IT System Specification

Example Implementation LADM

draft

25 March, 2011

Table of Contents

1 Implementation LADM

This package describes an example implementation of the complete LADM (DIS3 October 2010), which defines a reference Land Administration Domain Model (LADM) covering basic information-related components of Land Administration (including those over water as well as land, and elements above and below the surface of the earth).

The LADM provides a terminology for land administration, based on various national and international systems, allowing a shared description of different formal or informal practices and procedures in various jurisdictions. The LADM provides an abstract, conceptual schema with four basic packages related to

- **basic administrative units**, **rights**, **responsibilities**, and **restrictions** (to ownership rights);
- **parties** (people and organizations);
- **spatial units** (parcels, buildings and utility networks);
- spatial sources (surveying), and **spatial representations** (geometry and topology);

In the majority of cases the implementation is following the platform independent LADM as much as possible, however, based on arguments with respect to flexibility, performance, or simplicity, some alternative implementations have been made, which are documented in the following sections.

For each (group of) classes the decisions and comments with regard to implementation are described and visualised. In the following text some general decisions/assumptions are provided with regard to the implementation of the LADM classes in general.

- O/R mapper application framework
- Implementation of Attributes
- Implementation of Binary Large Objects
- Implementation of Inheritance Hierarchies

O/R mapper application framework
For implementation in the .Net environment and PostgreSQL database the tool **LLBLGenPro** (http://www.llblgen.com) is used, capable of generating a Visual Studio / .NET/ C# O/R mapper application framework and an implementation in the PostgreSQL/PostGIS database. The LLBLGen Pro designer allows a (*.Net platform specific*) "model first" approach (as opposed to "database-first") and support the implementation of decisions on how to implement model class hierarchies, and datatypes etc. The framework supports authorisation, validation, transaction/Unit of Work, entity collections, concurrency control, XML serialization & Webservice support, custom developed extensions, geographical datatypes [through type converters] etc.

Implementation of Attributes (.Net datatype, lenght, precision, etc.)

- *Name*: Varchar/String100
- *Description/Notes*: Varchar/String1000
- *LocationByText*: Varchar/String2000
- Inheritance Discriminator (*ObjectType* in class hierarchy): ObjectType, Varchar/String50
- *Id* (in combination with CodeProviderType): Varchar/String25
- *Image*: bytea/byte[]
- *Value/Amount* (in combination with CurrencyType): Single(12,2)

Implementation of Binary Large Objects
PostgreSQL provides two distinct ways to store binary data. Binary data can be stored in a table using the data type bytea (.Net System.Byte[]) or by using the Large Object feature which stores the binary data in a separate table in a special format and refers to that table by storing a value of type oid in your table.

Implementation of Inheritance Hierarchies
The instance hierarchies of super and sub classes in LADM can be used in object-oriented C#, a decision needs to be taken for the implementation in the PostgreSQL database. A number of inheritance hierarchies exist which are either implemented as "*target per entity*" (one table per class) or as "*target per entity hierarchy*" (one table for all super

and sub classes in a hierarchy). Both methods have their disadvantages. In "*target per entity*" many table are created with a possible effect on performance, and in "*target per entity hierarchy*" in some cases referential integrity of the database is at stake. Both implementations exist, for implementations related to the core elements, in principle "*target per entity*" will be selected:

Target per entity hierarchy

- **CodeList**: table CodeList (facilitating classes *BasicAdministrativeUnitType, BuildingUnitType, CodeProviderType, ContactMechanismType, CurrencyType, DimensionType, EventType, ExtLandCoverType, ExtLandUseType, ExtTaxationType, ExtTaxationType, GenderType, GroupPartyType, InterpolationType, LevelContentType, MeasureType, MonumentationType, MortgageType, PartyRoleType, PartyType, PointType, PointUomValueType, RegisterType, RelationshipType, ResponsibilityType, RestrictionType, RightType, ServiceType, ServiceWorkType, SourceType, SpatialUnitUomValueType, StructureType, SurfaceRelationType, UtilityNetworkStatusType,UtilityNetworkType)*
- **ObjectQuality**: table ObjectQuality (facilitating classes *SpatialUnitQuality, PointQuality, BasicAdministrativeUnitQuality, PartyQuality, RrrQuality*)
- **ObjectSource**: table ObjectSource (facilitating classes *SpatialUnitQuality, PointQuality, BasicAdministrativeUnitQuality, PartyQuality, RrrQuality*)
- **ExtParty:** table ExtParty (facilitating classes *NonNaturalPerson, NaturalPerson*)
- **ContactMechanisme** (the actual addresses, e-mail, phone numbers): table ContactMechanism (facilitating classes *Address* [with C*oordinate]*, *Internet, Phone*)

Target per entity

- **ObjectContactMechanism** (the reference to actual addresses, e-mail, phone numbers): tables ObjectContactMechanism (ExtPartyContactMechanism, PartyContactMechanism, SpatialUnitContactMechanism)
- **ObjectUnitOfMeasure** (for values with a certain measurement unit): tables ObjectUnitOfMeasure (PointUnitOf Measure, SpatialUnitOfMeasure)
- **PartyRole**: tables PartyRole (ServicePartyRole, ServiceWorkPartyRole, SourcePartyRole, RrrPartyRole)
- **Event**: tables Event (BasicAdministrationUnitEvent, BoundaryEvent, PointEvent, RrrEvent, SpatialUnitEvent, SpatialUnitForBasicAdministrativeUnitEvent)
- **Boundary**: tables Boundary (Face, FaceString)
- **Rrr**: tables Rrr (Responsibility, Restriction, Right, Mortgage

Implementation of Geometry

- TO DO: Fine tune approach around the *spatial* attributes of LADM:

"public"."ContactMechanism"."Coordinate" -> 'POINT'
"public"."Face"."Geometry" -> 'MULTISURFACE'
"public"."FaceString"."Geometry" -> 'MULTICURVE'
"public"."Point"."OriginalLocation" + "TransformedLocation" -> 'POINT'
"public"."SpatialUnit"."ReferencePoint" + "Geometry" -> 'POINT'/'GEOMETRY'
"public"."SpatialUnitGroup"."ReferencePoint" -> 'POINT'

- TO DO: *SpatialUnit.ReferencePoint* is redundant and derived based on association from *SpatialUnit* to *Point*.
- TO DO: *SpatialUnit.ReferenceGeometry* is redundant and derived based on *BoundaryFace* and *BoundaryFaceString*.

(*Version of Package: 25-3-2011*)

Diagram LADM CodeLists

The packages (Party, Administrative, SpatialUnit and Surveying & Representation) contain a number of CodeLists, all of which are implemented through one class (CodeList), with attributes Guid, Code [LADM value] and Description [alternative terminology/languages for use in GUI]. These attributes for the super class are all inherited by the various subtypes/codelists. Enumerations are also placed in the CodeList entity hierarchy. Column ObjectType is used for the name of the CodeList.

- TO DO: make database check *constraints* for Enumerations.
- TO DO: deal with *constraints* for specific values of CodeLists and related classes and associations, for example:

TO DO: The implied business rule on *LA_Party.type/LA_GroupParty.type* "if value is baunit it should apply the association with LA_BAUnit".
TO DO: LA_Party.type "group" is only available for LA_GroupParty
TO DO: LA_PartyType.baunit & LA_BAUnit.baunitAsParty.LA_Party).

Examples of CodeList:

- *BasicAdministrativeUnitType*
- *BuildingUnitType*
- *CodeProviderType*
- *ContactMechanismType*
- *CurrencyType*
- *DimensionType*
- *EventType*
- *ExtLandCoverType*
- *ExtLandUseType*
- *ExtTaxationType*
- *ExtTaxationType*
- *GenderType*
- *GroupPartyType*
- *InterpolationType*
- *LevelContentType*
- *MeasureType*
- *MonumentationType*
- *MortgageType*
- *PartyRoleType*
- *PartyType*
- *PointType*
- *PointUomValueType*
- *RegisterType*
- *RelationshipType*
- *ResponsibilityType*
- *RestrictionType*
- *RightType*
- *ServiceType*
- *ServiceWorkType*
- *SourceType*
- *SpatialUnitUomValueType*
- *StructureType*
- *SurfaceRelationType*
- *UtilityNetworkStatusType*
- *UtilityNetworkType*

(*Version of Diagram: 24-3-2011*)

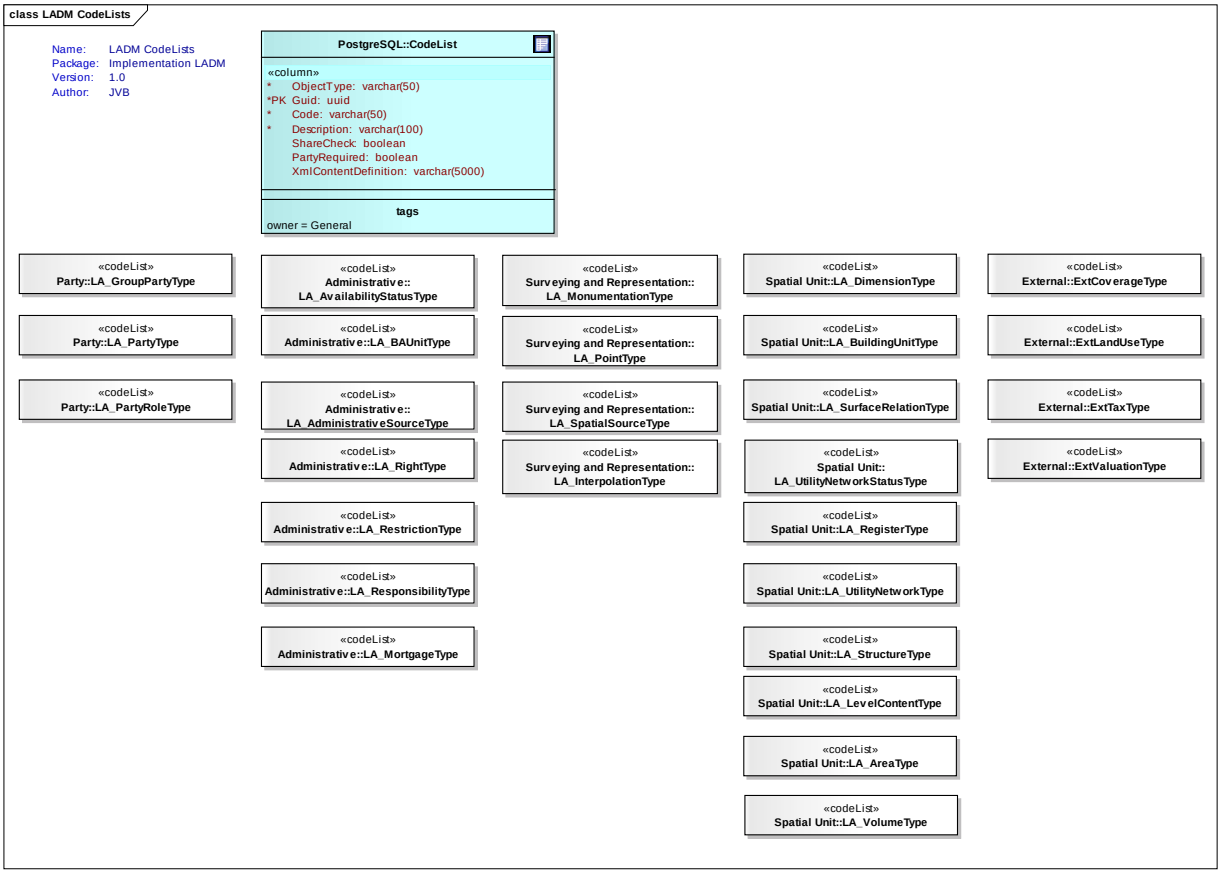


Figure: 1 : LADM CodeLists

Diagram VersionedObject [6.2.1]

Super class *VersionedObject has specialisations:*

- LA_RRR (LA_Mortgage)
- LA_BAUnit
- LA_RequiredRelationshipBAUnit
- LA_RequiredRelationshipSpatialUnit
- LA_SpatialUnit
- LA_SpatialUnitGroup
- LA_Level
- LA_BoundaryFace
- LA_BoundaryFaceString
- LA_Point
- LA_Party (LA_GroupParty)
- LA_PartyMember

The class *VersionedObject could be* implemented in the following manners:

- *quality*; as class *ObjectQuality* and specialisations, for example *PartyQuality*, *RrrQuality*, *BasicAdministrativeUnitQuality*, *SpatialUnitQuality* implemented in one table *ObjectQuality*.
- *source*: as class *ObjectSource* and specialisations, for example *PartySource*, *RrrSource*, *BasicAdministrativeUnitSource*, *SpatialUnitSource* implemented in one table *ObjectSource*.

- *beginLifespanVersion*: as attributes in classes, for example *BasicAdministrativeUnit.beginLifespanVersion*.
- *endLifespanVersion*: as attributes in classes, for example *BasicAdministrativeUnit.endLifespanVersion*

- TO DO: Decide on the implementation of *VersionedObject (begin & endLifespanVersion, source & quality)* for all associated classes.
- TO DO: Resolve the issue of *Source* (*Event*) causing *ObjectSource/Quality* to be obsolete. For example SpatialUnitSource, and SpatialUnitEvent have overlapping/redundant information; possibly for these situations the *beginLifespanVersion* and *endLifespanVersion* will be implemented, but not the specialisations of *ObjectSource & ObjectQuality?*

(*Version of Diagram: 25-3-2011*)

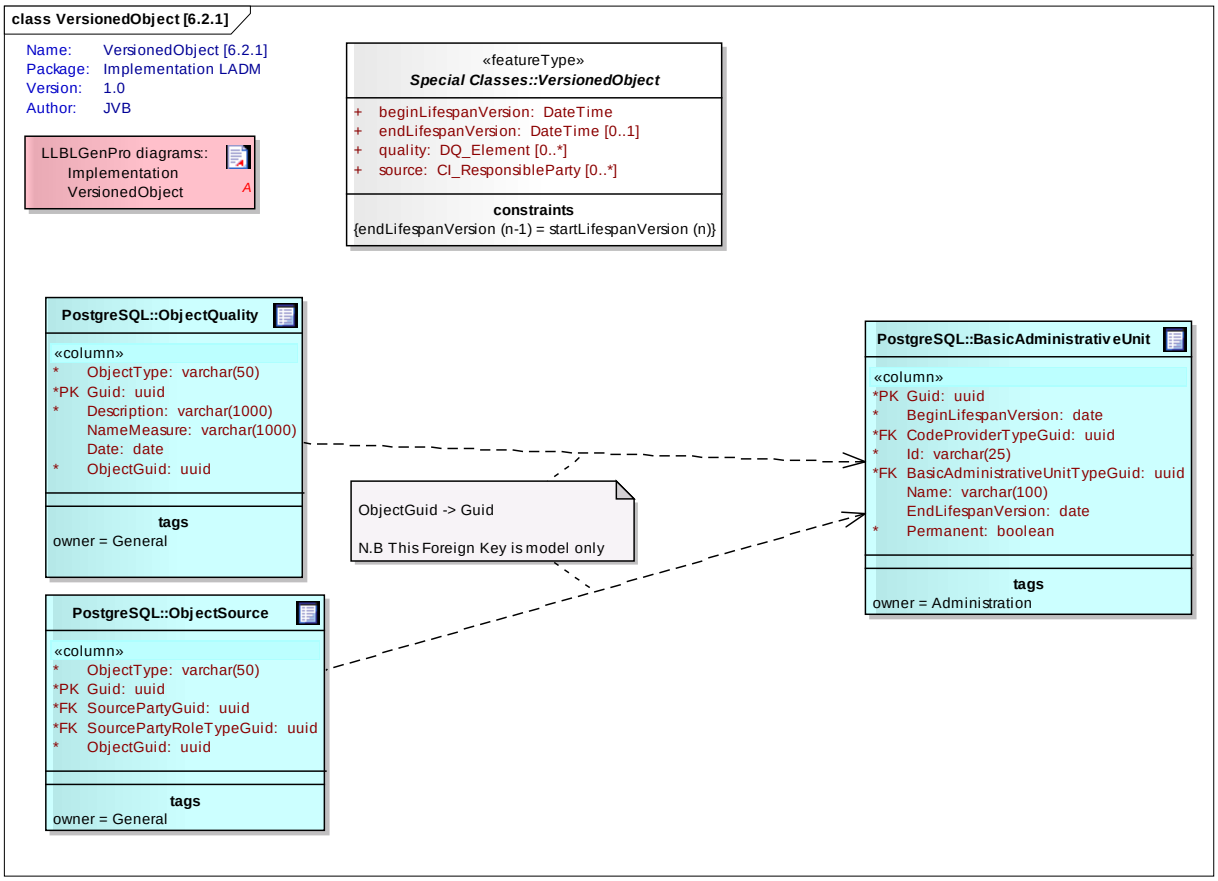


Figure: 2 : VersionedObject [6.2.1]

Diagram Oid [6.2.4]

A number of classes contain attributes are described by the class *Oid*. For example, the class *Party* contains one reference to an external party *extPID* and has one unique identifier *pID*. The *Oid* class and related attributes will be implemented as follows:

- The class with an *Oid* attribute will be associated with class *CodeProviderType* (a specialisation of class CodeList), and one attribute *Id* will be created, to implement the *Oid.locaId* datatype (*Oid.namespace* is represented by *CodeProviderType.Guid*).
- A unique constraint *UkCodeProviderId* will be based on *CodeProviderTypeGuid* and *Id*.
- TO DO: check if all relevant UK are implemented and if more can be made.
- For a selection of the core classes (for example *SpatialUnit*, *Party*) multiple values of external *Oid* can be maintained, see class *SpatialUnitCode* and *PartyCode*. This allows the storage of multiple (external) unique

references to a spatial unit (agency number, national UPN, etc) or for a person (passport, id-card)

- TO DO: Finetune approach with regard to Oid (*CodeProviderUk*).

Examples of CodeProviders/Namespaces:
- Cadastre Office
- Land Register Office 1
- Land Register Office 2
- Land Valuation Office
- Land Taxation Office
- Party
- Passport
- Social Fiscal
- SpatialUnit
- Basic Administrative Unit
- Right
- Restriction
- Responsibility
- Mortgage
- Process
- Boundary
- ContactMechanism
- Point
- SpatialUnitGroup (Province, District, Region, etc)

Notes:
- The local identifier should be unique within the namespace, i.e. no other spatial object should carry the same identifier.
- If INSPIRE compliance is needed, then he local identifier should only use the following set of characters: {"A" …"Z", "a"…"z", "0"…"9", "_", ".","," "-",}, i.e. only letters from the Latin alphabet, digits, underscores, periods, commas, and dashes are allowed.

