

INTERNATIONAL STANDARD

NORME INTERNATIONALE



Application integration at electric utilities – System interfaces for distribution management –

Part 11: Common information model (CIM) extensions for distribution

Intégration d'applications pour les services électriques – Interfaces système pour la gestion de la distribution –

Partie 11: Extensions du modèle d'information commun (CIM) pour la distribution



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2010 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester.

If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de la CEI ou du Comité national de la CEI du pays du demandeur.

Si vous avez des questions sur le copyright de la CEI ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de la CEI de votre pays de résidence.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,...). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch

Tel.: +41 22 919 02 11

Fax: +41 22 919 03 00

A propos de la CEI

La Commission Electrotechnique Internationale (CEI) est la première organisation mondiale qui élabore et publie des normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications CEI

Le contenu technique des publications de la CEI est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

- Catalogue des publications de la CEI: www.iec.ch/searchpub/cur_fut-f.htm

Le Catalogue en-ligne de la CEI vous permet d'effectuer des recherches en utilisant différents critères (numéro de référence, texte, comité d'études,...). Il donne aussi des informations sur les projets et les publications retirées ou remplacées.

- Just Published CEI: www.iec.ch/online_news/justpub

Restez informé sur les nouvelles publications de la CEI. Just Published détaille deux fois par mois les nouvelles publications parues. Disponible en-ligne et aussi par email.

- Electropedia: www.electropedia.org

Le premier dictionnaire en ligne au monde de termes électroniques et électriques. Il contient plus de 20 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans les langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International en ligne.

- Service Clients: www.iec.ch/webstore/custserv/custserv_entry-f.htm

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions, visitez le FAQ du Service clients ou contactez-nous:

Email: csc@iec.ch

Tél.: +41 22 919 02 11

Fax: +41 22 919 03 00

INTERNATIONAL STANDARD

NORME INTERNATIONALE



Application integration at electric utilities – System interfaces for distribution management –

Part 11: Common information model (CIM) extensions for distribution

Intégration d'applications pour les services électriques – Interfaces système pour la gestion de la distribution –

Partie 11: Extensions du modèle d'information commun (CIM) pour la distribution

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

PRICE CODE
CODE PRIX

XH

ICS 33.200

ISBN 978-2-88912-885-3

CONTENTS

FOREWORD.....	10
INTRODUCTION.....	12
1 Scope	13
2 Normative references	14
3 Terms and definitions	14
4 CIM specification	15
4.1 CIM modeling notation	15
4.2 CIM packages	15
4.2.1 General	15
4.2.2 TC 57 CIM packages.....	16
4.2.3 CIM extensions for distribution packages (this document)	16
4.3 CIM UML modelling	17
4.3.1 General	17
4.3.2 Scope of UML model.....	18
4.3.3 Extensibility	18
4.3.4 Message (profile) definition	19
4.4 DCIM model concepts and examples	19
4.4.1 General	19
4.4.2 Key classes in DCIM	19
4.4.3 Single-phase and unbalanced loads	20
4.4.4 Distribution line segments	21
4.4.5 Distribution transformers	26
4.4.6 Connectivity with unbalanced phases	30
4.4.7 Electrical model vs. physical model	32
4.4.8 Locations and graphical representations	33
4.4.9 Metering	33
4.4.10 PaymentMetering	35
4.5 Other.....	39
5 Detailed model	39
5.1 Overview	39
5.2 Context	39
6 Package architecture (normative).....	41
6.1 General.....	41
6.2 Top package IEC61968	41
6.2.1 IEC61968CIMVersion	43
6.2.2 Package Common	44
6.2.3 Package WiresExt.....	56
6.2.4 Package Assets	69
6.2.5 Package AssetModels	75
6.2.6 Package Work	94
6.2.7 Package Customers	96
6.2.8 Package Metering	105
6.2.9 Package LoadControl.....	129
6.2.10 Package PaymentMetering	132
Bibliography	162

Figure 1 – TC 57 CIM packages	16
Figure 2 – CIM extensions for distribution (DCIM) top-level packages	17
Figure 3 – DCIM key classes.....	20
Figure 4 – DCIM load model.....	21
Figure 5 – DCIM line connectivity model.....	22
Figure 6 – DCIM conductor (line and cable datasheet) model.....	23
Figure 7 – DCIM transformer connectivity model.....	26
Figure 8 – DCIM transformer datasheet model.....	27
Figure 9 – DCIM tap changer model	29
Figure 10 – Example of a distribution transformer that can be modelled with DCIM.....	30
Figure 11 – DCIM connectivity for two-Terminal devices	31
Figure 12 – DCIM connectivity for single-Terminal devices	32
Figure 13 – DCIM assets and relation to power system resources	33
Figure 14 – DCIM power system resource and asset locations.....	33
Figure 15 – DCIM metering model	34
Figure 16 – DCIM transacting model	35
Figure 17 – DCIM receipting model	36
Figure 18 – DCIM auxiliary agreement model	37
Figure 19 – DCIM pricing structure model.....	38
Figure 20 – Package diagram IEC61968::Main	41
Figure 21 – Package diagram IEC61968::Dependencies	42
Figure 22 – Package diagram IEC61968::StdCIM	43
Figure 23 – Logical diagram IEC61968::DCIMKeyClasses	43
Figure 24 – Logical diagram Common::CommonInheritance	44
Figure 25 – Logical diagram Common::CommonOverview	45
Figure 26 – Logical diagram Common::DCIMLocations	45
Figure 27 – Logical diagram WiresExt::WiresExtInheritance.....	56
Figure 28 – Logical diagram WiresExt::DCIMLoadModel	57
Figure 29 – Logical diagram WiresExt::DCIMLineModel	58
Figure 30 – Logical diagram WiresExt::DCIMTransformerModel	59
Figure 31 – Logical diagram WiresExt::DCIMTapChangerModel.....	60
Figure 32 – Logical diagram Assets::AssetsInheritance.....	69
Figure 33 – Logical diagram Assets::AssetsOverview	70
Figure 34 – Logical diagram Assets::DCIMAssetsAndPSRs	70
Figure 35 – Logical diagram AssetModels::AssetModelsInheritance	76
Figure 36 – Logical diagram AssetModels::AssetModelsOverview	77
Figure 37 – Logical diagram AssetModels::DCIMConductorInfo.....	78
Figure 38 – Logical diagram AssetModels::DCIMTransformerInfo.....	79
Figure 39 – Logical diagram Work::WorkInheritance	94
Figure 40 – Logical diagram Work::WorkOverview	94
Figure 41 – Logical diagram Customers::CustomersInheritance	96
Figure 42 – Logical diagram Customers::CustomersOverview	97

Figure 43 – Logical diagram Metering::MeteringInheritance	106
Figure 44 – Logical diagram Metering::MeteringOverviewShort	107
Figure 45 – Logical diagram Metering::MeteringOverview	108
Figure 46 – Logical diagram Metering::MeteringRelationships	109
Figure 47 – Logical diagram LoadControl::LoadControlInheritance	129
Figure 48 – Logical diagram LoadControl::LoadControlOverview	130
Figure 49 – Logical diagram PaymentMetering::PaymentMeteringInheritance	133
Figure 50 – Logical diagram PaymentMetering::PaymentMeteringOverview	134
Figure 51 – Logical diagram PaymentMetering::PaymentMeteringRelationships	135
Figure 52 – Logical diagram PaymentMetering::Transacting	136
Figure 53 – Logical diagram PaymentMetering::Receipting	137
Figure 54 – Logical diagram PaymentMetering::AuxiliaryAgreement	138
Figure 55 – Logical diagram PaymentMetering::TariffProfile	139
Table 1 – Open wye/open delta transformer bank connections	30
Table 2 – Attribute documentation	40
Table 3 – Association ends documentation	40
Table 4 – Enums documentation	40
Table 5 – Attributes of IEC61968::IEC61968CIMVersion	44
Table 6 – Attributes of Common::DateTimeInterval	46
Table 7 – Attributes of Common::Status	46
Table 8 – Attributes of Common::PostalAddress	46
Table 9 – Attributes of Common::StreetAddress	47
Table 10 – Attributes of Common::StreetDetail	47
Table 11 – Attributes of Common::TownDetail	47
Table 12 – Attributes of Common::ElectronicAddress	48
Table 13 – Attributes of Common::TelephoneNumber	48
Table 14 – Attributes of Common::ActivityRecord	48
Table 15 – Association ends of Common::ActivityRecord with other classes	49
Table 16 – Attributes of Common::Agreement	49
Table 17 – Association ends of Common::Agreement with other classes	50
Table 18 – Attributes of Common::CoordinateSystem	50
Table 19 – Association ends of Common::CoordinateSystem with other classes	50
Table 20 – Attributes of Common::Document	50
Table 21 – Association ends of Common::Document with other classes	51
Table 22 – Attributes of Common::Location	51
Table 23 – Association ends of Common::Location with other classes	52
Table 24 – Attributes of Common::Organisation	52
Table 25 – Attributes of Common::PositionPoint	53
Table 26 – Association ends of Common::PositionPoint with other classes	53
Table 27 – Attributes of Common::TimePoint	53
Table 28 – Association ends of Common::TimePoint with other classes	54

Table 29 – Attributes of Common::TimeSchedule	54
Table 30 – Association ends of Common::TimeSchedule with other classes	55
Table 31 – Attributes of Common::UserAttribute	55
Table 32 – Association ends of Common::UserAttribute with other classes	56
Table 33 – Attributes of WiresExt::DistributionLineSegment	61
Table 34 – Association ends of WiresExt::DistributionLineSegment with other classes	61
Table 35 – Attributes of WiresExt::DistributionTapChanger	62
Table 36 – Association ends of WiresExt::DistributionTapChanger with other classes	63
Table 37 – Attributes of WiresExt::DistributionTransformer	63
Table 38 – Association ends of WiresExt::DistributionTransformer with other classes	64
Table 39 – Attributes of WiresExt::DistributionTransformerWinding	65
Table 40 – Association ends of WiresExt::DistributionTransformerWinding with other classes	65
Table 41 – Attributes of WiresExt::PerLengthPhaseImpedance	66
Table 42 – Association ends of WiresExt::PerLengthPhaseImpedance with other classes	66
Table 43 – Attributes of WiresExt::PerLengthSequenceImpedance	66
Table 44 – Association ends of WiresExt::PerLengthSequenceImpedance with other classes	67
Table 45 – Attributes of WiresExt::PhaseImpedanceData	67
Table 46 – Association ends of WiresExt::PhaseImpedanceData with other classes	67
Table 47 – Attributes of WiresExt::TransformerBank	68
Table 48 – Association ends of WiresExt::TransformerBank with other classes	68
Table 49 – Attributes of WiresExt::WindingPiImpedance	68
Table 50 – Association ends of WiresExt::WindingPiImpedance with other classes	69
Table 51 – Attributes of Assets::AcceptanceTest	70
Table 52 – Literals of Assets::SealConditionKind	71
Table 53 – Literals of Assets::SealKind	71
Table 54 – Attributes of Assets::Asset	71
Table 55 – Association ends of Assets::Asset with other classes	72
Table 56 – Attributes of Assets::AssetContainer	73
Table 57 – Association ends of Assets::AssetContainer with other classes	73
Table 58 – Attributes of Assets::AssetFunction	74
Table 59 – Attributes of Assets::ComMediaAsset	74
Table 60 – Association ends of Assets::ComMediaAsset with other classes	75
Table 61 – Attributes of Assets::Seal	75
Table 62 – Association ends of Assets::Seal with other classes	75
Table 63 – Literals of AssetModels::AssetModelUsageKind	79
Table 64 – Literals of AssetModels::CorporateStandardKind	80
Table 65 – Literals of AssetModels::CableConstructionKind	80
Table 66 – Literals of AssetModels::CableOuterJacketKind	80
Table 67 – Literals of AssetModels::CableShieldMaterialKind	81
Table 68 – Literals of AssetModels::ConductorInsulationKind	81
Table 69 – Literals of AssetModels::ConductorMaterialKind	81

Table 70 – Literals of AssetModels::ConductorUsageKind.....	82
Table 71 – Attributes of AssetModels::AssetModel.....	82
Table 72 – Attributes of AssetModels::CableInfo.....	83
Table 73 – Association ends of AssetModels::CableInfo with other classes	83
Table 74 – Attributes of AssetModels::ConcentricNeutralCableInfo	84
Table 75 – Association ends of AssetModels::ConcentricNeutralCableInfo with other classes.....	84
Table 76 – Attributes of AssetModels::ConductorInfo.....	85
Table 77 – Association ends of AssetModels::ConductorInfo with other classes	85
Table 78 – Attributes of AssetModels::DistributionWindingTest	85
Table 79 – Association ends of AssetModels::DistributionWindingTest with other classes.....	86
Table 80 – Attributes of AssetModels::EndDeviceModel.....	86
Table 81 – Association ends of AssetModels::EndDeviceModel with other classes	86
Table 82 – Attributes of AssetModels::OpenCircuitTest.....	87
Table 83 – Association ends of AssetModels::OpenCircuitTest with other classes	87
Table 84 – Attributes of AssetModels::OverheadConductorInfo	87
Table 85 – Association ends of AssetModels::OverheadConductorInfo with other classes.....	88
Table 86 – Attributes of AssetModels::ShortCircuitTest.....	88
Table 87 – Association ends of AssetModels::ShortCircuitTest with other classes	89
Table 88 – Attributes of AssetModels::TapeShieldCableInfo.....	89
Table 89 – Association ends of AssetModels::TapeShieldCableInfo with other classes	90
Table 90 – Attributes of AssetModels::ToWindingSpec	90
Table 91 – Association ends of AssetModels::ToWindingSpec with other classes	90
Table 92 – Attributes of AssetModels::TransformerInfo	91
Table 93 – Association ends of AssetModels::TransformerInfo with other classes.....	91
Table 94 – Attributes of AssetModels::WindingInfo	91
Table 95 – Association ends of AssetModels::WindingInfo with other classes.....	92
Table 96 – Attributes of AssetModels::WireArrangement.....	92
Table 97 – Association ends of AssetModels::WireArrangement with other classes	93
Table 98 – Attributes of AssetModels::WireType.....	93
Table 99 – Association ends of AssetModels::WireType with other classes	94
Table 100 – Literals of Work::WorkKind.....	95
Table 101 – Attributes of Work::Work	95
Table 102 – Association ends of Work::Work with other classes.....	96
Table 103 – Literals of Customers::CustomerKind	97
Table 104 – Literals of Customers::RevenueKind.....	98
Table 105 – Literals of Customers::ServiceKind.....	98
Table 106 – Attributes of Customers::Customer.....	98
Table 107 – Association ends of Customers::Customer with other classes	99
Table 108 – Attributes of Customers::CustomerAccount.....	99
Table 109 – Association ends of Customers::CustomerAccount with other classes	100
Table 110 – Attributes of Customers::CustomerAgreement	100

Table 111 – Association ends of Customers::CustomerAgreement with other classes	101
Table 112 – Attributes of Customers::PricingStructure	101
Table 113 – Association ends of Customers::PricingStructure with other classes	102
Table 114 – Attributes of Customers::ServiceCategory	103
Table 115 – Association ends of Customers::ServiceCategory with other classes	103
Table 116 – Attributes of Customers::ServiceLocation	103
Table 117 – Association ends of Customers::ServiceLocation with other classes	104
Table 118 – Attributes of Customers::Tariff	105
Table 119 – Association ends of Customers::Tariff with other classes	105
Table 120 – Literals of Metering::DemandKind	110
Table 121 – Attributes of Metering::DynamicDemand	110
Table 122 – Literals of Metering::ReadingKind	110
Table 123 – Attributes of Metering::ComFunction	111
Table 124 – Association ends of Metering::ComFunction with other classes	111
Table 125 – Attributes of Metering::DemandResponseProgram	112
Table 126 – Association ends of Metering::DemandResponseProgram with other classes	112
Table 127 – Attributes of Metering::DeviceFunction	112
Table 128 – Association ends of Metering::DeviceFunction with other classes	113
Table 129 – Attributes of Metering::ElectricMeteringFunction	113
Table 130 – Association ends of Metering::ElectricMeteringFunction with other classes	114
Table 131 – Attributes of Metering::EndDeviceAsset	114
Table 132 – Association ends of Metering::EndDeviceAsset with other classes	116
Table 133 – Attributes of Metering::EndDeviceControl	117
Table 134 – Association ends of Metering::EndDeviceControl with other classes	117
Table 135 – Attributes of Metering::EndDeviceEvent	117
Table 136 – Association ends of Metering::EndDeviceEvent with other classes	118
Table 137 – Attributes of Metering::EndDeviceGroup	118
Table 138 – Association ends of Metering::EndDeviceGroup with other classes	118
Table 139 – Association ends of Metering::IntervalBlock with other classes	119
Table 140 – Attributes of Metering::IntervalReading	119
Table 141 – Association ends of Metering::IntervalReading with other classes	120
Table 142 – Attributes of Metering::MeterAsset	120
Table 143 – Association ends of Metering::MeterAsset with other classes	121
Table 144 – Attributes of Metering::MeterReading	122
Table 145 – Association ends of Metering::MeterReading with other classes	122
Table 146 – Attributes of Metering::MeterServiceWork	122
Table 147 – Association ends of Metering::MeterServiceWork with other classes	123
Table 148 – Attributes of Metering::Pending	123
Table 149 – Association ends of Metering::Pending with other classes	124
Table 150 – Attributes of Metering::Reading	124
Table 151 – Association ends of Metering::Reading with other classes	124
Table 152 – Attributes of Metering::ReadingQuality	125

