Binary Connector Subsystem

This subsystem describes the anatomy of a Binary Connector which connects from a position on one Node face to a different position on the same or another Node face. This includes the use of Tertiary Stems which branch off from a Binary Connector to connect a third Node face position forming a T-shaped line. Binary Connectors may form a straight line between Node faces or be bent at one or more corners.

Relationship numbering range: R100-R149

Class Descriptions

Anchored Binary Stem

This is an Anchored Stem that is one of the opposing (non-Tertiary) Stems in a Binary Connector.

Attributes

ID

Same as**Binary Stem.ID**

Connector

Same as**Binary Stem.Connector**

Identifiers

1. **ID** + **Connector**

Bend

A Bend is the line drawn between two Corners in a Bending Binary Connector. One is drawn for each user specified Path.

Attributes

T location

Type: Same as **Corner.Location**

P location

Type: Same as **Corner.Location**

Path

Type: Same as **Path.Sequence**

Connector

Same as **Path.Connector** and **Corner.Connector**

Identifiers

1. **T location** No two Bends can share the same Corner
2. **P location** Same reasoning as #1
3. **Path** + **Connector**

Bending Binary Connector

This is a Binary Connector that must turn one or more corners to connect its opposing Binary Stems. In such a case the two Binary Stems will be counterparts and we can arbitrarily start drawing a line from one of the Counterpart Binary Stems to the other. In fact, we could start from both ends and work toward the middle or start from the middle and work our way out. So the terms "start" and "end" could just as easily have been labeled "A" and "B".

Attributes

ID

Same as **Binary Connector.ID** and, for each of the two Counter Part Binary Stems, **Counterpart Binary Stem.Connector**

Start stem

Same as **Counterpart Binary Stem.ID**

End stem

Same as **Counterpart Binary Stem.ID** and not the same as the**Start stem** value

Identifiers

1. **ID**

Binary Connector

The defining property of a Binary Connector is that it connects two points, each on some Node face. Common examples are a transition from one state to another on a state diagram or an association between two classes on a class diagram.

While each Binary Stem must be in a unique position (Stems never overlap) both Binary Stems may be on the same Node or even on the same Node face in a Binary Connector. For example, a state may transition to itself or a class may be associated with itself via a reflexive association.

A Binary Connector may also include a Tertiary Stem which attaches to a third Node face position and then extends in a straight line to some point on the line connecting the two Binary Stems. Since the Tertiary Stem is a straight line, it cannot attache to the same Node as either of the Binary Stems in the Binary Connector. So the Tertiary Stem will be attached to the face of a Node that has neither of the Binary Stems attached.

At present, the only known example of a Tertiary Stem's usage is to represent an association class relationship on a class diagram.

Attributes

ID

Same as **Connector.ID**

Identifiers

ID

Binary Stem

This is a Stem that is one of the opposing (non-Tertiary) Stems in a Binary Connector. It's position may be specified by the user as an anchor point, or computed in the case of a Floating Binary Stem.

Attributes

ID

Same as**Binary Stem.ID**. Also the union of the ID values in each subclass.

Connector

Same as**Binary Stem.Connector** Also the union of the Connector values in each subclass.

Identifiers

1. **ID** + **Connector**

Corner

This is a point on the Canvas where two lines of a Bending Binary Connector meet at a right angle. Corners are not specified by the user, they are computed from user specified anchor positions and Paths.

Attributes

Location

The computed Canvas x and y coordinate of the Corner

Type: Position

Connector

Same as **Bending Binary Connector.ID**

Identifiers

1. **Location** (No two Corners may overlap)
2. **Location** + **Connector** (super identifier to support R111)

Counterpart Binary Stem

When a Binary Connector bends at least once the user must specify an anchor position for each Binary Stem. Since this means that we must have a pair of Anchored Binary Stems, we can think of these as required counterparts within such a Binary Connector.

Attributes

ID

Same as**Anchored Binary Stem.ID**

Connector

Same as**Anchored Binary Stem.Connector**

Identifiers

1. **ID** + **Connector**

Floating Binary Stem

The point where this Stem meets a Node Face is determined by drawing a straight line across from a Projecting Binary Stem in a Binary Connector. It “floats” because because the face position is computed rather than being specified by the user.

Attributes

ID

Same as**Binary Stem.ID**

Connector

Same as**Binary Stem.Connector**

Identifiers

1. **ID** + **Connector**

Lane

The corridor formed by either a Row or Column in the Grid. For the purpose of drawing a line as part of a Connector, Rows and Columns are regarded similarly.