(*Version of Diagram: 25-3-2011*)

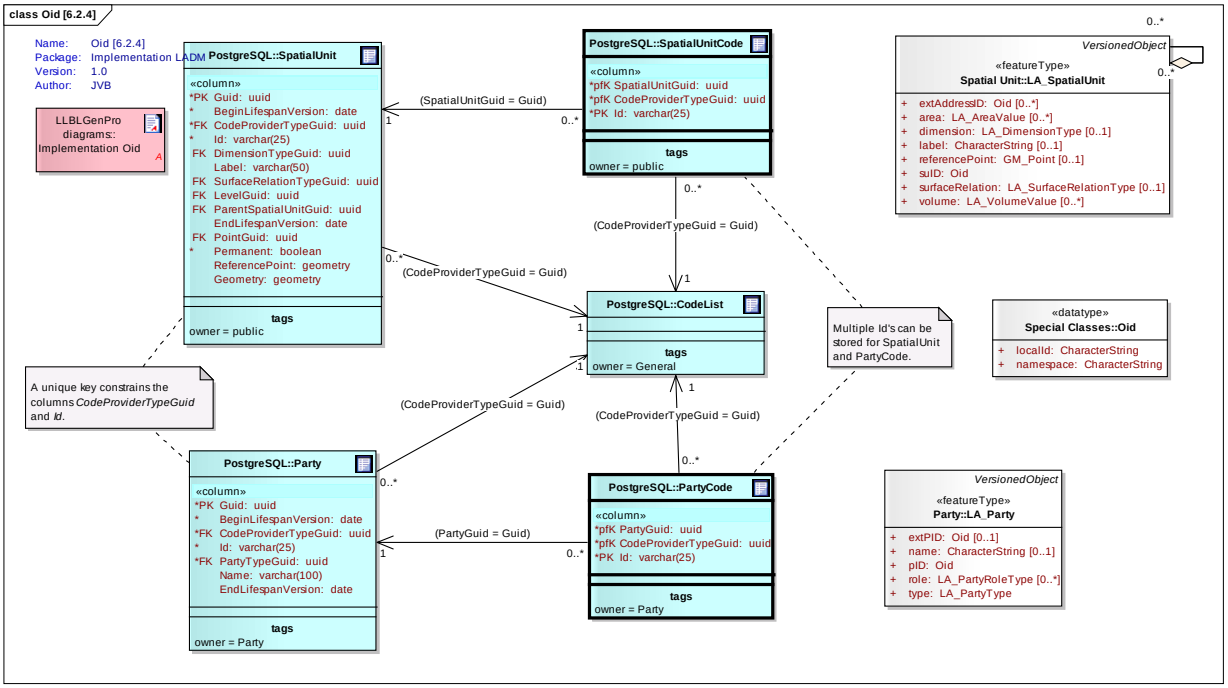


Figure: 3 : Oid [6.2.4]

Diagram PartyRoleType [6.3.4]

There are quite some associations between LA_Party and other classes to indicate the role of a party in the *data update and maintenance process*, in some cases more than one association (Consider LA_Source, LA_SpatialSource, LA_AdministrativeSource & their associations to LA_Party).

To provide flexibility for future associations to Party (from class *Rrr*, *Source*, *Service*, *ServiceWork* and others), these associations are *implemented* by an (association) class PartyRole (for example with specialisations *ServiceParty*, *ServiceWorkParty*, *SourceParty*, *ServiceParty*, *RrrParty and* their sub classes), covering the requirements of LADM, but flexible to allow more associations with regard to party role.The constraints to these associations could then be imposed/implemented with for example a business-rule engine or with database check constraints and/or triggers. (*Version of Diagram: 25-3-2011*)

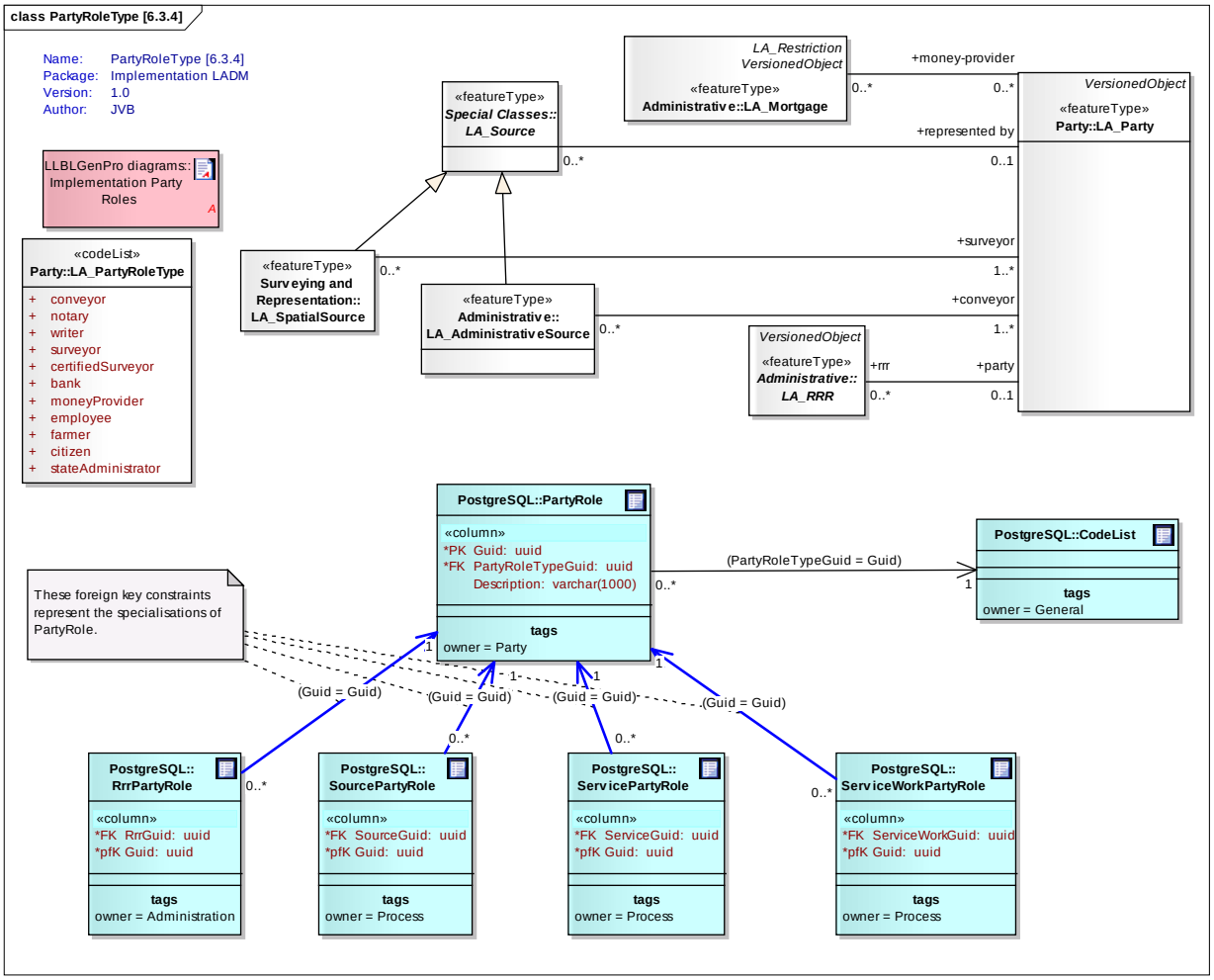


Figure: 4 : PartyRoleType [6.3.4]

Diagram Classes of Party Package [6.3]

The classes *LA_Party*, *LA_GroupParty*, and *LA_PartyMember* are implemented as follows:

- *LA_GroupParty.groupId* implemented similarly as *pID*, inherited from *LA_Party*.
- *LA_Party.role* (*LA_PartyRoleType*) will be implemented as classes *SourceParty*, *ServiceParty*, *RrrParty*
- TO DO: The constraint "*LA_GroupParty* has 2 or more [*] *LA_Parties*".
- TO DO: the "aggregation" association class LA_PartyMember; a LA_Party instance is only "party member" for one instance of LA_GroupParty.
- TO DO: The *Rational.numerator* and *denominator* are implemented as attributes for *LA_PartyMember*. A check constraint to make sure they are both filled or both empty and numerator/denominator < 1 and denominator != 0, implemented through check constraint and default values.
- None of the examples in Annex J describe LA_PartyMember.share, but it is assumed to be a share in the group, that in its turn is right holder for a RRR (which can be a share of the total as well)

- TO DO: Business rule "sum(LA_PartyMember.share)=1 per group"

(*Version of Diagram: 25-3-2011*)

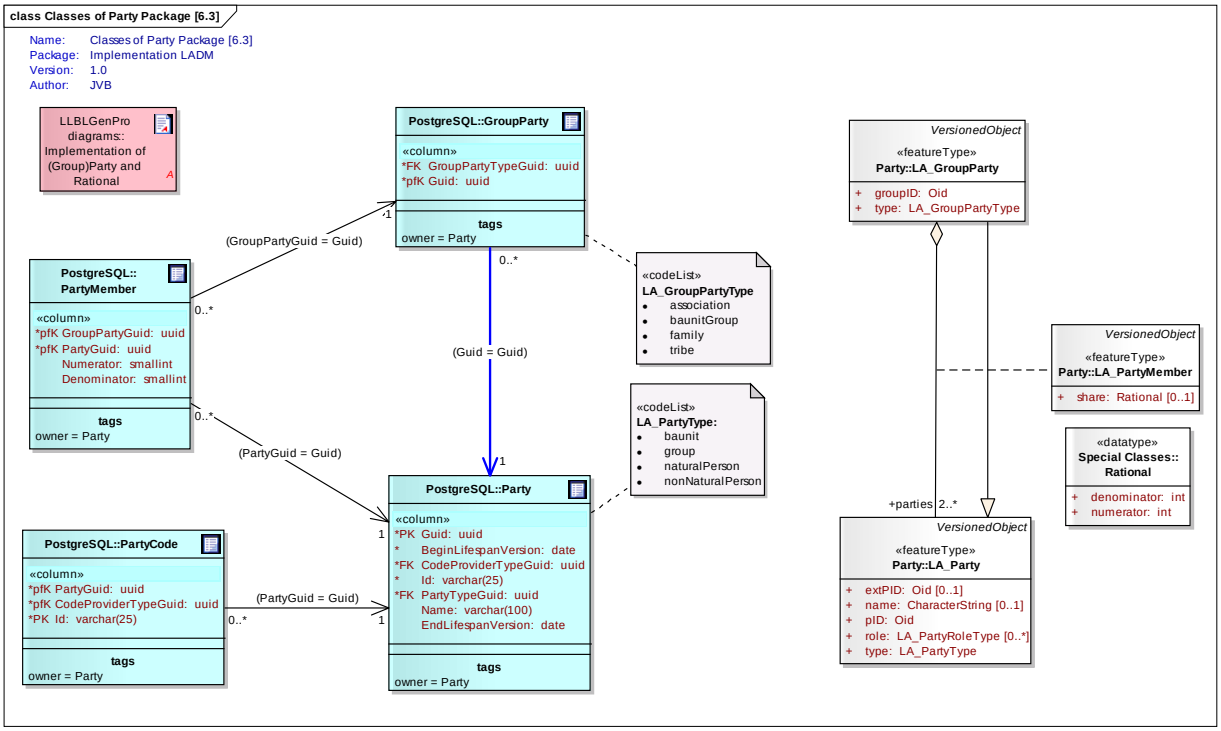


Figure: 5 : Classes of Party Package [6.3]

Diagram ExtParty and ExtAdresses [K.2 K.3]

LADM.ExtParty is implemented as a superclass *ExtParty* with subclasses *NaturalPerson* and *NonNaturalPerson*. The *Party.Id (LA_Party.extPID)* is populated with a value from *ExtParty.Id* (+*CodeProviderTypeGuid*). The *Party.Name* is **derived** from a selected *ExtParty* instance

LADM.ExtParty.extAddress [0..'] is implemented through *ExtPartyContactMechanism* (subclass of *ObjectContactMechanism*) and *ContactMechanism* (subclass *Address*, *Internet* and *Phone*). The addresses used for Party (*PartyContactMechanism*) can be selected from *ContactMechanism*.

The associations between (*Ext)Party* and *ContactMechanism (PartyContactMechanism* & *ExtPartyContactMechanism)* have an inherited association to class *ContactMechanismType* to indicate the value of the ContactMechanism to the Party (for example HomeAddress, HomeFax, BusinessFax, CorresspondenceAddress, Private E-mail Address, Business E-mail, Home Phone, Business Phone, Telephone, Mobile, Car) (*Version of Diagram: 25-3-2011*)

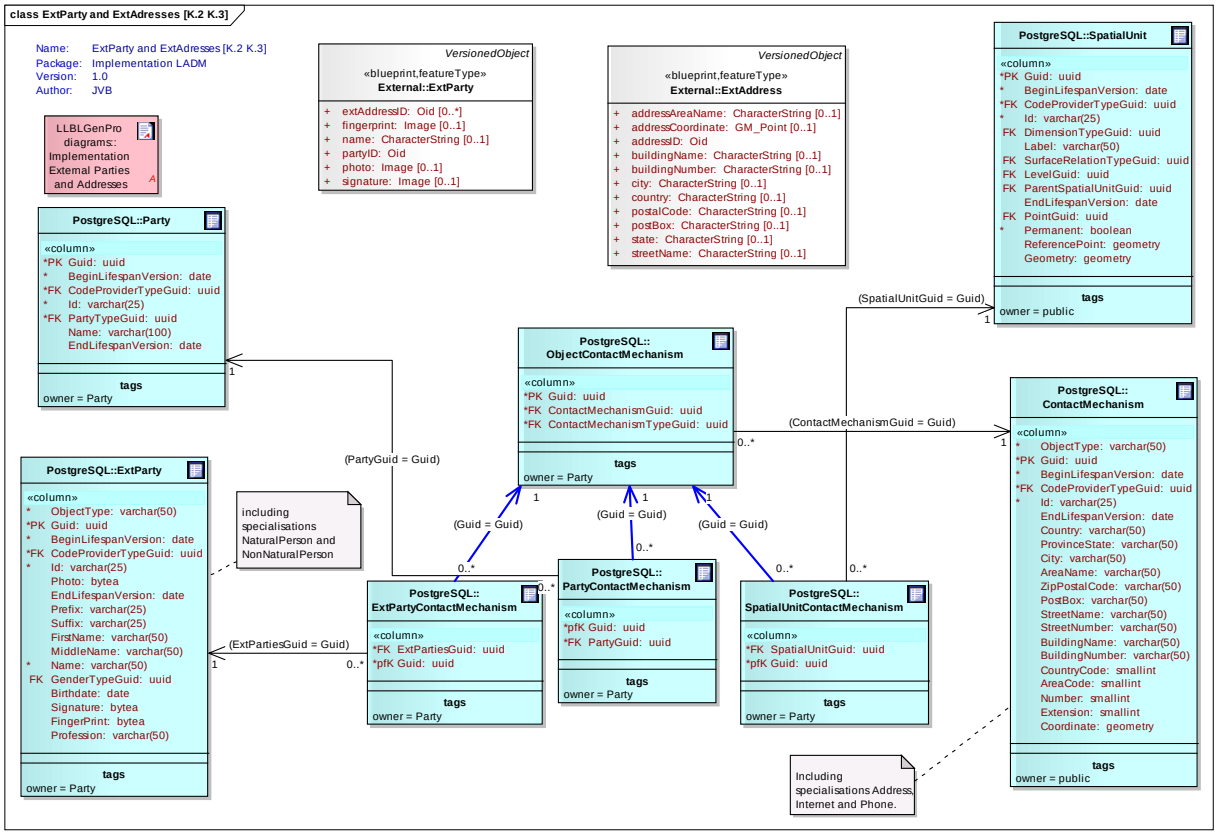


Figure: 6 : ExtParty and ExtAdresses [K.2 K.3]

Diagram Classes of Administrative Package [6.4] A

The classes with regard to LA_BasicAdministrativeUnit are implemented as follows:

- *LA_RequiredRelationshipBAUnit* is implemented as *RelationshipBasicAdministrativeUnit & RelationshipType (e.g. required), see also LA_RequiredRelationshipSpatialUnit.* This enable more flexibly to register recursive associations between classes.
- *LA_RequiredRelationshipBAUnit.relationship* implemented as *RelationshipBasicAdministrativeUnit.Description*
- Association *banunitAsParty [m:n]* implemented as class *BasicAdministrativeUnitAsParty*
- Association *LA_BAUnit <-> LA_AdministrativeSource* implemented as *BasicAdministrationUnitEvent*
- Association *LA_BAUnit <-> LA_SpatialUnit* implemented as *SpatialUnitForBasicAdministrativeUnit*

(*Version of Diagram: 25-3-2011*)

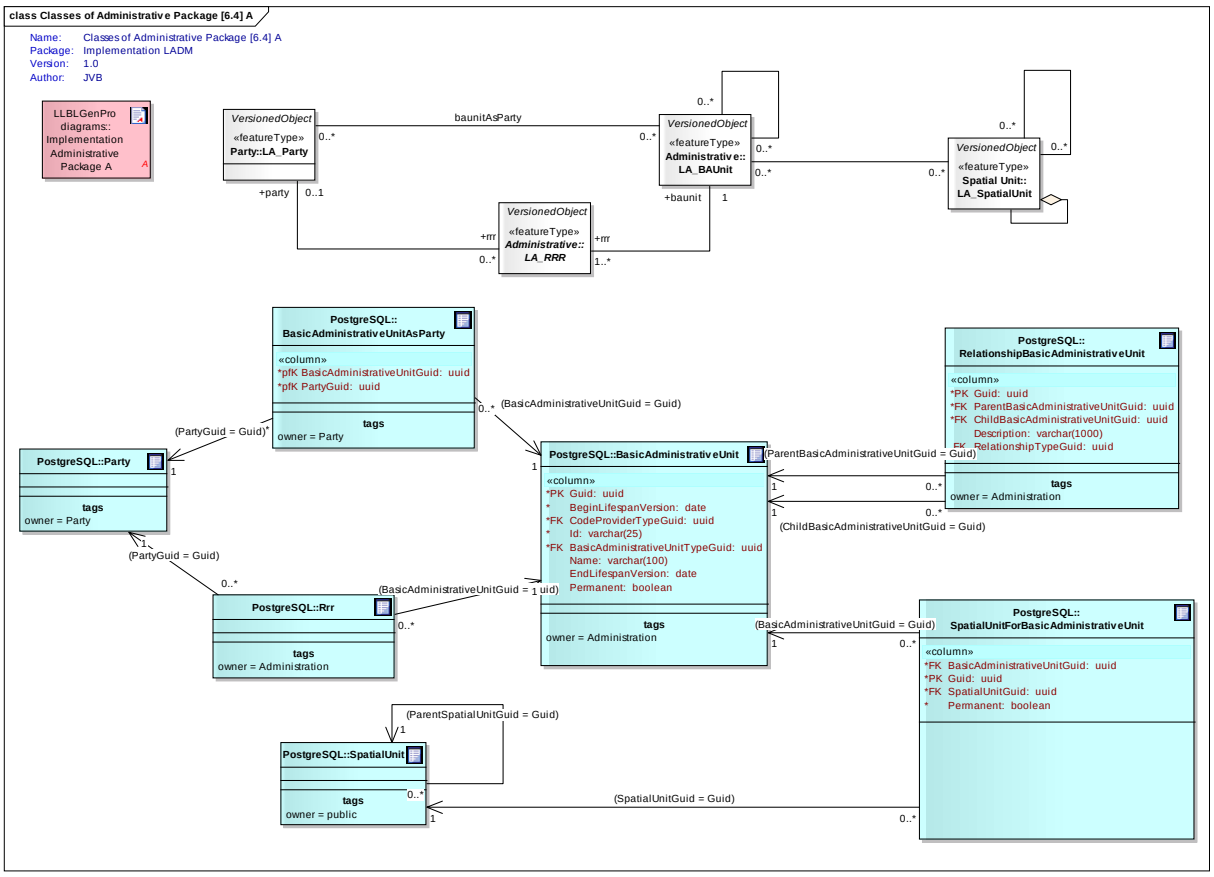


Figure: 7 : Classes of Administrative Package [6.4] A

Diagram Classes of Administrative Package [6.4] B

The classes with regard to LA_RRR are implemented as follows:

- Association *LA_Mortgage* <-> *LA_Party* implemented as *PartyRole.RrrParty*
- The ˜shareCheck" attribute of class LA_RRR (see 6.4.2) indicates wether 'share' is meaningless with regard to the type of right, restriction or responsibility. This has been implemented as attribute of RightType, RestrictionType, ResponsibilityType, instead of an attribute of Rrr.
- Attribute *LA_RRR. timeSpec* (ISO8601_Type) [0..1] is capable of handling other temporal descriptions, such as recurring patterns (every week-end, every summer, etc.). This means, for example, that a party can hold a right to use an apartment each year in March, or that a group of pastoralists has the right to cross a field each summer (for fuzzy time range specifications. This attribute has been implemented as a description/String (*Rrr.TimeSpecification*), but could be implemented as a special datatype or group of related classes to store the TimeSpecification. N.B.
- TO DO: a more structure implementation of Attribute *LA_RRR. timeSpec* (ISO8601_Type)
- TO DO: The constraint that no overlap is allowed between timeSpecs for the same RRR type and the same basic administrative unit.
- TO DO: constraints with regard to *share* and *shareCheck, and Numerator & Denominator*
- TO DO: constraints with regard to Mortgage.Amount (CurrencyType and value); both are filled or both are empty, as well as disabling CurrencyType?
- The association between *LA_Right* and *LA_Mortgage* has been implemented through *RightMortgage*; *RightMortgage.Sequence* implements the "ordered" attribute of the association.

