

开源基础上的低代码集成交 换平台研发与应用实践

目 录

CONTENTS

01 集成交换平台医易通简介

单击此处添加文本具体内容

02 低代码开发技术

单击此处添加文本具体内容

03 基于MC实现低代码可视化开发

单击此处添加文本具体内容

04 效果和下一步工作

单击此处添加文本具体内容

PART 01

开源集成交换引擎MC及二次开发的集⁰¹成交换网关医易通简介

平台基础上集成传统信息系统实现互联互通化道路的障碍

③ 系统数量多、规模大小不齐

医院快速增多的信息系统，极大地增加了医院信息部门的工作负担

④ 信息系统间标准不统一

不同系统的开发语言、开发规范、数据库选择、技术架构、以及采用的数据标准都不尽相同，容易产生信息孤岛

⑤ 数据交换共享效率低

传统的点对点开发方式，开发效率低，数据碎片化存储，系统间数据共享困难，容易形成网状结构

⑥ 不断飙升的维护成本

网状结构维护困难，系统的维护需要各个公司与信息部门开发人员相互配合维护，时间成本高，工作效率低

医疗健康
信息系统

医院数据交互技术的发展



20年前

SOA（面向服务的架构）的逐渐兴起，ESB（企业服务总线）技术的引入，被用于解决旧有的协议和专有的数据格式

由于信息化建设一直缺乏顶层设计，信息难以互联互通的现象普遍存在，而集成平台被认为是解决此难题的最佳工具



10年前

传统集成平台并没有达到理想效果。更多编程语言出现、分布式、微服务，容器编排，低代码等新兴技术的崛起，出现了**低代码聚合服务平台**



现在

现有的商品化集成引擎

× 主要缺点

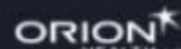
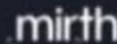
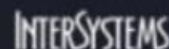
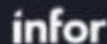
- ✓ 医疗行业对定制化需求很高，目前主流的医疗集成引擎以国外产品居多，国内产品大都基于国外的引擎进行二次开发。核心不开源，所以存在本地化程度低，如**协议标准的适配、英文界面等因素，从而导致使用效率低下**，最终的效果并不理想。
- ✓ 院内各个系统需要按照集成引擎的SDK进行对接，**因各个系统厂家技术水平不同，实施周期长，效果差，常造成漏数据、服务崩溃等问题。**
- ✓ 国内厂商自研产品专业水平较低，成熟度低，安全性能堪忧，专业化程度不高，自研自用为主。

△ 市面上常见的医疗集成引擎

- ❑ Orion Health – Rhapsody Engine
- ❑ Intersystem – Ensemble Engine
- ❑ Microsoft – BizTalk Engine
- ❑ Odin Health – Odin Engine