Table 153 – Association ends of Metering::ReadingQuality with other classes	125
Table 154 – Attributes of Metering::ReadingType	125
Table 155 – Association ends of Metering::ReadingType with other classes	126
Table 156 – Attributes of Metering::Register	126
Table 157 – Association ends of Metering::Register with other classes	127
Table 158 – Attributes of Metering::SDPLocation	127
Table 159 – Association ends of Metering::SDPLocation with other classes	128
Table 160 – Attributes of Metering::ServiceDeliveryPoint	128
Table 161 – Association ends of Metering::ServiceDeliveryPoint with other classes	129
Table 162 – Attributes of LoadControl::RemoteConnectDisconnectInfo	130
Table 163 – Attributes of LoadControl::ConnectDisconnectFunction	131
Table 164 – Association ends of LoadControl::ConnectDisconnectFunction with other classes	132
Table 165 – Attributes of PaymentMetering::AccountMovement	139
Table 166 – Attributes of PaymentMetering::AccountingUnit	140
Table 167 – Attributes of PaymentMetering::BankAccountDetail	140
Table 168 – Attributes of PaymentMetering::Due	140
Table 169 – Attributes of PaymentMetering::LineDetail	141
Table 170 – Literals of PaymentMetering::ChargeKind	141
Table 171 – Literals of PaymentMetering::ChequeKind	142
Table 172 – Literals of PaymentMetering::CreditKind	142
Table 173 – Literals of PaymentMetering::SupplierKind	142
Table 174 – Literals of PaymentMetering::TenderKind	142
Table 175 – Literals of PaymentMetering::TransactionKind	143
Table 176 – Attributes of PaymentMetering::AuxiliaryAccount	143
Table 177 – Association ends of PaymentMetering::AuxiliaryAccount with other classes	144
Table 178 – Attributes of PaymentMetering::AuxiliaryAgreement	144
Table 179 – Association ends of PaymentMetering::AuxiliaryAgreement with other classes	146
Table 180 – Attributes of PaymentMetering::Card	146
Table 181 – Association ends of PaymentMetering::Card with other classes	146
Table 182 – Attributes of PaymentMetering::Cashier	147
Table 183 – Association ends of PaymentMetering::Cashier with other classes	147
Table 184 – Attributes of PaymentMetering::CashierShift	147
Table 185 – Association ends of PaymentMetering::CashierShift with other classes	148
Table 186 – Attributes of PaymentMetering::Charge	148
Table 187 – Association ends of PaymentMetering::Charge with other classes	148
Table 188 – Attributes of PaymentMetering::Cheque	149
Table 189 – Association ends of PaymentMetering::Cheque with other classes	149
Table 190 – Attributes of PaymentMetering::ConsumptionTariffInterval	149
Table 191 – Association ends of PaymentMetering::ConsumptionTariffInterval with other classes	150
Table 192 – Attributes of PaymentMetering::MerchantAccount	150

Table 193 – Association ends of PaymentMetering::MerchantAccount with other classes	151
Table 194 – Attributes of PaymentMetering::MerchantAgreement.....	151
Table 195 – Association ends of PaymentMetering::MerchantAgreement with other classes	152
Table 196 – Attributes of PaymentMetering::PointOfSale	152
Table 197 – Association ends of PaymentMetering::PointOfSale with other classes.....	152
Table 198 – Attributes of PaymentMetering::Receipt.....	153
Table 199 – Association ends of PaymentMetering::Receipt with other classes	153
Table 200 – Attributes of PaymentMetering::ServiceSupplier	153
Table 201 – Association ends of PaymentMetering::ServiceSupplier with other classes	154
Table 202 – Attributes of PaymentMetering::Shift	155
Table 203 – Attributes of PaymentMetering::TariffProfile.....	156
Table 204 – Association ends of PaymentMetering::TariffProfile with other classes	156
Table 205 – Attributes of PaymentMetering::Tender.....	157
Table 206 – Association ends of PaymentMetering::Tender with other classes	157
Table 207 – Attributes of PaymentMetering::TimeTariffInterval.....	157
Table 208 – Association ends of PaymentMetering::TimeTariffInterval with other classes	158
Table 209 – Attributes of PaymentMetering::Transaction	158
Table 210 – Association ends of PaymentMetering::Transaction with other classes	159
Table 211 – Attributes of PaymentMetering::Transactor.....	159
Table 212 – Association ends of PaymentMetering::Transactor with other classes	160
Table 213 – Attributes of PaymentMetering::Vendor	160
Table 214 – Association ends of PaymentMetering::Vendor with other classes	160
Table 215 – Attributes of PaymentMetering::VendorShift	161
Table 216 – Association ends of PaymentMetering::VendorShift with other classes	161

DRAFT

INTERNATIONAL ELECTROTECHNICAL COMMISSION

APPLICATION INTEGRATION AT ELECTRIC UTILITIES – SYSTEM INTERFACES FOR DISTRIBUTION MANAGEMENT –

Part 11: Common information model (CIM) extensions for distribution

FOREWORD

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61968-11 has been prepared by IEC technical committee 57: Power systems management and associated information exchange.

This bilingual version (2013-01) corresponds to the monolingual English version, published in 2010-07.

The text of this standard is based on the following documents:

FDIS	Report on voting
57/1058/FDIS	57/1073/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

The French version of this standard has not been voted upon.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

In informative sections of this document, words printed in Tahoma Bold apply to terms that are used as tokens in the normative clauses (to facilitate the reading and the text search).

A list of all parts of the IEC 61968 series, under the general title: *Application integration at electric utilities – System interfaces for distribution management*, can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

Withdawn

INTRODUCTION

The IEC 61968 series of standards is intended to facilitate inter-application integration as opposed to intra-application integration. Intra-application integration is aimed at programs in the same application system, usually communicating with each other using middleware that is embedded in their underlying runtime environment, and tends to be optimised for close, real-time, synchronous connections and interactive request/reply or conversation communication models. Therefore, these interface standards are relevant to loosely coupled applications with more heterogeneity in languages, operating systems, protocols and management tools. This series of standards is intended to support applications that need to exchange data every few seconds, minutes, or hours rather than waiting for a nightly batch run. This series of standards, which are intended to be implemented with middleware services that exchange messages among applications, will complement, not replace utility data warehouses, database gateways, and operational stores.

As used in IEC 61968, a distribution management system (DMS) consists of various distributed application components for the utility to manage electrical distribution networks. These capabilities include monitoring and control of equipment for power delivery, management processes to ensure system reliability, voltage management, demand-side management, outage management, work management, automated mapping and facilities management. Standard interfaces are defined for each class of applications identified in the interface reference model (IRM), which is described in IEC 61968-1.

Withdrawn

APPLICATION INTEGRATION AT ELECTRIC UTILITIES – SYSTEM INTERFACES FOR DISTRIBUTION MANAGEMENT –

Part 11: Common information model (CIM) extensions for distribution

1 Scope

This International Standard specifies the distribution extensions of the Common Information Model (CIM) specified in IEC 61970-301. It defines a standard set of extensions of common information model (CIM), which support message definitions in Parts 3 to 9 of IEC 61968, IEC 61968-13 and IEC 61968-14¹⁾. The scope of this document is the information model that extends the base CIM for the needs of distribution networks, as well as for integration with enterprise-wide information systems typically used within electrical utilities. The information model is defined in UML which is platform-independent and electronically processable language that is then used to create message payload definitions in different required formats. In this way, this standard will not be impacted by the specification, development and/or deployment of next generation infrastructures, either through the use of standards or proprietary means.

For the purposes of this document, the distribution CIM (DCIM) model refers to the IEC TC 57 CIM model as defined by IEC 61970-301 and IEC 61968-11 (this document).

The Common Information Model (CIM) is an abstract model of the major objects in an electric utility enterprise typically involved in utility operations. By providing a standard way of representing power system resources as object classes and attributes, along with their relationships, the CIM facilitates the integration of software applications developed independently by different vendors. The CIM facilitates integration by defining a common language (i.e., semantics and syntax) based on the CIM to enable these applications or systems to access public data and exchange information independent of how such information is represented internally.

IEC 61970-301 defines a core CIM for Energy Management System (EMS) applications, including many classes that would be useful in a wider variety of applications. Due to its size, the CIM classes are grouped into logical Packages, and collections of these packages are maintained as separate International Standards. This document extends the core CIM with packages that focus on Distribution Management Systems (DMS) including Assets, Work, Customers, Load Control, Metering, and others. IEC 61970-302²⁾ extends the CIM with packages that focus on Financial, Energy Scheduling, Reservation, and other market-related applications. Other CIM extensions may be published as International Standards, each maintained by a separate group of domain experts. Depending on a project's needs, the integration of applications may require classes and packages from one or more of the CIM standards.

1) IEC 61968-5, IEC 61968-6, IEC 61968-7, IEC 61968-8 and IEC 61968-14 are under consideration.

2) Under consideration.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61968-1, *Application integration at electric utilities – System interfaces for distribution management – Part 1: Interface architecture and general requirements*

IEC 61968-2, *Application integration at electric utilities – System interfaces for distribution management – Part 2: Glossary*

IEC 61970-301, *Energy management system application program interface (EMS-API) – Part 301: Common information model (CIM) base*

IEC 61970-501, *Energy management system application program interface (EMS-API) – Part 501: Common Information Model Resource Description Framework (CIM RDF) schema*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 61968-2 and the following apply.

NOTE Refer to International Electrotechnical Vocabulary, IEC 60050, for general glossary definitions.

3.1 energy management system EMS

computer system comprising a software platform providing basic support services and a set of applications providing the functionality needed for the effective operation of electrical generation and transmission facilities so as to assure adequate security of energy supply at minimum cost

3.2 distribution management system DMS

computer system comprising a software platform providing basic support services and a set of applications providing the functionality needed for the effective operation of electrical distribution facilities so as to assure adequate security of energy supply at minimum cost

3.3 unified modeling language UML

formal and comprehensive descriptive language with diagramming techniques used to represent software systems, from requirements analysis, through design and implementation, to documentation

UML has evolved from a collection of methods contributed by different practitioners, into an International Standard. The CIM relies on UML for defining the model, and automated tools generate the documentation, schemas, and other artifacts directly from the UML. A basic understanding of UML is necessary to understand the CIM.

3.4 common information model with distribution extensions DCIM

union of the core CIM in IEC 61970-301 and additional packages defined in this document, IEC 61968-11

The DCIM is intended to address most of the domain modelling needs of a DMS; however, a specific project may require other CIM packages or extensions.

3.5 profile

subset of DCIM classes, associations and attributes needed to accomplish a specific type of interface

It may be expressed in XSD, RDF, and/or OWL files. A profile can be tested between applications. A profile is necessary in order to “use” the DCIM. Several profiles are defined in other parts of the IEC 61968 family of standards.

3.6 XML schema

schema used to define the structure, content, and semantics of extensible markup language (XML) files

XML schemas are generally found in files with an “xsd” extension. The DCIM uses XSD files to define inter-application messages in most domain areas, except for power system model exchange.

3.7 resource description framework RDF

web (W3C) standard used to represent information models

RDF is more powerful than XSD because it can describe a data model, not just an XML file. The DCIM uses a subset of RDF to support power system model exchange.

3.8 web ontology language OWL

another Web (W3C) standard that includes RDF and extends it

OWL is more powerful than RDF in supporting data types, enumerations, more details of class relationships and associations, etc. Future DCIM profiles may use OWL.

4 CIM specification

4.1 CIM modeling notation

The CIM is defined using object-oriented modeling techniques. Specifically, the CIM specification uses the Unified Modeling Language (UML) notation, which defines the CIM as a group of packages.

Each package in the CIM contains one or more class diagrams showing graphically all the classes in that package and their relationships. Each class is then defined in text in terms of its attributes and relationships to other classes.

The UML notation is described in Object Management Group (OMG) documents and several published textbooks.

4.2 CIM packages

4.2.1 General

The CIM is partitioned into a set of packages. A package is a general purpose means of grouping related model elements. The packages have been chosen to make the model easier to design, understand and review. The common information model consists of several sets of

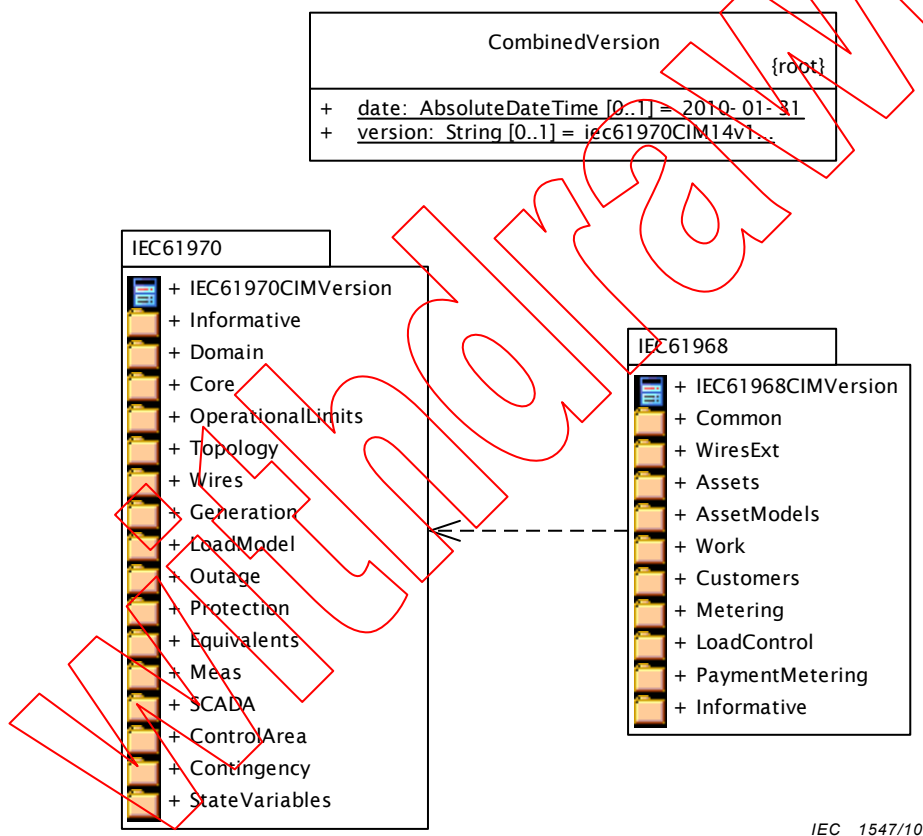
packages. Entities may have associations that cross many package boundaries. Each application will use information represented in several packages.

4.2.2 TC 57 CIM packages

The comprehensive CIM is partitioned into groups of packages for convenience. These groups include:

- IEC 61970-301 (base CIM, defining data types and power system resources as required by typical EMS and DMS control centre applications);
- IEC 61968-11 (this document).

Figure 1 shows all currently defined TC 57 CIM normative packages and their dependency relationships. The dashed line indicates a dependency relationship, with the arrowhead pointing from the dependent package to the package on which it has a dependency.



IEC 1547/10

Figure 1 – TC 57 CIM packages

NOTE The contents of the base CIM referred to from within this specification were auto-generated from the base CIM UML electronic model release IEC61970CIM14v13.

4.2.3 CIM extensions for distribution packages (this document)

The base CIM model as defined by IEC 61970-301 defines a set of sub-packages which includes **Wires**, **Topology**, **Measurements**, **Equivalents** and **Core**, as well as several other ones. Parts 3 to 9 of IEC 61968, IEC 61968-13 and IEC 61968-14 required extensions to the CIM model as specified by IEC 61970-301 in order to describe the objects and associated properties which are relevant to distribution modelling and information exchanges applicable not only to typical control room systems, but also with enterprise and partner systems. Therefore, just as applications in the distribution domain use classes from the base CIM, so might applications outside the distribution domain use classes defined in this document.

Figure 2 shows the packages defined for IEC 61968-11 CIM extensions for distribution. Notes on the left-hand side of the figure indicate the part of IEC 61968 that has been driving the definition of classes within the respective package. Note, however, that different documents in IEC 61968 series, as well as different applications that use CIM for information exchange will typically define messages using classes from several packages, including some defined outside of this document.

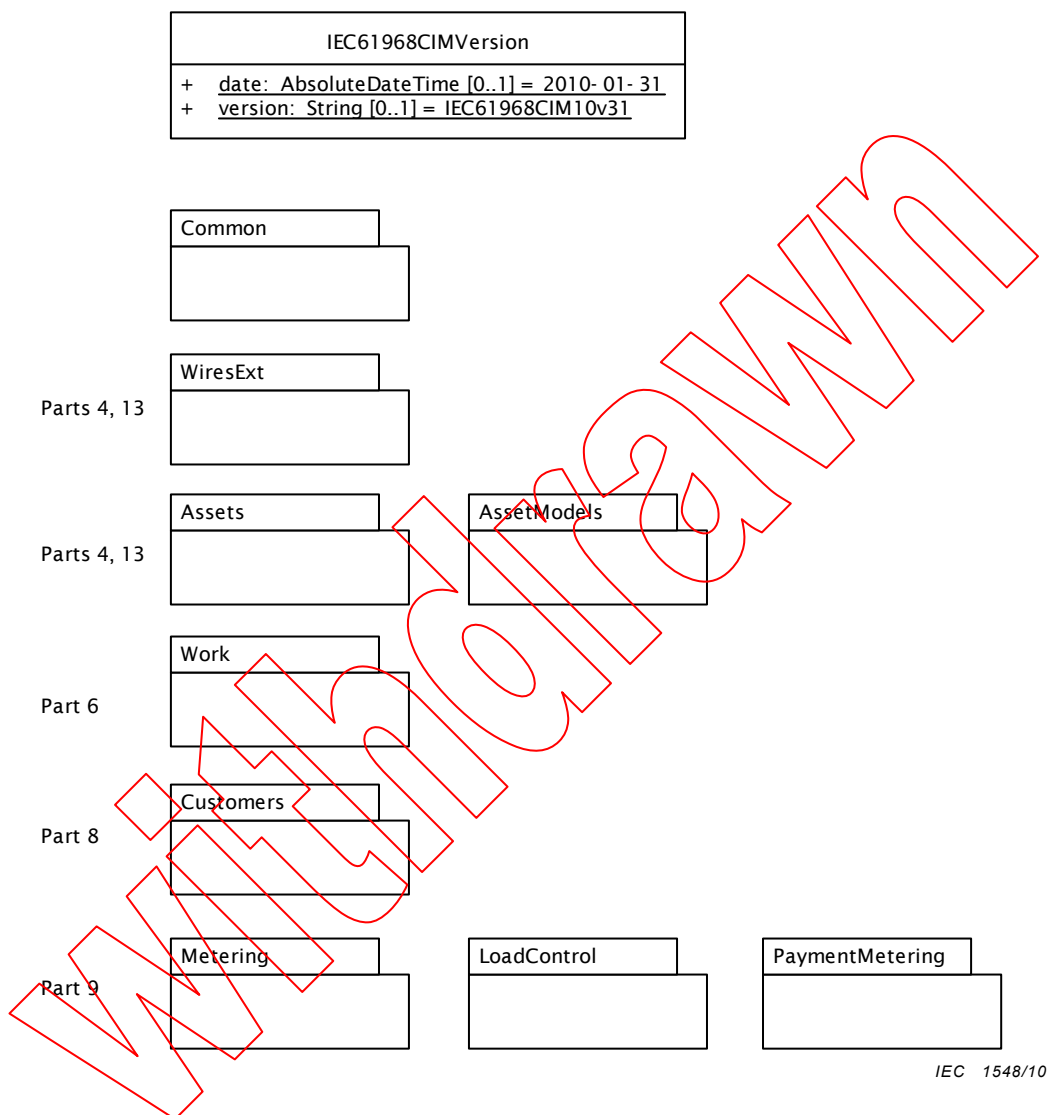


Figure 2 – CIM extensions for distribution (DCIM) top-level packages

Clause 6 contains the specification for each of the distribution CIM packages.

