

Schema documentation for DR-GW.xsd

november 22, 2023

Table of Contents

Namespace: "DR-GW"	9
Schema(s)	9
Main schema DR-GW.xsd	9
Included schema DR-GW-Call.xsd	9
Included schema DR-GW-Call.select.xsd	9
Included schema DR-GW-Call.types.xsd	9
Included schema DR-GW.types.xsd	9
Included schema DR-GW-Call.call.xsd	10
Included schema DR-GW-Call.ptt.xsd	10
Included schema DR-GW-Call.keyExchange.xsd	10
Included schema DR-GW-Call.unitInEmergency.xsd	10
Included schema DR-GW-Session.xsd	10
Included schema DR-GW-Session.check.xsd	10
Included schema DR-GW-Session.login.xsd	10
Included schema DR-GW-Session.types.xsd	10
Included schema DR-GW-Session.supervise.xsd	10
Included schema DR-GW-Session.logout.xsd	10
Included schema DR-GW-Sds.xsd	11
Included schema DR-GW-Sds.send.xsd	11
Included schema DR-GW-Sds.types.xsd	11
Included schema DR-GW-Sds.receive.xsd	11
Included schema DR-GW-Status.xsd	11
Included schema DR-GW-Status.send.xsd	11
Included schema DR-GW-Status.types.xsd	11
Included schema DR-GW-Status.receive.xsd	11
Included schema DR-GW-OrganisationBlock.xsd	11
Included schema DR-GW-OrganisationBlock.get.xsd	11
Included schema DR-GW-OrganisationBlock.types.xsd	12
Included schema DR-GW-OrganisationBlock.getList.xsd	12
Included schema DR-GW-OrganisationBlock.event.xsd	12
Included schema DR-GW-Group.xsd	12
Included schema DR-GW-Group.get.xsd	12
Included schema DR-GW-Group.types.xsd	12
Included schema DR-GW-Group.getList.xsd	12
Included schema DR-GW-Group.getRadioMembers.xsd	12
Included schema DR-GW-Group.getAppMembers.xsd	12
Included schema DR-GW-Group.getCombinations.xsd	13
Included schema DR-GW-Group.track.xsd	13
Included schema DR-GW-Group.addRadioMember.xsd	13
Included schema DR-GW-Group.removeRadioMember.xsd	13
Included schema DR-GW-Group.addCombination.xsd	13
Included schema DR-GW-Group.removeCombination.xsd	13
Included schema DR-GW-Group.subscribeData.xsd	13
Included schema DR-GW-Group.event.xsd	13
Included schema DR-GW-Radio.xsd	13
Included schema DR-GW-Radio.types.xsd	13
Included schema DR-GW-Radio.get.xsd	14
Included schema DR-GW-Radio.getList.xsd	14
Included schema DR-GW-Radio.getGroups.xsd	14
Included schema DR-GW-Radio.track.xsd	14
Included schema DR-GW-Radio.changeOpta.xsd	14
Included schema DR-GW-Radio.enableDisable.xsd	14
Included schema DR-GW-Radio.event.xsd	14
Included schema DR-GW-Application.xsd	14
Included schema DR-GW-Application.get.xsd	14
Included schema DR-GW-Application.types.xsd	14
Included schema DR-GW-Application.getList.xsd	15
Included schema DR-GW-System.xsd	15
Included schema DR-GW-System.tetraStates.xsd	15
Included schema DR-GW-System.types.xsd	15
Included schema DR-GW-System.log.xsd	15
Included schema DR-GW-System.event.xsd	15
Included schema DR-GW-Management.xsd	15

Included schema DR-GW-Management.get.xsd	15
Included schema DR-GW-Management.types.xsd	15
Element(s)	16
Element drgw	16
Element drgw / call	17
Element interfaceCall / select	18
Element typeRequest / requestId	18
Element typeCallSelect / target	18
Element typeSelection / level	19
Element typeSelection / target	19
Element typeAddress / subscriber	20
Element typeSubscriberAddress / ssi	20
Element typeSubscriberAddress / tsi	20
Element typeTSI / mnc	21
Element typeTSI / mcc	21
Element typeTSI / ssi	21
Element typeAddress / alias	21
Element typeAddress / msisdn	22
Element typeAddress / fssn	22
Element typeAddress / external	22
Element typeExternal / gatewayNumber	23
Element typeExternal / number	23
Element typeAddress / opta	23
Element typeAddress / cell	23
Element interfaceCall / request	24
Element typeEvent / requestId	24
Element typeEvent / result	25
Element typeResult / responseCode	25
Element typeResult / sourceSystem	25
Element typeResult / result	26
Element typeCallEvent / tetraCallId	26
Element typeCallEvent / action	26
Element typeCallEvent / attributes	27
Element typeCallAttributes / hook	27
Element typeCallAttributes / mode	28
Element typeCallAttributes / commtype	28
Element typeCallAttributes / priority	28
Element typeCallAttributes / encryption	28
Element typeCallAttributes / ambienceListen	29
Element typeCallAttributes / req2speak	29
Element typeCallAttributes / demandPriority	29
Element typeCallEvent / callingParty	29
Element typeCallEvent / calledParty	30
Element typeCallEvent / disconnectCause	31
Element typeDisconnectCause / protocol	32
Element typeDisconnectCause / code	32
Element typeDisconnectCause / text	32
Element interfaceCall / pttRequest	33
Element typeCallPTTRequest / action	33
Element typeCallPTTRequest / attributes	34
Element typeCallPTTRequest / talkingParty	34
Element typeCallPTTRequest / workstationId	35
Element interfaceCall / keyExchange	35
Element typeCallKeyExchange / action	36
Element interfaceCall / response	36
Element typeResponse / requestId	37
Element typeResponse / result	37
Element interfaceCall / selectEvent	37
Element typeCallSelectEvent / target	38
Element interfaceCall / event	38
Element interfaceCall / pttEvent	39
Element typeCallPTTEvent / tetraCallId	40
Element typeCallPTTEvent / granted	40
Element typeTxGranted / txGrant	41
Element typeTxGranted / talkingParty	42
Element typeTxGranted / encryption	42
Element typeTxGranted / txPriority	42
Element typeTxGranted / txInterrupt	43
Element typeTxGranted / txRepeat	43
Element typeTxGranted / workstationId	44
Element typeCallPTTEvent / ceased	44
Element typeCallPTTEvent / wait	44

Element interfaceCall / unitInEmergencyEvent	45
Element typeCallUnitInEmergencyEvent / group	46
Element typeCallUnitInEmergencyEvent / tetraCallId	46
Element typeCallUnitInEmergencyEvent / unitInEmg	46
Element typeCallUnitInEmergencyEvent / unitInEmgType	47
Element typeCallUnitInEmergencyEvent / emgInfo	47
Element typeCallUnitInEmergencyEvent / tstamp	48
Element interfaceCall / keyExchangeEvent	48
Element typeCallKeyExchangeEvent / state	49
Element typeCallKeyExchangeEvent / code	49
Element typeCallKeyExchangeEvent / priority	49
Element typeCallKeyExchangeEvent / interaction	50
Element typeCallKeyExchangeEvent / text	50
Element typeCallKeyExchangeEvent / tone	50
Element drgw / session	51
Element interfaceSession / login	51
Element typeSessionLogin / clientid	52
Element typeSessionLogin / supervise	52
Element typeSessionLogin / version	52
Element interfaceSession / logout	53
Element interfaceSession / supervise	53
Element interfaceSession / check	53
Element typeSessionCheck / clientid	54
Element interfaceSession / response	54
Element interfaceSession / loginEvent	55
Element typeSessionLoginEvent / issi	55
Element interfaceSession / superviseEvent	55
Element drgw / sds	56
Element interfaceSds / send	56
Element typeSdsSend / sds	57
Element typeSds / protocol	58
Element typeSds / sdsType	59
Element typeSds / msgRef	59
Element typeSds / report	59
Element typeSds / sdsdata	60
Element typeSdsData / data	60
Element typeSdsData / hexdata	60
Element typeSdsData / hexdatalength	60
Element typeSds / source	61
Element typeSds / target	61
Element typeSds / targetIsGroup	62
Element typeSds / forward	63
Element typeSds / validity	63
Element typeSds / tstamp	63
Element typeSds / encryption	64
Element typeSds / e2eegroup	64
Element interfaceSds / sendReport	64
Element typeSdsSendReport / target	65
Element typeSdsSendReport / msgRef	66
Element typeSdsSendReport / deliveryStatus	66
Element interfaceSds / response	66
Element interfaceSds / sendEvent	67
Element typeSdsSendEvent / msgRef	67
Element typeSdsSendEvent / sds	67
Element interfaceSds / receiveEvent	68
Element typeSdsReceiveEvent / sds	69
Element interfaceSds / reportEvent	70
Element typeSdsReportEvent / source	71
Element typeSdsReportEvent / target	72
Element typeSdsReportEvent / msgRef	73
Element typeSdsReportEvent / deliveryStatus	73
Element typeSdsReportEvent / tstamp	74
Element drgw / status	74
Element interfaceStatus / send	74
Element typeStatusSend / status	75
Element typeStatus / value	75
Element typeStatus / hexValue	75
Element typeStatus / source	76
Element typeStatus / target	76
Element typeStatus / tstamp	77
Element interfaceStatus / response	77
Element interfaceStatus / sendEvent	78

Element typeStatusSendEvent / status	78
Element interfaceStatus / receiveEvent	79
Element typeStatusReceiveEvent / status	79
Element drgw / org	80
Element interfaceOrg / get	81
Element typeOrgGet / orgblockId	81
Element typeOrganisationBlockId / orgblockId	81
Element typeOrganisationBlockIdNormal / id1	82
Element typeOrganisationBlockIdNormal / id2	82
Element typeOrganisationBlockIdNormal / id3	83
Element typeOrganisationBlockIdNormal / id4	83
Element typeOrganisationBlockIdNormal / id5	83
Element typeOrganisationBlockIdNormal / id6	83
Element typeOrganisationBlockId / orgblockIdSimple	84
Element interfaceOrg / getList	84
Element typeOrgGetList / orgblockId	84
Element interfaceOrg / response	85
Element interfaceOrg / getEvent	85
Element typeOrgGetEvent / orgblock	86
Element typeOrganisationBlock / orgblockId	86
Element typeOrganisationBlock / alias	86
Element interfaceOrg / getListEvent	87
Element typeOrgGetListEvent / orgblock	87
Element typeOrgGetListEvent / listEnd	87
Element interfaceOrg / event	88
Element typeOrgEvent / orgblock	88
Element typeOrgEvent / delete	89
Element drgw / group	89
Element interfaceGroup / get	91
Element typeGroupGet / group	92
Element interfaceGroup / getList	92
Element typeGroupGetList / orgblockId	92
Element interfaceGroup / getRadioMembers	93
Element typeGroupGetRadioMembers / group	93
Element interfaceGroup / getAppMembers	94
Element typeGroupGetAppMembers / group	94
Element interfaceGroup / getCombinations	94
Element typeGroupGetCombinations / group	95
Element interfaceGroup / track	95
Element typeGroupTrack / group	96
Element typeGroupTrack / mask	96
Element typeGroupTrack / stop	97
Element interfaceGroup / addRadioMember	97
Element typeGroupAddRadioMember / radio	97
Element typeGroupAddRadioMember / group	98
Element typeGroupAddRadioMember / membership	98
Element interfaceGroup / removeRadioMember	98
Element typeGroupRemoveRadioMember / radio	99
Element typeGroupRemoveRadioMember / group	99
Element interfaceGroup / addCombination	100
Element typeGroupAddCombination / group	100
Element typeGroupAddCombination / baseGroup	101
Element typeGroupAddCombination / force	101
Element interfaceGroup / removeCombination	101
Element typeGroupRemoveCombination / group	102
Element typeGroupRemoveCombination / baseGroup	102
Element interfaceGroup / subscribeData	103
Element typeGroupSubscribeData / group	103
Element typeGroupDataSubscription / addr	104
Element typeGroupDataSubscription / useSDS	104
Element typeGroupDataSubscription / useStatus	104
Element interfaceGroup / response	104
Element interfaceGroup / getEvent	105
Element typeGroupGetEvent / group	105
Element typeGroup / addr	106
Element typeGroup / alias	106
Element typeGroup / orgblockId	106
Element interfaceGroup / getListEvent	107
Element typeGroupGetListEvent / group	107
Element typeGroupGetListEvent / listEnd	107
Element interfaceGroup / getRadioMembersEvent	108
Element typeGroupGetRadioMembersEvent / group	108

Element typeGroupGetRadioMembersEvent / radio	109
Element typeGroupGetRadioMembersEvent / listEnd	109
Element interfaceGroup / getAppMembersEvent	109
Element typeGroupGetAppMembersEvent / app	110
Element typeGroupGetAppMembersEvent / listEnd	110
Element interfaceGroup / trackSubscriptionEvent	110
Element typeGroupTrackSubscriptionEvent / group	111
Element typeGroupTrackSubscriptionEvent / mask	111
Element typeGroupTrackSubscriptionEvent / stop	112
Element interfaceGroup / radioMemberEvent	112
Element typeGroupRadioMemberEvent / group	112
Element typeGroupRadioMemberEvent / radio	113
Element typeGroupRadioMemberEvent / delete	113
Element interfaceGroup / appMemberEvent	113
Element typeGroupAppMemberEvent / group	114
Element typeGroupAppMemberEvent / app	114
Element typeGroupAppMemberEvent / delete	115
Element interfaceGroup / combinationEvent	115
Element typeGroupCombinationEvent / group	116
Element typeGroupCombinationEvent / baseGroup	116
Element typeGroupCombinationEvent / constitGroup	116
Element interfaceGroup / addRadioMemberEvent	117
Element typeGroupAddRadioMemberEvent / radio	117
Element typeGroupAddRadioMemberEvent / group	118
Element interfaceGroup / removeRadioMemberEvent	118
Element typeGroupRemoveRadioMemberEvent / radio	119
Element typeGroupRemoveRadioMemberEvent / group	119
Element interfaceGroup / addCombinationEvent	119
Element typeGroupAddCombinationEvent / group	120
Element typeGroupAddCombinationEvent / baseGroup	120
Element interfaceGroup / removeCombinationEvent	120
Element typeGroupRemoveCombinationEvent / group	121
Element typeGroupRemoveCombinationEvent / baseGroup	121
Element interfaceGroup / subscribeDataEvent	122
Element typeGroupSubscribeDataEvent / group	122
Element interfaceGroup / event	123
Element typeGroupEvent / group	123
Element typeGroupEvent / delete	124
Element drgw / radio	124
Element interfaceRadio / get	125
Element typeRadioGet / radio	125
Element interfaceRadio / getList	126
Element typeRadioGetList / orgblockId	126
Element interfaceRadio / getGroups	127
Element typeRadioGetGroups / radio	127
Element typeRadio / issi	127
Element typeRadio / alias	128
Element typeRadio / orgblockId	128
Element typeRadio / opta	128
Element typeLastKnownOPTA / tstamp	129
Element typeLastKnownOPTA / opta	129
Element interfaceRadio / track	129
Element typeRadioTrack / radio	130
Element typeRadioTrack / stop	130
Element interfaceRadio / changeOpta	130
Element typeRadioChangeOpta / radio	131
Element typeRadioChangeOpta / opta	131
Element interfaceRadio / enable	132
Element typeRadioEnable / radio	132
Element typeRadioEnable / reason	133
Element typeRadioEnable / enable	133
Element interfaceRadio / disable	133
Element typeRadioDisable / radio	134
Element typeRadioDisable / reason	134
Element typeRadioDisable / enable	134
Element interfaceRadio / response	134
Element interfaceRadio / getEvent	135
Element typeRadioGetEvent / radio	135
Element interfaceRadio / getListEvent	136
Element typeRadioGetListEvent / radio	136
Element typeRadioGetListEvent / listEnd	137
Element interfaceRadio / getGroupsEvent	137

Element typeRadioGetGroupsEvent / radio	138
Element typeRadioGetGroupsEvent / group	138
Element typeRadioGroupSelection / group	138
Element typeRadioGroupSelection / level	139
Element typeRadioGetGroupsEvent / listEnd	139
Element interfaceRadio / trackSubscriptionEvent	139
Element typeRadioTrackSubscriptionEvent / radio	140
Element typeRadioTrackSubscriptionEvent / stop	140
Element interfaceRadio / trackEvent	141
Element typeRadioTrackEvent / trackingData	141
Element typeRadioTrackingData / radio	142
Element typeRadioTrackingData / registered	143
Element typeRadioTrackingData / exchangeId	143
Element typeRadioTrackingData / locationArea	143
Element typeRadioTrackingData / lastActive	143
Element typeRadioTrackingData / scanningOn	144
Element typeRadioTrackingData / status	144
Element typeStatusIndicator / value	144
Element typeStatusIndicator / time	144
Element typeRadioTrackingData / callType	145
Element typeRadioTrackingData / callParty	145
Element typeRadioTrackingData / dmoState	145
Element typeRadioTrackingData / emergency	145
Element interfaceRadio / groupsEvent	146
Element typeRadioGroupsEvent / radio	146
Element typeRadioGroupsEvent / group	147
Element typeRadioGroupsEvent / deletedGroup	147
Element interfaceRadio / changeOptaEvent	147
Element typeRadioChangeOptaEvent / radio	148
Element typeRadioChangeOptaEvent / opta	148
Element interfaceRadio / enableDisableEvent	148
Element typeRadioEnableDisableEvent / radio	149
Element typeRadioEnableDisableEvent / reason	149
Element typeRadioEnableDisableEvent / enabled	150
Element typeRadioEnableDisableEvent / overTheAir	150
Element interfaceRadio / event	150
Element typeRadioEvent / radio	151
Element typeRadioEvent / delete	151
Element drgw / app	151
Element interfaceApp / get	152
Element typeAppGet / app	152
Element interfaceApp / getList	153
Element typeAppGetList / orgblockId	153
Element interfaceApp / response	154
Element interfaceApp / getEvent	154
Element typeAppGetEvent / app	155
Element typeApplication / addr	155
Element typeApplication / alias	155
Element typeApplication / orgblockId	156
Element interfaceApp / getListEvent	156
Element typeAppGetListEvent / app	156
Element typeAppGetListEvent / listEnd	157
Element drgw / system	157
Element interfaceSystem / response	158
Element interfaceSystem / tetraStatesEvent	158
Element typeSystemTetraStatesEvent / tcsState	159
Element typeSystemTetraStatesEvent / dxtState	159
Element typeSystemTetraStatesEvent / cddconnectionState	159
Element typeSystemTetraStatesEvent / cddserverState	160
Element interfaceSystem / logEvent	160
Element typeSystemLogEvent / value	160
Element typeSystemLogEvent / text	161
Element interfaceSystem / event	161
Element typeSystemEvent / value	161
Element typeSystemEvent / text	162
Element drgw / management	162
Element interfaceManagement / get	163
Element typeManagementRequest / requestUri	163
Element typeManagementRequest / body	164
Element typeManagementRequest / authorization	164
Element typeManagementAuthorization / username	165
Element typeManagementAuthorization / password	165

Element typeManagementRequest / host	165
Element interfaceManagement / post	165
Element interfaceManagement / put	166
Element interfaceManagement / delete	167
Element interfaceManagement / response	168
Element interfaceManagement / event	169
Element typeManagementEvent / body	169
Element typeCallRequest / action	169
Element typeCallRequest / attributes	170
Element typeCallRequest / callingParty	171
Element typeCallRequest / calledParty	171
Element typeCallRequest / workstationId	172
Element check	172
Element typeSessionLogoutEvent / reason	173
Element typeSdsValidity / value	173
Element typeGroupGetCombinationsEvent / group	173
Element typeGroupGetCombinationsEvent / baseGroup	174
Element typeGroupGetCombinationsEvent / constitGroup	174
Complex Type(s)	175
Complex Type interfaceCall	175
Complex Type typeCallSelect	175
Complex Type typeRequest	176
Complex Type typeSelection	176
Complex Type typeAddress	177
Complex Type typeSubscriberAddress	177
Complex Type typeTSI	178
Complex Type typeExternal	179
Complex Type typeCallEvent	179
Complex Type typeEvent	180
Complex Type typeResult	180
Complex Type typeCallAttributes	181
Complex Type typeDisconnectCause	181
Complex Type typeCallPTTRequest	182
Complex Type typeCallKeyExchange	183
Complex Type typeResponse	183
Complex Type typeCallSelectEvent	183
Complex Type typeCallPTTEvent	184
Complex Type typeTxGranted	185
Complex Type typeEmpty	186
Complex Type typeCallUnitInEmergencyEvent	186
Complex Type typeCallKeyExchangeEvent	187
Complex Type interfaceSession	188
Complex Type typeSessionLogin	189
Complex Type typeSessionLogout	190
Complex Type typeSessionSupervise	190
Complex Type typeSessionCheck	191
Complex Type typeSessionLoginEvent	191
Complex Type typeSessionSuperviseEvent	192
Complex Type interfaceSds	192
Complex Type typeSdsSend	193
Complex Type typeSds	193
Complex Type typeSdsData	194
Complex Type typeSdsSendReport	195
Complex Type typeSdsSendEvent	196
Complex Type typeSdsReceiveEvent	196
Complex Type typeSdsReportEvent	197
Complex Type interfaceStatus	198
Complex Type typeStatusSend	198
Complex Type typeStatus	198
Complex Type typeStatusSendEvent	199
Complex Type typeStatusReceiveEvent	199
Complex Type interfaceOrg	200
Complex Type typeOrgGet	201
Complex Type typeOrganisationBlockId	201
Complex Type typeOrganisationBlockIdNormal	201
Complex Type typeOrgGetList	202
Complex Type typeOrgGetEvent	202
Complex Type typeOrganisationBlock	203
Complex Type typeOrgGetListEvent	203
Complex Type typeOrgEvent	204
Complex Type interfaceGroup	204
Complex Type typeGroupGet	206

Complex Type typeGroupGetList	207
Complex Type typeGroupGetRadioMembers	207
Complex Type typeGroupGetAppMembers	208
Complex Type typeGroupGetCombinations	208
Complex Type typeGroupTrack	209
Complex Type typeGroupAddRadioMember	209
Complex Type typeGroupRemoveRadioMember	210
Complex Type typeGroupAddCombination	210
Complex Type typeGroupRemoveCombination	211
Complex Type typeGroupSubscribeData	212
Complex Type typeGroupDataSubscription	212
Complex Type typeGroupGetEvent	212
Complex Type typeGroup	213
Complex Type typeGroupGetListEvent	213
Complex Type typeGroupGetRadioMembersEvent	214
Complex Type typeGroupGetAppMembersEvent	215
Complex Type typeGroupTrackSubscriptionEvent	215
Complex Type typeGroupRadioMemberEvent	216
Complex Type typeGroupAppMemberEvent	216
Complex Type typeGroupCombinationEvent	217
Complex Type typeGroupAddRadioMemberEvent	218
Complex Type typeGroupRemoveRadioMemberEvent	218
Complex Type typeGroupAddCombinationEvent	219
Complex Type typeGroupRemoveCombinationEvent	219
Complex Type typeGroupSubscribeDataEvent	220
Complex Type typeGroupEvent	221
Complex Type interfaceRadio	221
Complex Type typeRadioGet	223
Complex Type typeRadioGetList	223
Complex Type typeRadioGetGroups	223
Complex Type typeRadio	224
Complex Type typeLastKnownOPTA	224
Complex Type typeRadioTrack	225
Complex Type typeRadioChangeOpta	225
Complex Type typeRadioEnable	226
Complex Type typeRadioDisable	226
Complex Type typeRadioGetEvent	227
Complex Type typeRadioGetListEvent	228
Complex Type typeRadioGetGroupsEvent	228
Complex Type typeRadioGroupSelection	229
Complex Type typeRadioTrackSubscriptionEvent	229
Complex Type typeRadioTrackEvent	230
Complex Type typeRadioTrackingData	230
Complex Type typeStatusIndicator	231
Complex Type typeRadioGroupsEvent	231
Complex Type typeRadioChangeOptaEvent	232
Complex Type typeRadioEnableDisableEvent	232
Complex Type typeRadioEvent	233
Complex Type interfaceApp	233
Complex Type typeAppGet	234
Complex Type typeAppGetList	234
Complex Type typeAppGetEvent	235
Complex Type typeApplication	236
Complex Type typeAppGetListEvent	236
Complex Type interfaceSystem	236
Complex Type typeSystemTetraStatesEvent	237
Complex Type typeSystemLogEvent	238
Complex Type typeSystemEvent	238
Complex Type interfaceManagement	239
Complex Type typeManagementRequest	240
Complex Type typeManagementAuthorization	241
Complex Type typeManagementEvent	241
Complex Type typeCallRequest	242
Complex Type typeSessionLogoutEvent	242
Complex Type typeSdsValidity	243
Complex Type typeGroupGetCombinationsEvent	243
Simple Type(s)	244
Simple Type typeSelectionLevel	244
Simple Type typeDialString	245
Simple Type typeOpta	245
Simple Type typeResponseCode	245
Simple Type typeSourceSystem	246

Simple Type typeTetraCallId	246
Simple Type typeActionEvent	246
Simple Type typeCallMode	247
Simple Type typeCallType	248
Simple Type typeTxDemandPriority	248
Simple Type typeActionPTTRequest	249
Simple Type typeWorkstationId	249
Simple Type typeKeyExchangeAction	250
Simple Type typeTxGrant	250
Simple Type typeTxPriority	250
Simple Type typeUnitInEmergencyType	251
Simple Type typeEmergencyInfo	251
Simple Type typeKeyExchangeState	252
Simple Type typeKeyExchangeCode	252
Simple Type typeKeyExchangeTextPriority	253
Simple Type typeKeyExchangeText	253
Simple Type typeSuperviseTimeout	254
Simple Type typeSdsType	254
Simple Type typeReport	255
Simple Type typeOrganisationBlockIdSimple	255
Simple Type typeGroupTrackingMask	256
Simple Type typeMembershipType	256
Simple Type typeGroupSelectionLevel	256
Simple Type typeSystemElementState	257
Simple Type typeActionRequest	258
Simple Type typeKeyManagementTextPriority	259
Simple Type typeAddressingStyle	259
Simple Type typeGroupTrackingMaskValues	259
Attribute(s)	260
Attribute drgw / @version	260

Namespace: "DR-GW"

Schema(s)

Main schema DR-GW.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Call.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Call.select.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Call.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Call.call.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Call.ptt.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Call.keyExchange.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Call.unitInEmergency.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Session.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Session.check.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Session.login.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Session.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Session.supervise.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Session.logout.xsd

Namespace	DR-GW
Properties	attribute form default: qualified

	element form default: qualified
--	---------------------------------

Included schema DR-GW-Sds.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Sds.send.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Sds.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Sds.receive.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Status.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Status.send.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Status.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Status.receive.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-OrganisationBlock.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-OrganisationBlock.get.xsd

Namespace	DR-GW

Properties	attribute form default: qualified element form default: qualified
------------	--

Included schema DR-GW-OrganisationBlock.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-OrganisationBlock.getList.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-OrganisationBlock.event.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.get.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.getList.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.getRadioMembers.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.getAppMembers.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.getCombinations.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.track.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.addRadioMember.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.removeRadioMember.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.addCombination.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.removeCombination.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.subscribeData.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Group.event.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified

	element form default: qualified
--	---------------------------------

Included schema DR-GW-Radio.get.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.getList.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.getGroups.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.track.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.changeOpta.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.enableDisable.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Radio.event.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Application.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Application.get.xsd

Namespace	DR-GW
Properties	attribute form default: qualified element form default: qualified

Included schema DR-GW-Application.types.xsd

Namespace	DR-GW

Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-Application.getList.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-System.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-System.tetraStates.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-System.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-System.log.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-System.event.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-Management.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-Management.get.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Included schema DR-GW-Management.types.xsd

Namespace	DR-GW
Properties	attribute form default: qualified
	element form default: qualified

Element(s)

Element drgw

Namespace	DR-GW
Annotations	Root DR-GW element. Elements underneath represent various DR-GW interfaces.
Diagram	<pre> classDiagram class drgw { <<Root DR-GW element. Elements underneath represent various DR-GW interfaces.>> @Attributes version: Fixed 2.0 call session sds status org group radio app system management } </pre> <p>The diagram shows the <code>drgw</code> element as the root. It has attributes for version (Fixed 2.0) and contains nine child elements: <code>call</code>, <code>session</code>, <code>sds</code>, <code>status</code>, <code>org</code>, <code>group</code>, <code>radio</code>, <code>app</code>, and <code>system</code>. Each child element has a type attribute indicating its interface type: <code>interfaceCall</code>, <code>interfaceSession</code>, <code>interfaceSds</code>, <code>interfaceStatus</code>, <code>interfaceOrg</code>, <code>interfaceGroup</code>, <code>interfaceRadio</code>, <code>interfaceApp</code>, and <code>interfaceSystem</code>. A note on the left side of the diagram states: "Root DR-GW element. Elements underneath represent various DR-GW interfaces."</p>
Properties	content: complex
Model	call session sds status org group radio app system management
Children	app, call, group, management, org, radio, sds, session, status, system
Instance	<pre> <drgw version="2.0" xmlns="DR-GW"> <call>{1,1}</call> <session>{1,1}</session> <sds>{1,1}</sds> <status>{1,1}</status> <org>{1,1}</org> <group>{1,1}</group> <radio>{1,1}</radio> <app>{1,1}</app> <system>{1,1}</system> <management>{1,1}</management> </drgw> </pre>

Attributes	QName	Type	Fixed	Use	
	version		2.0	required	
Source	<pre><xs:element name="drgw"> <xs:annotation> <xs:documentation>Root DR-GW element. Elements underneath represent various DR-GW interfaces.</xs:documentation> </xs:annotation> <xs:complexType> <xs:choice> <xs:element name="call" type="interfaceCall"/> <xs:element name="session" type="interfaceSession"/> <xs:element name="sds" type="interfaceSds"/> <xs:element name="status" type="interfaceStatus"/> <xs:element name="org" type="interfaceOrg"/> <xs:element name="group" type="interfaceGroup"/> <xs:element name="radio" type="interfaceRadio"/> <xs:element name="app" type="interfaceApp"/> <xs:element name="system" type="interfaceSystem"/> <xs:element name="management" type="interfaceManagement"/> </xs:choice> <xs:attribute name="version" fixed="2.0" use="required"/> </xs:complexType> </xs:element></pre>				

Element drgw / call

Namespace	DR-GW
Diagram	<p>DR-GW-Call. Use for call control/ call monitoring. This is the only element, that can be used via both SIP/SOAP. When...</p>
Type	interfaceCall
Properties	content: complex
Model	select request pttRequest keyExchange response selectEvent event pttEvent unitInEmergencyEvent keyExchangeEvent
Children	event, keyExchange, keyExchangeEvent, pttEvent, pttRequest, request, response, select, selectEvent, unitInEmergencyEvent
Instance	<pre><call xmlns="DR-GW"> <select>{1,1}</select> <request>{1,1}</request> <pttRequest>{1,1}</pttRequest> <keyExchange>{1,1}</keyExchange> <response>{1,1}</response></pre>

	<pre> <selectEvent>{1,1}</selectEvent> <event>{1,1}</event> <pttEvent>{1,1}</pttEvent> <unitInEmergencyEvent>{1,1}</unitInEmergencyEvent> <keyExchangeEvent>{1,1}</keyExchangeEvent> </call> </pre>
Source	<pre><xss:element name="call" type="interfaceCall"/></pre>

Element interfaceCall / select

Namespace	DR-GW
Diagram	<p>The method reserves speech line for the targets of selection operation, sets the selection level of the targets of...</p>
Type	typeCallSelect
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeCallSelect
Properties	content: complex
Model	requestId , target
Children	requestId, target
Instance	<pre> <select xmlns="DR-GW"> <requestId>{1,1}</requestId> <target>{1,1}</target> </select> </pre>
Source	<pre><xss:element name="select" type="typeCallSelect"/></pre>

Element typeRequest / requestId

Namespace	DR-GW
Diagram	<p>Built-in derived type. The long datatype is derived from integer by setting the value of maxInclusive to be...</p>
Type	xs:long
Properties	content: simple
Source	<pre><xss:element name="requestId" type="xs:long"/></pre>

Element typeCallSelect / target

Namespace	DR-GW
Diagram	

Type	typeSelection
Properties	content: complex
Model	level , target
Children	level, target
Instance	<target xmlns="DR-GW"> <level>{1,1}</level> <target>{1,1}</target> </target>
Source	<xss:element name="target" type="typeSelection"/>

Element typeSelection / level

Namespace	DR-GW															
Diagram	<pre> graph LR level((level)) -- "-" --> typeSelectionLevel((typeSelectionLevel)) typeSelectionLevel -- "+" --> target classDef=callout classDefTitle="Defines how the target is monitored." </pre>															
Type	typeSelectionLevel															
Properties	content: simple															
Facets	<table> <tr> <td>enumeration</td> <td>no</td> <td>No selection. Used to remove selection.</td> </tr> <tr> <td>enumeration</td> <td>event</td> <td>Event monitoring.</td> </tr> <tr> <td>enumeration</td> <td>audio</td> <td>Audio monitoring.</td> </tr> <tr> <td>enumeration</td> <td>use</td> <td>Selection level use.</td> </tr> <tr> <td>enumeration</td> <td>a_use</td> <td>Selection level active use.</td> </tr> </table>	enumeration	no	No selection. Used to remove selection.	enumeration	event	Event monitoring.	enumeration	audio	Audio monitoring.	enumeration	use	Selection level use.	enumeration	a_use	Selection level active use.
enumeration	no	No selection. Used to remove selection.														
enumeration	event	Event monitoring.														
enumeration	audio	Audio monitoring.														
enumeration	use	Selection level use.														
enumeration	a_use	Selection level active use.														
Source	<xss:element name="level" type="typeSelectionLevel"/>															

Element typeSelection / target

Namespace	DR-GW
Diagram	<pre> graph LR target((target)) -- "-" --> typeAddress((typeAddress)) typeAddress -- "-" --> target typeAddress subscriber alias msisdn fssn external opta cell classDef=callout classDefTitle="Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA)." </pre>
Type	typeAddress
Properties	content: complex
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<target xmlns="DR-GW">

	<pre> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </target> </pre>
Source	<code><xs:element name="target" type="typeAddress" /></code>

Element typeAddress / subscriber

Namespace	DR-GW				
Diagram	<pre> classDiagram class subscriber class typeSubscriberAddress { <<ssi>> <<tsi>> } subscriber --> typeSubscriberAddress </pre>				
Type	typeSubscriberAddress				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	ssi tsi				
Children	ssi, tsi				
Instance	<pre> <subscriber xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </subscriber> </pre>				
Source	<code><xs:element name="subscriber" type="typeSubscriberAddress" minOccurs="0" /></code>				

Element typeSubscriberAddress / ssi

Namespace	DR-GW		
Diagram	<pre> classDiagram class ssi class xsUnsignedLong ssi --> xsUnsignedLong </pre> <p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>		
Type	xs:unsignedLong		
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> </table>	content:	simple
content:	simple		
Source	<code><xs:element name="ssi" type="xs:unsignedLong" /></code>		

Element typeSubscriberAddress / tsi

Namespace	DR-GW
Diagram	<pre> classDiagram class tsi class typeTSI { <<mnc>> <<mcc>> <<ssi>> } tsi --> typeTSI </pre> <p>Basic type for TETRA subscriber identity containing Network code(MNC) and Country code(MCC).</p>
Type	typeTSI

Properties	content: complex
Model	mnc , mcc , ssi
Children	mcc, mnc, ssi
Instance	<pre><tsi xmlns="DR-GW"> <mnc>{1,1}</mnc> <mcc>{1,1}</mcc> <ssi>{1,1}</ssi> </tsi></pre>
Source	<code><xss:element name="tsi" type="typeTSI" /></code>

Element typeTSI / mnc

Namespace	DR-GW
Diagram	<pre> graph LR mnc[mnc] --> xs[xs:unsignedShort] </pre> <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>
Type	xs:unsignedShort
Properties	content: simple
Source	<code><xss:element name="mnc" type="xs:unsignedShort" /></code>

Element typeTSI / mcc

Namespace	DR-GW
Diagram	<pre> graph LR mcc[mcc] --> xs[xs:unsignedShort] </pre> <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>
Type	xs:unsignedShort
Properties	content: simple
Source	<code><xss:element name="mcc" type="xs:unsignedShort" /></code>

Element typeTSI / ssi

Namespace	DR-GW
Diagram	<pre> graph LR ssi[ssi] --> xs[xs:unsignedLong] </pre> <p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>
Type	xs:unsignedLong
Properties	content: simple
Source	<code><xss:element name="ssi" type="xs:unsignedLong" /></code>

Element typeAddress / alias

Namespace	DR-GW
Diagram	<pre> graph LR alias[alias] --> xs[xs:normalizedString] </pre> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>

Type	xs:normalizedString
Properties	<p>content: simple</p> <p>minOccurs: 0</p>
Source	<xs:element name="alias" type="xs:normalizedString" minOccurs="0" />

Element typeAddress / msisdn

Namespace	DR-GW
Diagram	<pre> classDiagram class msisdn class typeDialString msisdn "0..1" -- "1" typeDialString note over typeDialString: Allowed characters are digits 0 - 9, *, #, A, B, C and D. Maximum length is 24 characters. </pre>
Type	typeDialString
Properties	<p>content: simple</p> <p>minOccurs: 0</p>
Facets	<p>maxLength 24</p>
Source	<xs:element name="msisdn" type="typeDialString" minOccurs="0" />

Element typeAddress / fssn

Namespace	DR-GW
Annotations	Fleet specific short number
Diagram	<pre> classDiagram class fssn class xsUnsignedLong fssn "0..1" -- "1" xsUnsignedLong note over fssn: Fleet specific short number note over xsUnsignedLong: Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of... </pre>
Type	xs:unsignedLong
Properties	<p>content: simple</p> <p>minOccurs: 0</p>
Source	<xs:element name="fssn" type="xs:unsignedLong" minOccurs="0" > <xs:annotation> <xs:documentation>Fleet specific short number</xs:documentation> </xs:annotation> </xs:element>

Element typeAddress / external

Namespace	DR-GW
Diagram	<pre> classDiagram class external class typeExternal class gatewayNumber class number external "0..1" -- "1" typeExternal typeExternal "0..1" -- "1" gatewayNumber typeExternal "0..1" -- "1" number note over typeExternal: External number consisting of Gateway number + DialString </pre>
Type	typeExternal
Properties	<p>content: complex</p> <p>minOccurs: 0</p>
Model	gatewayNumber , number
Children	gatewayNumber, number

Instance	<pre><external xmlns="DR-GW"> <gatewayNumber>{1,1}</gatewayNumber> <number>{1,1}</number> </external></pre>
Source	<pre><xss:element name="external" type="typeExternal" minOccurs="0" /></pre>

Element typeExternal / gatewayNumber

Namespace	DR-GW
Diagram	<pre> classDiagram class gatewayNumber { <<xs:unsignedLong>> } gatewayNumber < -- xs:unsignedLong </pre> <p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>
Type	xs:unsignedLong
Properties	content: simple
Source	<pre><xss:element name="gatewayNumber" type="xs:unsignedLong" /></pre>

Element typeExternal / number

Namespace	DR-GW
Diagram	<pre> classDiagram class number { <<typeDialString>> } number < -- typeDialString </pre> <p>Allowed characters are digits 0 - 9, *, #, A, B, C and D. Maximum length is 24 characters.</p>
Type	typeDialString
Properties	content: simple
Facets	maxLength 24
Source	<pre><xss:element name="number" type="typeDialString" /></pre>

Element typeAddress / opta

Namespace	DR-GW
Diagram	<pre> classDiagram class opta { <<typeOpta>> } opta < -- typeOpta </pre> <p>OPTA string. Maximum length is 24 characters.</p>
Type	typeOpta
Properties	content: simple minOccurs: 0
Facets	maxLength 24
Source	<pre><xss:element name="opta" type="typeOpta" minOccurs="0" /></pre>

Element typeAddress / cell

Namespace	DR-GW
Diagram	<pre> classDiagram class cell { <<xs:short>> } cell < -- xs:short </pre> <p>Built-in derived type. The short datatype is derived from int by setting the value of maxInclusive to be 32767 and...</p>
Type	xs:short
Properties	content: simple

	minOccurs:	0
Source	<xs:element name="cell" type="xs:short" minOccurs="0"/>	

Element interfaceCall / request

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeCallEvent typeCallEvent "0..1" -- "0..1" requestId : xs:short typeCallEvent "0..1" -- "0..1" result : xs:string typeCallEvent "0..1" -- "0..1" tetraCallId : xs:unsignedLong typeCallEvent "0..1" -- "0..1" action : xs:string typeCallEvent "0..1" -- "0..1" attributes : xs:string typeCallEvent "0..1" -- "0..1" callingParty : xs:string typeCallEvent "0..1" -- "0..1" calledParty : xs:string typeCallEvent "0..1" -- "0..1" disconnectCause : xs:string typeCallEvent "0..1" -- "0..1" request : xs:string </pre>
Type	typeCallEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeCallEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , tetraCallId{0,1} , action , attributes{0,1} , callingParty{0,1} , calledParty{0,1} , disconnectCause{0,1}
Children	action, attributes, calledParty, callingParty, disconnectCause, requestId, result, tetraCallId
Instance	<pre> <request xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <tetraCallId>{0,1}</tetraCallId> <action>{1,1}</action> <attributes>{0,1}</attributes> <callingParty>{0,1}</callingParty> <calledParty>{0,1}</calledParty> <disconnectCause>{0,1}</disconnectCause> </request> </pre>
Source	<xs:element name="request" type="typeCallEvent" />

Element typeEvent / requestId

Namespace	DR-GW				
Diagram	<pre> requestId <--> xs:unsignedLong </pre> <p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>				
Type	xs:unsignedLong				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				

Source	<code><xss:element name="requestId" type="xs:unsignedLong" minOccurs="0" /></code>
--------	--

Element typeEvent / result

Namespace	DR-GW				
Diagram	<pre> classDiagram class typeResult { responseCode sourceSystem result } class result typeResult < -- result </pre> <p>Common result values used in every response and optional specific subsystem result codes.</p>				
Type	typeResult				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	responseCode , sourceSystem{0,1} , result{0,1}				
Children	responseCode, result, sourceSystem				
Instance	<pre> <result xmlns="DR-GW"> <responseCode>{1,1}</responseCode> <sourceSystem>{0,1}</sourceSystem> <result>{0,1}</result> </result> </pre>				
Source	<code><xss:element name="result" type="typeResult" minOccurs="0" /></code>				

Element typeResult / responseCode

Namespace	DR-GW												
Diagram	<pre> classDiagram class responseCode { typeResponseCode } </pre>												
Type	typeResponseCode												
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> </table>	content:	simple										
content:	simple												
Facets	<table border="1"> <tr> <td>enumeration</td> <td>success</td> </tr> <tr> <td>enumeration</td> <td>final_response_pending</td> </tr> <tr> <td>enumeration</td> <td>error</td> </tr> <tr> <td>enumeration</td> <td>not_authorized_error</td> </tr> <tr> <td>enumeration</td> <td>temporary_failure</td> </tr> <tr> <td>enumeration</td> <td>subscription_failed</td> </tr> </table>	enumeration	success	enumeration	final_response_pending	enumeration	error	enumeration	not_authorized_error	enumeration	temporary_failure	enumeration	subscription_failed
enumeration	success												
enumeration	final_response_pending												
enumeration	error												
enumeration	not_authorized_error												
enumeration	temporary_failure												
enumeration	subscription_failed												
Source	<code><xss:element name="responseCode" type="typeResponseCode" /></code>												

Element typeResult / sourceSystem

Namespace	DR-GW				
Diagram	<pre> classDiagram class sourceSystem { typeSourceSystem } </pre>				
Type	typeSourceSystem				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Facets	<table border="1"> <tr> <td>enumeration</td> <td>DR-GW</td> </tr> </table>	enumeration	DR-GW		
enumeration	DR-GW				

	enumeration	TCS-API
	enumeration	TETRA
	enumeration	NEM-API
Source	<xs:element name="sourceSystem" type="typeSourceSystem" minOccurs="0" />	

Element typeResult / result

Namespace	DR-GW				
Diagram	<p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>				
Type	xs:unsignedLong				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xs:element name="result" type="xs:unsignedLong" minOccurs="0" />				

Element typeCallEvent / tetraCallId

Namespace	DR-GW				
Diagram	<p>TETRA callId, generated by the TCS. In the TCS-API it is defined as long, but Airbus doesn't have it under control if...</p>				
Type	typeTetraCallId				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xs:element name="tetraCallId" type="typeTetraCallId" minOccurs="0" />				

Element typeCallEvent / action

Namespace	DR-GW																		
Diagram	<p>All possible call actions.</p>																		
Type	typeActionEvent																		
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> </table>	content:	simple																
content:	simple																		
Facets	<table border="1"> <tr> <td>enumeration</td> <td>incoming</td> <td>This event fired when there is an incoming call. This is the first indication of a new incoming call.</td> </tr> <tr> <td>enumeration</td> <td>connected</td> <td>This event is used to inform that call has been connected and call setup is finished.</td> </tr> <tr> <td>enumeration</td> <td>held</td> <td>This event is used to inform TCS Client that individual call was put to hold.</td> </tr> <tr> <td>enumeration</td> <td>resumed</td> <td>This event is used to inform that individual call has been taken from hold.</td> </tr> <tr> <td>enumeration</td> <td>disconnected</td> <td>This event is used to inform that the call was disconnected.</td> </tr> <tr> <td>enumeration</td> <td>transferred</td> <td>This event is a response to transfer method call and indicates the</td> </tr> </table>	enumeration	incoming	This event fired when there is an incoming call. This is the first indication of a new incoming call.	enumeration	connected	This event is used to inform that call has been connected and call setup is finished.	enumeration	held	This event is used to inform TCS Client that individual call was put to hold.	enumeration	resumed	This event is used to inform that individual call has been taken from hold.	enumeration	disconnected	This event is used to inform that the call was disconnected.	enumeration	transferred	This event is a response to transfer method call and indicates the
enumeration	incoming	This event fired when there is an incoming call. This is the first indication of a new incoming call.																	
enumeration	connected	This event is used to inform that call has been connected and call setup is finished.																	
enumeration	held	This event is used to inform TCS Client that individual call was put to hold.																	
enumeration	resumed	This event is used to inform that individual call has been taken from hold.																	
enumeration	disconnected	This event is used to inform that the call was disconnected.																	
enumeration	transferred	This event is a response to transfer method call and indicates the																	

	result of the request.
Source	<xs:element name="action" type="typeActionEvent" />

Element typeCallEvent / attributes

Namespace	DR-GW				
Diagram	<p>Contains all attributes of the TETRA voice call.</p>				
Type	typeCallAttributes				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	hook{0,1} , mode{0,1} , commtype{0,1} , priority{0,1} , encryption{0,1} , ambienceListen{0,1} , req2speak{0,1} , demandPriority{0,1}				
Children	ambienceListen, commtype, demandPriority, encryption, hook, mode, priority, req2speak				
Instance	<pre> <attributes xmlns="DR-GW"> <hook>{0,1}</hook> <mode>{0,1}</mode> <commtype>{0,1}</commtype> <priority>{0,1}</priority> <encryption>{0,1}</encryption> <ambienceListen>{0,1}</ambienceListen> <req2speak>{0,1}</req2speak> <demandPriority>{0,1}</demandPriority> </attributes> </pre>				
Source	<xs:element name="attributes" type="typeCallAttributes" minOccurs="0" />				

Element typeCallAttributes / hook

Namespace	DR-GW				
Diagram					
Type	xs:boolean				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xs:element name="hook" type="xs:boolean" minOccurs="0" />				

Element typeCallAttributes / mode

Namespace	DR-GW
Diagram	<pre> graph LR mode["mode"] -- "0..1" --> typeCallMode["typeCallMode"] typeCallMode -- "Call mode attribute. Choices are simplex or duplex." --> callout[] </pre>
Type	typeCallMode
Properties	content: simple minOccurs: 0
Facets	enumeration simplex enumeration duplex
Source	<xss:element name="mode" type="typeCallMode" minOccurs="0" />

Element typeCallAttributes / commtype

Namespace	DR-GW
Diagram	<pre> graph LR commtype["commtype"] -- "0..1" --> typeCallType["typeCallType"] typeCallType -- "Call type attribute. Choices are Point2Point, Point2MultiPoint or Broadcast." --> callout[] </pre>
Type	typeCallType
Properties	content: simple minOccurs: 0
Facets	enumeration p2p enumeration p2mp enumeration bcast
Source	<xss:element name="commtype" type="typeCallType" minOccurs="0" />

Element typeCallAttributes / priority

Namespace	DR-GW
Diagram	<pre> graph LR priority["priority"] -- "0..1" --> xsUnsignedByte["xs:unsignedByte"] xsUnsignedByte -- "Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to..." --> callout[] </pre>
Type	xs:unsignedByte
Properties	content: simple minOccurs: 0 default: 1
Source	<xss:element name="priority" type="xs:unsignedByte" default="1" minOccurs="0" />

Element typeCallAttributes / encryption

Namespace	DR-GW
Diagram	<pre> graph LR encryption["encryption"] -- "0..1" --> xsBoolean["xs:boolean"] xsBoolean -- "Built-in primitive type. It defines the boolean values true and false." --> callout[] </pre>
Type	xs:boolean

Properties	content: simple minOccurs: 0 default: true
Source	<xs:element name="encryption" type="xs:boolean" default="true" minOccurs="0"/>

Element typeCallAttributes / ambienceListen

Namespace	DR-GW
Diagram	<pre> graph LR ambienceListen --> xsboolean note["Built-in primitive type. It defines the boolean values true and false."] </pre>
Type	xs:boolean
Properties	content: simple minOccurs: 0 default: 0
Source	<xs:element name="ambienceListen" type="xs:boolean" default="0" minOccurs="0"/>

Element typeCallAttributes / req2speak

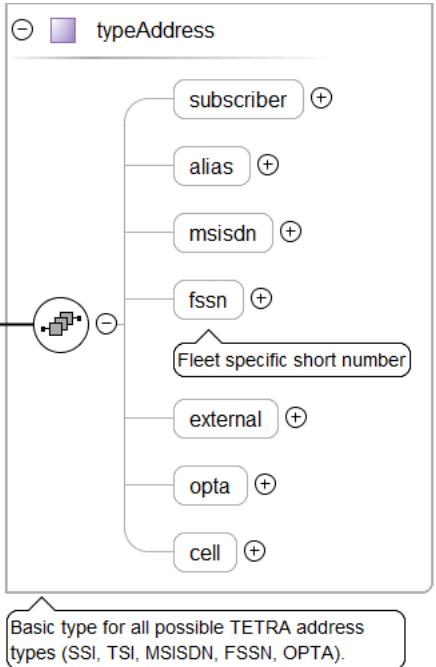
Namespace	DR-GW
Diagram	<pre> graph LR req2speak --> xsboolean note["Built-in primitive type. It defines the boolean values true and false."] </pre>
Type	xs:boolean
Properties	content: simple minOccurs: 0 default: 1
Source	<xs:element name="req2speak" type="xs:boolean" default="1" minOccurs="0"/>

Element typeCallAttributes / demandPriority

Namespace	DR-GW
Diagram	<pre> graph LR demandPriority --> typeTxDemandPriority note["Defines priority of speech item request: normal, pre-emptive, or emergency."] </pre>
Type	typeTxDemandPriority
Properties	content: simple minOccurs: 0 default: normal
Facets	enumeration normal enumeration preemptive enumeration emergency
Source	<xs:element name="demandPriority" type="typeTxDemandPriority" default="normal" minOccurs="0"/>

Element typeCallEvent / callingParty

Namespace	DR-GW
-----------	-------

Diagram	 <p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>				
Type	typeAddress				
Properties	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">content:</td><td style="padding: 2px;">complex</td></tr> <tr> <td style="padding: 2px;">minOccurs:</td><td style="padding: 2px;">0</td></tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}				
Children	alias, cell, external, fssn, msisdn, opta, subscriber				
Instance	<pre><callingParty xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </callingParty></pre>				
Source	<code><xss:element name="callingParty" type="typeAddress" minOccurs="0" /></code>				

Element typeCallEvent / calledParty

Namespace	DR-GW
-----------	-------

Diagram	<p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>				
Type	typeAddress				
Properties	<table border="1"> <tr> <td>content:</td><td>complex</td></tr> <tr> <td>minOccurs:</td><td>0</td></tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}				
Children	alias, cell, external, fssn, msisdn, opta, subscriber				
Instance	<pre><calledParty xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </calledParty></pre>				
Source	<code><xs:element name="calledParty" type="typeAddress" minOccurs="0" /></code>				

Element typeCallEvent / disconnectCause

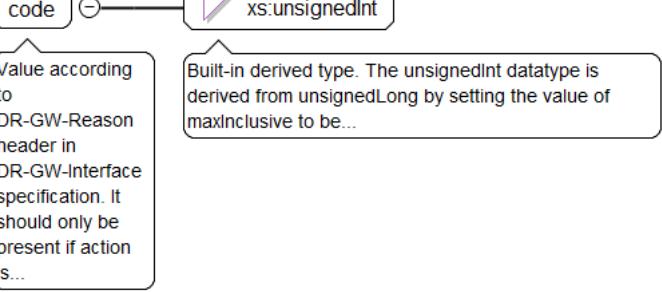
Namespace	DR-GW		
Diagram			
Type	typeDisconnectCause		
Properties	<table border="1"> <tr> <td>content:</td><td>complex</td></tr> </table>	content:	complex
content:	complex		

	minOccurs: 0
Model	protocol , code , text{0,1}
Children	code, protocol, text
Instance	<disconnectCause xmlns="DR-GW"> <protocol>{1,1}</protocol> <code>{1,1}</code> <text>{0,1}</text> </disconnectCause>
Source	<xss:element name="disconnectCause" type="typeDisconnectCause" minOccurs="0"/>

Element typeDisconnectCause / protocol

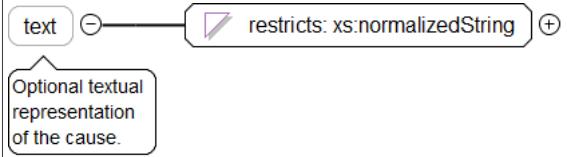
Namespace	DR-GW
Diagram	
Type	restriction of xs:normalizedString
Properties	content: simple
Facets	enumeration DR-GW enumeration TCS-API
Source	<xss:element name="protocol"> <xss:simpleType> <xss:restriction base="xs:normalizedString"> <xss:enumeration value="DR-GW"/> <xss:enumeration value="TCS-API"/> </xss:restriction> <br < xss:simpletype><br=""></br <> </xss:element>

Element typeDisconnectCause / code

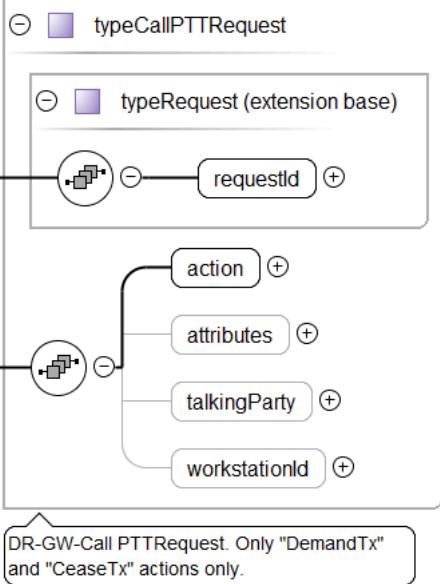
Namespace	DR-GW
Annotations	Value according to DR-GW-Reason header in DR-GW-Interface specification. It should only be present if action is "disconnected" and holds the reason for call disconnection.
Diagram	
Type	xs:unsignedInt
Properties	content: simple
Source	<xss:element name="code" type="xs:unsignedInt"> <xss:annotation> <xss:documentation>Value according to DR-GW-Reason header in DR-GW-Interface specification. It should only be present if action is "disconnected" and holds the reason for call disconnection.</xss:documentation> <br < xss:annotation><br=""></br <> </xss:element>

Element typeDisconnectCause / text

Namespace	DR-GW
Annotations	Optional textual representation of the cause.

