpoint](#)
This class represents a remote endpoint used by a proxy. Also this class encapsulate the DDS datawriter and the DDS datareader.
- class [eprosima::rpc::transport::dds::DDSAsyncTask](#)
This class represents a asynchronous task created to wait the reply from the server in an asynchronous call.
- class [eprosima::rpc::transport::dds::ProxyTransport](#)
This class is the base of all proxies that implement a transport using DDS.
- class [eprosima::rpc::transport::dds::ServerTransport](#)
This class is the base of all classes that implement a transport using DDS. This transport can be used by the servers.
- class [eprosima::rpc::transport::dds::TCPProxyTransport](#)
This class implements a transport using DDS over TCPv4. This transport can only be used by a server proxy.
- class [eprosima::rpc::transport::dds::TCPServerTransport](#)
This class implements a transport using DDS over TCPv4. This transport can only be used by a server.
- class [eprosima::rpc::transport::dds::Transport](#)
This class is the base of all classes that implement a transport using DDS. This transport could be used by both proxies and servers.
- class [eprosima::rpc::transport::dds::UDPPProxyTransport](#)
This class implements a transport using DDS over UDPv4. This transport only can be used by a server's proxy.
- class [eprosima::rpc::transport::dds::UDPServerTransport](#)
This class implements transport using DDS over UDPv4. This transport can only be used by a server.

Typedefs

- typedef enum
[eprosima::rpc::transport::TransportBehaviour](#) [eprosima::rpc::transport::TransportBehaviour](#)
This enumeration specifies the behaviour of the transport.

Enumerations

- enum [eprosima::rpc::transport::TransportBehaviour](#) { **PROXY_BEHAVIOUR**, **SERVER_BEHAVIOUR** }
This enumeration specifies the behaviour of the transport.

5.6.1 Detailed Description

Network transports that eProsima RPCDDS library offers. These transports define how a connection is established between a proxy and a server.

5.7 Protocols

Protocols used by the RPCs. They define how to serialize and deserialize the information and use a [eprosima::rpc::transport::Transport](#) to send it and receive it.

Collaboration diagram for Protocols:

Classes

- class [eprosima::rpc::protocol::dds::SampleIdentity_t](#)
This class is used to identify clients.
- class [eprosima::rpc::protocol::dds::RequestHeader](#)
Header information used in all generated request topics.
- class [eprosima::rpc::protocol::dds::ReplyHeader](#)
Header information used in all generated reply topics.
- class [eprosima::rpc::protocol::dds::RequestHeaderPlugin](#)
This class offers the functions needed by DDS middleware to use the class [RequestHeaderPlugin](#).
- class [eprosima::rpc::protocol::dds::ReplyHeaderPlugin](#)
This class offers the functions needed by DDS middleware to use the class [ReplyHeaderPlugin](#).
- class [eprosima::rpc::protocol::Protocol](#)
This abstract class represents the protocol used by the RPCs. It serializes and deserializes the information and uses a [eprosima::rpc::transport::Transport](#) to send it and receive it.

5.7.1 Detailed Description

Protocols used by the RPCs. They define how to serialize and deserialize the information and use a [eprosima::rpc::transport::Transport](#) to send it and receive it.

5.8 Generated API example for eProsima RPCDDS

This group contains the generated API by the tool *rpcddsgen* for a DDS example of an interface named *Foo*.

Classes

- interface [FooDDS::Foo](#)
This class represents the interface [Foo](#) defined by the user in the IDL file.
- class [FooDDS::Foo_FooProcedureCallbackHandler](#)
This abstract class defines the callbacks that eProsima RPC will call in an asynchronous call. These callback has to be implemented in a derived class.
- class [FooDDS::Foo_FooProcedureTask](#)
This class represents a asynchronous task created to wait the reply of the procedure [Foo::FooProcedure](#) from the server in an asynchronous call.
- class [eprosima::rpc::protocol::dds::FooDDSProtocol](#)
This class is responsible for serializing and deserializing the requests and responses of this application. It uses DDS.
- class [FooDDS::FooProxy](#)
This class implements a specific server's proxy for the defined interface [Foo](#).
- class [FooDDS::FooServer](#)
This class implements a specific server for the defined interface [Foo](#) by user.
- class [FooDDS::FooServerImpl](#)
This class is the skeleton of the servant and its remote procedures has to be implemented.
- class [FooDDS::Foo_FooProcedure_In](#)
This class represents the structure [Foo_FooProcedure_In](#) that can be used to send/receive requests for the operation [Foo::FooProcedure](#).
- class [FooDDS::Foo_FooProcedure_Out](#)
This class encapsulates output paramaters for operation [Foo::FooProcedure](#).
- class [FooDDS::Foo_FooProcedure_Result](#)
This class represents the structure [Foo_FooProcedure_Result](#) that can be used to send/receive replies for the operation [Foo::FooProcedure](#).
- class [FooDDS::Foo_Call](#)
This class represents the union used in the DDS topic to encapsulate the operations in request samples.
- class [FooDDS::Foo_Request](#)
This class represents the structure [Foo_Request](#) that can be used to send/receive requests for the interface [Foo](#).
- class [FooDDS::Foo_Return](#)
This class represents the union used in the DDS topic to encapsulate the operations in reply samples.
- class [FooDDS::Foo_Reply](#)
This class represents the structure [Foo_Reply](#) that can be used to send/receive replies for the interface [Foo](#).

5.8.1 Detailed Description

This group contains the generated API by the tool *rpcddsgen* for a DDS example of an interface named *Foo*.

Chapter 6

Class Documentation

6.1 eprosima::rpc::transport::AsyncTask Class Reference

This class represents a asynchronous task created to wait the reply from the server in an asynchronous call.

```
#include <AsyncTask.h>
```

Inheritance diagram for eprosima::rpc::transport::AsyncTask:

Protected Member Functions

- [AsyncTask](#) ()
Default constructor.
- virtual [~AsyncTask](#) ()
Destructor.

6.1.1 Detailed Description

This class represents a asynchronous task created to wait the reply from the server in an asynchronous call.

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

- includetmp/rpcdds/transports/AsyncTask.h

6.2 eprosima::rpc::transport::dds::AsyncThread Class Reference

This class is a separated thread used to manage asynchronous tasks.

```
#include <AsyncThread.h>
```

Public Member Functions

- [AsyncThread](#) ()
Default constructor.
- int [init](#) ()
This function initializes all internal objects.
- void [exit](#) ()
This function deletes the internal objects.

- int [addTask](#) (DDSQueryCondition *query, [DDSAsyncTask](#) *task, long timeout)
This function adds a new asynchronous task.
- void [deleteAssociatedAsyncTasks](#) ([ProxyProcedureEndpoint](#) *pe)
This function deletes all the asynchronous tasks associated with the [ProxyProcedureEndpoint](#) endpoint.

6.2.1 Detailed Description

This class is a separated thread used to manage asynchronous tasks.

6.2.2 Member Function Documentation

6.2.2.1 int `eprosima::rpc::transport::dds::AsyncThread::addTask (DDSQueryCondition * query, DDSAsyncTask * task, long timeout)`

This function adds a new asynchronous task.

Parameters

<i>query</i>	Associated DDS::QueryCondition to the asynchronous task. Cannot be NULL.
<i>task</i>	The new asynchronous task. Cannot be NULL.
<i>timeout</i>	The time in milliseconds to wait for the reply.

Returns

0 if the function succesfully works. -1 in other case

6.2.2.2 void `eprosima::rpc::transport::dds::AsyncThread::deleteAssociatedAsyncTasks (ProxyProcedureEndpoint * pe)`

This function deletes all the asynchronous tasks associated with the [ProxyProcedureEndpoint](#) endpoint.

Parameters

<i>pe</i>	Pointer to the ProxyProcedureEndpoint . It cannot be NULL.
-----------	--

6.2.2.3 int `eprosima::rpc::transport::dds::AsyncThread::init ()`

This function initializes all internal objects.

Returns

0 value is returned if all the objects was succesfully created. -1 in other case

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

- `includetmp/rpcdds/transport/dds/AsyncThread.h`

6.3 eprosima::rpc::exception::BadParamException Class Reference

This class is thrown as an exception when there is some bad paremeter in a object.

```
#include <BadParamException.h>
```

Inheritance diagram for `eprosima::rpc::exception::BadParamException`:

Collaboration diagram for `eprosima::rpc::exception::BadParamException`:

Public Member Functions

- FASTRPC_DIIAPI [BadParamException](#) (const std::string &message)
Default constructor.
- FASTRPC_DIIAPI [BadParamException](#) (const [BadParamException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [BadParamException](#) ([BadParamException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [BadParamException](#) & operator= (const [BadParamException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [BadParamException](#) & operator= ([BadParamException](#) &&ex)
Assignment operation.
- virtual FASTRPC_DIIAPI ~[BadParamException](#) () throw ()
Default destructor.
- virtual FASTRPC_DIIAPI void [raise](#) () const
This function throws the object as an exception.

Additional Inherited Members

6.3.1 Detailed Description

This class is thrown as an exception when there is some bad parameter in a object.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 FASTRPC_DIIAPI eprosimarpc::exception::BadParamException::BadParamException (const std::string & message) [inline]

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.3.2.2 FASTRPC_DIIAPI eprosimarpc::exception::BadParamException::BadParamException (const [BadParamException](#) & ex)

Default copy constructor.

Parameters

<i>ex</i>	BadParamException that will be copied.
-----------	--

6.3.2.3 FASTRPC_DIIAPI eprosimarpc::exception::BadParamException::BadParamException ([BadParamException](#) && ex)

Default move constructor.

Parameters

<i>ex</i>	BadParamException that will be moved.
-----------	---

6.3.3 Member Function Documentation

6.3.3.1 FASTRPC_DIIAPI [BadParamException](#)& eprosima::rpc::exception::BadParamException::operator= (const [BadParamException](#) & *ex*)

Assignment operation.

Parameters

<i>ex</i>	BadParamException that will be copied.
-----------	--

6.3.3.2 FASTRPC_DIIAPI [BadParamException](#)& eprosima::rpc::exception::BadParamException::operator= ([BadParamException](#) && *ex*)

Assignment operation.

Parameters

<i>ex</i>	BadParamException that will be moved.
-----------	---

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

- includetmp/rpcdds/exceptions/BadParamException.h

6.4 eprosima::rpc::exception::ClientInternalException Class Reference

This class is thrown as an exception when there is an error in the proxy side.

```
#include <ClientInternalException.h>
```

Inheritance diagram for eprosima::rpc::exception::ClientInternalException:

Collaboration diagram for eprosima::rpc::exception::ClientInternalException:

Public Member Functions

- FASTRPC_DIIAPI [ClientInternalException](#) (const std::string &message)
Default constructor.
- FASTRPC_DIIAPI [ClientInternalException](#) (const [ClientInternalException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [ClientInternalException](#) ([ClientInternalException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [ClientInternalException](#) & operator= (const [ClientInternalException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [ClientInternalException](#) & operator= ([ClientInternalException](#) &&ex)
Assignment operation.
- virtual FASTRPC_DIIAPI ~[ClientInternalException](#) () throw ()
Default constructor.
- virtual FASTRPC_DIIAPI void [raise](#) () const
This function throws the object as an exception.

Additional Inherited Members

6.4.1 Detailed Description

This class is thrown as an exception when there is an error in the proxy side.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 FASTRPC_DIIAPI eprosimarpc::exception::ClientInternalException::ClientInternalException (const std::string & message) [inline]

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.4.2.2 FASTRPC_DIIAPI eprosimarpc::exception::ClientInternalException::ClientInternalException (const ClientInternalException & ex)

Default copy constructor.

Parameters

<i>ex</i>	ClientInternalException that will be copied.
-----------	--

6.4.2.3 FASTRPC_DIIAPI eprosimarpc::exception::ClientInternalException::ClientInternalException (ClientInternalException && ex)

Default move constructor.

Parameters

<i>ex</i>	ClientInternalException that will be moved.
-----------	---

6.4.3 Member Function Documentation

6.4.3.1 FASTRPC_DIIAPI ClientInternalException& eprosimarpc::exception::ClientInternalException::operator= (const ClientInternalException & ex)

Assignment operation.

Parameters

<i>ex</i>	ClientInternalException that will be copied.
-----------	--

6.4.3.2 FASTRPC_DIIAPI ClientInternalException& eprosimarpc::exception::ClientInternalException::operator= (ClientInternalException && ex)

Assignment operation.

Parameters

<i>ex</i>	ClientInternalException that will be moved.
-----------	---

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

- `includetmp/rpcdds/exceptions/ClientInternalException.h`

6.5 eprosima::rpc::transport::dds::DDSAsyncTask Class Reference

This class represents a asynchronous task created to wait the reply from the server in an asynchronous call.

```
#include <DDSAsyncTask.h>
```

Inheritance diagram for `eprosima::rpc::transport::dds::DDSAsyncTask`:

Collaboration diagram for `eprosima::rpc::transport::dds::DDSAsyncTask`:

Public Member Functions

- [DDSAsyncTask](#) ()
default constructor
- virtual [~DDSAsyncTask](#) ()
default destructor
- void [execute](#) (DDSQueryCondition *query)
This function is called when the DDS WaitSet was wake up by the query condition of this asynchronous task. This function takes the reply.
- void [setProcedureEndpoint](#) (ProxyProcedureEndpoint *pe)
Sets the procedure endpoint.
- ProxyProcedureEndpoint * [getProcedureEndpoint](#) ()
Gets the procedure endpoint.
- virtual void [on_exception](#) (const [exception::SystemException](#) &ex)=0
This function executes the callback function when an exception occurs on the client's side. This function should be implemented by the generated asynchronous tasks.

Protected Member Functions

- virtual void [execute](#) ()=0
This function executes the callback functions when a reply is received or an exception was transmitted. This function should be implemented by the generated asynchronous tasks.
- virtual void * [getReplyInstance](#) ()=0
Returns the allocated memory that will be used when the reply is taken.

6.5.1 Detailed Description

This class represents a asynchronous task created to wait the reply from the server in an asynchronous call.

6.5.2 Member Function Documentation

6.5.2.1 void eprosima::rpc::transport::dds::DDSAsyncTask::execute (DDSQueryCondition * query)

This function is called when the DDS WaitSet was wake up by the query condition of this asynchronous task. This function takes the reply.

Parameters

<i>query</i>	Query condition associated with this asynchronous task.
--------------	---

6.5.2.2 ProxyProcedureEndpoint* eprosimas::rpc::transport::dds::DDSAsyncTask::getProcedureEndpoint ()

Gets the procedure endpoint.

Returns

Procedure endpoint with the DDS datawriter and datareader

6.5.2.3 virtual void* eprosimas::rpc::transport::dds::DDSAsyncTask::getReplyInstance () [protected], [pure virtual]

Returns the allocated memory that will be used when the reply is taken.

Returns

Pointer to the allocated memory.

Implemented in [FooDDS::Foo_FooProcedureTask](#).

6.5.2.4 virtual void eprosimas::rpc::transport::dds::DDSAsyncTask::on_exception (const exception::SystemException & ex) [pure virtual]

This function executes the callback function when an exception occurs on the client's side. This function should be implemented by the generated asynchronous tasks.

Parameters

<i>ex</i>	The exception that is sent to the user.
-----------	---

Implemented in [FooDDS::Foo_FooProcedureTask](#).

6.5.2.5 void eprosimas::rpc::transport::dds::DDSAsyncTask::setProcedureEndpoint (ProxyProcedureEndpoint * pe)

Sets the procedure endpoint.

Parameters

<i>pe</i>	Procedure endpoint with the DDS datawriter and datareader
-----------	---

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

- includetmp/rpcdds/transport/dds/DDSAsyncTask.h

6.6 eprosimas::rpc::transport::Endpoint Class Reference

This class represents an endpoint.

```
#include <Endpoint.h>
```

Inheritance diagram for eprosimas::rpc::transport::Endpoint:

Protected Member Functions

- [Endpoint](#) ()
Default constructor.
- virtual [~Endpoint](#) ()
Default destructor.

6.6.1 Detailed Description

This class represents an endpoint.

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

- includetmp/rpcdds/transports/components/Endpoint.h

6.7 eprosima::rpc::exception::Exception Class Reference

This abstract class is used to create exceptions.

```
#include <Exception.h>
```

Inheritance diagram for eprosima::rpc::exception::Exception:

Collaboration diagram for eprosima::rpc::exception::Exception:

Public Member Functions

- virtual [~Exception](#) () throw ()
Default destructor.
- virtual void [raise](#) () const =0
This function throws the object as exception.

Protected Member Functions

- [Exception](#) ()
Default constructor.
- [Exception](#) (const [Exception](#) &ex)
Default copy constructor.
- [Exception](#) ([Exception](#) &&ex)
Default move constructor.
- [Exception](#) & [operator=](#) (const [Exception](#) &ex)
Assignment operation.
- [Exception](#) & [operator=](#) ([Exception](#) &&)
Assignment operation.

6.7.1 Detailed Description

This abstract class is used to create exceptions.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 eprosima::rpc::exception::Exception::Exception (const Exception & ex) [protected]

Default copy constructor.

Parameters

<i>ex</i>	Exception that will be copied.
-----------	--

6.7.2.2 `eprosima::rpc::exception::Exception::Exception (Exception && ex)` `[protected]`

Default move constructor.

Parameters

<i>ex</i>	Exception that will be moved.
-----------	---

6.7.3 Member Function Documentation

6.7.3.1 `Exception& eprosima::rpc::exception::Exception::operator= (const Exception & ex)` `[protected]`

Assignment operation.

Parameters

<i>ex</i>	Exception that will be copied.
-----------	--

6.7.3.2 `Exception& eprosima::rpc::exception::Exception::operator= (Exception &&)` `[protected]`

Assignment operation.

Parameters

<i>ex</i>	Exception that will be moved.
-----------	---

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

- `includetmp/rpcdds/exceptions/Exception.h`

6.8 FooDDS::Foo Interface Reference

This class represents the interface [Foo](#) defined by the user in the IDL file.

```
#include <FooDDS.h>
```

Inheritance diagram for FooDDS::Foo:

Public Member Functions

- virtual void **FooProcedure** ()=0
- void **FooProcedure** ()

6.8.1 Detailed Description

This class represents the interface [Foo](#) defined by the user in the IDL file.

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

- `utils/doxygen/examples/dds/FooDDS.h`
- `utils/doxygen/examples/dds/FooDDS.idl`

6.9 FooDDS::Foo_Call Class Reference

This class represents the union used in the DDS topic to encapsulate the operations in request samples.

```
#include <FooDDSTopics.h>
```

Public Member Functions

- user_cpp_DllExport [Foo_Call](#) ()
Default constructor.
- user_cpp_DllExport [~Foo_Call](#) ()
Destructor.
- user_cpp_DllExport [Foo_Call](#) (const [Foo_Call](#) &x)
Copy constructor.
- user_cpp_DllExport [Foo_Call](#) ([Foo_Call](#) &&x)
Move constructor.
- user_cpp_DllExport [Foo_Call](#) & [operator=](#) (const [Foo_Call](#) &x)
Copy assignment.
- user_cpp_DllExport [Foo_Call](#) & [operator=](#) ([Foo_Call](#) &&x)
Move assignment.
- user_cpp_DllExport void [_d](#) (int32_t __d)
This function sets the discriminator value.
- user_cpp_DllExport int32_t [_d](#) () const
This function returns the value of the discriminator.
- user_cpp_DllExport int32_t & [_d](#) ()
This function returns a reference to the discriminator.
- user_cpp_DllExport void [unknown_operation](#) (eprosima::rpc::protocol::dds::UnknownOperation _unknown_operation)
- user_cpp_DllExport
eprosima::rpc::protocol::dds::UnknownOperation [unknown_operation](#) () const
- user_cpp_DllExport
eprosima::rpc::protocol::dds::UnknownOperation & [unknown_operation](#) ()
- user_cpp_DllExport void [FooProcedure](#) (const [Foo_FooProcedure_In](#) &_FooProcedure)
This function copies the value in member FooProcedure.
- user_cpp_DllExport void [FooProcedure](#) ([Foo_FooProcedure_In](#) &&_FooProcedure)
This function moves the value in member FooProcedure.
- user_cpp_DllExport const
[Foo_FooProcedure_In](#) & [FooProcedure](#) () const
This function returns a constant reference to member FooProcedure.
- user_cpp_DllExport
[Foo_FooProcedure_In](#) & [FooProcedure](#) ()
This function returns a reference to member FooProcedure.
- user_cpp_DllExport size_t [getSerializedSize](#) (size_t current_alignment=0) const
This function returns the serialized size of an object depending on the buffer alignment.
- user_cpp_DllExport void [serialize](#) (eprosima::fastcdr::Cdr &cdr) const
This function serializes an object using CDR serialization.
- user_cpp_DllExport void [deserialize](#) (eprosima::fastcdr::Cdr &cdr)
This function deserializes an object using CDR serialization.

Static Public Member Functions

- static user_cpp_DllExport size_t [getMaxCdrSerializedSize](#) (size_t current_alignment=0)
This function returns the maximum serialized size of an object depending on the buffer alignment.

6.9.1 Detailed Description

This class represents the union used in the DDS topic to encapsulate the operations in request samples.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 FooDDS::Foo_Call::Foo_Call (const Foo_Call & x)

Copy constructor.

Parameters

x	Reference to the object Foo_Call that will be copied.
---	---

6.9.2.2 FooDDS::Foo_Call::Foo_Call (Foo_Call && x)

Move constructor.

Parameters

x	Reference to the object Foo_Call that will be copied.
---	---

6.9.3 Member Function Documentation

6.9.3.1 void FooDDS::Foo_Call::_d (int32_t __d)

This function sets the discriminator value.

Parameters

__d	New value for the discriminator.
-----	----------------------------------

Exceptions

eprosima::rpc::exception::BadParamException	This exception is thrown if the new value doesn't correspond to the selected union member.
---	--

6.9.3.2 int32_t FooDDS::Foo_Call::_d () const

This function returns the value of the discriminator.

Returns

Value of the discriminator

6.9.3.3 int32_t & FooDDS::Foo_Call::_d ()

This function returns a reference to the discriminator.

Returns

Reference to the discriminator.

6.9.3.4 void FooDDS::Foo_Call::deserialize (eprosima::fastcdr::Cdr & cdr)

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.9.3.5 void FooDDS::Foo_Call::FooProcedure (const Foo_FooProcedure_In & _FooProcedure)

This function copies the value in member FooProcedure.

Parameters

<i>_FooProcedure</i>	New value to be copied in member FooProcedure
----------------------	---

6.9.3.6 void FooDDS::Foo_Call::FooProcedure (FooDDS::Foo_FooProcedure_In && _FooProcedure)

This function moves the value in member FooProcedure.

Parameters

<i>_FooProcedure</i>	New value to be moved in member FooProcedure
----------------------	--

6.9.3.7 const FooDDS::Foo_FooProcedure_In & FooDDS::Foo_Call::FooProcedure () const

This function returns a constant reference to member FooProcedure.

Returns

Constant reference to member FooProcedure

6.9.3.8 FooDDS::Foo_FooProcedure_In & FooDDS::Foo_Call::FooProcedure ()

This function returns a reference to member FooProcedure.

Returns

Reference to member FooProcedure

6.9.3.9 size_t FooDDS::Foo_Call::getMaxCdrSerializedSize (size_t current_alignment = 0) [static]

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.9.3.10 size_t FooDDS::Foo_Call::getSerializedSize (size_t current_alignment = 0) const

This function returns the serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Serialized size.

6.9.3.11 FooDDS::Foo_Call & FooDDS::Foo_Call::operator= (const Foo_Call & x)

Copy assignment.

Parameters

<i>x</i>	Reference to the object Foo_Call that will be copied.
----------	---

6.9.3.12 FooDDS::Foo_Call & FooDDS::Foo_Call::operator= (Foo_Call && x)

Move assignment.

Parameters

<i>x</i>	Reference to the object Foo_Call that will be copied.
----------	---

6.9.3.13 void FooDDS::Foo_Call::serialize (eprosima::fastcdr::Cdr & cdr) const

This function serializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

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

- `utils/doxygen/examples/dds/FooDDSTopics.h`
- `utils/doxygen/examples/dds/FooDDSTopics.cxx`

6.10 FooDDS::Foo_CallPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```

Static Public Member Functions

- static DDS_TypeCode * **get_typecode** ()

6.10.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`
- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.cxx`

6.11 FooDDS::Foo_FooProcedure_In Class Reference

This class represents the structure `Foo_FooProcedure_In` that can be used to send/receive requests for the operation `Foo::FooProcedure`.

```
#include <FooDDSTopics.h>
```

Public Member Functions

- `user_cpp_DllExport Foo_FooProcedure_In ()`
Default constructor.
- `user_cpp_DllExport ~Foo_FooProcedure_In ()`
Destructor.
- `user_cpp_DllExport Foo_FooProcedure_In (const Foo_FooProcedure_In &x)`
- `user_cpp_DllExport Foo_FooProcedure_In (Foo_FooProcedure_In &&x)`
Move constructor.
- `user_cpp_DllExport Foo_FooProcedure_In & operator= (const Foo_FooProcedure_In &x)`
Copy assignment.
- `user_cpp_DllExport Foo_FooProcedure_In & operator= (Foo_FooProcedure_In &&x)`
Copy assignment.
- `user_cpp_DllExport void dummy (int32_t _dummy)`
- `user_cpp_DllExport int32_t dummy () const`
- `user_cpp_DllExport int32_t & dummy ()`
- `user_cpp_DllExport size_t getSerializedSize (size_t current_alignment=0) const`
This function returns the serialized size of an object depending on the buffer alignment.
- `user_cpp_DllExport void serialize (eprosima::fastcdr::Cdr &cdr) const`
This function serializes an object using CDR serialization.
- `user_cpp_DllExport void deserialize (eprosima::fastcdr::Cdr &cdr)`
This function deserializes an object using CDR serialization.

