

Интеграция с БД

Lifehack по работе с JDBC в SAP PI

Шашурин Иван
Сентябрь 2021



Опыт



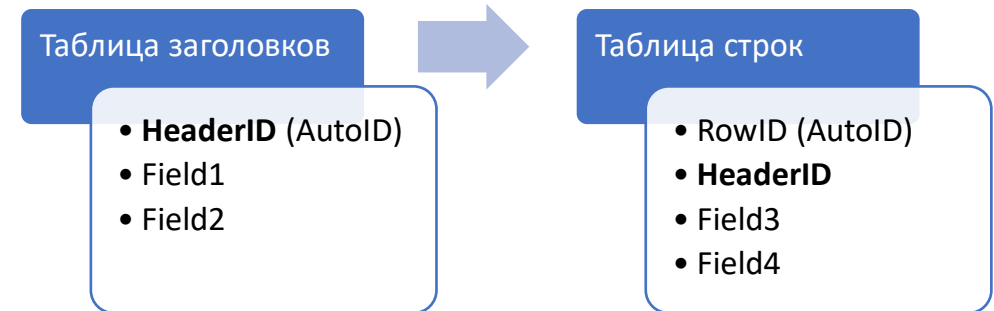
- Основная ERP система не имеет интерфейсов интеграции
- Дополнительно 3 системы, интегрируемые только через БД
- СУБД: Informix, Oracle, MySQL
- Более 200 JDBC каналов
- Сотни таблиц, десятки миллионов строк

Глобальные переменные в JDBC Receiver

Проблема: необходимость в JDBC Receiver использовать последовательность запись/чтение/запись

Решение: использовать глобальную переменную (в Oracle – контекст, PostgreSQL – сессионная переменная).

- Создаем хранимую процедуру по записи в таблицу заголовков. Она делает INSERT + сохранение lastInsertID в переменную
- Создаем хранимую процедуры для чтения переменной
- В JDBC Receiver вызываем шаги:
 - Вызов хранимой процедуры по сохранению в таблицу заголовка для первого документа.
 - Вызов INSERT для вставки позиций первого документа, где HeaderID берется из переменной.
 - Повтор для следующих документов.
 - В канале устанавливаем флаг Disconnect After Processing Each Message, чтобы сбрасывать переменную.



Timeout JDBC Receiver

Проблема: в случае сетевых ошибок между PI и одной из систем, канал JDBC может повиснуть без ошибок (DLNG). От этого занимают потоки JDBC, что приводит к очередям с другими системами.

Решение: устанавливать timeout в JDBC каналах

- `sqlQueryTimeout` – каждый statement
- `socketTimeout` – каждый statement или connection (зависит от JDBC драйвера)
- `loginTimeout`
- `syncTimeout` во вкладке Module
- Параметр в строке подключения

The screenshot shows the SAP NetWeaver Administrator (NWA) configuration interface for a JDBC Receiver. The 'Advanced' tab is active, displaying the 'Additional Parameters' section. A table at the bottom lists the configured timeout parameters.

Name	Value
sqlquerytimeout	2400
driver.socketTimeout	2400
driver.loginTimeout	1200

Буферные таблицы

Проблема: большая интенсивность онлайн обмена, блокировки в таблицах, лаги у пользователей из-за интеграции.



Дополнительно:

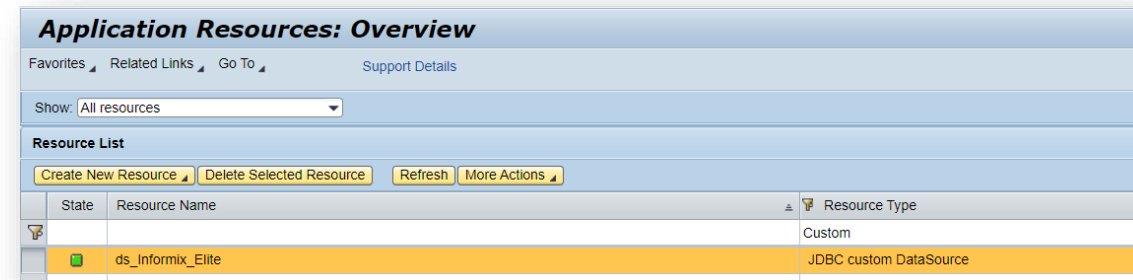
- Внешний мониторинг для контроля статусов
- Система регламентной очистки буферных таблиц

Data source

Проблема: в PI приходит множество больших XML (10-20 МБ), которые необходимо ощутимо преобразовать и положить в БД (update в кучу таблиц).

