ZWSOFT

平台基础知识

-- 模型历史管理简介

主讲人：黄恺斌

CONTENTS

概念&定义

- 内容提醒
- 边界表示 概念 & 定义
- 拓扑表 概念 & 定义
- 词汇 定义 & 缩写

旧数据结构&组织方式

- ZW3D 平台数据结构
- BREP 数据结构
- TT 数据结构

旧数据流同步&更新

- 拓扑表 入库流程
- 拓扑表 出库流程

C O N T E N T S

新架构介绍

- 数据 & 流程解耦
- 容器设计 & 数据管理
- 关联流程 演示&问答

其他

- 调试技巧

概念&定义

Content reminder

Tips : What you can know from this lesson & what can't.

你可以从这门课得到的

- 对基本概念的了解;
- 对数据结构的直观感受;
- 特征流程中关于B-rep和Tt的数据流向;
- 新旧模型历史数据管理流程的区别

你无法从这门课得到的

- 关于几何拓扑学的课本知识;
- 为什么采用这样的拓扑表示方式;
- 如何编写建模命令 & 有哪些常用的构建拓扑/几何对象的建模函数;
- 对基本数据格式的详细介绍;
- ZW3D怎么启动?
- 钝角

本课程假设你已经

- 了解如何使用ZW3D构建一个模型;
- 了解参数化建模基本概念：特征，重生成，重定义，回滚等;
- 了解执行/重定义/重生成历史树时的基本流程;
- 了解ZW3D存在由C编写的专用内存数据结构（如VsListObj ）;

PS :

如果课程讲述期间存在不了解的内容或者无法理解主讲人的表述，欢迎随时打断并提问，主讲人将根据问题的实际情况进行

- *当场解释;*
- *推荐 阅读Zwiki *** 页面 进行了解;*
- *推荐 咨询 *** 同事进行了解;*
- ~~*觉得你很萌并给你点赞;（小概率事件）*~~

边界表示 概念 & 定义

Tips： Mainly introduce the concept of B-rep and its definition.

三维模型表达

几何模型、参数化特征模型

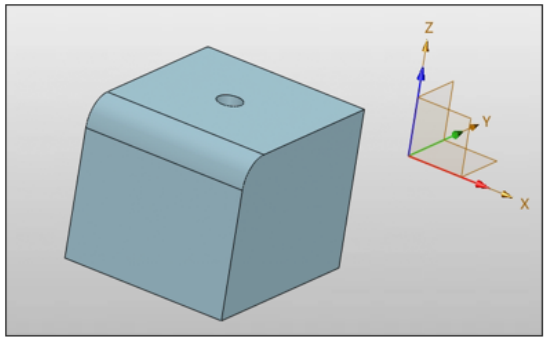
几何模型

边界表示法（Boundary representation，简称B-rep/Brep）

根据点，线，面这些几何元素构成的表面来对三维模型实体的**几何信息**进行描述，
用顶点，边，面，有向边，环，壳，体等拓扑元素对模型的**拓扑信息**进行描述。

建模结果

- 包含信息少，没有设计意图及其他生产需要的信息
- 无法使用这个表示方式进行设计和建模



- 为避免重复进行课程内容设计，此处直接使用已经教学完毕的课程课件内容进行展示。
- 若存在对这部分内容的疑问，请移步《**参数化历史建模和重生成概要**》课程PPT进行学习。

拓扑表 概念 & 定义

Tips : Mainly introduce the concept of topology/topologic table and its definition.

拓扑：

描述拓扑元素之间的**连接关系**，这种连接关系不依赖于特定的几何描述。比如说拓扑不会因为你使用隐式或者显式表达而不同。欧拉示性数描述一种拓扑元素间关系。ZW3D的拓扑是基于B-rep表达。

—— 《ZW3D拓扑基础》

拓扑表：

已一定的规则和方式储存一系列拓扑元素/几何元素的容器。在ZW3D中作为建模命令操作的对象集，建模命令对拓扑表进行运算操作，形成新的模型拓扑内容。

拓扑元素：

- 实体（VsShape）
- 壳（VsShell）
- 环（VsLoop）
- 裁剪边（VsPreEdge）
- 面（VsFace）
- 边（VsEdge）
- 顶点（VsVertex）

几何元素：

- 曲面（VsNurbSurf）
- 曲线（VsNurbCurv）
- 点（VsPoint）

词汇 定义 & 缩写

Tips： Introduce the abbreviation of high frequency terms.

词汇名	缩写
拓扑表	Tt, toptable
边界表示	Brep
实体	Shp
壳体	Shl
拓扑面	Fc, Face
拓扑边	Edge, Edg
拓扑环	lp, loop
裁剪边	Pedge
曲面	Nsurf, Nsrf, srf
曲线	Ncrv, Ncurv, crv
拓扑点	Pt, point
重生成	Regen
快速回滚	HQR
特征	Ftr

旧数据结构&组织方式

ZW3D 平台数据结构

Tips： Introduce the data structure of ZW3D DB/OM

文件对象

Struct VsObjBin：

基本功能：按Block分块，并序列化出一系列对象块用于存放具体的VsObject。

Struct VsObjDoc：

业务概念，包含多个bin，其中trgbin为实际文件内容储存的bin，其他bin在运行时临时使用。

基础数据对象

Struct VsObject

关键成员：

```
ZS_OM_OBJDATA void *data;
```

指向具体的派生数据对象。

以Fc为例：

在数据库中，一个Face的储存方式为：

```
VsObject -> data -> VsCdFace
```

派生数据对象

Struct VsCd (***)

VsCdBrep 【直接拥有】的 拓扑/几何 对象

包含以下几种：

- VsCdShape
- VsCdShell
- VsCdFace
- VsCdEdge
- VsCdPedge
- VsCdLoop
- VsCdNcurv
- VsCdNsurf

Tips：若希望了解更多关于DB/OM的知识，请参加数据引擎组相关课程，本课程将不在此赘述。

BREP 数据结构

Tips： Introduce the data organization structure of B-rep

数据对象结构

```
/* version 2 brep */
typedef struct
{
    int idx_shapes;   /* head of list of shape objects */
    int idx_shells;   /* head of list of shell objects */
    int idx_loops;    /* head of list of loop objects */
    int idx_edges;    /* head of list of edge objects */
    int idx_vertices; /* head of list of vertex objects */
    int idx_geom;     /* head of list of geometry objects */
    int last_ftr;     /* last active feature index */
    int last_shell;   /* last shell position code */
    int id_shell;     /* counter for generating shell pick id's */
    int id_face;      /* counter for generating face pick id's */
    int id_edge;      /* counter for generating edge pick id's */
    int idx_par;      /* index of parent object */
    int i1;           /* spare integer */
    int i2;           /* spare integer */

} VsCdBrep;
```

