Email
info@discon.co.za
Website
www.disconspecialists.com
Phone
(+27) 12 667 5975
LinkedIn
https://www.linkedin.com/company/discon-specialists

DISCON Specialists

EA Enabling Technique

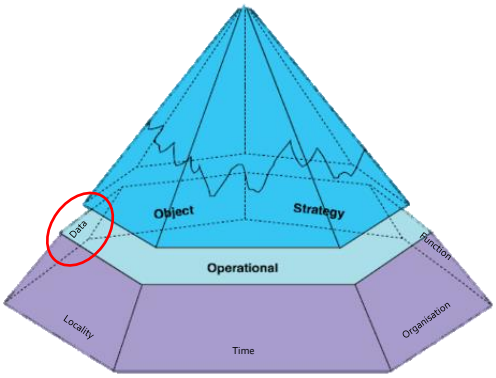
Attribute Dependency Diagram

Reason for Existence

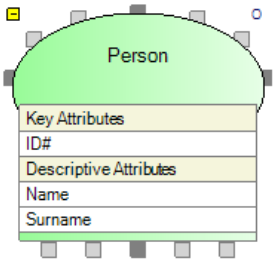
In the DISCON methodology, the Attribute Dependency Diagram is a mathematical representation of an organisation or business area's business rules. This is achieved by designing functional dependencies between attribute sets that are associated with events and objects of the business.

By applying pre-defined algorithms, the ADD provides certain outputs that are of value to the Business and the Business Engineering exercise in terms of Architecture and Data. The ADD represents the Data panel in the DISCON three-tiered pyramid.

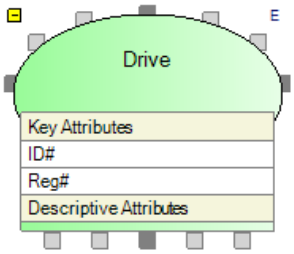
ADD models deliver a mathematical verification of the business rules, architecture & architectural priority, data in the 5th normal form and mathematically derived business objects.



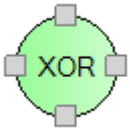
ADD Notation



Object Entity



Event Entity



Mutually Exclusive

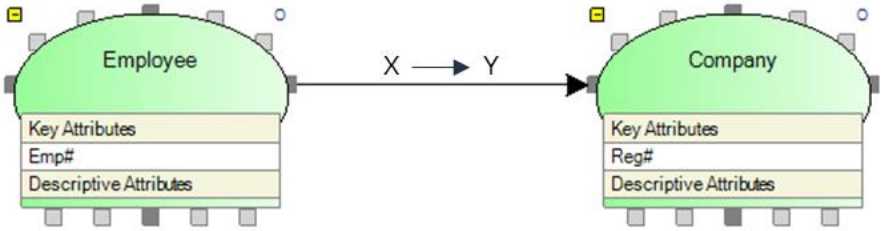
Functional Dependency

Subset of

Business Semantics

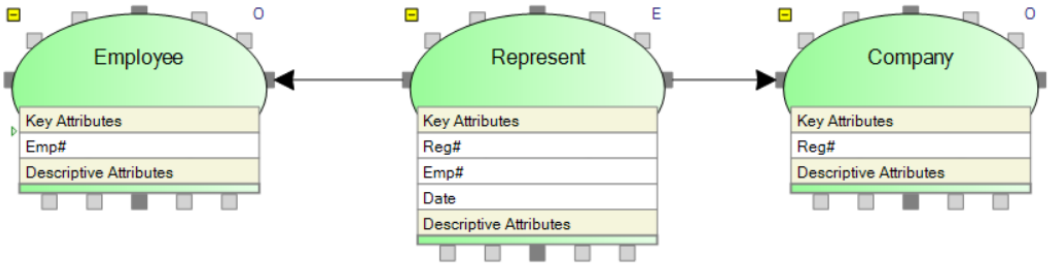
Functional Dependencies

In the following example there is a functional dependency between the key attribute set of the 'Employee' and the 'Company'. This implies that the key attribute of the employee object functionally determines the key attribute of the company object. For every employee there is always 1 and only 1 company (f(x);x -> y) and for every company there can be 0/1/ Many employees.



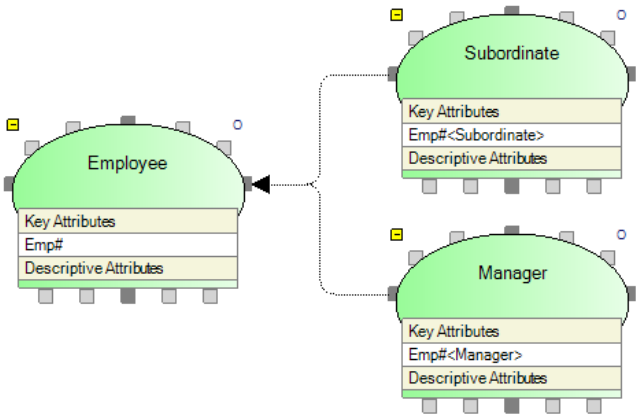
Key Attribute Sets

To enable the employee in the previous example to work for more than one company, an event describing the event of working or association entity needs to be considered. The event 'Represent' has a key attribute set that uniquely identifies an instance of the event and determines the business rule. The event is an association entity and allows for the many-to-many relationship we want between the key attribute sets of 'Company' and 'Employee'.



Subset Combinations

Depending on the business rule, subset combinations can be used to represent a generalised or a specialised perspective of the specific business rule. In the example below the employee object can be a 'Manager' or a 'Subordinate' or both. The dashed arrows represent an "either or" relationship between the originators of the arrows. Subordinates report to one manager, but the manager can also be a subordinate of another manager.

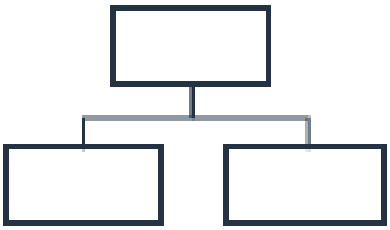


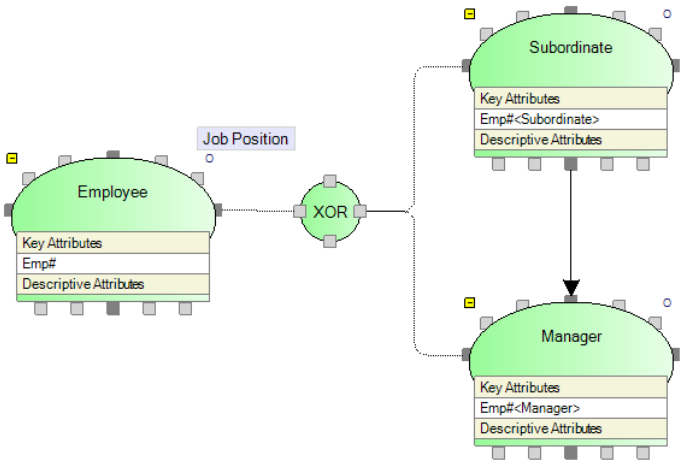
Hierarchical Structured Associations

In some cases data can be broken down into a tree structure. A data entity will have two subsets where the one subset will be functionally dependent on the other. Subsets are used to depict the precise role of the occurrence of the entity. The key attribute set of the subset entity is normally the same as the superset entity but in a different role.

Single - Level Hierarchy

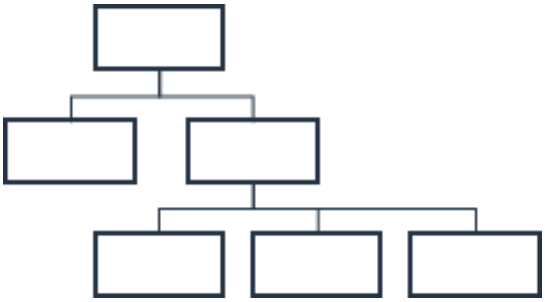
An example would be when each subordinate in a company has only one manager although the subordinates do not have anyone that reports to them.

