SUNLIGHTMQ 网关服务

系统接入规范

修订历史

版本号	修改日期	编写	评审	批准	修改内容
1.0.1					文档创建
1.0.2					更新部分内容描述，更新“2.6 编程建议”
1.0.3					更新部分内容描述，更新“2.2 接口定义规范”
1.0.4					更新部分内容描述，更新“2.4 异常响应报文返回情况”
1.0.5					新增应答报文参数返回结果不可以为“null”的相关说明。已加入到“2.1.3 报文样例”和“2.4 异常响应报文返回情况”两个小节
1.0.6					第三章“基本规范”新增“2.4 消息字符编码规范”
1.0.7					第二章“2.5 异常响应报文返回情况”更新，新增异常：响应码 00300 发送消息到 MQ 失败、响应码 00901 消息发送失败
1.0.8					新增“第三章访问方式”
1.0.9					“2.5 异常响应报文返回情况”说明更新： 1.消费者处理消息超时（消息接收器超时），超过 20 秒，可能原因是消费者应用接口异常； 2.生产者等待结果返回超时（消息发送器超时），超过 20 秒，可能原因是消息处理器异常关闭，或消息转发器异常，或网络传输问题；
1.1.0					新增第三章“异常响应”
1.1.1					第三章“异常响应”修改消息处理错误相关异常 0050X： 00501：Get 方式调用接口出错； 00502：调用混合队列出错； 00503：调用混合队列出错； 00504：调用 B2C 业务接口出错； 00505：调用普通业务接口出错； 00506：调用跨中心路由器失败；

版本号	修改日期	编写	评审	批准	修改内容
					00507：调用跨中心路由器出错
1.1.2					更新“3.2 节异常响应报文返回码”的“无调用权限 status=98”相关说明
1.1.3					更新“4.2 二阶段返回方式”相关说明
1.1.4					3.2.3 异常响应返回码 00600 更新：新增流量控制配置异常
1.1.5					文档名称与关键字修改
1.1.6					新增 1.6 消息序列化
1.1.7					3.2.3 异常响应返回码：00600，新增特殊字符异常情况
1.1.8					3.2.3 异常响应返回码：00600，新增消费者线程失效情况及网络配置错误
1.1.9					新增 1.7 消息异常

目录

第一章 基本原则

本规范提供 SUNLIGHTMQ 系统的相关标准 ,为各应用系统如何接入 SUNLIGHTMQ 系统，编写消息接口提供标准和参考依据。

1.1 报文形式

- 报文信息统一为基于 JAX-RS（Java API for RESTful Web Services）标准报文，并使用 JSON 格式作为资源的表述，即：HTTP+URI+JSON；
- 所有系统发布在 SUNLIGHTMQ 的服务采用统一的规范格式；
- 原则上今后开发的系统间的服务遵守本文档规范的报文格式，服务请求方通过 SUNLIGHTMQ 调用服务方在 SUNLIGHTMQ 发布的服务来实现各种功能。

1.2 报文内容

- 所有经过 SUNLIGHTMQ 的报文必须是标准格式报文；
- 请求方请求报文中的 JSON 报文体必须包含 SUNLIGHTMQ 统一指定的请求节点数据 ,其它节点需符合其事前与服务提供方约定的服务请求报文的应用数据报文定义；
- 服务响应报文中的 JSON 报文体必须包含 SUNLIGHTMQ 统一指定的返回节点数据，其它节点需符合其事前与服务请求方约定的服务响应报文的应用数据报文定义。

1.3 服务发布

- 服务提供方应提供服务名、接入秘钥，通过 SUNLIGHTMQ 发布接口；
- 服务请求方根据服务提供方发布的服务名以及服务的接入秘钥来发起服务请求。

1.4 智能路由

- 服务请求方需要将请求报文发送至 SUNLIGHTMQ 指定的服务端口，由 SUNLIGHTMQ 根据服务定义自动实现路由。

1.5 消息超时

- 对于原子服务由服务请求方统一处理超时时间，对于组合服务，由服务请求方与 SUNLIGHTMQ 管理组、服务提供方共同商定其对于每个服务的处理超时时间。

1.6 消息对象

- SUNLIGHTMQ 传输的消息体可以为二进制实体对象。如果消息体为二进制实体对象，则必须对该对象进行序列化。
- 序列化机制保存了实体对象的类型信息及其属性的类型信息和属性值。如果实体对象没有进行序列化或者没有设置 serialVersionUID，在网络中传输到达目标节点后，如果对方接口应用的实体类增加或减少了属性（filed），就有可能造成实体解析错误，报出异常，比如：“exception info：syntax error，expect {”，但如果设置了 serialVersionUID，就会将不一样的属性以缺省值反序列化，这样就可以避免不兼容问题。

