苏交科知识大脑KM系统对接方案

1. 需求定位

1.1 需求分析

在保留现有体系架构的前提下，将知识平台的常用标准库对接至可定制、自主可控的毕昇知识库及KnowledgeHub，并将原存储于润申的规范文件迁移至毕昇知识库系统进行实际存储。

2. 对接方案

- 标准库变更由KM系统发起推送，KnowledgeHub接收后记录并更新变更信息，同时负责将变更操作（含新增、替换、删除）同步推送至毕昇知识库。
- 规范文件迁移与同步方案：

① **首次执行完整迁移和同步**：KnowledgeHub从润申的数据库或接口中，批量拉取所有规范文件的元数据。同步下载润申文件对应URL的PDF资源，将文件上传至毕昇知识库系统保存，并生成毕昇平台专属的文件访问URL。

② **后续增量同步更新**：当标准知识库发生变更时，通过Webhook 或其他消息机制触发KnowledgeHub同步变更。若KnowledgeHub侧标准库发生新增或替换，则自动触发润申URL的PDF的下载，并上传至毕昇系统保存，同步更新对应的毕昇知识库 URL。

③ **每日定时维护及二次校验**：每日凌晨，KnowledgeHub启动定时任务，与标准知识库进行规范文件记录信息核对，重点校验 “最后修改时间” 与 “记录条数” 两项关键维度；若发现数据不一致，由KnowledgeHub优先更新自身数据，并同步联动更新毕昇知识库，确保三方数据一致性。

3. API 设计

3.1 基础信息

基础URL: `http://192.168.168.19:8088/sjkapi/knowledgehub`

API版本: v1.0.0

文档版本: 1.3.0

认证方式: 无需认证（AllowAny权限，无认证要求）

数据格式: JSON

字符编码: UTF-8

3.2 创建文档任务

接口: `POST /api/create-document-task`

描述: 创建新的文档同步任务

3.2.1 请求参数

代码块

```
{
    "kind": "create",
    "callback_url": "https://example.com/callback",
    "document": {
        "id": 560,
        "do_date": "2005-01-01",
        "create_time": "2024/10/23 Wednesday 19:57:10",
        "deleted": 0,
        "file_path": "http://192.168.60.205:8080/Bzton/pdfJs/web/viewer.html?file=/file/pdf/std_5366.pdf",
        "fileid": 5366,
        "is_buy": 2,
        "is_file": 1,
        "publish_date": "2004-09-04",
        "std_name": "公路沥青路面施工技术规范",
        "std_no": "JTG F40-2004",
        "std_status": "现行",
        "std_type": 1200,
        "expire_date": None,
        "parent_id": None,
        "children_id": 107625
    }
}
```

3.2.2 参数说明

参数	类型	必填	说明
kind	string	否	操作类型：create/update/delete，默认create
callback_url	string	否	回调URL地址

document	object	是	文档信息

3.2.3 document字段

字段	类型	必填	说明
id	integer	是	文档ID
std_name	string	是	标准名称
std_no	string	是	标准编号
std_type	integer	是	标准类型
file_path	string	否	文件路径
publish_date	string	否	发布日期
do_date	string	否	实施日期
expire_date	string	否	废止日期

3.2.4 响应格式

成功 (201):

代码块

```
{
    "status": 1,
    "message": "文档任务创建成功",
    "data": {
        "job_id": "550e8400-e29b-41d4-a716-446655440000"
    }
}
```

失败 (400):

代码块

```
{
    "status": 0,
```

```
    "message": "数据验证失败",
    "errors": {
        "document": ["该字段是必需的。"]
    }
}
```

3.3 查询任务状态

接口: `GET /api/document-task/{job_id}/`

描述: 根据任务ID查询任务状态

3.3.1 路径参数

参数	类型	必填	说明
job_id	string	是	任务ID（UUID格式）

3.3.2 响应格式

成功 (200):

代码块

```
{
    "status": 1,
    "message": "获取文档任务成功",
    "data": {
        "job_id": "550e8400-e29b-41d4-a716-446655440000",
        "task_status": "pending"
    }
}
```

失败 (404):

代码块

```
{
    "status": 0,
    "message": "未找到 job_id 为 550e8400-e29b-41d4-a716-446655440000 的任务",
```

```
    "errors": {
        "job_id": ["该任务不存在。"]
    }
}
```