NOTE The contents of the CIM defined in this specification were auto-generated from the CIM UML electronic model release IEC61968CIM10v31.

4.3 CIM UML modelling

4.3.1 General

The CIM model is defined and maintained using UML. The source and point of maintenance for the CIM model is currently an Enterprise Architect (EA) file. This permits the model to be viewed and maintained graphically. The same tool is used to generate web pages which can be viewed over the internet. From the EA file, XMI file is also generated that can be used within tools such as CIMTool to generate context specific messages in XSD, RDF, or OWL

format for Parts 3 to 9 of IEC 61968, IEC 61968-13 and IEC 61968-14. Other tool sets may also be used.

The purpose of this subclause (4.3) is to define some principles with respect to the IEC 61968 information model and associated information exchanges. For description of different UML constructs used in the CIM model, refer to Subclause 4.3 of IEC 61970-301: CIM classes and relationships.

4.3.2 Scope of UML model

It is not the intent of this specification and associated models to define models which satisfy all information requirements, as this would be an impossible task. The standard model is a canonical data model for enterprise systems integration and thus needs to satisfy requirements for information exchanges among different systems, i.e., message payloads defined in Parts 3 to 9 of IEC 61968, IEC 61968-13 and IEC 61968-14. Custom extensions are the matter of non-standard projects and products. They can be used to leverage CIM for information models of specific applications or systems, or for non-standard integration needs. However, custom extensions are not maintained by IEC.

The overall DCIM model has been evolving during many years, but has never been published as an IEC standard. For this first edition, the UML model has been split into normative and informative classes and their relationships. Only normative classes, with their attributes and normative relationships, are fully documented in Clause 6. At the time of editing, the normative classes and relationships are those required for IEC 61968-9 and IEC 61968-13. They are considered stable and are expected to change little.

In contrast, informative classes and their informative relationships are not documented in this specification. They are present in the electronic UML model and will be promoted to normative classes stepwise, with the new editions of Parts 3 to 9 of IEC 61968 and IEC 61968-13. Some of those classes will be kept informative as long as they are considered unstable and likely to change, and others might be removed because they do not participate in standard information exchange.

NOTE The next anticipated set of classes that is to be promoted from informative to normative in DCIM 11 for Edition 2 of this IEC 61968-11 are those needed to support the maintenance cycle of IEC 61968-3, IEC 61968-4, IEC 61968-9 and IEC 61968-13.

4.3.3 Extensibility

It is fully the intent to permit extensions to the information exchange model. Extensions should utilize a local namespace. The namespace shall also serve to identify the origination of the class or property.

An extension can be defined in a UML model and/or a physical model such as XML schema:

- If an extension is made in UML model, such extension should be made in a different package as a separate model layer or context rather than be added or modified into CIM model directly. A “targetNamespace” tagged value may be used for this purpose if such extension is meant for an XSD generation.
- If an extension is made for XSD only, a target namespace has to be different than a CIM XSD namespace to identify the origin of the extension.

The extension namespace may follow a naming convention: `http://<organization>/<version control>/<profile>`.

4.3.4 Message (profile) definition

4.3.4.1 General

A profile is a subset of DCIM classes, associations and attributes needed to accomplish a specific type of interface. A profile defines the message payload from CIM model (IEC 61968-11 and IEC 61970-301) and the header format and information to exchange (specific to profile document IEC 61968-3 through IEC 61968-9 and IEC 61968-13).

One way of defining standard messages (profiles) is using applications such as the open source CIMTool. Other methods may be used, including hand coding.

Currently, two grammars of XML are used for profile definition in IEC 61968 series.

4.3.4.2 Using XML schema

Messages defined within Parts 3 to 9 of IEC 61968 use combinations of nouns and verbs. The nouns usually refer to the classes defined within the DCIM model.

The verb usually indicates which attributes are required. In the case of **CREATE/CREATED** and **SHOW** verbs, typically all attributes defined in a profile are required, while with **GET**, **CANCEL/CANCELLED**, **DELETE/DELETED** and **CLOSE/CLOSED** verbs only object identifiers are typically required. The **CHANGE/CHANGED** verb would require an object identifier and the values of attributes to be changed.

The verbs defined for use within IEC 61968 compliant interfaces are specified in IEC 61968-1.

4.3.4.3 Using RDF schema

IEC 61970-501 defines a subset of RDF schema that is used for bulk network model exchanges. The profile format and header information is specified in IEC 61970-552-4³⁾, and is currently used by the profiles defined in IEC 61968-13.

4.4 DCIM model concepts and examples

4.4.1 General

This subclause (4.4) describes some examples of modeling in the distribution domain with DCIM. They are complementary to the base CIM examples presented in IEC 61970-301.

4.4.2 Key classes in DCIM

Base CIM in IEC 61970-301 mainly defines the function of electrical network elements through **PowerSystemResource** class and its subclasses, for the needs of information exchange in the context of control centre applications and systems. DCIM in this IEC 61968-11 document adds some key classes to support (a) physical description of those network elements, as well as (b) information exchange related to network operations and planning in the context of the whole utility enterprise.

Figure 3 shows some key classes in the DCIM.

³⁾ Under consideration.

<i>IdentifiedObject</i> Core::PowerSystemResource	<i>IdentifiedObject</i> Assets::Asset	<i>IdentifiedObject</i> Common::Document	<i>IdentifiedObject</i> Common::Location	<i>IdentifiedObject</i> Common::Organisation	<i>IdentifiedObject</i> Common::ActivityRecord
--	--	---	---	---	---

IEC 1549/10

Figure 3 – DCIM key classes

An **Asset** is a tangible resource of the utility, including power system equipment, vehicles, tools, cabinets, buildings, etc. For electrical network equipment, the role of the asset is defined through the PowerSystemResource hierarchy, defined mainly in the wires model (refer to IEC 61970-301 and model package **IEC61970::Wires**, and model package **IEC61968::WiresExt** under Clause 6). Asset description places emphasis on the physical characteristics of the equipment fulfilling that role.

A **Document** is a grouping of information collected, often managed as a part of a business process. It will frequently contain references to other objects, such as assets, persons and power system resources.

A **Location** is the place, scene, or point of something where someone or something has been, is, and/or will be at a given moment in time. It is defined with one or more position points (coordinates) in a given coordinate system.

Organisations may have roles as utilities, contractors, suppliers, manufacturers, customers, tax authorities, etc.

ActivityRecord records activity for an entity at a point in time. Activity may be for an event that has already occurred or for a planned activity.

NOTE 1 Classes of the CIM are often used in many disparate contexts. For example, the same instance of organisation may be for a customer role in a context dealing with service requests and a manufacturer role in a different context dealing with the maintenance of a particular asset.

NOTE 2 Several of these key classes have relationship to the other classes (not shown in the figure), and in particular there are relationships among their subclasses. For example, an **Asset** plays a particular role in the electrical network as defined by the **PowerSystemResource** it is associated with. The **PowerSystemResource** may have a schematic location on a one-line diagram and a geographic location, whereas the **Asset** filling that role has a geographic location that a crew can service. If an **Asset** is removed from service, refurbished and then installed at another **Location**, history of the logical network and the physical assets may both be tracked by having instances of **ActivityRecord** associated with the **PowerSystemResource** as well as the **Asset**. The **PowerSystemResource** may be serving a particular **Organisation**, such as a supplier. This supplier is simultaneously a customer.

4.4.3 Single-phase and unbalanced loads

Figure 4 shows the classes available to model distribution loads, which are often unbalanced among the three phases at a location. In some cases, single-phase and two-phase loads will occur.

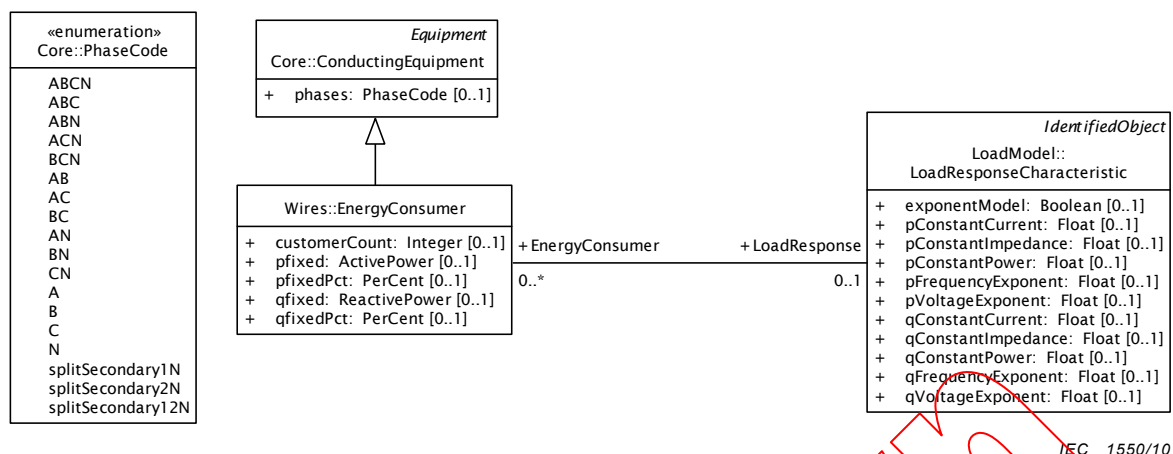


Figure 4 – DCIM load model

The **EnergyConsumer** class should be used to instantiate the load model, which can optionally be associated with a meter (through **ServiceDeliveryPoint**, not shown in the figure), but does not need to. **EnergyConsumer** inherits the **phases** attribute from **ConductingEquipment**, which may be assigned a **PhaseCode** enumeration literal. For example, **AN** describes a single-phase load from A to neutral, **BC** describes a single-phase load from B to C, **ABCN** describes a three-phase wye-grounded load, **ABC** describes a three-phase delta-connected load, etc.

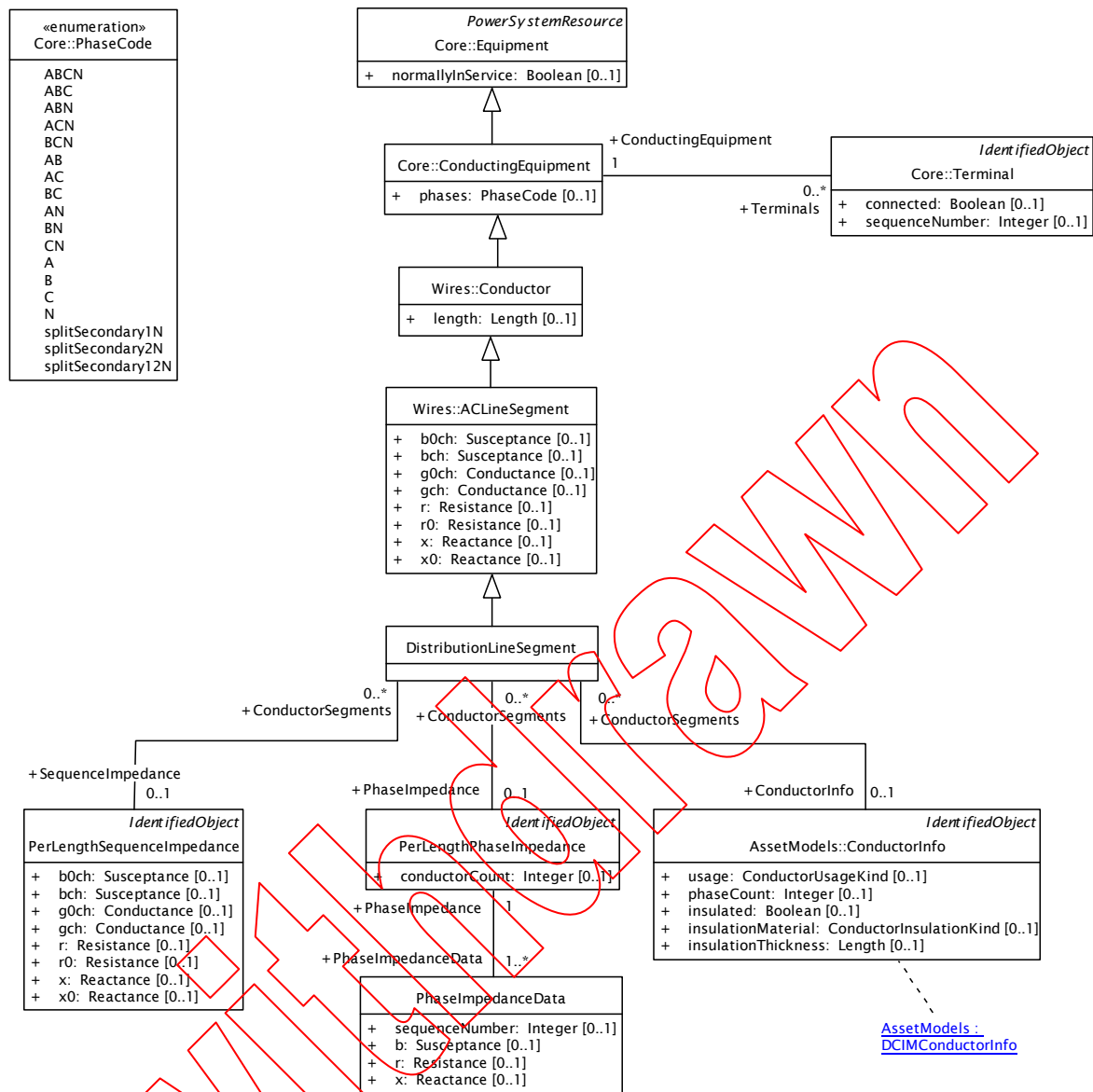
If a three-phase load is unbalanced, it can be modelled with three single-phase loads connected (through **Terminals**) to the same **ConnectivityNode**. With single-phase or two-phase loads, the model should also include conductors or transformers that establish connectivity back to the source.

The load model could be some combination of constant current, constant power, or constant impedance. In that case, an instance of **LoadResponseCharacteristic** should be associated with the **EnergyConsumer** instance.

4.4.4 Distribution line segments

Figure 5 shows the classes available to model distribution line segments (i.e., conductors). There are four ways to describe the impedance parameters of a **DistributionLineSegment**, as a function of the kind of model exchange and the data available in the exporting system.

NOTE This edition of IEC 61968-11 (documenting DCIM10) and the corresponding edition of IEC 61970-301 (documenting base CIM14) reflect separate line segment models for distribution and transmission (T&D), respectively. The next editions should present a consolidated T&D line segment model.



