# 

**投 标 文 件**

**项目名称：智慧农业一期**

**项目编号：JSJY-C2019G001**

**分包号：**

**分包名称：**

**投 标 人：**

**日 期：二〇一九年 月 日**

**目 录**

一、投标函 4

二、 开标一览表 5

三、投入要素明细表 6

四、 系统功能要求响应及偏离表 7

五、 技术服务方案说明 10

1. 项目理解 10

2. 微服务架构设计说明 11

3. 农业数据资源中心设计说明 11

4. 云服务管理支撑平台设计 11

4.1系统设计宗旨 11

4.3微服务架构系统标准 14

4.4系统技术要求 20

4.4系统功能设计 37

4.5系统框架设计 44

5. 其他说明 47

六、 项目实施及售后服务方案 48

1. 项目实施管理方案说明 48

1.1项目实施总流程 48

1.2项目实施准备阶段 48

1.3项目投产阶段 50

1.4项目验收阶段 51

1.5项目组织架构说明 52

2. 售后服务实施方案说明 53

2.1售后与日常维护服务保障措施 53

2.2系统验收后的保障 53

2.3售后及日常维护服务承诺 54

3. 其他说明 55

七、 人员配置情况 56

八、 业绩一览表 57

九、 商务条件响应及偏离表 58

十、 资格证明文件 60

附件一 法定代表人授权书 61

附件二 中小企业声明函 62

附件三 书面声明函 63

投标人营业执照副本 64

上一年度的财务状况报告（成立不满一年需提供资产负债表） 64

依法缴纳税收和社会保障资金的相关材料 64

# 三、投入要素明细表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 费用名称 | 数量 | 价格（元） | 备注 |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| …… |  |  |  |  |
|  |  |  |  |  |
| 合计总价（元） | | |  |  |
| 其中：小微企业分项总价 | | |  |  |
| 非小微企业分项总价 | | |  |  |

**注：1、投标人应根据本分包项目内容的分项价格情况填写上述表格。**

2、所有价格用人民币表示。

3、表中“合计总价”应与**本分包**“开标一览表”表中“投标总价”相一致。非小微企业分项总价+小微企业分项总价=合计总价

4、合计总价请保留小数点后两位。

5、小微企业一列中，如投标人是小微企业，且提供本企业制造的货物，由本企业承担工程、提供服务，或者提供其他小微企业制造的货物的，填小微。如投标人是小微企业，但提供非小微企业制造或使用非小微企业注册商标的货物的，视同为非小微。如投标人不是小微企业，填非小微。

# 一、投标函

致：江苏杰源工程管理咨询有限公司

根据贵方的智慧农业一期项目招标文件（项目编号：JSJY-C2019G001），投标人 阿里云计算有限公司 提交**（第一包：智慧农业核心框架平台）**下述文件：

一、投标函；二、开标一览表；三、投入要素明细表；[四、](#_Toc466984857)[系统功能要求响应及偏离表](#_Toc466984858)；五、技术服务方案说明；六、项目实施及售后服务方案；七、人员配置情况；八、业绩一览表；

（1）我方已详细审核全部招标文件，包括其有效补充文件等，我方同意放弃对此有不明或误解的权利。

（2）我方完全理解并同意贵方所保留的一切权利。

（3）我方完全理解贵方不一定要接受最低报价的投标或收到的任何投标。

（4）我方同意按招标文件规定提供与本投标有关的一切资料及贵方可能另外要求的与本投标有关的任何资料，并保证已提供和将提供的资料是真实的。

（5）我方同意投标有效期为自规定的投标截止之日起六十（60）天。

（6）如果我方有招标文件规定的投标保证金不予退还的事项发生，我方同意投标保证金不予退还。

（7）一旦我方中标，我们将根据法律法规和招标文件的规定，严格履行自己的责任和义务。

（8）与本投标有关的一切正式往来通讯请寄：

地址： 邮编：

联系人： 电话： 传真：

**投标人公章（CA签章）：**

**法定代表人或授权代表（CA签章）：**

**日期：**

# 开标一览表

|  |  |
| --- | --- |
| **项目名称** | 智慧农业一期 |
| **项目编号** | JSJY-C2019G001 |
| **分包号** | 第一包 |
| **分包名称** | 智慧农业核心框架平台 |
| **投标总价（元）** | 小写：￥  大写：人民币 |

注：1、“投标总价”请保留小数点后两位。

2、投标文件制作工具中手动填写的投标价格应与对应分包投标文件开标一览表中的投标总价一致。

# 系统功能要求响应及偏离表

1、按要求填写以下表格：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 招标系统功能要求 | 投标系统功能描述 | 有无偏离 | 偏离内容及原因 |
| 1 | 提供综合管理端即桌面应用云 | 基于云服务管理支撑平台的基础框架并对应用进行增减管理，提供一个用于电脑桌面的应用服务运营环境，可以承载所有面向桌面端的应用服务。 | 无 |  |
| 2 | 桌面应数据整合处理功能 | 数据资源中心的数据整合处理的功能界面在桌面端中实现。 | 无 |  |
| 3 | 桌面应用云基础管理功能 | 提供接入50个左右业务条线的信息系统，这50个左右系统实现单点登录，提供系统链接地址导航和备忘录功能。 | 无 |  |
| 4 | 桌面应用云资讯功能 | 提供资讯编辑、全市推送、定向群体推送等功能，并推送至移动端。 | 无 |  |
| 5 | 桌面应用云登录方式 | 支持账号密码登录、二维码登录、短信验证码登录。 |  |  |
| 6 | 本期精准治理系统功能 | 本期精准治理系统的功能界面将在桌面端中实现。 | 无 |  |
| 7 | 根据综合管理端（桌面端）应用对应的移动应用即移动应用云 | 移动端包括微信服务号、IOS版本和Android版本，以及常熟两平台的移动端应用，面向市民和企业的服务建设微信端，面向政府内部的应用建设APP和微信端，并将相关服务在常熟两平台同步发布。 | 无 |  |
| 8 | 移动应用云资讯 | 资讯功能数据来源由桌面端提供。 | 无 |  |
| 9 | 移动应用云功能模块入口及登录 | 用户是智慧农业的所有用户，用户根据不同角色的权限进行展示，用户可以自由增减功能模块并调整排序，还可以设定登录时的默认进入模块。 | 无 |  |
| 10 | 移动应用云农药管理平台功能 | 实现农药购买消息推送，查询农药购买记录查询和农药购买金额统计。 | 无 |  |
| 11 | 移动应用云登录方式 | 支持账号密码登录、短信验证码登录。 | 无 |  |

**注：1、投标人应根据本项目相应分包“第四章 项目需求”中要求对照第一包投标人投报系统的功能情况进行逐项评议。**

2、若无偏离，在“有无偏离”栏中填写“无”；若有偏离在“有无偏离”栏中填写 “有”并在“偏离内容及原因”栏中作出说明；因未标明或表述含糊导致的评审风险将由投标人承担。

3、表格不够可另接。

# 技术服务方案说明

主要内容应包括但不限于以下方面：

1. 项目理解

常熟智慧农业一期项目是基础建设期，主要目标是实现智慧农业的基础框架搭建、农业各类数据上图以及若干亮点应用的上线，重点推动管理数据化。建成健壮、高效、开放的基础框架，发挥农业数据资源中心和云服务管理支撑平台的架构优势，为应用的建设和运营提供强大的支撑能力。

农业数据资源中心，是常熟五大库的农业主题库，标准规范建设，通过数据梳理，形成标准规范，实现农业自然资源基础情况大摸底，根据标准规范，整合常熟农业全产业链数据，形成农业基准数据大统一。

云服务管理支撑平台，目标是搭建强大、开放、高效、稳定、安全、易用的分布式微服务架构平台，定位于可支撑未来智慧农业发展和面向互联网用户的总体需求规模；与智慧常熟融合，秉承共建共享理念，实现智慧常熟拓展智慧农业新领域；整合管理，对常熟智慧农业一期项目进行整合管理，为项目需求和系统建设的总目标提供相应的解决方案，总体项目管理。

针对常熟市农业智能云业务运营体系，对常熟市农业智能云的服务组件进行抽象设计，针对抽象出的基础组件、桌面应用云、移动应用云、便捷登录、服务管理平台、物联网管理平台、渠道支撑平台，通过对云服务支撑平台服务组件的详细设计、应用开发与测试、系统部署与前端业务集成实施、系统优化与技术赋能，以全局把控与敏捷研发的结合，构建智能云应用服务中心，体系化地落地应用运营和基础服务。

1. 微服务架构设计说明
2. 农业数据资源中心设计说明
3. 云服务管理支撑平台设计

4.1系统设计宗旨

4.1.1设计定位

根据系统需求的数据统一和服务开放原则，我们对整个系统设计定位为基于微服务分布式系统的大中台系统结构。

系统采用基于微服务的分布式架构设计，最大的特点是可扩展性，它能够适应需求变化而扩展。现实场景中需求经常随时间而不断变化，这也对应用平台提出了很高的要求。应用平台必须要能适应需求的变化，即具有可扩展性。随着业务规模不断增大，业务变得越来越复杂，并发用户请求越来越多，要处理的数据也越来越多，这个时候应用平台必须能够适应这些变化，支持高并发访问和海量数据处理。分布式系统有良好的可扩展性，可以通过增加服务器数量来增强分布式系统整体的处理能力，以应对业务增长带来的计算需求。

分布式系统的核心理念是让多台服务器协同工作，完成单台服务器无法处理的任务，尤其是高并发或者大数据量的任务。分布式系统由独立的服务器通过网络松散耦合组成的。每个服务器都是一台独立的虚拟机，服务器之间通过内部网络连接，内部网络速度一般比较快。因为分布式集群里的服务器是通过内部网络松散耦合，各节点之间的通讯有一定的网络开销，因此分布式系统在设计上尽可能减少节点间通讯。此外，因为网络传输瓶颈，单个节点的性能高低对分布式系统整体性能影响不大。比如，对分布式应用来说，采用不同编程语言开发带来的单个应用服务的性能差异，跟网络开销比起来都可以忽略不计。因此，分布式系统每个节点一般不采用高性能的物理服务器，而是性能相对一般的虚拟机服务器。提升分布式系统的整体性能是要通过横向扩展（增加更多的服务器），而不是纵向扩展（提升每个节点的服务器性能）。  
        分布式系统最大的好处是实现应用服务层面的弹性扩展。应用服务层面的弹性扩展是相对计算资源层面的弹性扩展而言的。分布式系统由软件来对硬件进行容错，通过软件来保证整体系统的高可靠性。

