Gittok Lecture Note

07 空間スキーマ

太田守重

2014

地物は位置，形（かたち），そして形同士の連結関係など，
空間的な性質をもつ.

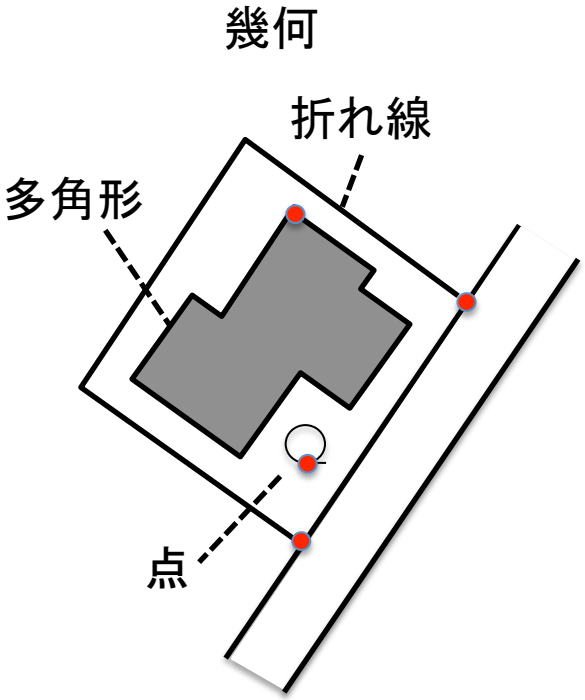
具体的にはどんな種類があるのか.
どのように記述するか.
XMLではどのように，表記するか.

空間スキーマとは

地物の空間特性を記述するスキーマ

初期の地理情報システムでは，典型的な空間特性は，地物の形状を表現する点，折れ線及び多角形といった，単純な幾何図形であった.

今日では，これらを含む幾何および幾何同士の接続関係をあわせて，空間属性と呼ぶ.



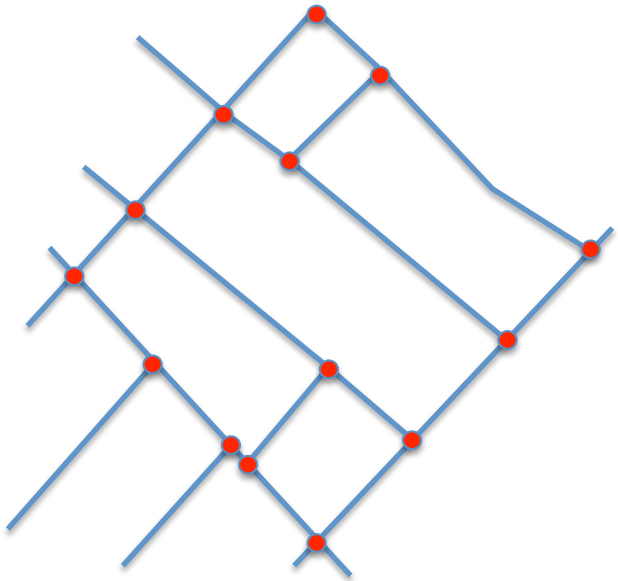
接続関係

孤立した点には線は接続しない.
折れ線同士は交点で接続する.
多角形は境界線で囲まれる.

地物の空間的な性質

地理院地図

道路網は道路と交差点からなる.



幾何
道路の中心線は，曲線（折れ線）
交差点は，点

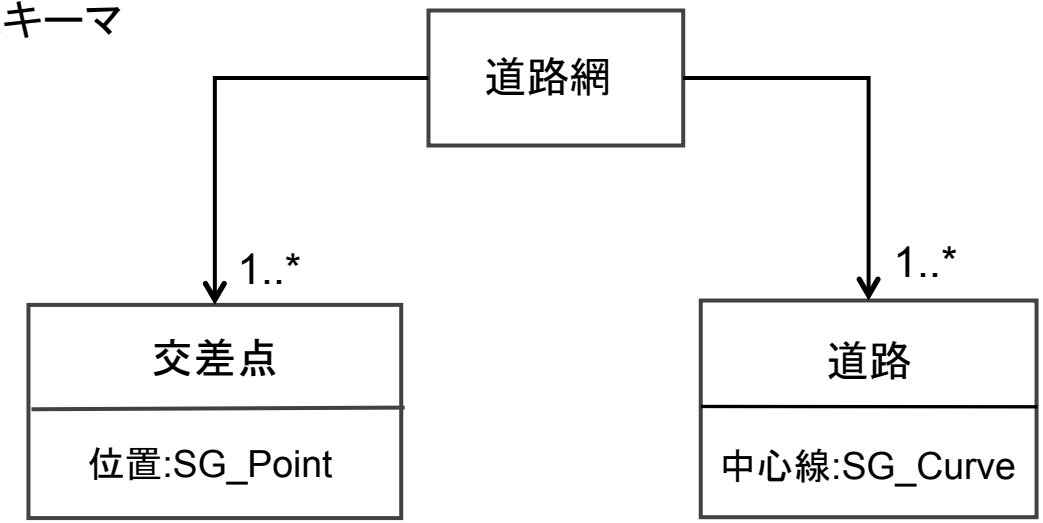
接続関係
交差点で複数の中心線が接続する.