储存位置

```
typedef struct VsCdPart
{
…… 【ignore】
int idx_brep;              /* b-rep sub-object (VBREP) */
…… 【ignore】
}
```

TT 数据结构

Tips： Introduce the data organization structure of toptable

数据对象结构

```
typedef struct VsApplTopTable
    {
        /* Topology information. */
        VsListObj *tbl_shape; /* List of VsShape structures. */
        VsListObj *tbl_shell; /* List of VsShell structures. */
        VsListObj *tbl_comp;  /* List of VsCompartment structures. */
        VsListObj *tbl_loop;  /* List of VsLoop structures. */
        VsListObj *tbl_edge;  /* List of VsEdge structures. */
        VsListObj *tbl_vrtx;  /* List of VsVertex structures. */
        /* Geometry information. */
        VsListObj *list_geom; /* List of VsLocGeomDat structures,
local NURB geometric entities */
        /* Database information. */
        VsListObj *list_idxs; /* List of VsDbIdxObj structures,
database indices for geometric
                              and topological entities */
        void *tbl_xref;       /* Cross reference table VsFcXRefTable */
    } VsApplTopTable;
```

储存位置

只储存于内存中。拓扑表为内存运行时对象，可以看作Brep的运行时格式。

建模算法在内存中操作拓扑表，并通过持久化方法将改动更新至模型Brep中。

Brep&Tt

从数据结构上看，Brep与Tt拥有相仿的数据组织方式。

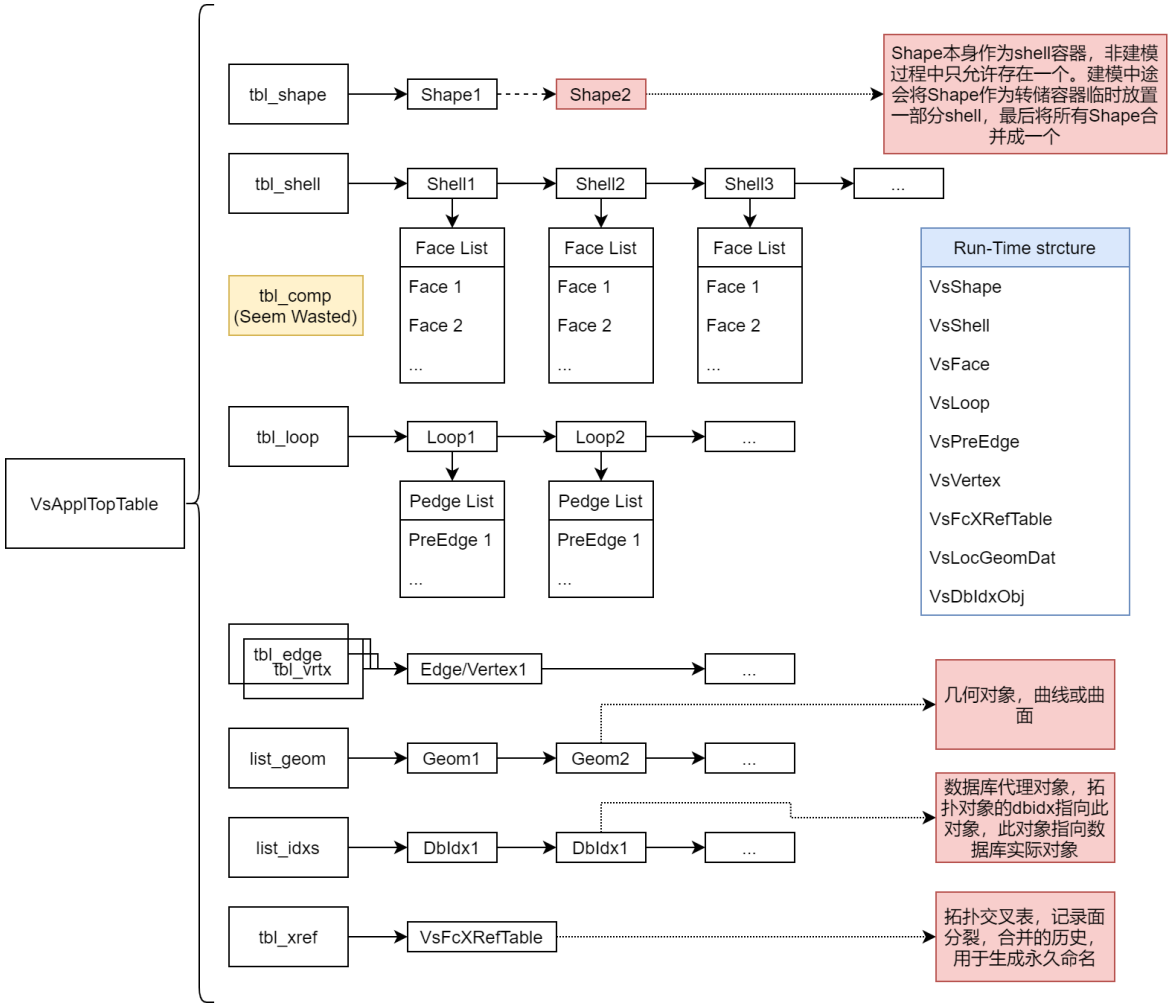
【同理对Brep内拓扑对象也成立，如VsFace <-> VsCdFace】

