Использование SQL-запросов и команд языков программирования для описания алгоритмов выборки данных в функциональных спецификациях на разработку ERP-систем

Петров Сергей Владимирович

Аннотация: в статье затрагивается вопрос описания алгоритмов выборки данных, содержащихся в функциональных спецификация на разработку ERP-систем. Вводится модель зрелости задания алгоритмов, связанная с опытностью технических консультантов, формирующих спецификации. Рассматриваются способы записывания логики селекции данных на основе краткого текста, SQL-запросов и комбинации структурированного языка запросов и выборочных команд из прикладных языков программирования ERP-системы. Показывается, чем ближе описание к текстовому, тем разработчику сложнее понять логику и наоборот.

Введение

Практический каждый проект внедрения корпоративных информационных систем требует их доработки. Для чего готовится документ функциональной спецификации на разработку, содержащий в себе описание исходного требования, верхнеуровневое понимание решения, а также технические детали реализации [1]. Документ готовится вне зависимости от применяемой методологии имплементации, не исключением являются Agile-подходы. В спецификации должен быть соблюден баланс бизнес и технических составляющих: пользователи должны увидеть свои исходные требования к системе и сформировать общую картину решения, в то время как разработчики – найти технические детали, достаточные для старта программирования решения.

Обычно эта задача решается за счет выделения в документе спецификации бизнес и технических частей, тем самым каждый из заинтересованных получает требуемую для работы информацию. К сожалению, нет какого-либо регламентированного подхода, описывающего структуру и содержание спецификации, поэтому в каждом проекте наполнение документа уникально. С технической точки зрения документ спецификации должен содержать описание всех необходимых атрибутов пользовательских экранов и алгоритмов их заполнения,

изменения и очистки. Правил по записи алгоритмов в спецификации так же нет, поэтому основное требование состоит в том, чтобы разработчику была понятна логика техрешения.

Чем опытнее функциональный консультант, тем детальнее он описывает алгоритмы выборки данных, приближая их к языку программирования ERP-системы. И, наоборот, новичок указывает минимум деталей в форме близкой к краткому тексту. Существует несколько подходов, как можно описать алгоритмы обработки данных. При этом они сильно зависят от зрелости и опытности технических специалистов, участвующих в ERP-проекте [2]. Давайте проанализируем несколько методов и дадим им оценку.

Цель и задачи

Цель статьи состоит в обзоре способов описания алгоритмов выборки данных в спецификациях на разработку ERP-систем. Чем качественнее написана функциональная спецификация, тем меньше алгоритмических ошибок ожидается при ее реализации. Достижение этой цели потребует выполнения следующих задач:

- рассмотрение способов описания алгоритмов;
- анализ SQL-запросов для использования в спецификациях на разработку;
- оценивание методов описания алгоритмов.

1. Алгоритмы в спецификациях на разработку ERP-систем

ERP-система является транзакционной, то есть ориентированной на обработку больших массивов данных. Преимущественно в системах подобного класса используются реляционные базы данных, таблицы в которых взаимосвязаны между собой по принципу «сущность-связь». Следовательно, вся информация, необходимая для пользователей хранится в таблицах баз данных и отображается в экранные формы программ по запросу. Подготовка функциональных спецификаций требует описания алгоритмов выборки данных, иными словами логики выбора информации из таблиц баз данных. Существует несколько способов, как можно описать подобные алгоритмы в спецификациях на разработку:

- ссылка на поле таблицы баз данных без указания деталей;
- применение SQL-запросов;
- использование SQL-запросов и прочих команд языков программирования.

Первый и самый распространенный метод описания алгоритмов заполнения полей в спецификации на разработку представляет собой явную ссылку на поле таблицы баз указания деталей, откуда изымается значение. Форма записи в этом случае принимает вид (1):

<Таблица>-<Поле>. (1)

Таким способом пользуются новички, так как приведенное описание не содержит технических деталей алгоритмов: входные данные, ограничения и прочие параметры. Более того, если программная разработка является сложной, то достаточно непросто передать всю логику обработки только используя (1), приходится пользоваться кратким текстом для описания алгоритма, что порождает разночтение. Пример использования (1) дан в таблице 1. Как видно из таблицы, описание правил заполнения схоже по принципу с оператором присвоения в языках программирования: слева указывается переменная, а справа – передаваемая ей значения, с той лишь разницей, что переменная характеризует поле на экране программы.