応用スキーマに見られる空間属性

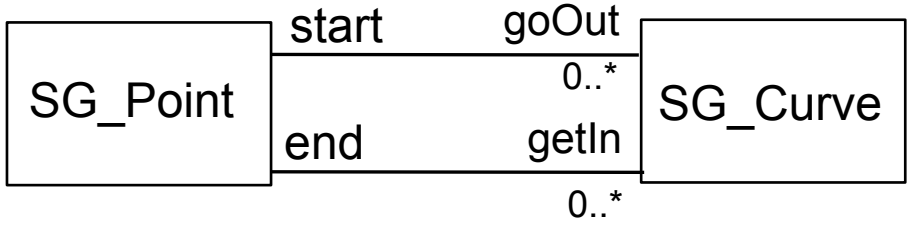
空間属性は，応用スキーマの中で空間属性のデータ型になる．しかも，空間属性は空間スキーマの要素として独自の構造をもつ．

右の例では，応用スキーマでは交差点道路の位置と道路の中心線は関係ないようにみえるが，
空間スキーマの中で接続関係をもつ．

応用スキーマ

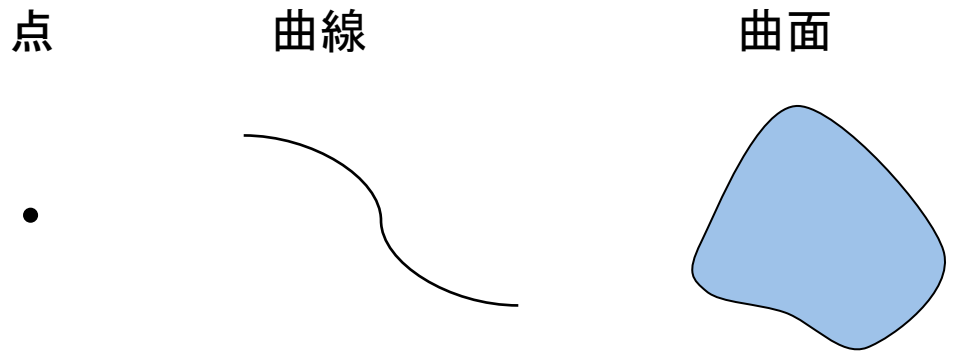


空間スキーマ

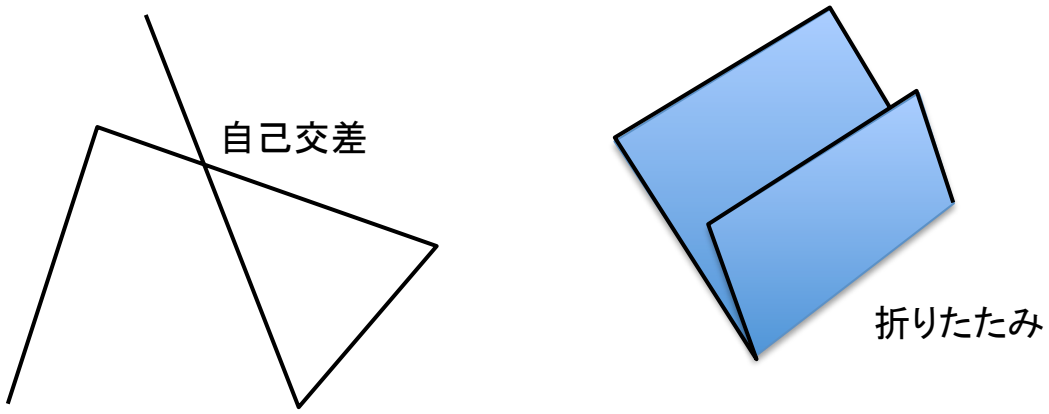


幾何プリミティブ

2次元空間中にある，それ以上不可分な幾何要素を幾何プリミティブという．幾何プリミティブには点，曲線，そして曲面がある．ここで，曲線及び曲面は自己交差や折り畳みがない，単純な形のものを指す．



