一、背景

> 场景：

随着市场的推广，我们的各类系统会部署到全球各地，用户也来自不同国家、地区。随之而来的，就是系统与用户交互的变更。

因此，我们的系统每扩展一个新的语言区域，就要针对这个语言区域进行国际化开发，并重新发不系统。这样会带来额外的人力成本。

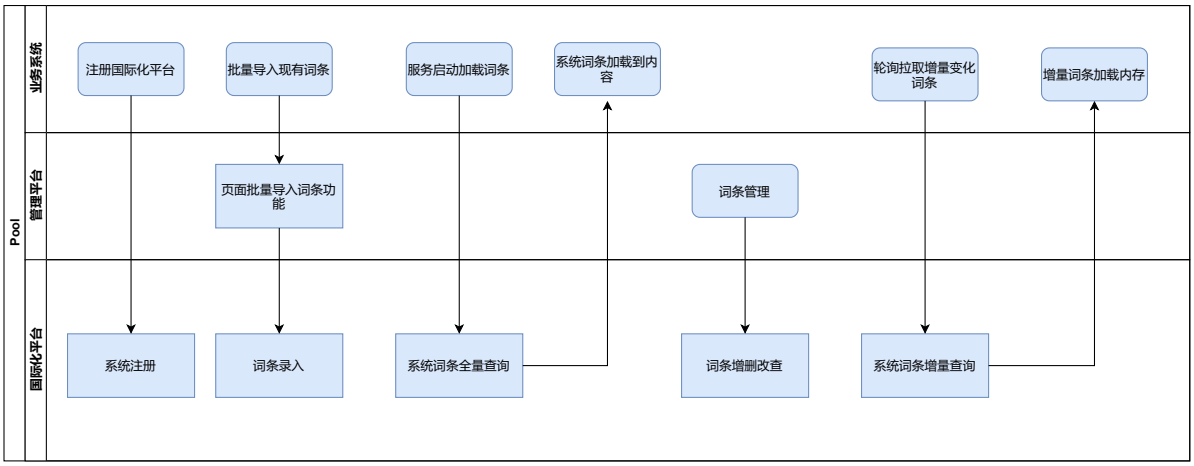
> 价值：

国际化平台系统的设计与实现，就是为了解决此类问题。

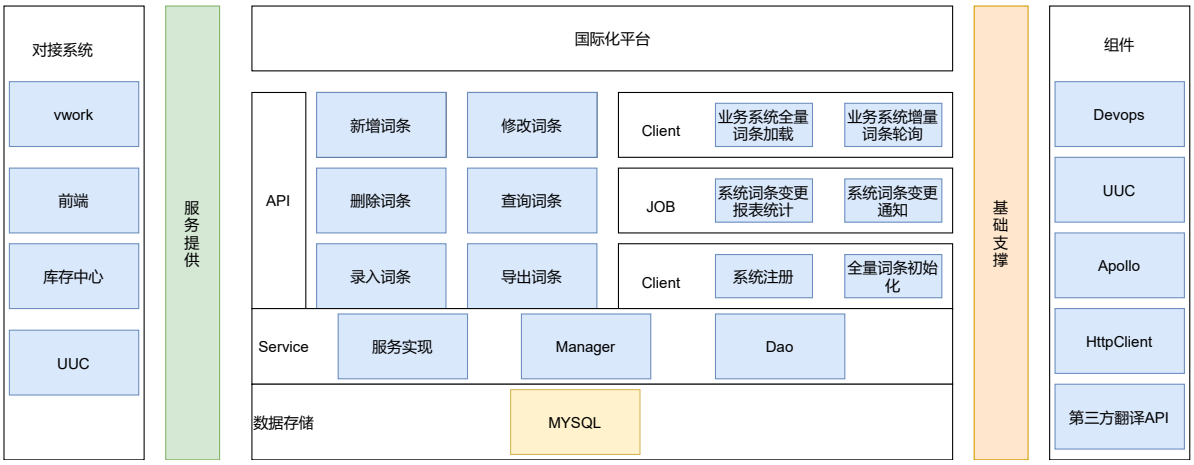
系统的国际化词条统一由平台管控，可以实现灵活的语种增加、自动翻译、实时加载。

且不依赖系统发布。

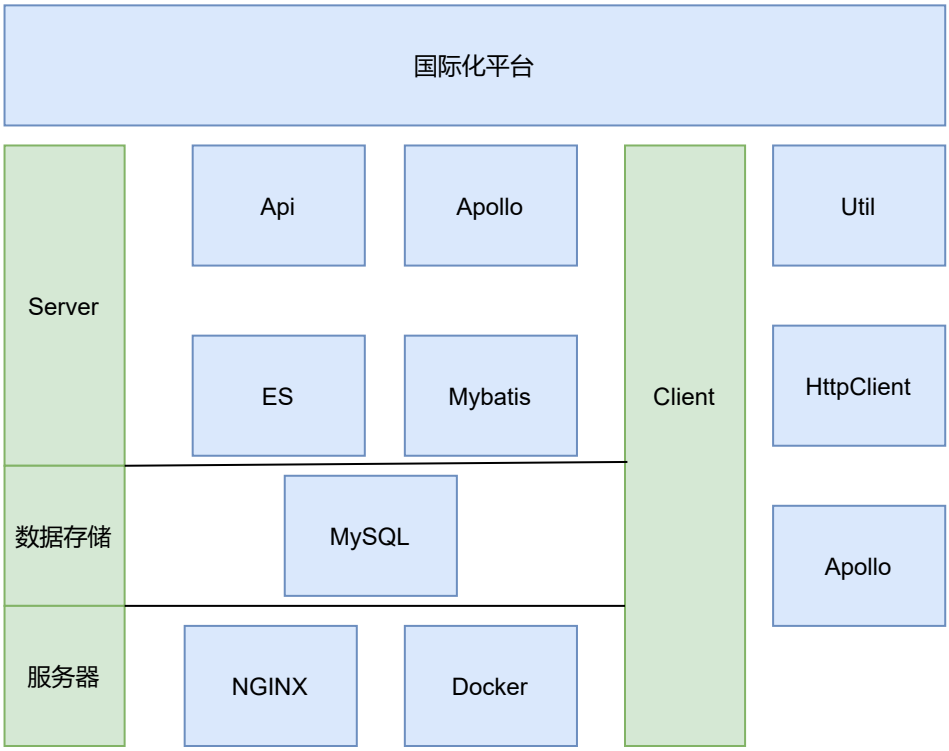
二、方案设计-系统交互



三、方案设计-系统架构



四、方案设计-应用架构



五、方案设计-表结构汇总

序号	表名	中文名	备注
1	t_project_manager	应用工程注册表	
2	t_project	工程信息明细表	
3	t_module	模块明细表	
4	t_international	系统模块支持语言配置表	
5	t_international_code	词条信息主表	
6	t_code_detail	词条信息明细表	
7	t_manual_dictionary	人工翻译字典表	
8	t_job_config	定时任务配置表	
9	t_translation_report	词条变更情况报表	
10	t_language	语言列表汇总	

> t_project_manager

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
project_code	varchar(8)	应用编码	N	注册时系统分配
module_code	varchar(8)	子应用编码	N	注册时系统分配
job_number	varchar(20)	工号	Y	注册时可选输入
project_id	varchar(64)	工程id（devops）	N	注册时输入
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除	N	

> t_project

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
project_name	varchar(64)	应用名称	N	注册时输入
app_id	varchar(64)	应用ID	N	注册时输入
project_code	varchar(8)	应用编码	N	注册时分配
description	varchar(127)	应用描述信息	Y	注册时输入
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除	N	

> t_module

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
app_id	varchar(64)	应用ID	N	注册时输入
module_name	varchar(64)	模块名称	N	注册时输入
module_id	varchar(64)	模块ID	N	注册时输入
module_code	varchar(8)	模块编码	N	注册时分配
description	varchar(127)	应用描述信息	Y	注册时输入
app_key	varchar(100)	app_id加密信息	Y	根据app_id进行md5加密
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除	N	