Diagram					
Type	restriction of xs:normalizedString				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Facets	maxLength 80				
Source	<pre><xs:element name="text" minOccurs="0"> <xs:annotation> <xs:documentation>Optional textual representation of the cause.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:normalizedString"> <xs:maxLength value="80"/> </xs:restriction> </xs:simpleType> </xs:element></pre>				

Element interfaceCall / pttRequest

Namespace	DR-GW
Diagram	
Type	typeCallPTTRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeCallPTTRequest
Properties	content: complex
Model	requestId , action , attributes{0,1} , talkingParty{0,1} , workstationId{0,1}
Children	action, attributes, requestId, talkingParty, workstationId
Instance	<pre><pttRequest xmlns="DR-GW"> <requestId>{1,1}</requestId> <action>{1,1}</action> <attributes>{0,1}</attributes> <talkingParty>{0,1}</talkingParty> <workstationId>{0,1}</workstationId> </pttRequest></pre>
Source	<pre><xs:element name="pttRequest" type="typeCallPTTRequest" /></pre>

Element typeCallPTTRequest / action

Namespace	DR-GW
-----------	-------

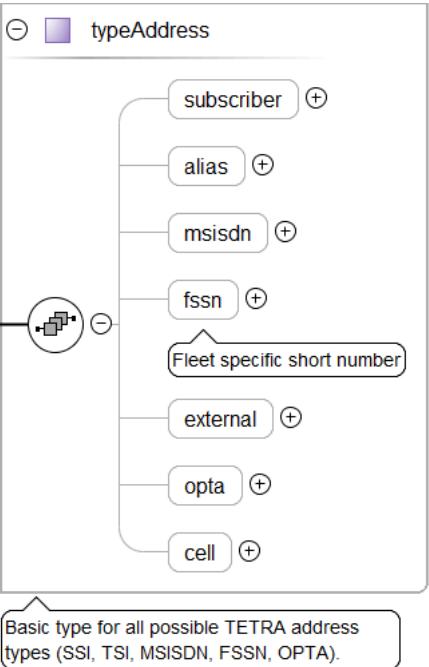
Diagram							
Type	typeActionPTTRequest						
Properties	content: simple						
Facets	<table> <tr> <td>enumeration</td> <td>demandtx</td> <td>This method can be used to request a speech item for a connected call.</td> </tr> <tr> <td>enumeration</td> <td>ceasetx</td> <td>This method is used to inform the system that the speech item is not needed any more.</td> </tr> </table>	enumeration	demandtx	This method can be used to request a speech item for a connected call.	enumeration	ceasetx	This method is used to inform the system that the speech item is not needed any more.
enumeration	demandtx	This method can be used to request a speech item for a connected call.					
enumeration	ceasetx	This method is used to inform the system that the speech item is not needed any more.					
Source	<code><xs:element name="action" type="typeActionPTTRequest"/></code>						

Element typeCallPTTRequest / attributes

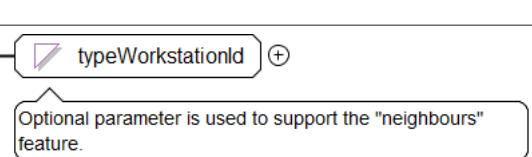
Namespace	DR-GW				
Diagram					
Type	typeCallAttributes				
Properties	<table> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	hook{0,1} , mode{0,1} , commtype{0,1} , priority{0,1} , encryption{0,1} , ambienceListen{0,1} , req2speak{0,1} , demandPriority{0,1}				
Children	ambienceListen, commtype, demandPriority, encryption, hook, mode, priority, req2speak				
Instance	<code><attributes xmlns="DR-GW"> <hook>{0,1}</hook> <mode>{0,1}</mode> <commtype>{0,1}</commtype> <priority>{0,1}</priority> <encryption>{0,1}</encryption> <ambienceListen>{0,1}</ambienceListen> <req2speak>{0,1}</req2speak> <demandPriority>{0,1}</demandPriority> </attributes></code>				
Source	<code><xs:element name="attributes" type="typeCallAttributes" minOccurs="0"/></code>				

Element typeCallPTTRequest / talkingParty

Namespace	DR-GW
-----------	-------

Diagram					
Type	typeAddress				
Properties	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">content:</td><td style="padding: 2px;">complex</td></tr> <tr> <td style="padding: 2px;">minOccurs:</td><td style="padding: 2px;">0</td></tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}				
Children	alias, cell, external, fssn, msisdn, opta, subscriber				
Instance	<pre><talkingParty xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </talkingParty></pre>				
Source	<code><xs:element name="talkingParty" type="typeAddress" minOccurs="0" /></code>				

Element typeCallPTTRequest / workstationId

Namespace	DR-GW				
Diagram					
Type	typeWorkstationId				
Properties	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">content:</td><td style="padding: 2px;">simple</td></tr> <tr> <td style="padding: 2px;">minOccurs:</td><td style="padding: 2px;">0</td></tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xs:element name="workstationId" type="typeWorkstationId" minOccurs="0" /></code>				

Element interfaceCall / keyExchange

Namespace	DR-GW
-----------	-------

Diagram	<pre> classDiagram typeCallKeyExchange < -- typeRequest typeCallKeyExchange "0..1" --> "1..1" requestId typeCallKeyExchange "0..1" --> "1..1" action note over typeCallKeyExchange: For triggering the group key exchange. Key exchange events are sent in Call_KeyXEvent. </pre>
Type	typeCallKeyExchange
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeCallKeyExchange
Properties	content: complex
Model	requestId , action
Children	action, requestId
Instance	<pre> <keyExchange xmlns="DR-GW"> <requestId>{1,1}</requestId> <action>{1,1}</action> </keyExchange> </pre>
Source	<code><xss:element name="keyExchange" type="typeCallKeyExchange" /></code>

Element typeCallKeyExchange / action

Namespace	DR-GW				
Diagram	<pre> classDiagram typeKeyExchangeAction "0..1" --> "1..1" action note over typeKeyExchangeAction: Action type for key exchange request. </pre>				
Type	typeKeyExchangeAction				
Properties	content: simple				
Facets	<table border="1"> <tr> <td>enumeration</td> <td>start</td> </tr> <tr> <td>enumeration</td> <td>stop</td> </tr> </table>	enumeration	start	enumeration	stop
enumeration	start				
enumeration	stop				
Source	<code><xss:element name="action" type="typeKeyExchangeAction" /></code>				

Element interfaceCall / response

Namespace	DR-GW
Diagram	<pre> classDiagram typeResponse "0..1" --> "1..1" requestId typeResponse "0..1" --> "1..1" result note over typeResponse: Response contains result of execution of any method. </pre>
Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result

Instance	<pre><response xmlns="DR-GW"> <requestId>{1,1}</requestId> <result>{1,1}</result> </response></pre>
Source	<pre><xs:element name="response" type="typeResponse" /></pre>

Element typeResponse / requestId

Namespace	DR-GW
Diagram	<p>The diagram shows a class named 'requestId' with a multiplicity of 1..1. It has a directed association labeled 'xs:unsignedLong' pointing to it. A callout box indicates: 'Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...'.</p>
Type	xs:unsignedLong
Properties	content: simple
Source	<pre><xs:element name="requestId" type="xs:unsignedLong" /></pre>

Element typeResponse / result

Namespace	DR-GW
Diagram	<p>The diagram shows a class named 'result' with a multiplicity of 1..1. It has a directed association labeled 'typeResult' pointing to it. Inside the 'typeResult' block, there are three children: 'responseCode' (multiplicity 0..1), 'sourceSystem' (multiplicity 0..1), and 'result' (multiplicity 0..1). A callout box indicates: 'Common result values used in every response and optional specific subsystem result codes.'</p>
Type	typeResult
Properties	content: complex
Model	responseCode , sourceSystem{0,1} , result{0,1}
Children	responseCode, result, sourceSystem
Instance	<pre><result xmlns="DR-GW"> <responseCode>{1,1}</responseCode> <sourceSystem>{0,1}</sourceSystem> <result>{0,1}</result> </result></pre>
Source	<pre><xs:element name="result" type="typeResult" /></pre>

Element interfaceCall / selectEvent

Namespace	DR-GW
-----------	-------

Diagram	<pre> classDiagram class typeEvent { <<extension base>> requestId *--> selectEvent result *--> target } selectEvent --> typeEvent target --> typeEvent note over typeEvent: The event informs about the actual state of the selection requested before using the "select" request. </pre>
Type	typeCallSelectEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeCallSelectEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , target
Children	requestId, result, target
Instance	<pre> <selectEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <target>{1,1}</target> </selectEvent> </pre>
Source	<xss:element name="selectEvent" type="typeCallSelectEvent"/>

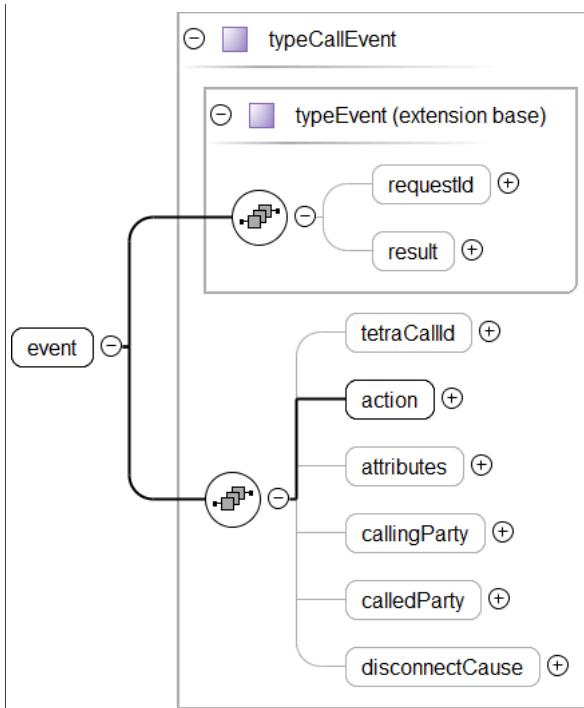
Element typeCallSelectEvent / target

Namespace	DR-GW
Diagram	<pre> classDiagram class typeSelection { level *--> target target *--> typeSelection } </pre>
Type	typeSelection
Properties	content: complex
Model	level , target
Children	level, target
Instance	<pre> <target xmlns="DR-GW"> <level>{1,1}</level> <target>{1,1}</target> </target> </pre>
Source	<xss:element name="target" type="typeSelection"/>

Element interfaceCall / event

Namespace	DR-GW
-----------	-------

Diagram



Type	typeCallEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeCallEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , tetraCallId{0,1} , action , attributes{0,1} , callingParty{0,1} , calledParty{0,1} , disconnectCause{0,1}
Children	action, attributes, calledParty, callingParty, disconnectCause, requestId, result, tetraCallId
Instance	<pre> <event xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <tetraCallId>{0,1}</tetraCallId> <action>{1,1}</action> <attributes>{0,1}</attributes> <callingParty>{0,1}</callingParty> <calledParty>{0,1}</calledParty> <disconnectCause>{0,1}</disconnectCause> </event> </pre>
Source	<code><xss:element name="event" type="typeCallEvent" /></code>

Element interfaceCall / pttEvent

Namespace	DR-GW
-----------	-------

Diagram	<pre> classDiagram typeEvent < -- typeCallPTTEvent typeCallPTTEvent { requestId? result? tetraCallId? granted? ceased? wait? } </pre> <p>The diagram shows the UML class structure for <code>typeCallPTTEvent</code>. It is an extension of <code>typeEvent</code>. The class contains attributes: <code>requestId</code> (optional), <code>result</code> (optional), <code>tetraCallId</code> (optional), <code>granted</code> (optional), <code>ceased</code> (optional), and <code>wait</code> (optional). There are also three callouts providing descriptions for the <code>ceased</code>, <code>wait</code>, and <code>tetraCallId</code> attributes.</p>
Type	<code>typeCallPTTEvent</code>
Type hierarchy	<ul style="list-style-type: none"> <code>typeEvent</code> <code>typeCallPTTEvent</code>
Properties	content: complex
Model	<code>requestId{0,1}</code> , <code>result{0,1}</code> , <code>tetraCallId{0,1}</code> , (<code>granted</code> <code>ceased</code> <code>wait</code>)
Children	<code>ceased</code> , <code>granted</code> , <code>requestId</code> , <code>result</code> , <code>tetraCallId</code> , <code>wait</code>
Instance	<pre> <pttEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <tetraCallId>{0,1}</tetraCallId> <granted>{1,1}</granted> <ceased>{1,1}</ceased> <wait>{1,1}</wait> </pttEvent> </pre>
Source	<code><xss:element name="pttEvent" type="typeCallPTTEvent" /></code>

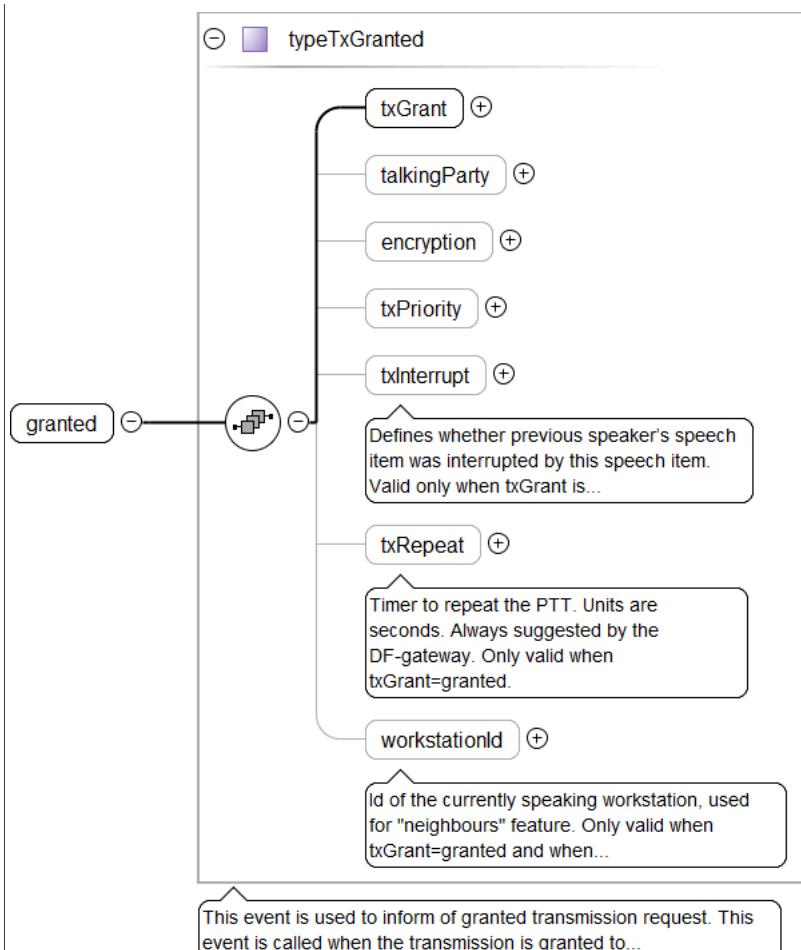
Element `typeCallPTTEvent` / `tetraCallId`

Namespace	DR-GW				
Diagram	<p>The diagram shows the UML class structure for <code>tetraCallId</code>. It extends <code>typeTetraCallId</code>. A callout provides a note about the TETRA callId.</p>				
Type	<code>typeTetraCallId</code>				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="tetraCallId" type="typeTetraCallId" minOccurs="0" /></code>				

Element `typeCallPTTEvent` / `granted`

Namespace	DR-GW
-----------	-------

Diagram



Type	typeTxGranted
Properties	content: complex
Model	txGrant , talkingParty{0,1} , encryption{0,1} , txPriority{0,1} , txInterrupt{0,1} , txRepeat{0,1} , workstationId{0,1}
Children	encryption, talkingParty, txGrant, txInterrupt, txPriority, txRepeat, workstationId
Instance	<pre> <granted xmlns="DR-GW"> <txGrant>{1,1}</txGrant> <talkingParty>{0,1}</talkingParty> <encryption>{0,1}</encryption> <txPriority>{0,1}</txPriority> <txInterrupt>{0,1}</txInterrupt> <txRepeat>{0,1}</txRepeat> <workstationId>{0,1}</workstationId> </granted></pre>
Source	<xss:element name="granted" type="typeTxGranted" />

Element typeTxGranted / txGrant

Namespace	DR-GW								
Diagram	<p>txGrant — points to typeTxGrant.</p> <p>Defines to whom speech item was granted.</p>								
Type	typeTxGrant								
Properties	content: simple								
Facets	<table border="1"> <tbody> <tr> <td>enumeration</td> <td>granted</td> </tr> <tr> <td>enumeration</td> <td>notGranted</td> </tr> <tr> <td>enumeration</td> <td>queued</td> </tr> <tr> <td>enumeration</td> <td>granted2another</td> </tr> </tbody> </table>	enumeration	granted	enumeration	notGranted	enumeration	queued	enumeration	granted2another
enumeration	granted								
enumeration	notGranted								
enumeration	queued								
enumeration	granted2another								

Source	<code><xss:element name="txGrant" type="typeTxGrant" /></code>
--------	--

Element typeTxGranted / talkingParty

Namespace	DR-GW				
Diagram	<pre> classDiagram typeAddress { subscriber alias msisdn fssn external opta cell } talkingParty { <<dr-gw>> <<subscriber>> <<alias>> <<msisdn>> <<fssn>> <<external>> <<opta>> <<cell>> } typeAddress "0..1" o-- "1..1" talkingParty </pre> <p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>				
Type	typeAddress				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}				
Children	alias, cell, external, fssn, msisdn, opta, subscriber				
Instance	<pre> <talkingParty xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </talkingParty> </pre>				
Source	<code><xss:element name="talkingParty" type="typeAddress" minOccurs="0" /></code>				

Element typeTxGranted / encryption

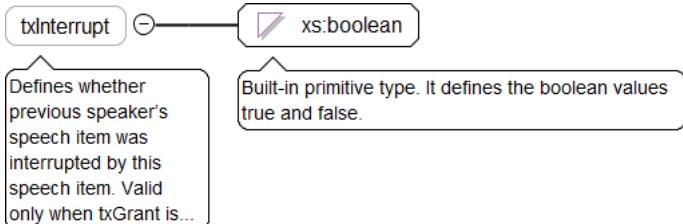
Namespace	DR-GW						
Diagram	<pre> classDiagram xs:boolean encryption { <<dr-gw>> <<xs:boolean>> } encryption "0..1" o-- "1..1" xs:boolean </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>						
Type	xs:boolean						
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>default:</td> <td>true</td> </tr> </table>	content:	simple	minOccurs:	0	default:	true
content:	simple						
minOccurs:	0						
default:	true						
Source	<code><xss:element name="encryption" type="xs:boolean" default="true" minOccurs="0" /></code>						

Element typeTxGranted / txPriority

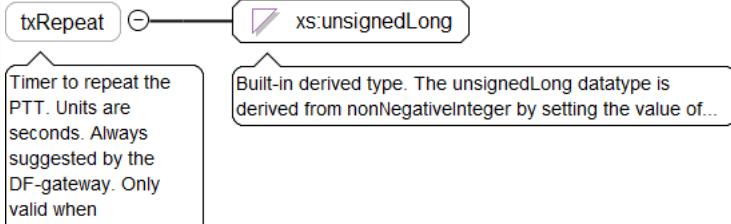
Namespace	DR-GW
-----------	-------

Diagram							
Type	typeTxPriority						
Properties	<table border="1"> <tr><td>content:</td><td>simple</td></tr> <tr><td>minOccurs:</td><td>0</td></tr> <tr><td>default:</td><td>normal</td></tr> </table>	content:	simple	minOccurs:	0	default:	normal
content:	simple						
minOccurs:	0						
default:	normal						
Facets	<table border="1"> <tr><td>enumeration</td><td>normal</td></tr> <tr><td>enumeration</td><td>emergency</td></tr> </table>	enumeration	normal	enumeration	emergency		
enumeration	normal						
enumeration	emergency						
Source	<code><xss:element name="txPriority" type="typeTxPriority" minOccurs="0" default="normal"/></code>						

Element typeTxGranted / txInterrupt

Namespace	DR-GW						
Annotations	Defines whether previous speaker's speech item was interrupted by this speech item. Valid only when txGrant is granted2another.						
Diagram							
Type	xs:boolean						
Properties	<table border="1"> <tr><td>content:</td><td>simple</td></tr> <tr><td>minOccurs:</td><td>0</td></tr> <tr><td>default:</td><td>false</td></tr> </table>	content:	simple	minOccurs:	0	default:	false
content:	simple						
minOccurs:	0						
default:	false						
Source	<pre><xss:element name="txInterrupt" type="xs:boolean" default="false" minOccurs="0"> <xss:annotation> <xss:documentation>Defines whether previous speaker's speech item was interrupted by this speech item. Valid only when txGrant is granted2another.</xss:documentation> </xss:annotation> </xss:element></pre>						

Element typeTxGranted / txRepeat

Namespace	DR-GW						
Annotations	Timer to repeat the PTT. Units are seconds. Always suggested by the DF-gateway. Only valid when txGrant=granted.						
Diagram							
Type	xs:unsignedLong						
Properties	<table border="1"> <tr><td>content:</td><td>simple</td></tr> <tr><td>minOccurs:</td><td>0</td></tr> <tr><td>default:</td><td>0</td></tr> </table>	content:	simple	minOccurs:	0	default:	0
content:	simple						
minOccurs:	0						
default:	0						
Source	<pre><xss:element name="txRepeat" type="xs:unsignedLong" minOccurs="0" default="0"> <xss:annotation> <xss:documentation>Timer to repeat the PTT. Units are seconds. Always suggested by the DF-gateway. Only valid when txGrant=granted.</xss:documentation> </xss:annotation> </xss:element></pre>						

```

</xs:annotation>
</xs:element>

```

Element typeTxGranted / workstationId

Namespace	DR-GW				
Annotations	<p>Id of the currently speaking workstation, used for "neighbours" feature. Only valid when txGrant=granted and when supplied by the DF-client in PTT request.</p>				
Diagram	<pre> classDiagram workstationId "1..>" xs:normalizedString xs:normalizedString < -- "1..>" normalizedString normalizedString < -- "1..>" string string < -- "1..>" whitespace whitespace < -- "1..>" normalizedString </pre> <p>workstationId → xs:normalizedString xs:normalizedString → normalizedString normalizedString → string string → whitespace whitespace → normalizedString</p> <p>Annotations: Id of the currently speaking workstation, used for "neighbours" feature. Only valid when txGrant=granted and when... Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>				
Type	xs:normalizedString				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<pre> <xs:element name="workstationId" type="xs:normalizedString" minOccurs="0"> <xs:annotation> <xs:documentation>Id of the currently speaking workstation, used for "neighbours" feature. Only valid when txGrant=granted and when supplied by the DF-client in PTT request.</xs:documentation> </xs:annotation> </xs:element> </pre>				

Element typeCallPTTEvent / ceased

Namespace	DR-GW		
Annotations	This event is used to inform that transmission is ceased and nobody has the speech item.		
Diagram	<pre> classDiagram ceased "1..>" typeEmpty typeEmpty < -- "1..>" emptyType emptyType < -- "1..>" anyType anyType < -- "1..>" simpleType simpleType < -- "1..>" complexType complexType < -- "1..>" restriction restriction < -- "1..>" typeEmpty </pre> <p>ceased → typeEmpty typeEmpty → emptyType emptyType → anyType anyType → simpleType simpleType → complexType complexType → restriction restriction → typeEmpty</p> <p>Annotations: This event is used to inform that transmission is ceased and nobody has the speech item. Explicit type specification for elements that shall be empty.</p>		
Type	typeEmpty		
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> </table>	content:	complex
content:	complex		
Source	<pre> <xs:element name="ceased" type="typeEmpty"> <xs:annotation> <xs:documentation>This event is used to inform that transmission is ceased and nobody has the speech item.</xs:documentation> </xs:annotation> </xs:element> </pre>		

Element typeCallPTTEvent / wait

Namespace	DR-GW
Annotations	This event is used to inform that the call is temporarily paused e.g. if radio subscriber has roamed to a new cell and there are no free resources available.

Diagram	
Type	typeEmpty
Properties	content: complex
Source	<pre><xss:element name="wait" type="typeEmpty"> <xss:annotation> <xss:documentation>This event is used to inform that the call is temporarily paused e.g. if radio subscriber has roamed to a new cell and there are no free resources available.</xss:documentation> </xss:annotation> </xss:element></pre>

Element interfaceCall / unitInEmergencyEvent

Namespace	DR-GW
Diagram	
Type	typeCallUnitInEmergencyEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeCallUnitInEmergencyEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , tetraCallId{0,1} , unitInEmg , unitInEmgType , emgInfo , tstamp
Children	emgInfo, group, requestId, result, tetraCallId, tstamp, unitInEmg, unitInEmgType
Instance	<pre><unitInEmergencyEvent xmlns="DR-GW"> <requestId>{0,1}</requestId></pre>

	<pre> <result>{0,1}</result> <group>{1,1}</group> <tetraCallId>{0,1}</tetraCallId> <unitInEmg>{1,1}</unitInEmg> <unitInEmgType>{1,1}</unitInEmgType> <emgInfo>{1,1}</emgInfo> <tstamp>{1,1}</tstamp> </unitInEmergencyEvent> </pre>
Source	<code><xs:element name="unitInEmergencyEvent" type="typeCallUnitInEmergencyEvent" /></code>

Element typeCallUnitInEmergencyEvent / group

Namespace	DR-GW
Diagram	<pre> classDiagram class group class typeSubscriberAddress class ssi class tsi group --> typeSubscriberAddress typeSubscriberAddress --> ssi typeSubscriberAddress --> tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group> </pre>
Source	<code><xs:element name="group" type="typeSubscriberAddress" /></code>

Element typeCallUnitInEmergencyEvent / tetraCallId

Namespace	DR-GW				
Diagram	<pre> classDiagram class tetraCallId class typeTetraCallId tetraCallId --> typeTetraCallId </pre> <p>TETRA callId, generated by the TCS. In the TCS-API it is defined as long, but Airbus doesn't have it under control if...</p>				
Type	typeTetraCallId				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xs:element name="tetraCallId" type="typeTetraCallId" minOccurs="0" /></code>				

Element typeCallUnitInEmergencyEvent / unitInEmg

Namespace	DR-GW
-----------	-------

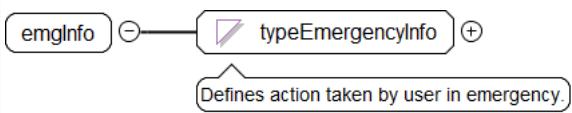
Diagram	<pre> classDiagram class typeAddress { subscriber *0..1 alias *0..1 msisdn *0..1 fssn *0..1 external *0..1 opta *0..1 cell *0..1 } unitInEmg *0..1 --> typeAddress note over typeAddress: Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA). </pre>
Type	typeAddress
Properties	content: complex
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre> <unitInEmg xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </unitInEmg> </pre>
Source	<xss:element name="unitInEmg" type="typeAddress" />

Element typeCallUnitInEmergencyEvent / unitInEmgType

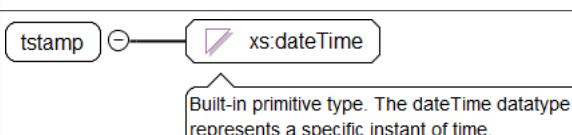
Namespace	DR-GW										
Diagram	<pre> classDiagram class unitInEmgType class typeUnitInEmergencyType unitInEmgType --> typeUnitInEmergencyType note over typeUnitInEmergencyType: Defines type of the subscriber. Refer to type tcsCallSubscriberType_t of the TCS-API. </pre>										
Type	typeUnitInEmergencyType										
Properties	content: simple										
Facets	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">enumeration</td> <td style="padding: 2px;">dummy</td> </tr> <tr> <td style="padding: 2px;">enumeration</td> <td style="padding: 2px;">ms</td> </tr> <tr> <td style="padding: 2px;">enumeration</td> <td style="padding: 2px;">g4wif</td> </tr> <tr> <td style="padding: 2px;">enumeration</td> <td style="padding: 2px;">external</td> </tr> <tr> <td style="padding: 2px;">enumeration</td> <td style="padding: 2px;">ws</td> </tr> </table>	enumeration	dummy	enumeration	ms	enumeration	g4wif	enumeration	external	enumeration	ws
enumeration	dummy										
enumeration	ms										
enumeration	g4wif										
enumeration	external										
enumeration	ws										
Source	<xss:element name="unitInEmgType" type="typeUnitInEmergencyType" />										

Element typeCallUnitInEmergencyEvent / emgInfo

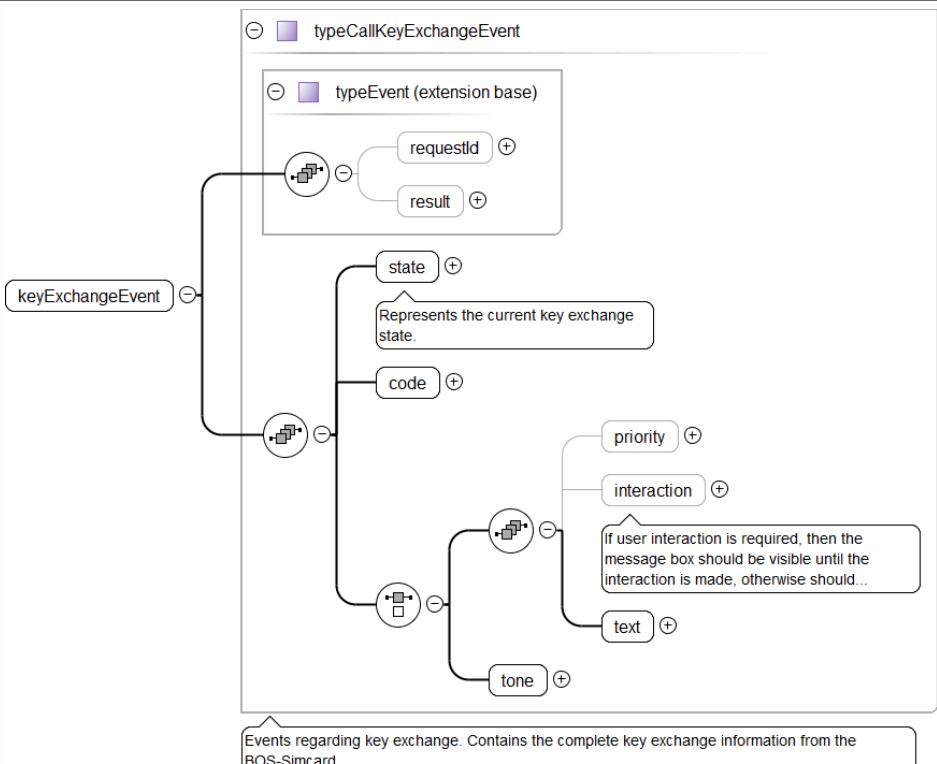
Namespace	DR-GW
-----------	-------

Diagram	 Defines action taken by user in emergency.
Type	typeEmergencyInfo
Properties	content: simple
Facets	enumeration addTx enumeration add enumeration ceased enumeration demandTx enumeration removed enumeration emergencyCallDisconnected
Source	<xs:element name="emgInfo" type="typeEmergencyInfo"/>

Element typeCallUnitInEmergencyEvent / tstamp

Namespace	DR-GW
Diagram	 Built-in primitive type. The dateTime datatype represents a specific instant of time.
Type	xs:dateTime
Properties	content: simple
Source	<xs:element name="tstamp" type="xs:dateTime"/>

Element interfaceCall / keyExchangeEvent

Namespace	DR-GW
Diagram	 Events regarding key exchange. Contains the complete key exchange information from the BOS-Simcard.
Type	typeCallKeyExchangeEvent
Type hierarchy	• typeEvent

	<ul style="list-style-type: none"> • typeCallKeyExchangeEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , state , code , ((priority{0,1} , interaction{0,1} , text) tone)
Children	code, interaction, priority, requestId, result, state, text, tone
Instance	<pre><keyExchangeEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <state>{1,1}</state> <code>{1,1}</code> <priority>{0,1}</priority> <interaction>{0,1}</interaction> <text>{1,1}</text> <tone>{1,1}</tone> </keyExchangeEvent></pre>
Source	<code><xss:element name="keyExchangeEvent" type="typeCallKeyExchangeEvent" /></code>

Element typeCallKeyExchangeEvent / state

Namespace	DR-GW									
Annotations	Represents the current key exchange state.									
Diagram	<pre> classDiagram state "–" typeKeyExchangeState state --> stateNote typeKeyExchangeState --> stateNote stateNote : Represents the current key exchange state. </pre> <p>The diagram shows a class named 'state' connected to another class named 'typeKeyExchangeState' via a directed association. A callout box labeled 'Represents the current key exchange state.' points to the 'state' class.</p>									
Type	typeKeyExchangeState									
Properties	content: simple									
Facets	<table> <tr> <td>enumeration</td> <td>keyValid</td> <td>current key is valid, no user action required.</td> </tr> <tr> <td>enumeration</td> <td>keyInvalid</td> <td>Key invalid, user must request key exchange.</td> </tr> <tr> <td>enumeration</td> <td>keyExchangeInProgress</td> <td>Key exchange in progress, user may abort exchange or wait until it gets finished.</td> </tr> </table>	enumeration	keyValid	current key is valid, no user action required.	enumeration	keyInvalid	Key invalid, user must request key exchange.	enumeration	keyExchangeInProgress	Key exchange in progress, user may abort exchange or wait until it gets finished.
enumeration	keyValid	current key is valid, no user action required.								
enumeration	keyInvalid	Key invalid, user must request key exchange.								
enumeration	keyExchangeInProgress	Key exchange in progress, user may abort exchange or wait until it gets finished.								
Source	<pre><xss:element name="state" type="typeKeyExchangeState"> <xss:annotation> <xss:documentation>Represents the current key exchange state.</xss:documentation> </xss:annotation> </xss:element></pre>									

Element typeCallKeyExchangeEvent / code

Namespace	DR-GW
Diagram	<pre> classDiagram code "–" typeKeyExchangeCode code --> codeNote typeKeyExchangeCode --> codeNote codeNote : See "Table 5.3: Status words of the commands" of the E-to-E Encryption SIM-ME Interface (Version 4.0.5) for all... </pre> <p>The diagram shows a class named 'code' connected to another class named 'typeKeyExchangeCode' via a directed association. A callout box labeled 'See "Table 5.3: Status words of the commands" of the E-to-E Encryption SIM-ME Interface (Version 4.0.5) for all...' points to the 'code' class.</p>
Type	typeKeyExchangeCode
Properties	content: simple
Facets	length 2
Source	<code><xss:element name="code" type="typeKeyExchangeCode" /></code>

Element typeCallKeyExchangeEvent / priority

Namespace	DR-GW
Diagram	<pre> classDiagram priority "–" typeKeyExchangeTextPriority priority --> priorityNote typeKeyExchangeTextPriority --> priorityNote priorityNote : Defines the priority of the KeyExchange information. </pre> <p>The diagram shows a class named 'priority' connected to another class named 'typeKeyExchangeTextPriority' via a directed association. A callout box labeled 'Defines the priority of the KeyExchange information.' points to the 'priority' class.</p>
Type	typeKeyExchangeTextPriority

Properties	content: simple minOccurs: 0 default: normal
Facets	enumeration normal enumeration high
Source	<code><xs:element name="priority" type="typeKeyExchangeTextPriority" minOccurs="0" default="normal" /></code>

Element typeCallKeyExchangeEvent / interaction

Namespace	DR-GW
Annotations	If user interaction is required, then the message box should be visible until the interaction is made, otherwise should be hidden after delay.
Diagram	<pre> graph LR interaction[interaction] -- "-" --> xsBoolean[xs:boolean] subgraph Annotation [] direction TB A1[If user interaction is required, then the message box should be visible until the interaction is made, otherwise should...] A2[Built-in primitive type. It defines the boolean values true and false.] A1 --- A2 end </pre>
Type	xs:boolean
Properties	content: simple minOccurs: 0 default: false
Source	<code><xs:element name="interaction" type="xs:boolean" minOccurs="0" default="false"></code> <code> <xs:annotation></code> <code> <xs:documentation>If user interaction is required, then the message box should be visible until the interaction is made, otherwise should be hidden after delay.</xs:documentation></code> <code> </xs:annotation></code> <code></xs:element></code>

Element typeCallKeyExchangeEvent / text

Namespace	DR-GW
Diagram	<pre> graph LR text[text] -- "-" --> typeKeyExchangeText[typeKeyExchangeText] subgraph Annotation [] direction TB A1[The textual information supplied by the BOS-simcard and sent from the DF-Gateway to the DF-client.] end </pre>
Type	typeKeyExchangeText
Properties	content: simple
Facets	maxLength 100
Source	<code><xs:element name="text" type="typeKeyExchangeText" /></code>

Element typeCallKeyExchangeEvent / tone

Namespace	DR-GW
Diagram	<pre> graph LR tone[tone] -- "-" --> xsBoolean[xs:boolean] subgraph Annotation [] direction TB A1[Built-in primitive type. It defines the boolean values true and false.] end </pre>
Type	xs:boolean
Properties	content: simple fixed: true
Source	<code><xs:element name="tone" type="xs:boolean" fixed="true" /></code>

Element drgw / session

Namespace	DR-GW
Diagram	<pre> classDiagram class session { <<Type interfaceSession>> } class interfaceSession { login logout supervise check response loginEvent superviseEvent } session < -- interfaceSession </pre> <p>DR-GW-Session. In order to use the rest of DR-GW interface the DF-Client must establish a DR-GW session and maintain it...</p>
Type	interfaceSession
Properties	content: complex
Model	login logout supervise check response loginEvent superviseEvent
Children	check, login, loginEvent, logout, response, supervise, superviseEvent
Instance	<pre> <session xmlns="DR-GW"> <login>{1,1}</login> <logout>{1,1}</logout> <supervise>{1,1}</supervise> <check>{1,1}</check> <response>{1,1}</response> <loginEvent>{1,1}</loginEvent> <superviseEvent>{1,1}</superviseEvent> </session> </pre>
Source	<xs:element name="session" type="interfaceSession" />

Element interfaceSession / login

Namespace	DR-GW
Diagram	<pre> classDiagram class login class typeSessionLogin { <<typeRequest (extension base)>> } class typeRequest login < -- typeSessionLogin </pre> <p>Login procedure. The username, password and the complete authentication is done using mechanisms of the transport...</p>

Type	typeSessionLogin
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSessionLogin
Properties	content: complex
Model	requestId , clientid , supervise{0,1} , version{0,1}
Children	clientid, requestId, supervise, version
Instance	<pre><login xmlns="DR-GW"> <requestId>{1,1}</requestId> <clientid>{1,1}</clientid> <supervise>{0,1}</supervise> <version>{0,1}</version> </login></pre>
Source	<code><xss:element name="login" type="typeSessionLogin"/></code>

Element typeSessionLogin / clientid

Namespace	DR-GW
Diagram	<p>A UML class diagram fragment showing a dependency relationship between two elements. On the left, there is a rounded rectangle labeled "clientid". A line with a hollow circle at the start (representing a dependency) connects to another rounded rectangle on the right labeled "xs:string". Below this diagram, a callout box contains the text: "Built-in primitive type. The string datatype represents character strings in XML."</p>
Type	xs:string
Properties	content: simple
Source	<code><xss:element name="clientid" type="xs:string"/></code>

Element typeSessionLogin / supervise

Namespace	DR-GW						
Diagram	<p>A UML class diagram fragment showing a dependency relationship between two elements. On the left, there is a rounded rectangle labeled "supervise". A line with a hollow circle at the start (representing a dependency) connects to another rounded rectangle on the right labeled "typeSuperviseTimeout". Above the "typeSuperviseTimeout" box is a plus sign (+), indicating zero or one occurrence. Below this diagram, a callout box contains the text: "Accepted supervise timeout values."</p>						
Type	typeSuperviseTimeout						
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>default:</td> <td>60</td> </tr> </table>	content:	simple	minOccurs:	0	default:	60
content:	simple						
minOccurs:	0						
default:	60						
Facets	<table border="1"> <tr> <td>enumeration</td> <td>20</td> </tr> <tr> <td>enumeration</td> <td>30</td> </tr> <tr> <td>enumeration</td> <td>60</td> </tr> </table>	enumeration	20	enumeration	30	enumeration	60
enumeration	20						
enumeration	30						
enumeration	60						
Source	<code><xss:element name="supervise" type="typeSuperviseTimeout" default="60" minOccurs="0"/></code>						

Element typeSessionLogin / version

Namespace	DR-GW				
Diagram	<p>A UML class diagram fragment showing a dependency relationship between two elements. On the left, there is a rounded rectangle labeled "version". A line with a hollow circle at the start (representing a dependency) connects to another rounded rectangle on the right labeled "xs:string". Below this diagram, a callout box contains the text: "Built-in primitive type. The string datatype represents character strings in XML."</p>				
Type	xs:string				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="version" type="xs:string" minOccurs="0"/></code>				

Element interfaceSession / logout

Namespace	DR-GW
Diagram	<pre> classDiagram typeSessionLogout < -- typeRequest {extension base} typeSessionLogout "0..1" --> "0..1" logout : logout logout --> requestId : requestId </pre>
Type	typeSessionLogout
Type hierarchy	<ul style="list-style-type: none"> • typeRequest <ul style="list-style-type: none"> • typeSessionLogout
Properties	content: complex
Model	requestId
Children	requestId
Instance	<pre> <logout xmlns="DR-GW"> <requestId>{1,1}</requestId> </logout> </pre>
Source	<pre> <xss:element name="logout" type="typeSessionLogout" /> </pre>

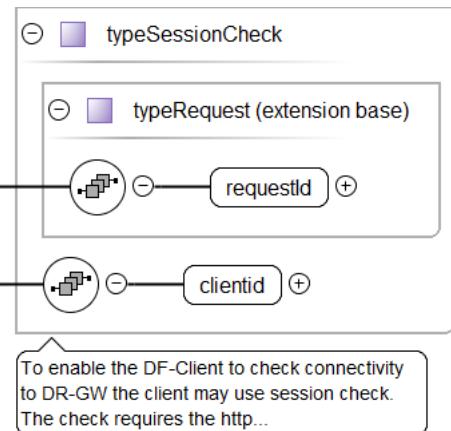
Element interfaceSession / supervise

Namespace	DR-GW
Diagram	<pre> classDiagram typeSessionSupervise < -- typeRequest {extension base} typeSessionSupervise "0..1" --> "0..1" supervise : supervise supervise --> requestId : requestId </pre>
Type	typeSessionSupervise
Type hierarchy	<ul style="list-style-type: none"> • typeRequest <ul style="list-style-type: none"> • typeSessionSupervise
Properties	content: complex
Model	requestId
Children	requestId
Instance	<pre> <supervise xmlns="DR-GW"> <requestId>{1,1}</requestId> </supervise> </pre>
Source	<pre> <xss:element name="supervise" type="typeSessionSupervise" /> </pre>

Element interfaceSession / check

Namespace	DR-GW
-----------	-------

Diagram



Type	typeSessionCheck
------	------------------

Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSessionCheck
----------------	---

Properties	content: complex
------------	------------------

Model	requestId , clientid
-------	----------------------

Children	clientid, requestId
----------	---------------------

Instance	<pre><check xmlns="DR-GW"> <requestId>{1,1}</requestId> <clientid>{1,1}</clientid> </check></pre>
----------	---

Source	<code><xs:element name="check" type="typeSessionCheck" /></code>
--------	--

Element typeSessionCheck / clientid

Namespace	DR-GW
-----------	-------

Diagram	<p>Built-in primitive type. The string datatype represents character strings in XML.</p>
---------	--

Type	xs:string
------	-----------

Properties	content: simple
------------	-----------------

Source	<code><xs:element name="clientid" type="xs:string" /></code>
--------	--

Element interfaceSession / response

Namespace	DR-GW
-----------	-------

Diagram	
---------	--

Type	typeResponse
------	--------------

Properties	content: complex
------------	------------------

Model	requestId , result
-------	--------------------

Children	requestId, result
----------	-------------------

Instance	<pre><response xmlns="DR-GW"> <requestId>{1,1}</requestId></pre>
----------	--

	<pre> <result>{1,1}</result> </response> </pre>
Source	<xs:element name="response" type="typeResponse" />

Element interfaceSession / loginEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeSessionLoginEvent < -- typeEvent typeSessionLoginEvent < -- loginEvent typeSessionLoginEvent < -- issi typeSessionLoginEvent < -- requestId typeSessionLoginEvent < -- result </pre>
Type	typeSessionLoginEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSessionLoginEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , issi{0,1}
Children	issi, requestId, result
Instance	<pre> <loginEvent xmlns="DR-GW"> <requestIds>{0,1}</requestIds> <result>{0,1}</result> <issi>{0,1}</issi> </loginEvent> </pre>
Source	<xs:element name="loginEvent" type="typeSessionLoginEvent" />

Element typeSessionLoginEvent / issi

Namespace	DR-GW				
Diagram	<pre> issi < -- xs:string </pre> <p>Built-in primitive type. The string datatype represents character strings in XML.</p>				
Type	xs:string				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xs:element name="issi" type="xs:string" minOccurs="0" />				

Element interfaceSession / superviseEvent

Namespace	DR-GW
Diagram	<pre> superviseEvent < -- typeEvent </pre>

Type	typeSessionSuperviseEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSessionSuperviseEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1}
Children	requestId, result
Instance	<pre><superviseEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> </superviseEvent></pre>
Source	<code><xss:element name="superviseEvent" type="typeSessionSuperviseEvent" /></code>

Element drgw / sds

Namespace	DR-GW
Diagram	<pre> classDiagram class sds { <<Type>> <<interfaceSds>> } interfaceSds "1..1" -- "1..1" sds interfaceSds <<interfaceSds>> interfaceSds <<+send>> interfaceSds <<+sendReport>> interfaceSds <<+response>> interfaceSds <<+sendEvent>> interfaceSds <<+receiveEvent>> interfaceSds <<+reportEvent>> note over interfaceSds: DR-GW-Sds. Use to send/receive SDS messages. Use only via SOAP. </pre>
Type	interfaceSds
Properties	content: complex
Model	send sendReport response sendEvent receiveEvent reportEvent
Children	receiveEvent, reportEvent, response, send, sendEvent, sendReport
Instance	<pre><sds xmlns="DR-GW"> <send>{1,1}</send> <sendReport>{1,1}</sendReport> <response>{1,1}</response> <sendEvent>{1,1}</sendEvent> <receiveEvent>{1,1}</receiveEvent> <reportEvent>{1,1}</reportEvent> </sds></pre>
Source	<code><xss:element name="sds" type="interfaceSds" /></code>

Element interfaceSds / send

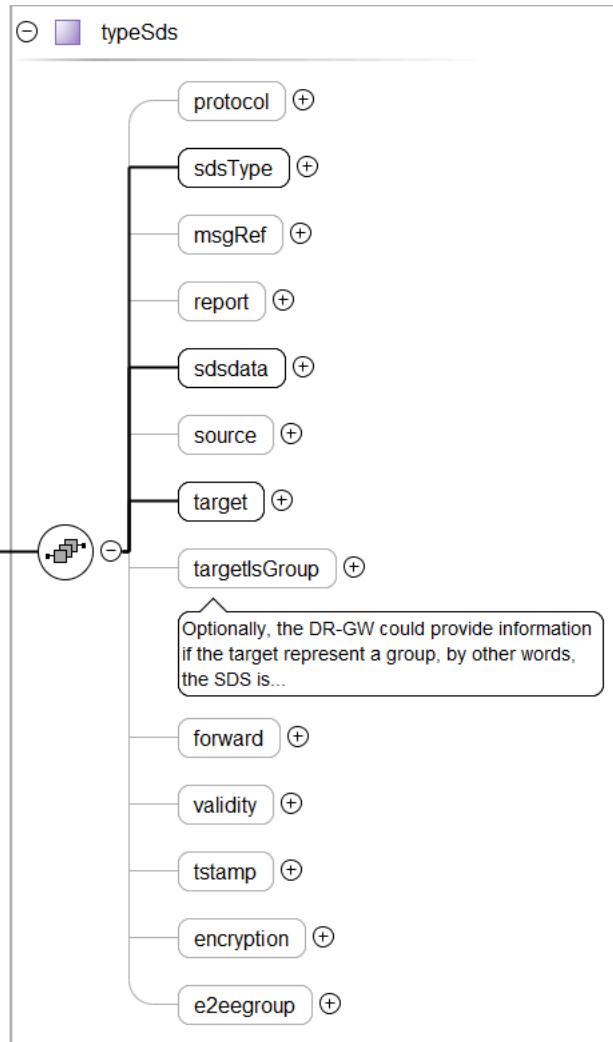
Namespace	DR-GW
-----------	-------

Diagram	<p>This type is used to send SDS message.</p>
Type	typeSdsSend
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSdsSend
Properties	content: complex
Model	requestId , sds
Children	requestId, sds
Instance	<pre><send xmlns="DR-GW"> <requestId>{1,1}</requestId> <sds>{1,1}</sds> </send></pre>
Source	<code><xss:element name="send" type="typeSdsSend" /></code>

Element typeSdsSend / sds

Namespace	DR-GW
-----------	-------

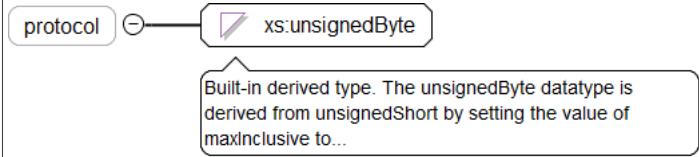
Diagram



Type	<code>typeSds</code>
Properties	content: complex
Model	<code>protocol{0,1}</code> , <code>sdsType</code> , <code>msgRef{0,1}</code> , <code>report{0,1}</code> , <code>sdsdata</code> , <code>source{0,1}</code> , <code>target</code> , <code>targetIsGroup{0,1}</code> , <code>forward{0,1}</code> , <code>validity{0,1}</code> , <code>tstamp{0,1}</code> , <code>encryption{0,1}</code> , <code>e2eegroup{0,1}</code>
Children	<code>e2eegroup</code> , <code>encryption</code> , <code>forward</code> , <code>msgRef</code> , <code>protocol</code> , <code>report</code> , <code>sdsType</code> , <code>sdsdata</code> , <code>source</code> , <code>target</code> , <code>targetIsGroup</code> , <code>tstamp</code> , <code>validity</code>
Instance	<pre> <sds xmlns="DR-GW"> <protocol>{0,1}</protocol> <sdsType>{1,1}</sdsType> <msgRef>{0,1}</msgRef> <report>{0,1}</report> <sdsdata>{1,1}</sdsdata> <source>{0,1}</source> <target>{1,1}</target> <targetIsGroup>{0,1}</targetIsGroup> <forward>{0,1}</forward> <validity>{0,1}</validity> <tstamp>{0,1}</tstamp> <encryption>{0,1}</encryption> <e2eegroup>{0,1}</e2eegroup> </sds> </pre>
Source	<code><xss:element name="sds" type="typeSds" /></code>

Element `typeSds` / `protocol`

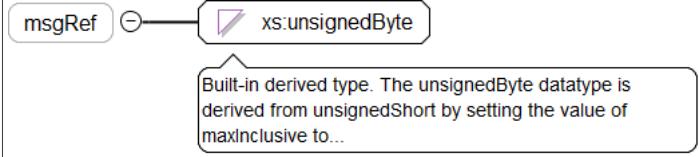
Namespace	DR-GW
-----------	-------

Diagram					
Type	xs:unsignedByte				
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="protocol" type="xs:unsignedByte" minOccurs="0"/></code>				

Element typeSds / sdsType

Namespace	DR-GW																		
Diagram																			
Type	typeSdsType																		
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> </table>	content:	simple																
content:	simple																		
Facets	<table> <tr> <td>enumeration</td> <td>0</td> <td>SDS1.</td> </tr> <tr> <td>enumeration</td> <td>1</td> <td>SDS2.</td> </tr> <tr> <td>enumeration</td> <td>2</td> <td>SDS3.</td> </tr> <tr> <td>enumeration</td> <td>3</td> <td>SDS4.</td> </tr> <tr> <td>enumeration</td> <td>4</td> <td>SDS-TL.</td> </tr> <tr> <td>enumeration</td> <td>5</td> <td>Status.</td> </tr> </table>	enumeration	0	SDS1.	enumeration	1	SDS2.	enumeration	2	SDS3.	enumeration	3	SDS4.	enumeration	4	SDS-TL.	enumeration	5	Status.
enumeration	0	SDS1.																	
enumeration	1	SDS2.																	
enumeration	2	SDS3.																	
enumeration	3	SDS4.																	
enumeration	4	SDS-TL.																	
enumeration	5	Status.																	
Source	<code><xss:element name="sdsType" type="typeSdsType"/></code>																		

Element typeSds / msgRef

Namespace	DR-GW				
Diagram					
Type	xs:unsignedByte				
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="msgRef" type="xs:unsignedByte" minOccurs="0"/></code>				

Element typeSds / report

Namespace	DR-GW								
Diagram									
Type	typeReport								
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>default:</td> <td>none</td> </tr> </table>	content:	simple	minOccurs:	0	default:	none		
content:	simple								
minOccurs:	0								
default:	none								
Facets	<table> <tr> <td>enumeration</td> <td>none</td> </tr> <tr> <td>enumeration</td> <td>delivery</td> </tr> <tr> <td>enumeration</td> <td>consume</td> </tr> <tr> <td>enumeration</td> <td>both</td> </tr> </table>	enumeration	none	enumeration	delivery	enumeration	consume	enumeration	both
enumeration	none								
enumeration	delivery								
enumeration	consume								
enumeration	both								

Source	<code><xss:element name="report" type="typeReport" default="none" minOccurs="0" /></code>
--------	---

Element typeSds / sdsdata

Namespace	DR-GW
Diagram	<pre> classDiagram class typeSdsData { data hexdata hexdatalength } class sdsdata { <<typeSdsData>> } sdsdata --> typeSdsData </pre> <p>2 ways of encoding the SDS. When sent from DF-Client to DF-Gateway at least one node must be present, otherwise it will...</p>
Type	typeSdsData
Properties	content: complex
Model	data{0,1} , hexdata{0,1} , hexdatalength{0,1}
Children	data, hexdata, hexdatalength
Instance	<pre> <sdsdata xmlns="DR-GW"> <data>{0,1}</data> <hexdata>{0,1}</hexdata> <hexdatalength>{0,1}</hexdatalength> </sdsdata> </pre>
Source	<code><xss:element name="sdsdata" type="typeSdsData" /></code>

Element typeSdsData / data

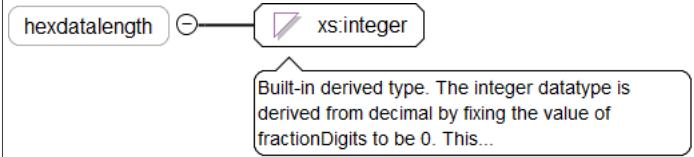
Namespace	DR-GW				
Diagram	<pre> classDiagram class typeSdsData { xs:string data } class xs:string typeSdsData --> xs:string </pre> <p>Built-in primitive type. The string datatype represents character strings in XML.</p>				
Type	xs:string				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="data" type="xs:string" minOccurs="0" /></code>				

Element typeSdsData / hexdata

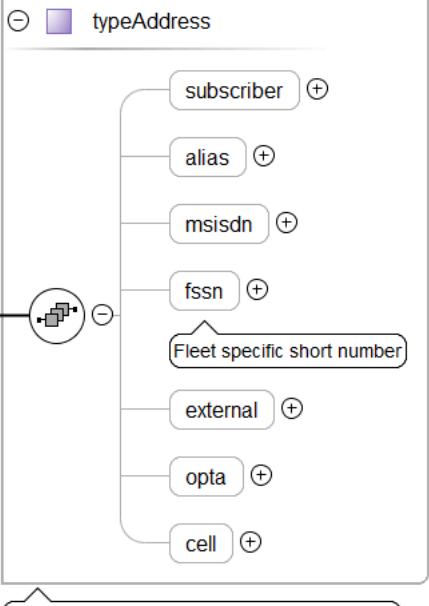
Namespace	DR-GW				
Diagram	<pre> classDiagram class typeSdsData { xs:hexBinary hexdata } class xs:hexBinary typeSdsData --> xs:hexBinary </pre> <p>Built-in primitive type. The hexBinary datatype represents arbitrary hex-encoded binary data.</p>				
Type	xs:hexBinary				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="hexdata" type="xs:hexBinary" minOccurs="0" /></code>				