Решение: использовать Java Mapping или Java Proxy (JPR) для преобразования и осуществления оптимальной записи в БД.

- Создаем data source в NWA
- В Java коде занимаемся парсингом и выстраиванием запросов:
 - Вместо UPDATE используем DELETE + INSERT
 - Первым шагом группируем блоки DELETE
 - Вторым шагом группируем блоки INSERT (по несколько тысяч строк за один INSERT)
 - Запускаем скрипты блоками



Favorites ▾ Related Links ▾ Go To ▾ Support Details		
Show: All resources ▾		
Resource List		
Create New Resource ▾ Delete Selected Resource Refresh More Actions ▾		
State	Resource Name	Resource Type
🟢	ds_Informix_Elite	JDBC custom DataSource

```
InitialContext ctx = new InitialContext();
DataSource ds = (DataSource)ctx.lookup("jdbc/notx/ds_Informix_Elite");
Connection dbConnection = ds.getConnection();
Statement sqlDbStatement = dbConnection.createStatement();
```

```
sqlDbStatement.addBatch("START TRANSACTION;");
sqlDbStatement.addBatch("DELETE ...");
sqlDbStatement.addBatch("INSERT ...");
sqlDbStatement.addBatch("COMMIT;");
sqlDbStatement.executeBatch();
sqlDbStatement.clearBatch();
```

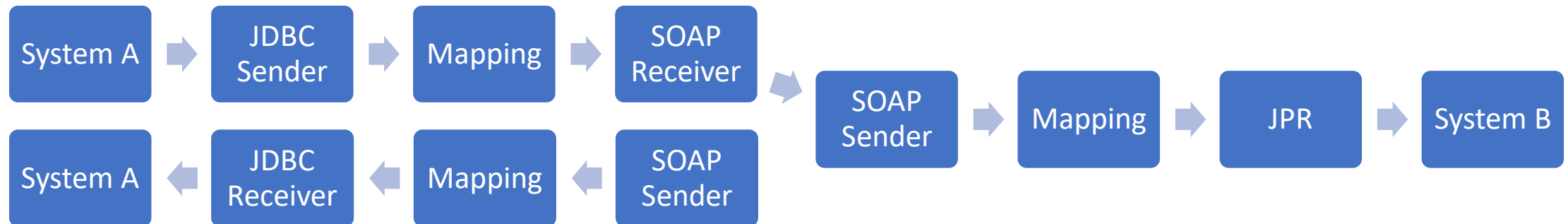
Async/sync мосты & Java proxy (JPR)

Проблема: необходимость реализации схемы Sync JDBC ↔ mapping ↔ Sync JPR, т.е. в системе А необходимо взять данные по JDBC, записать их в систему Б, ответ от системы Б отправить в систему А.

- JDBC Sender в режиме Best Effort игнорирует XML SQL, выполняется Update SQL Statement из Sender Канала
- Реализация модулей на async/sync не совместима с реализацией вызова JPR
 - При реализации на receiver, ответное сообщение от JPR падает с ошибкой «Queue unavailable JPRResp», второй ICO не отрабатывает
 - При реализации на sender, ответное сообщение падает с ошибкой «CPAObjectNotFoundException»

Решение: использование async/sync мост одновременно с петлей в PI

- ICO 1: System A (JDBC, async) ↔ PI (SOAP, async-sync) = отправка в петлю
- ICO 2: PI (SOAP, Sync) ↔ System B (JPR, Sync) = доставка из петли получателю
- ICO 3: PI (SOAP, async-sync) ↔ System A (JDBC, async) = ответное сообщение

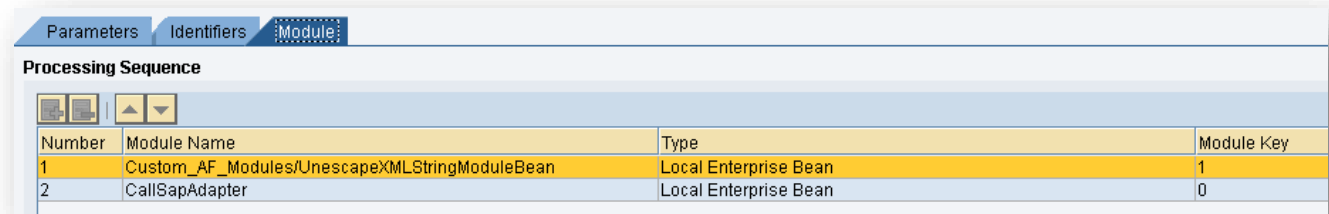


Генерация xml в СУБД & валидация в PI

Проблема: при генерации XML средствами СУБД (Oracle – xmlelement, Informix – genxml, самодельные варианты), после отправки через JDBC Sender, в стандартном модуле происходит автоматическая конвертация всех тегов < и > в < и >, что приводит к невозможности использовать валидацию в ICO.

