iS3 概要设计规约

小 组 成 员

1751062 陈恬恬

1752461 刘 雨

1754067 徐海琪

修订历史

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 编写日期 | SEPG | 版本 | 说明 | 作者 | 评审时  间 | 参与人  员 | 批准日  期 | 确认人  员 |
| 2019-11-30 | iS3 | 1.0 | 初稿 | 徐海琪 |  |  |  |  |
| 2019-12-10 | iS3 | 2.0 | 系统架构图 | 徐海琪 |  |  |  |  |
| 2020-01-05 | iS3 | 3.0 | 定稿 | 徐海琪 |  |  |  |  |

目录

1. [引言 5](#_bookmark0)
   1. [目的 5](#_bookmark1)
   2. [概要设计依据 6](#_bookmark2)
   3. [参考资料 6](#_bookmark3)
   4. [假定和约束 6](#_bookmark4)
   5. [定义 6](#_bookmark5)
2. [概要设计 7](#_bookmark6)
   1. [系统总体架构设计 7](#_bookmark7)
      1. [数据模块 8](#_bookmark8)
      2. [模型层 8](#_bookmark9)
      3. [控制器层 9](#_bookmark10)
      4. [视图层 9](#_bookmark11)
   2. [系统软件结构设计 9](#_bookmark13)
      1. [体系结构图 11](#_bookmark15)
   3. [接口设计 14](#_bookmark16)
      1. [内部接口 14](#_bookmark17)
      2. [外部接口 15](#_bookmark18)
   4. [界面设计 16](#_bookmark20)
      1. 平台[登录界面 16](#_bookmark21)
      2. [服务器端口配置界面 17](#_bookmark22)
      3. [项目选择界面 18](#_bookmark23)
      4. [平台展示界面 20](#_bookmark24)
   5. [数据库设计 41](#_bookmark52)
      1. [实体属性 43](#_bookmark54)
   6. [系统出错处理设置 46](#_bookmark55)
      1. [系统出错处理设计 46](#_bookmark56)
      2. [系统维护设计 48](#_bookmark57)

1 引言

1.1 目的

本文档旨在将软件系统功能进一步细化，将原本复杂的功能拆分成不同的模块，并逐步开发健壮的系统结构，使软件系统需求能够很好的转化为未来设计，并进一步提高软件各方面性能。本文档依据高内聚低耦合的原则，将结构分解为不同的模块，各模块分别负责自己的功 能，且模块与模块之间，模块与环境之间的调用关系尽可能简单。同时，在进行概要设计分析的过程中，发现并解决了系统中各个模块之间功能重合、冗余和功能短缺的部分，为详细设计和代码编写提供了保障。

本文档的目标读者为本平台的开发人员和二次开发人员。

1.2 概要设计依据

本文档依据需求分析规约的功能建模部分重写。

1.3 参考资料

参考《计算机软件产品开发文件编制指南（GB8567-88）》。

［1］iS3项目团队 iS3 需求分析规约（说明书）

［2］GB8567-88，计算机软件产品开发文件编制指南

1.4 假定和约束

**开发条件**：成员自备电脑，并配置相关环境

**经费支持**：无经费支持

**设备条件：**iS3 主程序采用语言 c# + Python，基于.Net 4.5 Framework 框架开发而成，

建议采用64 位 windows 操作系统，

系统至少为 Windows 7 SP1，

处理器至少 1.6 GHz ，4 GB RAM， 并具备 11 GB 可用硬盘空间。

1.5 定义

|  |  |
| --- | --- |
| **词汇** | **说明** |
| Project | 用户在平台上可查看的每个项目叫Project。 |

|  |  |
| --- | --- |
| Domain | 每个Project中，有不同的土木专业方面的域的划分，每个域是一个Domain。 |
| DGObjects | 每个Domain中，有很多数字化对象组，每个数字化对象组就是一个DGObjects。 |
| DGObject | 每个DGObjects中，有一个个不同的小单元，每个对象叫数字化对象，即DGObject。 |
| 用户 | 只浏览查看现有的iS3项目的用户。 |
| 开发人员 | 对该平台进行维护的人员。 |
| 二次开发人员 | 通过使用配置工具和二次开发模块来对iS3平台进行二次开发的人员。 |