Element typeSdsData / hexdatalength

Namespace	DR-GW
-----------	-------

Diagram	
Type	xs:integer
Properties	content: simple minOccurs: 0
Source	<code><xs:element name="hexdatalength" type="xs:integer" minOccurs="0"/></code>

Element typesDs / source

Namespace	DR-GW
Diagram	
Type	typeAddress
Properties	content: complex minOccurs: 0
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre><source xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </source></pre>
Source	<code><xs:element name="source" type="typeAddress" minOccurs="0"/></code>

Element typesDs / target

Namespace	DR-GW
-----------	-------

Diagram	<p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>
Type	typeAddress
Properties	content: complex
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre><target xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </target></pre>
Source	<code><xs:element name="target" type="typeAddress" /></code>

Element typeSds / targetIsGroup

Namespace	DR-GW				
Annotations	Optionally, the DR-GW could provide information if the target represent a group, by other words, the SDS is group-addressed. Also the Df-Client can provide this information to help DR-GW decide which resource to use for actual sending.				
Diagram					
Type	xs:boolean				
Properties	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">content:</td> <td style="padding: 2px;">simple</td> </tr> <tr> <td style="padding: 2px;">minOccurs:</td> <td style="padding: 2px;">0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<pre><xs:element name="targetIsGroup" type="xs:boolean" minOccurs="0"> <xs:annotation> <xs:documentation>Optionally, the DR-GW could provide information if the target represent a group, by other words, the SDS is group-addressed. Also the Df-Client can provide this information to help DR-GW decide which resource to use for actual sending.</xs:documentation> </xs:annotation> </xs:element></pre>				

Element typeSds / forward

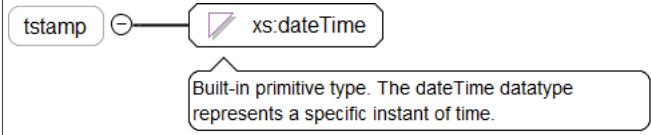
Namespace	DR-GW				
Diagram	<pre> classDiagram class typeAddress { subscriber * alias * msisdn * fssn * external * opta * cell * } forward --> typeAddress note over typeAddress: Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA). </pre>				
Type	typeAddress				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}				
Children	alias, cell, external, fssn, msisdn, opta, subscriber				
Instance	<pre> <forward xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </forward> </pre>				
Source	<code><xss:element name="forward" type="typeAddress" minOccurs="0" /></code>				

Element typeSds / validity

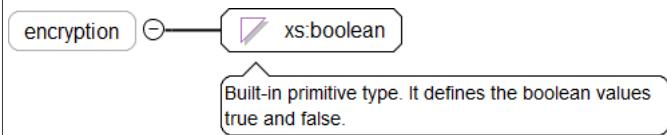
Namespace	DR-GW				
Diagram	<pre> classDiagram class validity { xs:unsignedByte } note over validity: Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to... </pre>				
Type	xs:unsignedByte				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="validity" type="xs:unsignedByte" minOccurs="0" /></code>				

Element typeSds / tstamp

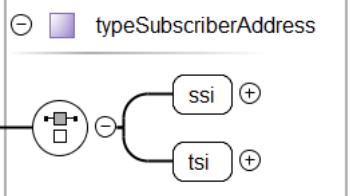
Namespace	DR-GW
-----------	-------

Diagram	
Type	xs:dateTime
Properties	<p>content: simple</p> <p>minOccurs: 0</p>
Source	<xs:element name="tstamp" type="xs:dateTime" minOccurs="0"/>

Element typeSds / encryption

Namespace	DR-GW
Diagram	
Type	xs:boolean
Properties	<p>content: simple</p> <p>minOccurs: 0</p> <p>default: true</p>
Source	<xs:element name="encryption" type="xs:boolean" default="true" minOccurs="0"/>

Element typeSds / e2eegroup

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	<p>content: complex</p> <p>minOccurs: 0</p>
Model	ssi tsi
Children	ssi, tsi
Instance	<e2eegroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </e2eegroup>
Source	<xs:element name="e2eegroup" type="typeSubscriberAddress" minOccurs="0"/>

Element interfaceSds / sendReport

Namespace	DR-GW
-----------	-------

Diagram	<pre> classDiagram typeSdsSendReport < -- typeRequest typeSdsSendReport { requestId target msgRef deliveryStatus } note over msgRef: An message reference is returned in the response for later message identification in case delivery and/or consume... </pre>
Type	typeSdsSendReport
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSdsSendReport
Properties	content: complex
Model	requestId , target , msgRef , deliveryStatus
Children	deliveryStatus, msgRef, requestId, target
Instance	<pre> <sendReport xmlns="DR-GW"> <requestId>{1,1}</requestId> <target>{1,1}</target> <msgRef>{1,1}</msgRef> <deliveryStatus>{1,1}</deliveryStatus> </sendReport> </pre>
Source	<code><xss:element name="sendReport" type="typeSdsSendReport" /></code>

Element typeSdsSendReport / target

Namespace	DR-GW
Diagram	<pre> classDiagram typeAddress { subscriber alias msisdn fssn external opta cell } note over fssn: Fleet specific short number note over typeAddress: Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA). </pre>
Type	typeAddress

Properties	content: complex
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre><target xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </target></pre>
Source	<code><xss:element name="target" type="typeAddress" /></code>

Element typeSdssSendReport / msgRef

Namespace	DR-GW
Diagram	<pre> graph LR msgRef[msgRef] --> xs[xs:unsignedByte] </pre> <p>Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to...</p>
Type	xs:unsignedByte
Properties	content: simple
Source	<code><xss:element name="msgRef" type="xs:unsignedByte" /></code>

Element typeSdssSendReport / deliveryStatus

Namespace	DR-GW
Diagram	<pre> graph LR deliveryStatus[deliveryStatus] --> xs[xs:unsignedByte] </pre> <p>Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to...</p>
Type	xs:unsignedByte
Properties	content: simple
Source	<code><xss:element name="deliveryStatus" type="xs:unsignedByte" /></code>

Element interfaceSds / response

Namespace	DR-GW
Diagram	<pre> graph TD typeResponse["typeResponse"] --> response[response] response --> requestId[requestId] response --> result[result] </pre> <p>Response contains result of execution of any method.</p>
Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<code><response xmlns="DR-GW"></code>

	<pre><requestId>{1,1}</requestId> <result>{1,1}</result> </response></pre>
Source	<code><xss:element name="response" type="typeResponse" /></code>

Element interfaceSds / sendEvent

Namespace	DR-GW
Diagram	<p>This type is used as a final response to previous send request and contains final result if the message was sent or...</p>
Type	typeSdsSendEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSdsSendEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , msgRef{0,1} , sds
Children	msgRef, requestId, result, sds
Instance	<pre><sendEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <msgRef>{0,1}</msgRef> <sds>{1,1}</sds> </sendEvent></pre>
Source	<code><xss:element name="sendEvent" type="typeSdsSendEvent" /></code>

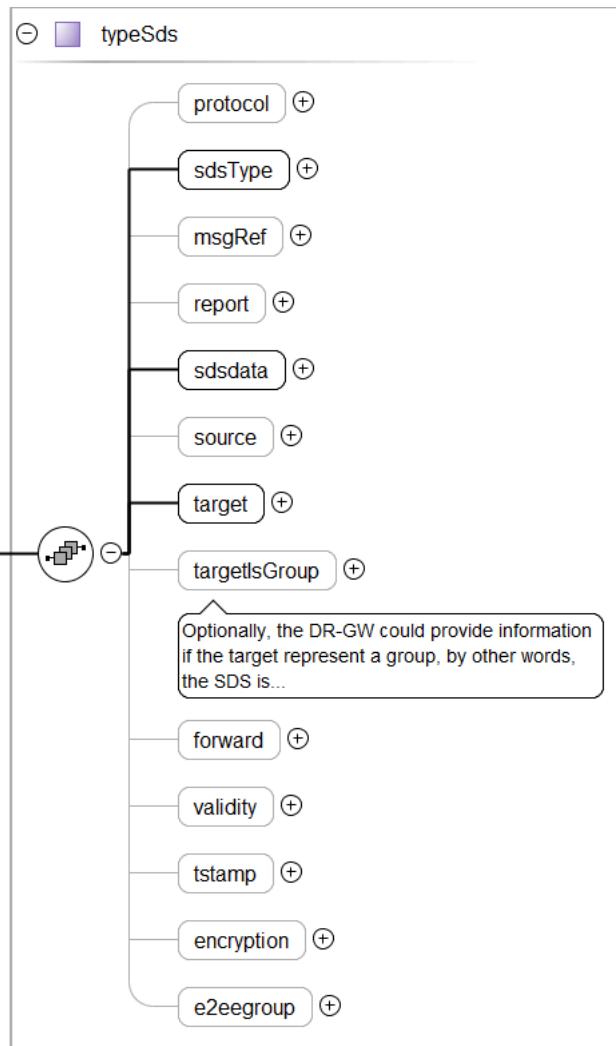
Element typeSdsSendEvent / msgRef

Namespace	DR-GW						
Diagram	<p>Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to...</p>						
Type	xs:unsignedByte						
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>default:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0	default:	0
content:	simple						
minOccurs:	0						
default:	0						
Source	<code><xss:element name="msgRef" type="xs:unsignedByte" minOccurs="0" default="0" /></code>						

Element typeSdsSendEvent / sds

Namespace	DR-GW
-----------	-------

Diagram

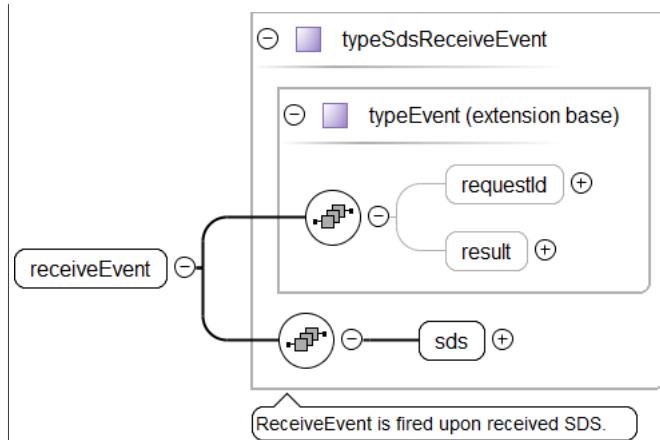


Type	<code>typeSds</code>
Properties	content: complex
Model	<code>protocol{0,1}</code> , <code>sdsType</code> , <code>msgRef{0,1}</code> , <code>report{0,1}</code> , <code>sdsdata</code> , <code>source{0,1}</code> , <code>target</code> , <code>targetIsGroup{0,1}</code> , <code>forward{0,1}</code> , <code>validity{0,1}</code> , <code>tstamp{0,1}</code> , <code>encryption{0,1}</code> , <code>e2eegroup{0,1}</code>
Children	<code>e2eegroup</code> , <code>encryption</code> , <code>forward</code> , <code>msgRef</code> , <code>protocol</code> , <code>report</code> , <code>sdsType</code> , <code>sdsdata</code> , <code>source</code> , <code>target</code> , <code>targetIsGroup</code> , <code>tstamp</code> , <code>validity</code>
Instance	<pre> <sds xmlns="DR-GW"> <protocol>{0,1}</protocol> <sdsType>{1,1}</sdsType> <msgRef>{0,1}</msgRef> <report>{0,1}</report> <sdsdata>{1,1}</sdsdata> <source>{0,1}</source> <target>{1,1}</target> <targetIsGroup>{0,1}</targetIsGroup> <forward>{0,1}</forward> <validity>{0,1}</validity> <tstamp>{0,1}</tstamp> <encryption>{0,1}</encryption> <e2eegroup>{0,1}</e2eegroup> </sds> </pre>
Source	<code><xss:element name="sds" type="typeSds" /></code>

Element interfaceSds / receiveEvent

Namespace	DR-GW
-----------	-------

Diagram

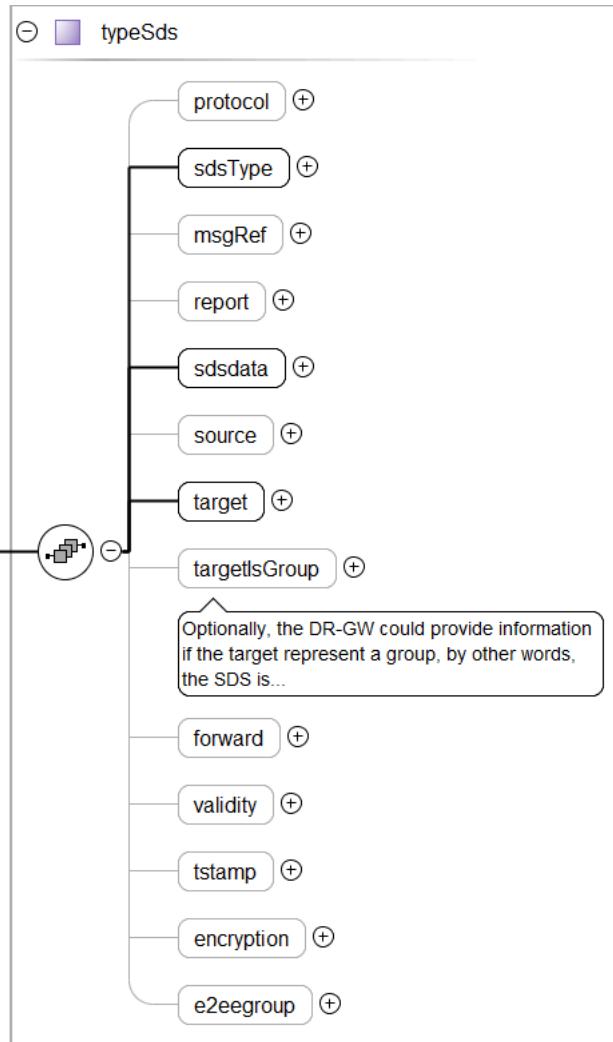


Type	<code>typeSdsReceiveEvent</code>
Type hierarchy	<ul style="list-style-type: none"> <code>typeEvent</code> <code>typeSdsReceiveEvent</code>
Properties	content: complex
Model	<code>requestId{0,1}</code> , <code>result{0,1}</code> , <code>sds</code>
Children	<code>requestId</code> , <code>result</code> , <code>sds</code>
Instance	<pre> <receiveEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <sds>{1,1}</sds> </receiveEvent> </pre>
Source	<code><xss:element name="receiveEvent" type="typeSdsReceiveEvent" /></code>

Element `typeSdsReceiveEvent` / `sds`

Namespace	DR-GW
-----------	-------

Diagram



Type	typeSds
Properties	content: complex
Model	protocol{0,1} , sdsType , msgRef{0,1} , report{0,1} , sdsdata , source{0,1} , target , targetIsGroup{0,1} , forward{0,1} , validity{0,1} , tstamp{0,1} , encryption{0,1} , e2eegroup{0,1}
Children	e2eegroup, encryption, forward, msgRef, protocol, report, sdsType, sdsdata, source, target, targetIsGroup, tstamp, validity
Instance	<pre> <sds xmlns="DR-GW"> <protocol>{0,1}</protocol> <sdsType>{1,1}</sdsType> <msgRef>{0,1}</msgRef> <report>{0,1}</report> <sdsdata>{1,1}</sdsdata> <source>{0,1}</source> <target>{1,1}</target> <targetIsGroup>{0,1}</targetIsGroup> <forward>{0,1}</forward> <validity>{0,1}</validity> <tstamp>{0,1}</tstamp> <encryption>{0,1}</encryption> <e2eegroup>{0,1}</e2eegroup> </sds> </pre>
Source	<xss:element name="sds" type="typeSds" />

Element interfaceSds / reportEvent

Namespace	DR-GW
-----------	-------

Diagram	<p>The diagram shows the structure of the <code>typeSdsReportEvent</code> element. It is an extension of the <code>typeEvent</code> base type. The <code>reportEvent</code> element contains attributes: <code>requestId</code>, <code>result</code>, <code>source</code>, <code>target</code>, <code>msgRef</code>, <code>deliveryStatus</code>, and <code>tstamp</code>. A note at the bottom states: "ReportEvent is fired whenever the delivery or consume report is received."</p>
Type	<code>typeSdsReportEvent</code>
Type hierarchy	<ul style="list-style-type: none"> • <code>typeEvent</code> • <code>typeSdsReportEvent</code>
Properties	content: complex
Model	<code>requestId{0,1}</code> , <code>result{0,1}</code> , <code>source</code> , <code>target</code> , <code>msgRef</code> , <code>deliveryStatus</code> , <code>tstamp</code>
Children	<code>deliveryStatus</code> , <code>msgRef</code> , <code>requestId</code> , <code>result</code> , <code>source</code> , <code>target</code> , <code>tstamp</code>
Instance	<pre><reportEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <source>{1,1}</source> <target>{1,1}</target> <msgRef>{1,1}</msgRef> <deliveryStatus>{1,1}</deliveryStatus> <tstamp>{1,1}</tstamp> </reportEvent></pre>
Source	<code><xss:element name="reportEvent" type="typeSdsReportEvent" /></code>

Element `typeSdsReportEvent` / `source`

Namespace	DR-GW
-----------	-------

Diagram	<p>typeAddress</p> <ul style="list-style-type: none"> source (multiplicity 0..1) subscriber alias msisdn fssn (note: Fleet specific short number) external opta cell <p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>
Type	typeAddress
Properties	content: complex
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre><source xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </source></pre>
Source	<code><xss:element name="source" type="typeAddress" /></code>

Element typeSdsReportEvent / target

Namespace	DR-GW
-----------	-------

Diagram	<pre> classDiagram class typeAddress { subscriber {0..1} alias {0..1} msisdn {0..1} fssn {0..1} external {0..1} opta {0..1} cell {0..1} } target "0..1" --> typeAddress typeAddress < -- basicTypeForAllPossibleTETRAAddressTypes </pre>
Type	typeAddress
Properties	content: complex
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre> <target xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </target> </pre>
Source	<xss:element name="target" type="typeAddress" />

Element typeSdsReportEvent / msgRef

Namespace	DR-GW
Diagram	<pre> classDiagram element msgRef --> xs:unsignedByte xs:unsignedByte < -- builtInDerivedType </pre>
Type	xs:unsignedByte
Properties	content: simple
Source	<xss:element name="msgRef" type="xs:unsignedByte" />

Element typeSdsReportEvent / deliveryStatus

Namespace	DR-GW
Diagram	<pre> classDiagram element deliveryStatus --> xs:unsignedByte xs:unsignedByte < -- builtInDerivedType </pre>
Type	xs:unsignedByte

Properties	content: simple
Source	<xs:element name="deliveryStatus" type="xs:unsignedByte"/>

Element typeSdsReportEvent / tstamp

Namespace	DR-GW
Diagram	<p>tstamp</p> <p>xs:dateTime</p> <p>Built-in primitive type. The dateTime datatype represents a specific instant of time.</p>
Type	xs:dateTime
Properties	content: simple
Source	<xs:element name="tstamp" type="xs:dateTime"/>

Element drgw / status

Namespace	DR-GW
Diagram	<p>status</p> <p>Type interfaceStatus</p> <p>interfaceStatus</p> <p>send</p> <p>response</p> <p>sendEvent</p> <p>receiveEvent</p> <p>DR-GW-Status element. Use to send/receive status messages. Use only via SOAP.</p>
Type	interfaceStatus
Properties	content: complex
Model	send response sendEvent receiveEvent
Children	receiveEvent, response, send, sendEvent
Instance	<pre><status xmlns="DR-GW"> <send>{1,1}</send> <response>{1,1}</response> <sendEvent>{1,1}</sendEvent> <receiveEvent>{1,1}</receiveEvent> </status></pre>
Source	<xs:element name="status" type="interfaceStatus"/>

Element interfaceStatus / send

Namespace	DR-GW
Diagram	<p>typeStatusSend</p> <p>typeRequest (extension base)</p> <p>requestId</p> <p>status</p>

Type	typeStatusSend
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeStatusSend
Properties	content: complex
Model	requestId , status
Children	requestId, status
Instance	<pre><send xmlns="DR-GW"> <requestId>{1,1}</requestId> <status>{1,1}</status> </send></pre>
Source	<code><xss:element name="send" type="typeStatusSend"/></code>

Element typeStatusSend / status

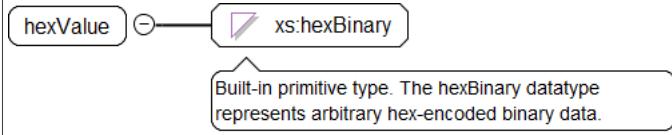
Namespace	DR-GW
Diagram	
Type	typeStatusSend
Properties	content: complex
Model	(value hexValue) , source{0,1} , target , tstamp{0,1}
Children	hexValue, source, target, tstamp, value
Instance	<pre><status xmlns="DR-GW"> <value>{1,1}</value> <hexValue>{1,1}</hexValue> <source>{0,1}</source> <target>{1,1}</target> <tstamp>{0,1}</tstamp> </status></pre>
Source	<code><xss:element name="status" type="typeStatus"/></code>

Element typeStatus / value

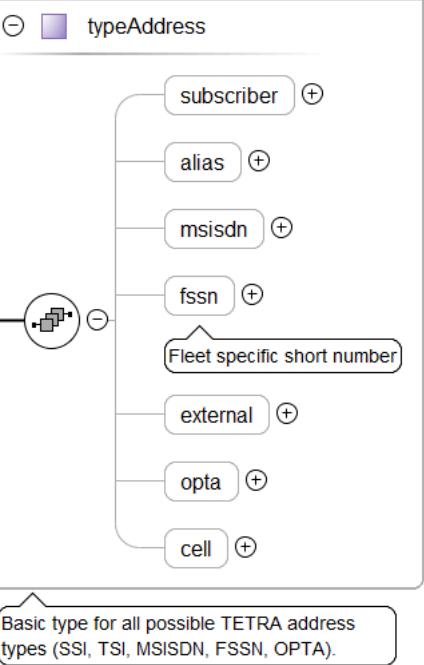
Namespace	DR-GW
Diagram	<p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>
Type	xs:unsignedShort
Properties	content: simple
Source	<code><xss:element name="value" type="xs:unsignedShort"/></code>

Element typeStatus / hexValue

Namespace	DR-GW
-----------	-------

Diagram	
Type	xs:hexBinary
Properties	content: simple
Source	<code><xs:element name="hexValue" type="xs:hexBinary" /></code>

Element typeStatus / source

Namespace	DR-GW
Diagram	
Type	typeAddress
Properties	content: complex minOccurs: 0
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre><source xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </source></pre>
Source	<code><xs:element name="source" type="typeAddress" minOccurs="0" /></code>

Element typeStatus / target

Namespace	DR-GW
-----------	-------

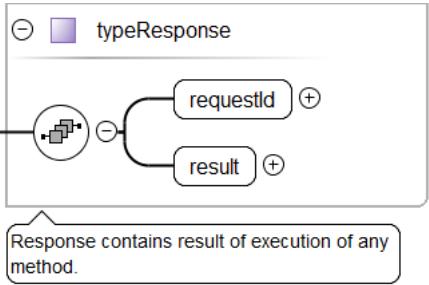
Diagram	<p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>
Type	typeAddress
Properties	content: complex
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Instance	<pre><target xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </target></pre>
Source	<code><xss:element name="target" type="typeAddress" /></code>

Element typeStatus / tstamp

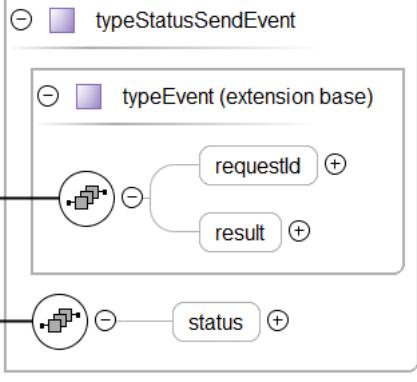
Namespace	DR-GW				
Diagram	<p>Built-in primitive type. The dateTime datatype represents a specific instant of time.</p>				
Type	xs:dateTime				
Properties	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">content:</td> <td style="padding: 2px;">simple</td> </tr> <tr> <td style="padding: 2px;">minOccurs:</td> <td style="padding: 2px;">0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="tstamp" type="xs:dateTime" minOccurs="0" /></code>				

Element interfaceStatus / response

Namespace	DR-GW
-----------	-------

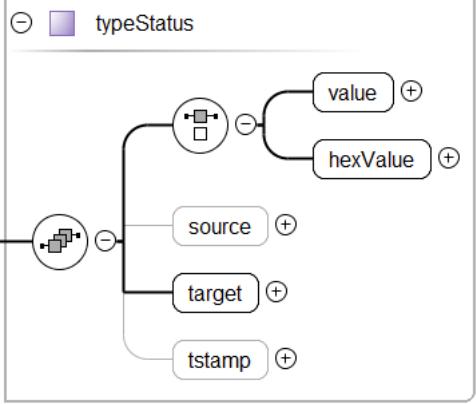
Diagram	 <p>Response contains result of execution of any method.</p>
Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<pre><response xmlns="DR-GW"> <requestIds>{1,1}</requestIds> <result>{1,1}</result> </response></pre>
Source	<code><xss:element name="response" type="typeResponse" /></code>

Element interfaceStatus / sendEvent

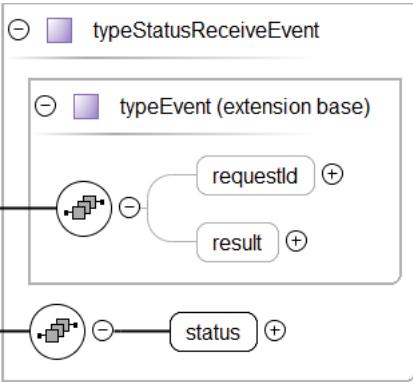
Namespace	DR-GW
Diagram	
Type	typeStatusSendEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeStatusSendEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , status{0,1}
Children	requestId, result, status
Instance	<pre><sendEvent xmlns="DR-GW"> <requestIds>{0,1}</requestIds> <result>{0,1}</result> <status>{0,1}</status> </sendEvent></pre>
Source	<code><xss:element name="sendEvent" type="typeStatusSendEvent" /></code>

Element typeStatusSendEvent / status

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeStatus
Properties	<p>content: complex</p> <p>minOccurs: 0</p>
Model	(value hexValue) , source{0,1} , target , tstamp{0,1}
Children	hexValue, source, target, tstamp, value
Instance	<pre><status xmlns="DR-GW"> <value>{1,1}</value> <hexValue>{1,1}</hexValue> <source>{0,1}</source> <target>{1,1}</target> <tstamp>{0,1}</tstamp> </status></pre>
Source	<code><xss:element name="status" type="typeStatus" minOccurs="0"/></code>

Element interfaceStatus / receiveEvent

Namespace	DR-GW
Diagram	
Type	typeStatusReceiveEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeStatusReceiveEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , status
Children	requestId, result, status
Instance	<pre><receiveEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <status>{1,1}</status> </receiveEvent></pre>
Source	<code><xss:element name="receiveEvent" type="typeStatusReceiveEvent"/></code>

Element typeStatusReceiveEvent / status

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeStatus
Properties	content: complex
Model	(value hexValue) , source{0,1} , target , tstamp{0,1}
Children	hexValue, source, target, tstamp, value
Instance	<pre><status xmlns="DR-GW"> <value>{1,1}</value> <hexValue>{1,1}</hexValue> <source>{0,1}</source> <target>{1,1}</target> <tstamp>{0,1}</tstamp> </status></pre>
Source	<code><xss:element name="status" type="typeStatus"/></code>

Element drgw / org

Namespace	DR-GW
Diagram	
Type	interfaceOrg
Properties	content: complex
Model	get getList response getEvent getListEvent event
Children	event, get, getEvent, getList, getListEvent, response
Instance	<pre><org xmlns="DR-GW"> <get>{1,1}</get> <getList>{1,1}</getList> <response>{1,1}</response> <getEvent>{1,1}</getEvent> <getListEvent>{1,1}</getListEvent> <event>{1,1}</event> </org></pre>

Source

```
<xss:element name="org" type="interfaceOrg" />
```

Element interfaceOrg / get

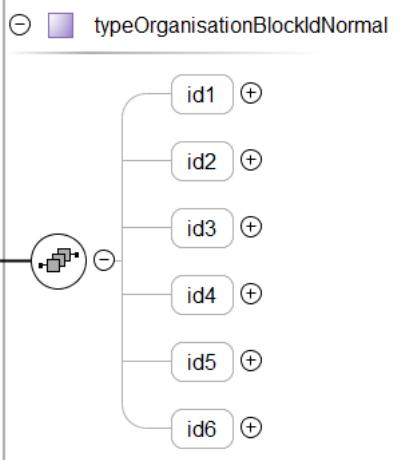
Namespace	DR-GW
Diagram	<pre> classDiagram typeOrgGet { typeRequest get } typeRequest { requestId orgblockId } typeOrgGet < -- typeRequest typeRequest --> get </pre>
Type	typeOrgGet
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeOrgGet
Properties	content: complex
Model	requestId , orgblockId
Children	orgblockId, requestId
Instance	<pre> <get xmlns="DR-GW"> <requestId>{1,1}</requestId> <orgblockId>{1,1}</orgblockId> </get> </pre>
Source	<pre><xss:element name="get" type="typeOrgGet" /></pre>

Element typeOrgGet / orgblockId

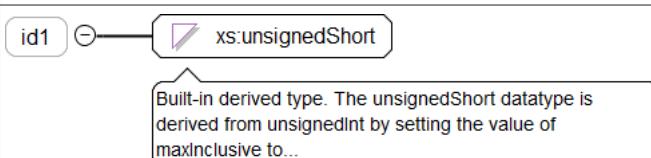
Namespace	DR-GW
Diagram	<pre> classDiagram typeOrganisationBlockId { orgblockId } orgblockId { orgblockIdSimple } typeOrganisationBlockId < -- orgblockId orgblockId --> orgblockIdSimple </pre>
Type	typeOrganisationBlockId
Properties	content: complex
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<pre> <orgblockId xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <orgblockIdSimple>{1,1}</orgblockIdSimple> </orgblockId> </pre>
Source	<pre><xss:element name="orgblockId" type="typeOrganisationBlockId" /></pre>

Element typeOrganisationBlockId / orgblockId

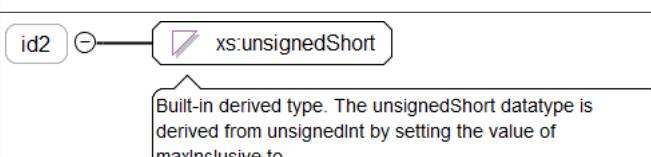
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeOrganisationBlockIdNormal
Properties	content: complex
Model	id1{0,1} , id2{0,1} , id3{0,1} , id4{0,1} , id5{0,1} , id6{0,1}
Children	id1, id2, id3, id4, id5, id6
Instance	<pre><orgblockId xmlns="DR-GW"> <id1>{0,1}</id1> <id2>{0,1}</id2> <id3>{0,1}</id3> <id4>{0,1}</id4> <id5>{0,1}</id5> <id6>{0,1}</id6> </orgblockId></pre>
Source	<code><xs:element name="orgblockId" type="typeOrganisationBlockIdNormal" /></code>

Element typeOrganisationBlockIdNormal / id1

Namespace	DR-GW
Diagram	 <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>
Type	xs:unsignedShort
Properties	content: simple minOccurs: 0
Source	<code><xs:element name="id1" type="xs:unsignedShort" minOccurs="0" /></code>

Element typeOrganisationBlockIdNormal / id2

Namespace	DR-GW
Diagram	 <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>
Type	xs:unsignedShort
Properties	content: simple minOccurs: 0
Source	<code><xs:element name="id2" type="xs:unsignedShort" minOccurs="0" /></code>

Element typeOrganisationBlockIdNormal / id3

Namespace	DR-GW				
Diagram	<p>id3</p> <p>xs:unsignedShort</p> <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>				
Type	xs:unsignedShort				
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="id3" type="xs:unsignedShort" minOccurs="0" /></code>				

Element typeOrganisationBlockIdNormal / id4

Namespace	DR-GW				
Diagram	<p>id4</p> <p>xs:unsignedShort</p> <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>				
Type	xs:unsignedShort				
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="id4" type="xs:unsignedShort" minOccurs="0" /></code>				

Element typeOrganisationBlockIdNormal / id5

Namespace	DR-GW				
Diagram	<p>id5</p> <p>xs:unsignedShort</p> <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>				
Type	xs:unsignedShort				
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="id5" type="xs:unsignedShort" minOccurs="0" /></code>				

Element typeOrganisationBlockIdNormal / id6

Namespace	DR-GW				
Diagram	<p>id6</p> <p>xs:unsignedShort</p> <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>				
Type	xs:unsignedShort				
Properties	<table> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="id6" type="xs:unsignedShort" minOccurs="0" /></code>				

Element typeOrganisationBlockId / orgblockIdSimple

Namespace	DR-GW
Diagram	<p>The diagram shows a class named "typeOrganisationBlockIdSimple" with a multiplicity of 0..1. A note below it states: "Organisation block send as simple normalized string. The pattern is: id1-id2-id3-id4-id5-id6".</p>
Type	typeOrganisationBlockIdSimple
Properties	content: simple
Facets	<p>pattern</p> <pre>(([0-9] [1-9]\d{0,3} [1-5]\d{4} 6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])-){0,5}([0-9] [1-9]\d{0,3} [1-5]\d{4} 6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])</pre>
Source	<code><xs:element name="orgblockIdSimple" type="typeOrganisationBlockIdSimple"/></code>

Element interfaceOrg / getList

Namespace	DR-GW
Diagram	<p>The diagram shows a class named "typeOrgGetList" which is an extension base for "typeRequest". It has two attributes: "requestId" and "orgblockId", both with multiplicity 0..1.</p>
Type	typeOrgGetList
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeOrgGetList
Properties	content: complex
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Instance	<code><getList xmlns="DR-GW"> <requestId>{1,1}</requestId> <orgblockId>{0,1}</orgblockId> </getList></code>
Source	<code><xs:element name="getList" type="typeOrgGetList"/></code>

Element typeOrgGetList / orgblockId

Namespace	DR-GW
Diagram	<p>The diagram shows a class named "typeOrganisationBlockId" with an attribute "orgblockId" (multiplicity 0..1). This attribute points to another attribute "orgblockIdSimple" (multiplicity 0..1).</p>
Type	typeOrganisationBlockId

Properties	content: complex minOccurs: 0
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<orgblockId xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <orgblockIdSimple>{1,1}</orgblockIdSimple> </orgblockId>
Source	<xss:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0"/>

Element interfaceOrg / response

Namespace	DR-GW
Diagram	<pre> classDiagram class typeResponse { response * --> requestId 1..1 response * --> result 1..1 } note over requestId, result: Response contains result of execution of any method. </pre>
Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<response xmlns="DR-GW"> <requestId>{1,1}</requestId> <result>{1,1}</result> </response>
Source	<xss:element name="response" type="typeResponse" />

Element interfaceOrg / getEvent

Namespace	DR-GW
Diagram	<pre> classDiagram class typeOrgGetEvent { getEvent * --> requestId 1..1 getEvent * --> result 1..1 getEvent --> orgblock 1..1 } class typeEvent { <<extension base>> } note over typeEvent: typeEvent (extension base) </pre>
Type	typeOrgGetEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeOrgGetEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , orgblock
Children	orgblock, requestId, result
Instance	<getEvent xmlns="DR-GW"> <requestId>{0,1}</requestId>

	<pre><result>{0,1}</result> <orgblock>{1,1}</orgblock> </getEvent></pre>
Source	<code><xs:element name="getEvent" type="typeOrgGetEvent" /></code>

Element typeOrgGetEvent / orgblock

Namespace	DR-GW
Diagram	<pre> classDiagram class typeOrganisationBlock { orgblock* --> orgblockId orgblock* --> alias } orgblockId < -- orgblockIdSimple </pre>
Type	typeOrganisationBlock
Properties	content: complex
Model	orgblockId , alias
Children	alias, orgblockId
Instance	<pre> <orgblock xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <alias>{1,1}</alias> </orgblock> </pre>
Source	<code><xs:element name="orgblock" type="typeOrganisationBlock" /></code>

Element typeOrganisationBlock / orgblockId

Namespace	DR-GW
Diagram	<pre> classDiagram class typeOrganisationBlockId { orgblockId* --> orgblockId orgblockId* --> orgblockIdSimple } orgblockIdSimple < -- orgblockIdSimple </pre>
Type	typeOrganisationBlockId
Properties	content: complex
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<pre> <orgblockId xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <orgblockIdSimple>{1,1}</orgblockIdSimple> </orgblockId> </pre>
Source	<code><xs:element name="orgblockId" type="typeOrganisationBlockId" /></code>

Element typeOrganisationBlock / alias

Namespace	DR-GW
Diagram	<p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>
Type	xs:normalizedString
Properties	content: simple
Source	<code><xs:element name="alias" type="xs:normalizedString" /></code>

Element interfaceOrg / getListEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeOrgGetListEvent typeEvent { requestId result } typeOrgGetListEvent { orgblock *--> listEnd } orgblock { orgblockId alias } </pre>
Type	typeOrgGetListEvent
Type hierarchy	<ul style="list-style-type: none"> typeEvent typeOrgGetListEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , orgblock* , listEnd
Children	listEnd, orgblock, requestId, result
Instance	<pre> <getListEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <orgblock>{0,unbounded}</orgblock> <listEnd>{1,1}</listEnd> </getListEvent> </pre>
Source	<xss:element name="getListEvent" type="typeOrgGetListEvent" />

Element typeOrgGetListEvent / orgblock

Namespace	DR-GW						
Diagram	<pre> typeOrganisationBlock < -- typeOrgGetListEvent typeOrganisationBlock { orgblockId alias } alias { orgblockId alias } </pre>						
Type	typeOrganisationBlock						
Properties	<table> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>maxOccurs:</td> <td>unbounded</td> </tr> </table>	content:	complex	minOccurs:	0	maxOccurs:	unbounded
content:	complex						
minOccurs:	0						
maxOccurs:	unbounded						
Model	orgblockId , alias						
Children	alias, orgblockId						
Instance	<pre> <orgblock xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <alias>{1,1}</alias> </orgblock> </pre>						
Source	<xss:element name="orgblock" type="typeOrganisationBlock" minOccurs="0" maxOccurs="unbounded" />						

Element typeOrgGetListEvent / listEnd

Namespace	DR-GW
-----------	-------

Diagram	 Built-in primitive type. It defines the boolean values true and false.
Type	xs:boolean
Properties	content: simple
Source	<xs:element name="listEnd" type="xs:boolean" />

Element interfaceOrg / event

Namespace	DR-GW
Diagram	
Type	typeOrgEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeOrgEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , orgblock , delete
Children	delete, orgblock, requestId, result
Instance	<pre> <event xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <orgblock>{1,1}</orgblock> <delete>{1,1}</delete> </event> </pre>
Source	<xs:element name="event" type="typeOrgEvent" />

Element typeOrgEvent / orgblock

Namespace	DR-GW
Diagram	
Type	typeOrganisationBlock
Properties	content: complex
Model	orgblockId , alias
Children	alias, orgblockId
Instance	<pre> <orgblock xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> </orgblock> </pre>

	<pre> <alias>{1,1}</alias> </orgblock> </pre>
Source	<pre><xss:element name="orgblock" type="typeOrganisationBlock"/></pre>

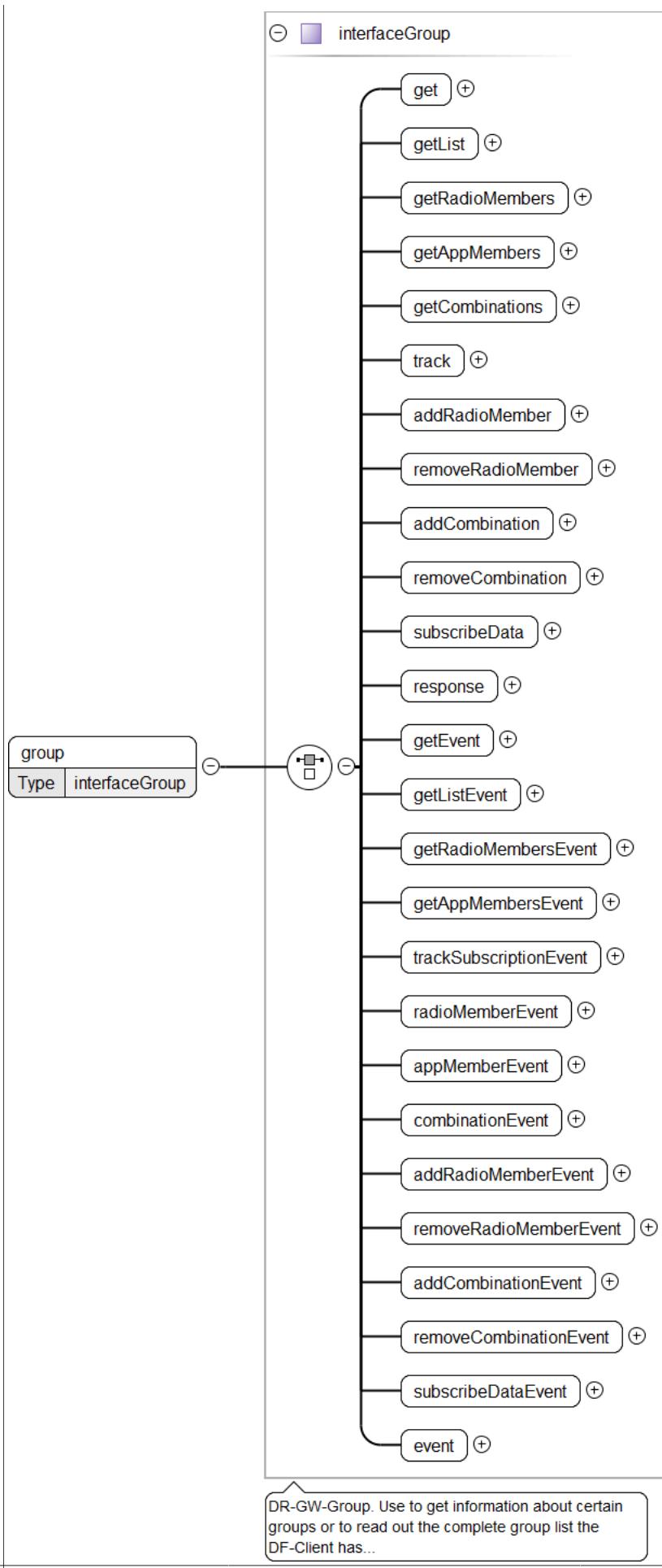
Element typeOrgEvent / delete

Namespace	DR-GW
Diagram	<p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<pre><xss:element name="delete" type="xs:boolean"/></pre>

Element drgw / group

Namespace	DR-GW
-----------	-------

Diagram



Type	interfaceGroup
Properties	content: complex
Model	get getList getRadioMembers getAppMembers getCombinations track addRadioMember removeRadioMember addCombination removeCombination subscribeData response getEvent getListEvent getRadioMembersEvent getAppMembersEvent trackSubscriptionEvent radioMemberEvent appMemberEvent combinationEvent addRadioMemberEvent removeRadioMemberEvent addCombinationEvent removeCombinationEvent subscribeDataEvent event
Children	addCombination, addCombinationEvent, addRadioMember, addRadioMemberEvent, appMemberEvent, combinationEvent, event, get, getAppMembers, getAppMembersEvent, getCombinations, getEvent, getList, getListEvent, getRadioMembers, getRadioMembersEvent, radioMemberEvent, removeCombination, removeCombinationEvent, removeRadioMember, removeRadioMemberEvent, response, subscribeData, subscribeDataEvent, track, trackSubscriptionEvent
Instance	<pre> <group xmlns="DR-GW"> <get>{1,1}</get> <getList>{1,1}</getList> <getRadioMembers>{1,1}</getRadioMembers> < getAppMembers>{1,1}</ getAppMembers> <getCombinations>{1,1}</getCombinations> <track>{1,1}</track> <addRadioMember>{1,1}</addRadioMember> <removeRadioMember>{1,1}</removeRadioMember> <addCombination>{1,1}</addCombination> <removeCombination>{1,1}</removeCombination> <subscribeData>{1,1}</subscribeData> <response>{1,1}</response> <getEvent>{1,1}</getEvent> <getListEvent>{1,1}</getListEvent> <getRadioMembersEvent>{1,1}</getRadioMembersEvent> < getAppMembersEvent>{1,1}</ getAppMembersEvent> <trackSubscriptionEvent>{1,1}</trackSubscriptionEvent> <radioMemberEvent>{1,1}</radioMemberEvent> <appMemberEvent>{1,1}</appMemberEvent> <combinationEvent>{1,1}</combinationEvent> <addRadioMemberEvent>{1,1}</addRadioMemberEvent> <removeRadioMemberEvent>{1,1}</removeRadioMemberEvent> <addCombinationEvent>{1,1}</addCombinationEvent> <removeCombinationEvent>{1,1}</removeCombinationEvent> <subscribeDataEvent>{1,1}</subscribeDataEvent> <event>{1,1}</event> </group></pre>
Source	<xss:element name="group" type="interfaceGroup" />

Element interfaceGroup / get

Namespace	DR-GW
Diagram	<pre> classDiagram class typeGroupGet { <<interface group>> <<extension base="typeRequest">> <<operations>> <<get>> : <<requestId>> <<group>> <<group>> : <<group>> </operations> } </pre>
Type	typeGroupGet
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGet
Properties	content: complex
Model	requestId , group
Children	group, requestId
Instance	<pre> <get xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,1}</group> </get></pre>
Source	<xss:element name="get" type="typeGroupGet" />

Element typeGroupGet / group

Namespace	DR-GW
Diagram	<pre> classDiagram class typeGroupGet { <<group>> } class typeSubscriberAddress { <<ssi>> <<tsi>> } typeGroupGet "1..1" --> "1..1" typeSubscriberAddress : group typeSubscriberAddress "1..1" --> "1..1" ssi typeSubscriberAddress "1..1" --> "1..1" tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group> </pre>
Source	<pre> <xss:element name="group" type="typeSubscriberAddress"/> </pre>

Element interfaceGroup / getList

Namespace	DR-GW
Diagram	<pre> classDiagram class typeGroupGetList { <<getList>> } class typeRequest { <<requestId>> <<orgblockId>> } typeGroupGetList "1..1" --> "1..1" typeRequest : getList typeRequest "1..1" --> "1..1" requestId typeRequest "1..1" --> "1..1" orgblockId </pre>
Type	typeGroupGetList
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGetList
Properties	content: complex
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Instance	<pre> <getList xmlns="DR-GW"> <requestId>{1,1}</requestId> <orgblockId>{0,1}</orgblockId> </getList> </pre>
Source	<pre> <xss:element name="getList" type="typeGroupGetList"/> </pre>

Element typeGroupGetList / orgblockID

Namespace	DR-GW
Diagram	<pre> classDiagram class typeGroupGetList { <<orgblockID>> } class typeOrganisationBlockId { <<orgblockId>> <<orgblockIdSimple>> } typeGroupGetList "1..1" --> "1..1" typeOrganisationBlockId : orgblockID typeOrganisationBlockId "1..1" --> "1..1" orgblockId typeOrganisationBlockId "1..1" --> "1..1" orgblockIdSimple </pre>

Type	typeOrganisationBlockId
Properties	content: complex minOccurs: 0
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<pre><orgblockId xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <orgblockIdSimple>{1,1}</orgblockIdSimple> </orgblockId></pre>
Source	<code><xss:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0" /></code>

Element interfaceGroup / getRadioMembers

Namespace	DR-GW
Diagram	<pre> classDiagram typeRequest < -- typeGroupGetRadioMembers typeGroupGetRadioMembers "1..1" -- "1..1" requestId typeGroupGetRadioMembers "1..1" -- "1..1" group </pre>
Type	typeGroupGetRadioMembers
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGetRadioMembers
Properties	content: complex
Model	requestId , group
Children	group, requestId
Instance	<pre><getRadioMembers xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,1}</group> </getRadioMembers></pre>
Source	<code><xss:element name="getRadioMembers" type="typeGroupGetRadioMembers" /></code>

Element typeGroupGetRadioMembers / group

Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress "1..1" -- "1..1" group group "1..1" -- "1..1" ssi group "1..1" -- "1..1" tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress" /></code>

Element interfaceGroup / getAppMembers

Namespace	DR-GW
Diagram	<pre> classDiagram typeGroupGetAppMembers < -- typeRequest typeGroupGetAppMembers "1..1" --> requestId : String typeGroupGetAppMembers "0..1" --> group : String </pre>
Type	typeGroupGetAppMembers
Type hierarchy	<ul style="list-style-type: none"> • typeRequest <ul style="list-style-type: none"> • typeGroupGetAppMembers
Properties	content: complex
Model	requestId , group
Children	group, requestId
Instance	<pre> <getAppMembers xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,1}</group> </getAppMembers> </pre>
Source	<code><xss:element name="getAppMembers" type="typeGroupGetAppMembers" /></code>

Element typeGroupGetAppMembers / group

Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress < -- typeGroup typeSubscriberAddress "1..1" --> ssi : String typeSubscriberAddress "1..1" --> tsi : String </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group> </pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress" /></code>

Element interfaceGroup / getCombinations

Namespace	DR-GW
-----------	-------

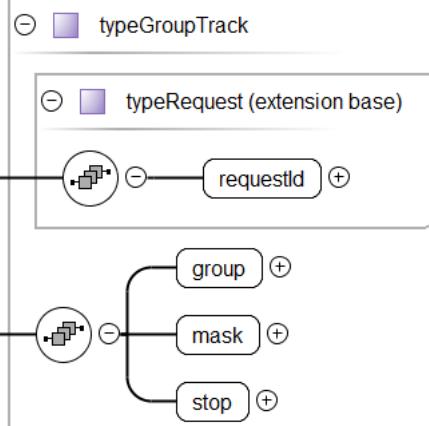
Diagram	<pre> classDiagram typeGroupGetCombinations < -- typeRequest typeGroupGetCombinations "0..1" --> "1..1" requestId typeGroupGetCombinations "0..1" --> "1..1" group note over typeGroupGetCombinations: The method requests the groups that belong to the same combined group as the group specified. </pre>
Type	typeGroupGetCombinations
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGetCombinations
Properties	content: complex
Model	requestId , group
Children	group, requestId
Instance	<pre> <getCombinations xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,1}</group> </getCombinations> </pre>
Source	<code><xs:element name="getCombinations" type="typeGroupGetCombinations" /></code>

Element typeGroupGetCombinations / group

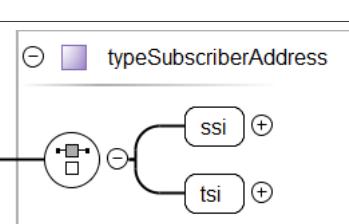
Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress "0..1" --> "1..1" ssi typeSubscriberAddress "0..1" --> "1..1" tsi note over typeSubscriberAddress: The method requests the groups that belong to the same combined group as the group specified. </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group> </pre>
Source	<code><xs:element name="group" type="typeSubscriberAddress" /></code>

Element interfaceGroup / track

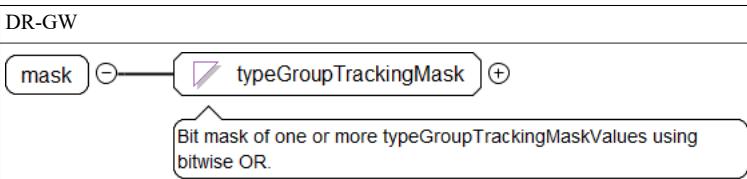
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeGroupTrack
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupTrack
Properties	content: complex
Model	requestId , group , mask , stop
Children	group, mask, requestId, stop
Instance	<pre><track xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,1}</group> <mask>{1,1}</mask> <stop>{1,1}</stop> </track></pre>
Source	<code><xss:element name="track" type="typeGroupTrack" /></code>

Element typeGroupTrack / group

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress" /></code>

Element typeGroupTrack / mask

Namespace	DR-GW
Diagram	
Type	typeGroupTrackingMask

Properties	content: simple
Source	<xs:element name="mask" type="typeGroupTrackingMask"/>

Element typeGroupTrack / stop

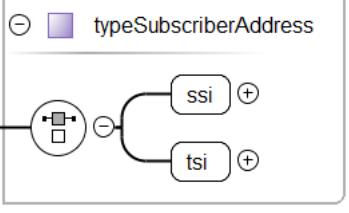
Namespace	DR-GW
Diagram	<p>The diagram shows a class named 'stop' with a multiplicity of 0..1. It has a directed association labeled 'xs:boolean' with a target compartment containing the icon for the xs:boolean type. A callout box states: 'Built-in primitive type. It defines the boolean values true and false.'</p>
Type	xs:boolean
Properties	content: simple
Source	<xs:element name="stop" type="xs:boolean"/>

Element interfaceGroup / addRadioMember

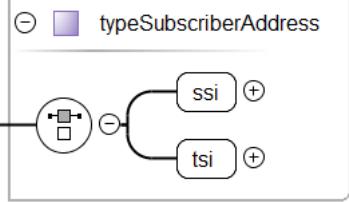
Namespace	DR-GW
Diagram	<p>The diagram shows a class named 'addRadioMember' with a multiplicity of 0..1. It has a directed association labeled 'typeRequest (extension base)' with a target compartment containing the icon for 'typeRequest'. This association is marked with a circled plus sign. Below this, there are three more associations: one to 'requestId' (multiplicity 0..1), one to 'radio' (multiplicity 1..1), one to 'group' (multiplicity 1..1), and one to 'membership' (multiplicity 0..1). A callout box states: 'Requests the addition of a radio subscriber to a group. This might cause DGNA operation in the air interface.'</p>
Type	typeGroupAddRadioMember
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupAddRadioMember
Properties	content: complex
Model	requestId , radio , group , membership{0,1}
Children	group, membership, radio, requestId
Instance	<pre><addRadioMember xmlns="DR-GW"> <requestId>{1,1}</requestId> <radio>{1,1}</radio> <group>{1,1}</group> <membership>{0,1}</membership> </addRadioMember></pre>
Source	<xs:element name="addRadioMember" type="typeGroupAddRadioMember"/>

Element typeGroupAddRadioMember / radio

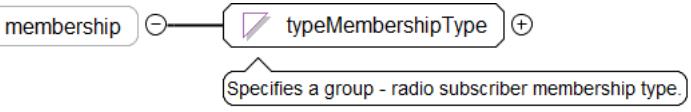
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress"/></code>

Element typeGroupAddRadioMember / group

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress"/></code>

Element typeGroupAddRadioMember / membership

Namespace	DR-GW
Diagram	 Specifies a group - radio subscriber membership type.
Type	typeMembershipType
Properties	content: simple minOccurs: 0
Facets	enumeration unknown enumeration permanent enumeration visiting
Source	<code><xss:element name="membership" type="typeMembershipType" minOccurs="0" /></code>

Element interfaceGroup / removeRadioMember

Namespace	DR-GW
-----------	-------

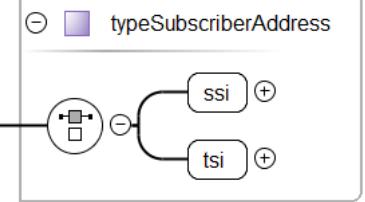
Diagram	<pre> classDiagram typeGroupRemoveRadioMember < -- typeRequest typeGroupRemoveRadioMember { <> requestId : <<radio>> <> radio : <<radio>> <> group : <<radio>> } </pre> <p>Requests removing a radio subscriber from a group. This might cause DGNA operation in the air interface.</p>
Type	typeGroupRemoveRadioMember
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupRemoveRadioMember
Properties	content: complex
Model	requestId , radio , group
Children	group, radio, requestId
Instance	<removeRadioMember xmlns="DR-GW"> <requestId>{1,1}</requestId> <radio>{1,1}</radio> <group>{1,1}</group> </removeRadioMember>
Source	<xss:element name="removeRadioMember" type="typeGroupRemoveRadioMember"/>

Element typeGroupRemoveRadioMember / radio

Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress { <> radio : <<radio>> <> ssi : <<radio>> <> tsi : <<radio>> } </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio>
Source	<xss:element name="radio" type="typeSubscriberAddress"/>