4.1.2设计理念

系统在具体设计过程中，基于微服务分布式中台系统的定位，我们秉承了分布式系统设计中的几个核心设计理念。

1. **横向可扩展**

横向可扩展性（Scale Out）是指通过增加服务器数量来提升集群整体性能。纵向可扩展性（Scale Up）是指提升每台服务器性能进而提升集群整体性能。纵向可扩展性的上限非常明显，单台服务器的性能不可能无限提升，而且跟服务器性能相比，网络开销才是分布式系统最大的瓶颈。横向可扩展性的上限空间比较大，集群总能很方便地增加服务器。而且分布式系统会尽可能保证横向扩展带来集群整体性能的（准）线性提升。

1. **不允许单点失效**

单点失效是指，某个应用服务只有一份实例运行在某一台服务器上，这台服务器一旦挂掉，那么这个应用服务必然也受影响而挂掉，导致整个服务不可用。

不要让服务器满负荷运行，服务器长时间满负荷运行的话，出故障的概率显著升高。所以分布式系统采用一大堆中低性能的服务器，尽可能把负载均摊到所有服务器上，让每台服务器的负载都不高，保证集群整体稳定性。

1. **减少节点间通讯开销**

分布式系统的整体性能瓶颈在于内部网络开销。目前网络传输的速度还赶不上CPU读取内存或硬盘的速度，所以减少网络通讯开销，让CPU尽可能处理内存的数据或本地硬盘的数据，能显著提高分布式系统的性能。

1. **请求无状态**

应用服务的状态是指运行时程序因为处理服务请求而存在内存的数据。分布式应用服务最好是设计成无状态。因为如果应用程序是有状态的，那么一旦服务器宕机就会使得应用服务程序受影响而挂掉，那存在内存的数据也就丢失了，这显然不是高可靠的服务。把应用服务设计成无状态的，让程序把需要保存的数据都保存在专门的存储上，这样应用服务程序可以任意重启而不丢失数据，方便分布式系统在服务器宕机后恢复应用服务。

4.1.3技术成熟度

目前，基于微服务的分布式系统框架已经在市场上广泛应用于系统中台设计中，有了丰富的技术和经验积累。

其中，架构核心技术现状：

1. SpringCloud、Dubbo等微服务主流开发框架盛行。
2. Docker: 容器和微服务相辅相成，两大技术成熟的时间点非常契合。容器技术的成熟为微服务提供了得天独厚的客观条件。轻量化的容器是微服务的最佳运行环境，微服务应用只有在容器环境下才能保障运维效率的提升。同时，微服务应用架构对外在组件的管理会变得困难，需要用容器平台去管理中间件，才能发挥出更大价值。
3. 采用微服务架构改造应用系统，不仅仅是选择开发框架本身，还要建设一套完整的体系架构。既要实现应用模块之间的解耦，还要实现统一管理。服务化体系，包括开发框架、以及周边配套工具链和组件，比如服务注册、服务发现、API网关、负载均衡、服务治理、配置中心、安全管理、与容器的结合、监控管理等等。

4.3微服务架构系统标准

微服务分布式系统中台设计过程中，我们制定了系统搭建的数据标准，设计标准和开发标准规范，保证整个系统构建过程符合标准，从而有保障的实现系统建设目标。

4.3.1数据标准

微服务架构数据设计，遵循以下几个要点：

1. **数据易用性**

数据无论用什么方式实现，其最终目的都是为业务（或者是客户）使用的。因此，在对外提供服务的时候，数据的易用性非常关键。

1. **主、副数据及数据解耦**

每个微服务 API 的数据完全独立是不太现实的，在数据设计的时候，需要考虑降低数据间的相互依赖性。因此，首先需要确定每个微服务 API 的主数据和副数据。主数据指微服务 API 的核心数据，这种数据的增删改主要集中在某个微服务 API 中。其次，为了降低数据之间的耦合度，用数据关联表来表征数据间的关系。如果想去掉数据间的关联关系，直接去掉关联表即可，对数据本身的没有任何影响。

1. **分库分表**

随着业务数据量不断增加，单一数据库或单一数据表中会积累大量的数据。当数据累积到一定程度后，数据操作的性能会大幅下降。所以，在数据架构设计阶段就应该考虑数据的分库分表。

1. **多源数据适配**

传统的关系型数据库之外，有多种多样的数据源，比如图像、声音、视频等多媒体数据文件或数据流，CSV、TXT、Doc、Excle、PDF、XML 等各种异构数。这些数据都需要做相应的处理，转换成可管理的数据信息。因此在数据架构设计的时候，需要给不同性质的数据源配置相对应的读写适配器，同时也需要有统一调度的地方。

1. **多源数据缓存**

数据处理的性能除了处理逻辑的复杂度以外，还有很大一部分是目标数据的操作时长（含对硬件磁盘设备的读写以及网络的传输）。网络速度特别是光纤的使用后已经大幅度提高，但机器磁盘的读写效率并没有显著提高，因此减少磁盘读写是提高效率的一个重要途径。数据缓存就是把常用的数据（不会经常更改的数据）、最近使用数据放到内存中。这样就可以大幅降低系统对硬件磁盘设备的操作开销，提高整个数据系统的性能。

1. **数据集市**

数据集市是一个很大的话题。当现有的数据不能简单地通过几个表数据关联以及简单加工后就可以供业务使用的时候，就需要考虑构建数据集市。数据集市以数据运用的观点来分析加工数据，通过多源数据的导入、清洗、加工、视图做成等一系列的数据操作后，为业务提供可用的、稳定的数据源。

4.3.2技术标准

微服务架构设计，我们制定了以下设计技术标准：

1. **高内聚低耦合**

紧密关联的事物应该放在一起，每个服务是针对一个单一职责的业务能力的封装，专注做好一件事情（每次只有一个更改它的理由）。将相关的事物放在一起，可以减少不必要的服务相互调用消耗。

使用轻量级的通信方式，同步RESTful（GET/PUT/POST...），基于http，能让服务间的通信变得标准化并且无状态。

避免在服务与服务之间共享数据库。

1. **高度自治**

* 独立部署运行和扩展
  + 每个服务能够独立被部署并运行在一个进程内
  + 这种运行和部署方式能够赋予系统灵活的代码组织方式和发布节奏，使得快速交付和应对变化成为可能。
* 独立开发和演进
  + 技术选型灵活，不受遗留系统技术栈的约束。
  + 合适的业务问题可以选择合适的技术栈，可以独立的演进
  + 服务与服务之间采取与语言无关的API进行集成
* 独立的团队和自治
  + 团队对服务的整个生命周期负责，工作在独立的上下文中， 谁开发，谁维护。

1. **弹性设计**

* 设计可容错的系统
  + 拥抱失败，为已知的错误而设计
  + 依赖的服务挂掉
  + 网络连接问题
* 设计具有自我保护能力的系统
  + 服务隔离
  + 服务降级
  + 限制使用资源
  + 防止级联错误

1. **日志与监控**

当产品环境出错时，需要快速的定位问题，检测可能发生的意外和故障。而日志与监控是快速定位和预防的不二选择，在微服务架构中更是至关重要。

高度可观察，我们需要对正在发生的事情有一个整体的视角。

聚合你的日志，聚合你的数据，从而当你遇到问题时，可以深入分析原因。

当需要重现令人讨厌的问题，或仅仅查看你的系统在生产环境如何交互时，关联标识可以帮助你跟踪系统间的调用。

监控主要包括服务可用状态、请求流量、调用链、错误计数，结构化的日志、服务依赖关系可视化等内容，以便发现问题及时修复，实时调整系统负载，必要时进行服务降级，过载保护等等，从而让系统和环境提供高效高质量的服务。

1. **自动化**

在微服务架构下，面临如下挑战：

* 分布式系统
* 多服务，多实例
* 手动测试，部署，发布太消耗时间
* 反馈周期太长

传统的手工运维方式必然要被淘汰，微服务的实施是有一定的先决条件：那就是自动化，当服务规模化后需要更多自动化和标准化的手段来提升效能和降低成本。

* 自动化测试必不可少，因为对比单块系统，确保我们大量的服务正常工作是一个更复杂的过程。
* 调用一个统一的命令行，以相同的方式把系统部署到各个环境。
* 考虑使用环境定义来帮助你明确不同环境间的差异，但同时保持使用统一的方式进行部署的能力。
* 考虑创建自定义镜像来加快部署，并且拥抱全自动化创建不可变服务器的实践。

自动化一切可以自动化的，降低部署和发布的难度, 比如： 在持续集成和持续交付中，自动化编译，测试，安全扫描，打包，集成测试，部署，随着服务越来越多，在发布过程中，需要进一步自动化蓝绿部署（做到老版本到新版本的平滑过渡）还可以使用pipeline as code的实践，用代码来描述你的流水线。关于部署有很多选择，可以使用虚拟机，容器docker，或者流行的无服务架构lambda。

4.3.3开发标准

微服务架构的分布式系统开发过程，需要遵循以下要求：

1. **命名规范**

无论服务命名还是代码文件命名，都需要遵循统一的命名规范，避免产生代码和服务产生冲突。

1. **依赖和仓库配置**

在项目的pom文件添加以下父依赖,用于实现微服务基础框架构建、健康检查、路由追踪等能力在pom文件中，请按照以下规则命名groupId和artifactId： groupId:定义当前maven项目隶属的实际项目或应用，和微服务命名规范类似，按照组织-项目/应用进行命名。artifactId:请使用service-name，定义artifactId。

1. **服务配置**

将服务配置分为多部分，分别对应不同环境配置，可以通过profile进行简单切换。

1. **日志格式**

路径：为方便日志收集，请将日志输出至指定路径，可以根据需要对日志文件进行切割。  
格式：为方便日志分析，请按照标准格式生成日志，其中Exception字段可以不输出，也可以自定义其他日志。