1.7 消息异常

- REST 接口发生异常不直接抛出，而是将异常信息作为处理结果返回给消息生产者（接口请求方），以方便 SUNLIGHTMQ 用户调试程序。

第二章 基本规范

2.1 报文规范

报文信息统一为基于 JAX-RS（Java API for RESTful Web Services）标准报文，并使用 JSON 格式作为资源的表述，即：HTTP+URI+JSON；

2.1.1 请求报文规范

SUNLIGHTMQ 请求报文中必需包含下列域：

域名	中文含义	说明	备注
businessName	业务名称	String	有明确含义的名称
appKey	SUNLIGHTMQ 分配给业务系统（消息生产者）的唯一标识	string(16 字符)	如：1000S4AV43RC5PL2
syncFlag	同步标记	String	取值为 1 的时候为同步调用，0 为异步调用
messageJson	消息内容	String	标准 Json 格式字符串，需使用 utf-8 编码格式，所有参数均使用小写 如： messageJson={"name":"test"}

2.1.2 响应报文规范

SUNLIGHTMQ 响应报文中必需包含下列域：

域名	中文含义	说明	备注
status	调用状态	string(20-100 字符)	调用状态码（1 成功，-1 失败）
message	调用结果提示信息	string(0-100 字符)	调用结果提示信息
messageJson	返回的消息体	Json 格式	返回的消息体，所有参数均使用小写

2.1.3 报文样例

- 请求报文

http://60.30.215.42:8080/router?businessName=userregister&syncFlag=1&appKey=1014CBMAGXEEE6H4&messageJson={"name":"test"}

- 应答报文
  - 同步消息应答报文如下：

{"message":"邮箱，手机号，密码和微信 OpenId 都不能为空","messageJson":"","status":"-1"}

{"messageJson":{"balance_day":"40060","deal_response_code":"00","deal_code

```
":"3006"},"message":"","status":"1"}
```

✧ 异步消息应答报文如下：

```
{"message":"异步调用成功","messageJson":"","status":"1"}
```

- 应答报文参数返回结果不可以为“null”，如下是错误的

```
{"status":null,"message":null,"messageJson":null}
```

2.2 接口定义规范

各业务系统自己定义 REST 接口方法，必须同时支持 GET 和 POST 方式。

格式如下：

传输协议://主机名:端口/系统应用名/webapi/模块名/资源服务名?查询字符串

- 协议：协议名称，由于是 Web 应用，一般使用 HTTP；
- 主机名：DNS 名称或 IP 地址；
- 端口：协议所使用的端口，HTTP 默认 80（可以不写），如果不是默认 80，端口必须明确写上；
- 系统应用名：即系统应用的简称或缩写，能够简单、直观、准确的表达系统应用的意思。比如：erp、b2b 等；
- webapi/api：推荐使用的常量定义，其含义为 Web 方式的 API；在 URI 上，区别于系统应用中其它部分；
- 模块名：系统中提供 Web API 的模块名称，比如：ordermodule，能够简单、直观、准确地描述模块；
- 资源服务名：即服务接口方法的映射名称，命名规则，与模块名一样，简单、

直观、准确表达含义；

- 查询字符串：查询字符串，方法作用域信息，传递参数信息。

GET 方式调用如下：

```
http://api.闪来云.com/encrypttest/api/AddEncrypt?orderno=1&erpCode=2&qty=1&mobile=3&userID=4
```

2.3 服务关联命名规范

2.3.1 服务命名规范

- 服务的命名规范为：actionobject
  - action 是服务中所有操作行为的抽象描述，单词必须小写；
  - object 是服务中所有操作实例的抽象描述，单词必须小写。

2.3.2 服务操作命名规范

- 服务中的操作的命名规范为：actionobject
  - action 是服务操作的行为定义
    - action 的具体表达可以是：search、get/inquire、add/create、update、delete 等；
    - 表达具体 action 的单词必须小写；
    - search：查询特定名称的实例列表；
    - get/inquire：查询特定名称的详细实例；

- ✧ add/create：增加一个实例；
- ✧ update：更改一个实例中的属性；
- ✧ delete ：删除一个实例。

➢ object 是实例的名称（如 customer）。

2.4 消息字符编码规范

消息在编码、传输、解析过程中，统一使用 UTF-8 编码格式。设置字符编码的方法如下，请参考：

- 设置 HTTP 消息头 content-charset 参数

```
filePost.getParams().setParameter(HttpMethodParams.HTTP_CONTENT_CHARSET,"utf-8");
```

- 设置 Tomcat 服务器，修改 server.xml 配置文件

```
<Connector port="8080" protocol="HTTP/1.1" maxThreads="150" connectionTimeout="20000" redirectPort="8443" URIEncoding="utf-8"/>
```

- 消息传输前定义字符编码

```
String replyMsg = new String("返回消息".getBytes(), "UTF-8");
requestParas.put("messageJson",URLEncoder.encode(messageJson, "UTF-8"));
```