Element typeGroupRemoveRadioMember / group

Namespace	DR-GW
-----------	-------

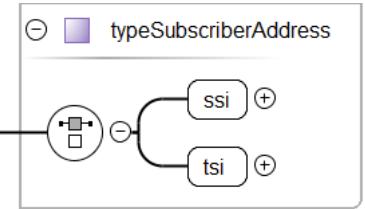
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress"/></code>

Element interfaceGroup / addCombination

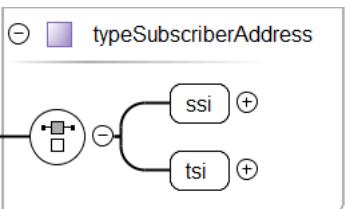
Namespace	DR-GW
Diagram	<p>Requests the addition of a group to a combined group.</p>
Type	typeGroupAddCombination
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupAddCombination
Properties	content: complex
Model	requestId , group , baseGroup , force{0,1}
Children	baseGroup, force, group, requestId
Instance	<pre><addCombination xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,1}</group> <baseGroup>{1,1}</baseGroup> <force>{0,1}</force> </addCombination></pre>
Source	<code><xss:element name="addCombination" type="typeGroupAddCombination"/></code>

Element typeGroupAddCombination / group

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress"/></code>

Element typeGroupAddCombination / baseGroup

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><baseGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </baseGroup></pre>
Source	<code><xss:element name="baseGroup" type="typeSubscriberAddress"/></code>

Element typeGroupAddCombination / force

Namespace	DR-GW
Diagram	 Built-in primitive type. It defines the boolean values true and false.
Type	xs:boolean
Properties	content: simple minOccurs: 0 default: true
Source	<code><xss:element name="force" type="xs:boolean" minOccurs="0" default="true"/></code>

Element interfaceGroup / removeCombination

Namespace	DR-GW
-----------	-------

Diagram	<pre> classDiagram typeGroupRemoveCombination < -- typeRequest typeGroupRemoveCombination "0..1" --> "1..1" removeCombination typeGroupRemoveCombination "0..1" --> "1..1" requestId typeGroupRemoveCombination "0..1" --> "1..1" group typeGroupRemoveCombination "0..1" --> "1..1" baseGroup </pre> <p>Requests removing a group from a combined group.</p>
Type	typeGroupRemoveCombination
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupRemoveCombination
Properties	content: complex
Model	requestId , group , baseGroup
Children	baseGroup, group, requestId
Instance	<pre> <removeCombination xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,1}</group> <baseGroup>{1,1}</baseGroup> </removeCombination> </pre>
Source	<code><xss:element name="removeCombination" type="typeGroupRemoveCombination"/></code>

Element typeGroupRemoveCombination / group

Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress "0..1" --> "1..1" group typeSubscriberAddress "0..1" --> "1..1" ssi typeSubscriberAddress "0..1" --> "1..1" tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group> </pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress"/></code>

Element typeGroupRemoveCombination / baseGroup

Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress "0..1" --> "1..1" baseGroup typeSubscriberAddress "0..1" --> "1..1" ssi typeSubscriberAddress "0..1" --> "1..1" tsi </pre>

Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<baseGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </baseGroup>
Source	<xss:element name="baseGroup" type="typeSubscriberAddress" />

Element interfaceGroup / subscribeData

Namespace	DR-GW
Diagram	<pre> classDiagram typeRequest < -- typeGroupSubscribeData typeRequest { requestId group } typeGroupSubscribeData { +requestId +group } typeGroupSubscribeData < -- subscribeData subscribeData { +group } </pre>
Type	typeGroupSubscribeData
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupSubscribeData
Properties	content: complex
Model	requestId , group+
Children	group, requestId
Instance	<subscribeData xmlns="DR-GW"> <requestId>{1,1}</requestId> <group>{1,unbounded}</group> </subscribeData>
Source	<xss:element name="subscribeData" type="typeGroupSubscribeData" />

Element typeGroupSubscribeData / group

Namespace	DR-GW				
Diagram	<pre> classDiagram typeGroupDataSubscription < -- group typeGroupDataSubscription { addr useSDS useStatus } group { +group } </pre>				
Type	typeGroupDataSubscription				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>maxOccurs:</td> <td>unbounded</td> </tr> </table>	content:	complex	maxOccurs:	unbounded
content:	complex				
maxOccurs:	unbounded				
Model	addr , useSDS , useStatus				
Children	addr, useSDS, useStatus				
Instance	<group xmlns="DR-GW"> <addr>{1,1}</addr> <useSDS>{1,1}</useSDS> <useStatus>{1,1}</useStatus> </group>				

Source

```
<xss:element name="group" type="typeGroupDataSubscription" maxOccurs="unbounded"/>
```

Element typeGroupDataSubscription / addr

Namespace	DR-GW
Diagram	<pre> classDiagram class addr { <<addr>> <<typeSubscriberAddress>> <<ssi>> <<tsi>> } addr < -- typeSubscriberAddress typeSubscriberAddress < -- ssi typeSubscriberAddress < -- tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <addr xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </addr> </pre>
Source	<pre><xss:element name="addr" type="typeSubscriberAddress"/></pre>

Element typeGroupDataSubscription / useSDS

Namespace	DR-GW
Diagram	<pre> classDiagram class useSDS { <<useSDS>> <<xs:boolean>> } useSDS < -- xs:boolean </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<pre><xss:element name="useSDS" type="xs:boolean"/></pre>

Element typeGroupDataSubscription / useStatus

Namespace	DR-GW
Diagram	<pre> classDiagram class useStatus { <<useStatus>> <<xs:boolean>> } useStatus < -- xs:boolean </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<pre><xss:element name="useStatus" type="xs:boolean"/></pre>

Element interfaceGroup / response

Namespace	DR-GW
Diagram	<pre> classDiagram class response { <<response>> <<typeResponse>> <<requestId>> <<result>> } response < -- typeResponse typeResponse < -- requestId typeResponse < -- result </pre> <p>Response contains result of execution of any method.</p>

Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<response xmlns="DR-GW"> <requestId>{1,1}</requestId> <result>{1,1}</result> </response>
Source	<xss:element name="response" type="typeResponse" />

Element interfaceGroup / getEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeGroupGetEvent typeGroupGetEvent "1..1" -- "1..1" getEvent typeGroupGetEvent "1..1" -- "1..1" requestId typeGroupGetEvent "1..1" -- "1..1" result typeGroupGetEvent "1..1" -- "1..1" group </pre>
Type	typeGroupGetEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupGetEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group
Children	group, requestId, result
Instance	<getEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> </getEvent>
Source	<xss:element name="getEvent" type="typeGroupGetEvent" />

Element typeGroupGetEvent / group

Namespace	DR-GW
Diagram	<pre> typeGroup "1..1" -- "1..1" group typeGroup "1..1" -- "1..1" addr typeGroup "1..1" -- "1..1" alias typeGroup "1..1" -- "1..1" orgblockId </pre>
Type	typeGroup
Properties	content: complex
Model	addr , alias , orgblockId
Children	addr, alias, orgblockId
Instance	<group xmlns="DR-GW"> <addr>{1,1}</addr>

	<pre><alias>{1,1}</alias> <orgblockId>{1,1}</orgblockId> </group></pre>
Source	<code><xs:element name="group" type="typeGroup" /></code>

Element typeGroup / addr

Namespace	DR-GW
Diagram	<pre> classDiagram class addr class typeSubscriberAddress { <<complex type>> <<content: complex>> ssi tsi } addr --> typeSubscriberAddress </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><addr xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </addr></pre>
Source	<code><xs:element name="addr" type="typeSubscriberAddress" /></code>

Element typeGroup / alias

Namespace	DR-GW
Diagram	<pre> classDiagram class alias class xsnormalizedString { <<xs:normalizedString>> <<Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...>> } alias --> xsnormalizedString </pre>
Type	xs:normalizedString
Properties	content: simple
Source	<code><xs:element name="alias" type="xs:normalizedString" /></code>

Element typeGroup / orgblockId

Namespace	DR-GW
Diagram	<pre> classDiagram class orgblockId class typeOrganisationBlockId { <<complex type>> <<content: complex>> orgblockId orgblockIdSimple } orgblockId --> typeOrganisationBlockId </pre>
Type	typeOrganisationBlockId
Properties	content: complex
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<pre><orgblockId xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <orgblockIdSimple>{1,1}</orgblockIdSimple> </orgblockId></pre>
Source	<code><xs:element name="orgblockId" type="typeOrganisationBlockId" /></code>

Element interfaceGroup / getListEvent

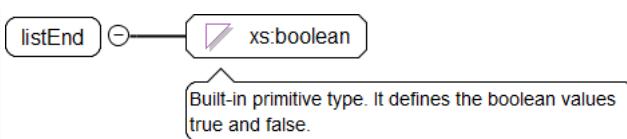
Namespace	DR-GW
Diagram	<pre> classDiagram typeGroupGetListEvent < -- typeEvent typeEvent "0..∞" --> listEnd : group typeEvent "0..∞" --> requestId : requestId typeEvent "0..∞" --> result : result getListEvent --> typeEvent : getListEvent </pre>
Type	typeGroupGetListEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupGetListEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group* , listEnd
Children	group, listEnd, requestId, result
Instance	<pre> <getListEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{0,unbounded}</group> <listEnd>{1,1}</listEnd> </getListEvent> </pre>
Source	<code><xss:element name="getListEvent" type="typeGroupGetListEvent" /></code>

Element typeGroupGetListEvent / group

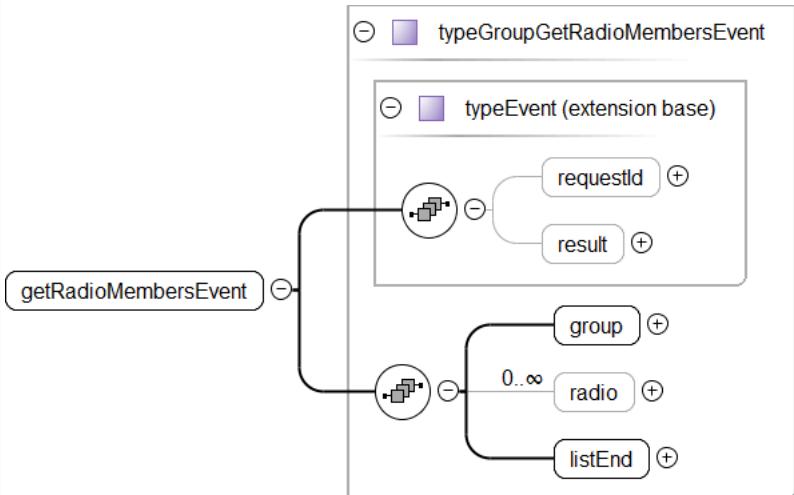
Namespace	DR-GW						
Diagram	<pre> classDiagram typeGroupGetListEvent < -- typeGroup typeGroup < -- typeGroupGetListEvent typeGroup < -- typeEvent typeGroup "0..∞" --> addr : addr typeGroup "0..∞" --> alias : alias typeGroup "0..∞" --> orgblockId : orgblockId group --> typeGroup : group </pre>						
Type	typeGroup						
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>maxOccurs:</td> <td>unbounded</td> </tr> </table>	content:	complex	minOccurs:	0	maxOccurs:	unbounded
content:	complex						
minOccurs:	0						
maxOccurs:	unbounded						
Model	addr , alias , orgblockId						
Children	addr, alias, orgblockId						
Instance	<pre> <group xmlns="DR-GW"> <addr>{1,1}</addr> <alias>{1,1}</alias> <orgblockId>{1,1}</orgblockId> </group> </pre>						
Source	<code><xss:element name="group" type="typeGroup" minOccurs="0" maxOccurs="unbounded" /></code>						

Element typeGroupGetListEvent / listEnd

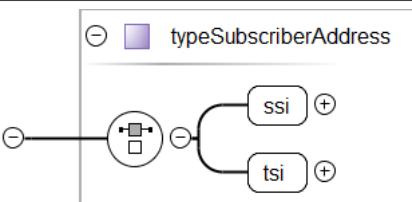
Namespace	DR-GW
-----------	-------

Diagram	
Type	xs:boolean
Properties	content: simple
Source	<xs:element name="listEnd" type="xs:boolean" />

Element interfaceGroup / getRadioMembersEvent

Namespace	DR-GW
Diagram	
Type	typeGroupGetRadioMembersEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupGetRadioMembersEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , radio* , listEnd
Children	group, listEnd, radio, requestId, result
Instance	<pre><getRadioMembersEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <radio>{0,unbounded}</radio> <listEnd>{1,1}</listEnd> </getRadioMembersEvent></pre>
Source	<xs:element name="getRadioMembersEvent" type="typeGroupGetRadioMembersEvent" />

Element typeGroupGetRadioMembersEvent / group

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi

Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<pre><xss:element name="group" type="typeSubscriberAddress" /></pre>

Element typeGroupGetRadioMembersEvent / radio

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	<p>content: complex</p> <p>minOccurs: 0</p> <p>maxOccurs: unbounded</p>
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<pre><xss:element name="radio" type="typeSubscriberAddress" minOccurs="0" maxOccurs="unbounded" /></pre>

Element typeGroupGetRadioMembersEvent / listEnd

Namespace	DR-GW
Diagram	
Type	xs:boolean
Properties	content: simple
Source	<pre><xss:element name="listEnd" type="xs:boolean" /></pre>

Element interfaceGroup / getAppMembersEvent

Namespace	DR-GW
Diagram	
Type	typeGroupGetAppMembersEvent

Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeGroupGetAppMembersEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , app* , listEnd
Children	app, listEnd, requestId, result
Instance	<pre><getAppMembersEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <app>{0,unbounded}</app> <listEnd>{1,1}</listEnd> </getAppMembersEvent></pre>
Source	<code><xss:element name="getAppMembersEvent" type="typeGroupGetAppMembersEvent" /></code>

Element typeGroupGetAppMembersEvent / app

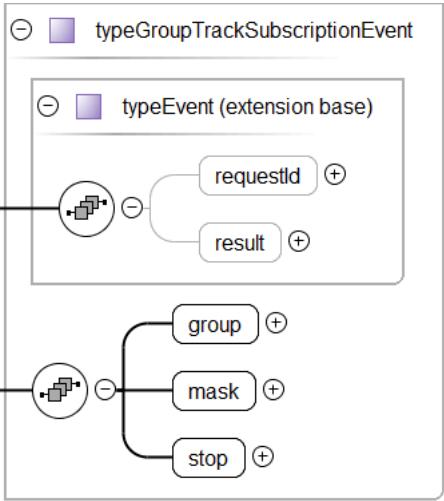
Namespace	DR-GW						
Diagram							
Type	typeSubscriberAddress						
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>maxOccurs:</td> <td>unbounded</td> </tr> </table>	content:	complex	minOccurs:	0	maxOccurs:	unbounded
content:	complex						
minOccurs:	0						
maxOccurs:	unbounded						
Model	ssi tsi						
Children	ssi, tsi						
Instance	<pre><app xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </app></pre>						
Source	<code><xss:element name="app" type="typeSubscriberAddress" minOccurs="0" maxOccurs="unbounded" /></code>						

Element typeGroupGetAppMembersEvent / listEnd

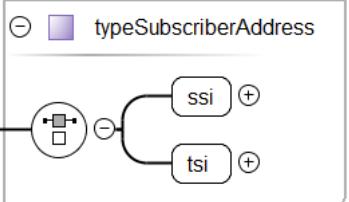
Namespace	DR-GW
Diagram	
Type	xs:boolean
Properties	content: simple
Source	<code><xss:element name="listEnd" type="xs:boolean" /></code>

Element interfaceGroup / trackSubscriptionEvent

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeGroupTrackSubscriptionEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupTrackSubscriptionEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , mask , stop
Children	group, mask, requestId, result, stop
Instance	<pre><trackSubscriptionEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <mask>{1,1}</mask> <stop>{1,1}</stop> </trackSubscriptionEvent></pre>
Source	<code><xss:element name="trackSubscriptionEvent" type="typeGroupTrackSubscriptionEvent" /></code>

Element typeGroupTrackSubscriptionEvent / group

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress" /></code>

Element typeGroupTrackSubscriptionEvent / mask

Namespace	DR-GW
Diagram	 <p>Bit mask of one or more typeGroupTrackingMaskValues using bitwise OR.</p>

Type	typeGroupTrackingMask
Properties	content: simple
Source	<xs:element name="mask" type="typeGroupTrackingMask" />

Element typeGroupTrackSubscriptionEvent / stop

Namespace	DR-GW
Diagram	<p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<xs:element name="stop" type="xs:boolean" />

Element interfaceGroup / radioMemberEvent

Namespace	DR-GW
Diagram	
Type	typeGroupRadioMemberEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupRadioMemberEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , radio , delete
Children	delete, group, radio, requestId, result
Instance	<pre> <radioMemberEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <radio>{1,1}</radio> <delete>{1,1}</delete> </radioMemberEvent> </pre>
Source	<xs:element name="radioMemberEvent" type="typeGroupRadioMemberEvent" />

Element typeGroupRadioMemberEvent / group

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress"/></code>

Element typeGroupRadioMemberEvent / radio

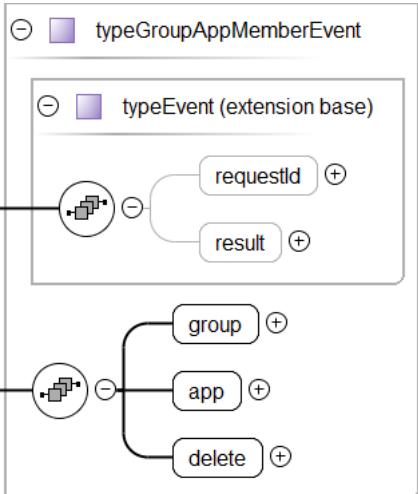
Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress"/></code>

Element typeGroupRadioMemberEvent / delete

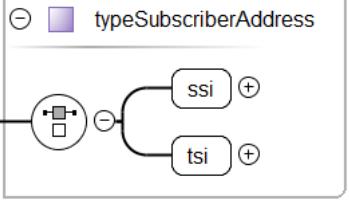
Namespace	DR-GW
Diagram	 A callout box points to the xs:boolean element with the text: "Built-in primitive type. It defines the boolean values true and false.".
Type	xs:boolean
Properties	content: simple
Source	<code><xss:element name="delete" type="xs:boolean"/></code>

Element interfaceGroup / appMemberEvent

Namespace	DR-GW
-----------	-------

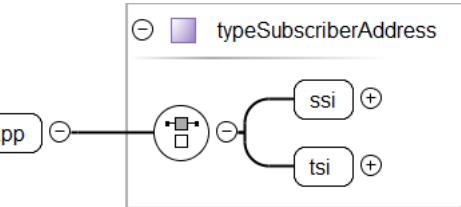
Diagram	
Type	typeGroupAppMemberEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupAppMemberEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , app , delete
Children	app, delete, group, requestId, result
Instance	<pre><appMemberEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <app>{1,1}</app> <delete>{1,1}</delete> </appMemberEvent></pre>
Source	<code><xs:element name="appMemberEvent" type="typeGroupAppMemberEvent" /></code>

Element typeGroupAppMemberEvent / group

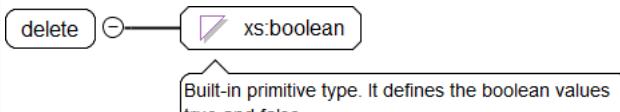
Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xs:element name="group" type="typeSubscriberAddress" /></code>

Element typeGroupAppMemberEvent / app

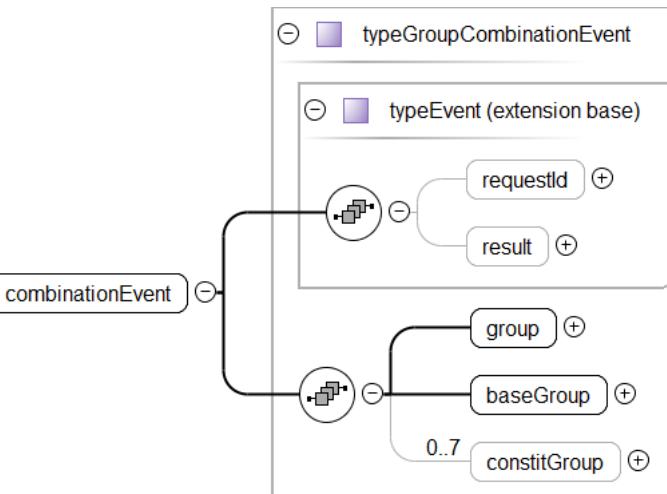
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<app xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </app>
Source	<xs:element name="app" type="typeSubscriberAddress"/>

Element typeGroupAppMemberEvent / delete

Namespace	DR-GW
Diagram	 A callout box points to xs:boolean with the text: Built-in primitive type. It defines the boolean values true and false.
Type	xs:boolean
Properties	content: simple
Source	<xs:element name="delete" type="xs:boolean"/>

Element interfaceGroup / combinationEvent

Namespace	DR-GW
Diagram	
Type	typeGroupCombinationEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupCombinationEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , baseGroup , constitGroup{0,7}
Children	baseGroup, constitGroup, group, requestId, result
Instance	<combinationEvent xmlns="DR-GW">

	<pre> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <baseGroup>{1,1}</baseGroup> <constitGroup>{0,7}</constitGroup> </combinationEvent> </pre>
Source	<xss:element name="combinationEvent" type="typeGroupCombinationEvent" />

Element typeGroupCombinationEvent / group

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group> </pre>
Source	<xss:element name="group" type="typeSubscriberAddress" />

Element typeGroupCombinationEvent / baseGroup

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <baseGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </baseGroup> </pre>
Source	<xss:element name="baseGroup" type="typeSubscriberAddress" />

Element typeGroupCombinationEvent / constitGroup

Namespace	DR-GW
Diagram	

Type	typeSubscriberAddress
Properties	<p>content: complex</p> <p>minOccurs: 0</p> <p>maxOccurs: 7</p>
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><constitGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </constitGroup></pre>
Source	<pre><xss:element name="constitGroup" type="typeSubscriberAddress" minOccurs="0" maxOccurs="7" /></pre>

Element interfaceGroup / addRadioMemberEvent

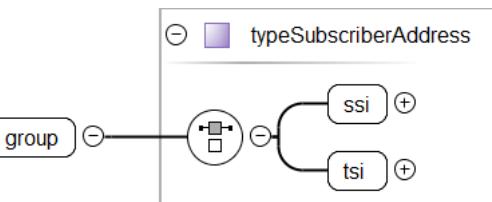
Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeGroupAddRadioMemberEvent typeGroupAddRadioMemberEvent { <<extension base>> attribute requestId attribute result attribute radio attribute group } </pre>
Type	typeGroupAddRadioMemberEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupAddRadioMemberEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , group
Children	group, radio, requestId, result
Instance	<pre><addRadioMemberEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <group>{1,1}</group> </addRadioMemberEvent></pre>
Source	<pre><xss:element name="addRadioMemberEvent" type="typeGroupAddRadioMemberEvent" /></pre>

Element typeGroupAddRadioMemberEvent / radio

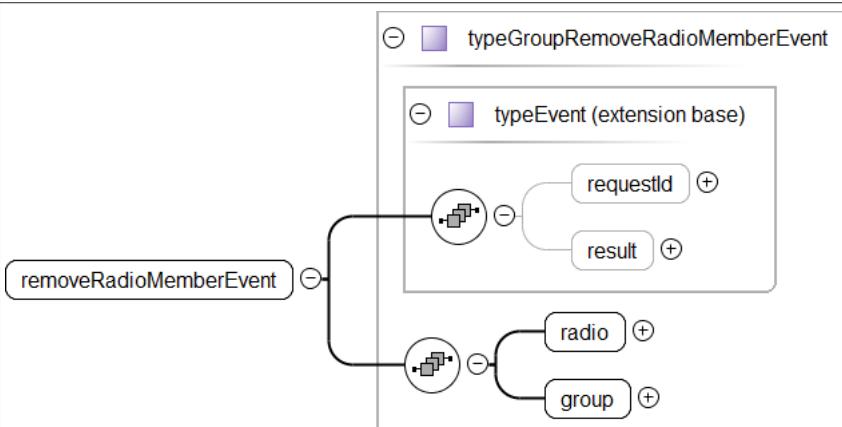
Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeSubscriberAddress typeSubscriberAddress { <<extension base>> attribute ssi attribute tsi } </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi

Children	ssi, tsi
Instance	<radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio>
Source	<xss:element name="radio" type="typeSubscriberAddress"/>

Element typeGroupAddRadioMemberEvent / group

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group>
Source	<xss:element name="group" type="typeSubscriberAddress"/>

Element interfaceGroup / removeRadioMemberEvent

Namespace	DR-GW
Diagram	
Type	typeGroupRemoveRadioMemberEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupRemoveRadioMemberEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , group
Children	group, radio, requestId, result
Instance	<removeRadioMemberEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <group>{1,1}</group> </removeRadioMemberEvent>
Source	<xss:element name="removeRadioMemberEvent" type="typeGroupRemoveRadioMemberEvent"/>

Element typeGroupRemoveRadioMemberEvent / radio

Namespace	DR-GW
Diagram	<pre> classDiagram class radio class typeSubscriberAddress { <<radio>> <<ssi>> <<tsi>> } radio --> typeSubscriberAddress typeSubscriberAddress --> ssi typeSubscriberAddress --> tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio>
Source	<xss:element name="radio" type="typeSubscriberAddress"/>

Element typeGroupRemoveRadioMemberEvent / group

Namespace	DR-GW
Diagram	<pre> classDiagram class group class typeSubscriberAddress { <<group>> <<ssi>> <<tsi>> } group --> typeSubscriberAddress typeSubscriberAddress --> ssi typeSubscriberAddress --> tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group>
Source	<xss:element name="group" type="typeSubscriberAddress"/>

Element interfaceGroup / addCombinationEvent

Namespace	DR-GW
Diagram	<pre> classDiagram class addCombinationEvent class typeGroupAddCombinationEvent { <<addCombinationEvent>> <<typeEvent (extension base)>> <<requestId>> <<result>> <<group>> <<baseGroup>> } addCombinationEvent --> typeGroupAddCombinationEvent typeGroupAddCombinationEvent --> requestId typeGroupAddCombinationEvent --> result typeGroupAddCombinationEvent --> group typeGroupAddCombinationEvent --> baseGroup </pre>

Type	typeGroupAddCombinationEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupAddCombinationEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , baseGroup
Children	baseGroup, group, requestId, result
Instance	<pre><addCombinationEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <baseGroup>{1,1}</baseGroup> </addCombinationEvent></pre>
Source	<code><xss:element name="addCombinationEvent" type="typeGroupAddCombinationEvent" /></code>

Element typeGroupAddCombinationEvent / group

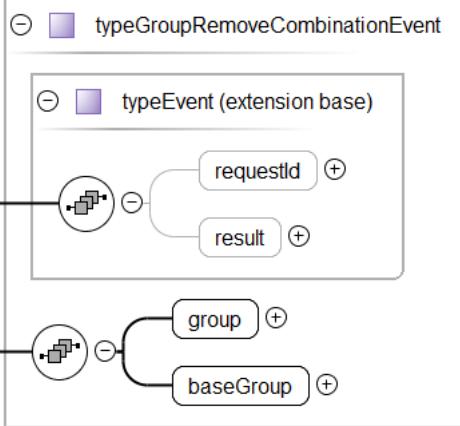
Namespace	DR-GW
Diagram	<pre> classDiagram class group class ssi class tsi group "1..1" *-- "1..1" ssi group "1..1" *-- "1..1" tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress" /></code>

Element typeGroupAddCombinationEvent / baseGroup

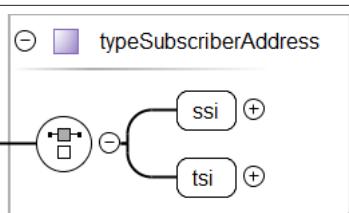
Namespace	DR-GW
Diagram	<pre> classDiagram class baseGroup class ssi class tsi baseGroup "1..1" *-- "1..1" ssi baseGroup "1..1" *-- "1..1" tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><baseGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </baseGroup></pre>
Source	<code><xss:element name="baseGroup" type="typeSubscriberAddress" /></code>

Element interfaceGroup / removeCombinationEvent

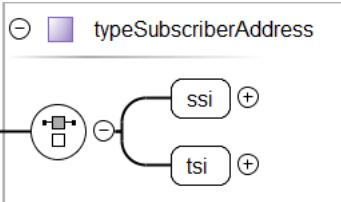
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeGroupRemoveCombinationEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupRemoveCombinationEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , baseGroup
Children	baseGroup, group, requestId, result
Instance	<pre><removeCombinationEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <baseGroup>{1,1}</baseGroup> </removeCombinationEvent></pre>
Source	<code><xss:element name="removeCombinationEvent" type="typeGroupRemoveCombinationEvent" /></code>

Element typeGroupRemoveCombinationEvent / group

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress" /></code>

Element typeGroupRemoveCombinationEvent / baseGroup

Namespace	DR-GW
Diagram	

Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<baseGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </baseGroup>
Source	<xs:element name="baseGroup" type="typeSubscriberAddress" />

Element interfaceGroup / subscribeDataEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeGroupSubscribeDataEvent typeEvent { <<extension base>> requestId result } typeGroupSubscribeDataEvent { <<subscribeDataEvent>> group } </pre>
Type	typeGroupSubscribeDataEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupSubscribeDataEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group
Children	group, requestId, result
Instance	<subscribeDataEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> </subscribeDataEvent>
Source	<xs:element name="subscribeDataEvent" type="typeGroupSubscribeDataEvent" />

Element typeGroupSubscribeDataEvent / group

Namespace	DR-GW
Diagram	<pre> classDiagram typeGroupDataSubscription < -- group typeGroupDataSubscription { <<typeGroupDataSubscription>> addr useSDS useStatus } group { <<group>> --> typeGroupDataSubscription } </pre>
Type	typeGroupDataSubscription
Properties	content: complex
Model	addr , useSDS , useStatus
Children	addr, useSDS, useStatus
Instance	<group xmlns="DR-GW"> <addr>{1,1}</addr>

	<pre><useSDS>{1,1}</useSDS> <useStatus>{1,1}</useStatus> </group></pre>
Source	<code><xss:element name="group" type="typeGroupDataSubscription" /></code>

Element interfaceGroup / event

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeGroupEvent event --> typeGroupEvent typeGroupEvent "0..1" requestId typeGroupEvent "0..1" result typeGroupEvent "1..1" group typeGroupEvent "1..1" delete </pre>
Type	typeGroupEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , group , delete
Children	delete, group, requestId, result
Instance	<pre> <event xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <group>{1,1}</group> <delete>{1,1}</delete> </event> </pre>
Source	<code><xss:element name="event" type="typeGroupEvent" /></code>

Element typeGroupEvent / group

Namespace	DR-GW
Diagram	<pre> typeGroup "0..1" group typeGroup "0..1" addr typeGroup "0..1" alias typeGroup "0..1" orgblockId </pre>
Type	typeGroup
Properties	content: complex
Model	addr , alias , orgblockId
Children	addr, alias, orgblockId
Instance	<pre> <group xmlns="DR-GW"> <addr>{1,1}</addr> <alias>{1,1}</alias> <orgblockId>{1,1}</orgblockId> </group> </pre>
Source	<code><xss:element name="group" type="typeGroup" /></code>

Element typeGroupEvent / delete

Namespace	DR-GW
Diagram	<p>Diagram illustrating the 'delete' element:</p> <pre> graph LR delete([delete]) --> xsBoolean(xs:boolean) </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<code><xs:element name="delete" type="xs:boolean"/></code>

Element drgw / radio

Namespace	DR-GW
Diagram	<p>Diagram illustrating the 'radio' element:</p> <pre> graph LR radio[radio Type interfaceRadio] --> interfaceRadio(interfaceRadio) </pre> <p>DR-GW-Radio. Use to get information about certain radios or to read out the complete radio list the DF-Client has...</p>
Type	interfaceRadio

Properties	content: complex
Model	get getList getGroups track changeOpta enable disable response getEvent getListEvent getGroupsEvent trackSubscriptionEvent trackEvent groupsEvent changeOptaEvent enableDisableEvent event
Children	changeOpta, changeOptaEvent, disable, enable, enableDisableEvent, event, get, getEvent, getGroups, getGroupsEvent, getList, getListEvent, groupsEvent, response, track, trackEvent, trackSubscriptionEvent
Instance	<pre><radio xmlns="DR-GW"> <get>{1,1}</get> <getList>{1,1}</getList> <getGroups>{1,1}</getGroups> <track>{1,1}</track> <changeOpta>{1,1}</changeOpta> <enable>{1,1}</enable> <disable>{1,1}</disable> <response>{1,1}</response> <getEvent>{1,1}</getEvent> <getListEvent>{1,1}</getListEvent> <getGroupsEvent>{1,1}</getGroupsEvent> <trackSubscriptionEvent>{1,1}</trackSubscriptionEvent> <trackEvent>{1,1}</trackEvent> <groupsEvent>{1,1}</groupsEvent> <changeOptaEvent>{1,1}</changeOptaEvent> <enableDisableEvent>{1,1}</enableDisableEvent> <event>{1,1}</event> </radio></pre>
Source	<code><xss:element name="radio" type="interfaceRadio"/></code>

Element interfaceRadio / get

Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioGet < -- typeRequest typeRadioGet "1..1" -- "1..1" requestId typeRadioGet "1..1" -- "1..1" radio </pre>
Type	typeRadioGet
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioGet
Properties	content: complex
Model	requestId , radio
Children	radio, requestId
Instance	<pre><get xmlns="DR-GW"> <requestId>{1,1}</requestId> <radio>{1,1}</radio> </get></pre>
Source	<code><xss:element name="get" type="typeRadioGet"/></code>

Element typeRadioGet / radio

Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress "1..1" -- "1..1" radio typeSubscriberAddress "1..1" -- "1..1" ssi typeSubscriberAddress "1..1" -- "1..1" tsi </pre>

Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio>
Source	<xss:element name="radio" type="typeSubscriberAddress"/>

Element interfaceRadio / getList

Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioGetList < -- typeRequest typeRadioGetList "1..1" --> getList typeRadioGetList "1..1" --> orgblockId typeRadioGetList "1..1" --> requestID </pre>
Type	typeRadioGetList
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioGetList
Properties	content: complex
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Instance	<getList xmlns="DR-GW"> <requestId>{1,1}</requestId> <orgblockId>{0,1}</orgblockId> </getList>
Source	<xss:element name="getList" type="typeRadioGetList"/>

Element typeRadioGetList / orgblockID

Namespace	DR-GW
Diagram	<pre> classDiagram typeOrganisationBlockId "1..1" --> orgblockID typeOrganisationBlockId "1..1" --> orgblockIDSimple </pre>
Type	typeOrganisationBlockId
Properties	<p>content: complex</p> <p>minOccurs: 0</p>
Model	orgblockID orgblockIDSimple
Children	orgblockID, orgblockIDSimple
Instance	<orgblockID xmlns="DR-GW"> <orgblockID>{1,1}</orgblockID> <orgblockIDSimple>{1,1}</orgblockIDSimple> </orgblockID>
Source	<xss:element name="orgblockID" type="typeOrganisationBlockId" minOccurs="0"/>

Element interfaceRadio / getGroups

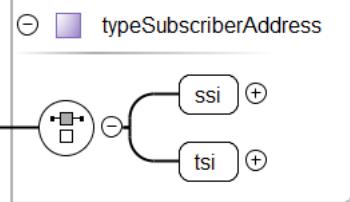
Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioGetGroups < -- typeRequest typeRadioGetGroups "0..1" --> "1..1" requestId typeRadioGetGroups "0..1" --> "1..1" radio </pre>
Type	typeRadioGetGroups
Type hierarchy	<ul style="list-style-type: none"> • typeRequest <ul style="list-style-type: none"> • typeRadioGetGroups
Properties	content: complex
Model	requestId , radio
Children	radio, requestId
Instance	<pre> <getGroups xmlns="DR-GW"> <requestId>{1,1}</requestId> <radio>{1,1}</radio> </getGroups> </pre>
Source	<xss:element name="getGroups" type="typeRadioGetGroups"/>

Element typeRadioGetGroups / radio

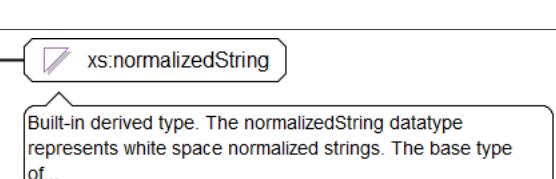
Namespace	DR-GW
Diagram	<pre> classDiagram typeRadio "0..1" --> "1..1" issi typeRadio "0..1" --> "1..1" alias typeRadio "0..1" --> "1..1" orgblockId typeRadio "0..1" --> "1..1" opta </pre>
Type	typeRadio
Properties	content: complex
Model	issi , alias{0,1} , orgblockId{0,1} , opta{0,1}
Children	alias, issi, opta, orgblockId
Instance	<pre> <radio xmlns="DR-GW"> <issi>{1,1}</issi> <alias>{0,1}</alias> <orgblockId>{0,1}</orgblockId> <opta>{0,1}</opta> </radio> </pre>
Source	<xss:element name="radio" type="typeRadio"/>

Element typeRadio / issi

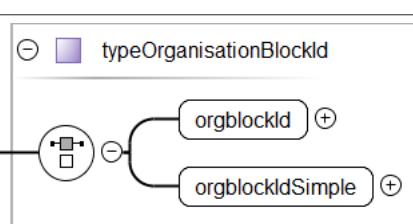
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<issi xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </issi>
Source	<xss:element name="issi" type="typeSubscriberAddress" />

Element typeRadio / alias

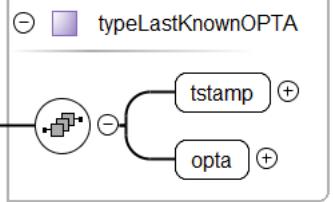
Namespace	DR-GW
Diagram	
Type	xs:normalizedString
Properties	content: simple minOccurs: 0
Source	<xss:element name="alias" type="xs:normalizedString" minOccurs="0" />

Element typeRadio / orgblockId

Namespace	DR-GW
Diagram	
Type	typeOrganisationBlockId
Properties	content: complex minOccurs: 0
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<orgblockId xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <orgblockIdSimple>{1,1}</orgblockIdSimple> </orgblockId>
Source	<xss:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0" />

Element typeRadio / opta

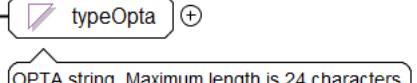
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeLastKnownOPTA
Properties	<p>content: complex</p> <p>minOccurs: 0</p>
Model	tstamp , opta
Children	opta, tstamp
Instance	<pre><opta xmlns="DR-GW"> <tstamp>{1,1}</tstamp> <opta>{1,1}</opta> </opta></pre>
Source	<code><xss:element name="opta" type="typeLastKnownOPTA" minOccurs="0" /></code>

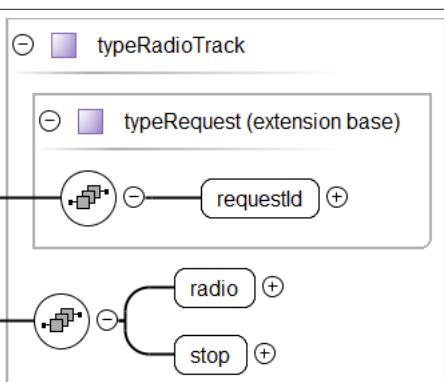
Element typeLastKnownOPTA / tstamp

Namespace	DR-GW
Diagram	 Built-in primitive type. The dateTime datatype represents a specific instant of time.
Type	xs:dateTime
Properties	content: simple
Source	<code><xss:element name="tstamp" type="xs:dateTime" /></code>

Element typeLastKnownOPTA / opta

Namespace	DR-GW
Diagram	 OPTA string. Maximum length is 24 characters.
Type	typeOpta
Properties	content: simple
Facets	maxLength 24
Source	<code><xss:element name="opta" type="typeOpta" /></code>

Element interfaceRadio / track

Namespace	DR-GW
Diagram	

Type	typeRadioTrack
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioTrack
Properties	content: complex
Model	requestId , radio , stop
Children	radio, requestId, stop
Instance	<pre><track xmlns="DR-GW"> <requestId>{1,1}</requestId> <radio>{1,1}</radio> <stop>{1,1}</stop> </track></pre>
Source	<code><xss:element name="track" type="typeRadioTrack" /></code>

Element typeRadioTrack / radio

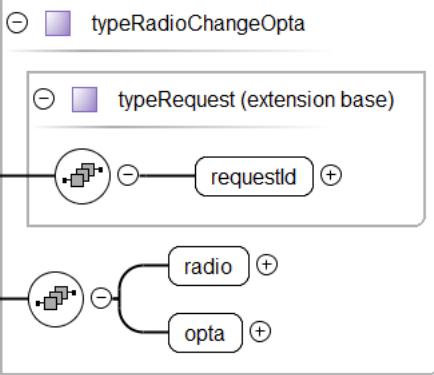
Namespace	DR-GW
Diagram	<pre> classDiagram class typeRadioTrack { requestId radio stop } class radio { ssi tsi } class ssi class tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress" /></code>

Element typeRadioTrack / stop

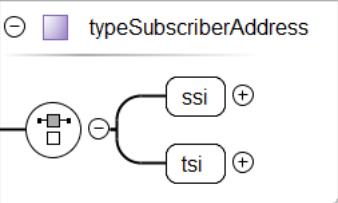
Namespace	DR-GW
Diagram	<pre> classDiagram class typeRadioTrack { requestId radio stop } class stop { xs:boolean } class xs:boolean </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<code><xss:element name="stop" type="xs:boolean" /></code>

Element interfaceRadio / changeOpta

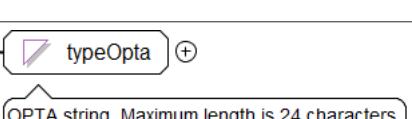
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeRadioChangeOpta
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioChangeOpta
Properties	content: complex
Model	requestId , radio , opta
Children	opta, radio, requestId
Instance	<pre><changeOpta xmlns="DR-GW"> <requestIds>{1,1}</requestIds> <radio>{1,1}</radio> <opta>{1,1}</opta> </changeOpta></pre>
Source	<code><xss:element name="changeOpta" type="typeRadioChangeOpta"/></code>

Element typeRadioChangeOpta / radio

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress"/></code>

Element typeRadioChangeOpta / opta

Namespace	DR-GW
Diagram	
Type	typeOpta
Properties	content: simple
Facets	maxLength 24

Source	<code><xss:element name="opta" type="typeOpta" /></code>
--------	--

Element interfaceRadio / enable

Namespace	DR-GW
Diagram	<pre> classDiagram class typeRadioEnable { <<typeRequest (extension base)>> <<radio>> <<enable>> } typeRadioEnable "1..1" -- "1..1" typeRequest typeRequest "1..1" -- "1..1" radio radio "1..1" -- "1..1" enable radio "1..1" -- "1..1" reason radio "1..1" -- "1..1" radio note over radio: This method is used to Enable the radio terminal over the air. </pre>
Type	typeRadioEnable
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioEnable
Properties	content: complex
Model	requestId , radio , radio , reason{0,1} , enable
Children	enable, radio, reason, requestId
Instance	<pre> <enable xmlns="DR-GW"> <requestId>{1,1}</requestId> <radio>{1,1}</radio> <radio>{1,1}</radio> <reason>{0,1}</reason> <enable>{1,1}</enable> </enable> </pre>
Source	<code><xss:element name="enable" type="typeRadioEnable" /></code>

Element typeRadioEnable / radio

Namespace	DR-GW
Diagram	<pre> classDiagram class typeSubscriberAddress { <<radio>> <<ssi>> <<tsi>> } radio "1..1" -- "1..1" ssi radio "1..1" -- "1..1" tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio> </pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress" /></code>

Element typeRadioEnable / reason

Namespace	DR-GW
Diagram	<p>Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to...</p>
Type	xs:unsignedByte
Properties	content: simple minOccurs: 0
Source	<code><xss:element name="reason" type="xs:unsignedByte" minOccurs="0" /></code>

Element typeRadioEnable / enable

Namespace	DR-GW
Diagram	<p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<code><xss:element name="enable" type="xs:boolean" /></code>

Element interfaceRadio / disable

Namespace	DR-GW
Diagram	<p>This method is used to disable the radio terminal over the air. If no reason is supplied, then the DF-Gateway sets the...</p>
Type	typeRadioDisable
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioDisable
Properties	content: complex
Model	requestId , radio , radio , reason{0,1} , enable
Children	enable, radio, reason, requestId

Instance	<pre><disable xmlns="DR-GW"> <requestId>{1,1}</requestId> <radio>{1,1}</radio> <radio>{1,1}</radio> <reason>{0,1}</reason> <enable>{1,1}</enable> </disable></pre>
Source	<pre><xss:element name="disable" type="typeRadioDisable" /></pre>

Element typeRadioDisable / radio

Namespace	DR-GW
Diagram	<pre> classDiagram class typeRadioDisable { <<radio>> --> radio radio --> ssi radio --> tsi } </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<pre><xss:element name="radio" type="typeSubscriberAddress" /></pre>

Element typeRadioDisable / reason

Namespace	DR-GW
Diagram	<pre> classDiagram class typeRadioDisable { <<reason>> --> xs:unsignedByte } </pre> <p>Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to...</p>
Type	xs:unsignedByte
Properties	content: simple minOccurs: 0
Source	<pre><xss:element name="reason" type="xs:unsignedByte" minOccurs="0" /></pre>

Element typeRadioDisable / enable

Namespace	DR-GW
Diagram	<pre> classDiagram class typeRadioDisable { <<enable>> --> xs:boolean } </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<pre><xss:element name="enable" type="xs:boolean" /></pre>

Element interfaceRadio / response

Namespace	DR-GW
-----------	-------

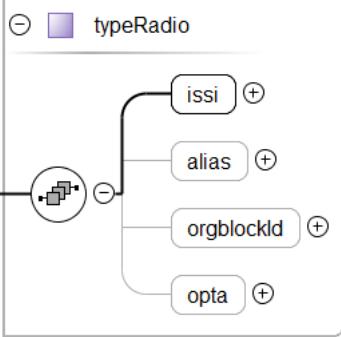
Diagram	<p>typeResponse</p> <p>response</p> <p>requestId</p> <p>result</p> <p>Response contains result of execution of any method.</p>
Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<pre><response xmlns="DR-GW"> <requestIds>{1,1}</requestIds> <result>{1,1}</result> </response></pre>
Source	<code><xss:element name="response" type="typeResponse" /></code>

Element interfaceRadio / getEvent

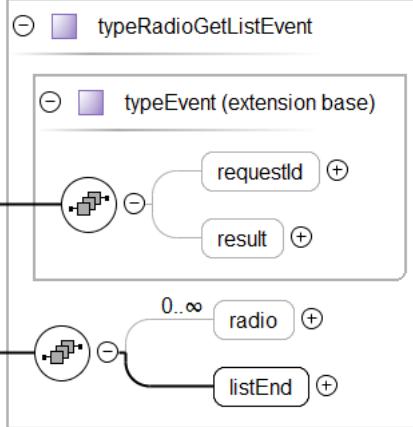
Namespace	DR-GW
Diagram	<p>typeRadioGetEvent</p> <p>getEvent</p> <p>requestId</p> <p>result</p> <p>radio</p> <p>typeEvent (extension base)</p> <p>radio</p> <p>radio is an extension of typeEvent</p>
Type	typeRadioGetEvent
Type hierarchy	<ul style="list-style-type: none"> typeEvent typeRadioGetEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio
Children	radio, requestId, result
Instance	<pre><getEvent xmlns="DR-GW"> <requestIds>{0,1}</requestIds> <result>{0,1}</result> <radio>{1,1}</radio> </getEvent></pre>
Source	<code><xss:element name="getEvent" type="typeRadioGetEvent" /></code>

Element typeRadioGetEvent / radio

Namespace	DR-GW
-----------	-------

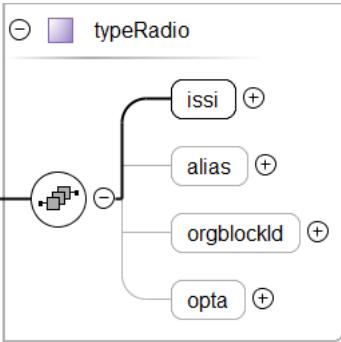
Diagram	
Type	typeRadio
Properties	content: complex
Model	issi , alias{0,1} , orgblockId{0,1} , opta{0,1}
Children	alias, issi, opta, orgblockId
Instance	<pre><radio xmlns="DR-GW"> <issi>{1,1}</issi> <alias>{0,1}</alias> <orgblockId>{0,1}</orgblockId> <opta>{0,1}</opta> </radio></pre>
Source	<code><xss:element name="radio" type="typeRadio"/></code>

Element interfaceRadio / getListEvent

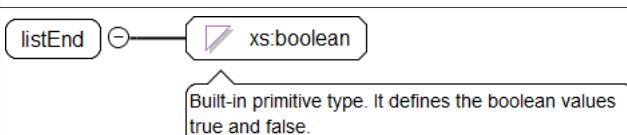
Namespace	DR-GW
Diagram	
Type	typeRadioGetListEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioGetListEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio* , listEnd
Children	listEnd, radio, requestId, result
Instance	<pre><getListEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{0,unbounded}</radio> <listEnd>{1,1}</listEnd> </getListEvent></pre>
Source	<code><xss:element name="getListEvent" type="typeRadioGetListEvent"/></code>

Element typeRadioGetListEvent / radio

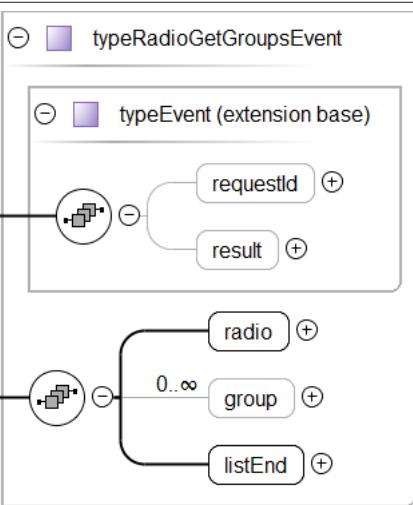
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeRadio
Properties	<p>content: complex</p> <p>minOccurs: 0</p> <p>maxOccurs: unbounded</p>
Model	issi , alias{0,1} , orgblockId{0,1} , opta{0,1}
Children	alias, issi, opta, orgblockId
Instance	<pre><radio xmlns="DR-GW"> <issi>{1,1}</issi> <alias>{0,1}</alias> <orgblockId>{0,1}</orgblockId> <opta>{0,1}</opta> </radio></pre>
Source	<code><x:element name="radio" type="typeRadio" minOccurs="0" maxOccurs="unbounded"/></code>

Element typeRadioGetListEvent / listEnd

Namespace	DR-GW
Diagram	
Type	xs:boolean
Properties	content: simple
Source	<code><x:element name="listEnd" type="xs:boolean"/></code>

Element interfaceRadio / getGroupsEvent

Namespace	DR-GW
Diagram	
Type	typeRadioGetGroupsEvent

Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeRadioGetGroupsEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , group* , listEnd
Children	group, listEnd, radio, requestId, result
Instance	<pre><getGroupsEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <group>{0,unbounded}</group> <listEnd>{1,1}</listEnd> </getGroupsEvent></pre>
Source	<code><xss:element name="getGroupsEvent" type="typeRadioGetGroupsEvent" /></code>

Element typeRadioGetGroupsEvent / radio

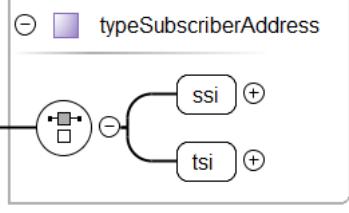
Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress" /></code>

Element typeRadioGetGroupsEvent / group

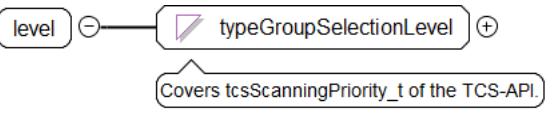
Namespace	DR-GW						
Diagram							
Type	typeRadioGroupSelection						
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> <tr> <td>maxOccurs:</td> <td>unbounded</td> </tr> </table>	content:	complex	minOccurs:	0	maxOccurs:	unbounded
content:	complex						
minOccurs:	0						
maxOccurs:	unbounded						
Model	group , level						
Children	group, level						
Instance	<pre><group xmlns="DR-GW"> <group>{1,1}</group> <level>{1,1}</level> </group></pre>						
Source	<code><xss:element name="group" type="typeRadioGroupSelection" minOccurs="0" maxOccurs="unbounded" /></code>						

Element typeRadioGroupSelection / group

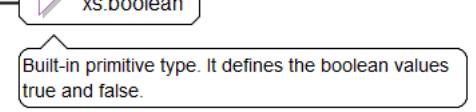
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress"/></code>

Element typeRadioGroupSelection / level

Namespace	DR-GW
Diagram	
Type	typeGroupSelectionLevel
Properties	content: simple
Facets	enumeration notScanned enumeration low enumeration normal enumeration selected enumeration high enumeration background
Source	<code><xss:element name="level" type="typeGroupSelectionLevel"/></code>

Element typeRadioGetGroupsEvent / listEnd

Namespace	DR-GW
Diagram	
Type	xs:boolean
Properties	content: simple
Source	<code><xss:element name="listEnd" type="xs:boolean"/></code>

Element interfaceRadio / trackSubscriptionEvent

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeRadioTrackSubscriptionEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioTrackSubscriptionEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , stop
Children	radio, requestId, result, stop
Instance	<pre><trackSubscriptionEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <stop>{1,1}</stop> </trackSubscriptionEvent></pre>
Source	<code><xss:element name="trackSubscriptionEvent" type="typeRadioTrackSubscriptionEvent" /></code>

Element typeRadioTrackSubscriptionEvent / radio

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress" /></code>

Element typeRadioTrackSubscriptionEvent / stop

Namespace	DR-GW
Diagram	
Type	xs:boolean

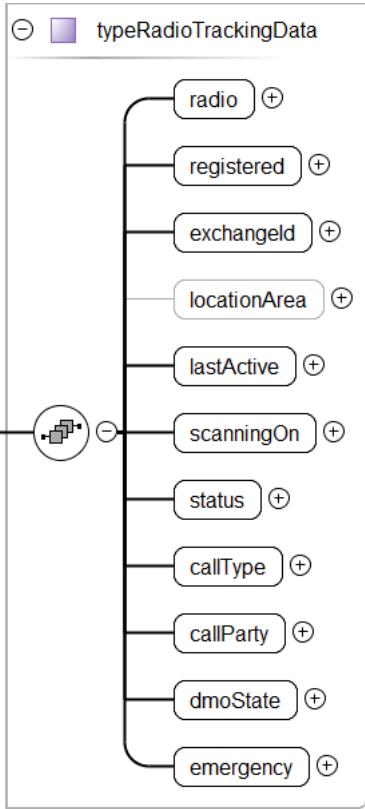
Properties	content: simple
Source	<xs:element name="stop" type="xs:boolean"/>

Element interfaceRadio / trackEvent

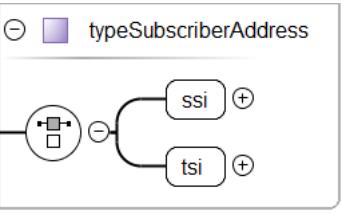
Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioTrackEvent < -- typeEvent typeRadioTrackEvent "0..1" --> "1..1" trackingData : trackingData typeEvent "0..1" --> "1..1" requestId : requestId typeEvent "0..1" --> "1..1" result : result </pre>
Type	typeRadioTrackEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioTrackEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , trackingData
Children	requestId, result, trackingData
Instance	<pre> <trackEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <trackingData>{1,1}</trackingData> </trackEvent> </pre>
Source	<xs:element name="trackEvent" type="typeRadioTrackEvent"/>