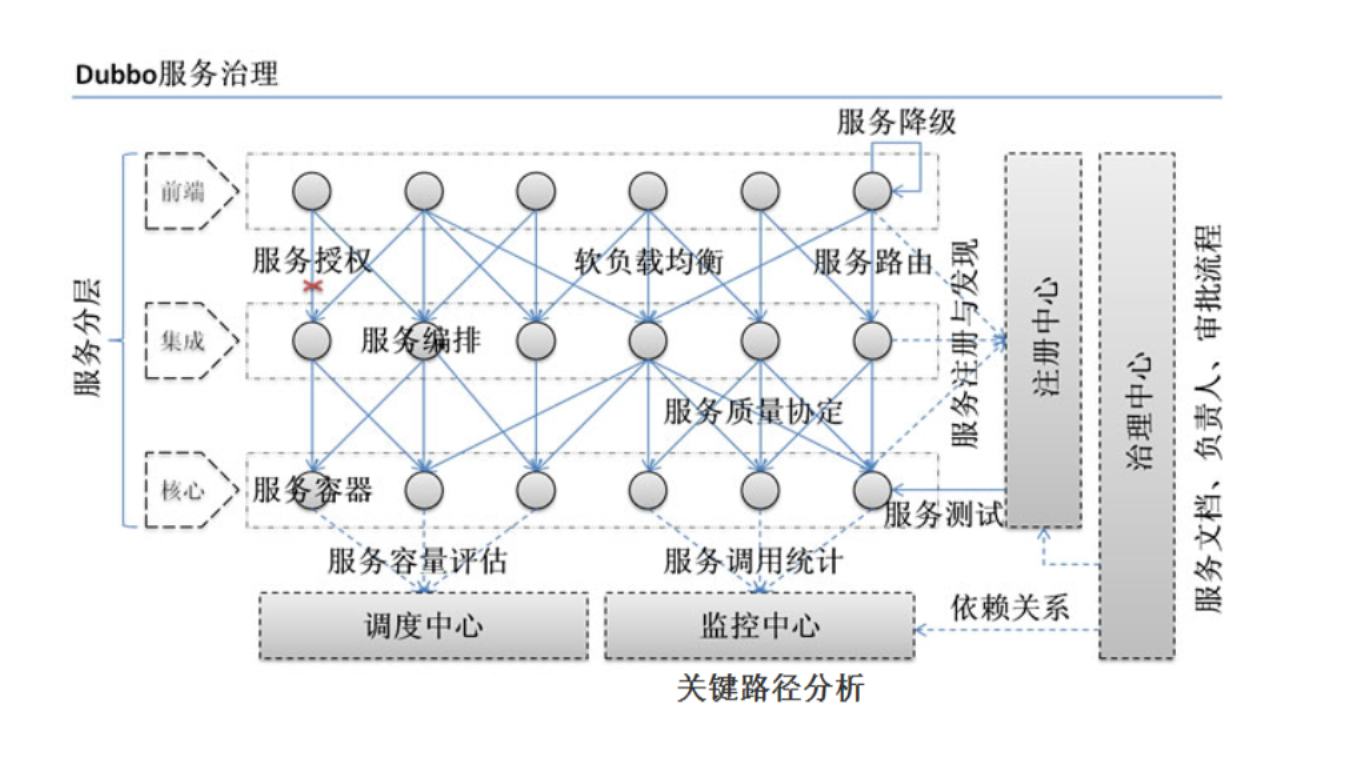
4.4系统技术要求

4.4.1 微服务架构

在本项目中，我们需要建立云管理支撑平台的微服务架构体系规范，其中重点需要解决以下问题：

* 服务配置
* 服务之间的负载均衡
* 服务监控
* API网关
* 服务鉴权
* 服务运行参数管理

为了解决上述问题，需要对服务进行治理，经过调研，选型采用Dubbo+Zookeeper的服务治理框架。



1. **服务配置中心**

**服务治理领域最重要的问题就是服务发现与注册。引入注册中心的概念，解决服务的注册与发现的问题。**

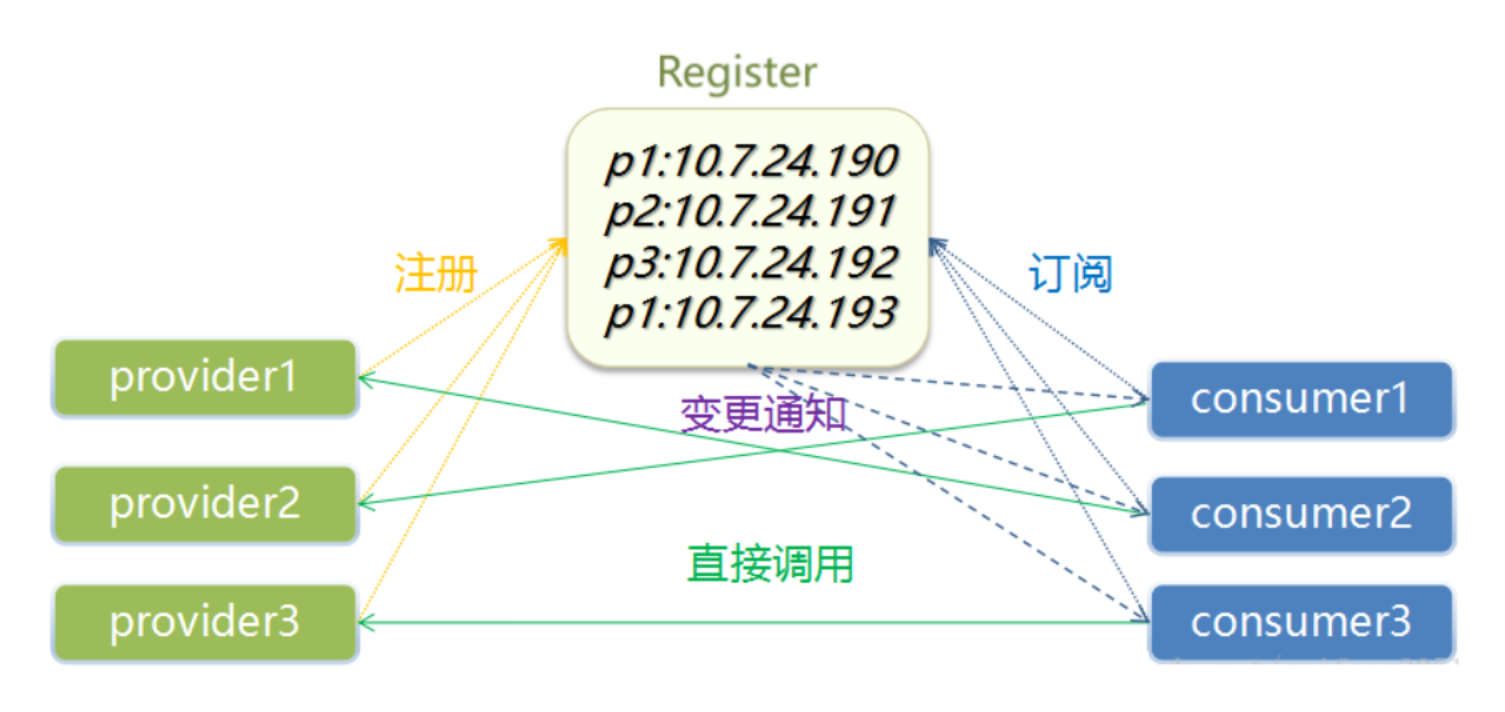
**服务提供者启动，向注册中心注册自己提供的服务**

**消费者启动，向注册中心订阅自己需要的服务**

**注册中心返回服务提供者的列表给消费者**

**消费者从服务提供者列表中，按照软负载均衡算法，选择一台发起请求**

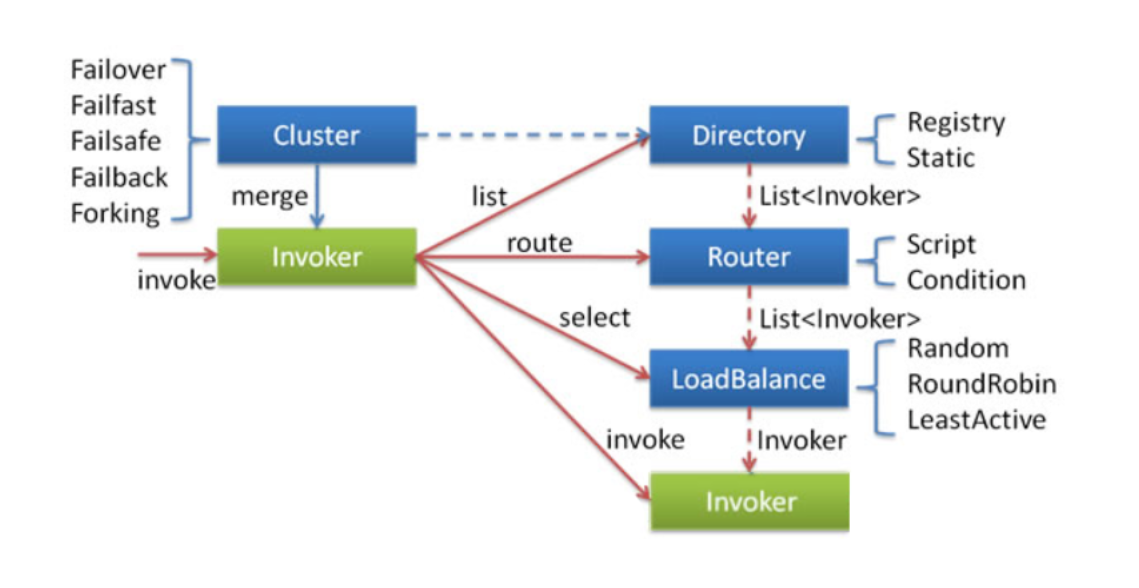
**服务注册与订阅结构如下图：**

****

**（2） 微服务架构水平扩展与负载均衡**

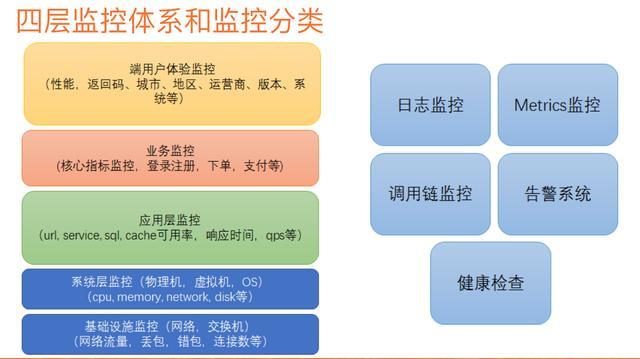
**微服务架构在设计初期应充分考虑架构的水平扩展性和压力逐步增高后的负载均衡需求。**