2. 概要设计

2.1 系统总体架构设计



## 平台总体架构图

本平台客户端设计以实现各模块间的低耦合及模块内部的高内聚为导向，采用 MVC 作为总体架构；

服务器端访问 MySQL 数据库进行数据交互；

同时在系统层面设计了安全保障模块、缓存模块，为系统平台的安全和高效保驾护航。

* + 1. 数据模块

本平台作为基础设施智慧建设的平台，对于大量基础设施的数据，如钻孔，监测，土质信息等，以及2D数据、3D数据等都有涉及。

本平台使用MySQL关系型数据库存储信息。MySQL 数据库作为一个快速、多线程编程的数据库，不仅满足本系统对于访问速度的要求，而且具有强大的查询功能， 可以在同一查询中混用来自不同数据库的表，查询方便快捷。

* + 1. 模型层

模型层负责处理本平台中应用程序的数据逻辑部分，对数据对象进行存取操作。本模块访问 MySQL 数据库，采用 WPF 框架中的EntityFramework映射规则自动完成数据库表到类的映射。

* + 1. 控制器层

控制器层是本平台处理用户请求的核心层，负责对用户的 http 请求作出响应，处理本平台的核心业务逻辑。控制器层通过 Filter 过滤来自用户的请求，检查用户请求的数据格式及用户当前的登录状态，并根据判断结果对请求进行回复。

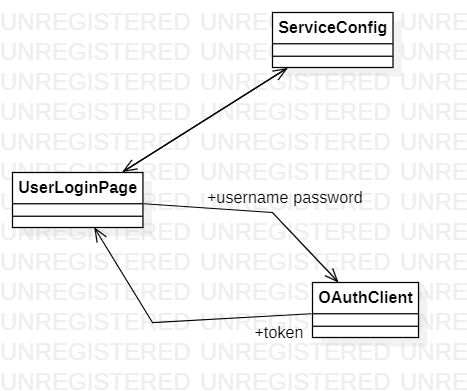
* + 1. 视图层

视图层是本平台的数据显示部分，是与用户交互的接口。负责平台的前端业务逻辑，将从服务端获得的数据渲染出来。

2.2 系统软件结构设计

* + 1. 体系结构图

# 登录功能类调用图



登陆时，UserLoginPage类先调用ServiceConfig类得到默认用户名、密码和Server端IP地址和密码。用户点击登录按钮后，UserLoginPage调用OAuthClient的函数进行身份验证，如果身份正确，则返回一个token。

# 查看项目列表功能类调用图



查看项目列表时，需要先加载数据库里的全部项目数据。因此先要调用RepositoryForProject类，RepositoryForProject类调用OAuthClient类向Server端发送请求。

# 查看项目类调用图

****

查看项目时，需要对项目和视图进行加载。在加载项目时，需要调用Project类的加载函数，分别对项目定义信息ProjectDefinition和Domain进行加载，分别调用这两个类的加载函数。在加载Domain时，需要对Domain下属的每个数字对象组进行加载，因此调用DGOjbects类。在加载DGObjects时，需要加载所有相关数据，因此调用DGObjectRepository向Server端发送请求，DGObjectRepository调用OAuthClient发送请求。

# Click

进行数据联动时，点击IS3DataGrid里的一行，与其触发器绑定的MainFrame的监听器被触发，将这个变化广播到IS3DataGrid和PlanView，并调用DGObjects类的函数查询数据中被选中的数字对象的信息。同时PlanView将该变化传到IS3View，视图上该数据对象渲染成被选中状态。