- 在 Get 或 Post 接口设置编码方式

```
@Produces("application/json; charset=UTF-8")
```

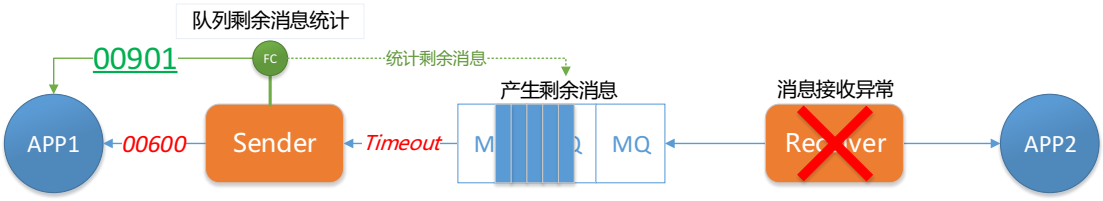
第三章 异常响应

3.1 节点异常与流量控制说明

3.1.1 消息接收异常情况

当消息接收异常时，队列中会产生剩余消息。消息发送器 Sender 返回 APP1 "00600" 超时错误编码。

如果增加流量控制功能，FC 流量控制器收到消息后，首先监控队列中是否产生剩余消息，如果超出相关阀值，按照一定概率，直接返回 APP1 "00901" 队列剩余消息超出阀值错误编码。



3.1.2 消息处理异常情况（自动签收）

当消息处理异常时 ,消息接收器Receiver 返回处理异常编码 "0050X",消息发送器Sender继续返回APP1 "0050X" 异常错误编码。

使用自动签收（未使用手动签收功能），消息处理异常，队列中无剩余消息。

如果增加流量控制功能，FC流量控制器统计连续收到 "0050X" 的错误情况，如果超出相关阀值，按照一定概率，后续收到该队列消息后，直接返回APP1 "00902" 超出阀值错误编码。

3.1.3 消息处理异常情况（异步消息手动签收）

当消息处理异常时，消息接收器 Receiver 返回处理异常编码 "0050X"，消息发送器 Sender 继续返回 APP1 "0050X" 异常错误编码。

异步消息使用手动签收功能，发生消息处理异常（异常编码 "0050X"），队列中产生剩余消息。同时该消息由 MQ 系统自动重发，重发间隔当前系统设置为 30 秒。如果接口始终没有恢复正常，24 小时后（当前系统设置为 24 小时）异步消息自动失效，重发停止；在 24 小时之内接口恢复正常，异步消息签收成功，重发停止。

如果增加流量控制功能，FC 流量控制器收到消息后，首先监控队列中是否产生剩余消息，如果超出相关阀值，按照一定概率，直接返回 APP1 "00901" 队列剩余消息超出阀值错误编码。

3.1.4 MQ 组件异常情况

当MQ组件异常时，消息发送器Sender返回APP1 "00300" 组件异常错误编码。流量控制器对该异常不作处理。

3.1.5 消息发送器异常情况

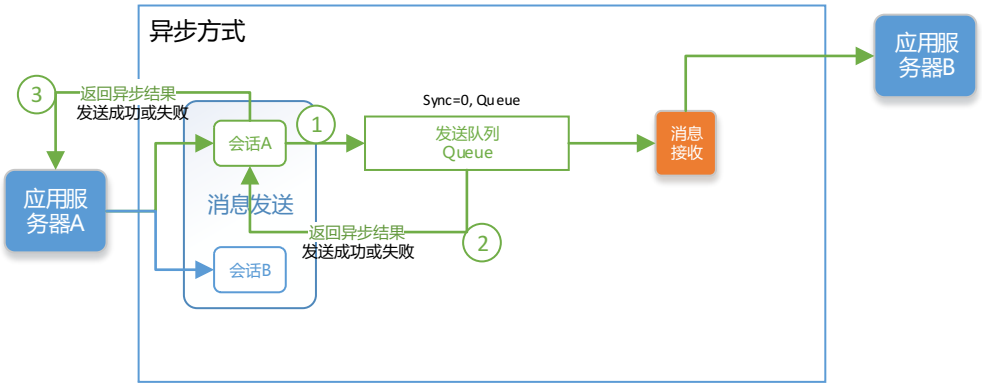
当消息发送器 Sender 异常时，APP1 发生"Connection refused" 异常。流量控制器对该异常不作处理。