**基于Dubbo+Zookeeper建立集群容错体系**

****

1. **服务监控**

监控是微服务治理的重要环节。一般分为以下四层



我们提供对于项目整体资源监控功能，随时监控各个应用资源使用情况。主要监控点包括：

* RPC事务：RPC响应时间分析，RPC吞吐量统计，RPC慢事务追踪。
* 数据库：调用数据库延迟分析，数据库吞吐量统计，慢查询慢SQL统计。
* 缓存：访问redis延迟和redis吞吐量统计分析。
* 访问域名：域名维度前端监控，包含白屏时间，首屏时间，下载时间，渲染时间时间。
* 访问页面：URI维度前端监控，白屏时间，首屏时间，下载时间，渲染时间。
* 定位分析：从页面下载时间，首屏时间，白屏时间，页面渲染时间等维度定位伪问题。
* Ajax 接口分析：从延迟和吞吐量维度分析ajax接口性能。
* 脚本错误：统计js执行错误信息以及出错页面。
* 浏览器性能：从流量类型维度统计分析延迟和吞吐量。
* APP性能：监控移动终端的APP性能，包括请求响应时间、不同设备的加载性能。
* 摘要：HTTP流程维度分析前端性能，包括DNS解析，建连，下载，渲染等过程。
* 地理：在地理位置维度分析页面延迟，吞吐量，统计各个地区的性能。
* 链路追踪：通过模拟用户发送的 HTTP 请求，从 servlet 接收到 request 到返回 response 过程中，对数据库、缓存等中间件等调用都进行全链路记录并在展示。
* 错误分析：记录应用中的每一次 error 级别日志打印时的异常。
* 主动监控：实时检测URL在线状态及性能。

监控的内容分为五个大类：日志监控，Metrics监控（服务调用情况），调用链监控，告警系统和健康检查。

日志监控，使用ELK+KAFKA来实现。

健康检查和Metrics，使用spring boot框架实现。

另外我们还会使用Dubbo Monitor来做服务性能监控，依赖Zabbix实现告警系统

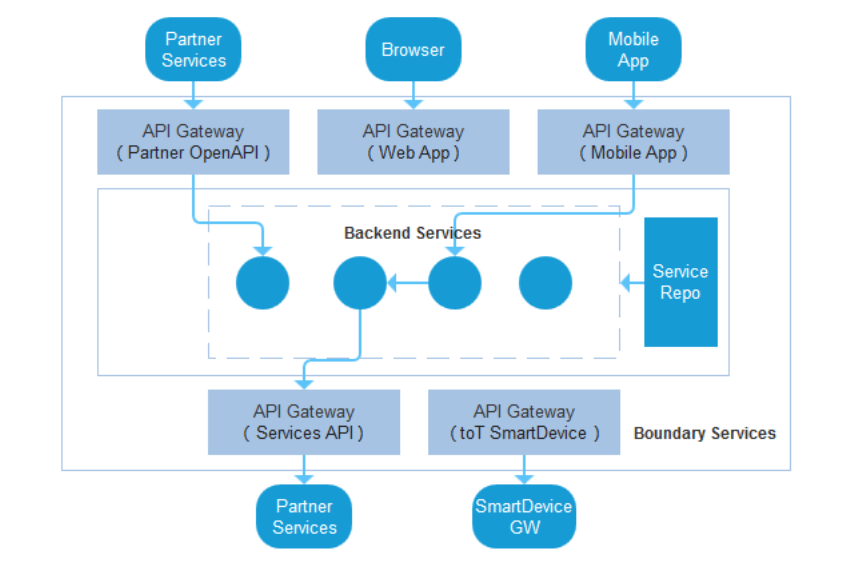
并将监控情况作为基础数据源，可以提供数据分析支持，对接桌面分析、移动分析、数据大屏进行展现

**（4）API网关**

API 网关，是保护内部服务而设计的一道屏障，可以提供高性能、高可用的 API托管服务，从而帮助应用服务的开发人员便捷地对外提供服务，而不用考虑安全控制、流量控制、审计日志等问题，统一在网关层将安全认证，流量控制，审计日志，黑白名单等实现。网关的下一层，是内部服务，内部服务只需开发和关注具体业务相关的实现。网关可以提供API发布、管理、维护等主要功能。开发人员只需要简单的配置操作即可把自己开发的服务发布出去，同时置于网关的保护之下。

API网关需要关注解决以下问题：

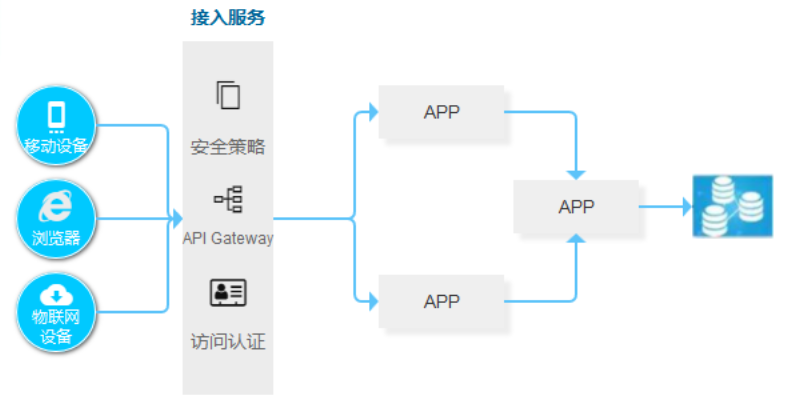
* 网关层对外部和内部进行隔离，保障后台服务的安全性。
* 对外访问控制由网络层面转换成了运维层面，减少变更的流程和错误成本
* 减少客户端与服务的耦合，服务可以独立发展。通过网关层来做映射。
* 通过网关层聚合，减少外部访问的频次，提升访问效率。
* 节约后端服务开发成本，减少上线风险。
* 为服务熔断，灰度发布，线上测试提供简单方案。
* 便于扩展。



在本系统中，通过API网关实现对于API请求进行网关管理，提供API请求网关服务，包含通用网关管理，开放平台，协议转换等。从而解决服务寻址和鉴权问题。

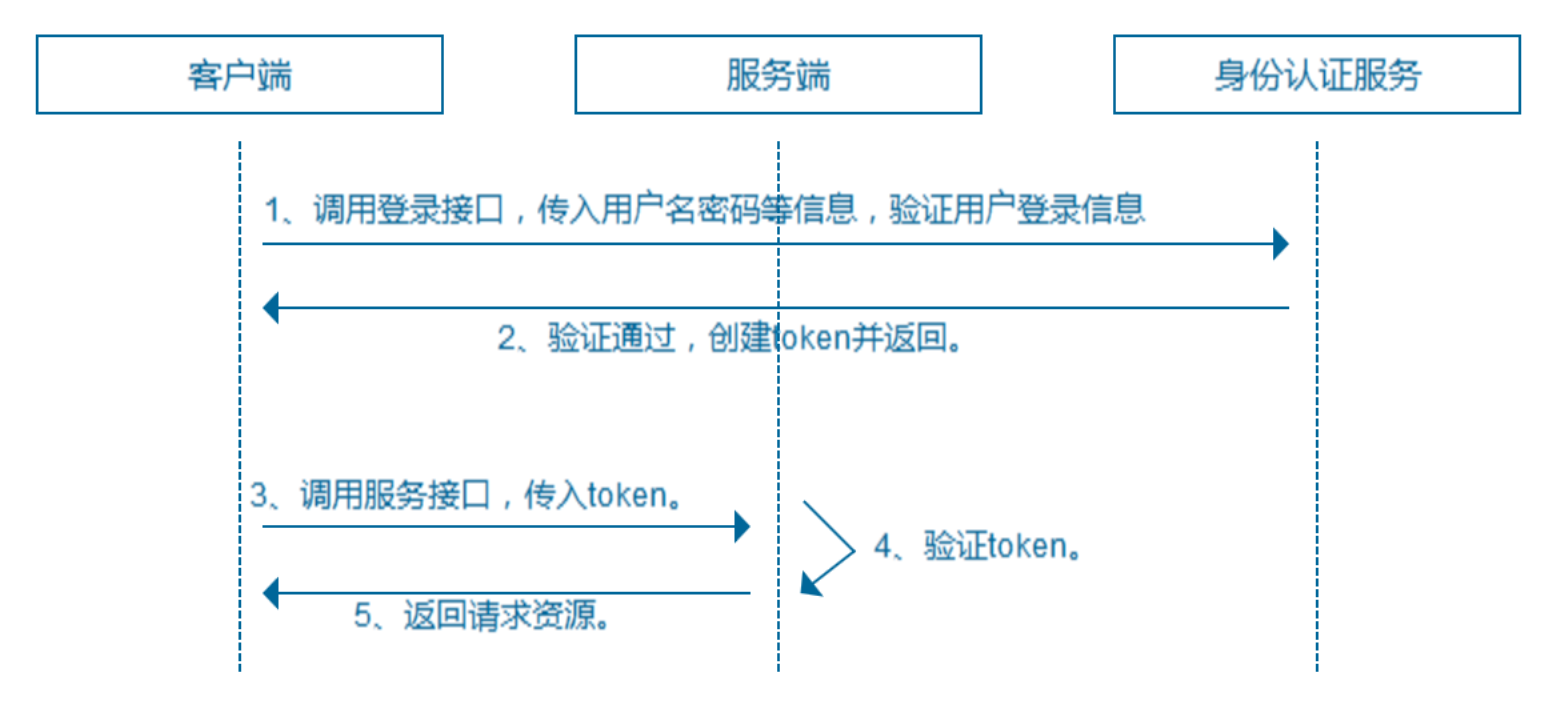
（5）服务鉴权

在微服务架构下，一个应用会被拆分成若干个微应用，每个微应用都需要对访问进行鉴权，每个微应用都需要明确当前访问用户以及其权限。尤其当访问来源不只是浏览器，还包括其他服务的调用时，单体应用架构下的鉴权方式就不是特别合适了。在为服务架构下，要考虑外部应用接入的场景、用户 - 服务的鉴权、服务 - 服务的鉴权等多种鉴权场景。