- Attribute *LA_RRR.partyRequired* indicates whether a party is required for the registration of the restriction in the association to LA_Party. *PartyRequired* is set to TRUE (by default), if for the registration of the restriction a party is required, and to FALSE, if the restriction is considered as an spatial unit restriction. The spatial unit restriction is always via a baunit. The Attribute *LA_RRR.partyRequired* is implemented as an attribute *RestrictionType.PartyRequired*)

(*Version of Diagram: 25-3-2011*)

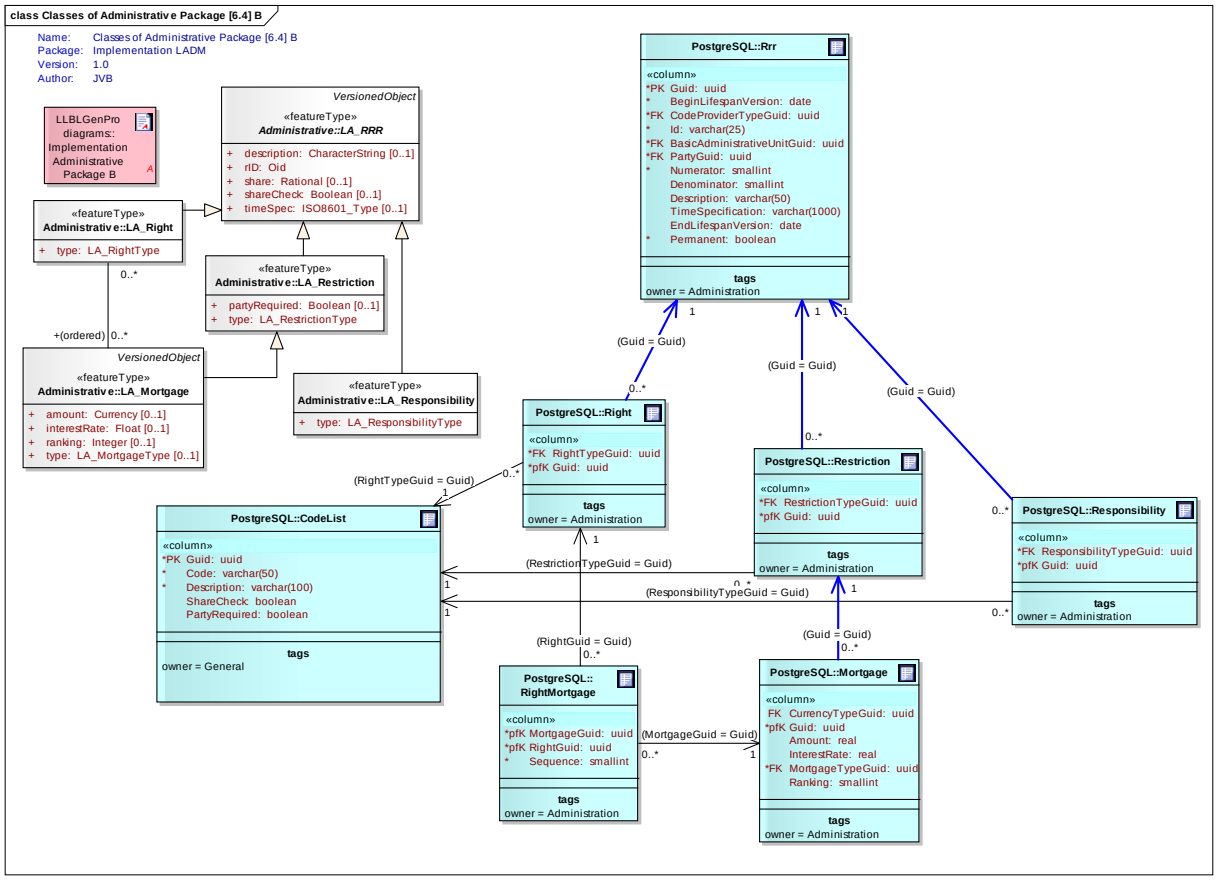


Figure: 8 : Classes of Administrative Package [6.4] B

Diagram Source [6.2.2]

Source Documents

The source documents (~events / instruments) are modelled through the classes *LA_Source*, *LA_SpatialSource*, *LA_AdministrativeSource*. The definition of the types of source documents (consider *LA_SpatialSourceType* and *LA_AdministrativeSourceType*) may vary considerably (in the detail recorded) even within national boundaries; with all this possible current and future variation it is unlikely that a single definition will suffice for even the simplest class of source documents (See comments in OSCAR related doc's [http://source.otago.ac.nz/oscar/Osca_home]). Therefore an approach is recommended which leads to a situation where system administrators can define any number of source document types with any combination of fields and(date)types without the need to modify the source code of a land administration system.

The classes with regard to source documents are implemented as follows:

- *LA_SpatialSourceType* and *LA_AdministrativeSourceType* are implemented through one class *SourceType*, the expected content of these source types is stored in *SourceType.XmlContentDefinition (sometimes referred to as "**standard application forms**")*.
- *LA_Source.submission* and *lifeSpanStamp* are implemented through *Source.Submission*; the date of submission

of the source by a party to the registering authority & the moment that the event represented by the instance of *LA_Source* is further processed in the land administration system (the actual start of a service).

- *LA_Source.acceptance* are implemented through *Source.Acceptance*; the date of force of law of the source by an authority, the actual date of the source document (signature/stamp).
- *LA_Source.sID* and *LA_Source.extArchiveID* are implemented through *CodeProviderType[Guid]* and *Id*. The assumption is that either the source is uniquely identified by an *extArchiveID* or by an internal *sID*.
- *LA_AdministrativeSource.text* is implemented through *Source.XmlContent*.
- TO DO: Further elaborate on the XML definitions in columns *Source.XmlContent & SourceType.XmlContentDefinition*.

The following attributes are maintained in the "external" description *SourceType.XmlContentDefinition*.

- LA_Source.recordation
- LA_Source.maintype
- LA_AdministrativeSource.availibilityStatus
- LA_SpatialSource.measurements
- LA_SpatialSource.procedure

Service & Work
LADM does not contain classes with regard to the applications by right holders and the land administrations s*ervices* and *work* that are being performed.The classes *Service* and *ServiceWork* have been added, where *ServiceWork* has an association with the actual *Source* document that is being provided for a certain application.

- TO DO: Further elaborate on these Service related classes

Events (from source documents)
The *source* documents describe events that result in the creation, update or archive of a certain instance. The events are associated with the *ServiceWork*. For the core elements a specialisation of class *Event* was defined (*RrrEvent*, *BasicAdministrationUnitEvent*, *SpatialUnitEvent*, etc) and these form the implementation of the many:many association between *LA_Source*, *LA_SpatialSource*, *LA_AdministrativeSource* and core class *LA_RRR*, *LA_BAUnit* etc.This will enable automatic handling of core objects within a process/service, and can be used for derivation of history between *BasicAdministrativeUnits* and *SpatialUnits*. On behalf of this (automatic) processing, indicating the temporary nature of the new instances, an attribute *Permanent* (boolean, default false) has been introduced for the core classes. (*Version of Diagram: 25-3-2011*)

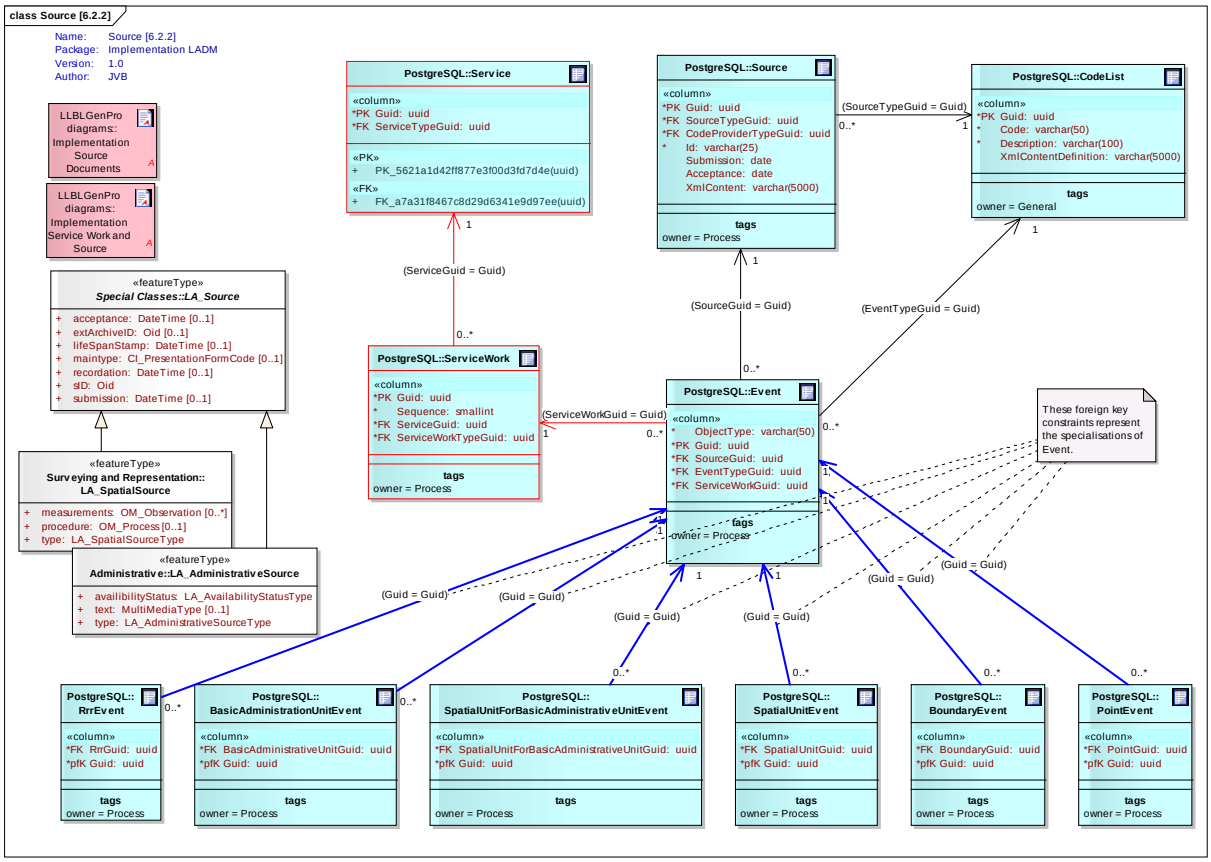


Figure: 9 : Source [6.2.2]

Diagram SpatialUnit [6.5.1]

The *LA_SpatialUnit* is implemented as follows:

- *LA_SpatialUnit.extAddressID* is implemented through *SpatialUnitContactMechanism*.
- *LA_SpatialUnit.suID* is implemented through *CodeProviderType[Guid] & Id*
- In relation to the *LA_SpatialUnit.suID*, the class *SpatialUnitCode* allows for storing multiple unique (external) identifications for *SpatialUnit*.
- *LA_SpatialUnit.area* and *volume[0..*]* are implemented through *SpatialUnitOfMeasure (see Diagram UnitOfMeasure).*
- TO DO: methods *AreaClosed, volumeClosed, computeArea, computeVolume, createArea, createVolume*
- The aggregation association for *SpatialUnit* has been implemented through a self referencing foreign key *ParentSpatialUnitGuid*.
- *LA_RequiredRelationshipSpatialUnit* is implemented as *RelationshipSpatialUnit & RelationshipType (e.g. required), see also LA_RequiredRelationshipBAUnit.* This enable more flexibly to register recursive associations between classes.
- *LA_RequiredRelationshipSpatialUnit.relationship* implemented as *RelationshipSpatialUnit.Description*
- The aggregation association for *SpatialUnitGroup* has been implemented through a self referencing foreign key *ParentSpatialUnitGroupGuid*.
- *LA_LegalSpaceBuildingUnit.buildingUnitId & LA_LegalSpaceUtilityNetwork.extPhysicalNetworkID are* implemented through inheritance of *SpatialUnit.CodeProviderType[Guid]Id and class SpatialUnitCode*
- TO DO: *SpatialUnit.Dimension* could be derived from *Boundary.Face* or *Boundary.FaceString.*

(*Version of Diagram: 25-3-2011*)

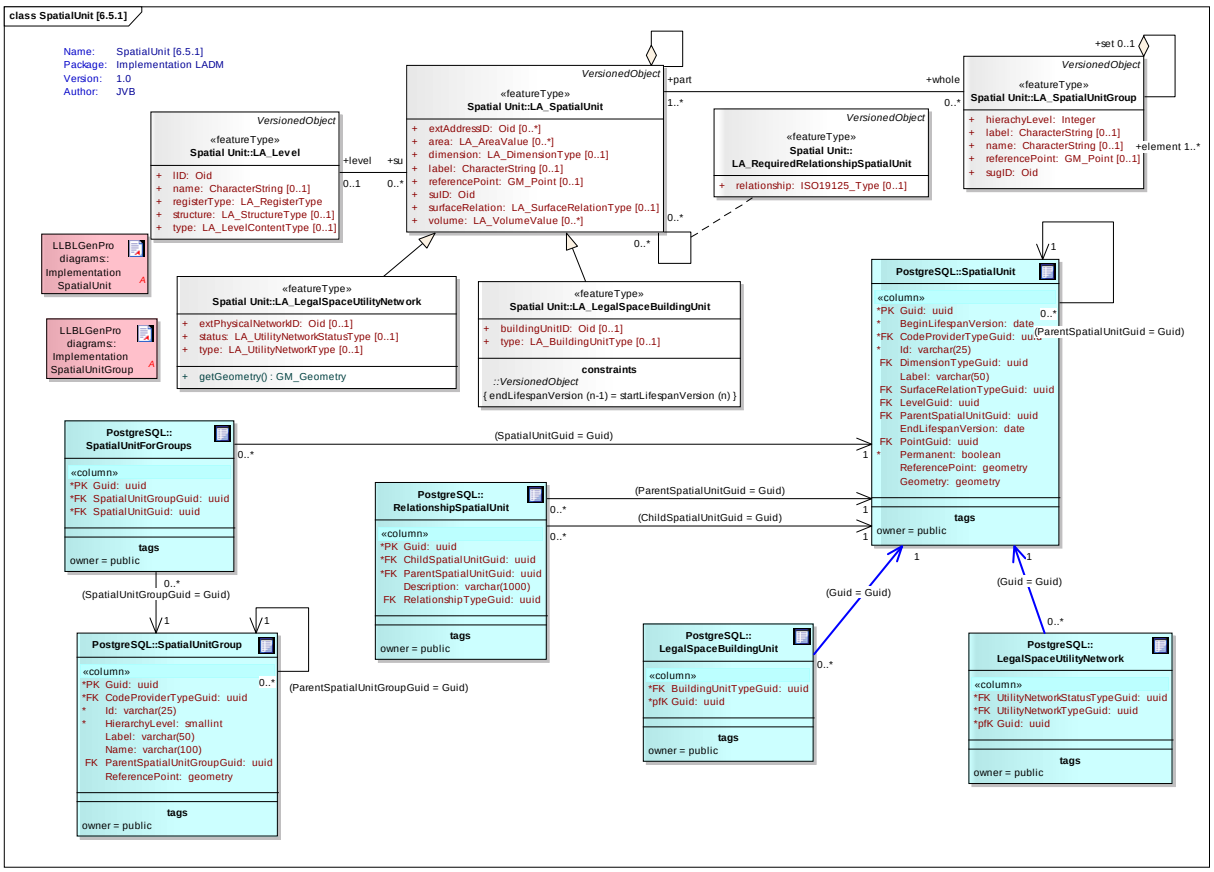


Figure: 10 : SpatialUnit [6.5.1]

Diagram UnitOfMeasure

On several occasions, the LADM uses some form of Measurement Unit to describe the characteristics of an object. For example for a *SpatialUnit*, multiple values can be recorded for Calculated area, legal area, official volume, etc; the class *Point* registers an *EstimatedAccuracy* [Length]. These values of a certain kind of measurement unit are implemented through *PointUnitOfMeasure, SpatialUnitOfMeasure*

- Class *MeasureType* contains instances of e.g. Area, Length, Angle, Time, Velocity, Volume, Scale, Weight, Count.
- Class *UnitOfMeasure* containts instances of e.g. Meter, Feet, Square Meter, Cubic Meter
- Class *SpatialUnitUomValueType* implements the classes *LA_AreaType, LA_VolumeType c*lass, e.g. official[legal], nonOfficial, calculated, surveyed (*PointUomValueType:* estimatedAccuracy)
- *PointUnitOfMeasure* and *SpatialUnitOfMeasure* contain the actual values (implementing LA_SpatialUnit.area & volume as well as , *LA_Point.estimatedAccuracy*.
- TO DO: Constraints with regard of preventing (or allowing) multiple version for the same characterisitc, e.g. muliple values for Point.EstimatedAccuracy?

