

Glodon 广联达

— 数字建设方 —

企业直达项目 项目直达客户

# 产品组件集成器-方案设计

工程产品部 | 杨震

2023-02-02



# CONTENT 目录

1 问题

2 技术选型

3 设计方案

4 方案落地

# 01

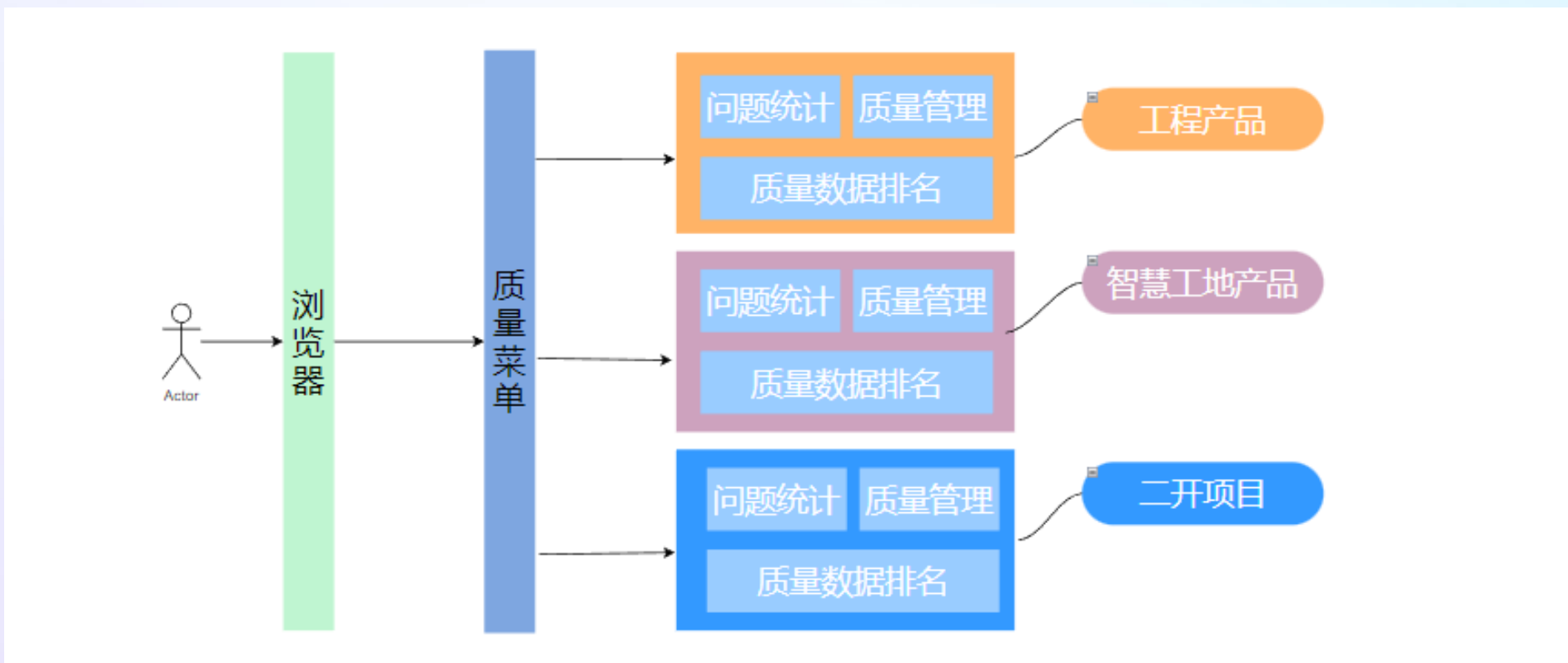
问题

标准+二开

# 01 想要达到的组件集成效果



## 前端架构模式-目前





### 实施层

- 配置模板需要在融合器**不同产品**中找到相对应的组件
- 菜单配置挂接融合器模板时需要区分是哪个产品的模板

### 产品层

- 需要考虑**模块放在哪个产品中**做才能配置到对应菜单中
- 相**类似模块太多**，无法定位哪个产品组去做

### 技术层

- 跨产品线组件开发
- 不同框架组件 (vue,react)
- **各产品重复组件**的越来越多,业务变动后组件遗漏废弃

# 02 技术选型

## 02 微前端和其他方案（体现价值）

	优点	缺点
微前端	<ol style="list-style-type: none"><li>1. 技术迭代更新，应用可以自治，可维护性高</li><li>2. 解决多团队技术栈不同问题，实现react和vue等框架的结合</li><li>3. 有一整套的解决方案，符合现有框架</li><li>4. 各产品线独立部署</li></ol>	<ol style="list-style-type: none"><li>1. 所有依赖于同一微前端基座，后续需要持续维护</li></ol>
非微前端	<ol style="list-style-type: none"><li>1. 可选择的技术多，可以根据对应场景开发</li></ol>	<ol style="list-style-type: none"><li>1. 开发js和css隔离器</li><li>2. 通讯机制</li><li>3. 解决各产品路由管理</li><li>4. 考虑按需加载性能问题</li><li>5. 写一套解决方案</li></ol>



## 02 什么是微前端

微前端是一种类似于微服务的架构，它将微服务的理念应用于浏览器端，即将 Web 应用由单一的单体应用转变为多个小型前端应用聚合为一的应用。各个前端应用可以独立运行、独立开发、独立部署。

### 优点

- 1.可以与时俱进，不断引入新技术/新框架
- 2.局部/增量升级
- 3.代码简洁、解耦、更易维护
- 4.独立部署