本系统中我们选型采用基于Oauth2.0结合JWT的鉴权方式

随着 Restful API、微服务的兴起，基于 Token 的认证现在已经越来越普遍。Token 和 Session ID 不同，并非只是一个 key。Token 一般会包含用户的相关信息，通过验证 Token 就可以完成身份校验。



* 用户输入登录信息（或者调用 Token 接口，传入用户信息），发送到身份认证服务进行认证（身份认证服务可以和服务端在一起，也可以分离，看微服务拆分情况了）。
* 身份验证服务验证登录信息是否正确，返回接口（一般接口中会包含用户基础信息、权限范围、有效时间等信息），客户端存储接口，可以存储在 Session 或者数据库中。
* 用户将 Token 放在 HTTP 请求头中，发起相关 API 调用。
* 被调用的微服务，验证 Token 权限。
* 服务端返回相关资源和数据。

**基于 Token 认证的好处如下：**

* 服务端无状态：Token 机制在服务端不需要存储 session 信息，因为 Token 自身包含了所有用户的相关信息。
* 性能较好，因为在验证 Token 时不用再去访问数据库或者远程服务进行权限校验，自然可以提升不少性能。
* 支持移动设备。
* 支持跨程序调用，Cookie 是不允许垮域访问的，而 Token 则不存在这个问题。

**JSON Web Token（JWT）是为了在网络应用环境间传递声明而执行的一种基于 JSON 的开放标准（RFC 7519）。来自 JWT RFC 7519 标准化的摘要说明：JSON Web Token 是一种紧凑的，URL 安全的方式，表示要在双方之间传输的声明。JWT 一般被用来在身份提供者和服务提供者间传递被认证的用户身份信息，以便于从资源服务器获取资源，也可以增加一些额外的其它业务逻辑所必须的声明信息，该 Token 也可直接被用于认证，也可被加密。**

**JWT 是由三段信息构成的，第一段为头部（Header），第二段为载荷（Payload)，第三段为签名（Signature）。每一段内容都是一个 JSON 对象，将每一段 JSON 对象采用 BASE64 编码，将编码后的内容用. 链接一起就构成了 JWT 字符串。**

**OAuth 是一种开放的协议，为桌面程序或者基于 BS 的 web 应用提供了一种简单的，标准的方式去访问需要用户授权的 API 服务。OAUTH 认证授权具有以下特点：**

**简单：不管是 OAuth 服务提供者还是应用开发者，都很容易于理解与使用；**

**安全：没有涉及到用户密钥等信息，更安全更灵活；**

**开放：任何服务提供商都可以实现 OAuth，任何软件开发商都可以使用 OAuth；**

**在本系统微服务架构场景下，我们将 Oauth 和 JWT 结合使用，**

**Oauth 用于第三方接入的场景，管理对外的权限，和 API 网关结合，针对于外部的访问进行鉴权。**

**JWT 更加轻巧，实现微服务之间进行访问鉴权，并且可以避免在流转过程中和身份认证服务打交道。**

**（6）配置中心**

**将系统主要运行变量参数化，建立独立的服务中心进行运行参数配置管理和分发。**

**主要解决以下问题：**

**减少配置参数变化带来的代码变更**

**避免参数调整带来服务重启停机消耗**

**降低各应用模块耦合度**

**对应用整体运行状态进行统一管理**

**通过运行参数分发降低系统频繁加载消耗**

系统中相关参数需要开放配置，可以动态配置项目内所有应用的环境变量，从而进应用运行状态及功能进行线上实时调整。

4.4.2 开发管理体系

**（1）开发管理架构**

**系统中我们采用基于微服务的分布式架构体系来搭建整个系统应用。采用**项目>应用>微服务的系统管理体系。

微服务也是一种服务化，不过其和 SOA 架构的服务化概念也是有区别的，可以从以下几个关键字来理解：

松耦合：每个微服务内部都可以使用 DDD（领域驱动设计）的思想进行设计领域模型，服务间尽量减少同步的调用，多使用消息的方式让服务间的领域事件来进行解耦。

轻量级协议：Dubbo 是 SOA 的开源的标准实现之一。微服务更倾向于使用 Restful 风格的 API，轻量级的协议可以很好地支持跨语言开发的服务。

高度自治和持续集成：从底层的角度来说，SOA 更加倾向于基于虚拟机或者服务器的部署，每个应用都部署在不同的机器上。微服务可以很好得和容器技术结合，容器技术比微服务出现得晚，但是容器技术的出现让微服务的实施更加简便，目前 Docker 已经成为很多微服务实践的基础容器。因为容器的特色，所以一台机器上可以部署几十个、几百个不同的微服务。如果某个微服务流量压力比其他微服务大，可以在不增加机器的情况下，在一台机器上多分配一些该微服务的容器实例。同时，因为 Docker 的容器编排社区日渐成熟，类似 Mesos、Kubernetes 及 Docker 官方提供的 Swarm 都可以作为持续集成部署的技术选择。

微服务的分布式不仅仅是容器应用层面的分布式，其为了高度自治，底层的存储体系也应该互相独立，并且也不是所有的微服务都需要持久化的存储服务。

微服务中的分布式场景除了服务本身需要有服务发现、负载均衡，微服务依赖的底层存储也会有分布式的场景：为了高可用性和性能需要处理数据库的复制、分区，并且在存储的分库情况下，微服务需要能保证分布式事务的一致性。

**（2）工单管理**

建立工单管理系统，集中展示微服务应用内待处理问题，包括告警、代码质量问题、自定义任务，主要功能包括

事件管理：事件管理的目标是在约定的时间内快速恢复意外中断的服务。快速响应和多渠道接入能力，确保事件在第一时间接入服务团队；灵活的分类分派机制实现服务资源的最优分配，及时高效的恢复服务。

问题管理：问题管理通过追查事件背后的根本原因，制定合适的解决方案或采取预防措施防止事件再次发生。由事件触发和主动的问题管理，跟踪问题分析过程、确认根本原因、规避措施、变更计划，闭环管理问题的整个生命周期。

变更与发布管理：变更和发布管理的目标是在有效控制风险和成本的同时，按照标准的步骤和方法，高效完成IT系统的变更和发布。

监控管理：监控系统实时获取在线使用的设备和系统的状态。与监控的集成可以实时获取到这些信息，并与 系统 中的信息对比，及时发现偏差并创建工单进行跟踪。当监控系统发出后，可自动生成工单，并分派给相应人员处理，同时修改配置项的状态。

1. **代码仓库**

版本控制工具也就是把团队每个成员修改的代码做一次备份，每提交或签入一次都会有记录，避免重复拷贝，集中管理的特点。这样在开发过程中省去了很多麻烦，团队联网开发，也可以很高效的工作。版本控制工具一般都是管理员的操作会相对麻烦相对难一点，比如把文件放到哪个位置、安装版本控制工具环境、设置账户密码等等一系列操作。用户也有个小麻烦，小编最常遇到的是代码冲突的问题，当两个人同时修改一个文件的时候就会产出该问题，冲突在各个版本控制工具中都有相应的处理方式，及时解决就可以了。

git可以说是世界上最先进的版本控制系统，大多语句的执行为linux语句，也不难怪，，起初他就是为了帮助开发linux开发内核而使用。

git有以下功能：

* 从服务器上克隆完整的Git仓库（包括代码和版本信息）到单机上。
* 在自己的机器上根据不同的开发目的，创建分支，修改代码。
* 在单机上自己创建的分支上提交代码。
* 在单机上合并分支。
* 把服务器上最新版的代码fetch下来，然后跟自己的主分支合并。
* 生成补丁（patch），把补丁提交。
* 一般开发者之间解决冲突的方法，开发者之间可以使用pull 命令解决冲突，解决完冲突之后再提交。

Git主要优势：

* 适合分布式开发，强调个体。
* 公共服务器压力和数据量都不会太大。
* 速度快、灵活。
* 任意两个开发者之间可以很容易的解决冲突。
* 离线工作。

在项目中，我们选用高性能远端 Git 仓库，分布式计算和存储，保护分支权限控制。

**（4）应用开发运维全链路管理（CI/CD）**

提供灵活的流水线构建能力，通过可视化编辑描述构建流程。支持分布式任务实现应用构建、版本管理、应用部署和应用监控。

我们使用Jenkins+Git实现应用的自动化构建和持续集成。

Jenkins 是一个可扩展的持续集成引擎。接下来，我将通过两篇文章来全面介绍jenkins的基本概念，安装、配置、自动构建、监控、部署，以及在生产环境的高级应用。

主要用途：

* 持续、自动地构建/测试软件项目。
* 监控一些定时执行的任务。

Jenkins特性：

* 易于安装-本文提供四种配置方式。
* 易于配置-所有配置都是通过其提供的web界面实现。
* 集成RSS/E-mail通过RSS发布构建结果或当构建完成时通过e-mail通知。
* 生成JUnit/TestNG测试报告。
* 分布式构建支持Jenkins能够让多台计算机一起构建/测试。
* 文件识别:Jenkins能够跟踪哪次构建生成哪些jar，哪次构建使用哪个版本的jar等。
* 插件支持:支持扩展插件，你可以开发适合自己团队使用的工具。
* Jenkins一切配置都可以在web界面上完成。有些配置如MAVEN\_HOME和Email，只需要配置一次，所有的项目就都能用。当然也可以通过修改XML进行配置。
* 支持Maven的模块(Module)，Jenkins对Maven做了优化，因此它能自动识别Module，每个Module可以配置成一个job。相当灵活。
* 测试报告聚合，所有模块的测试报告都被聚合在一起，结果一目了然，使用其他CI，这几乎是件不可能完成的任务。
* 构件指纹(artifact fingerprint)，每次build的结果构件都被很好的自动管理，无需任何配置就可以方便的浏览下载。

1. **容器云平台**

系统依托Docker容器技术，结合DevOps理念以及自动化运维体系，为企业带来了优化的IT架构，灵活、敏捷、自动化的IT流程。以标准化的交付方式，加快新业务的推出时间，及时响应客户的需求。

