微语协作平台技术手册

技术架构与部署指南

北京微语天下科技有限公司

2025年9月1日

Table of Contents

# 微语协作平台技术手册

## 基于AI全新打造的开源企业级多租户团队协作工具

### 技术架构与部署指南

**🤖 文档说明：本技术手册由AI基于官方文档自动生成，包含技术架构、部署方案和开发指南等内容。**

## 📚 产品概述

**微语**是基于AI全新打造的一款开源企业级多租户团队协作工具，集成企业IM、在线客服、企业知识库、工单系统、AI Agent智能体、工作流、客户之声、呼叫中心、视频客服、开放平台等多种功能于一体。

### 技术特点

* **多租户架构** - 支持SaaS模式部署，租户数据完全隔离
* **模块化设计** - 各功能模块独立，可按需部署
* **微服务架构** - 支持分布式部署和水平扩展
* **开源可定制** - 提供完整源码版本，支持深度定制

## 🏗️ 系统架构

### 分层架构

系统采用经典的分层架构设计：

* **展示层** - Web前端、移动端APP、API接口
* **业务层** - 业务逻辑处理、工作流引擎、AI引擎
* **数据层** - 数据库、缓存、文件存储
* **基础设施层** - 容器化部署、监控、日志

### 模块架构

* **通讯模块** - 企业IM、消息推送、实时通讯
* **客服模块** - 多渠道客服、智能分配、会话管理
* **AI模块** - 大模型集成、智能问答、自动回复
* **知识库模块** - 文档管理、全文检索、权限控制
* **工单模块** - 工单流转、状态跟踪、SLA管理
* **工作流模块** - 流程设计、自动化执行、审批管理

### 技术栈

* **后端技术** - Java Spring Boot、Redis、MySQL、MongoDB
* **前端技术** - React、Vue.js、TypeScript、WebRTC
* **移动端** - Flutter、React Native
* **部署方式** - Docker、Kubernetes、Docker Compose

## 🔧 部署方案

### Docker部署（推荐）

#### 系统要求

* **操作系统** - Linux/macOS/Windows
* **内存** - 最低4GB，推荐8GB以上
* **存储** - 最低20GB，推荐50GB以上
* **CPU** - 最低2核，推荐4核以上

#### 快速启动

# 下载部署包
git clone https://github.com/Bytedesk/bytedesk.git
cd bytedesk

# 启动服务
docker-compose up -d

# 查看服务状态
docker-compose ps

#### 服务端口

* **Web管理后台** - http://localhost:9003
* **H5客服端** - http://localhost:9006
* **API接口** - http://localhost:9003/api

### Kubernetes部署

适用于大规模生产环境：

# 部署到K8s集群
kubectl apply -f k8s/

# 查看部署状态
kubectl get pods -n bytedesk

### 源码部署

#### 后端部署

# 编译后端
cd server
mvn clean package

# 运行服务
java -jar target/bytedesk-server.jar

#### 前端部署

# 编译前端
cd admin
npm install
npm run build

# 部署到Web服务器
cp -r dist/\* /var/www/html/

## ⚙️ 配置说明

### 数据库配置

支持MySQL、PostgreSQL、MongoDB等数据库：

spring:
 datasource:
 url: jdbc:mysql://localhost:3306/bytedesk
 username: root
 password: password

 data:
 mongodb:
 uri: mongodb://localhost:27017/bytedesk

### Redis配置

用于缓存和会话管理：

spring:
 redis:
 host: localhost
 port: 6379
 password: password
 database: 0

### AI模型配置

支持对接多种大语言模型：

ai:
 providers:
 - name: deepseek
 apiKey: your-api-key
 baseUrl: https://api.deepseek.com
 - name: openai
 apiKey: your-api-key
 baseUrl: https://api.openai.com

## 📱 客户端集成

### Web集成

#### React集成

import { BytedeskChat } from '@bytedesk/react';

function App() {
 return (
 <BytedeskChat
 orgUid="your-org-uid"
 workGroupUid="your-workgroup-uid"
 visitor={{
 nickname: "访客",
 avatar: "avatar-url"
 }}
 />
 );
}

#### Vue集成

<template>
 <BytedeskChat
 :orgUid="orgUid"
 :workGroupUid="workGroupUid"
 :visitor="visitor"
 />
</template>

<script>
import { BytedeskChat } from '@bytedesk/vue';

export default {
 components: {
 BytedeskChat
 },
 data() {
 return {
 orgUid: 'your-org-uid',
 workGroupUid: 'your-workgroup-uid',
 visitor: {
 nickname: '访客',
 avatar: 'avatar-url'
 }
 }
 }
}
</script>

### 移动端集成

#### Flutter集成

import 'package:bytedesk\_flutter/bytedesk\_flutter.dart';

// 初始化
BytedeskFlutter.init(
 orgUid: 'your-org-uid',
 appKey: 'your-app-key',
);

// 开始对话
BytedeskFlutter.startChat(
 workGroupUid: 'your-workgroup-uid',
 visitor: {
 'nickname': '访客',
 'avatar': 'avatar-url'
 },
);