开源集成交换引擎Mirth Connect

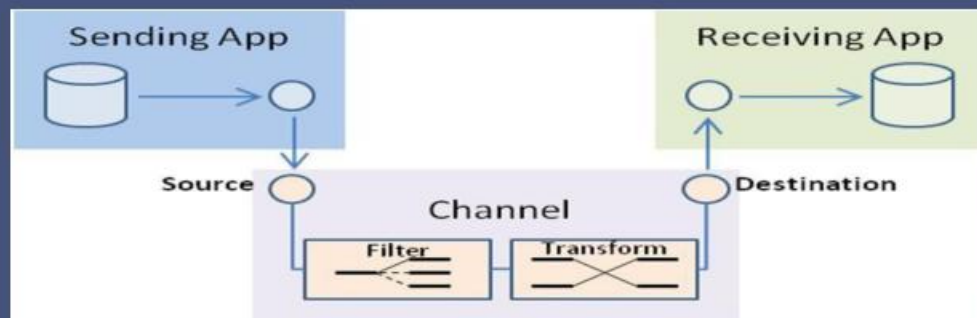


Mirth® Connect
by NextGen Healthcare



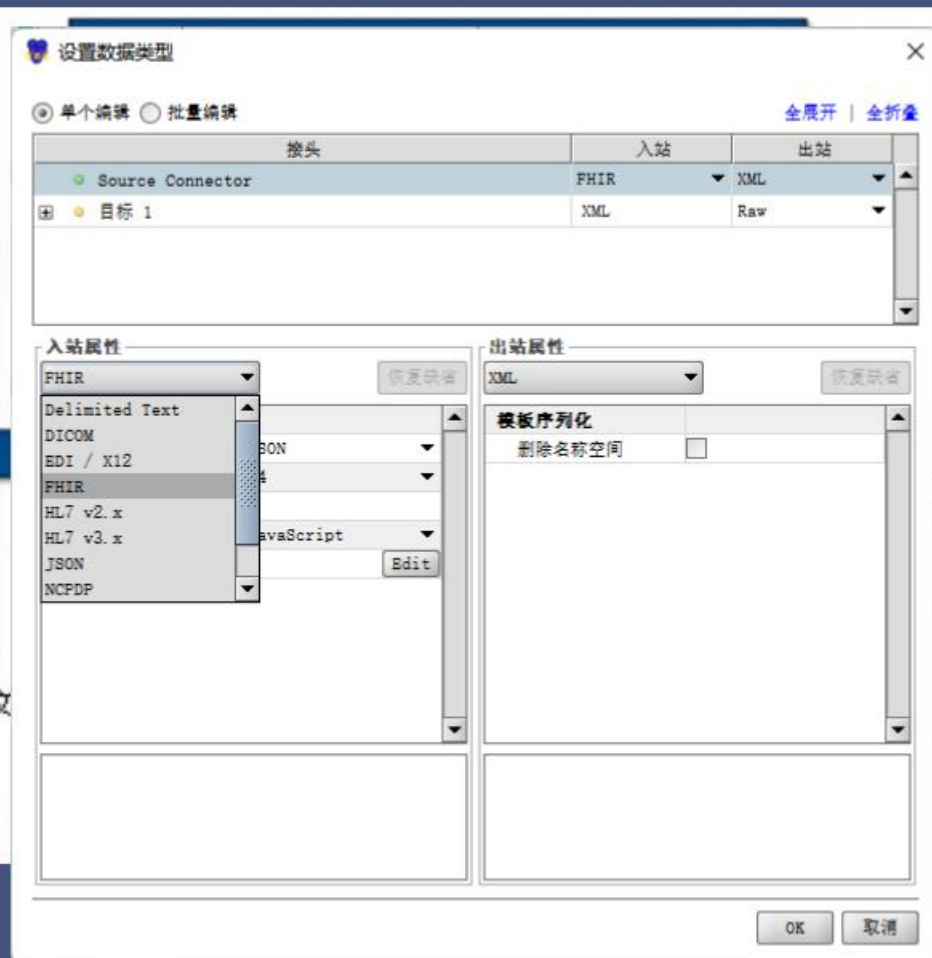
主要特点

- MC是开源的多标准接口引擎和集成平台，用在不同的医疗保健系统之间传递和转换数据。主要特点：
 - 支持多种数据交换标准：支持多种数据交换格式，包括 HL7、DICOM、XML、JSON 和 CSV 等。
 - 易于使用：提供了一个直观的图形用户界面，使用户可以轻松地创建和管理数据交换通道和转换规则。
 - 可扩展性：MC具有高度可扩展性，可以轻松地添加新的数据源、转换器和目标系统。支持JAVA/JS/XML的 DEVOPS环境，可视化现场开发，热部署。支持历史版本管理。
 - 安全性：MC提供了安全的数据传输和身份验证功能，以确保数据的保密性和完整性。它也可以充当内外网之间的网闸，实现数据交换和应用协同。
 - 高可靠性：MC 具有高度可靠性和容错能力，能够处理大量数据和故障情况。
 - 灵活部署，多场景跨平台支持，移动边缘嵌入式到服务器， Docker容器部署、系统服务模式，单节点和集群
 - 开源：MC是一款开源软件，可以免费使用和定制。





开源集成交换引擎MirthConnect简介



应用系统、文件和数据库



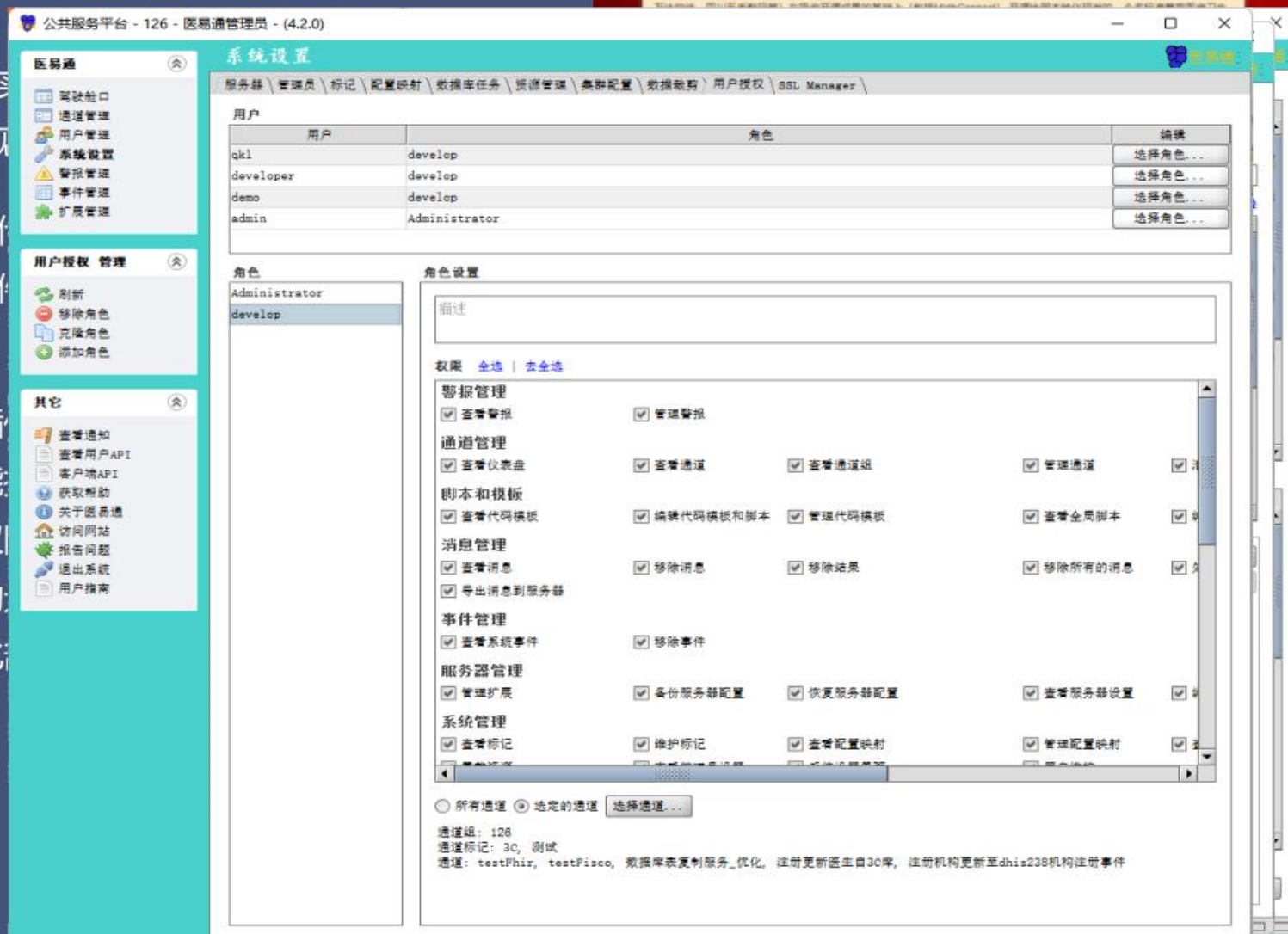
✓ 主要特点

★ 是一个开源 和平台独立的面向卫生健康行业 集成交换解决方案，可以部署在本地或云端，并具有丰富的医疗卫生行业专用的内置特色功能，可以轻松集成来自多个来源和标准规范的数据，包括：

- HL7 v2.x、v3.x、FHIR
- DICOM 数据交换标准
- 各种关系数据库连接
- 网络交换协议标准（TCP/IP, HTTPS/HTTP, FTP/SFTP, WEBDAV, NFS, SMTP）
- RAW、格式文本、XML 和 JSON 数据交换标准支持