我们选用阿里云容器服务，管理所有应用实例，支持一键部署、服务编排、实例伸缩、资源超卖、环境隔离、版本回滚、容器日志、容器监控等功能。

容器服务提供高性能可伸缩的容器应用管理服务，支持用 Docker 和 Kubernetes 进行容器化应用的生命周期管理，提供多种应用发布方式和持续交付能力并支持微服务架构。容器服务简化了容器管理集群的搭建工作，整合了阿里云虚拟化、存储、网络和安全能力，打造云端最佳容器运行环境。

主要功能包括：

* 集群管理，灵活的地域和网络环境选择

用户可以根据自己的需求，选择不同的地域创建和删除集群。

可灵活配置专有网络 VPC 环境。

* 多种服务器托管方式

支持授权容器服务创建云服务器加入到指定集群。

支持将已购买的云服务器添加到指定集群。

* 一站式容器生命周期管理

网络：

支持跨宿主机容器间互联，支持高性能 VPC 网络驱动。

存储：

支持数据卷管理，支持 OSSFS、云盘和文件存储（Network Attached Storage，简称 NAS）。

日志：

支持日志自动采集和阿里云日志服务集成。您还可以和第三方开源日志解决方案进行集成。

监控：

支持容器级别和VM级别的监控。您还可以和第三方开源监控解决方案进行集成。

调度：

支持跨可用区高可用和异常节点的reschedule等策略。

路由：

支持4层和7层的请求转发和后端绑定。

权限：

支持集群级别的RAM授权管理。

* Swarm 和 Kubernetes 双支持

Swarm 和 Kubernetes 双支持，无缝将已有系统从线下迁移至云上。

* 阿里云环境特有的增值能力，更好的体验

整合专有网络 VPC，提供安全、高性能、支持混合云的部署方案。

扩展 Compose 模板定义，增强生命周期管理。

整合负载均衡，提供容器的访问能力。

* 高可用调度策略，轻松打通上下游交付流程

支持服务级别的亲和性策略和横向扩展。

支持跨可用区高可用和灾难恢复。

支持集群和应用管理的 OpenAPI，轻松对接持续集成和私有部署系统。

4.4.3 中间件平台

提供系统运行需要的存储、消息、搜索等类型的中间件，支持一键拉起。提供不同规格的中间件，包括单实例版、高可用版。提供MySQL、Redis、Zookeeper、Elasticsearch、RabbitMQ、RocketMQ、Kafka等

Mysql：作为主要关系型数据存储空间，选用阿里云RDS进行统一数据存储和管理。

Redis：系统中的缓存服务，提供高热度数据缓存功能，提升系统性能和响应效率，另外也可提供分布式锁服务，保证分布式应用中的高并发场景下的数据可靠性。

Zookeeper: 在系统中作为服务注册中心，配合Dubbo作为微服务架构的中心治理组件，管理系统中各个服务注册、上下线、运行状态监控等

Elasticsearch：具有分布式多用户能力的全文搜索引擎，作为支撑新闻资讯搜索功能的功能组件，提升系统全文检索效率。

MQ：消息队列，通过发布、订阅的方式，将系统相关逻辑解耦，一方面让各服务之间依赖性降低，另一方面将部分处理转为异步操作，进一步缩短客户操作的响应时间，另外还可以在瞬时高PV场景中对请求进行削峰。

4.4系统功能设计

4.4.1 基础组件

云服务管理支撑平台，建立统一组件中心，将数据统一并提供公开服务，供其他周边相关系统调用，具体组件包括:

**（1）用户管理组件**

建立统一用户中心，提供统一用户身份识别及用户管理相关功能。

主要包括：

用户基本信息库：建立常熟农业基础用户信息库，收集用户身份，人脸特征，相关角色等信息。

用户身份识别: 提供统一用户身份识别功能，可以通过用户密码、市民卡读卡，人脸识别等多重途径获取用户身份信息。

用户信息维护更新：当用户数据发生变化时，提供实时和定时更新功能。

1. **统一认证组件（单点登录）**

**建立基于Oauth2.0+JWT的统一身份认证体系和单点登录功能。**

**所有系统中的用户、角色、权限信息可以由组件进行统一管理和维护，面向全平台所有**桌面端和移动端应用，面向所有用户，给用户提供全局统一的用户身份账户。

用户可以在系统中进行登录后，系统会通过统一权限管理，获取用户所有有权限访问的应用，用户可以在应用中任意切换，不用重新登录。

同时，提供第三方系统智慧农业账号认证接口，实现访问已实现对接的第三方系统时，可提供智慧农业用户体系的登录方式

**（3）应用管理组件**

应用管理主要对智慧农业云服务平台各类应用进行管理，主要功能包括：

应用注册：各类应用可以在平台中进行注册登记，方便应用管理

应用查询：在系统中查询所有已注册应用，查看应用当前状态和运行情况

应用注销：应用下线时，需要通知注册中心，进行应用注销操作，将应用可靠下线

安全监控：提供统一日志服务，记录各个应用的运行状态，异常信息，用户访问记录等，另外提供应用异常信息监控与上报功能

**（4）权限管理组件**

结合单点登录的用户体系，构建全面的权限管理组件。针对全平台的各个应用和服务，进行统一的用户身份认证和访问控制。主要包括

用户注册：对接统一用户中心的用户维护组件，实现用户前端注册功能

用户组管理：可以对用户设置多个相关用户组，每个用户组都对应一套权限集合，用户组用户可以进行相应授权操作

权限维护：维护各系统的权限点信息，可以细化到具体操作按钮和请求。

权限分配：对用户组进行权限设置

访问控制：根据用户所属用户组，对用户的请求进行统一访问控制，拦截用户未授权操作，可以细化到具体操作按钮和请求。

**（5）消息服务组件**

消息服务组件主要提供了，系统到用户以及用户到用户的多重信息互通方式，包括系统消息服务（IM）和短信消息服务（SMS）。

系统消息服务：主要提供线上系统内的实时消息服务，实现系统给用户的消息推送（跨平台），以及用户与用户之间的实时聊天功能

短信消息服务：由系统推送给用户的离线消息服务，主要传输途径为手机短信（SMS），用户也可以通过短信上行，来对系统进行相应和操作。

**（6）安全监控组件**

安全监控是指出于安全考虑，对全平台所有应用和服务等相关内容进行统一监控，主要包括：

日志管理：建立统一日志中心，收集各系统的操作访问日志、异常操作日志、系统异常信息等，对日志的统一管理，为今后对系统运行状态、用户操作轨迹、用户操作习惯等打好数据基础。

访问痕迹：对用户操作访问痕迹进行记录，通过用户操作埋点，记录用户访问信息，获取请求数据，一方面为系统安全，问题排查提供数据支持，另一方面也为今后系统升级更新提供方案建议。

性能监控：对各个应用及服务节点进行运行状态监控，包括请求的响应情况以及物理机的运行状态等。

身份认证与权限控制：结合单点登录的用户体系，构建全面的权限管理系统。针对全平台的各个应用和服务，进行统一的身份认证和权限控制。

数据加密：包括数据的传输加密和保存加密，系统统一采用HTTPS的加密协议进行数据传输，另外对于密级较高的数据，包括用户密码等在存储前都要提前进行加密处理，不允许明码储存

4.4.2 API服务管理

1. **服务管理平台**

通过服务管理平台，对接装入平台的服务、API进行全生命周期管理，主要功能包括

服务注册管理：构建服务自动注册与发现平台，当服务上线后，自动请求管理平台进行注册，管理平台也可以主动扫描，发现线上离散服务，并对其进行注册管理。

服务授权管理：对服务的访问权限进行控制，分为外部访问和服务内部访问两套授权管理体系。

服务目录管理：系统中，总会存在多个服务节点同时工作，并且节点数量会随着网络规模的变化而动态增减，服务节点也有可能发生宕机与恢复。面对着动态增减的服务节点，我们通过临时节点创建与自动删除来掌握服务节点的动态增减

服务运行监控：服务管理平台，通过心跳轮询方式，对各服务进行运行状态监控，对不健康的服务进行报警及请求流量控制，对宕机的服务，进行强制下线处理。

1. **人脸认证**

提供人脸认证API，通过认证组件功能实现人脸识别，供相关业务应用系统直接调用，实现人脸认证功能

1. **市民卡验证**

与常熟市市民卡系统进行对接，封装市民卡统一验证API，支持业务应用系统直接调用API实现市民卡验证的功能。

4.4.3 物联网管理平台

物联网设备对接，主要是通云服务支撑管理平台中的物联网管理平台进行统一的设备管理，包括：

1. 物联网网关管理：**连接感知网络与传统通信网络的纽带。作为网关设备，物联网网关可以实现感知网络与通信网络，以及不同类型感知网络之间的协议转换．既可以实现广域互联．也可以实现局域互联。此外物联网网关还需要具备设备管理功能，运营商通过物联网网关设备可以管理底层的各感知节点，了解各节点的相关信息，并实现远程控制。具备广泛接入能力、可管理能力和协议转换能力。**
2. **连接管理：对物联网中上网及连接的设备进行管理，包括网络资源管理，Sim卡管控，连接管理，设备管理，故障管理，网络资源用量管理，资费管理，套餐管理，账单管理，服务托管等功能。**
3. **运行支撑管理：实现设备运行整体监视，实际运转结果，设备信号监控，设备信号履历等监控控制管理功能。**
4. **安全管理：由于物联网设备会彻底跳过防火墙建立与第三方服务的长期连接，所以在建设过程中，我们考虑那些设备拥有的网络访问的层级，有多少数据进行传输，设备信息安全方面的成熟度等方面，通过建立加密算法引擎、固件保护认证、物联网设备安全、上传设备信号轨迹等方面集合实现对于物联网安全体系管理。**

