DSpace class schema derived from the IFC class schema

Carl Schultz

Date started: 11 May 2010
Release #1

1 Introduction

This document presents the first release of the DSpace class schema as adapted from the IFC class schema. It defines object-oriented classes, inheritance hierarchies, and other relevant architectural domain concepts that provide a schema for DSpace models (i.e. architectural design files).

I have tried to follow the IFC naming conventions to make it easy to understand the mapping between an IFC model and a DSpace model. Specifically, DSpace class names usually take the least abstract IFC class name that is applicable, and DSpace class-attribute names take the equivalent IFC class-attribute names. For example, DsRoot corresponds to IfcRoot; DsWallStandardCase corresponds to IfcWallStandardCase; DsBuildingElement combines IfcBuildingElement and IfcElement; and so on.

The remainder of this document is organised as follows. The next section refers to relevant reading material that can assist in linking the DSpace classes to the appropriate IFC classes. This is followed by a summary of the classes, inheritance hierarchies, and the aggregation relationship in order to introduce some of the more important underlying organisation features of the schema. Section 4 presents the concept of a "project" which manages global design settings and acts as the highest spatial container. Section 5 provides a more complete description of spatial containment relationship and defines some spatial objects. Section 6 presents the building elements that can be used in the current release of DSpace, and Section 7 specifies how objects in a design can be geometrically represented in a plan-view. Section 8 presents the detailed specificaion of the DSpace classes. Finally, Section 9 exhaustively lists the features and concepts that are included in the relevant portions of IFC to provide a roadmap for further DSpace schema developement, and Section 10 provides a tentative schedule for future releases.

2 Relevant reading material

The resources given in this section are very useful for navigating IFC, and may assist in understanding particular DSpace classes (by looking for IFC classes with similar names).

The complete IFC specification (version IFC2x3):
http://www.iai-tech.org/ifc/IFC2x3/TC1/html/index.htm

To navigate to the IFC classes:
- Go to the frame in the upper-left corner (under the heading "Browsing documentation by:") and click "alphabetical listing" ("go->").
- Next go to the frame below (middle-left) and click "Entities".
- Finally, go to the lowest frame (lower-left) and search for the required IFC class name (e.g. "IfcWall").
- The API is at the bottom of each entry.

A useful guide to the IFC is:
http://www.iai-tech.org/downloads/accompanying-documents/guidelines/IFC2x%20Model%20Implementation%20Guide%20V2-0b.pdf

3 Class summary and the DsRoot class

The DSpace schema is an object-oriented model that consists of classes and instances. The root class for all objects in an architectural design is "DsRoot". The root class contains information that is necessary for all object instances that are relevant for describing architectural designs at a high level (as opposed to object instances for geometric modelling such as points and lines) including concrete building elements such as doors and walls, and salient relationships between objects, such as a "decomposition relationship".

The root class defines three key fields. Every object that inherits from DsRoot has:
- a global unique identifier that lasts throughout the entire project lifecycle
- an arbitrary name
- an arbitrary description

In the domain of construction IT physical objects are called "elements" (such as chairs, doors, regions of space, etc.); please note the distinction between the term "object" (i.e. generic class instance which includes things like instances of "relationships") and the term "element" (i.e. class instance that refers to architecture domain objects that have a physical component such as rooms and spaces). Basic concept categories currently relevant to DSpace are:

- project
- spatial organisation (specifically spatial containment and spatial aggregation) and elements that consist purely of "space"
- physical building elements referred to as "products"
- geometric representations of elements (i.e. the geometric shape of a wall in a 2D plan)

Following is the class inheritance hierarchy for the current release of the DSpace schema. Abstract classes (that cannot be instantiated) are unbold and concrete classes are bolded.

- DsRoot
  - DsObject
    - **DsProject**
    - DsProduct <*omit "DsElement" to avoid confusion with IfcElement*>
      - DsSpatialStructureElement
        - **DsBuilding**
        - **DsBuildingStorey**
      - DsBuildingElement <*IfcBuildingElement + IfcElement*>
        - **DsWallStandardCase**
  - DsRelationship
    - **DsRelAggregates**
    - **DsRelContainedInSpatialStructure**

- **DsLocalPlacement** <*attribute of DsProduct - specifies (absolute) location (DsCartesianPoint)*>
- **DsShapeRepresentation** <*attribute of DsProject - list of DsPolylines*>

- **DsPolyline** <*list of DsCartesianPoints*>
- **DsCartesianPoint** <*a pair of reals*>