因为本质上Tt与Brep就是对同一份数据对象的不同表达方式。

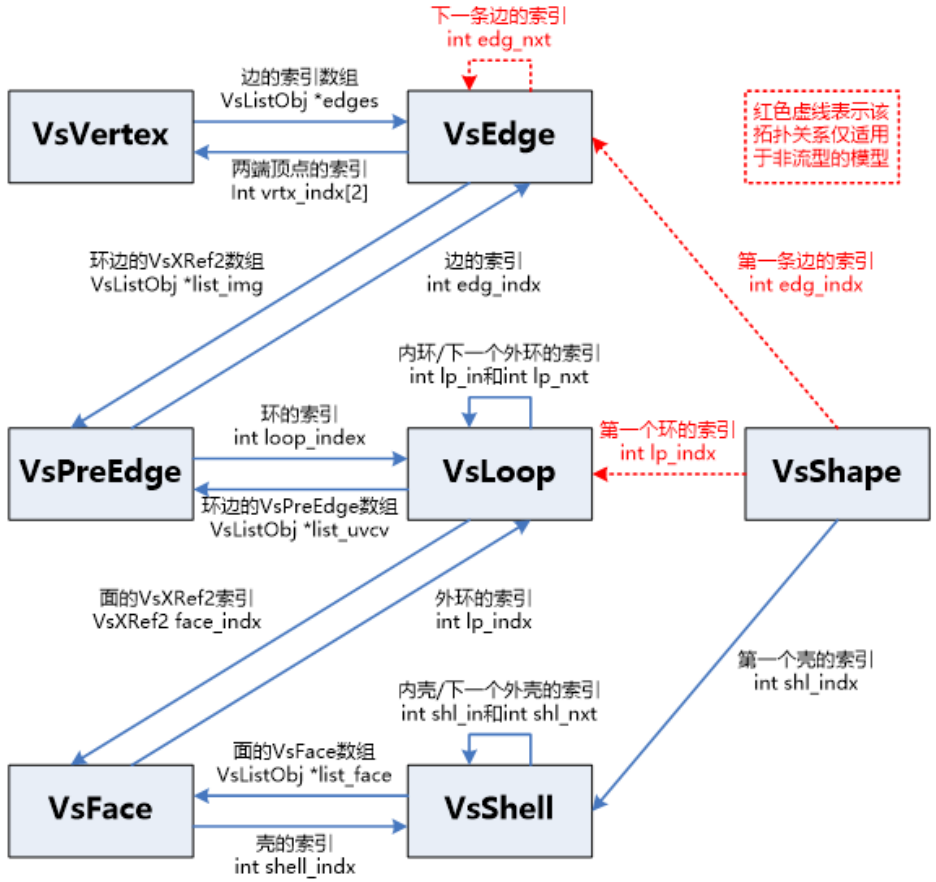
TT 数据结构

Tips : Introduce the data organization structure of toptable

数据对象说明



拓扑结构说明



旧数据流同步&更新

拓扑表 入库流程

Tips: How topology table being dumped into the VsCdBrep object.

定义

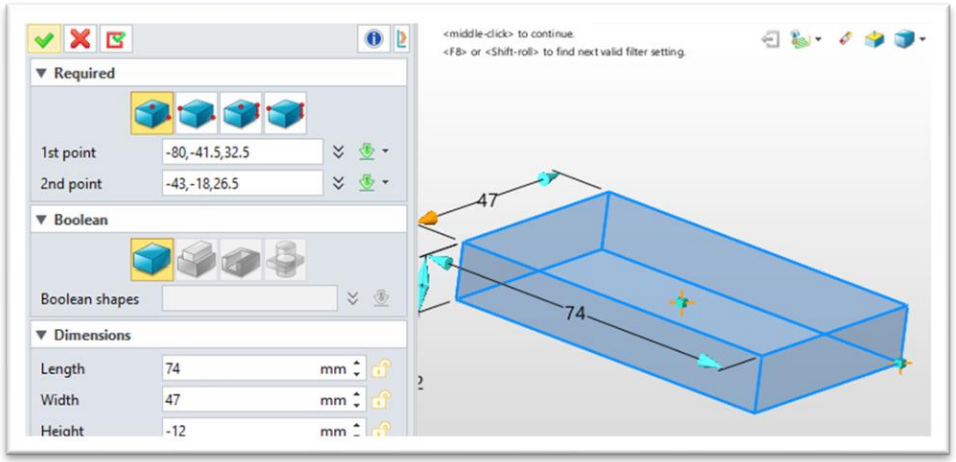
拓扑表入库流程：指的是将被**建模算法**进行**修改**后的**拓扑表**内容同步至**模型Brep对象**内的流程。

建模算法：一系列修改拓扑表的操作的统称。在零件环境下，一般由执行特征命令调用。

修改：对拓扑表进行以下操作均视为广义上的修改：增加对象；删除对象；修改对象内容；调换、调整对象所处数组的位置。

拓扑表：若无特殊说明，均代指类型为VsApplTopTable的全局变量VgTt。

模型Brep对象：若无特殊说明，均代指零件对象VsCdPart下的idx_brep对象。

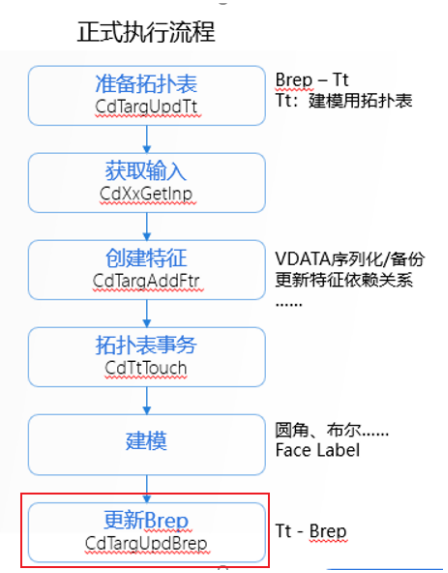


总体概述

函数入口：CdTargUpdBrep

主体函数：CdTtToBrep

入库流程位于特征执行流程的尾部，在用户点击建模特征并选择"√"执行后，通过模板命令调用一系列的建模操作修改拓扑表，最后执行入库流程。流程图如右图所示。



拓扑表 入库流程

Tips: How topology table being dumped into the VsCdBrep object.

—预处理环节与后处理环节基本与入库的主要流程无关，本课程不对此部分进行展开，详细内容可阅读代码或查看《拓扑表入库》文章内容。

流程总览

- 预处理环节
- 入库环节
- 后处理环节

预处理环节

- Label相关
- 显示相关
- 全局变量相关
- 静态设置
- 拓扑表检查

后处理环节

- 显示相关
- 全局变量相关
- 静态设置

入库环节

入库流程可以大致分为五个部分：预处理死亡（DELETED）拓扑对象，将拓扑表新增内容同步至DB，删除死亡拓扑对象，将拓扑对象数据库对象状态同步至brep，后处理。分别对应以下几个函数：

- TtToBinStart
- TtToBin
- TtDelObj
- TtListToBrep

下面对这4步进行展开描述。（《模型历史管理课程》省略此部分）

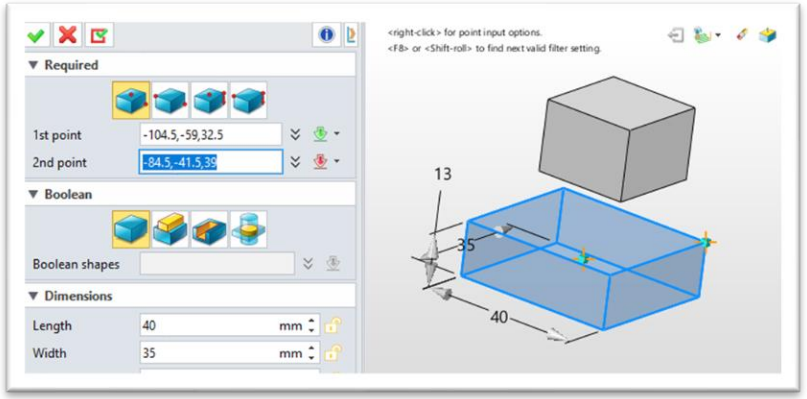
PS：后处理部分都是一些其他附加流程或者功能按需添加的处理，本课程不对此部分进行展开，这部分可以更加关注实际代码。

拓扑表 出库流程

Tips: How topology table being update from the VsCdBrep object.