Element typeRadioTrackEvent / trackingData

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeRadioTrackingData
Properties	content: complex
Model	radio , registered , exchangeId , locationArea{0,1} , lastActive , scanningOn , status , callType , callParty , dmoState , emergency
Children	callParty, callType, dmoState, emergency, exchangeId, lastActive, locationArea, radio, registered, scanningOn, status
Instance	<pre><trackingData xmlns="DR-GW"> <radio>{1,1}</radio> <registered>{1,1}</registered> <exchangeId>{1,1}</exchangeId> <locationArea>{0,1}</locationArea> <lastActive>{1,1}</lastActive> <scanningOn>{1,1}</scanningOn> <status>{1,1}</status> <callType>{1,1}</callType> <callParty>{1,1}</callParty> <dmoState>{1,1}</dmoState> <emergency>{1,1}</emergency> </trackingData></pre>
Source	<code><xs:element name="trackingData" type="typeRadioTrackingData"/></code>

Element typeRadioTrackingData / radio

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi

Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<pre><xs:element name="radio" type="typeSubscriberAddress" /></pre>

Element typeRadioTrackingData / registered

Namespace	DR-GW
Diagram	<pre>graph LR; registered --> xsboolean[xs:boolean];</pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<pre><xs:element name="registered" type="xs:boolean" /></pre>

Element typeRadioTrackingData / exchangeId

Namespace	DR-GW
Diagram	<pre>graph LR; exchangeId --> xsunsignedLong[xs:unsignedLong];</pre> <p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>
Type	xs:unsignedLong
Properties	content: simple
Source	<pre><xs:element name="exchangeId" type="xs:unsignedLong" /></pre>

Element typeRadioTrackingData / locationArea

Namespace	DR-GW
Diagram	<pre>graph LR; locationArea --> xsunsignedShort[xs:unsignedShort];</pre> <p>Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...</p>
Type	xs:unsignedShort
Properties	content: simple minOccurs: 0
Source	<pre><xs:element name="locationArea" type="xs:unsignedShort" minOccurs="0" /></pre>

Element typeRadioTrackingData / lastActive

Namespace	DR-GW
Diagram	<pre>graph LR; lastActive --> xsdateTime[xs:dateTime];</pre> <p>Built-in primitive type. The dateTime datatype represents a specific instant of time.</p>
Type	xs:dateTime
Properties	content: simple
Source	<pre><xs:element name="lastActive" type="xs:dateTime" /></pre>

Element typeRadioTrackingData / scanningOn

Namespace	DR-GW
Diagram	<p>The diagram shows a rounded rectangle labeled "scanningOn" connected by a line to a purple icon representing "xs:boolean". A callout box below the connection states: "Built-in primitive type. It defines the boolean values true and false."</p>
Type	xs:boolean
Properties	content: simple
Source	<xs:element name="scanningOn" type="xs:boolean" />

Element typeRadioTrackingData / status

Namespace	DR-GW
Diagram	<p>The diagram shows a rounded rectangle labeled "status" connected by a line to a purple icon representing "typeStatusIndicator". Inside "typeStatusIndicator", there is a "value" node and a "time" node, both with plus signs indicating they are children. A callout box above the connection states: "typeStatusIndicator".</p>
Type	typeStatusIndicator
Properties	content: complex
Model	value , time
Children	time, value
Instance	<status xmlns="DR-GW"> <value>{1,1}</value> <time>{1,1}</time> </status>
Source	<xs:element name="status" type="typeStatusIndicator" />

Element typeStatusIndicator / value

Namespace	DR-GW
Diagram	<p>The diagram shows a rounded rectangle labeled "value" connected by a line to a purple icon representing "xs:unsignedLong". A callout box below the connection states: "Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of..."</p>
Type	xs:unsignedLong
Properties	content: simple
Source	<xs:element name="value" type="xs:unsignedLong" />

Element typeStatusIndicator / time

Namespace	DR-GW
Diagram	<p>The diagram shows a rounded rectangle labeled "time" connected by a line to a purple icon representing "xs:dateTime". A callout box below the connection states: "Built-in primitive type. The dateTime datatype represents a specific instant of time."</p>
Type	xs:dateTime
Properties	content: simple
Source	<xs:element name="time" type="xs:dateTime" />

Element typeRadioTrackingData / callType

Namespace	DR-GW						
Diagram	<pre> graph LR callType --> typeCallType typeCallType --> p2p typeCallType --> p2mp typeCallType --> bcast </pre> <p>Call type attribute. Choices are Point2Point, Point2MultiPoint or Broadcast.</p>						
Type	typeCallType						
Properties	content: simple						
Facets	<table> <tr> <td>enumeration</td> <td>p2p</td> </tr> <tr> <td>enumeration</td> <td>p2mp</td> </tr> <tr> <td>enumeration</td> <td>bcast</td> </tr> </table>	enumeration	p2p	enumeration	p2mp	enumeration	bcast
enumeration	p2p						
enumeration	p2mp						
enumeration	bcast						
Source	<code><xss:element name="callType" type="typeCallType" /></code>						

Element typeRadioTrackingData / callParty

Namespace	DR-GW
Diagram	<pre> graph LR callParty --> typeSubscriberAddress typeSubscriberAddress --> ssi typeSubscriberAddress --> tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<code><callParty xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </callParty></code>
Source	<code><xss:element name="callParty" type="typeSubscriberAddress" /></code>

Element typeRadioTrackingData / dmoState

Namespace	DR-GW
Diagram	<pre> graph LR dmoState --> xsboolean </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>
Type	xs:boolean
Properties	content: simple
Source	<code><xss:element name="dmoState" type="xs:boolean" /></code>

Element typeRadioTrackingData / emergency

Namespace	DR-GW
Diagram	<pre> graph LR emergency --> xsboolean </pre> <p>Built-in primitive type. It defines the boolean values true and false.</p>

Type	xs:boolean
Properties	content: simple
Source	<xss:element name="emergency" type="xs:boolean"/>

Element interfaceRadio / groupsEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioGroupsEvent < -- typeEvent typeEvent < -- groupsEvent groupsEvent < -- radio radio < -- group radio < -- deletedGroup </pre>
Type	typeRadioGroupsEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioGroupsEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , (group+ deletedGroup)
Children	deletedGroup, group, radio, requestId, result
Instance	<groupsEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <group>{1,unbounded}</group> <deletedGroup>{1,1}</deletedGroup> </groupsEvent>
Source	<xss:element name="groupsEvent" type="typeRadioGroupsEvent"/>

Element typeRadioGroupsEvent / radio

Namespace	DR-GW
Diagram	<pre> typeRadioGroupsEvent < -- typeSubscriberAddress typeSubscriberAddress < -- radio radio < -- ssi radio < -- tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio>
Source	<xss:element name="radio" type="typeSubscriberAddress"/>

Element typeRadioGroupsEvent / group

Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioGroupSelection < -- group typeRadioGroupSelection < -- level typeRadioGroupSelection < -- group </pre>
Type	typeRadioGroupSelection
Properties	content: complex maxOccurs: unbounded
Model	group , level
Children	group, level
Instance	<pre> <group xmlns="DR-GW"> <group>{1,1}</group> <level>{1,1}</level> </group> </pre>
Source	<pre> <xss:element name="group" type="typeRadioGroupSelection" maxOccurs="unbounded" /> </pre>

Element typeRadioGroupsEvent / deletedGroup

Namespace	DR-GW
Diagram	<pre> classDiagram typeSubscriberAddress < -- deletedGroup typeSubscriberAddress < -- ssi typeSubscriberAddress < -- tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <deletedGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </deletedGroup> </pre>
Source	<pre> <xss:element name="deletedGroup" type="typeSubscriberAddress" /> </pre>

Element interfaceRadio / changeOptaEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioChangeOptaEvent < -- changeOptaEvent typeRadioChangeOptaEvent < -- radio typeRadioChangeOptaEvent < -- opta </pre>

Type	typeRadioChangeOptaEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioChangeOptaEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , opta
Children	opta, radio, requestId, result
Instance	<pre><changeOptaEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <opta>{1,1}</opta> </changeOptaEvent></pre>
Source	<code><xss:element name="changeOptaEvent" type="typeRadioChangeOptaEvent" /></code>

Element typeRadioChangeOptaEvent / radio

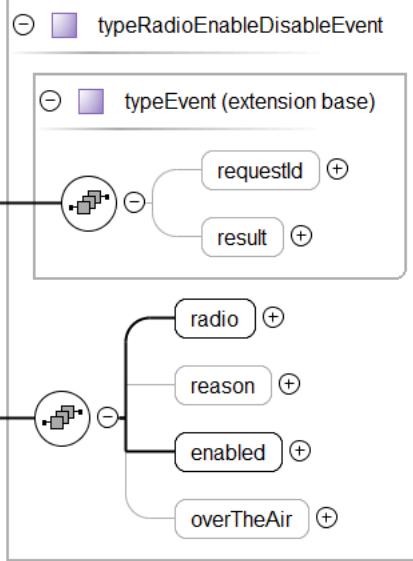
Namespace	DR-GW
Diagram	<pre> classDiagram class typeSubscriberAddress { radio ssi tsi } radio --> typeSubscriberAddress ssi --> typeSubscriberAddress tsi --> typeSubscriberAddress </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xss:element name="radio" type="typeSubscriberAddress" /></code>

Element typeRadioChangeOptaEvent / opta

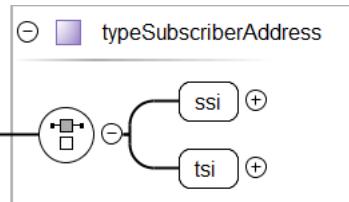
Namespace	DR-GW
Diagram	<pre> classDiagram class typeOpta { opta } opta --> typeOpta </pre> <p>OPTA string. Maximum length is 24 characters.</p>
Type	typeOpta
Properties	content: simple
Facets	maxLength 24
Source	<code><xss:element name="opta" type="typeOpta" /></code>

Element interfaceRadio / enableDisableEvent

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeRadioEnableDisableEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioEnableDisableEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , reason{0,1} , enabled , overTheAir{0,1}
Children	enabled, overTheAir, radio, reason, requestId, result
Instance	<pre><enableDisableEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <reason>{0,1}</reason> <enabled>{1,1}</enabled> <overTheAir>{0,1}</overTheAir> </enableDisableEvent></pre>
Source	<code><xs:element name="enableDisableEvent" type="typeRadioEnableDisableEvent" /></code>

Element typeRadioEnableDisableEvent / radio

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><radio xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </radio></pre>
Source	<code><xs:element name="radio" type="typeSubscriberAddress" /></code>

Element typeRadioEnableDisableEvent / reason

Namespace	DR-GW
-----------	-------

Diagram	<pre> graph LR reason((reason)) -- "0..1" --> xsUnsignedByte[xs:unsignedByte] xsUnsignedByte -- "Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to..." --> info </pre>				
Type	xs:unsignedByte				
Properties	<table border="1"> <tr> <td>content:</td><td>simple</td></tr> <tr> <td>minOccurs:</td><td>0</td></tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xss:element name="reason" type="xs:unsignedByte" minOccurs="0" />				

Element typeRadioEnableDisableEvent / enabled

Namespace	DR-GW				
Diagram	<pre> graph LR enabled((enabled)) -- "0..1" --> xsBoolean[xs:boolean] xsBoolean -- "Built-in primitive type. It defines the boolean values true and false." --> info </pre>				
Type	xs:boolean				
Properties	<table border="1"> <tr> <td>content:</td><td>simple</td></tr> <tr> <td>minOccurs:</td><td>0</td></tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xss:element name="enabled" type="xs:boolean" minOccurs="0" />				

Element typeRadioEnableDisableEvent / overTheAir

Namespace	DR-GW				
Diagram	<pre> graph LR overTheAir((overTheAir)) -- "0..1" --> xsBoolean[xs:boolean] xsBoolean -- "Built-in primitive type. It defines the boolean values true and false." --> info </pre>				
Type	xs:boolean				
Properties	<table border="1"> <tr> <td>content:</td><td>simple</td></tr> <tr> <td>minOccurs:</td><td>0</td></tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xss:element name="overTheAir" type="xs:boolean" minOccurs="0" />				

Element interfaceRadio / event

Namespace	DR-GW
Diagram	<pre> graph LR event((event)) -- "0..1" --> typeRadioEvent[typeRadioEvent] typeRadioEvent --> requestId((requestId)) typeRadioEvent --> result((result)) typeRadioEvent --> radio((radio)) typeRadioEvent --> delete((delete)) </pre>
Type	typeRadioEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent

	<ul style="list-style-type: none"> • typeRadioEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , radio , delete
Children	delete, radio, requestId, result
Instance	<pre><event xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <radio>{1,1}</radio> <delete>{1,1}</delete> </event></pre>
Source	<code><xss:element name="event" type="typeRadioEvent" /></code>

Element typeRadioEvent / radio

Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioEvent { radio issi alias orgblockId opta } radio { <> } </pre>
Type	typeRadio
Properties	content: complex
Model	issi , alias{0,1} , orgblockId{0,1} , opta{0,1}
Children	alias, issi, opta, orgblockId
Instance	<pre><radio xmlns="DR-GW"> <issi>{1,1}</issi> <alias>{0,1}</alias> <orgblockId>{0,1}</orgblockId> <opta>{0,1}</opta> </radio></pre>
Source	<code><xss:element name="radio" type="typeRadio" /></code>

Element typeRadioEvent / delete

Namespace	DR-GW
Diagram	<pre> classDiagram typeRadioEvent { delete xs:boolean } delete { <> } </pre>
Type	xs:boolean
Properties	content: simple
Source	<code><xss:element name="delete" type="xs:boolean" /></code>

Element drgw / app

Namespace	DR-GW
-----------	-------

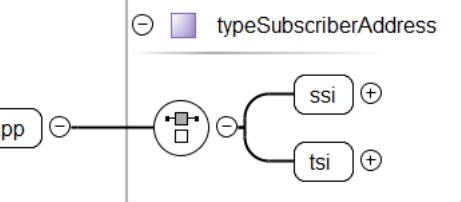
Diagram	<p>DR-GW-Application. Use to get information about certain client application or to read out the complete client...</p>
Type	interfaceApp
Properties	content: complex
Model	get getList response getEvent getListEvent
Children	get, getEvent, getList, getListEvent, response
Instance	<pre><app xmlns="DR-GW"> <get>{1,1}</get> <getList>{1,1}</getList> <response>{1,1}</response> <getEvent>{1,1}</getEvent> <getListEvent>{1,1}</getListEvent> </app></pre>
Source	<code><xss:element name="app" type="interfaceApp" /></code>

Element interfaceApp / get

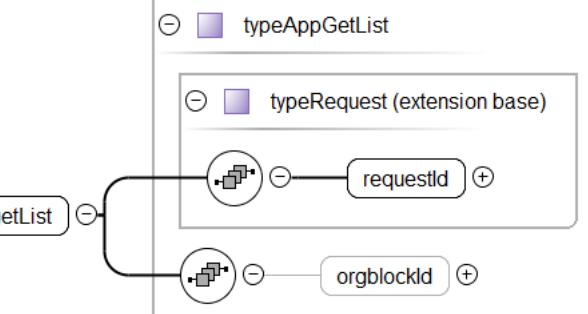
Namespace	DR-GW
Diagram	
Type	typeAppGet
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeAppGet
Properties	content: complex
Model	requestId , app
Children	app, requestId
Instance	<pre><get xmlns="DR-GW"> <requestId>{1,1}</requestId> <app>{1,1}</app> </get></pre>
Source	<code><xss:element name="get" type="typeAppGet" /></code>

Element typeAppGet / app

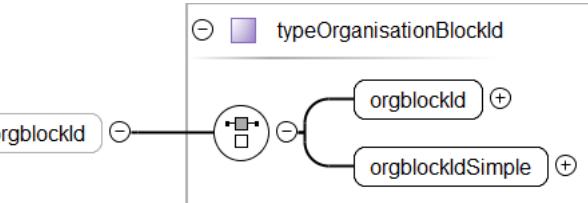
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<app xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </app>
Source	<xss:element name="app" type="typeSubscriberAddress"/>

Element interfaceApp / getList

Namespace	DR-GW
Diagram	
Type	typeAppGetList
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeAppGetList
Properties	content: complex
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Instance	<getList xmlns="DR-GW"> <requestId>{1,1}</requestId> <orgblockId>{0,1}</orgblockId> </getList>
Source	<xss:element name="getList" type="typeAppGetList"/>

Element typeAppGetList / orgblockId

Namespace	DR-GW				
Diagram					
Type	typeOrganisationBlockId				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				

Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<pre><orgblockId xmlns="DR-GW"> <orgblockId>{1,1}</orgblockId> <orgblockIdSimple>{1,1}</orgblockIdSimple> </orgblockId></pre>
Source	<code><xs:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0" /></code>

Element interfaceApp / response

Namespace	DR-GW
Diagram	<p>typeResponse</p> <p>response</p> <p>requestId</p> <p>result</p> <p>Response contains result of execution of any method.</p>
Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<pre><response xmlns="DR-GW"> <requestId>{1,1}</requestId> <result>{1,1}</result> </response></pre>
Source	<code><xs:element name="response" type="typeResponse" /></code>

Element interfaceApp / getEvent

Namespace	DR-GW
Diagram	<p>typeAppGetEvent</p> <p>getEvent</p> <p>requestId</p> <p>result</p> <p>app</p>
Type	typeAppGetEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeAppGetEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , app
Children	app, requestId, result
Instance	<pre><getEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <app>{1,1}</app> </getEvent></pre>

Source

```
<xss:element name="getEvent" type="typeAppGetEvent" />
```

Element typeAppGetEvent / app

Namespace	DR-GW
Diagram	<pre> classDiagram class typeApplication { <<typeApplication>> } class app { <<app>> } typeApplication "1..1" *-- "1..1" app typeApplication "1..1" *-- "1..1" addr typeApplication "1..1" *-- "1..1" alias typeApplication "1..1" *-- "1..1" orgblockId </pre>
Type	typeApplication
Properties	content: complex maxOccurs: 1
Model	addr , alias , orgblockId
Children	addr, alias, orgblockId
Instance	<pre> <app xmlns="DR-GW"> <addr>{1,1}</addr> <alias>{1,1}</alias> <orgblockId>{1,1}</orgblockId> </app> </pre>
Source	<pre><xss:element name="app" type="typeApplication" maxOccurs="1" /></pre>

Element typeApplication / addr

Namespace	DR-GW
Diagram	<pre> classDiagram class typeSubscriberAddress { <<typeSubscriberAddress>> } class addr { <<addr>> } typeSubscriberAddress "1..1" *-- "1..1" addr typeSubscriberAddress "1..1" *-- "1..1" ssi typeSubscriberAddress "1..1" *-- "1..1" tsi </pre>
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre> <addr xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </addr> </pre>
Source	<pre><xss:element name="addr" type="typeSubscriberAddress" /></pre>

Element typeApplication / alias

Namespace	DR-GW
Diagram	<pre> classDiagram class alias { <<alias>> } class xsnormalizedString { <<xsnormalizedString>> } alias --> xsnormalizedString </pre> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>
Type	xsnormalizedString
Properties	content: simple

Source	<code><xss:element name="alias" type="xs:normalizedString"/></code>
--------	---

Element typeApplication / orgblockId

Namespace	DR-GW
Diagram	<pre> classDiagram class typeOrganisationBlockId { <<complex type>> <<content>> orgblockId *--> orgblockId orgblockIdSimple *--> orgblockIdSimple } </pre>
Type	typeOrganisationBlockId
Properties	content: complex
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Instance	<code><orgblockId xmlns="DR-GW"></code> <code> <orgblockId>{1,1}</orgblockId></code> <code> <orgblockIdSimple>{1,1}</orgblockIdSimple></code> <code></orgblockId></code>
Source	<code><xss:element name="orgblockId" type="typeOrganisationBlockId"/></code>

Element interfaceApp / getListEvent

Namespace	DR-GW
Diagram	<pre> classDiagram class typeEvent { <<extension base>> <<content>> requestId result } class typeAppGetListEvent { <<content>> getListEvent *--> getListEvent app *--> listEnd } </pre>
Type	typeAppGetListEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeAppGetListEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , app* , listEnd{0,1}
Children	app, listEnd, requestId, result
Instance	<code><getListEvent xmlns="DR-GW"></code> <code> <requestId>{0,1}</requestId></code> <code> <result>{0,1}</result></code> <code> <app>{0,unbounded}</app></code> <code> <listEnd>{0,1}</listEnd></code> <code></getListEvent></code>
Source	<code><xss:element name="getListEvent" type="typeAppGetListEvent"/></code>

Element typeAppGetListEvent / app

Namespace	DR-GW
-----------	-------

Diagram	
Type	typeApplication
Properties	<p>content: complex</p> <p>minOccurs: 0</p> <p>maxOccurs: unbounded</p>
Model	addr , alias , orgblockId
Children	addr, alias, orgblockId
Instance	<pre><app xmlns="DR-GW"> <addr>{1,1}</addr> <alias>{1,1}</alias> <orgblockId>{1,1}</orgblockId> </app></pre>
Source	<code><xss:element name="app" type="typeApplication" minOccurs="0" maxOccurs="unbounded" /></code>

Element typeAppGetListEvent / listEnd

Namespace	DR-GW
Diagram	
Type	xs:boolean
Properties	<p>content: simple</p> <p>minOccurs: 0</p>
Source	<code><xss:element name="listEnd" type="xs:boolean" minOccurs="0" /></code>

Element drgw / system

Namespace	DR-GW
Diagram	
Type	interfaceSystem
Properties	<p>content: complex</p>
Model	response tetraStatesEvent logEvent event
Children	event, logEvent, response, tetraStatesEvent

Instance	<pre><system xmlns="DR-GW"> <response>{1,1}</response> <tetraStatesEvent>{1,1}</tetraStatesEvent> <logEvent>{1,1}</logEvent> <event>{1,1}</event> </system></pre>
Source	<code><xss:element name="system" type="interfaceSystem" /></code>

Element interfaceSystem / response

Namespace	DR-GW
Diagram	<p>Response contains result of execution of any method.</p>
Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<pre><response xmlns="DR-GW"> <requestId>{1,1}</requestId> <result>{1,1}</result> </response></pre>
Source	<code><xss:element name="response" type="typeResponse" /></code>

Element interfaceSystem / tetraStatesEvent

Namespace	DR-GW
Diagram	<p>Indication of the subsystem state updates. Contains current states of all subsystem known by the TCS.</p>
Type	typeSystemTetraStatesEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSystemTetraStatesEvent
Properties	content: complex

Model	requestId{0,1} , result{0,1} , tcsState{0,1} , dxtState{0,1} , cddconnectionState{0,1} , cddserverState{0,1}
Children	cddconnectionState, cddserverState, dxtState, requestId, result, tcsState
Instance	<pre><tetraStatesEvent xmlns="DR-GW"> <requestIds>{0,1}</requestId> <result>{0,1}</result> <tcsState>{0,1}</tcsState> <dxtState>{0,1}</dxtState> <cddconnectionState>{0,1}</cddconnectionState> <cddserverState>{0,1}</cddserverState> </tetraStatesEvent></pre>
Source	<code><xss:element name="tetraStatesEvent" type="typeSystemTetraStatesEvent" /></code>

Element typeSystemTetraStatesEvent / tcsState

Namespace	DR-GW									
Diagram										
Type	typeSystemElementState									
Properties	<table> <tr> <td>content:</td> <td>simple</td> <td></td> </tr> <tr> <td>minOccurs:</td> <td>0</td> <td></td> </tr> </table>	content:	simple		minOccurs:	0				
content:	simple									
minOccurs:	0									
Facets	<table> <tr> <td>enumeration</td> <td>unknown</td> <td>Unknown state.</td> </tr> <tr> <td>enumeration</td> <td>ok</td> <td>Connection or server is working.</td> </tr> <tr> <td>enumeration</td> <td>n_Ok</td> <td>Connection or server is not working.</td> </tr> </table>	enumeration	unknown	Unknown state.	enumeration	ok	Connection or server is working.	enumeration	n_Ok	Connection or server is not working.
enumeration	unknown	Unknown state.								
enumeration	ok	Connection or server is working.								
enumeration	n_Ok	Connection or server is not working.								
Source	<code><xss:element name="tcsState" type="typeSystemElementState" minOccurs="0" /></code>									

Element typeSystemTetraStatesEvent / dxtState

Namespace	DR-GW									
Diagram										
Type	typeSystemElementState									
Properties	<table> <tr> <td>content:</td> <td>simple</td> <td></td> </tr> <tr> <td>minOccurs:</td> <td>0</td> <td></td> </tr> </table>	content:	simple		minOccurs:	0				
content:	simple									
minOccurs:	0									
Facets	<table> <tr> <td>enumeration</td> <td>unknown</td> <td>Unknown state.</td> </tr> <tr> <td>enumeration</td> <td>ok</td> <td>Connection or server is working.</td> </tr> <tr> <td>enumeration</td> <td>n_Ok</td> <td>Connection or server is not working.</td> </tr> </table>	enumeration	unknown	Unknown state.	enumeration	ok	Connection or server is working.	enumeration	n_Ok	Connection or server is not working.
enumeration	unknown	Unknown state.								
enumeration	ok	Connection or server is working.								
enumeration	n_Ok	Connection or server is not working.								
Source	<code><xss:element name="dxtState" type="typeSystemElementState" minOccurs="0" /></code>									

Element typeSystemTetraStatesEvent / cddconnectionState

Namespace	DR-GW									
Diagram										
Type	typeSystemElementState									
Properties	<table> <tr> <td>content:</td> <td>simple</td> <td></td> </tr> <tr> <td>minOccurs:</td> <td>0</td> <td></td> </tr> </table>	content:	simple		minOccurs:	0				
content:	simple									
minOccurs:	0									
Facets	<table> <tr> <td>enumeration</td> <td>unknown</td> <td>Unknown state.</td> </tr> <tr> <td>enumeration</td> <td>ok</td> <td>Connection or server is working.</td> </tr> <tr> <td>enumeration</td> <td>n_Ok</td> <td>Connection or server is not working.</td> </tr> </table>	enumeration	unknown	Unknown state.	enumeration	ok	Connection or server is working.	enumeration	n_Ok	Connection or server is not working.
enumeration	unknown	Unknown state.								
enumeration	ok	Connection or server is working.								
enumeration	n_Ok	Connection or server is not working.								
Source	<code><xss:element name="cddconnectionState" type="typeSystemElementState" minOccurs="0" /></code>									

Element typeSystemTetraStatesEvent / cddserverState

Namespace	DR-GW									
Diagram	<p>cddserverState \ominus— typeSystemElementState \oplus Specifies connection, server or unit state.</p>									
Type	typeSystemElementState									
Properties	content: simple minOccurs: 0									
Facets	<table> <tr> <td>enumeration</td> <td>unknown</td> <td>Unknown state.</td> </tr> <tr> <td>enumeration</td> <td>ok</td> <td>Connection or server is working.</td> </tr> <tr> <td>enumeration</td> <td>n_Ok</td> <td>Connection or server is not working.</td> </tr> </table>	enumeration	unknown	Unknown state.	enumeration	ok	Connection or server is working.	enumeration	n_Ok	Connection or server is not working.
enumeration	unknown	Unknown state.								
enumeration	ok	Connection or server is working.								
enumeration	n_Ok	Connection or server is not working.								
Source	<code><xs:element name="cddserverState" type="typeSystemElementState" minOccurs="0" /></code>									

Element interfaceSystem / logEvent

Namespace	DR-GW
Diagram	<p>logEvent \ominus— typeSystemLogEvent +— typeEvent (extension base) +—requestId \oplus +—result \oplus +—value \oplus +—text \oplus DR-GW vendor-specific logging information (errors, notices) in both numeric and textual form. See DR-GW vendor-specific...</p>
Type	typeSystemLogEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeSystemLogEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , value{0,1} , text{0,1}
Children	requestId, result, text, value
Instance	<pre><logEvent xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <value>{0,1}</value> <text>{0,1}</text> </logEvent></pre>
Source	<code><xs:element name="logEvent" type="typeSystemLogEvent" /></code>

Element typeSystemLogEvent / value

Namespace	DR-GW
Diagram	<p>value \ominus— xs:hexBinary Built-in primitive type. The hexBinary datatype represents arbitrary hex-encoded binary data.</p>

Type	xs:hexBinary
Properties	<p>content: simple</p> <p>minOccurs: 0</p>
Source	<xs:element name="value" type="xs:hexBinary" minOccurs="0"/>

Element typeSystemLogEvent / text

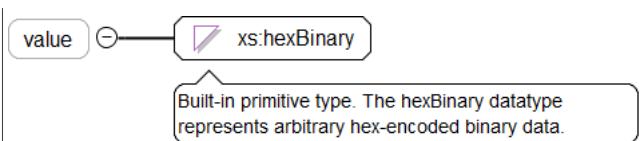
Namespace	DR-GW
Diagram	<p>Diagram illustrating the derivation of the 'text' element from the 'normalizedString' datatype. The 'text' element is marked as simple and has a multiplicity of 0..1. It is derived from the 'normalizedString' datatype, which is described as a built-in derived type representing whitespace-normalized strings.</p>
Type	xs:normalizedString
Properties	<p>content: simple</p> <p>minOccurs: 0</p>
Source	<xs:element name="text" type="xs:normalizedString" minOccurs="0"/>

Element interfaceSystem / event

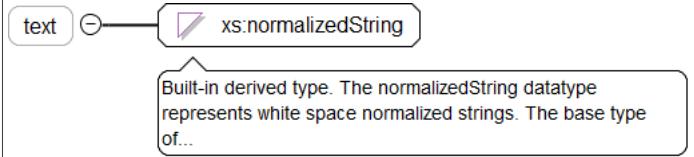
Namespace	DR-GW
Diagram	<p>Diagram illustrating the type hierarchy for 'event'. 'event' is defined as an extension base for 'typeEvent'. It has a multiplicity of 0..1 and is marked as complex. It has three associations: 'requestId' (multiplicity 0..1), 'result' (multiplicity 0..1), and 'value' (multiplicity 0..1). The 'value' association further branches to 'text' (multiplicity 0..1) and 'result' (multiplicity 0..1). A callout box provides vendor-specific information about system events.</p>
Type	typeSystemEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSystemEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , value{0,1} , text{0,1}
Children	requestId, result, text, value
Instance	<pre><event xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <value>{0,1}</value> <text>{0,1}</text> </event></pre>
Source	<xs:element name="event" type="typeSystemEvent" />

Element typeSystemEvent / value

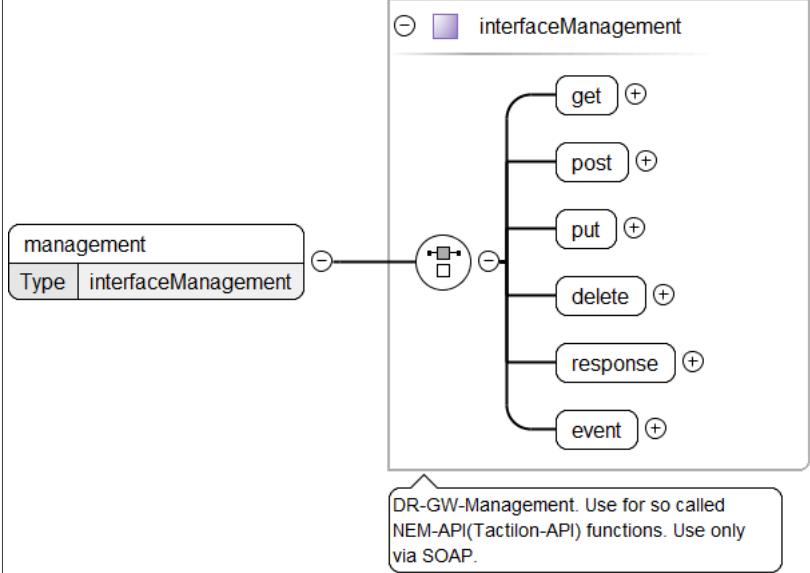
Namespace	DR-GW
-----------	-------

Diagram	
Type	xs:hexBinary
Properties	content: simple minOccurs: 0
Source	<xs:element name="value" type="xs:hexBinary" minOccurs="0"/>

Element typeSystemEvent / text

Namespace	DR-GW
Diagram	
Type	xs:normalizedString
Properties	content: simple minOccurs: 0
Source	<xs:element name="text" type="xs:normalizedString" minOccurs="0"/>

Element drgw / management

Namespace	DR-GW
Diagram	
Type	interfaceManagement
Properties	content: complex
Model	get post put delete response event
Children	delete, event, get, post, put, response
Instance	<pre> <management xmlns="DR-GW"> <get>{1,1}</get> <post>{1,1}</post> <put>{1,1}</put> <delete>{1,1}</delete> <response>{1,1}</response> <event>{1,1}</event> </management> </pre>

Source

<xss:element name="management" type="interfaceManagement" />

Element interfaceManagement / get

Namespace	DR-GW
Diagram	
Type	typeManagementRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeManagementRequest
Properties	content: complex
Model	requestId , requestUri , body{0,1} , authorization{0,1} , host{0,1}
Children	authorization, body, host, requestId, requestUri
Instance	<pre><get xmlns="DR-GW"> <requestId>{1,1}</requestId> <requestUri>{1,1}</requestUri> <body>{0,1}</body> <authorization>{0,1}</authorization> <host>{0,1}</host> </get></pre>
Source	<xss:element name="get" type="typeManagementRequest" />

Element typeManagementRequest / requestUri

Namespace	DR-GW
Diagram	

Type	xs:string
Properties	content: simple
Source	<code><xss:element name="requestUri" type="xs:string"/></code>

Element typeManagementRequest / body

Namespace	DR-GW
Annotations	Some management requests, typically PUT and POST require the presence of the body. See Tactilon-API specification for details on how to construct the body.
Diagram	<p>The diagram shows a class named 'body' with a multiplicity of 0..1. It has a directed association labeled 'xs:string' pointing to a string icon. A callout box from the association points to a note: 'Some management requests, typically PUT and POST require the presence of the body. See Tactilon-API specification for details on how to construct the body.' Another callout box from the string icon points to a note: 'Built-in primitive type. The string datatype represents character strings in XML.'</p>
Type	xs:string
Properties	content: simple minOccurs: 0
Source	<pre><xss:element name="body" type="xs:string" minOccurs="0"> <xss:annotation> <xss:documentation>Some management requests, typically PUT and POST require the presence of the body. See Tactilon-API specification for details on how to construct the body.</xss:documentation> </xss:annotation> </xss:element></pre>

Element typeManagementRequest / authorization

Namespace	DR-GW
Annotations	Optionally, Df-Client can provide his own authorization credentials to access Tactilon. If not provided, the DR-GW shall use its own, globally configured.
Diagram	<p>The diagram shows a class named 'authorization' with a multiplicity of 0..1. It has a directed association labeled 'typeManagementAuthorization' pointing to a square icon. This association is part of a composite structure named 'typeManagementAuthorization'. Inside this structure, there are two elements: 'username' and 'password', each with a multiplicity of 0..1. Callout boxes point from the association to the note: 'Optionally, Df-Client can provide his own authorization credentials to access Tactilon. If not provided, the DR-GW...' and from the square icon to the note: 'typeManagementAuthorization'.</p>
Type	typeManagementAuthorization
Properties	content: complex minOccurs: 0
Model	username , password
Children	password, username
Instance	<pre><authorization xmlns="DR-GW"> <username>{1,1}</username> <password>{1,1}</password> </authorization></pre>
Source	<code><xss:element name="authorization" type="typeManagementAuthorization" minOccurs="0"></code>

```

<xs:annotation>
  <xs:documentation>Optionally, Df-Client can provide his own authorization credentials to access Tactilon. If not provided, the DR-GW shall use its own, globally configured.</xs:documentation>
</xs:annotation>
</xs:element>

```

Element typeManagementAuthorization / username

Namespace	DR-GW
Diagram	<p>Built-in primitive type. The string datatype represents character strings in XML.</p>
Type	xs:string
Properties	content: simple
Source	<xs:element name="username" type="xs:string" />

Element typeManagementAuthorization / password

Namespace	DR-GW
Diagram	<p>Built-in primitive type. The string datatype represents character strings in XML.</p>
Type	xs:string
Properties	content: simple
Source	<xs:element name="password" type="xs:string" />

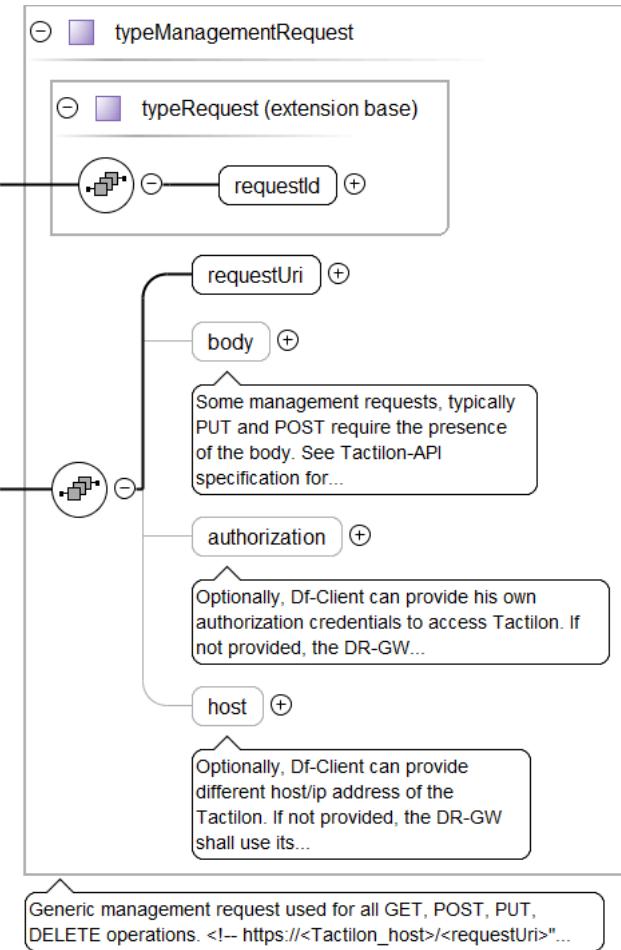
Element typeManagementRequest / host

Namespace	DR-GW
Annotations	Optionally, Df-Client can provide different host/ip address of the Tactilon. If not provided, the DR-GW shall use its own, globally configured.
Diagram	<p>Optionally, Df-Client can provide different host/ip address of the Tactilon. If not provided, the DR-GW shall use its...</p> <p>Built-in primitive type. The string datatype represents character strings in XML.</p>
Type	xs:string
Properties	content: simple minOccurs: 0
Source	<pre> <xs:element name="host" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>Optionally, Df-Client can provide different host/ip address of the Tactilon. If not provided, the DR-GW shall use its own, globally configured.</xs:documentation> </xs:annotation> </xs:element> </pre>

Element interfaceManagement / post

Namespace	DR-GW
-----------	-------

Diagram

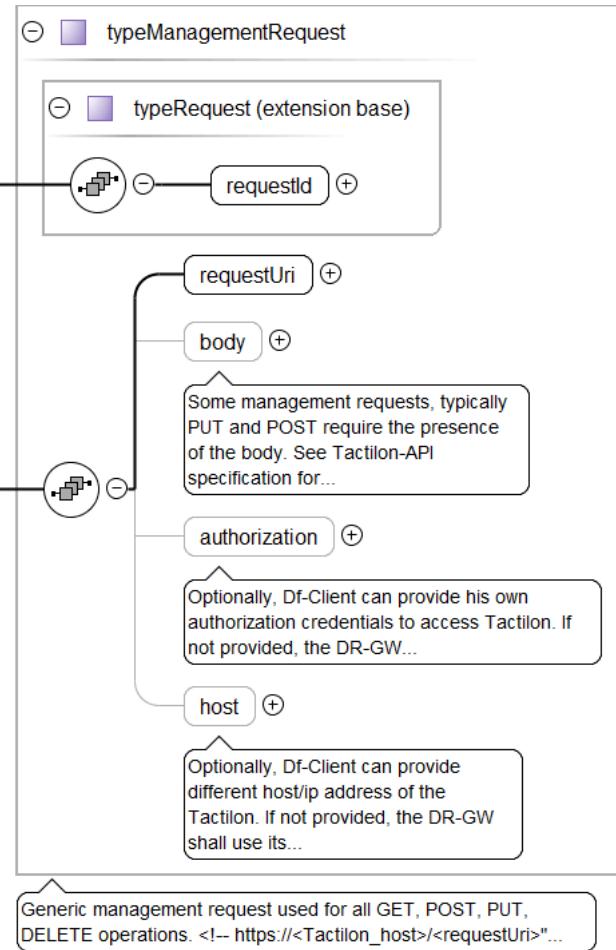


Type	<code>typeManagementRequest</code>
Type hierarchy	<ul style="list-style-type: none"> <code>typeRequest</code> <code>typeManagementRequest</code>
Properties	content: complex
Model	<code>requestId , requestUri , body{0,1} , authorization{0,1} , host{0,1}</code>
Children	<code>authorization, body, host, requestId, requestUri</code>
Instance	<pre> <post xmlns="DR-GW"> <requestId>{1,1}</requestId> <requestUri>{1,1}</requestUri> <body>{0,1}</body> <authorization>{0,1}</authorization> <host>{0,1}</host> </post> </pre>
Source	<code><xss:element name="post" type="typeManagementRequest" /></code>

Element interfaceManagement / put

Namespace	DR-GW
-----------	-------

Diagram

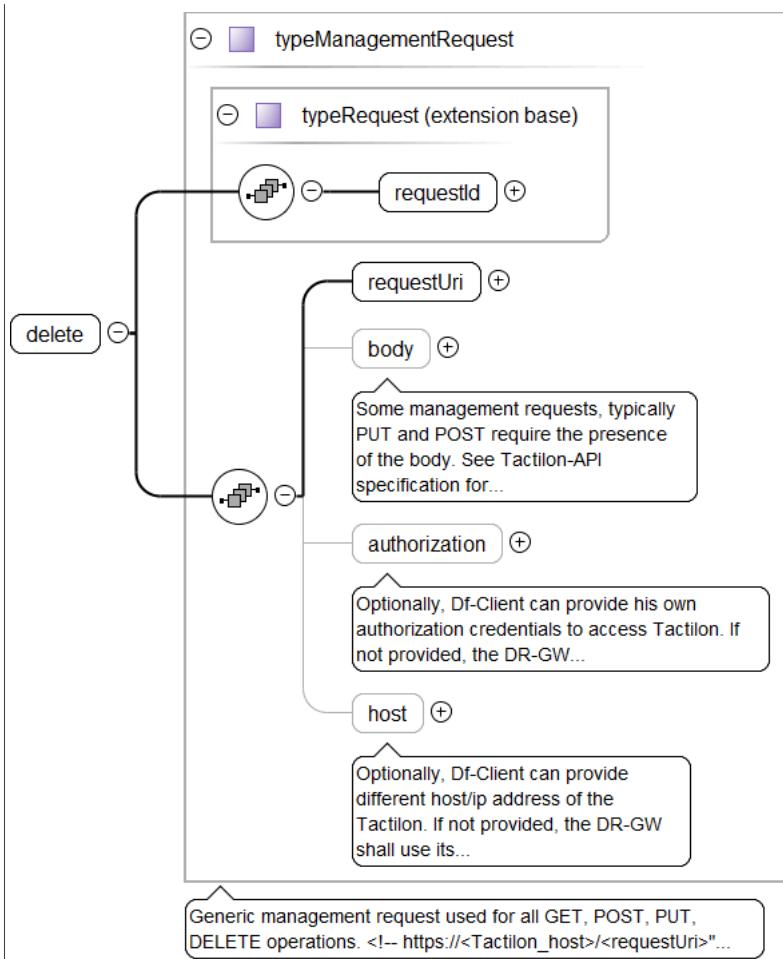


Type	<code>typeManagementRequest</code>
Type hierarchy	<ul style="list-style-type: none"> • <code>typeRequest</code> • <code>typeManagementRequest</code>
Properties	content: complex
Model	<code>requestId , requestUri , body{0,1} , authorization{0,1} , host{0,1}</code>
Children	authorization, body, host, requestId, requestUri
Instance	<pre><put xmlns="DR-GW"> <requestId>{1,1}</requestId> <requestUri>{1,1}</requestUri> <body>{0,1}</body> <authorization>{0,1}</authorization> <host>{0,1}</host> </put></pre>
Source	<code><xss:element name="put" type="typeManagementRequest" /></code>

Element interfaceManagement / delete

Namespace	DR-GW
-----------	-------

Diagram



Type	<code>typeManagementRequest</code>
------	------------------------------------

Type hierarchy	<ul style="list-style-type: none"> <code>typeRequest</code> <code>typeManagementRequest</code>
----------------	--

Properties	content: complex
------------	------------------

Model	<code>requestId</code> , <code>requestUri</code> , <code>body</code> {0,1} , <code>authorization</code> {0,1} , <code>host</code> {0,1}
-------	---

Children	<code>authorization</code> , <code>body</code> , <code>host</code> , <code>requestId</code> , <code>requestUri</code>
----------	---

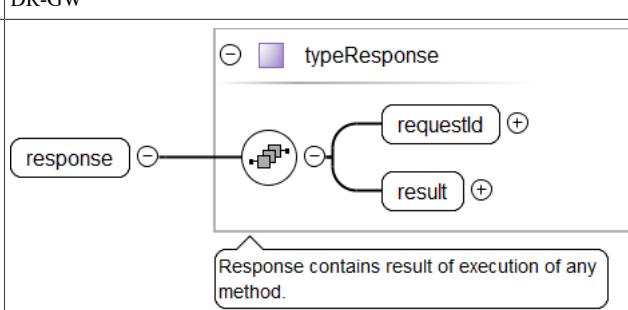
Instance	<pre><delete xmlns="DR-GW"> <requestId>{1,1}</requestId> <requestUri>{1,1}</requestUri> <body>{0,1}</body> <authorization>{0,1}</authorization> <host>{0,1}</host> </delete></pre>
----------	--

Source	<code><xss:element name="delete" type="typeManagementRequest" /></code>
--------	---

Element interfaceManagement / response

Namespace	DR-GW
-----------	-------

Diagram



Type	typeResponse
Properties	content: complex
Model	requestId , result
Children	requestId, result
Instance	<response xmlns="DR-GW"> <requestId>{1,1}</requestId> <result>{1,1}</result> </response>
Source	<xss:element name="response" type="typeResponse" />

Element interfaceManagement / event

Namespace	DR-GW
Diagram	<pre> classDiagram typeManagementEvent < -- typeEvent typeEvent < -- event typeManagementEvent "0..1" requestId typeManagementEvent "0..1" result typeManagementEvent "0..1" body note over typeManagementEvent: Generic management event, is fired from DR-GW as a final response to previous get, post, put or delete request from... </pre>
Type	typeManagementEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeManagementEvent
Properties	content: complex
Model	requestId{0,1} , result{0,1} , body{0,1}
Children	body, requestId, result
Instance	<event xmlns="DR-GW"> <requestId>{0,1}</requestId> <result>{0,1}</result> <body>{0,1}</body> </event>
Source	<xss:element name="event" type="typeManagementEvent" />

Element typeManagementEvent / body

Namespace	DR-GW				
Diagram	<pre> classDiagram body < -- xs:string note over body: Built-in primitive type. The string datatype represents character strings in XML. </pre>				
Type	xs:string				
Properties	<table border="1"> <tr> <td>content:</td> <td>simple</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<xss:element name="body" type="xs:string" minOccurs="0" />				

Element typeCallRequest / action

Namespace	DR-GW
-----------	-------

Diagram	<pre> graph LR action((action)) --> typeActionRequest[typeActionRequest] typeActionRequest --> callActions[All possible call actions.] </pre>		
Type	typeActionRequest		
Properties	content: simple		
Facets	enumeration	setup	This method is used to initiate a new call setup. For a call setup to be successful it is required that the resources have been reserved prior this method call.
	enumeration	connect	This method is used to connect an incoming call.
	enumeration	hold	This method requests to put an individual call to hold.
	enumeration	unhold	This method is a request for resuming an individual call from hold.
	enumeration	disconnect	This method is used to disconnect a call.
	enumeration	transfer	This method is used to transfer an individual call to a new recipient.
	enumeration	releasecall	This method is used to release radio subscriber's individual call.
Source	<xs:element name="action" type="typeActionRequest"/>		

Element typeCallRequest / attributes

Namespace	DR-GW
Diagram	<pre> graph LR attributes((attributes)) --> typeCallAttributes[typeCallAttributes] typeCallAttributes --> callAttributes[Contains all attributes of the TETRA voice call.] </pre>
Type	typeCallAttributes
Properties	<p>content: complex</p> <p>minOccurs: 0</p>
Model	hook{0,1} , mode{0,1} , commtype{0,1} , priority{0,1} , encryption{0,1} , ambienceListen{0,1} , req2speak{0,1} , demandPriority{0,1}
Children	ambienceListen, commtype, demandPriority, encryption, hook, mode, priority, req2speak
Instance	<attributes xmlns="DR-GW"> <hook>{0,1}</hook>

	<pre> <mode>{0,1}</mode> <commtyp>{0,1}</commtyp> <priority>{0,1}</priority> <encryption>{0,1}</encryption> <ambienceListen>{0,1}</ambienceListen> <req2speak>{0,1}</req2speak> <demandPriority>{0,1}</demandPriority> </attributes> </pre>
Source	<xs:element name="attributes" type="typeCallAttributes" minOccurs="0" />

Element typeCallRequest / callingParty

Namespace	DR-GW				
Diagram	<p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>				
Type	typeAddress				
Properties	<table border="1"> <tr> <td>content:</td> <td>complex</td> </tr> <tr> <td>minOccurs:</td> <td>0</td> </tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}				
Children	alias, cell, external, fssn, msisdn, opta, subscriber				
Instance	<pre> <callingParty xmlns="DR-GW"> <subscriber>{0,1}</subscriber> <alias>{0,1}</alias> <msisdn>{0,1}</msisdn> <fssn>{0,1}</fssn> <external>{0,1}</external> <opta>{0,1}</opta> <cell>{0,1}</cell> </callingParty> </pre>				
Source	<xs:element name="callingParty" type="typeAddress" minOccurs="0" />				

Element typeCallRequest / calledParty

Namespace	DR-GW
-----------	-------

Diagram	<p><code><calledParty xmlns="DR-GW"></code> <code><subscriber>{0..1}</subscriber></code> <code><alias>{0..1}</alias></code> <code><msisdn>{0..1}</msisdn></code> <code><fssn>{0..1}</fssn></code> <code><external>{0..1}</external></code> <code><opta>{0..1}</opta></code> <code><cell>{0..1}</cell></code> <code></calledParty></code></p> <p>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</p>				
Type	typeAddress				
Properties	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">content:</td><td style="padding: 2px;">complex</td></tr> <tr> <td style="padding: 2px;">minOccurs:</td><td style="padding: 2px;">0</td></tr> </table>	content:	complex	minOccurs:	0
content:	complex				
minOccurs:	0				
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}				
Children	alias, cell, external, fssn, msisdn, opta, subscriber				
Instance	<pre><calledParty xmlns="DR-GW"> <subscriber>{0..1}</subscriber> <alias>{0..1}</alias> <msisdn>{0..1}</msisdn> <fssn>{0..1}</fssn> <external>{0..1}</external> <opta>{0..1}</opta> <cell>{0..1}</cell> </calledParty></pre>				
Source	<code><xss:element name="calledParty" type="typeAddress" minOccurs="0" /></code>				

Element typeCallRequest / workstationId

Namespace	DR-GW				
Diagram	<p>Optional parameter is used to support the "neighbours" feature.</p>				
Type	typeWorkstationId				
Properties	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">content:</td><td style="padding: 2px;">simple</td></tr> <tr> <td style="padding: 2px;">minOccurs:</td><td style="padding: 2px;">0</td></tr> </table>	content:	simple	minOccurs:	0
content:	simple				
minOccurs:	0				
Source	<code><xss:element name="workstationId" type="typeWorkstationId" minOccurs="0" /></code>				

Element check

Namespace	DR-GW
-----------	-------

Diagram	<p>To enable the DF-Client to check connectivity to DR-GW the client may use session check. The check requires the http...</p>
Type	typeSessionCheck
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSessionCheck
Properties	content: complex
Model	requestId , clientid
Children	clientid, requestId
Instance	<pre><check xmlns="DR-GW"> <requestId>{1,1}</requestId> <clientid>{1,1}</clientid> </check></pre>
Source	<code><xs:element name="check" type="typeSessionCheck" /></code>

Element typeSessionLogoutEvent / reason

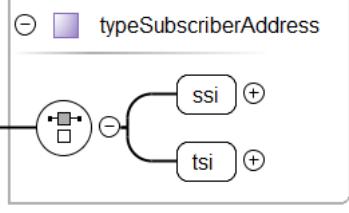
Namespace	DR-GW
Diagram	<p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>
Type	xs:unsignedLong
Properties	content: simple minOccurs: 0
Source	<code><xs:element name="reason" type="xs:unsignedLong" minOccurs="0" /></code>

Element typeSdsValidity / value

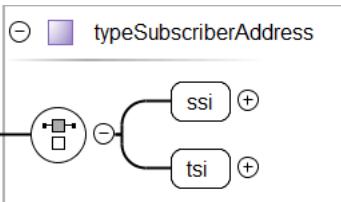
Namespace	DR-GW
Diagram	<p>Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...</p>
Type	xs:unsignedLong
Properties	content: simple
Source	<code><xs:element name="value" type="xs:unsignedLong" /></code>

Element typeGroupGetCombinationsEvent / group

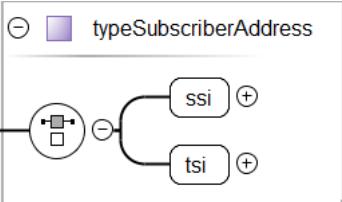
Namespace	DR-GW
-----------	-------

Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><group xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </group></pre>
Source	<code><xss:element name="group" type="typeSubscriberAddress"/></code>

Element typeGroupGetCombinationsEvent / baseGroup

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><baseGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </baseGroup></pre>
Source	<code><xss:element name="baseGroup" type="typeSubscriberAddress"/></code>

Element typeGroupGetCombinationsEvent / constitGroup

Namespace	DR-GW
Diagram	
Type	typeSubscriberAddress
Properties	content: complex maxOccurs: 7
Model	ssi tsi
Children	ssi, tsi
Instance	<pre><constitGroup xmlns="DR-GW"> <ssi>{1,1}</ssi> <tsi>{1,1}</tsi> </constitGroup></pre>
Source	<code><xss:element name="constitGroup" type="typeSubscriberAddress" maxOccurs="7"/></code>

Complex Type(s)

Complex Type interfaceCall

Namespace	DR-GW
Annotations	DR-GW-Call. Use for call control/ call monitoring. This is the only element, that can be used via both SIP/SOAP. When used via SOAP it can only be used for call event monitoring.
Diagram	<p>DR-GW-Call. Use for call control/ call monitoring. This is the only element, that can be used via both SIP/SOAP. When...</p>
Used by	Element drgw/call
Model	select request pttRequest keyExchange response selectEvent event pttEvent unitInEmergencyEvent keyExchangeEvent
Children	event, keyExchange, keyExchangeEvent, pttEvent, pttRequest, request, response, select, selectEvent, unitInEmergencyEvent
Source	<pre> <xs:complexType name="interfaceCall"> <xs:annotation> <xs:documentation>DR-GW-Call. Use for call control/ call monitoring. This is the only element, that can be used via both SIP/SOAP. When used via SOAP it can only be used for call event monitoring.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="select" type="typeCallSelect"/> <xs:element name="request" type="typeCallEvent"/> <xs:element name="pttRequest" type="typeCallPTTRequest"/> <xs:element name="keyExchange" type="typeCallKeyExchange"/> <xs:element name="response" type="typeResponse"/> <xs:element name="selectEvent" type="typeCallSelectEvent"/> <xs:element name="event" type="typeCallEvent"/> <xs:element name="pttEvent" type="typeCallPTTEvent"/> <xs:element name="unitInEmergencyEvent" type="typeCallUnitInEmergencyEvent"/> <xs:element name="keyExchangeEvent" type="typeCallKeyExchangeEvent"/> </xs:choice> </xs:complexType></pre>

Complex Type typeCallSelect

Namespace	DR-GW
Annotations	The method reserves speech line for the targets of selection operation, sets the selection level of the targets of selection operation to given values.