3.4 状态码说明

3.4.1 HTTP状态码

状态码	说明
200	请求成功
201	创建成功
400	请求参数错误
404	资源不存在
500	服务器内部错误

3.4.2 业务状态码

状态	说明
pending	未启动
processing	处理中
completed	完成
failed	失败

3.5 回调机制

3.5.1 回调通知

当任务完成时，系统会向指定的 `callback_url` 发送POST请求。

3.5.2 回调数据格式

代码块

```
{
    "job_id": "550e8400-e29b-41d4-a716-446655440000",
    "task_status": "completed",
    "sync_status": "success",
    "sync_time": "2024-01-01T10:02:00Z",
    "error_message": null,
    "external_id": "560",
    "sync_type": "create"
}
```

3.5.3 回调重试机制

重试间隔：10s, 30s, 1min, 5min, 15min（可配置）

最大重试次数：5次（可配置）

重试状态：pending → retrying → success/failed

重试检查：通过 Celery 定时任务自动检查

3.6 错误处理

3.6.1 常见错误

错误信息	原因	解决方案
"document字段必须是字典格式"	document参数类型错误	确保document是对象类型
"该字段是必需的"	缺少必填字段	检查请求参数完整性
"未找到 job_id 为 xxx 的任务"	任务不存在	检查job_id是否正确
"数据验证失败"	参数验证不通过	检查参数格式和内容

3.6.2 错误处理示例

代码块

```python
import requests

def safe_request(url, method='GET', **kwargs):
    """安全的请求方法，包含错误处理"""
    try:
        if method.upper() == 'GET':
            response = requests.get(url, **kwargs)
        else:
            response = requests.post(url, **kwargs)

        response.raise_for_status()
        return response.json()

    except requests.exceptions.Timeout:
        return {'error': '请求超时'}
    except requests.exceptions.ConnectionError:
        return {'error': '连接失败'}
    except requests.exceptions.HTTPError as e:
        return {'error': f'HTTP错误: {e.response.status_code}'}
    except Exception as e:
        return {'error': f'未知错误: {str(e)}'}

# 使用示例
result = safe_request('http://api.example.com/data')
if 'error' in result:
    print(f"请求失败: {result['error']}")
else:
    print(f"请求成功: {result}")
```

3.7 请求示例

3.7.1 Python示例

代码块

```python
import requests
import json

# 创建任务
def create_document_task(document_data, callback_url=None):
    url = "http://192.168.168.19:8088/sjkapi/knowledgehub/api/create-document-task"

    payload = {
```

```
        "kind": "create",
        "callback_url": callback_url,
        "document": document_data
    }

    headers = {
        "Content-Type": "application/json"
    }

    response = requests.post(url, json=payload, headers=headers)
    return response.json()

# 查询状态
def get_task_status(job_id):
    url = f"http://192.168.168.19:8088/sjkapi/knowledgehub/api/document-task/{job_id}/"
    response = requests.get(url)
    return response.json()

# 使用示例
document = {
    "id": 560,
    "std_name": "公路沥青路面施工技术规范",
    "std_no": "JTG F40-2004",
    "std_type": 1200
}

# 创建任务
result = create_document_task(document, "https://example.com/callback")
if result['status'] == 1:
    job_id = result['data']['job_id']
    print(f"任务创建成功: {job_id}")

    # 查询状态
    status_result = get_task_status(job_id)
    if status_result['status'] == 1:
        print(f"任务状态: {status_result['data']['task_status']}")
else:
    print(f"创建失败: {result['message']}")
```

3.7.2 JavaScript示例

代码块

```javascript
// 创建任务
async function createDocumentTask(documentData, callbackUrl) {
    const url = 'http://192.168.168.19:8088/sjkapi/knowledgehub/api/create-document-task';

    const payload = {
        kind: 'create',
        callback_url: callbackUrl,
        document: documentData
    };

    try {
        const response = await fetch(url, {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify(payload)
        });

        const result = await response.json();
        return result;
    } catch (error) {
        console.error('请求失败:', error);
        throw error;
    }
}

// 查询状态
async function getTaskStatus(jobId) {
    const url = `http://192.168.168.19:8088/sjkapi/knowledgehub/api/document-task/${jobId}/`;

    try {
        const response = await fetch(url);
        const result = await response.json();
        return result;
    } catch (error) {
        console.error('请求失败:', error);
        throw error;
    }
}

// 使用示例
const document = {
    id: 560,
    std_name: "公路沥青路面施工技术规范",
```

```
    std_no: "JTG F40-2004",
    std_type: 1200
};

createDocumentTask(document, "https://example.com/callback")
    .then(result => {
        if (result.status === 1) {
            const jobId = result.data.job_id;
            console.log(`任务创建成功: ${jobId}`);

            return getTaskStatus(jobId);
        } else {
            console.log(`创建失败: ${result.message}`);
        }
    })
    .then(statusResult => {
        if (statusResult && statusResult.status === 1) {
            console.log(`任务状态: ${statusResult.data.task_status}`);
        }
    })
    .catch(error => {
        console.error('操作失败:', error);
    });
```