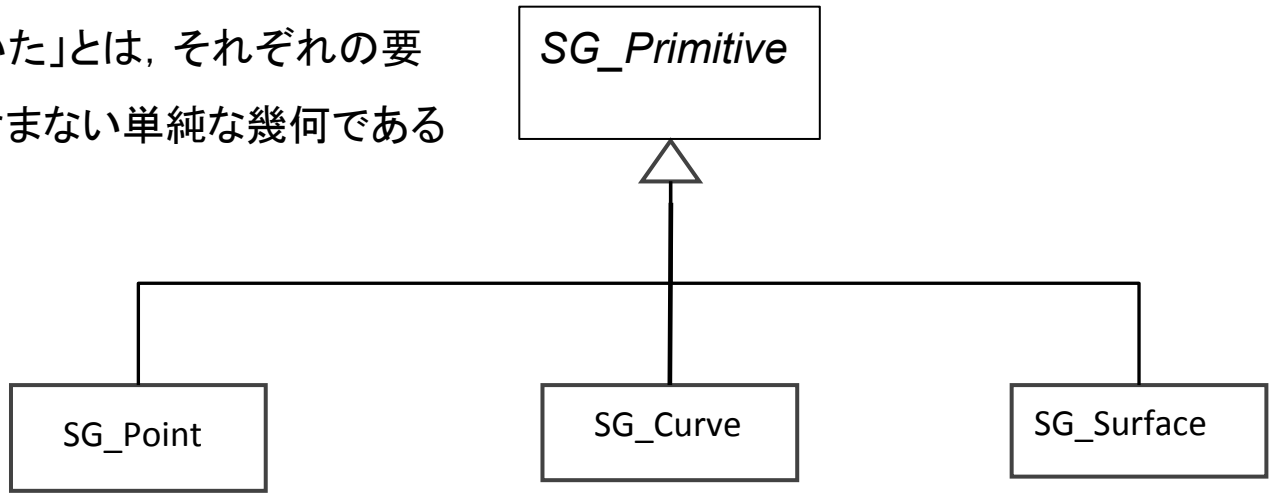
単純な幾何プリミティブ



これらは，単純な幾何プリミティブではない．

SG_Primitive

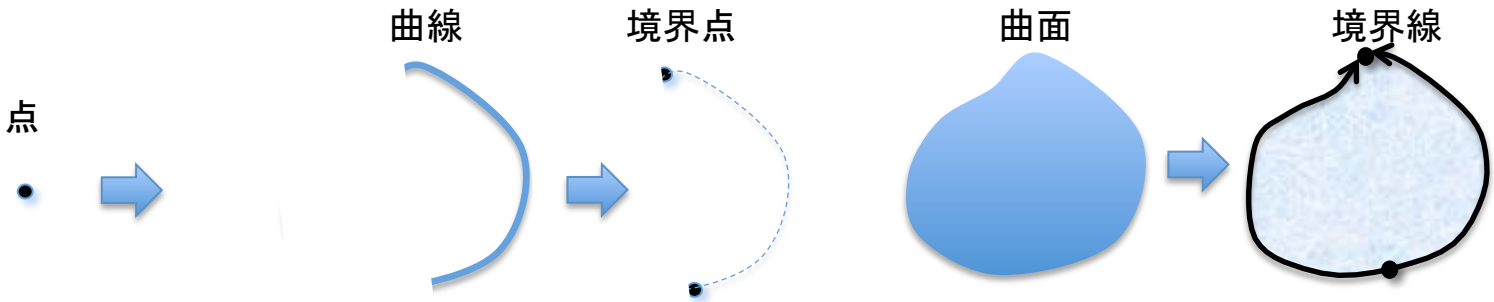
不可分な，開いた幾何形状を示す抽象クラス.その下位に点，曲線，及び曲面がある.「開いた」とは，それぞれの要素が境界を含まない単純な幾何であることを示す.



点に境界はない

曲線の境界は二つの点

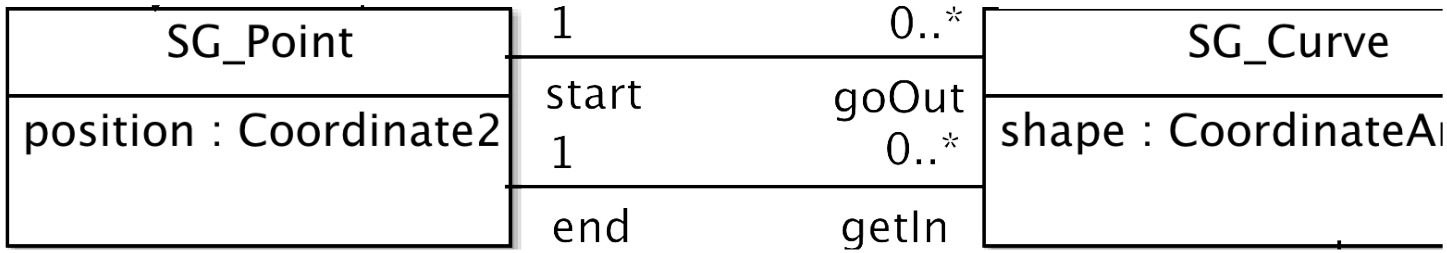
曲面の境界は，閉じた曲線になる，向きをもった曲線の列

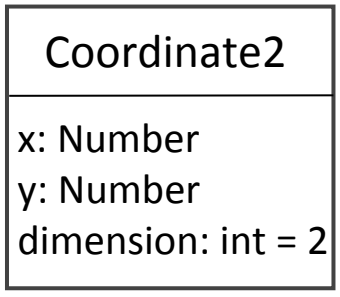


境界は幾何プリミティブが参照する，１次元低い*別の*幾何プリミティブ.

点 (SG_Point)

点は幾何プリミティブで，位置 (position)をもつ. gittok では，位置は2次元座標 (coordinate2) で表現する.
点から曲線が出る (goOut) ことがあり，逆に入る (getIn)こともある. その数は0以上である.

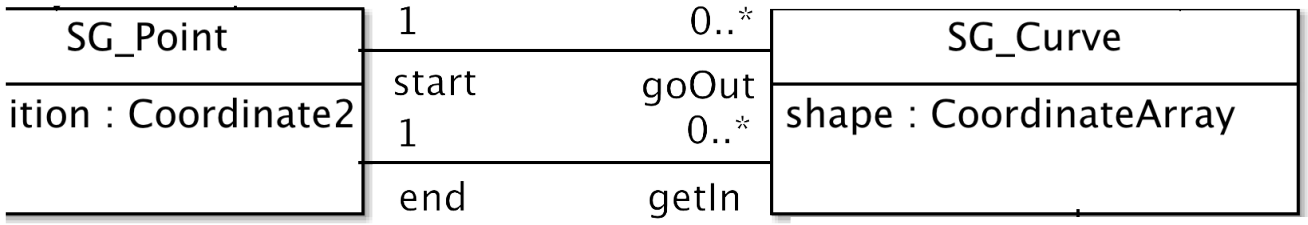


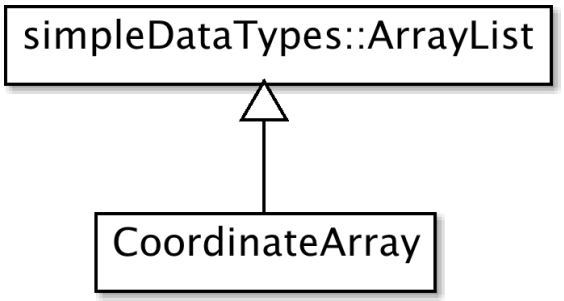


2次元の座標といったって，
長さの単位は？原点は？

座標要素の基準となる座標系は別に定義される.　(09 参照系）をみること).