第四章 访问方式

4.1 异步访问方式

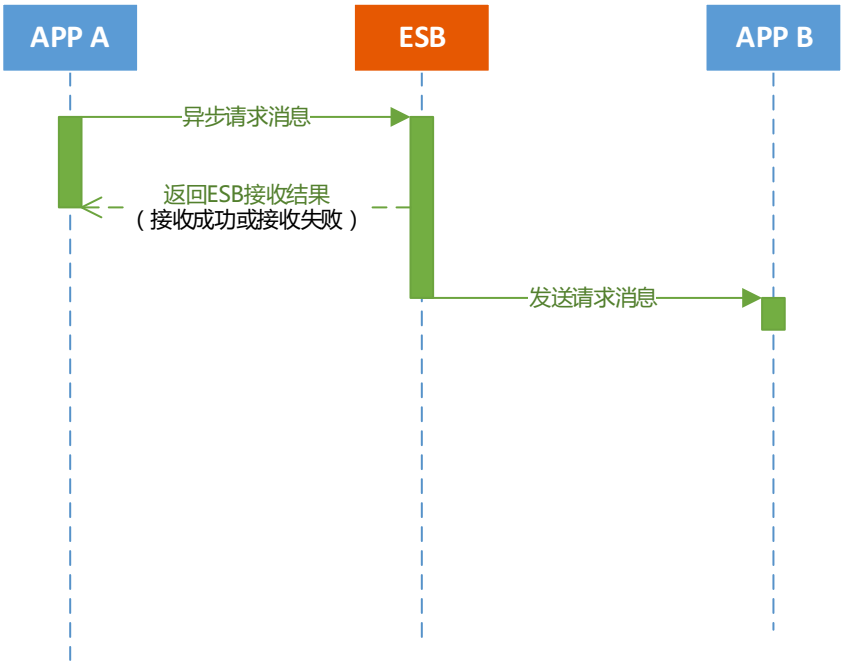
应用服务器 A 成功将请求发送到 SUNLIGHTMQ 后，即可关闭连接，不再等待接收应用服务器 B 的返回结果。

（1）消息流转组件图如下：



异步方式：应用服务器A成功将请求发送到ESB后，即可关闭连接，不再等待接收应用服务器B的返回结果。

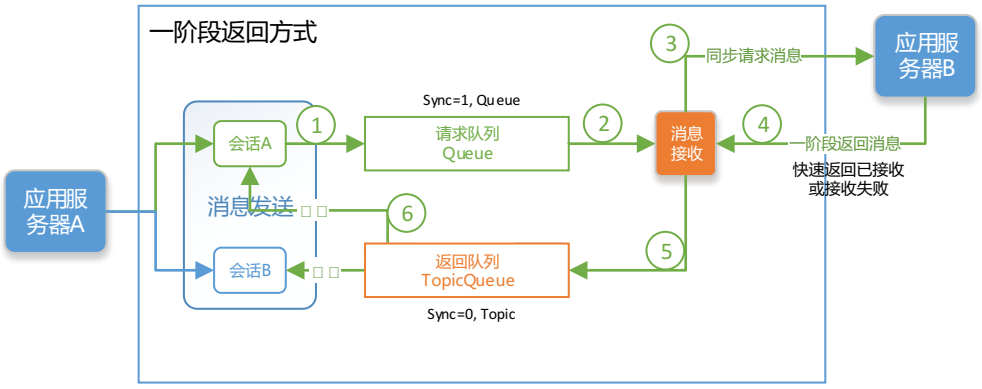
（2）消息流转时序图如下：



4.2 一阶段返回方式

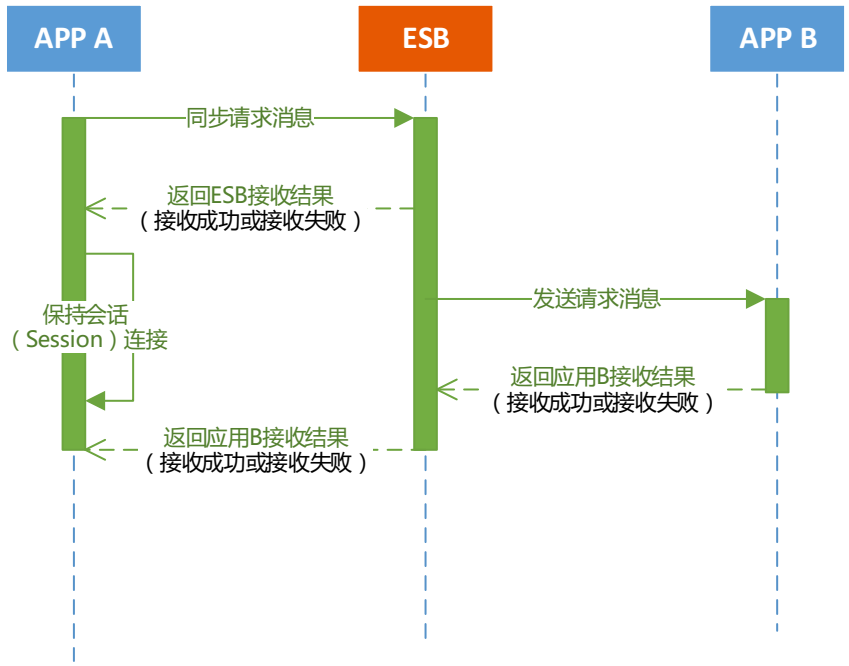
应用服务器 A 成功将请求发送到 SUNLIGHTMQ 后，保持连接，等待接收应用服务器 B 的返回结果。应用服务器 B 收到请求消息以后，直接返回接收结果：已接收或接收失败，不返回最终处理结果。