其他数据联动方式相同，即触发MainFrame的监听器，MainFrame广播该变化，其他组件做出响应。

2.3 接口设计

* + 1. 内部接口
       1. 核心类接口

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 类型 | 接口名 | 接口描述 |
| interface IService | 变量 | IPrivilegeService PrivilegeService | 权限服务 |
| Dictionary<DbServiceType,IDataService>  dataServiceDict | 数据服务列表 |
| IDataService DataService | 当前数据服务 |
| IFileService FileService | 文件服务 |
| IMapService MapService | 地图服务 |
|  | 方法 | bool IsDataServiceConnect  (DbServiceType dbServiceType) | 判断选择数据服务能否连  接 |
| void SetNowDataService  (DbServiceType dbServiceType) | 设置当前数据服务 |
| interface IMainFrame | 变量 | Project prj | 获取当前工程 |
| 方法 | IEnumerable<IView> views | 获取当前所有的视图 |
| IView activeView | 获取当前焦点视图 |
| Task<IView> addView  (EngineeringMap eMap, bool canClose); | 增加视图 |
| void output(string str); | 输出相关信息到 python |
| void runStatements(string statements); | 运行 python 命令 |
| void runPythonScripts(string file); | 运行 python 脚本 |
| event EventHandler<ObjSelectionChangedEventArgs>  objSelectionChangedTrigger; | 对象选取监听 |
| event EventHandler projectLoaded; | 项目加载事件 |
| interface IView | 变量 | Project prj | View 对应的工程 |
| ViewType type | 视图类别 |
| string name | 视图名 |
| EngineeringMap eMap | 视图内的工程地图 |
| IEnumerable<IGraphicsLayer> layers | 视图内图层 |
| ISpatialReference spatialReference | 视图坐标系 |
| IGraphicsLayer drawingLayer | 视图的绘制图层 |
| 方法 | void initializeView() | 初始化视图 |
| void onClose() | 关闭视图 |
| void highlightObject  (DGObject obj, bool on = true) | 高亮视图内对象 |
| void highlightObjects(IEnumerable<DGObject>  objs, bool on = true) | 高亮视图内对象 |
| void highlightObjects(IEnumerable<DGObject>  objs, string layerID, bool on = true) | 高亮视图内对象 |
| void highlightAll(bool on = true) | 高亮视图内所有对象 |
| IMapPoint screenToLocation  (System.Windows.Point screenPoint) | 屏幕点和真实点转换 |
| System.Windows.Point  locationToScreen(IMapPoint mapPoint) | 真实点和屏幕点转换 |
| void addSeletableLayer(string layerID) | 增加选择图层 |
| void removeSelectableLayer(string layerID) | 去除选择图层 |
| void zoomTo(IGeometry geom) | 缩放至 |
| void addLayer(IGraphicsLayer layer) | 增加图层 |
| IGraphicsLayer getLayer(string layerID) | 获取图层 |
| IGraphicsLayer removeLayer(string layerID) | 删除图层 |
| void addLocalTiledLayer  (string filePath, string layerID) | 增加 tpk 图层 |
| Task<IGraphicsLayer> addGdbLayer(LayerDef  layerDef, string dbFile, int start = 0, int maxFeatures = 0) | 增加 geodatabase 图层 |
| Task<IGraphicsLayer> addShpLayer(LayerDef  layerDef,string shpFile, int start = 0, int maxFeatures = 0) | 增加 shp 图层 |
| int syncObjects() | 同步地图和数据库对象 |
| Task loadPredefinedLayers() | 加载预设图层 |
| void objSelectionChangedListener(object  sender, ObjSelectionChangedEventArgs e) | 对象选取事件监听 |

* + - 1. 二次开发模块
         1. 该模块有的xxx接口
         2. 注册接口
         3. 登出接口
         4. 个人信息修改接口
    1. 外部接口
       1. 项目管理模块
       2. 项目配置模块
       3. 二次开发模块