#### iOS集成

#import <BytedeskSDK/BytedeskSDK.h>

// 初始化
[[BytedeskAPI sharedInstance] initWithOrgUid:@"your-org-uid"
 appKey:@"your-app-key"];

// 开始对话
[[BytedeskAPI sharedInstance] startChatWithWorkGroupUid:@"your-workgroup-uid"
 visitor:visitor];

#### Android集成

import com.bytedesk.android.BytedeskSDK;

// 初始化
BytedeskSDK.getInstance().init(context, "your-org-uid", "your-app-key");

// 开始对话
BytedeskSDK.getInstance().startChat(context, "your-workgroup-uid", visitor);

## 🔌 API接口

### RESTful API

#### 认证接口

POST /api/v1/auth/login
Content-Type: application/json

{
 "username": "admin",
 "password": "password"
}

#### 消息接口

POST /api/v1/messages/send
Authorization: Bearer {token}
Content-Type: application/json

{
 "type": "text",
 "content": "消息内容",
 "toUid": "接收者ID"
}

#### 工单接口

POST /api/v1/tickets/create
Authorization: Bearer {token}
Content-Type: application/json

{
 "title": "工单标题",
 "content": "工单内容",
 "priority": "high",
 "categoryUid": "分类ID"
}

### WebSocket API

实时消息推送：

const ws = new WebSocket('ws://localhost:9003/websocket');

ws.onopen = function() {
 // 连接建立
 console.log('WebSocket连接已建立');
};

ws.onmessage = function(event) {
 // 接收消息
 const message = JSON.parse(event.data);
 console.log('收到消息:', message);
};

ws.send(JSON.stringify({
 type: 'text',
 content: '发送消息',
 toUid: 'receiver-id'
}));

## 🔒 安全特性

### 数据安全

* **数据加密** - 传输加密（HTTPS/WSS）、存储加密
* **权限控制** - RBAC权限模型、细粒度权限控制
* **数据隔离** - 多租户数据完全隔离
* **审计日志** - 完整的操作日志记录

### 系统安全

* **身份认证** - JWT Token、OAuth2.0、SSO集成
* **访问控制** - IP白名单、API限流、防CSRF
* **漏洞防护** - SQL注入防护、XSS防护
* **合规支持** - 符合GDPR、等保2.0要求

## 📊 性能优化

### 性能指标

* **并发用户** - 支持10万+在线用户
* **消息吞吐** - 每秒10万+消息处理
* **响应时间** - API响应时间<100ms
* **可用性** - 99.9%系统可用性

### 优化策略

* **缓存优化** - Redis缓存、CDN加速
* **数据库优化** - 读写分离、分库分表
* **负载均衡** - Nginx负载均衡、服务集群
* **监控告警** - Prometheus + Grafana监控

## 🛠️ 运维管理

### 监控指标

* **系统监控** - CPU、内存、磁盘、网络
* **应用监控** - 接口性能、错误率、吞吐量
* **业务监控** - 用户活跃度、消息量、服务质量

### 日志管理

* **应用日志** - 业务操作日志、错误日志
* **访问日志** - API访问日志、用户行为日志
* **系统日志** - 系统运行日志、安全日志

### 备份策略

* **数据备份** - 定时全量备份、增量备份
* **文件备份** - 上传文件、配置文件备份
* **容灾恢复** - 异地备份、快速恢复机制

## 📞 技术支持

### 技术文档

* **开发者文档** - API文档、SDK文档
* **部署文档** - 安装部署、配置说明
* **运维文档** - 监控运维、故障排查

### 技术支持服务

* **在线支持** - 技术论坛、在线客服
* **专业服务** - 技术咨询、定制开发
* **培训服务** - 产品培训、技术培训

### 联系方式

* **技术支持邮箱** - support@weiyuai.cn
* **商务合作** - sales@weiyuai.cn
* **微信支持** - 扫码联系，备注：微语

## 📋 版本历史

### v2.0.0 (2025-09-01)

* **新增功能** - AI Agent智能体、工作流引擎
* **性能优化** - 消息处理性能提升50%
* **安全增强** - 增加多因子认证、API签名验证

### v1.5.0 (2025-06-01)

* **新增功能** - 视频客服、呼叫中心
* **界面升级** - 全新管理后台界面
* **集成优化** - 支持更多第三方系统集成

### v1.0.0 (2025-01-01)

* **首次发布** - 基础功能模块
* **多租户支持** - 完整的SaaS架构
* **开源发布** - 社区版本开源

## ⚖️ 许可协议

### 社区版许可

* **开源协议** - Apache 2.0许可证
* **商业使用** - 允许商业使用和修改
* **版权声明** - 需保留原始版权声明

### 商业版许可

* **商业许可** - 专有商业许可证
* **技术支持** - 官方技术支持服务
* **定制开发** - 支持定制开发服务

*本技术手册由AI基于官方文档自动生成，内容准确性以官方文档为准* *手册版本：v2.0.0* *生成时间：2025年9月1日*