# 五、插件模块划分

## 1）多维矢量模块（责任人：王增杰）

① C/S架构，初定使用MongoDB数据库/…（原版插件里有，也有基于吴明光老师数据库引擎的实现）

② 以OGC/geos/DRM模型为基础，结合几何代数特征表达加速（MVTree结构的集成）

③ 需要考虑以上模型是否支持三维，能否实现我们之前所有的功能

④ 需要实现/集成的功能包括：

-场景集成（建模与可视化）

-场景空间关系动态计算（基于GA算子的空间关系计算：相交关系、拓扑关系）

重点：（动态特征，约束（拓扑，空间关系等）的嵌入）

## 2）栅格与场模块（责任人：张正方、刘袁）

① 文件系统架构，单独的管理控件

-由于场数据可能包含多个维度，并存在数据量大等问题，需要具有基本的文件内存映射功能（即不直接读取文件，只读取数据的变量、维度信息，需要可视化或者分析时再动态读入数据）- 已有系统原型（刘袁，见第6部分插件系统架构实验）

-支持层次张量方式的数据管理，已有算法（健健）但底层张量库需要替换(详见张量库开发方案： E:\workspace\CAUSTA\_3\_0\其它待集成的源码\健健系统\张量库开发方案－V1.00823.docx)

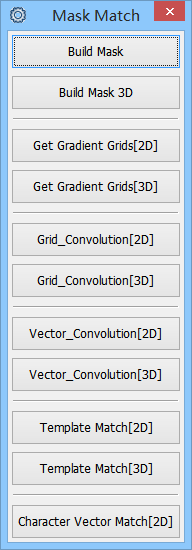
-支持常见的栅格文件（asc,nc,bin,img等）

② 需要实现/集成的功能包括

-基于张量的特征联动可视化（张量分解和可视化已有算法，但界面交互未实现）

-系统模板与特征模板算法（已有算法，待优化与集成）

-多层多时相场数据管理与分析（南极海地冰，基本的分析和可视化已有算法，需要对数据建模，设计交互界面）



**Mask Match** 

――Build Mask [Cbuild\_mask] 构建二维模板

――Build Mask 3D [Cbuild\_mask\_3D] 构建三维模板

――Gradient Grids[2D] [CgetGradientGrids] 二维梯度

――Get Gradient Grids[3D] [CgetGradientGrids3D] 三维梯度

――Convolution[2D] [CGridConvolution2D] 二维栅格卷积

――Grid\_Convolution[3D] [CGridConvolution3D] 三维栅格卷积

――Vector Convolution[2D] [CVectorConvolution2D] 二维矢量卷积

――Vector\_Convolution[3D] [CVectorConvolution3D] 三维矢量卷积

――Template Match[2D] [CTemplateMatch2D] 二维模板匹配

――Template Match[3D] [CTemplateMatch3D] 三维模板匹配

――Character Vector Match[2D] [CCharVectorConvolution2D] 特征模板匹配

## 3）网络模块（责任人：傅蓉）

① 文件系统架构，单独的管理控件（treectl）

-层次网络管理（朱帅有原型，但相关算法还需要集成）

-MCOP算法（有原型算法，需要集成优化）

② 需要实现/集成的功能包括

集成：

Network**--**Compute

――Data process 【CDataProcess】

――Max Flow 【Cmax\_flow】

――Shortest Path[Module] 【CModelShortestPath】

――Node Constraint 【CNodeConstraint\_Path】

――Shortest Path 【CShortestPath】

## 4) 时空框架（责任人：车骁宇）

要求：基于SRM框架，体现绝对、相对、局部和整体

## 5）应急疏散功能模块（责任人：张丽莹）

已实现2D的系统，3D的demo

实现：

MCOP、应急疏散 相关分析插件集成

重点：多重约束、动态网络（权重、拓扑，及由此引起的层次结构的更新）、多重权重约束

## 6）点云模块（责任人：王赟）

① 文件系统架构，球树索引（袁帅点云代码及模型，算法需优化与集成）

② 需要实现/集成的功能包括

-点云结构提取，点云简化

-点云匹配、影像匹配（几何形状算子）

## 7）PIR传感器相关分析模块（责任人：周春烨）

① C/S架构

② 提供场景和传感器可视化，传感器数据查询，数据叠加显示功能

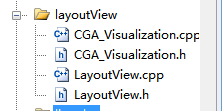
③ 实现目标数目识别，语义提取，特征轨迹提取（参数可控，并能够可视化到图形界面上）