2.4 界面设计

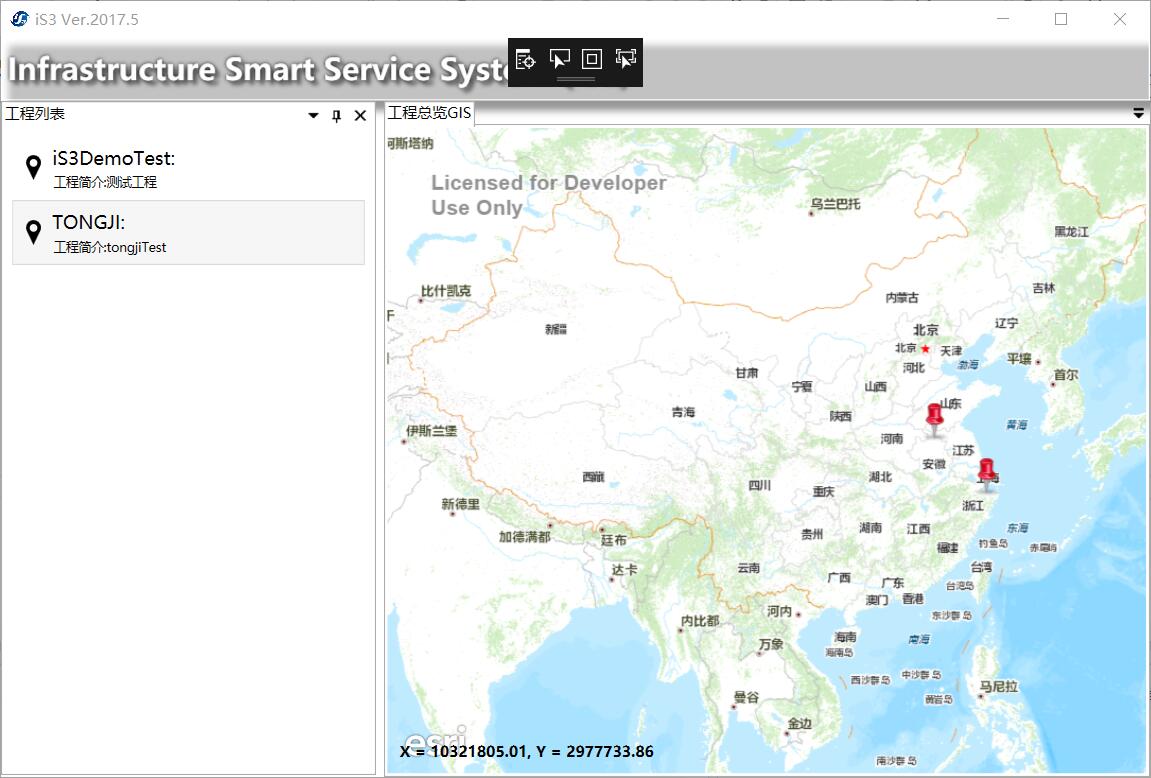
* + 1. 平台登录界面



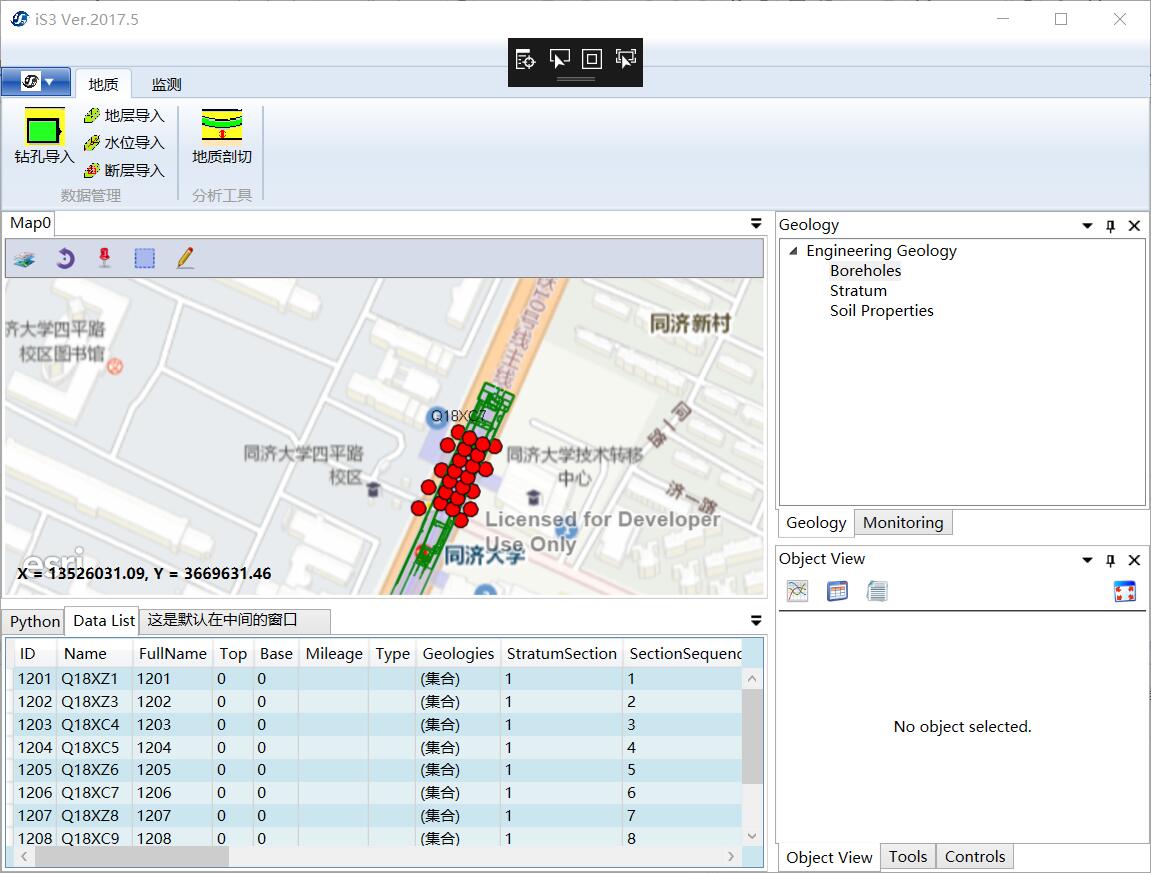
* + 1. 服务器端口配置界面



* + 1. 项目选择界面

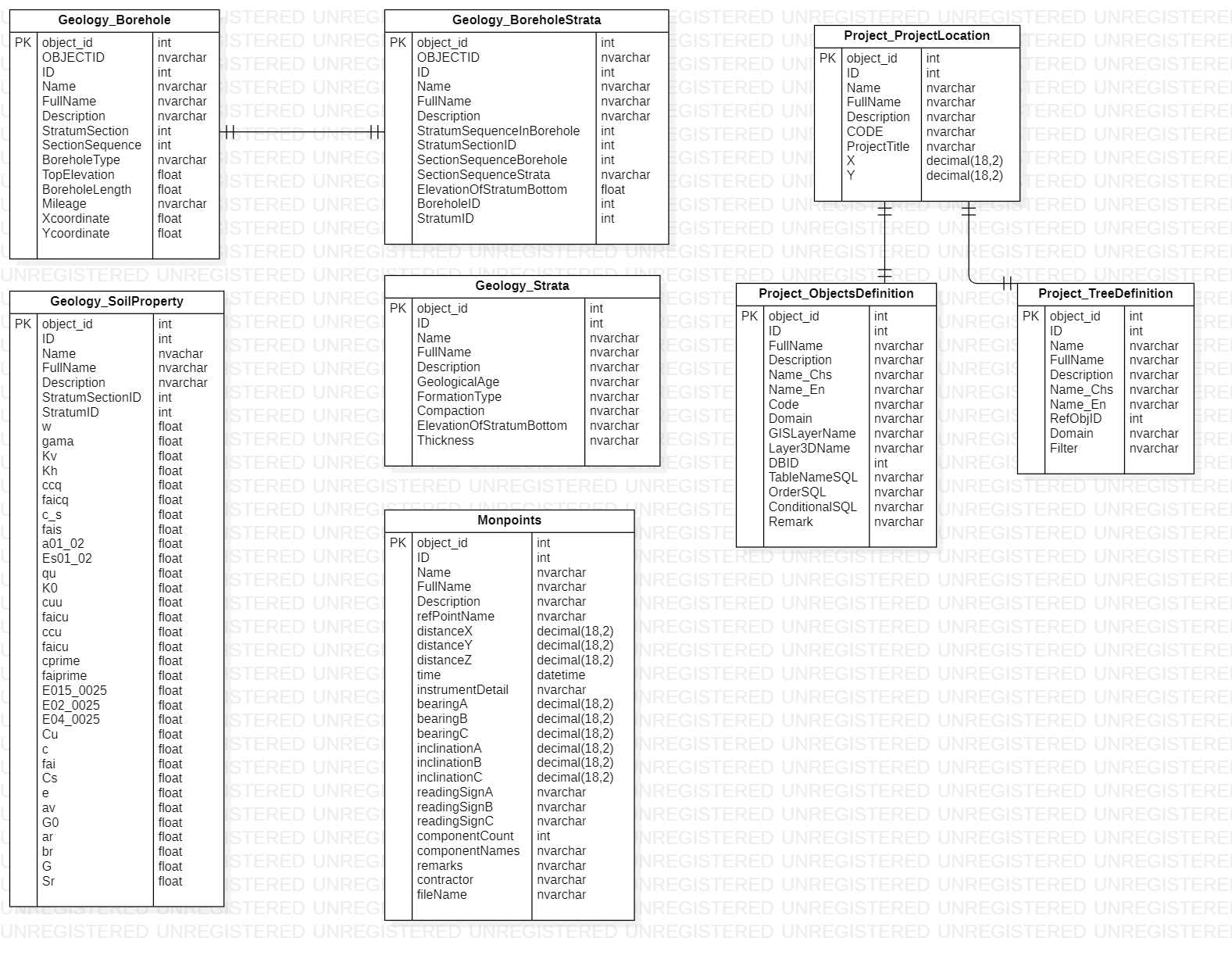


* + 1. 平台展示界面



2.5 数据库设计

本平台的TONGJI样例中，各个DGObjectd的数据及其关系数据存储在 MySQL 数据库中，依据对《需求分析规约》中的数据建模，对数据库逻辑结构进一步详细设计，得到本节中的数据库设计。



* + 1. 实体属性
       1. Borhole（钻孔）

|  |  |  |
| --- | --- | --- |
| 属性名 | 类型 | 描述 |
| object\_id | int | 主键 |
| OBJECTID | String |  |
| ID | int |  |
| Name | String | 钻孔的名字 |
| Description | String | 钻孔的描述 |
| FullName | String | 钻孔的全称 |
| StratumSection | int | 土层的区域 |
| SectionSequence | int |  |
| BoreholeType | float | 钻孔的类型 |
| TopElevation | float |  |
| BoreholeLength | float | 钻孔的长度 |
| Mileage | string |  |
| Xcoordinate | float |  |
| Ycoordinate | float |  |

* + - 1. Monpoints（监测点）

|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 描述 |
| object\_id | int | 主键 |
| ID | int |  |
| Name | string | 监测点的名字 |
| FullName | string | 监测点的全称 |
| Description | String | 监测点的描述 |