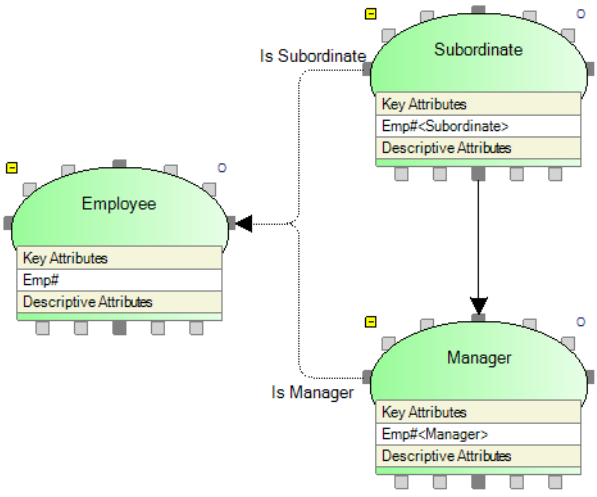




Multi-Level Hierarchy

An example would be when each subordinate in a company has one manager although the manager can be a subordinate of another manager. The employee of a company can either be a subordinate or a manager or both. Any subordinate must have a manager.



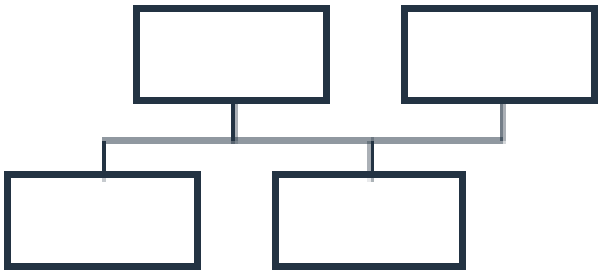


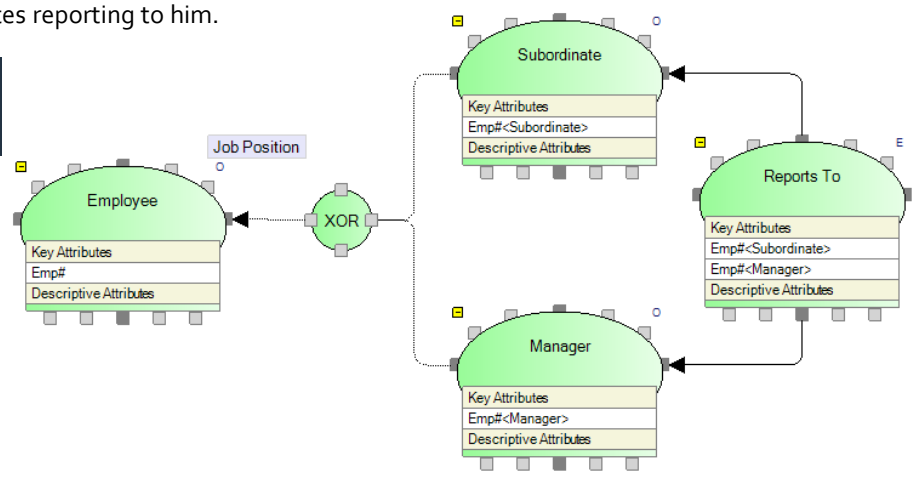
Network Structured Associations

In cases where the associations are of a many-to-many nature, the structure is known as a network structure. An entity will also have two subsets where the two subsets will have a many to many relationship between them. Subsets are again used to depict the precise role. The unique attribute set of the subset entity is normally also the same as the superset entity but in a different role.

Single - Level Network

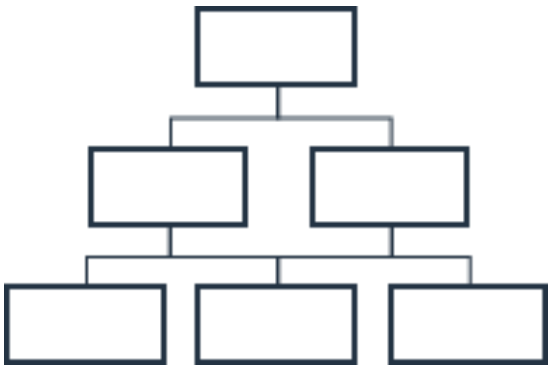
This indicates that a subordinate can never be both the manager and subordinate; and the manager can never be both the manager and subordinate. The employee of a company can either be a subordinate or a manager and not both. Any subordinate can report to multiple managers and a manager can have multiple subordinates reporting to him.

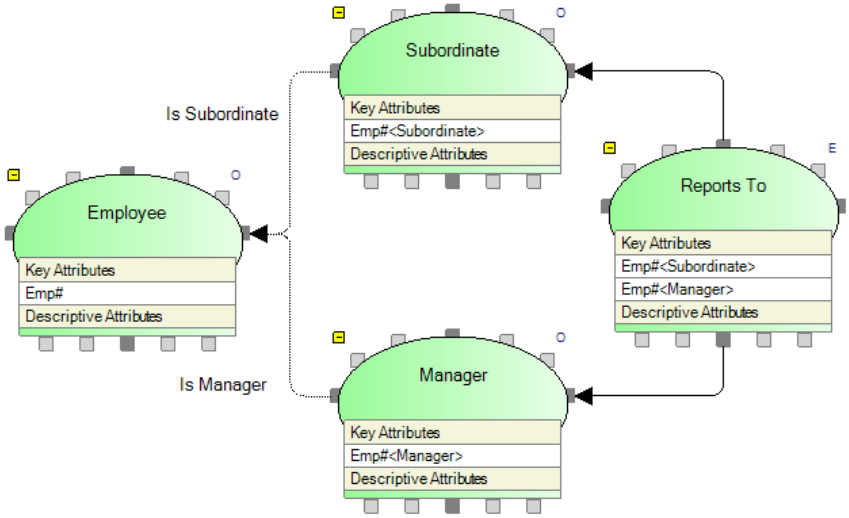




Multi-Level Network

An example would be when multiple subordinates in a company report to a number of managers and a manager can have a number of subordinates reporting to him. The employee of a company can either be a subordinate or a manager or both. Any subordinate can report to more than one manager and a manager can have multiple subordinates reporting to him. This could also be called a reporting structure.

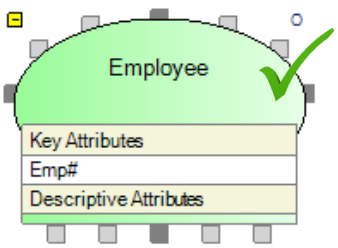


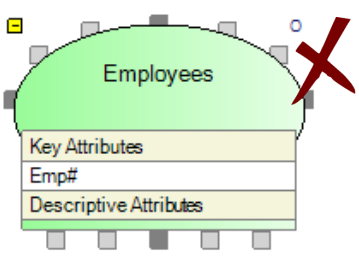


Heuristic Rules

Plural

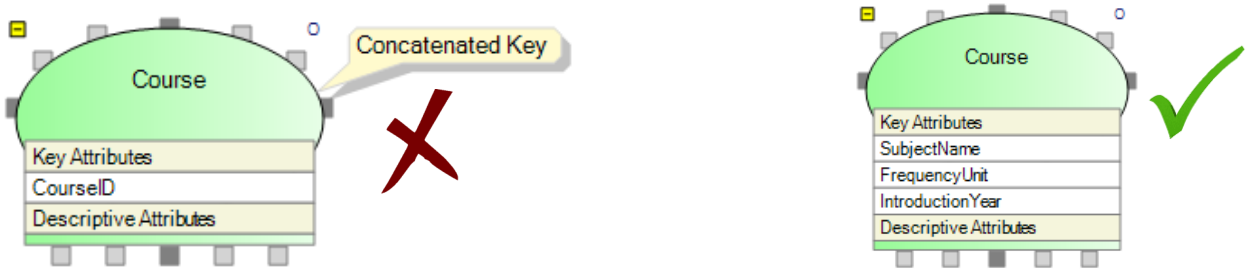
A Data Entity or attribute is singular by nature and not plural. To express an entity we must never refer to the entity as a plural. Employees, Vehicles and Organisations are all examples of incorrectly formulated entities. Where as Employee, Vehicle and Organisation are all properly formulated entities.