曲線 (SG_Curve)

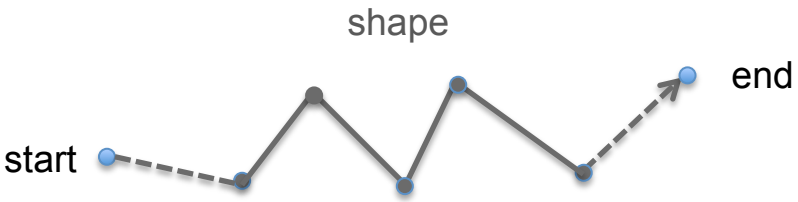




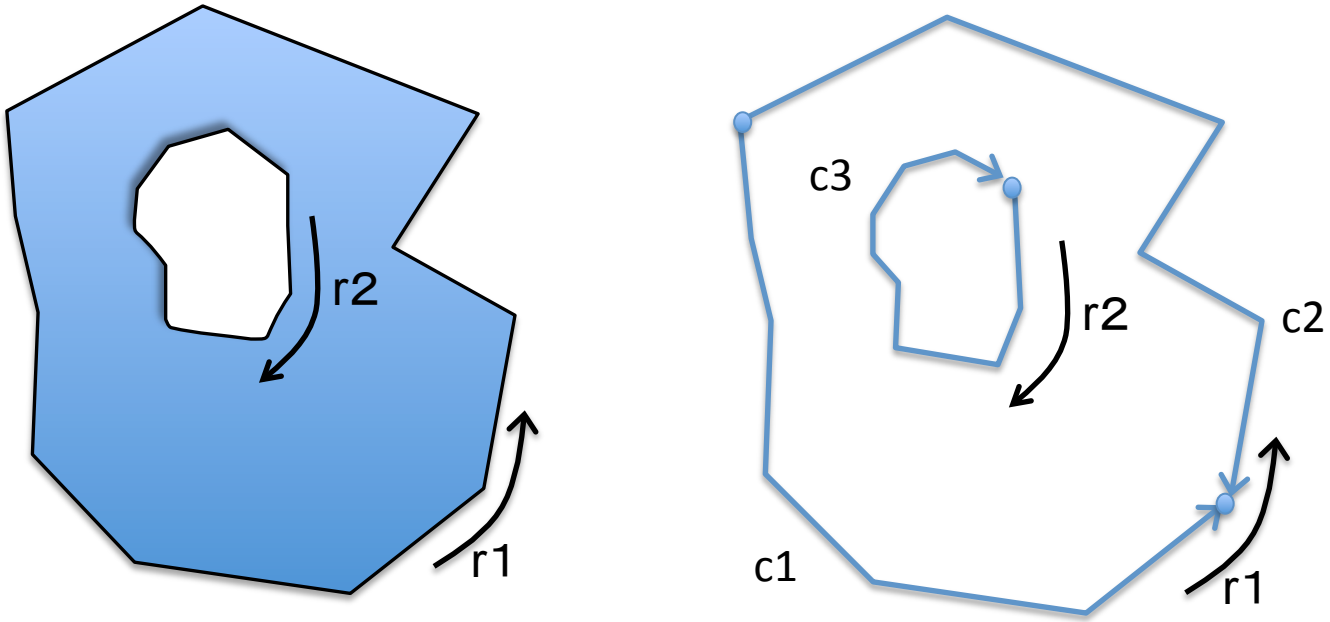
曲線は形状 (shape) 及び，始点 (start) と終点 (end) への関連をもつ．属性“shape” の型は座標の配列リスト（06 単純データ型と主題属性、参照）であり，曲線の“内側”を表現する．