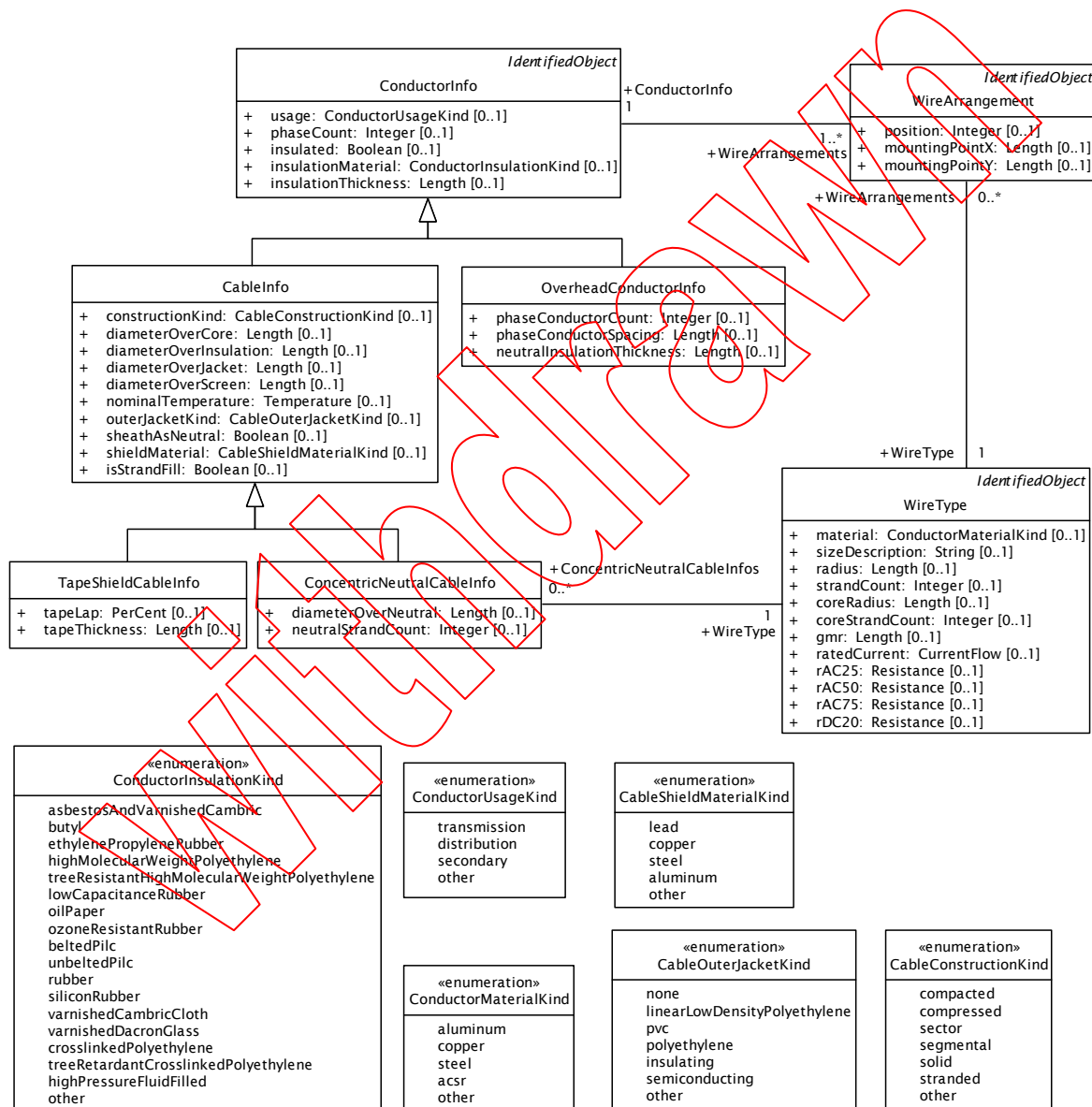
IEC 1551/10

Figure 5 – DCIM line connectivity model

- Provide an association to **ConductorInfo**, detailed in Figure 6, for calculation of impedance and admittance per unit length from Carson's equations using physical wire and geometry data. **Conductor.length** attribute is required, because the instance electrical parameters are defined as (per-unit length parameters) * (**Conductor.length**). See also 4.4.4.3 below.
- Provide an association to **PerLengthPhaseImpedance**, which references pre-calculated NxN symmetric impedance and admittance matrices per unit length. The **conductorCount** (N) has to be at least equal to the number of phases, but it could be higher if the neutral (or other grounded conductor) is retained in the matrix. **PhaseImpedanceData** implements Z and Y matrix elements, stored in column order. The attributes **r**, **x**, and **sequenceNumber** are all required, while **b** is optional. Only the lower triangular elements are stored, so that a 3x3 matrix would have 6 elements (i.e., instances of **PhaseImpedanceData**). Because the **PhaseCode** enumeration is not ordered, the matrix rows and columns have to be in phase order. For **phases=ABC**, row 1 is always phase **A**, row 2 is phase **B**, and row 3 is phase **C**. This might describe a "line code" with phase **A** on the left, **B** in the middle, and **C** on the right. If the phasing is **BCA**, from left to right, that would require a different instance of **PerLengthPhaseImpedance** because the matrices will differ. **Conductor.length** is required for the **DistributionLineSegment**. See also 4.4.4.2 below.

- Provide an association to **PerLengthSequenceImpedance**. This class implements a library of “line codes” that have sequence impedances and line charging per unit length. **Conductor.length** is required. See 4.4.4.2 for usage of sequence parameters with 1-phase and 2-phase line segments.
- The inherited **ACLineSegment** attributes may be used if none of the preceding three associations is provided. **Conductor.length** is optional, and not used in the impedance calculation.

Figure 6 shows the **ConductorInfo** class hierarchy, used for defining line segment impedances from physical data (sometimes referred to as “line codes”). **ConductorInfo** is a base class that should not be instantiated directly.



IEC 1552/10

Figure 6 – DCIM conductor (line and cable datasheet) model

OverheadConductorInfo identifies the line physical data. Any conductors numbered above the **ConductorInfo.phaseCount** are assumed to be continuously grounded; the application may eliminate these conductors from the impedance and admittance matrices through Kron reduction. The **phaseConductorCount** and **phaseConductorSpacing** attributes of **OverheadConductorInfo** refer to sub-conductor bundling, which is not common for distribution lines, but may appear on high-voltage transmission lines in the model.

also allows for triplex secondary lines, by specifying the **insulated**, **insulationMaterial**, **insulationThickness**, and **neutralInsulationThickness** attributes.

WireType defines a library of wires. The attributes **material**, **sizeDescription**, **strandCount**, and **coreStrandCount** (for ACSR, aluminium conductor steel reinforced wire) help to identify the wire, and are often coded into the instance local name. The minimum required electrical attributes are **gmr**, **radius**, and **rAC50**. A complete wire table would usually have resistances defined at other temperatures and frequencies, such as **rAC25**, **rAC75**, and **rDC20**. For ACSR wires, the **coreRadius** is optional for frequency-dependent calculation of the **gmr**. The **coreRadius** is zero for non-ACSR wires. The **ratedCurrent** is the ampacity at 50 °C. The **ratedCurrent** is necessary to interpret power flow output, but not for the power flow solution itself.

WireArrangement defines the horizontal (**mountingPointX**) and vertical (**mountingPointY**) coordinates of each wire on the pole cross section. All three attributes are required. The wire height above ground is **mountingPointY**, including any sag effects. The wire horizontal position, **mountingPointX**, is measured from an arbitrary but consistent reference line. Common choices for the horizontal reference are the pole centerline, and the left-most wire position. The **WireArrangement** class mediates a many-to-many relationship between **ConductorInfo** and **WireType**. This model allows a different **WireType** for the neutral, and for each phase.

Underground cables will use **WireArrangement** and **WireType** in the same way as overhead lines. The two concrete classes that may be instantiated are **TapeShieldCableInfo** and **ConcentricNeutralCableInfo**. For concentric neutral cables, the copper neutral wires are defined by an association to **WireType**, with number and positioning determined by the **diameterOverNeutral** and **neutralStrandCount** attributes.

4.4.4.1 Using sequence impedances (balanced case)

The positive and zero sequence impedances may be transferred through the **r**, **x**, **r0**, and **x0** attributes of **PerLengthSequenceImpedance** associated with a **DistributionLineSegment** instance. The **bch**, **b0ch**, **gch**, and **g0ch** attributes are not important for overhead distribution lines. For three phases, this describes a balanced three-phase, or perfectly transposed, line. The attributes in **PerLengthSequenceImpedance** are expressed in units per length, so it is necessary to multiply their values with the **length** attribute of **Conductor**.

NOTE This is equivalent to the attributes of **Wires::ACLineSegment**, which are pre-calculated for the whole length of the segment and thus must be defined on each instance of the segment. In contrast, **PerLengthSequenceImpedance** is referenceable, and as such can be used (through association) by several segment instances, thus decreasing the amount of data transferred in data exchanges.

If the **DistributionLineSegment** has only one or two phases, a balanced model can still be transferred through the **r**, **x**, **r0**, and **x0** attributes. This represents an impedance matrix with equal complex diagonal elements, Z_s , and equal complex off-diagonal elements, Z_m . For a single-phase line, the attributes to transfer are:

$$Z_1 = Z_0 = Z_s$$

For a two-phase or three-phase line, the attributes to transfer are:

$$Z_1 = Z_s - Z_m$$

$$Z_0 = Z_s + (n - 1) Z_m$$

where n is the number of phases. Upon receipt of **r**, **x**, **r0**, and **x0**, the balanced two-phase or three-phase impedance matrix is constructed from:

$$Z_s = (Z_0 + (n - 1) Z_1) / n$$

$$Z_m = (Z_0 - Z_1) / n$$

The inherited **phases** attribute of **DistributionLineSegment** should be assigned an appropriate **PhaseCode** enumeration literal to show the phases actually present, such as **A**, **B**, **C**, **AB**, **BC**, **AC**, or **ABC**. The neutral, **N**, should not appear because any neutral conductor must have been incorporated into the earth return when sequence impedances are used.

For underground distribution cables, the sequence impedances are also appropriate, including the **bch** and **b0ch** attributes.

4.4.4.2 Using phase impedances (unbalanced case)

Calculated matrix parameters may be transferred by referencing a **PerLengthPhaseImpedance** instance, in lieu of the physical model described in 4.4.4.3. A matrix model is useful when:

- the target application has no means of calculating parameters from the physical data;
- the underlying physical data is not readily available;
- it is necessary to match the unbalanced line parameters as closely as possible.

For a two-phase line, with the neutral already reduced, the Z and Y matrices will be 2x2, but due to symmetry, there will only be 3 unique matrix elements. That leads to three associated instances of **PhaseImpedanceData** with column-wise storage:

- **sequenceNumber** = 1 for row 1, column 1 of the matrix;
- **sequenceNumber** = 2 for row 2, column 1 of the matrix;
- **sequenceNumber** = 3 for row 2, column 2 of the matrix.

This instance of **PerLengthPhaseImpedance** could be referenced from a **DistributionLineSegment** having phases **AB**, **AC**, or **BC**. Row 1 always corresponds to the first phase, which is **A**, **B**, or **C**, respectively. If the phase wires are physically transposed, such that **C** (for example) should be in row 1, a new instance of **PerLengthPhaseImpedance** is required.

4.4.4.3 Using physical parameters

For overhead lines, a physical model can be transferred through reference to an **OverheadConductorInfo** instance, which is associated with further classes shown in Figure 6. This will support calculation of an unbalanced phase impedance matrix through the use of Carson's equations, or an equivalent method of handling the earth return. For example, suppose there are three phase wires, plus a different size neutral wire, on a pole with horizontal crossarm. This requires one **OverheadConductorInfo** instance, four **WireArrangement** instances that describe the four conductor positions, and two **WireType** instances describing the phase and neutral wire types. The **length** attribute of **Conductor** must be used, and many **DistributionLineSegments** will typically refer to the same **OverheadConductorInfo** instance.

The **WireArrangement** instances link **WireTypes** and **OverheadConductorInfos**, in addition to defining the mounting points. These **WireArrangements** must be sequenced in order to identify the phase wire assignments. The resistance attribute of **WireType** should be supplied for power frequency, and at the wire's desired operating temperature for calculations.

A single-phase, concentric neutral cable requires one instance of **ConcentricNeutralCableInfo**, one **WireArrangement** instance to specify the burial depth, and one **WireType** for the core conductor. A second **WireType** specifies the neutral strand; this is associated directly to **ConcentricNeutralCableInfo** rather than through a **WireArrangement**. A three-phase type

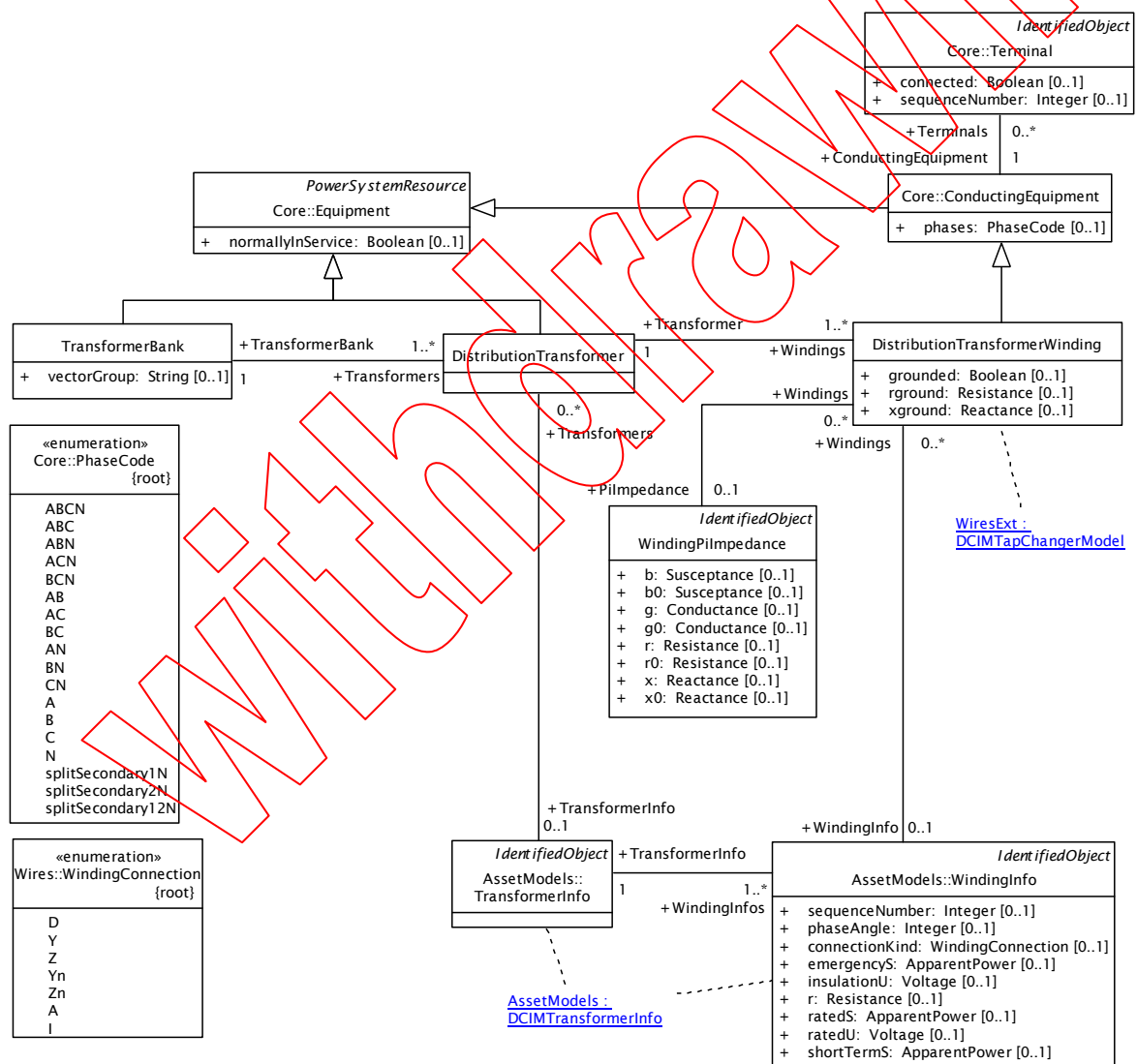
shielded cable, with bare neutral conductor, would require one instance of **TapeShieldCableInfo** and four **WireArrangements** for the three phases and neutral. There would also be two **WireTypes**, one for the phase core conductor and another for the neutral.

4.4.5 Distribution transformers

4.4.5.1 Electrical model

Figure 7 shows the classes that model distribution transformer instances. They rely exclusively on a library of transformer types, detailed in Figure 8, to obtain most of the transformer parameters.

NOTE This edition of IEC 61968-11 (documenting DCIM10) and the corresponding edition of IEC 61970-301 (documenting base CIM14) reflect separate transformer models for distribution and transmission (T&D), respectively. The next editions should present a consolidated T&D transformer model.



IEC 1553/10

Figure 7 – DCIM transformer connectivity model

TransformerBank organizes the component three-phase and/or single-phase transformers. Some banks contain just one three-phase transformer, while others contain two or more single-phase transformers. At the transmission level, EHV transformer banks may also contain single-phase transformers, which need not be identical, especially when a spare is in

service. This class is equivalent to the **PowerTransformer** defined in IEC 61970-301, and it inherits from **Equipment**. Locations are often associated to the bank. The **vectorGroup** attribute for protective relaying is derived from the internal winding connections and phase angles; it uses standard nomenclature to describe any number of windings that may be included in the bank.

DistributionTransformer must reside in a transformer bank, and it organizes a flexible number of transformer windings. It references the **TransformerInfo** class in Figure 8 to obtain most of the data.

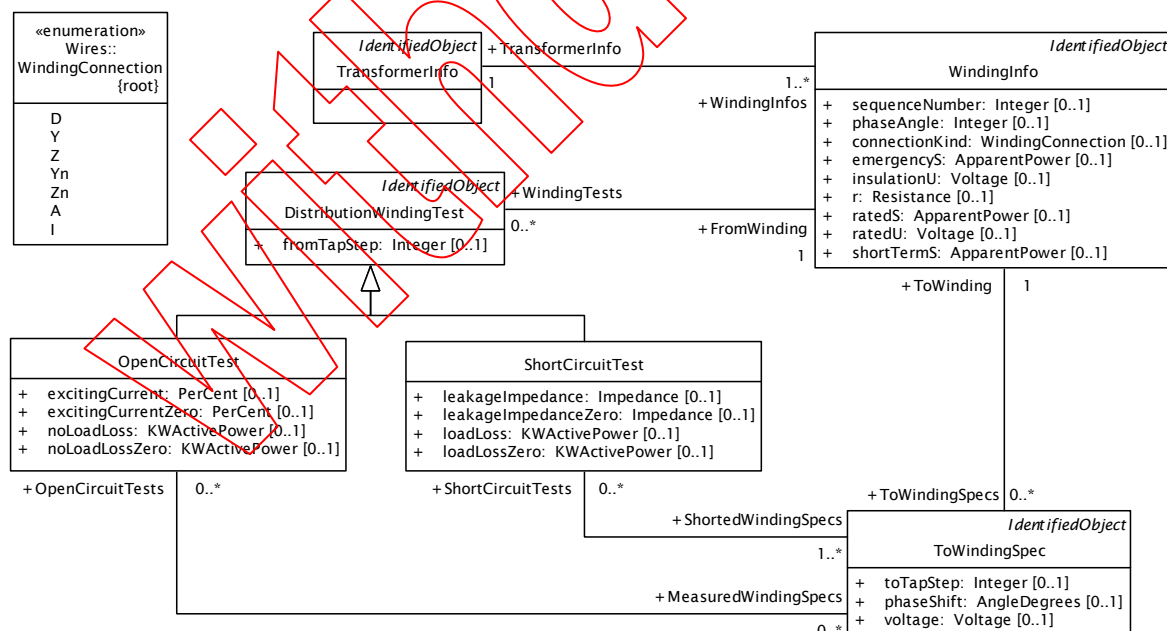
DistributionTransformerWinding is a **ConductingEquipment**, with phasing information and associated **Terminals**. It references the **WindingInfo** class in Figure 8 to obtain data on the winding ratings. In addition to **ConductingEquipment.phases**, the other instance attributes define the grounding options:

- solidly grounded: **grounded** = true, **rground** = 0, **xground** = 0;
- impedance grounded: **grounded** = true, **rground** ≥ 0, **xground** ≥ 0,
- ungrounded: **grounded** = false.