（1）消息流转组件图如下：



一阶段返回方式：应用服务器A成功将请求发送到ESB后，保持连接，等待接收应用服务器B的返回结果。应用服务器B收到请求消息以后，直接返回接收结果：已接收或接收失败，不返回最终处理结果。

（2）消息流转时序图如下：



4.3 二阶段返回方式

第一阶段分为以下两种方式：

第一阶段采用同步请求方式：应用服务器 A 成功将请求发送到 SUNLIGHTMQ 后，保持连接，等待接收应用服务器 B 的返回结果。应用服务器 B 收到请求消息以后，直接返回接收结果：已接收或接收失败，不返回最终处理结果；

第一阶段采用异步请求方式：应用服务器 A 成功将请求发送到 SUNLIGHTMQ 后，结束连接，不等待接收应用服务器 B 的返回结果，即采用异步请求方式，该方式已在 B2C3.0 项目中应用。

第二阶段：应用服务器 B 处理完成后，按照异步方式，通过“返回接口”将处理结果发送到“结果队列”，消息接收器调用应用服务器 A 的“接收接口”方法，返回结果给应用服务器 A。

使用“二阶段返回方式”，应用服务器 A 需定义以下两个接口：

请求接口：请求业务服务

接收接口：接收处理结果

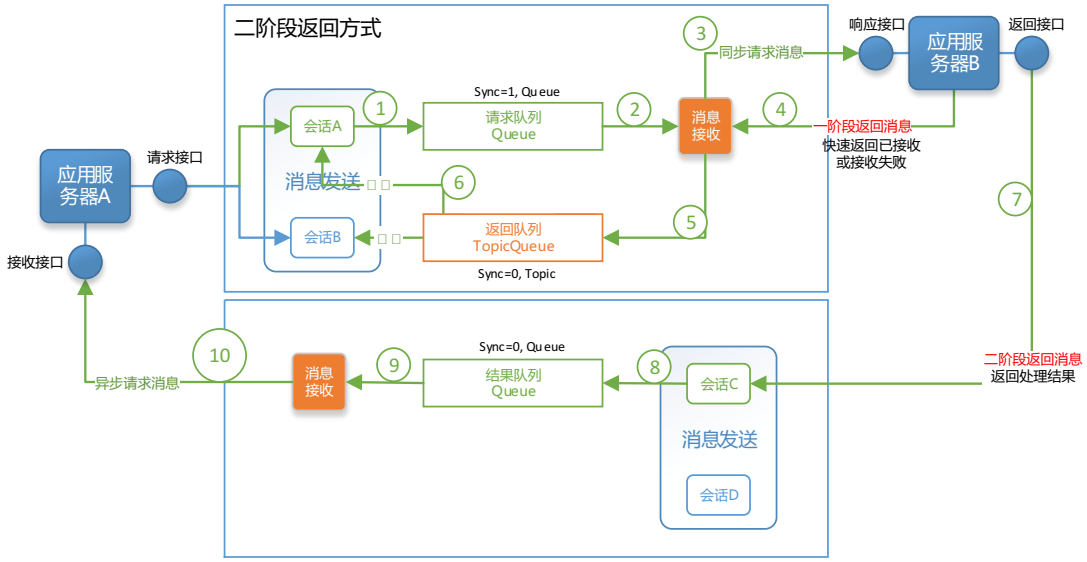
同时应用服务器 B 需定义对应的以下两个接口：

响应接口：处理业务请求

返回接口：返回处理结果