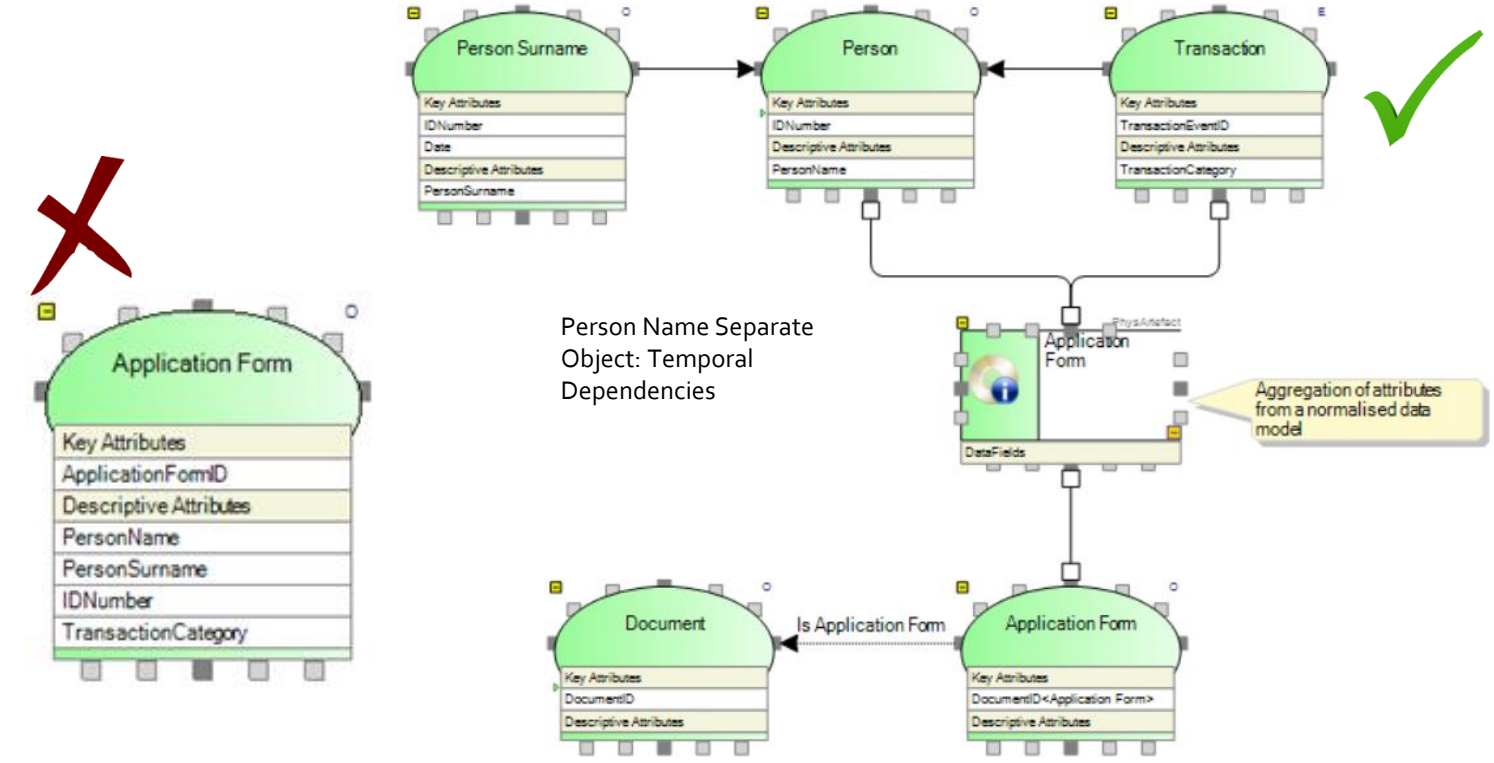
Natural Key Attribute Set

A Key Attribute Set should represent the most natural combination of attributes that can be used to uniquely identify each instance of an Entity. This will ensure that the functional dependencies between the Key Attributes Sets are more correct. For example: CourseId typically consist of a concatenation of a number of other attributes which means it is not in it's most natural form. CourseId could potentially be a concatenation of SubjectName, FrequencyUnit (Monthly, Semester, Year) and IntroductionYear.



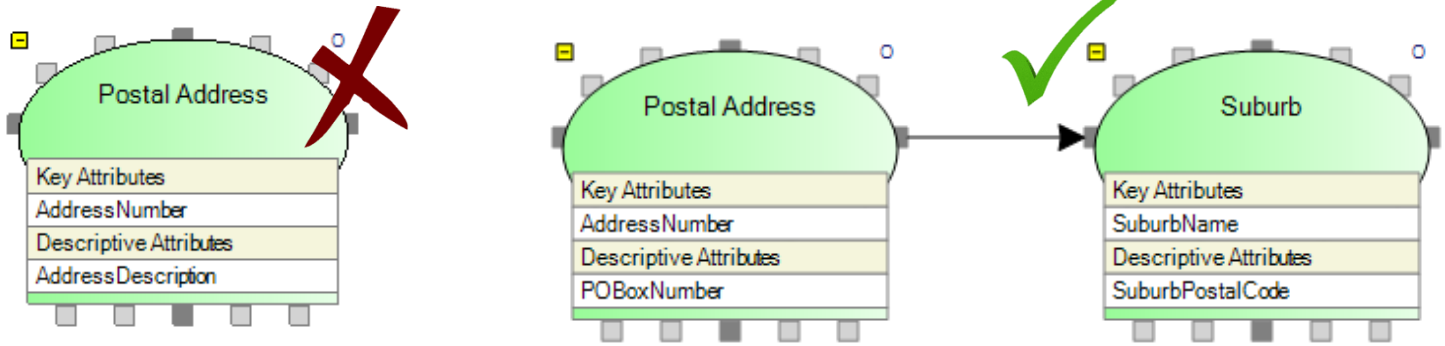
Modelling Data Artifacts as Data Entities

Care should be taken when modelling Data Artefacts such as application forms, contracts, policy documents and others. The ideal notation for representing such artefacts is the Data View which represents an aggregation of data attributes selected from a normalised structure and which is used for a specific purpose within a process. Data Artefacts can be modelled as Data Entities but this would typically be done for modelling Record or Document Management semantics. The attributes on the Physical Data Artefact can however be reversed engineered and used to generate a logical data model.



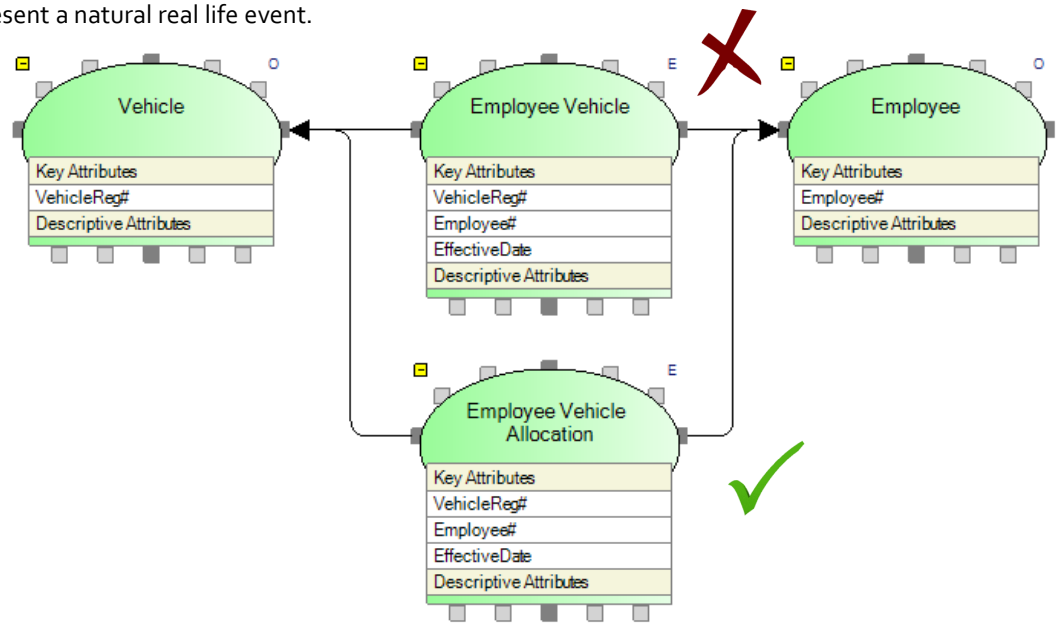
Natural Data Attribute Names

Attempt to avoid using system generated attributes, concatenate attribute or attributes that are not in their smallest atomic form when creating a logical data model. A Description or Comment Attribute is not in its smallest atomic form. For example the Address Description attribute should rather become PO Box Number, Suburb Name, City Name and Country Name.