The **WindingPILimpedance** class allows for compatibility with IEC 61970-301 transformer models.

4.4.5.2 Physical model

Figure 8 shows the classes that allow for exchange of transformer physical data, sometimes referred to as “transformer codes” in applications.



IEC 1554/10

Figure 8 – DCIM transformer datasheet model

WindingConnection enumeration includes the standard **D**, **Y**, **Z**, **Yn**, and **Zn** nomenclature to describe delta, wye, zig-zag, and neutral connections in three-phase transformer vector groups. **A** is used for a common autotransformer winding, and **I** for a single-phase transformer winding.

TransformerInfo is referenced by the **DistributionTransformer**. It defines most of the transformer data. In the model at present, this class has no attributes of its own, and serves only to organize the winding attributes into a library. **TransformerInfo** collects associations to **WindingInfo**, such that the **WindingType** enumeration is not used. This generalizes to any number of windings.

WindingInfo includes all of the winding rating information, including **ratedU**, **ratedS**, **connectionKind**, **phaseAngle**, **shortTermS**, **emergencyS**, and **insulationU**. **r** is the winding's DC resistance. **sequenceNumber** is the winding's order in the **vectorGroup**; the high-voltage winding is always 1. The **sequenceNumber** increases as the **ratedU** decreases, but some transformers have two windings with the same **ratedU**, in which case the **sequenceNumber** distinguishes between them.

DistributionWindingTest with its attributes and subclasses covers short-circuit and open-circuit tests, for either single-phase or three-phase transformers, and in the case of three-phase transformers, either positive-sequence or zero-sequence excitation. For both open-circuit and short-circuit tests, only one winding ("from") is excited. The "from" winding is associated to the parent **DistributionWindingTest** class. For short-circuit tests, one or more other ("to") windings will be short-circuited. For open-circuit tests, there may be voltage and angle data on the other "to" windings.

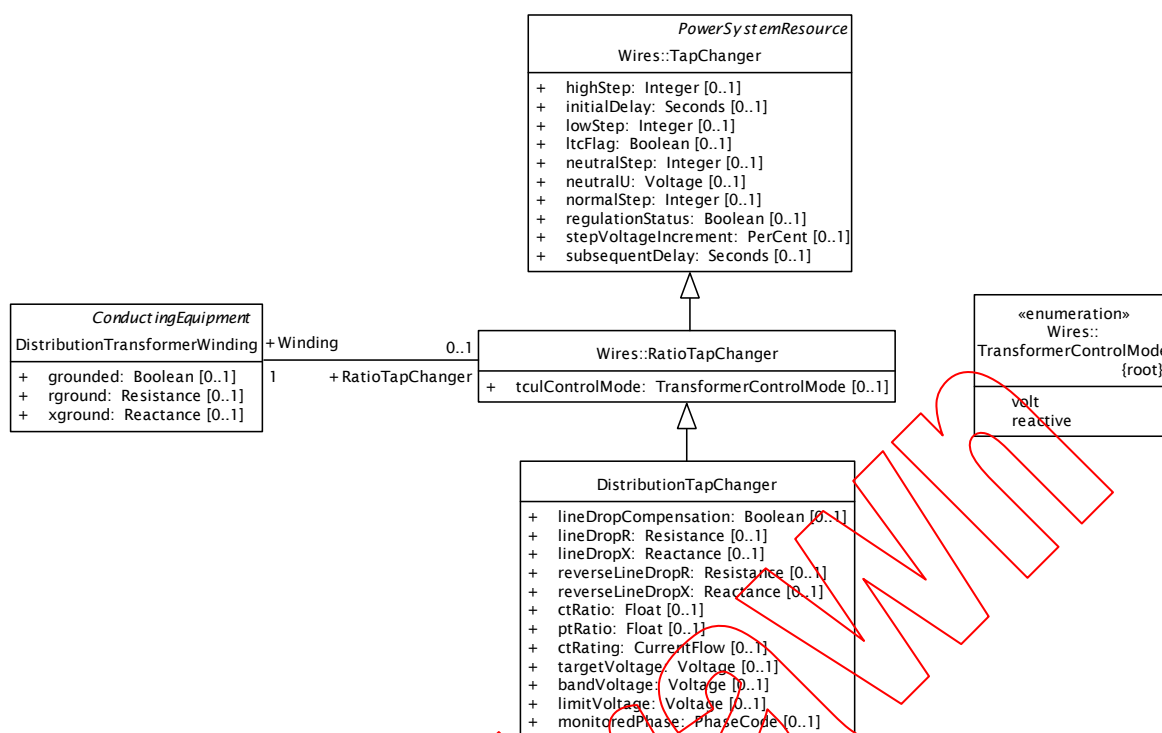
OpenCircuitTest's attributes **excitingCurrent**, **excitingCurrentZero**, **noLoadLoss**, and **noLoadLossZero** are all measured on the excited ("from") winding. Positive and zero sequence test data can be reported in the same instance of **OpenCircuitTest**. This test is generally done with rated voltage applied to the excited winding. The test may include measurements of induced voltage and angle on other windings; these values would be reported in associated instances of **ToWindingSpec**.

ShortCircuitTest is done by circulating rated current through the "from" winding, with one or more "to" windings short circuited. Positive and zero sequence test data can be reported in the same instance of **ShortCircuitTest**. The AC resistances derived from **loadLoss** and **loadLossZero** are likely to differ from the winding DC resistances, **WindingInfo.r**, which are obtained from a separate test.

ToWindingSpec class defines the windings short circuited for a given **ShortCircuitTest**, and only the **toTapStep** attribute applies. For an **OpenCircuitTest**, the **voltage** and **phaseShift** attributes contain measured voltage and angle on the associated "to" winding. Instances of this class must be associated to either an **OpenCircuitTest** or a **ShortCircuitTest**, but not both.

4.4.5.3 Tap changer model

Figure 9 shows the classes used to model an unbalanced voltage regulator. A **DistributionTapChanger**, which inherits from **RatioTapChanger**, is associated to a **DistributionTransformerWinding**. Each regulator uses autonomous local control, so that **RegulationSchedule** and **TapSchedule** (present in IEC 61970-301 for power transformers) are not used. Some important attributes of those classes were copied to **DistributionTapChanger**. Phase angle regulators and variation curves are also not generally used on distribution systems.



IEC 1555/10

Figure 9 – DCIM tap changer model

A three-phase line voltage regulator usually has three independent regulators to help correct voltage unbalance. The regulators are usually connected in wye. The model starts with a **TransformerBank** containing three **DistributionTransformers**, and a total of six **DistributionTransformerWindings**. There will be three instances of **DistributionTapChanger**, each associated to a different **DistributionTransformerWinding**. The **monitoredPhase** attribute of **DistributionTapChanger** should match the phase connection of its associated winding. The tap positions, and sometimes the other attributes, will not be the same in each phase of the regulator. An open-delta regulator is also fairly common; this consists of two single-phase regulators connected line-to-line in a bank, with partial capability to correct voltage unbalance.

A three-phase substation voltage regulator usually changes all three taps together, with no ability to correct voltage unbalance. This could be modelled with a bank of one three-phase transformer and a total of two three-phase **DistributionTransformerWindings**. There would be just one **DistributionTapChanger** associated to one winding. The **monitoredPhase** attribute of **DistributionTapChanger** can be **A**, **B**, or **C** if the potential transformer is connected line-to-ground. It can also be **AB**, **AC**, or **BC** for line-to-line potential transformers. Typically, only one potential transformer controls the regulator.

4.4.5.4 Example distribution transformer

The transformer in Figure 10 shows an open wye/open delta bank, which is used to supply inexpensive three-phase service to smaller customers.

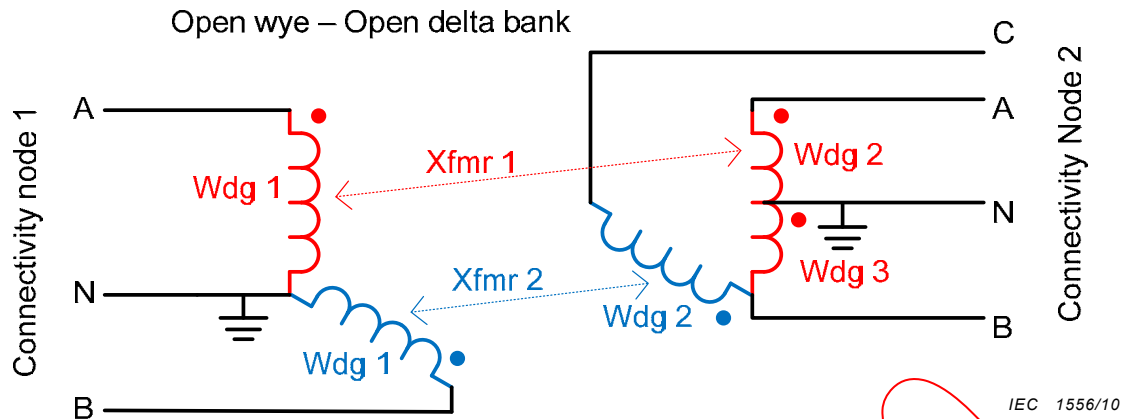


Figure 10 – Example of a distribution transformer that can be modelled with DCIM

Table 1 shows some of the important attribute values for this example.

Table 1 – Open wye/open delta transformer bank connections

Distribution Transformer	Distribution TransformerWinding	ratedU	ratedS	connection Type	phaseAngle	phaseCode
Xfmr 1	Wdg 1	7200	100e3	I	0	AN
	Wdg 2	120	50e3	I	0	AN
	Wdg 3	120	50e3	I	6	BN
Xfmr 2	Wdg 1	7200	50e3	I	0	BN
	Wdg 2	240	50e3	I	0	BC

A phase shift “6” indicates that Wdg 3 is actually from N to B, rather than B to N. Through the **Terminals**, **ConnectivityNode 1** will have phases **ABN** present from this transformer bank. Other connected equipment, such as a line segment, could add phase **C**. **ConnectivityNode 2** will have phases **ABCN** present from this transformer bank. The “lighting leg” (Xfmr 1) usually has a different rating than the “power leg” (Xfmr 2). This means that phase and rating assignments to the bank might be ambiguous, and thus need to be specified on **DistributionTransformerWinding**.

4.4.6 Connectivity with unbalanced phases

4.4.6.1 General

The distribution model connectivity follows the CIM connectivity model, i.e., it relies on **ConductingEquipment**, **Terminal**, and **ConnectivityNode** classes (refer to IEC 61970-301, Subclause 4.4.2 Connectivity model). **ConductingEquipment** has a **phase** attribute to describe the phase information of conducting equipment. **ConnectivityNodes** are points where **Terminals** of **ConductingEquipment** are connected together with zero impedance. It does not have phase information. Relationship between a **ConnectivityNode** and its container is modelled through **ConnectivityNodeContainer** class.

NOTE The presented CIM connectivity and phasing model supports the exchange of an as-built network model. However, in distribution network operations there are cases where a **ConductingEquipment** has different state for each **phase**. For example, a three phase switch might be operated differently on each phase so that phase **A** is electrically disconnected and phases **BC** are electrically connected; as a result, the line that is connected to the switch may have different electrical state on each phase as well. Currently the CIM measurement model can not represent such cases since no phase information is provided on **Measurement**. That capability is much needed for distribution operation and control and may be present in the next edition of this IEC 61968-11 in support of IEC 61968-3.

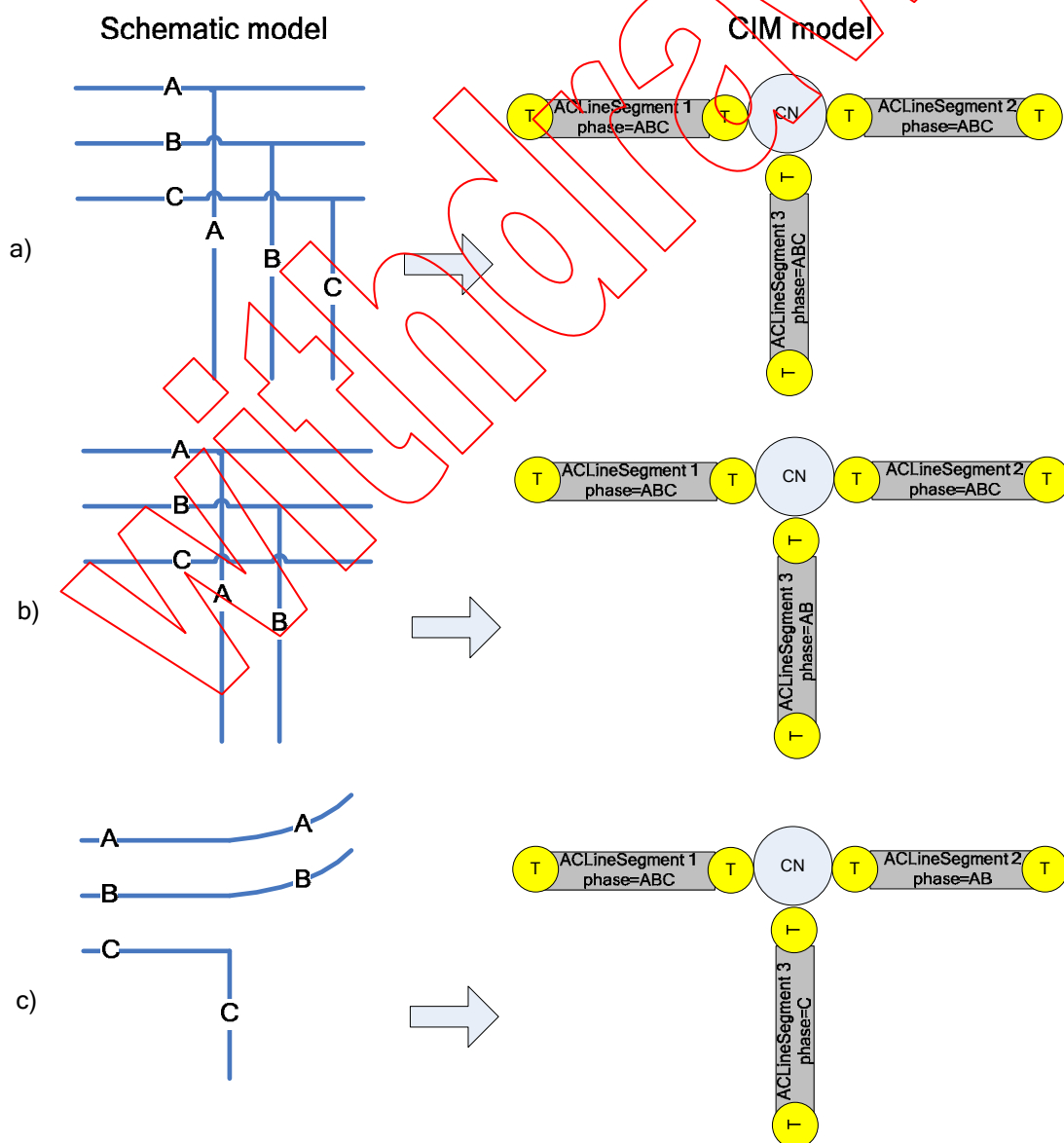
For an extensive example of balanced three-phase sample network, refer to IEC 61970-301, Subclause 4.4.2.1 Connectivity and containment example. The examples provided in the next subclause illustrate the usage of connectivity model for unbalanced phase cases.

4.4.6.2 Connectivity examples for unbalanced phases

Generally speaking, a **ConnectivityNode** is created as the point where two or more devices are connected together. The devices, which are modelled as subtypes of **ConductingEquipment**, carry the phasing information through the **phase** attribute of **ConductingEquipment**.

Consider as example **ACLineSegment** or its subtype **DistributionLineSegment**, which are used to model electrical connectivity and characteristics (impedances) of overhead lines and cables. The physical line or cable needs to be electrically modelled in such a way that an **ACLineSegment** or **DistributionLineSegment** have one and only one **phase** attribute.

Figure 11 illustrates three examples with different phase configurations that are common in distribution networks.



IEC 1557/10

Figure 11 – DCIM connectivity for two-Terminal devices

Similar connectivity modelling is to be applied for any other CIM two-**Terminal** devices such as series shunts and switches.

Figure 12 illustrates two examples for modelling loads: balanced and unbalanced.

