Location (Logical Data Model)

Ver 0.1 (23-02-2019).

CIM Standard | CIM Standard but modified (i.e. attribut/entity renamed) | Non-CIM (extensions)

All enumerations are modelled in the database as plain tables, including english and local alias texts plus sort order. This to support easy customization and utilization in database driven applications such a QGIS.

CIM: "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 can be defined with one or more postition points (coordinates) in a given coordinate system"

Click me, to see YouTube introduction video

Location
- PK Id
- MRID (uuid)
- Coordinates (put on concrete entities)
- Direction-Information (text)
- LocationOriginKind (enum)

Jesper/DAX: The CIM location concept is modelled in the our logical relational data model using a table called Location, that acts as the superclass (or super-entity) for all other location-ish entities (that you'll find just below it).

Regarding the Master Resource Identifier (MRID): Most classes in the CIM standard inherit from identified object, and typical though many other classes in a comprehensive inheritance hierarchy. However, we don't want to model that explicitly in our logical data model. Instead we put a master resource identified (MRID), and possible other attributes from inherited classes, on super-entities of choice. The reason for such "inheritance denormalization hack" is that explicitly modelling the complete CIM inheritance structure in a logical data model will be insane. I'm pretty sure that database people, GIS people, and super users would "kill" us, if we did so. And if you're a software engineer reading this, don't wory, becasue REST services etc. can still utilized a more comprehensive modelling paradigm of choice.

Regarding coordinates: In CIM the coordinat information is related directly to the Location class – i.e. conceptually modelled as a value property of Location. This make sense in a conceptual model, but in the logical model it give us trouble, because many database driven GIS applications (such as QGIS) prefer/require that one table only holds one type of geometry (i.e. points, lines or polygons). That's why the Coordinate property/attribut is moved down to the concrete entities such as Access Address, Building etc. At that level we know if points, lines or polygons are prefered by the customer. It's again a "denormalisation hack" I choose to do, because big parts of the GIS-world (both software and humans) are not ready for the idea of mixing geometry types in the same property. But I'm all ears, if you can prove it was a wrong decision. I was thinking of solving this problem using database views, but it also complicates things unnecessarily.

Overall: The goal is to create a logical data model that is easy for humans to learn, discuss and improve. I have bad experience with UML models like used in CIM, when it comes to collaboration with users, and even developers. A UML model can quickly gets unmanageable in terms of human comprehension. A logical data model is easier to comprehend, expecially if you also have a database to play with. But that does not mean we don't have a conceptual model as well. However, OWL, not UML, is ised to model the conceptual model. Again because UML is not good for humans (from my experience). Using tools like web protegy, is a more efficient way of doing conceptual modelling in collaboration with domain experts (customers /business users). Because the most crutial thing to get success, its to get the domain experts in the loop!

Click me, to access Web Protegé projects: https://daxgrid.org/webprotege

Building Structure Kind Enumeration
- unknown
- singleDwellingUnit
- multiDwellingUnit

Zone Kind Enumeration
- centralOfficeZone
- flexZone
- customerZone

Unit Ownership Kind Enumeration
- unknown
- private
- housingAsscociation
- commercial
- govermental
- other

Unit Usage Kind Enumeration
- unknown
- household
- accommodation
- office
- retail
- institutional
- industrial
- other

Location Origin Kind Enumeration
- official
- pendingOfficial
- unofficial

General Zone entity as defined by CIM: Area divided off from other areas. It may be part of the electrical network, a land area where special restrictions apply...

Zone
- PK FK LocationId
- Name (text)
- ZoneKind (enum)
- Coordinates (PolygonZ)

There's a compound value object called "Street Address" in CIM. I changed it to be a specialized location entity called "Access Address", because it can also be an address along a rail road or in the middle of nowhere (i.e. a wind mill etc.).

Access Address
- PK FK LocationId
- FK ExternalAccess-AddressId (text/uuid)
- FK PostalCode (text)
- FK MunicipalCode (text)
- FK RoadSegmentId
- HouseNumber (text)
- Coordinates (PointZ)

There's a compound value object called "Street Details" in CIM containing attributes such as suitNumber etc. I changed it to be a specialized location entity called "Unit Address" and added a floor name attribute because it fits the Danish way of handling addresses better.

Unit Address
- PK FK LocationId
- FK AccessAdressId
- FK ExternalUnit-AddressId (text/uuid)
- FloorName (text)
- RoomName (text)
- UnitUsageKind (enum)
- UnitOwnership-Kind (enum)

There's no building entity in CIM. However, buildings are very important in the process of telco planning, so we add it to the model as first class location citizen.

Building
- PK FK LocationId
- FK ExternalBuildingId (text/uuid)
- Name (text)
- BuildingStructure-Kind (enum)
- Coordinates (PolygonZ)

There's no building access point entity in CIM. However, these are important in the process of telco planning, so we add it to the model as first class location citizen.

Building Access Point
- PK FK LocationId
- FK BuildingId
- FK AccessAddressId
- Coordinates (PointZ)

There's no road entity in CIM. However, these are very important in the process of telco planning, so we add it to the model as first class location citizen.

Road Segment
- PK FK LocationId
- FK RoadCode (text)
- FK MunicipalCode (text)
- Name (text)
- Coordinates (PolylineZ)

Postal District
- PK Code (text)
- Name (text)

Municipal
- PK Code
- Name (text)

CIM: (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.

DAX/Jesper: This data model supports an external addressing system in greater details by allowing external ids on access addresses, unit addresses and buildings. Moreover road segments can be identified with both a municipal code and a road code, which is used to uniquely identify road segments in Denmark.

Route Model

Notice that these entities has an optional relation to one of the location entities defined above.

Route Node
- PK Id
- MRID (uuid)
- FK LocationId
- Coordinates (PointZ)

Route Segment
- PK Id
- MRID (uuid)
- FK LocationId
- FK FromNodeId
- FK ToNodeId
- RouteSegmentKind (enum)
- Coordinates (PolylineZ)

Route Segment Kind Enumeration
- unknown
- buried
- areal
- indoor
- drilling
- roadCrossover-Drilling
- roadCrossover-DuctBank
- microTrenching
- tunnel