Static Public Member Functions

- `static user_cpp_DllExport size_t getMaxCdrSerializedSize (size_t current_alignment=0)`
This function returns the maximum serialized size of an object depending on the buffer alignment.

6.11.1 Detailed Description

This class represents the structure `Foo_FooProcedure_In` that can be used to send/receive requests for the operation `Foo::FooProcedure`.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 FooDDS::Foo_FooProcedure_In::Foo_FooProcedure_In (Foo_FooProcedure_In && x)

Move constructor.

Parameters

<i>x</i>	Reference to the object Foo_FooProcedure_In that will be copied.
----------	--

6.11.3 Member Function Documentation

6.11.3.1 void FooDDS::Foo_FooProcedure_In::deserialize (eprosima::fastcdr::Cdr & *cdr*)

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.11.3.2 size_t FooDDS::Foo_FooProcedure_In::getMaxCdrSerializedSize (size_t *current_alignment* = 0) [static]

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.11.3.3 size_t FooDDS::Foo_FooProcedure_In::getSerializedSize (size_t *current_alignment* = 0) const

This function returns the serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Serialized size.

6.11.3.4 FooDDS::Foo_FooProcedure_In & FooDDS::Foo_FooProcedure_In::operator= (const Foo_FooProcedure_In & *x*)

Copy assignment.

Parameters

<i>x</i>	Reference to the object Foo_FooProcedure that will be copied.
----------	---

6.11.3.5 FooDDS::Foo_FooProcedure_In & FooDDS::Foo_FooProcedure_In::operator= (FooDDS::Foo_FooProcedure_In && *x*)

Copy assignment.

Parameters

<i>x</i>	Reference to the object Foo_FooProcedure that will be copied.
----------	---

6.11.3.6 void FooDDS::Foo_FooProcedure_In::serialize (eprosima::fastcdr::Cdr & cdr) const

This function serializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

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

- utils/doxygen/examples/dds/FooDDSTopics.h
- utils/doxygen/examples/dds/FooDDSTopics.cxx

6.12 FooDDS::Foo_FooProcedure_Out Class Reference

This class encapsulates output paramaters for operation Foo::FooProcedure.

```
#include <FooDDSTopics.h>
```

Public Member Functions

- user_cpp_DllExport [Foo_FooProcedure_Out](#) ()
Default constructor.
- user_cpp_DllExport [~Foo_FooProcedure_Out](#) ()
Destructor.
- user_cpp_DllExport [Foo_FooProcedure_Out](#) (const [Foo_FooProcedure_Out](#) &x)
Copy constructor.
- user_cpp_DllExport [Foo_FooProcedure_Out](#) ([Foo_FooProcedure_Out](#) &&x)
Move constructor.
- user_cpp_DllExport [Foo_FooProcedure_Out](#) & operator= (const [Foo_FooProcedure_Out](#) &x)
Copy assignment.
- user_cpp_DllExport [Foo_FooProcedure_Out](#) & operator= ([Foo_FooProcedure_Out](#) &&x)
Copy assignment.
- user_cpp_DllExport void **dummy** (int32_t _dummy)
- user_cpp_DllExport int32_t **dummy** () const
- user_cpp_DllExport int32_t & **dummy** ()
- user_cpp_DllExport size_t [getSerializedSize](#) (size_t current_alignment=0) const
This function returns the serialized size of an object depending on the buffer alignment.
- user_cpp_DllExport void [serialize](#) (eprosima::fastcdr::Cdr &cdr) const
This function serializes an object using CDR serialization.
- user_cpp_DllExport void [deserialize](#) (eprosima::fastcdr::Cdr &cdr)
This function deserializes an object using CDR serialization.

Static Public Member Functions

- static user_cpp_DllExport size_t [getMaxCdrSerializedSize](#) (size_t current_alignment=0)
This function returns the maximum serialized size of an object depending on the buffer alignment.

6.12.1 Detailed Description

This class encapsulates output paramaters for operation Foo::FooProcedure.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 FooDDS::Foo_FooProcedure_Out::Foo_FooProcedure_Out (const Foo_FooProcedure_Out & x)

Copy constructor.

Parameters

<i>x</i>	Reference to the object Foo_FooProcedure_Out that will be copied.
----------	---

6.12.2.2 FooDDS::Foo_FooProcedure_Out::Foo_FooProcedure_Out (Foo_FooProcedure_Out && x)

Move constructor.

Parameters

<i>x</i>	Reference to the object Foo_FooProcedure_Out that will be copied.
----------	---

6.12.3 Member Function Documentation

6.12.3.1 void FooDDS::Foo_FooProcedure_Out::deserialize (eprosima::fastcdr::Cdr & cdr)

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.12.3.2 size_t FooDDS::Foo_FooProcedure_Out::getMaxCdrSerializedSize (size_t current_alignment = 0) [static]

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.12.3.3 size_t FooDDS::Foo_FooProcedure_Out::getSerializedSize (size_t current_alignment = 0) const

This function returns the serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Serialized size.

6.12.3.4 FooDDS::Foo_FooProcedure_Out & FooDDS::Foo_FooProcedure_Out::operator= (const Foo_FooProcedure_Out & x)

Copy assignment.

Parameters

<code>x</code>	Reference to the object Foo_FooProcedure that will be copied.
----------------	---

6.12.3.5 FooDDS::Foo_FooProcedure_Out & FooDDS::Foo_FooProcedure_Out::operator= (FooDDS::Foo_FooProcedure_Out && x)

Copy assignment.

Parameters

<code>x</code>	Reference to the object Foo_FooProcedure that will be copied.
----------------	---

6.12.3.6 void FooDDS::Foo_FooProcedure_Out::serialize (eprosima::fastcdr::Cdr & cdr) const

This function serializes an object using CDR serialization.

Parameters

<code>cdr</code>	CDR serialization object.
------------------	---------------------------

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

- utils/doxygen/examples/dds/FooDDSTopics.h
- utils/doxygen/examples/dds/FooDDSTopics.cxx

6.13 FooDDS::Foo_FooProcedure_Result Class Reference

This class represents the structure [Foo_FooProcedure_Result](#) that can be used to send/receive replies for the operation Foo::FooProcedure.

```
#include <FooDDSTopics.h>
```

Public Member Functions

- user_cpp_DllExport [Foo_FooProcedure_Result](#) ()
Default constructor.
- user_cpp_DllExport [~Foo_FooProcedure_Result](#) ()
Destructor.
- user_cpp_DllExport [Foo_FooProcedure_Result](#) (const [Foo_FooProcedure_Result](#) &x)
Copy constructor.
- user_cpp_DllExport [Foo_FooProcedure_Result](#) ([Foo_FooProcedure_Result](#) &&x)
Move constructor.
- user_cpp_DllExport [Foo_FooProcedure_Result](#) & operator= (const [Foo_FooProcedure_Result](#) &x)

- Copy assignment.*
- user_cpp_DllExport [Foo_FooProcedure_Result](#) & operator= ([Foo_FooProcedure_Result](#) &&x)
- Copy assignment.*
- user_cpp_DllExport void [_d](#) (int32_t __d)
- This function sets the discriminator value.*
- user_cpp_DllExport int32_t [_d](#) () const
- This function returns the value of the discriminator.*
- user_cpp_DllExport int32_t & [_d](#) ()
- This function returns a reference to the discriminator.*
- user_cpp_DllExport void **unknown_exception** (eprosima::rpc::protocol::dds::UnknownException _unknown_exception)
- user_cpp_DllExport eprosima::rpc::protocol::dds::UnknownException **unknown_exception** () const
- user_cpp_DllExport eprosima::rpc::protocol::dds::UnknownException & **unknown_exception** ()
- user_cpp_DllExport void [out_](#) (const [Foo_FooProcedure_Out](#) &_out_)
- This function copies the value in member out_.*
- user_cpp_DllExport void [out_](#) ([Foo_FooProcedure_Out](#) &&_out_)
- This function moves the value in member out_.*
- user_cpp_DllExport const [Foo_FooProcedure_Out](#) & [out_](#) () const
- This function returns a constant reference to member out_.*
- user_cpp_DllExport [Foo_FooProcedure_Out](#) & [out_](#) ()
- This function returns a reference to member out_.*
- user_cpp_DllExport void **sysx_** (eprosima::rpc::ReturnMessage _sysx_)
- user_cpp_DllExport eprosima::rpc::ReturnMessage **sysx_** () const
- user_cpp_DllExport eprosima::rpc::ReturnMessage & **sysx_** ()
- user_cpp_DllExport size_t [getSerializedSize](#) (size_t current_alignment=0) const
- This function returns the serialized size of an object depending on the buffer alignment.*
- user_cpp_DllExport void [serialize](#) (eprosima::fastcdr::Cdr &cdr) const
- This function serializes an object using CDR serialization.*
- user_cpp_DllExport void [deserialize](#) (eprosima::fastcdr::Cdr &cdr)
- This function deserializes an object using CDR serialization.*

Static Public Member Functions

- static user_cpp_DllExport size_t [getMaxCdrSerializedSize](#) (size_t current_alignment=0)
- This function returns the maximum serialized size of an object depending on the buffer alignment.*

6.13.1 Detailed Description

This class represents the structure [Foo_FooProcedure_Result](#) that can be used to send/receive replies for the operation Foo::FooProcedure.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 FooDDS::Foo_FooProcedure_Result::Foo_FooProcedure_Result (const Foo_FooProcedure_Result & x)

Copy constructor.

Parameters

<i>x</i>	Reference to the object Foo_FooProcedure_Result that will be copied.
----------	--

6.13.2.2 FooDDS::Foo_FooProcedure_Result::Foo_FooProcedure_Result (Foo_FooProcedure_Result && x)

Move constructor.

Parameters

<i>x</i>	Reference to the object Foo_FooProcedure_Result that will be copied.
----------	--

6.13.3 Member Function Documentation

6.13.3.1 void FooDDS::Foo_FooProcedure_Result::_d (int32_t __d)

This function sets the discriminator value.

Parameters

<i>__d</i>	New value for the discriminator.
------------	----------------------------------

Exceptions

eprosima::rpc::exception::BadParamException	This exception is thrown if the new value doesn't correspond to the selected union member.
---	--

6.13.3.2 int32_t FooDDS::Foo_FooProcedure_Result::_d () const

This function returns the value of the discriminator.

Returns

Value of the discriminator

6.13.3.3 int32_t & FooDDS::Foo_FooProcedure_Result::_d ()

This function returns a reference to the discriminator.

Returns

Reference to the discriminator.

6.13.3.4 void FooDDS::Foo_FooProcedure_Result::deserialize (eprosima::fastcdr::Cdr & cdr)

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.13.3.5 size_t FooDDS::Foo_FooProcedure_Result::getMaxCdrSerializedSize (size_t current_alignment = 0) [static]

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.13.3.6 `size_t FooDDS::Foo_FooProcedure_Result::getSerializedSize (size_t current_alignment = 0) const`

This function returns the serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Serialized size.

6.13.3.7 `FooDDS::Foo_FooProcedure_Result & FooDDS::Foo_FooProcedure_Result::operator= (const Foo_FooProcedure_Result & x)`

Copy assignment.

Parameters

<i>x</i>	Reference to the object <code>Foo_FooProcedure</code> that will be copied.
----------	--

6.13.3.8 `FooDDS::Foo_FooProcedure_Result & FooDDS::Foo_FooProcedure_Result::operator= (FooDDS::Foo_FooProcedure_Result && x)`

Copy assignment.

Parameters

<i>x</i>	Reference to the object <code>Foo_FooProcedure</code> that will be copied.
----------	--

6.13.3.9 `void FooDDS::Foo_FooProcedure_Result::out_ (const Foo_FooProcedure_Out & _out_)`

This function copies the value in member `out_`.

Parameters

<i>out</i>	New value to be copied in member <code>out_</code>
---------------------------------	--

6.13.3.10 `void FooDDS::Foo_FooProcedure_Result::out_ (FooDDS::Foo_FooProcedure_Out && _out_)`

This function moves the value in member `out_`.

Parameters

<code>out</code>	New value to be moved in member out_
---------------------------------------	--------------------------------------

6.13.3.11 `const FooDDS::Foo_FooProcedure_Out & FooDDS::Foo_FooProcedure_Result::out_ () const`

This function returns a constant reference to member out_.

Returns

Constant reference to member out_

Exceptions

<code>eprosima::rpc::exception::BadParamException</code>	This exception is thrown if the requested union member is not the current selection.
--	--

6.13.3.12 `FooDDS::Foo_FooProcedure_Out & FooDDS::Foo_FooProcedure_Result::out_ ()`

This function returns a reference to member out_.

Returns

Reference to member out_

Exceptions

<code>eprosima::rpc::exception::BadParamException</code>	This exception is thrown if the requested union member is not the current selection.
--	--

6.13.3.13 `void FooDDS::Foo_FooProcedure_Result::serialize (eprosima::fastcdr::Cdr & cdr) const`

This function serializes an object using CDR serialization.

Parameters

<code>cdr</code>	CDR serialization object.
------------------	---------------------------

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

- `utils/doxygen/examples/dds/FooDDSTopics.h`
- `utils/doxygen/examples/dds/FooDDSTopics.cxx`

6.14 FooDDS::Foo_FooProcedureCallbackHandler Class Reference

This abstract class defines the callbacks that eProsima RPC will call in an asynchronous call. These callback has to be implemented in a derived class.

```
#include <FooDDSAsyncCallbackHandlers.h>
```

Public Member Functions

- virtual void [`FooProcedure`](#) ()=0
- virtual void [`on_exception`](#) (const [`eprosima::rpc::exception::SystemException`](#) &ex)=0

This function is called when an exception occurs. This exception can be launched in the server's side or in the client's side.

6.14.1 Detailed Description

This abstract class defines the callbacks that eProsimas RPC will call in an asynchronous call. These callback has to be implemented in a derived class.

6.14.2 Member Function Documentation

6.14.2.1 `virtual void FooDDS::Foo_FooProcedureCallbackHandler::FooProcedure () [pure virtual]`

This function is called when is received the reply from the server.

6.14.2.2 `virtual void FooDDS::Foo_FooProcedureCallbackHandler::on_exception (const eprosimas::rpc::exception::-SystemException & ex) [pure virtual]`

This function is called when an exception occurs. This exception can be launched in the server's side or in the client's side.

Parameters

<code>ex</code>	The exception that will be launched.
-----------------	--------------------------------------

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

- `utils/doxygen/examples/dds/FooDDSAsyncCallbackHandlers.h`

6.15 FooDDS::Foo_FooProcedureTask Class Reference

This class represents a asynchronous task created to wait the reply of the procedure `Foo::FooProcedure` from the server in an asynchronous call.

```
#include <FooDDSDDSASyncSupport.h>
```

Inheritance diagram for `FooDDS::Foo_FooProcedureTask`:

Collaboration diagram for `FooDDS::Foo_FooProcedureTask`:

Public Member Functions

- `Foo_FooProcedureTask (Foo_FooProcedureCallbackHandler &obj)`
The default constructor.
- `virtual ~Foo_FooProcedureTask ()`
Destructor.
- `virtual void execute ()`
This function is called when the reply sample is received.
- `virtual void on_exception (const eprosimas::rpc::exception::SystemException &ex)`
This function is called when an exception occurs. This exception can be launched in the server's side or in the client's side.
- `Foo_FooProcedureCallbackHandler & getObject ()`
This function returns the object used by the task.
- `virtual void * getReplyInstance ()`
This function returns the allocated reply sample.

Additional Inherited Members

6.15.1 Detailed Description

This class represents a asynchronous task created to wait the reply of the procedure Foo::FooProcedure from the server in an asynchronous call.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 FooDDS::Foo_FooProcedureTask::Foo_FooProcedureTask (Foo_FooProcedureCallbackHandler & *obj*)

The default constructor.

Parameters

<i>obj</i>	Object that implements the callbacks that FastRPC will call when the reply will be received or and exception will be launched.
<i>client</i>	Pointer to the server's proxy. Cannot be NULL.

6.15.3 Member Function Documentation

6.15.3.1 Foo_FooProcedureCallbackHandler & FooDDS::Foo_FooProcedureTask::getObject ()

This function returns the object used by the task.

Returns

The object that implements the callbacks.

6.15.3.2 void * FooDDS::Foo_FooProcedureTask::getReplyInstance () [virtual]

This function returns the allocated reply sample.

Returns

Pointer to the allocated reply sample.

Implements [eprosima::rpc::transport::dds::DDSAsyncTask](#).

6.15.3.3 void FooDDS::Foo_FooProcedureTask::on_exception (const eprosima::rpc::exception::SystemException & *ex*) [virtual]

This function is called when an exception occurs. This exception can be launched in the server's side or in the client's side.

Parameters

<i>ex</i>	The exception that will be launched.
-----------	--------------------------------------

Implements [eprosima::rpc::transport::dds::DDSAsyncTask](#).

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

- `utils/doxygen/examples/dds/FooDDSDDSAsyncSupport.h`
- `utils/doxygen/examples/dds/FooDDSDDSAsyncSupport.cxx`

6.16 FooDDS::Foo_Reply Class Reference

This class represents the structure [Foo_Reply](#) that can be used to send/receive replies for the interface [Foo](#).

```
#include <FooDDSTopics.h>
```

Public Member Functions

- user_cpp_DllExport [Foo_Reply](#) ()
Default constructor.
- user_cpp_DllExport [~Foo_Reply](#) ()
Destructor.
- user_cpp_DllExport **Foo_Reply** (const [Foo_Reply](#) &x)
- user_cpp_DllExport [Foo_Reply](#) ([Foo_Reply](#) &&x)
Move constructor.
- user_cpp_DllExport [Foo_Reply](#) & operator= (const [Foo_Reply](#) &x)
Copy assignment.
- user_cpp_DllExport [Foo_Reply](#) & operator= ([Foo_Reply](#) &&x)
Copy assignment.
- user_cpp_DllExport void [header](#) (const [eprosima::rpc::protocol::dds::ReplyHeader](#) &_header)
This method sets the reply header information.
- user_cpp_DllExport void [header](#) ([eprosima::rpc::protocol::dds::ReplyHeader](#) &&_header)
This method sets the reply header information.
- user_cpp_DllExport const [eprosima::rpc::protocol::dds::ReplyHeader](#) & [header](#) () const
This method returns the reply header information.
- user_cpp_DllExport [eprosima::rpc::protocol::dds::ReplyHeader](#) & [header](#) ()
This method returns the reply header information.
- user_cpp_DllExport void [reply](#) (const [Foo_Return](#) &_reply)
This method sets the union that encapsulates the interface operations.
- user_cpp_DllExport void [reply](#) ([Foo_Return](#) &&_reply)
This method sets the union that encapsulates the interface operations.
- user_cpp_DllExport const [Foo_Return](#) & [reply](#) () const
This method sets the union that encapsulates the interface operations.
- user_cpp_DllExport [Foo_Return](#) & [reply](#) ()
This method sets the union that encapsulates the interface operations.
- user_cpp_DllExport size_t [getSerializedSize](#) (size_t current_alignment=0) const
This function returns the serialized size of an object depending on the buffer alignment.
- user_cpp_DllExport void [serialize](#) ([eprosima::fastcdr::Cdr](#) &cdr) const
This function serializes an object using CDR serialization.
- user_cpp_DllExport void [deserialize](#) ([eprosima::fastcdr::Cdr](#) &cdr)
This function deserializes an object using CDR serialization.

Static Public Member Functions

- static user_cpp_DllExport size_t [getMaxCdrSerializedSize](#) (size_t current_alignment=0)
This function returns the maximum serialized size of an object depending on the buffer alignment.

6.16.1 Detailed Description

This class represents the structure [Foo_Reply](#) that can be used to send/receive replies for the interface [Foo](#).

6.16.2 Constructor & Destructor Documentation

6.16.2.1 FooDDS::Foo_Reply::Foo_Reply (Foo_Reply && x)

Move constructor.

Parameters

<i>x</i>	Reference to the object Foo_Reply that will be copied.
----------	--

6.16.3 Member Function Documentation

6.16.3.1 void FooDDS::Foo_Reply::deserialize (eprosima::fastcdr::Cdr & cdr)

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.16.3.2 size_t FooDDS::Foo_Reply::getMaxCdrSerializedSize (size_t *current_alignment* = 0) [static]

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.16.3.3 size_t FooDDS::Foo_Reply::getSerializedSize (size_t *current_alignment* = 0) const

This function returns the serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Serialized size.

6.16.3.4 user_cpp_DllExport void FooDDS::Foo_Reply::header (const eprosima::rpc::protocol::dds::ReplyHeader & _header) [inline]

This method sets the reply header information.

Parameters

<code>_header</code>	Reply header.
----------------------	---------------

6.16.3.5 `user_cpp_DllExport void FooDDS::Foo_Reply::header (eprosima::rpc::protocol::dds::ReplyHeader && _header) [inline]`

This method sets the reply header information.

Parameters

<code>_header</code>	Reply header.
----------------------	---------------

6.16.3.6 `user_cpp_DllExport const eprosima::rpc::protocol::dds::ReplyHeader& FooDDS::Foo_Reply::header () const [inline]`

This method returns the reply header information.

Returns

Reply header.

6.16.3.7 `user_cpp_DllExport eprosima::rpc::protocol::dds::ReplyHeader& FooDDS::Foo_Reply::header () [inline]`

This method returns the reply header information.

Returns

Reply header.

6.16.3.8 `FooDDS::Foo_Reply & FooDDS::Foo_Reply::operator= (const Foo_Reply & x)`

Copy assignment.

Parameters

<code>x</code>	Reference to the object Foo that will be copied.
----------------	--

6.16.3.9 `FooDDS::Foo_Reply & FooDDS::Foo_Reply::operator= (Foo_Reply && x)`

Copy assignment.

Parameters

<code>x</code>	Reference to the object Foo that will be copied.
----------------	--

6.16.3.10 `user_cpp_DllExport void FooDDS::Foo_Reply::reply (const Foo_Return & _reply) [inline]`

This method sets the union that encapsulates the interface operations.

Parameters

<code>_reply</code>	Union.
---------------------	--------

6.16.3.11 `user_cpp_DllExport void FooDDS::Foo_Reply::reply (Foo_Return && _reply) [inline]`

This method sets the union that encapsulates the interface operations.

Parameters

<code>_reply</code>	Union.
---------------------	--------

6.16.3.12 `user_cpp_DllExport const Foo_Return& FooDDS::Foo_Reply::reply () const [inline]`

This method sets the union that encapsulates the interface operations.

Returns

Union.

6.16.3.13 `user_cpp_DllExport Foo_Return& FooDDS::Foo_Reply::reply () [inline]`

This method sets the union that encapsulates the interface operations.

Returns

Union.

6.16.3.14 `void FooDDS::Foo_Reply::serialize (eprosima::fastcdr::Cdr & cdr) const`

This function serializes an object using CDR serialization.