Таблица 1. Пример описания логики заполнения поля на основе ссылки

№	Название поля	Описание поля	Правило	Алгоритм
1	BUDAT	Дата проводки/Posting date	=	MKPF-BUDAT
2	MATNR	Материал/Material	=	MSEG-MATNR
3	MENGE	Коли-во/Quantity	=	MSEG-MENGE

Следующим подходом для указания логики заполнения полей служит применение языка структурированных запросов SQL (Structured Query Language). Структура SQL единообразна и не зависит от языка программирования. Среда программирования может слегка менять синтаксис SQL, добавляя в него специфичный диалект, однако принципиальный разночтений он не дает. Команд обработки данных в языке SQL немного. Примеры самых частых из них приведены на рис. 1. В спецификациях на разработку в 90% случаев используется оператор SELECT,

все остальные применяются гораздо реже. Этот оператор позволяет осуществить выборку данных из таблицы при указании начальных ограничений (2):

*SELECT * FROM <Таблица> INTO <Временный Массив> WHERE <Условия>*, (2)

где INTO <Временный Массив> является опциональной специфичный командой языка программирования, позволяющей сохранять найденные значения во временную переменную. Обычно переменная именуется схожим с названием таблицы образом, чтобы не путать разработчика. Рассмотрим пример, данный в табл. 2. Из него легко заметить, что использование SQL-запросов значительно упрощает задание выборки. Однако в случае, если на экранной форме нужно выдать сразу несколько полей, значения которых хранятся в той же таблице баз данных, описание кажется слишком нагромождённым. Ощущается необходимость разовой записи SQL-запроса, а далее лишь ссылки на результаты выборки. Поэтому переходим к следующему, финальному способу описания.

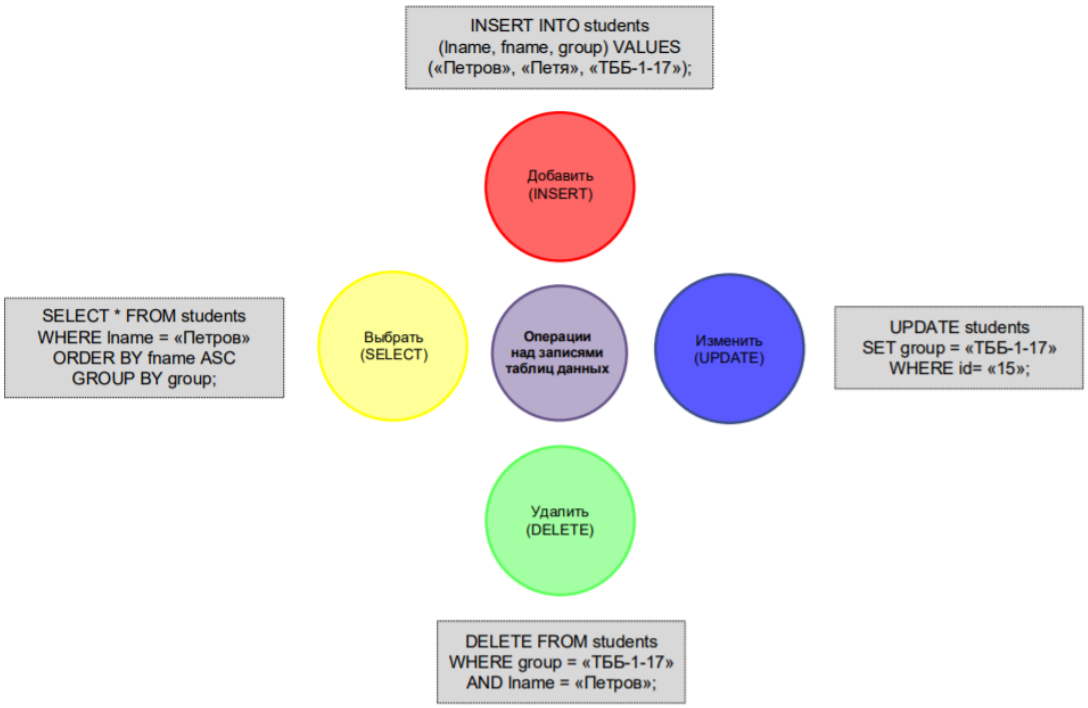

Рис.1. *Типовые SQL-запросы и примеры их использования*

Таблица 2. Пример описания алгоритма выборки данных на основе SQL-запросов

№	Название поля	Описание поля	Правило	Алгоритм
1	BUDAT	Дата проводки /Posting date	=	Select BUDAT from MKPF where MBLNR = «Номер документа материала» селекционного экрана and MJAHR = «Год документа материала» селекционного экрана
2	MATNR	Материал/Material	=	Select MATNR from MSEG where MBLNR = «Номер документа материала» селекционного экрана and MJAHR = «Год документа материала» селекционного экрана
3	MENGE	Коли-во/Quantity	=	Select MENGE from MSEG where MBLNR = «Номер документа материала» селекционного экрана and MJAHR = «Год документа материала» селекционного экрана