④ 实现场景模拟功能（利用人体行为模型生成人体轨迹，进一步求得传感器响应结果，利用结果求得人体行为语义，将其同预设定的行为模型对比）

（5）传感器数据接入（并行接入、更新）

## 8）数据管理与可视化（责任人：罗文）

① 可视化基于

健健

GDI中添加VTK相关可视化

可参照 CCGA\_Visualization 中的VTK显示功能

系统中的触发方式为：

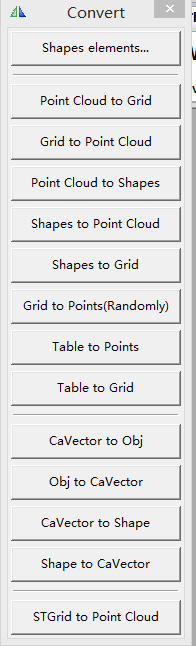
1> Window菜单→Show layout Window 弹出Layout窗口；

2> Window → Show ToolBars → Layout ToolBar。点击添加页面工具 ，在弹出按钮中选择 Add 3D Viewer 。 在Layout窗口中弹出新的三维可视化页面；

3> 在三维可视化页面中 右健点右键可以弹出相关可视化功能

② 数据管理控件（treectrl）,数据转换，数据的基本操作……（需要集成其它控件的数据相关的代码）

――Shapes element…

――――Points to Line[CLines\_From\_Points] CPoints\_to\_Lines

――――Line to points[CPoints\_From\_Lines] CLine\_to\_points

――――Line to Polygons[CPolygons\_From\_Lines] CLine\_to\_Polygons

――――Polygon to Lines[CLines\_From\_Polygons] CPolygon\_to\_Lines

――――Polygon to points[CPolygon\_To\_Points] CPolygon\_To\_Points

――――――――――――――

――Point Cloud to Grid[CPC\_To\_Grid] CPC\_To\_Grid

――Grid to Point Cloud[CPC\_From\_Grid] CGrid\_To\_PC

――Point Cloud to Shapes[CPC\_To\_Shapes] CPC\_To\_Shapes

――Shapes to Point Cloud[CPC\_From\_Shapes] CShapes\_To\_PC

――Shapes to Grid[CShapes2Grid] CShapes\_To\_Grid

――Grid to Points(Randomly)[CGrid\_To\_Points\_Random] CGrid\_To\_Points\_Random

――Table to Points[CPoints\_From\_Table] CTable\_To\_Points

――Table to Grid[CTable2Grid] CTable\_To\_Grid

――――――――――――――

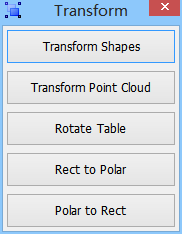
――CaVector to Obj[CCaVector2Obj]

――Obj to CaVector[CObj2CaVector]

――CaVector to Shape[CCaVector2Shape]

――Shape to CaVector[CShapes2Cav]

――STGrid to Point Cloud[Cstgrid2ptc]

**Transform** **--** Organize

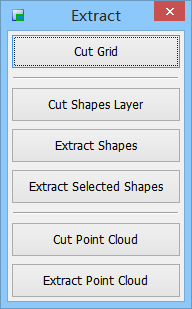
――Transform Shapes[CTransformShapes] CTransform\_Shapes

――Transform Point Cloud[CPC\_Transform] CTransform\_PC

――Rotate Table[CTable\_Rotate] CRotate\_Table

――――Rect to Polar[CCost\_RectToPolar] CCost\_RectToPolar

――――Polar to Rect[CCost\_PolarToRect] CCost\_PolarToRect

**Extract** **--** Organize

――Cut Grid[CGrid\_Cut] CGrid\_Cut

――――――――――――――

――Cut Shapes Layer[CShapes\_Cut\_Interactive] CShapes\_Cut\_Interactive

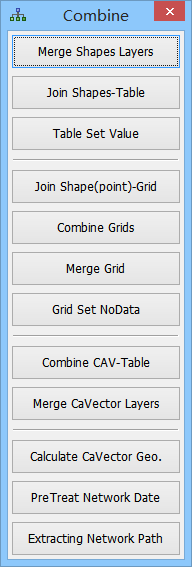
――Extract Shapes[CShapes\_Cut] CShapes\_Cut