Parameters

<code>cdr</code>	CDR serialization object.
------------------	---------------------------

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

- `utils/doxygen/examples/dds/FooDDSTopics.h`
- `utils/doxygen/examples/dds/FooDDSTopics.cxx`

6.17 FooDDS::Foo_ReplyDataReader Class Reference

Reply DataReader.

```
#include <FooDDSTopicsPlugin.h>
```

Inheritance diagram for FooDDS::Foo_ReplyDataReader:

Collaboration diagram for FooDDS::Foo_ReplyDataReader:

Public Member Functions

- **Foo_ReplyDataReader** (DDSDataReader *impl)

6.17.1 Detailed Description

Reply DataReader.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`

6.18 FooDDS::Foo_ReplyDataWriter Class Reference

Reply DataWriter.

```
#include <FooDDSTopicsPlugin.h>
```

Inheritance diagram for FooDDS::Foo_ReplyDataWriter:

Collaboration diagram for FooDDS::Foo_ReplyDataWriter:

Public Member Functions

- **Foo_ReplyDataWriter** (DDSDDataWriter *impl)

6.18.1 Detailed Description

Reply DataWriter.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`

6.19 FooDDS::Foo_ReplyPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```

Inheritance diagram for FooDDS::Foo_ReplyPlugin:

Collaboration diagram for FooDDS::Foo_ReplyPlugin:

Public Member Functions

- DDSDataReader * **create_datareaderl** (DDSDataReader *dataReader)
- DDS_ReturnCode_t **destroy_datareaderl** (DDSDataReader *dataReader)
- DDSDataWriter * **create_datawriterl** (DDSDataWriter *dataWriter)
- DDS_ReturnCode_t **destroy_datawriterl** (DDSDataWriter *dataWriter)

Static Public Member Functions

- static const char * **get_typename** ()
- static [FooDDS::Foo_Reply](#) * **create_data** (void)
- static void **destroy_data** ([FooDDS::Foo_Reply](#) *sample)
- static void **copy_data** ([FooDDS::Foo_Reply](#) *dst, const [FooDDS::Foo_Reply](#) *src)
- static unsigned int **get_serialized_sample_max_size** (PRESTypePluginEndpointData endpoint_data, RTI-Bool include_encapsulation, RTIEncapsulationId encapsulation_id, unsigned int current_alignment)

- static unsigned int **get_serialized_sample_size** (PRESTypePluginEndpointData endpoint_data, RTIBool include_encapsulation, RTIEncapsulationId encapsulation_id, unsigned int current_alignment, const [FooDDS::Foo_Reply](#) *sample)
- static unsigned int **get_serialized_sample_min_size** (PRESTypePluginEndpointData endpoint_data, RTIBool include_encapsulation, RTIEncapsulationId encapsulation_id, unsigned int current_alignment)
- static PRESTypePluginParticipantData **on_participant_attached** (void *registration_data, const struct PRESTypePluginParticipantInfo *participant_info, RTIBool top_level_registration, void *container_plugin_context, RTICdrTypeCode *typeCode)
- static void **on_participant_detached** (PRESTypePluginParticipantData participant_data)
- static PRESTypePluginEndpointData **on_endpoint_attached** (PRESTypePluginParticipantData participant_data, const struct PRESTypePluginEndpointInfo *endpoint_info, RTIBool top_level_registration, void *container_plugin_context)
- static void **on_endpoint_detached** (PRESTypePluginEndpointData endpoint_data)
- static RTIBool **copy_sample** (PRESTypePluginEndpointData endpoint_data, [FooDDS::Foo_Reply](#) *dst, const [FooDDS::Foo_Reply](#) *src)
- static RTIBool **serialize** (PRESTypePluginEndpointData endpoint_data, const [FooDDS::Foo_Reply](#) *sample, struct RTICdrStream *stream, RTIBool serialize_encapsulation, RTIEncapsulationId encapsulation_id, RTIBool serialize_sample, void *endpoint_plugin_qos)
- static RTIBool **deserialize** (PRESTypePluginEndpointData endpoint_data, [FooDDS::Foo_Reply](#) **sample, RTIBool *drop_sample, struct RTICdrStream *stream, RTIBool deserialize_encapsulation, RTIBool deserialize_sample, void *endpoint_plugin_qos)
- static PRESTypePluginKeyKind **get_key_kind** (void)
- static DDS_TypeCode * **get_typecode** ()
- static struct PRESTypePlugin * **new_plugin** (void)
- static void **delete_plugin** (struct PRESTypePlugin *plugin)
- static bool **register_type** (DDSDomainParticipant *participant, const char *type_name)

6.19.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

6.19.2 Member Function Documentation

6.19.2.1 **bool FooDDS::Foo_ReplyPlugin::register_type (DDSDomainParticipant * participant, const char * type_name)**
[static]

TODO Mover al transporte

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

- utils/doxygen/examples/dds/FooDDSTopicsPlugin.h
- utils/doxygen/examples/dds/FooDDSTopicsPlugin.cxx

6.20 FooDDS::Foo_Request Class Reference

This class represents the structure [Foo_Request](#) that can be used to send/receive requests for the interface [Foo](#).

```
#include <FooDDSTopics.h>
```

Public Member Functions

- user_cpp_DLlexport [Foo_Request](#) ()
Default constructor.

- user_cpp_DllExport [~Foo_Request](#) ()
Destructor.
- user_cpp_DllExport [Foo_Request](#) (const [Foo_Request](#) &x)
Copy constructor.
- user_cpp_DllExport [Foo_Request](#) ([Foo_Request](#) &&x)
Move constructor.
- user_cpp_DllExport [Foo_Request](#) & operator= (const [Foo_Request](#) &x)
Copy assignment.
- user_cpp_DllExport [Foo_Request](#) & operator= ([Foo_Request](#) &&x)
Copy assignment.
- user_cpp_DllExport void [header](#) (const [eprosima::rpc::protocol::dds::RequestHeader](#) &_header)
This method sets the request header information.
- user_cpp_DllExport void [header](#) ([eprosima::rpc::protocol::dds::RequestHeader](#) &&_header)
This method sets the request header information.
- user_cpp_DllExport const [eprosima::rpc::protocol::dds::RequestHeader](#) & [header](#) () const
This method returns the request header information.
- user_cpp_DllExport [eprosima::rpc::protocol::dds::RequestHeader](#) & [header](#) ()
This method returns the request header information.
- user_cpp_DllExport void [request](#) (const [Foo_Call](#) &_request)
This method sets the union that encapsulates the interface operations.
- user_cpp_DllExport void [request](#) ([Foo_Call](#) &&_request)
This method sets the union that encapsulates the interface operations.
- user_cpp_DllExport const [Foo_Call](#) & [request](#) () const
This method returns the union that encapsulates the interface operations.
- user_cpp_DllExport [Foo_Call](#) & [request](#) ()
This method returns the union that encapsulates the interface operations.
- user_cpp_DllExport size_t [getSerializedSize](#) (size_t current_alignment=0) const
This function returns the serialized size of an object depending on the buffer alignment.
- user_cpp_DllExport void [serialize](#) ([eprosima::fastcdr::Cdr](#) &cdr) const
This function serializes an object using CDR serialization.
- user_cpp_DllExport void [deserialize](#) ([eprosima::fastcdr::Cdr](#) &cdr)
This function deserializes an object using CDR serialization.

Static Public Member Functions

- static user_cpp_DllExport size_t [getMaxCdrSerializedSize](#) (size_t current_alignment=0)
This function returns the maximum serialized size of an object depending on the buffer alignment.

6.20.1 Detailed Description

This class represents the structure [Foo_Request](#) that can be used to send/receive requests for the interface [Foo](#).

6.20.2 Constructor & Destructor Documentation

6.20.2.1 [FooDDS::Foo_Request::Foo_Request](#) (const [Foo_Request](#) & x)

Copy constructor.

Parameters

<i>x</i>	Reference to the object Foo_Request that will be copied.
----------	--

6.20.2.2 FooDDS::Foo_Request::Foo_Request (Foo_Request && x)

Move constructor.

Parameters

<i>x</i>	Reference to the object Foo_Request that will be copied.
----------	--

6.20.3 Member Function Documentation

6.20.3.1 void FooDDS::Foo_Request::deserialize (eprosima::fastcdr::Cdr & cdr)

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.20.3.2 size_t FooDDS::Foo_Request::getMaxCdrSerializedSize (size_t current_alignment = 0) [static]

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.20.3.3 size_t FooDDS::Foo_Request::getSerializedSize (size_t current_alignment = 0) const

This function returns the serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Serialized size.

6.20.3.4 user_cpp_DllExport void FooDDS::Foo_Request::header (const eprosima::rpc::protocol::dds::Request-Header & _header) [inline]

This method sets the request header information.

Parameters

<code>_header</code>	Request header.
----------------------	-----------------

6.20.3.5 `user_cpp_DllExport void FooDDS::Foo_Request::header (eprosima::rpc::protocol::dds::RequestHeader && _header) [inline]`

This method sets the request header information.

Parameters

<code>_header</code>	Request header.
----------------------	-----------------

6.20.3.6 `user_cpp_DllExport const eprosima::rpc::protocol::dds::RequestHeader& FooDDS::Foo_Request::header () const [inline]`

This method returns the request header information.

Returns

Request header.

6.20.3.7 `user_cpp_DllExport eprosima::rpc::protocol::dds::RequestHeader& FooDDS::Foo_Request::header () [inline]`

This method returns the request header information.

Returns

Request header.

6.20.3.8 `FooDDS::Foo_Request & FooDDS::Foo_Request::operator= (const Foo_Request & x)`

Copy assignment.

Parameters

<code>x</code>	Reference to the object Foo that will be copied.
----------------	--

6.20.3.9 `FooDDS::Foo_Request & FooDDS::Foo_Request::operator= (Foo_Request && x)`

Copy assignment.

Parameters

<code>x</code>	Reference to the object Foo that will be copied.
----------------	--

6.20.3.10 `user_cpp_DllExport void FooDDS::Foo_Request::request (const Foo_Call & _request) [inline]`

This method sets the union that encapsulates the interface operations.

Parameters

<code>_request</code>	Union.
-----------------------	--------

6.20.3.11 `user_cpp_DllExport void FooDDS::Foo_Request::request (Foo_Call &&_request) [inline]`

This method sets the union that encapsulates the interface operations.

Parameters

<code>_request</code>	Union.
-----------------------	--------

6.20.3.12 `user_cpp_DllExport const Foo_Call& FooDDS::Foo_Request::request () const [inline]`

This method returns the union that encapsulates the interface operations.

Returns

Union.

6.20.3.13 `user_cpp_DllExport Foo_Call& FooDDS::Foo_Request::request () [inline]`

This method returns the union that encapsulates the interface operations.

Returns

Union.

6.20.3.14 `void FooDDS::Foo_Request::serialize (eprosima::fastcdr::Cdr & cdr) const`

This function serializes an object using CDR serialization.

Parameters

<code>cdr</code>	CDR serialization object.
------------------	---------------------------

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

- `utils/doxygen/examples/dds/FooDDSTopics.h`
- `utils/doxygen/examples/dds/FooDDSTopics.cxx`

6.21 FooDDS::Foo_RequestDataReader Class Reference

Request DataReader.

```
#include <FooDDSTopicsPlugin.h>
```

Inheritance diagram for FooDDS::Foo_RequestDataReader:

Collaboration diagram for FooDDS::Foo_RequestDataReader:

Public Member Functions

- **Foo_RequestDataReader** (DDSDDataReader *impl)

6.21.1 Detailed Description

Request DataReader.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`

6.22 FooDDS::Foo_RequestDataWriter Class Reference

Request DataWriter.

```
#include <FooDDSTopicsPlugin.h>
```

Inheritance diagram for FooDDS::Foo_RequestDataWriter:

Collaboration diagram for FooDDS::Foo_RequestDataWriter:

Public Member Functions

- **Foo_RequestDataWriter** (DDSDDataWriter *impl)

6.22.1 Detailed Description

Request DataWriter.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`

6.23 FooDDS::Foo_RequestPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```

Inheritance diagram for FooDDS::Foo_RequestPlugin:

Collaboration diagram for FooDDS::Foo_RequestPlugin:

Public Member Functions

- DDSDDataReader * **create_datareaderl** (DDSDDataReader *dataReader)
- DDS_ReturnCode_t **destroy_datareaderl** (DDSDDataReader *dataReader)
- DDSDDataWriter * **create_datawriterl** (DDSDDataWriter *dataWriter)
- DDS_ReturnCode_t **destroy_datawriterl** (DDSDDataWriter *dataWriter)

Static Public Member Functions

- static const char * **get_typename** ()
- static [FooDDS::Foo_Request](#) * **create_data** (void)
- static void **destroy_data** ([FooDDS::Foo_Request](#) *sample)
- static void **copy_data** ([FooDDS::Foo_Request](#) *dst, const [FooDDS::Foo_Request](#) *src)
- static unsigned int **get_serialized_sample_max_size** (PRESTypePluginEndpointData endpoint_data, RTI-Bool include_encapsulation, RTIEncapsulationId encapsulation_id, unsigned int current_alignment)

- static unsigned int **get_serialized_sample_size** (PRESTypePluginEndpointData endpoint_data, RTIBool include_encapsulation, RTIEncapsulationId encapsulation_id, unsigned int current_alignment, const [FooDDS::Foo_Request](#) *sample)
- static unsigned int **get_serialized_sample_min_size** (PRESTypePluginEndpointData endpoint_data, RTIBool include_encapsulation, RTIEncapsulationId encapsulation_id, unsigned int current_alignment)
- static PRESTypePluginParticipantData **on_participant_attached** (void *registration_data, const struct PRESTypePluginParticipantInfo *participant_info, RTIBool top_level_registration, void *container_plugin_context, RTICdrTypeCode *typeCode)
- static void **on_participant_detached** (PRESTypePluginParticipantData participant_data)
- static PRESTypePluginEndpointData **on_endpoint_attached** (PRESTypePluginParticipantData participant_data, const struct PRESTypePluginEndpointInfo *endpoint_info, RTIBool top_level_registration, void *container_plugin_context)
- static void **on_endpoint_detached** (PRESTypePluginEndpointData endpoint_data)
- static RTIBool **copy_sample** (PRESTypePluginEndpointData endpoint_data, [FooDDS::Foo_Request](#) *dst, const [FooDDS::Foo_Request](#) *src)
- static RTIBool **serialize** (PRESTypePluginEndpointData endpoint_data, const [FooDDS::Foo_Request](#) *sample, struct RTICdrStream *stream, RTIBool serialize_encapsulation, RTIEncapsulationId encapsulation_id, RTIBool serialize_sample, void *endpoint_plugin_qos)
- static RTIBool **deserialize** (PRESTypePluginEndpointData endpoint_data, [FooDDS::Foo_Request](#) **sample, RTIBool *drop_sample, struct RTICdrStream *stream, RTIBool deserialize_encapsulation, RTIBool deserialize_sample, void *endpoint_plugin_qos)
- static PRESTypePluginKeyKind **get_key_kind** (void)
- static DDS_TypeCode * **get_typecode** ()
- static struct PRESTypePlugin * **new_plugin** (void)
- static void **delete_plugin** (struct PRESTypePlugin *plugin)
- static bool **register_type** (DDSDomainParticipant *participant, const char *type_name)

6.23.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

6.23.2 Member Function Documentation

6.23.2.1 **bool FooDDS::Foo_RequestPlugin::register_type (DDSDomainParticipant * participant, const char * type_name)**
[static]

TODO Mover al transporte

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

- utils/doxygen/examples/dds/FooDDSTopicsPlugin.h
- utils/doxygen/examples/dds/FooDDSTopicsPlugin.cxx

6.24 FooDDS::Foo_Return Class Reference

This class represents the union used in the DDS topic to encapsulate the operations in reply samples.

```
#include <FooDDSTopics.h>
```

Public Member Functions

- user_cpp_DLlexport [Foo_Return](#) ()
Default constructor.

- user_cpp_DllExport [~Foo_Return](#) ()
Destructor.
- user_cpp_DllExport [Foo_Return](#) (const [Foo_Return](#) &x)
Copy constructor.
- user_cpp_DllExport [Foo_Return](#) ([Foo_Return](#) &&x)
Move constructor.
- user_cpp_DllExport [Foo_Return](#) & [operator=](#) (const [Foo_Return](#) &x)
Copy assignment.
- user_cpp_DllExport [Foo_Return](#) & [operator=](#) ([Foo_Return](#) &&x)
Move assignment.
- user_cpp_DllExport void [_d](#) (int32_t __d)
This function sets the discriminator value.
- user_cpp_DllExport int32_t [_d](#) () const
This function returns the value of the discriminator.
- user_cpp_DllExport int32_t & [_d](#) ()
This function returns a reference to the discriminator.
- user_cpp_DllExport void **unknown_operation** (eprosima::rpc::protocol::dds::UnknownOperation _unknown_operation)
- user_cpp_DllExport
eprosima::rpc::protocol::dds::UnknownOperation **unknown_operation** () const
- user_cpp_DllExport
eprosima::rpc::protocol::dds::UnknownOperation & **unknown_operation** ()
- user_cpp_DllExport void [FooProcedure](#) (const [Foo_FooProcedure_Result](#) &_FooProcedure)
This function copies the value in member FooProcedure.
- user_cpp_DllExport void [FooProcedure](#) ([Foo_FooProcedure_Result](#) &&_FooProcedure)
This function moves the value in member FooProcedure.
- user_cpp_DllExport const
[Foo_FooProcedure_Result](#) & [FooProcedure](#) () const
This function returns a constant reference to member FooProcedure.
- user_cpp_DllExport
[Foo_FooProcedure_Result](#) & [FooProcedure](#) ()
This function returns a reference to member FooProcedure.
- user_cpp_DllExport size_t [getSerializedSize](#) (size_t current_alignment=0) const
This function returns the serialized size of an object depending on the buffer alignment.
- user_cpp_DllExport void [serialize](#) (eprosima::fastcdr::Cdr &cdr) const
This function serializes an object using CDR serialization.
- user_cpp_DllExport void [deserialize](#) (eprosima::fastcdr::Cdr &cdr)
This function deserializes an object using CDR serialization.

Static Public Member Functions

- static user_cpp_DllExport size_t [getMaxCdrSerializedSize](#) (size_t current_alignment=0)
This function returns the maximum serialized size of an object depending on the buffer alignment.

6.24.1 Detailed Description

This class represents the union used in the DDS topic to encapsulate the operations in reply samples.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 FooDDS::Foo_Return::Foo_Return (const Foo_Return & x)

Copy constructor.

Parameters

<i>x</i>	Reference to the object Foo_Return that will be copied.
----------	---

6.24.2.2 FooDDS::Foo_Return::Foo_Return (Foo_Return && x)

Move constructor.

Parameters

<i>x</i>	Reference to the object Foo_Return that will be copied.
----------	---

6.24.3 Member Function Documentation

6.24.3.1 void FooDDS::Foo_Return::_d (int32_t __d)

This function sets the discriminator value.

Parameters

<i>__d</i>	New value for the discriminator.
------------	----------------------------------

Exceptions

eProsima::rpc::exception::BadParamException	This exception is thrown if the new value doesn't correspond to the selected union member.
---	--

6.24.3.2 int32_t FooDDS::Foo_Return::_d () const

This function returns the value of the discriminator.

Returns

Value of the discriminator

6.24.3.3 int32_t & FooDDS::Foo_Return::_d ()

This function returns a reference to the discriminator.

Returns

Reference to the discriminator.

6.24.3.4 void FooDDS::Foo_Return::deserialize (eprosima::fastcdr::Cdr & cdr)

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.24.3.5 void FooDDS::Foo_Return::FooProcedure (const Foo_FooProcedure_Result & _FooProcedure)

This function copies the value in member FooProcedure.

Parameters

<i>_FooProcedure</i>	New value to be copied in member FooProcedure
----------------------	---

6.24.3.6 `void FooDDS::Foo_Return::FooProcedure (FooDDS::Foo_FooProcedure_Result && _FooProcedure)`

This function moves the value in member FooProcedure.

Parameters

<i>_FooProcedure</i>	New value to be moved in member FooProcedure
----------------------	--

6.24.3.7 `const FooDDS::Foo_FooProcedure_Result & FooDDS::Foo_Return::FooProcedure () const`

This function returns a constant reference to member FooProcedure.

Returns

Constant reference to member FooProcedure

6.24.3.8 `FooDDS::Foo_FooProcedure_Result & FooDDS::Foo_Return::FooProcedure ()`

This function returns a reference to member FooProcedure.

Returns

Reference to member FooProcedure

6.24.3.9 `size_t FooDDS::Foo_Return::getMaxCdrSerializedSize (size_t current_alignment = 0) [static]`

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.24.3.10 `size_t FooDDS::Foo_Return::getSerializedSize (size_t current_alignment = 0) const`

This function returns the serialized size of an object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Serialized size.

6.24.3.11 `FooDDS::Foo_Return & FooDDS::Foo_Return::operator= (const Foo_Return & x)`

Copy assignment.

Parameters

<i>x</i>	Reference to the object Foo_Return that will be copied.
----------	---

6.24.3.12 FooDDS::Foo_Return & FooDDS::Foo_Return::operator=(Foo_Return && x)

Move assignment.

Parameters

<i>x</i>	Reference to the object Foo_Return that will be copied.
----------	---

6.24.3.13 void FooDDS::Foo_Return::serialize (eprosima::fastcdr::Cdr & cdr) const

This function serializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

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

- `utils/doxygen/examples/dds/FooDDSTopics.h`
- `utils/doxygen/examples/dds/FooDDSTopics.cxx`

6.25 FooDDS::Foo_ReturnPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```

Static Public Member Functions

- static DDS_TypeCode * **get_typecode** ()

6.25.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`
- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.cxx`

6.26 eprosima::rpc::protocol::dds::FooDDSProtocol Class Reference

This class is responsible for serializing and deserializing the requests and responses of this application. It uses DDS.

```
#include <FooDDSDDSProtocol.h>
```

Inheritance diagram for `eprosima::rpc::protocol::dds::FooDDSProtocol`:

Collaboration diagram for `eprosima::rpc::protocol::dds::FooDDSProtocol`:

Public Member Functions

- [FooDDSProtocol](#) ()
Default constructor.
- virtual [~FooDDSProtocol](#) ()
Destructor.
- virtual bool [setTransport](#) ([eprosima::rpc::transport::Transport](#) &transport)
This method sets the transport for the communications.
- bool [activateInterface](#) (const char *interfaceName)
This function activates needed DDS entities to use an interface.
- void [FooDDS_Foo_FooProcedure](#) ()
This method implements the proxy part of the protocol for the operation FooProcedure. It is called from the Proxy interface.
- void [FooDDS_Foo_FooProcedure_async](#) ([FooDDS::Foo_FooProcedureCallbackHandler](#) &obj)
This asynchronous method implements the proxy part of the protocol for the operation FooProcedure. It is called from the Proxy interface.

Static Public Member Functions

- static void [FooDDS_Foo_serve](#) ([eprosima::rpc::protocol::Protocol](#) &protocol, void *data, [eprosima::rpc::transport::Endpoint](#) *endpoint)
This method implements the server part of the protocol for the interface Foo. It is called when a request sample is received.

Additional Inherited Members

6.26.1 Detailed Description

This class is responsible for serializing and deserializing the requests and responses of this application. It uses DDS.

6.26.2 Member Function Documentation

6.26.2.1 bool FooDDSProtocol::activateInterface (const char * *interfaceName*) [virtual]

This function activates needed DDS entities to use an interface.

Parameters

<i>interfaceName</i>	Interface name.
----------------------	-----------------

Returns

Whether the activation works successfully.

Implements [eprosima::rpc::protocol::FooDDSProtocol](#).

6.26.2.2 void FooDDSProtocol::FooDDS_Foo_serve ([eprosima::rpc::protocol::Protocol](#) & *protocol*, void * *data*, [eprosima::rpc::transport::Endpoint](#) * *endpoint*) [static]

This method implements the server part of the protocol for the interface Foo. It is called when a request sample is received.

Parameters

<i>protocol</i>	DDS protocol object that is in used.
<i>data</i>	Pointer to the received request sample. Cannot be NULL.
<i>endpoint</i>	Pointer to the endpoint that sent the request reply. Cannot be NULL.

6.26.2.3 `virtual bool eprosima::rpc::protocol::dds::FooDDSProtocol::setTransport (eprosima::rpc::transport::Transport & transport) [virtual]`

This method sets the transport for the communications.

Parameters

<i>transport</i>	Transport to use
------------------	------------------

Returns

True if the assignment is successful, false otherwise

Implements [eprosima::rpc::protocol::FooDDSProtocol](#).

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

- `utils/doxygen/examples/dds/FooDDSDDSProtocol.h`
- `utils/doxygen/examples/dds/FooDDSDDSProtocol.cxx`

6.27 eprosima::rpc::protocol::FooDDSProtocol Class Reference

[Protocol](#) base class for the specific application.

```
#include <FooDDSProtocol.h>
```

Inheritance diagram for `eprosima::rpc::protocol::FooDDSProtocol`:

Collaboration diagram for `eprosima::rpc::protocol::FooDDSProtocol`:

Public Member Functions

- virtual bool [setTransport](#) ([eprosima::rpc::transport::Transport](#) &transport)=0
This method sets the transport for the communications. It has to be implemented by the children classes.
- virtual bool [activateInterface](#) (const char *interfaceName)=0
In some protocols this function activates needed entities to use an interface.
- void [linkFooDDS_FooImpl](#) ([FooDDS::FooServerImpl](#) &impl)
This method links a specific servant with the protocol.
- virtual void [FooDDS_Foo_FooProcedure](#) ()=0
This method implements the proxy part of the protocol for the operation FooProcedure. It has to be implemented by the child classes.
- virtual void [FooDDS_Foo_FooProcedure_async](#) ([FooDDS::Foo_FooProcedureCallbackHandler](#) &obj)=0
This asynchronous method implements the proxy part of the protocol for the operation FooProcedure. It has to be implemented by the child classes.

Protected Attributes

- [FooDDS::FooServerImpl](#) * [_FooDDS_Foo_impl](#)

Additional Inherited Members

6.27.1 Detailed Description

[Protocol](#) base class for the specific application.

6.27.2 Member Function Documentation

6.27.2.1 `virtual bool eprosimarpc::protocol::FooDDSProtocol::activateInterface (const char * interfaceName) [pure virtual]`

In some protocols this function activates needed entities to use an interface.

Parameters

<i>interfaceName</i>	Interface name.
----------------------	-----------------

Returns

Whether the activation works successfully.

Implemented in [eprosimarpc::protocol::dds::FooDDSProtocol](#).

6.27.2.2 `void eprosimarpc::protocol::FooDDSProtocol::linkFooDDS_FooImpl (FooDDS::FooServerImpl & impl) [inline]`

This method links a specific servant with the protocol.