医易通：基于MC的本地化开发和扩展

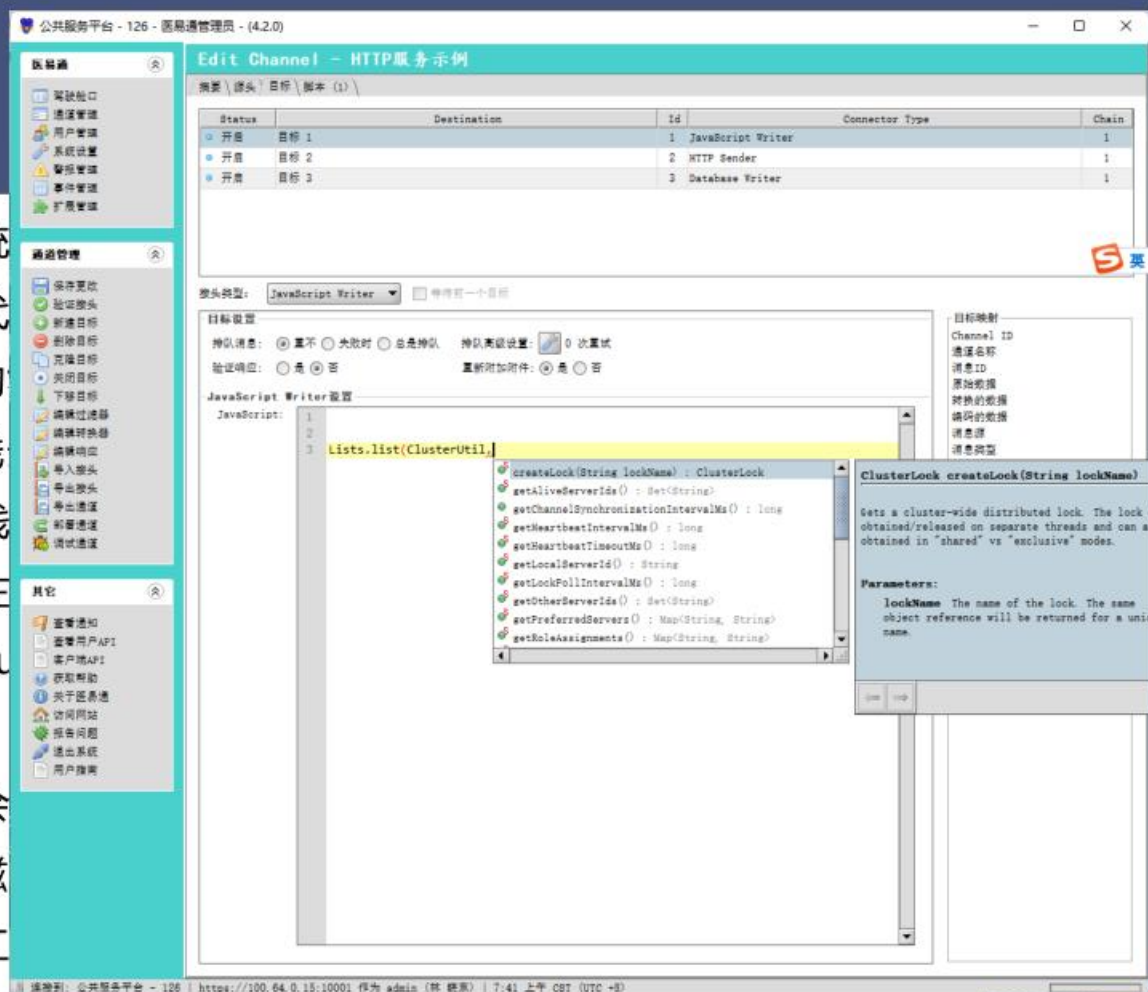
- 经过多年的研究与应用，对MC进行本地化改造，实现插件开发。
- 跟进最新的开源版本迭代。
- 开发了大量功能扩展插件：
 - 支持FHIR标准
 - 支持串口通信
 - 支持集群高可用
 - 支持高级报警功能
 - 支持灵活的用户权限
 - 支持多节点部署
 - 支持Docker容器



低代码集成交换服务平台

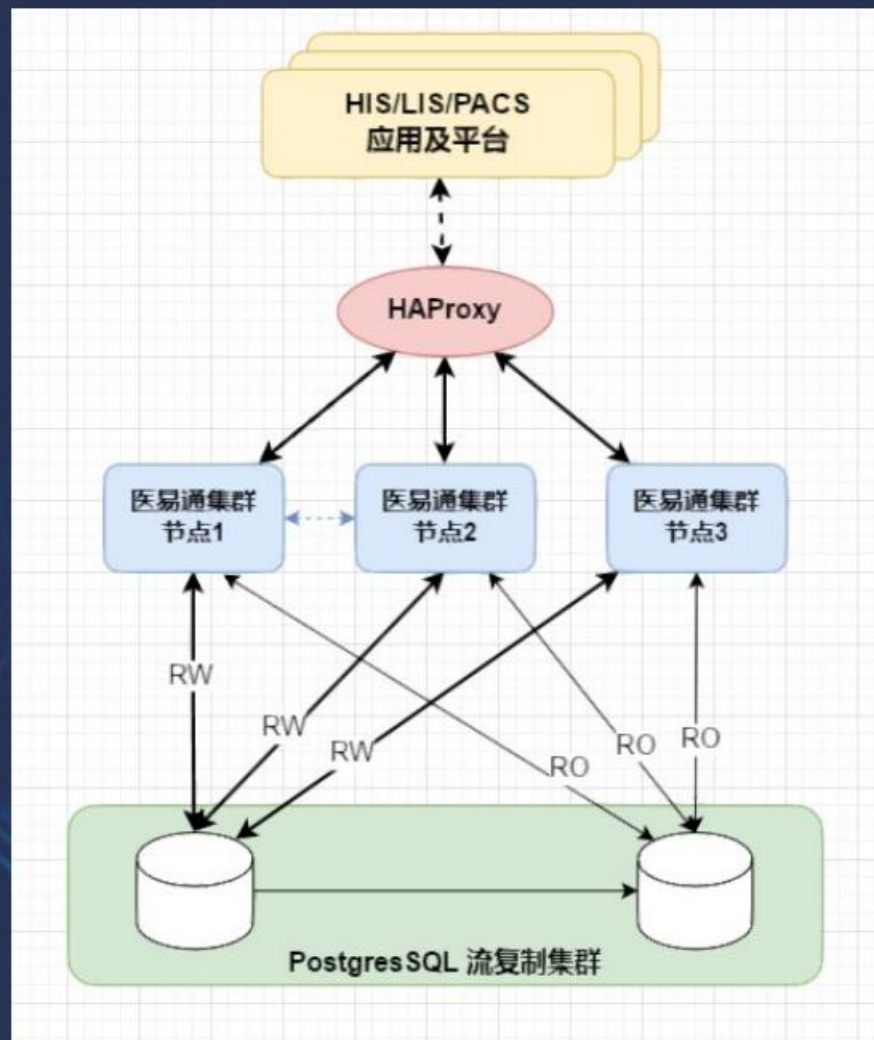
✓ 平台主要特点

- ✓ 多种数据源可视化便捷接入，包括多个系统
- ✓ 通过界面支持对接口入参，出参进行在线代
- ✓ 通过界面配置即可控制SQL的流向，数据的
- ✓ 通过界面可视化配置FaaS接口，支持js在线
- ✓ 通过界面可视化配置消息生产与消费与在线
- ✓ 通过界面可视化配置主索引接口，主索引在
- ✓ 轻量：一键部署服务集群，服务的内存/cpu性能高，速度快
- ✓ 动态扩容：流量较大、并发较高时，平台会
- ✓ 服务自愈：当节点的资源消耗(cpu/内存/磁
- ✓ 平台并不存储数据，只是院内数据的搬运工