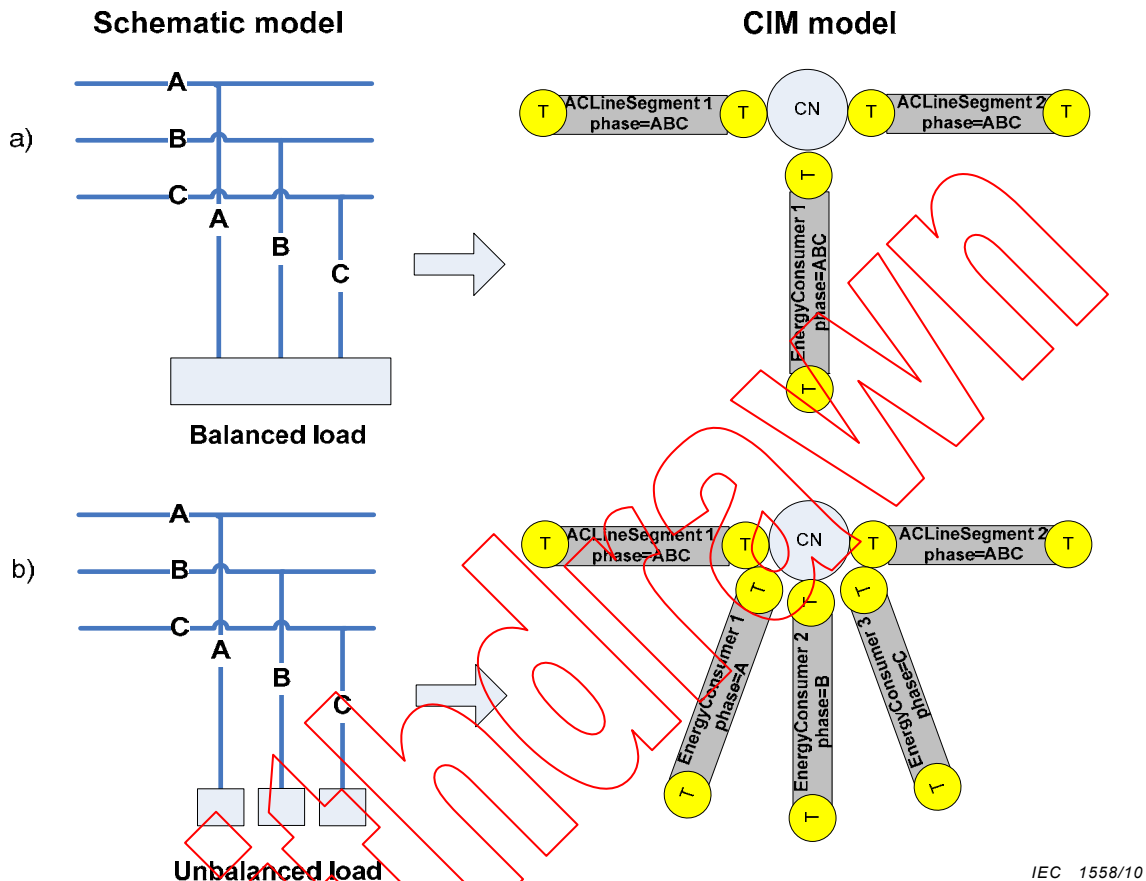


Figure 12 – DCIM connectivity for single-Terminal devices

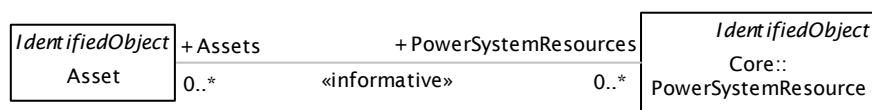
Similar connectivity modelling is to be applied for any other CIM single-**Terminal** devices such as shunts, generators and transformer windings.

NOTE This edition of IEC 61968-11 does not support multi-state switches (modelled by **CompositeSwitch**) and does not provide any grouping concept for a switch bank (where phases can have different status), or for loads (with potentially different load characteristics).

4.4.7 Electrical model vs. physical model

The distribution CIM covers both electrical and physical representation of an object. The **PowerSystemResource** class models the electrical representation and is often used for network operation, monitoring, outage management, and operations planning, while the **Asset** class models an object's physical representation and is mainly used for asset and work management. The physical representation may also be essential in deriving attributes for the electrical representation, even if no explicit **Asset** class is used. Subclause 4.4.4 explains how the asset model can help calculate the electrical characteristic of a distribution line. The relationship between the two aspects also provides key information for extension planning, work management and outage management.

Figure 13 illustrates how **Asset** and **PowerSystemResource** are related in CIM for the purpose of some data exchange scenarios.



IEC 1559/10

Figure 13 – DCIM assets and relation to power system resources

The relationship between **Asset** and **PowerSystemResource** allows for navigation between the two different representations (models) of a real world object. Connectivity and operational aspects (such as ratings, energisation status) are always modelled through **PowerSystemResource** and its subclasses, while physical aspects (such as data sheet ratings, dimensions, asset lifecycle) are always modelled through **Asset** and its related classes.

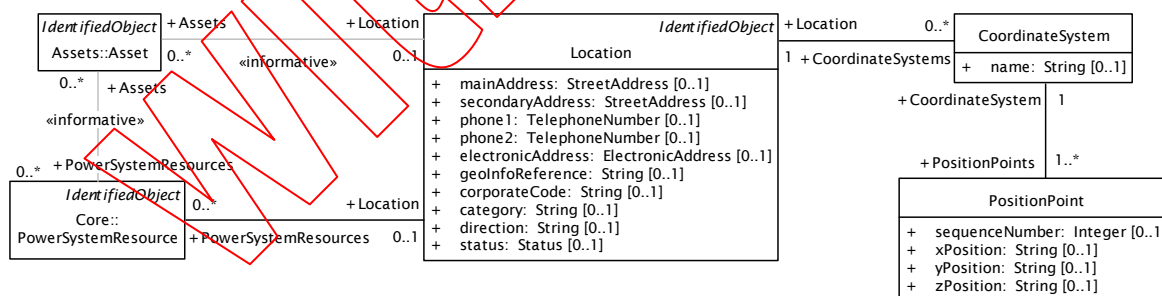
NOTE This edition of IEC 61968-11 contains **Asset** and **AssetModel** classes because they are used in the context of metering (IEC 61968-9), but not in the context of distribution network model exchange (IEC 61968-13). The limited set of assets-related classes in this edition contains those classes that support conductor and transformer modelling only, and the relationship **Asset-PowerSystemResource** is not used, thus tagged as informative. The next edition of this IEC 61968-11 should contain more complete asset model for the needs of distribution network model exchanges.

4.4.8 Locations and graphical representations

To allow for locating different entities in space, DCIM provides the class **Location**. Entities can be located by their different kinds of addresses as well as by position coordinates, **PositionPoint** in a given **CoordinateSystem**.

Location class is void of any graphical data exchange, although its **PositionPoints** can be used to exchange coordinates that may be used by graphical applications.

Figure 14 shows how **Location** class can be used in relation with **Assets** and **PowerSystemResources**.



IEC 1560/10

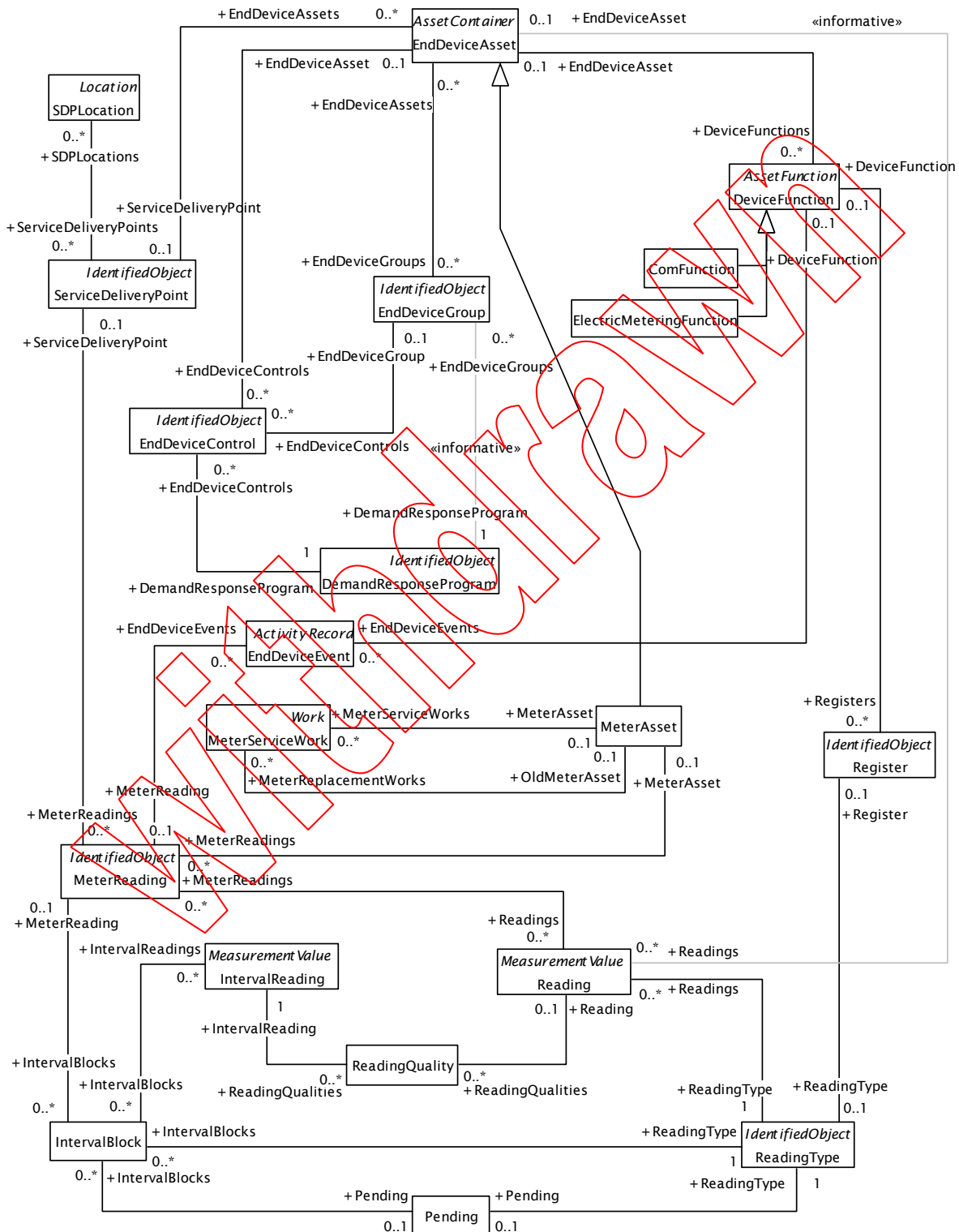
Figure 14 – DCIM power system resource and asset locations

NOTE Support for single line diagram exchanges, used in operations of transmission and some distribution control centres, is planned to be supported in base CIM, IEC 61970-301, and may be referred from a future edition of this IEC 61968-11. Support for geo-spatial data exchanges is out of the scope of this IEC 61968-11, since there are widely used and adopted standards for other domains that can be applied to electrical networks as well.

4.4.9 Metering

The **Metering** package introduces classes, shown in Figure 15 that are needed for the enterprise integration of metering systems and the information they exchange with other enterprise systems. A logical model for a meter is provided by **MeterAsset**, which inherits from **EndDeviceAsset**. **EndDeviceAssets** can detect and report **EndDeviceEvents**, report **MeterReadings** and accept **EndDeviceControls**. Within the model for meter readings, each

reading has a specific type and can be associated with an interval. **EndDeviceAsset** communication can be facilitated by individual addresses or group addresses. **EndDeviceAssets** are deployed at **ServiceDeliveryPoints**, providing the means to relate to locations and customers. A **MeterServiceWork** class is introduced to facilitate work related to the installation, maintenance and changeout of meters.



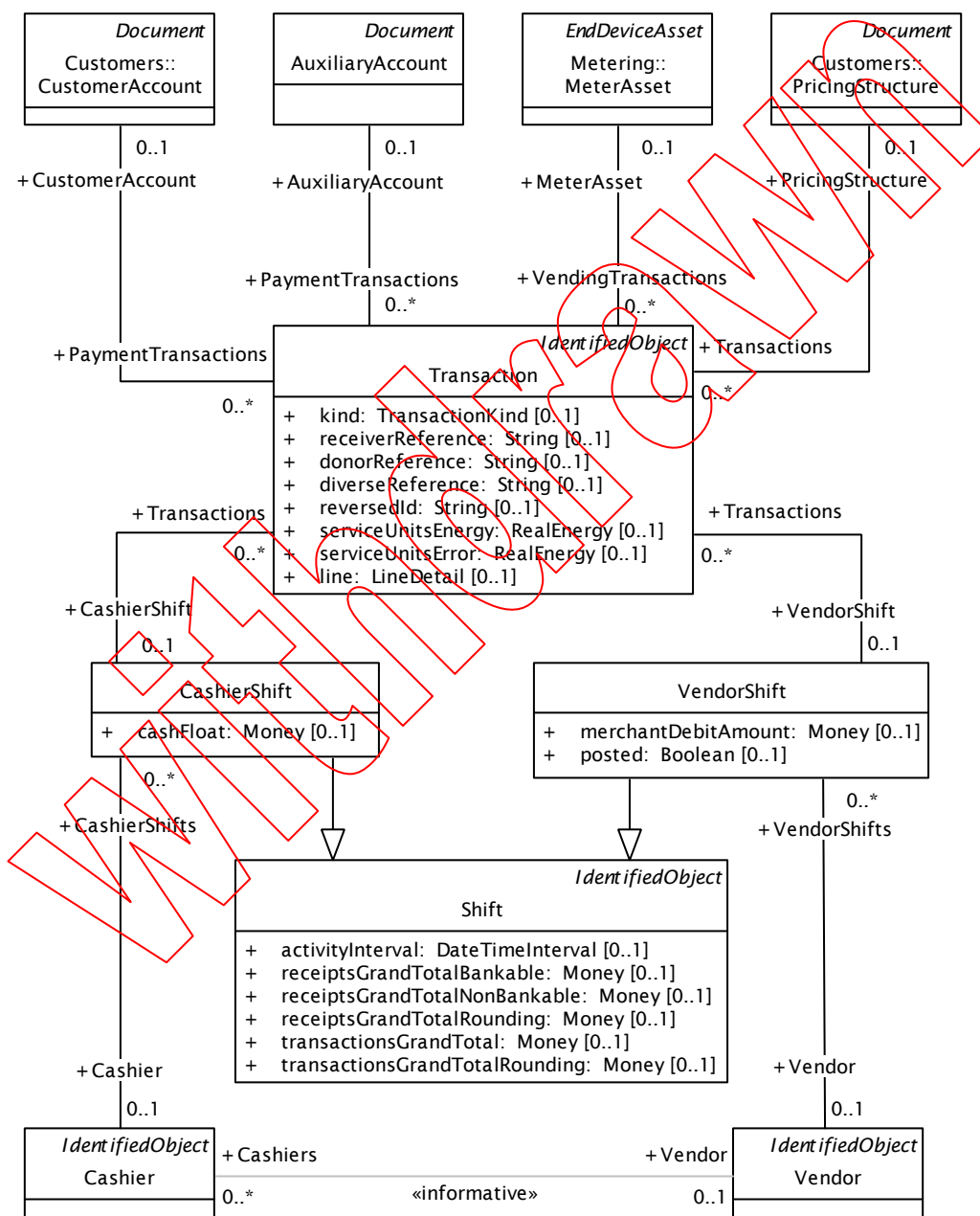
IEC 1561/10

Figure 15 – DCIM metering model

4.4.10 PaymentMetering

4.4.10.1 Transacting

A payment metering system generally facilitates financial transactions between a customer and a service provider. The relevant information describing these transactions are typically recorded in the payment system and this information is subsequently exchanged with another system such as the customer information or billing system. A typical example of realising such an information recording scheme using some of the CIM classes found in the **PaymentMetering** package is shown in Figure 16.



IEC 1562/10

Figure 16 – DCIM transacting model

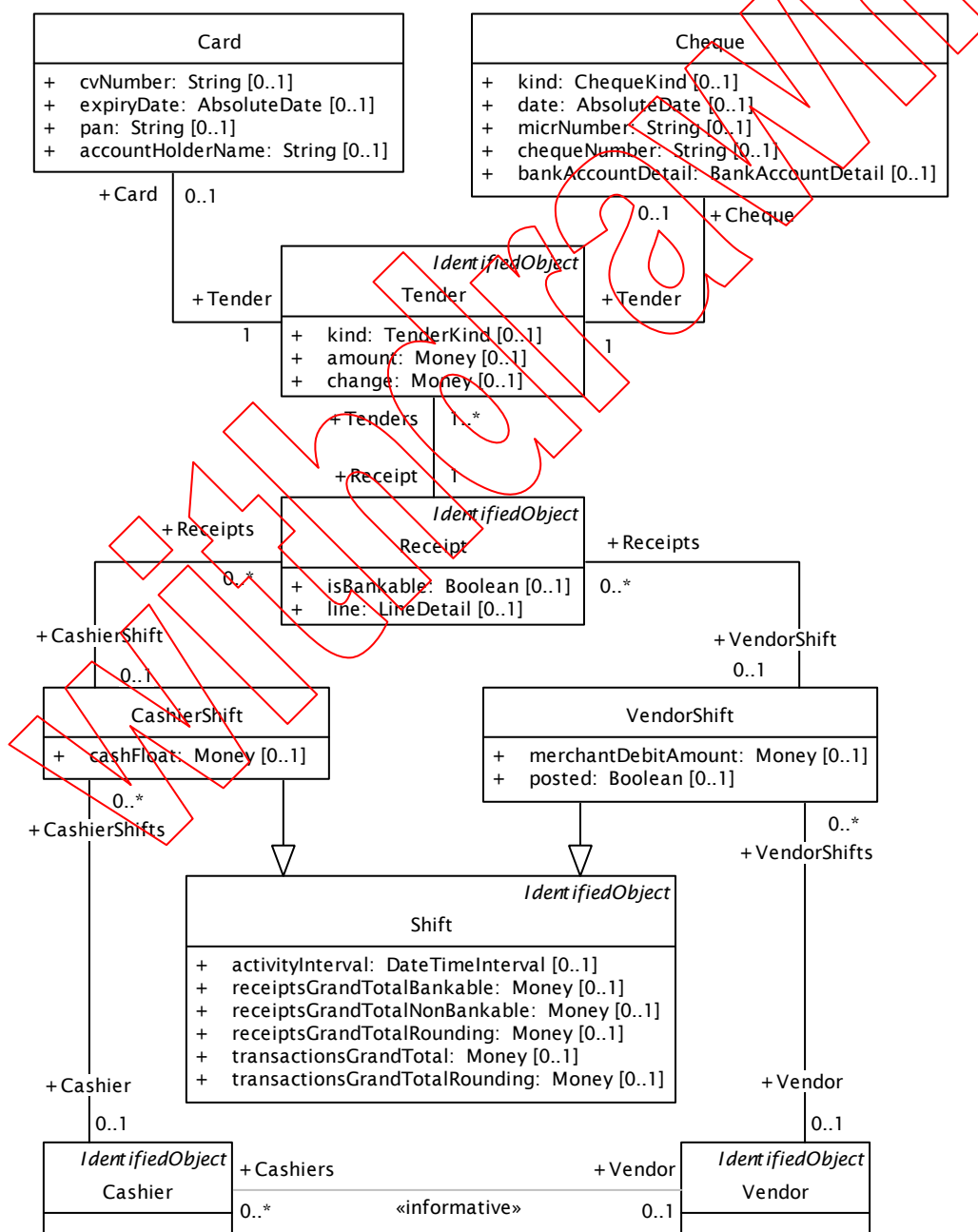
The core of information in this model is the **Transaction** class, which captures all the relevant information about the transaction and also includes extended information such as when a payment is made against a **CustomerAccount**, payment against an **AuxiliaryAccount**.

purchase of a prepaid **Token** for a prepayment service meter and the pricing that was used to calculate the amount charged for such a sale.