(*Version of Diagram: 25-3-2011*)

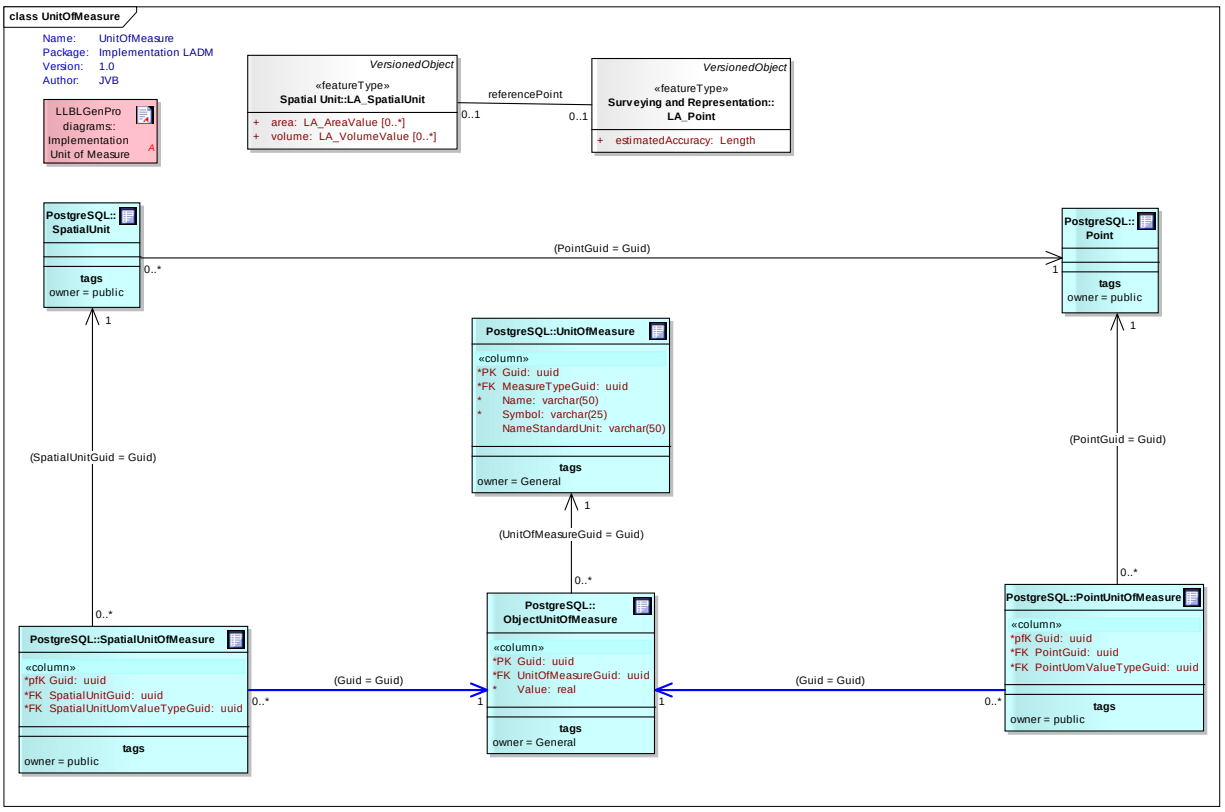


Figure: 11 : UnitOfMeasure

Diagram Surveying [6.6]

The classes of the Surveying Package are implemented as follows

- *LA_Point* is implemented as class *Point*.
- The [0..1] association between *LA_Point* and *LA_SpatialUnit* has been implemented as a Foreign Key from *SpatialUnit* to *Point*.
- TO DO*: GetTransResult()*
- The derived attribute LA*_Point.productionMethod* is implemented through *PointQuality.Description*.
- *LA_Point.transAndResult* [transformation & transformedLocation] [0..*] is implemented through *PointQuality.Description* and *Point.TransformedLocation*.
- *LA_BoundaryFace* (for 3D boundary representations) is implemented as *Boundary* and specialisation *Face*.
- *LA_BoundaryFaceString* (for 2D boundary representations) is implemented as *Boundary* and specialisation *FaceString*.
- The associations between *LA_BoundaryFace*, *LA_BoundaryFaceString*, *LA_SpatialUnit* and *LA_Point* are implemented as class *SpatialUnitBoundary* and *BoundaryPoint*.
- The association attribute *plus* or *minus* is implemented as *SpatialUnitBoundary.Plus* [boolean]
- TO DO: Constraint: for polygon-based (2D) or polyhedron-based (3D) *SpatialUnits*: no **minus** and at least one **plus** [*LA_BoundaryFace(String)*], for *topology*-based *SpatialUnits*: at least one **plus** or **minus**.

(*Version of Diagram: 25-3-2011*)

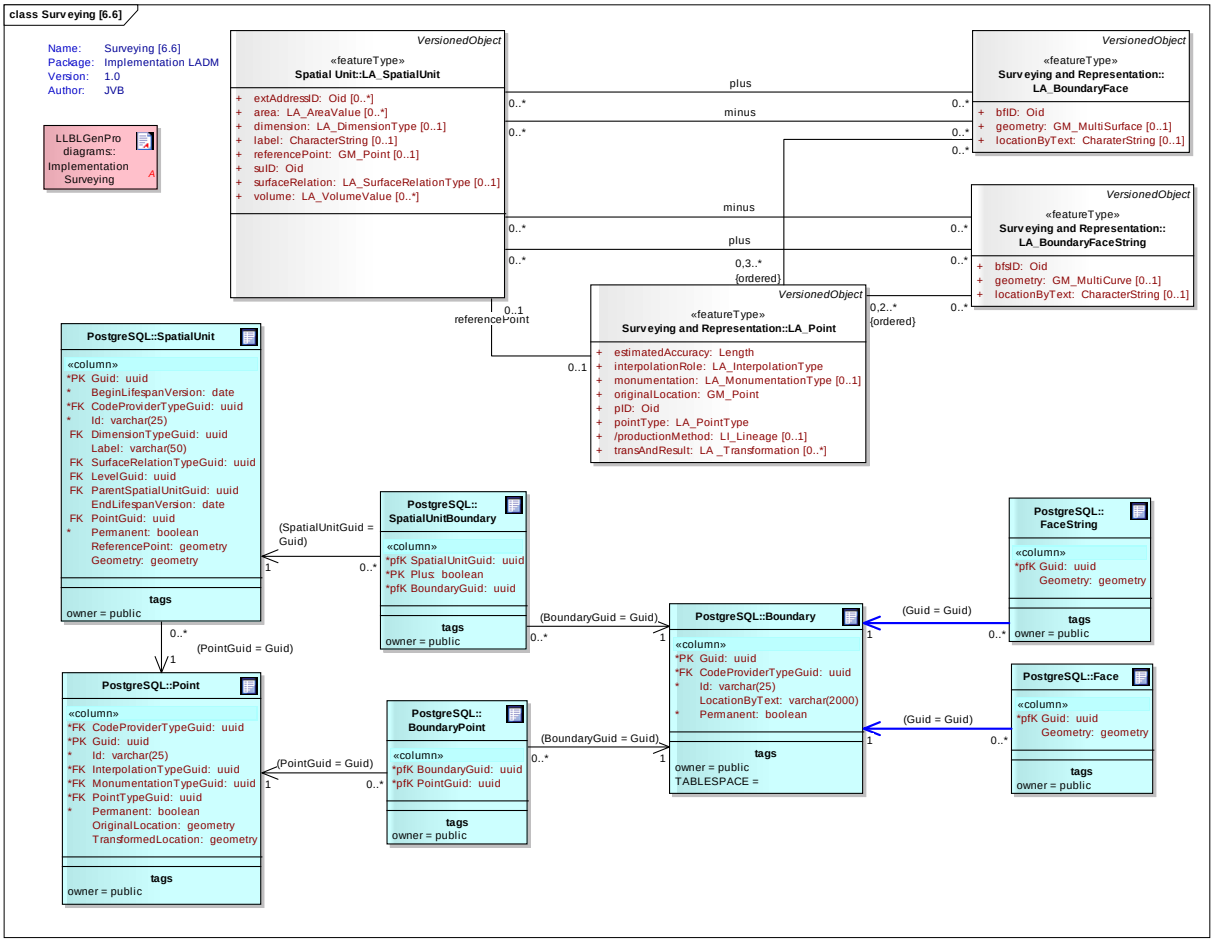


Figure: 12 : Surveying [6.6]

Diagram External Classes [Annex K]

The external classes are implemented as follows:

- *ExtValuation*, *ExtTaxation*, *ExtLandUse* are implemented under similar names
- *ExtCoverageType* is implemented as *ExtLandCoverType* (consistancy in terminology)
- Class *ExtArchive* is not implemented, all attributes can be recorded through *Source*.
- *ExtParty* and *ExtAddress* are implemented through *ExtParty*, *NonNaturalPerson*, *NaturalPerson*, see Diagram for *ExtParty* and *ExtAddresses* [K.2 K.3]
- TO DO: *ExtPhysicalUtilityNetwork* and elements *ExtNetworkSegment* & *ExtNetworkNode*
- Class ExtLegalBuffer is encountered in EnterPrise Architect file, but not in PDF version of LADM, will not be implemented.
- TO DO: Constraint with regard to CurrencyType and Value.

(*Version of Diagram: 25-3-2011*)

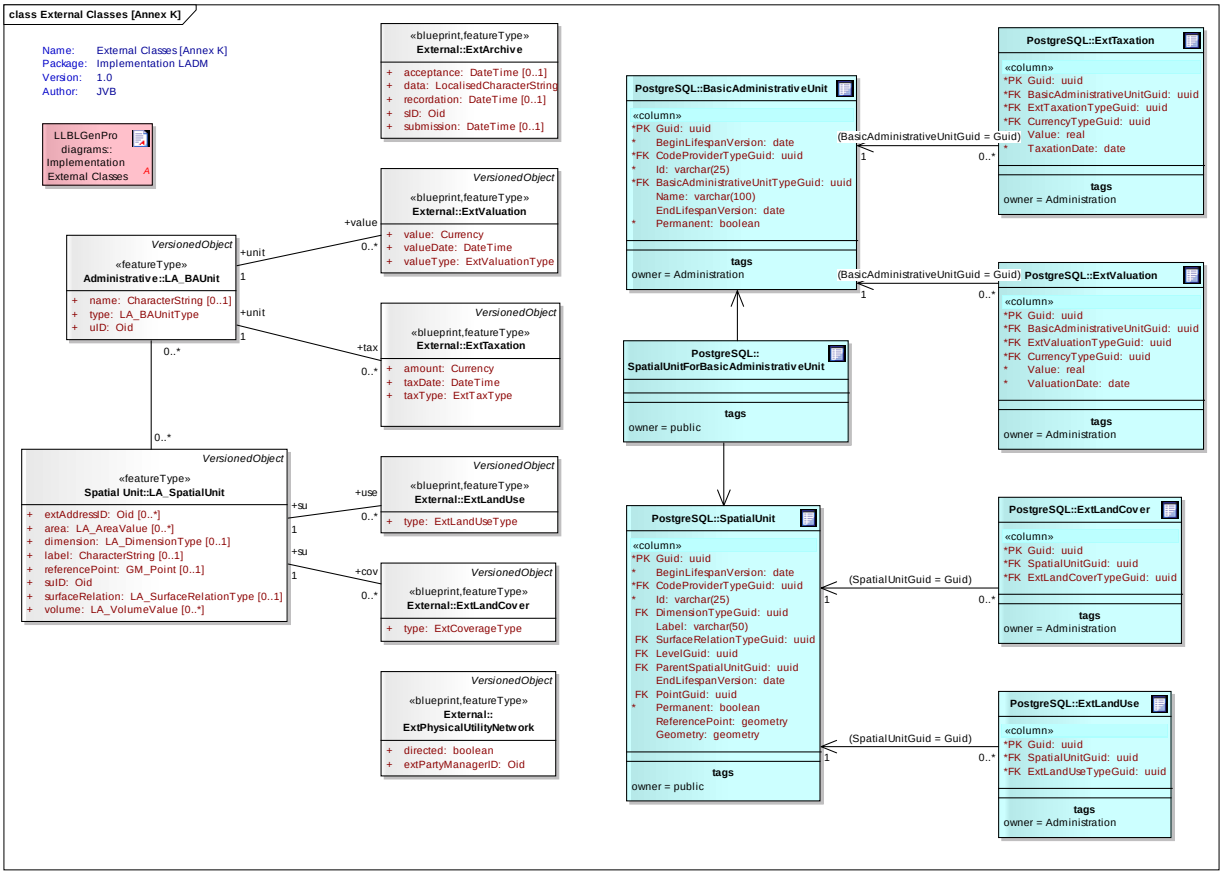


Figure: 13 : External Classes [Annex K]

1.1 PostgreSQL

The tables generated for the PostgreSQL/PostGIS rdbms are described in the following sections.
(*Version of Package: 25-3-2011*)

1.2 Class «table»: BasicAdministrationUnitEvent

The event for a **basic administrative unit**, a specialisation of **event** to describe the relation with **source** documents.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
BasicAdministrativeUnitGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.3 Class «table»: BasicAdministrativeUnit

An administrative entity consisting of zero or more **spatial units** against which (one or more) unique and homogeneous **rights** (e.g. ownership right or land use right), **responsibilities** or **restrictions** are associated to the whole entity, as included in a **Land Administration** system.

For example:

- A condominium unit with two spatial units (e.g. an apartment and a garage), or a farm lot made of one spatial unit (e.g. parcel of land), a servitude made of one spatial unit (e.g. the road representing the right-of-way), or a land consolidation area.

Notes:

- By unique is meant that a right, or restriction, or responsibility is held by one, or several parties (e.g. owners or users) for the whole basic administrative unit. By homogeneous is meant that a right, or restriction, or responsibility (e.g. ownership, use, social tenure, lease, or easement) affects the whole basic administrative unit. For a restriction zero parties are possible.
- An basic administrative unit may play the role of party.
- A basic administrative unit should get an unique identifier when registered, or recorded.
- Basic administrative units are needed, among other things, to register ‘basic property units’, which consist of several spatial units, belonging to a party, under the same right (a right must be 'homogeneous' over the whole baunit). There must be a unique right for each basic administrative unit in order to establish a unique combination between an instance of a Party, an instance of a subclass of Rrr, and an instance of basic administrative unit.
- basic administrative unit allows the association of one right to a combination of spatial units (e.g. an apartment and a parking place).
- A constraint states that, for one basic administrative unit, the sum of all the shares must be equal to 1 for the same subclass of class Rrr, unless ‘share’ is meaningless with regard to the type of right, restriction or responsibility. This is indicated by the ‘shareCheck’ attribute of class RightType, RestrictionType, RespnsibilityType.
- It is possible that no spatial unit exists for a basic administrative unit, thus allowing for the support of special administrative situations (e.g. deeds registration without mapping).
- With class basic administrative unit it is possible to register spatial units from different levels as one unit. If (parts of) spatial units are included, or eliminated from the basic administrative unit, the ID stays the same, with a different version. In this approach, a mortgage can only be established on the complete basic administrative

unit, not on one or more of the registered spatial units.
- A (group of) basic administrative units may be a party.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
BeginLifespanVersion	date	True	False	Start time of a specific instance version.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
BasicAdministrativeUnitTypeGuid	uuid	True	False	The type of the basic administrative unit.
Name	varchar	False	False	The name of the basic administrative unit.
EndLifespanVersion	date	False	False	End time of a specific instance version.
Permanent	boolean	True	False	Indicates the status of the instance (initially temporary ~ Permanent = false).

1.4 Class «table»: BasicAdministrativeUnitAsParty

The implementation of the many-to-many association between a **basic administrative unit** that acts as a **party**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
BasicAdministrativeUnitGuid	uuid	True	True	
PartyGuid	uuid	True	True	

1.5 Class «table»: Boundary

A set that represents the limit of an entity [ISO 19107:2003]

Notes:
- LADM supports the increasing use of 3D representations of spatial units, without putting an additional burden on the existing 2D representations. Another feature of the spatial representation within LADM is that there is no mismatch between spatial units that are represented in 2D and spatial units that are represented in 3D. Furthermore, LADM is based on accepted and available spatial schemata, such as published in ISO 19107:2003.
- The geometry is either derived from associated class Point, or based on captured linear/surface geometry.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.

CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
LocationByText	varchar	False	False	The boundary represented in text.
Permanent	boolean	True	False	

1.6 Class «table»: BoundaryEvent

The event for a **boundary (face or facestring)**, a specialisation of **event** to describe the relation with **source** document.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
BoundaryGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.7 Class «table»: BoundaryPoint

The **points** for **boundaries**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
BoundaryGuid	uuid	True	True	
PointGuid	uuid	True	True	

1.8 Class «table»: CodeList

The values for all **code lists** (incl. enumerations) within LADM.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ObjectType	varchar	True	False	
Guid	uuid	True	True	The system global unique identifier.
Code	varchar	True	False	
Description	varchar	True	False	
ShareCheck	boolean	False	False	Indicates whether the constraint on nominator and denominator in class Rrr and PartyMember is applicable.
PartyRequired	boolean	False	False	Indicates whether a party is required for the registration of the restriction in the association to Party.

XmlContentDefinition	varchar	False	False	

1.9 Class «table»: ContactMechanism

The **contact mechanisms** (addresses, e-mail, phone numbers, etc) that are used, shared and allocated to various LADM objects (e.g. **parties, spatial units**).

Notes:

- Some of these contact mechanisme may use/link to SpatialUnitGroups.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ObjectType	varchar	True	False	
Guid	uuid	True	True	The system global unique identifier.
BeginLifespanVersion	date	True	False	Start time of a specific instance version.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
EndLifespanVersion	date	False	False	End time of a specific instance version.
Country	varchar	False	False	
ProvinceState	varchar	False	False	
City	varchar	False	False	
AreaName	varchar	False	False	The address area name of the address.
ZipPostalCode	varchar	False	False	
PostBox	varchar	False	False	
StreetName	varchar	False	False	
StreetNumber	varchar	False	False	
BuildingName	varchar	False	False	
BuildingNumber	varchar	False	False	
CountryCode	smallint	False	False	
AreaCode	smallint	False	False	
Number	smallint	False	False	
Extension	smallint	False	False	
Coordinate	geometry	False	False	The coordinate of the address.

1.10 Class «table»: Event

The **event** that connects core elements (e.g. basic administrative unit, spatial unit) to **source**s.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ObjectType	varchar	True	False	
Guid	uuid	True	True	The system global unique identifier.
SourceGuid	uuid	True	False	
EventTypeGuid	uuid	True	False	
ServiceWorkGuid	uuid	True	False	

1.11 Class «table»: ExtLandCover

A class for the external registration of land cover data; land cover is the observed (bio)physical cover on the earth's surface.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
SpatialUnitGuid	uuid	True	False	
ExtLandCoverTypeGuid	uuid	True	False	

1.12 Class «table»: ExtLandUse

A class for the registration of land use data; land use is an arrangement, activity or input people undertake in certain land cover type, to produce, change or maintain it.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
SpatialUnitGuid	uuid	True	False	
ExtLandUseTypeGuid	uuid	True	False	

1.13 Class «table»: ExtParty

A class for an external registration of parties, to be used in LADM parties.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ObjectType	varchar	True	False	
Guid	uuid	True	True	The system global unique identifier.
BeginLifespanVersion	date	True	False	Start time of a specific instance version.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
Photo	bytea	False	False	The photo (digital) of the party.
EndLifespanVersion	date	False	False	End time of a specific instance version.
Prefix	varchar	False	False	
Suffix	varchar	False	False	
FirstName	varchar	False	False	
MiddleName	varchar	False	False	
Name	varchar	True	False	The (last) name of the party
GenderTypeGuid	uuid	False	False	The type of gender of the party.
Birthdate	date	False	False	
Signature	bytea	False	False	The signature (digital) of the party.
FingerPrint	bytea	False	False	The fingerprint (digital) of the party.
Profession	varchar	False	False	

1.14 Class «table»: ExtPartyContactMechanism

The **contact mechanism**s (addresses, e-mail, phone numbers, etc) for a **party**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ExtPartiesGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.15 Class «table»: ExtTaxation

A class for the registration of taxation data.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
BasicAdministrativeUnitGuid	uuid	True	False	

ExtTaxationTypeGuid	uuid	True	False	
CurrencyTypeGuid	uuid	True	False	
Value	real	True	False	The amount of taxation.
TaxationDate	date	True	False	The date of taxation.