定义

拓扑表出库流程：指的是将Brep表示的几何数据加载到拓扑表中，基本所有建模算法都需要在拓扑表上执行。



总体概述

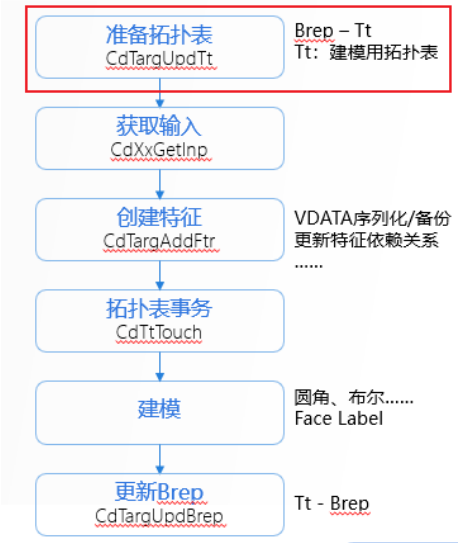
函数入口：CdTargUpdTt

主体函数：CdTtFromBrep

入库流程位于特征执行流程的开头，在用户点击建模特征并选择“√”执行后，流程会先检查当前拓扑表是否和文件Brep为完全映射（或者说是否过时&无效），若过时，会执行拓扑表出库流程，根据文件现存Brep重建一份拓扑表。

PS：一般我们称重建后的拓扑表为“干净拓扑表”，而建模入库完毕后的拓扑表为“脏拓扑表”。因为拓扑表在出库后并不会删除数组元素，而会保存，故建模后不进行删除重建流程的拓扑表会存在两种状态的对象【Unmodified & Deleted】。

正式执行流程



拓扑表 出库流程

Tips: How topology table being update from the VsCdBrep object.

流程总览

相较于入库流程，出库流程仅需要考虑如何按结构填充拓扑表，同时查询是否存在拓扑表对象映射较为容易（查看是否有元素对应db_idx一致），映射成功时，对象字段也为完全覆盖，只是会存在不同的状态Flag修改，在流程上较为简单。主要包含以下几步：

- **初始化数据**

CdBrepInitTT根据VsCdBrep中shape，shell，loop，edge和vertice的数量用VxTopTabResiz计算出拓扑表大小并初始化。其中VmTopTabInit初始化拓扑表的链表。初始化VsTtInput的list_ent链表。

- **把brep数据加载到拓扑表**

对VsCdBrep的成员调用各自的**V_TT方法**，在拓扑表中创建db对象对应的tt对象。

- **设置拓扑对象间的连接关系**

对V_TT创建出的tt对象（inp.list_ent）调用TtEntMap，把第二步生成的tt对象关联起来。

PS： db对象中记录着对象之间的关系，比如一条边所关联的两个点，这一步相当于把db的这个关联也在tt对象中创建出来。具体参考TtEntMap的实现，处理了VERTEX，EDGE，PEDGE，LOOP，FACE，SHELL和SHAPE对象。

- **后处理**

包含时间记录，孔面信息更新，错误处理等

新架构介绍

数据&流程解耦

Tips： Introduce the goat of new architecture & basic concept.

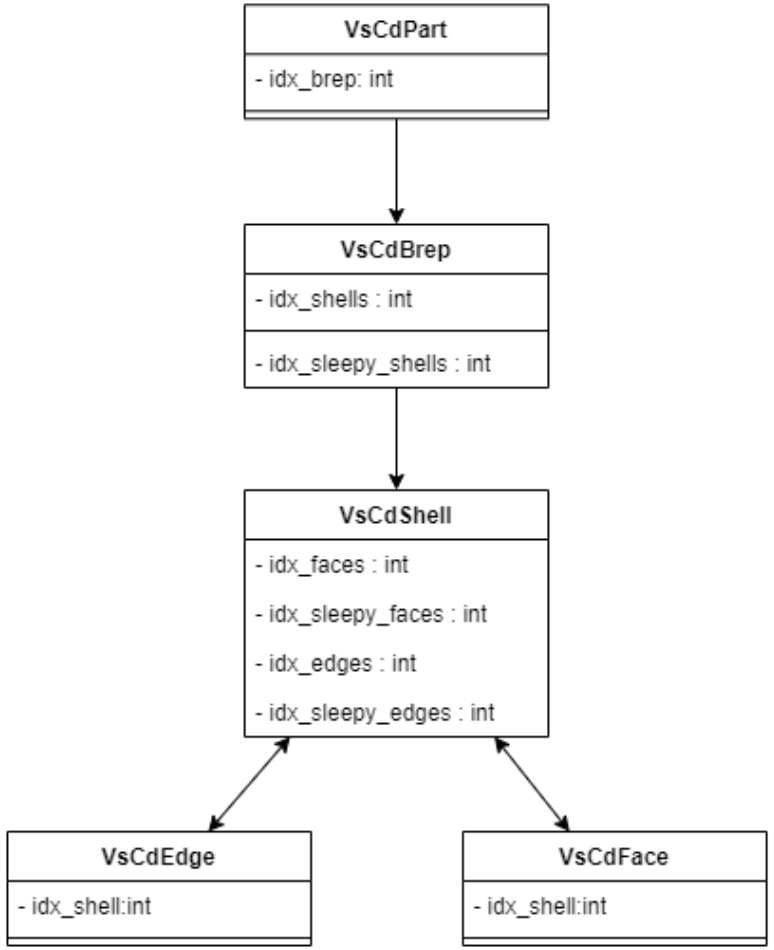
数据对象解耦

新架构下主要思路和方向在于将各个模块的数据封装进自身模块之中，以达成**模块独立**和**数据流隔离**的效果。

经过划分过后，BREP对象被分成了两块：

- 几何拓扑数据：由独立内核管理其数据；
- 应用功能数据：由上层对象管理其数据。（原VsCdBrep及其子对象）

简化的Brep结构



容器设计&数据管理

Tips ： Introduce the data organization structure of B-rep

Partition

设计意图：让内核具有历史数据管理能力，帮助解决多实体效率问题，重生成问题，快速回滚和静态问题。

特点：

- 以body为单位对内核对象数据进行管理，大部分情况下是一个Partition对应一个body；
- 具有事务能力，用于支撑内核的历史数据管理；
- 一个Partition对应一个拓扑表；
- 不同的Partition不能共享拓扑几何数据；