Aggregation

Some very fundamental relationships between elements are modelled as object instances. In the object hierarchy, note the existence of the class DsRelationship. One key relationship is aggregation, modelled using the DsRelAggregates class. Aggregation defines the part-whole relationship between objects; for example a car can be decomposed into its constituent parts such as an engine, wheels, doors, and so on. Similarly a building can be decomposed into its constituent storeys; for example (in vaguely Java notation):

```
DsRelAggregates aggRelExample = new DsRelAggregates{
        relatingObject    = (DsBuilding) cartesiumBremen
        relatedObjects    = {(DsBuildingStorey) floor1,
                             ...,
                             (DsBuildingStorey) floor5}};
```

Aggregation relationships form a strict hierarchy. The details of when aggregation should be applied are specified in the following sections.

4 Project container and base settings

This section presents the notion of a "project". A project is modelled as an instance of the class DsProject. All design files have exactly one DsProject. The DsProject instance holds global definitions of presentation context (world coordinate system) and the units within the global context. Moreover, with respect to the aggregation hierarchy in a design file, the DsProject instance acts as the uppermost container class; i.e. DsProject can not be combined with other objects to decompose any other object - conversely the DsProject instance is necessarily decomposed into a set of building objects.

Measure types and units

DSpace schema models measuring types (e.g. length-measure, area-measure, etc.; these correspond to IFC classes such as IfcLengthMeasure) and the associated units (e.g. refer to IfcSIUnit). For the current release all available measuring types and the associated units will be hardcoded as follows:

- distance measures (such as length) are specified in metres (m)
- area measures are specified in square metres (m^2)
- volume measures are specified in cubic metres (m^3)
- angular measures are specified in radians (rad)

Representation context

Each element in a design has a geometric representation. These representations are embedded in a representation context which includes a world coordinate system, dimensionality, the direction of true north, and so on. In this release the world coordinate system is hardcoded as follows:

- dimensionality is hardcoded as 2D
- centred on natural origin (0,0)
- x-axis is in the direction of (1,0) and the y-axis is in the direction of (0,1)

5 Spatial structure and space elements

IFC describes spatial structuring as a "breakdown of the project model into manageable subsets according to spatial arrangements". Spatial structures are elements that occupy some region of space (e.g. a building storey) without directly having a physical "body". Spatial structures are composed of other elements (by aggregation, e.g. a building is composed of storeys) and can spatially contain other elements (e.g. a building storey can contain walls).

Spatial structures are modelled by the DsSpatialStructureElement class. Spatial structures they inherit all concepts associated with "DsProducts" (refer to Section 3), notably:

- aggregation and decomposition (described in Section 3)
- geometric placement (described in Section 7)
- element in spatial structure (described in this section)

Two spatial structure classes are defined in this release: DsBuilding and DsBuildingStorey.

Spatial containment

Two important features are aggregation (refer to Section 3) and spatial containment. The distinction is that the aggregation relationship means that the object can be decomposed into constituent parts (e.g. a building is decomposed into a set of storeys) whereas the spatial containment relationship means that the contained object in inside the container object (e.g. a chair is contained inside a room).

The DSpace class for the spatial containment relationship is DsRelContainsInSpatialStructure.

Building the spatial structures

This section describes how the spatial organisation of a design (particularly aggregation and spatial containment) is modelled in DSpace. DsProject, DsBuilding, and DsBuildingStorey are mandatory for all DSpace models. They are hierarchically organised such that DsProject is decomposed into a number of DsBuildings, and each DsBuilding is decomposed into a number of DsBuildingStoreys. In this release each DsProject contains exactly one DsBuilding and exactly one DsBuildingStorey.

DsBuildings are associated directly with a DsProject (using the DsRelAggregates relationship where the DsBuilding instance is one of the relatedObjects). DsBuildings are decomposed into DsBuildingStoreys (using the DsRelAggregates relationship where the DsBuilding instance is the relatingObject). DsBuildings are not directly associated with a geometry as this is provided by the constituent elements.

DsBuildingStoreys are associated directly to a DsBuilding (using the DsRelAggregates relationship where the DsBuildingStorey instance is one of the relatedObjects). DsBuildingStoreys are not directly associated with a geometry as this is provided by the constituent elements. All DsBuildingElements are assigned to the DsBuildingStorey (using the DsRelContainsInSpatialStructure relationship where the DsBuildingStorey instance is the relatingStructure).