1.16 Class «table»: ExtValuation

A class for the registration of valuation data.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
BasicAdministrativeUnitGuid	uuid	True	False	
ExtValuationTypeGuid	uuid	True	False	
CurrencyTypeGuid	uuid	True	False	
Value	real	True	False	
ValuationDate	date	True	False	

1.17 Class «table»: Face

A **face** that is used in the 3-dimensional representation of a **boundary** of a **spatial unit**.

For Example:

- Boundary faces are used when the implied vertical and unbounded faces of a boundary face string are not sufficient to describe 3D spatial units. Boundary faces close volumes in height (e.g. every apartment floor), or in depth (e.g. an underground parking garage), or in all other directions to form a bounded volume. The volumes represent legal space (in contrast with physical space).

Notes:

- 3D representations of spatial units use *boundary faces* class Face.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier (from super class).
Geometry	geometry	False	False	The boundary represented via a surface in 3D.

1.18 Class «table»: FaceString

A **boundary** forming part of the outside of a **spatial unit**.

Notes:
- Boundary face strings are used to represent the boundaries of spatial units via linestrings in 2D. This 2D representation implies in a 2D land administration system a 2D boundary, or in a 3D land administration system a series of vertical boundary faces. In that case an unbounded volume is assumed, surrounded by boundary faces, which intersect the earth's surface (such as traditionally depicted in the cadastral map).

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier (from super class).
Geometry	geometry	False	False	The boundary represented via a curve at ground level (2D).

1.19 Class «table»: GroupParty

Any number of **parties**, forming together a distinct entity, with each **party** registered.

For example:
- A partnership (with each partner registered as a party), or two tribes (with each tribe registered as a party).

Notes
- A group party may be a party member of another group party.
- There is a constraint stating, that the sum of the shares of the group party members, equals to 1. This constraint is only enforced, if there exists a class PartyMember.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
GroupPartyTypeGuid	uuid	True	False	The type of a group party.
Guid	uuid	True	True	The system global unique identifier (from super class).

1.20 Class «table»: LegalSpaceBuildingUnit

A component of **building** (the legal, recorded or informal space of the physical entity).

For example:
- An apartment, a stairs, a threshold, a garage, a parking place, or a laundry space.

Notes:
- A building unit is for different purposes (e.g. living or commercial), or it can be a construction work.
- A building unit concerns *legal space*, which does not necessarily coincide with the physical space of a building.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
BuildingUnitTypeGuid	uuid	True	False	The type of the building unit.
Guid	uuid	True	True	The system global unique identifier (from super class).

1.21 Class «table»: LegalSpaceUtilityNetwork

A **network** describing the topology of a utility.

For example:
- The legal space needed to access and to keep in repair a cable or pipeline utility network.

Notes:
- A utility network may be attributed with information about its legal, recorded or informal space.
- A utility network can also be modelled as a basic administrative unit.
- A utility network concerns *legal space*, which does not necessarily coincide with the physical space of a utility network.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
UtilityNetworkStatusTypeGuid	uuid	True	False	The status of the utility network.
UtilityNetworkTypeGuid	uuid	True	False	The type of the utility network.
Guid	uuid	True	True	The system global unique identifier (from super class).

1.22 Class «table»: Level

A set of **spatial units**, with a geometric, and/or topological, and/or thematic coherence.

For example:
- One level for an urban cadastre and another level for a rural cadastre.
- One level with rights and another level with restrictions.
- One level with formal rights, a second level with informal rights and a third level with customary rights.
- One level with point based spatial units, a second level with line based spatial units, and a third level with polygon based spatial units.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
RegisterTypeGuid	uuid	True	False	The register type of the content of the level.
Name	varchar	False	False	The name of the level.
LevelContentTypeGuid	uuid	True	False	The type of the content of the level.

StructureTypeGuid	uuid	True	False	The structure of the level geometry.

1.23 Class «table»: Mortgage

A ***mortgage*** is a special restriction of the ownership **right**. It concerns the conveyance of a property by a debtor to a creditor, as a security for a financial loan, with the condition that the property is returned, when the loan is paid off. A **mortgage** is associated to class **Right** (the right that is the basis for the mortgage).

Notes:
- ISO 4217 should be used for the list of currencies in the ISO/TS 19103 measure.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
CurrencyTypeGuid	uuid	False	False	
Guid	uuid	True	True	The system global unique identifier (from super class).
Amount	real	False	False	The amount of money of the mortgage.
InterestRate	real	False	False	Interest rate of the mortgage (percentage).
MortgageTypeGuid	uuid	True	False	The type of the mortgage.
Ranking	smallint	False	False	The ranking order, if more than one mortgage applies to a right (or rights).

1.24 Class «table»: ObjectContactMechanism

The contact mechanisms (addresses, e-mail, phone numbers, etc) for a Object (super class).
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
ContactMechanismGuid	uuid	True	False	
ContactMechanismTypeGuid	uuid	True	False	

1.25 Class «table»: ObjectQuality

To manage and maintain historical data in the database with regard to the *Quality* of a specific instance version
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ObjectType	varchar	True	False	

Guid	uuid	True	True	The system global unique identifier.
Description	varchar	True	False	
NameMeasure	varchar	False	False	
Date	date	False	False	
ObjectGuid	uuid	True	False	

1.26 Class «table»: ObjectSource

To manage and maintain historical data in the database with regard to the responsible organization of a specific instance version.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ObjectType	varchar	True	False	
Guid	uuid	True	True	The system global unique identifier.
SourcePartyGuid	uuid	True	False	
SourcePartyRoleTypeGuid	uuid	True	False	
ObjectGuid	uuid	True	False	

1.27 Class «table»: ObjectUnitOfMeasure

The super class of SpatialUnitOfMeasure and PointUnitOfMeasure.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
UnitOfMeasureGuid	uuid	True	False	
Value	real	True	False	

1.28 Class «table»: Party

A person or organisation that plays a role in a **rights** transaction [ISO/CD 19153].

For example:

- A juridical person may be: a company, a municipality, the state, a tribe, a farmer cooperation, or a church community (with each juridical person represented by a delegate: a director, chief, CEO, etc.).

Notes:

- In order to be registered as a party not all members need to be identified and registered individually.

- A basic administrative unit may be a party because it may hold a right of e.g. easement.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
BeginLifespanVersion	date	True	False	Start time of a specific instance version.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
PartyTypeGuid	uuid	True	False	The type of the party.
Name	varchar	False	False	The name of the party.
EndLifespanVersion	date	False	False	End time of a specific instance version.

1.29 Class «table»: PartyCode

Multiple unique identifications (by different code providers) for party.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
PartyGuid	uuid	True	True	
CodeProviderTypeGuid	uuid	True	True	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	True	The identifier, unique within the specified Code Provider (namespace).

1.30 Class «table»: PartyContactMechanism

The contact mechanisms (addresses, e-mail, phone numbers, etc) for a Party.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier (from super class).
PartyGuid	uuid	True	False	

1.31 Class «table»: PartyMember

A **party** registered and identified as a constituent of a **group party.**

Notes:

- PartyMember supports rational numbers (fractions); e.g. ½ or ¾. A fraction is written as a pair of numbers, the top number called the numerator and the bottom number called the denominator.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
GroupPartyGuid	uuid	True	True	
PartyGuid	uuid	True	True	The type of the party.
Numerator	smallint	False	False	The top number in the notation of a fraction.
Denominator	smallint	False	False	The bottom number in the notation of a fraction.

1.32 Class «table»: PartyRole

The role of the party in the data update and maintenance process.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
PartyRoleTypeGuid	uuid	True	False	
Description	varchar	False	False	

1.33 Class «table»: Point

A 0-dimensional geometric primitive, representing a position [ISO 19107:2003].

Notes:
- A point may be used to define one or more boundary faces or boundary face strings.
- Points can be observed by e.g. terrestrial surveying, but also by photo interpretation, image interpretation, or identification on an existing map.
- *Points* can be acquired in the field (with classical surveys, or with satellite navigation systems), in an office, or compiled from various sources, for example using forms, field sketches, ortho-images, or orthophotos. The acquisition of points (a survey) may concern the identification of spatial units on a photograph, on an image, or on a topographic map; cycloramas or pictometry methods (multiple images from different angles) may also be used for that purpose.
- Coordinates themselves either come from *points*, or are captured as linear geometry. Spatial units may share the same representation structure: existing 2D data, whether topologically structured or not, or polygons, or unstructured boundaries, or simply point or textual descriptions, can be included.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.

Guid	uuid	True	True	The system global unique identifier.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
InterpolationTypeGuid	uuid	True	False	The role of point in the structure of a straight line or curve.
MonumentationTypeGuid	uuid	True	False	The type of monumentation.
PointTypeGuid	uuid	True	False	The type of point.
Permanent	boolean	True	False	
OriginalLocation	geometry	False	False	The calculated co-ordinates, based on measurements and observations.
TransformedLocation	geometry	False	False	The transformed location.

1.34 Class «table»: PointEvent

The event for a **point**, a specialisation of Event to describe the relation with Source.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
PointGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.35 Class «table»: PointUnitOfMeasure

Extra characteristics and values for a **point** (e.g. estimated accuracy).
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier (from super class).
PointGuid	uuid	True	False	
PointUomValueTypeGuid	uuid	True	False	

1.36 Class «table»: RelationshipBasicAdministrativeUnit

A explicit association between **basic administrative units.**

Notes:

- RelationshipBasicAdministrativeUnit allows creating instances of relationships between basic administrative units. It allows maintaining explicit spatial relationships in the absence of spatial units to describe the basic administrative units, or in the presence of inaccurate geometry of spatial units to generate reliable implicit spatial relationships.
- Instances of RelationshipBasicAdministrativeUnit override implicit relationships, established through

geospatial overlaying techniques.

- Even if the geometry of spatial units is accurate, there may be legal reasons to establish required relationships between basic administrative units.
- TO DO: RelationshipBasicAdministrativeUnit is a versioned object class. Different life cycle attributes than the versioned object ones, can be added using the attribute 'relationship'.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
ParentBasicAdministrativeUnitGuid	uuid	True	False	
ChildBasicAdministrativeUnitGuid	uuid	True	False	
Description	varchar	False	False	The description of the required relationship.
RelationshipTypeGuid	uuid	False	False	

1.37 Class «table»: RelationshipSpatialUnit

A explicit association between **spatial units.**

Notes:

- Due to inaccurate geometries, or missing geometries, geospatial overlay techniques may generate invalid, or no relationships between spatial units, which can be introduced by required relationships.
- Relationships for spatial units may be defined with ISO 19125-2 types.
- *Required relationships* are explicit links between spatial units. Sometimes there is a need for these links, when the geometry of the spatial units is not accurate enough to give reliable results, when applying geospatial overlaying techniques (e.g. a building, in reality inside a parcel, is reported to fall outside the parcel; the same applies to the geometry of a right, e.g. an easement). Required relationships override implicit relationships, established through geospatial overlaying techniques.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
ChildSpatialUnitGuid	uuid	True	False	
ParentSpatialUnitGuid	uuid	True	False	
Description	varchar	False	False	The description of the required relationship (e.g. ISO 19125-2 spatial type).
RelationshipTypeGuid	uuid	False	False	

1.38 Class «table»: Responsibility

A formal or informal obligation to do something.

For example:
- The responsibility to clean a ditch, to keep a snow-free pavement or to remove icicles from the roof during winter, or to maintain a monument.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ResponsibilityTypeGuid	uuid	True	False	The type of the responsibility.
Guid	uuid	True	True	The system global unique identifier (from super class).

1.39 Class «table»: Restriction

a formal or informal entitlement to refrain from doing something.

For example:
- It is not allowed to build within 200 metres of a fuel station.
- a servitude or mortgage as a restriction to the ownership right.
- Restrictions usually "run with the land", meaning that they remain valid, even when the right to the land is transferred after the right was created (and registered).

Notes:
- Attribute *RestrictionType.PartyRequired* is set to TRUE (by default), if for the registration of the restriction a party is required, and to FALSE, if the restriction is considered as an spatial unit restriction. The spatial unit restriction is always via a basic administrative unit.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
RestrictionTypeGuid	uuid	True	False	The type of the restriction.
Guid	uuid	True	True	The system global unique identifier (from super class).

1.40 Class «table»: Right

An action, activity or class of actions that a system participant may perform on or using an associated resource [ISO 19132:2007].

For example:
- Ownership right, apartment right, tenancy right, possessions, customary right, Islamic right (e.g. miri or milk), indigenous right, or informal right.

Notes:
- A right may provide a formal or informal entitlement to own or do something.
- this International Standard deals with *real* rights and *personal* rights. Real rights are rights over or in respect of spatial units (e.g. ownership, or usufruct). Personal rights are rights that parties have (e.g. fishing rights, grazing rights, or using rights).

- Rights may be overlapping, or may be in disagreement.
- Rights are primarily in the domain of private or customary law. Ownership rights are generally based on (national) legislation, and code lists in LADM are in support of this.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
RightTypeGuid	uuid	True	False	The type of the right.
Guid	uuid	True	True	The system global unique identifier (from super class).

1.41 Class «table»: RightMortgage

The **mortgages** for a **right**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
MortgageGuid	uuid	True	True	
RightGuid	uuid	True	True	
Sequence	smallint	True	False	

1.42 Class «table»: Rrr

The abstract class Rrr generalises **right**, **responsibility** and **restriction**.

Notes:

- Rrr supports rational numbers (fractions); e.g. ½ or ¾. A fraction is written as a pair of numbers, the top number called the numerator and the bottom number called the denominator.
- An instance of a subclass of Rrr is a right (or social tenure relationship), a restriction, or a responsibility. If it is a right or responsibility, then it is associated to exactly one [1] party, and exactly one [1] basic administrative unit. If it is a restriction, then it is associated to zero or one [0..1] parties, and exactly one [1] basic administrative unit. The latter allows for the registration of restrictions (e.g. right-of-way, right-to-harvest-fruit), with, or without an association to Party.
- There is a constraint in class basic administrative unit, that the sum of all shares is equal to 1, e.g. two parties, each holding a share of ½ in a right of ownership; or one party holding ¼ and another holding ¾.
- Attribute TimeSpecificationis capable of handling other temporal descriptions, such as recurring patterns (every week-end, every summer, etc.).
- There is a constraint that no overlap is allowed between TimeSpecifications for the same Rrr type and the same basic administrative unit.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.

BeginLifespanVersion	date	True	False	Start time of a specific instance version.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
BasicAdministrativeUnitGuid	uuid	True	False	
PartyGuid	uuid	True	False	
Numerator	smallint	True	False	The top number in the notation of a fraction.
Denominator	smallint	False	False	The bottom number in the notation of a fraction.
Description	varchar	False	False	Description regarding the right, restriction or responsibility.
TimeSpecification	varchar	False	False	Operational use of a right in time sharing.
EndLifespanVersion	date	False	False	End time of a specific instance version.
Permanent	boolean	True	False	Indicates the status of the instance (initially temporary ~ Permanent = false).

1.43 Class «table»: RrrEvent

The event for a **rights, restrictions and responsibilitues**, a specialisation of Event to describe the relation with Source.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
RrrGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.44 Class «table»: RrrPartyRole

The **parties** involved in the data update and maintenance process of a **right, restriction or responsibility**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
RrrGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.45 Class «table»: Service

A **land administration** process of determining, recording and disseminating information about the relationship between people and **land**.

Notes:

- In many countries land administration information is determined, recorded and disseminated under the umbrella of cadastre and land registry. Both institutions can be unified in a single (state) organization.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
ServiceTypeGuid	uuid	True	False	

1.46 Class «table»: ServicePartyRole

The **parties** involved in the data update and maintenance process of a **service**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ServiceGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.47 Class «table»: ServiceWork

The **work** performed as part of a **service**, with an association to **events**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
Sequence	smallint	True	False	
ServiceGuid	uuid	True	False	
ServiceWorkTypeGuid	uuid	True	False	

1.48 Class «table»: ServiceWorkPartyRole

The **parties** involved in the data update and maintenance process of **service work**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
ServiceWorkGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.49 Class «table»: Source

source with the administrative description of the **parties** involved, the **rights**, **restrictions** and **responsibilities** created and the **basic administrative units** affected, or the spatial representation of one (part of) or more **spatial units.**

For Example:

- It is the evidence of a party's right to a basic administrative unit.
- A document describing a transaction (a deed), or a judgement of the register holder.
- A field survey sketch, an orthophoto, or a satellite image with evidence on the location of boundaries (collected from the field).

Notes:

- In principle, all rights, restrictions and responsibilities are based on an administrative source, as instances from class Source.
- A survey is documented with *sources*, either final (sometimes formal) documents, or all documents related to a survey. Sometimes, several documents are the result of a single survey. A spatial source may be official, or not (i.e. a registered survey plan, or an aerial photograph). Paper based documents (which may be scanned) can be considered as an integral part of the land administration system.
- A set of measurements with observations (distances, bearings, etc.) of points, is an attribute of Source. The individual points are instances of class Point, which is associated to Source. While it is not required that the complete spatial unit is represented, a spatial source may be associated to several points. Geodetic control points, including multiple sets of coordinates for points, and with multiple reference systems, are all supported in LADM.
- The fact that all different (public or private law) rights find their base in some kind of transacting document is represented by the association between Rrr and Source. The party responsible for drafting the document is connected to the latter as 'conveyer', 'notary', or 'writer' (see PartyRole).
- In some Land Administration systems, sources are needed to perform the transactions, but not archived afterwards. Then the registration itself provides evidence.
- Measurements are the basis for mapping, and for historical reconstruction of the location of (parts of) the spatial unit in the field.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
SourceTypeGuid	uuid	True	False	The type of spatial/administrative source.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
Submission	date	False	False	The date of submission of the source by a party.
Acceptance	date	False	False	The date of force of law of the source by an authority.
XmlContent	varchar	False	False	The content of a source document in xml format.

1.50 Class «table»: SourcePartyRole

The **parties** involved in the data update and maintenance process of a **source** document.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
SourceGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.51 Class «table»: SpatialUnit

A single area (or multiple areas) of land and/or water, or a single volume (or multiple volumes) of space.

Notes:

- A single area is the general case and multiple areas are the exception.
- *Spatial units* are structured in a way to support the creation and management of *basic administrative units*.
- This standard supports either 2-dimensional (2D), 3-dimensional (3D), or mixed (2D and 3D) representations of spatial units, which may be described in text ("from this tree to that river"), or based on a single point, or represented as a set of unstructured lines, or as a surface, or as a 3D volume.
- Independent from spatial units represented with a single point, text, or a set of unstructured lines, a spatial unit may have an area equal to zero for administrative reasons.
- Spatial units may be grouped into two forms: as *spatial unit groups or* as sub spatial units, or *subparcels*, that is a grouping of a spatial unit into its parts. This is realized by an recursive relationship (aggregation) of a SpatialUnit onto itself (ParentSpatialUnitGuid). Parts, in their turn, may be grouped into subparts (*subsubparcels*), and so on.
- Spatial units are refined into two specializations: *building units* (LegalSpaceBuildingUnit) or u*ility networks* (LegalSpaceUtilityNetwork, as instances of class LA_LegalSpaceUtilityNetwork.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
BeginLifespanVersion	date	True	False	Start time of a specific instance version.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
DimensionTypeGuid	uuid	False	False	The dimension of the spatial unit.
Label	varchar	False	False	Short textual description of the spatial unit.
SurfaceRelationTypeGuid	uuid	False	False	
LevelGuid	uuid	False	False	
ParentSpatialUnitGuid	uuid	False	False	
EndLifespanVersion	date	False	False	End time of a specific instance version.
PointGuid	uuid	False	False	

Permanent	boolean	True	False	Indicates the status of the instance (initially temporary ~ Permanent = false).
ReferencePoint	geometry	False	False	The coordinates of a point inside the spatial unit.
Geometry	geometry	False	False	The derived geometry from the spatial unit.