| refPointName | string |  |
| distanceX | Decimal(18,2) |  |
| distanceY | Decimal(18,2) |  |
| distanceZ | Decimal(18,2) |  |
| time | datetime | 监测时间 |
| componentCount | int | 组件数 |
| componentNames | string |  |
| remarks | string |  |
| contractor | string |  |
| fileName | string |  |

* + - 1. SoilProperty（土壤特性）

|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 描述 |
| object\_id | int | 主键 1 |
| ID | int |  |
| Name | string | 土壤特性的名字 |
| FullName | string | 全称 |
| Description | string | 土质信息的描述 |
| StrarumSectionID | int |  |
| StratumID | int | 土层ID |
| w | float |  |
| gama | float |  |
| Kv | float |  |
| K0 | float |  |
| c\_s | float |  |
| ccu | float |  |
| E015\_0025 | float |  |
| E02\_0025 | float |  |
| Cu | float |  |
| c | float |  |
| fai | float |  |
| G | float |  |
| Sr | float |  |

* + - 1. Project\_ProjectLocation（项目位置）

|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 描述 |
| object\_id | int | 主键 |
| ID | int |  |
| FullName | string | 全称 |
| Description | string | 描述 |
| CODE | string | 代码 |
| ProjectTitle | string | 项目名字 |
| X | Decimal(18,2) |  |
| Y | Decimal(18,2) |  |

* + - 1. Project\_ObjectsDefinition（对象定义）

|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 描述 |
| object\_id | int | 主键 |
| ID | int |  |
| FullName | string | 全称 |
| Decription | string | 描述 |
| Name\_Chs | string |  |
| Name\_En | string |  |
| Code | string | 代码 |
| Domain | string | 域 |
| GISLayerName | string | GIS层的名字 |
| Layer3DName | string | 3d层的名字 |
| DBID | int | 数据库id |
| TableNameSQL | string |  |
| OrderSQL | string | 命令的SQL语句 |
| ConditionarySQL | string |  |
| Remark | string | 标记 |

* + - 1. Project\_TreeDefinition（树的定义）

|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 描述 |
| object\_id | int | 主键 |
| ID | int |  |
| Name | string | 名字 |
| FullName | string | 全称 |
| Description | strings | 描述 |
| Name\_Chs | string |  |
| Name\_En | string |  |
| RefObjID | int |  |
| Domain | string | 域 |
| Filter | string | 过滤器 |