6 Building elements

Building elements are the major functional parts of a building. They are logically contained in spatial structures and can decompose other elements (i.e. defining an assembly of building components). The DSpace class for modelling building elements is DsBuildingElement. All building elements inherit the following properties from DsProduct

- aggregation and decomposition (described in Section 3)
- geometric placement (described in Section 7)
- element in spatial structure (described in Section 5)

Additionally, building elements have a number of properties that spatial structures do not have such as having openings (holes), being able to fill holes and providing boundaries to spatial structures. Each building element has a geometric representation as presented in Section 7.

The current release only has one building element subclass: DsWallStandardCase. These walls are defined as have a straight-line wall path and a constant height.

7 Shape representation of elements

Each concrete subclass of DsProduct has a geometric representation. The two main pieces of information are:

- object placement
- geometric shape describing the object (currently we are focusing only on the plan view)

Object placement

DsProduct instances need to be given a location to mark their physical position in the design with respect to the world coordinate origin. The DSpace model class for representing object placement is DsLocalPlacement. In this release all DsProducts are placed with respect to absolute coordinates (relative to the default world coordinate system).

Object placement is specified by a geometric point. Points are specified using cartesian coordinates with the class DsCartesianPoint. This point is interpreted as providing the *translation* for the shape representation (discussed below). For example, if the shape representation is a square with the coordinates

((0,0), (0,5),(5,5), (5,0), (0,0))

then an object placement point of (10, 10) simply translates the squares coordinates into

((10,10), (10,15), (15,15),(15,10), (10,10)).

Shape representation

Each DsProduct has a plan-view geometric representation, that is, a numerically defined "shape". The DSpace model class for representing the geometric shape is DsShapeRepresentation. Each DsShapeRepresentation contains a list of polylines (DsPolyline) where a polyline is a list of *n* points that specify *n*-1 line segments. In an instance of DsShapeRepresentation each polyline defines a simple closed one-piece region; specifically each polyline has the following properties:

- simple (does not cross itself)
- loop (the endpoint of the polyline is the same as the start point)

Moreover the region defined by the polyline contains its boundary, i.e. the region also includes the polyline.

8 Detailed class specifications

This section presents a detailed specification for each DSpace class.

Architectural design

DsRoot

Class type:	abstract
Description:	the root class for all objects in an architectural design; contains information necessary for every design object
Subclass of:	(none)
Superclass of:	DsObject, DsRelationship
Attributes:	globalId (String 22 characters, case sensitive) name (String) description (String)
Relationships:	(no direct)

DsObject

Class type:	abstract
Description:	entities in an architectural design (in contrast to "relationship" instances)
Subclassof:	DsRoot
Superclassof:	DsProject, DsProduct
Attributes:	(none)
Relationships:	DsRelAggregates (via relatingObject, relatedObjects)

DsProject

Class type:	concrete
Description:	Defines design file's representation context and other global settings
Subclassof:	DsObject
Superclassof:	(none)
Attributes:	(none)
Relationships:	(no direct)

DsProduct

Class type:	abstract
Description:	defines objects that have a spatial context
Subclassof:	DsObject
Superclassof:	DsSpatialStructureElement, DsBuildingElement
Attributes:	objectPlacement (DsLocalPlacement)
	representation (DsShapeRepresentation)
Relationships:	DsRelContainedInSpatialStructure (via relatedElements)

DsSpatialStructureElement

Class type:	abstract
Description:	Element that occupies a region of space without directly consisting of a physical body
Subclassof:	DsProduct
Superclassof:	DsBuilding, DsBuildingStorey
Attributes:	(none)
Relationships:	DsRelContainedInSpatialStructure (via relatingElements)

DsBuilding

Class type:	concrete
Description:	decomposes the project, and is decomposed into building storeys
Subclassof:	DsSpatialStructureElement
Superclassof:	(none)
Attributes:	(none)
Relationships:	(no direct)

DsBuildingStorey

Class type:	concrete
Description:	decomposes buildings, and is the lowest level container (i.e. contains all building elements)
Subclassof:	DsSpatialStructureElement
Superclassof:	(none)
Attributes:	(none)
Relationships:	(no direct)

DsBuildingElement

Class type:	abstract
Description:	major functional units of a design; distinct from spatial structures by being able to have openings, fill holes and provide boundaries for spatial structures
Subclassof:	DsProduct
Superclassof:	DsWallStandardCase
Attributes:	(none)
Relationships:	(no direct)

DsWallStandardCase

Class type:	concrete
Description:	wall with a straight-line wall path and constant height
Subclassof:	DsBuildingElement
Superclassof:	(none)
Attributes:	(none)
Relationships:	(no direct)

DsRelationship

Class type:	abstract
Description:	associations between entities in an architectural design (in contrast to the actual entities)
Subclassof:	DsRoot
Superclassof:	DsRelAggregates, DsRelContainedInSpatialStructure
Attributes:	(none)
Relationships:	(no direct)