> t_international

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
app_id	varchar(64)	应用ID	N	注册时输入
project_code	varchar(8)	应用编码	N	注册时分配
project_name	varchar(64)	应用名称	N	注册时输入
module_name	varchar(64)	模块名称	N	注册时输入
module_id	varchar(64)	模块ID	N	注册时输入
module_code	varchar(8)	模块编码	N	注册时分配
support_language	varchar(1024)	支持的语言	N	注册时输入
description	varchar(127)	应用描述信息	Y	注册时输入
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除	N	

> t_international_code

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
scene_code	varchar(8)	词条场景编码	N	
default_description	varchar(512)	词条key	Y	
code	varchar(16)	词条编码	N	
app_id	varchar(64)	应用ID	N	
project_code	varchar(8)	应用编码	N	
module_code	varchar(8)	模块编码	N	
word_character	varchar(32)	词性：名词、动词、复合	Y	
module	varchar(255)	词条归属模块	Y	
usage_scenario	varchar(1024)	词条使用场景	Y	
resource_key	varchar(512)	预留字段	Y	
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除	N	

> t_code_detail

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
code	varchar(16)	词条编码	N	
language_code	varchar(16)	所属语种	N	
description	varchar(4096)	语言对应的描述信息	N	
how_to_translate	varchar(32)	翻译方式：手动（MANUAL），自动（AUTOMATIC）	Y	
app_id	varchar(64)	应用ID	N	
project_code	varchar(8)	应用编码	N	
module_code	varchar(8)	模块编码	N	
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除	N	

> t_manual_dictionary

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
chinese	varchar(1000)	中文	N	
language_code	varchar(16)	所属语种	N	
description	varchar(4096)	中文对应语种的翻译信息	N	
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	

> t_job_config

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
job_name	varchar(128)	任务名称	N	
job_type	tinyint	类型：1-新增词条统计并发送V消息	N	
deal_time	int	任务处理时长	N	
job_param	varchar(1024)	要处理的JOB参数	Y	
report_user	varchar(4096)	报告接受人列表	Y	
report_user_name	varchar(4096)	报告联系人姓名列表	Y	
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
update_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除，0：未删除	N	

> t_translation_report

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
app_id	varchar(64)	应用ID	N	
module_id	varchar(64)	模块ID	N	
report_type	tinyint	报告类型：1-新增词条 2-修改词条	Y	
report_num	int	报告的数量	Y	
start_time	datetime(3)	报告开始时间	Y	
end_time	datetime(3)	报告结束时间	Y	
create_time	datetime	创建时间	Y	
update_time	datetime	更新时间	Y	

> t_language

- index id

column name	data type	comment	nullable	desc
id	bigint	id	N	
language_name	varchar(32)	语言名称	N	
language_code	varchar(16)	语言编码	N	
description	varchar(127)	语言描述	Y	
create_time	datetime	创建时间	Y	
create_user_id	varchar(32)	创建人	Y	
update_time	datetime	更新时间	Y	
pdate_user_id	varchar(32)	更新人	Y	
del_flag	tinyint(1)	是否删除，1：已删除，0：未删除	N	

六、功能范围

- 已实现或正在实现的功能

序号	功能点	类型	详细描述
1	系统注册	http	根据appid和projectId注册到国际化平台
2	增删改查	http	国际化词条增删改查功能
3	词条导入、导出	http	按模板导入、导出词条
4	系统词条拷贝	http	根据不同环境需要差异化管理时，快速复制词条到新系统下
5	客户端全量数据拉取	http	业务系统启动时，全量拉取系统词条
6	客户端增量数据轮询	http	业务系统定时轮询拉取增量变化词条
7	客户端国际化词条Util	Util	根据key+language获取国际化词条的工具类
8	系统词条增量变化通知	JOB	统计一段时间内的系统新增词条信息，并生成词条明细文档，V消息发给系统负责人
9	系统词条变化报表统计	JOB	每天统计系统词条增量变化数据量
10	人工翻译字典数据增删改查	http	翻译组人工翻译的信息录入字典表，翻译时优先取人工翻译