曲線って滑らかじゃないの？

gittok では，c^0級の曲線，つまり，折れ線のみを扱う．



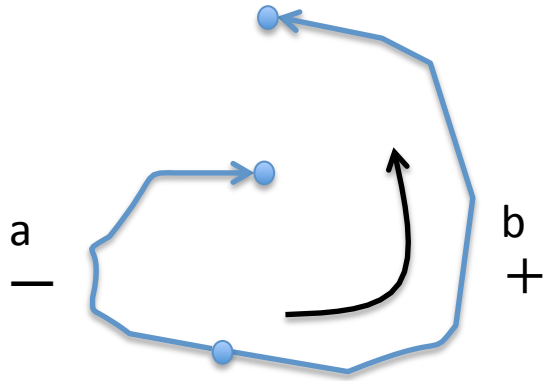
曲面とは？



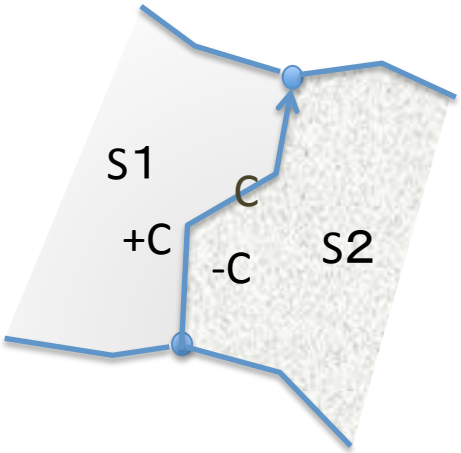
閉じた境界線（輪）の進行方向左側が曲面．境界線は，向きがついた曲線，つまり有向曲線である．曲面は内側の境界をもつことがある．輪は要素となる曲線の列である．曲線がもともと左に面の内側を見るように方向付けられていれば，面にとって，曲線の方向は+．上記の曲面では，外側境界 (r1)の輪は (+c1, –c2) ，内側境界の輪(r2)は(+c3)で示される．なお，現状 gittok では内側の境界は実装できない．

曲面の境界

もともと曲線は，始点から終点への方向をもつ．しかし，曲面の境界を構成する曲線は，異なる方向になる場合がある．そこで，曲線に向きの記号をもたせることで，方向をそろえる．



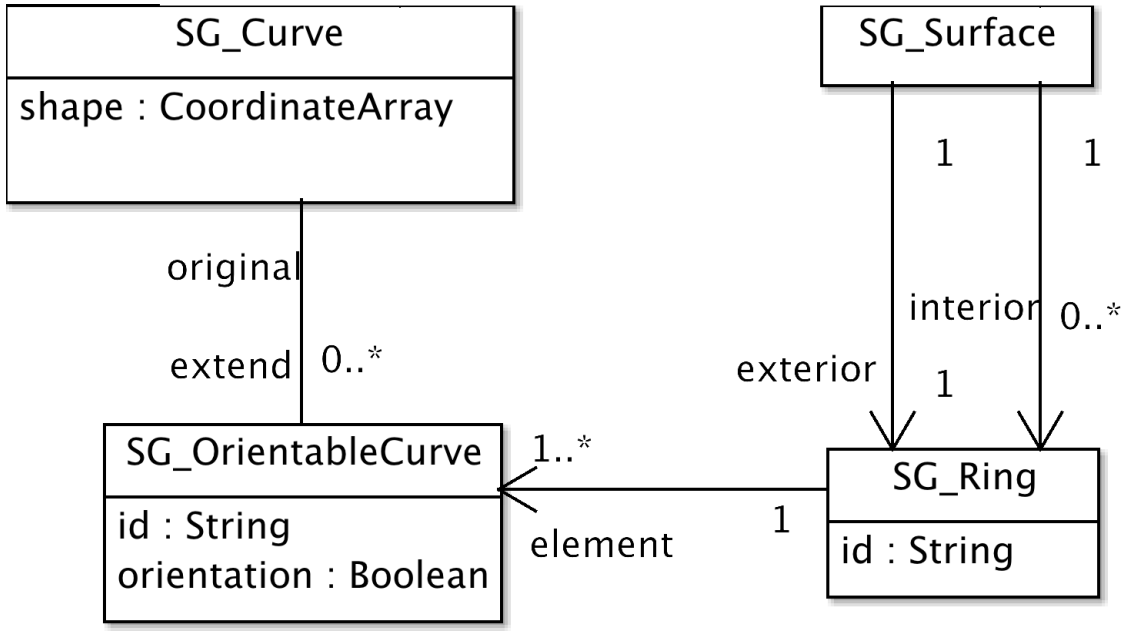
この曲線は，有向曲線-aと+bの列で表現できる．



曲面S1にとっては，Cは正方向，
曲面S2にとっては，反方向

曲面（SG_Surface）

SG_Surface (曲面) は，幾何プリミティブであり，外側境界(exterior)をもち，複数の内側境界 (interior) をもつことがある．
面の境界は輪(SG_Ring)であるが，これは，向き(orientation)をもつ有向曲線 (SG_OrientableCurve)の列（element）で表現される．



曲線自身は，どの曲面の境界か知ってるの？

いまのままでは，分かりません．今後検討します．

幾何複体とは

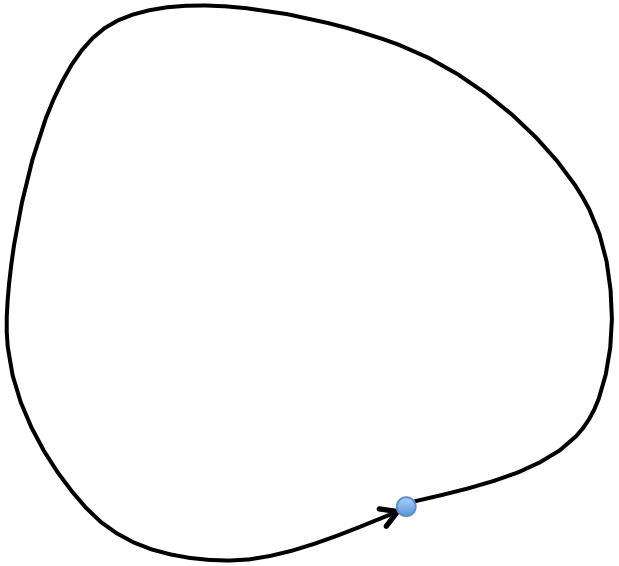
互いに接続する幾何プリミティブの集りを，幾何複体(complex)という．ただし，幾何複体の要素になるプリミティブ同士は以下の条件を満たさなければいけない．