3.7.3 cURL示例

代码块

```
# 创建任务
curl -X POST "http://192.168.168.19:8088/sjkapi/knowledgehub/api/create-document-task" \
  -H "Content-Type: application/json" \
  -d '{
    "kind": "create",
    "callback_url": "https://example.com/callback",
    "document": {
      "id": 560,
      "std_name": "公路沥青路面施工技术规范",
      "std_no": "JTG F40-2004",
      "std_type": 1200
    }
  }'

# 查询状态
```

```
curl -X GET "http://192.168.168.19:8088/sjkapi/knowledgehub/api/document-task/550e8400-e29b-41d4-a716-446655440000/"
```

3.8 常见问题 (FAQ)

Q1: 任务创建后多久开始处理?

A: 任务创建后立即进入 `pending` 状态，处理时间取决于系统负载和任务队列长度。

Q2: 如何处理任务失败?

A: 可以通过查询任务状态获取失败原因，系统会记录详细的错误信息。

Q3: 如何取消正在处理的任务?

A: 当前版本不支持任务取消，任务会继续执行直到完成或失败。

Q4: 回调URL需要什么格式?

A: 回调URL必须是有效的HTTP/HTTPS地址，系统会发送POST请求。

Q5: 任务数据会保存多久?

A: 任务数据默认长期保存，可通过配置设置清理策略。

3.9 限制说明

3.9.1 请求限制

请求超时时间：30秒（回调请求）

数据格式：JSON，建议单次请求数据量不超过1MB

并发处理：受服务器资源限制

3.9.2 任务限制

任务状态：pending → processing → completed/failed

回调重试：最多5次，间隔递增

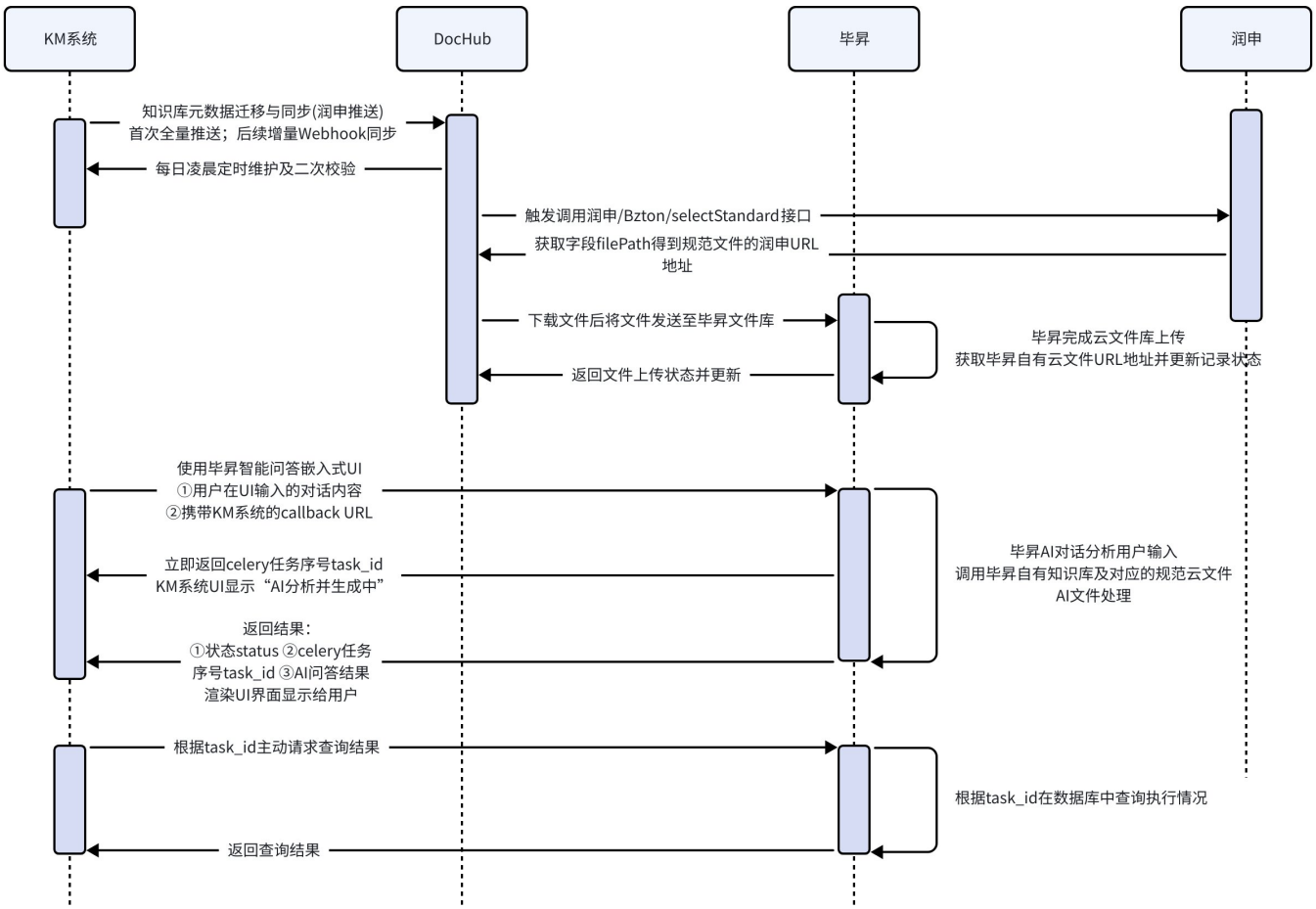
任务保存：默认长期保存（可配置清理策略）

4. 数据模型Models设计

字段名	中文含义	说明
id	规范索引号	主键；毕昇规范文件唯一标识符
standard_id	润申标准编号	润申规范文件唯一标识符
std_name	规范名称	
std_no	规范文件发行编号	如：GB/T 30948-2021 建办城〔2021〕47号