Parameters

<i>impl</i>	Servant implementation.
-------------	-------------------------

6.27.2.3 `virtual bool eprosimarpc::protocol::FooDDSProtocol::setTransport (eprosimarpc::transport::Transport & transport) [pure virtual]`

This method sets the transport for the communications. It has to be implemented by the children classes.

Parameters

<i>transport</i>	Transport to use.
------------------	-------------------

Returns

True if the assignment is successful, false otherwise

Implements [eprosimarpc::protocol::Protocol](#).

Implemented in [eprosimarpc::protocol::dds::FooDDSProtocol](#).

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

- `utils/doxygen/examples/dds/FooDDSProtocol.h`
- `utils/doxygen/examples/dds/FooDDSDDSProtocol.cxx`

6.28 FooDDS::FooPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```


Classes

- class [FooProcedure_InPlugin](#)
This class encapsulates the methods used on DDS topics by DDS middleware.
- class [FooProcedure_OutPlugin](#)
This class encapsulates the methods used on DDS topics by DDS middleware.
- class [FooProcedure_ResultPlugin](#)
This class encapsulates the methods used on DDS topics by DDS middleware.

6.28.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`

6.29 FooDDS::FooPlugin::FooProcedure_InPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```

Static Public Member Functions

- static DDS_TypeCode * **get_typecode** ()

6.29.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`
- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.cxx`

6.30 FooDDS::FooPlugin::FooProcedure_OutPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```

Static Public Member Functions

- static DDS_TypeCode * **get_typecode** ()

6.30.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

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

- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.h`
- `utils/doxygen/examples/dds/FooDDSTopicsPlugin.cxx`

6.31 FooDDS::FooPlugin::FooProcedure_ResultPlugin Class Reference

This class encapsulates the methods used on DDS topics by DDS middleware.

```
#include <FooDDSTopicsPlugin.h>
```

Inheritance diagram for FooDDS::FooPlugin::FooProcedure_ResultPlugin:

Collaboration diagram for FooDDS::FooPlugin::FooProcedure_ResultPlugin:

Static Public Member Functions

- static DDS_TypeCode * **get_typecode** ()

6.31.1 Detailed Description

This class encapsulates the methods used on DDS topics by DDS middleware.

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

- utils/doxygen/examples/dds/FooDDSTopicsPlugin.h
- utils/doxygen/examples/dds/FooDDSTopicsPlugin.cxx

6.32 FooDDS::FooProxy Class Reference

This class implements a specific server's proxy for the defined interface [Foo](#).

```
#include <FooDDSProxy.h>
```

Inheritance diagram for FooDDS::FooProxy:

Collaboration diagram for FooDDS::FooProxy:

Public Member Functions

- [FooProxy](#) ([eprosima::rpc::transport::ProxyTransport](#) &transport, [eprosima::rpc::protocol::FooDDSProtocol](#) &protocol)

This constructor sets the transport that will be used by the server's proxy.

- virtual [~FooProxy](#) ()

Destructor.

- void [FooProcedure](#) ()

Proxy method for the operation FooProcedure.

- void [FooProcedure_async](#) ([FooDDS::Foo_FooProcedureCallbackHandler](#) &obj)

Proxy asynchronous method for the operation FooProcedure.

Additional Inherited Members

6.32.1 Detailed Description

This class implements a specific server's proxy for the defined interface [Foo](#).

6.32.2 Constructor & Destructor Documentation

6.32.2.1 FooDDS::FooProxy::FooProxy (eprosima::rpc::transport::ProxyTransport & *transport*, eprosima::rpc::protocol::FooDDSProtocol & *protocol*)

This constructor sets the transport that will be used by the server's proxy.

Parameters

<i>transport</i>	The network transport that server's proxy has to use. This transport's object is not deleted by this class in its destructor. Cannot be NULL.
<i>protocol</i>	The protocol used to send the information over the transport. This protocol's object is not deleted by this class in its destructor. Cannot be NULL.

Exceptions

<i><code>eprosima::rpc::exception::InitializeException</code></i>	This exception is thrown when the initialization was wrong.
---	---

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

- `utils/doxygen/examples/dds/FooDDSPoxy.h`
- `utils/doxygen/examples/dds/FooDDSPoxy.cxx`

6.33 FooDDS::FooServer Class Reference

This class implements a specific server for the defined interface [Foo](#) by user.

```
#include <FooDDSServer.h>
```

Inheritance diagram for FooDDS::FooServer:

Collaboration diagram for FooDDS::FooServer:

Public Member Functions

- [FooServer](#) ([eprosima::rpc::strategy::ServerStrategy](#) &strategy, [eprosima::rpc::transport::ServerTransport](#) &transport, [eprosima::rpc::protocol::FooDDSProtocol](#) &protocol, [FooServerImpl](#) &servant)
This constructor sets the transport that will be used by the server.
- virtual [~FooServer](#) ()
Destructor.

Additional Inherited Members

6.33.1 Detailed Description

This class implements a specific server for the defined interface [Foo](#) by user.

6.33.2 Constructor & Destructor Documentation

- 6.33.2.1 **FooDDS::FooServer::FooServer** ([eprosima::rpc::strategy::ServerStrategy](#) & *strategy*, [eprosima::rpc::transport::ServerTransport](#) & *transport*, [eprosima::rpc::protocol::FooDDSProtocol](#) & *protocol*, [FooServerImpl](#) & *servant*)

This constructor sets the transport that will be used by the server.

Parameters

<i>strategy</i>	Strategy used by server to work with new requests. This class doesn't delete this object in its destructor. Cannot be NULL.
-----------------	---

<i>transport</i>	The network transport that the server has to use. This transport's object is not deleted by this class in its destructor. Cannot be NULL.
<i>protocol</i>	Generated protocol that the server has to use. This class has the information to process requests and build responses for this application environment.
<i>servant</i>	Servant that the server will use to invoke user's functions.

Exceptions

<i>eProsima::RPCDDS::InitializeException</i>	This exception is thrown when the initialization was wrong.
--	---

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

- `utils/doxygen/examples/dds/FooDDSServer.h`
- `utils/doxygen/examples/dds/FooDDSServer.cxx`

6.34 FooDDS::FooServerImpl Class Reference

This class is the skeleton of the servant and its remote procedures has to be implemented.

```
#include <FooDDSServerImpl.h>
```

Inheritance diagram for FooDDS::FooServerImpl:

Collaboration diagram for FooDDS::FooServerImpl:

Public Member Functions

- [FooServerImpl \(\)](#)
The default constructor.
- virtual [~FooServerImpl \(\)](#)
Destructor.

6.34.1 Detailed Description

This class is the skeleton of the servant and its remote procedures has to be implemented.

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

- `utils/doxygen/examples/dds/FooDDSServerImpl.h`

6.35 eprosima::rpc::protocol::dds::GUID_t Class Reference

Public Member Functions

- **GUID_t** (const [GUID_t](#) &guid)
- [GUID_t & operator=](#) (const [GUID_t](#) &guid)
- [uint8_t * value](#) ()
- void [serialize](#) (eprosima::fastcdr::Cdr &cdr) const
This function serializes the [GUID_t](#) object using CDR serialization.
- void [deserialize](#) (eprosima::fastcdr::Cdr &cdr)
This function deserializes the [GUID_t](#) object using CDR serialization.

Static Public Member Functions

- static `size_t getMaxCdrSerializedSize (size_t current_alignment)`

This function returns the maximum serialized size of a [GUID_t](#) object depending on the buffer alignment.

6.35.1 Member Function Documentation

6.35.1.1 `void eprosima::rpc::protocol::dds::GUID_t::deserialize (eprosima::fastcdr::Cdr & cdr)`

This function deserializes the [GUID_t](#) object using CDR serialization.

Parameters

<code>cdr</code>	CDR serialization object.
------------------	---------------------------

6.35.1.2 `static size_t eprosima::rpc::protocol::dds::GUID_t::getMaxCdrSerializedSize (size_t current_alignment)` [static]

This function returns the maximum serialized size of a [GUID_t](#) object depending on the buffer alignment.

Parameters

<code>current_ - alignment</code>	Buffer alignment.
-----------------------------------	-------------------

Returns

Maximum serialized size.

6.35.1.3 `void eprosima::rpc::protocol::dds::GUID_t::serialize (eprosima::fastcdr::Cdr & cdr) const`

This function serializes the [GUID_t](#) object using CDR serialization.

Parameters

<code>cdr</code>	CDR serialization object.
------------------	---------------------------

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

- `includetmp/rpcdds/protocols/dds/MessageHeader.h`

6.36 eprosima::rpc::protocol::dds::GUID_tPlugin Class Reference

Static Public Member Functions

- static `DDS_TypeCode * get_typecode ()`

This function returns the TypeCode.

6.36.1 Member Function Documentation

6.36.1.1 `static DDS_TypeCode* eprosima::rpc::protocol::dds::GUID_tPlugin::get_typecode ()` [static]

This function returns the TypeCode.

Returns

The TypeCode.

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

- includetmp/rpcdds/protocols/dds/MessageHeaderPlugin.h

6.37 eprosima::rpc::exception::IncompatibleException Class Reference

This class is thrown as an exception when a selected protocol and transport are incompatible.

```
#include <IncompatibleException.h>
```

Inheritance diagram for eprosima::rpc::exception::IncompatibleException:

Collaboration diagram for eprosima::rpc::exception::IncompatibleException:

Public Member Functions

- FASTRPC_DIIAPI [IncompatibleException](#) (const std::string &message)
Default constructor.
- FASTRPC_DIIAPI [IncompatibleException](#) (const [IncompatibleException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [IncompatibleException](#) ([IncompatibleException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [IncompatibleException](#) & operator= (const [IncompatibleException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [IncompatibleException](#) & operator= ([IncompatibleException](#) &&ex)
Assignment operation.
- virtual FASTRPC_DIIAPI [~IncompatibleException](#) () throw ()
Default constructor.
- virtual FASTRPC_DIIAPI void [raise](#) () const
This function throws the object as an exception.

Additional Inherited Members

6.37.1 Detailed Description

This class is thrown as an exception when a selected protocol and transport are incompatible.

6.37.2 Constructor & Destructor Documentation

6.37.2.1 FASTRPC_DIIAPI eprosima::rpc::exception::IncompatibleException::IncompatibleException (const std::string &message) [inline]

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.37.2.2 FASTRPC_DIIAPI eprosima::rpc::exception::IncompatibleException::IncompatibleException (const IncompatibleException & ex)

Default copy constructor.

Parameters

<i>ex</i>	IncompatibleException that will be copied.
-----------	--

6.37.2.3 FASTRPC_DIIAPI eprosima::rpc::exception::IncompatibleException::IncompatibleException (IncompatibleException && ex)

Default move constructor.

Parameters

<i>ex</i>	IncompatibleException that will be moved.
-----------	---

6.37.3 Member Function Documentation

6.37.3.1 FASTRPC_DIIAPI IncompatibleException& eprosima::rpc::exception::IncompatibleException::operator= (const IncompatibleException & ex)

Assignment operation.

Parameters

<i>ex</i>	IncompatibleException that will be copied.
-----------	--

6.37.3.2 FASTRPC_DIIAPI IncompatibleException& eprosima::rpc::exception::IncompatibleException::operator= (IncompatibleException && ex)

Assignment operation.

Parameters

<i>ex</i>	IncompatibleException that will be moved.
-----------	---

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

- `includetmp/rpcdds/exceptions/IncompatibleException.h`

6.38 eprosima::rpc::exception::InitializeException Class Reference

This class is thrown as an exception when there is an error initializing an object.

```
#include <InitializeException.h>
```

Inheritance diagram for `eprosima::rpc::exception::InitializeException`:

Collaboration diagram for `eprosima::rpc::exception::InitializeException`:

Public Member Functions

- FASTRPC_DIIAPI [InitializeException](#) (const std::string &message)
Default constructor.
- FASTRPC_DIIAPI [InitializeException](#) (const [InitializeException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [InitializeException](#) ([InitializeException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [InitializeException](#) & [operator=](#) (const [InitializeException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [InitializeException](#) & [operator=](#) ([InitializeException](#) &&ex)
Assignment operation.
- virtual FASTRPC_DIIAPI [~InitializeException](#) () throw ()
Default destructor.
- virtual FASTRPC_DIIAPI void [raise](#) () const
This function throws the object as an exception.

Additional Inherited Members

6.38.1 Detailed Description

This class is thrown as an exception when there is an error initializing an object.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 FASTRPC_DIIAPI eprosima::rpc::exception::InitializeException::InitializeException (const std::string & message)
[inline]

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.38.2.2 FASTRPC_DIIAPI eprosima::rpc::exception::InitializeException::InitializeException (const [InitializeException](#) & ex)

Default copy constructor.

Parameters

<i>ex</i>	InitializeException that will be copied.
-----------	--

6.38.2.3 FASTRPC_DIIAPI eprosima::rpc::exception::InitializeException::InitializeException ([InitializeException](#) && ex)

Default move constructor.

Parameters

<i>ex</i>	InitializeException that will be moved.
-----------	---

6.38.3 Member Function Documentation

6.38.3.1 FASTRPC_DIIAPI InitializeException& eprosima::rpc::exception::InitializeException::operator= (const InitializeException & *ex*)

Assignment operation.

Parameters

<i>ex</i>	InitializeException that will be copied.
-----------	--

6.38.3.2 FASTRPC_DIIAPI InitializeException& eprosima::rpc::exception::InitializeException::operator= (InitializeException && *ex*)

Assignment operation.

Parameters

<i>ex</i>	InitializeException that will be moved.
-----------	---

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

- includetmp/rpcdds/exceptions/InitializeException.h

6.39 eprosima::rpc::protocol::Protocol Class Reference

This abstract class represents the protocol used by the RPCs. It serializes and deserializes the information and uses a [eprosima::rpc::transport::Transport](#) to send it and receive it.

```
#include <Protocol.h>
```

Inheritance diagram for eprosima::rpc::protocol::Protocol:

Public Member Functions

- virtual bool [setTransport](#) ([eprosima::rpc::transport::Transport](#) &transport)=0
This method sets a [eprosima::rpc::transport::Transport](#) object, used for the communications.

Protected Member Functions

- [Protocol](#) ()
Default constructor.
- virtual [~Protocol](#) ()
Default destructor.
- [eprosima::rpc::transport::Transport](#) & [getTransport](#) () const
This method returns the [eprosima::rpc::transport::Transport](#) object, used for the communications.
- void [_setTransport](#) ([eprosima::rpc::transport::Transport](#) &transport)
This method sets a [eprosima::rpc::transport::Transport](#) object, used for the communications.

6.39.1 Detailed Description

This abstract class represents the protocol used by the RPCs. It serializes and deserializes the information and uses a [eprosimarpc::transport::Transport](#) to send it and receive it.

6.39.2 Member Function Documentation

6.39.2.1 `void eprosimarpc::protocol::Protocol::_setTransport (eprosimarpc::transport::Transport & transport)`
`[inline], [protected]`

This method sets a [eprosimarpc::transport::Transport](#) object, used for the communications.

Parameters

<i>transport</i>	eprosimarpc::transport::Transport to use for the communication.
------------------	---

6.39.2.2 `eprosimarpc::transport::Transport& eprosimarpc::protocol::Protocol::getTransport () const`
`[inline], [protected]`

This method returns the [eprosimarpc::transport::Transport](#) object, used for the communications.

Returns

[eprosimarpc::transport::Transport](#) used for the communications.

6.39.2.3 `virtual bool eprosimarpc::protocol::Protocol::setTransport (eprosimarpc::transport::Transport & transport)`
`[pure virtual]`

This method sets a [eprosimarpc::transport::Transport](#) object, used for the communications.

Parameters

<i>transport</i>	eprosimarpc::transport::Transport to use for the communications.
------------------	--

Implemented in [eprosimarpc::protocol::dds::FooDDSProtocol](#), and [eprosimarpc::protocol::FooDDSProtocol](#).

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

- `includetmp/rpcdds/protocols/Protocol.h`

6.40 eprosimarpc::proxy::Proxy Class Reference

This class implements the common functionalities that all server's proxies have.

```
#include <Proxy.h>
```

Inheritance diagram for `eprosimarpc::proxy::Proxy`:

Protected Member Functions

- [Proxy](#) ([eprosimarpc::transport::ProxyTransport](#) &transport, [eprosimarpc::protocol::Protocol](#) &protocol)
[Proxy](#) constructor.
- virtual [~Proxy](#) ()
The default destructor.
- [eprosimarpc::protocol::Protocol](#) & [getProtocol](#) () const

Method to obtain the protocol.

- `eprosima::rpc::transport::ProxyTransport & getTransport () const`

Method to get the transport.

6.40.1 Detailed Description

This class implements the common functionalities that all server's proxies have.

6.40.2 Constructor & Destructor Documentation

- 6.40.2.1 `eprosima::rpc::proxy::Proxy::Proxy (eprosima::rpc::transport::ProxyTransport & transport, eprosima::rpc::protocol::Protocol & protocol) [protected]`

`Proxy` constructor.

Parameters

<i>transport</i>	The transport that will be used by the server's proxy. This class doesn't delete this object in its destructor.
<i>protocol</i>	The protocol used to send information over the transport. This class doesn't delete this object in its destructor.

Exceptions

<i>InitializeException</i>	This exception is thrown when the initialization went wrong.
----------------------------	--

6.40.3 Member Function Documentation

- 6.40.3.1 `eprosima::rpc::protocol::Protocol& eprosima::rpc::proxy::Proxy::getProtocol () const [inline], [protected]`

Method to obtain the protocol.

Returns

The protocol used to send information over the transport

- 6.40.3.2 `eprosima::rpc::transport::ProxyTransport& eprosima::rpc::proxy::Proxy::getTransport () const [inline], [protected]`

Method to get the transport.

Returns

The transport used used by the proxy

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

- `includetmp/rpcdds/client/Proxy.h`

6.41 eprosima::rpc::transport::dds::ProxyProcedureEndpoint Class Reference

This class represents a remote endpoint used by a proxy. It also encapsulates the DDS datawriter and the DDS datareader.

```
#include <ProxyProcedureEndpoint.h>
```

Inheritance diagram for eprosimarpc::transport::dds::ProxyProcedureEndpoint:

Collaboration diagram for eprosimarpc::transport::dds::ProxyProcedureEndpoint:

Public Member Functions

- [ProxyProcedureEndpoint](#) ([ProxyTransport](#) &transport)
Default constructor.
- virtual [~ProxyProcedureEndpoint](#) ()
Default destructor.
- int [initialize](#) (const char *name, const char *writertypename, const char *readertypename, bool eprosimatypes, Transport::Copy_data copy_data, int dataSize)
This function initializes the proxy procedure endpoint.
- void [finalize](#) ()
This function finalizes the proxy procedure endpoint. All entities and objects created by this procedure endpoint are deleted.
- eprosimarpc::ReturnMessage [send](#) (void *request, void *reply)
This function sends a synchronous RPC call. It sends the request to the server and waits for the reply. The wait mechanism is implemented with a DDS WaitSet.
- eprosimarpc::ReturnMessage [send_async](#) (void *request, [DDSAsyncTask](#) *task)
This function sends an asynchronous RPC call. It sends the request to the server and does not wait for the reply. Instead, the corresponding callback inside the [DDSAsyncTask](#) object will be invoked when the response arrives.
- void [freeQuery](#) (DDSQueryCondition *query)
Frees a DDS query condition.
- eprosimarpc::ReturnMessage [takeReply](#) (void *reply, DDSQueryCondition *query)
This function takes a sample from the datareader.

Additional Inherited Members

6.41.1 Detailed Description

This class represents a remote endpoint used by a proxy. It also encapsulates the DDS datawriter and the DDS datareader.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 eprosimarpc::transport::dds::ProxyProcedureEndpoint::ProxyProcedureEndpoint ([ProxyTransport](#) & transport)

Default constructor.

Parameters

Transport	that is creating the proxy procedure endpoint. It cannot be NULL.
---------------------------	---

6.41.3 Member Function Documentation

6.41.3.1 void eprosimarpc::transport::dds::ProxyProcedureEndpoint::freeQuery (DDSQueryCondition * query)

Frees a DDS query condition.

Parameters

<i>query</i>	Query condition to free.
--------------	--------------------------

6.41.3.2 `int eprosima::rpc::transport::dds::ProxyProcedureEndpoint::initialize (const char * name, const char * writertypename, const char * readertypename, bool eprosima_types, Transport::Copy_data copy_data, int dataSize)`

This function initializes the proxy procedure endpoint.

Parameters

<i>name</i>	The name associated with this proxy procedure endpoint. It cannot be NULL.
<i>writertypename</i>	The type name of the topic that the proxy procedure endpoint uses in the datawriter. It cannot be NULL.
<i>readertypename</i>	The type name of the topic that the proxy procedure endpoint uses in the datareader. It cannot be NULL.
<i>copy_data</i>	Pointer to the function used to copy the data when it is received.

Returns

0 if the initialization works. -1 in other case. TODO

6.41.3.3 `eprosima::rpc::ReturnMessage eprosima::rpc::transport::dds::ProxyProcedureEndpoint::send (void * request, void * reply)`

This function sends a synchronous RPC call. It sends the request to the server and waits for the reply. The wait mechanism is implemented with a DDS WaitSet.

Parameters

<i>request</i>	Pointer to the allocated request. It cannot be NULL.
<i>reply</i>	Pointer to the allocated reply. This memory will be filled with the incoming data. The pointer can be NULL and this means that the RPC call is oneway.

Returns

Operation status

Exceptions

<code>eprosima::rpc::exception::-</code> <code>ServerTimeoutException.</code>	
--	--

6.41.3.4 `eprosima::rpc::ReturnMessage eprosima::rpc::transport::dds::ProxyProcedureEndpoint::send_async (void * request, DDSAsyncTask * task)`

This function sends an asynchronous RPC call. It sends the request to the server and does not wait for the reply. Instead, the corresponding callback inside the [DDSAsyncTask](#) object will be invoked when the response arrives.

Parameters

<i>request</i>	Pointer to the allocated request. It cannot be NULL.
----------------	--

<i>task</i>	Object containing information of the asynchronous task.
-------------	---

Returns

Operation status. It can be CLIENT_INTERNAL_ERROR or NO_SERVER

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

- includetmp/rpcdds/transport/dds/components/ProxyProcedureEndpoint.h

6.42 eprosima::rpc::transport::dds::ProxyTransport Class Reference

This class is the base of all proxies that implement a transport using DDS.

```
#include <ProxyTransport.h>
```

Inheritance diagram for eprosima::rpc::transport::dds::ProxyTransport:

Collaboration diagram for eprosima::rpc::transport::dds::ProxyTransport:

Public Member Functions

- virtual FASTRPC_DIIAPI [~ProxyTransport](#) ()
Default destructor.
- virtual FASTRPC_DIIAPI const char * [getType](#) () const
This abstract function returns the type of the transport. This function has to be implemented by the child classes.
- FASTRPC_DIIAPI const char * [getRemoteServiceName](#) () const
This function returns the DDS service name.
- FASTRPC_DIIAPI const char * [getInstanceName](#) () const
- FASTRPC_DIIAPI long [getTimeout](#) ()
This function gets the timeout value.
- FASTRPC_DIIAPI [eprosima::rpc::transport::Endpoint * createProcedureEndpoint](#) (const char *name, const char *writertypename, const char *readertypename, bool eprosima_types, Transport::Create_data create_data, Transport::Copy_data copy_data, Transport::Destroy_data destroy_data, Transport::ProcessFunc processFunc, int dataSize)
This function creates a new proxy procedure endpoint. This proxy procedure endpoint manages the DDS datawriter and the DDS datareader.
- FASTRPC_DIIAPI int [addAsyncTask](#) (DDSQueryCondition *query, [DDSAsyncTask](#) *task, long timeout)
This function adds a asynchronous task to the asynchronous thread.
- FASTRPC_DIIAPI void [deleteAssociatedAsyncTasks](#) ([ProxyProcedureEndpoint](#) *pe)
This function deletes all the asynchronous tasks associated with the [ProxyProcedureEndpoint](#) endpoint.

Protected Member Functions

- virtual FASTRPC_DIIAPI int [setTransport](#) (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant)=0
This abstract function sets the QoS to use a specific transport.
- FASTRPC_DIIAPI [ProxyTransport](#) (const char *const &remoteServiceName, const char *const &instanceName, int domainId=0, long milliseconds=10000L)
Default constructor.

Additional Inherited Members

6.42.1 Detailed Description

This class is the base of all proxies that implement a transport using DDS.

6.42.2 Constructor & Destructor Documentation

6.42.2.1 `FASTRPC_DllAPI eprosima::rpc::transport::dds::ProxyTransport::ProxyTransport (const char *const & remoteServiceName, const char *const & instanceName, int domainId = 0, long milliseconds = 10000L)`
[protected]

Default constructor.

Parameters

<i>domainId</i>	Optional parameter that specifies the domain identifier will be used in DDS.
-----------------	--

6.42.3 Member Function Documentation

6.42.3.1 `FASTRPC_DllAPI int eprosima::rpc::transport::dds::ProxyTransport::addAsyncTask (DDSQueryCondition * query, DDSAsyncTask * task, long timeout)`

This function adds a asynchronous task to the asynchronous thread.

