Реализовано:

1. Основные таблицы модуля:
   1. IKP\_TASKS. Основная таблица, содержит перечень заданий. ID – по сиквенсу S\_IKP\_TASKS.
   2. IKP\_TASKS\_EVENTS. Дочерняя к основной. Действия в заданиях. ID по сиквенсу S\_IKP\_TASK\_EVENTS. Уникальный индекс – ID родительской задачи + №п/п, для однозначного порядка выполнения.
   3. IKP\_TASK\_PRESETS. Пресеты (предустановленные настройки). Справочник для действий, пока не будем использовать, можно не отвлекаться. По сути – содержит заранее подготовленные PL/SQL блоки или имена пакетов/процедур для выполнения типичных действий.
   4. IKP\_LOG. Журнал отладки.
   5. IKP\_TMP\_FILE. Временная таблица с одним полем типа BLOB и, по сути, одной записью в рамках сессии. Работает ON COMMIT PRESERVE ROWS. Нужна для загрузки файла в BLOB поле для дальнейшей обработки в пакетах и выгрузки сформированного в пакетах файла из этого же поля.
2. Пакеты:
   1. IKP\_ADM. Основной пакет. Функции инициализации задания, действий в рамках задания, выполнение действий. Вся работа с действиями идет по №пп в рамках задания. Данные действий (ID действия, имя файла (загружаемого или сгенерированного для выгрузки), произвольный набор параметров) хранятся в пакете, т.е. существуют в рамках сессии для одного пользователя.
   2. IKP\_DEBUG. Пакет для отладки. Пишет отладочную таблицу (пока только таблицу, можно будет подумать с выводом в DBMS, можно добавить вывод полного журнала за сессию в текст). Работа идет также в рамках сессии, сообщение пишется или не пишется в таблицу исходя из заданного в рамках сессии уровня отладки (Все – Отладка – Инфо – Только ошибки).

В архиве IKP.7z есть runme.sql, которым можно создать все объекты (сиквенсы, таблицы, пакеты, синонимы, гранты); есть fill.sql, который добавит по одной записи в таблицы заданий и действий – для примера; example.sql – пример запуска задания на PL/SQL, его нужно будет реализовывать на Java.

Итак, что нужно на данный момент от интерфейса.

1. Основная форма. В верхней части основная таблица (IKP\_TASKS) с тулбаром (добавить/редактировать/удалить/запустить). В нижней части – дочерняя таблица (IKP\_TASK\_EVENTS) с тулбаром (добавить/редактировать/удалить). Связка по внешнему ключу (через iEventTaskID – iTaskID).
2. Две формы для добавления/редактирования записей в верхней и нижней таблицах.
3. Окно запуска задания. По кнопке запустить из основной формы поднимаем окно запуска. В заголовок – наименование задания (cTaskName), из элементов на форме – кнопка Пуск/стоп, комбобокс с уровнем отладки (ALL=1,DEBUG=2,INFO=3,ERROR=4), основную часть формы занимает таблица с выводом из IKP\_LOG полей STRTYPE,MSGTXT,DT; сортировка по SESSID,STRNO. Крайне желательно в этом окне делать отдельный коннект к базе, чтобы для выполнения каждого задания мы имели свою сессию. Сессию **USERENV** ('SESSIONID') тоже можно выводить в заголовок окна для информации. Содержимое дебаг-таблицы обновляем после завершения (полного завершения задания либо по кнопке пуск/стоп); можно подумать как обновлять по таймеру (совершенно не первоочередная задача).

На параметры задания пока не смотрим – они у нас пока что всегда будут на стороне клиента, периодичность/интервал/частота повторения значения не имеет, т.к. запускаем пока по кнопке. Так что в клиентской части для таблицы заданий имеют значение только ID задачи и ее наименование.

Запуск выполняем как в примере example.sql.

Пропишу алгоритм словами:

1. Инициализируем задание, передавая как параметр ID задания
2. Запускаем задание (при этом взводится признак что задание выполняется, чтобы в других сессиях не могли запустить).
3. Далее циклом по всем действиям в рамках задания с сортировкой по NPP выполняем:
   1. Инициализация действия (передаем номер п/п)
   2. Если загрузка файла – передаем имя файла для действия в пакет IKP\_ADM и загружаем файл во временную таблицу.
   3. В любом случае – выполняем действие
   4. Если выгрузка файла – берем имя файла для действия из пакета IKP\_ADM и выгружаем файл из временной таблицы.
4. Финишируем задание (снимается признак выполнения).

Действия у нас на данный момент будут либо пресет, либо PLSQL блок (0-preset, 1-PLSQL, 2-script, 3-OS cmd), для упрощения берем только PLSQL – тип 1.

Основная задача – отработать загрузку и выгрузку файла через BLOB поля. Т.е. действие - загрузка файла: берем входящую папку из настроек действия (cEventInDir), хватаем файл из нее, загружаем в поле во временной таблице и толкаем действие на выполнение. Для выгрузки – в обратном порядке. С данными в BLOB-поле уже будем работать в пакетах.

Пока готовится интерфейс – займусь обвязкой пакетов для работы с DBF и XLSX в BLOB-полях, чтобы тестировать потом уже все вместе.