### 缺点

- 1.拆分的粒度越小，便意味着架构变得复杂、维护成本变高

## 02 微前端技术种类

技术类型	优点	缺点	采用度
qiankun	<ol style="list-style-type: none"><li>1. 资源预加载</li><li>2. 按需加载各产品组件</li><li>3. 技术社区丰富活跃</li></ol>	<ol style="list-style-type: none"><li>1. 上下文不一致，<b>共享状态</b>等等需要通过总线方式传递</li></ol>	<ol style="list-style-type: none"><li>1. 容器模式，符合目前业务场景</li></ol>
iframe	<ol style="list-style-type: none"><li>1. 硬隔离样式和项目</li><li>2. 模块分离，便于更改</li></ol>	<ol style="list-style-type: none"><li>1. 根据同源策略，不能操作组件外元素。比如<b>弹窗</b>只能显示出一部分</li><li>2. 需要设计一套通讯机制</li></ol>	<ol style="list-style-type: none"><li>1. <b>不考虑</b></li></ol>
Web Components	<ol style="list-style-type: none"><li>1. 组件化方案</li><li>2. 更程度的封装</li></ol>	<ol style="list-style-type: none"><li>1. 兼容谷歌34+，其他浏览器不支持</li><li>2. 需要设计一套定制的<b>通信机制</b></li></ol>	<ol style="list-style-type: none"><li>1. <b>不考虑</b></li></ol>
EMP (Module Federation)	<ol style="list-style-type: none"><li>1. 去中心化</li><li>2. 隔离应用</li><li>3. 资源共享和通信</li></ol>	<ol style="list-style-type: none"><li>1. 基于webpack5，底层不兼容目前无法涵盖所有框架</li><li>2. 拆分出去的各产品项目打包输出方式改动很大</li></ol>	<ol style="list-style-type: none"><li>1. 模块模式，各项目可以异步加载其他项目模块</li></ol>