Parameters

<i>query</i>	The DDS query condition that is used to take the request. Cannot be NULL.
<i>task</i>	The asynchronos task created and associated with a request. Cannot be NULL.
<i>timeout</i>	The timeout used for this request.

Returns

A 0 value is returned if function works successfully. In any other case, -1 is returned.

6.42.3.2 `FASTRPC_DllAPI eprosima::rpc::transport::Endpoint* eprosima::rpc::transport::dds::ProxyTransport::create-ProcedureEndpoint (const char * name, const char * writertypename, const char * readertypename, bool eprosima_types, Transport::Create_data create_data, Transport::Copy_data copy_data, Transport::Destroy_data destroy_data, Transport::ProcessFunc processFunc, int dataSize)` [virtual]

This function creates a new proxy procedure endpoint. This proxy procedure endpoint manages the DDS datawriter and the DDS datareader.

TODO Actualizar

Parameters

<i>name</i>	The name associated with this proxy procedure endpoint. It cannot be NULL.
<i>writertypename</i>	The type name of the topic that the procedure endpoint uses in the datawriter. It cannot be NULL.
<i>readertypename</i>	The type name of the topic that the procedure endpoint uses in the datareader. It cannot be NULL.

<i>initialize_data</i>	Pointer to the function to initialize DataReader received data
<i>copy_data</i>	Pointer to the function used to copy the data when it is received.
<i>finalize_data</i>	Pointer to the function to finalize DataReader received data
<i>ProcessFunc</i>	Pointer to the function invoked when a message is received from the server
<i>dataSize</i>	Size of the DataReader data structure

Returns

0 if the function works. -1 in other case. TODO

Implements [eprosimarpc::transport::dds::Transport](#).

6.42.3.3 FASTRPC_DIIAPI void eprosimarpc::transport::dds::ProxyTransport::deleteAssociatedAsyncTasks (ProxyProcedureEndpoint * pe)

This function deletes all the asynchronous tasks associated with the [ProxyProcedureEndpoint](#) endpoint.

Parameters

<i>pe</i>	Pointer to the ProxyProcedureEndpoint . It cannot be NULL.
-----------	--

6.42.3.4 FASTRPC_DIIAPI const char* eprosimarpc::transport::dds::ProxyTransport::getRemoteServiceName () const

This function returns the DDS service name.

Returns

DDS service name.

6.42.3.5 FASTRPC_DIIAPI long eprosimarpc::transport::dds::ProxyTransport::getTimeout ()

This function gets the timeout value.

Returns

Timeout value.

6.42.3.6 virtual FASTRPC_DIIAPI int eprosimarpc::transport::dds::ProxyTransport::setTransport (DDS_DomainParticipantQos & participantQos, DDSDomainParticipant * participant) [protected], [pure virtual]

This abstract function sets the QoS to use a specific transport.

Parameters

<i>participantQos</i>	Reference to the DDS domain participant QoS.
<i>participant</i>	The domain participant that will be set to use a specific transport.

Implements [eprosimarpc::transport::dds::Transport](#).

Implemented in [eprosimarpc::transport::dds::UDPProxyTransport](#).

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

- `includetmp/rpcdds/transport/dds/ProxyTransport.h`

6.43 eprosima::rpc::transport::ProxyTransport Class Reference

This interface is the base of all classes that implement a transport that can be used by the proxy.

```
#include <ProxyTransport.h>
```

Inheritance diagram for eprosima::rpc::transport::ProxyTransport:

Collaboration diagram for eprosima::rpc::transport::ProxyTransport:

Public Member Functions

- [ProxyTransport](#) ()
Default constructor.
- virtual [~ProxyTransport](#) ()
Default destructor.
- virtual const char * [getType](#) () const =0
This function returns the type of the transport. This function has to be implemented by the child classes.
- [TransportBehaviour](#) [getBehaviour](#) () const
This function returns the behaviour of the transport.
- virtual bool [connect](#) ()=0
Abstract method. It must start a connection with the server.
- virtual bool [send](#) (const void *buffer, const size_t bufferSize)=0
Abstract method. It must send a request to the server.
- virtual int [receive](#) (void *buffer, const size_t bufferSize, size_t &dataToRead)=0
Abstract method. It must receive a reply from the server.

6.43.1 Detailed Description

This interface is the base of all classes that implement a transport that can be used by the proxy.

6.43.2 Member Function Documentation

6.43.2.1 virtual bool eprosima::rpc::transport::ProxyTransport::connect () [pure virtual]

Abstract method. It must start a connection with the server.

Returns

true if the operation is successful, false otherwise.

6.43.2.2 [TransportBehaviour](#) eprosima::rpc::transport::ProxyTransport::getBehaviour () const [inline],
[virtual]

This function returns the behaviour of the transport.

Returns

The behaviour of the transport.

Implements [eprosima::rpc::transport::Transport](#).

6.43.2.3 virtual int eprosima::rpc::transport::ProxyTransport::receive (void * buffer, const size_t bufferSize, size_t &dataToRead) [pure virtual]

Abstract method. It must receive a reply from the server.

Parameters

<i>buffer</i>	Buffer that will contain the HTTP message.
<i>bufferSize</i>	Size of the buffer.
<i>dataToRead</i>	Number of bytes received.

Returns

-1 if the operation fails.

6.43.2.4 `virtual bool eprosima::rpc::transport::ProxyTransport::send (const void * buffer, const size_t bufferSize)` [pure virtual]

Abstract method. It must send a request to the server.

Parameters

<i>buffer</i>	Buffer containing the request
<i>bufferSize</i>	Buffer size

Returns

true if the operation is successful, false otherwise.

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

- includetmp/rpcdds/transport/ProxyTransport.h

6.44 eprosima::rpc::protocol::dds::ReplyHeader Class Reference

Header information used in all generated reply topics.

```
#include <MessageHeader.h>
```

Public Member Functions

- [ReplyHeader](#) ()
Default constructor.
- [ReplyHeader](#) (const [ReplyHeader](#) &header)
Copy constructor.
- [ReplyHeader](#) ([ReplyHeader](#) &&header)
Copy constructor.
- [~ReplyHeader](#) ()
Destructor.
- [ReplyHeader](#) & [operator=](#) (const [ReplyHeader](#) &header)
Copy assignment.
- [ReplyHeader](#) & [operator=](#) ([ReplyHeader](#) &&header)
Copy assignment.
- void [request_id](#) (const [SampleIdentity_t](#) &_request_id)
This function sets the client identifier.
- void [request_id](#) ([SampleIdentity_t](#) &&_request_id)
This function sets the client identifier.
- const [SampleIdentity_t](#) & [request_id](#) () const

This function returns the client identifier.

- [SampleIdentity_t](#) & [request_id](#) ()

This function returns the client identifier.

- void [serialize](#) (eprosima::fastcdr::Cdr &cdr) const

This function serializes the [ReplyHeader](#) object using CDR serialization.

- void [deserialize](#) (eprosima::fastcdr::Cdr &cdr)

This function deserializes the [ReplyHeader](#) object using CDR serialization.

Static Public Member Functions

- static size_t [getMaxCdrSerializedSize](#) (size_t current_alignment)

This function returns the maximum serialized size of a [ReplyHeader](#) object depending on the buffer alignment.

6.44.1 Detailed Description

Header information used in all generated reply topics.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 eprosima::rpc::protocol::dds::ReplyHeader::ReplyHeader (const ReplyHeader & header)

Copy constructor.

Parameters

<i>header</i>	ReplyHeader object to be copied.
---------------	--

6.44.2.2 eprosima::rpc::protocol::dds::ReplyHeader::ReplyHeader (ReplyHeader && header)

Copy constructor.

Parameters

<i>header</i>	ReplyHeader object to be copied.
---------------	--

6.44.3 Member Function Documentation

6.44.3.1 void eprosima::rpc::protocol::dds::ReplyHeader::deserialize (eprosima::fastcdr::Cdr & cdr)

This function deserializes the [ReplyHeader](#) object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.44.3.2 static size_t eprosima::rpc::protocol::dds::ReplyHeader::getMaxCdrSerializedSize (size_t current_alignment) [static]

This function returns the maximum serialized size of a [ReplyHeader](#) object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.44.3.3 ReplyHeader& eprosima::rpc::protocol::dds::ReplyHeader::operator= (const ReplyHeader & header)

Copy assignment.

Parameters

<i>header</i>	ReplyHeader object to be copied.
---------------	--

6.44.3.4 ReplyHeader& eprosima::rpc::protocol::dds::ReplyHeader::operator= (ReplyHeader && header)

Copy assignment.

Parameters

<i>header</i>	ReplyHeader object to be copied.
---------------	--

6.44.3.5 void eprosima::rpc::protocol::dds::ReplyHeader::request_id (const SampleIdentity_t & _request_id)
[inline]

This function sets the client identifier.

Parameters

<i>_clientId</i>	Client identifier
------------------	-------------------

6.44.3.6 void eprosima::rpc::protocol::dds::ReplyHeader::request_id (SampleIdentity_t && _request_id) [inline]

This function sets the client identifier.

Parameters

<i>_clientId</i>	Client identifier
------------------	-------------------

6.44.3.7 const SampleIdentity_t& eprosima::rpc::protocol::dds::ReplyHeader::request_id () const [inline]

This function returns the client identifier.

Returns

Client identifier

6.44.3.8 SampleIdentity_t& eprosima::rpc::protocol::dds::ReplyHeader::request_id () [inline]

This function returns the client identifier.

Returns

Client identifier

6.44.3.9 `void eprosima::rpc::protocol::dds::ReplyHeader::serialize (eprosima::fastcdr::Cdr & cdr) const`

This function serializes the [ReplyHeader](#) object using CDR serialization.

Parameters

<code>cdr</code>	CDR serialization object.
------------------	---------------------------

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

- `includetmp/rpcdds/protocols/dds/MessageHeader.h`

6.45 eprosima::rpc::protocol::dds::ReplyHeaderPlugin Class Reference

This class offers the functions needed by DDS middleware to use the class [ReplyHeaderPlugin](#).

```
#include <MessageHeaderPlugin.h>
```

Static Public Member Functions

- static `DDS_TypeCode * get_typecode ()`

This function returns the TypeCode.

6.45.1 Detailed Description

This class offers the functions needed by DDS middleware to use the class [ReplyHeaderPlugin](#).

6.45.2 Member Function Documentation

6.45.2.1 `static DDS_TypeCode* eprosima::rpc::protocol::dds::ReplyHeaderPlugin::get_typecode () [static]`

This function returns the TypeCode.

Returns

The TypeCode.

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

- `includetmp/rpcdds/protocols/dds/MessageHeaderPlugin.h`

6.46 eprosima::rpc::protocol::dds::RequestHeader Class Reference

Header information used in all generated request topics.

```
#include <MessageHeader.h>
```

Public Member Functions

- [RequestHeader](#) ()
Default constructor.
- [RequestHeader](#) (const [RequestHeader](#) &header)
Copy constructor.
- [RequestHeader](#) ([RequestHeader](#) &&header)
Copy constructor.
- [~RequestHeader](#) ()
Destructor.
- [RequestHeader](#) & operator= (const [RequestHeader](#) &header)
Copy assignment.
- [RequestHeader](#) & operator= ([RequestHeader](#) &&header)
Copy assignment.
- void [request_id](#) (const [SampleIdentity_t](#) &_request_id)
This function sets the client identifier.
- void [request_id](#) ([SampleIdentity_t](#) &&_request_id)
This function sets the client identifier.
- const [SampleIdentity_t](#) & [request_id](#) () const
This function returns the client identifier.
- [SampleIdentity_t](#) & [request_id](#) ()
This function returns the client identifier.
- void [remote_service_name](#) (const char *_remote_service_name)
This function sets the server service name.
- const char * [remote_service_name](#) () const
This function returns the server service name.
- void [instance_name](#) (const char *_instance_name)
This function sets the server service name.
- const char * [instance_name](#) () const
This function returns the server service name.
- void [serialize](#) (eprosima::fastcdr::Cdr &cdr) const
This function serializes the [RequestHeader](#) object using CDR serialization.
- void [deserialize](#) (eprosima::fastcdr::Cdr &cdr)
This function deserializes the [RequestHeader](#) object using CDR serialization.

Static Public Member Functions

- static size_t [getMaxCdrSerializedSize](#) (size_t current_alignment)
This function returns the maximum serialized size of a [RequestHeader](#) object depending on the buffer alignment.

6.46.1 Detailed Description

Header information used in all generated request topics.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 eprosima::rpc::protocol::dds::RequestHeader::RequestHeader (const RequestHeader & header)

Copy constructor.

Parameters

<i>header</i>	RequestHeader object to be copied.
---------------	--

6.46.2.2 `eprosima::rpc::protocol::dds::RequestHeader::RequestHeader (RequestHeader && header)`

Copy constructor.

Parameters

<i>header</i>	RequestHeader object to be copied.
---------------	--

6.46.3 Member Function Documentation

6.46.3.1 `void eprosima::rpc::protocol::dds::RequestHeader::deserialize (eprosima::fastcdr::Cdr & cdr)`

This function deserializes the [RequestHeader](#) object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.46.3.2 `static size_t eprosima::rpc::protocol::dds::RequestHeader::getMaxCdrSerializedSize (size_t current_alignment)`
[static]

This function returns the maximum serialized size of a [RequestHeader](#) object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.46.3.3 `void eprosima::rpc::protocol::dds::RequestHeader::instance_name (const char * _instance_name)` [inline]

This function sets the server service name.

Parameters

<i>_remoteService-Name</i>	Server service name.
----------------------------	----------------------

6.46.3.4 `const char* eprosima::rpc::protocol::dds::RequestHeader::instance_name () const` [inline]

This function returns the server service name.

Returns

Server service name.

6.46.3.5 `RequestHeader& eprosima::rpc::protocol::dds::RequestHeader::operator= (const RequestHeader & header)`

Copy assignment.

Parameters

<i>header</i>	RequestHeader object to be copied.
---------------	--

6.46.3.6 RequestHeader& eprosima::rpc::protocol::dds::RequestHeader::operator= (RequestHeader && header)

Copy assignment.

Parameters

<i>header</i>	RequestHeader object to be copied.
---------------	--

6.46.3.7 void eprosima::rpc::protocol::dds::RequestHeader::remote_service_name (const char * _remote_service_name)
[inline]

This function sets the server service name.

Parameters

<i>_remoteService- Name</i>	Server service name.
---------------------------------	----------------------

6.46.3.8 const char* eprosima::rpc::protocol::dds::RequestHeader::remote_service_name () const [inline]

This function returns the server service name.

Returns

Server service name.

6.46.3.9 void eprosima::rpc::protocol::dds::RequestHeader::request_id (const SampleIdentity_t & _request_id)
[inline]

This function sets the client identifier.

Parameters

<i>_clientId</i>	Client identifier
------------------	-------------------

6.46.3.10 void eprosima::rpc::protocol::dds::RequestHeader::request_id (SampleIdentity_t && _request_id)
[inline]

This function sets the client identifier.

Parameters

<i>_clientId</i>	Client identifier
------------------	-------------------

6.46.3.11 const SampleIdentity_t& eprosima::rpc::protocol::dds::RequestHeader::request_id () const [inline]

This function returns the client identifier.

Returns

Client identifier

6.46.3.12 SampleIdentity_t& eprosima::rpc::protocol::dds::RequestHeader::request_id () [inline]

This function returns the client identifier.

Returns

Client identifier

6.46.3.13 void eprosima::rpc::protocol::dds::RequestHeader::serialize (eprosima::fastcdr::Cdr & cdr) const

This function serializes the [RequestHeader](#) object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

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

- includetmp/rpcdds/protocols/dds/MessageHeader.h

6.47 eprosima::rpc::protocol::dds::RequestHeaderPlugin Class Reference

This class offers the functions needed by DDS middleware to use the class [RequestHeaderPlugin](#).

```
#include <MessageHeaderPlugin.h>
```

Static Public Member Functions

- static DDS_TypeCode * [get_typecode](#) ()
This function returns the TypeCode.

6.47.1 Detailed Description

This class offers the functions needed by DDS middleware to use the class [RequestHeaderPlugin](#).

6.47.2 Member Function Documentation

6.47.2.1 static DDS_TypeCode* eprosima::rpc::protocol::dds::RequestHeaderPlugin::get_typecode () [static]

This function returns the TypeCode.

Returns

The TypeCode.

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

- includetmp/rpcdds/protocols/dds/MessageHeaderPlugin.h

6.48 eprosima::rpc::protocol::dds::SampleIdentity_t Class Reference

This class is used to identify clients.

```
#include <MessageHeader.h>
```

Public Member Functions

- [SampleIdentity_t](#) ()
Default constructor.
- [SampleIdentity_t](#) (const [SampleIdentity_t](#) &id)
Copy constructor.
- [SampleIdentity_t](#) ([SampleIdentity_t](#) &&id)
Copy constructor.
- [~SampleIdentity_t](#) ()
Destructor.
- [SampleIdentity_t](#) & operator= (const [SampleIdentity_t](#) &id)
Copy assignment.
- [SampleIdentity_t](#) & operator= ([SampleIdentity_t](#) &&id)
Copy assignment.
- void **guid** (const [GUID_t](#) &_guid)
- void **guid** ([GUID_t](#) &&_guid)
- const [GUID_t](#) & **guid** () const
This function returns the client identifier.
- [GUID_t](#) & **guid** ()
This function returns the client identifier.
- void **sequence_number** (int64_t _sequence_number)
This function sets the fourth value of the client identifier.
- int64_t **sequence_number** () const
This function returns the fourth value of the client identifier.
- int64_t & **sequence_number** ()
This function returns the fourth value of the client identifier.
- void **serialize** (eprosima::fastcdr::Cdr &cdr) const
This function serializes the [SampleIdentity_t](#) object using CDR serialization.
- void **deserialize** (eprosima::fastcdr::Cdr &cdr)
This function deserializes the [SampleIdentity_t](#) object using CDR serialization.

Static Public Member Functions

- static size_t **getMaxCdrSerializedSize** (size_t current_alignment)
This function returns the maximum serialized size of a [SampleIdentity_t](#) object depending on the buffer alignment.

6.48.1 Detailed Description

This class is used to identify clients.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 eprosima::rpc::protocol::dds::SampleIdentity_t::SampleIdentity_t (const SampleIdentity_t & id)

Copy constructor.

Parameters

<i>id</i>	SampleIdentity_t object to be copied.
-----------	---

6.48.2.2 `eprosima::rpc::protocol::dds::SampleIdentity_t::SampleIdentity_t (SampleIdentity_t && id)`

Copy constructor.

Parameters

<i>id</i>	SampleIdentity_t object to be copied.
-----------	---

6.48.3 Member Function Documentation

6.48.3.1 `void eprosima::rpc::protocol::dds::SampleIdentity_t::deserialize (eprosima::fastcdr::Cdr & cdr)`

This function deserializes the [SampleIdentity_t](#) object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.48.3.2 `static size_t eprosima::rpc::protocol::dds::SampleIdentity_t::getMaxCdrSerializedSize (size_t current_alignment)`
[static]

This function returns the maximum serialized size of a [SampleIdentity_t](#) object depending on the buffer alignment.

Parameters

<i>current_ - alignment</i>	Buffer alignment.
-----------------------------	-------------------

Returns

Maximum serialized size.

6.48.3.3 `const GUID_t& eprosima::rpc::protocol::dds::SampleIdentity_t::guid () const` [inline]

This function returns the client identifier.

Returns

Client identifier

6.48.3.4 `GUID_t& eprosima::rpc::protocol::dds::SampleIdentity_t::guid ()` [inline]

This function returns the client identifier.

Returns

Client identifier

6.48.3.5 `SampleIdentity_t& eprosima::rpc::protocol::dds::SampleIdentity_t::operator= (const SampleIdentity_t & id)`

Copy assignment.

Parameters

<i>id</i>	SampleIdentity_t object to be copied.
-----------	---

6.48.3.6 [SampleIdentity_t](#) & eprosima::rpc::protocol::dds::SampleIdentity_t::operator= ([SampleIdentity_t](#) && *id*)

Copy assignment.

Parameters

<i>id</i>	SampleIdentity_t object to be copied.
-----------	---

6.48.3.7 void eprosima::rpc::protocol::dds::SampleIdentity_t::sequence_number (int64_t *sequence_number*) [inline]

This function sets the fourth value of the client identifier.

Parameters

<i>_value_4</i>	Fourth value of the client identifier.
-----------------	--

6.48.3.8 int64_t eprosima::rpc::protocol::dds::SampleIdentity_t::sequence_number () const [inline]

This function returns the fourth value of the client identifier.

Returns

Fourth value of the client identifier.

6.48.3.9 int64_t& eprosima::rpc::protocol::dds::SampleIdentity_t::sequence_number () [inline]

This function returns the fourth value of the client identifier.

Returns

Fourth value of the client identifier.

6.48.3.10 void eprosima::rpc::protocol::dds::SampleIdentity_t::serialize (eprosima::fastcdr::Cdr & *cdr*) const

This function serializes the [SampleIdentity_t](#) object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

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

- includetmp/rpcdds/protocols/dds/MessageHeader.h

6.49 eprosima::rpc::protocol::dds::SampleIdentity_tPlugin Class Reference

Static Public Member Functions

- static DDS_TypeCode * [get_typecode](#) ()

This function returns the TypeCode.

6.49.1 Member Function Documentation

6.49.1.1 static DDS_TypeCode* eprosima::rpc::protocol::dds::SampleIdentity_tPlugin::get_typecode () [static]

This function returns the TypeCode.

Returns

The TypeCode.

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

- includetmp/rpcdds/protocols/dds/MessageHeaderPlugin.h

6.50 eprosima::rpc::server::Server Class Reference

This class implements the common functionalities that any server has.

```
#include <Server.h>
```

Inheritance diagram for eprosima::rpc::server::Server:

Public Member Functions

- void [serve](#) ()

This function makes the server starts listening requests.

Exceptions

eprosima::rpc::exception::InitializeException	<i>This exception is thrown when the initialization fails for any reason.</i>
---	---

- void [stop](#) ()

This function closes the server's communications.

Static Public Member Functions

- static void [process](#) ([Server](#) &server, void *data, [eprosima::rpc::transport::Endpoint](#) &endpoint)

This callback is invoked by the ServerStrategy. It processes a request.

Protected Member Functions

- [Server](#) ([eprosima::rpc::strategy::ServerStrategy](#) &strategy, [eprosima::rpc::transport::ServerTransport](#) &transport, [eprosima::rpc::protocol::Protocol](#) &protocol)

A constructor. The associated domain participant is created.

- virtual [~Server](#) ()

The default destructor.

6.50.1 Detailed Description

This class implements the common functionalities that any server has.

6.50.2 Constructor & Destructor Documentation

6.50.2.1 eprosima::rpc::server::Server::Server (eprosima::rpc::strategy::ServerStrategy & *strategy*, eprosima::rpc::transport::ServerTransport & *transport*, eprosima::rpc::protocol::Protocol & *protocol*)
[protected]

A constructor. The associated domain participant is created.

Parameters

<i>serviceName</i>	The service's name that proxies will use to connect with the server.
<i>strategy</i>	The strategy used by the server to execute new requests. This class doesn't delete this object in its destructor. It cannot be NULL.
<i>transport</i>	The transport that will use the server. This class doesn't delete this object in its destructor. If the pointer is NULL, then a default UDPTransport will be used.
<i>domainId</i>	The domain id's value that the server proxy will set in the domain participant.

Exceptions

<i>InitializeException</i>	This exception is thrown when the initialization was wrong.
----------------------------	---

6.50.3 Member Function Documentation

6.50.3.1 `static void eprosimarpc::server::Server::process (Server & server, void * data, eprosimarpc::transport::Endpoint & endpoint)` [static]

This callback is invoked by the ServerStrategy. It processes a request.

Parameters

<i>server</i>	The invoked server.
<i>data</i>	The request data.
<i>endpoint</i>	The request endpoint.

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

- `includetmp/rpcdds/server/Server.h`

6.51 eprosimarpc::exception::ServerInternalException Class Reference

This class is thrown as an exception when there is an error in the server side.

```
#include <ServerInternalException.h>
```

Inheritance diagram for eprosimarpc::exception::ServerInternalException:

Collaboration diagram for eprosimarpc::exception::ServerInternalException:

Public Member Functions

- FASTRPC_DIIAPI [ServerInternalException](#) (const std::string &message)
Default constructor.
- FASTRPC_DIIAPI [ServerInternalException](#) (const [ServerInternalException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [ServerInternalException](#) ([ServerInternalException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [ServerInternalException](#) & operator= (const [ServerInternalException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [ServerInternalException](#) & operator= ([ServerInternalException](#) &&ex)
Assignment operation.
- virtual FASTRPC_DIIAPI [~ServerInternalException](#) () throw ()
Default destructor.

- virtual FASTRPC_DllAPI void [raise](#) () const
This function throws the object as an exception.

Additional Inherited Members

6.51.1 Detailed Description

This class is thrown as an exception when there is an error in the server side.