（1）消息流转组件图如下：

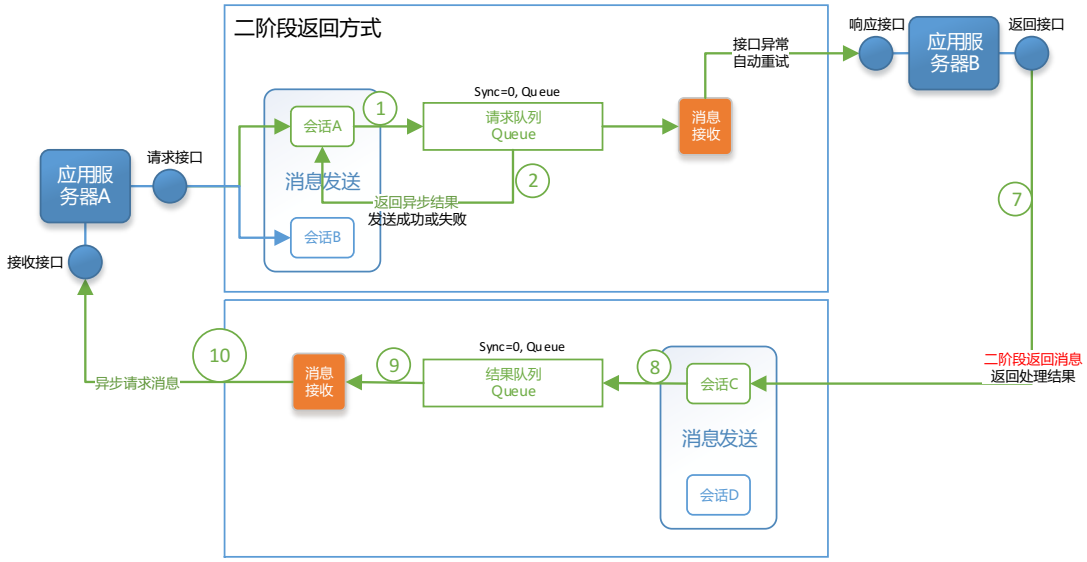
1）第一阶段采用同步请求方式：



二阶段返回方式：
第一阶段采用同步请求方式：应用服务器A成功将请求发送到ESB后，保持连接，等待接收应用服务器B的返回结果。应用服务器B收到请求消息以后，直接返回接收结果：已接收或接收失败，不返回最终处理结果。
第二阶段：应用服务器B处理完成后，按照异步方式，通过“返回接口”将处理结果发送到“结果队列”，消息接收器调用应用服务器A的“接收接口”方法，返回结果给应用服务器A。

使用“二阶段返回方式”，应用服务器A需定义以下两个接口：
（1）请求接口：请求业务服务
（2）接收接口：接收处理结果
同时应用服务器B需定义对应的以下两个接口：
（1）响应接口：处理业务请求
（2）返回接口：返回处理结果

2）第一阶段采用异步请求方式

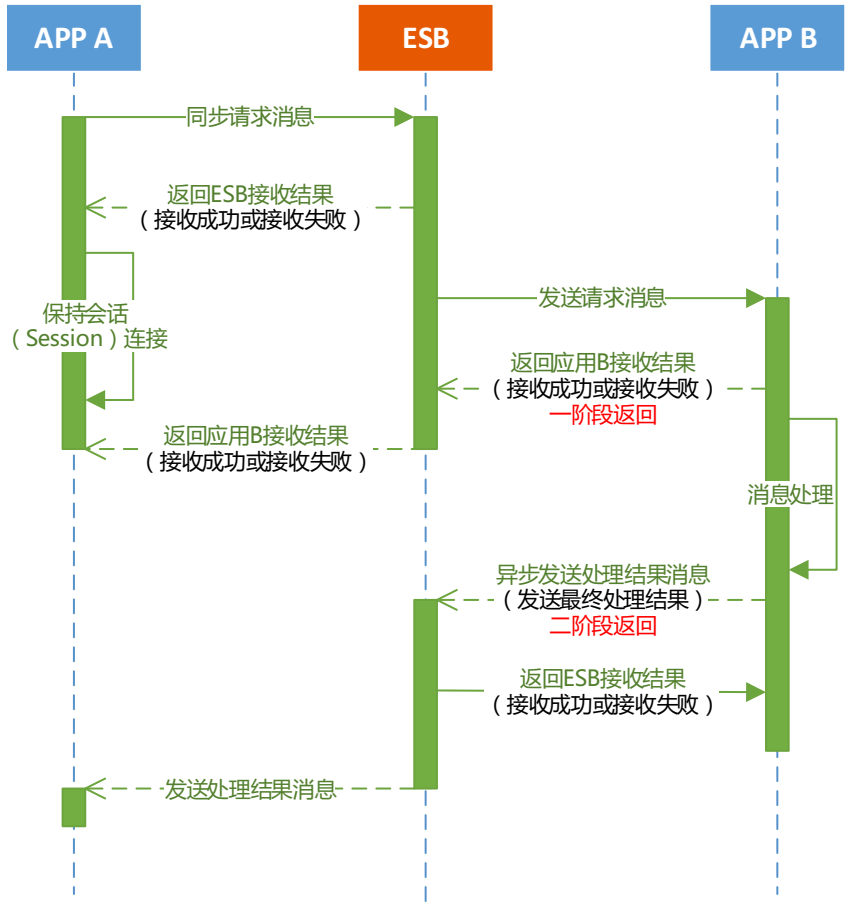


二阶段返回方式：
第一阶段采用异步请求方式，应用服务器A成功将请求发送到ESB后，结束连接，不等待接收应用服务器B的返回结果，即采用异步请求方式，该方式已在B2C3.0项目中应用。
第二阶段：应用服务器B处理完成后，按照异步方式，通过“返回接口”将处理结果发送到“结果队列”，消息接收器调用应用服务器A的“接收接口”方法，返回结果给应用服务器A。