Diagram	<p>The method reserves speech line for the targets of selection operation, sets the selection level of the targets of...</p>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeCallSelect
Used by	Element interfaceCall/select
Model	requestId , target
Children	requestId, target
Source	<pre><xs:complexType name="typeCallSelect"> <xs:annotation> <xs:documentation>The method reserves speech line for the targets of selection operation, sets the selection level of the targets of selection operation to given values.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="target" type="typeSelection"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeRequest

Namespace	DR-GW
Diagram	
Used by	Complex Types
	typeAppGet, typeAppGetList, typeCallKeyExchange, typeCallPTTRequest, typeCallRequest, typeCallSelect, typeGroupAddCombination, typeGroupAddRadioMember, typeGroupGet, typeGroupGetAppMembers, typeGroupGetCombinations, typeGroupGetList, typeGroupGetRadioMembers, typeGroupRemoveCombination, typeGroupRemoveRadioMember, typeGroupSubscribeData, typeGroupTrack, typeManagementRequest, typeOrgGet, typeOrgGetList, typeRadioChangeOpta, typeRadioDisable, typeRadioEnable, typeRadioGet, typeRadioGetGroups, typeRadioGetList, typeRadioTrack, typeSdsSend, typeSdsSendReport, typeSessionCheck, typeSessionLogin, typeSessionLogout, typeSessionSupervise, typeStatusSend
Model	requestId
Children	requestId
Source	<pre><xs:complexType name="typeRequest"> <xs:sequence> <xs:element name="requestId" type="xs:long"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeSelection

Namespace	DR-GW
Annotations	
Diagram	
Used by	Elements typeCallSelect/target, typeCallSelectEvent/target

Model	level , target
Children	level, target
Source	<pre><xss:complexType name="typeSelection"> <xss:annotation> <xss:documentation/> </xss:annotation> <xss:sequence> <xss:element name="level" type="typeSelectionLevel"/> <xss:element name="target" type="typeAddress"/> </xss:sequence> </xss:complexType></pre>

Complex Type typeAddress

Namespace	DR-GW
Annotations	Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).
Diagram	<pre> classDiagram class typeAddress { <<Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).>> } class subscriber class alias class msisdn class fssn class external class opta class cell typeAddress < -- subscriber typeAddress < -- alias typeAddress < -- msisdn typeAddress < -- fssn typeAddress < -- external typeAddress < -- opta typeAddress < -- cell </pre>
Used by	<p>Elements</p> <p>typeCallEvent/calledParty, typeCallEvent/callingParty, typeCallPTTRequest/talkingParty, typeCallRequest/calledParty, typeCallRequest/callingParty, typeCallUnitInEmergencyEvent/unitInEmg, typeSds/forward, typeSds/source, typeSds/target, typeSdsReportEvent/source, typeSdsReportEvent/target, typeSdsSendReport/target, typeSelection/target, typeStatus/source, typeStatus/target, typeTxGranted/talkingParty</p>
Model	subscriber{0,1} , alias{0,1} , msisdn{0,1} , fssn{0,1} , external{0,1} , opta{0,1} , cell{0,1}
Children	alias, cell, external, fssn, msisdn, opta, subscriber
Source	<pre><xss:complexType name="typeAddress"> <xss:annotation> <xss:documentation>Basic type for all possible TETRA address types (SSI, TSI, MSISDN, FSSN, OPTA).</xss:documentation> </xss:annotation> <xss:sequence> <xss:element name="subscriber" type="typeSubscriberAddress" minOccurs="0"/> <xss:element name="alias" type="xs:normalizedString" minOccurs="0"/> <xss:element name="msisdn" type="typeDialString" minOccurs="0"/> <xss:element name="fssn" type="xs:unsignedLong" minOccurs="0"> <xss:annotation> <xss:documentation>Fleet specific short number</xss:documentation> </xss:annotation> </xss:element> <xss:element name="external" type="typeExternal" minOccurs="0"/> <xss:element name="opta" type="typeOpta" minOccurs="0"/> <xss:element name="cell" type="xs:short" minOccurs="0"/> </xss:sequence> </xss:complexType></pre>

Complex Type typeSubscriberAddress

Namespace	DR-GW
Annotations	

Diagram	A UML statechart diagram showing a choice node with two outgoing transitions. The top transition leads to a state labeled 'ssi' with a plus sign (+) indicating multiplicity. The bottom transition leads to a state labeled 'tsi' with a plus sign (+) indicating multiplicity.
Used by	Elements typeAddress/subscriber, typeAppGet/app, typeApplication/addr, typeCallUnitInEmergencyEvent/group, typeGroup/addr, typeGroupAddCombination/baseGroup, typeGroupAddCombination/group, typeGroupAddCombinationEvent/baseGroup, typeGroupAddCombinationEvent/group, typeGroupAddRadioMember/group, typeGroupAddRadioMember/radio, typeGroupAddRadioMemberEvent/group, typeGroupAddRadioMemberEvent/radio, typeGroupAppMemberEvent/app, typeGroupAppMemberEvent/group, typeGroupCombinationEvent/baseGroup, typeGroupCombinationEvent/constitGroup, typeGroupCombinationEvent/group, typeGroupDataSubscription/addr, typeGroupGet/group, typeGroupGetAppMembers/group, typeGroupGetAppMembersEvent/app, typeGroupGetCombinations/group, typeGroupGetCombinationsEvent/baseGroup, typeGroupGetCombinationsEvent/constitGroup, typeGroupGetCombinationsEvent/group, typeGroupGetRadioMembers/group, typeGroupGetRadioMembersEvent/group, typeGroupGetRadioMembersEvent/radio, typeGroupRadioMemberEvent/group, typeGroupRadioMemberEvent/radio, typeGroupRemoveCombination/baseGroup, typeGroupRemoveCombinationEvent/group, typeGroupRemoveCombinationEvent/baseGroup, typeGroupRemoveCombinationEvent/group, typeGroupRemoveRadioMember/group, typeGroupRemoveRadioMemberEvent/radio, typeGroupRemoveRadioMemberEvent/group, typeGroupRemoveRadioMemberEvent/radio, typeGroupTrack/group, typeGroupTrackSubscriptionEvent/group, typeRadio/issi, typeRadioChangeOpta/radio, typeRadioChangeOptaEvent/radio, typeRadioDisable/radio, typeRadioEnable/radio, typeRadioEnableDisableEvent/radio, typeRadioGet/radio, typeRadioGetGroupsEvent/radio, typeRadioGroupSelection/group, typeRadioGroupsEvent/deletedGroup, typeRadioGroupsEvent/radio, typeRadioTrack/radio, typeRadioTrackSubscriptionEvent/radio, typeRadioTrackingData/callParty, typeRadioTrackingData/radio, typeSds/e2eegroup
Model	ssi tsi
Children	ssi, tsi
Source	<pre><xs:complexType name="typeSubscriberAddress"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:choice> <xs:element name="ssi" type="xs:unsignedLong"/> <xs:element name="tsi" type="typeTSI"/> </xs:choice> </xs:complexType></pre>

Complex Type typeTSI

Namespace	DR-GW
Annotations	Basic type for TETRA subscriber identity containing Network code(MNC) and Country code(MCC) .
Diagram	A UML statechart diagram showing a sequence node with three outgoing transitions. The first transition leads to a state labeled 'mnc' with a plus sign (+) indicating multiplicity. The second transition leads to a state labeled 'mcc' with a plus sign (+) indicating multiplicity. The third transition leads to a state labeled 'ssi' with a plus sign (+) indicating multiplicity. A callout box above the sequence node contains the text: 'Basic type for TETRA subscriber identity containing Network code(MNC) and Country code(MCC)'.
Used by	Element typeSubscriberAddress/tsi
Model	mnc , mcc , ssi
Children	mcc, mnc, ssi
Source	<pre><xs:complexType name="typeTSI"> <xs:annotation> <xs:documentation>Basic type for TETRA subscriber identity containing Network code(MNC) and Country code(MCC).</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="mnc" type="xs:unsignedShort"/> <xs:element name="mcc" type="xs:unsignedShort"/> <xs:element name="ssi" type="xs:unsignedLong"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeExternal

Namespace	DR-GW
Annotations	External number consisting of Gateway number + DialString
Diagram	<pre> classDiagram typeExternal < -- typeEvent typeExternal { gatewayNumber number } gatewayNumber *--> number </pre> <p>External number consisting of Gateway number + DialString</p>
Used by	Element typeAddress/external
Model	gatewayNumber, number
Children	gatewayNumber, number
Source	<pre> <xss:complexType name="typeExternal"> <xss:annotation> <xss:documentation>External number consisting of Gateway number + DialString</xss:documentation> </xss:annotation> <xss:sequence> <xss:element name="gatewayNumber" type="xs:unsignedLong"/> <xss:element name="number" type="typeDialString"/> </xss:sequence> </xss:complexType> </pre>

Complex Type typeCallEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeCallEvent typeEvent { requestId result } typeCallEvent { tetraCallId action attributes callingParty calledParty disconnectCause } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeCallEvent
Used by	Elements interfaceCall/event, interfaceCall/request
Model	requestId{0,1}, result{0,1}, tetraCallId{0,1}, action, attributes{0,1}, callingParty{0,1}, calledParty{0,1}, disconnectCause{0,1}
Children	action, attributes, calledParty, callingParty, disconnectCause, requestId, result, tetraCallId
Source	<pre> <xss:complexType name="typeCallEvent"> <xss:annotation> <xss:documentation> </xss:documentation> </xss:annotation> <xss:complexContent> <xss:extension base="typeEvent"> ... </xss:extension> </xss:complexContent> </xss:complexType> </pre>

```

<xs:sequence>
  <xs:element name="tetraCallId" type="typeTetraCallId" minOccurs="0"/>
  <xs:element name="action" type="typeActionEvent"/>
  <xs:element name="attributes" type="typeCallAttributes" minOccurs="0"/>
  <xs:element name="callingParty" type="typeAddress" minOccurs="0"/>
  <xs:element name="calledParty" type="typeAddress" minOccurs="0"/>
  <xs:element name="disconnectCause" type="typeDisconnectCause" minOccurs="0"/>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeEvent

Namespace	DR-GW
Diagram	
Used by	Complex Types typeAppGetEvent, typeAppGetListEvent, typeCallEvent, typeCallKeyExchangeEvent, typeCallPT-TEvent, typeCallSelectEvent, typeCallUnitInEmergencyEvent, typeGroupAddCombinationEvent, typeGroupAddRadioMemberEvent, typeGroupAppMemberEvent, typeGroupCombinationEvent, typeGroupEvent, typeGroupGetAppMembersEvent, typeGroupGetCombinationsEvent, typeGroupGetEvent, typeGroupGetListEvent, typeGroupGetRadioMembersEvent, typeGroupRadioMemberEvent, typeGroupRemoveCombinationEvent, typeGroupRemoveRadioMemberEvent, typeGroupSubscribeDataEvent, typeGroupTrackSubscriptionEvent, typeManagementEvent, typeOrgEvent, typeOrgGetEvent, typeOrgGetListEvent, typeRadioChangeOptaEvent, typeRadioEnableDisableEvent, typeRadioEvent, typeRadioGetEvent, typeRadioGetGroupsEvent, typeRadioGetListEvent, typeRadioGroupsEvent, typeRadioTrackEvent, typeRadioTrackSubscriptionEvent, typeSdsReceiveEvent, typeSdsReportEvent, typeSdsSendEvent, typeSessionLoginEvent, typeSessionLogoutEvent, typeSessionSuperviseEvent, typeStatusReceiveEvent, typeStatusSendEvent, typeSystemEvent, typeSystemLogEvent, typeSystemTetraStatesEvent
Model	requestId{0,1} , result{0,1}
Children	requestId, result
Source	<pre> <xs:complexType name="typeEvent"> <xs:sequence> <xs:element name="requestId" type="xs:unsignedLong" minOccurs="0"/> <xs:element name="result" type="typeResult" minOccurs="0"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeResult

Namespace	DR-GW
Annotations	Common result values used in every response and optional specific subsystem result codes.
Diagram	
Used by	Elements typeEvent/result, typeResponse/result
Model	responseCode , sourceSystem{0,1} , result{0,1}
Children	responseCode, sourceSystem
Source	<pre> <xs:complexType name="typeResult"> <xs:annotation> <xs:documentation>Common result values used in every response and optional specific subsystem result codes.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="responseCode" type="typeResponseCode"/> <xs:element name="sourceSystem" type="typeSourceSystem" minOccurs="0"/> <xs:element name="result" type="xs:unsignedLong" minOccurs="0"/> </xs:sequence> </xs:complexType> </pre>

<code></xs:complexType></code>

Complex Type typeCallAttributes

Namespace	DR-GW
Annotations	Contains all attributes of the TETRA voice call.
Diagram	<pre> classDiagram class typeCallAttributes { hook mode commtype priority encryption ambienceListen req2speak demandPriority } </pre> <p>Contains all attributes of the TETRA voice call.</p>
Used by	Elements typeCallEvent/attributes, typeCallPTTRequest/attributes, typeCallRequest/attributes
Model	hook{0,1} , mode{0,1} , commtype{0,1} , priority{0,1} , encryption{0,1} , ambienceListen{0,1} , req2speak{0,1} , demandPriority{0,1}
Children	ambienceListen, commtype, demandPriority, encryption, hook, mode, priority, req2speak
Source	<pre> <xs:complexType name="typeCallAttributes"> <xs:annotation> <xs:documentation>Contains all attributes of the TETRA voice call.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="hook" type="xs:boolean" minOccurs="0"/> <xs:element name="mode" type="typeCallMode" minOccurs="0"/> <xs:element name="commtype" type="typeCallType" minOccurs="0"/> <xs:element name="priority" type="xs:unsignedByte" default="1" minOccurs="0"/> <xs:element name="encryption" type="xs:boolean" default="true" minOccurs="0"/> <xs:element name="ambienceListen" type="xs:boolean" default="0" minOccurs="0"/> <xs:element name="req2speak" type="xs:boolean" default="1" minOccurs="0"/> <xs:element name="demandPriority" type="typeTxDemandPriority" default="normal" minOccurs="0"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeDisconnectCause

Namespace	DR-GW
Diagram	<pre> classDiagram class typeDisconnectCause { protocol code text } </pre> <p>Value according to DR-GW-Reason header in DR-GW-Interface specification. It should only be present if action is...</p> <p>Optional textual representation of the cause.</p>
Used by	Element typeCallEvent/disconnectCause
Model	protocol , code , text{0,1}
Children	code, protocol, text

Source	<pre> <xs:complexType name="typeDisconnectCause"> <xs:sequence> <xs:element name="protocol"> <xs:simpleType> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="DR-GW"/> <xs:enumeration value="TCS-API"/> </xs:restriction> </xs:simpleType> </xs:element> <xs:element name="code" type="xs:unsignedInt"> <xs:annotation> <xs:documentation>Value according to DR-GW-Reason header in DR-GW-Interface specification. It should only be present if action is "disconnected" and holds the reason for call disconnection.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="text" minOccurs="0"> <xs:annotation> <xs:documentation>Optional textual representation of the cause.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:normalizedString"> <xs:maxLength value="80"/> </xs:restriction> </xs:simpleType> </xs:element> </xs:sequence> </xs:complexType> </pre>
--------	---

Complex Type typeCallPTTRequest

Namespace	DR-GW
Annotations	DR-GW-Call PTTRequest. Only "DemandTx" and "CeaseTx" actions only.
Diagram	<pre> classDiagram typeRequest("typeRequest (extension base)") { requestId action attributes talkingParty workstationId } typeCallPTTRequest("typeCallPTTRequest") { <<DR-GW-Call PTTRequest. Only "DemandTx" and "CeaseTx" actions only.>> } typeRequest < -- typeCallPTTRequest </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeCallPTTRequest
Used by	Element interfaceCall/pttRequest
Model	requestId , action , attributes{0,1} , talkingParty{0,1} , workstationId{0,1}
Children	action, attributes, requestId, talkingParty, workstationId
Source	<pre> <xs:complexType name="typeCallPTTRequest"> <xs:annotation> <xs:documentation>DR-GW-Call PTTRequest. Only "DemandTx" and "CeaseTx" actions only.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="action" type="typeActionPTTRequest"/> <xs:element name="attributes" type="typeCallAttributes" minOccurs="0"/> <xs:element name="talkingParty" type="typeAddress" minOccurs="0"/> <xs:element name="workstationId" type="typeWorkstationId" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeCallKeyExchange

Namespace	DR-GW
Annotations	For triggering the group key exchange. Key exchange events are sent in Call_KeyXEvent.
Diagram	<pre> classDiagram typeRequest < -- typeCallKeyExchange typeRequest { <<extension base="typeRequest">> <<sequence>> <<element name="requestId" type="xs:unsignedLong"/>> <<element name="action" type="typeKeyExchangeAction"/>> } typeCallKeyExchange { <<documentation>>For triggering the group key exchange. Key exchange events are sent in Call_KeyXEvent. } </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeCallKeyExchange
Used by	Element interfaceCall/keyExchange
Model	requestId , action
Children	action, requestId
Source	<pre> <xs:complexType name="typeCallKeyExchange"> <xs:annotation> <xs:documentation>For triggering the group key exchange. Key exchange events are sent in Call_KeyXEvent.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="action" type="typeKeyExchangeAction"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeResponse

Namespace	DR-GW
Annotations	Response contains result of execution of any method.
Diagram	<pre> classDiagram typeResponse { <<documentation>>Response contains result of execution of any method. } typeResponse --> typeResponse : requestId --> result </pre>
Used by	Elements interfaceApp/response, interfaceCall/response, interfaceGroup/response, interfaceManagement/response, interfaceOrg/response, interfaceRadio/response, interfaceSds/response, interfaceSession/response, interfaceStatus/response, interfaceSystem/response
Model	requestId , result
Children	requestId, result
Source	<pre> <xs:complexType name="typeResponse"> <xs:annotation> <xs:documentation>Response contains result of execution of any method.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="requestId" type="xs:unsignedLong"/> <xs:element name="result" type="typeResult"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeCallSelectEvent

Namespace	DR-GW
Annotations	The event informs about the actual state of the selection requested before using the "select" request.

Diagram	<pre> classDiagram typeEvent { requestId result } typeCallSelectEvent { target } typeEvent < -- typeCallSelectEvent </pre> <p>The event informs about the actual state of the selection requested before using the "select" request.</p>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeCallSelectEvent
Used by	Element interfaceCall/selectEvent
Model	requestId{0,1} , result{0,1} , target
Children	requestId, result, target
Source	<pre> <xss:complexType name="typeCallSelectEvent"> <xss:annotation> <xss:documentation>The event informs about the actual state of the selection requested before using the "select" request.</xss:documentation> </xss:annotation> <xss:complexContent> <xss:extension base="typeEvent"> <xss:sequence> <xss:element name="target" type="typeSelection"/> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

Complex Type typeCallPTTEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent { requestId result } typeCallPTTEvent { tetraCallId granted ceased wait } typeEvent < -- typeCallPTTEvent </pre> <p>This event is used to inform that transmission is ceased and nobody has the speech item.</p> <p>This event is used to inform that the call is temporarily paused e.g. if radio subscriber has roamed to a new cell and...</p>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeCallPTTEvent

Used by	Element	interfaceCall/pttEvent
Model		requestId{0,1} , result{0,1} , tetraCallId{0,1} , (granted ceased wait)
Children		ceased, granted, requestId, result, tetraCallId, wait
Source		<pre> <xs:complexType name="typeCallPTTEvent"> <xs:annotation> <xs:documentation> </xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="tetraCallId" type="typeTetraCallId" minOccurs="0"/> <xs:choice> <xs:element name="granted" type="typeTxGranted"/> <xs:element name="ceased" type="typeEmpty"> <xs:annotation> <xs:documentation>This event is used to inform that transmission is ceased and nobody has the speech item.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="wait" type="typeEmpty"> <xs:annotation> <xs:documentation>This event is used to inform that the call is temporarily paused e.g. if radio subscriber has roamed to a new cell and there are no free resources available.</xs:documentation> </xs:annotation> </xs:element> </xs:choice> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeTxGranted

Namespace	DR-GW
Annotations	This event is used to inform of granted transmission request. This event is called when the transmission is granted to client, to another user or the request has been queued.
Diagram	<p>The diagram illustrates the structure of the typeTxGranted complex type. It features a central class box labeled "typeTxGranted" with a purple square icon. A line with a hollow circle at the end connects it to a rounded rectangle containing the text: "This event is used to inform of granted transmission request. This event is called when the transmission is granted to...". From the "typeTxGranted" box, a line with a solid circle at the end connects to a sequence of five rounded rectangles: "txGrant", "talkingParty", "encryption", "txPriority", and "txInterrupt". Each of these five rectangles has a small circle with a plus sign (+) to its right. Below this sequence, another line with a solid circle at the end connects to a rounded rectangle containing the text: "Defines whether previous speaker's speech item was interrupted by this speech item. Valid only when txGrant is...". Further down, another line with a solid circle at the end connects to a rounded rectangle containing the text: "Timer to repeat the PTT. Units are seconds. Always suggested by the DF-gateway. Only valid when txGrant=granted.". At the bottom, a line with a solid circle at the end connects to a rounded rectangle containing the text: "Id of the currently speaking workstation, used for \"neighbours\" feature. Only valid when txGrant=granted and when...".</p>
Used by	Element
	typeCallPTTEvent/granted
Model	txGrant , talkingParty{0,1} , encryption{0,1} , txPriority{0,1} , txInterrupt{0,1} , txRepeat{0,1} , workstationId{0,1}

Children	encryption, talkingParty, txGrant, txInterrupt, txPriority, txRepeat, workstationId
Source	<pre> <xs:complexType name="typeTxGranted"> <xs:annotation> <xs:documentation>This event is used to inform of granted transmission request. This event is called when the transmission is granted to client, to another user or the request has been queued.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="txGrant" type="typeTxGrant"/> <xs:element name="talkingParty" type="typeAddress" minOccurs="0"/> <xs:element name="encryption" type="xs:boolean" default="true" minOccurs="0"/> <xs:element name="txPriority" type="typeTxPriority" minOccurs="0" default="normal"/> <xs:element name="txInterrupt" type="xs:boolean" default="false" minOccurs="0"> <xs:annotation> <xs:documentation>Defines whether previous speaker's speech item was interrupted by this speech item. Valid only when txGrant is granted2another.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="txRepeat" type="xs:unsignedLong" minOccurs="0" default="0"> <xs:annotation> <xs:documentation>Timer to repeat the PTT. Units are seconds. Always suggested by the DF-gateway. Only valid when txGrant=granted.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="workstationId" type="xs:normalizedString" minOccurs="0"> <xs:annotation> <xs:documentation>Id of the currently speaking workstation, used for "neighbours" feature. Only valid when txGrant=granted and when supplied by the DF-client in PTT request.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType></pre>

Complex Type typeEmpty

Namespace	DR-GW
Annotations	Explicit type specification for elements that shall be empty.
Diagram	<p>Explicit type specification for elements that shall be empty.</p>
Used by	Elements typeCallPTTEvent/ceased, typeCallPTTEvent/wait
Source	<pre> <xs:complexType name="typeEmpty"> <xs:annotation> <xs:documentation>Explicit type specification for elements that shall be empty.</xs:documentation> </xs:annotation> </xs:complexType></pre>

Complex Type typeCallUnitInEmergencyEvent

Namespace	DR-GW
Annotations	This event is used to inform DF-Client that a subscriber is in emergency state during an emergency group call including the ending of its emergency situation. Also queuing of emergency speech item requests is indicated using this event. Event is fired every time the TETRA system informs the Gateway that subscriber's emergency information is changed. For example, based on this information TCS Client could use pre-emptive speech item to request the current speaker to stop in order to let the unit in emergency to speak.

Diagram	<pre> classDiagram typeEvent { requestId result } typeCallUnitInEmergencyEvent { group tetraCallId unitInEmg unitInEmgType emgInfo tstamp } typeEvent < -- typeCallUnitInEmergencyEvent note over typeCallUnitInEmergencyEvent: This event is used to inform DF-Client that a subscriber is in emergency state during an emergency group call including... </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeCallUnitInEmergencyEvent
Used by	Element interfaceCall/unitInEmergencyEvent
Model	requestId{0,1} , result{0,1} , group , tetraCallId{0,1} , unitInEmg , unitInEmgType , emgInfo , tstamp
Children	emgInfo, group, requestId, result, tetraCallId, tstamp, unitInEmg, unitInEmgType
Source	<pre> <xs:complexType name="typeCallUnitInEmergencyEvent"> <xs:annotation> <xs:documentation>This event is used to inform DF-Client that a subscriber is in emergency state during an emergency group call including the ending of its emergency situation. Also queuing of emergency speech item requests is indicated using this event. Event is fired every time the TETRA system informs the Gateway that subscriber's emergency information is changed. For example, based on this information TCS Client could use pre-emptive speech item to request the current speaker to stop in order to let the unit in emergency to speak.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="tetraCallId" type="typeTetraCallId" minOccurs="0"/> <xs:element name="unitInEmg" type="typeAddress"/> <xs:element name="unitInEmgType" type="typeUnitInEmergencyType"/> <xs:element name="emgInfo" type="typeEmergencyInfo"/> <xs:element name="tstamp" type="xs:dateTime"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeCallKeyExchangeEvent

Namespace	DR-GW
Annotations	Events regarding key exchange. Contains the complete key exchange information from the BOS-Simcard.

Diagram	<pre> classDiagram typeEvent { requestId result } typeCallKeyExchangeEvent { state code priority interaction text tone } typeEvent < -- typeCallKeyExchangeEvent state --> "Represents the current key exchange state." code --> "Represents the current key exchange state." priority --> "If user interaction is required, then the message box should be visible until the interaction is made, otherwise should..." interaction --> "If user interaction is required, then the message box should be visible until the interaction is made, otherwise should..." text --> "If user interaction is required, then the message box should be visible until the interaction is made, otherwise should..." tone --> "If user interaction is required, then the message box should be visible until the interaction is made, otherwise should..." </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeCallKeyExchangeEvent
Used by	Element interfaceCall/keyExchangeEvent
Model	requestId{0,1} , result{0,1} , state , code , ((priority{0,1} , interaction{0,1} , text) tone)
Children	code, interaction, priority, requestId, result, state, text, tone
Source	<pre> <xs:complexType name="typeCallKeyExchangeEvent"> <xs:annotation> <xs:documentation>Events regarding key exchange. Contains the complete key exchange information from the BOS-Simcard.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="state" type="typeKeyExchangeState"> <xs:annotation> <xs:documentation>Represents the current key exchange state.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="code" type="typeKeyExchangeCode" /> <xs:choice> <xs:sequence> <xs:element name="priority" type="typeKeyExchangeTextPriority" minOccurs="0" default="normal"/> <xs:element name="interaction" type="xs:boolean" minOccurs="0" default="false"> <xs:annotation> <xs:documentation>If user interaction is required, then the message box should be visible until the interaction is made, otherwise should be hidden after delay.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="text" type="typeKeyExchangeText" /> </xs:sequence> <xs:element name="tone" type="xs:boolean" fixed="true" /> </xs:choice> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type interfaceSession

Namespace	DR-GW
Annotations	DR-GW-Session. In order to use the rest of DR-GW interface the DF-Client must establish a DR-GW session and maintain it using supervise in a timely manner. Use only via SOAP.

Diagram	
Used by	Element drgw/session
Model	login logout supervise check response loginEvent superviseEvent
Children	check, login, loginEvent, logout, response, supervise, superviseEvent
Source	<pre><xs:complexType name="interfaceSession"> <xs:annotation> <xs:documentation>DR-GW-Session. In order to use the rest of DR-GW interface the DF-Client must establish a DR-GW session and maintain it using supervise in a timely manner. Use only via SOAP.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="login" type="typeSessionLogin"/> <xs:element name="logout" type="typeSessionLogout"/> <xs:element name="supervise" type="typeSessionSupervise"/> <xs:element name="check" type="typeSessionCheck"/> <xs:element name="response" type="typeResponse"/> <xs:element name="loginEvent" type="typeSessionLoginEvent"/> <xs:element name="superviseEvent" type="typeSessionSuperviseEvent"/> </xs:choice> </xs:complexType></pre>

Complex Type typeSessionLogin

Namespace	DR-GW
Annotations	Login procedure. The username, password and the complete authentication is done using mechanisms of the transport protocol, digest access authentication on HTTP.
Diagram	
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSessionLogin
Used by	Element interfaceSession/login
Model	requestId , clientid , supervise{0,1} , version{0,1}
Children	clientid, requestId, supervise, version
Source	<pre><xs:complexType name="typeSessionLogin"> <xs:annotation></pre>

```

<xs:documentation>Login procedure. The username, password and the complete authentication
is done using mechanisms of the transport protocol, digest access authentication on HTTP.</
xs:documentation>
</xs:annotation>
<xs:complexContent>
  <xs:extension base="typeRequest">
    <xs:sequence>
      <xs:element name="clientId" type="xs:string"/>
      <xs:element name="supervise" type="typeSuperviseTimeout" default="60" minOccurs="0"/>
      <xs:element name="version" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:extension>
</xs:complexContent>

```

Complex Type typeSessionLogout

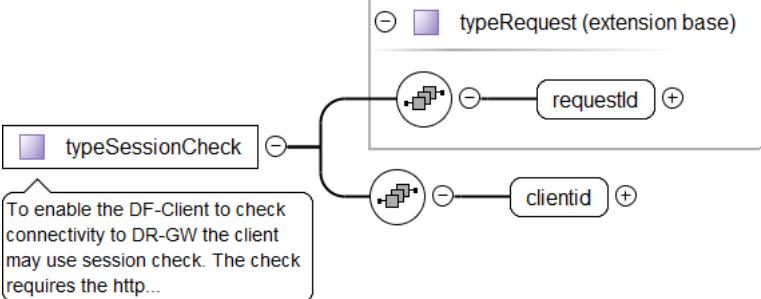
Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRequest < -- typeSessionLogout typeRequest "1..1" -- "1..1" typeSessionLogout typeRequest "1..1" -- "1..1" requestId typeSessionLogout "1..1" -- "1..1" requestId typeRequest "1..1" -- "1..1" clientId typeRequest "1..1" -- "1..1" supervise typeRequest "1..1" -- "1..1" version typeSessionLogout "1..1" -- "1..1" clientId typeSessionLogout "1..1" -- "1..1" supervise typeSessionLogout "1..1" -- "1..1" version </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSessionLogout
Used by	Element interfaceSession/logout
Model	requestId
Children	requestId
Source	<pre> <xs:complexType name="typeSessionLogout"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeSessionSupervise

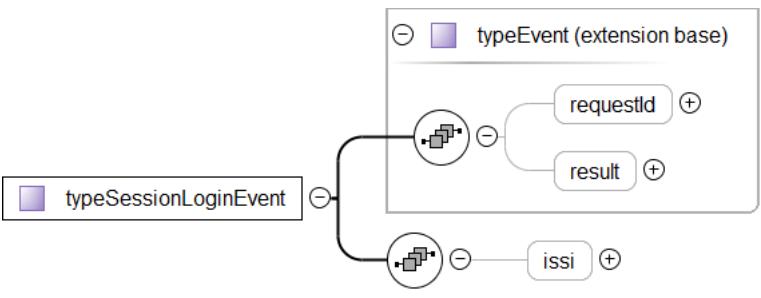
Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRequest < -- typeSessionSupervise typeRequest "1..1" -- "1..1" typeSessionSupervise typeRequest "1..1" -- "1..1" requestId typeRequest "1..1" -- "1..1" clientId typeRequest "1..1" -- "1..1" supervise typeRequest "1..1" -- "1..1" version typeSessionSupervise "1..1" -- "1..1" requestId typeSessionSupervise "1..1" -- "1..1" clientId typeSessionSupervise "1..1" -- "1..1" supervise typeSessionSupervise "1..1" -- "1..1" version </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSessionSupervise
Used by	Element interfaceSession/supervise
Model	requestId
Children	requestId
Source	<pre> <xs:complexType name="typeSessionSupervise"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

<pre></xs:complexType></pre>

Complex Type typeSessionCheck

Namespace	DR-GW
Annotations	To enable the DF-Client to check connectivity to DR-GW the client may use session check. The check requires the http authentication as it would be needed for a session login. Once the session check is OK, the client should be able to login later on. There is no resource allocation associated with the session check.
Diagram	 <pre> classDiagram typeRequest < -- typeSessionCheck typeRequest { -requestId -clientid } typeSessionCheck { note "To enable the DF-Client to check connectivity to DR-GW the client may use session check. The check requires the http..." } </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSessionCheck
Used by	Elements check, interfaceSession/check
Model	requestId , clientid
Children	clientid, requestId
Source	<pre> <xs:complexType name="typeSessionCheck"> <xs:annotation> <xs:documentation>To enable the DF-Client to check connectivity to DR-GW the client may use session check. The check requires the http authentication as it would be needed for a session login. Once the session check is OK, the client should be able to login later on. There is no resource allocation associated with the session check.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="clientid" type="xs:string"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeSessionLoginEvent

Namespace	DR-GW
Annotations	
Diagram	 <pre> classDiagram typeEvent < -- typeSessionLoginEvent typeEvent { -requestId -result } typeSessionLoginEvent { -issi } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSessionLoginEvent
Used by	Element interfaceSession/loginEvent

Model	requestId{0,1} , result{0,1} , issi{0,1}
Children	issi, requestId, result
Source	<pre><xs:complexType name="typeSessionLoginEvent"> <xs:annotation> <xs:documentation> </xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="issi" type="xs:string" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeSessionSuperviseEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeSessionSuperviseEvent typeEvent { +requestId +result } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSessionSuperviseEvent
Used by	Element interfaceSession/superviseEvent
Model	requestId{0,1} , result{0,1}
Children	requestId, result
Source	<pre><xs:complexType name="typeSessionSuperviseEvent"> <xs:annotation> <xs:documentation> </xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent" /> </xs:complexContent> </xs:complexType></pre>

Complex Type interfaceSds

Namespace	DR-GW
Annotations	DR-GW-Sds. Use to send/receive SDS messages. Use only via SOAP.
Diagram	<pre> interfaceSds { +send +sendReport +response +sendEvent +receiveEvent +reportEvent } </pre> <p>DR-GW-Sds. Use to send/receive SDS messages. Use only via SOAP.</p>
Used by	Element drgw/sds

Model	send sendReport response sendEvent receiveEvent reportEvent
Children	receiveEvent, reportEvent, response, send, sendEvent, sendReport
Source	<pre><xs:complexType name="interfaceSds"> <xs:annotation> <xs:documentation>DR-GW-Sds. Use to send/receive SDS messages. Use only via SOAP.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="send" type="typeSdsSend"/> <xs:element name="sendReport" type="typeSdsSendReport"/> <xs:element name="response" type="typeResponse"/> <xs:element name="sendEvent" type="typeSdsSendEvent"/> <xs:element name="receiveEvent" type="typeSdsReceiveEvent"/> <xs:element name="reportEvent" type="typeSdsReportEvent"/> </xs:choice> </xs:complexType></pre>

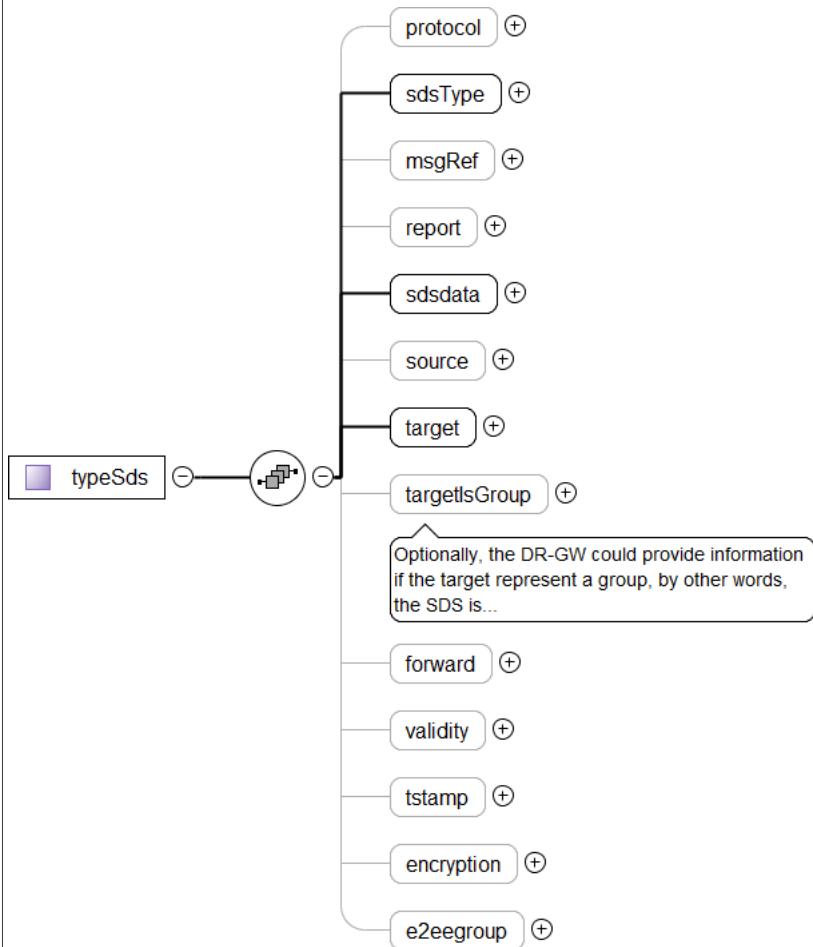
Complex Type typeSdsSend

Namespace	DR-GW
Annotations	This type is used to send SDS message.
Diagram	<pre> classDiagram typeRequest < -- typeSdsSend typeRequest "0..1" -- "1..1" requestId typeRequest "0..1" -- "1..1" sds note over typeSdsSend: This type is used to send SDS message. </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSdsSend
Used by	Element interfaceSds/send
Model	requestId , sds
Children	requestId, sds
Source	<pre><xs:complexType name="typeSdsSend"> <xs:annotation> <xs:documentation>This type is used to send SDS message.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="sds" type="typeSds"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeSds

Namespace	DR-GW
Annotations	

Diagram



Used by	Elements typeSdsReceiveEvent/sds, typeSdsSend/sds, typeSdsSendEvent/sds
Model	protocol{0,1} , sdsType , msgRef{0,1} , report{0,1} , sdsdata , source{0,1} , target , targetIsGroup{0,1} , forward{0,1} , validity{0,1} , tstamp{0,1} , encryption{0,1} , e2eegroup{0,1}
Children	e2eegroup, encryption, forward, msgRef, protocol, report, sdsType, sdsdata, source, target, targetIsGroup, tstamp, validity
Source	<pre> <xs:complexType name="typeSds"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="protocol" type="xs:unsignedByte" minOccurs="0"/> <xs:element name="sdsType" type="typeSdsType"/> <xs:element name="msgRef" type="xs:unsignedByte" minOccurs="0"/> <xs:element name="report" type="typeReport" default="none" minOccurs="0"/> <xs:element name="sdsdata" type="typeSdsData"/> <xs:element name="source" type="typeAddress" minOccurs="0"/> <xs:element name="target" type="typeAddress"/> <xs:element name="targetIsGroup" type="xs:boolean" minOccurs="0"> <xs:annotation> <xs:documentation>Optional, the DR-GW could provide information if the target represent a group, by other words, the SDS is group-addressed. Also the Df-Client can provide this information to help DR-GW decide which resource to use for actual sending.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="forward" type="typeAddress" minOccurs="0"/> <xs:element name="validity" type="xs:unsignedByte" minOccurs="0"/> <xs:element name="tstamp" type="xs:dateTime" minOccurs="0"/> <xs:element name="encryption" type="xs:boolean" default="true" minOccurs="0"/> <xs:element name="e2eegroup" type="typeSubscriberAddress" minOccurs="0"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeSdsData

Namespace	DR-GW
Annotations	2 ways of encoding the SDS. When sent from DF-Client to DF-Gateway at least

one node must be present, otherwise it will be discarded as not valid. When sent from DF-Gateway to DF-Client both nodes must be present, as it is unclear if the DF-Client supports the encoding inside raw "hexdata", so the readable decoded content must be present to. The default charset used within the "data" node is ISO-8859-15.

Diagram	<pre> classDiagram typeRequest < -- typeSdsData typeSdsData { <> data : string <> hexdata : hexBinary <> hexdatalength : integer } </pre> <p>2 ways of encoding the SDS. When sent from DF-Client to DF-Gateway at least one node must be present, otherwise it will...</p>
Used by	Element typeSds/sdsdata
Model	data{0,1} , hexdata{0,1} , hexdatalength{0,1}
Children	data, hexdata, hexdatalength
Source	<pre> <xs:complexType name="typeSdsData"> <xs:annotation> <xs:documentation>2 ways of encoding the SDS. When sent from DF-Client to DF-Gateway at least one node must be present, otherwise it will be discarded as not valid. When sent from DF-Gateway to DF-Client both nodes must be present, as it is unclear if the DF-Client supports the encoding inside raw "hexdata", so the readable decoded content must be present to. The default charset used within the "data" node is ISO-8859-15.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="data" type="xs:string" minOccurs="0"/> <xs:element name="hexdata" type="xs:hexBinary" minOccurs="0"/> <xs:element name="hexdatalength" type="xs:integer" minOccurs="0"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeSdsSendReport

Namespace	DR-GW
Annotations	An message reference is returned in the response for later message identification in case delivery and/or consume requests were asked. Is only valid for SDS-TL.
Diagram	<pre> classDiagram typeRequest < -- typeSdsSendReport typeSdsSendReport { <> requestId : string <> target : string <> msgRef : string <> deliveryStatus : string } </pre> <p>An message reference is returned in the response for later message identification in case delivery and/or consume...</p>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeSdsSendReport
Used by	Element interfaceSds/sendReport
Model	requestId , target , msgRef , deliveryStatus
Children	deliveryStatus, msgRef, requestId, target
Source	<pre> <xs:complexType name="typeSdsSendReport"> <xs:annotation> <xs:documentation>An message reference is returned in the response for later message identification in case delivery and/or consume requests were asked. Is only valid for SDS-TL.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> ... </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

```

<xs:element name="target" type="typeAddress" />
<xs:element name="msgRef" type="xs:unsignedByte" />
<xs:element name="deliveryStatus" type="xs:unsignedByte" />
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeSdsSendEvent

Namespace	DR-GW
Annotations	<p>This type is used as a final response to previous send request and contains final result if the message was sent or not. A message reference is returned in the response for later message identification in case delivery and/or consume requests were asked. Is valid only for SDS-TL.</p>
Diagram	<pre> classDiagram typeEvent < -- typeSdsSendEvent typeEvent "0..1" -- "0..1" msgRef typeEvent "0..1" -- "0..1" sds typeEvent "0..1" -- "0..1" requestId typeEvent "0..1" -- "0..1" result </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSdsSendEvent
Used by	Element interfaceSds/sendEvent
Model	requestId{0,1} , result{0,1} , msgRef{0,1} , sds
Children	msgRef, requestId, result, sds
Source	<pre> <xs:complexType name="typeSdsSendEvent"> <xs:annotation> <xs:documentation>This type is used as a final response to previous send request and contains final result if the message was sent or not. A message reference is returned in the response for later message identification in case delivery and/or consume requests were asked. Is valid only for SDS-TL.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="msgRef" type="xs:unsignedByte" minOccurs="0" default="0"/> <xs:element name="sds" type="typeSds" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeSdsReceiveEvent

Namespace	DR-GW
Annotations	ReceiveEvent is fired upon received SDS.
Diagram	<pre> classDiagram typeEvent < -- typeSdsReceiveEvent typeEvent "0..1" -- "0..1" sds typeEvent "0..1" -- "0..1" requestId typeEvent "0..1" -- "0..1" result </pre>

Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSdsReceiveEvent
Used by	Element interfaceSds/receiveEvent
Model	requestId{0,1} , result{0,1} , sds
Children	requestId, result, sds
Source	<pre><xs:complexType name="typeSdsReceiveEvent"> <xs:annotation> <xs:documentation>ReceiveEvent is fired upon received SDS.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="sds" type="typeSds" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeSdsReportEvent

Namespace	DR-GW
Annotations	ReportEvent is fired whenever the delivery or consume report is received.
Diagram	<pre> classDiagram typeEvent("typeEvent (extension base)") { requestId result } typeSdsReportEvent("typeSdsReportEvent") { source target msgRef deliveryStatus tstamp } typeEvent < -- typeSdsReportEvent note over typeSdsReportEvent: ReportEvent is fired whenever the delivery or consume report is received. </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSdsReportEvent
Used by	Element interfaceSds/reportEvent
Model	requestId{0,1} , result{0,1} , source , target , msgRef , deliveryStatus , tstamp
Children	deliveryStatus, msgRef, requestId, result, source, target, tstamp
Source	<pre><xs:complexType name="typeSdsReportEvent"> <xs:annotation> <xs:documentation>ReportEvent is fired whenever the delivery or consume report is received.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="source" type="typeAddress" /> <xs:element name="target" type="typeAddress" /> <xs:element name="msgRef" type="xs:unsignedByte" /> <xs:element name="deliveryStatus" type="xs:unsignedByte" /> <xs:element name="tstamp" type="xs:dateTime" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

```
</xs:complexContent>
</xs:complexType>
```

Complex Type interfaceStatus

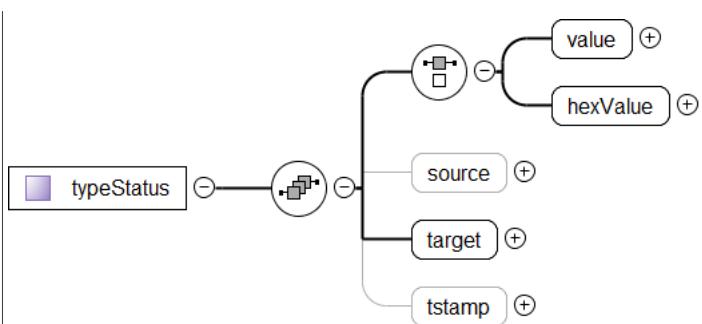
Namespace	DR-GW
Annotations	DR-GW-Status element. Use to send/receive status messages. Use only via SOAP.
Diagram	<pre> classDiagram typeRequest < -- interfaceStatus interfaceStatus < -- send interfaceStatus < -- response interfaceStatus < -- sendEvent interfaceStatus < -- receiveEvent </pre> <p>DR-GW-Status element. Use to send/receive status messages. Use only via SOAP.</p>
Used by	Element drgw/status
Model	send response sendEvent receiveEvent
Children	receiveEvent, response, send, sendEvent
Source	<pre> <xs:complexType name="interfaceStatus"> <xs:annotation> <xs:documentation>DR-GW-Status element. Use to send/receive status messages. Use only via SOAP.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="send" type="typeStatusSend"/> <xs:element name="response" type="typeResponse"/> <xs:element name="sendEvent" type="typeStatusSendEvent"/> <xs:element name="receiveEvent" type="typeStatusReceiveEvent"/> </xs:choice> </xs:complexType> </pre>

Complex Type typeStatusSend

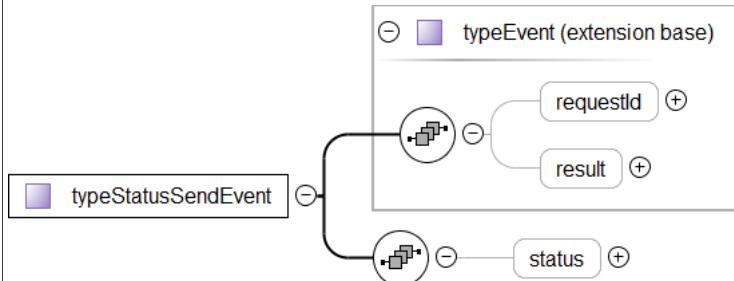
Namespace	DR-GW
Diagram	<pre> typeRequest < -- typeStatusSend typeStatusSend < -- requestId typeStatusSend < -- status </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeStatusSend
Used by	Element interfaceStatus/send
Model	requestId, status
Children	requestId, status
Source	<pre> <xs:complexType name="typeStatusSend"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="status" type="typeStatus"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeStatus

Namespace	DR-GW
-----------	-------

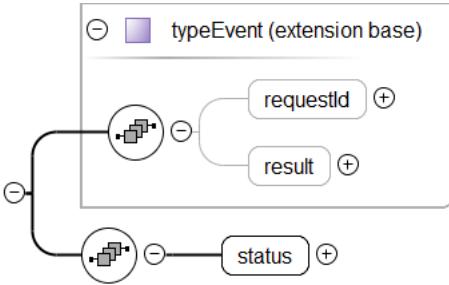
Diagram	
Used by	Elements typeStatusReceiveEvent/status, typeStatusSend/status, typeStatusSendEvent/status
Model	(value hexValue) , source{0,1} , target , tstamp{0,1}
Children	hexValue, source, target, tstamp, value
Source	<pre><xs:complexType name="typeStatus"> <xs:sequence> <xs:choice> <xs:element name="value" type="xs:unsignedShort"/> <xs:element name="hexValue" type="xs:hexBinary"/> </xs:choice> <xs:element name="source" type="typeAddress" minOccurs="0"/> <xs:element name="target" type="typeAddress"/> <xs:element name="tstamp" type="xs:dateTime" minOccurs="0"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeStatusSendEvent

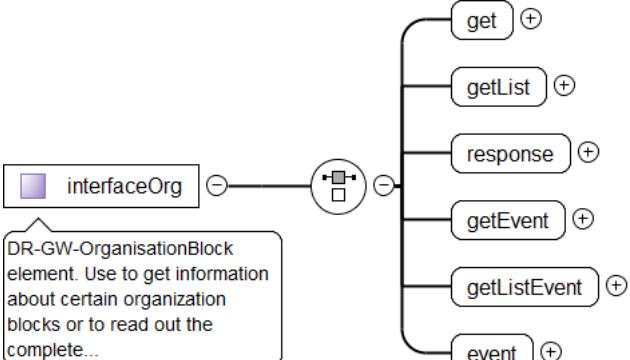
Namespace	DR-GW
Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeStatusSendEvent
Used by	Element interfaceStatus/sendEvent
Model	requestId{0,1} , result{0,1} , status{0,1}
Children	requestId, result, status
Source	<pre><xs:complexType name="typeStatusSendEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="status" type="typeStatus" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeStatusReceiveEvent

Namespace	DR-GW
-----------	-------

Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeStatusReceiveEvent
Used by	Element interfaceStatus/receiveEvent
Model	requestId{0,1} , result{0,1} , status
Children	requestId, result, status
Source	<pre><xs:complexType name="typeStatusReceiveEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="status" type="typeStatus" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type interfaceOrg

Namespace	DR-GW
Annotations	DR-GW-OrganisationBlock element. Use to get information about certain organization blocks or to read out the complete organization schema the DF-Client has rights for. Use only via SOAP.
Diagram	 <p>DR-GW-OrganisationBlock element. Use to get information about certain organization blocks or to read out the complete organization schema the DF-Client has rights for. Use only via SOAP.</p>
Used by	Element drgw/org
Model	get getList response getEvent getListEvent event
Children	event, get, getEvent, getList, getListEvent, response
Source	<pre><xs:complexType name="interfaceOrg"> <xs:annotation> <xs:documentation>DR-GW-OrganisationBlock element. Use to get information about certain organization blocks or to read out the complete organization schema the DF-Client has rights for. Use only via SOAP.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="get" type="typeOrgGet"/> <xs:element name="getList" type="typeOrgGetList"/> <xs:element name="response" type="typeResponse"/> <xs:element name="getEvent" type="typeOrgGetEvent"/> <xs:element name="getListEvent" type="typeOrgGetListEvent"/> <xs:element name="event" type="typeOrgEvent"/> </xs:choice> </xs:complexType></pre>

Complex Type typeOrgGet

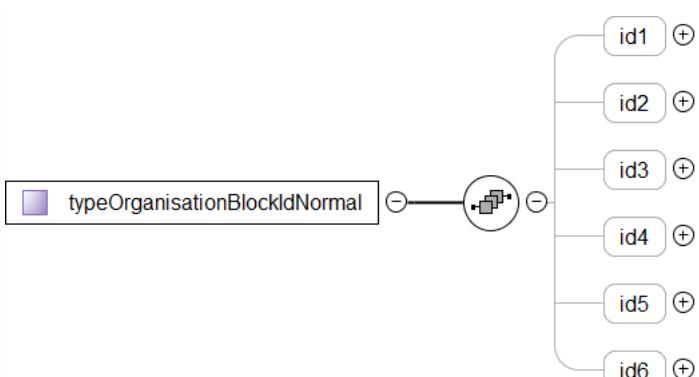
Namespace	DR-GW
Diagram	<pre> classDiagram typeRequest < -- typeOrgGet typeOrgGet { -requestId -orgblockId } </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeOrgGet
Used by	Element interfaceOrg/get
Model	requestId , orgblockId
Children	orgblockId, requestId
Source	<pre> <xs:complexType name="typeOrgGet"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="orgblockId" type="typeOrganisationBlockId"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeOrganisationBlockId

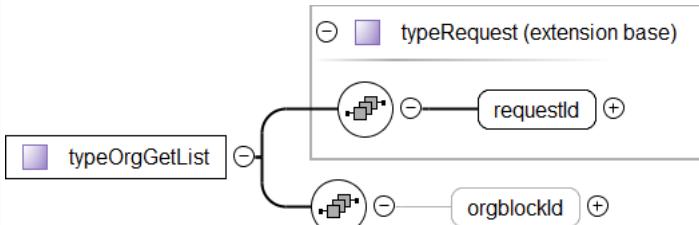
Namespace	DR-GW
Annotations	
Diagram	<pre> typeOrganisationBlockId { --orgblockId --orgblockIdSimple } </pre>
Used by	Elements typeAppGetList/orgblockId, typeApplication/orgblockId, typeGroup/orgblockId, typeGroupGetList/orgblockId, typeOrgGet/orgblockId, typeOrgGetList/orgblockId, typeOrganisationBlock/orgblockId, typeRadio/orgblockId, typeRadioGetList/orgblockId
Model	orgblockId orgblockIdSimple
Children	orgblockId, orgblockIdSimple
Source	<pre> <xs:complexType name="typeOrganisationBlockId"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:choice> <xs:element name="orgblockId" type="typeOrganisationBlockIdNormal"/> <xs:element name="orgblockIdSimple" type="typeOrganisationBlockIdSimple"/> </xs:choice> </xs:complexType> </pre>

Complex Type typeOrganisationBlockIdNormal

Namespace	DR-GW
Annotations	

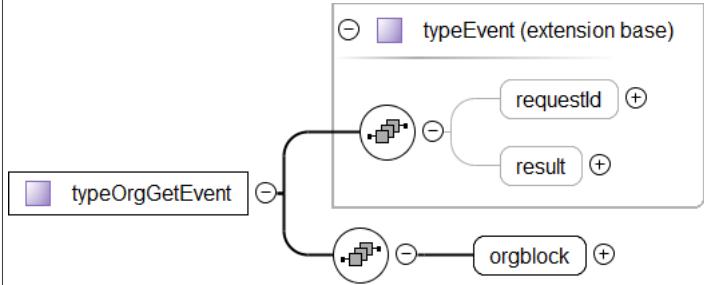
Diagram	
Used by	Element typeOrganisationBlockId/orgblockId
Model	id1{0,1} , id2{0,1} , id3{0,1} , id4{0,1} , id5{0,1} , id6{0,1}
Children	id1, id2, id3, id4, id5, id6
Source	<pre><xs:complexType name="typeOrganisationBlockIdNormal"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="id1" type="xs:unsignedShort" minOccurs="0"/> <xs:element name="id2" type="xs:unsignedShort" minOccurs="0"/> <xs:element name="id3" type="xs:unsignedShort" minOccurs="0"/> <xs:element name="id4" type="xs:unsignedShort" minOccurs="0"/> <xs:element name="id5" type="xs:unsignedShort" minOccurs="0"/> <xs:element name="id6" type="xs:unsignedShort" minOccurs="0"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeOrgGetList

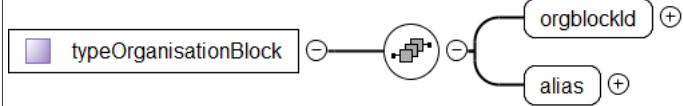
Namespace	DR-GW
Diagram	
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeOrgGetList
Used by	Element interfaceOrg/getList
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Source	<pre><xs:complexType name="typeOrgGetList"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeOrgGetEvent

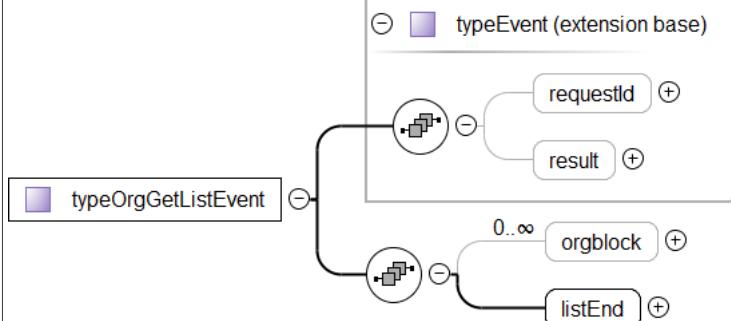
Namespace	DR-GW
-----------	-------

Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeOrgGetEvent
Used by	Element interfaceOrg/getEvent
Model	requestId{0,1} , result{0,1} , orgblock
Children	orgblock, requestId, result
Source	<pre><xs:complexType name="typeOrgGetEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="orgblock" type="typeOrganisationBlock"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeOrganisationBlock

Namespace	DR-GW
Annotations	
Diagram	
Used by	Elements typeOrgEvent/orgblock, typeOrgGetEvent/orgblock, typeOrgGetListEvent/orgblock
Model	orgblockId , alias
Children	alias, orgblockId
Source	<pre><xs:complexType name="typeOrganisationBlock"> <xs:annotation> <xs:documentation></xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="orgblockId" type="typeOrganisationBlockId"/> <xs:element name="alias" type="xs:normalizedString"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeOrgGetListEvent

Namespace	DR-GW
Diagram	

Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeOrgGetListEvent
Used by	Element interfaceOrg/getListEvent
Model	requestId{0,1} , result{0,1} , orgblock* , listEnd
Children	listEnd, orgblock, requestId, result
Source	<pre><xs:complexType name="typeOrgGetListEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="orgblock" type="typeOrganisationBlock" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="listEnd" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

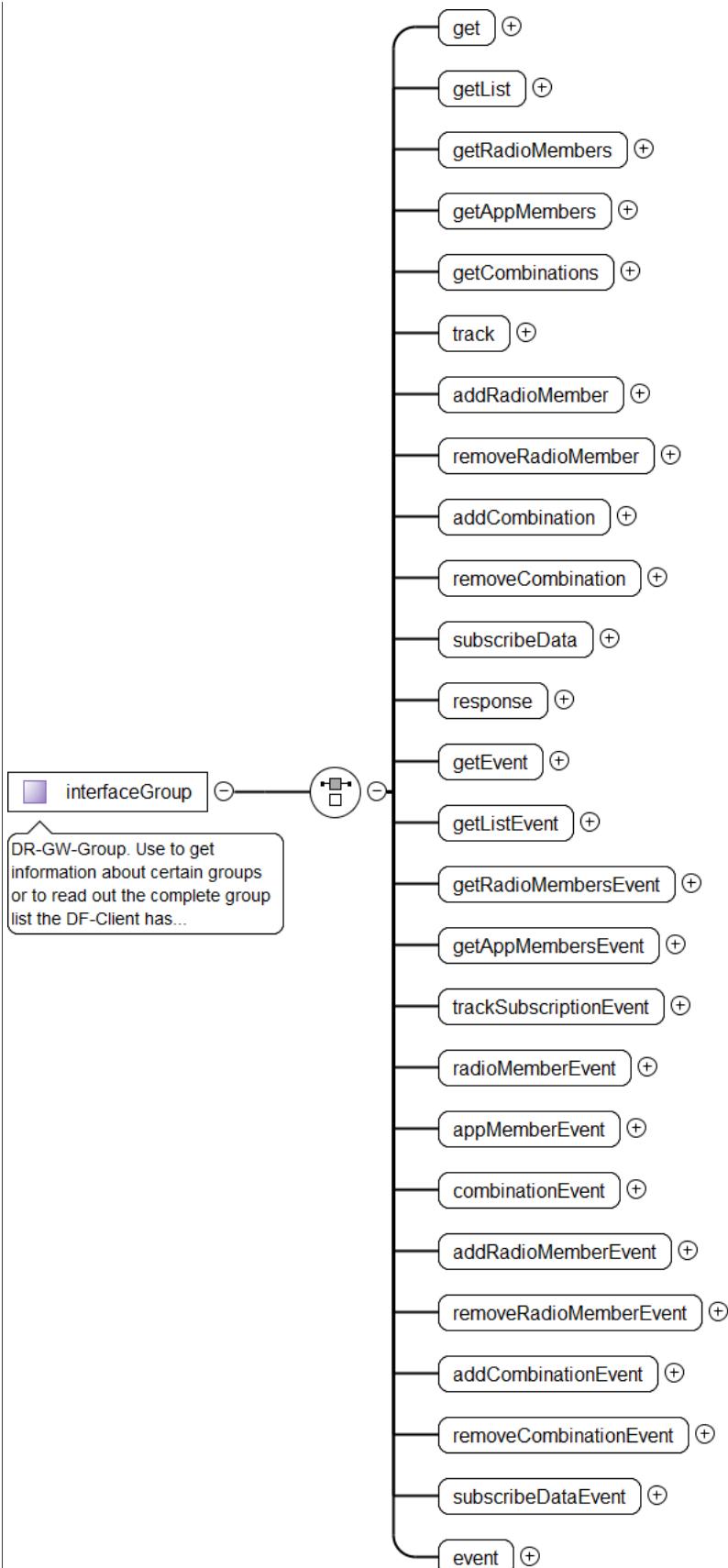
Complex Type typeOrgEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent { requestId result } typeOrgEvent { orgblock delete } typeEvent < -- typeOrgEvent </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeOrgEvent
Used by	Element interfaceOrg/event
Model	requestId{0,1} , result{0,1} , orgblock , delete
Children	delete, orgblock, requestId, result
Source	<pre><xs:complexType name="typeOrgEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="orgblock" type="typeOrganisationBlock"/> <xs:element name="delete" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type interfaceGroup

Namespace	DR-GW
Annotations	DR-GW-Group. Use to get information about certain groups or to read out the complete group list the DF-Client has rights for. There are also many other methods to execute over a certain group. See each method description. Use only via SOAP.