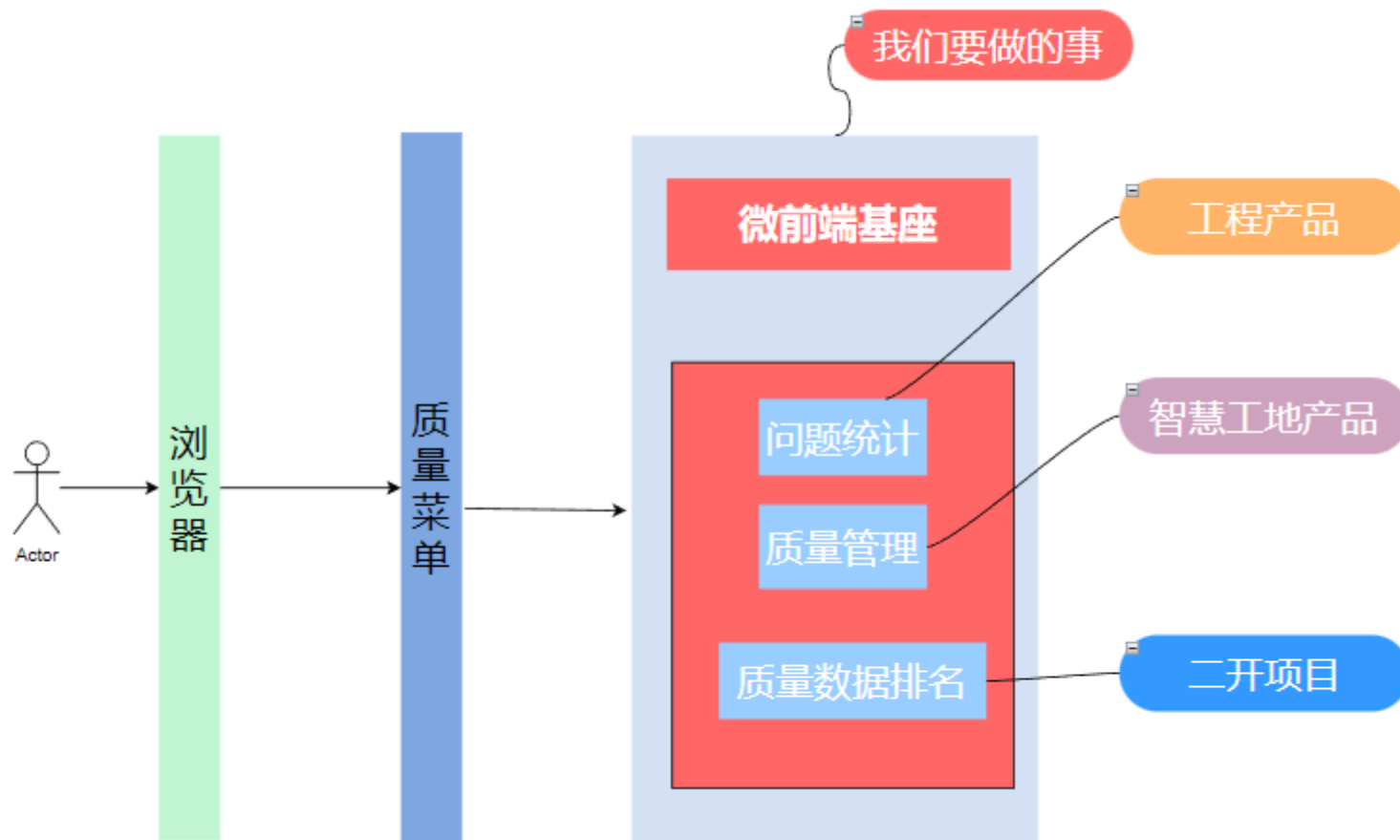
# 03


## 设计方案

### 基于微前端理念改造