典型的生产环境高可用部署架构

- 基于数据库集群提供服务集群所需要的数据库高可用。
- 利用医易通高级集群扩展插件支持，不是2个以上的医易通运行实例节点。
- 利用软/硬负载均衡器实现负载均衡和屏蔽MC节点实效对应用的影响。



PART TWO

低代码开发技术简介

02

PART

什么是低代码开发技术？

- 低代码开发技术是一种用于快速构建应用程序的开发方法。它使用可视化界面和图形化工具来代替传统的编程语言和代码编写，使得开发人员可以更快速地创建和部署应用程序。
- 低代码开发技术的优点
 - 提高开发效率：低代码开发技术可以大幅缩短应用程序的开发周期，使得开发人员可以更快速地交付产品。
 - 降低开发成本：低代码开发技术可以减少开发人员的编码工作量，从而降低开发成本。
 - 更好的可维护性：低代码开发技术可以提高应用程序的可维护性和易读性，使得开发人员可以更方便地进行代码修改和维护。
 - 增强创新能力：低代码开发技术可以让开发人员更加专注于业务逻辑和用户体验，从而增强创新能力和竞争力。

低代码开发技术的典型方式

- 低代码开发技术实现的主要方式包括：

- 统一的服务框架，和任务模型，简化自定义，又能支持多标准
- 自定义代码模板或函数库
- 可视化、交互式、配置式编制服务、接口和任务。并能自动生成规范代码
- 拖拽式编制代码或服务接口
- 流程化分步骤式定义业务规则和流程。
- 组合集成现有服务和任务。
- 第三方成熟库加载
- 开放架构支持三方集组件成和扩动件式态展

- 总之，低代码开发技术是一种快速、高效和灵活的开发方法，可以帮助企业和开发人员更好地应对市场需求和业务变化。

平台能力-业务系统的对接

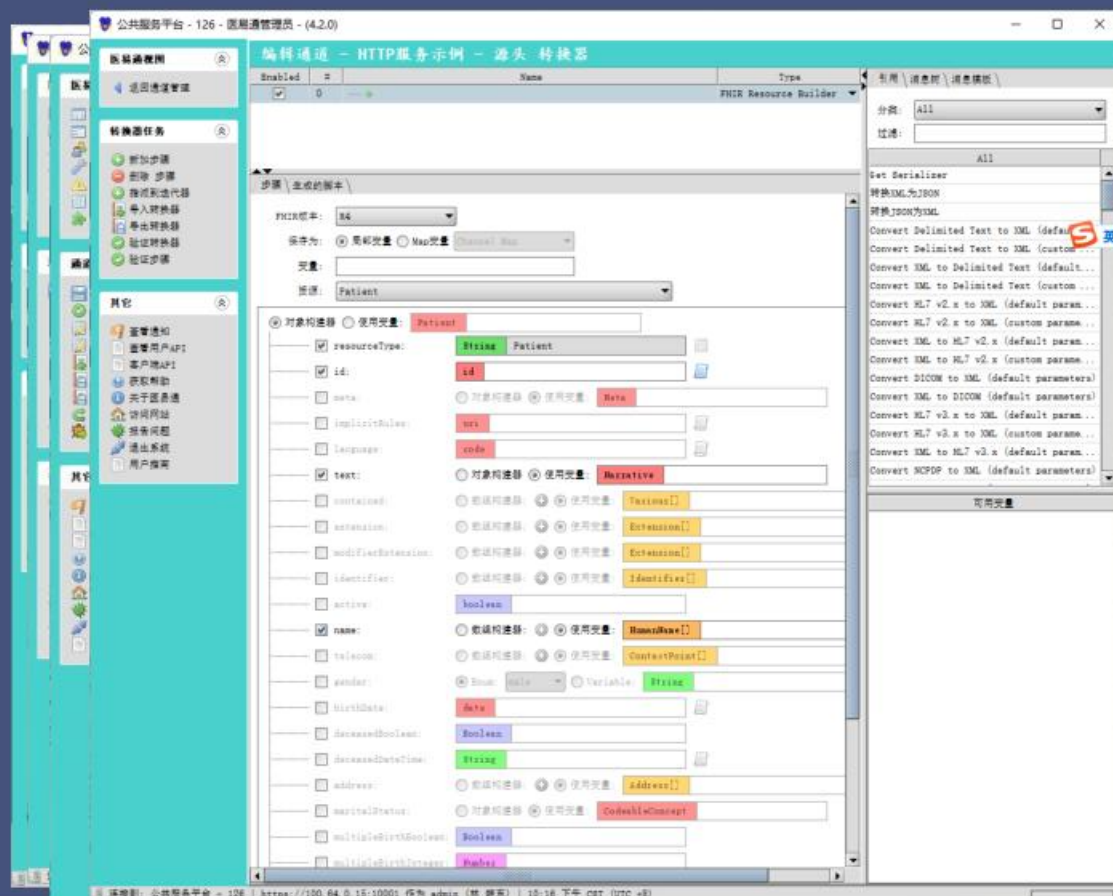
★ 痛点问题：院内存在众多业务系统，接口规范各不相同，重复利用难度高。需要统一资源，统一调用协议，统一数据格式