Diagram



Used by	Element	drgw/group
Model	get getList getRadioMembers getAppMembers getCombinations track addRadioMember removeRadioMember addCombination removeCombination subscribeData response getEvent getListEvent getRadioMembersEvent getAppMembersEvent	

	trackSubscriptionEvent radioMemberEvent appMemberEvent combinationEvent addRadioMemberEvent removeRadioMemberEvent addCombinationEvent removeCombinationEvent subscribeDataEvent event
Children	addCombination, addCombinationEvent, addRadioMember, addRadioMemberEvent, appMemberEvent, combinationEvent, event, get, getAppMembers, getAppMembersEvent, getCombinations, getEvent, getList, getListEvent, getRadioMembers, getRadioMembersEvent, radioMemberEvent, removeCombination, removeCombinationEvent, removeRadioMember, removeRadioMemberEvent, response, subscribeData, subscribeDataEvent, track, trackSubscriptionEvent
Source	<pre> <xs:complexType name="interfaceGroup"> <xs:annotation> <xs:documentation>DR-GW-Group. Use to get information about certain groups or to read out the complete group list the DF-Client has rights for. There are also many other methods to execute over a certain group. See each method description. Use only via SOAP.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="get" type="typeGroupGet"/> <xs:element name="getList" type="typeGroupGetList"/> <xs:element name="getRadioMembers" type="typeGroupGetRadioMembers"/> <xs:element name="getAppMembers" type="typeGroupGetAppMembers"/> <xs:element name="getCombinations" type="typeGroupGetCombinations"/> <xs:element name="track" type="typeGroupTrack"/> <xs:element name="addRadioMember" type="typeGroupAddRadioMember"/> <xs:element name="removeRadioMember" type="typeGroupRemoveRadioMember"/> <xs:element name="addCombination" type="typeGroupAddCombination"/> <xs:element name="removeCombination" type="typeGroupRemoveCombination"/> <xs:element name="subscribeData" type="typeGroupSubscribeData"/> <xs:element name="response" type="typeResponse"/> <xs:element name="getEvent" type="typeGroupGetEvent"/> <xs:element name="getListEvent" type="typeGroupGetListEvent"/> <xs:element name="getRadioMembersEvent" type="typeGroupGetRadioMembersEvent"/> <xs:element name="getAppMembersEvent" type="typeGroupGetAppMembersEvent"/> <xs:element name="trackSubscriptionEvent" type="typeGroupTrackSubscriptionEvent"/> <xs:element name="radioMemberEvent" type="typeGroupRadioMemberEvent"/> <xs:element name="appMemberEvent" type="typeGroupAppMemberEvent"/> <xs:element name="combinationEvent" type="typeGroupCombinationEvent"/> <xs:element name="addRadioMemberEvent" type="typeGroupAddRadioMemberEvent"/> <xs:element name="removeRadioMemberEvent" type="typeGroupRemoveRadioMemberEvent"/> <xs:element name="addCombinationEvent" type="typeGroupAddCombinationEvent"/> <xs:element name="removeCombinationEvent" type="typeGroupRemoveCombinationEvent"/> <xs:element name="subscribeDataEvent" type="typeGroupSubscribeDataEvent"/> <xs:element name="event" type="typeGroupEvent"/> </xs:choice> </xs:complexType></pre>

Complex Type typeGroupGet

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRequest "extension base" typeGroupGet "typeGroupGet" typeGroupGet --> typeRequest typeGroupGet "requestId" typeGroupGet "group" </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGet
Used by	Element interfaceGroup/get
Model	requestId , group
Children	group, requestId
Source	<pre> <xs:complexType name="typeGroupGet"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

```

    </xs:complexContent>
</xs:complexType>

```

Complex Type typeGroupGetList

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRequest < -- typeGroupGetList typeGroupGetList "0..1" --> requestId : String typeGroupGetList "0..1" --> orgblockId : String </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGetList
Used by	Element interfaceGroup/getList
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Source	<pre> <xs:complexType name="typeGroupGetList"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeGroupGetRadioMembers

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRequest < -- typeGroupGetRadioMembers typeGroupGetRadioMembers "0..1" --> requestId : String typeGroupGetRadioMembers "0..1" --> group : String </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGetRadioMembers
Used by	Element interfaceGroup/getRadioMembers
Model	requestId , group
Children	group, requestId
Source	<pre> <xs:complexType name="typeGroupGetRadioMembers"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

```

</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeGroupGetAppMembers

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRequest < -- typeGroupGetAppMembers typeGroupGetAppMembers "1" --> requestId typeGroupGetAppMembers "1" --> group </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGetAppMembers
Used by	Element interfaceGroup/getAppMembers
Model	requestId , group
Children	group, requestId
Source	<pre> <xs:complexType name="typeGroupGetAppMembers"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeGroupGetCombinations

Namespace	DR-GW
Annotations	The method requests the groups that belong to the same combined group as the group specified.
Diagram	<pre> classDiagram typeRequest < -- typeGroupGetCombinations typeGroupGetCombinations "1" --> requestId typeGroupGetCombinations "1" --> group </pre> <p>The method requests the groups that belong to the same combined group as the group specified.</p>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupGetCombinations
Used by	Element interfaceGroup/getCombinations
Model	requestId , group
Children	group, requestId
Source	<pre> <xs:complexType name="typeGroupGetCombinations"> <xs:annotation> <xs:documentation>The method requests the groups that belong to the same combined group as the group specified.</xs:documentation> </xs:annotation> </xs:complexType> </pre>

```

<xs:complexContent>
  <xs:extension base="typeRequest">
    <xs:sequence>
      <xs:element name="group" type="typeSubscriberAddress"/>
    </xs:sequence>
  </xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeGroupTrack

Namespace	DR-GW
Annotations	
Diagram	<pre> graph TD typeRequest["typeRequest (extension base)"] typeGroupTrack["typeGroupTrack"] requestId["requestId"] group["group"] mask["mask"] stop["stop"] typeGroupTrack --> typeRequest typeGroupTrack -.- requestId typeGroupTrack -.- group typeGroupTrack -.- mask typeGroupTrack -.- stop </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupTrack
Used by	Element interfaceGroup/track
Model	requestId , group , mask , stop
Children	group, mask, requestId, stop
Source	<pre> <xs:complexType name="typeGroupTrack"> <xs:annotation> <xs:documentation> Requests the addition of a radio subscriber to a group. This might cause DGNA operation in the air interface. </xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="mask" type="typeGroupTrackingMask"/> <xs:element name="stop" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeGroupAddRadioMember

Namespace	DR-GW
Annotations	<p>Requests the addition of a radio subscriber to a group. This might cause DGNA operation in the air interface.</p>
Diagram	<pre> graph TD typeRequest["typeRequest (extension base)"] typeGroupAddRadioMember["typeGroupAddRadioMember"] requestId["requestId"] radio["radio"] group["group"] membership["membership"] typeGroupAddRadioMember --> typeRequest typeGroupAddRadioMember -.- requestId typeGroupAddRadioMember -.- radio typeGroupAddRadioMember -.- group typeGroupAddRadioMember -.- membership </pre> <p>Requests the addition of a radio subscriber to a group. This might cause DGNA operation in the air interface.</p>

Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupAddRadioMember
Used by	Element interfaceGroup/addRadioMember
Model	requestId , radio , group , membership{0,1}
Children	group, membership, radio, requestId
Source	<pre><xs:complexType name="typeGroupAddRadioMember"> <xs:annotation> <xs:documentation>Requests the addition of a radio subscriber to a group. This might cause DGNA operation in the air interface.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="membership" type="typeMembershipType" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupRemoveRadioMember

Namespace	DR-GW
Annotations	Requests removing a radio subscriber from a group. This might cause DGNA operation in the air interface.
Diagram	<pre> classDiagram typeRequest --> typeGroupRemoveRadioMember typeGroupRemoveRadioMember "Requests removing a radio subscriber from a group. This might cause DGNA operation in the air interface." typeGroupRemoveRadioMember <> requestId typeGroupRemoveRadioMember <> radio typeGroupRemoveRadioMember <> group </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupRemoveRadioMember
Used by	Element interfaceGroup/removeRadioMember
Model	requestId , radio , group
Children	group, radio, requestId
Source	<pre><xs:complexType name="typeGroupRemoveRadioMember"> <xs:annotation> <xs:documentation>Requests removing a radio subscriber from a group. This might cause DGNA operation in the air interface.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="group" type="typeSubscriberAddress"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupAddCombination

Namespace	DR-GW
Annotations	Requests the addition of a group to a combined group.

Diagram	<pre> classDiagram typeRequest < -- typeGroupAddCombination typeGroupAddCombination { requestId group baseGroup force } </pre> <p>Requests the addition of a group to a combined group.</p>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupAddCombination
Used by	Element interfaceGroup/addCombination
Model	requestId , group , baseGroup , force{0,1}
Children	baseGroup, force, group, requestId
Source	<pre> <xss:complexType name="typeGroupAddCombination"> <xss:annotation> <xss:documentation>Requests the addition of a group to a combined group.</xss:documentation> </xss:annotation> <xss:complexContent> <xss:extension base="typeRequest"> <xss:sequence> <xss:element name="group" type="typeSubscriberAddress"/> <xss:element name="baseGroup" type="typeSubscriberAddress"/> <xss:element name="force" type="xs:boolean" minOccurs="0" default="true"/> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

Complex Type typeGroupRemoveCombination

Namespace	DR-GW
Annotations	Requests removing a group from a combined group.
Diagram	<pre> classDiagram typeRequest < -- typeGroupRemoveCombination typeGroupRemoveCombination { requestId group baseGroup } </pre> <p>Requests removing a group from a combined group.</p>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupRemoveCombination
Used by	Element interfaceGroup/removeCombination
Model	requestId , group , baseGroup
Children	baseGroup, group, requestId
Source	<pre> <xss:complexType name="typeGroupRemoveCombination"> <xss:annotation> <xss:documentation>Requests removing a group from a combined group.</xss:documentation> </xss:annotation> <xss:complexContent> <xss:extension base="typeRequest"> <xss:sequence> <xss:element name="group" type="typeSubscriberAddress"/> <xss:element name="baseGroup" type="typeSubscriberAddress"/> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

```

</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeGroupSubscribeData

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRequest < -- typeGroupSubscribeData typeRequest "1..1" --> requestId typeGroupSubscribeData "1..infinity" --> group </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeGroupSubscribeData
Used by	Element interfaceGroup/subscribeData
Model	requestId , group+
Children	group, requestId
Source	<pre> <xs:complexType name="typeGroupSubscribeData"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="group" type="typeGroupDataSubscription" maxOccurs="unbounded" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeGroupDataSubscription

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeGroupDataSubscription "1..1" --> addr typeGroupDataSubscription "1..1" --> useSDS typeGroupDataSubscription "1..1" --> useStatus </pre>
Used by	Elements typeGroupSubscribeData/group, typeGroupSubscribeDataEvent/group
Model	addr , useSDS , useStatus
Children	addr, useSDS, useStatus
Source	<pre> <xs:complexType name="typeGroupDataSubscription"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="addr" type="typeSubscriberAddress" /> <xs:element name="useSDS" type="xs:boolean" /> <xs:element name="useStatus" type="xs:boolean" /> </xs:sequence> </xs:complexType> </pre>

Complex Type typeGroupGetEvent

Namespace	DR-GW
-----------	-------

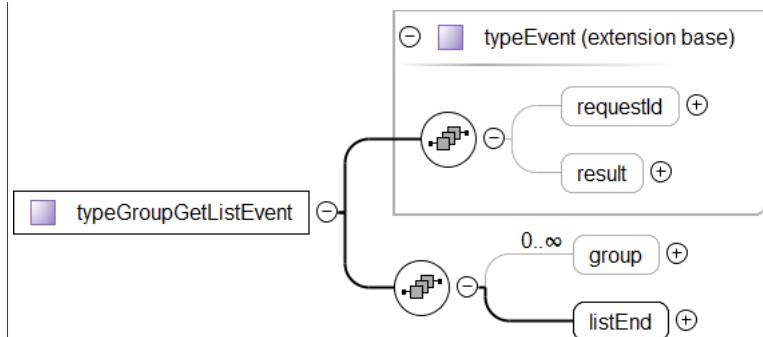
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeGroupGetEvent typeEvent "0..1" *--> requestId typeEvent "0..1" *--> result typeGroupGetEvent "0..1" *--> group </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupGetEvent
Used by	Element interfaceGroup/getEvent
Model	requestId{0,1} , result{0,1} , group
Children	group, requestId, result
Source	<pre> <xss:complexType name="typeGroupGetEvent"> <xss:annotation> <xss:documentation/> </xss:annotation> <xss:complexContent> <xss:extension base="typeEvent"> <xss:sequence> <xss:element name="group" type="typeGroup" /> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

Complex Type typeGroup

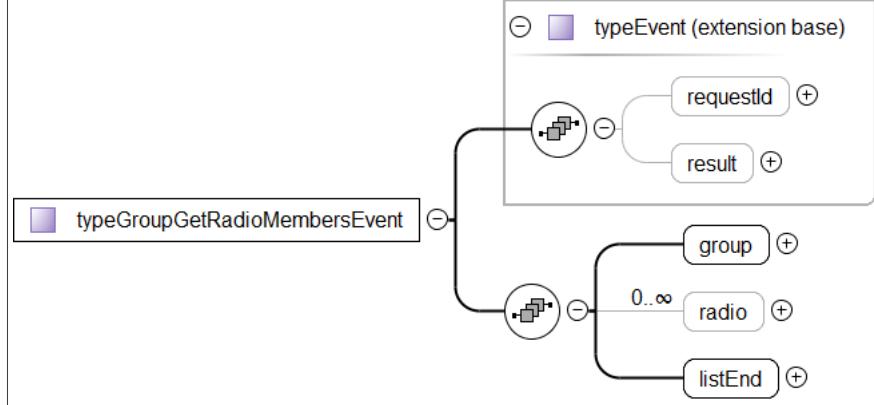
Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeGroup "0..1" *--> addr typeGroup "0..1" *--> alias typeGroup "0..1" *--> orgblockId </pre>
Used by	Elements typeGroupEvent/group, typeGroupGetEvent/group, typeGroupGetListEvent/group
Model	addr , alias , orgblockId
Children	addr, alias, orgblockId
Source	<pre> <xss:complexType name="typeGroup"> <xss:annotation> <xss:documentation/> </xss:annotation> <xss:sequence> <xss:element name="addr" type="typeSubscriberAddress" /> <xss:element name="alias" type="xs:normalizedString" /> <xss:element name="orgblockId" type="typeOrganisationBlockId" /> </xss:sequence> </xss:complexType> </pre>

Complex Type typeGroupGetListEvent

Namespace	DR-GW
Annotations	

Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupGetListEvent
Used by	Element interfaceGroup/getListEvent
Model	requestId{0,1} , result{0,1} , group* , listEnd
Children	group, listEnd, requestId, result
Source	<pre><xs:complexType name="typeGroupGetListEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeGroup" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="listEnd" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupGetRadioMembersEvent

Namespace	DR-GW
Annotations	
Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupGetRadioMembersEvent
Used by	Element interfaceGroup/getRadioMembersEvent
Model	requestId{0,1} , result{0,1} , group , radio* , listEnd
Children	group, listEnd, radio, requestId, result
Source	<pre><xs:complexType name="typeGroupGetRadioMembersEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeRadio" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="listEnd" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

```
<xs:extension base="typeEvent">
  <xs:sequence>
    <xs:element name="group" type="typeSubscriberAddress"/>
    <xs:element name="radio" type="typeSubscriberAddress" minOccurs="0" maxOccurs="unbounded" />
    <xs:element name="listEnd" type="xs:boolean"/>
  </xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
```

Complex Type typeGroupGetAppMembersEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeGroupGetAppMembersEvent typeEvent { -requestId -result +app +listEnd } typeGroupGetAppMembersEvent { +app } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeGroupGetAppMembersEvent
Used by	Element interfaceGroup/getAppMembersEvent
Model	requestId{0,1} , result{0,1} , app* , listEnd
Children	app, listEnd, requestId, result
Source	<pre> <xs:complexType name="typeGroupGetAppMembersEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="app" type="typeSubscriberAddress" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="listEnd" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeGroupTrackSubscriptionEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram class typeEvent { <<extension base>> +requestId +result } class typeGroupTrackSubscriptionEvent { <<extension>> -> typeEvent +group +mask +stop } typeEvent < -- typeGroupTrackSubscriptionEvent </pre> <p>The diagram illustrates an UML Class Diagram. It features two classes: typeEvent and typeGroupTrackSubscriptionEvent. The typeEvent class is labeled as an extension base with two attributes: requestId and result, both marked with a plus sign. A directed association connects typeEvent to typeGroupTrackSubscriptionEvent, indicated by a line with a hollow arrowhead and a minus sign. The typeGroupTrackSubscriptionEvent class has three attributes: group, mask, and stop, all marked with a plus sign.</p>

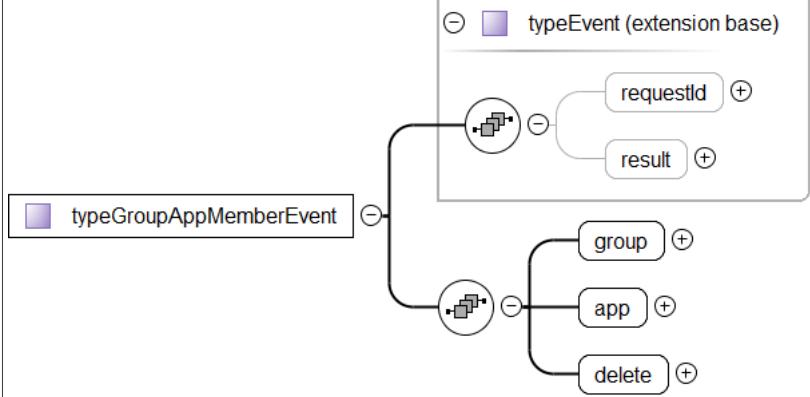
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupTrackSubscriptionEvent
Used by	Element interfaceGroup/trackSubscriptionEvent
Model	requestId{0,1} , result{0,1} , group , mask , stop
Children	group, mask, requestId, result, stop
Source	<pre><xs:complexType name="typeGroupTrackSubscriptionEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="mask" type="typeGroupTrackingMask"/> <xs:element name="stop" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupRadioMemberEvent

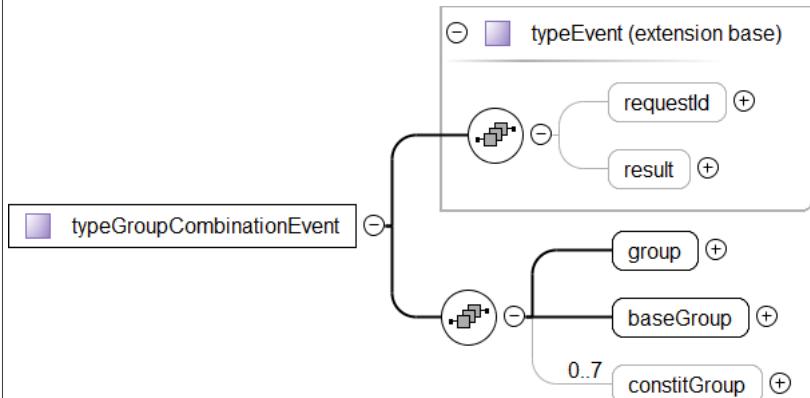
Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeGroupRadioMemberEvent typeEvent "0..1" -- "0..1" typeGroupRadioMemberEvent : requestId typeEvent "0..1" -- "0..1" typeGroupRadioMemberEvent : result typeEvent "0..1" -- "0..1" typeGroupRadioMemberEvent : group typeEvent "0..1" -- "0..1" typeGroupRadioMemberEvent : radio typeEvent "0..1" -- "0..1" typeGroupRadioMemberEvent : delete </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupRadioMemberEvent
Used by	Element interfaceGroup/radioMemberEvent
Model	requestId{0,1} , result{0,1} , group , radio , delete
Children	delete, group, radio, requestId, result
Source	<pre><xs:complexType name="typeGroupRadioMemberEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="delete" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupAppMemberEvent

Namespace	DR-GW
Annotations	

Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupAppMemberEvent
Used by	Element interfaceGroup/appMemberEvent
Model	requestId{0,1} , result{0,1} , group , app , delete
Children	app, delete, group, requestId, result
Source	<pre><xs:complexType name="typeGroupAppMemberEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="app" type="typeSubscriberAddress"/> <xs:element name="delete" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupCombinationEvent

Namespace	DR-GW
Annotations	
Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupCombinationEvent
Used by	Element interfaceGroup/combinationEvent
Model	requestId{0,1} , result{0,1} , group , baseGroup , constitGroup{0,7}
Children	baseGroup, constitGroup, group, requestId, result
Source	<pre><xs:complexType name="typeGroupCombinationEvent"></pre>

```

<xs:annotation>
  <xs:documentation/>
</xs:annotation>
<xs:complexType>
  <xs:extension base="typeEvent">
    <xs:sequence>
      <xs:element name="group" type="typeSubscriberAddress" />
      <xs:element name="baseGroup" type="typeSubscriberAddress" />
      <xs:element name="constitGroup" type="typeSubscriberAddress" minOccurs="0" maxOccurs="7" />
    </xs:sequence>
  </xs:extension>
</xs:complexType>
</xs:complexType>

```

Complex Type typeGroupAddRadioMemberEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent --> typeGroupAddRadioMemberEvent typeGroupAddRadioMemberEvent { requestId result radio group } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupAddRadioMemberEvent
Used by	Element interfaceGroup/addRadioMemberEvent
Model	requestId{0,1} , result{0,1} , radio , group
Children	group, radio, requestId, result
Source	<pre> <xs:complexType name="typeGroupAddRadioMemberEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="group" type="typeSubscriberAddress" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeGroupRemoveRadioMemberEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent --> typeGroupRemoveRadioMemberEvent typeGroupRemoveRadioMemberEvent { requestId result radio group } </pre>
Source	

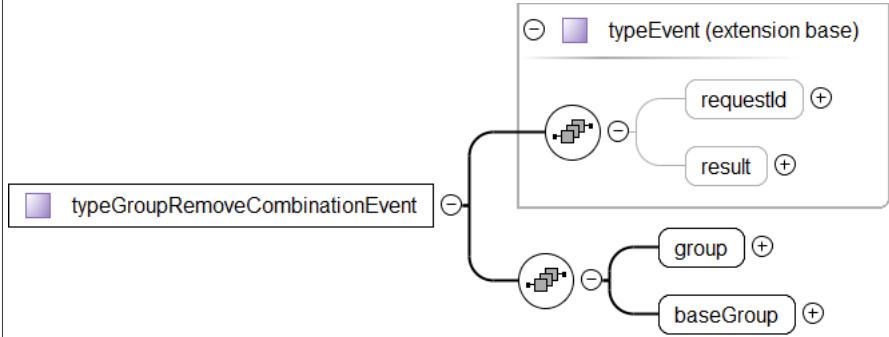
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeGroupRemoveRadioMemberEvent
Used by	Element interfaceGroup/removeRadioMemberEvent
Model	requestId{0,1} , result{0,1} , radio , group
Children	group, radio, requestId, result
Source	<pre> <xs:complexType name="typeGroupRemoveRadioMemberEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="group" type="typeSubscriberAddress"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupAddCombinationEvent

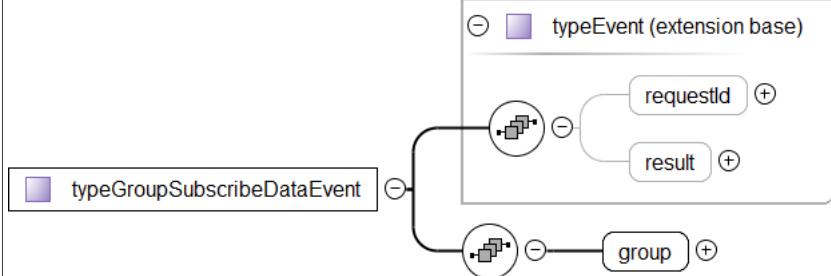
Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeGroupAddCombinationEvent typeEvent { -requestId -result -group -baseGroup } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeGroupAddCombinationEvent
Used by	Element interfaceGroup/addCombinationEvent
Model	requestId{0,1} , result{0,1} , group , baseGroup
Children	baseGroup, group, requestId, result
Source	<pre> <xss:complexType name="typeGroupAddCombinationEvent"> <xss:annotation> <xss:documentation/> </xss:annotation> <xss:complexContent> <xss:extension base="typeEvent"> <xss:sequence> <xss:element name="group" type="typeSubscriberAddress"/> <xss:element name="baseGroup" type="typeSubscriberAddress"/> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

Complex Type typeGroupRemoveCombinationEvent

Namespace	DR-GW
Annotations	

Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupRemoveCombinationEvent
Used by	Element interfaceGroup/removeCombinationEvent
Model	requestId{0,1} , result{0,1} , group , baseGroup
Children	baseGroup, group, requestId, result
Source	<pre><xs:complexType name="typeGroupRemoveCombinationEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="baseGroup" type="typeSubscriberAddress"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeGroupSubscribeDataEvent

Namespace	DR-GW
Annotations	
Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupSubscribeDataEvent
Used by	Element interfaceGroup/subscribeDataEvent
Model	requestId{0,1} , result{0,1} , group
Children	group, requestId, result
Source	<pre><xs:complexType name="typeGroupSubscribeDataEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeGroupDataSubscription"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

```

</xs:complexContent>
</xs:complexType>

```

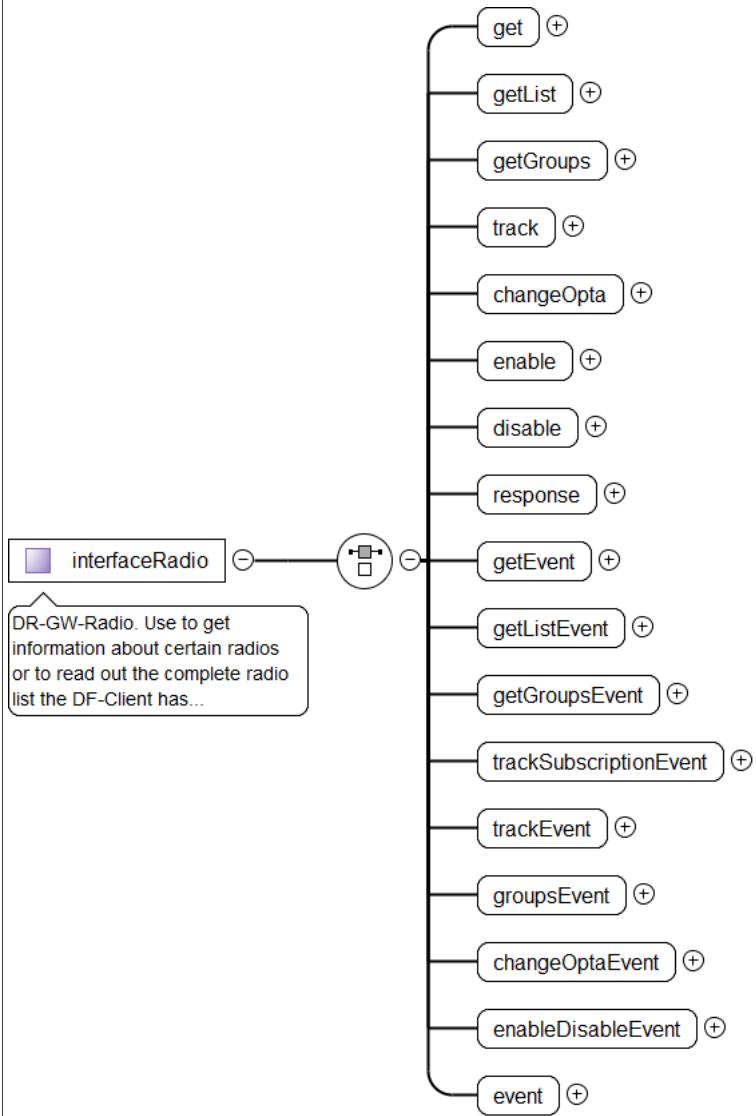
Complex Type typeGroupEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeGroupEvent typeEvent "0..1" --> requestId typeEvent "0..1" --> result typeGroupEvent "0..1" --> group typeGroupEvent "0..1" --> delete </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupEvent
Used by	Element interfaceGroup/event
Model	requestId{0,1} , result{0,1} , group , delete
Children	delete, group, requestId, result
Source	<pre> <xs:complexType name="typeGroupEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeGroup"/> <xs:element name="delete" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type interfaceRadio

Namespace	DR-GW
Annotations	DR-GW-Radio. Use to get information about certain radios or to read out the complete radio list the DF-Client has rights for. There are also many other methods to execute over a certain radio. See each method description. Use only via SOAP.

Diagram



Used by	Element drgw/radio
Model	get getList getGroups track changeOpta enable disable response getEvent getListEvent getGroupsEvent trackSubscriptionEvent trackEvent groupsEvent changeOptaEvent enableDisableEvent event
Children	changeOpta, changeOptaEvent, disable, enable, enableDisableEvent, event, get, getList, getEvent, getGroups, getGroupsEvent, getGroupsEvent, getListEvent, groupsEvent, response, track, trackEvent, trackSubscriptionEvent
Source	<pre> <xs:complexType name="interfaceRadio"> <xs:annotation> <xs:documentation>DR-GW-Radio. Use to get information about certain radios or to read out the complete radio list the DF-Client has rights for. There are also many other methods to execute over a certain radio. See each method description. Use only via SOAP.</xs:documentation> <xs:annotation> <xs:choice> <xs:element name="get" type="typeRadioGet"/> <xs:element name="getList" type="typeRadioGetList"/> <xs:element name="getGroups" type="typeRadioGetGroups"/> <xs:element name="track" type="typeRadioTrack"/> <xs:element name="changeOpta" type="typeRadioChangeOpta"/> <xs:element name="enable" type="typeRadioEnable"/> <xs:element name="disable" type="typeRadioDisable"/> <xs:element name="response" type="typeResponse"/> <xs:element name="getEvent" type="typeRadioGetEvent"/> <xs:element name="getListEvent" type="typeRadioGetListEvent"/> <xs:element name="getGroupsEvent" type="typeRadioGetGroupsEvent"/> <xs:element name="trackSubscriptionEvent" type="typeRadioTrackSubscriptionEvent"/> <xs:element name="trackEvent" type="typeRadioTrackEvent"/> <xs:element name="groupsEvent" type="typeRadioGroupsEvent"/> <xs:element name="changeOptaEvent" type="typeRadioChangeOptaEvent"/> <xs:element name="enableDisableEvent" type="typeRadioEnableDisableEvent"/> <xs:element name="event" type="typeRadioEvent"/> </xs:choice> </xs:annotation> </pre>

```

</xs:choice>
</xs:complexType>

```

Complex Type typeRadioGet

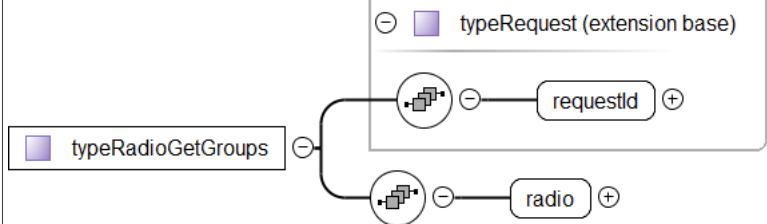
Namespace	DR-GW
Diagram	<pre> classDiagram typeRequest < -- typeRadioGet typeRequest "1..1" requestId typeRequest "1..1" radio typeRadioGet "1..1" requestId typeRadioGet "1..1" radio </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioGet
Used by	Element interfaceRadio/get
Model	requestId , radio
Children	radio, requestId
Source	<pre> <xs:complexType name="typeRadioGet"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeRadioGetList

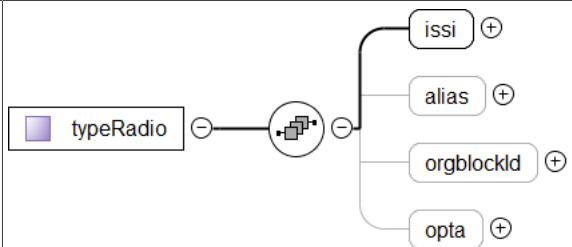
Namespace	DR-GW
Diagram	<pre> classDiagram typeRequest < -- typeRadioGetList typeRequest "1..1" requestId typeRequest "1..1" orgblockId typeRadioGetList "1..1" requestId typeRadioGetList "1..1" orgblockId </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioGetList
Used by	Element interfaceRadio/getList
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Source	<pre> <xs:complexType name="typeRadioGetList"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeRadioGetGroups

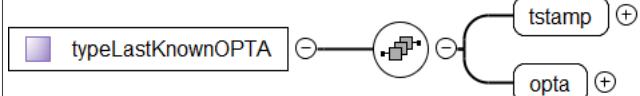
Namespace	DR-GW
-----------	-------

Diagram	
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioGetGroups
Used by	Element interfaceRadio/getGroups
Model	requestId , radio
Children	radio, requestId
Source	<pre><xs:complexType name="typeRadioGetGroups"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeRadio"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeRadio

Namespace	DR-GW
Annotations	
Diagram	
Used by	Elements typeRadioEvent/radio, typeRadioGetEvent/radio, typeRadioGetGroups/radio, typeRadioGetListEvent/radio
Model	issi , alias{0,1} , orgblockId{0,1} , opta{0,1}
Children	alias, issi, opta, orgblockId
Source	<pre><xs:complexType name="typeRadio"> <xs:annotation> <xs:documentation></xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="issi" type="typeSubscriberAddress"/> <xs:element name="alias" type="xs:normalizedString" minOccurs="0"/> <xs:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0"/> <xs:element name="opta" type="typeLastKnownOPTA" minOccurs="0"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeLastKnownOPTA

Namespace	DR-GW
Annotations	
Diagram	

Used by	Element	typeRadio/opta
Model	tstamp , opta	
Children	opta, tstamp	
Source		<pre><xs:complexType name="typeLastKnownOPTA"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="tstamp" type="xs:dateTime"/> <xs:element name="opta" type="typeOpta"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeRadioTrack

Namespace	DR-GW	
Diagram	<pre> classDiagram typeRequest < -- typeRadioTrack typeRequest { -requestId -radio -stop } typeRadioTrack { -radio -stop } </pre>	
Type	extension of typeRequest	
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioTrack 	
Used by	Element interfaceRadio/track	
Model	requestId , radio , stop	
Children	radio, requestId, stop	
Source		<pre><xs:complexType name="typeRadioTrack"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="stop" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeRadioChangeOpta

Namespace	DR-GW
Diagram	<pre> classDiagram typeRequest < -- typeRadioChangeOpta typeRequest { -requestId -radio -opta } typeRadioChangeOpta { -opta } </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioChangeOpta
Used by	Element interfaceRadio/changeOpta

Model	requestId , radio , opta
Children	opta, radio, requestId
Source	<pre><xs:complexType name="typeRadioChangeOpta"> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="opta" type="typeOpta" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeRadioEnable

Namespace	DR-GW
Annotations	This method is used to Enable the radio terminal over the air.
Diagram	<pre> classDiagram typeRequest "0..1" -- "1..1" typeRadioEnable typeRequest "0..1" -- "1..1" radio typeRadioEnable "0..1" -- "1..1" reason typeRadioEnable "0..1" -- "1..1" enable note over typeRadioEnable: This method is used to Enable the radio terminal over the air. </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioEnable
Used by	Element interfaceRadio/enable
Model	requestId , radio , radio , reason{0,1} , enable
Children	enable, radio, reason, requestId
Source	<pre><xs:complexType name="typeRadioEnable"> <xs:annotation> <xs:documentation>This method is used to Enable the radio terminal over the air.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="reason" type="xs:unsignedByte" minOccurs="0" /> <xs:element name="enable" type="xs:boolean" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeRadioDisable

Namespace	DR-GW
Annotations	<p>This method is used to disable the radio terminal over the air. If no reason is supplied, then the DF-Gateway sets the default reason. There is no default reason value, it depends on the DF-Gateway configuration what reason is used when no reason is supplied by DF-Client.</p> <p>See TCS API Description for all possible reasons for disabling.</p>

Diagram	<pre> classDiagram typeRequest < -- typeRadioDisable typeRequest "0..1" requestId typeRadioDisable "0..1" radio typeRadioDisable "0..1" reason typeRadioDisable "0..1" enable </pre> <p>This method is used to disable the radio terminal over the air. If no reason is supplied, then the DF-Gateway sets the...</p>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeRadioDisable
Used by	Element interfaceRadio/disable
Model	requestId , radio , radio , reason{0,1} , enable
Children	enable, radio, reason, requestId
Source	<pre> <xs:complexType name="typeRadioDisable"> <xs:annotation> <xs:documentation>This method is used to disable the radio terminal over the air. If no reason is supplied, then the DF-Gateway sets the default reason. There is no default reason value, it depends on the DF-Gateway configuration what reason is used when no reason is supplied by DF-Client. See TCS API Description for all possible reasons for disabling.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="reason" type="xs:unsignedByte" minOccurs="0" /> <xs:element name="enable" type="xs:boolean" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeRadioGetEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioGetEvent typeEvent "0..1" requestId typeEvent "0..1" result typeRadioGetEvent "0..1" radio </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioGetEvent
Used by	Element interfaceRadio/getEvent
Model	requestId{0,1} , result{0,1} , radio
Children	radio, requestId, result
Source	<pre> <xs:complexType name="typeRadioGetEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

```

<xs:element name="radio" type="typeRadio"/>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeRadioGetListEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioGetListEvent typeEvent { requestId result } typeRadioGetListEvent { radio * "0..∞" } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioGetListEvent
Used by	Element interfaceRadio/getListEvent
Model	requestId{0,1} , result{0,1} , radio* , listEnd
Children	listEnd, radio, requestId, result
Source	<pre> <xs:complexType name="typeRadioGetListEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeRadio" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="listEnd" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeRadioGetGroupsEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioGetGroupsEvent typeEvent { requestId result } typeRadioGetGroupsEvent { radio group * "0..∞" } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioGetGroupsEvent
Used by	Element interfaceRadio/getGroupsEvent
Model	requestId{0,1} , result{0,1} , radio , group* , listEnd
Children	group, listEnd, radio, requestId, result

Source	<pre> <xs:complexType name="typeRadioGetGroupsEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="group" type="typeRadioGroupSelection" minOccurs="0" maxOccurs="unbounded" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>
--------	---

Complex Type typeRadioGroupSelection

Namespace	DR-GW
Annotations	
Diagram	<pre> graph LR typeEvent["typeEvent (extension base)"] typeRadioGroupSelection["typeRadioGroupSelection"] typeRadioGroupSelection --> group typeRadioGroupSelection --> level group level </pre>
Used by	Elements typeRadioGetGroupsEvent/group, typeRadioGroupsEvent/group
Model	group , level
Children	group, level
Source	<pre> <xs:complexType name="typeRadioGroupSelection"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:element name="level" type="typeGroupSelectionLevel"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeRadioTrackSubscriptionEvent

Namespace	DR-GW
Diagram	<pre> graph LR typeEvent["typeEvent (extension base)"] typeRadioTrackSubscriptionEvent["typeRadioTrackSubscriptionEvent"] typeRadioTrackSubscriptionEvent --> requestId typeRadioTrackSubscriptionEvent --> result typeRadioTrackSubscriptionEvent --> radio typeRadioTrackSubscriptionEvent --> stop requestId result radio stop </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioTrackSubscriptionEvent
Used by	Element interfaceRadio/trackSubscriptionEvent
Model	requestId{0,1} , result{0,1} , radio , stop
Children	radio, requestId, result, stop
Source	<pre> <xs:complexType name="typeRadioTrackSubscriptionEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="stop" type="xs:boolean"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeRadioTrackEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioTrackEvent typeEvent "0..1" *--> requestId typeEvent "0..1" *--> result typeRadioTrackEvent "0..1" *--> trackingData </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioTrackEvent
Used by	Element interfaceRadio/trackEvent
Model	requestId{0,1} , result{0,1} , trackingData
Children	requestId, result, trackingData
Source	<pre> <xss:complexType name="typeRadioTrackEvent"> <xss:complexContent> <xss:extension base="typeEvent"> <xss:sequence> <xss:element name="trackingData" type="typeRadioTrackingData"/> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

Complex Type typeRadioTrackingData

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeRadioTrackingData "0..1" *--> radio typeRadioTrackingData "0..1" *--> registered typeRadioTrackingData "0..1" *--> exchangeId typeRadioTrackingData "0..1" *--> locationArea typeRadioTrackingData "0..1" *--> lastActive typeRadioTrackingData "0..1" *--> scanningOn typeRadioTrackingData "0..1" *--> status typeRadioTrackingData "0..1" *--> callType typeRadioTrackingData "0..1" *--> callParty typeRadioTrackingData "0..1" *--> dmoState typeRadioTrackingData "0..1" *--> emergency </pre>
Used by	Element typeRadioTrackEvent/trackingData
Model	radio , registered , exchangeId , locationArea{0,1} , lastActive , scanningOn , status , callType , callParty , dmoState , emergency

Children	callParty, callType, dmoState, emergency, exchangeId, lastActive, locationArea, radio, registered, scanningOn, status
Source	<pre><xs:complexType name="typeRadioTrackingData"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="registered" type="xs:boolean" /> <xs:element name="exchangeId" type="xs:unsignedLong" /> <xs:element name="locationArea" type="xs:unsignedShort" minOccurs="0" /> <xs:element name="lastActive" type="xs:dateTime" /> <xs:element name="scanningOn" type="xs:boolean" /> <xs:element name="status" type="typeStatusIndicator" /> <xs:element name="callType" type="typeCallType" /> <xs:element name="callParty" type="typeSubscriberAddress" /> <xs:element name="dmoState" type="xs:boolean" /> <xs:element name="emergency" type="xs:boolean" /> </xs:sequence> </xs:complexType></pre>

Complex Type typeStatusIndicator

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeStatusIndicator typeStatusIndicator { <<xs:element name="value" type="xs:unsignedLong"/>> <<xs:element name="time" type="xs:dateTime"/>> } </pre>
Used by	Element typeRadioTrackingData/status
Model	value , time
Children	time, value
Source	<pre><xs:complexType name="typeStatusIndicator"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="value" type="xs:unsignedLong" /> <xs:element name="time" type="xs:dateTime" /> </xs:sequence> </xs:complexType></pre>

Complex Type typeRadioGroupsEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioGroupsEvent typeRadioGroupsEvent { <<xs:element name="requestId" type="xs:string"/>> <<xs:element name="result" type="xs:string"/>> <<xs:element name="radio" type="typeRadio"/>> <<xs:element name="group" type="group" minOccurs="1..infinity"/>> <<xs:element name="deletedGroup" type="group" />> } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioGroupsEvent
Used by	Element interfaceRadio/groupsEvent
Model	requestId{0,1} , result{0,1} , radio , (group+ deletedGroup)
Children	deletedGroup, group, radio, requestId, result

Source	<pre><xs:complexType name="typeRadioGroupsEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:choice> <xs:element name="group" type="typeRadioGroupSelection" maxOccurs="unbounded"/> <xs:element name="deletedGroup" type="typeSubscriberAddress"/> </xs:choice> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>
--------	---

Complex Type typeRadioChangeOptaEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioChangeOptaEvent typeEvent { +requestId +result } typeRadioChangeOptaEvent { +radio +opta } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent <ul style="list-style-type: none"> • typeRadioChangeOptaEvent
Used by	Element interfaceRadio/changeOptaEvent
Model	requestId{0,1} , result{0,1} , radio , opta
Children	opta, radio, requestId, result
Source	<pre><xs:complexType name="typeRadioChangeOptaEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress"/> <xs:element name="opta" type="typeOpta"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeRadioEnableDisableEvent

Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioEnableDisableEvent typeEvent { +requestId +result } typeRadioEnableDisableEvent { +radio +reason +enabled +overTheAir } </pre>

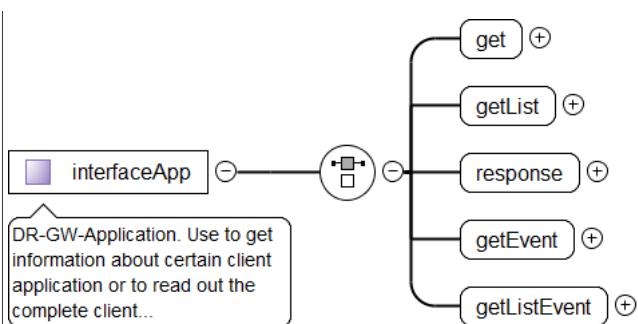
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioEnableDisableEvent
Used by	Element interfaceRadio/enableDisableEvent
Model	requestId{0,1} , result{0,1} , radio , reason{0,1} , enabled , overTheAir{0,1}
Children	enabled, overTheAir, radio, reason, requestId, result
Source	<pre><xs:complexType name="typeRadioEnableDisableEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeSubscriberAddress" /> <xs:element name="reason" type="xs:unsignedByte" minOccurs="0" /> <xs:element name="enabled" type="xs:boolean" /> <xs:element name="overTheAir" type="xs:boolean" minOccurs="0" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeRadioEvent

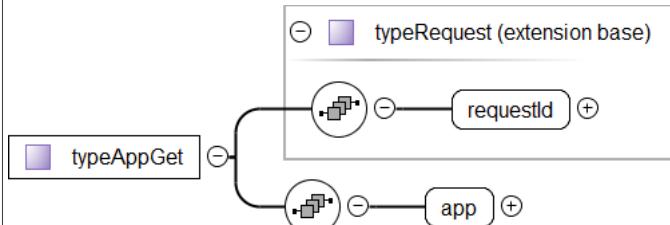
Namespace	DR-GW
Diagram	<pre> classDiagram typeEvent < -- typeRadioEvent typeEvent { -requestId -result } typeRadioEvent { -radio -delete } </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeRadioEvent
Used by	Element interfaceRadio/event
Model	requestId{0,1} , result{0,1} , radio , delete
Children	delete, radio, requestId, result
Source	<pre><xs:complexType name="typeRadioEvent"> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="radio" type="typeRadio" /> <xs:element name="delete" type="xs:boolean" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type interfaceApp

Namespace	DR-GW
Annotations	DR-GW-Application. Use to get information about certain client application or to read out the complete client application list the DF-Client has rights for. Use only via SOAP.