std_type	规范文件类别	1100,1，国家标准 1200,4，行业标准 1300，其他标准 1400，企业标准 1500，团体标准 6，地方标准 2，国外标准 3,法规 5图集, 7 资料
status	状态	可选：Active（现行）、Repealed（废止）、Draft（草案）
publish_date	发布日期	
do_date	实施日期	
expire_date	废止日期	状态为'Repealed'时填写
replaces_standard	替换的标准规范（替换了谁）	可能有多个，以数组形式存储标准编号
replaced_by_standard	被替换的标准规范（被谁替换）	可能有多个，以数组形式存储标准编号
runshen_source_url	润申原始PDF地址	
runshen_source_url_datetime	润申原始PDF地址更新时间	

file_status	毕昇知识库状态	可选：Pending（待更新）、Download（下载中）、Upload（上传中）、Completed（已完成）、Failed（失败）
file_url	毕昇知识库库中的URL地址	
file_url_datetime	毕昇知识库库中的URL地址更新时间	
ai_processing_status	AI处理状态	可选：Pending（待处理）、Processing（处理中）、Completed（已完成）、Failed（失败）
create_time	记录创建时间	
modify_time	记录修改时间	
modify_user	修改人员	
deleted_at	记录软删除标记	

5. 调用逻辑时序图


KM系统
DocHub
毕昇
润申
知识库元数据迁移与同步(润申推送)
首次全量推送；后续增量Webhook同步
每日凌晨定时维护及二次校验
触发调用润申/Bzton/selectStandard接口
获取字段filePath得到规范文件的润申URL
地址
下载文件后将文件发送至毕昇文件库
返回文件上传状态并更新
毕昇完成云文件库上传
获取毕昇自有云文件URL地址并更新记录状态
使用毕昇智能问答嵌入式UI
①用户在UI输入的对话内容
②携带KM系统的callback URL
立即返回celery任务序号task_id
KM系统UI显示“AI分析并生成中”
毕昇AI对话分析用户输入
调用毕昇自有知识库及对应的规范云文件
AI文件处理
返回结果：
①状态status ②celery任务
序号task_id ③AI问答结果
渲染UI界面显示给用户
根据task_id主动请求查询结果
根据task_id在数据库中查询执行情况
返回查询结果