1. 要素となるプリミティブ同士は共通部分をもたない．
2. 幾何複体は境界となるプリミティブを含む．

例えば，複数の筆で構成される街区の空間属性は，幾何複体である．また，右図に見られるように，複数の土地利用区域を含む領域は，幾何複体で表現できる．

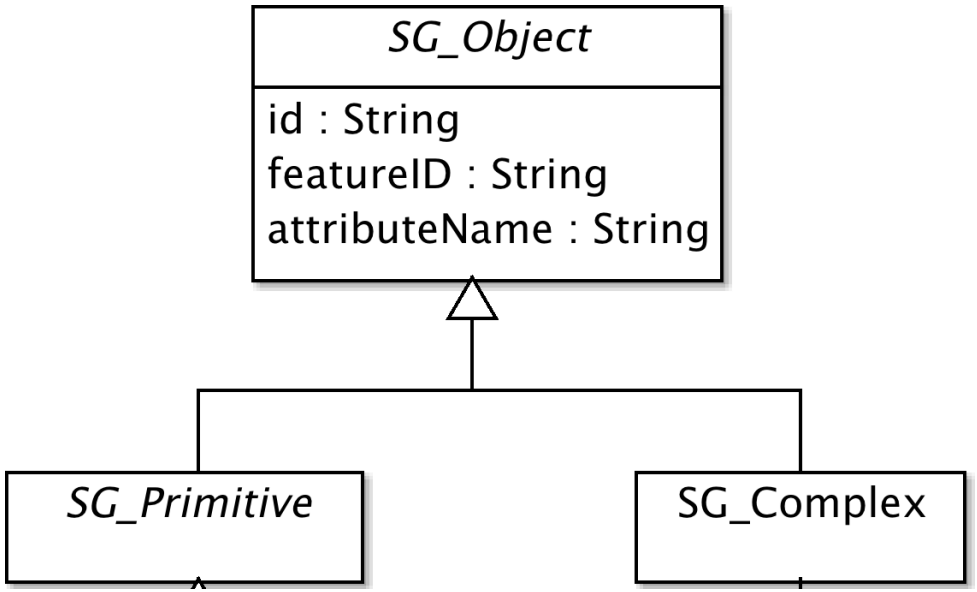
幾何複体（SG_Complex）

SG_Complexは点，曲線及び曲面それぞれの集合からなる，幾何データ全体の部分集合である．最も単純な幾何複体は，端点が一致する曲線とその端点の組み合わせである．