6.51.2 Constructor & Destructor Documentation

6.51.2.1 FASTRPC_DllAPI eprosimarpc::exception::ServerInternalException::ServerInternalException (const std::string & *message*) [inline]

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.51.2.2 FASTRPC_DllAPI eprosimarpc::exception::ServerInternalException::ServerInternalException (const ServerInternalException & *ex*)

Default copy constructor.

Parameters

<i>ex</i>	ServerInternalException that will be copied.
-----------	--

6.51.2.3 FASTRPC_DllAPI eprosimarpc::exception::ServerInternalException::ServerInternalException (ServerInternalException && *ex*)

Default move constructor.

Parameters

<i>ex</i>	ServerInternalException that will be moved.
-----------	---

6.51.3 Member Function Documentation

6.51.3.1 FASTRPC_DllAPI ServerInternalException& eprosimarpc::exception::ServerInternalException::operator= (const ServerInternalException & *ex*)

Assignment operation.

Parameters

<i>ex</i>	ServerInternalException that will be copied.
-----------	--

6.51.3.2 FASTRPC_DllAPI ServerInternalException& eprosimarpc::exception::ServerInternalException::operator= (ServerInternalException && *ex*)

Assignment operation.

Parameters

<i>ex</i>	ServerInternalException that will be moved.
-----------	---

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

- `includetmp/rpcdds/exceptions/ServerInternalException.h`

6.52 eprosima::rpc::exception::ServerNotFoundException Class Reference

This class is thrown as an exception when the server is not found.

```
#include <ServerNotFoundException.h>
```

Inheritance diagram for `eprosima::rpc::exception::ServerNotFoundException`:

Collaboration diagram for `eprosima::rpc::exception::ServerNotFoundException`:

Public Member Functions

- FASTRPC_DIIAPI [ServerNotFoundException](#) (const std::string &message)
Default constructor.
- FASTRPC_DIIAPI [ServerNotFoundException](#) (const [ServerNotFoundException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [ServerNotFoundException](#) ([ServerNotFoundException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [ServerNotFoundException](#) & operator= (const [ServerNotFoundException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [ServerNotFoundException](#) & operator= ([ServerNotFoundException](#) &&ex)
Assignment operation.
- virtual FASTRPC_DIIAPI ~[ServerNotFoundException](#) () throw ()
Default constructor.
- virtual FASTRPC_DIIAPI void [raise](#) () const
This function throws the object as an exception.

Additional Inherited Members

6.52.1 Detailed Description

This class is thrown as an exception when the server is not found.

6.52.2 Constructor & Destructor Documentation

- 6.52.2.1 FASTRPC_DIIAPI `eprosima::rpc::exception::ServerNotFoundException::ServerNotFoundException (const std::string & message) [inline]`

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.52.2.2 FASTRPC_DllAPI eprosimarpc::exception::ServerNotFoundException::ServerNotFoundException (const ServerNotFoundException & ex)

Default copy constructor.

Parameters

<i>ex</i>	ServerNotFoundException that will be copied.
-----------	--

6.52.2.3 FASTRPC_DllAPI eprosimarpc::exception::ServerNotFoundException::ServerNotFoundException (ServerNotFoundException && ex)

Default move constructor.

Parameters

<i>ex</i>	ServerNotFoundException that will be moved.
-----------	---

6.52.3 Member Function Documentation

6.52.3.1 FASTRPC_DllAPI ServerNotFoundException& eprosimarpc::exception::ServerNotFoundException::operator= (const ServerNotFoundException & ex)

Assignment operation.

Parameters

<i>ex</i>	ServerNotFoundException that will be copied.
-----------	--

6.52.3.2 FASTRPC_DllAPI ServerNotFoundException& eprosimarpc::exception::ServerNotFoundException::operator= (ServerNotFoundException && ex)

Assignment operation.

Parameters

<i>ex</i>	ServerNotFoundException that will be moved.
-----------	---

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

- includetmp/rpcdds/exceptions/ServerNotFoundException.h

6.53 eprosimarpc::transport::dds::ServerProcedureEndpoint Class Reference

This class represents a remote endpoint used by a proxy. Also this class encapsulate the DDS datawriter and the DDS datareader.

```
#include <ServerProcedureEndpoint.h>
```

Inheritance diagram for eprosimarpc::transport::dds::ServerProcedureEndpoint:

Collaboration diagram for eprosimarpc::transport::dds::ServerProcedureEndpoint:

Public Member Functions

- FASTRPC_DIIAPI [ServerProcedureEndpoint](#) ([ServerTransport](#) &transport)
Default constructor.
- virtual FASTRPC_DIIAPI [~ServerProcedureEndpoint](#) ()
Default destructor.
- FASTRPC_DIIAPI int [initialize](#) (const char *name, const char *writertypename, const char *readertypename, Transport::Create_data create_data, Transport::Destroy_data destroy_data, Transport::ProcessFunc, int dataSize)
Initializes the endpoint.
- FASTRPC_DIIAPI int [start](#) (const char *const &serviceName, const char *const &instanceName)
This method creates the DDS entities needed to run this DDS [Endpoint](#).
- FASTRPC_DIIAPI void [stop](#) ()
This method deletes the DDS entities needed to run this DDS [Endpoint](#).
- FASTRPC_DIIAPI Transport::ProcessFunc [getProcessFunc](#) ()
Gets the callback used to processes a request.
- FASTRPC_DIIAPI int [sendReply](#) (void *data)
Sends the reply.
- virtual FASTRPC_DIIAPI void [on_data_available](#) (DDSDataReader *reader)
DDS callback.
- virtual FASTRPC_DIIAPI void [on_requested_deadline_missed](#) (DDSDataReader *reader, const DDS_RequestedDeadlineMissedStatus &status)
DDS callback.
- virtual FASTRPC_DIIAPI void [on_requested_incompatible_qos](#) (DDSDataReader *reader, const DDS_RequestedIncompatibleQosStatus &status)
DDS callback.
- virtual FASTRPC_DIIAPI void [on_sample_rejected](#) (DDSDataReader *reader, const DDS_SampleRejectedStatus &status)
DDS callback.
- virtual FASTRPC_DIIAPI void [on_liveliness_changed](#) (DDSDataReader *reader, const DDS_LivelinessChangedStatus &status)
DDS callback.
- virtual FASTRPC_DIIAPI void [on_sample_lost](#) (DDSDataReader *reader, const DDS_SampleLostStatus &status)
DDS callback.
- virtual FASTRPC_DIIAPI void [on_subscription_matched](#) (DDSDataReader *reader, const DDS_SubscriptionMatchedStatus &status)
DDS callback.

Additional Inherited Members

6.53.1 Detailed Description

This class represents a remote endpoint used by a proxy. Also this class encapsulate the DDS datawriter and the DDS datareader.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 FASTRPC_DIIAPI eprosima::rpc::transport::dds::ServerProcedureEndpoint::ServerProcedureEndpoint ([ServerTransport](#) & *transport*)

Default constructor.

Parameters

<i>Transport</i>	that creates the proxy procedure endpoint. It cannot be NULL.
----------------------------------	---

6.53.3 Member Function Documentation

6.53.3.1 **FASTRPC_DIIAPI** `Transport::ProcessFunc eprosimarpc::transport::dds::ServerProcedureEndpoint::getProcessFunc () [inline]`

Gets the callback used to processes a request.

Returns

Function callback used to processes a request.

6.53.3.2 **FASTRPC_DIIAPI** `int eprosimarpc::transport::dds::ServerProcedureEndpoint::initialize (const char * name, const char * writertypename, const char * readertypename, Transport::Create_data create_data, Transport::Destroy_data destroy_data, Transport::ProcessFunc , int dataSize)`

Initializes the endpoint.

TODO Actualizar

Parameters

<i>name</i>	The name associated with this procedure endpoint. It cannot be NULL.
<i>writertypename</i>	The type name of the topic that the procedure endpoint uses in the datawriter. It cannot be NULL.
<i>readertypename</i>	The type name of the topic that the procedure endpoint uses in the datareader. It cannot be NULL.
<i>initialize_data</i>	Pointer to the function to initialize DataReader received data
<i>finalize_data</i>	Pointer to the function to finalize DataReader received data
<i>ProcessFunc</i>	Pointer to the function invoked when a message is received from the server
<i>dataSize</i>	Size of the DataReader data structure

6.53.3.3 **FASTRPC_DIIAPI** `int eprosimarpc::transport::dds::ServerProcedureEndpoint::sendReply (void * data)`

Sends the reply.

Parameters

<i>serviceName</i>	Name of the service.
--------------------	----------------------

6.53.3.4 **FASTRPC_DIIAPI** `int eprosimarpc::transport::dds::ServerProcedureEndpoint::start (const char *const & serviceName, const char *const & instanceName)`

This method creates the DDS entities needed to run this DDS [Endpoint](#).

Parameters

<i>serviceName</i>	Name of the service.
--------------------	----------------------

6.53.3.5 **FASTRPC_DIIAPI** `void eprosimarpc::transport::dds::ServerProcedureEndpoint::stop ()`

This method deletes the DDS entities needed to run this DDS [Endpoint](#).

Parameters

<i>serviceName</i>	Name of the service.
--------------------	----------------------

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

- `includetmp/rpcdds/transports/dds/components/ServerProcedureEndpoint.h`

6.54 eprosima::rpc::strategy::ServerStrategy Class Reference

This class is the base of all classes that implement a server strategy. that could be used by the server.

```
#include <ServerStrategy.h>
```

Inheritance diagram for eprosima::rpc::strategy::ServerStrategy:

Public Member Functions

- [ServerStrategy](#) ()
Default constructor.
- virtual [~ServerStrategy](#) ()
Default destructor.
- virtual [ServerStrategyImpl](#) * [getImpl](#) ()=0
Gets the implementation of the strategy using Boost library.

6.54.1 Detailed Description

This class is the base of all classes that implement a server strategy. that could be used by the server.

6.54.2 Member Function Documentation

6.54.2.1 `virtual ServerStrategyImpl* eprosima::rpc::strategy::ServerStrategy::getImpl () [pure virtual]`

Gets the implementation of the strategy using Boost library.

Returns

Implementation of the strategy.

Implemented in [eprosima::rpc::strategy::ThreadPoolStrategy](#), [eprosima::rpc::strategy::SingleThreadStrategy](#), and [eprosima::rpc::strategy::ThreadPerRequestStrategy](#).

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

- `includetmp/rpcdds/strategies/ServerStrategy.h`

6.55 eprosima::rpc::strategy::ServerStrategyImpl Class Reference

This class is the base of all classes that implement a server strategy. that could be used by the server.

```
#include <ServerStrategyImpl.h>
```

Public Member Functions

- [ServerStrategyImpl](#) ()
Default constructor.
- virtual [~ServerStrategyImpl](#) ()
Default destructor.
- virtual void [schedule](#) (boost::function< void()> callback)=0
This function schedules an incoming request. This function has to be implemented by the derived classes.

6.55.1 Detailed Description

This class is the base of all classes that implement a server strategy. that could be used by the server.

6.55.2 Member Function Documentation

6.55.2.1 virtual void eprosimarpc::strategy::ServerStrategyImpl::schedule (boost::function< void()> *callback*) [pure virtual]

This function schedules an incoming request. This function has to be implemented by the derived classes.

Parameters

<i>callback</i>	The Server's method to invoke when a request arrives.
-----------------	---

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

- includetmp/rpcdds/strategies/ServerStrategyImpl.h

6.56 eprosimarpc::exception::ServerTimeoutException Class Reference

This class is thrown as an exception when the remote procedure call exceeds the maximum time.

```
#include <ServerTimeoutException.h>
```

Inheritance diagram for eprosimarpc::exception::ServerTimeoutException:

Collaboration diagram for eprosimarpc::exception::ServerTimeoutException:

Public Member Functions

- FASTRPC_DIIAPI [ServerTimeoutException](#) (const std::string &message)
Default constructor.
- FASTRPC_DIIAPI [ServerTimeoutException](#) (const [ServerTimeoutException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [ServerTimeoutException](#) ([ServerTimeoutException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [ServerTimeoutException](#) & operator= (const [ServerTimeoutException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [ServerTimeoutException](#) & operator= ([ServerTimeoutException](#) &&ex)
Assignment operation.
- virtual FASTRPC_DIIAPI [~ServerTimeoutException](#) () throw ()
Default constructor.

- virtual FASTRPC_DllAPI void [raise](#) () const
This function throws the object as an exception.

Additional Inherited Members

6.56.1 Detailed Description

This class is thrown as an exception when the remote procedure call exceeds the maximum time.

6.56.2 Constructor & Destructor Documentation

6.56.2.1 FASTRPC_DllAPI eprosima::rpc::exception::ServerTimeoutException::ServerTimeoutException (const std::string & *message*) [inline]

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.56.2.2 FASTRPC_DllAPI eprosima::rpc::exception::ServerTimeoutException::ServerTimeoutException (const ServerTimeoutException & *ex*)

Default copy constructor.

Parameters

<i>ex</i>	ServerTimeoutException that will be copied.
-----------	---

6.56.2.3 FASTRPC_DllAPI eprosima::rpc::exception::ServerTimeoutException::ServerTimeoutException (ServerTimeoutException && *ex*)

Default move constructor.

Parameters

<i>ex</i>	ServerTimeoutException that will be moved.
-----------	--

6.56.3 Member Function Documentation

6.56.3.1 FASTRPC_DllAPI ServerTimeoutException& eprosima::rpc::exception::ServerTimeoutException::operator= (const ServerTimeoutException & *ex*)

Assignment operation.

Parameters

<i>ex</i>	ServerTimeoutException that will be copied.
-----------	---

6.56.3.2 FASTRPC_DllAPI ServerTimeoutException& eprosima::rpc::exception::ServerTimeoutException::operator= (ServerTimeoutException && *ex*)

Assignment operation.

Parameters

<i>ex</i>	ServerTimeoutException that will be moved.
-----------	--

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

- `includetmp/rpcdds/exceptions/ServerTimeoutException.h`

6.57 eprosima::rpc::transport::ServerTransport Class Reference

This interface is the base of all classes that implement a transport that can be used by the server.

```
#include <ServerTransport.h>
```

Inheritance diagram for eprosima::rpc::transport::ServerTransport:

Collaboration diagram for eprosima::rpc::transport::ServerTransport:

Public Member Functions

- [ServerTransport](#) ()
Default constructor.
- virtual [~ServerTransport](#) ()
Default destructor.
- void [setStrategy](#) (eprosima::rpc::strategy::ServerStrategy &strategy)
Sets the threading strategy.
- void [linkProtocol](#) (eprosima::rpc::protocol::Protocol &protocol)
Sets the communication protocol.
- [eprosima::rpc::protocol::Protocol](#) & [getLinkedProtocol](#) ()
Gets the communication protocol.
- [eprosima::rpc::strategy::ServerStrategy](#) & [getStrategy](#) () const
Gets the threading strategy.
- [ServerTransport_Callback](#) [getCallback](#) () const
Gets the callback that will proccess the requests.
- void [setCallback](#) ([ServerTransport_Callback](#) callback)
Gets the callback that will proccess the requests.
- [TransportBehaviour](#) [getBehaviour](#) () const
This function returns the behaviour of the transport.
- virtual const char * [getType](#) () const =0
This function returns the type of the transport. This function has to be implemented by the child classes.
- virtual void [run](#) ()=0
This method runs the TCP server needed for the HTTP connections.
- virtual void [stop](#) ()=0
This method stops the TCP server needed for the HTTP connections.
- virtual void [sendReply](#) (void *data, size_t dataLength, [Endpoint](#) *endpoint)=0
This function is used to send a reply to a proxy.
- virtual int [receive](#) (char *buffer, size_t bufferLength, size_t &dataToRead, [Endpoint](#) *endpoint)=0
This function is used to send a reply to a proxy.

6.57.1 Detailed Description

This interface is the base of all classes that implement a transport that can be used by the server.

6.57.2 Member Function Documentation

6.57.2.1 TransportBehaviour `eprosima::rpc::transport::ServerTransport::getBehaviour () const` `[inline]`,
`[virtual]`

This function returns the behaviour of the transport.

Returns

The behaviour of the transport.

Implements [eprosima::rpc::transport::Transport](#).

6.57.2.2 ServerTransport_Callback `eprosima::rpc::transport::ServerTransport::getCallback () const` `[inline]`

Gets the callback that will process the requests.

Returns

Callback that will process the requests.

6.57.2.3 eprosima::rpc::protocol::Protocol& `eprosima::rpc::transport::ServerTransport::getLinkedProtocol ()`
`[inline]`

Gets the communication protocol.

Returns

Communication protocol.

6.57.2.4 eprosima::rpc::strategy::ServerStrategy& `eprosima::rpc::transport::ServerTransport::getStrategy () const`
`[inline]`

Gets the threading strategy.

Returns

Threading strategy.

6.57.2.5 void `eprosima::rpc::transport::ServerTransport::linkProtocol (eprosima::rpc::protocol::Protocol & protocol)`
`[inline]`

Sets the communication protocol.

Parameters

<i>protocol</i>	Communication protocol.
-----------------	-------------------------

6.57.2.6 virtual int `eprosima::rpc::transport::ServerTransport::receive (char * buffer, size_t bufferLength, size_t & dataToRead, Endpoint * endpoint)` `[pure virtual]`

This function is used to send a reply to a proxy.

Parameters

<i>buffer</i>	Buffer to allocate the received data
<i>bufferLength</i>	Size of the buffer
<i>dataToRead</i>	Size of the data to read
<i>endpoint</i>	Endpoint to receive the data from

Implemented in [eprosimarpc::transport::dds::ServerTransport](#).

6.57.2.7 `virtual void eprosimarpc::transport::ServerTransport::sendReply (void * data, size_t dataLength, Endpoint * endpoint) [pure virtual]`

This function is used to send a reply to a proxy.

Parameters

<i>data</i>	Response to send.
<i>dataLength</i>	Length of the data to send.
<i>endpoint</i>	Target endpoint to send the data to.

Implemented in [eprosimarpc::transport::dds::ServerTransport](#).

6.57.2.8 `void eprosimarpc::transport::ServerTransport::setCallback (ServerTransport_Callback callback) [inline]`

Gets the callback that will process the requests.

Parameters

<i>Callback</i>	Callback that will process the requests.
-----------------	--

6.57.2.9 `void eprosimarpc::transport::ServerTransport::setStrategy (eprosimarpc::strategy::ServerStrategy & strategy) [inline]`

Sets the threading strategy.

Parameters

<i>strategy</i>	Threading strategy.
-----------------	---------------------

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

- `includetmp/rpcdds/transport/ServerTransport.h`

6.58 eprosimarpc::transport::dds::ServerTransport Class Reference

This class is the base of all classes that implement a transport using DDS. This transport can be used by the servers.

```
#include <ServerTransport.h>
```

Inheritance diagram for `eprosimarpc::transport::dds::ServerTransport`:

Collaboration diagram for `eprosimarpc::transport::dds::ServerTransport`:

Public Member Functions

- `virtual FASTRPC_DIIAPI ~ServerTransport ()`
Default destructor.

- virtual FASTRPC_DIIAPI const char * [getType](#) () const

This function returns the type of the transport. This function has to be implemented by the child classes.

- FASTRPC_DIIAPI [eprosima::rpc::transport::Endpoint](#) * [createProcedureEndpoint](#) (const char *name, const char *writertypename, const char *readertypename, bool eprosima_types, Transport::Create_data create_data, Transport::Copy_data copy_data, Transport::Destroy_data destroy_data, Transport::ProcessFunc processFunc, int dataSize)

This function creates a new proxy procedure endpoint. This proxy procedure endpoint manages the DDS datawriter and the DDS datareader.

- FASTRPC_DIIAPI void [process](#) ([ServerProcedureEndpoint](#) *endpoint, void *data)

This method is invoked once for each incoming request.

- FASTRPC_DIIAPI void [run](#) ()

This method starts all the DDS Datawriters and Datareaders.

- FASTRPC_DIIAPI void [stop](#) ()

This function does not apply to DDS transport.

- void [sendReply](#) (void *data, size_t dataLength, [Endpoint](#) *endpoint)

This function is used to send a reply to a proxy.

- FASTRPC_DIIAPI int [receive](#) (char *buffer, size_t bufferLength, size_t &dataToRead, [Endpoint](#) *endpoint)

This function does not apply to DDS transport.

Protected Member Functions

- virtual FASTRPC_DIIAPI int [setTransport](#) (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant)=0

This abstract function sets the QoS of DDS to use a specific transport.

- FASTRPC_DIIAPI [ServerTransport](#) (const char *const &serviceName, const char *const &instanceName, int domainId=0)

Default constructor.

Additional Inherited Members

6.58.1 Detailed Description

This class is the base of all classes that implement a transport using DDS. This transport can be used by the servers.

6.58.2 Constructor & Destructor Documentation

- 6.58.2.1 FASTRPC_DIIAPI [eprosima::rpc::transport::dds::ServerTransport::ServerTransport](#) (const char *const & *serviceName*, const char *const & *instanceName*, int *domainId* = 0) [protected]

Default constructor.

Parameters

<i>domainId</i>	Optional parameter that specifies the domain identifier that will be used in DDS.
-----------------	---

6.58.3 Member Function Documentation

6.58.3.1 `FASTRPC_DllAPI eprosimarpc::transport::Endpoint* eprosimarpc::transport::dds::ServerTransport::create-ProcedureEndpoint (const char * name, const char * writertypename, const char * readertypename, bool eprosimatypes, Transport::Create_data create_data, Transport::Copy_data copy_data, Transport::Destroy_data destroy_data, Transport::ProcessFunc processFunc, int dataSize) [virtual]`

This function creates a new proxy procedure endpoint. This proxy procedure endpoint manages the DDS datawriter and the DDS datareader.

TODO Actualizar

Parameters

<i>name</i>	The name associated with this proxy procedure endpoint. It cannot be NULL.
<i>writertypename</i>	The type name of the topic that the procedure endpoint uses in the datawriter. It cannot be NULL.
<i>readertypename</i>	The type name of the topic that the procedure endpoint uses in the datareader. It cannot be NULL.
<i>initialize_data</i>	Pointer to the function to initialize DataReader received data
<i>copy_data</i>	Pointer to the function used to copy the data when it is received.
<i>finalize_data</i>	Pointer to the function to finalize DataReader received data
<i>ProcessFunc</i>	Pointer to the function invoked when a message is received from the server
<i>dataSize</i>	Size of the DataReader data structure

Returns

0 if the function successfully works, -1 in other case TODO

Implements [eprosimarpc::transport::dds::Transport](#).

6.58.3.2 `FASTRPC_DllAPI void eprosimarpc::transport::dds::ServerTransport::process (ServerProcedureEndpoint * endpoint, void * data)`

This method is invoked once for each incoming request.

Parameters

<i>data</i>	The request data.
<i>endpoint</i>	The request endpoint.

6.58.3.3 `void eprosimarpc::transport::dds::ServerTransport::sendReply (void * data, size_t dataLength, Endpoint * endpoint) [virtual]`

This function is used to send a reply to a proxy.

Parameters

<i>data</i>	Data to send.
<i>dataLength</i>	Length of the data to send.
<i>endpoint</i>	Endpoint meant to send the data.

Implements [eprosimarpc::transport::ServerTransport](#).

6.58.3.4 `virtual FASTRPC_DllAPI int eprosimarpc::transport::dds::ServerTransport::setTransport (DDS_DomainParticipantQos & participantQos, DDSDomainParticipant * participant) [protected], [pure virtual]`

This abstract function sets the QoS of DDS to use a specific transport.

Parameters

<i>participantQos</i>	Reference to the DDS domain participant QoS.
<i>participant</i>	The domain participant that will be set to use a specific transport.

Implements [eprosima::rpc::transport::dds::Transport](#).

Implemented in [eprosima::rpc::transport::dds::TCPServerTransport](#), and [eprosima::rpc::transport::dds::UDP-ServerTransport](#).

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

- `includetmp/rpcdds/transport/dds/ServerTransport.h`

6.59 eprosima::rpc::strategy::SingleThreadStrategy Class Reference

This class implements the sigle thread strategy. The server uses a reception thread to execute all the requests.

```
#include <SingleThreadStrategy.h>
```

Inheritance diagram for `eprosima::rpc::strategy::SingleThreadStrategy`:

Collaboration diagram for `eprosima::rpc::strategy::SingleThreadStrategy`:

Public Member Functions

- [SingleThreadStrategy](#) ()
Default constructor.
- virtual [~SingleThreadStrategy](#) ()
Default destructor.
- [ServerStrategyImpl](#) * [getImpl](#) ()
Gets the implementation of the strategy using Boost library.

6.59.1 Detailed Description

This class implements the sigle thread strategy. The server uses a reception thread to execute all the requests.

6.59.2 Member Function Documentation

6.59.2.1 `ServerStrategyImpl* eprosima::rpc::strategy::SingleThreadStrategy::getImpl ()` [virtual]

Gets the implementation of the strategy using Boost library.

Returns

Implementation of the strategy

Implements [eprosima::rpc::strategy::ServerStrategy](#).