- ✓ 统一各个业务系统的接口协议
- ✓ 支持数据库，MQ，WebService，HTTP/HTTPs，CDC等接口对接



平台能力-服务接口和任务的开发

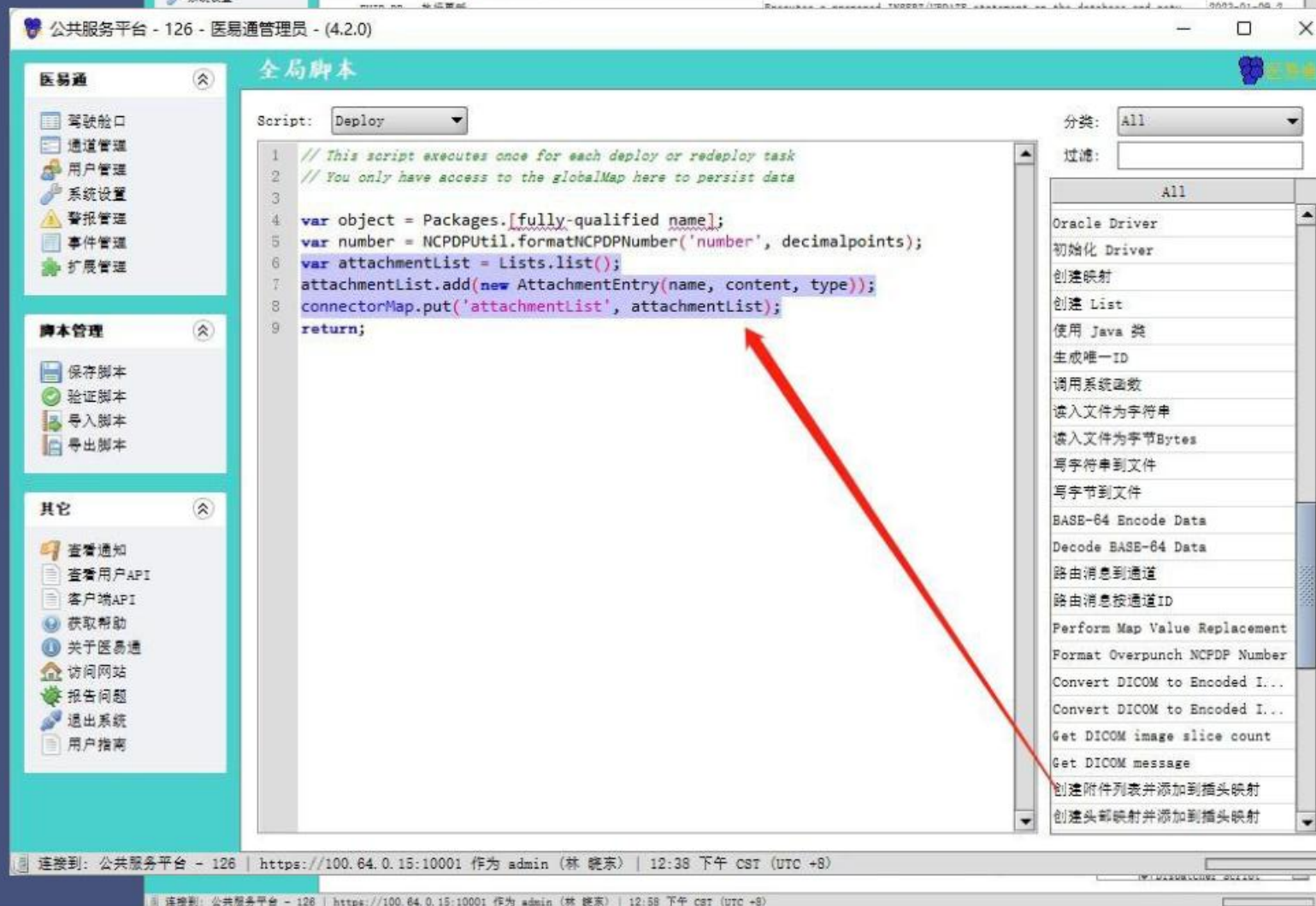
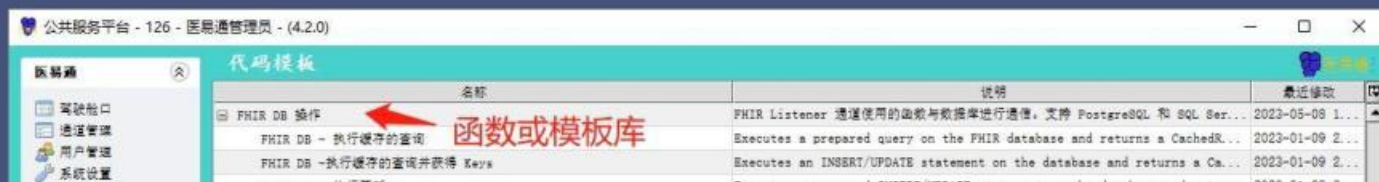
- ✓ 痛点问题：很多业务都存在灵活多变的需求，协调工程师开发一个新需求或接口，迭代版本，从需求评审、开发、测试、上线等流程下来，费时间也费力。



完全通过界面可视化配置接口各种参数，基于标准的JavaScript和Java、SQL语言在线编辑，可视化配置，就能完成一个接口服务和数据治理和业务流程的开发，只需要具备SQL、JS/Java的编写能力，就可以完成大部分通用业务的在线开发

- ❑ 可视化配置、现成可视化编辑
- ❑ 界面逻辑控制语义支持
(JAVA/JavaScript,SQL)
- ❑ 在线调试
- ❑ 一键保存即可自动发布
- ❑ 历史版本管理