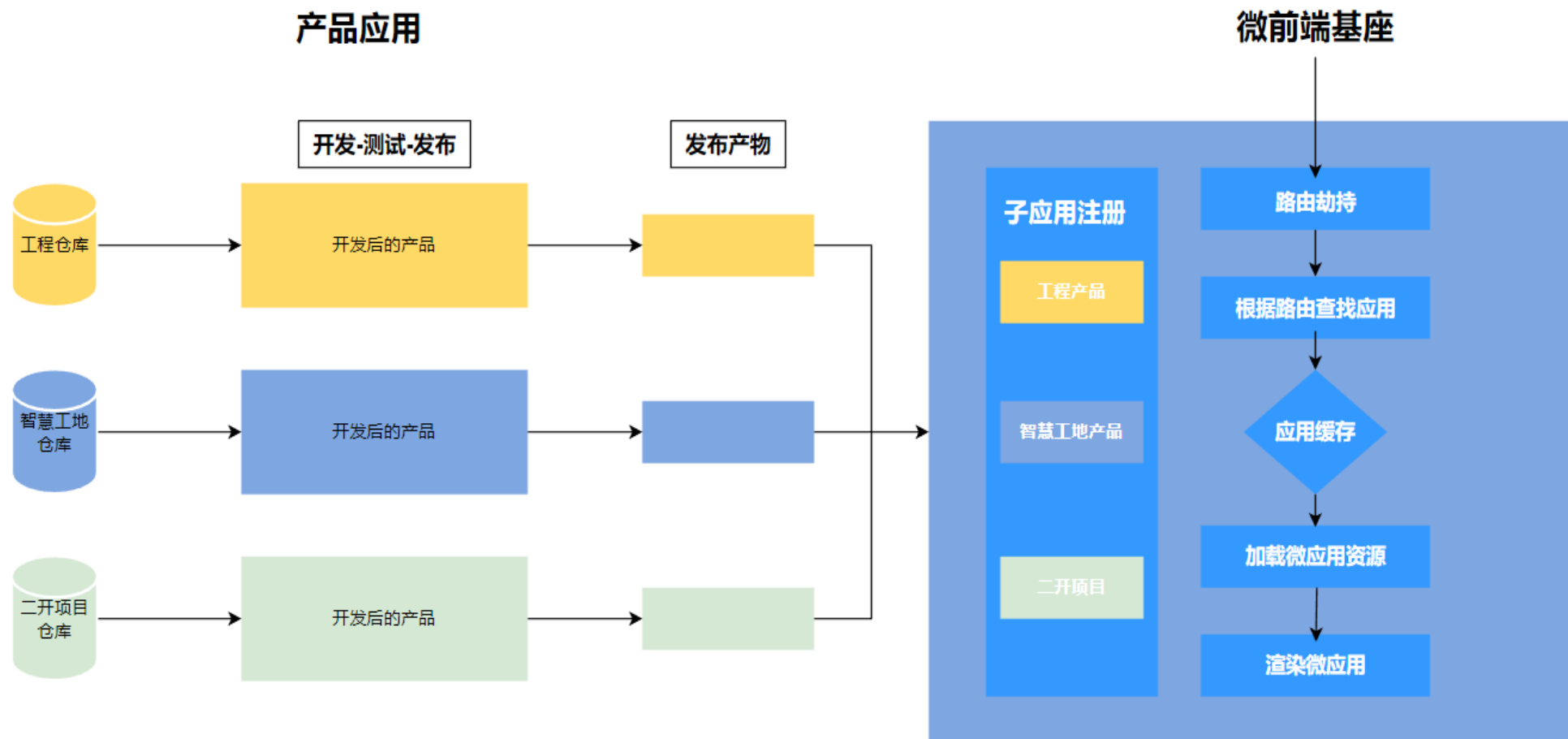
# 03 设计 - 前端架构模式-改造后