Решение: java модуль в канал

- Разэкранирование с помощью `org.apache.commons.text.StringEscapeUtils`
- Логирование, при необходимости



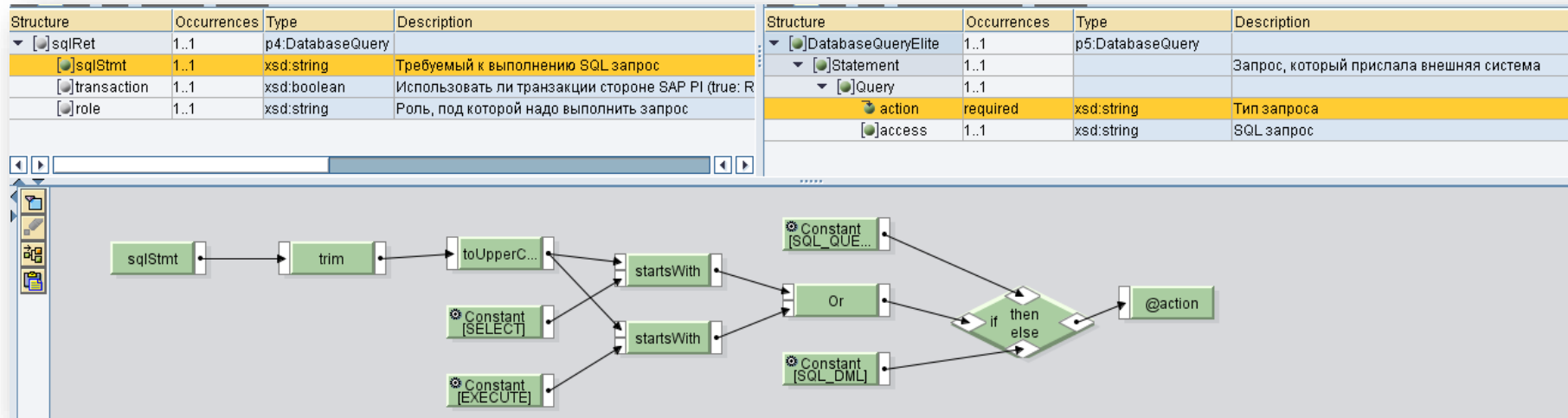
The screenshot shows a configuration window with three tabs: 'Parameters', 'Identifiers', and 'Module'. The 'Module' tab is active, displaying a 'Processing Sequence' table. The table has four columns: 'Number', 'Module Name', 'Type', and 'Module Key'. It contains two entries: entry 1 is 'Custom_AF_Modules/UnescapeXMLStringModuleBean' of type 'Local Enterprise Bean' with key '1'; entry 2 is 'CallSapAdapter' of type 'Local Enterprise Bean' with key '0'.

Number	Module Name	Type	Module Key
1	Custom_AF_Modules/UnescapeXMLStringModuleBean	Local Enterprise Bean	1
2	CallSapAdapter	Local Enterprise Bean	0

```
/**
 * Замена символов экранирования, например &lt; → <
 * @param inputMessage
 * @return
 * @throws ModuleException
 */
private String unescapeXmlTags(String inputMessage) throws ModuleException {
    try {
        return StringEscapeUtils.unescapeXml( inputMessage );
    } catch (Exception e) {
        returnError("UnescapeXMLStringModuleBean failed: exception while trying to unescape");
    }
    return null;
}
```


Прочие полезности

- Разные пользователи, для разных процессов, особенно для бизнес таблиц (отслеживать активность, убивать сессии)
- При падении сообщения на JDBC Sender при отключенном сохранении payload (BI/VI), происходит откат транзакции.
- При использовании async-sync мостов на JDBC Receiver: если произошла ошибка во втором ICO, но в нём не настроено сохранение payload (BI/VI), то отката транзакции на стороне получателя не происходит.
- Для запуска хранимок в JDBC Receiver (EXECUTE) можно использовать SQL_DML, если ХП ничего не возвращает, но SQL_QUERY, если ХП возвращает результаты своей работы.



Благодарю за внимание