2.6 系统出错处理设置

* + 1. 系统出错处理设计
       1. **config.exe加载ProjectList时失败**

问题描述：在加载ProjectList.xml时，如果该文件不存在，程序抛出异常结束运行。

问题缘由：在加载ProjectList.xml时，未对是否存在ProjectList.xml加以判断，加载文件失败直接抛出异常。

问题解决：

1. 增加对ProjectList.xml文件是否存在的判断。
2. 增加对ProjectList函数返回值为null情况的处理。
   * + 1. **config.exe新建xml失败**

问题描述：在使用config.exe生成xml时，只能在已有xml文件且xml里有标签时才能成功，否则程序崩溃并自动退出。

问题缘由：在Step 3 - Config domains of the project中的LoadProject函数，当指定文件名的xml不存在或不存在时，返回的project值为null，无法进行后续生成步骤。程序缺少对project值为null情况的处理，不够健壮。

问题解决：在.\iS3\_2.0\_Demo-master\iS3-Desktop-Client\iS3-Config\App.xmal.cs文件中修改StartConfig函数中的在Step 3 - Config domains of the project的LoadProject函数执行后，在LoadProject函数后增加对返回值为null的project的处理：新构造一个Project。

* + - 1. **config.exe的下生成xml标签<UNKNOWN>**

问题描述：生成后的xml文件标签下的Domain类型标签为<UNKNOWN>。

问题缘由：在Step 3时生成Domain时初始化为Unknown，后续选择后只修改了Domain的Name和Type，但没有修改Domain的root属性，而root属性被初始化为Unkown。

问题解决：在.\iS3\_2.0\_Demo-master\iS3-Desktop-Client\iS3-Config\AppDomainWindow.xaml.cs中修改函数OKBtn\_Click，增加新的初始化。

* + - 1. **选中Tree Panel根项程序崩溃**

问题描述：选中Tree Panel里的根项程序会崩溃。

问题缘由：Tree Panel里的根项虽然是DGObject，但无法像Borehole之类的DGObject一样获取数据。但在代码中没有对选择的DGObject是否有数据加以判断，因此会崩溃。

问题解决：在IS3DataGrid.xaml.cs中修改DGObjectsSelectionChangedListener函数，增加对获得的objs为空的情况的判断。

* + - 1. **点击2D模型不存在的Object的DataGrid程序崩溃**

问题描述：如果一个Object的数据在DataGrid里显示，但没有对应的2D模型渲染出来，在DataGrid里选中该行，程序会抛出异常。

问题缘由：当Object对应的2D模型不存在时，选中它的DataGrid，其SelectedItem值为null，而在代码直接使用SelectedItem的属性，但没有考虑到SelectedItem是null的可能性。

问题解决：在IS3DataGrid.xaml.cs中修改DGObjectDataGrid\_SelectionChanged函数，增加对SelectedItem为空的情况的判断。

* + 1. 系统维护设计

说明为了系统维护的方便而在程序内部设计中作出的安排，包括在程序中专门安排用于系统的检查与维护的检测点和专用模块（或）类。

本系统采用 Critterism 框架检测程序运行情况和崩溃信息，开发人员可以第一时间知道程序发生错误并及时 Debug。同时，如果用户在使用网站的过程中发现任何问题，可以以邮件方式通知开发人员。系统尽可能保证每一个月发布一次更新补丁。

使用 CVS 进行版本控制，跟踪文件和文档的变更，同时可以防止覆盖其他设计者开发者在同一项目中的文件。

考虑浏览器差异兼容问题，加入 IE 浏览器条件式、CSS Hack 和其他解决方案。

保障网站的服务器安全，保证服务器所在地的物理安全，防止意外事件和人物破坏物理设备。监测系统日志，通过运行系统日志程序，系统会记录下所有用户使用系统的情形，包括最近登录时间、使用的账号、进行的活动等。日志程序会定期生成报表，通过对报表进行分析， 可以知道是否有异常现象。定期对服务器进行备份 为防止不能预料的系统故障或用户不小心的非法操作，必须对系统进行安全备份。除了对全系统进行每月一次的备份外，还应对修改过的数据进行每周一次的备份。同时，应该将修改过的重要系统文件存放在不同的服务器上，以便出现系统崩溃时(通常是硬盘出错)，可及时地将系统恢复到正常状态。