DsRelAggregates

Class type:	concrete
Description:	defines a part-whole relationship between objects
Subclassof:	DsRelationship
Superclassof:	(none)
Attributes:	relatingObject (DsObject)
	relatedObjects (set <DsObject>)
Relationships:	(no direct)

DsRelContainedInSpatialStructure

Class type:	concrete
Description:	defines spatial containment, where a collection of elements can be inside some spatial element
Subclassof:	DsRelaion
Superclassof:	(none)
Attributes:	relatingStructure (DsSpatialStructureElement)
	relatedElements (set <DsProduct>)
Relationships:	(no direct)

<u>Product representation</u>

DsLocalPlacement

Class type:	concrete
Description:	positions a DsProduct instance with respect to the default absolute world coordinates; the point is interpreted as a geometric translation of the coordinates specified in the shape representation
Subclassof:	(none)
Superclassof:	(none)
Attributes:	relativePlacement (DsCartesianPoint)
Relationships:	(no direct)

DsShapeRepresentation

Class type:	concrete
Description:	plan-view geometry of the DsProduct defined as a closed region bounded by a Jordan curve
Subclassof:	(none)
Superclassof:	(none)
Attributes:	items (set <DsPolyline>)
Relationships:	(no direct)

<u>Geometric modelling</u>

DsPolyline

Class type:	concrete
Description:	a curve consisting of *n* points that define *n*-1 contiguous line segments; must have at least 2 points
Subclassof:	(none)
Superclassof:	(none)
Attributes:	points (list <DsCartesianPoint>)
Relationships:	(no direct)

DsCartesianPoint

Class type:	concrete
Description:	a point on a 2D grid of reals
Subclassof:	(none)
Superclassof:	(none)
Attributes:	x (real), y (real)
Relationships:	(no direct)

9 Features adapted from IFC by class

This sections presents a rough list of features derived directly from IFC that are being considered for the DSpace schema. This list includes features that are beyond the scope of the current release.

Blue features are required for this release.
Red features are currently determined to be out of scope for what is needed in the DSpace project.
Black features are currently determine to appear in some future release.

- DsRoot
  - globally unique identifier
  - name and description
- DsProject
  - measure types and units
    - global unit assignment
    - local unit assignment
  - representation context <DsRepresentationContext>
    - defining the world coordinate system
    - integer dimension count

        - degree of precision afforded by geometric models (a real)
        - specifying true north
        - context type for geometric representation context (plan view, model view, etc.)
- DsObject
    - element quantities such as area, length, etc.
    - handling custom object “types”
- DsSpatialStructureElement
    - complex and partial composition types (e.g. Building sections and partial building storeys)
    - using IfcSite and IfcSpace (i.e. for rooms)
    - each building storey usually declares its own coordinate system positioned relative to the next higher spatial structure (same with spaces, buildings, and sites) - this applies to aggregates in general
    - allowing more building storeys
    - allowing more buildings
    - DsSpace light weight classifications (e.g. ISO standards for basic room types like kitchen)
    - DsZones to group DsSpaces
- DsMappedItem
    - DsGeometricSet <set of DsPolyLines> to generalise shapes
- DsBuildingElement
    - “has openings” relationship - i.e. openings and recesses in walls
    - “fills void” relationship - i.e. a window or a door can fill the opening in a wall
    - “provides boundary” relationship
    - “is connected to” relationship
    - DsWallStandardCase
        - standard walls with straight line path
        - standard walls with circular wall path
        - constant height
        - changing height
        - non-standard walls (wall path is neither a straight line nor a circular path)
    - DsDoor
    - DsWindow
    - DsStair
    - DsStairFlight
    - DsBeam
    - DsColumn
    - DsRamp
    - DsRampFlight
    - DsSlab
    - DsRoof
- DsRelationship
- DsRelAggregates
- DsRelContainedInSpatialStructure
- DsLocalPlacement
    - relative placement (specifying another object’s DsLocalPlacement)
    - absolute placement (leaving relative placement unspecified)
    - direction (as a direction ratio) - conventions must be applied to determine the meanings of direction for each object, e.g. DsDoor direction is the orientation that the door swings out towards
- DsProductDefinitionShape
    - allows an object to have multiple representations
- DsShapeRepresentation
    - allow a set of polylines that define Jordan curves - each curve specifies the boundary of a closed region
    - use regions with other properties such as open regions
    - allow other shapes
- DsPolyline
- DsCartesianPoint
    - 2D
    - 3D

10 Feature release schedule