1.52 Class «table»: SpatialUnitBoundary

The boundaries for a spatial unit, the basis for deriving the SpatialUnit.Geometry.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
SpatialUnitGuid	uuid	True	True	
Plus	boolean	True	True	
BoundaryGuid	uuid	True	True	

1.53 Class «table»: SpatialUnitCode

Multiple unique identifications (by different code providers) for spatial unit.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
SpatialUnitGuid	uuid	True	True	
CodeProviderTypeGuid	uuid	True	True	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	True	The identifier, unique within the specified Code Provider (namespace).

1.54 Class «table»: SpatialUnitContactMechanism

The contact mechanisms (addresses, e-mail, phone numbers, etc) for a SpatialUnit.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
SpatialUnitGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.55 Class «table»: SpatialUnitEvent

The event for a **spatial unit**, a specialisation of Event to describe the relation with Source.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
SpatialUnitGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.56 Class «table»: SpatialUnitForBasicAdministrativeUnit

The spatial units as part of a basic administrative unit.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
BasicAdministrativeUnitGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier.
SpatialUnitGuid	uuid	True	False	
Permanent	boolean	True	False	Indicates the status of the instance (initially temporary ~ Permanent = false).

1.57 Class «table»: SpatialUnitForBasicAdministrativeUnitEvent

The event for a the spatial units assigned to a basic administrative unit, a specialisation of Event to describe the relation with Source.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
SpatialUnitForBasicAdministrativeUnitGuid	uuid	True	False	
Guid	uuid	True	True	The system global unique identifier (from super class).

1.58 Class «table»: SpatialUnitForGroups

The **spatial unit**s as part of a **spatial unit group**.
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
SpatialUnitGroupGuid	uuid	True	False	
SpatialUnitGuid	uuid	True	False	

1.59 Class «table»: SpatialUnitGroup

Any number of **spatial units**, considered as an entity.

For example:
- Spatial units forming together an administrative zone such as a section, a canton, a municipality, a department, a province, or a country.
- Spatial units within a planning area.

Notes:
- The highest level in the hierarchy of a subdivision (*country*) is 1; lower levels are incremented by 1.
- The spatial units in a spatial unit group are not necessarily continuous.
- A spatial unit group may be a grouping of other spatial unit groups. In implementations of LADM, this is to enable the inclusion of spatial unit identifiers in hierarchical zones.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
CodeProviderTypeGuid	uuid	True	False	The identifier for the provider of the unique code in *Id*.
Id	varchar	True	False	The identifier, unique within the specified Code Provider (namespace).
HierarchyLevel	smallint	True	False	The level in the hierarchy of an administrative, or zoning subdivision.
Label	varchar	False	False	Short textual description of the spatial unit group.
Name	varchar	False	False	The name of the spatial unit group.
ParentSpatialUnitGroupGuid	uuid	False	False	
ReferencePoint	geometry	False	False	The coordinates of a point within the spatial unit group.

1.60 Class «table»: SpatialUnitOfMeasure

Extra characteristics and values for a point (e.g. area, volume).
(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier (from super class).
SpatialUnitGuid	uuid	True	False	
SpatialUnitUomValueTypeGuid	uuid	True	False	

1.61 Class «table»: UnitOfMeasure

The measurement units that can be used for extra characteristics of instances of for example Spatial Unit.

(*Version of Class: 25-3-2011*)

Attributes

Name	Data Type	Not Null	Static (Unique)	Notes
Guid	uuid	True	True	The system global unique identifier.
MeasureTypeGuid	uuid	True	False	
Name	varchar	True	False	
Symbol	varchar	True	False	
NameStandardUnit	varchar	False	False	

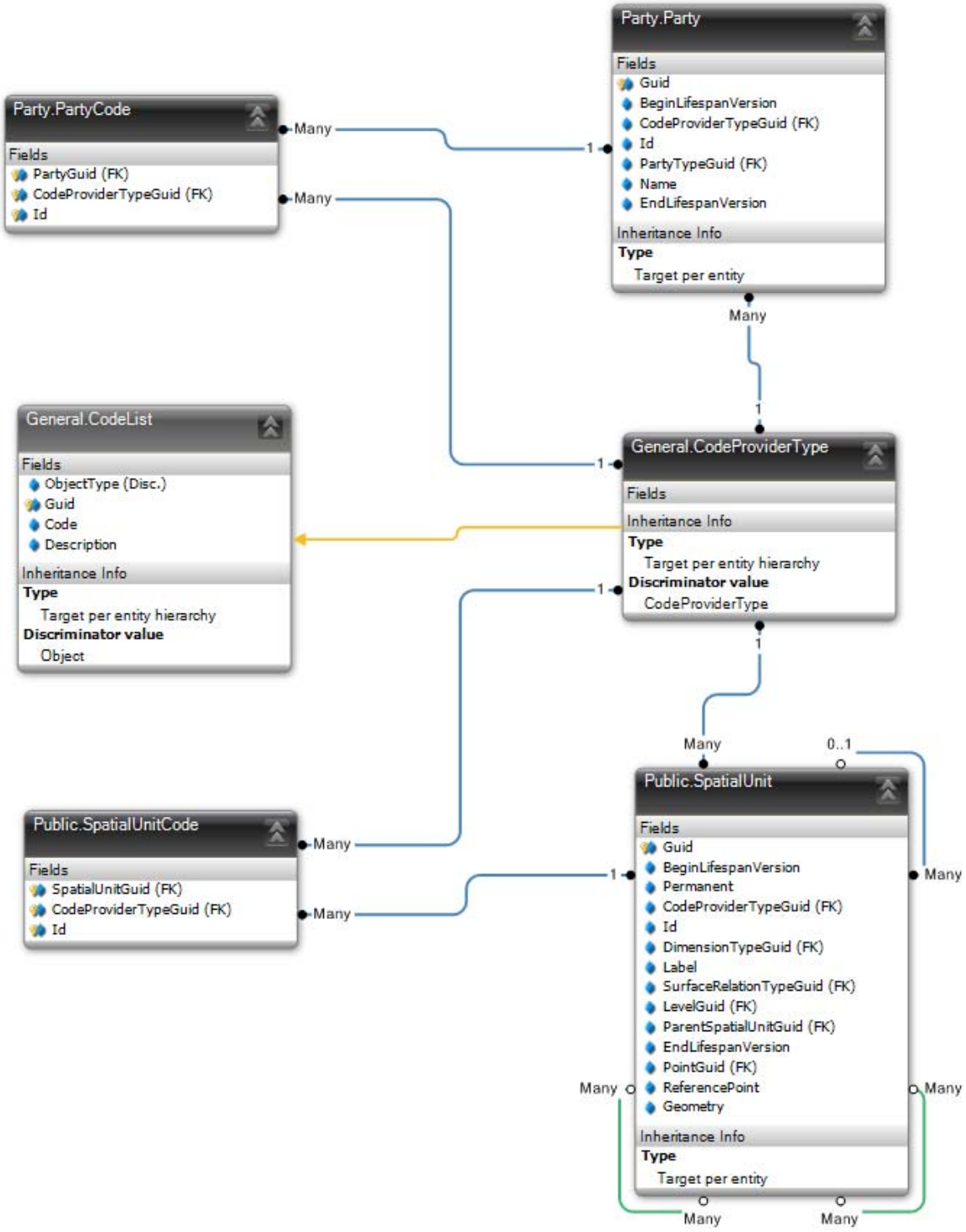
1.62 LLBLGenPro diagrams

The .Net classes, mapped to the PostgreSQL tables are presented in this package.
(*Version of Package: 25-3-2011*)

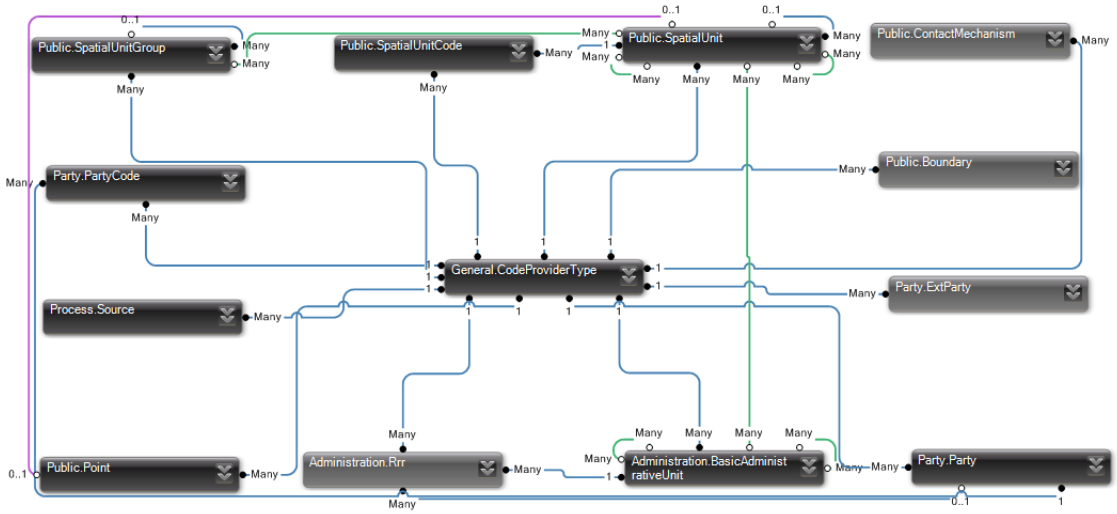
1.63 Artifact «document»: Implementation Oid

(*Version of Artifact: 25-3-2011*)

LLBLGenPro Designer Model: **LadmOid624**:



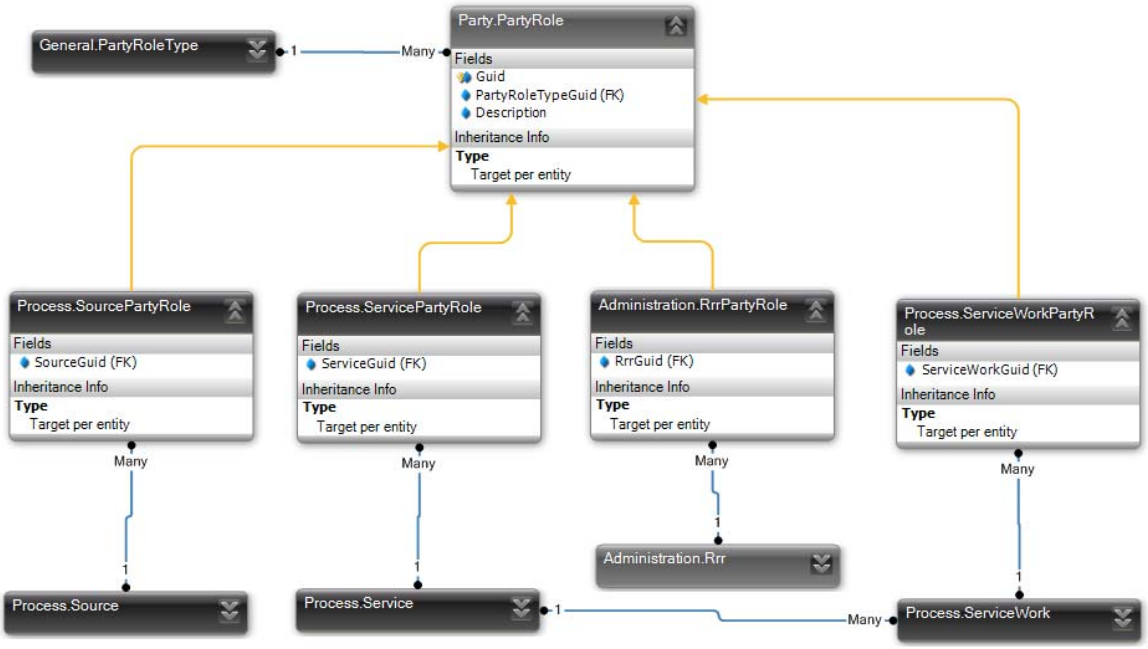
LadmCodeProvider:



1.64 Artifact «document»: Implementation Party Roles

(*Version of Artifact: 25-3-2011*)

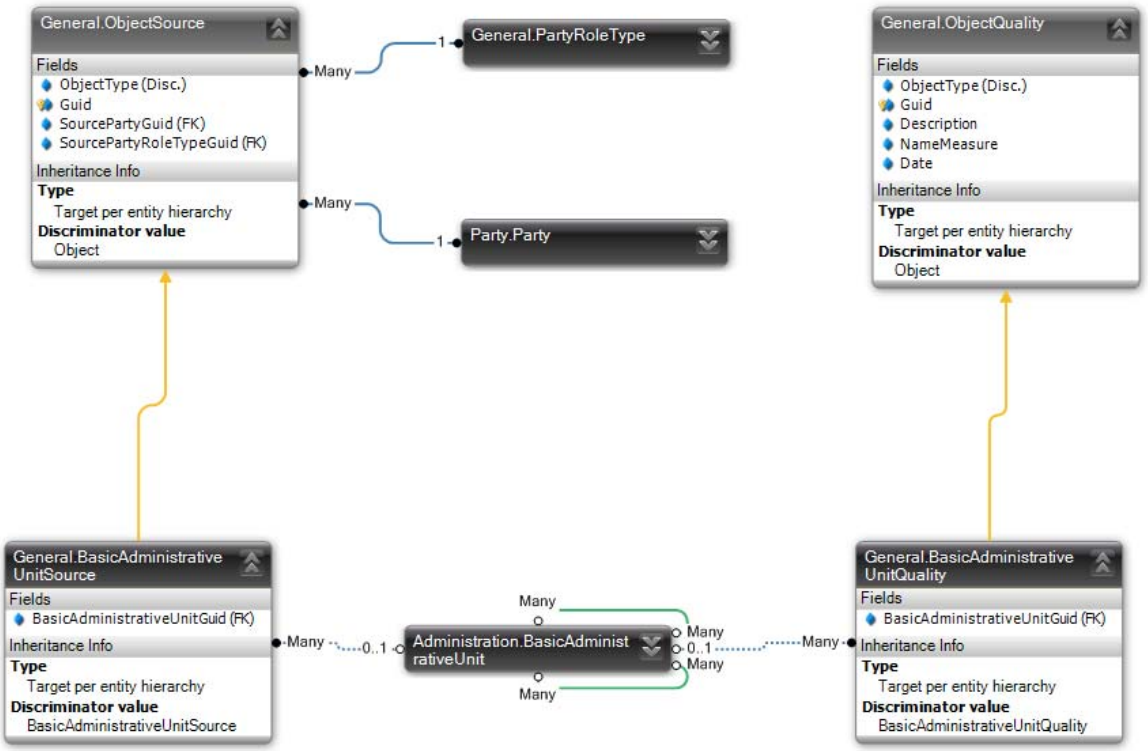
LLBLGenPro Designer Model: **LadmPartyRoleType634**:



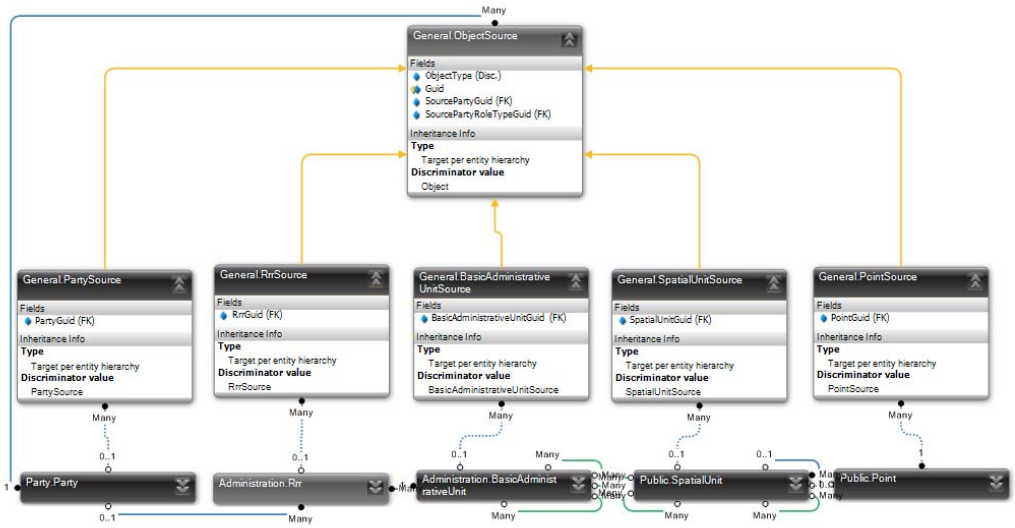
1.65 Artifact «document»: Implementation VersionedObject

(*Version of Artifact: 25-3-2011*)

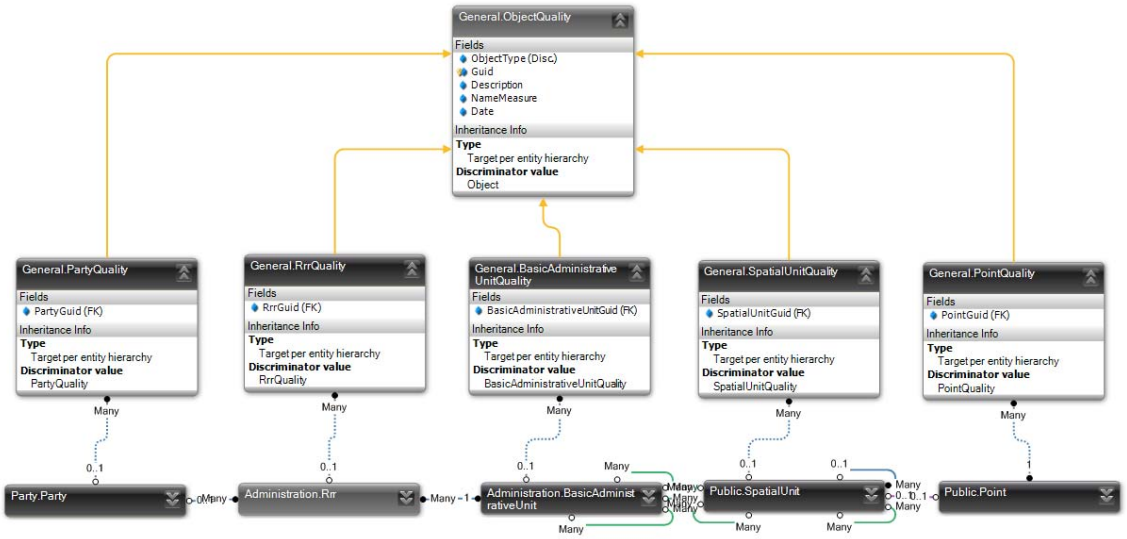
LLBLGenPro Designer Model: **LadmVersionedObject621**