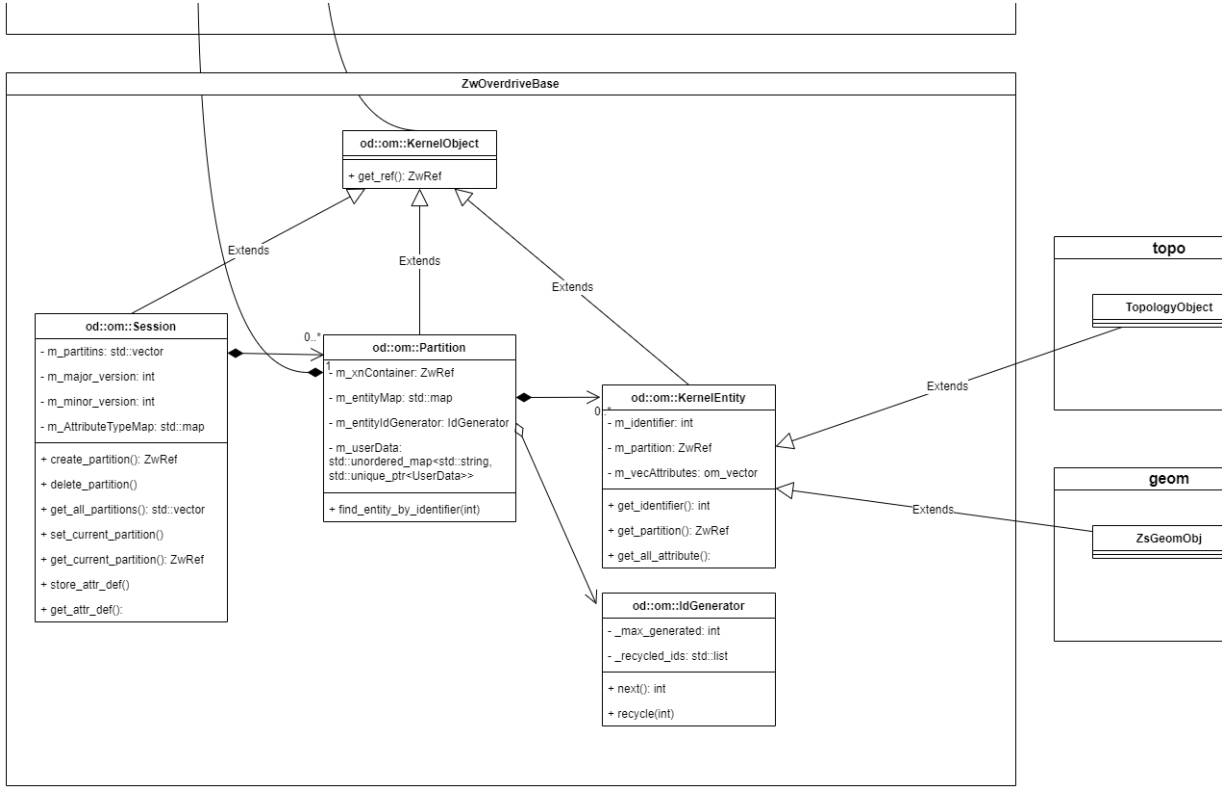
Pmark

设计意图：记录Partition某个状态下的数据，属于增量记录，undo/redo或者选择性重生成等操作时，需要借助PMark实现Partition数据的跳转，不需要重新进行建模运算。

特点：

- 发生回滚和前滚时，Partition的状态只能去到某一个PMark上。
- 每个Partition有一个Initial PMark(初始状态)和 Current PMark (当前状态)。
- PMark是树状结构。 （参考为什么pmark是树状结构）

UML



容器设计&数据管理

Tips： Introduce the data organization structure of B-rep

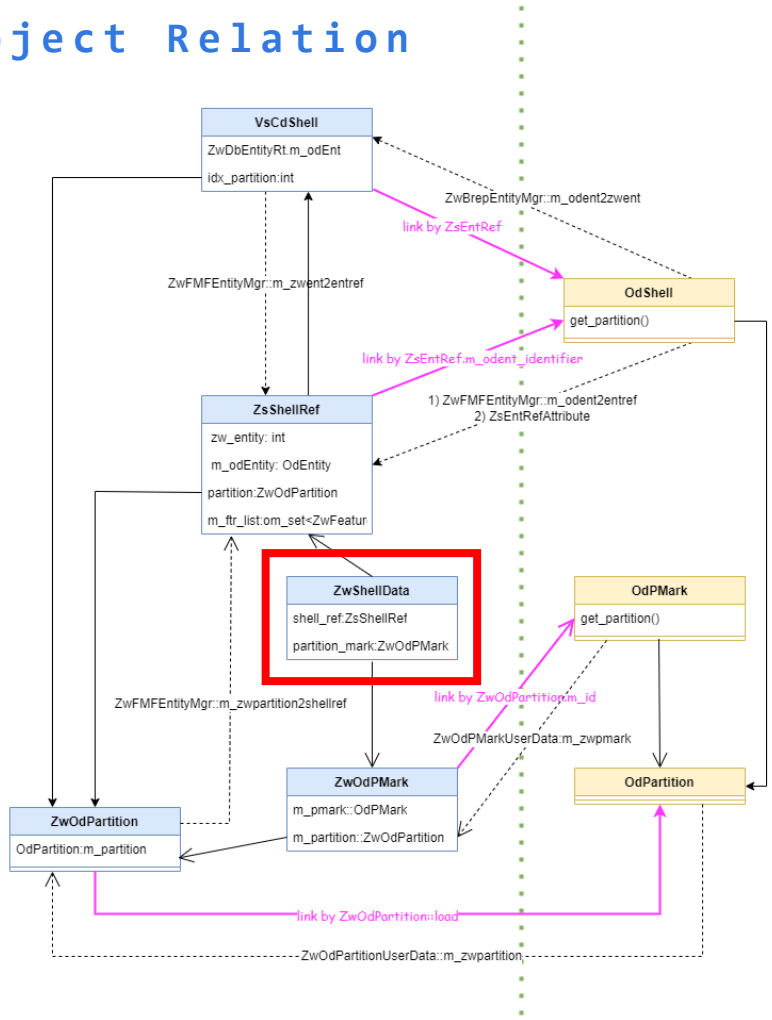
ZwShellData

设计意图：内核提供了Partition以及Pmark结构，应用层需要记录某个实体对应的Partition的节点状态。Shell Data就是用于记录某个体映射的Partition的状态变化的类。

