# 引言

## 编写目的与基线

*【--说明编写详细设计方案的主要目的。*

*说明书编制的目的是说明一个软件系统各个层次中的每个程序（每个模块或子程序）和数据库系统的设计考虑，为程序员编码提供依据。*

*如果一个软件系统比较简单，层次很少，本文件可以不单独编写，和概要设计说明书中不重复部分合并编写。*

*方案重点是模块的执行流程和数据库系统详细设计的描述。--】*

***该文档针对XX设计的XX模块的XX版本进行编写的***

## 参考资料

*【--列出详细设计报告引用的文献或资料，资料的作者、标题、出版单位和出版日期等信息，必要时说明如何得到这些资料。--】*

## 术语定义及说明

*【--列出本文档中用到的可能会引起混淆的专门术语、定义和缩写词的原文。--】*

# 需求及设计决策

## 输入项

### 输入项的需求

### 输入项的决策

## 输出项

### 输出项的需求

### 输出项的决策

## 约束条件

安全性、保密性、私密性的需求及选择的方法

# 配置项详细设计

## 配置项体系结构

*【--对系统组成、逻辑结构及层次进行确认，对应用系统、支撑系统及各自实现的功能进行确认，细化集成设计及系统工作流程，特别要注意因软件的引进造成的系统本身结构和公司其他系统的结构变化。包括：*

* *系统组成、逻辑结构及层次确认*
* *应用系统结构确认*
* *支撑系统结构确认*
* *系统集成确认*

*系统工作流程确认--】*

### 单元划分

### 层级结构

## 全局数据结构

### 常量

### 变量

### 数据结构

## 接口设计

### 接口标识与接口图

### XX接口

#### 接口的优先级别

#### 接口类型

#### 元素及集合特征

#### 通讯方法

#### 协议的特征

#### 兼容性

### XX接口

## 单元详细设计

*【与3.1.1一一对应】*

### XX单元

#### 限制条件以及非常规特性

#### 软件逻辑

#### 启动时内部起的作用

#### 控制交予其他配置项的条件

#### 对每个输入的响应及响应时间，包括数据转换、重命名、数据传送操作

#### 操作序列和动态控制序列

#### 异常与错误

### XX单元

### XX单元

### XX单元

# 需求可追溯性

# 注解

*【--本章应包含有助于理解本文档的一般信息（例如原理本章应包含为理解本文档需要的术语和定义，所有缩略语和它们在文档中的含义的字母序列表--】*

# 外包要求

## 检测分析模块-任务分配与管理组件（2c模式）

### 组件示意图





### 组件功能描述

任务分配页面需求如下：

1. 如示意图所示，能够加载并展示任务信息。
2. 加载的基础数据包括任务集合、分析员集合、段集合
3. 分工的概念：对于已经分配好的任务可以修改、移除；可以选择分工，修改现有的分工情况。
4. 对于以分工的段：对于已分工的段在下拉列表中就不在显示，分工只能连续分工，如：下拉的第三项到第五项被分配了，那么剩下的第一项只能与第二项合成一个任务，不能直接由第一项到第六项或者更后面（与4c模式分工逻辑一致）
5. 分工的单位：以段为单位。第一段->第二段区间时显示 第一段-第二段。（接口实现获取数据）
6. 包括：线路、段范围选择、分析员、截止日期等内容。此外，在表格中展示已经分配的任务信息；（具体页面需要后期找甲方确认）
7. 需要检测任务是否完全分配，对于没有完全分配的任务，需要提示用户，是否强制保存。

### 组件接口方法描述

/// <summary>

/// 提交任务事件

/// </summary>

/// <param name="taskList"></param>

event Action<ObservableCollection<TaskModel>> SaveEvent;

其中，TaskModel结构如下：

/// <summary>

/// 分配任务表实体类

/// <summary>

public class TaskModel: NotifyPropertyChanged

{

private string \_id;

/// <summary>

/// 任务Id，数据库自增

/// <summary>

public string ID { get => \_id; set { \_id = value; OnPropertyChanged("ID"); } }

private string \_taskname;

/// <summary>

/// 任务名称

/// <summary>

public string TaskName { get => \_taskname; set { \_taskname = value; OnPropertyChanged("TaskName"); } }

private string \_tasktype;