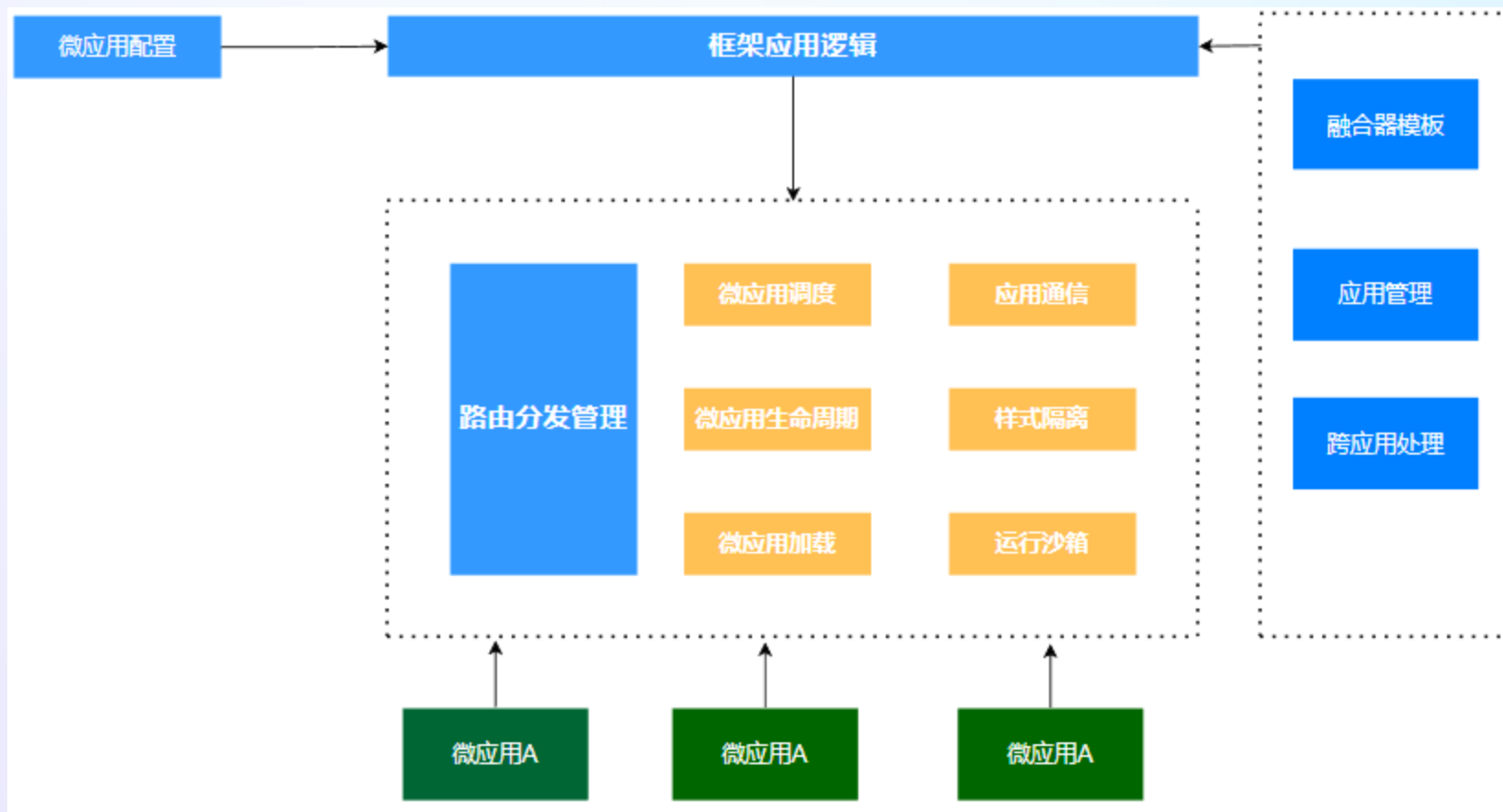
# 04 设计方案 基于Qiankun (容器模式)