Transaction information may be further aggregated and sorted into **CashierShift** and **VendorShift** groupings for accounting and reconciliation purposes against a **Cashier** and **Vendor** who are accountable for the revenue collected during the particular **Transaction**.

4.4.10.2 Receipting

A transaction generally involves the receipt of revenue from the customer, which may take the form of cash, cheque or card for example. The capture and subsequent exchange of information describing the properties of this revenue may be realised by means of the model example shown in Figure 17.



IEC 1563/10

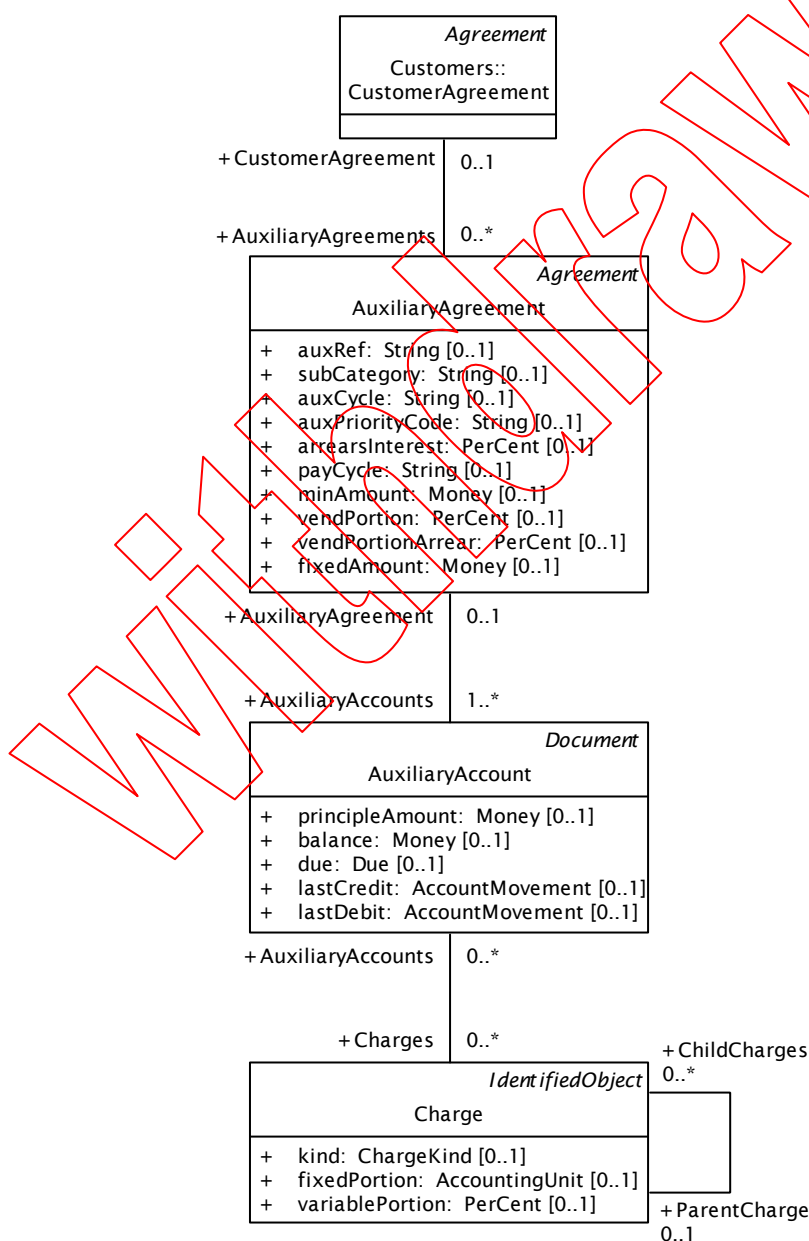
Figure 17 – DCIM receipting model

When a customer tenders payment during a transaction, the information is typically captured in the **Receipt**, **Tender**, **Card** and **Cheque** classes.

Receipt information may be further aggregated and sorted into **CashierShift** and **VendorShift** groupings for accounting and reconciliation purposes against a **Cashier** and **Vendor** who are accountable for the revenue collected during the particular **Transaction**.

4.4.10.3 Auxiliary payments

In addition to the typical payments made by customers for services provided by the service provider such as a utility, it is often required to receipt payments for other items such as debt, rates, taxes, municipal fines, TV licences, garbage collection charges, etc. The collection of such revenue may be integrated with token sales and customer account payments by means of auxiliary agreements and auxiliary accounts, an example of which is shown in Figure 18.

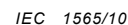


IEC 1564/10

Figure 18 – DCIM auxiliary agreement model

Charge allows for nested structures of charges to be levied against the **CustomerAccount** in accordance with the rules set in **AuxiliaryAgreement** and allows for fixed charges, variable charges and percentage charges.

Pricing structures may contain tariffs, which are often quite complex in structure and operation. Most tariffs levy charges that are time-based or consumption-based, both of which are interval-based. A model to realise such complex tariff structures is shown in Figure 19.



TariffProfile determines the cycle of operation for the **Tariff**, such as hourly, daily, weekly, monthly, etc, at the end of which it resets to start at the beginning of the process again.

TimeTariffInterval determines the starting time for a particular interval and several instances of **TimeTariffInterval** may be used to construct a series of time intervals to realise a time of use tariff for example.

Alternatively **ConsumptionTimeInterval** determines the starting value of a consumption interval and several instances of **ConsumptionTimeInterval** may be used to construct a series of consumption intervals to realise a block tariff or a step tariff for example.

The price per service unit per time interval or per consumption interval is determined by **Charge** class, which provides for nested charge structures and allows for fixed charges, variable charges and percentage charges.

For very complex tariff structures, the **TimeTariffInterval** and **ConsumptionTimeInterval** may be combined to provide for time-based and consumption-based charges simultaneously.

4.5 Other

Subclauses 4.5 to 4.8 of IEC 61970-301 describe CIM modelling tools, CIM extensions, and implementation conventions. The following points have changed since the last published edition 1.0 in 2003.

- Enterprise Architect is now used, instead of Rational Rose, to maintain the UML that defines the DCIM. The most current Enterprise Architect file (*.eap) provides the most current DCIM documentation.
- CIMTool provides a method of extending the DCIM and generating profiles, but other methods may be used as well.
- The **Naming** class is now the **IdentifiedObject** class. Interoperability testing has proven the need to maintain unique and persistent object identifiers across model domains. In practice, one way of achieving this would be the usage of universally unique identifiers (UUIDs) for the **mRID** attribute of **IdentifiedObject**. However, this is not the requirement.
- Older versions of CIM UML use three standard UML stereotypes: **Primitive**, enumeration and **Datatype**. The latter one is however used with a specific CIM semantics (not the standard UML semantics): for a triple of attributes {value, unit, multiplier}, which implies custom mapping to serialisation artefacts (RDFS, OWL, XSD). Therefore, a new custom UML stereotype, **Compound**, has been introduced to “fulfil” the semantics of standard UML datatype: group of values without identity. Classes with this stereotype never participate in relationships (generalisation, association), but are simply used as types for attributes.

5 Detailed model

5.1 Overview

The common information model (CIM) represents a comprehensive logical view of information exchanged among different systems in electrical utilities. This definition includes the public classes and attributes, as well as the relationships between them.

5.2 Context

The CIM is partitioned into subpackages. Classes within the packages are listed alphabetically. Native class attributes are listed first, followed by inherited attributes. Native associations are listed first for each class, followed by inherited associations. The associations are described according to the role of each class participating in the association.

Figure 1 shows that distribution CIM (this document) depends on base CIM (IEC 61970-301). This document includes the detailed description of the contents of **IEC61968** package only, and references several classes, attributes and association ends included in **IEC61970** package.

For each package, the model information for each class is fully described. Attribute and association end information for native and inherited attributes is documented as in Table 2 and Table 3 . For any inherited attributes or association ends, the “note” column will contain text indicating the attributes is inherited from a specific class. The note column for native attributes and association ends contains the actual description.

Table 2 – Attribute documentation

name	type	description
native1	Float	A floating point native attribute of the class is described here.
native2	ActivePower	Documentation for another native attribute of type ActivePower.
name	String	inherited from: IdentifiedObject

In the Attributes table, in some cases, an attribute is a constant, in which case the phrase “(const)” is added in the name column of the attributes table. In such cases the attribute normally has an initial value also which is preceded by an equal sign and appended to the attribute name.

Table 3 – Association ends documentation

[mult from]	[mult to] name	type	description
[0..*]	[0..*] OperatedBy_Companies	Company	inherited from: PowerSystemResource
[1]	[0..*] Contains_Measurements	Measurement	inherited from: PowerSystemResource
[1]	[0..*] OperatingShare	OperatingShare	inherited from: PowerSystemResource
[1..*]	[0..1] ModelingAuthoritySet	ModelingAuthoritySet	inherited from: IdentifiedObject

In the association ends table, the first column describes the multiplicity at this end of the association (i.e., how this class participates in association). Second column describes the other association end. Its multiplicity is included in brackets. The association end name is listed in plain text. The class at the other end of the association is in the third column. A multiplicity of zero indicates an optional association. A multiplicity of “*” indicates any number is allowed. For example, a multiplicity of [1..*] indicates a range from 1 to any larger number is allowed.

In the case that a class is an enumeration, the attributes table is replaced by the enums documentation as in Table 4 , since the type of each enum within the enumeration is not defined.

Table 4 – Enums documentation

literal	description
steel	
lead	
lock	
other	

6 Package architecture (normative)

6.1 General

This specification is automatically generated from the CIM model file iec61970cim14v13_iec61968cim10v31_combined.eap.

6.2 Top package IEC61968

The IEC 61968 subpackages of the CIM are developed, standardized and maintained by IEC TC57 Working Group 14: System interfaces for distribution management (WG14).

Currently, normative parts of the model support the needs of information exchange defined in IEC 61968-9 and in IEC 61968-13.

Figure 20 shows package diagram Main.

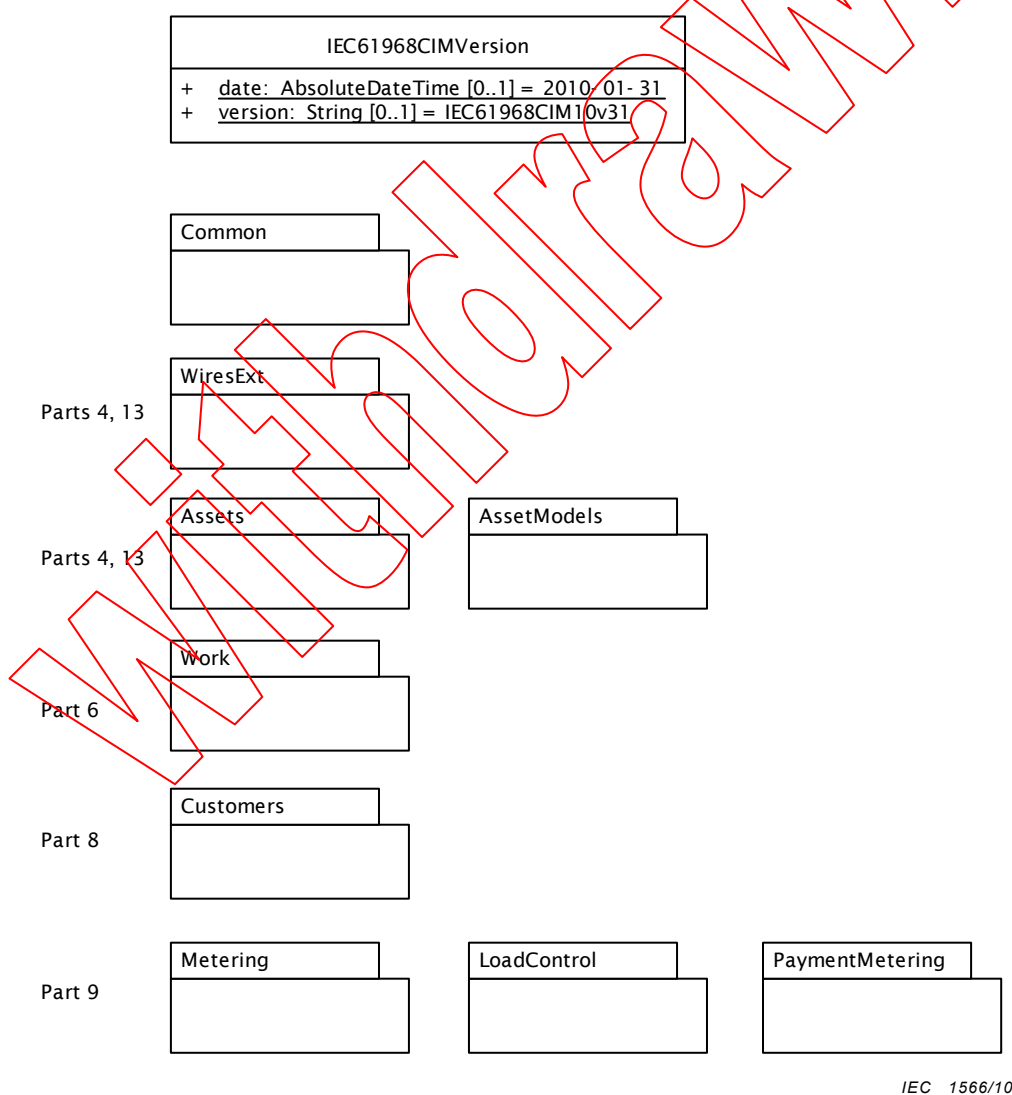
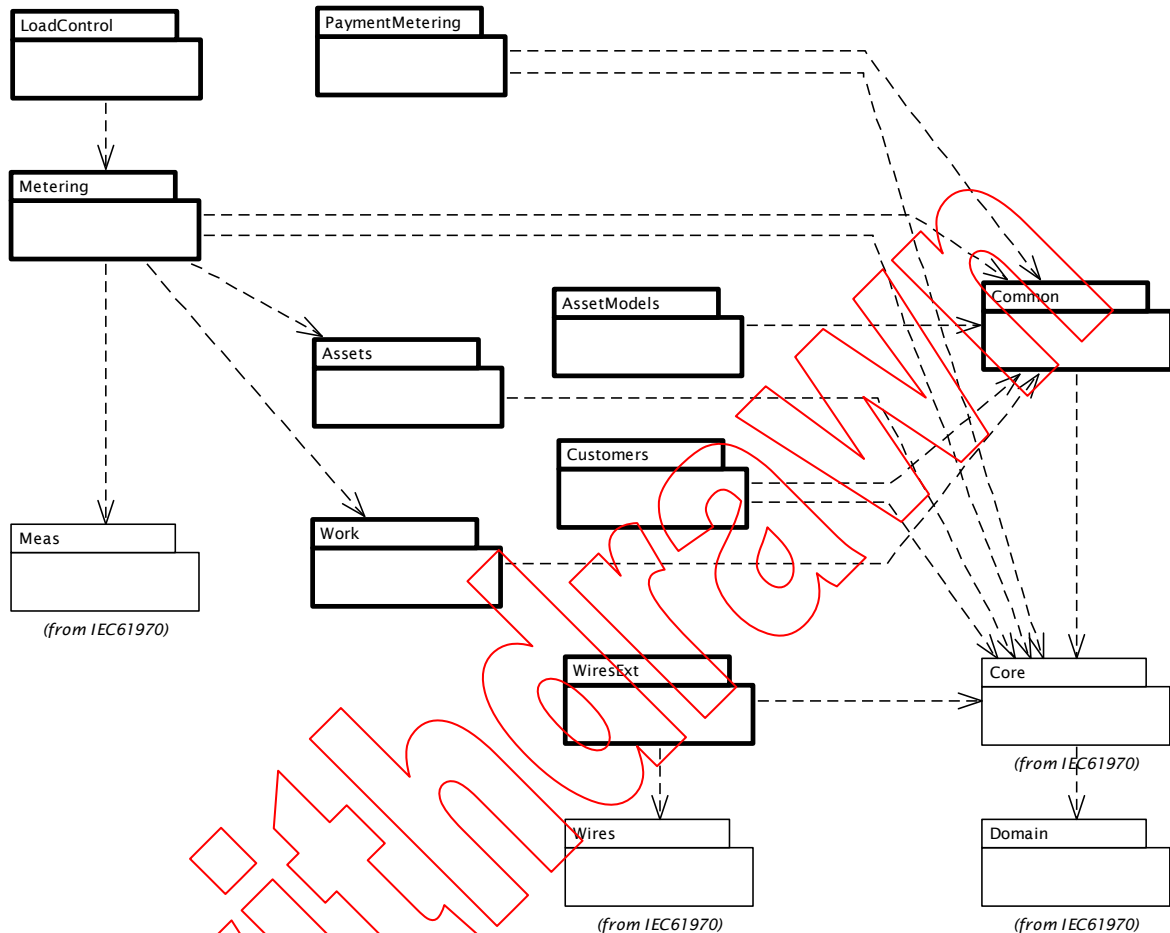


Figure 20 – Package diagram IEC61968::Main

This diagram shows version and normative contents of the CIM extensions for distribution.