- 规划中的功能

序号	功能点	类型	详细描述
1	新系统接入能力提升	http	支持properties属性文件录入功能
2	自定义翻译优先级	http	1、支持对接谷歌翻译API 2、不同系统支持翻译优先级排序：人工、百度、谷歌

七、外部系统对接

7.1、添加maven依赖

```
<dependency>
    <groupId>xxx</group>
    <artifactId>hydra-spring-boot-starter</artifactId>
    <version>xxx</version>
</dependency>
```

7.2、启动类添加注解

```
@EnableHydra
```

7.3、客户端覆盖Spring默认的MessageResource Bean，使用自定义的Bean

```
@Component("I18nMessageSource")
@ConditionalOnProperty(
         name = {"hydra.internaltional.enabled",
         havingValue = "true"}
)
@Primary
public class I18nMessageSource implements MessageSource {

    public I18nMessageSource() {

    }

    @Override
    public String getMessage(String code, Object[] args, Locale locale) {
        return HydraInternationalUtil.getInternationalCodeInfoByLanuage(code,
local.toLanguageTag());
    }

}
```

八、缓存内存结构体设计

```
// key：词条key
ConcurrentHashMap<String, InternationalCodeInfo> internationalMap = new
ConcurrentHashMap<>();
```

```
public class InternationalCodeInfo implements Serializable {
    private String appId;

    private String code;

    private String key;

    // key：语言环境 value：对应语言的翻译词
    private Map<String, String> languages;
}
```

九、服务启动&增量拉取词条代码伪代码实现

```
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Inherited
@Import({HydraConfiguration.class})
public @interface EnableHydra {

}
```

```
@Configuration
public class HydraConfiguration {

    @Bean
    public HydraDataWrappper hydraDataWrapper() {
        return new HydraDataWrapper();
    }

    @Bean
    public HydraListener hydraListener() {
        return new HydraListener();
    }
}
```

```
@Component
public class HydraDataWrapper {

    @Value("${hydra.app.id:})
    private String appId;

    @Value("hydra.app.meta:")
    private String host;

    @Resource
    private RestTemplate restTemplate;

    @PostConstruce
    private void initData() {
        // ...
        String url = host + "/xxx?appId=" + id;
        ResponseEntity<Map<String, InternationalCodeInfo>> response =
restTemplate.exchange(url, ...);
        // 放入缓存
```

```
        HydraInternationalUtil.putAllInternationalCodeInfo(reponse.getBody);
    }

}
```

```
public class HydraListener implements Runnable {

    @Value("${hydra.app.id:}")
    private String appId;

    @Value("${hydra.app.meta:}")
    private String host;

    // 长轮询连接超时时间
    private static final int LONG_POLLING_CONNECT_TIMEOUT = 5000;
    // 长轮询读取超时时间
    private static final int LONG_POLLING_READ_TIMEOUT = 65000;
    // 线程休眠时间
    private static final int SLEEP_TIME = 60000;

    private RestTemplate restTemplate;

    @PostConstruct
    private void initClient() {
        new RestTemplate(getClientHttpRequestFactory());
        new Thread(this).start();
    }

    @Override
    public void run() {
        while (true) {
            try {
                Thread.sleep(SLEEP_TIME);
                String url = host + "/xxx?appId=" + appId + "&ip=" + getIp();
                ResponseEntity<Map<String, InternationalCodeInfo>> response =
 restTemplate.exchange(url, ...);
                // 放入缓存

 HydraInternationalUtil.putAllInternationalCodeInfo(response.getBody());
            } catch(...) {
                ...
            }
        }
    }

    private String getIp() {
        try {
            InetAddress localHost = Inet4Address.getLocalHost();
            return localHost.getHostAddress();
        } catch (...) {
            ...
        }
    }

    private SimpleClientHttpRequestFactory() {
```

```
        SimpleClientHttpRequestFactory clientHttpRequestFactory = new
SimpleClientHttpRequestFactory();
        clientHttpRequestFactory.setConnectTimeout(LONG_POLLING_CONNECT_TIMEOUT);
        clientHttpRequestFactory.setReadTimeOut(LONG_POLLING_READ_TIMEOUT);
        return clientHttpRequestFactory;
    }

}
```