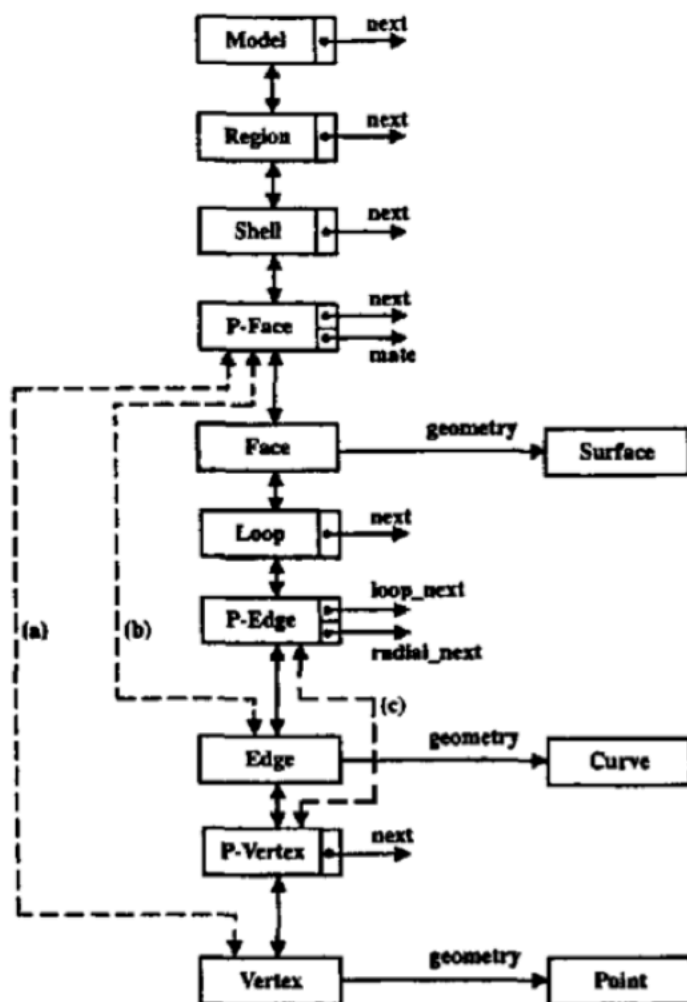


Representation scheme: Partial-Entity data structure

(Sang Hun Lee & Kunwoo Lee, ACM Solid Modeling, 2001)

Compact Non-Manifold Boundary Representation Based on Partial Topological Entities



```
class Entity {
    int    _id;
    Attribute *_attribute;
};

class Model : public Entity {
    Model *_next;      // next model
    Region *_region;   // list of regions
};

class Region : public Entity {
    Region *_next;     // link field of the region list of a model
    Model *_model;     // parent model
    Shell *_shell;     // peripheral shell
};

class Shell : public Entity {
    Shell *_next;      // next void shell
    Region *_region;   // parent region
    Pface *_pface;     // partial face
};

class Pface : public Entity { // partial face (p-face) class
    Pface *_next;      // next p-face
    Shell *_shell;     // parent shell
    Entity *_child;    // child entity: a face, an edge, or a vertex
    Orient _orient;    // orientation flag w.r.t. the face normal
    Pface *_mate;     // mate p-face
};

class Face : public Entity {
    Pface *_pface;    // one of two incident p-faces
    Loop *_loop;      // peripheral loop
    Surface *_geometry; // surface
};
```

```
class Loop : public Entity {
    Loop *_next;      // next hole loop
    Face *_face;      // parent face
    Pedge *_pedge;    // a p-edge in a loop
};

class Pedge : public Entity { // partial edge (p-edge) class
    Loop *_loop;      // parent loop
    Entity *_child;    // child entity: an edge or a p-vertex
    Orient *_orient;   // orientation flag w.r.t. the edge direction
    Pvertex *_pvertex; // start p-vertex
    Pedge *_looped_prev; // previous p-edge in the loop cycle
    Pedge *_looped_next; // next p-edge in the loop cycle
    Pedge *_radial_prev; // previous p-edge in the radial cycle
    Pedge *_radial_next; // next p-edge in the radial cycle
};

class Edge : public Entity {
    Entity *_parent;   // parent entity: a p-edge or a p-face
    Pvertex *_pvertex[2]; // two end p-vertices
    Curve *_geometry;  // curve
};

class Pvertex : public Entity { // partial vertex (p-vertex) class
    Pvertex *_next;    // another p-vertex associated with _vertex
    Entity *_parent;   // parent entity: an edge or a p-edge
    Vertex *_vertex;   // mother vertex
};

class Vertex : public Entity {
    Entity *_parent;   // parent entity: a p-vertex or a p-face
    Point *_geometry;  // position
};
```