/// <summary>

/// 任务类型：1C/2C/4C

/// <summary>

public string TaskType { get => \_tasktype; set { \_tasktype = value; OnPropertyChanged("TaskType"); } }

private int \_processtype;

/// <summary>

/// 1:检测任务 2:标定任务（目前只存在检测任务）

/// <summary>

public int ProcessType { get => \_processtype; set { \_processtype = value; OnPropertyChanged("ProcessType"); } }

private string \_analystid;

/// <summary>

/// 分析员用户ID

/// <summary>

public string AnalystID { get => \_analystid; set { \_analystid = value; OnPropertyChanged("AnalystID"); } }

private string \_leaderid;

/// <summary>

/// 组长用户ID

/// <summary>

public string LeaderID { get => \_leaderid; set { \_leaderid = value; OnPropertyChanged("LeaderID"); } }

private DateTime? \_taskstarttime;

/// <summary>

/// 任务指定开始时间，默认为当前时间

/// <summary>

public DateTime? TaskStartTime { get => \_taskstarttime; set { \_taskstarttime = value; OnPropertyChanged("TaskStartTime"); } }

private DateTime? \_taskendtime;

/// <summary>

/// 任务指定完成时间

/// <summary>

public DateTime? TaskEndTime { get => \_taskendtime; set { \_taskendtime = value; OnPropertyChanged("TaskEndTime"); } }

private int \_processedfilecount;

/// <summary>

/// 用户已经查看的文件数，数据库提供

/// <summary>

public int ProcessedFileCount { get => \_processedfilecount; set { \_processedfilecount = value; OnPropertyChanged("ProcessedFileCount"); } }

private int \_totalfilecount;

/// <summary>

/// 用户任务中包含的文件总数，数据库提供

/// <summary>

public int TotalFileCount { get => \_totalfilecount; set { \_totalfilecount = value; OnPropertyChanged("TotalFileCount"); } }

private DateTime? \_completetime;

/// <summary>

///

/// <summary>

public DateTime? CompleteTime { get => \_completetime; set { \_completetime = value; OnPropertyChanged("CompleteTime"); } }

private int \_isabandoned;

/// <summary>

/// 是否放弃了本次任务

/// <summary>

public int IsAbandoned { get => \_isabandoned; set { \_isabandoned = value; OnPropertyChanged("IsAbandoned"); } }

private string \_lineid;

/// <summary>

/// 线路ID

/// <summary>

public string LineID { get => \_lineid; set { \_lineid = value; OnPropertyChanged("LineID"); } }

private string \_startsiteid;

/// <summary>

/// 起始站区ID，可缺省

/// <summary>

public string StartSiteID { get => \_startsiteid; set { \_startsiteid = value; OnPropertyChanged("StartSiteID"); } }

private string \_endsiteid;

/// <summary>

/// 结束站区ID，可缺省

/// <summary>

public string EndSiteID { get => \_endsiteid; set { \_endsiteid = value; OnPropertyChanged("EndSiteID"); } }

private string \_startpoleid;

/// <summary>

/// 起始杆号，可缺省

/// <summary>

public string StartPoleID { get => \_startpoleid; set { \_startpoleid = value; OnPropertyChanged("StartPoleID"); } }

private string \_endpoleid;

/// <summary>

/// 结束杆号，可缺省

/// <summary>

public string EndPoleID { get => \_endpoleid; set { \_endpoleid = value; OnPropertyChanged("EndPoleID"); } }

private string \_linedirection;

/// <summary>

/// 线路方向：上行/下行

/// <summary>

public string LineDirection { get => \_linedirection; set { \_linedirection = value; OnPropertyChanged("LineDirection"); } }

private string \_origindatapatchid;

/// <summary>

/// 上传文件包Id

/// <summary>

public string OriginDataPatchID { get => \_origindatapatchid; set { \_origindatapatchid = value; OnPropertyChanged("OriginDataPatchID"); } }

private string \_remark;

/// <summary>

/// 备注，新加字段，用来存储选择端的范围(第一段-第二段)

/// <summary>

public string Remark { get => \_remark; set { \_remark = value; OnPropertyChanged("Remark"); } }

}