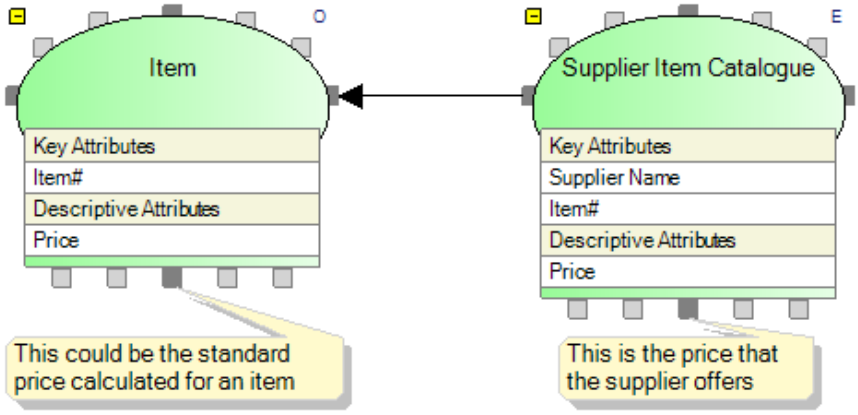
Natural Data Entity Names

Use natural names for the Data Entities that represent an Object or Event as a natural construct. Through concatenating two Entity names to create an Event Entity does not represent a natural real life event.



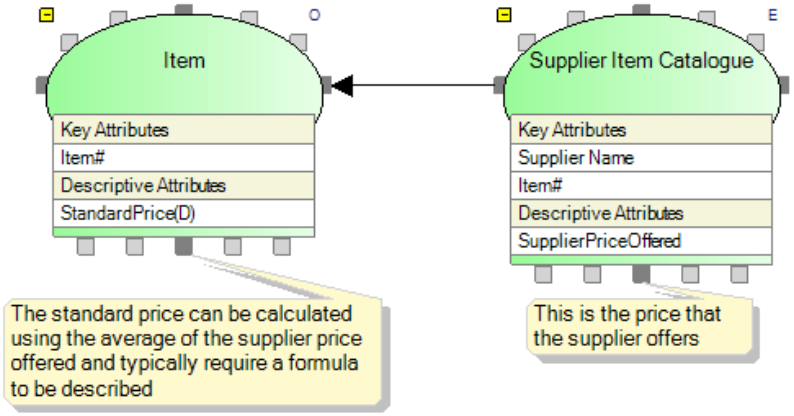
A Descriptive Attribute Must Describe the Entity Directly

Place the Descriptive Attributes on the Entities that it best describes to ensure a normalised structure. The Descriptive Attribute should not indirectly describe the Entity.



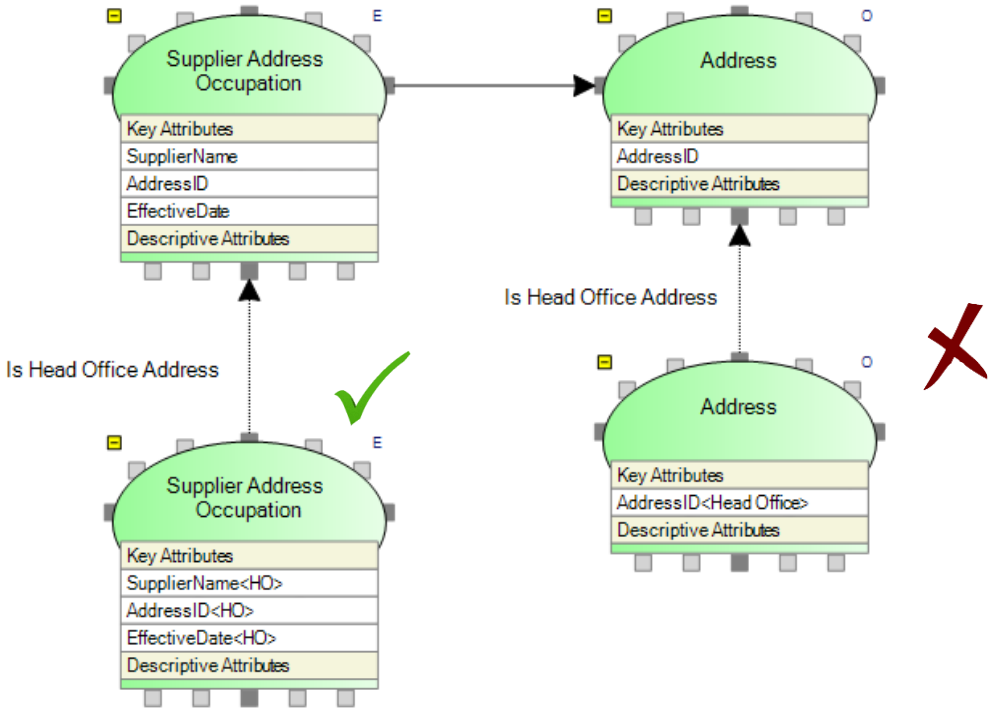
Identify Derivable Attribute Values

When the attribute value of an entity can be derived from the attributes of another entity, it is a derivable attribute. These attributes typically imply Logic (Algorithm/Formula) to generate the values for the attribute.



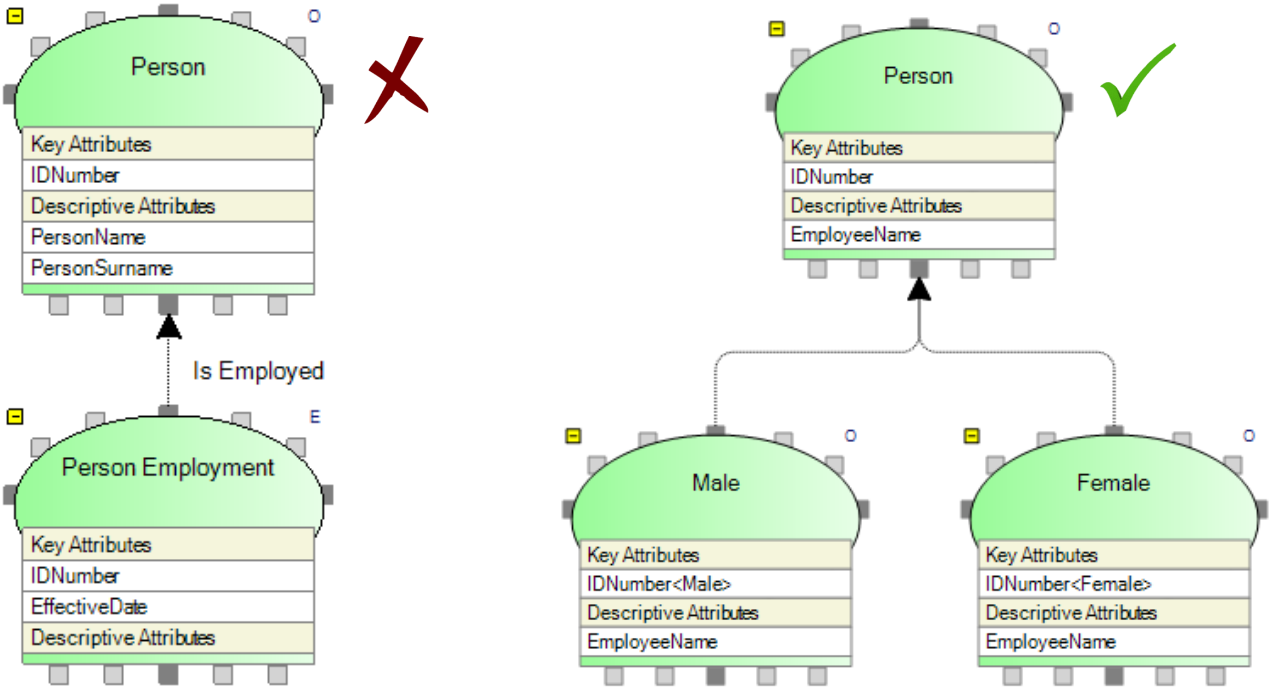
Place the Subset on the Appropriate Superset Entity

Test if the Subset Entity cannot be allocated to another Superset Entity that will more accurately represent the business semantics.