――Extract Selected Shapes[CNewLayerFromSelectedShapes] CNewLayerFromSelectedShapes

――――――――――――――

――Cut Point Cloud[CPC\_Cut\_Interactive] CPC\_Cut\_Interactive

――Extract Point Cloud[CPC\_Cut] CPC\_Cut

**Combine** **--** Organize

――Merge Shapes Layers[CShapes\_Merge]

――Join Shapes-Table[CShapes\_Assign\_Table]

――Table Set Value[TableSetValue] --General**--** Organize

――――――――――――――

――Join Shape(point)-Grid[CGrid\_Values\_AddTo\_Points]

――Combine Grids[ CCombineGrids]

――Merge Grid[CGrid\_Merge]

――Grid Set NoData[CgridSetNoData] --General**--** Organize

――――――――――――――

――Combine CAV-Table [CCaVector\_Assign\_Table]

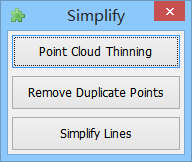
――Merge CaVector Layers [CCaVector\_Merge]

――――――――――――――

――Calculate CaVector Geo. [CCaVector\_Geometry\_Calc]

――PreTreat Network Date [CPreTreatNetworkDate] --General**--** Organize

――Extracting Network Path [Cbuildlinefromtxt] --General**--** Organize

**simplify**  Organize

――Point Cloud Thinning (simple)[ CPC\_Thinning\_Simple]

――Remove Duplicate Points[CRemove\_Duplicates]

——Simplify Lines[CSimplifyShapes] //取消

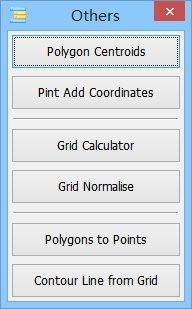
**Space Transform** -- GA Space

――Projective Embedding[Ctransform2affine]

――Stereographic Embedding[transform2stereographic]

――Conformal Embedding[CONFORMAL1]

――Polynomial Embedding[CONFORMAL2]

**Others** **--**Compute -- General

――Polygon Centroids[CPolygon\_Centroids]

――Pint Add Coordinates[CAddCoordinates]

――――――――――――――

――Grid Calculator[CGrid\_Calculator]

――Grid Normalise [CGrid\_Normalise]

――――――――――――――

――Contour Line from Grid[CGrid\_To\_Contour]

――Gradient from Grid[CGrid\_To\_Gradient]

## 9) 模板算法模块（责任人：未来规划）

## 10）编译引擎（责任人：胡老师）

## 11) 并行与高性能计算模块（责任人：所有人均涉及）

## 12）高维智能计算（责任人：未来规划）

（影像分割，模式匹配）

基于生成、滤波、优化的网络结构

## 13）基于OSGearth的DEMO运行环境（责任人：车骁宇）

要求：独立于saga，有开源的代码，熟悉osg场景文件的生成，可对界面做基本的定制与修改，可添加简单的交互

## 14）基于WebGL的DEMO运行环境（责任人：杨力）

要求：独立于saga，主要面向展示，添加简单的交互

SRM：时空框架

EDCS:属性与语义

DRM:数据模型

# 六 可构建的DEMO

基于OSGearth重新实现之前的demo，核心在于earth数据的生成，可设计简单的交互，主要的分析功能还是在插件中完成

## 1）南极

已有基础数据，需要生成earth数据，设计交互

## 2）PIR传感器

如果要表现好，需要能进入场景内部，目前的模型可能太过简单

## 3）应急疏散

可以再找一个范围大一点的场景，也需要有近景的视图，因此需要有精细的模型；另外一个思路是像吴老师那样，场景更复杂，疏散的点更多，但对算法的性能提出考验

## 4）模式模拟

王增杰已实现基础算法，需要加强流程控制和可视化的展示。目前先在saga中写插件，后期也需要把流程与可视化搬到osgearth里去

## 5）网络

大数据的网络，实现基于分层网络模型的多约束求解（由于传统算法的效率已经很高了，需要构建复杂的分析需求），可视化。同样对性能要求很高。

## 6）voronoi算法

不基于osgearth，简单的小例子

# 七、插件系统架构实验

TClimate插件，先构建一个用于管理时空数据的类（继承于CSG\_Module），在这个类的基础上再添加功能：

1）管理类设计

类名称：CCA\_ClimateDataManage

属性：NC数据，STData数组，tensor数组，Tucker数组

方法：数据检查/获取，数据转换

★ 定义一个全局常量CCA\_ClimateDataManage g\_Manage;