LLBLGenPro Designer Diagram: **LadmObjectSourceOverview**



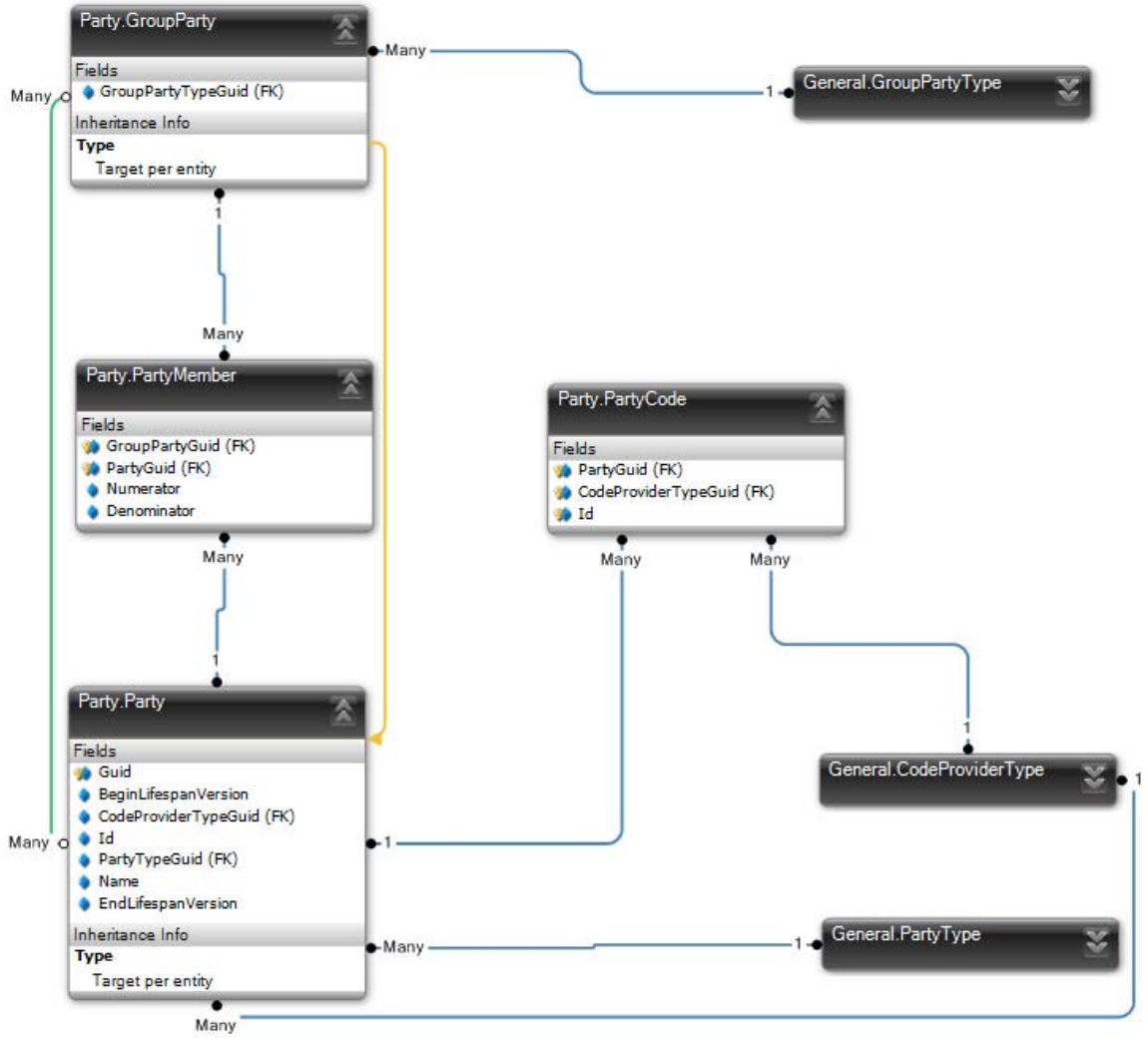
LadmObjectQualityOverview



1.66 Artifact «document»: Implementation of (Group)Party and Rational

(*Version of Artifact: 25-3-2011*)

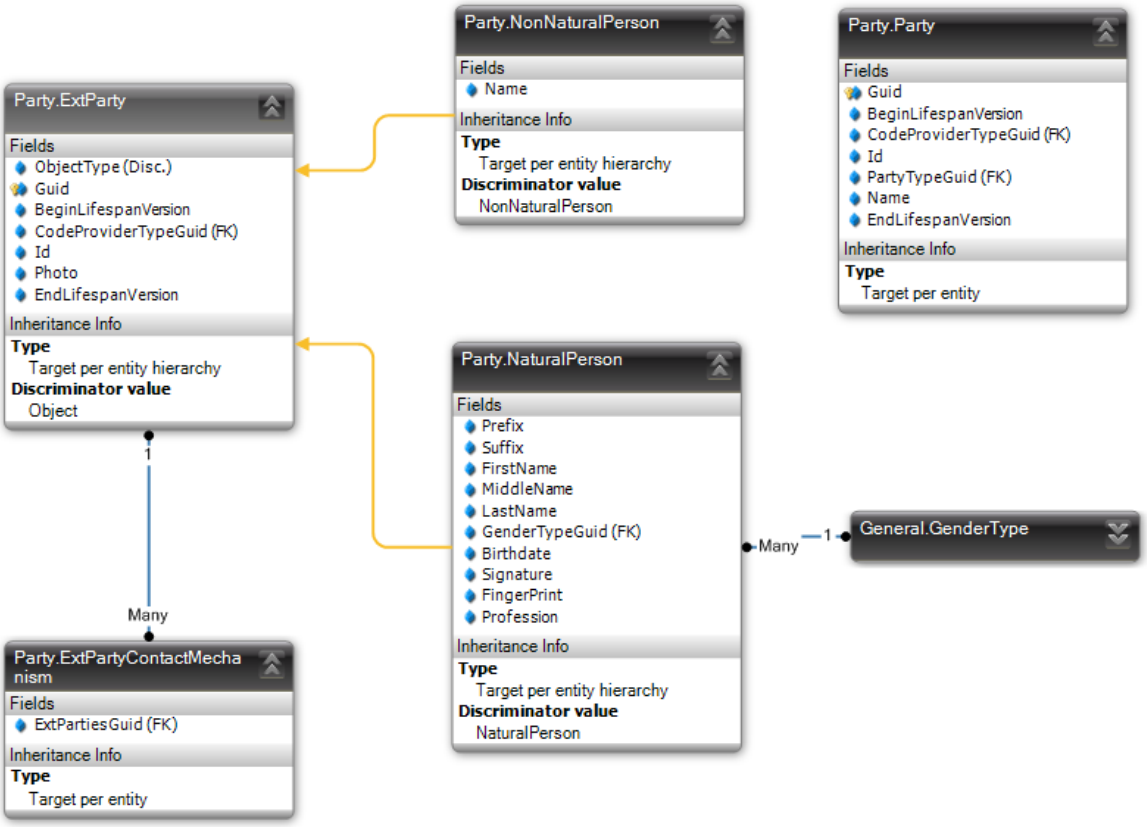
LLBLGenPro Designer Model: **LadmParty63**:



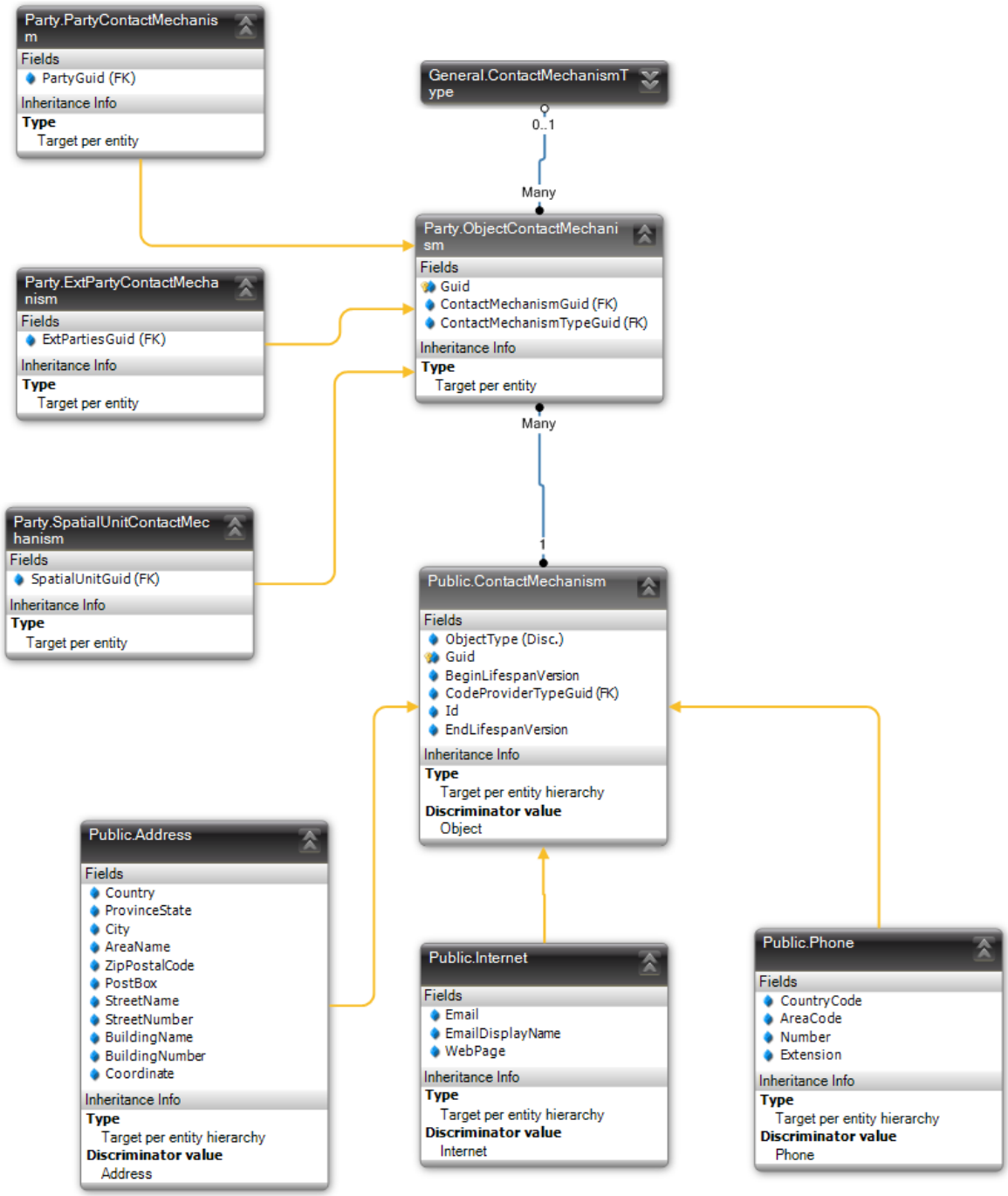
1.67 Artifact «document»: Implementation External Parties and Addresses

(*Version of Artifact: 25-3-2011*)

LLBLGenPro Designer Model: **LadmExtPartyK2**:



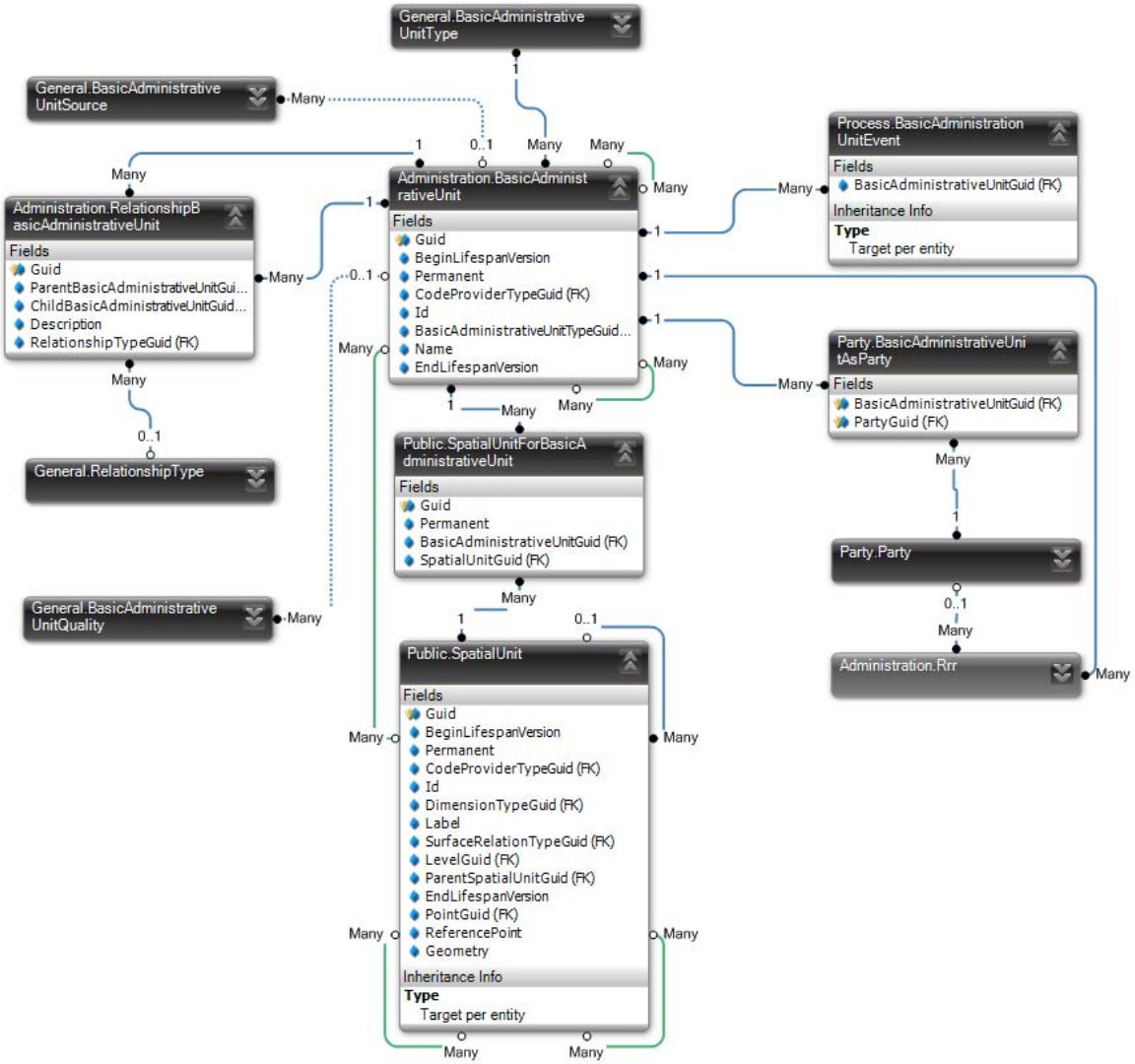
LLBLGenPro Designer Model: **LadmExtAdressK3**:



1.68 Artifact «document»: Implementation Administrative Package A

(*Version of Artifact: 25-3-2011*)

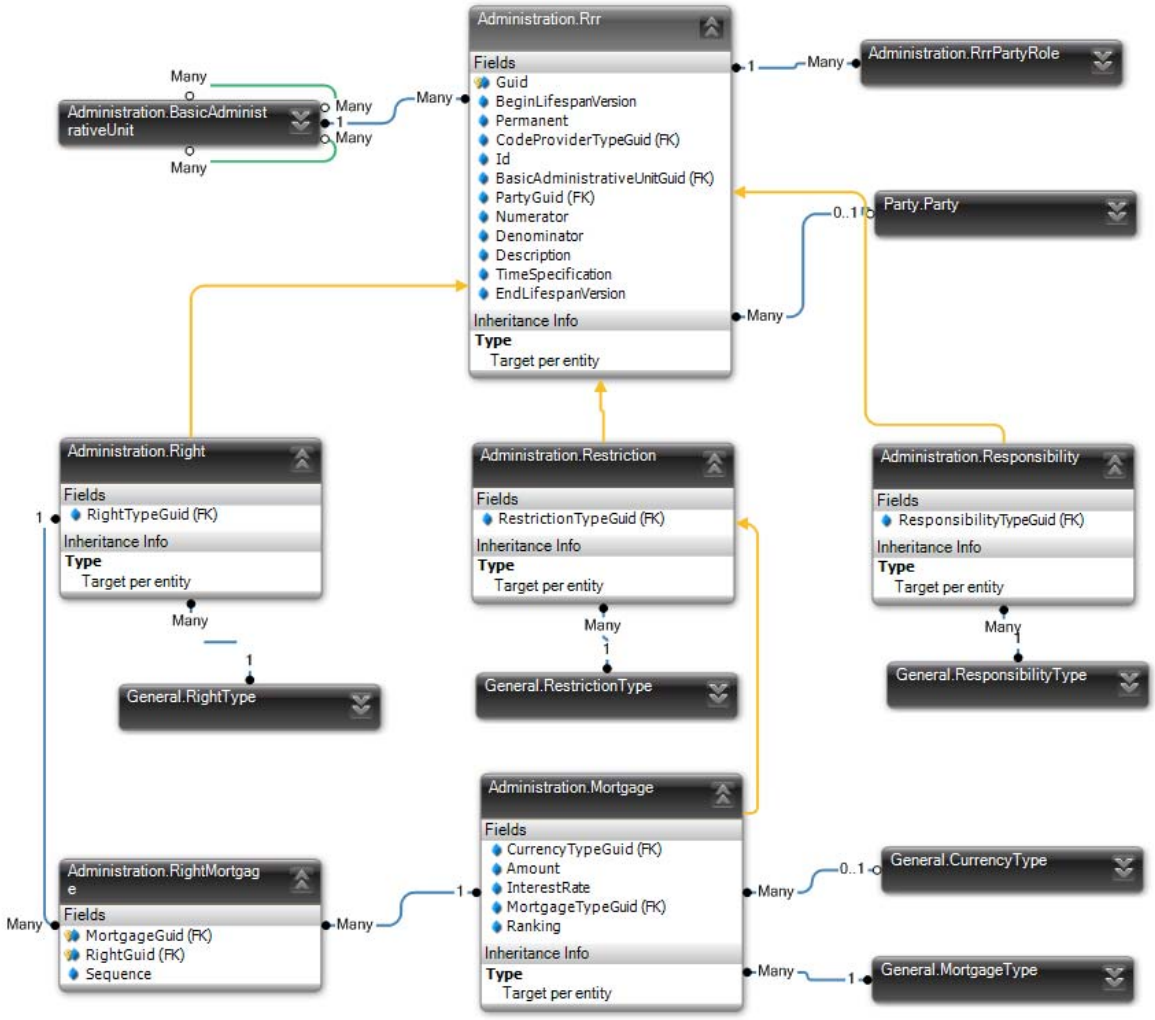
LLBLGenPro Designer Model: **LadmBasicAdministrativeUnit64A**:



1.69 Artifact «document»: Implementation Administrative Package B

(*Version of Artifact: 25-3-2011*)

