**控件名称：progress\_contrast\_analysis**

**方案id：progress\_ca\_v1.0**

**1.页面初始化后台返回数据**

{

method：'init', //当前调用接口方法名

data:[ // 提醒信息数据结构

{

task\_name:"p4挖土", // 任务名称

task\_status:"1" // 任务状态 1：已完成 2：已滞后

},

{

task\_name:"p5挖土",

task\_status:"2"

},

]

}

**2.用户点击提醒信息弹框的确定后**

①前端主动调用获取截止到当前日期的所有任务数据方法

接口名称：getTaskListData

发送到后台参数：""

model.invoke("getTaskListData","")

②后台返回数据

{

method:"getTaskListData",

data:[ // 截止到当前日期的所有任务数据

{

        wbs:"G",  //wbs任务代码

        task\_name:"工作5",  //wbs任务名称

id:"123",

        owner:"张三",   //承担人

        is\_certical\_task:true,  //是否关键任务

        task\_status:"1",  //任务状态  1 未开始  2 进行中  3 已完成

        handle\_status:"0"  //处理状态  0 待处理  1 已处理

      },

      {

        wbs:"H",

        task\_name:"工作6",

id:"1234",

        owner:"李四",

        is\_certical\_task:true,

        task\_status:"2",

        handle\_status:"0"

      },

]

}

**3.用户进行筛选操作时：**

①前端主动调用获取当前点击任务的右侧需要显示的相关数据的接口

接口名称：getFilterData

发送到后台参数：filterCondition:{

conditionOne:1,  //显示选项  1 全部  2 仅显示差异

      conditionTwo:1,  //时间范围  1 全部  2 范围  时间为timeRange值 ["2020-03-01","2020-03-15"]

      conditionThree:1, //任务状态 1 全部  2 未开始  3 进行中  4 已完成

      timeRange:[]  //时间范围 为2时 日期范围所选值 数组["2020-03-01","2020-03-15"]

}

model.invoke("getFilterData",filterCondition)

②后台返回数据

{

method:"getFilterData",

data:[ // 筛选后的任务数据

{

        wbs:"G",  //wbs任务代码

        task\_name:"工作5",  //wbs任务名称

id:"123",

        owner:"张三",   //承担人

        is\_certical\_task:true,  //是否关键任务

        task\_status:"1",  //任务状态  1 未开始  2 进行中  3 已完成

        handle\_status:"0"  //处理状态  0 待处理  1 已处理

      },

      {

        wbs:"H",

        task\_name:"工作6",

id:"1234",

        owner:"李四",

        is\_certical\_task:true,

        task\_status:"2",

        handle\_status:"0"

      },

]

}

**4.用户点击某条任务时：**

①前端主动调用获取当前点击任务的右侧需要显示的相关数据的接口

接口名称：getClickedTaskDetailData

发送到后台参数：task\_id

model.invoke("getClickedTaskDetailData",task\_id)

②后台返回数据

{

method:"getClickedTaskDetailData",

data:{

beforeTaskInfo:[ // 紧前任务对当前任务影响情况表数据结构

{

before\_task\_name:"工作4",  //前置任务名称

handling\_measures:"成本压缩100元" //当前任务处理措施

},

{

before\_task\_name:"工作5",

handling\_measures:"成本压缩200元"

},

],

currentTaskInfo:[ // 当前任务情况表数据结构

{

task\_id:"1",  //任务id

task\_name:"p4挖土", //任务名称

task\_status:"1",  //任务状态

plan\_duration:"10", //计划工期

plan\_quantities:"150", //计划工程量

duration\_after\_adjustment:"12", //调整后工期

actual\_duration:"15",  //实际工期

actual\_quantities:"170",  //实际工程量

free\_time:0,  //自由时差

free\_time\_exceed:0, //超出自由时差

free\_time\_analysis\_conclusion:"自由时差分析结论", //自由时差分析结论

total\_time:0, //总时差

total\_time\_exceed:0, //超出总时差

total\_time\_analysis\_conclusion:"总时差分析结论", //总时差分析结论

}

]

}

}

**5.用户点击选择后续任务 放大镜按钮时**

①前端主动调用接口通知后台拉起选择后续任务弹窗

接口名称：pickFollowTasks

发送到后台参数：task\_id //当前任务id

model.invoke("pickFollowTasks",task\_id)

②后台返回数据

{

method:"pickFollowTasks",

data:[

{

id:"12", // 后续任务id

task\_name:"后续任务1", // 后续任务名称

},

{

id:"13",

task\_name:"后续任务2",

},

{

task\_id:"14",

task\_name:"后续任务3",

}

]

}

**6.用户选取任务后点击任务汇报详情按钮时**

①前端主动调用接口通知后台发送所选任务汇报详情数据

接口名称：getReportDetailData

发送到后台参数：task\_id //当前所选任务id

model.invoke("getReportDetailData",task\_id)

②后台返回数据

{

method:"getReportDetailData",

data:[

{

plan\_duration:"20",  //计划工期

      actual\_duration:"25", //实际工期

      duration\_completion\_rate:"90%", //工期完成率

      plan\_quantities:"40",  //计划工程量

      actual\_quantities:"35",  //实际工程量

      quantities\_completion\_rate:"76%",  //工程量完成率

      remaining\_duration:"5"  //剩余工期

}

]

}

**7.用户点击任务汇报列表详情时**

①前端主动调用接口通知后台跳转到任务汇报详情列表页面

接口名称：goTaskReportDetailListPage

发送到后台参数：task\_id //当前所选任务id

model.invoke("goTaskReportDetailListPage",task\_id)

**8.用户点击同步到进度维护按钮时**

①前端主动调用接口将后续任务决策措施数据发送到后台，后台跳转到进度维护页面

接口名称：syncToScheduleMaintenance

发送到后台参数：param:[

{

id:"12", //后续任务id

task\_name:"后续任务1"， // 后续任务名称

handling\_measures:"压缩工期5天" //后续任务处理决策

},

{

id:"13" ,

task\_name:"后续任务2"，

handling\_measures:"压缩工期8天"

}

]

model.invoke("getReportDetailData",param)