需要接入的物联网数据包括：

1. **农机北斗设备**

农机北斗定位设备安装在多辆农机设备上，当前通过农机调度系统进行管理和应用，在本项目中心需要通过物联网管理平台对农机北斗设备进行管理。

1. **传感器**

传感器安装在江苏常熟国家农业科技园区管理委员会和相关湿地的检测站中，需要通过物联网管理平台对相关传感器进行管理。

1. **农业监控摄像头**

农业监控摄像头安装在江苏常熟国家农业科技园区管理委员会等相关单位，需要通过物联网管理平台对相关监控摄像头进行管理。

1. **常熟市民卡读卡器**

常熟市民卡读卡器部署在常熟90家左右的农资店，需要通过物联网管理平台对常熟市民卡读卡器进行管理。

1. **人脸识别设备**

人脸识别设备部署在常熟90家左右的农资店，需要通过物联网管理平台对人脸识别设备进行管理。

1. **农药指导录音设备**

农药指导录音设备部署在常熟90家左右的农资店，需要通过物联网管理平台对农药指导录音设备进行管理。

4.4.4 系统桌面端

1. **系统登录**

支持账号密码登录、二维码登录、短信验证码登录

1. **统一系统框架**

基于Web基础技术等的桌面端框架，需要同时支持IE内核和Chrome内核的浏览器，提供桌面端的底层框架、用户体验规范等，用于支撑搭载桌面业务性功能应用，为新建应用系统提供统一的用户体系、统一的基础API、统一的开发标准、统一的用户体验标准、统一的安全策略等

1. **基础门户**

实现接入系统的单点登录，需要接入的系统包括农业数据资源中心中的数据管理平台、云服务管理支撑平台中的基础组件和API服务管理平台（API网关）及物联网管理平台、农业农村局项目管理系统、以及农业综合统计信息系统等四个新建设的系统

1. **信息系统接入**

接入50个左右业务条线的信息系统，这50个左右系统不实现单点登录，只需要提供系统链接地址导航和备忘录功能。需要提供用户自定义系统分类导航，并为每个系统提供备忘录管理功能，用户可以编辑备忘录的内容，打开备忘录时需要进行身份验证，确保安全性

4.4.5 系统移动端

1. **系统登录**

支持账号密码登录、二维码登录、短信验证码登录

1. **统一系统框架**

基于H5技术进行APP开发，包括iOS和Android，支撑搭载移动端业务性功能应用，为应用提供统一的用户体系、统一的基础API、统一的开发标准、统一的用户体验标准、统一的安全策略等

1. **基础门户**

提供移动端的基础门户，实现接入系统的单点登录，需要接入的系统包括农业综合统计信息系统等四个当前需要建设的系统，提供农业数据资源中心的数据管理平台的移动端查询入口

1. **农药管理平台**

实现农药购买消息推送，查询农药购买记录查询和农药购买金额统计

4.5系统框架设计

云服务管理支撑平台基于分布式微服务框架构建，提供统一的运行环境，支撑智慧农业所有应用与服务，及后续拓展。云服务管理支撑平台需要提供统一的桌面端和移动端框架，统一的管理标准、统一的数据标准、统一的技术标准、统一的用户体验标准、统一的信息安全标准，统一的开发标准等，实现统一的用户体系，统一的API管理，统一的应用管理，统一的界面交互设计等全局性标准规范，确保智慧农业的高拓展性、高效运维和持续运营。

4.5.1项目总体架构

根据农业智能云应用服务平台蓝图业务功能需求，并充分考虑农业智能云应用服务平台运营所带来的高稳定高可用、业务需求快速响应、大数据分析支持、灵活可扩展、全方位安全保证的技术能力要求，以微服务及共享服务中心为核心的架构模式， 农业智能云应用服务平台体系系统蓝图如下：



4.5.2接入层

对接业务运营蓝图中的新体验新渠道的建设需求，提供了基础组件、桌面应用云、移动应用云、单点登录、服务管理平台、物联网管理平台、渠道支撑平台、数据管理平台多种接入手段，数据分析应用提供了桌面分析、移动分析、数据大屏等方式。

4.5.3应用服务层

1. 云服务管理支撑平台在农业资源数据中心的基础上，通过与五大库、公共信息服务平台、市民卡系统、其他部门数据对接，向外部系统获取或者提供数据，通过获取的数据对本地数据进行比对、更新；通过提供数据，支持外部系统运行和数据更新。
2. 物联网设备对接，主要是通云服务支撑管理平台中的物联网管理平台进行统一的设备管理，包括物联网网关管理、连接管理、运行支撑管理、安全管理等。

4.5.4 Paas层

包括两类服务，中间件服务和数据服务：

1. 中间件服务，基于微服务架构理念，集成信息系统交互过程中的公共能力，以全面覆盖微服务、消息、事务的中间件，支持各应用或微服务之间的连接和交互，以应对不确定的业务需求、海量并发的用户访问。
2. 数据服务，基于农业数据资源中心提供的分布式计算与存储服务体系构建，以分布式数据节点为上层应用提供云化的数据服务，包括关系型数据库服务、NoSql数据库服务、数据管理服务等。

4.5.5 Laas层

基础资源层，包括计算服务、存储服务、网络服务、安全服务、运维服务等组件，基于虚拟化等云计算技术，将普通基础资源整合在一起，以集群的方式给提供基础设施服务，从云平台安全、云产品安全和云租户安全三个维度提供多层面一体化的安全防护服务。

1. 其他说明

# 项目实施及售后服务方案

主要内容应包括但不限于以下方面：

1. 项目实施管理方案说明

1.1项目实施总流程

项目实施除了前期准备外（确定项目计划、项目组织等），包括需求分析、信息调研、系统设计、开发及系统测试和UAT、系统部署、推广主演阶段。这些阶段结束，系统进入项目验收前的运行维护阶段。

1.2项目实施准备阶段

项目实施准备阶段的主要任务包括：

1. 确定项目计划。
2. 确定双方的项目经理人选和项目组织架构。
3. 确定项目开发场地及项目实施所需要的基础设施。
4. 准备项目启动会所需要的事宜。
5. 召开项目启动会议。
6. 需求分析阶段 ：根据项目招标文件，对需求内容进行分析、确认、整理，并形成正式的业务需求列表，由相关人员确认。
7. 信息调研阶段 ：信息调研阶段根据需求分析的结果，对可能涉及到的数据源进行调研，以确定数据对需求的支持程度，了解数据质量、数据源系统抽取数据的压力等情况，并据此编制业务需求功能规格说明书。该阶段主要工作包括：

* 根据需求分析结果进行信息调研。
* 根据调研结果编制业务需求功能规格说明书。
* 根据最终确定的功能规格说明书，对项目计划进行调整。根据功能规格说明书的具体情况，与客户交流项目的实施计划后，如果发现原有计划有问题，应要对计划进行变更，变更后的计划应再和客户交流确认。

1. 系统设计阶段：根据前两个阶段的工作成果进行系统设计工作，系统设计阶段的主要工作内容包括:

* 体系架构设计。
* 对系统的设计考虑，包括系统的基本处理流程，系统的组织结构，模块划分，功能分配，接口设计，运行设计，数据结构设计和出错处理设计等。
* 逻辑模型设计与应用设计。

(9)系统设计阶段：根据前两个阶段的工作成果进行系统设计工作，系统设计阶段的主要

开发及系统测试和用户测试阶段。该阶段按照设计阶段的设计文档，开发实现业务需 求功能规格说明书所要求的功能，主要工作内容包括：

* 数据加载、转换、整合、加工程序开发和单元测试，完成接口数据和物理数据库设计的数据对映，并据此设计整个后台的脚本程序，按照设计进行开发；开发汇总表的脚本；开发数据加工程序和脚本；在每一部分开发完成后分别进行各自的单元测试。
* 前端应用功能的开发和单元测试，开发本期业务范围之内各业务系统的业务功能模块。
* 模拟测试，在开发与单元测试的基础上，进行系统集成测试，看各部分功能的配合协调情况，对发现的问题进行纠正。
* 进行用户验收测试和数据核对，验收所开发功能的数量、业务逻辑符合功能规格说明书的要求，确保来自各个业务系统的数据经过整合后，依旧符合业务过程的实际情况，确保基于这些基础数据进行统计分析结果的准确性。

1.3项目投产阶段

系统实施部署阶段：系统部署阶段对系统进行投产部署，部署工作从收到经过几点测试、及网络环境测试后的可开展应用部署的设备及环境开始，主要包括如下内容：

1. 系统环境部署，包括本系统相关的系统软件安装、配置，网络环境和本系统应用相关的配置，与本系统应用相关目录、空间等的分配。
2. 系统应用部署，包括本系统的数据库、表等环境建立和初始化，系统前后台程序、脚本的安装部署，所需基础配置数据的初始化等。
3. 数据部署，进行本系统相关的初始数据加载和日常数据追加，确保初始数据进入系统，并在系统交付后可以使用。
4. 对前后台进行交付前的联调测试、调优，进行前后台联动测试，并根据数据加载情况，对后台数据处理和前端应用模块进行相应的性能测试、调优（包括功能、性能、压力测试等）。
5. 对用户进行培训，使其掌握系统的操作应用，能够在上线后开始使用系统对内、对外提供服务，或者与项目组一道进行系统运行维护，在工作中进行知识转移。 系统推广阶段 。
6. 进行推广培训，按照本系统的功能，编写相应的培训材料，进行培训，并根据各地在使用中的情况进行系统使用中的支持。

1.4项目验收阶段

完成部署和投产后，整个项目进入验收前的运行维护阶段，但项目管理小组和项目组的主要人员，还需要负责完成该项目验收工作。

验收工作包括：

1. 验收申请。项目经理根据合同规定的验收时间和验收条款，向客户提出验收申请报告，提交相关的验收资料，并和客户交流确认。
2. 明确验收流程和人员。在客户同意验收申请后，项目经理应和客户共同组织验收的人员，并就验收的内容和验收的成员进行沟通交流。
3. 项目验收。项目经理和客户的项目负责人一起组织验收人员，对项目提交物、系统运行情况进行验证和确认，根据合同的相应条款对系统进行验收。
4. 签署验收报告。双方验收人员根据系统验收结果，编写系统验收报告，并把系统验收报告提交客户方签字盖章。
5. 系统移交。验收完成后应把系统的设备、系统运行环境及用户和密码、项目相关的基础设施、项目全部交付物、系统运行维护期间的工作记录和文档、维护指导书等资料，向客户方指定的人员进行移交，并和客户方相关负责人进行交接。
6. 验收阶段接收后，系统进入正常运行阶段。我方将按照合同条款规定，对系统进入验收后的支持服务阶段。对于验收中存在遗留问题的，我方将继续组织人员对验收后遗留的问题进行处理，并在完成后，单独进行和相关负责人的交付，对遗留问题的处理后交付，不再组织另外的验收。