特点：

- 通过shell data将shellref和odpmark关联起来，一个特征修改了几个shell，就会产生几个shell data；
- 会使用VeShellDataBehavior标记自身的状态（创建，修改，死亡）；

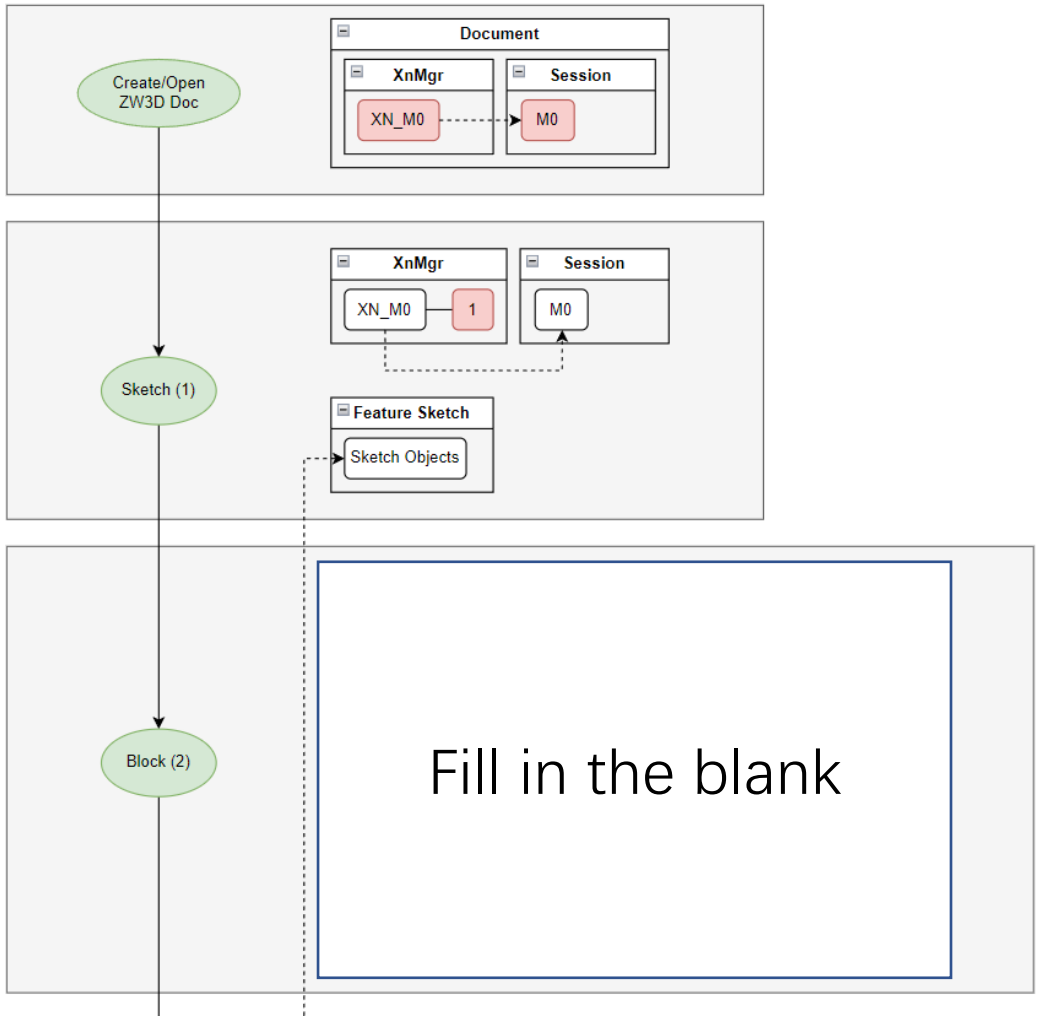
Object Relation



关联功能演示：创建特征

Tips ： Introduce the whole procedure of feature execution.

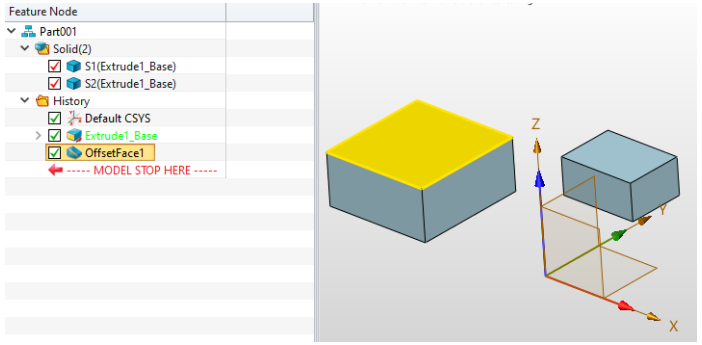
Short Quiz(1)