代码模板库管理



平台能力-复杂接口开

- ✓ 痛点：有些业务逻辑复杂，除访
算法等逻辑，传统模式开发下来

传统医生站检查

预约申请

各专科系统检查

预约申请

线上各端检查

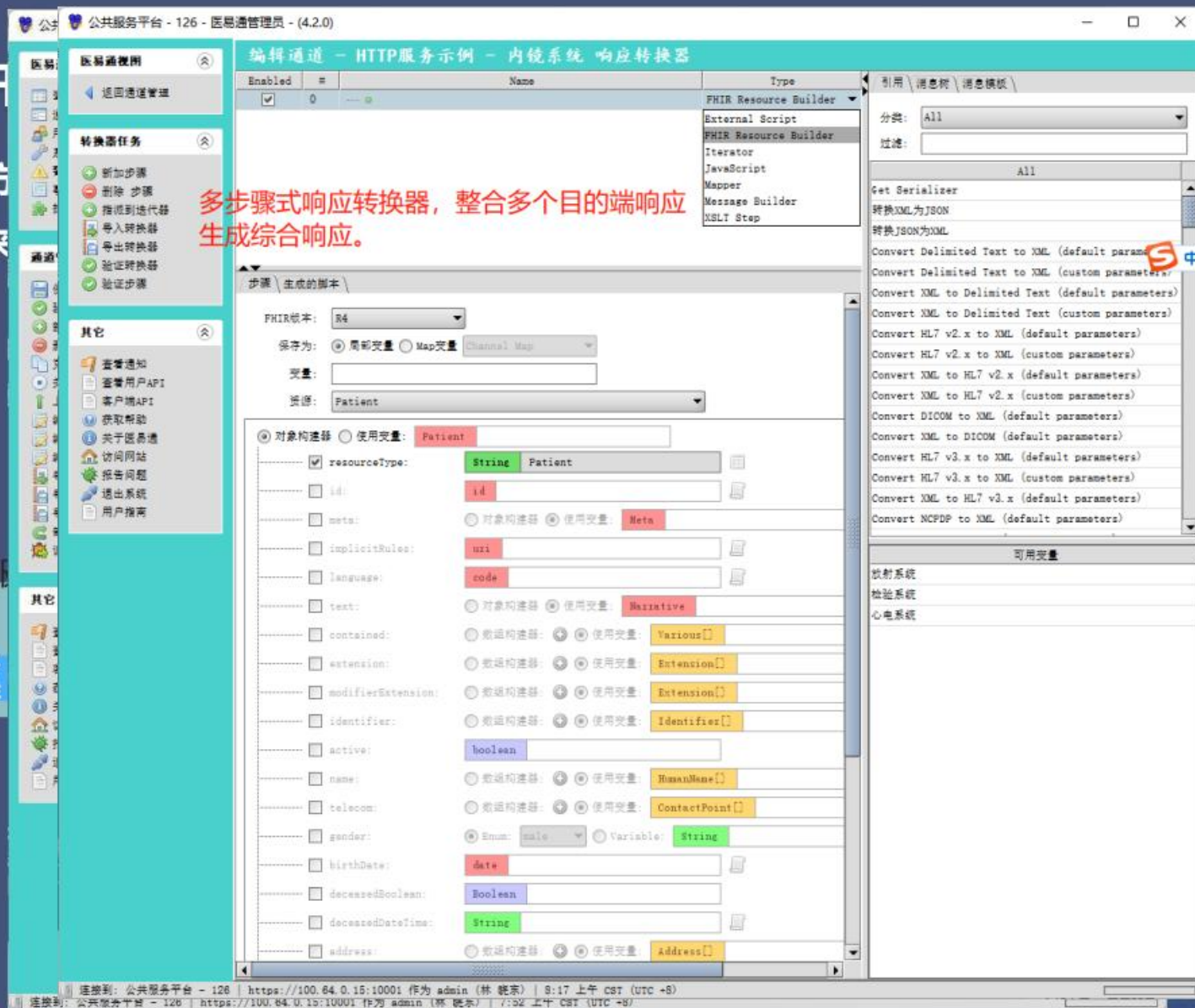
预约申请

聚合预约

多转换多
目标编排

安全保证：VM沙
箱中执行
结果返回主线程

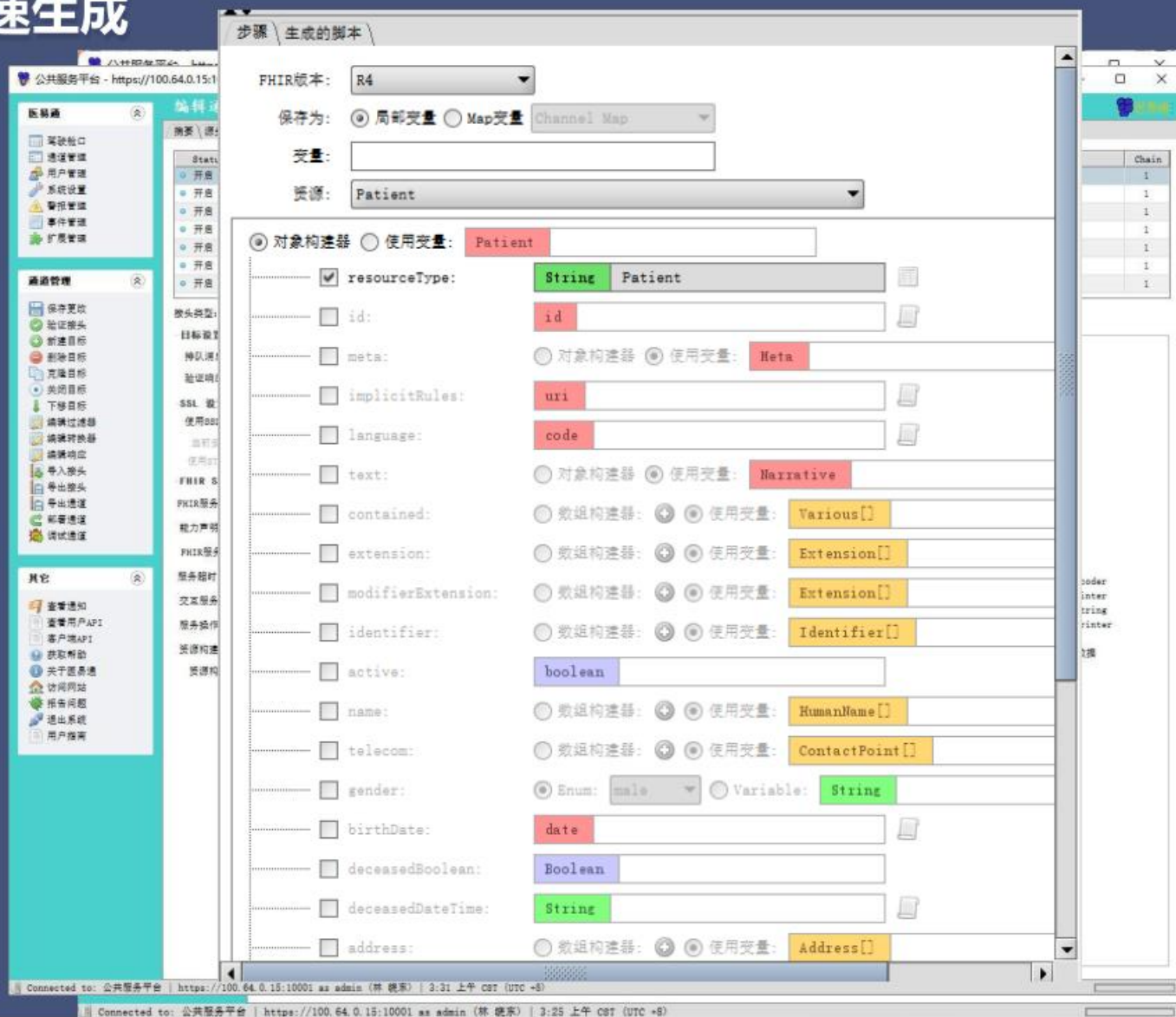
预约申请



平台能力-FHIR资源及服务快速生成

★ 痛点问题：标准化的HL7-FHIR标准资源和服务的如何快速定义和实现。

- 交互式配置服务
- 可视化资源定义



平台能力-轻量化部署

- ✓ 痛点问题：如何支持在不同应用场景灵活部署满足多种需求
- ✓ 既支持集群化部署，又支持在轻量的边缘计算如树莓派低资源部署。还支持容器化部署。
- ✓ 不管ESB或集成引擎，部署和集群管理并不是一件很容易的事情，宕机恢复时间较长，需要强有力的运维能力才能保证7*24h正常使用。