Annotation on the left side of the diagram indicates the part of IEC 61968 that has been mainly driving modelling requirements for the respective package(s).

Figure 21 shows package diagram Dependencies.



IEC 1567/10

Figure 21 – Package diagram IEC61968::Dependencies

This diagram shows in bold contents of this package, as well as the dependencies based on inheritance only.

Figure 22 shows package diagram StdCIM.

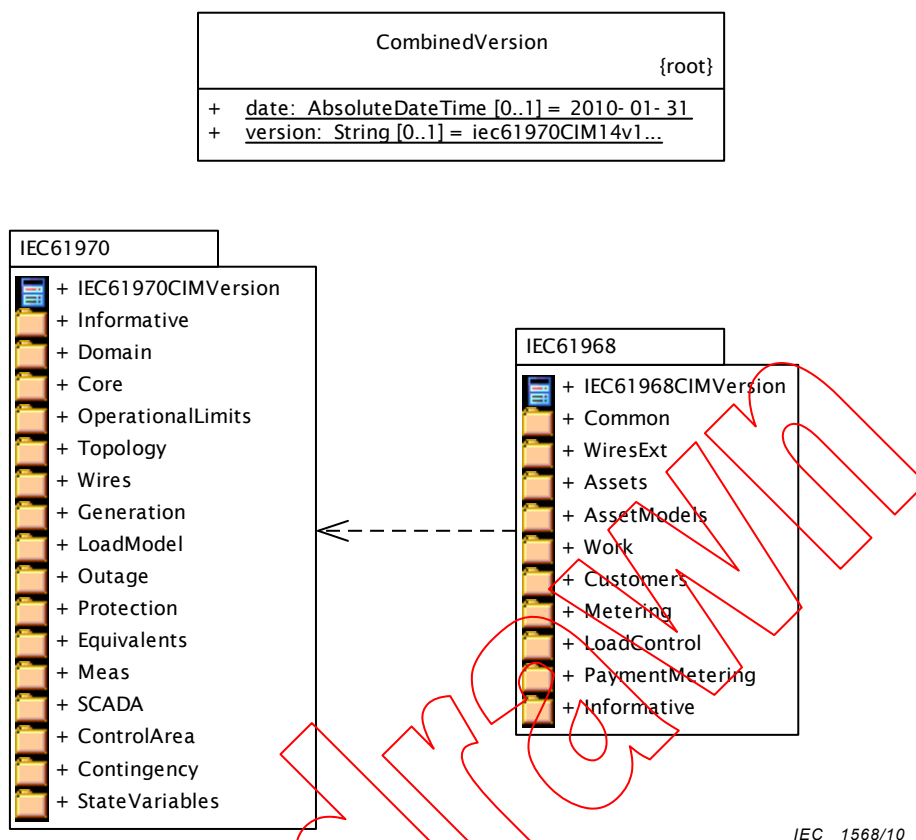


Figure 22 – Package diagram IEC61968::StdCIM

This diagram shows version and normative contents of the currently defined TC 57 CIM model.

Figure 23 shows logical diagram DCIMKeyClasses.

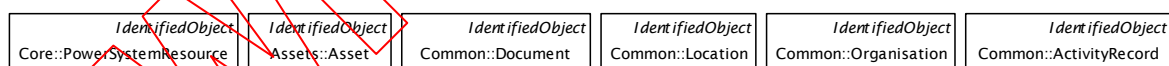


Figure 23 – Logical diagram IEC61968::DCIMKeyClasses

This diagram shows key classes in DCIM.

6.2.1 IEC61968CIMVersion

IEC 61968 version number assigned to this UML model.

Table 5 shows all attributes of IEC61968CIMVersion.

Table 5 – Attributes of IEC61968::IEC61968CIMVersion

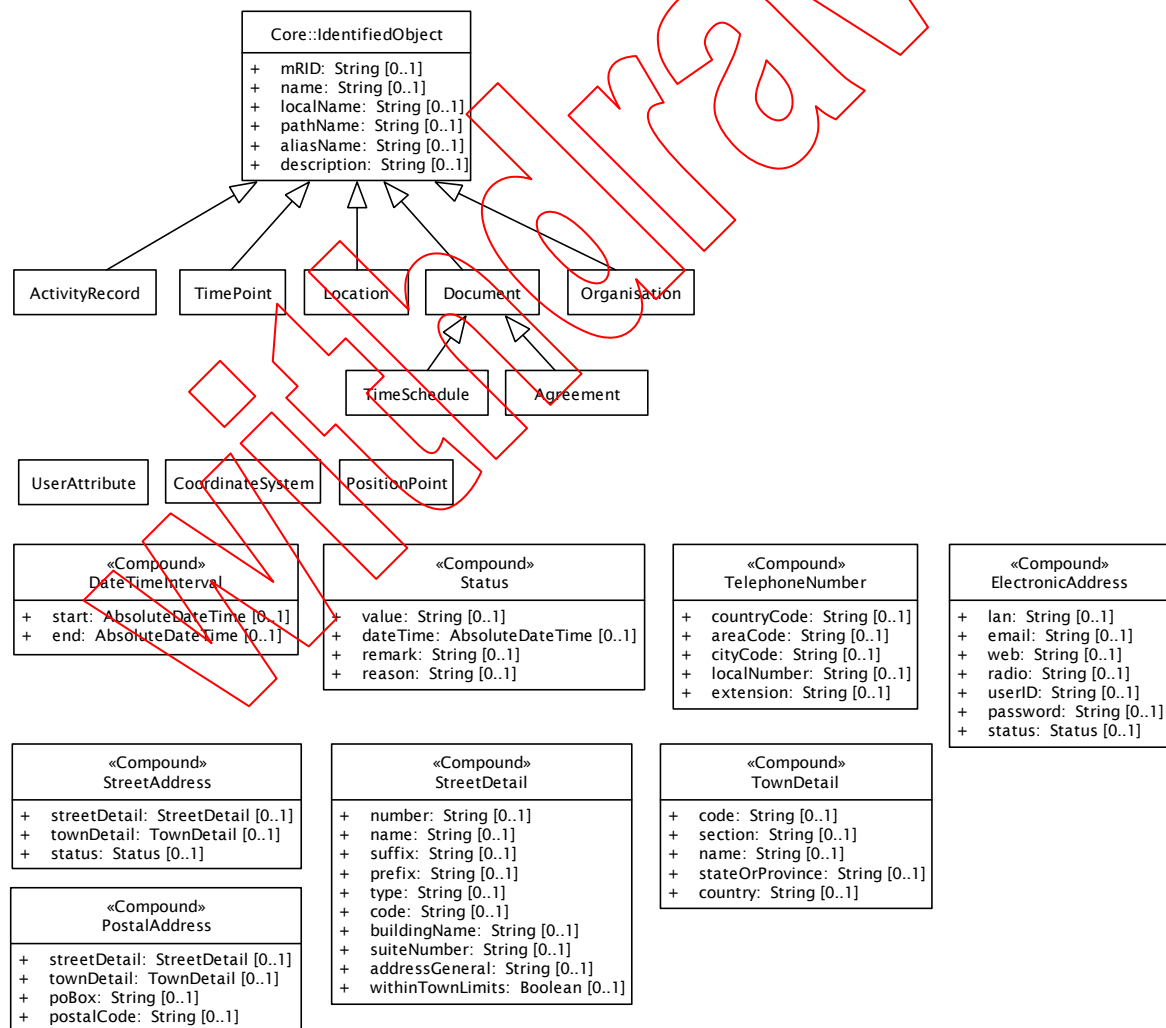
name	type	description
date=2010-01-31 (const)	AbsoluteDateTime	Form is YYYY-MM-DD for example for January 5, 2009 it is 2009-01-05.
version=IEC61968CIM10v31 (const)	String	Form is IEC61968CIMXXvYY where XX is the major CIM package version and the YY is the minor version. For example IEC61968CIM10v17.

6.2.2 Package Common

6.2.2.1 General

This package contains the information classes that support distribution management in general.

Figure 24 shows logical diagram CommonInheritance.

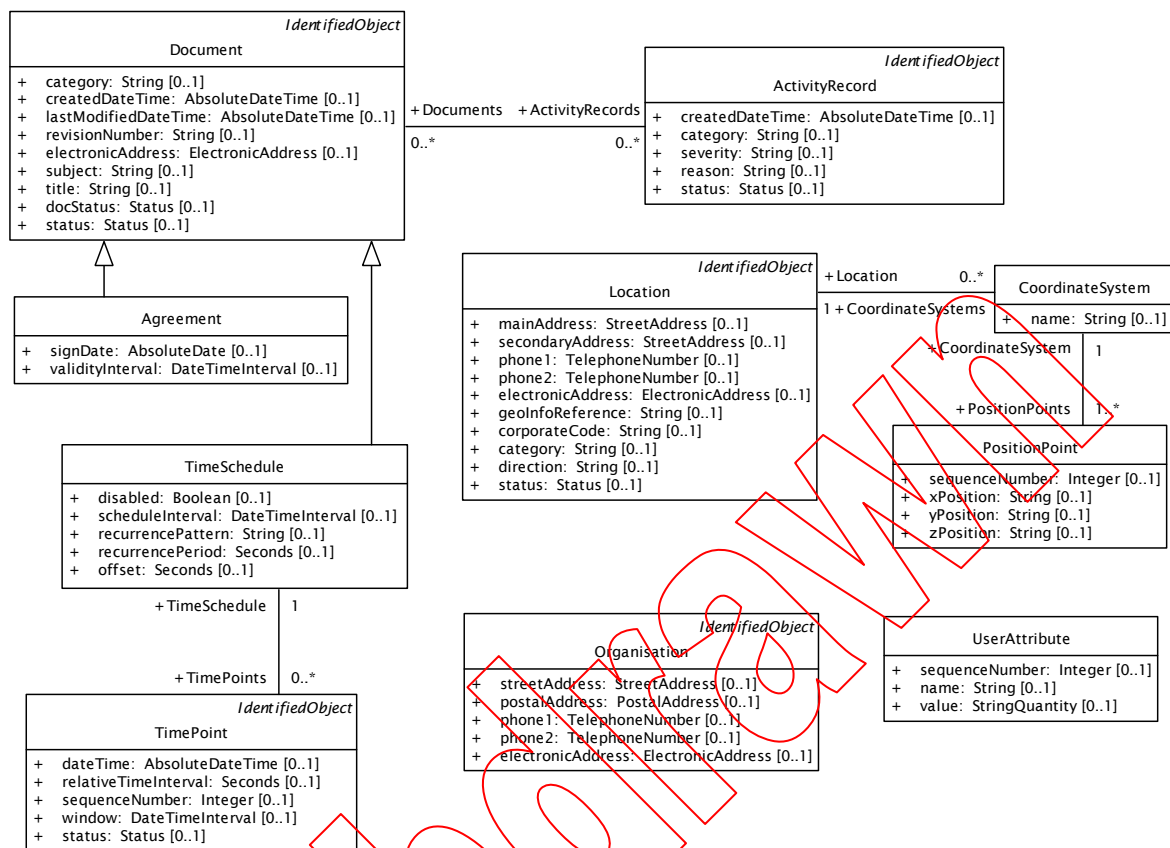


IEC 1570/10

Figure 24 – Logical diagram Common::CommonInheritance

This diagram shows inheritance hierarchy for normative classes from this package, as well as enumerations and compound types.

Figure 25 shows logical diagram CommonOverview.

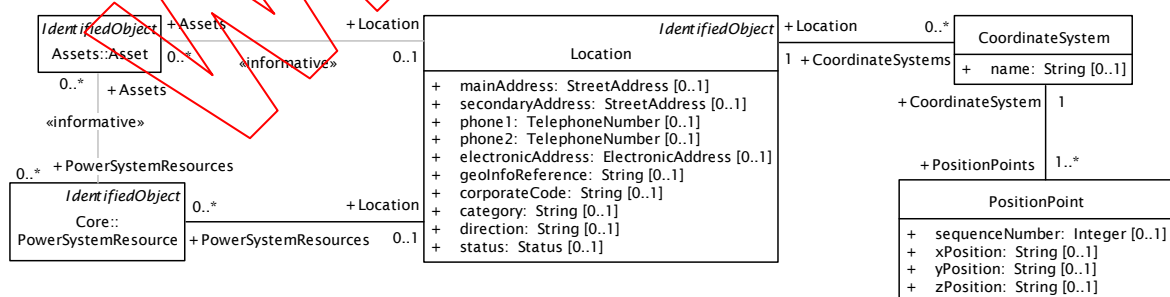


IEC 1571/10

Figure 25 – Logical diagram Common::CommonOverview

This diagram shows normative classes from this package.

Figure 26 shows logical diagram DCIMLocations.



IEC 1572/10

Figure 26 – Logical diagram Common::DCIMLocations

This diagram shows classes used to model locations of power system resources and assets in DCIM.

6.2.2.2 DateTimeInterval compound

Interval of date and time.

Table 6 shows all attributes of DateTimeInterval.

Table 6 – Attributes of Common::DateTimeInterval

name	type	description
start	AbsoluteDateTime	Date and time that this interval started.
end	AbsoluteDateTime	Date and time that this interval ended.

6.2.2.3 Status compound

Current status information relevant to an entity.

Table 7 shows all attributes of Status.

Table 7 – Attributes of Common::Status

name	type	description
value	String	Status value at 'dateTime'; prior status changes may have been kept in instances of ActivityRecords associated with the object to which this Status applies.
dateTime	AbsoluteDateTime	Date and time for which status 'value' applies.
remark	String	Pertinent information regarding the current 'value', as free form text.
reason	String	Reason code or explanation for why an object went to the current status 'value'.

6.2.2.4 PostalAddress compound

General purpose postal address information.

Table 8 shows all attributes of PostalAddress.

Table 8 – Attributes of Common::PostalAddress

name	type	description
streetDetail	StreetDetail	Street detail.
townDetail	TownDetail	Town detail.
poBox	String	Post office box.
postalCode	String	Postal code for the address.

6.2.2.5 StreetAddress compound

General purpose street address information.

Table 9 shows all attributes of StreetAddress.

Table 9 – Attributes of Common::StreetAddress

name	type	description
streetDetail	StreetDetail	Street detail.
townDetail	TownDetail	Town detail.
status	Status	Status of this address.

6.2.2.6 StreetDetail compound

Street details, in the context of address.

Table 10 shows all attributes of StreetDetail.

Table 10 – Attributes of Common::StreetDetail

name	type	description
number	String	Designator of the specific location on the street.
name	String	Name of the street.
suffix	String	Suffix to the street name. For example: North, South, East, West.
prefix	String	Prefix to the street name. For example: North, South, East, West.
type	String	Type of street. Examples include: street, circle, boulevard, avenue, road, drive, etc.
code	String	(if applicable) Utilities often make use of external reference systems, such as those of the town-planner's department or surveyor general's mapping system, that allocate global reference codes to streets.
buildingName	String	(if applicable) In certain cases, the physical location of the place of interest does not have a direct point of entry from the street, but may be located inside a larger structure such as a building, complex, office block, apartment, etc.
suiteNumber	String	Number of the apartment or suite.
addressGeneral	String	Additional address information, for example a mailstop.
withinTownLimits	Boolean	True if this street is within the legal geographical boundaries of the specified town (default).

6.2.2.7 TownDetail compound

Town details, in the context of address.

Table 11 shows all attributes of TownDetail.

Table 11 – Attributes of Common::TownDetail

name	type	description
code	String	Town code.
section	String	Town section. For example, it is common for there to be 36 sections per township.
name	String	Town name.
stateOrProvince	String	Name of the state or province.
country	String	Name of the country.

6.2.2.8 ElectronicAddress compound

Electronic address information.

Table 12 shows all attributes of ElectronicAddress.

Table 12 – Attributes of Common::ElectronicAddress

name	type	description
lan	String	Address on local area network.
email	String	Email address.
web	String	World wide web address.
radio	String	Radio address.
userID	String	User ID needed to log in, which can be for an individual person, an organisation, a location, etc.
password	String	Password needed to log in.
status	Status	Status of this electronic address.

6.2.2.9 TelephoneNumber compound

Telephone number.

Table 13 shows all attributes of TelephoneNumber.

Table 13 – Attributes of Common::TelephoneNumber

name	type	description
countryCode	String	Country code.
areaCode	String	Area or region code.
cityCode	String	(if applicable) City code.
localNumber	String	Main (local) part of this telephone number.
extension	String	(if applicable) Extension for this telephone number.

6.2.2.10 ActivityRecord

Records activity for an entity at a point in time; activity may be for an event that has already occurred or for a planned activity.

Table 14 shows all attributes of ActivityRecord.

Table 14 – Attributes of Common::ActivityRecord

name	type	description
createdDateTime	AbsoluteDateTime	Date and time this activity record has been created (different from the 'status.dateTime', which is the time of a status change of the associated object, if applicable).
category	String	Category of event resulting in this activity record.
severity	String	Severity level of event resulting in this activity.