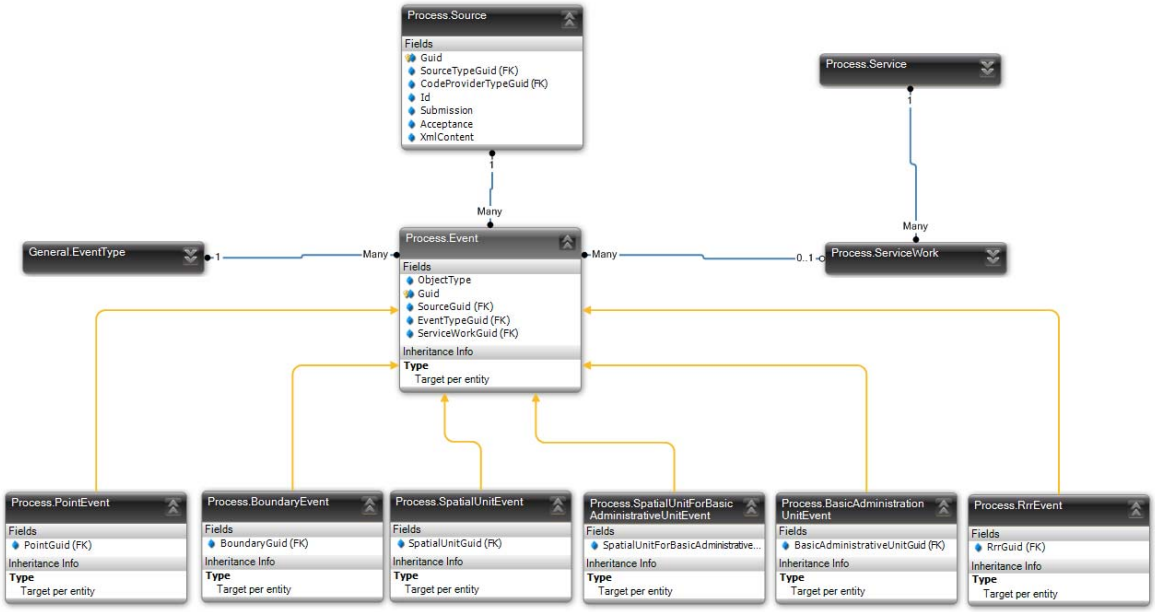
LLBLGenPro Designer Model: **LadmRightRestrictionResponsibility64B**:



1.70 Artifact «document»: Implementation Source Documents

(*Version of Artifact: 25-3-2011*)

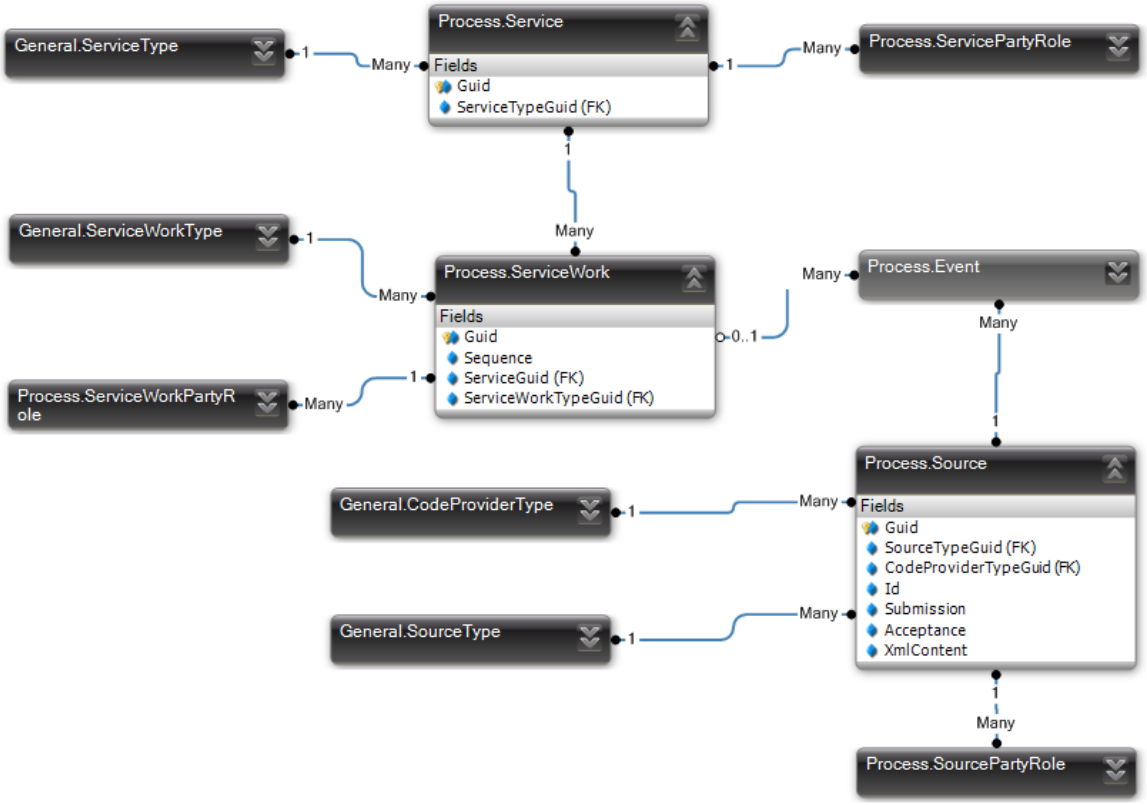
LLBLGenPro Designer Model: **LadmSource622**:



1.71 Artifact «document»: Implementation Service Work and Source

(*Version of Artifact: 25-3-2011*)

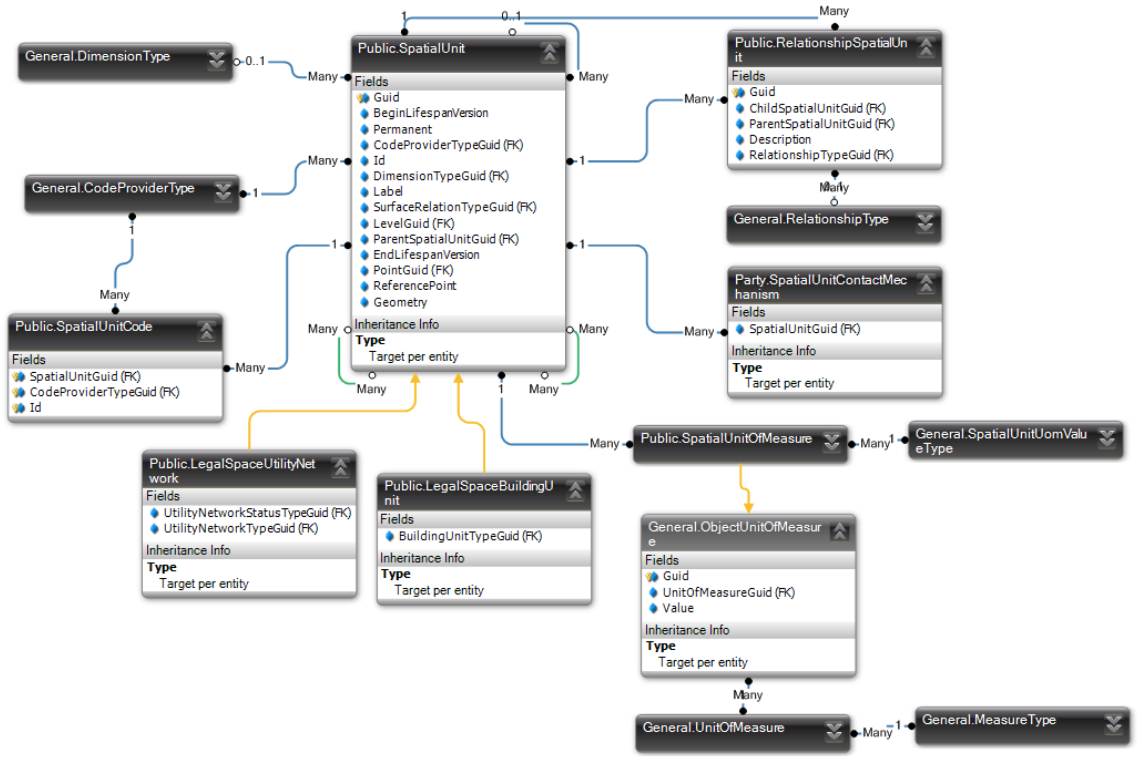
LLBLGenPro Designer Model: **LadmServiceAndSource**:



1.72 Artifact «document»: Implementation SpatialUnit

(*Version of Artifact: 25-3-2011*)

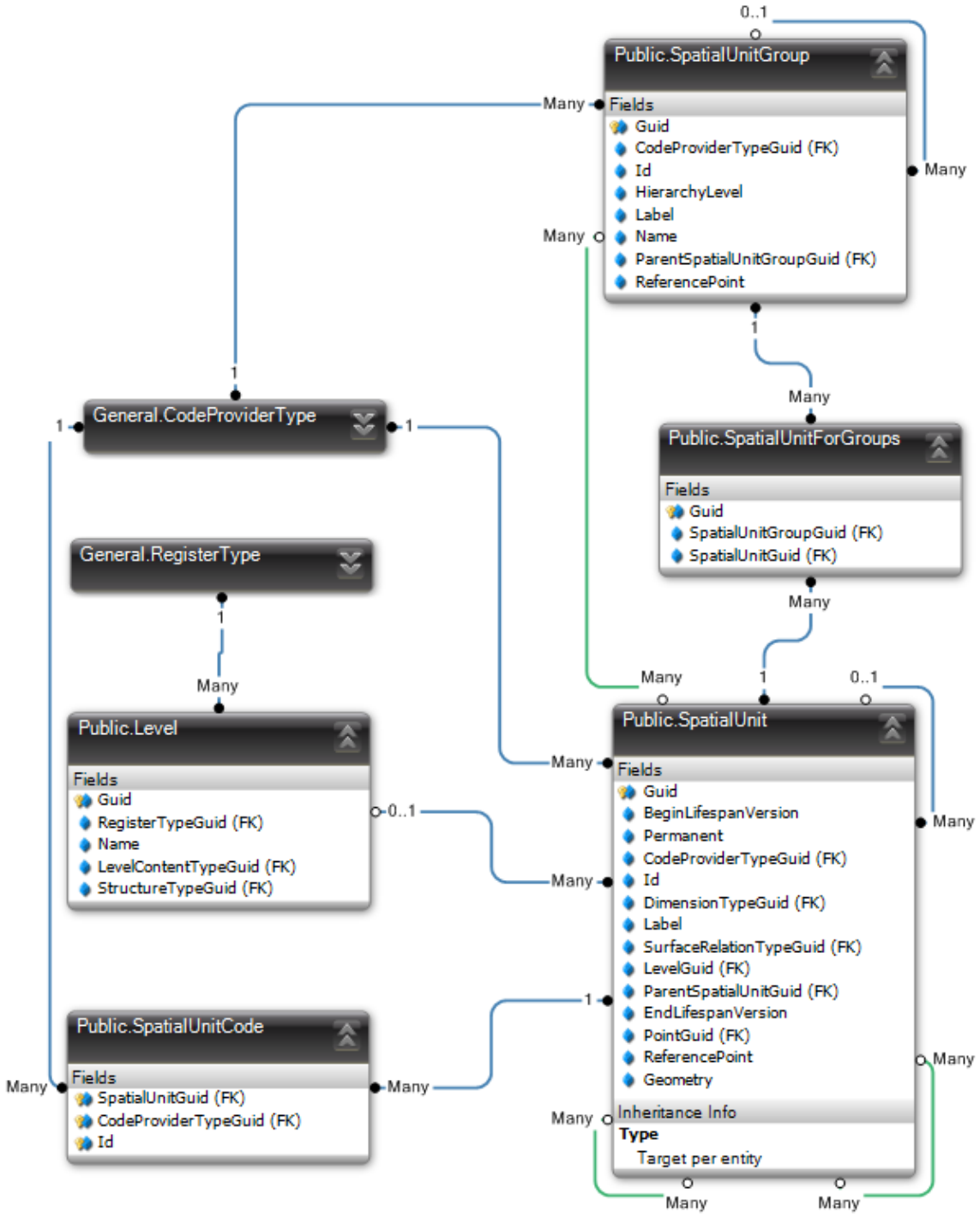
LLBLGenPro Designer Model: **LadmSpatialUnit651**:



1.73 Artifact «document»: Implementation SpatialUnitGroup

(*Version of Artifact: 25-3-2011*)

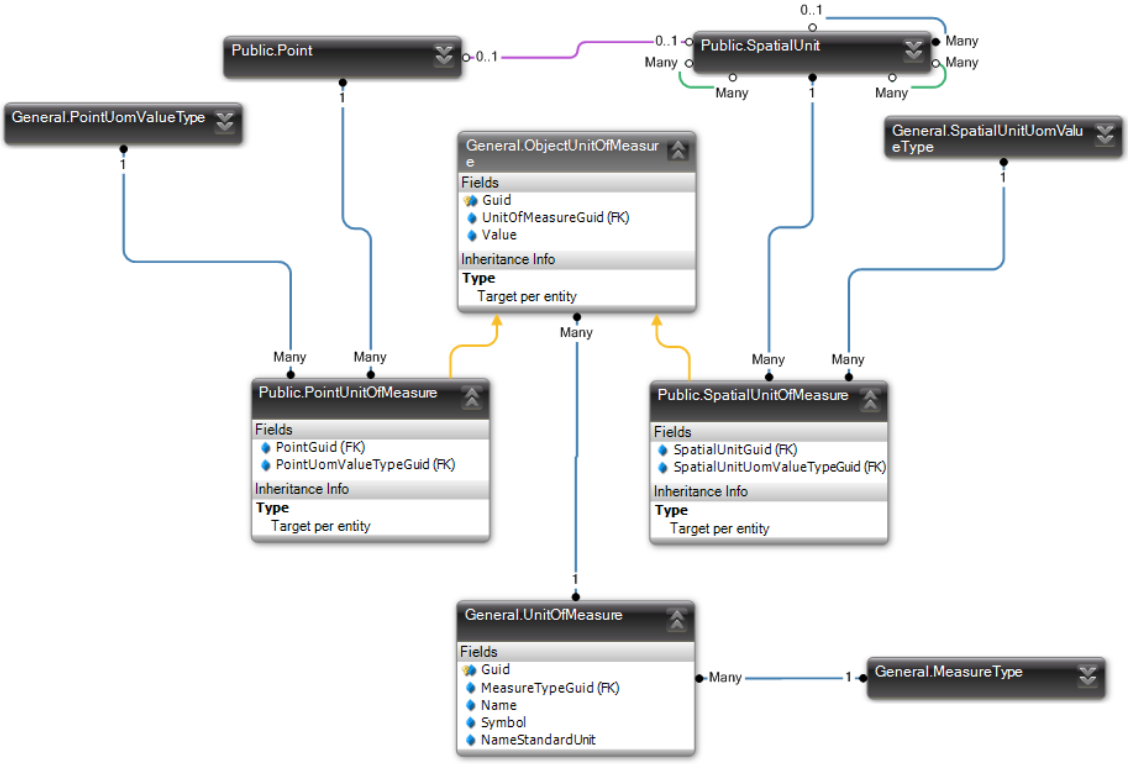
LLBLGenPro Designer Model: **LadmSpatialUnitGroup652**:


Public.SpatialUnitGroup
Fields
Guid
CodeProviderTypeGuid (FK)
Id
HierarchyLevel
Label
Name
ParentSpatialUnitGroupGuid (FK)
ReferencePoint
General.CodeProviderType
General.RegisterType
Public.SpatialUnitForGroups
Fields
Guid
SpatialUnitGroupGuid (FK)
SpatialUnitGuid (FK)
Public.Level
Fields
Guid
RegisterTypeGuid (FK)
Name
LevelContentTypeGuid (FK)
StructureTypeGuid (FK)
Public.SpatialUnit
Fields
Guid
BeginLifespanVersion
Permanent
CodeProviderTypeGuid (FK)
Id
DimensionTypeGuid (FK)
Label
SurfaceRelationTypeGuid (FK)
LevelGuid (FK)
ParentSpatialUnitGuid (FK)
EndLifespanVersion
PointGuid (FK)
ReferencePoint
Geometry
Inheritance Info
Type
Target per entity
Public.SpatialUnitCode
Fields
SpatialUnitGuid (FK)
CodeProviderTypeGuid (FK)
Id
0..1
Many
1

1.74 Artifact «document»: Implementation Unit of Measure

(*Version of Artifact: 25-3-2011*)

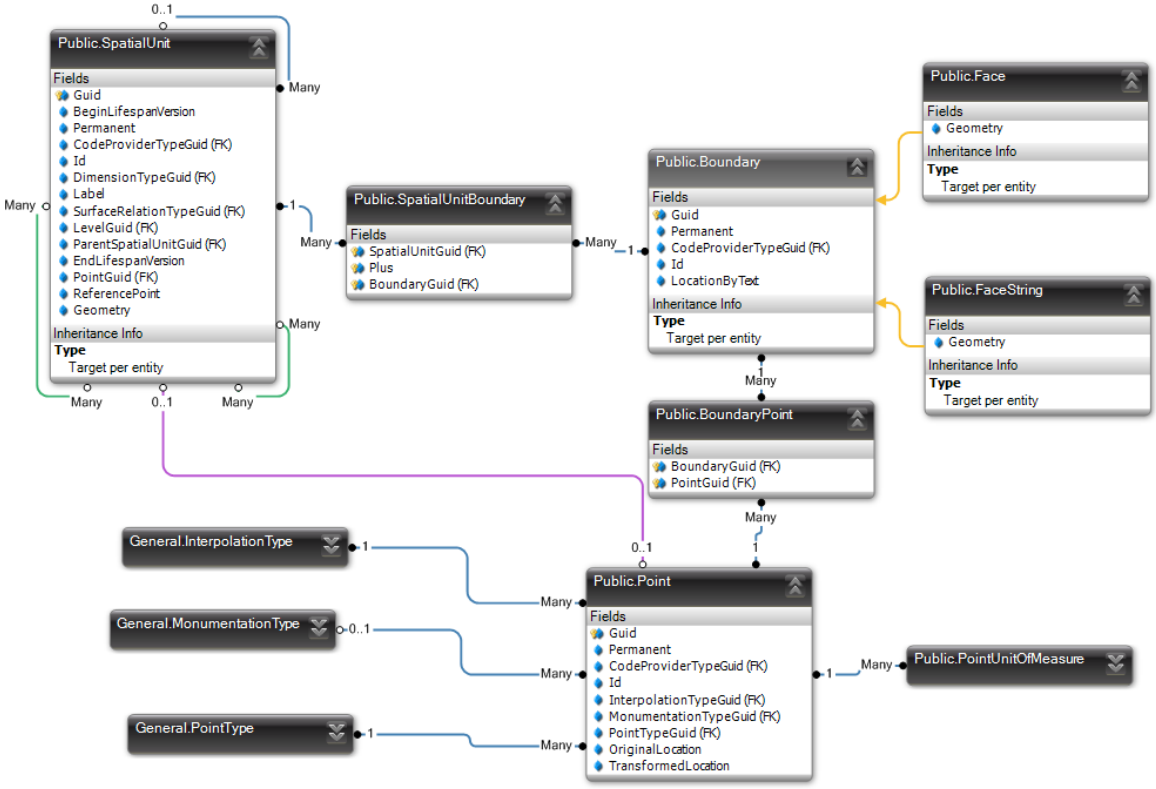
LLBLGenPro Designer Model: **LadmUnitOfMeasure**:



1.75 Artifact «document»: Implementation Surveying

(*Version of Artifact: 25-3-2011*)

LLBLGenPro Designer Model: **LadmSurveying66**:



1.76 Artifact «document»: Implementation External Classes

(*Version of Artifact: 25-3-2011*)

LLBLGenPro Designer Model: **LadmExternalClasses**: