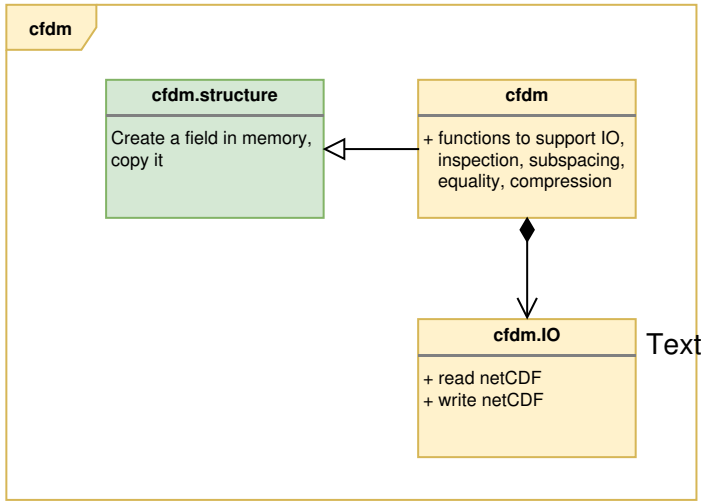
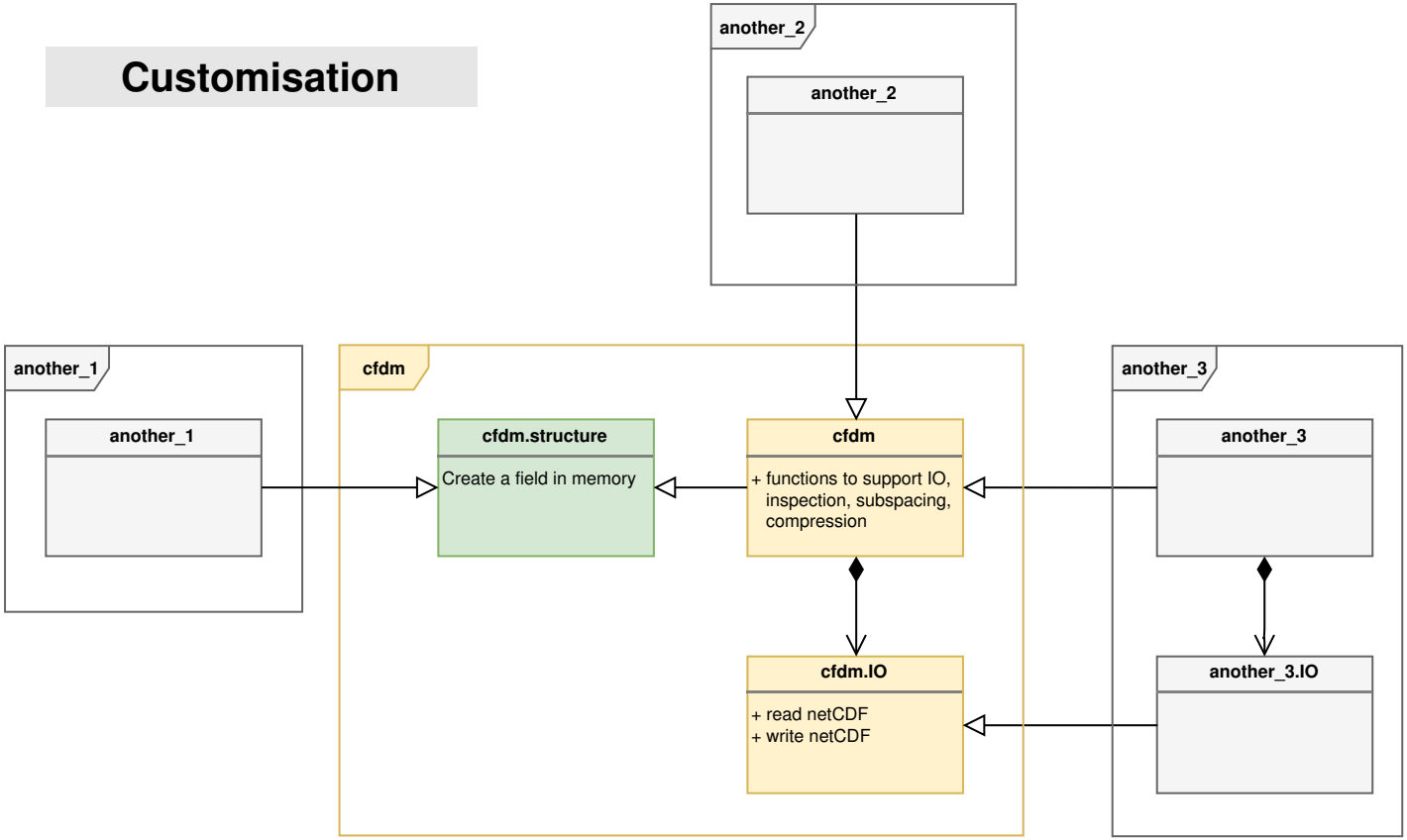
CF data model reference implementation

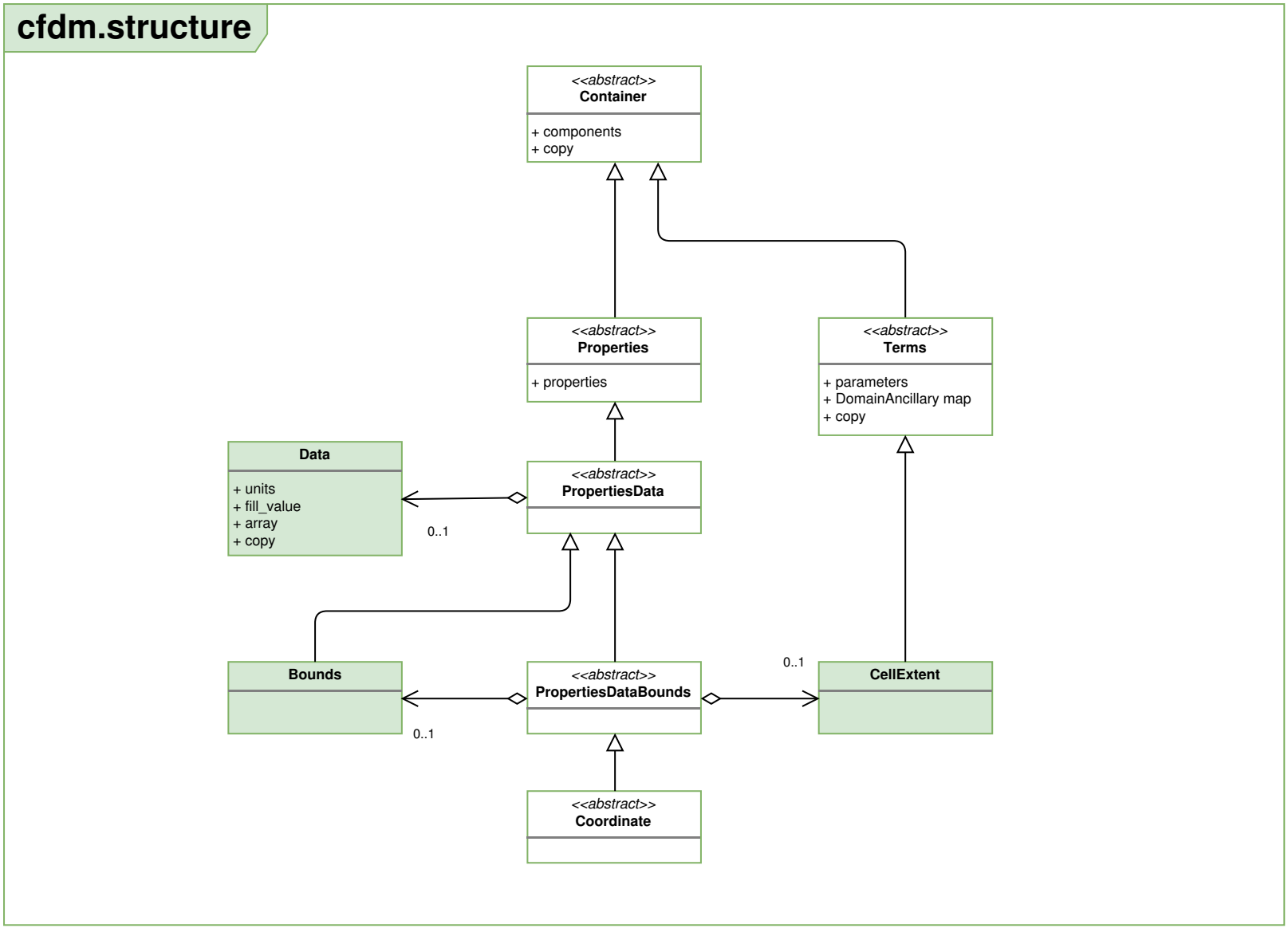


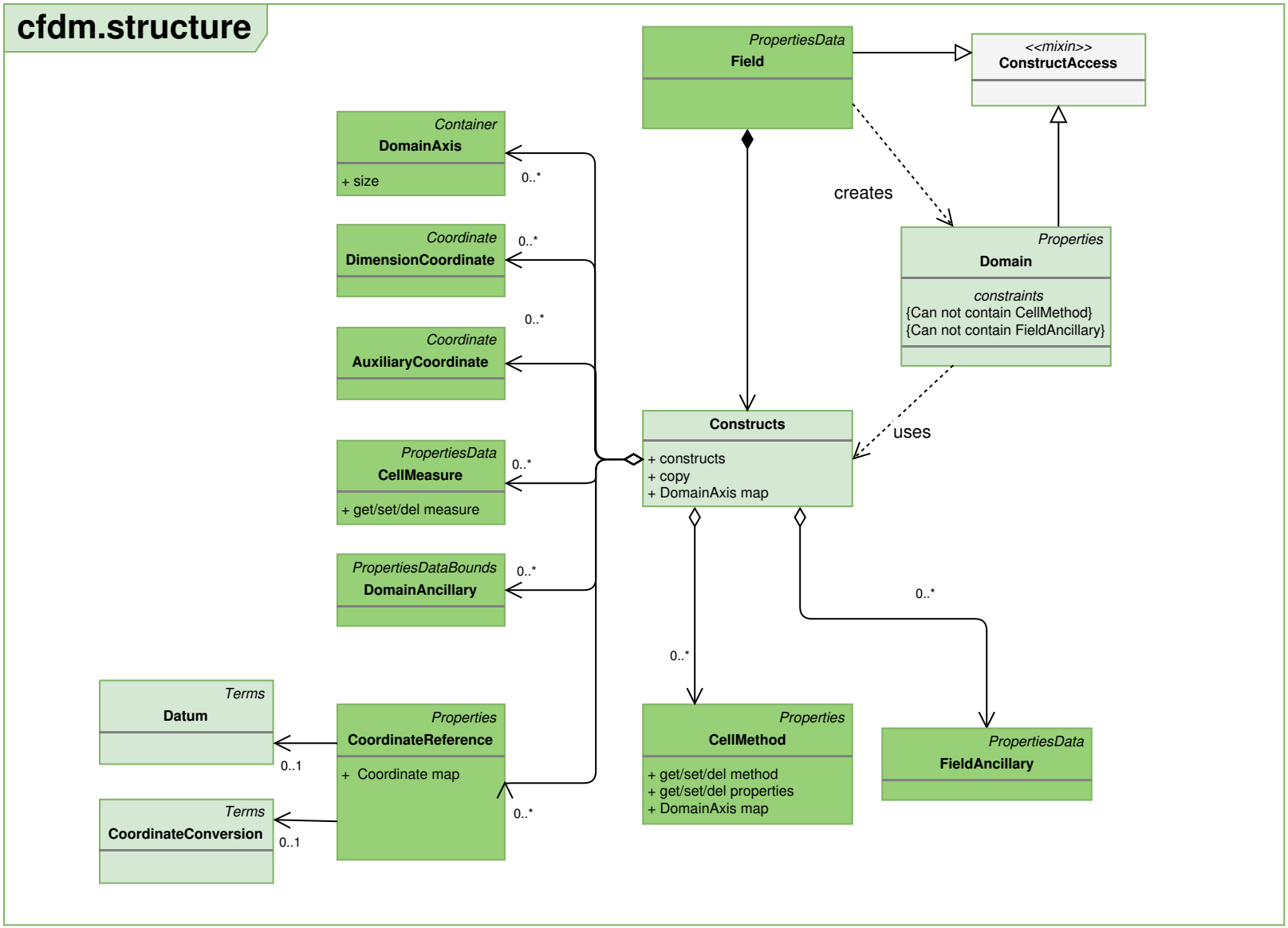

```
>>> import cfdm
>>> f = cfdm.structure.Field()
>>> f.set_property('standard_name', 'iron_growth_limitation_of_picophytoplankton'
>>> print f.get_property('standard_name')
'iron_growth_limitation_of_picophytoplankton'
```

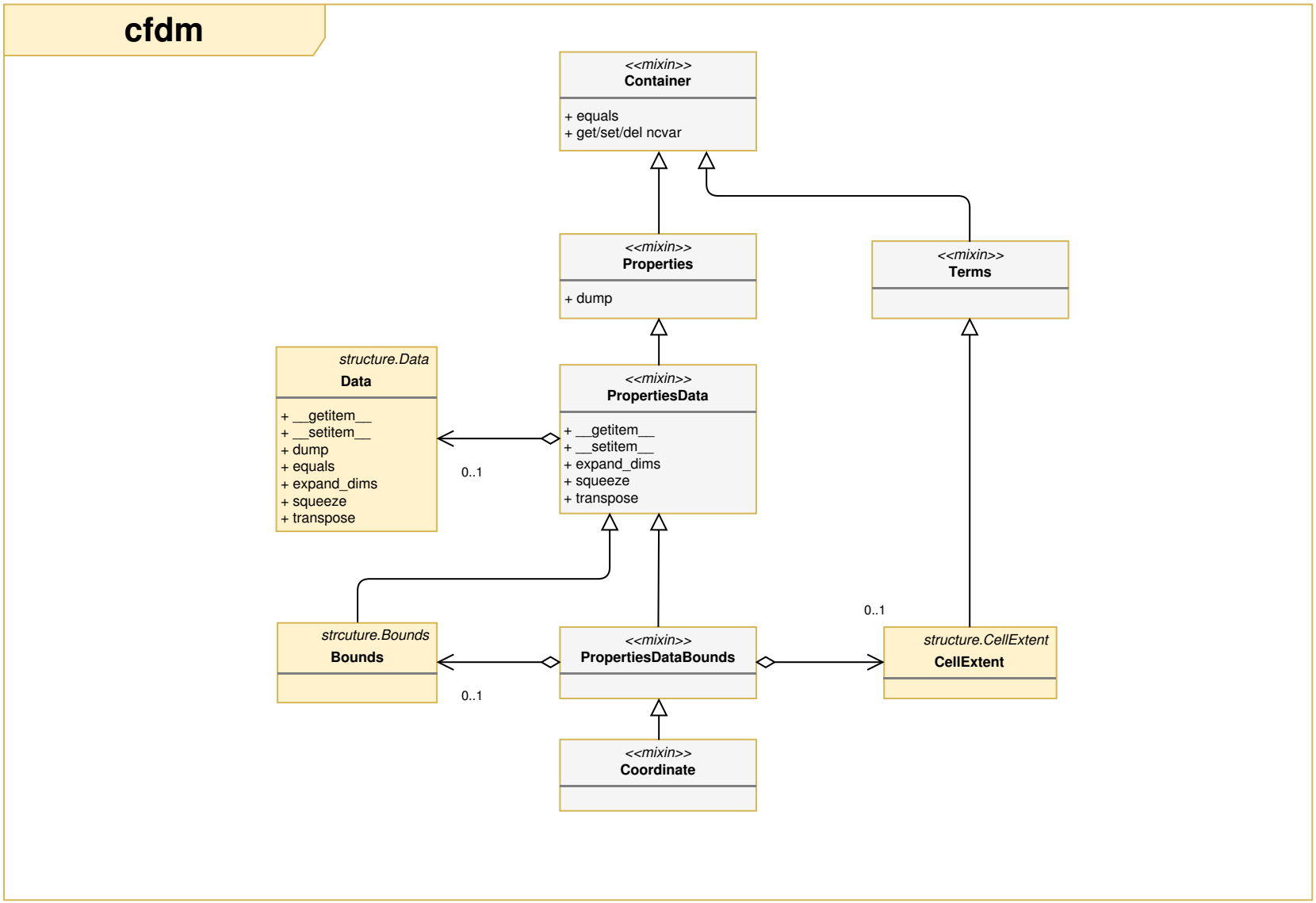
```
>>> import cfdm
>>> f = cfdm.read('file.nc')
>>> print f.equals(f)
True
>>> f[:, 0] = -99
>>> cfdm.write(f, 'newfile.nc')
```

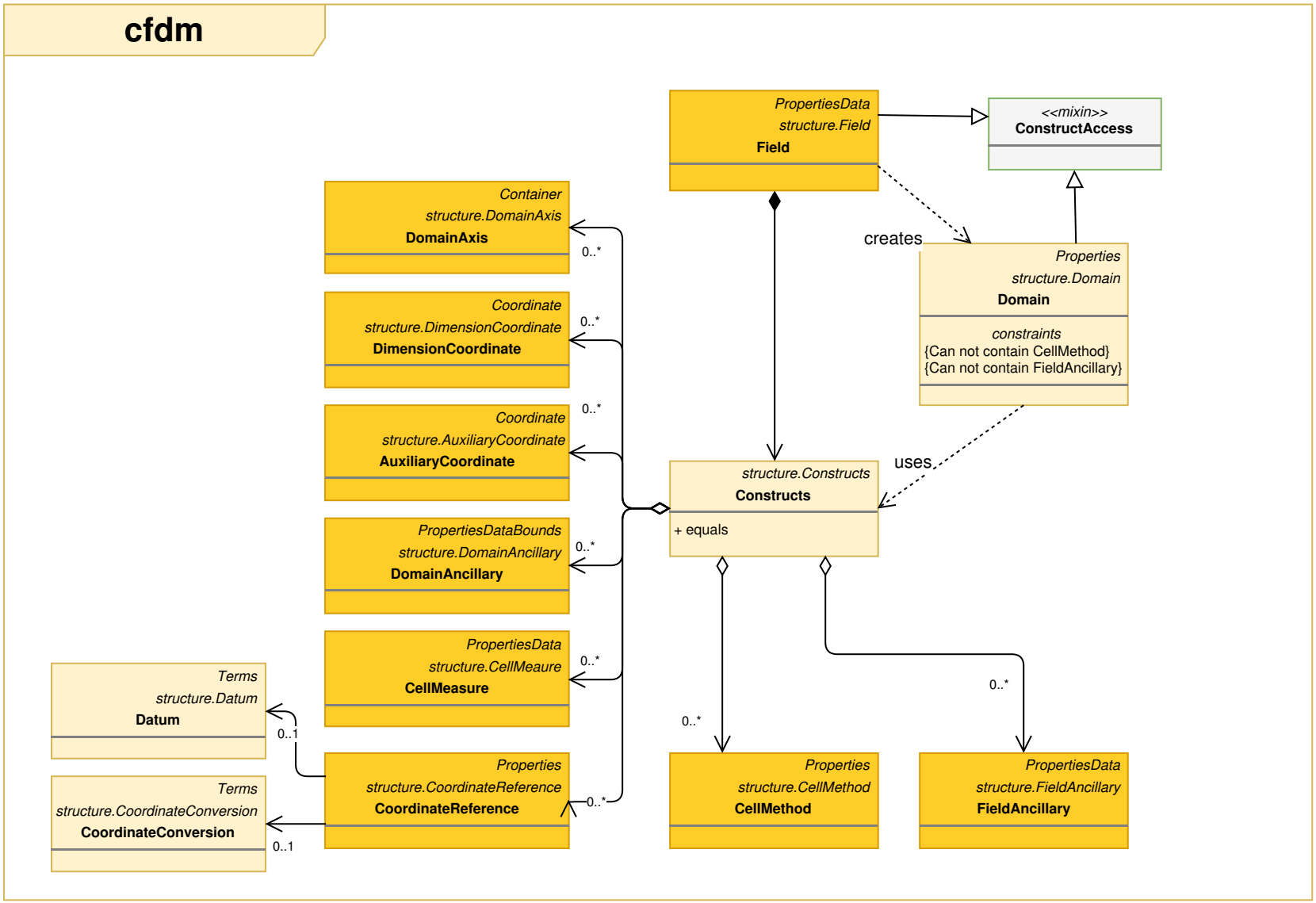
Customisation

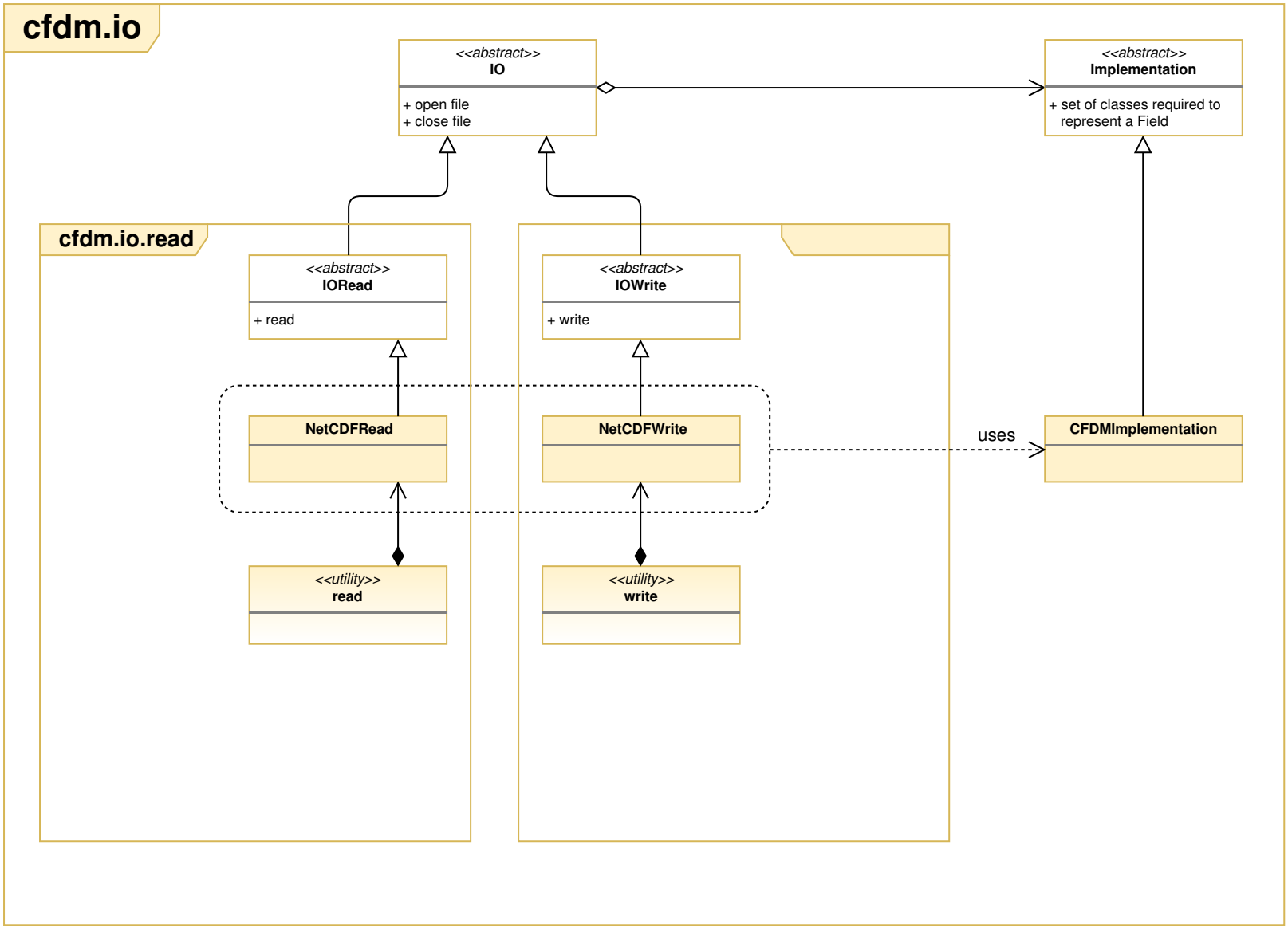


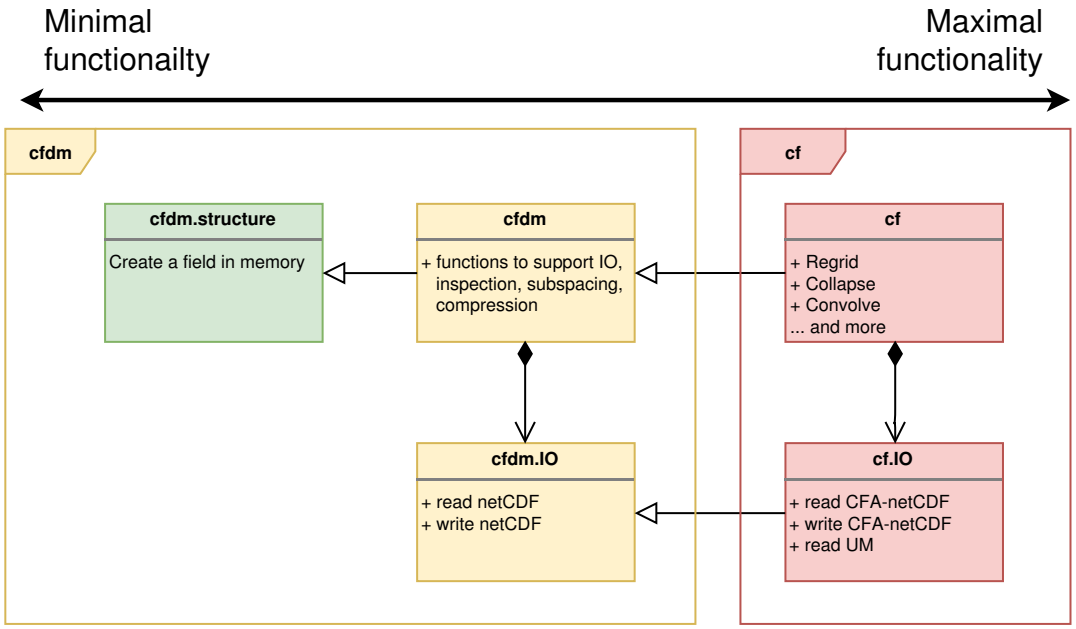

cfdm.structure
<<abstract>>
Container
+ components
+ copy
<<abstract>>
Properties
+ properties
<<abstract>>
Terms
+ parameters
+ DomainAncillary map
+ copy
Data
+ units
+ fill_value
+ array
+ copy
<<abstract>>
PropertiesData
0..1
Bounds
<<abstract>>
PropertiesDataBounds
0..1
0..1
CellExtent
<<abstract>>
Coordinate


cfdm.structure
PropertiesData
Field
<<mixin>>
ConstructAccess
Container
DomainAxis
+ size
0..*
creates
Coordinate
DimensionCoordinate
0..*
Properties
Domain
constraints
{Can not contain CellMethod}
{Can not contain FieldAncillary}
0..*
Coordinate
AuxiliaryCoordinate
Constructs
+ constructs
+ copy
+ DomainAxis map
uses
PropertiesData
CellMeasure
+ get/set/del measure
0..*
PropertiesDataBounds
DomainAncillary
0..*
0..*
0..*
Terms
Datum
Properties
CoordinateReference
+ Coordinate map
Properties
CellMethod
+ get/set/del method
+ get/set/del properties
+ DomainAxis map
PropertiesData
FieldAncillary
0..1
Terms
CoordinateConversion
0..1
0..*


cfdm
<<mixin>>
Container
+ equals
+ get/set/del ncvar
<<mixin>>
Properties
+ dump
<<mixin>>
Terms
structure.Data
Data
+ __getitem__
+ __setitem__
+ dump
+ equals
+ expand_dims
+ squeeze
+ transpose
0..1
<<mixin>>
PropertiesData
+ __getitem__
+ __setitem__
+ expand_dims
+ squeeze
+ transpose
strcuture.Bounds
Bounds
0..1
<<mixin>>
PropertiesDataBounds
0..1
structure.CellExtent
CellExtent
<<mixin>>
Coordinate


cfdm
PropertiesData
structure.Field
Field
<<mixin>>
ConstructAccess
creates
Container
structure.DomainAxis
DomainAxis
0..*
Coordinate
structure.DimensionCoordinate
DimensionCoordinate
0..*
Coordinate
structure.AuxiliaryCoordinate
AuxiliaryCoordinate
0..*
PropertiesDataBounds
structure.DomainAncillary
DomainAncillary
0..*
PropertiesData
structure.CellMeaure
CellMeasure
0..*
Properties
structure.Domain
Domain
constraints
{Can not contain CellMethod}
{Can not contain FieldAncillary}
uses
structure.Constructs
Constructs
+ equals
Terms
structure.Datum
Datum
0..1
Terms
structure.CoordinateConversion
CoordinateConversion
0..1
Properties
structure.CoordinateReference
CoordinateReference
0..*
0..*
0..*
Properties
structure.CellMethod
CellMethod
PropertiesData
structure.FieldAncillary
FieldAncillary

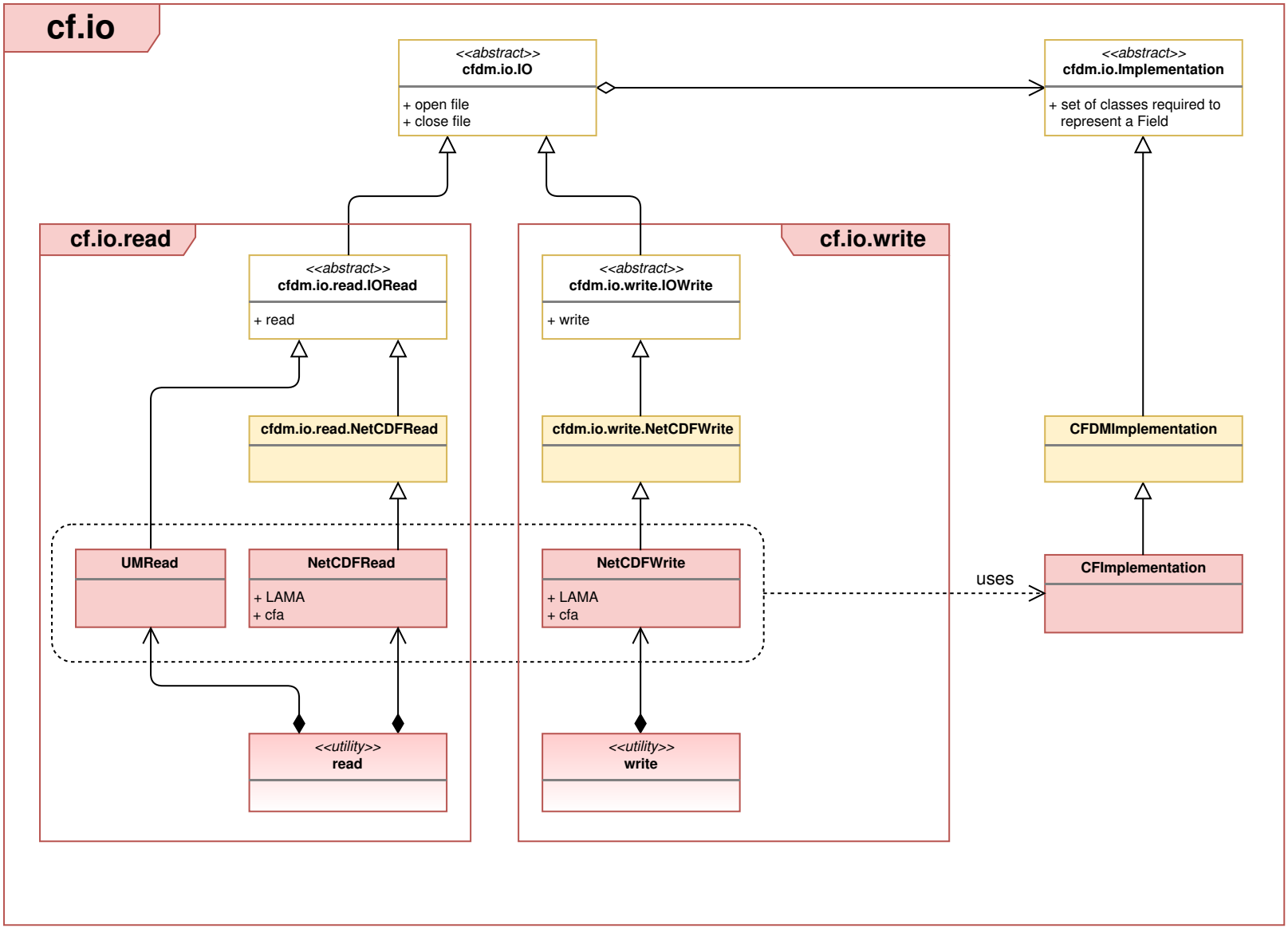

cfdm.io
<<abstract>>
IO
+ open file
+ close file
<<abstract>>
Implementation
+ set of classes required to represent a Field
cfdm.io.read
<<abstract>>
IORead
+ read
<<abstract>>
IOWrite
+ write
NetCDFRead
NetCDFWrite
uses
CFDMImplementation
<<utility>>
read
<<utility>>
write




```
>>> import cfdm
>>> f = cfdm.structure.Field()
>>> f.set_property('standard_name', 'iron_growth_limitation_of_picophytoplankton'
>>> print f.get_property('standard_name')
'iron_growth_limitation_of_picophytoplankton'
```

```
>>> import cfdm
>>> f = cfdm.read('file.nc')
>>> print f.equals(f)
True
>>> f[:, 0] = -99
>>> cfdm.write(f, 'newfile.nc')
```

```
>>> import cf
>>> f = cf.read('file[1-9].pp')
>>> f.standard_name = 'sea_water_practical_salinity'
>>> g = f.collapse('T: variance', weights='T')
>>> g = g.subspace(longitude=cf.wi(-10, 45))
>>> cf.write(g, 'newfile.nc', fmt='CFA4')
```


cf.io
<<abstract>>
cfdm.io.IO
+ open file
+ close file
<<abstract>>
cfdm.io.Implementation
+ set of classes required to represent a Field
cf.io.read
<<abstract>>
cfdm.io.read.IORead
+ read
cf.io.write
<<abstract>>
cfdm.io.write.IOWrite
+ write
cfdm.io.read.NetCDFRead
cfdm.io.write.NetCDFWrite
CFDMImplementation
UMRead
NetCDFRead
+ LAMA
+ cfa
NetCDFWrite
+ LAMA
+ cfa
uses
CFImplementation
<<utility>>
read
<<utility>>
write