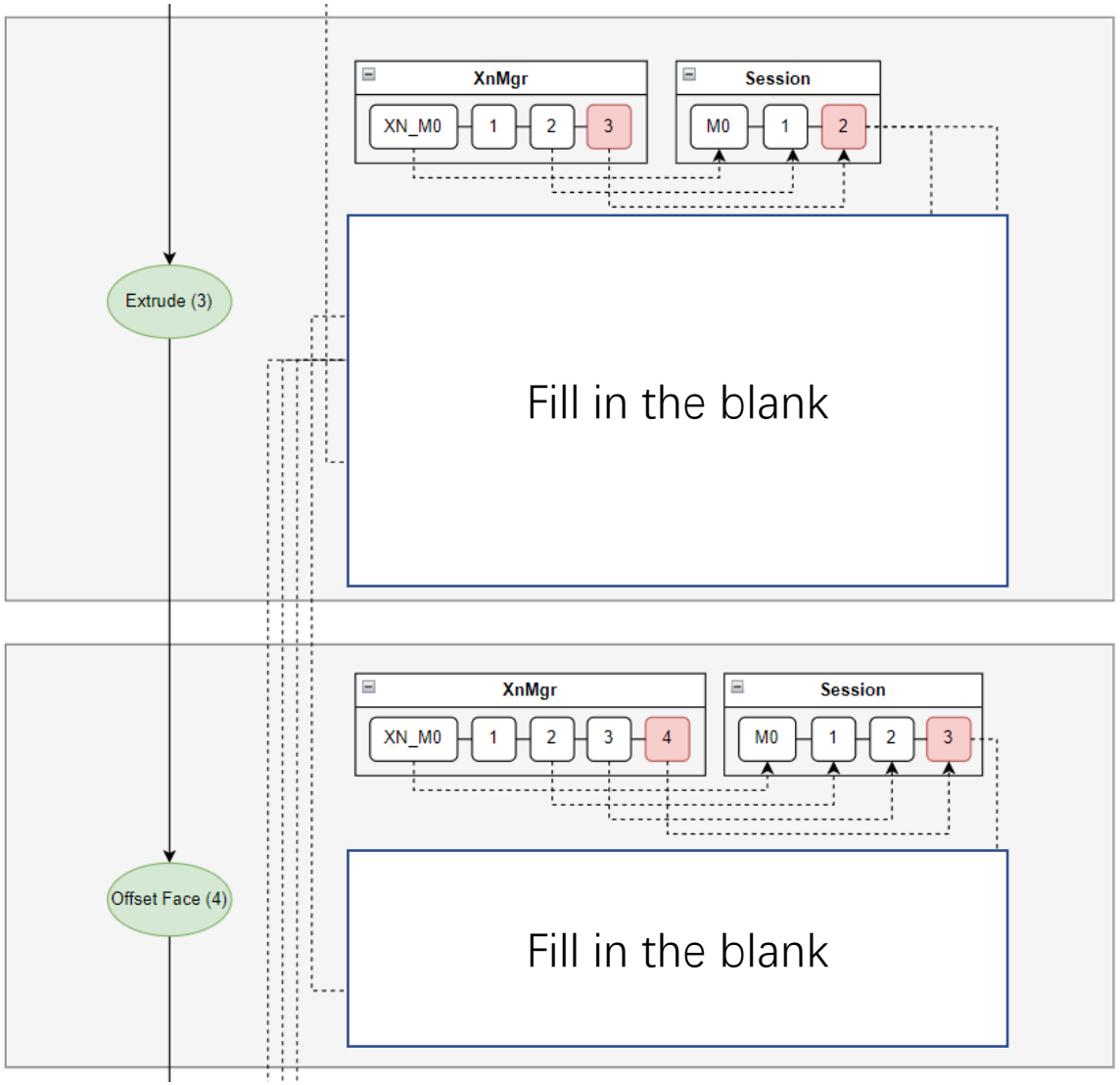


关联功能演示：修饰特征

Tips ： Introduce the whole procedure of feature execution.

Short Quiz(2)

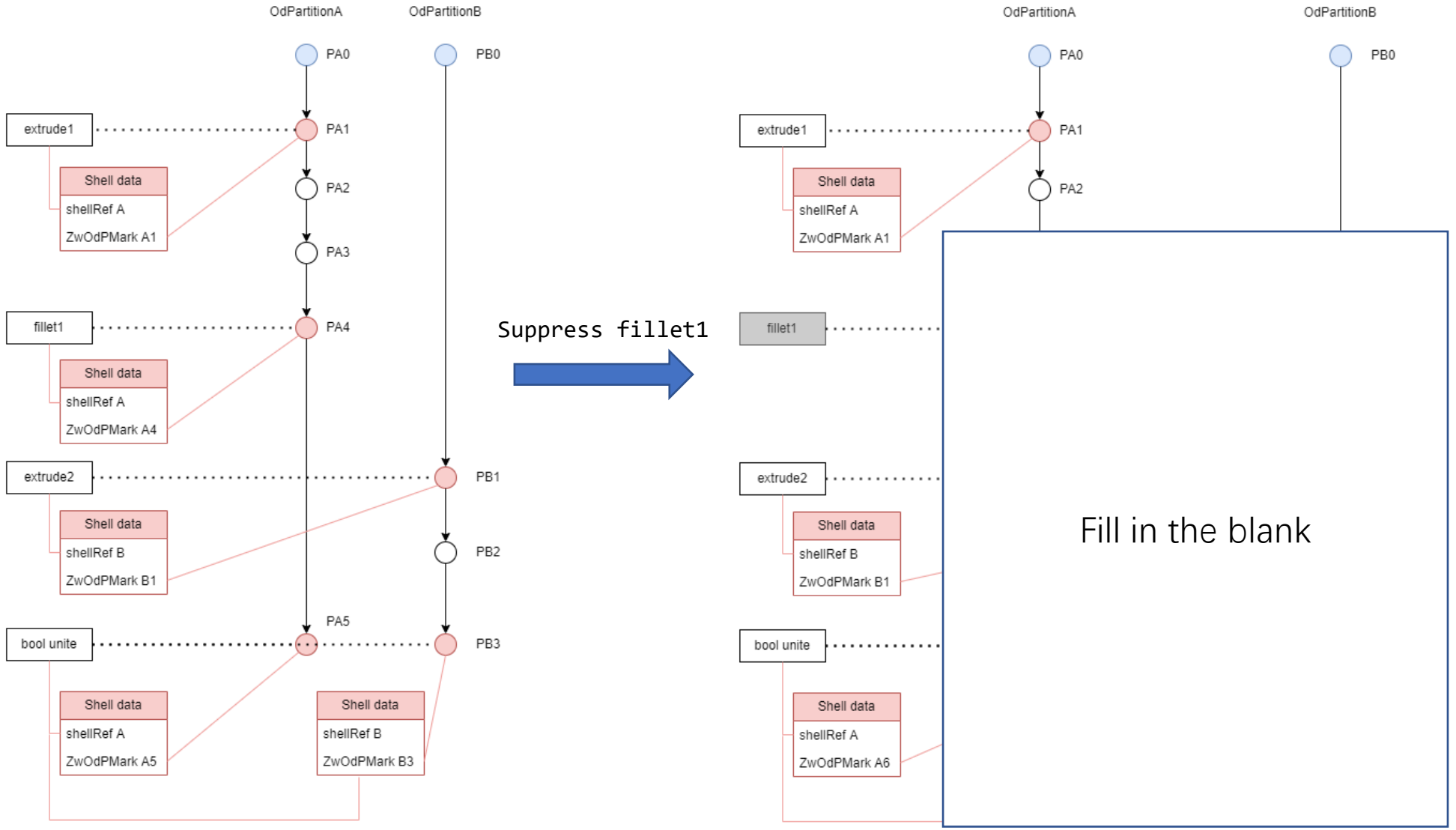




关联功能演示：抑制

Tips ： Introduce the whole procedure of feature execution.

