ALGEBRAIC SPECIFICATION EXAMPLES, CUBE

```
GenericCube = {

      ;the structure of the SPACE embodying cubeness

      USE[ UnitVector ] = { i j k }
      V = { 0 - 1 }
      D = { 0 1 }
      < vi_V, vj_V, vk_V > = [vi, vj, vk] * [i, j, k]T
      < di_D, dj_D, dk_D > = [di, dj, dk] * [i, j, k]T

      origin = < 0, 0, 0 >
      center = < .5, .5, .5 >

      ;the PARTS of a cube, the DOMAIN of elementary elements

      PART = { < vi_, vj_, vk_> }
      VIRTEX = { < di_, dj_, dk_ > }
      EDGE = { <di_, dj_, ->  <di_, -, dk_>  <-, dj_, dk_> }
      FACE = { <di_, -, ->  <-, dj_, ->  <-, -, dk_> }
      SELF = { <-, -, -> }

      ;the operator which yields properties of the cube

      (p1_PART ^* p2_PART) = < ^*[p1.i p2.i] ^*[p1.j p2.j] ^*[p1.k p2.k] >

      ^*[ 0 0 ] = 0
      ^*[ 0 - ] = 0
      ^*[ 0 1 ] = -
      ^*[ 1 - ] = 1
      ^*[ 1 1 ] = 1
      ^*[ - - ] = -

      ;properties

      parallel[ p1_PART p2_PART ] = {

            p1_EDGE ^* p2_EDGE = _FACE
            p1_EDGE ^* p2_EDGE = _SOLID
            p1_EDGE ^* p2_FACE = _EDGE
            p1_EDGE ^* p2_FACE = _FACE
            p1_FACE ^* p2_FACE = _SOLID
            }
```

```
      perpendicular[ p1_PART p2_PART] = {

            p1_EDGE ^* p2_EDGE = _VIRTEX
            p1_EDGE ^* p2_FACE = _VIRTEX
            p1_FACE ^* p2_FACE = _EDGE
            }


      skew[ p1_PART p2_PART ] =

            p1_EDGE ^* p2_EDGE = _EDGE


      on[ p1_PART, p2_PART ] = {

            p1_VIRTEX ^* p2_VIRTEX = _VIRTEX
            p1_VIRTEX ^* p2_EDGE = _VIRTEX
            p1_VIRTEX ^* p2_FACE = _VIRTEX
            }


      connectedby[ p1_PART, p2_PART ] = {

            p1_VIRTEX ^* p2_VIRTEX = _EDGE
            p1_VIRTEX ^* p2_VIRTEX = _FACE
            p1_VIRTEX ^* p2_VIRTEX = _SOLID
            p1_VIRTEX ^* p2_EDGE = _EDGE
            p1_VIRTEX ^* p2_EDGE = _FACE
            p1_VIRTEX ^* p2_FACE = _FACE
              }

GenericBlock = {

      USE[ GenericCube ] = { cube }
      BLOCK = { [a1_ARITH, a2_ARITH, a3_ARITH] * [cube.i, cube.j, cube.k]T }
      IsCube[ b_BLOCK ] = ( b.a1 = b.a2 = b.a3 )
      }
```

```
CubesInaCube = {

      USE[ GenericBlock ] = { world b1 ... }
      worldscale = [1000, 1000, 1000]
      bigworld = worldscale * world
      location[ b_ ] = < bi, bj, bk >
      InWorld[b_] =
        (( <0,0,0> <= location[ b ] >= worldscale * <1,1,1> ) = true )
      location[b1] = <0,0,0>
      location[b2] = <1,0,0>
      }


StackOfBlocks = {

      USE[ GenericBlock ] = { world b1 ... }
      STACK = { [[ b1_ ... ]] }
      CONFIGURATION = { [[ b1___ ]]__ }

      emptytable = [[ ]]
      [[ ]] [[ ]] = [[ ]]
      location[ emptytable ] = < _, 0, _ >

      PutBlockOnStack[ b1_, s_CONFIGURATION ] =
        ( [[ b1 ]] [[ s ]] = [[ b1, s ]] )

      TakeBlockOffStack[ b1_, s_CONFIGURATION ] =
        ( [[ b1, s ]] = [[ b1 ]] [[ s ]] )

      On[ b1_, b2_ ] =
        ( [[ ___, b1, b2, ___ ]] = true )

      Above[ b1_, b2_ ] =
        ( [[ ___, b1, ___, b2, ___ ]] = true )

      OnTable[ b_ ] = ( [[ ___, b ]] = true )

      OnTopOfStack[ b_ ] = ( [[ b, ___ ]] = true )
```