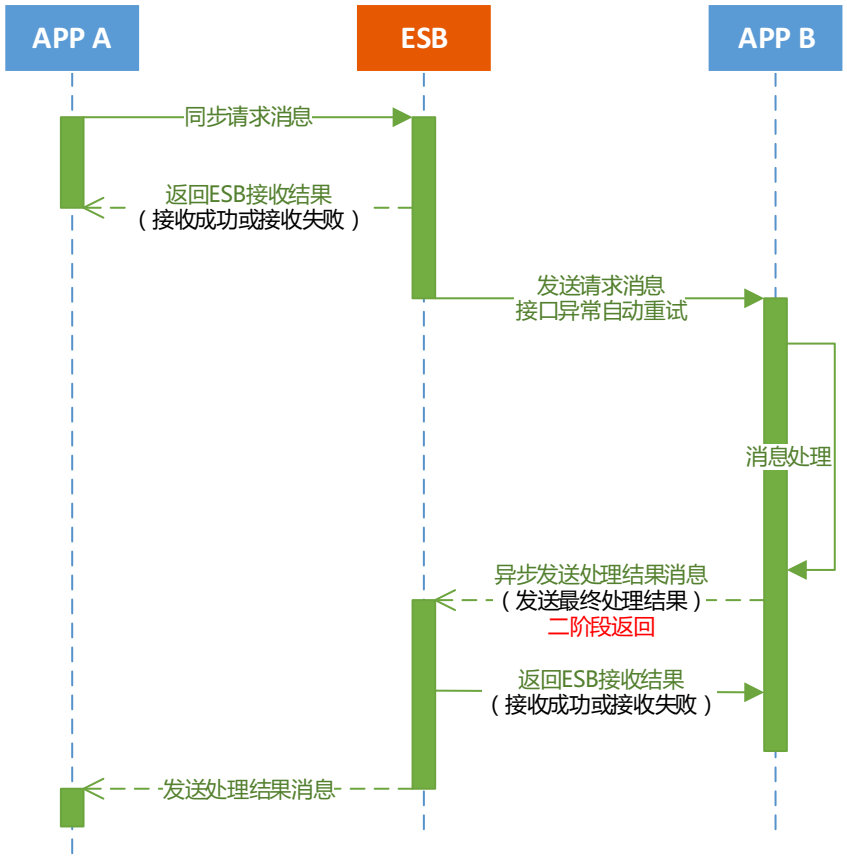
使用“二阶段返回方式”，应用服务器A需定义以下两个接口：
（1）请求接口：请求业务服务
（2）接收接口：接收处理结果
同时应用服务器B需定义对应的以下两个接口：
（1）响应接口：处理业务请求
（2）返回接口：返回处理结果

（2）消息流转时序图如下：

1）第一阶段采用同步请求方式：



2）第一阶段采用异步请求方式

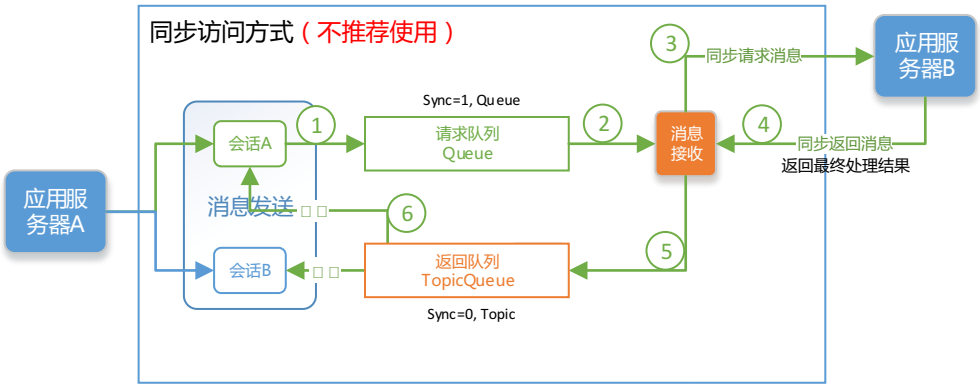

APP A
ESB
APP B
同步请求消息
返回ESB接收结果
（接收成功或接收失败）
发送请求消息
接口异常自动重试
消息处理
异步发送处理结果消息
（发送最终处理结果）
二阶段返回
返回ESB接收结果
（接收成功或接收失败）
发送处理结果消息

4.4 同步访问方式（不推荐使用）

同步访问方式（不推荐使用）：应用服务器 A 成功将请求发送到 SUNLIGHTMQ 后，保持连接，等待接收应用服务器 B 的处理结果。应用服务器 B 收到请求消息以后，处理请求并返回最终处理结果。