名称	环境
操作系统	跨操作系统+JAVA
部署环境	Java 1.8以上/Docker 18+
计算单元	>1台
单台CPU数量	4 核
单台内存	4G
单台硬盘大小	SSD 10G
数据库	Postgresql/MySQL/Oracle/Sqlserver/derby/rdbms

- ❑ 在容器化环境下，一键就可以部署启动（非K8S）
- ❑ 单个节点实际使用内存100M以下，一台虚拟机可以启动多个节点并行处理请求
- ❑ 单个节点秒级启动
- ❑ 整个集群10秒内完成启动或重启
- ❑ 服务所在的虚拟机/裸机只需40G磁盘
- ❑ 界面服务集群状态可视化监控，手动点击即可移出服务集群

平台-更多能力

- ❑ 主索引界面配置
- ❑ 定时任务在线代码制作
- ❑ 组件仓库（内置JWT，短信，加解密等直接使用）
- ❑ 单次调用链路可视化展示
- ❑ 细粒度的权限控制
- ❑ 接口流量控制能力和手动调节
- ❑ 磁盘空间的检查与告警
- ❑ 内存的定时检查与告警
- ❑ 第三方数据库连接池信息的检查与告警
- ❑ 慢API查询的检查与告警
- ❑ 关键性能指标达到阈值的自我停止服务，重启和再次加入
- ❑ 消费节点健康状态定时检查
- ❑ 异常数据的重试流程和配置

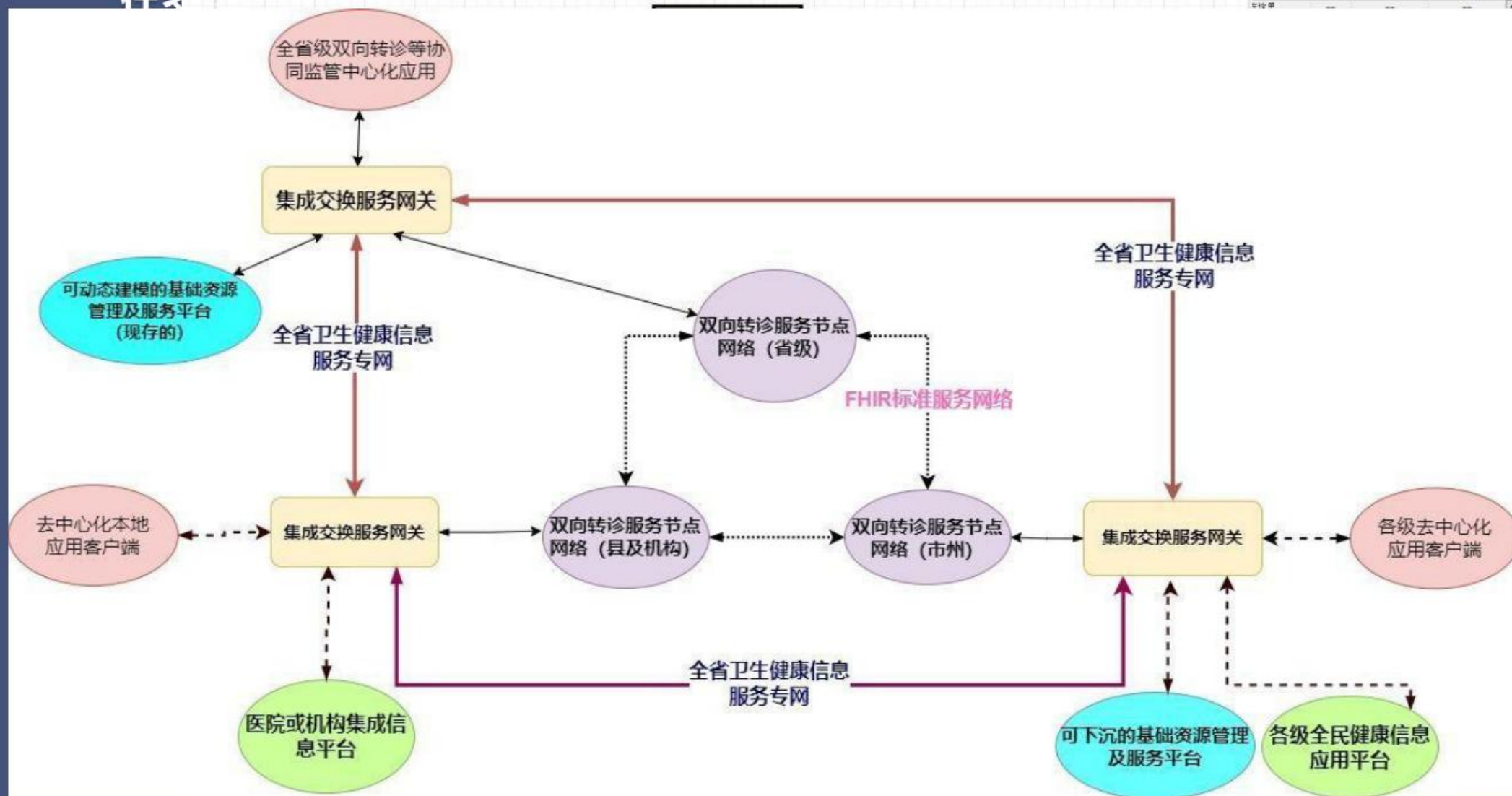
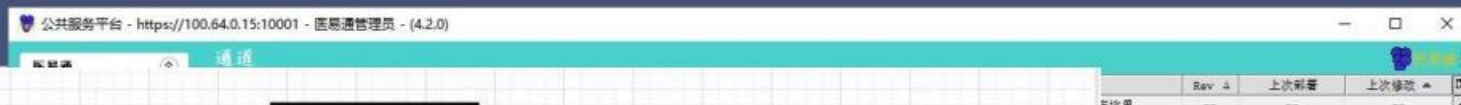
PART THREE