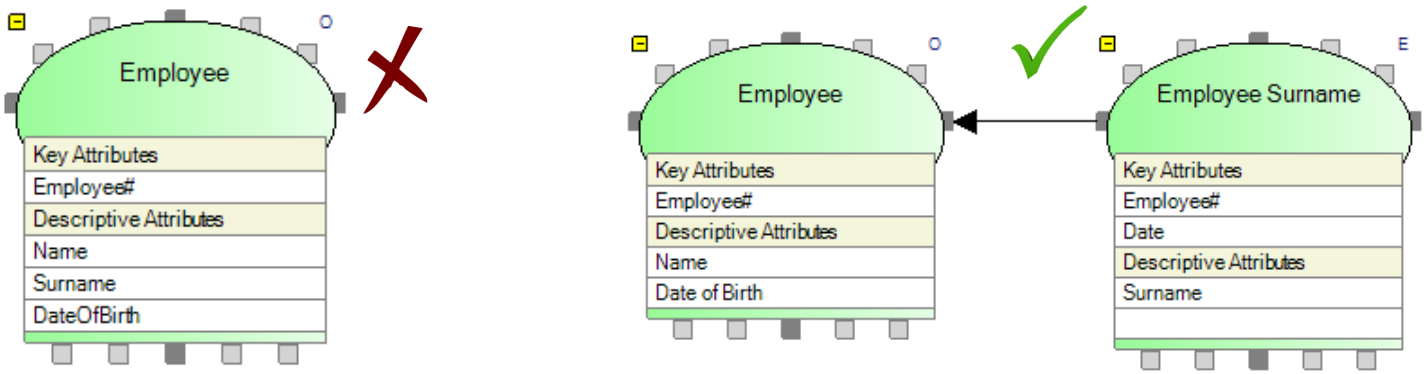
A Subset Always Inherits the Entity Type of the Subset

A subset Entity cannot be of a different type as the superset Entity.



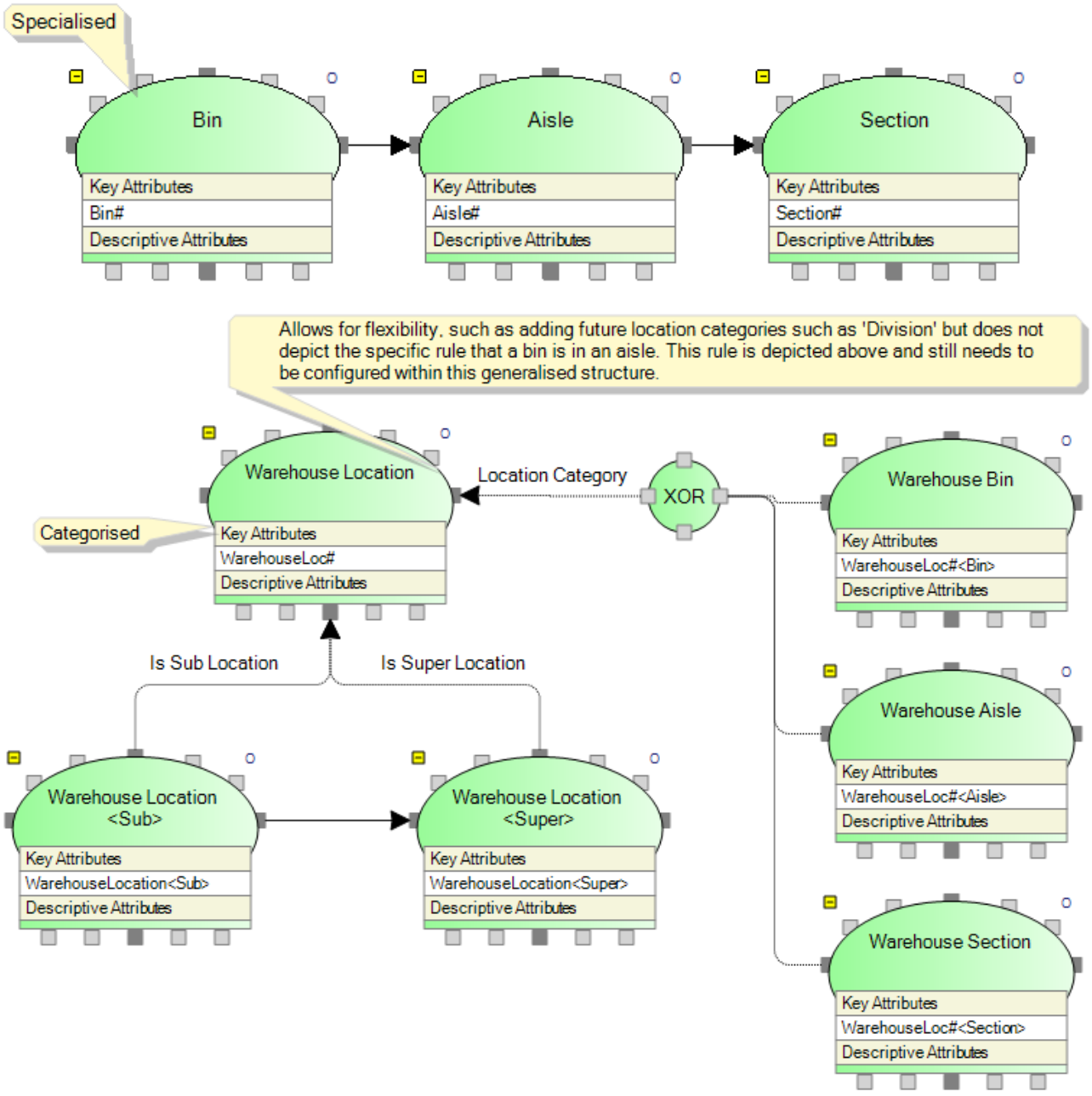
Temporal Dependencies

A dependency that is not always true over time is called a temporary dependency and should rather be avoided. The dependency between Employee ID and Surname is only true until the surname of the employee changes.



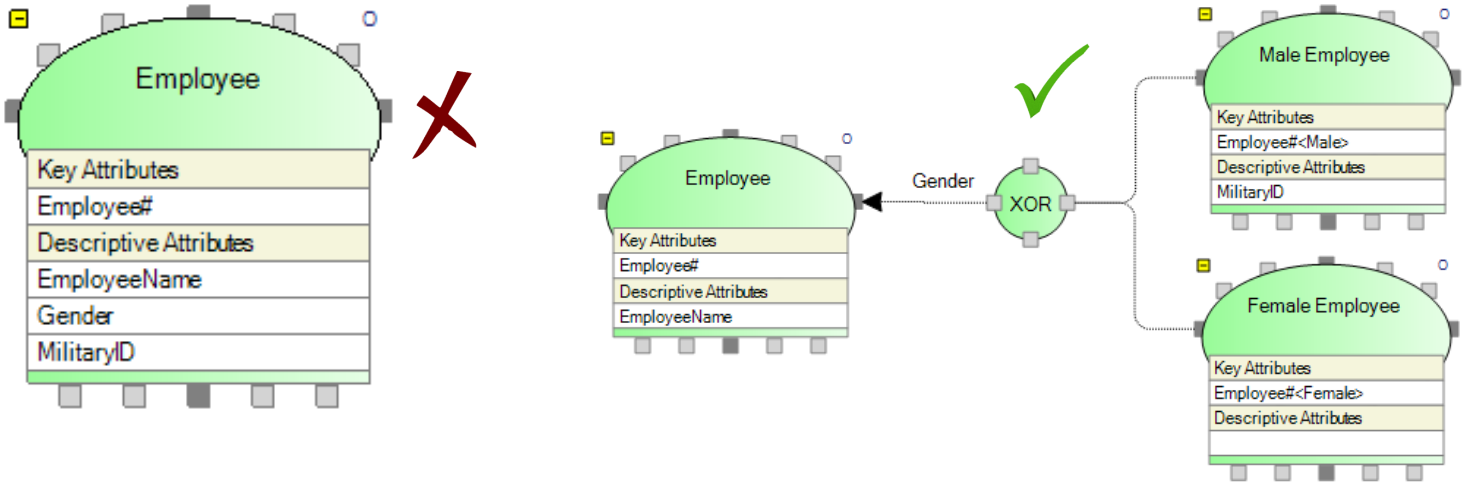
Use Generalised vs Specialised Models Carefully

Both instances of models are valuable. Ideally, specialised models need to be created first to better understand the business semantics and to be able to later configure the rules within a more generalised data model. Generalised data models allow for more flexibility but tend to be more difficult to be interpreted.



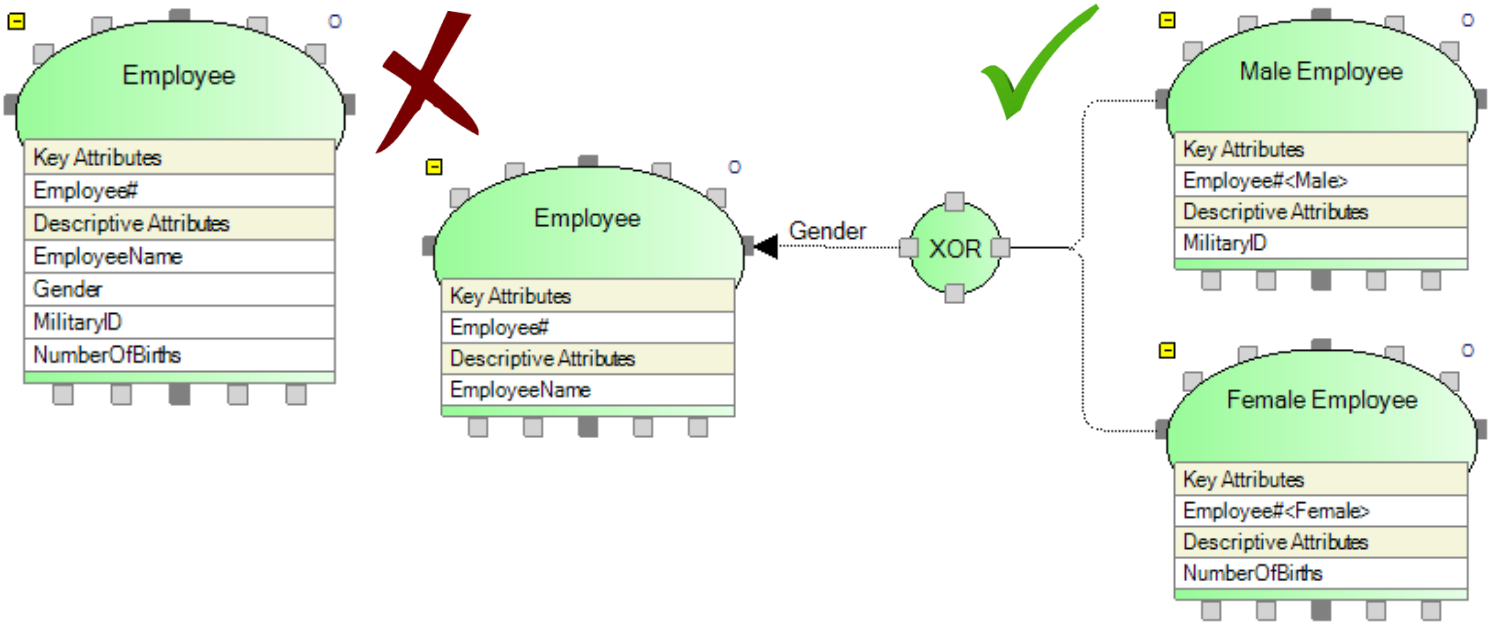
Conditional Dependencies

When the descriptive attribute of an entity can be conditionally dependent on the key attribute set of the entity, it is called a conditional dependency. In this case the business rule is not complete and subsets should be used to depict the structure more clearly. In the example below the business would like to know what the Military Id of all male employees is. In the first instance the Military Id is a Conditional Dependency of Employee Id. The dependency is not true for all the employees. If however the business rule was depicted as in the second instance the dependency will be true for all the male employees.



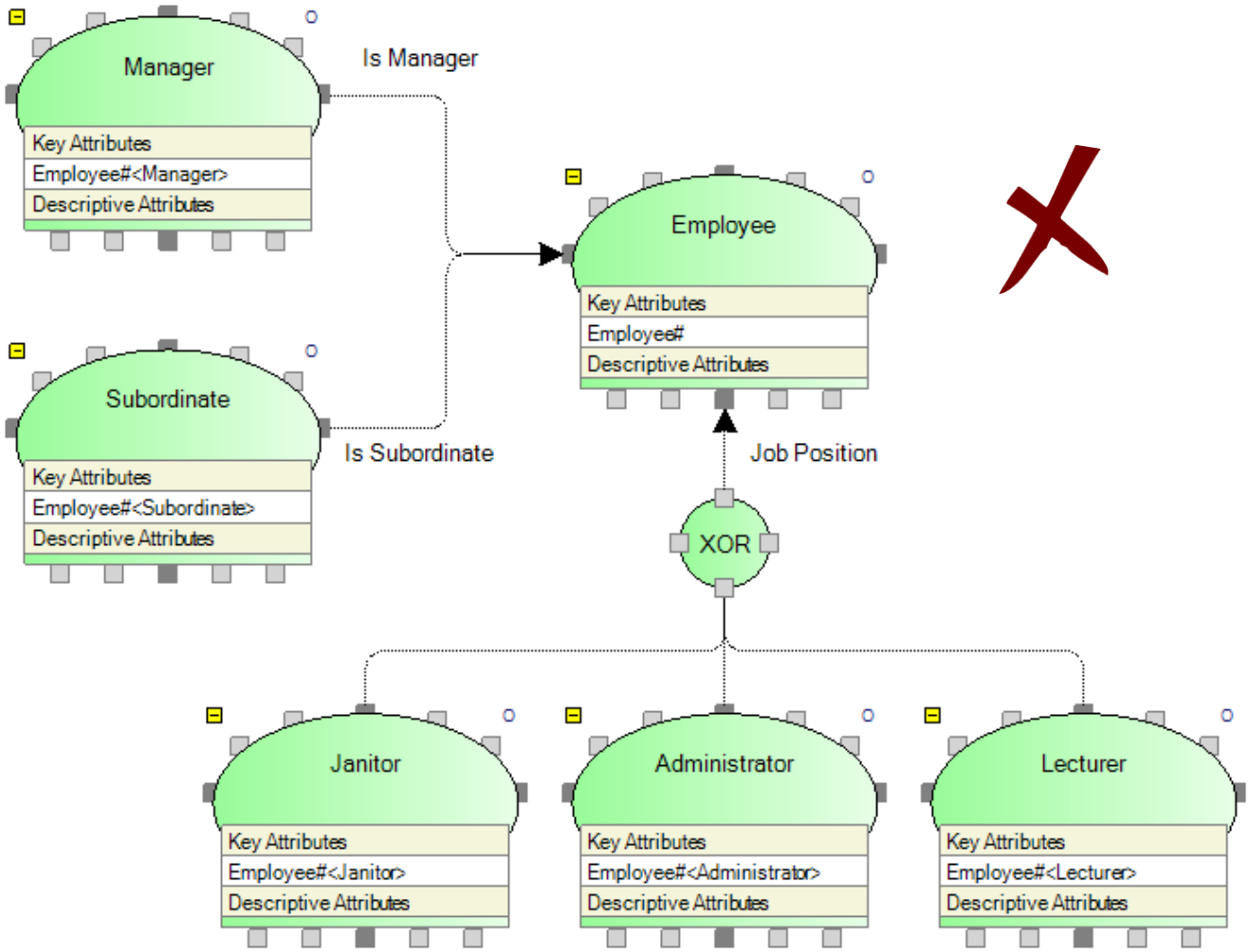
Nil-Value Dependencies

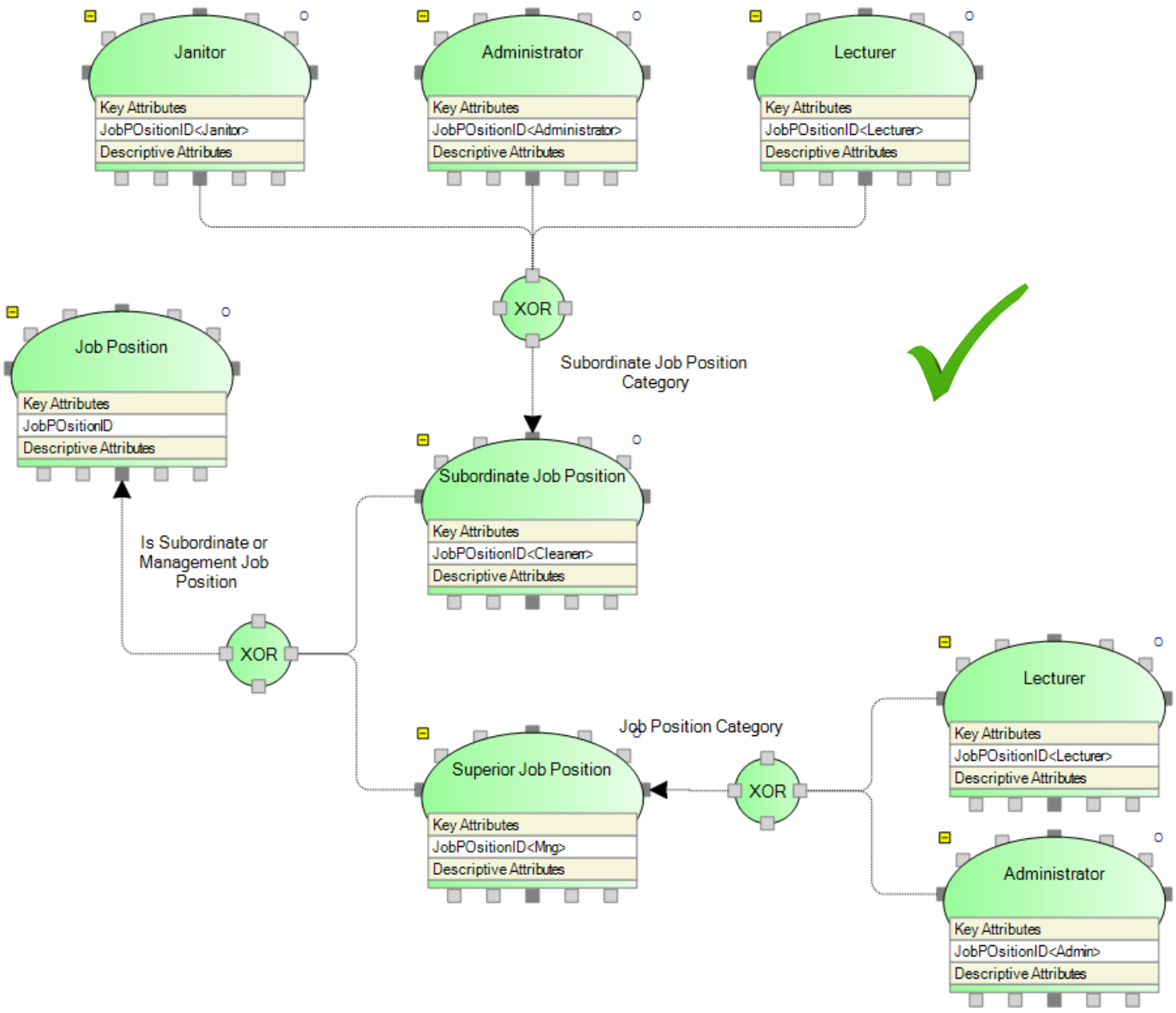
Descriptive Attributes of a Data Entity should only be functionally dependent on the Key Attribute Set and are not allowed to be functionally dependent on the values of any of the other descriptive attributes. If they are, these attributes could potentially have Nil-values. For example: The values of the attribute MilitaryID can have a value of nil if the gender is female and the value of the attribute NumberOfBirths can have a value of nil if the gender is male.



Contradictory Subset Combinations

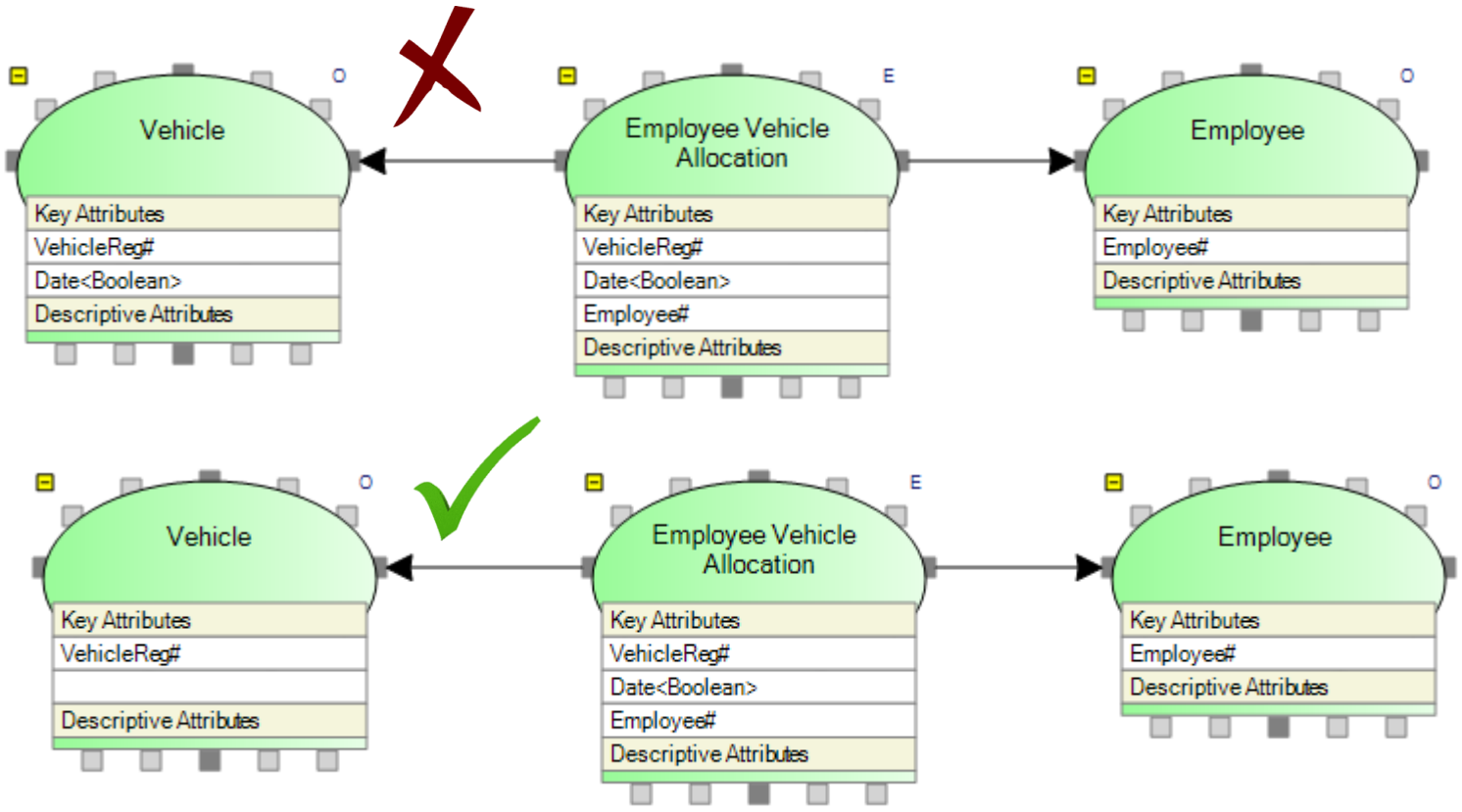
When there are multiple subset structures that have the same superset entity there might be some of the subsets that potentially contradict each other. Especially when there is a combination of "exclusive or" and "either or" subset combinations, contradicting subset combinations might occur. The first subset structure defines that an employee can either be a Janitor Employee, Administrator Employee or a Lecturer Employee. The second subset structure defines that an employee can be a Subordinate Employee or a Manager Employee or both. The two subset structures are contradicting each other. For this business a Janitor cannot be a manager although this is shown in the example. The semantics of the model are not correct.