1.5项目组织架构说明

1. 项目的组织由甲乙双方共同组成。甲方和乙方将指派人员执行本项目所需的角色，乙方实际指派到项目中的人员可能充当不同的角色。甲乙双方个指派一名项目经理，双方项目经理共同负责。
2. 整个项目在由双方领导组成的项目领导组领导下进行，由双方的项目经理组成项目管理小组，共同管理项目。
3. 项目管理小组对项目领导组报告，负责项目的计划、项目管理、项目状况跟踪、项目问题的协调解决、项目日常报告等工作。
4. 业务专家组提供在进行系统规划、需求分析、系统设计开发时的业务咨询工作。 质量控制组直接对软件开发项目组负责，对应用程序设计编制及测试过程、内容、结果等各种项目质量要素进行过程控制和阶段复审，具有质量否决权。
5. 设计组由相关的技术专家组成，负责项目所包含的所有系统的需求分析、系统设计、技术管理、组织与项目开发有关的技术培训以及对开发小组的协调等工作。
6. 体系架构和集成组负责整个系统的体系架构设计，并负责进行系统集成，负责协调和解决集成中涉及架构调整等方面的工作。
7. 后台开发组和前台应用组是项目的开发小组，分别负责按照设计文档的要求，完成项目中的后台数据处理、转换及相关程序开发，和前端Java开发及基于BI工具的开发，以及单元测试和集成测试工作。
8. 售后服务实施方案说明

2.1售后与日常维护服务保障措施

为解决客户在系统使用中遇到的问题，保证该系统的正常运转，并以实际需要向客户提供及时有效的服务。我们承诺以健全的售后服务体系、多样的服务方式及丰富的服务内容，对您提供满意的售后服务和技术支持。

我们为客户提供多种形式的售后服务体系，包括线上值守服务、现场支持及紧急服务、定期保养服务、电话技术支持、电子邮件支持等。服务内容包括：网上展示展销虚拟平台的数据维护、服务器安全维护、版本升级服务、服务器与数据库性能维护等。

具体标准如下：

1. 项目完成提交及安装调试，我们承诺本项目平台按时安装至客户指定服务器。
2. 系统平台试运行，在系统平台试运行期间，如果发现由于我们的责任造成任何平台的功能和性能不符合技术规范书的要求，或由于设备故障的发生导致设备停止运行，我们承诺在6小时内进行维护系统，所有费用亦由我们承担，同时相应顺延产品试运行期。

2.2系统验收后的保障

1. 固定专业技术人员负责维护系统。
2. 响应时间，提供7×24小时技术支持，在平台系统使用中不论何种原因造成的故障，我们将在接到客户要求维护的通知后，1小时内响应，2小时内到达现场并提出解决方案，6小时内处理完毕。
3. 日常保障，专项技术服务组将长期负责该项目的日常维护，系统正常运行期间，维护工程师将每周进行一次例行系统检查，每月进行一次例行系统维护，最大程度的避免系统隐患的发生。
4. 紧急支援保障，对于本项目的所有素材文件原始资料在维护期内均开辟专门的服务器进行整理和备份，当由于客户人为操作或其他未知因素产生数据资料丢失时，我们会及时提供备份文件和资料将客户的损失减少到最低。

2.3售后及日常维护服务承诺

为了更好地为客户提供及时、迅速、优质周到的服务，我们特做出以下承诺：

1. 成立专门的项目工作组。
2. 免费维护服务，我们负责完成系统平台的设计和开发，并提供相应的免费维护服务。
3. 技术支持方式包括热线电话支持、电子邮件支持、网站支持等各类支持方式，并承诺保证客户在该阶段掌握平台操作、调试、验收的基本技能，保证系统正常安装、运行，并提交安装、调试、验收实施计划书，并承担但不限于以下服务：
4. 专家跟踪监督服务。
5. 免费维护期内服务内容。
6. 升级服务。
7. 承诺在平台建设完成后，根据实际应用需要进行软件完善及升级工作。
8. 其他说明

# 人员配置情况

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **姓名** | **在本项目中担任的职务** | **年龄** | **学历** | **职业资格、职称** | **从业**  **年限** | **主要资历、经验或承担过的项目** | **备注** |
| 1 |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |

注：1、投标人应将承担本项目投入的有关人员配备情况依次填报在上述表格中。

2、提供所投入人员的身份证、学历证书、资格证书、职称证书、技术能力等相关证明材料。

3、提供所投入人员在投标人单位的近六个月的社保证明。

4、因表述含糊导致的评标风险将由供应商自行承担。

5、如有需要说明的事项，请在备注中列明。

# 业绩一览表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 项目名称 | 合同金额 | 实施时间 | 业主联系人、联系电话 | 备注 |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| … |  |  |  |  |  |

注：1、上述业绩应为投标人2016年1月1日至今过数据资源中心平台、云服务支撑管理平台的类似业绩合同。

2、请投标人提供上述表格中主要业绩的合同（合同首页、标的金额所在页、签字盖章页）等证明材料。

3、如有需要说明的事项，请在备注中列明。

4、我方在此声明，我方所提供的以上资料均真实和准确；我方保证上述信息的真实和准确，并愿意承担因我方就此弄虚作假所引起的一切法律后果。

# 商务条件响应及偏离表

|  |  |  |
| --- | --- | --- |
| **合同条款** | **有无偏离** | **偏离内容及原因** |
| 第一条 定义 |  |  |
| 第二条 组成本合同的文件 |  |  |
| 第四条 权利保证及保密 |  |  |
| 第五条 乙方义务 |  |  |
| 第六条 甲方义务 |  |  |
| 第七条 工期规定 |  |  |
| 第八条 验收和测试 |  |  |
| 第九条 人员变更 |  |  |
| 第十条 培训 |  |  |
| 第十一条 伴随服务 |  |  |
| 第十二条 售后服务和技术支持 |  |  |
| 第十三条 付款 |  |  |
| 第十四条 违约责任 |  |  |
| 第十五条 不可抗力 |  |  |
| 第十六条 合同的补充 |  |  |
| 第十七条 合同的变更、中止 |  |  |
| 第十八条 合同的终止 |  |  |
| 第十九条 合同的转让 |  |  |
| 第二十条 履约保证金 |  |  |
| 第二十一条 争议解决方式 |  |  |
| 第二十二条 合同生效及其他 |  |  |

注：1、投标人应据实、详细填写上述表格，因表述含糊导致的评标风险将由投标人承担。

2、若无偏离，在“有无偏离”栏中填写 “无”；若有偏离在“有无偏离”栏中填写 “有”并在“偏离内容及原因”栏中作出说明，因未标明或表述含糊导致的评审风险将由投标人承担。

3、投标人如不能满足本项目付款方式要求，按无效投标处理。

# 资格证明文件

\*法定代表人授权书（法定代表人签署投标文件的不需提供，格式见附件一）

注：附授权代表居民身份证、法定代表人居民身份证

\*投标人营业执照副本

\*上一年度的财务状况报告（成立不满一年需提供资产负债表）

\*依法缴纳税收和社会保障资金的相关材料

\*书面声明函（（参考格式附件三）

金融机构或中介组织出具的投标人资信证明文件（如果有的话）

投标人认为需要提供的其他证明文件

**其中加“\*”项目若有缺失或无效，将导致其投标无效且不允许在开标后补正。**

## 附件一 法定代表人授权书

江苏杰源工程管理咨询有限公司：

本授权委托书声明：注册于 （投标人地址） 的 （投标人全称） 的 （法定代表人姓名） ，代表本公司授权 （授权代表姓名） 为本公司（工厂）的合法代理人，就贵方组织的智慧农业一期项目（项目编号：JSJY-C2019G001），以本公司（工厂）名义处理一切与之有关的事务。

本授权书于 2019 年 月 日生效，被授权人无转委权。

特此声明。

**投标人公章（CA签章）**：

地 址：

**法定代表人（CA签章）**：

职 务：

授权代表**姓名**：

公民身份号码：

职 务：

## 附件二 中小企业声明函

江苏杰源工程管理咨询有限公司：

本公司郑重声明，根据《政府采购促进中小企业发展暂行办法》（财库[2011]181号）的规定，本公司为\_\_\_\_\_\_（请填写：中型、小微）企业。即，本公司同时满足以下条件：

　1、根据《工业和信息化部、国家统计局、国家发展和改革委员会、财政部关于印发中小企业划型标准规定的通知》（工信部联企业[2011]300号）规定的划分标准，本公司为\_\_\_\_\_\_（请填写：中型、小微）企业。

　2、本公司参加 单位的 项目（项目编号：JSJY-C2019G001）采购活动提供本企业制造的货物，由本企业承担工程、提供服务，或者提供其他\_\_\_\_\_\_（请填写：中型、小微）企业制造的货物。本条所称货物不包括使用大型企业注册商标的货物。

附：（1）提供小微企业承诺函（附件二中小企业声明函）。

（2）提供小微企业名录（http://xwqy.gsxt.gov.cn)查询本公司记录截图。

注：如两者不一致，以县级以上中小企业主管部门的书面证明材料为准；如两者不一致且又未提供主管部门书面证明材料则不享受价格评审优惠。

本公司对上述声明的真实性负责。如有虚假，将依法承担相应责任。

　 企业名称：

日 期：

## 附件三 书面声明函

江苏杰源工程管理咨询有限公司：

本公司愿就由贵公司组织实施的智慧农业一期项目（项目编号：JSJY-C2019G001）招标活动进行投标。并作以下郑重声明：

1、我公司具备履行本项采购合同所必需的设备和专业技术能力，为履行本项采购合同我公司具备如下主要设备和主要专业技术能力：

主要设备有： 。

主要专业技术能力有： 。

2、参加本次政府采购活动前3年内，我单位在经营活动中没有因违法经营受到刑事处罚或者责令停产停业、吊销许可证或者执照、较大数额罚款等行政处罚。

3、本公司所提交的投标文件中所有关于投标资格的文件、证明材料和陈述均是真实的、准确的。

本公司对上述声明的真实性负责。如有虚假，将依法承担相应责任。

企业名称：

日 期：

## 投标人营业执照副本

## 上一年度的财务状况报告（成立不满一年需提供资产负债表）

## 依法缴纳税收和社会保障资金的相关材料