实践成果及展示

03

PART

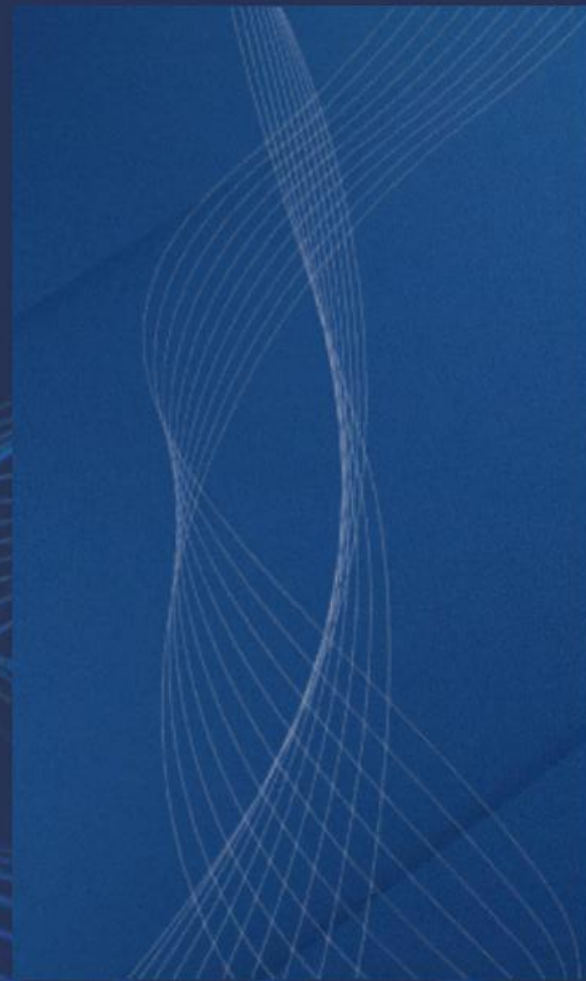
在多



PART 01

未来工作规划

04



未来工作规划

30%

继续完善平台的现有功能，增强平台的低代码能力，加强容错机制和问题定位能力，减轻用户的使用负担。



60%

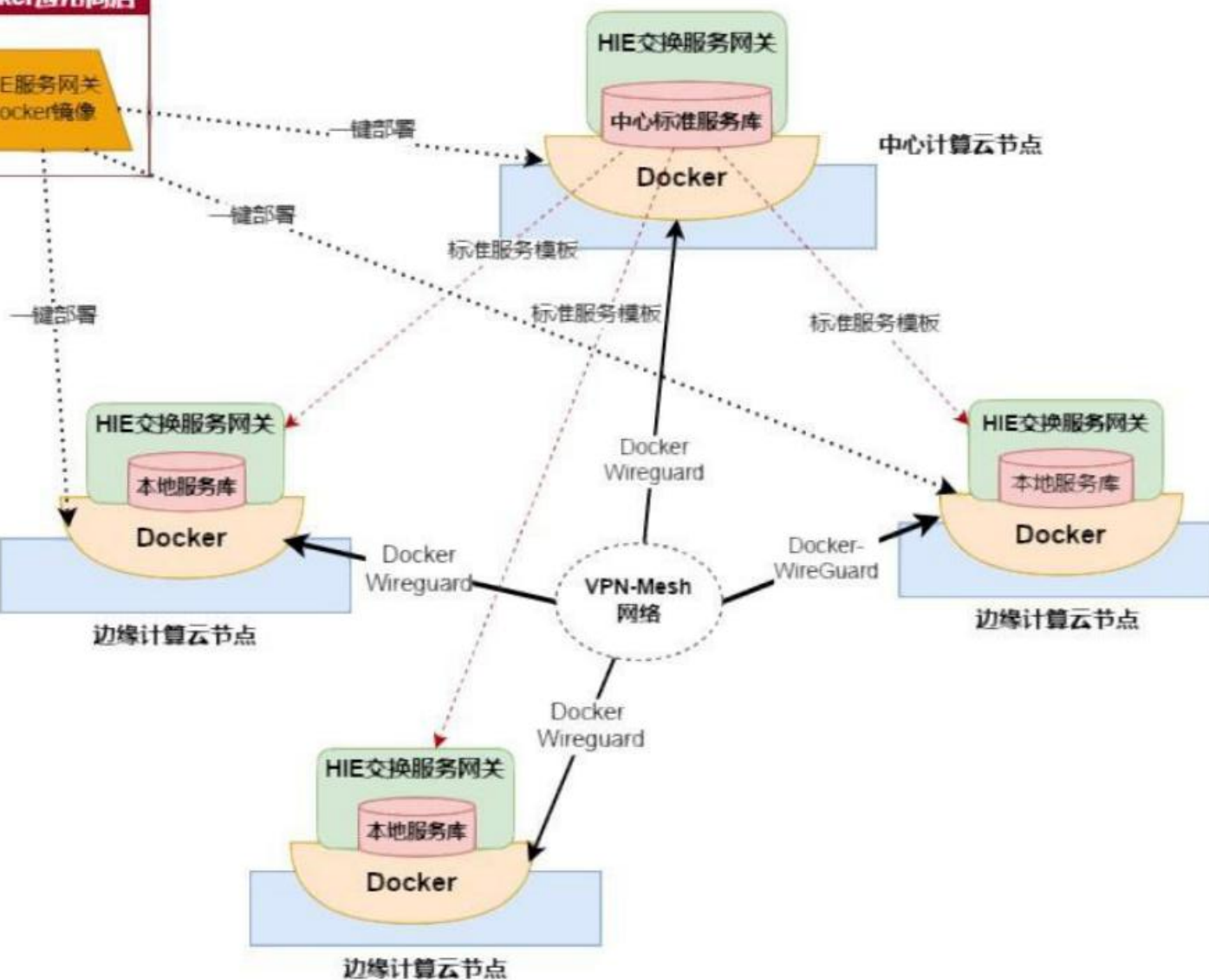
为平台添加智能运维服务，可实时观察容器状态，并且一切运维操作，如服务节点弹性伸缩，节点重启等，都可在界面可视化操作。



90%

深化低代码聚合服务平台应用，聆听用户的需求与建议，反哺并赋能平台，使平台具有更强、更稳定的能力。





可动

应用

用客户端