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

- `includetmp/rpcdds/strategies/SingleThreadStrategy.h`

6.60 eprosimarpc::exception::SystemException Class Reference

This abstract class is used to create internal FASTRPC exceptions.

```
#include <SystemException.h>
```

Inheritance diagram for eprosimarpc::exception::SystemException:

Collaboration diagram for eprosimarpc::exception::SystemException:

Public Member Functions

- virtual FASTRPC_DIIAPI [~SystemException](#) () throw ()
Default destructor.
- FASTRPC_DIIAPI const int32_t & [minor](#) () const
This function returns the number associated with the system exception.
- FASTRPC_DIIAPI void [minor](#) (const int32_t &minor)
This function sets the number that will be associated with the system exception.
- virtual FASTRPC_DIIAPI void [raise](#) () const =0
This function throws the object as an exception.
- virtual FASTRPC_DIIAPI const char * [what](#) () const throw ()
This function returns the error message.

Protected Member Functions

- FASTRPC_DIIAPI [SystemException](#) (const char *const &message)
Default constructor.
- FASTRPC_DIIAPI [SystemException](#) (const [SystemException](#) &ex)
Default copy constructor.
- FASTRPC_DIIAPI [SystemException](#) ([SystemException](#) &&ex)
Default move constructor.
- FASTRPC_DIIAPI [SystemException](#) (const char *const &message, int32_t [minor](#))
Constructor.
- FASTRPC_DIIAPI [SystemException](#) & [operator=](#) (const [SystemException](#) &ex)
Assignment operation.
- FASTRPC_DIIAPI [SystemException](#) & [operator=](#) ([SystemException](#) &&ex)
Assignment operation.

6.60.1 Detailed Description

This abstract class is used to create internal FASTRPC exceptions.

6.60.2 Constructor & Destructor Documentation

6.60.2.1 FASTRPC_DIIAPI eprosimarpc::exception::SystemException (const char *const & message) [explicit], [protected]

Default constructor.

Parameters

<i>message</i>	An error message. This message is copied.
----------------	---

6.60.2.2 FASTRPC_DllAPI eprosima::rpc::exception::SystemException::SystemException (const SystemException & ex) [protected]

Default copy constructor.

Parameters

<i>ex</i>	SystemException that will be copied.
-----------	--

6.60.2.3 FASTRPC_DllAPI eprosima::rpc::exception::SystemException::SystemException (SystemException && ex) [protected]

Default move constructor.

Parameters

<i>ex</i>	SystemException that will be moved.
-----------	---

6.60.2.4 FASTRPC_DllAPI eprosima::rpc::exception::SystemException::SystemException (const char *const & message, int32_t minor) [explicit], [protected]

Constructor.

Parameters

<i>message</i>	An error message. This message is copied.
<i>minor</i>	The number that will be associated with the system exception.

6.60.3 Member Function Documentation

6.60.3.1 FASTRPC_DllAPI const int32_t& eprosima::rpc::exception::SystemException::minor () const

This function returns the number associated with the system exception.

Returns

The number associated with the system exception.

6.60.3.2 FASTRPC_DllAPI void eprosima::rpc::exception::SystemException::minor (const int32_t & minor)

This function sets the number that will be associated with the system exception.

Parameters

<i>minor</i>	The number that will be associated with the system exception.
--------------	---

6.60.3.3 FASTRPC_DllAPI SystemException& eprosima::rpc::exception::SystemException::operator= (const SystemException & ex) [protected]

Assignment operation.

Parameters

<i>ex</i>	SystemException that will be copied.
-----------	--

6.60.3.4 FASTRPC_DIIAPI SystemException& eprosima::rpc::exception::SystemException::operator= (SystemException && ex) [protected]

Assignment operation.

Parameters

<i>ex</i>	SystemException that will be moved.
-----------	---

6.60.3.5 virtual FASTRPC_DIIAPI const char* eprosima::rpc::exception::SystemException::what () const throw () [virtual]

This function returns the error message.

Returns

The error message.

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

- includetmp/rpcdds/exceptions/SystemException.h

6.61 eprosima::rpc::protocol::dds::SystemExceptionCodePlugin Class Reference

Static Public Member Functions

- static DDS_TypeCode * [get_typecode](#) ()
This function returns the TypeCode.

6.61.1 Member Function Documentation

6.61.1.1 static DDS_TypeCode* eprosima::rpc::protocol::dds::SystemExceptionCodePlugin::get_typecode () [static]

This function returns the TypeCode.

Returns

The TypeCode.

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

- includetmp/rpcdds/protocols/dds/MessageHeaderPlugin.h

6.62 eprosima::rpc::transport::dds::TCPProxyTransport Class Reference

This class implements a transport using DDS over TCPv4. This transport can only be used by a server proxy.

```
#include <TCPProxyTransport.h>
```

Inheritance diagram for eprosima::rpc::transport::dds::TCPProxyTransport:

Collaboration diagram for eprosima::rpc::transport::dds::TCPProxyTransport:

Public Member Functions

- FASTRPC_DIIAPI [TCPProxyTransport](#) (const char *const &to_connect, const char *const &remoteServiceName, const char *const &instanceName, int domainId=0, long timeout=10000L)

Default constructor for the proxies.

- virtual FASTRPC_DIIAPI [~TCPProxyTransport](#) ()

Default destructor.

- virtual FASTRPC_DIIAPI int [setTransport](#) (DDS::DomainParticipantQos &participantQos, DDS::DomainParticipant *participant)

This function sets the DDS' QoS to use the TCPv4 transport.

Additional Inherited Members

6.62.1 Detailed Description

This class implements a transport using DDS over TCPv4. This transport can only be used by a server proxy.

6.62.2 Constructor & Destructor Documentation

- 6.62.2.1 FASTRPC_DIIAPI `eprosima::rpc::transport::dds::TCPProxyTransport::TCPProxyTransport (const char *const &to_connect, const char *const &remoteServiceName, const char *const &instanceName, int domainId = 0, long timeout = 10000L)`

Default constructor for the proxies.

Parameters

<i>to_connect</i>	Public address and port where the server can be found by the proxy. By example: "218.18.-3.133:7600"
<i>remoteServiceName</i>	Name of the remote service
<i>domainId</i>	Optional parameter that specifies the domain identifier to be used in DDS.
<i>timeout</i>	The time in milliseconds to wait for the reply.

6.62.3 Member Function Documentation

- 6.62.3.1 virtual FASTRPC_DIIAPI int `eprosima::rpc::transport::dds::TCPProxyTransport::setTransport (DDS::DomainParticipantQos &participantQos, DDS::DomainParticipant *participant)` [virtual]

This function sets the DDS' QoS to use the TCPv4 transport.

Parameters

<i>participantQos</i>	Reference to the DDS domain participant QoS.
<i>participant</i>	The domain participant that will be set to use TCPv4 transport.

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

- `includetmp/rpcdds/transport/dds/TCPProxyTransport.h`

6.63 eprosima::rpc::transport::dds::TCPServerTransport Class Reference

This class implements a transport using DDS over TCPv4. This transport can only be used by a server.

```
#include <TCPServerTransport.h>
```

Inheritance diagram for eprosima::rpc::transport::dds::TCPServerTransport:

Collaboration diagram for eprosima::rpc::transport::dds::TCPServerTransport:

Public Member Functions

- FASTRPC_DIIAPI [TCPServerTransport](#) (const char *const &public_address, const char *const &server_bind_port, const char *const &serviceName, const char *const &instanceName, int domainId=0)
Default constructor for servers.
- virtual FASTRPC_DIIAPI [~TCPServerTransport](#) ()
Default destructor.
- virtual FASTRPC_DIIAPI int [setTransport](#) (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant)
This function sets the QoS to use the TCPv4 transport.

Additional Inherited Members

6.63.1 Detailed Description

This class implements a transport using DDS over TCPv4. This transport can only be used by a server.

6.63.2 Constructor & Destructor Documentation

- 6.63.2.1 FASTRPC_DIIAPI eprosima::rpc::transport::dds::TCPServerTransport::TCPServerTransport (const char *const &public_address, const char *const &server_bind_port, const char *const &serviceName, const char *const &instanceName, int domainId = 0)

Default constructor for servers.

Parameters

<i>public_address</i>	Public address and port of the server. The server should be accesible in this address. The user has to configure his router for this purpose. For example: "218.18.3.133:7600"
<i>server_bind_port</i>	Port used by the server in his machine. This port will be used in the router for port forwarding between the public port and this port.
<i>serviceName</i>	Name of the remote service
<i>domainId</i>	Optional parameter that specifies the domain identifier to be used in DDS.

6.63.3 Member Function Documentation

- 6.63.3.1 virtual FASTRPC_DIIAPI int eprosima::rpc::transport::dds::TCPServerTransport::setTransport (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant) [virtual]

This function sets the QoS to use the TCPv4 transport.

Parameters

<i>participantQos</i>	Reference to the DDS domain participant QoS.
<i>participant</i>	The domain participant that will be set to use TCPv4 transport.

Implements [eprosima::rpc::transport::dds::ServerTransport](#).

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

- includetmp/rpcdds/transport/dds/TCPServerTransport.h

6.64 eprosima::rpc::strategy::ThreadPerRequestStrategy Class Reference

This class implements the thread per request strategy. The server creates a new thread for every new incoming request.

```
#include <ThreadPerRequestStrategy.h>
```

Inheritance diagram for eprosima::rpc::strategy::ThreadPerRequestStrategy:

Collaboration diagram for eprosima::rpc::strategy::ThreadPerRequestStrategy:

Public Member Functions

- [ThreadPerRequestStrategy](#) ()
Default constructor.
- virtual [~ThreadPerRequestStrategy](#) ()
Default destructor.
- [ServerStrategyImpl](#) * [getImpl](#) ()
Gets the implementation of the strategy using Boost library.

6.64.1 Detailed Description

This class implements the thread per request strategy. The server creates a new thread for every new incoming request.

6.64.2 Member Function Documentation

6.64.2.1 [ServerStrategyImpl](#)* eprosima::rpc::strategy::ThreadPerRequestStrategy::getImpl () [virtual]

Gets the implementation of the strategy using Boost library.

Returns

Strategy implementation.

Implements [eprosima::rpc::strategy::ServerStrategy](#).

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

- `includetmp/rpcdds/strategies/ThreadPerRequestStrategy.h`

6.65 eprosima::rpc::strategy::ThreadPoolStrategy Class Reference

This class implements a thread pool strategy. The server schedules the incoming requests in a free thread of the thread pool.

```
#include <ThreadPoolStrategy.h>
```

Inheritance diagram for eprosima::rpc::strategy::ThreadPoolStrategy:

Collaboration diagram for eprosima::rpc::strategy::ThreadPoolStrategy:

Public Member Functions

- [ThreadPoolStrategy](#) (unsigned int threadCount=FASTRPC_MIN_THREADS_DEFAULT)

Default constructor.

- [~ThreadPoolStrategy](#) ()

Default destructor.

- [ServerStrategyImpl](#) * [getImpl](#) ()

Gets the implementation of the strategy using Boost library.

6.65.1 Detailed Description

This class implements a thread pool strategy. The server schedules the incoming requests in a free thread of the thread pool.

6.65.2 Constructor & Destructor Documentation

6.65.2.1 `eprosima::rpc::strategy::ThreadPoolStrategy::ThreadPoolStrategy (unsigned int threadCount = FASTRPC_MIN_THREADS_DEFAULT)`

Default constructor.

Parameters

<i>threadCount</i>	Number of threads the thread pool will manage. Default value: 5.
--------------------	--

6.65.3 Member Function Documentation

6.65.3.1 `ServerStrategyImpl* eprosima::rpc::strategy::ThreadPoolStrategy::getImpl () [virtual]`

Gets the implementation of the strategy using Boost library.

Returns

Implementation of the strategy.

Implements [eprosima::rpc::strategy::ServerStrategy](#).

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

- `includetmp/rpcdds/strategies/ThreadPoolStrategy.h`

6.66 eprosima::rpc::transport::dds::Transport Class Reference

This class is the base of all classes that implement a transport using DDS. This transport could be used by both proxies and servers.

```
#include <Transport.h>
```

Inheritance diagram for `eprosima::rpc::transport::dds::Transport`:

Public Types

- `typedef void (* Create_data)(void)`
- `typedef void (* Copy_data)(void *dst, void *src)`
- `typedef void (* Destroy_data)(void *data)`
- `typedef void (* ProcessFunc)(eprosima::rpc::protocol::Protocol &, void *, eprosima::rpc::transport::Endpoint *)`

Public Member Functions

- virtual [~Transport](#) ()
Default destructor.
- void [initialize](#) ()
Initializes all the DDS elements: creates the topic, the participant, the publisher and the subscriber.
- DDSDomainParticipant * [getParticipant](#) () const
Gets the domain participant.
- DDSPublisher * [getPublisher](#) () const
Gets the publisher.
- DDSSubscriber * [getSubscriber](#) () const
Gets the subscriber.
- virtual [eprosima::rpc::transport::Endpoint](#) * [createProcedureEndpoint](#) (const char *name, const char *writertypename, const char *readertypename, bool eprosima_types, Create_data create_data, Copy_data copy_data, Destroy_data destroy_data, ProcessFunc processFunc, int dataSize)=0
This function creates a new procedure endpoint. This proxy procedure endpoint manages the DDS datawriter and the DDS datareader.

Protected Member Functions

- virtual int [setTransport](#) (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant)=0
This abstract function sets the QoS to use a specific transport.
- [Transport](#) (int domainId=0)
Default constructor.

6.66.1 Detailed Description

This class is the base of all classes that implement a transport using DDS. This transport could be used by both proxies and servers.

6.66.2 Constructor & Destructor Documentation

6.66.2.1 `eprosima::rpc::transport::dds::Transport::Transport (int domainId = 0)` [protected]

Default constructor.

Parameters

<i>domainId</i>	Optional parameter that specifies the domain identifier that will be used in DDS.
-----------------	---

6.66.3 Member Function Documentation

6.66.3.1 `virtual eprosima::rpc::transport::Endpoint* eprosima::rpc::transport::dds::Transport::createProcedureEndpoint (const char * name, const char * writertypename, const char * readertypename, bool eprosima_types, Create_data create_data, Copy_data copy_data, Destroy_data destroy_data, ProcessFunc processFunc, int dataSize)` [pure virtual]

This function creates a new procedure endpoint. This proxy procedure endpoint manages the DDS datawriter and the DDS datareader.

TODO Actualizar

Parameters

<i>name</i>	The name associated with this proxy procedure endpoint. It cannot be NULL.
<i>writertypename</i>	The type name of the topic that the procedure endpoint uses in the datawriter. It cannot be NULL.
<i>readertypename</i>	The type name of the topic that the procedure endpoint uses in the datareader. It cannot be NULL.
<i>initialize_data</i>	Pointer to the function to initialize DataReader received data
<i>copy_data</i>	Pointer to the function used to copy the data when it is received.
<i>finalize_data</i>	Pointer to the function to finalize DataReader received data
<i>ProcessFunc</i>	Pointer to the function invoked when a message is received from the server
<i>dataSize</i>	Size of the DataReader data structure

Returns

0 if the function ends successfully, -1 otherwise. TODO

Implemented in [eprosima::rpc::transport::dds::ProxyTransport](#), and [eprosima::rpc::transport::dds::ServerTransport](#).

6.66.3.2 DDSDomainParticipant* eprosima::rpc::transport::dds::Transport::getParticipant () const [inline]

Gets the domain participant.

Returns

DDS domain participant.

6.66.3.3 DDSPublisher* eprosima::rpc::transport::dds::Transport::getPublisher () const [inline]

Gets the publisher.

Returns

DDS publisher.

6.66.3.4 DDSSubscriber* eprosima::rpc::transport::dds::Transport::getSubscriber () const [inline]

Gets the subscriber.

Returns

DDS subscriber.

6.66.3.5 virtual int eprosima::rpc::transport::dds::Transport::setTransport (DDS_DomainParticipantQos & participantQos, DDSDomainParticipant * participant) [protected], [pure virtual]

This abstract function sets the QoS to use a specific transport.

Parameters

<i>participantQos</i>	Reference to the DDS domain participant QoS.
-----------------------	--

<i>participant</i>	The domain participant that will be set to use a specific transport.
--------------------	--

Implemented in [eprosima::rpc::transport::dds::ProxyTransport](#), [eprosima::rpc::transport::dds::ServerTransport](#), [eprosima::rpc::transport::dds::UDPProxyTransport](#), [eprosima::rpc::transport::dds::TCPServerTransport](#), and [eprosima::rpc::transport::dds::UDPServerTransport](#).

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

- `includetmp/rpcdds/transport/dds/Transport.h`

6.67 eprosima::rpc::transport::Transport Class Reference

This class is the base of all classes that implement a transport that could be used by the proxy or the server.

```
#include <Transport.h>
```

Inheritance diagram for `eprosima::rpc::transport::Transport`:

Public Member Functions

- [Transport](#) ()
Default constructor.
- virtual [~Transport](#) ()
Default destructor.
- virtual const char * [getType](#) () const =0
This function returns the type of the transport. This function has to be implemented by the child classes.
- virtual [TransportBehaviour](#) [getBehaviour](#) () const =0

6.67.1 Detailed Description

This class is the base of all classes that implement a transport that could be used by the proxy or the server.

6.67.2 Member Function Documentation

6.67.2.1 virtual [TransportBehaviour](#) `eprosima::rpc::transport::Transport::getBehaviour () const` [pure virtual]

2brief This function returns the behaviour of the transport.

Returns

The behaviour of the transport.

Implemented in [eprosima::rpc::transport::ServerTransport](#), and [eprosima::rpc::transport::ProxyTransport](#).

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

- `includetmp/rpcdds/transport/Transport.h`

6.68 eprosima::rpc::transport::dds::UDPProxyTransport Class Reference

This class implements a transport using DDS over UDPv4. This transport only can be used by a server's proxy.

```
#include <UDPProxyTransport.h>
```

Inheritance diagram for `eprosima::rpc::transport::dds::UDPProxyTransport`:

Collaboration diagram for `eprosima::rpc::transport::dds::UDPProxyTransport`:

Public Member Functions

- FASTRPC_DIIAPI [UDPProxyTransport](#) (const char *const &remoteServiceName, const char *const &instanceName, int domainId=0, long timeout=10000L)
Default constructor for server's proxies.
- FASTRPC_DIIAPI [UDPProxyTransport](#) (const char *const &to_connect, const char *const &remoteServiceName, const char *const &instanceName, int domainId=0, long timeout=10000L)
Constructor for server's proxies.
- virtual FASTRPC_DIIAPI [~UDPProxyTransport](#) ()
Default destructor.
- virtual FASTRPC_DIIAPI int [setTransport](#) (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant)
This function sets the QoS of DDS to use the UDPv4 transport.

Additional Inherited Members

6.68.1 Detailed Description

This class implements a transport using DDS over UDPv4. This transport only can be used by a server's proxy.

6.68.2 Constructor & Destructor Documentation

- 6.68.2.1 FASTRPC_DIIAPI eprosimarpc::transport::dds::UDPProxyTransport::UDPProxyTransport (const char *const &remoteServiceName, const char *const &instanceName, int domainId = 0, long timeout = 10000L)

Default constructor for server's proxies.

Parameters

<i>remoteServiceName</i>	Name of the service
<i>domainId</i>	Optional parameter that specifies the domain identifier to be used in DDS.
<i>timeout</i>	The time in milliseconds to wait for the reply.

- 6.68.2.2 FASTRPC_DIIAPI eprosimarpc::transport::dds::UDPProxyTransport::UDPProxyTransport (const char *const &to_connect, const char *const &remoteServiceName, const char *const &instanceName, int domainId = 0, long timeout = 10000L)

Constructor for server's proxies.

Parameters

<i>to_connect</i>	IP address where the server can be found by the proxy. For example: "192.168.1.3"
<i>remoteServiceName</i>	Name of the service
<i>domainId</i>	Optional parameter that specifies the domain identifier to be used in DDS.
<i>timeout</i>	The time in milliseconds to wait for the reply.

6.68.3 Member Function Documentation

- 6.68.3.1 virtual FASTRPC_DIIAPI int eprosimarpc::transport::dds::UDPProxyTransport::setTransport (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant) [virtual]

This function sets the QoS of DDS to use the UDPv4 transport.

Parameters

<i>participantQos</i>	Reference to the DDS domain participant QoS.
<i>participant</i>	The domain participant that will be set to use UDPv4 transport.

Implements [eprosima::rpc::transport::dds::ProxyTransport](#).

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

- `includetmp/rpcdds/transport/dds/UDPProxyTransport.h`

6.69 eprosima::rpc::transport::dds::UDPServerTransport Class Reference

This class implements transport using DDS over UDPv4. This transport can only be used by a server.

```
#include <UDPServerTransport.h>
```

Inheritance diagram for `eprosima::rpc::transport::dds::UDPServerTransport`:

Collaboration diagram for `eprosima::rpc::transport::dds::UDPServerTransport`:

Public Member Functions

- `FASTRPC_DIIAPI UDPServerTransport (const char *const &serviceName, const char *const &instanceName, int domainId=0)`
Default constructor for servers.
- `virtual FASTRPC_DIIAPI ~UDPServerTransport ()`
Default destructor.
- `virtual FASTRPC_DIIAPI int setTransport (DDS_DomainParticipantQos &participantQos, DDSDomainParticipant *participant)`
This function sets the DDS' QoS to use the UDPv4 transport.

Additional Inherited Members

6.69.1 Detailed Description

This class implements transport using DDS over UDPv4. This transport can only be used by a server.

6.69.2 Constructor & Destructor Documentation

- 6.69.2.1 `FASTRPC_DIIAPI eprosima::rpc::transport::dds::UDPServerTransport::UDPServerTransport (const char *const &serviceName, const char *const &instanceName, int domainId = 0)`

Default constructor for servers.

Parameters

<i>remoteServiceName</i>	Name of the service
<i>domainId</i>	Optional parameter that specifies the domain identifier that will be used in DDS.

6.69.3 Member Function Documentation

6.69.3.1 virtual FASTRPC_DllAPI int eprosima::rpc::transport::dds::UDPServerTransport::setTransport (DDS_DomainParticipantQos & *participantQos*, DDSDomainParticipant * *participant*) [virtual]

This function sets the DDS' QoS to use the UDPv4 transport.

Parameters

<i>participantQos</i>	Reference to the DDS domain participant QoS.
<i>participant</i>	The domain participant that will be set to use UDPv4 transport.

Implements [eprosima::rpc::transport::dds::ServerTransport](#).

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

- `includetmp/rpcdds/transport/dds/UDPServerTransport.h`

6.70 eprosima::rpc::protocol::dds::UnknownExceptionPlugin Class Reference

Static Public Member Functions

- static DDS_TypeCode * [get_typecode](#) ()
This function returns the TypeCode.

6.70.1 Member Function Documentation

6.70.1.1 static DDS_TypeCode* eprosima::rpc::protocol::dds::UnknownExceptionPlugin::get_typecode () [static]

This function returns the TypeCode.

Returns

The TypeCode.

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

- `includetmp/rpcdds/protocols/dds/MessageHeaderPlugin.h`

6.71 eprosima::rpc::protocol::dds::UnknownOperationPlugin Class Reference

Static Public Member Functions

- static DDS_TypeCode * [get_typecode](#) ()
This function returns the TypeCode.

6.71.1 Member Function Documentation

6.71.1.1 static DDS_TypeCode* eprosima::rpc::protocol::dds::UnknownOperationPlugin::get_typecode () [static]

This function returns the TypeCode.

Returns

The TypeCode.

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

- `includetmp/rpcdds/protocols/dds/MessageHeaderPlugin.h`

6.72 eprosimarpc::exception::UserException Class Reference

This abstract class is used to create user exceptions.

```
#include <UserException.h>
```

Inheritance diagram for eprosimarpc::exception::UserException:

Collaboration diagram for eprosimarpc::exception::UserException:

Public Member Functions

- virtual [~UserException](#) () throw ()
Default destructor.
- virtual void [raise](#) () const =0
This function throws the object as exception.

Protected Member Functions

- [UserException](#) ()
Default constructor.
- [UserException](#) (const [UserException](#) &ex)
Default copy constructor.
- [UserException](#) ([UserException](#) &&ex)
Default move constructor.
- [UserException](#) & [operator=](#) (const [UserException](#) &ex)
Assignment operation.
- [UserException](#) & [operator=](#) ([UserException](#) &&ex)
Assignment operation.

6.72.1 Detailed Description

This abstract class is used to create user exceptions.

6.72.2 Constructor & Destructor Documentation

6.72.2.1 eprosimarpc::exception::UserException::UserException (const [UserException](#) & ex) [protected]

Default copy constructor.

Parameters

<i>ex</i>	UserException that will be copied.
-----------	--

6.72.2.2 eprosimarpc::exception::UserException::UserException ([UserException](#) && ex) [protected]

Default move constructor.

Parameters

<i>ex</i>	UserException that will be moved.
-----------	---

6.72.3 Member Function Documentation

6.72.3.1 `UserException& eprosima::rpc::exception::UserException::operator= (const UserException & ex)` [protected]

Assignment operation.

Parameters

<i>ex</i>	UserException that will be copied.
-----------	--

6.72.3.2 `UserException& eprosima::rpc::exception::UserException::operator= (UserException && ex)` [protected]

Assignment operation.