Diagram	
Used by	Element drgw/app
Model	get getList response getEvent getListEvent
Children	get, getList, getEvent, getListEvent, response
Source	<pre><xs:complexType name="interfaceApp"> <xs:annotation> <xs:documentation>DR-GW-Application. Use to get information about certain client application or to read out the complete client application list the DF-Client has rights for. Use only via SOAP.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="get" type="typeAppGet"/> <xs:element name="getList" type="typeAppGetList"/> <xs:element name="response" type="typeResponse"/> <xs:element name="getEvent" type="typeAppGetEvent"/> <xs:element name="getListEvent" type="typeAppGetListEvent"/> </xs:choice> </xs:complexType></pre>

Complex Type typeAppGet

Namespace	DR-GW
Annotations	
Diagram	
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeAppGet
Used by	Element interfaceApp/get
Model	requestId , app
Children	app, requestId
Source	<pre><xs:complexType name="typeAppGet"> <xs:annotation> <xs:documentation></xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="app" type="typeSubscriberAddress"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeAppGetList

Namespace	DR-GW
-----------	-------

Annotations	
Diagram	<pre> classDiagram typeRequest < -- typeAppGetList typeRequest "0..1" requestId typeRequest "0..1" orgblockId typeAppGetList "0..1" app </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeAppGetList
Used by	Element interfaceApp/getList
Model	requestId , orgblockId{0,1}
Children	orgblockId, requestId
Source	<pre> <xss:complexType name="typeAppGetList"> <xss:annotation> <xss:documentation></xss:documentation> </xss:annotation> <xss:complexContent> <xss:extension base="typeRequest"> <xss:sequence> <xss:element name="orgblockId" type="typeOrganisationBlockId" minOccurs="0"/> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

Complex Type typeAppGetEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeAppGetEvent typeEvent "0..1" requestId typeEvent "0..1" result typeAppGetEvent "0..1" app </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeAppGetEvent
Used by	Element interfaceApp/getEvent
Model	requestId{0,1} , result{0,1} , app
Children	app, requestId, result
Source	<pre> <xss:complexType name="typeAppGetEvent"> <xss:annotation> <xss:documentation></xss:documentation> </xss:annotation> <xss:complexContent> <xss:extension base="typeEvent"> <xss:sequence> <xss:element name="app" type="typeApplication" maxOccurs="1"/> </xss:sequence> </xss:extension> </xss:complexContent> </xss:complexType> </pre>

Complex Type typeApplication

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeApplication < -- typeEvent typeEvent < -- typeAppGetListEvent typeAppGetListEvent < --> addr typeAppGetListEvent < --> alias typeAppGetListEvent < --> orgblockId </pre>
Used by	Elements typeAppGetEvent/app, typeAppGetListEvent/app
Model	addr, alias, orgblockId
Children	addr, alias, orgblockId
Source	<pre> <xs:complexType name="typeApplication"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:sequence> <xs:element name="addr" type="typeSubscriberAddress"/> <xs:element name="alias" type="xs:normalizedString"/> <xs:element name="orgblockId" type="typeOrganisationBlockId"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeAppGetListEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent < -- typeAppGetListEvent typeAppGetListEvent < --> requestId typeAppGetListEvent < --> result typeAppGetListEvent < --> app typeAppGetListEvent < --> listEnd </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeAppGetListEvent
Used by	Element interfaceApp/getListEvent
Model	requestId{0,1}, result{0,1}, app*, listEnd{0,1}
Children	app, listEnd, requestId, result
Source	<pre> <xs:complexType name="typeAppGetListEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="app" type="typeApplication" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="listEnd" type="xs:boolean" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type interfaceSystem

Namespace	DR-GW
-----------	-------

Annotations	DR-GW-System. Use to get TETRA system events or DR-GW vendor-specific log and/or system events. Use only via SOAP.
Diagram	<pre> classDiagram class interfaceSystem { <<DR-GW-System. Use to get TETRA system events or DR-GW vendor-specific log and/or system events. Use only via SOAP.>> } class tetraStatesEvent { <<Indication of the subsystem state updates. Contains current states of all subsystem known by the TCS.>> } interfaceSystem < -- tetraStatesEvent tetraStatesEvent < -- response tetraStatesEvent < -- logEvent tetraStatesEvent < -- event </pre>
Used by	Element drgw/system
Model	response tetraStatesEvent logEvent event
Children	event, logEvent, response, tetraStatesEvent
Source	<pre> <xss:complexType name="interfaceSystem"> <xss:annotation> <xss:documentation>DR-GW-System. Use to get TETRA system events or DR-GW vendor-specific log and/or system events. Use only via SOAP.</xss:documentation> </xss:annotation> <xss:choice> <xss:element name="response" type="typeResponse"/> <xss:element name="tetraStatesEvent" type="typeSystemTetraStatesEvent"/> <xss:element name="logEvent" type="typeSystemLogEvent"/> <xss:element name="event" type="typeSystemEvent"/> </xss:choice> </xss:complexType> </pre>

Complex Type typeSystemTetraStatesEvent

Namespace	DR-GW
Annotations	Indication of the subsystem state updates. Contains current states of all subsystem known by the TCS.
Diagram	<pre> classDiagram class typeEvent { <<Indication of the subsystem state updates. Contains current states of all subsystem known by the TCS.>> } class typeSystemTetraStatesEvent { <<Indication of the subsystem state updates. Contains current states of all subsystem known by the TCS.>> } typeEvent < -- typeSystemTetraStatesEvent typeSystemTetraStatesEvent < -- requestId typeSystemTetraStatesEvent < -- result typeSystemTetraStatesEvent < -- tcsState typeSystemTetraStatesEvent < -- dxtState typeSystemTetraStatesEvent < -- cddconnectionState typeSystemTetraStatesEvent < -- cddserverState </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSystemTetraStatesEvent
Used by	Element interfaceSystem/tetraStatesEvent
Model	requestId{0,1} , result{0,1} , tcsState{0,1} , dxtState{0,1} , cddconnectionState{0,1} , cddserverState{0,1}
Children	cddconnectionState, cddserverState, dxtState, requestId, result, tcsState
Source	<pre> <xss:complexType name="typeSystemTetraStatesEvent"> <xss:annotation> <xss:documentation>Indication of the subsystem state updates. Contains current states of all subsystem known by the TCS.</xss:documentation> </xss:annotation> <xss:complexContent> <xss:extension base="typeEvent"> <xss:sequence> </pre>

```

<xs:element name="tcsState" type="typeSystemElementState" minOccurs="0" />
<xs:element name="dxtState" type="typeSystemElementState" minOccurs="0" />
<xs:element name="cddconnectionState" type="typeSystemElementState" minOccurs="0" />
<xs:element name="cddserverState" type="typeSystemElementState" minOccurs="0" />
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

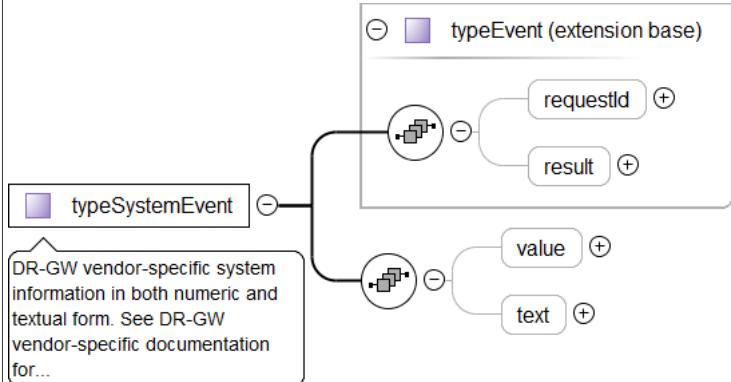
```

Complex Type typeSystemLogEvent

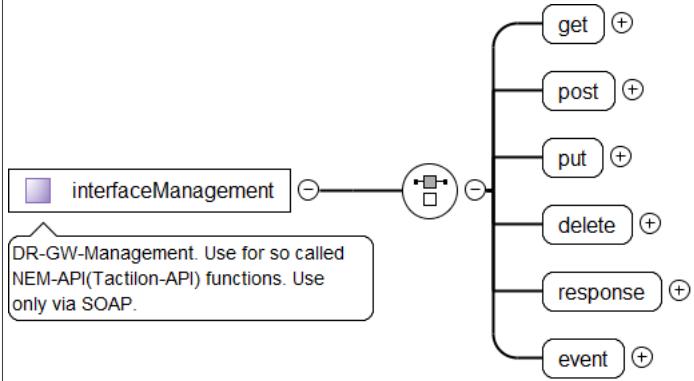
Namespace	DR-GW
Annotations	DR-GW vendor-specific logging information (errors, notices) in both numeric and textual form. See DR-GW vendor-specific documentation for the details.
Diagram	<pre> classDiagram typeEvent("typeEvent (extension base)") { requestId result value text } typeSystemLogEvent("typeSystemLogEvent") { <<DR-GW vendor-specific logging information (errors, notices) in both numeric and textual form. See DR-GW vendor-specific...>> } typeEvent < -- typeSystemLogEvent typeSystemLogEvent --> requestId typeSystemLogEvent --> result typeSystemLogEvent --> value typeSystemLogEvent --> text </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSystemLogEvent
Used by	Element interfaceSystem/logEvent
Model	requestId{0,1} , result{0,1} , value{0,1} , text{0,1}
Children	requestId, result, text, value
Source	<pre> <xs:complexType name="typeSystemLogEvent"> <xs:annotation> <xs:documentation>DR-GW vendor-specific logging information (errors, notices) in both numeric and textual form. See DR-GW vendor-specific documentation for the details.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="value" type="xs:hexBinary" minOccurs="0" /> <xs:element name="text" type="xs:normalizedString" minOccurs="0" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeSystemEvent

Namespace	DR-GW
Annotations	DR-GW vendor-specific system information in both numeric and textual form. See DR-GW vendor-specific documentation for the details.

Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSystemEvent
Used by	Element interfaceSystem/event
Model	requestId{0,1} , result{0,1} , value{0,1} , text{0,1}
Children	requestId, result, text, value
Source	<pre><xs:complexType name="typeSystemEvent"> <xs:annotation> <xs:documentation>DR-GW vendor-specific system information in both numeric and textual form. See DR-GW vendor-specific documentation for the details.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="value" type="xs:hexBinary" minOccurs="0"/> <xs:element name="text" type="xs:normalizedString" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type interfaceManagement

Namespace	DR-GW
Annotations	DR-GW-Management. Use for so called NEM-API(Tactilon-API) functions. Use only via SOAP.
Diagram	
Used by	Element drgw/management
Model	get post put delete response event
Children	delete, event, get, post, put, response
Source	<pre><xs:complexType name="interfaceManagement"> <xs:annotation> <xs:documentation>DR-GW-Management. Use for so called NEM-API(Tactilon-API) functions. Use only via SOAP.</xs:documentation> </xs:annotation> <xs:choice> <xs:element name="get" type="typeManagementRequest"/> <xs:element name="post" type="typeManagementRequest"/></pre>

```

<xs:element name="put" type="typeManagementRequest" />
<xs:element name="delete" type="typeManagementRequest" />
<xs:element name="response" type="typeResponse" />
<xs:element name="event" type="typeManagementEvent" />
</xs:choice>
</xs:complexType>

```

Complex Type typeManagementRequest

Namespace	DR-GW
Annotations	Generic management request used for all GET, POST, PUT, DELETE operations.
Diagram	<p>The diagram illustrates the structure of the typeManagementRequest complex type. It is based on the typeRequest extension base. The attributes defined are:</p> <ul style="list-style-type: none"> requestId requestUri body authorization host <p>Annotations provide additional context for each attribute:</p> <ul style="list-style-type: none"> requestId: "Some management requests, typically PUT and POST require the presence of the body. See Tactilon-API specification for..." requestUri: "Optional, Df-Client can provide his own authorization credentials to access Tactilon. If not provided, the DR-GW..." body: "Optional, Df-Client can provide different host/ip address of the Tactilon. If not provided, the DR-GW shall use its..." authorization: "See Tactilon-API specification for details on how to construct the body." host: "See Tactilon-API specification for details on how to construct the body."
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeManagementRequest
Used by	Elements interfaceManagement/delete, interfaceManagement/get, interfaceManagement/post, interfaceManagement/put
Model	requestId, requestUri, body{0,1}, authorization{0,1}, host{0,1}
Children	authorization, body, host, requestId, requestUri
Source	<pre> <xs:complexType name="typeManagementRequest"> <xs:annotation> <xs:documentation>Generic management request used for all GET, POST, PUT, DELETE operations. <!-- https://<Tactilon_host>/<requestUri> --> </xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="requestUri" type="xs:string" /> <xs:element name="body" type="xs:string" minOccurs="0" /> <xs:annotation> <xs:documentation>Some management requests, typically PUT and POST require the presence of the body. See Tactilon-API specification for details on how to construct the body.</xs:documentation> </xs:annotation> </xs:sequence> <xs:element name="authorization" type="typeManagementAuthorization" minOccurs="0" /> <xs:annotation> <xs:documentation>Optional, Df-Client can provide his own authorization credentials to access Tactilon. If not provided, the DR-GW shall use its own, globally configured.</xs:documentation> </xs:annotation> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

```

<xs:element name="host" type="xs:string" minOccurs="0">
    <xs:annotation>
        <xs:documentation> Optionally, Df-Client can provide different host/ip address of the Tactilon. If not provided, the DR-GW shall use its own, globally configured.</xs:documentation>
    </xs:annotation>
</xs:element>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeManagementAuthorization

Namespace	DR-GW
Annotations	
Diagram	
Used by	Element typeManagementRequest/authorization
Model	username , password
Children	password, username
Source	<pre> <xs:complexType name="typeManagementAuthorization"> <xs:annotation> <xs:documentation> </xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="username" type="xs:string"/> <xs:element name="password" type="xs:string"/> </xs:sequence> </xs:complexType> </pre>

Complex Type typeManagementEvent

Namespace	DR-GW
Annotations	Generic management event, is fired from DR-GW as a final response to previous get, post, put or delete request from Df-Client and contains the actual response from Tactilon.
Diagram	
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeManagementEvent
Used by	Element interfaceManagement/event
Model	requestId{0,1} , result{0,1} , body{0,1}
Children	body, requestId, result
Source	<pre> <xs:complexType name="typeManagementEvent"> <xs:annotation> <xs:documentation> Generic management event, is fired from DR-GW as a final response to previous get, post, put or delete request from Df-Client and contains the actual response from Tactilon.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> </pre>

```

<xs:sequence>
  <xs:element name="body" type="xs:string" minOccurs="0" />
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Complex Type typeCallRequest

Namespace	DR-GW
Annotations	The method is used to accomplish all call related operations.
Diagram	<pre> classDiagram typeRequest "typeRequest (extension base)" { requestId action attributes callingParty calledParty workstationId } typeCallRequest "typeCallRequest" { <<The method is used to accomplish all call related operations.>> } typeCallRequest --> typeRequest typeCallRequest <<The method is used to accomplish all call related operations.>> </pre>
Type	extension of typeRequest
Type hierarchy	<ul style="list-style-type: none"> • typeRequest • typeCallRequest
Model	requestId , action , attributes{0,1} , callingParty{0,1} , calledParty{0,1} , workstationId{0,1}
Children	action, attributes, calledParty, callingParty, requestId, workstationId
Source	<pre> <xs:complexType name="typeCallRequest"> <xs:annotation> <xs:documentation>The method is used to accomplish all call related operations.</xs:documentation> </xs:annotation> <xs:complexContent> <xs:extension base="typeRequest"> <xs:sequence> <xs:element name="action" type="typeActionRequest" /> <xs:element name="attributes" type="typeCallAttributes" minOccurs="0" /> <xs:element name="callingParty" type="typeAddress" minOccurs="0" /> <xs:element name="calledParty" type="typeAddress" minOccurs="0" /> <xs:element name="workstationId" type="typeWorkstationId" minOccurs="0" /> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

Complex Type typeSessionLogoutEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> classDiagram typeEvent "typeEvent (extension base)" { requestId result reason } typeSessionLogoutEvent "typeSessionLogoutEvent" { <<The method is used to accomplish all session logout related operations.>> } typeSessionLogoutEvent --> typeEvent typeSessionLogoutEvent <<The method is used to accomplish all session logout related operations.>> </pre>

Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeSessionLogoutEvent
Model	requestId{0,1} , result{0,1} , reason{0,1}
Children	reason, requestId, result
Source	<pre><xs:complexType name="typeSessionLogoutEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="reason" type="xs:unsignedLong" minOccurs="0"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Complex Type typeSdsValidity

Namespace	DR-GW
Annotations	Validity of the SDS in case store and forward center is used. The unit is seconds. Infinte validity is represented by 0xFFFFFFFF
Diagram	<pre> graph LR A["typeSdsValidity"] --> B["value"] B --> C["value"] C --> D["value"] </pre> <p>Validity of the SDS in case store and forward center is used. The unit is seconds. Infinte validity is represented by...</p>
Model	value
Children	value
Source	<pre><xs:complexType name="typeSdsValidity"> <xs:annotation> <xs:documentation>Validity of the SDS in case store and forward center is used. The unit is seconds. Infinte validity is represented by 0xFFFFFFFF</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="value" type="xs:unsignedLong"/> </xs:sequence> </xs:complexType></pre>

Complex Type typeGroupGetCombinationsEvent

Namespace	DR-GW
Annotations	
Diagram	<pre> graph LR A["typeEvent (extension base)"] B["typeGroupGetCombinationsEvent"] A --- C["requestId"] A --- D["result"] B --- E["group"] B --- F["baseGroup"] B --- G["constitGroup"] </pre>
Type	extension of typeEvent
Type hierarchy	<ul style="list-style-type: none"> • typeEvent • typeGroupGetCombinationsEvent

Model	requestId{0,1} , result{0,1} , group , baseGroup , constitGroup{1,7}
Children	baseGroup, constitGroup, group, requestId, result
Source	<pre><xs:complexType name="typeGroupGetCombinationsEvent"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:complexContent> <xs:extension base="typeEvent"> <xs:sequence> <xs:element name="group" type="typeSubscriberAddress"/> <xs:sequence minOccurs="0"> <xs:element name="baseGroup" type="typeSubscriberAddress"/> <xs:element name="constitGroup" type="typeSubscriberAddress" maxOccurs="7"/> </xs:sequence> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType></pre>

Simple Type(s)

Simple Type typeSelectionLevel

Namespace	DR-GW															
Annotations	Defines how the target is monitored.															
Diagram																
Type	restriction of xs:normalizedString															
Facets	<table> <tr> <td>enumeration</td> <td>no</td> <td>No selection. Used to remove selection.</td> </tr> <tr> <td>enumeration</td> <td>event</td> <td>Event monitoring.</td> </tr> <tr> <td>enumeration</td> <td>audio</td> <td>Audio monitoring.</td> </tr> <tr> <td>enumeration</td> <td>use</td> <td>Selection level use.</td> </tr> <tr> <td>enumeration</td> <td>a_use</td> <td>Selection level active use.</td> </tr> </table>	enumeration	no	No selection. Used to remove selection.	enumeration	event	Event monitoring.	enumeration	audio	Audio monitoring.	enumeration	use	Selection level use.	enumeration	a_use	Selection level active use.
enumeration	no	No selection. Used to remove selection.														
enumeration	event	Event monitoring.														
enumeration	audio	Audio monitoring.														
enumeration	use	Selection level use.														
enumeration	a_use	Selection level active use.														
Used by	Element typeSelection/level															
Source	<pre><xs:simpleType name="typeSelectionLevel"> <xs:annotation> <xs:documentation>Defines how the target is monitored.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="no"> <xs:annotation> <xs:documentation>No selection. Used to remove selection.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="event"> <xs:annotation> <xs:documentation>Event monitoring.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="audio"> <xs:annotation> <xs:documentation>Audio monitoring.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="use"> <xs:annotation> <xs:documentation>Selection level use.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="a_use"> <xs:annotation> <xs:documentation>Selection level active use.</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType></pre>															

Simple Type typeDialString

Namespace	DR-GW
Annotations	Allowed characters are digits 0 - 9, *, #, A, B, C and D. Maximum length is 24 characters.
Diagram	<pre> graph LR typeDialString["typeDialString"] -- ⊂ --> xsNormalizedString["xs:normalizedString"] </pre> <p>Allowed characters are digits 0 - 9, *, #, A, B, C and D. Maximum length is 24 characters.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>
Type	restriction of xs:normalizedString
Facets	maxLength 24
Used by	Elements typeAddress/msisdn, typeExternal/number
Source	<pre> <xs:simpleType name="typeDialString"> <xs:annotation> <xs:documentation>Allowed characters are digits 0 - 9, *, #, A, B, C and D. Maximum length is 24 characters.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:maxLength value="24"/> </xs:restriction> </xs:simpleType> </pre>

Simple Type typeOpta

Namespace	DR-GW
Annotations	OPTA string. Maximum length is 24 characters.
Diagram	<pre> graph LR typeOpta["typeOpta"] -- ⊂ --> xsNormalizedString["xs:normalizedString"] </pre> <p>OPTA string. Maximum length is 24 characters.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>
Type	restriction of xs:normalizedString
Facets	maxLength 24
Used by	Elements typeAddress/opta, typeLastKnownOPTA/opta, typeRadioChangeOpta/opta, typeRadioChangeOptaEvent/opta
Source	<pre> <xs:simpleType name="typeOpta"> <xs:annotation> <xs:documentation>OPTA string. Maximum length is 24 characters.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:maxLength value="24"/> </xs:restriction> </xs:simpleType> </pre>

Simple Type typeResponseCode

Namespace	DR-GW										
Diagram	<pre> graph LR typeResponseCode["typeResponseCode"] -- ⊂ --> xsNormalizedString["xs:normalizedString"] </pre> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>										
Type	restriction of xs:normalizedString										
Facets	<table border="1"> <tr> <td>enumeration</td> <td>success</td> </tr> <tr> <td>enumeration</td> <td>final_response_pending</td> </tr> <tr> <td>enumeration</td> <td>error</td> </tr> <tr> <td>enumeration</td> <td>not_authorized_error</td> </tr> <tr> <td>enumeration</td> <td>temporary_failure</td> </tr> </table>	enumeration	success	enumeration	final_response_pending	enumeration	error	enumeration	not_authorized_error	enumeration	temporary_failure
enumeration	success										
enumeration	final_response_pending										
enumeration	error										
enumeration	not_authorized_error										
enumeration	temporary_failure										

	enumeration	subscription_failed
Used by	Element	typeResult/responseCode
Source	<pre><xs:simpleType name="typeResponseCode"> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="success"/> <xs:enumeration value="final_response_pending"/> <xs:enumeration value="error"/> <xs:enumeration value="not_authorized_error"/> <xs:enumeration value="temporary_failure"/> <xs:enumeration value="subscription_failed"/> </xs:restriction> </xs:simpleType></pre>	

Simple Type typeSourceSystem

Namespace	DR-GW								
Diagram	<p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>								
Type	restriction of xs:normalizedString								
Facets	<table border="1"> <tr> <td>enumeration</td> <td>DR-GW</td> </tr> <tr> <td>enumeration</td> <td>TCS-API</td> </tr> <tr> <td>enumeration</td> <td>TETRA</td> </tr> <tr> <td>enumeration</td> <td>NEM-API</td> </tr> </table>	enumeration	DR-GW	enumeration	TCS-API	enumeration	TETRA	enumeration	NEM-API
enumeration	DR-GW								
enumeration	TCS-API								
enumeration	TETRA								
enumeration	NEM-API								
Used by	Element typeResult/sourceSystem								
Source	<pre><xs:simpleType name="typeSourceSystem"> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="DR-GW"/> <xs:enumeration value="TCS-API"/> <xs:enumeration value="TETRA"/> <xs:enumeration value="NEM-API"/> </xs:restriction> </xs:simpleType></pre>								

Simple Type typeTetraCallId

Namespace	DR-GW
Annotations	TETRA callId, generated by the TCS. In the TCS-API it is defined as long, but Airbus doesn't have it under control if it can only be unsigned, therefore modified to string in DR-GW.
Diagram	<p>TETRA callId, generated by the TCS. In the TCS-API it is defined as long, but Airbus doesn't have it under control if...</p> <p>Built-in primitive type. The string datatype represents character strings in XML.</p>
Type	xs:string
Used by	Elements typeCallEvent/tetraCallId, typeCallPTTEvent/tetraCallId, typeCallUnitInEmergencyEvent/tetraCallId
Source	<pre><xs:simpleType name="typeTetraCallId"> <xs:annotation> <xs:documentation>TETRA callId, generated by the TCS. In the TCS-API it is defined as long, but Airbus doesn't have it under control if it can only be unsigned, therefore modified to string in DR-GW.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"/> </xs:simpleType></pre>

Simple Type typeActionEvent

Namespace	DR-GW
-----------	-------

Annotations	All possible call actions.	
Diagram		
Type	restriction of xs:normalizedString	
Facets	enumeration	incoming This event fired when there is an incoming call. This is the first indication of a new incoming call.
	enumeration	connected This event is used to inform that call has been connected and call setup is finished.
	enumeration	held This event is used to inform TCS Client that individual call was put to hold.
	enumeration	resumed This event is used to inform that individual call has been taken from hold.
	enumeration	disconnected This event is used to inform that the call was disconnected.
	enumeration	transferred This event is a response to transfer method call and indicates the result of the request.
Used by	Element	typeCallEvent/action
Source	<pre> <xs:simpleType name="typeActionEvent"> <xs:annotation> <xs:documentation>All possible call actions.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="incoming"> <xs:annotation> <xs:documentation>This event fired when there is an incoming call. This is the first indication of a new incoming call.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="connected"> <xs:annotation> <xs:documentation>This event is used to inform that call has been connected and call setup is finished.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="held"> <xs:annotation> <xs:documentation>This event is used to inform TCS Client that individual call was put to hold.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="resumed"> <xs:annotation> <xs:documentation>This event is used to inform that individual call has been taken from hold.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="disconnected"> <xs:annotation> <xs:documentation>This event is used to inform that the call was disconnected.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="transferred"> <xs:annotation> <xs:documentation>This event is a response to transfer method call and indicates the result of the request.</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType> </pre>	

Simple Type typeCallMode

Namespace	DR-GW
-----------	-------

Annotations	Call mode attribute. Choices are simplex or duplex.				
Diagram	<p>Call mode attribute. Choices are simplex or duplex.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>				
Type	restriction of xs:normalizedString				
Facets	<table> <tr> <td>enumeration</td> <td>simplex</td> </tr> <tr> <td>enumeration</td> <td>duplex</td> </tr> </table>	enumeration	simplex	enumeration	duplex
enumeration	simplex				
enumeration	duplex				
Used by	Element typeCallAttributes/mode				
Source	<pre><xs:simpleType name="typeCallMode"> <xs:annotation> <xs:documentation>Call mode attribute. Choices are simplex or duplex.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="simplex"/> <xs:enumeration value="duplex"/> </xs:restriction> </xs:simpleType></pre>				

Simple Type typeCallType

Namespace	DR-GW						
Annotations	Call type attribute. Choices are Point2Point, Point2MultiPoint or Broadcast.						
Diagram	<p>Call type attribute. Choices are Point2Point, Point2MultiPoint or Broadcast.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>						
Type	restriction of xs:normalizedString						
Facets	<table> <tr> <td>enumeration</td> <td>p2p</td> </tr> <tr> <td>enumeration</td> <td>p2mp</td> </tr> <tr> <td>enumeration</td> <td>bcast</td> </tr> </table>	enumeration	p2p	enumeration	p2mp	enumeration	bcast
enumeration	p2p						
enumeration	p2mp						
enumeration	bcast						
Used by	Elements typeCallAttributes/commtyp, typeRadioTrackingData/callType						
Source	<pre><xs:simpleType name="typeCallType"> <xs:annotation> <xs:documentation>Call type attribute. Choices are Point2Point, Point2MultiPoint or Broadcast.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="p2p"/> <xs:enumeration value="p2mp"/> <xs:enumeration value="bcast"/> </xs:restriction> </xs:simpleType></pre>						

Simple Type typeTxDemandPriority

Namespace	DR-GW				
Annotations	Defines priority of speech item request: normal, pre-emptive, or emergency.				
Diagram	<p>Defines priority of speech item request: normal, pre-emptive, or emergency.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>				
Type	restriction of xs:normalizedString				
Facets	<table> <tr> <td>enumeration</td> <td>normal</td> </tr> <tr> <td>enumeration</td> <td>preemptive</td> </tr> </table>	enumeration	normal	enumeration	preemptive
enumeration	normal				
enumeration	preemptive				

	enumeration	emergency
Used by	Element	typeCallAttributes/demandPriority
Source	<pre><xs:simpleType name="typeTxDemandPriority"> <xs:annotation> <xs:documentation>Defines priority of speech item request: normal, pre-emptive, or emergency.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="normal"/> <xs:enumeration value="preemptive"/> <xs:enumeration value="emergency"/> </xs:restriction> </xs:simpleType></pre>	

Simple Type typeActionPTTRequest

Namespace	DR-GW							
Annotations	All possible PTT requests.							
Diagram	<pre> classDiagram typeActionPTTRequest "1" -- "0..1" xs:normalizedString typeActionPTTRequest "1..*" xs:normalizedString "0..1" </pre> <p>All possible PTT requests.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>							
Type	restriction of xs:normalizedString							
Facets	<table> <tr> <td>enumeration</td> <td>demandtx</td> <td>This method can be used to request a speech item for a connected call.</td> </tr> <tr> <td>enumeration</td> <td>ceasetx</td> <td>This method is used to inform the system that the speech item is not needed any more.</td> </tr> </table>		enumeration	demandtx	This method can be used to request a speech item for a connected call.	enumeration	ceasetx	This method is used to inform the system that the speech item is not needed any more.
enumeration	demandtx	This method can be used to request a speech item for a connected call.						
enumeration	ceasetx	This method is used to inform the system that the speech item is not needed any more.						
Used by	Element typeCallPTTRequest/action							
Source	<pre><xs:simpleType name="typeActionPTTRequest"> <xs:annotation> <xs:documentation>All possible PTT requests.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="demandtx"> <xs:annotation> <xs:documentation>This method can be used to request a speech item for a connected call.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="ceasetx"> <xs:annotation> <xs:documentation>This method is used to inform the system that the speech item is not needed any more.</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType></pre>							

Simple Type typeWorkstationId

Namespace	DR-GW	
Annotations	Optional parameter is used to support the "neighbours" feature.	
Diagram	<pre> classDiagram typeWorkstationId "1" -- "0..1" xs:normalizedString typeWorkstationId "1..*" xs:normalizedString "0..1" </pre> <p>Optional parameter is used to support the "neighbours" feature.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>	
Type	xs:normalizedString	
Used by	Elements typeCallPTTRequest/workstationId, typeCallRequest/workstationId	
Source	<pre><xs:simpleType name="typeWorkstationId"> <xs:annotation></pre>	

```

<xs:documentation>Optional parameter is used to support the "neighbours" feature.</
xs:documentation>
</xs:annotation>
<xs:restriction base="xs:normalizedString" />
</xs:simpleType>

```

Simple Type typeKeyExchangeAction

Namespace	DR-GW				
Annotations	Action type for key exchange request.				
Diagram	<pre> classDiagram typeKeyExchangeAction < -- xs:normalizedString </pre> <p>The diagram shows a UML class named 'typeKeyExchangeAction' with a generalization arrow pointing to the 'xs:normalizedString' class. A callout box below 'typeKeyExchangeAction' states 'Action type for key exchange request.' A callout box below 'xs:normalizedString' states 'Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...'.</p>				
Type	restriction of xs:normalizedString				
Facets	<table> <tr> <td>enumeration</td> <td>start</td> </tr> <tr> <td>enumeration</td> <td>stop</td> </tr> </table>	enumeration	start	enumeration	stop
enumeration	start				
enumeration	stop				
Used by	Element typeCallKeyExchange/action				
Source	<pre> <xs:simpleType name="typeKeyExchangeAction"> <xs:annotation> <xs:documentation>Action type for key exchange request.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString" /> <xs:enumeration value="start"/> <xs:enumeration value="stop"/> </xs:restriction> </xs:simpleType> </pre>				

Simple Type typeTxGrant

Namespace	DR-GW								
Annotations	Defines to whom speech item was granted.								
Diagram	<pre> classDiagram typeTxGrant < -- xs:normalizedString </pre> <p>The diagram shows a UML class named 'typeTxGrant' with a generalization arrow pointing to the 'xs:normalizedString' class. A callout box below 'typeTxGrant' states 'Defines to whom speech item was granted.' A callout box below 'xs:normalizedString' states 'Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...'.</p>								
Type	restriction of xs:normalizedString								
Facets	<table> <tr> <td>enumeration</td> <td>granted</td> </tr> <tr> <td>enumeration</td> <td>notGranted</td> </tr> <tr> <td>enumeration</td> <td>queued</td> </tr> <tr> <td>enumeration</td> <td>granted2another</td> </tr> </table>	enumeration	granted	enumeration	notGranted	enumeration	queued	enumeration	granted2another
enumeration	granted								
enumeration	notGranted								
enumeration	queued								
enumeration	granted2another								
Used by	Element typeTxGranted/txGrant								
Source	<pre> <xs:simpleType name="typeTxGrant"> <xs:annotation> <xs:documentation>Defines to whom speech item was granted.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString" /> <xs:enumeration value="granted"/> <xs:enumeration value="notGranted"/> <xs:enumeration value="queued"/> <xs:enumeration value="granted2another"/> </xs:restriction> </xs:simpleType> </pre>								

Simple Type typeTxPriority

Namespace	DR-GW
Annotations	Defines the priority of the transmission.

Diagram	<pre> graph LR typeTxPriority["typeTxPriority"] --> xsNormalizedString["xs:normalizedString"] typeTxPriority --- note1[Defines the priority of the transmission.] xsNormalizedString --- note2[Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...] </pre>				
Type	restriction of xs:normalizedString				
Facets	<table> <tr> <td>enumeration</td> <td>normal</td> </tr> <tr> <td>enumeration</td> <td>emergency</td> </tr> </table>	enumeration	normal	enumeration	emergency
enumeration	normal				
enumeration	emergency				
Used by	Element typeTxGranted/txPriority				
Source	<pre> <xs:simpleType name="typeTxPriority"> <xs:annotation> <xs:documentation>Defines the priority of the transmission.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="normal"/> <xs:enumeration value="emergency"/> </xs:restriction> </xs:simpleType> </pre>				

Simple Type typeUnitInEmergencyType

Namespace	DR-GW										
Annotations	Defines type of the subscriber. Refer to type tcsCallSubscriberType_t of the TCS-API.										
Diagram	<pre> graph LR typeUnitInEmergencyType["typeUnitInEmergencyType"] --> xsNormalizedString["xs:normalizedString"] typeUnitInEmergencyType --- note1[Defines type of the subscriber. Refer to type tcsCallSubscriberType_t of the TCS-API.] xsNormalizedString --- note2[Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...] </pre>										
Type	restriction of xs:normalizedString										
Facets	<table> <tr> <td>enumeration</td> <td>dummy</td> </tr> <tr> <td>enumeration</td> <td>ms</td> </tr> <tr> <td>enumeration</td> <td>g4wif</td> </tr> <tr> <td>enumeration</td> <td>external</td> </tr> <tr> <td>enumeration</td> <td>ws</td> </tr> </table>	enumeration	dummy	enumeration	ms	enumeration	g4wif	enumeration	external	enumeration	ws
enumeration	dummy										
enumeration	ms										
enumeration	g4wif										
enumeration	external										
enumeration	ws										
Used by	Element typeCallUnitInEmergencyEvent/unitInEmgType										
Source	<pre> <xs:simpleType name="typeUnitInEmergencyType"> <xs:annotation> <xs:documentation>Defines type of the subscriber. Refer to type tcsCallSubscriberType_t of the TCS-API.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="dummy"/> <xs:enumeration value="ms"/> <xs:enumeration value="g4wif"/> <xs:enumeration value="external"/> <xs:enumeration value="ws"/> </xs:restriction> </xs:simpleType> </pre>										

Simple Type typeEmergencyInfo

Namespace	DR-GW
Annotations	Defines action taken by user in emergency.
Diagram	<pre> graph LR typeEmergencyInfo["typeEmergencyInfo"] --> xsNormalizedString["xs:normalizedString"] typeEmergencyInfo --- note1[Defines action taken by user in emergency.] xsNormalizedString --- note2[Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...] </pre>
Type	restriction of xs:normalizedString

Facets	enumeration	addTx	
	enumeration	add	
	enumeration	ceased	
	enumeration	demandTx	
	enumeration	removed	
	enumeration	emergencyCallDisconnected	
Used by	Element	typeCallUnitInEmergencyEvent/emgInfo	
Source	<pre><xs:simpleType name="typeEmergencyInfo"> <xs:annotation> <xs:documentation>Defines action taken by user in emergency.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="addTx"/> <xs:enumeration value="add"/> <xs:enumeration value="ceased"/> <xs:enumeration value="demandTx"/> <xs:enumeration value="removed"/> <xs:enumeration value="emergencyCallDisconnected"/> </xs:restriction> </xs:simpleType></pre>		

Simple Type typeKeyExchangeState

Namespace	DR-GW											
Annotations	Represents current key state.											
Diagram	<pre> classDiagram typeKeyExchangeState "1" -- "0..1" xs:normalizedString typeKeyExchangeState "1..1" -- "1..1" "Represents current key state." xs:normalizedString "1..1" -- "1..1" "Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of..." </pre>											
Type	restriction of xs:normalizedString											
Facets	<table> <tr> <td>enumeration</td> <td>keyValid</td> <td>current key is valid, no user action required.</td> </tr> <tr> <td>enumeration</td> <td>keyInvalid</td> <td>Key invalid, user must request key exchange.</td> </tr> <tr> <td>enumeration</td> <td>keyExchangeInProgress</td> <td>Key exchange in progress, user may abort exchange or wait until it gets finished.</td> </tr> </table>			enumeration	keyValid	current key is valid, no user action required.	enumeration	keyInvalid	Key invalid, user must request key exchange.	enumeration	keyExchangeInProgress	Key exchange in progress, user may abort exchange or wait until it gets finished.
enumeration	keyValid	current key is valid, no user action required.										
enumeration	keyInvalid	Key invalid, user must request key exchange.										
enumeration	keyExchangeInProgress	Key exchange in progress, user may abort exchange or wait until it gets finished.										
Used by	Element typeCallKeyExchangeEvent/state											
Source	<pre><xs:simpleType name="typeKeyExchangeState"> <xs:annotation> <xs:documentation>Represents current key state.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="keyValid"> <xs:annotation> <xs:documentation>current key is valid, no user action required.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="keyInvalid"> <xs:annotation> <xs:documentation>Key invalid, user must request key exchange.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="keyExchangeInProgress"> <xs:annotation> <xs:documentation>Key exchange in progress, user may abort exchange or wait until it gets finished.</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType></pre>											

Simple Type typeKeyExchangeCode

Namespace	DR-GW	
Annotations	See "Table 5.3: Status words of the commands" of the E-to-E Encryption	

	SIM-ME Interface (Version 4.0.5) for all possible code values.
Diagram	<p>See "Table 5.3: Status words of the commands" of the E-to-E Encryption SIM-ME Interface (Version 4.0.5) for all...</p> <p>Built-in primitive type. The hexBinary datatype represents arbitrary hex-encoded binary data.</p>
Type	restriction of xs:hexBinary
Facets	length 2
Used by	Element typeCallKeyExchangeEvent/code
Source	<pre><xs:simpleType name="typeKeyExchangeCode"> <xs:annotation> <xs:documentation>See "Table 5.3: Status words of the commands" of the E-to-E Encryption SIM-ME Interface (Version 4.0.5) for all possible code values.</xs:documentation> </xs:annotation> <xs:restriction base="xs:hexBinary"> <xs:length value="2"/> </xs:restriction> </xs:simpleType></pre>

Simple Type typeKeyExchangeTextPriority

Namespace	DR-GW				
Annotations	Defines the priority of the KeyExchange information.				
Diagram	<p>Defines the priority of the KeyExchange information.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>				
Type	restriction of xs:normalizedString				
Facets	<table> <tr> <td>enumeration</td> <td>normal</td> </tr> <tr> <td>enumeration</td> <td>high</td> </tr> </table>	enumeration	normal	enumeration	high
enumeration	normal				
enumeration	high				
Used by	Element typeCallKeyExchangeEvent/priority				
Source	<pre><xs:simpleType name="typeKeyExchangeTextPriority"> <xs:annotation> <xs:documentation>Defines the priority of the KeyExchange information.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="normal"/> <xs:enumeration value="high"/> </xs:restriction> </xs:simpleType></pre>				

Simple Type typeKeyExchangeText

Namespace	DR-GW
Annotations	The textual information supplied by the BOS-simcard and sent from the DF-Gateway to the DF-client.
Diagram	<p>The textual information supplied by the BOS-simcard and sent from the DF-Gateway to the DF-client.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>
Type	restriction of xs:normalizedString
Facets	maxLength 100
Used by	Element typeCallKeyExchangeEvent/text
Source	<pre><xs:simpleType name="typeKeyExchangeText"> <xs:annotation> <xs:documentation>The textual information supplied by the BOS-simcard and sent from the DF-Gateway to the DF-client.</xs:documentation> </xs:annotation></pre>

```

<xs:restriction base="xs:normalizedString">
  <xsmaxLength value="100"/>
</xs:restriction>
</xs:simpleType>

```

Simple Type typeSuperviseTimeout

Namespace	DR-GW						
Annotations	Accepted supervise timeout values.						
Diagram	<p>The diagram shows two rounded rectangles representing UML classes. The left one is labeled "typeSuperviseTimeout" and the right one is labeled "xs:unsignedByte". An association line connects them with a hollow diamond symbol at the "typeSuperviseTimeout" end. Below the diagram are two callout boxes. The left one, pointing to "typeSuperviseTimeout", contains the text "Accepted supervise timeout values.". The right one, pointing to "xs:unsignedByte", contains the text "Built-in derived type. The unsignedByte datatype is derived from unsignedShort by setting the value of maxInclusive to...".</p>						
Type	restriction of xs:unsignedByte						
Facets	<table> <tr> <td>enumeration</td> <td>20</td> </tr> <tr> <td>enumeration</td> <td>30</td> </tr> <tr> <td>enumeration</td> <td>60</td> </tr> </table>	enumeration	20	enumeration	30	enumeration	60
enumeration	20						
enumeration	30						
enumeration	60						
Used by	Element typeSessionLogin/supervise						
Source	<pre> <xs:simpleType name="typeSuperviseTimeout"> <xs:annotation> <xs:documentation>Accepted supervise timeout values.</xs:documentation> </xs:annotation> <xs:restriction base="xs:unsignedByte"> <xs:enumeration value="20"/> <xs:enumeration value="30"/> <xs:enumeration value="60"/> </xs:restriction> </xs:simpleType> </pre>						

Simple Type typeSdsType

Namespace	DR-GW																		
Annotations																			
Diagram	<p>The diagram shows two rounded rectangles representing UML classes. The left one is labeled "typeSdsType" and the right one is labeled "xs:byte". An association line connects them with a hollow diamond symbol at the "typeSdsType" end. Below the diagram is a callout box pointing to "typeSdsType" containing the text "Built-in derived type. The byte datatype is derived from short by setting the value of maxInclusive to be 127 and...".</p>																		
Type	restriction of xs:byte																		
Facets	<table> <tr> <td>enumeration</td> <td>0</td> <td>SDS1.</td> </tr> <tr> <td>enumeration</td> <td>1</td> <td>SDS2.</td> </tr> <tr> <td>enumeration</td> <td>2</td> <td>SDS3.</td> </tr> <tr> <td>enumeration</td> <td>3</td> <td>SDS4.</td> </tr> <tr> <td>enumeration</td> <td>4</td> <td>SDS-TL.</td> </tr> <tr> <td>enumeration</td> <td>5</td> <td>Status.</td> </tr> </table>	enumeration	0	SDS1.	enumeration	1	SDS2.	enumeration	2	SDS3.	enumeration	3	SDS4.	enumeration	4	SDS-TL.	enumeration	5	Status.
enumeration	0	SDS1.																	
enumeration	1	SDS2.																	
enumeration	2	SDS3.																	
enumeration	3	SDS4.																	
enumeration	4	SDS-TL.																	
enumeration	5	Status.																	
Used by	Element typeSds/sdsType																		
Source	<pre> <xs:simpleType name="typeSdsType"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:restriction base="xs:byte"> <xs:enumeration value="0"> <xs:annotation> <xs:documentation>SDS1.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="1"> <xs:annotation> <xs:documentation>SDS2.</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType> </pre>																		

```

</xs:enumeration>
<xs:enumeration value="2">
  <xs:annotation>
    <xs:documentation>SDS3.</xs:documentation>
  </xs:annotation>
</xs:enumeration>
<xs:enumeration value="3">
  <xs:annotation>
    <xs:documentation>SDS4.</xs:documentation>
  </xs:annotation>
</xs:enumeration>
<xs:enumeration value="4">
  <xs:annotation>
    <xs:documentation>SDS-TL.</xs:documentation>
  </xs:annotation>
</xs:enumeration>
<xs:enumeration value="5">
  <xs:annotation>
    <xs:documentation>Status.</xs:documentation>
  </xs:annotation>
</xs:enumeration>
</xs:restriction>
</xs:simpleType>

```

Simple Type typeReport

Namespace	DR-GW								
Annotations									
Diagram	<p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>								
Type	restriction of xs:normalizedString								
Facets	<table> <tr> <td>enumeration</td> <td>none</td> </tr> <tr> <td>enumeration</td> <td>delivery</td> </tr> <tr> <td>enumeration</td> <td>consume</td> </tr> <tr> <td>enumeration</td> <td>both</td> </tr> </table>	enumeration	none	enumeration	delivery	enumeration	consume	enumeration	both
enumeration	none								
enumeration	delivery								
enumeration	consume								
enumeration	both								
Used by	Element typeSds/report								
Source	<pre> <xs:simpleType name="typeReport"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="none"/> <xs:enumeration value="delivery"/> <xs:enumeration value="consume"/> <xs:enumeration value="both"/> </xs:restriction> </xs:simpleType> </pre>								

Simple Type typeOrganisationBlockIdSimple

Namespace	DR-GW		
Annotations	Organisation block send as simple normalized string. The pattern is: id1-id2-id3-id4-id5-id6		
Diagram	<p>Organisation block send as simple normalized string. The pattern is: id1-id2-id3-id4-id5-id6</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>		
Type	restriction of xs:normalizedString		
Facets	<table> <tr> <td>pattern</td> <td>(([0-9] [1-9]\d{0,3} [1-5]\d{4} 6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])-){0,5}(([0-9] [1-9]\d{0,3} [1-5]\d{4} </td> </tr> </table>	pattern	(([0-9] [1-9]\d{0,3} [1-5]\d{4} 6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])-){0,5}(([0-9] [1-9]\d{0,3} [1-5]\d{4}
pattern	(([0-9] [1-9]\d{0,3} [1-5]\d{4} 6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])-){0,5}(([0-9] [1-9]\d{0,3} [1-5]\d{4}		

		6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])
Used by	Element	typeOrganisationBlockId/orgblockIdSimple
Source		<pre><xss:simpleType name="typeOrganisationBlockIdSimple"> <xss:annotation> <xss:documentation>Organisation block send as simple normalized string. The pattern is: id1-id2- id3-id4-id5-id6</xss:documentation> </xss:annotation> <xss:restriction base="xss:normalizedString"> <xss:pattern value="(([0-9] 1-9)\d{0,3} [1-5]\d{4} 6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])-){0,5}(([0-9] 1-9)\d{0,3} [1-5]\d{4} 6[0-4]\d{3} 65[0-4]\d{2} 655[0-2]\d 6553[0-5])"/> </xss:restriction> </xss:simpleType></pre>

Simple Type typeGroupTrackingMask

Namespace	DR-GW
Annotations	Bit mask of one or more typeGroupTrackingMaskValues using bitwise OR.
Diagram	<p>The diagram illustrates the derivation of the typeGroupTrackingMask type. It shows a box labeled "typeGroupTrackingMask" connected by a line with a hollow circle to a box labeled "xs:unsignedShort". A callout box below "typeGroupTrackingMask" states: "Bit mask of one or more typeGroupTrackingMaskValues using bitwise OR.". A callout box below "xs:unsignedShort" states: "Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...".</p>
Type	xs:unsignedShort
Used by	Elements typeGroupTrack/mask, typeGroupTrackSubscriptionEvent/mask
Source	<pre><xss:simpleType name="typeGroupTrackingMask"> <xss:annotation> <xss:documentation>Bit mask of one or more typeGroupTrackingMaskValues using bitwise OR.</xss:documentation> </xss:annotation> <xss:restriction base="xs:unsignedShort" /> </xss:simpleType></pre>

Simple Type typeMembershipType

Namespace	DR-GW						
Annotations	Specifies a group - radio subscriber membership type.						
Diagram	<p>The diagram illustrates the derivation of the typeMembershipType type. It shows a box labeled "typeMembershipType" connected by a line with a hollow circle to a box labeled "xs:normalizedString". A callout box below "typeMembershipType" states: "Specifies a group - radio subscriber membership type.". A callout box below "xs:normalizedString" states: "Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...".</p>						
Type	restriction of xs:normalizedString						
Facets	<table border="1"> <tr> <td>enumeration</td> <td>unknown</td> </tr> <tr> <td>enumeration</td> <td>permanent</td> </tr> <tr> <td>enumeration</td> <td>visiting</td> </tr> </table>	enumeration	unknown	enumeration	permanent	enumeration	visiting
enumeration	unknown						
enumeration	permanent						
enumeration	visiting						
Used by	Element typeGroupAddRadioMember/membership						
Source	<pre><xss:simpleType name="typeMembershipType"> <xss:annotation> <xss:documentation>Specifies a group - radio subscriber membership type.</xss:documentation> </xss:annotation> <xss:restriction base="xs:normalizedString"> <xss:enumeration value="unknown" /> <xss:enumeration value="permanent" /> <xss:enumeration value="visiting" /> </xss:restriction> </xss:simpleType></pre>						

Simple Type typeGroupSelectionLevel

Namespace	DR-GW
-----------	-------

Annotations	Covers tcsScanningPriority_t of the TCS-API.													
Diagram		<p>Covers tcsScanningPriority_t of the TCS-API.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>												
Type	restriction of xs:normalizedString													
Facets	<table> <tr><td>enumeration</td><td>notScanned</td></tr> <tr><td>enumeration</td><td>low</td></tr> <tr><td>enumeration</td><td>normal</td></tr> <tr><td>enumeration</td><td>selected</td></tr> <tr><td>enumeration</td><td>high</td></tr> <tr><td>enumeration</td><td>background</td></tr> </table>		enumeration	notScanned	enumeration	low	enumeration	normal	enumeration	selected	enumeration	high	enumeration	background
enumeration	notScanned													
enumeration	low													
enumeration	normal													
enumeration	selected													
enumeration	high													
enumeration	background													
Used by	Element typeRadioGroupSelection/level													
Source	<pre> <xs:simpleType name="typeGroupSelectionLevel"> <xs:annotation> <xs:documentation>Covers tcsScanningPriority_t of the TCS-API.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="notScanned"/> <xs:enumeration value="low"/> <xs:enumeration value="normal"/> <xs:enumeration value="selected"/> <xs:enumeration value="high"/> <xs:enumeration value="background"/> </xs:restriction> </xs:simpleType> </pre>													

Simple Type typeSystemElementState

Namespace	DR-GW										
Annotations	Specifies connection, server or unit state.										
Diagram		<p>Specifies connection, server or unit state.</p> <p>Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...</p>									
Type	restriction of xs:normalizedString										
Facets	<table> <tr><td>enumeration</td><td>unknown</td><td>Unknown state.</td></tr> <tr><td>enumeration</td><td>ok</td><td>Connection or server is working.</td></tr> <tr><td>enumeration</td><td>n_Ok</td><td>Connection or server is not working.</td></tr> </table>		enumeration	unknown	Unknown state.	enumeration	ok	Connection or server is working.	enumeration	n_Ok	Connection or server is not working.
enumeration	unknown	Unknown state.									
enumeration	ok	Connection or server is working.									
enumeration	n_Ok	Connection or server is not working.									
Used by	Elements typeSystemTetraStatesEvent/cddconnectionState, typeSystemTetraStatesEvent/cddserverState, typeSystemTetraStatesEvent/dxtState, typeSystemTetraStatesEvent/tcsState										
Source	<pre> <xs:simpleType name="typeSystemElementState"> <xs:annotation> <xs:documentation>Specifies connection, server or unit state.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="unknown"> <xs:annotation> <xs:documentation>Unknown state.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="ok"> <xs:annotation> <xs:documentation>Connection or server is working.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="n_Ok"> <xs:annotation> <xs:documentation>Connection or server is not working.</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType> </pre>										

```

    </xs:restriction>
</xs:simpleType>

```

Simple Type typeActionRequest

Namespace	DR-GW																						
Annotations	All possible call actions.																						
Diagram	<p>The diagram shows a UML class named "typeActionRequest" with a hollow circle symbol indicating it is derived from another type. A line connects "typeActionRequest" to a box labeled "xs:normalizedString". Below "typeActionRequest" is a callout box containing "All possible call actions.". Below "xs:normalizedString" is a callout box containing "Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...".</p>																						
Type	restriction of xs:normalizedString																						
Facets	<table> <tr> <td>enumeration</td> <td>setup</td> <td>This method is used to initiate a new call setup. For a call setup to be successful it is required that the resources have been reserved prior this method call.</td> </tr> <tr> <td>enumeration</td> <td>connect</td> <td>This method is used to connect an incoming call.</td> </tr> <tr> <td>enumeration</td> <td>hold</td> <td>This method requests to put an individual call to hold.</td> </tr> <tr> <td>enumeration</td> <td>unhold</td> <td>This method is a request for resuming an individual call from hold.</td> </tr> <tr> <td>enumeration</td> <td>disconnect</td> <td>This method is used to disconnect a call.</td> </tr> <tr> <td>enumeration</td> <td>transfer</td> <td>This method is used to transfer an individual call to a new recipient.</td> </tr> <tr> <td>enumeration</td> <td>releasecall</td> <td>This method is used to release radio subscriber's individual call.</td> </tr> </table>		enumeration	setup	This method is used to initiate a new call setup. For a call setup to be successful it is required that the resources have been reserved prior this method call.	enumeration	connect	This method is used to connect an incoming call.	enumeration	hold	This method requests to put an individual call to hold.	enumeration	unhold	This method is a request for resuming an individual call from hold.	enumeration	disconnect	This method is used to disconnect a call.	enumeration	transfer	This method is used to transfer an individual call to a new recipient.	enumeration	releasecall	This method is used to release radio subscriber's individual call.
enumeration	setup	This method is used to initiate a new call setup. For a call setup to be successful it is required that the resources have been reserved prior this method call.																					
enumeration	connect	This method is used to connect an incoming call.																					
enumeration	hold	This method requests to put an individual call to hold.																					
enumeration	unhold	This method is a request for resuming an individual call from hold.																					
enumeration	disconnect	This method is used to disconnect a call.																					
enumeration	transfer	This method is used to transfer an individual call to a new recipient.																					
enumeration	releasecall	This method is used to release radio subscriber's individual call.																					
Used by	Element typeCallRequest/action																						
Source	<pre> <xs:simpleType name="typeActionRequest"> <xs:annotation> <xs:documentation>All possible call actions.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="setup"> <xs:annotation> <xs:documentation>This method is used to initiate a new call setup. For a call setup to be successful it is required that the resources have been reserved prior this method call.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="connect"> <xs:annotation> <xs:documentation>This method is used to connect an incoming call.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="hold"> <xs:annotation> <xs:documentation>This method requests to put an individual call to hold.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="unhold"> <xs:annotation> <xs:documentation>This method is a request for resuming an individual call from hold.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="disconnect"> <xs:annotation> <xs:documentation>This method is used to disconnect a call.</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="transfer"> <xs:annotation> <xs:documentation>This method is used to transfer an individual call to a new recipient.</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType> </pre>																						

```

<xs:enumeration value="releasecall">
  <xs:annotation>
    <xs:documentation>This method is used to release radio subscriber's individual call.</xs:documentation>
  </xs:annotation>
</xs:enumeration>
</xs:restriction>
</xs:simpleType>

```

Simple Type typeKeyManagementTextPriority

Namespace	DR-GW				
Annotations	Defines the priority of the keymanagement information.				
Diagram	<p>The diagram shows a UML class named "typeKeyManagementTextPriority" with a hollow circle symbol indicating it is a restriction. It is connected by a line to another class named "xs:normalizedString". Below the classes are two text boxes: one stating "Defines the priority of the keymanagement information." and another stating "Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...".</p>				
Type	restriction of xs:normalizedString				
Facets	<table border="1"> <tr> <td>enumeration</td> <td>normal</td> </tr> <tr> <td>enumeration</td> <td>high</td> </tr> </table>	enumeration	normal	enumeration	high
enumeration	normal				
enumeration	high				
Source	<pre> <xs:simpleType name="typeKeyManagementTextPriority"> <xs:annotation> <xs:documentation>Defines the priority of the keymanagement information.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="normal"/> <xs:enumeration value="high"/> </xs:restriction> </xs:simpleType> </pre>				

Simple Type typeAddressingStyle

Namespace	DR-GW				
Annotations	Describes the IP addressing style. Unicast or multicast.				
Diagram	<p>The diagram shows a UML class named "typeAddressingStyle" with a hollow circle symbol indicating it is a restriction. It is connected by a line to another class named "xs:normalizedString". Below the classes are two text boxes: one stating "Describes the IP addressing style. Unicast or multicast." and another stating "Built-in derived type. The normalizedString datatype represents white space normalized strings. The base type of...".</p>				
Type	restriction of xs:normalizedString				
Facets	<table border="1"> <tr> <td>enumeration</td> <td>ucast</td> </tr> <tr> <td>enumeration</td> <td>mcast</td> </tr> </table>	enumeration	ucast	enumeration	mcast
enumeration	ucast				
enumeration	mcast				
Source	<pre> <xs:simpleType name="typeAddressingStyle"> <xs:annotation> <xs:documentation>Describes the IP addressing style. Unicast or multicast.</xs:documentation> </xs:annotation> <xs:restriction base="xs:normalizedString"> <xs:enumeration value="ucast"/> <xs:enumeration value="mcast"/> </xs:restriction> </xs:simpleType> </pre>				

Simple Type typeGroupTrackingMaskValues

Namespace	DR-GW
Annotations	
Diagram	<p>The diagram shows a UML class named "typeGroupTrackingMaskValues" with a hollow circle symbol indicating it is a restriction. It is connected by a line to another class named "xs:unsignedShort". Below the classes is a single text box stating "Built-in derived type. The unsignedShort datatype is derived from unsignedInt by setting the value of maxInclusive to...".</p>
Type	restriction of xs:unsignedShort

Facets	enumeration 0 enumeration 1 enumeration 2 enumeration 4 enumeration 8 enumeration 16 enumeration 65535	TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_BASIC_C TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_RS_ADD_REMOVE_C TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_G4WIF_ADD_REMOVE_C TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_WSUSER_ADD_REMOVE_C TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_CBR_REMOVE_C TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_GROUP_ADD_REMOVE_C TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_ALL_C
Source	<pre> <xs:simpleType name="typeGroupTrackingMaskValues"> <xs:annotation> <xs:documentation/> </xs:annotation> <xs:restriction bases="xs:unsignedShort"> <xs:enumeration value="0"> <xs:annotation> <xs:documentation>TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_BASIC_C</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="1"> <xs:annotation> <xs:documentation>TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_RS_ADD_REMOVE_C</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="2"> <xs:annotation> <xs:documentation>TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_G4WIF_ADD_REMOVE_C</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="4"> <xs:annotation> <xs:documentation>TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_WSUSER_ADD_REMOVE_C</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="8"> <xs:annotation> <xs:documentation>TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_CBR_REMOVE_C</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="16"> <xs:annotation> <xs:documentation>TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_GROUP_ADD_REMOVE_C</xs:documentation> </xs:annotation> </xs:enumeration> <xs:enumeration value="65535"> <xs:annotation> <xs:documentation>TCS_GROUP_SUBSCRIPTION_MASK_VALUES_T_ALL_C</xs:documentation> </xs:annotation> </xs:enumeration> </xs:restriction> </xs:simpleType> </pre>	

Attribute(s)

Attribute drgw / @version

Namespace	DR-GW
Properties	use: required fixed: 2.0
Used by	Element drgw
Source	<xs:attribute name="version" fixed="2.0" use="required"/>