同步访问方式会造成会话（session）长时间等待，高并发时影响系统整体稳定性，因此不推荐使用。需要返回最终处理结果的业务，推荐使用 “二阶段返回方式”。

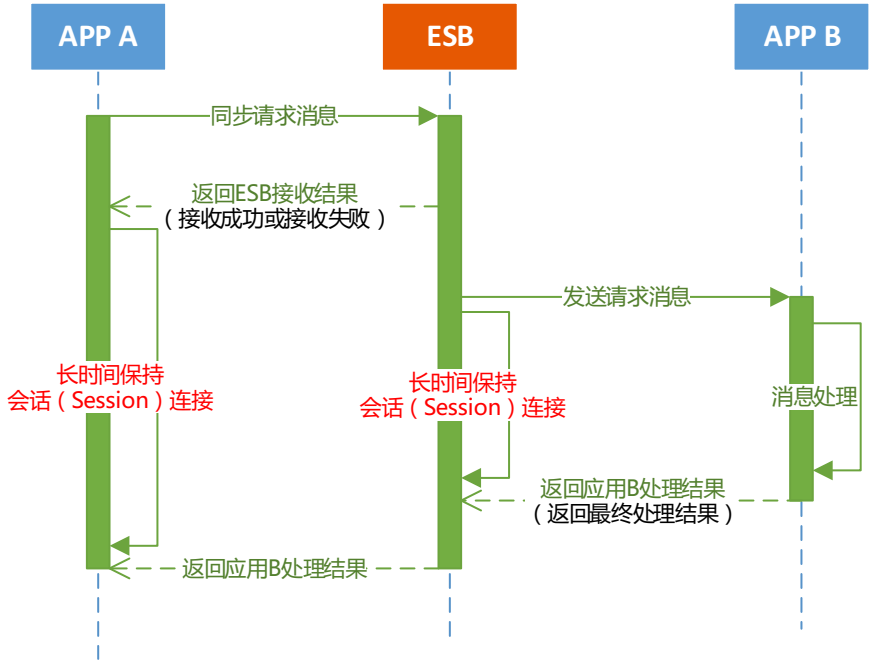
（1）消息流转组件图如下：



同步访问方式（不推荐使用）：应用服务器A成功将请求发送到ESB后，保持连接，等待接收应用服务器B的处理结果。应用服务器B收到请求消息以后，处理请求并返回最终处理结果。

同步访问方式会造成会话（session）长时间等待，高并发时影响系统整体稳定性，因此不推荐使用。需要返回最终处理结果的业务，推荐使用 “二阶段返回方式”。

（2）消息流转时序图如下：


APP A
ESB
APP B
同步请求消息
返回ESB接收结果
（接收成功或接收失败）
发送请求消息
长时间保持
会话（Session）连接
长时间保持
会话（Session）连接
消息处理
返回应用B处理结果
（返回最终处理结果）
返回应用B处理结果

第五章 开发建议

5.1 建议设计开发步骤

以下内容为按序排列的针对新接入应用系统的建议开发步骤：

- 公共处理：
  - 与 SUNLIGHTMQ 管理组商定接入系统参数；
- 服务提供方：
  - 划分设计服务和服务操作；
  - 对服务进行命名、对服务操作进行命名；
  - 接入系统作为服务提供方与 SUNLIGHTMQ 管理组商定并发处理性能；
  - 通过 SUNLIGHTMQ 平台为服务分别进行注册；
- 服务请求方：
  - 服务请求方与 SUNLIGHTMQ 管理组、服务提供方，共同商定其对于每个服务的处理超时时间；
  - 根据服务方提供的参考数据进行评估和开发；

5.2 编程建议

Java 和 C# REST 接口开发时请注意：必须同时实现 GET 和 POST 接口方法，不能只实现 GET 方法。

Java 代码举例：

```
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;

import com.bean.Result;
```

```
@Path("/find/")
public interface IHello {

    /**
     * 登录
     * @param userName 帐号
     * @param userPass 密码
     * @return
     */
    @POST
    @Path("/login")
    Result login(@FormParam("userName")String userName,@FormParam("userPass")String
userPass);

    /**
     * 按名称查询
     * @param who 名字
     * @return
     */
    @GET
    @Path("/findByName/{who}")
    Result findByName(@PathParam("who")String who);

}
```

C# 代码举例：

```
public class TestController : ApiController
    {
        // GET: api/Test
        public LpdReturnDto<dynamic> Get()
        {
            return new LpdReturnDto<dynamic>
            {
                status=1,
                message=string.Empty,
                messageJson = new { Code="Hello World" }
            };
        }
        // GET: api/Test/5
        public string Get(int id)
        {
            return "value";
```

```
        }
        // POST: api/Test
        public LpdReturnDto<dynamic> Post([FromBody]string value)
        {
            return new LpdReturnDto<dynamic>
            {
                status = 1,
                message = string.Empty,
                messageJson = new { Code = "Hello World" }
            };
        }

        // PUT: api/Test/5
        public void Put(int id, [FromBody]string value)
        {
        }

        // DELETE: api/Test/5
        public void Delete(int id)
        {
        }
    }
```