# 03 设计 - 注册应用

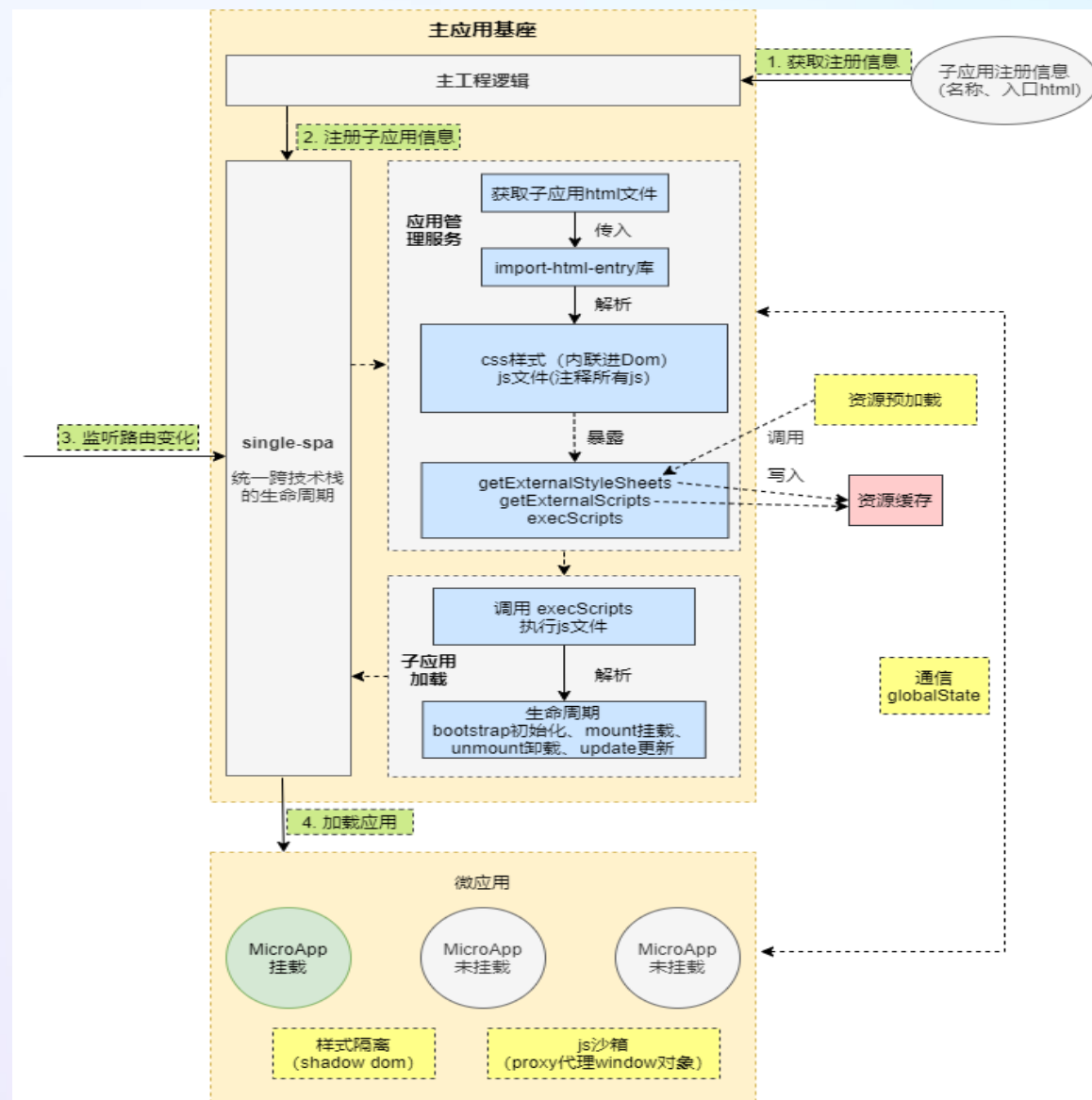




# 03 设计 - 运行期



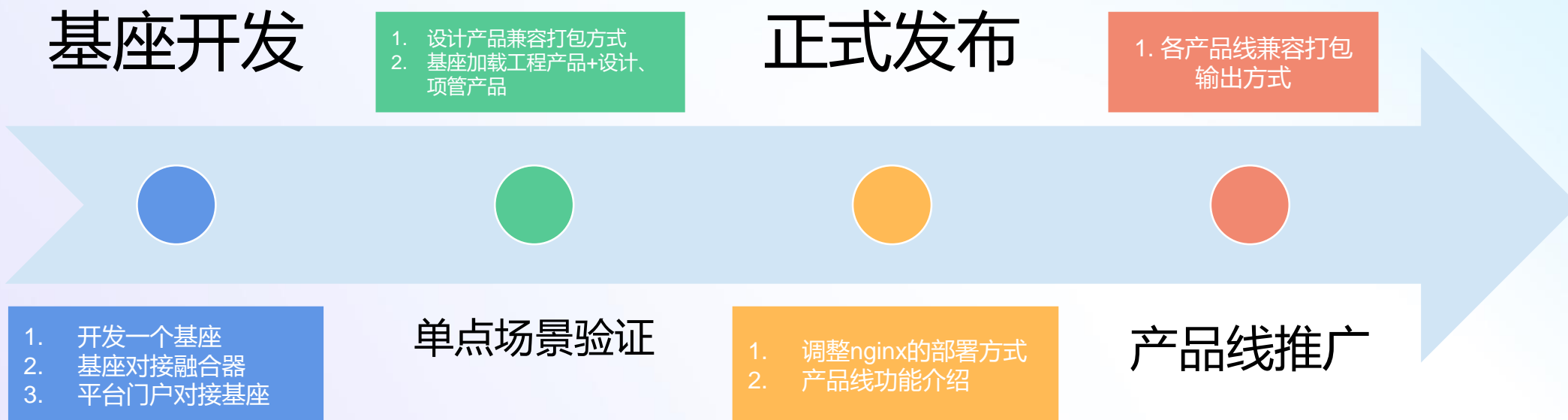
# 03 设计-运行原理



# 05 方案落地



# 04 关键路标



# 06 实施方案

# 01 主应用-关键代码

```
/**
 * 获取产品标识
 * @param {Object} item 融合器组件数据, 根据productId判断是哪个产品线的功能
 * @returns {id:,type} type: 智慧工地/ics-jfpark 工程监管/jfpark
 */
getProductSign(item) {
  if (!item?.productId) return "组件没有产品唯一标识! ";
  switch (item.productId) {
    case "jfpark":
      return {
        id: item?.productId + this.uuid(),
        type: "jfpark",
      };
    case "ics-jfpark":
      return {
        id: item?.productId + this.uuid(),
        type: "ics-jfpark",
      };
    default:
      break;
  }
},
```

```
/*
 * @Author: yangz
 * @Date: 2022-03-09 17:09:09
 * @LastEditors: Please set LastEditors
 * @LastEditTime: 2023-02-21 09:47:20
 * @Description: 乾坤主应用入口
 */
import { registerMicroApps, start } from "qiankun";

import { filterApps, microConfig } from "../registerApp";

export { qiankunActions } from "../globalState";

/**
 * @name: yangz-k
 * @Description: 微应用注册
 * @param {*} subConfig 子应用信息
 * @return {*}
 */
export const registerApps = (subConfig) => {
  const _apps = filterApps(subConfig);
  registerMicroApps(_apps, microConfig);
  start({
    prefetch: "all", // 可选, 是否开启预加载, 默认为 true。
    // 不开启沙箱模式-ui组件select,DatePicker...选择是挂接到body上, 沙箱硬隔离后到时位置有问题
    sandbox: { strictStyleIsolation: false }, // 可选, 是否开启沙箱, 默认为 true。// 从而确保微应用的样式不会对全局造成影响。
    singular: false // 可选, 是否为单实例场景, 单实例指的是同一时间只会渲染一个微应用。默认为 true。
  });
};
```



## 02 主应用-关键代码

```
/**
 * @name: yangz-k
 * @Description: 获取入口地址, 根据产品id的类型
 * @param {*} type 产品id, 对应融合器里面的productId字段
 * @return {*} 产品地址
 */
const getEntryByProductIdType = (type) => {