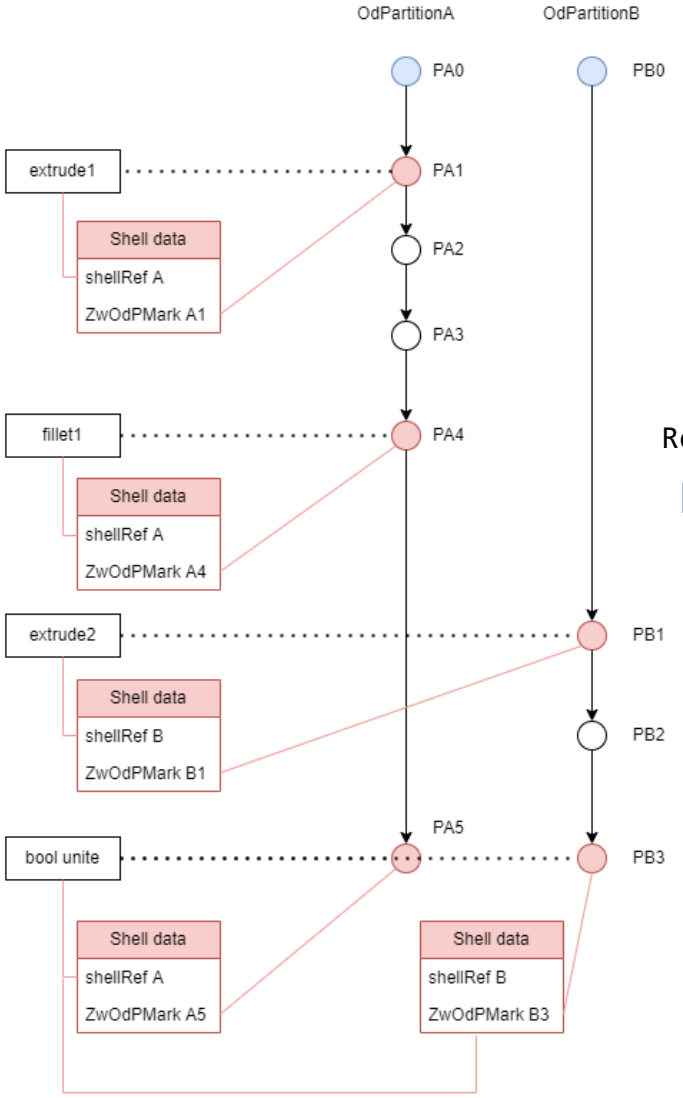
Short Quiz(3)



关联功能演示：重定义

Tips ： Introduce the whole procedure of feature execution.

Course Quiz(4)



Redefine fillet1

?

其他

调试技巧

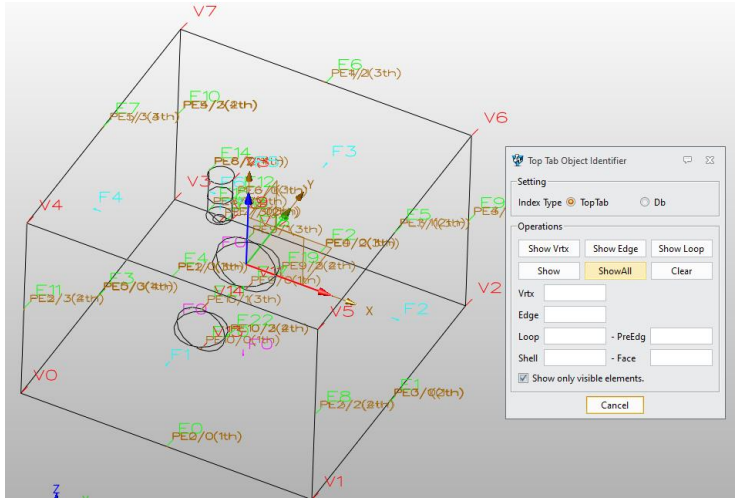
Tips: Some basic rules of topo object & debugging tips.

调试技巧

Hint：不涉及程序调试的相关技巧，如有此方面需求，请查看Zwiki文档《 ZW3D调试技巧介绍.docx 》

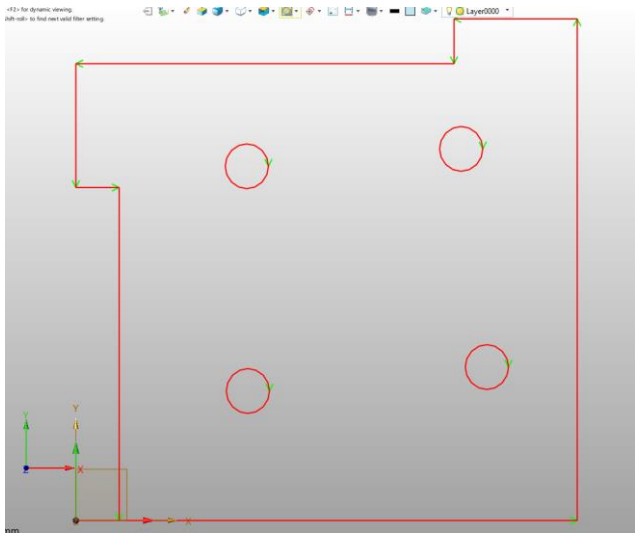
如何查看拓扑信息：

- ❑ 使用Debug→CheckTt功能
- ❑ 点击“Show Ids”即可开开始查看
- ❑ 期望查看所有Edge的编号以及在模型区中对应的内容，可点击Show Edge。也可以针对想查看的已知编号的拓扑对象进行查看。下图是点击“Show All”后绘图区的表现。



如何查看面的环：

- ❑ 开启多ZW3D实例调试
- ❑ 点击Utility中的Start Another ZW3D，启动另一个ZW3D软件实例
- ❑ 新的ZW3D中新建一个Z3Prt文件，如Part001.Z3PRT
- ❑ 在新的ZW3D中也启动“Remote Graphic Debugging”
- ❑ 回到原来的ZW3D窗口中，点击Debug→Util →Disp Face Uvs选项并选择一个面
- ❑ 切换至另一个实例，选择TopView.



详细流程 & 图例请参考《拓扑表结构》。

参考文献

Tips: reference.

[1] “Partition&Brep management” https://zwiki.zwcax.com/x/cRowBw

[2] “history transaction” https://zwiki.zwcax.com/x/HQu4Bw

[3] “Kernel objects” https://zwiki.zwcax.com/x/jRowBw

[4] “Partition Design” https://zwiki.zwcax.com/x/x604C

[5] “为什么pmark是树状结构” https://zwiki.zwcax.com/x/Gao4C

[6] “《拓扑表入库》” https://zwiki.zwcax.com/x/NAFeBQ

[7] “《拓扑表出库》” https://zwiki.zwcax.com/x/LAFeBQ

[8] “《拓扑表结构》” https://zwiki.zwcax.com/x/noEwBg

[9] “Parameter-less Brep(Partition)” https://zwiki.zwcax.com/x/Jgu4Bw


感谢聆听
Q & A