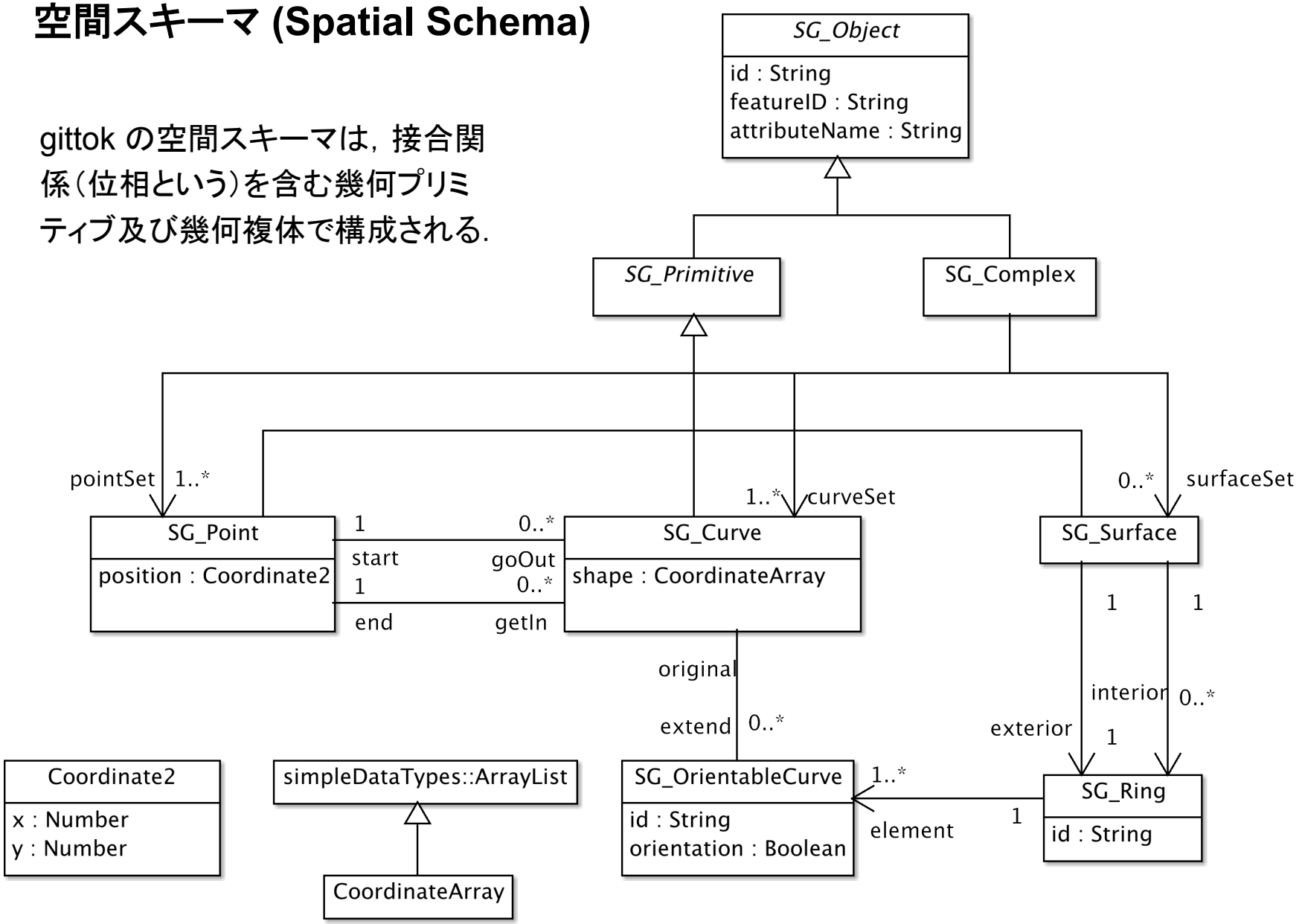
幾何オブジェクト（SG_Object）

幾何オブジェクト (SG_Object) は抽象クラスであり，幾何クラスのルートクラスとして，識別子 (id)，地物インスタンスの識別子 (featureID)，及び属性 (attributeName)の名称を，下位クラスに継承させる.
スクリーン上に表示された地図に示されるのは幾何属性であり，これが地物を代理 (proxy) するので，それがもつ featureID を使って，地物インスタンスを特定することができる.



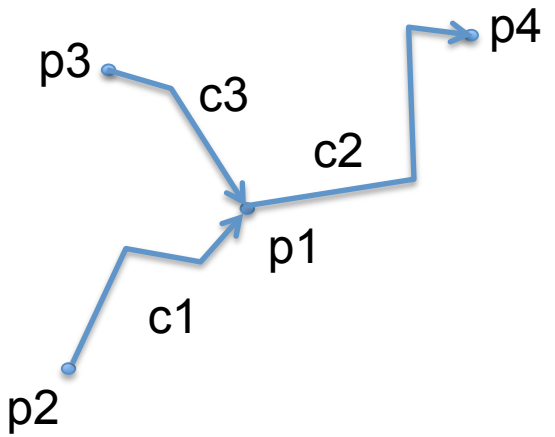
空間スキーマ (Spatial Schema)

gittok の空間スキーマは，接合関係（位相という）を含む幾何プリミティブ及び幾何複体で構成される.



点のXML 表現

SG_PointはSG_Primitive及びその上位のSG_Objectを継承し，位置(position)をもち，入る曲線(getIn)，出る曲線(goOut)への参照をもつ．
もし，この点が地物インスタンスを代理 (proxy) するときは，そのインスタンスのidがfeatureIDの値になる．また，proxyになる属性の名称がattributeNameの値になる．

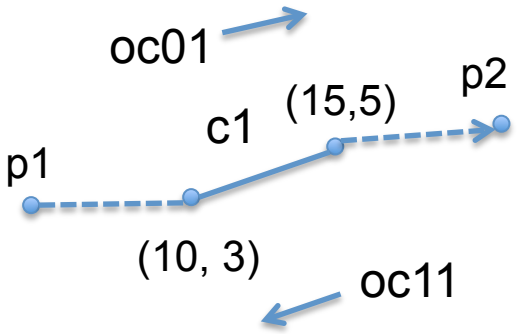



```
<SG_Point>
    <inheritance>
        <SG_Primitive>
            <inheritance>
                <SG_Object id="p1" featureID="" attributeName=""/>
            </inheritance>
        </SG_Primitive>
    </inheritance>
    <position>
        <Coordinate component="35.70,139.73" dimension="2"/>
    </position>
    <getIn idref="c1,c3"/>
    <goOut idref="c2"/>
</SG_Point>
```

XML表現については，（04 XML入門），（05 インスタンスの表現）を参照のこと．

曲線のインスタンスのXML表現

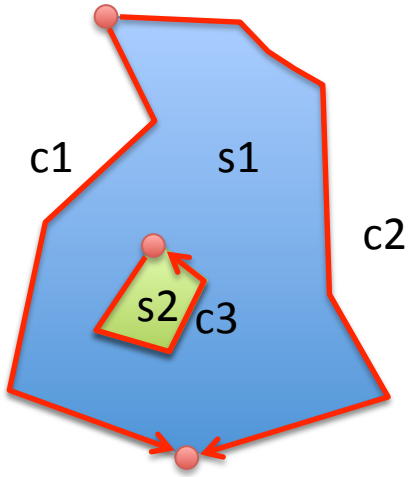
SG_Curveは始点 (start), 終点 (end)
を参照し, 自らの形状 (shape) を座
標列として保持する.




```
<SG_Curve>
    <inheritance>
        <SG_Primitive>
            <inheritance>
                <SG_Object id=“c1" featureID="” atributeName=“” />
            </inheritance>
        <SG_Primitive>
    </inheritance>
    <start idref=“p1"/>
    <end idref=“p2"/>
    <shape>
        <CoordinateArray element=“10,3,15,5" dimension="2"/>
    </shape>
    <extend idref=“oc01, oc11"/>
</SG_Curve>
```

extendの idref 要素数
0: 両側に面がない
1: 左右どちらかに面がある
2: 両側に面がある

曲面のインスタンスのXML表現



SG_Surface
s1=(r1, r3)
s2=(r2)