  switch (type) {
    case "jfpark":
      return VUE_APP_MICRO_JFPARK;
    case "ics-jfpark":
      return VUE_APP_MICRO_ZHGD;
    default:
      return VUE_APP_MICRO_JFPARK;
  }
}
/**
 * 重构apps, 解析路由往下个页面挂接参数
 * idArr [Array] 要挂接的基座载体
 * name: 微应用名称 - 具有唯一性
 * entry: 微应用入口 - 通过该地址加载微应用, 这里我们使用 config 配置
 * container: 微应用挂载节点 - 微应用加载完成后将挂载在该节点上
 * activeRule: 微应用触发的路由规则 - 触发路由规则后将加载该微应用
 */
export const filterApps = (subConfig) => {
  // 解析路由上的参数代入每个微应用页面
  const params = analysisRoute();
  const microApps = []
  for (let index = 0; index < subConfig.length; index++) {
    const item = {};
    item.entry = getEntryByProductIdType(subConfig[index]?.type)
    item.name = `${subConfig[index]?.id}-app`;
    // 必选, 微应用的容器节点的选择器或者 Element 实例。
    item.container = `#${subConfig[index]?.id}`;
    item.activeRule = "/";
    // 可选, 主应用需要传递给微应用的数据。
    item.props = {
      routerBase: item.activeRule, // 下发基础路由
      globalState: initialState, // 下发全局数据方法
      routeParams: params,
      componentCode: subConfig[index]?.componentCode || "empty"
    };
    microApps.push(item)
  }
  return microApps;
};
```

```
/**
 * @Author: yangz
 * @Date: 2022-03-09 17:09:09
 * @LastEditors: Please set LastEditors
 * @LastEditTime: 2023-03-15 14:52:37
 * @Description: 状态数据管理
 */

import store from "../store/index";
import { initGlobalState } from "qiankun";

// 定义全局下发的数据
export const initialState = {
  // 写一些数据信息
};

// 初始化全局下发的数据
export const qiankunActions = initGlobalState(initialState);

// 检测全局下发数据的改变
qiankunActions.onGlobalStateChange((state) => {
  // 修改全局下发的数据
  for (const key in state) {
    if (Object.prototype.hasOwnProperty.call(state, key)) { }
  }
});
```