Attributes

Number

Type: Same as either**Column.Number or Row.Number**

Orientation

Type: Row_Column `:: [ row | column ]`

Identifiers

Number + **Direction**

Since you can have both a Row and Column with the same number in the Grid, we need the direction to distinguish them.

Path

If a Bending Binary Connector requires more than one Corner, it will be necessary for the user to specify where to place each Corner to Corner stretch. Depending on the orientation, either a Row or Column is chosen along with an alignment preference.

Attributes

Connector

Same as **Bending Binary Connector.ID**

Sequence

Paths are sequenced from the Node in the T position toward the Node in the P position.

Type: Ordinal

Lane

Type: Same as **Lane.Number**

Direction

Type: Same as **Lane.Direction**

Rut

We can imagine the Path guided a long a rut somewhere in the Lane. In a horizontal lane this could be the center, top or bottom. Finer gradations in a Lane are possible. For now only three rut positions will be available, but finer gradations should eventually be supported.

Type: Lane Placement

Identifiers

Sequence + **Connector**

Paths are numbered uniquely within each Connector

Projecting Binary Stem

This is an Anchored Binary Stem participating on one of the opposing (non-Tertiary) sides of a Binary Connector. It could be one component of a pair of opposing Anchored Binary Stems in the case where the Binary Connector bends around at least one corner or a Projecting Binary Stem that establishes the position of it opposing Floating Binary Stem

Attributes

ID

Same as**Anchored Binary Stem.ID**

Connector

Same as**Anchored Binary Stem.Connector**

Identifiers

1. **ID** + **Connector**

Straight Binary Connector

This is a Binary Connector drawn as a single straight vertical or horizontal line. Since the line is straight, only one of its Binary Stems has an anchor position specified by the user. The opposite Binary Stem will be placed where the straight line projecting from the Anchored Binary Stem intersects the target Node face. At that point a Floating Binary Stem is drawn which won't necessarily line up with any specific Face Placement position on the Node face.

The anchored Stem is called a Projecting Binary Stem with the non-anchored Stem referred to as a Floating Binary Stem.

Attributes

ID

Same as **Binary Connector.ID**, **Floating Binary Stem.Connector** and **Projecting Binary Stem.Connector**

Floating stem

Same as **Floating Binary Stem.ID**

Projecting stem

Same as **Projecting Binary Stem.ID**

Identifiers

ID

Tertiary Stem

Drawn from a Node face to the middle of a Binary Connector where the root end is on the Node and the vine end touches the Binary Connector between its two Opposing Stems.

Attributes

ID

Same as **Anchored Stem.ID**

Connector

Same as **Anchored Stem.Connector** and **Binary Connector.ID**

Identifiers

1. **ID** + **Connector**

Relationship Descriptions

R100 / 1:1

Projecting Binary Stem, establishes x or y coordinate of *one* **Floating Binary Stem**

Floating Binary Stem gets x or y coordinate from *one* **Projecting Binary Stem**

If a Binary Connector is unbent (straight) we want to specify an anchor position on one Node face and then just draw the Connector line straight across to stop on the opposite Node face. What we don't want to do is try to connect anchor to anchor since that will lead to a diagonal line if the anchors on each side aren't on the same x or y axis.

So we pair an Anchored Binary Stem which we will call the Projecting Binary Stem with a non-Anchored Binary Stem which we call the Floating Binary Stem. Thus the x or y value of the Floating Binary Stem is strictly determined by that of its paired Projecting Binary Stem anchor position. This pairing then forms a Straight Binary Connector.

Formalization

Straight Binary Connector association class

R101 / Generalization

Anchored Binary Stem is a **Projecting Binary Stem**or **Counterpart Binary Stem**

These are the two roles played by a Binary Stem that has a user specified anchor position. In the projecting case, the Stem serves to establish the x or y coordinate of a line shared by a corresponding Floating Binary Stem. In the counterpart case, two Anchored Binary Stems are the terminating points of a line bent at least once.

Formalization

The identifier in each of the subclasses referring to the superclass identifier.

R102 / 1:M

Bending Binary Connector turns at right angle on *one or many* **Corner**

Corner is a right angle turn of *one* **Bending Binary Connector**

By definition, a there is at least one right angle turn in a Binary Bending Connector and hence, one Corner.

Formalization

Referential attribute in the Corner class

R103 / Generalization

Binary Connector is a **Straight Binary Connector**or **Bending Binary Connector**

A Straight Binary Connector is a single horizontal or vertical line that connects both of its non-tertiary Stems. Only one of the two non-tertiary Stems is an Anchored Stem since the opposing Stem is positioned based on the intersection of the connector line and the opposing Node face. Thus the face position of only one Stem, the Anchored Stem, need be specified by the user.