SG_Ring
r1=(oc01, oc12)
r2=(oc03)
r3=(oc13)

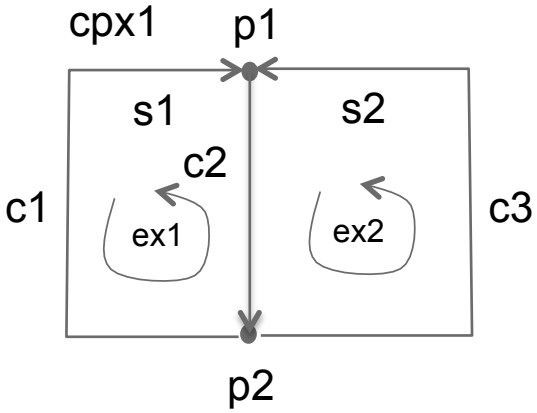
SG_OrientableCurve
oc01=+c1
oc12=-c2
oc03=+c3
oc13=-c3

```
<SG_Surface>
 <inheritance>
        .....
 </inheritance>
 <exterior idref=”r1" />
 <interior idref=“r3” />
</Surface>
<SG_Surface>
 <inheritance>
        .....
 </inheritance>
 <exterior idref=”r2" />
</SG_Surface>
```

```
<SG_Ring id=“r1”>
   <element idref=”oc01, oc12" />
</SG_Ring>
<SG_Ring id=“r2”>
   <element idref=”oc03" />
</SG_Ring>
<SG_Ring id=”r3”>
   <element idref=”oc13" />
</SG_Ring>

<SG_OrientableCurve id=”oc01" orientation="+”>
   <original idref=“c1”>
</SG_OrientableCurve>
<SG_OrientableCurve id=”oc12" orientation="-”>
   <original idref=“c2”>
</SG_OrientableCurve>
<SG_OrientableCurve id=”oc03" orientation="+”>
   <original idref=“c3”>
</SG_OrientableCurve>
<SG_OrientableCurve id=”oc13" orientation=“-”>
   <original idref=“c3”>
</SG_OrientableCurve>
```

SG_Complexの実装



幾何プリミティブは境界を含まないが，幾何複体はそれ自体が境界を含む“閉じた”幾何である．
例えば，上記の複体の場合，境界の輪は，(-c1, +c3)であるが，これらの有向曲線は幾何複体 cpx1 に含まれる．

幾何複体（右上グレーの部分）は，幾何プリミティブなどへの参照で構成される．

幾何複体 cpx1は，以下のように示すことができる．

```
cpx1 = (pointSet, curveSet, surfaceSet,
                orientableCurveSet, ringSet)

pointSet = (p1, p2)
curveSet = (c1, c2, c3)
surfaceSet = (s1, s2)
orientableCurveSet = (oc11, oc02, oc12, oc03)
ringSet = (ex1, ex2)
```

s1 = (ex1), s2 = (ex2)

ex1 = (oc11, oc12), ex2 = (oc02, oc03)

oc11 = −c1, oc02 = +c2, oc12 = −c2, oc03 = +c3

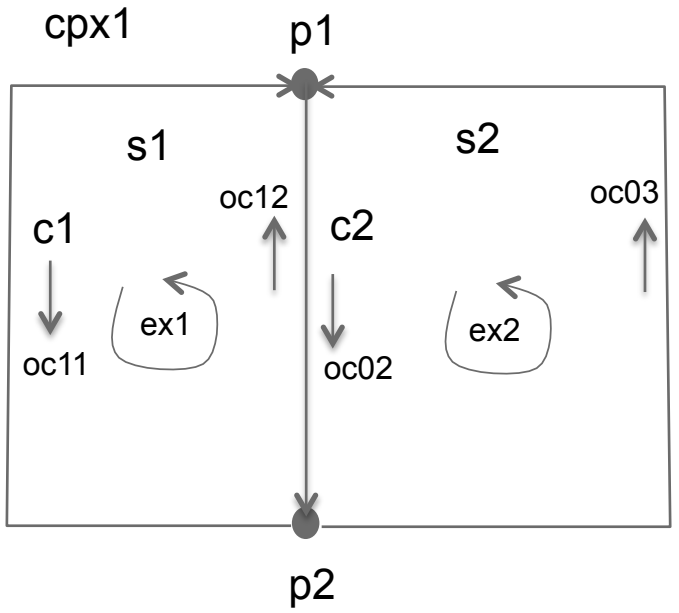
c1 = (p2, p1, shape1)
c2 = (p1, p2, shape2)
c3 = (p2, p1, shape3)

p1 = (getIn1, goOut1, position1)
p2 = (getIn2, goOut2, position2)

getIn1 = (c1, c3), goOut1 = (c2)
getIn2 = (c2), goOut2 = (c1, c3)

＊shapeおよびpositionの記述は省略

幾何複体のインスタンスのXML表現




```
<SG_Complex>
        <inheritance>
                <SG_Object id=”cpx1" featureID=”…” attributeName=“…”>
        </inheritance>
        <pointSet idref="p1,p2"/>
        <curveSet idref="c1,c2,c3"/>
        <surfaceSet idref="s1,s2"/>
        <orientableCurveSet = "oc11,oc02,oc12,oc03"/>
        <ringSet="ex1,ex2"/>
</SG_Complex>
```