具体的管理类设计待续……

2）功能类设计

① NC文件连接类

功能：连接给定NC文件（虚拟指针），并将其指针赋给g\_Manage中的NC数组

输入：NC文件路径

输出：无

② NC文件属性查询类

功能：选择指定NC文件（从g\_Manage获取），列出该NC文件中所有的属性，给出属性的维度，单位等信息

输入：指定NC文件（Add\_Choice）

输出：Table文件

③ NC文件中特定属性数据导出类

功能：选择指定NC文件（从g\_Manage获取），选择该NC文件中特定属性，导出该属性为STData数据，并将其指针赋给g\_Manage中的STData数组

输入：指定NC文件（Add\_Choice），指定属性（Add\_Choice）

输出：无（系统中无显示结果，实际存储在内存中）

功能类待续……

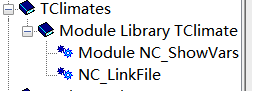
3）在系统GUI中添加ClimateData管理的Tree，并连接插件功能

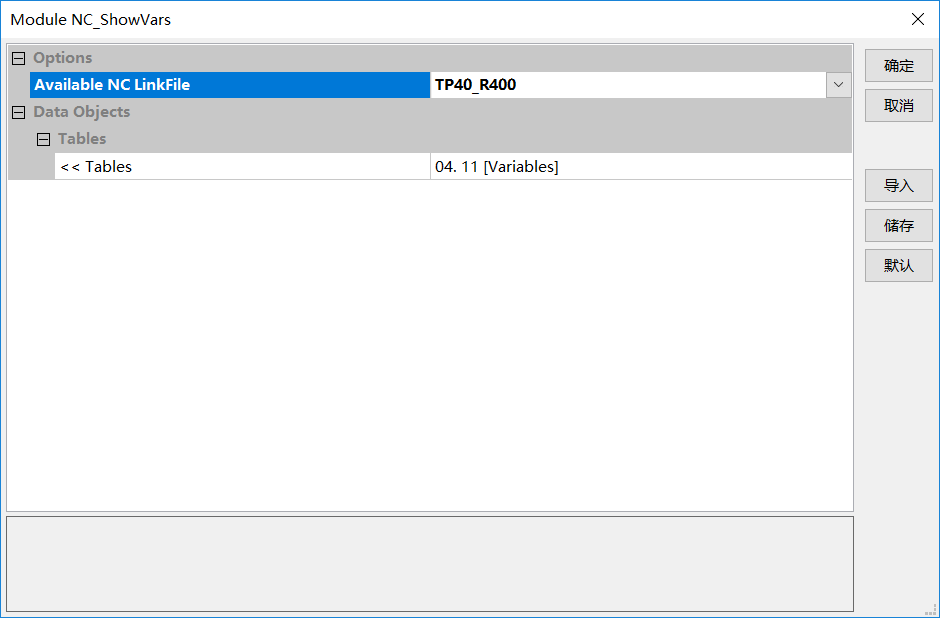
完成：

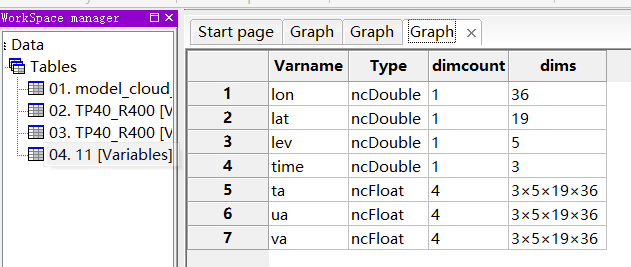
1) 管理类CCA\_ClimateDataManage，目前只实现了NC文件的管理

2）NC文件链接类CNC\_LinkFile，记录NC文件路径和名称，并保存到管理类中

3）NC文件属性列出类CNC\_ShowVars，选定NC文件，列出属性表







3）显示类（需要用到GDI）

健健前期已经实现集成VTK的GDI