Наконец, третий способ проектирования алгоритмов выборки данных подразумевает применение как SQL-запросов, так и части команд прикладных языков программирования. В отличие от предыдущего способа, SQL-запросы выносятся в отдельные специи описания, находящиеся много раньше логики заполнения полей. Здесь работает подход: сначала выбираются все данные во временные массивы, которые затем используются для заполнения полей экранных форм. Поля же, в свою очередь, ссылаются на них. Более того, в случае необходимости задания сложной логики расчета, применяются команды языков программирования. Самая применяемая из которых LOOP AT (3):

Loop at <Таблица/Временный Массив>, (3)

позволяющая выполнять циклическую обработку данных для всех записей массива. Команда, в частности, сильно помогает, когда данные объекта нормализованы и разнесены по нескольким таблицам, например, заголовка и позиций. Проанализируем пример из табл. 3. Обратите внимание, что алгоритмы выбора данных 1-2 описаны в начале секции, причем оба из них сохраняют найденные значения во временные массивы. Алгоритм 2 использует команду LOOP AT для циклического выбора данных из нормализованной таблицы, относящейся к уровню позиции. Для заполнения поля 1, находящегося на уровне заголовка, применяется значение из временного массива, найденного согласно алгоритму 1, в то время, как поля 2-3 ссылаются на данные, полученные на основе алгоритма 2. Тем самым исключается дублирование описания, как это было отмечено в прошлом методе. Данный способ применяют более продвинутые функциональные специалисты в виду сложности.

Таблица 3. Пример описания логики обработки данных на основе SQL-запросов и прочих команд из языков программирования

№	Название поля	Описание поля	Правило	Алгоритм
Алгоритмы выборки				
A1	Select * from MKPF Into MKPF where MBLNR = «Номер документа материала» селекционного экрана and MJAHR = «Год документа материала» селекционного экрана			
A2	Loop at MKPF (шага A1) Select * from MSEG Into MSEG where MBLNR = MKPF-MBLNR and MJAHR = MKPF-MJAHR			
Заполнение полей				
1	BUDAT	Дата проводки/Posting date	=	MKPF-BUDAT (шаг A1)
2	MATNR	Материал/Material	=	MSEG-MATNR (шаг A2)
3	MENGE	Коли-во/Quantity	=	MSEG-MENGE (шаг A2)

Заключение

В виду того, что нет какого-либо общего правила по описанию алгоритмов в спецификациях на разработку для проектов внедрения ERP-систем, каждый функциональный консультант использует свой подход. Если отталкиваться от опытности специалистов, то можно выделить три модели зрелости задания логики обработки данных. Первая модель подразумевает использование краткого текста для описания алгоритмов или проставления ссылки на конкретные поля таблиц данных, где хранится требуемая информация. При этом входные параметры выборок, тонкости алгоритмов и скорость их работы обычно игнорируется. По существу разработчику говорится, что нужно, а вот способ реализации он определяет сам.

Вторая модель зрелости подразумевает применение SQL-запросов с целью записи шагов выборки данных. Этот способ весьма популярен, так как язык структурированных запросов имеет единую форму записи во всех странах и знаком каждому разработчику. Из множества доступных SQL-запросов в спецификациях преимущественно используется команда SELECT. Проблемы в данном случае остаются лишь в том, как записать логику, требующую сложную формулы расчета. Третья модель зрелости предполагает применение как SQL-запросов, так и команд языков программирования, например, LOOP для задания циклов обработки данных, что дает возможность записать даже самые незаурядные формулы. Однако стоит помнить, вне зависимости от модели зрелости, всегда остается возможность снабжения алгоритмов комментариями.

Литература

1. Степанов Д.Ю. Формирование универсальных требований к пользовательским программам при подготовке спецификации на ABAP-разработку // Актуальные проблемы современной науки. – 2014. – т.78, №4. – с.258-268. – URL: https://stepanovd.com/science/26-article-2014-4-design.
2. Гроff Д.Р., Вайнберг П.Н., Оппель Э.Д. SQL. Полное руководство. – М.: Вильямс, 2014. – 960 с.

Выходные данные статьи

Петров С.В. Использование SQL-запросов и команд языков программирования для описания алгоритмов выборки данных в функциональных спецификациях на разработку ERP-систем // Корпоративные информационные системы. – 2018. – №2 – С. 30-37. – URL: https://corpinfosys.ru/archive/issue-2/140-2018-2-sqlspecifications.

Об авторе

Петров Сергей Владимирович – эксперт по разработке программных решений в банковской, торговой и производственной сферах. Специализируется на языках программирования высокого уровня C++, Java и Transact SQL. Имеет более чем 10-летний опыт разработки приложений. Принимал участие в проектах разработки аналитических, экспертных, биотехнических и корпоративных систем. Электронный адрес: mail@corpinfosys.ru.