拥抱云时代的前端开发架构——微前端

原创 克军 Alibaba F2E 2019-11-29

微前端架构旨在解决单体应用在一个相对长的时间跨度下，由于参与的人员、团队的增加，从一个普通应用演变成一个巨石应用(Frontend Monolith)，随之而来的应用不可维护的问题。这类问题在企业级 Web 应用中尤为常见。

微前端的价值

阿里云提供的很多商业化产品和服务，本质上是对外提供「能力」，普惠中小企业。目前，能力输出主要是通过 OpenAPI，用以集成到企业自己的业务场景中，这里主要解决的还是企业底层的能力问题——无需雇佣算法工程师，就可以拥有语音、图像识别等能力。安全也是一样，不需要找安全专家，普通的工程师就可以通过控制台高效地处理各种安全事件。

但是，随着**云技术**不断的下沉，与产业结合的越来越紧密，OpenAPI 唯有把粒度做得越来越细，才能满足各种各样的业务场景，但同时企业侧的学习成本和开发复杂度自然就上去了。控制台做为管（理）控（制）这些能力的工具，目前也只能算是「标品」，必须为了满足不同体量、不同业务特点的需求，灵活地组合和部署，就像是用户自己开发的一样。

综上所述，微前端的价值有 3 点：

1. 解决产品侧的扩展性和组合性。化整为零，自由组合。
2. 解决能力输出的「最后一公里」。
3. 云生态中的「新物种」 — 微应用。

如果微前端只存在**工程上**的价值，那它是不值得大张旗鼓去做的。

我认为，前端团队需要在这个方面做出业务价值。如果你问我 Ant Design 有什么技术价值？它的价值就是有大量的企业在用，形成某种能力依赖，不需要找设计师或者多么资深的前端工程师，就可以做出看上去很专业的后台界面。

在这条价值链路上，OpenAPI 太底层，控制台不灵活，UI 库太通用。其中的空白点是**绑定能力的商业化组件**。举个例子，企业的后台管理页上，可以直接 inside 一个「漏洞管理」的微前端应用，和一个 DataV 的微前端应用展示数据，只需要简单配一下即可，不用开发，就能做到“就像自己开发的一样”。反过来也一样，ISV 在阿里云的产品平台上，不仅可以通过小程序的形式，也可以通过微前端应用的形式输入自己的服务。

微前端的问题域

简单地说，搞微前端目的就是要将产品原子化（跟原子化的 OpenAPI 一个道理），再根据客户业务场景组合。每个功能模块能单独迭代，自由集成当然好，但维护成本怎么控制。怎么调试、公共组件版本控制、众多同窗微应用之间怎么“和谐相处”等等。微前端并非只是解决在页面上异步加载一个模块就完事了，更多的是将改造引发的一系列问题需要通过体系化的方案解决，否则就变成反生产力工具。

目前，阿里的微前端方案有 qiankun(乾坤)、Magix、icestack、以及内部很多的微前端解决方案。或多或少都带有一些自身的业务特色，没有明确提出标准，或者明确定义微前端的技术体系到底包含哪些内容。这方面有项目落地的团队真应该再进一步瞄准更高的价值点做，同时广泛交流，这样才能更快得出标准化的东西。我们团队也在实践中，这里我抛出一些开放性问题讨论。

首先必须明确**微前端不是框架、不是工具/库，而是一套架构体系，它包括若干库、工具、中心化治理平台以及相关配套设施。**

微前端包括 3 部分：

- 微前端的基础设施。这是目前讨论得最多的，一个微应用如何通过一个组件基座加载进来、脚手架工具、怎么单独构建和部署、怎么联调。
- 微前端配置中心：标准化的配置文件格式，支持灰度、回滚、红蓝、A/B 等发布策略。
- 微前端的可观察性工具：对于任何分布式特点的架构，线上/线下治理都很重要。

微前端具体要解决好的 10 个问题：

1. 微应用的注册、异步加载和生命周期管理；
2. 微应用之间、主从之间的消息机制；
3. 微应用之间的安全隔离措施；
4. 微应用的框架无关、版本无关；
5. 微应用之间、主从之间的公共依赖的库、业务逻辑(utils)以及版本怎么管理；
6. 微应用独立调试、和主应用联调的方式，快速定位报错（发射问题）；
7. 微应用的发布流程；
8. 微应用打包优化问题；
9. 微应用专有云场景的出包方案；
10. 渐进式升级：用微应用方案平滑重构老项目。

通过问题理解问题是一种思考方式，相信大家能沟通通过微前端三大组成部分和它要解决的 10 个问题，能够有一个大概的理解。下面，看一下我归纳的微前端的架构体系（如图）：

通过上图，很明显的看出前后端分工，以及线上微应用相关配置流程。整体运行环境以及开发流程是非常复杂的，留到大会的时候再详细讲解。

微前端的基本原理

如下图所示，微前端的工程化是从传统前端工程化体系升级上来的。

比如构建，增加微应用类型的项目构建，有动态的打包策略。传统项目管理工具通常是命令行工具，包括构建、发布、测试，会升级为项目工作台，通过 Web 界面管理项目。一个项目包括哪些微应用，版本，发布策略等在配置中心统一管理。一个大型应用被「碎片化」后，还要能做到「一目了然」。哪个模块报错，加载失败等异常发生第一时间反应在配置中心里。

下面的原型图，就是一个最基本的配置中心的样貌。**微前端体系要可控、可观察**。

通过多个微应用的组合，能够在变化如此复杂的需求中，更好的更快的赋能业务。

云时代的前端开发模式

前端开发从 PC 时代到移动时代，从刀耕火种的原始运维到云计算时代，回顾起来，我们会发现——开发模式跟时代背景真是密不可分。前端奋斗 20 年才把页面写好，而现在又变成「切页面」了，只是此「切」非彼「切」。云时代的开发模式注定是「碎片化」的，开发是面向模块的，而页面只是一种组合场景，一种运行时容器。

我想，未来的产品开发主要时间是在「编排」——编排服务、编排逻辑、编排组件、编排访问策略、编排流程。到了云时代，一家企业只要招几个前端工程师就可以了，兼顾开发和运维、资产全部上云，运维任务通过控制台就能完成。开发借助 Serverless 和编排工具就能实现无服务端。在未来，无论是前端工程师还是全栈工程师，都将不复存在，应该叫**端到端(F2E -> E2E)工程师**了。

D2 微前端专场

本届 D2 微前端专场将邀请在微前端领域具有丰富实践的工程师，一起来为大家分享他们的理解和思考，希望能让大家对微前端有更加清晰的认识。