# 03 子应用-关键代码

```
// webpack配置
chainWebpack: config => {
  config.plugins.delete("prefetch")
  // 修复HMR
  config.resolve.symlinks(true)
  // 打包分析
  if (argMode === "development") {
    config.plugin("webpack-report").use(BundleAnalyzerPlugin, [
      {
        analyzerMode: "static"
      }
    ])
  }
}
config.module
  .rule("images")
  .test(/\.(jpg|jpeg|png|gif|ico|ttf|otf|eot|woff|woff2)$/i) // 给这些图片类型做压缩
  .use("url-loader") // url-loader要搭配file-loader做图片压缩
  .loader("url-loader")
  .tap((options) => Object.assign(options, { name: '/static/jfpark/[name].[ext]', limit: 500 * 1024, esModule: false })); // 50M
// 解析文件 licz, 2年前 • 打包优化
config.module
  .rule(/\.(webm|ogg|mp3|wav|flac|aac|mov)(\?.*)?$/i)
  .test(/\.(webm|ogg|mp3|wav|flac|aac|mov)(\?.*)?$/i)
  .use("url-loader")
  .loader("url-loader")
  .end()
config.optimization.splitChunks({ ...
})
},
configureWebpack: {
  resolve: { ...
},
  output: {
    library: `${packageName}-${name}`,
    libraryTarget: "umd", // 把微应用打包成 umd 库格式
    jsonpFunction: `webpackJsonp_${packageName}`,
    filename: `[name].[hash].js`,
    chunkFilename: `[name].[hash].js`
  },
  performance: {
    hints: false
  },
  externals: { AMap: "AMap" },
}
```

```
// 初始化项目前，先加载环境项目配置
const initApp = async (props) => {
  window["getUrlVar"] = getUrlVar
  try {
    const res = await $get(
      api.getTenantConfig(),
      { apiType: "bimone" },
      "",
      false
    )
    window["glodonTenantConfig"] = res?.value?.length
      ? JSON.parse(res.value)
      : {}
    initView(props)
    loadTheme()
  } catch (error) {
    window["glodonTenantConfig"] = {}
    initView(props)
    loadTheme()
  }
}
// 动态加载对应环境的样式文件
const loadTheme = () => {
  if (window["glodonTenantConfig"]?.code === "cdgd") {
    setTheme("cdgd")
  }
}
/**
 * yangz
 * 项目入口配置
 * 独立运行、微应用被吊起
 */
// 1. 独立运行
if (!window["__POWERED_BY_QIANKUN__"]) {
  initApp(undefined)
}
// 2. 微应用被吊起
export async function bootstrap(props) {}
export async function mount(props) {
  initApp(props)
}
export async function unmount() {
  instance.$destroy()
  instance.$el.innerHTML = ""
  instance = null
}
```

## 04 子应用-关键代码

```
// 独立运行
if (!window["__POWERED_BY_QIANKUN__"]) {
  router = new Router({
    // 解决vue框架页面跳转有白色不可追踪色块的bug
    scrollBehavior: () => ({ x: 0, y: 0 }),
    // scrollBehavior: () => ({ y: 0 }),
    routes: _CONSTANTS_ROUTERS
  })
} else {
  router = new Router({
    mode: "abstract",
    base: process.env.BASE_URL,
    // 解决vue框架页面跳转有白色不可追踪色块的bug
    scrollBehavior: () => ({ x: 0, y: 0 }),
    // scrollBehavior: () => ({ y: 0 }),
    routes: [
      // You, 2周前 • 路由懒加载 ...
    ]
  })
}

// 解决ElementUI导航栏中的vue-router在3.0版本以上重复点菜单报错问题
const originalPush = Router.prototype.push
Router.prototype.push = function push(location) {
  return originalPush.call(this, location).catch(err => err)
}
export default router
```



数字赋能建设方  
让每一个工程项目成功



Q&A