Parameters

<i>ex</i>	UserException that will be moved.
-----------	---

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

- `includetmp/rpcdds/exceptions/UserException.h`

Index

- `_d`
 - `FooDDS::Foo_Call`, 32
 - `FooDDS::Foo_FooProcedure_Result`, 41
 - `FooDDS::Foo_Return`, 59
 - `_setTransport`
 - `eprosima::rpc::protocol::Protocol`, 75
- `activateInterface`
 - `eprosima::rpc::protocol::dds::FooDDSProtocol`, 62
 - `eprosima::rpc::protocol::FooDDSProtocol`, 64
- `addAsyncTask`
 - `eprosima::rpc::transport::dds::ProxyTransport`, 80
- `addTask`
 - `eprosima::rpc::transport::dds::AsyncThread`, 22
- `BadParamException`
 - `eprosima::rpc::exception::BadParamException`, 23
- `Client Module`, 12
- `ClientInternalException`
 - `eprosima::rpc::exception::ClientInternalException`, 25
- `connect`
 - `eprosima::rpc::transport::ProxyTransport`, 82
- `createProcedureEndpoint`
 - `eprosima::rpc::transport::dds::ProxyTransport`, 80
 - `eprosima::rpc::transport::dds::ServerTransport`, 108
 - `eprosima::rpc::transport::dds::Transport`, 118
- `deleteAssociatedAsyncTasks`
 - `eprosima::rpc::transport::dds::AsyncThread`, 22
 - `eprosima::rpc::transport::dds::ProxyTransport`, 81
- `deserialize`
 - `eprosima::rpc::protocol::dds::GUID_t`, 70
 - `eprosima::rpc::protocol::dds::ReplyHeader`, 84
 - `eprosima::rpc::protocol::dds::RequestHeader`, 88
 - `eprosima::rpc::protocol::dds::SampleIdentity_t`, 92
 - `FooDDS::Foo_Call`, 32
 - `FooDDS::Foo_FooProcedure_In`, 36
 - `FooDDS::Foo_FooProcedure_Out`, 38
 - `FooDDS::Foo_FooProcedure_Result`, 41
 - `FooDDS::Foo_Reply`, 47
 - `FooDDS::Foo_Request`, 53
 - `FooDDS::Foo_Return`, 59
- `eProsima RPCDDS API Reference`, 11
- `eprosima::rpc::exception::BadParamException`, 22
 - `BadParamException`, 23
 - `operator=`, 24
- `eprosima::rpc::exception::ClientInternalException`, 24
 - `ClientInternalException`, 25
 - `operator=`, 25
- `eprosima::rpc::exception::Exception`, 28
 - `Exception`, 29, 30
 - `operator=`, 30
- `eprosima::rpc::exception::IncompatibleException`, 71
 - `IncompatibleException`, 71, 72
 - `operator=`, 72
- `eprosima::rpc::exception::InitializeException`, 72
 - `InitializeException`, 73
 - `operator=`, 74
- `eprosima::rpc::exception::ServerInternalException`, 96
 - `operator=`, 97
 - `ServerInternalException`, 97
- `eprosima::rpc::exception::ServerNotFoundException`, 98
 - `operator=`, 99
 - `ServerNotFoundException`, 98, 99
- `eprosima::rpc::exception::ServerTimeoutException`, 103
 - `operator=`, 104
 - `ServerTimeoutException`, 104
- `eprosima::rpc::exception::SystemException`, 111
 - `minor`, 112
 - `operator=`, 112, 113
 - `SystemException`, 111, 112
 - `what`, 113
- `eprosima::rpc::exception::UserException`, 125
 - `operator=`, 126
 - `UserException`, 125
- `eprosima::rpc::protocol::FooDDSProtocol`, 63
 - `activateInterface`, 64
 - `linkFooDDS_FooImpl`, 64
 - `setTransport`, 64
- `eprosima::rpc::protocol::Protocol`, 74
 - `_setTransport`, 75
 - `getTransport`, 75
 - `setTransport`, 75
- `eprosima::rpc::protocol::dds::FooDDSProtocol`, 61
 - `activateInterface`, 62
 - `FooDDS_Foo_serve`, 62
 - `setTransport`, 63
- `eprosima::rpc::protocol::dds::GUID_t`, 69
 - `deserialize`, 70
 - `getMaxCdrSerializedSize`, 70
 - `serialize`, 70
- `eprosima::rpc::protocol::dds::GUID_tPlugin`, 70
 - `get_typecode`, 70
- `eprosima::rpc::protocol::dds::ReplyHeader`, 83
 - `deserialize`, 84

- getMaxCdrSerializedSize, 84
 - operator=, 85
 - ReplyHeader, 84
 - request_id, 85
 - serialize, 86
- eprosima::rpc::protocol::dds::ReplyHeaderPlugin, 86
 - get_typecode, 86
- eprosima::rpc::protocol::dds::RequestHeader, 86
 - deserialize, 88
 - getMaxCdrSerializedSize, 88
 - instance_name, 88
 - operator=, 88, 89
 - remote_service_name, 89
 - request_id, 89
 - RequestHeader, 87, 88
 - serialize, 90
- eprosima::rpc::protocol::dds::RequestHeaderPlugin, 90
 - get_typecode, 90
- eprosima::rpc::protocol::dds::SampleIdentity_t, 90
 - deserialize, 92
 - getMaxCdrSerializedSize, 92
 - guid, 92
 - operator=, 92, 93
 - SampleIdentity_t, 91, 92
 - sequence_number, 93
 - serialize, 93
- eprosima::rpc::protocol::dds::SampleIdentity_tPlugin, 93
 - get_typecode, 94
- eprosima::rpc::protocol::dds::SystemExceptionCode-Plugin, 113
 - get_typecode, 113
- eprosima::rpc::protocol::dds::UnknownExceptionPlugin, 124
 - get_typecode, 124
- eprosima::rpc::protocol::dds::UnknownOperationPlugin, 124
 - get_typecode, 124
- eprosima::rpc::proxy::Proxy, 75
 - getProtocol, 76
 - getTransport, 76
 - Proxy, 76
- eprosima::rpc::server::Server, 94
 - process, 96
 - Server, 95
- eprosima::rpc::strategy::ServerStrategy, 102
 - getImpl, 102
- eprosima::rpc::strategy::ServerStrategyImpl, 102
 - schedule, 103
- eprosima::rpc::strategy::SingleThreadStrategy, 110
 - getImpl, 110
- eprosima::rpc::strategy::ThreadPerRequestStrategy, 116
 - getImpl, 116
- eprosima::rpc::strategy::ThreadPoolStrategy, 116
 - getImpl, 117
 - ThreadPoolStrategy, 117
- eprosima::rpc::transport::AsyncTask, 21
- eprosima::rpc::transport::Endpoint, 27
- eprosima::rpc::transport::ProxyTransport, 82
 - connect, 82
 - getBehaviour, 82
 - receive, 82
 - send, 83
- eprosima::rpc::transport::ServerTransport, 105
 - getBehaviour, 106
 - getCallback, 106
 - getLinkedProtocol, 106
 - getStrategy, 106
 - linkProtocol, 106
 - receive, 106
 - sendReply, 107
 - setCallback, 107
 - setStrategy, 107
- eprosima::rpc::transport::Transport, 120
 - getBehaviour, 120
- eprosima::rpc::transport::dds::AsyncThread, 21
 - addTask, 22
 - deleteAssociatedAsyncTasks, 22
 - init, 22
- eprosima::rpc::transport::dds::DDSAsyncTask, 26
 - execute, 26
 - getProcedureEndpoint, 27
 - getReplyInstance, 27
 - on_exception, 27
 - setProcedureEndpoint, 27
- eprosima::rpc::transport::dds::ProxyProcedureEndpoint, 76
 - freeQuery, 77
 - initialize, 78
 - ProxyProcedureEndpoint, 77
 - send, 78
 - send_async, 78
- eprosima::rpc::transport::dds::ProxyTransport, 79
 - addAsyncTask, 80
 - createProcedureEndpoint, 80
 - deleteAssociatedAsyncTasks, 81
 - getRemoteServiceName, 81
 - getTimeout, 81
 - ProxyTransport, 80
 - setTransport, 81
- eprosima::rpc::transport::dds::ServerProcedureEndpoint, 99
 - getProcessFunc, 101
 - initialize, 101
 - sendReply, 101
 - ServerProcedureEndpoint, 100
 - start, 101
 - stop, 101
- eprosima::rpc::transport::dds::ServerTransport, 107
 - createProcedureEndpoint, 108
 - process, 109
 - sendReply, 109
 - ServerTransport, 108
 - setTransport, 109
- eprosima::rpc::transport::dds::TCPProxyTransport, 113

- setTransport, 114
 - TCPProxyTransport, 114
- eprosima::rpc::transport::dds::TCPServerTransport, 114
 - setTransport, 115
 - TCPServerTransport, 115
- eprosima::rpc::transport::dds::Transport, 117
 - createProcedureEndpoint, 118
 - getParticipant, 119
 - getPublisher, 119
 - getSubscriber, 119
 - setTransport, 119
 - Transport, 118
- eprosima::rpc::transport::dds::UDPProxyTransport, 120
 - setTransport, 121
 - UDPProxyTransport, 121
- eprosima::rpc::transport::dds::UDPServerTransport, 122
 - setTransport, 122
 - UDPServerTransport, 122
- Exception
 - eprosima::rpc::exception::Exception, 29, 30
- Exceptions, 14
- execute
 - eprosima::rpc::transport::dds::DDSAsyncTask, 26
- Foo_Call
 - FooDDS::Foo_Call, 32
- Foo_FooProcedure_In
 - FooDDS::Foo_FooProcedure_In, 35
- Foo_FooProcedure_Out
 - FooDDS::Foo_FooProcedure_Out, 38
- Foo_FooProcedure_Result
 - FooDDS::Foo_FooProcedure_Result, 40, 41
- Foo_FooProcedureTask
 - FooDDS::Foo_FooProcedureTask, 45
- Foo_Reply
 - FooDDS::Foo_Reply, 47
- Foo_Request
 - FooDDS::Foo_Request, 52, 53
- Foo_Return
 - FooDDS::Foo_Return, 58, 59
- FooDDS::Foo, 30
- FooDDS::Foo_Call, 31
 - _d, 32
 - deserialize, 32
 - Foo_Call, 32
 - FooProcedure, 33
 - getMaxCdrSerializedSize, 33
 - getSerializedSize, 33
 - operator=, 34
 - serialize, 34
- FooDDS::Foo_CallPlugin, 34
- FooDDS::Foo_FooProcedure_In, 35
 - deserialize, 36
 - Foo_FooProcedure_In, 35
 - getMaxCdrSerializedSize, 36
 - getSerializedSize, 36
 - operator=, 36
 - serialize, 37
- FooDDS::Foo_FooProcedure_Out, 37
 - deserialize, 38
 - Foo_FooProcedure_Out, 38
 - getMaxCdrSerializedSize, 38
 - getSerializedSize, 38
 - operator=, 39
 - serialize, 39
- FooDDS::Foo_FooProcedure_Result, 39
 - _d, 41
 - deserialize, 41
 - Foo_FooProcedure_Result, 40, 41
 - getMaxCdrSerializedSize, 41
 - getSerializedSize, 42
 - operator=, 42
 - out_, 42, 43
 - serialize, 43
- FooDDS::Foo_FooProcedureCallbackHandler, 43
 - FooProcedure, 44
 - on_exception, 44
- FooDDS::Foo_FooProcedureTask, 44
 - Foo_FooProcedureTask, 45
 - getObject, 45
 - getReplyInstance, 45
 - on_exception, 45
- FooDDS::Foo_Reply, 46
 - deserialize, 47
 - Foo_Reply, 47
 - getMaxCdrSerializedSize, 47
 - getSerializedSize, 47
 - header, 47, 48
 - operator=, 48
 - reply, 48, 49
 - serialize, 49
- FooDDS::Foo_ReplyDataReader, 49
- FooDDS::Foo_ReplyDataWriter, 50
- FooDDS::Foo_ReplyPlugin, 50
 - register_type, 51
- FooDDS::Foo_Request, 51
 - deserialize, 53
 - Foo_Request, 52, 53
 - getMaxCdrSerializedSize, 53
 - getSerializedSize, 53
 - header, 53, 54
 - operator=, 54
 - request, 54, 55
 - serialize, 55
- FooDDS::Foo_RequestDataReader, 55
- FooDDS::Foo_RequestDataWriter, 56
- FooDDS::Foo_RequestPlugin, 56
 - register_type, 57
- FooDDS::Foo_Return, 57
 - _d, 59
 - deserialize, 59
 - Foo_Return, 58, 59
 - FooProcedure, 59, 60
 - getMaxCdrSerializedSize, 60
 - getSerializedSize, 60
 - operator=, 60, 61

- serialize, 61
- FooDDS::Foo_ReturnPlugin, 61
- FooDDS::FooPlugin, 64
- FooDDS::FooPlugin::FooProcedure_InPlugin, 65
- FooDDS::FooPlugin::FooProcedure_OutPlugin, 65
- FooDDS::FooPlugin::FooProcedure_ResultPlugin, 66
- FooDDS::FooProxy, 66
 - FooProxy, 67
- FooDDS::FooServer, 68
 - FooServer, 68
- FooDDS::FooServerImpl, 69
- FooDDS_Foo_serve
 - eprosima::rpc::protocol::dds::FooDDSProtocol, 62
- FooProcedure
 - FooDDS::Foo_Call, 33
 - FooDDS::Foo_FooProcedureCallbackHandler, 44
 - FooDDS::Foo_Return, 59, 60
- FooProxy
 - FooDDS::FooProxy, 67
- FooServer
 - FooDDS::FooServer, 68
- freeQuery
 - eprosima::rpc::transport::dds::ProxyProcedure-Endpoint, 77
- Generated API example for eProsimas RPCDDS, 19
- get_typecode
 - eprosima::rpc::protocol::dds::GUID_tPlugin, 70
 - eprosima::rpc::protocol::dds::ReplyHeaderPlugin, 86
 - eprosima::rpc::protocol::dds::RequestHeader-Plugin, 90
 - eprosima::rpc::protocol::dds::SampleIdentity_t-Plugin, 94
 - eprosima::rpc::protocol::dds::SystemException-CodePlugin, 113
 - eprosima::rpc::protocol::dds::UnknownException-Plugin, 124
 - eprosima::rpc::protocol::dds::UnknownOperation-Plugin, 124
- getBehaviour
 - eprosima::rpc::transport::ProxyTransport, 82
 - eprosima::rpc::transport::ServerTransport, 106
 - eprosima::rpc::transport::Transport, 120
- getCallback
 - eprosima::rpc::transport::ServerTransport, 106
- getImpl
 - eprosima::rpc::strategy::ServerStrategy, 102
 - eprosima::rpc::strategy::SingleThreadStrategy, 110
 - eprosima::rpc::strategy::ThreadPerRequest-Strategy, 116
 - eprosima::rpc::strategy::ThreadPoolStrategy, 117
- getLinkedProtocol
 - eprosima::rpc::transport::ServerTransport, 106
- getMaxCdrSerializedSize
 - eprosima::rpc::protocol::dds::GUID_t, 70
 - eprosima::rpc::protocol::dds::ReplyHeader, 84
 - eprosima::rpc::protocol::dds::RequestHeader, 88
 - eprosima::rpc::protocol::dds::SampleIdentity_t, 92
 - FooDDS::Foo_Call, 33
 - FooDDS::Foo_FooProcedure_In, 36
 - FooDDS::Foo_FooProcedure_Out, 38
 - FooDDS::Foo_FooProcedure_Result, 41
 - FooDDS::Foo_Reply, 47
 - FooDDS::Foo_Request, 53
 - FooDDS::Foo_Return, 60
- getObject
 - FooDDS::Foo_FooProcedureTask, 45
- getParticipant
 - eprosima::rpc::transport::dds::Transport, 119
- getProcedureEndpoint
 - eprosima::rpc::transport::dds::DDSAsyncTask, 27
- getProcessFunc
 - eprosima::rpc::transport::dds::ServerProcedure-Endpoint, 101
- getProtocol
 - eprosima::rpc::proxy::Proxy, 76
- getPublisher
 - eprosima::rpc::transport::dds::Transport, 119
- getRemoteServiceName
 - eprosima::rpc::transport::dds::ProxyTransport, 81
- getReplyInstance
 - eprosima::rpc::transport::dds::DDSAsyncTask, 27
 - FooDDS::Foo_FooProcedureTask, 45
- getSerializedSize
 - FooDDS::Foo_Call, 33
 - FooDDS::Foo_FooProcedure_In, 36
 - FooDDS::Foo_FooProcedure_Out, 38
 - FooDDS::Foo_FooProcedure_Result, 42
 - FooDDS::Foo_Reply, 47
 - FooDDS::Foo_Request, 53
 - FooDDS::Foo_Return, 60
- getStrategy
 - eprosima::rpc::transport::ServerTransport, 106
- getSubscriber
 - eprosima::rpc::transport::dds::Transport, 119
- getTimeout
 - eprosima::rpc::transport::dds::ProxyTransport, 81
- getTransport
 - eprosima::rpc::protocol::Protocol, 75
 - eprosima::rpc::proxy::Proxy, 76
- guid
 - eprosima::rpc::protocol::dds::SampleIdentity_t, 92
- header
 - FooDDS::Foo_Reply, 47, 48
 - FooDDS::Foo_Request, 53, 54
- IncompatibleException
 - eprosima::rpc::exception::IncompatibleException, 71, 72
- init
 - eprosima::rpc::transport::dds::AsyncThread, 22
- initialize
 - eprosima::rpc::transport::dds::ProxyProcedure-Endpoint, 78

- eprosima::rpc::transport::dds::ServerProcedure-Endpoint, [101](#)
- InitializeException
 - eprosima::rpc::exception::InitializeException, [73](#)
- instance_name
 - eprosima::rpc::protocol::dds::RequestHeader, [88](#)
- linkFooDDS_FooImpl
 - eprosima::rpc::protocol::FooDDSProtocol, [64](#)
- linkProtocol
 - eprosima::rpc::transport::ServerTransport, [106](#)
- minor
 - eprosima::rpc::exception::SystemException, [112](#)
- on_exception
 - eprosima::rpc::transport::dds::DDSAsyncTask, [27](#)
 - FooDDS::Foo_FooProcedureCallbackHandler, [44](#)
 - FooDDS::Foo_FooProcedureTask, [45](#)
- operator=
 - eprosima::rpc::exception::BadParamException, [24](#)
 - eprosima::rpc::exception::ClientInternalException, [25](#)
 - eprosima::rpc::exception::Exception, [30](#)
 - eprosima::rpc::exception::IncompatibleException, [72](#)
 - eprosima::rpc::exception::InitializeException, [74](#)
 - eprosima::rpc::exception::ServerInternalException, [97](#)
 - eprosima::rpc::exception::ServerNotFoundException, [99](#)
 - eprosima::rpc::exception::ServerTimeoutException, [104](#)
 - eprosima::rpc::exception::SystemException, [112](#), [113](#)
 - eprosima::rpc::exception::UserException, [126](#)
 - eprosima::rpc::protocol::dds::ReplyHeader, [85](#)
 - eprosima::rpc::protocol::dds::RequestHeader, [88](#), [89](#)
 - eprosima::rpc::protocol::dds::SampleIdentity_t, [92](#), [93](#)
 - FooDDS::Foo_Call, [34](#)
 - FooDDS::Foo_FooProcedure_In, [36](#)
 - FooDDS::Foo_FooProcedure_Out, [39](#)
 - FooDDS::Foo_FooProcedure_Result, [42](#)
 - FooDDS::Foo_Reply, [48](#)
 - FooDDS::Foo_Request, [54](#)
 - FooDDS::Foo_Return, [60](#), [61](#)
- out_
 - FooDDS::Foo_FooProcedure_Result, [42](#), [43](#)
- process
 - eprosima::rpc::server::Server, [96](#)
 - eprosima::rpc::transport::dds::ServerTransport, [109](#)
- Protocols, [18](#)
- Proxy
 - eprosima::rpc::proxy::Proxy, [76](#)
- ProxyProcedureEndpoint
 - eprosima::rpc::transport::dds::ProxyProcedure-Endpoint, [77](#)
- ProxyTransport
 - eprosima::rpc::transport::dds::ProxyTransport, [80](#)
- receive
 - eprosima::rpc::transport::ProxyTransport, [82](#)
 - eprosima::rpc::transport::ServerTransport, [106](#)
- register_type
 - FooDDS::Foo_ReplyPlugin, [51](#)
 - FooDDS::Foo_RequestPlugin, [57](#)
- remote_service_name
 - eprosima::rpc::protocol::dds::RequestHeader, [89](#)
- reply
 - FooDDS::Foo_Reply, [48](#), [49](#)
- ReplyHeader
 - eprosima::rpc::protocol::dds::ReplyHeader, [84](#)
- request
 - FooDDS::Foo_Request, [54](#), [55](#)
- request_id
 - eprosima::rpc::protocol::dds::ReplyHeader, [85](#)
 - eprosima::rpc::protocol::dds::RequestHeader, [89](#)
- RequestHeader
 - eprosima::rpc::protocol::dds::RequestHeader, [87](#), [88](#)
- SampleIdentity_t
 - eprosima::rpc::protocol::dds::SampleIdentity_t, [91](#), [92](#)
- schedule
 - eprosima::rpc::strategy::ServerStrategyImpl, [103](#)
- send
 - eprosima::rpc::transport::dds::ProxyProcedure-Endpoint, [78](#)
 - eprosima::rpc::transport::ProxyTransport, [83](#)
- send_async
 - eprosima::rpc::transport::dds::ProxyProcedure-Endpoint, [78](#)
- sendReply
 - eprosima::rpc::transport::dds::ServerProcedure-Endpoint, [101](#)
 - eprosima::rpc::transport::dds::ServerTransport, [109](#)
 - eprosima::rpc::transport::ServerTransport, [107](#)
- sequence_number
 - eprosima::rpc::protocol::dds::SampleIdentity_t, [93](#)
- serialize
 - eprosima::rpc::protocol::dds::GUID_t, [70](#)
 - eprosima::rpc::protocol::dds::ReplyHeader, [86](#)
 - eprosima::rpc::protocol::dds::RequestHeader, [90](#)
 - eprosima::rpc::protocol::dds::SampleIdentity_t, [93](#)
 - FooDDS::Foo_Call, [34](#)
 - FooDDS::Foo_FooProcedure_In, [37](#)
 - FooDDS::Foo_FooProcedure_Out, [39](#)
 - FooDDS::Foo_FooProcedure_Result, [43](#)
 - FooDDS::Foo_Reply, [49](#)
 - FooDDS::Foo_Request, [55](#)
 - FooDDS::Foo_Return, [61](#)
- Server

- `eprosima::rpc::server::Server`, 95
- Server Module, 13
- ServerInternalException
 - `eprosima::rpc::exception::ServerInternalException`, 97
- ServerNotFoundException
 - `eprosima::rpc::exception::ServerNotFoundException`, 98, 99
- ServerProcedureEndpoint
 - `eprosima::rpc::transport::dds::ServerProcedure-Endpoint`, 100
- ServerTimeoutException
 - `eprosima::rpc::exception::ServerTimeoutException`, 104
- ServerTransport
 - `eprosima::rpc::transport::dds::ServerTransport`, 108
- setCallback
 - `eprosima::rpc::transport::ServerTransport`, 107
- setProcedureEndpoint
 - `eprosima::rpc::transport::dds::DDSAsyncTask`, 27
- setStrategy
 - `eprosima::rpc::transport::ServerTransport`, 107
- setTransport
 - `eprosima::rpc::protocol::dds::FooDDSProtocol`, 63
 - `eprosima::rpc::protocol::FooDDSProtocol`, 64
 - `eprosima::rpc::protocol::Protocol`, 75
 - `eprosima::rpc::transport::dds::ProxyTransport`, 81
 - `eprosima::rpc::transport::dds::ServerTransport`, 109
 - `eprosima::rpc::transport::dds::TCPProxyTransport`, 114
 - `eprosima::rpc::transport::dds::TCPServerTransport`, 115
 - `eprosima::rpc::transport::dds::Transport`, 119
 - `eprosima::rpc::transport::dds::UDPProxyTransport`, 121
 - `eprosima::rpc::transport::dds::UDPServerTransport`, 122
- start
 - `eprosima::rpc::transport::dds::ServerProcedure-Endpoint`, 101
- stop
 - `eprosima::rpc::transport::dds::ServerProcedure-Endpoint`, 101
- Strategies, 15
- SystemException
 - `eprosima::rpc::exception::SystemException`, 111, 112
- TCPProxyTransport
 - `eprosima::rpc::transport::dds::TCPProxyTransport`, 114
- TCPServerTransport
 - `eprosima::rpc::transport::dds::TCPServerTransport`, 115
- ThreadPoolStrategy
 - `eprosima::rpc::strategy::ThreadPoolStrategy`, 117
- Transport
 - `eprosima::rpc::transport::dds::Transport`, 118
- Transports, 16
- UDPProxyTransport
 - `eprosima::rpc::transport::dds::UDPProxyTransport`, 121
- UDPServerTransport
 - `eprosima::rpc::transport::dds::UDPServerTransport`, 122
- UserException
 - `eprosima::rpc::exception::UserException`, 125
- what
 - `eprosima::rpc::exception::SystemException`, 113