Janitor
Key Attributes
JobPOsitionID<Janitor>
Descriptive Attributes
Administrator
Key Attributes
JobPOsitionID<Administrator>
Descriptive Attributes
Lecturer
Key Attributes
JobPOsitionID<Lecturer>
Descriptive Attributes
XOR
Subordinate Job Position Category
Job Position
Key Attributes
JobPOsitionID
Descriptive Attributes
Subordinate Job Position
Key Attributes
JobPOsitionID<Cleaner>
Descriptive Attributes
Is Subordinate or Management Job Position
XOR
Job Position Category
Superior Job Position
Key Attributes
JobPOsitionID<Mng>
Descriptive Attributes
XOR
Lecturer
Key Attributes
JobPOsitionID<Lecturer>
Descriptive Attributes
Administrator
Key Attributes
JobPOsitionID<Admin>
Descriptive Attributes

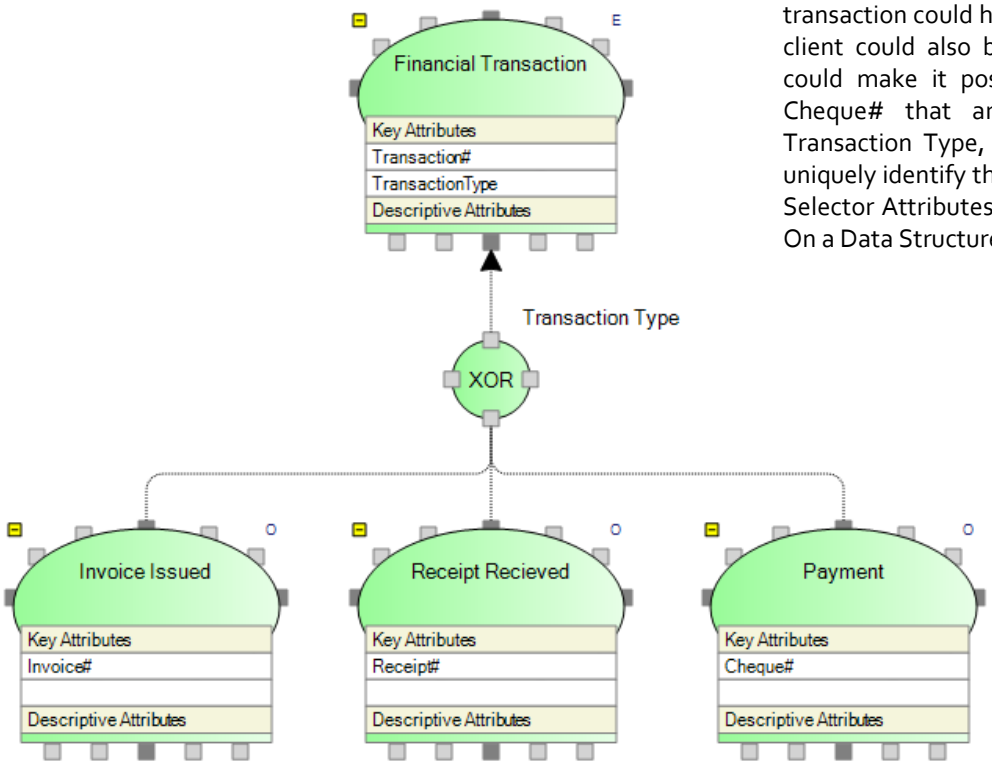
Consider Dates & Time Attributes For Each Event Entity

An Event Entity is an occurrence which implies that the Date and Time of occurrence is normally applicable for the business semantic.



Selector Attributes Within The Unique Identifier Sets

When the key attribute set of an entity does not uniquely identify the entity, the selector attribute associated with the entity can be used. The combination of the selector attribute and the key attribute set will then uniquely identify the entity.

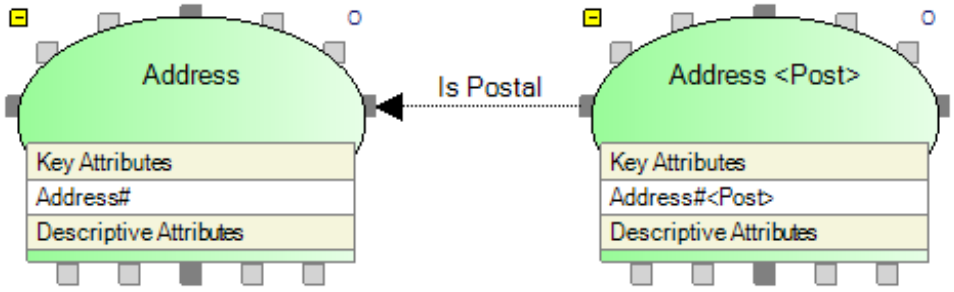


The different document types involved in the financial transaction could have the same document number. A different client could also be involved with each transaction and this could make it possible to have an Invoice#, Receipt# and Cheque# that are the same. If the selector attribute, Transaction Type, forms part of the key attribute set, it will uniquely identify the Financial Transaction entity.

Selector Attributes (for Subset Dependencies) must be named. On a Data Structure model this becomes a column.

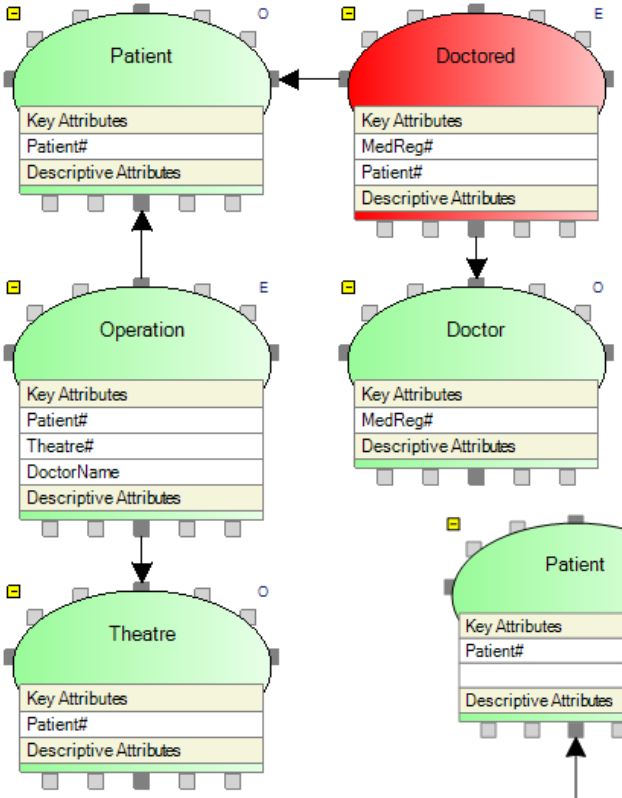
Naming Subset Entities or The Key Attribute Sets of Subset Entities

If a specific and natural Key Attribute Set exists for a Subset Entity, it could be used. Alternatively, the "in the role of" notation is used: ("AttributeName<SupersetEntityName>").



Derivable Entities

An entity can be derived from using the key sets of two or more entities. A derivable entity will normally not be shown on a model except when the derivable entity is part of another functional dependency. The entity Doctored could have been derived from the entity Operation. From the entity Operation it is possible to determine which patients the Doctor has doctored.



Only when, for example, many patients are consulted by many doctors in one room will there be a functional dependency between Doctored and Consulting Room, and would the Doctored entity appear on the model. A more meaningful name could also have been given to the entity Doctored such as Consulted.