A Bending Binary Connector has at least one corner and requires all of its Stems to be Anchored Stems (fixed on user specified node face positions).

Formalization

The identifier in each of the subclasses referring to the superclass identifier

R104 / 1:1

Counterpart Binary Stem, starts line toward *one* **Counterpart Binary Stem**

Counterpart Binary Stem ends line from *one* **Counterpart Binary Stem**

In a Bending Binary Connector, a line is drawn between two Counterpart Binary Stems. The terms "start" and "end" are used to establish an arbitrary ordering of the Bends so that we can refer to a next or previous Bend while copumting and drawing the lines.

Formalization

Bending Binary Connector association class

R105 / 1:M

Bending Binary Connector turns at right angle on *one or many* **Bend**

Bend is a right angle turn of *one* **Bending Binary Connector**

The line forming a Bending Binary Connector must, by definition, bend at least once. At each Bend the line turns 90 degrees in either direction and proceeds to the end Stem or to the next Bend.

We would like most of the connector lines in our model diagrams to be straight as much as possible. This can be achieved more easily for non-Binary Connectors. For now at least bending is only supported for Binary Connectors and, therefore, a Bend can only exist as part of a Binary Connector.

Formalization

Referential attributes in the Bend class

R107 / 1:Mc

Bending Binary Connector takes *zero, one or many* **Path**

Path is taken by *one* **Bending Binary Connector**

In the simplest and most common case, a Bending Binary Connector turns at only one point forming a single Corner. In this case there is no need for the user to specify a Path as the anchor positions on each Stem establish the single Corner location. You just find the intersection of the lines projecting from each Stem.

When more than one Corner is desired, the user must choose where to place each pair of Corners. Consider two Nodes in the same Row where the Connector will be drawn between the top face of the T node to the top face of the P node. Each Corner will lie somewhere above each Node on the same y coordinate, since we want a straight line. But where is the y coordinate? One Row above? Two Rows above? It's up the the user to decide.

A Path represents both the choice of a Row or Column (Lane) and the alignment within that Lane.

The number of Paths that must be specified is equal to one less than the number of desired Corners in the Bending Binary Connector. So, zero in the case of a single Corner as previously discussed and then incrementing from there.

A Path is defined specific to its Bending Binary Connector. A variety of constraints will prevent two Paths from different Connectors from overlapping, such as the enforcing the uniqueness of Corner coordinates.

Formalization

Referential attributes in the Path class

R109 / Generalization

Binary Stem is a **Floating Binary Stem**or **Anchored Binary Stem**

A Stem used in a Binary Connector may or may not be anchored (user specified). In the case of a Straight Binary Connector, one will be anchored with the other left floating (derived from the anchored Stem). With a Bending Binary Connector each Stem must be anchored.

Formalization

The identifier in each of the subclasses referring to the superclass identifier. Also the superclass identifier is defined as the union of the corresponding identifiers in each of the subclasses.

R110 / 1:1c

Tertiary Stem connects to the middle of *one* **Binary Connector**

Binary Connector connects with *zero or one* **Tertiary Stem**

A third Node face position may be connected into a Binary Connector, effectively making it tertiary. Rather than define a new kind of Connector we just say that a Tertiary Stem may or may not latch onto the middle of any given Binary Connector. This is because the properties of a Binary Connector, bent or straight, are not affected by the existence of any optional third Stem.

When we say "middle of" we mean anywhere between the vine ends of the Binary Connector's two Binary Stems.

Formalization

Referential attribute in the Tertiary Stem class

R111 / 1c:1c

Corner is toward the P/T anchor of *zero or one* **Corner**

When there are more than two Corners in a Bending Binary Connector the Corners are connected in sequence proceeding from the T node to the P node. This establishes an arbitrary but consistent sequence for the purpose of determining how all of the Bends are interconnected. So if we know which Node is designated as T, we can proceed from one Path to the next filling in Bends to get to the P node.

For a Bending Binary Connector with only one Corner there are no Bends.

Formalization

Referential attributes in the Bend association class

R112 / 1:1

Bend is drawn along *one* **Path**

Path establishes line of *one* **Bend**

Whereas a Path is a user specified request for the placement of a line, a Bend is the actual line computed between two Corners.

For each Path specified by the user it is necessary to compute the corresponding Bend so that it can be drawn.

Formalization

Referential attribute in the Bend class