К решению проблем миграции данных

Р. Фишбах, И. Лахманн, М. Виннемут (ECS)

При переносе (миграции) "старых данных" в современные информационные системы необходимо быть готовым практически к любым неожиданностям и различного уровня сложностям. При переводе старых данных в формат новых рекомендуется использовать промежуточный формат. Показано, что скриптовый язык Python оптимально приспособлен для трансформации данных в промежуточный формат. Применение языка Python – одна из возможностей повысить эффективность процесса миграции.

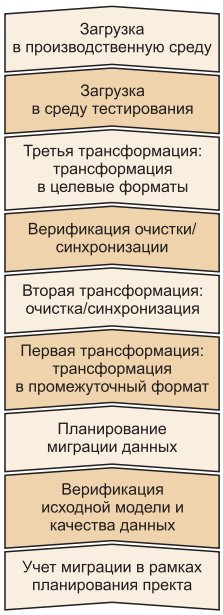
"Как богу удалось в шесть дней создать мир?" – спрашивается в старой IT-шутке. Ответ: "Ему не приходилось иметь дело с уже инсталлированным базисом". Это напоминает IT-экспертам о безнадежности их мечты когда-нибудь начать работу с "белого листа", "с нуля", с самого начала. Там, где сегодня вводятся новые информационные системы, уже существовали старые, функции которых и данные предстоит интегрировать в новые системы для дальнейшего их использования. Сложность этой само собой разумеющейся задачи очень часто недооценивается, объем кроющихся затрат зачастую становится очевидным только на поздних стадиях проекта, когда бюджет уже на исходе и сроки крепко поджимают.

В мире информационных технологий, где начать "с нуля" невозможно, рекомендуется параллельно к введению новой системы планировать миграцию ее данных. Это, как правило, означает их полное обновление, так как использовать данные в том виде, в котором они представлены в "старой" системе, практически невозможно. Причина не только в том, что организация БД в "старой" системе отличается от таковой в новой, но также требуется реорганизация в структуре классов, отношений и атрибутов. Старые данные к тому же подвержены ряду других "недугов", среди которых:

- описание идентичных обстоятельств десятком различных способов;
- различные допустимые значения для одних и тех же атрибутов в различных системах;
- неполнота данных старой системы с точки зрения новой;
- различная степень обработки документов и процессов, хранящихся в различных источниках, что дополняется наличием "мертвых душ", то есть данных в БД, не обрабатывавшихся годами;
- работа с различными форматами данных одной и той же прикладной программы, эксплуатируемой в различных филиалах предприятия. Эти различия возникают на основе недокументированных отклонений от когда-то определенной, но устаревшей схемы.

При миграции "старых данных" необходимо быть готовым практически ко всему. Особенно это касается метаданных – здесь часто исходят из того, что в описанных документах нет отклонений от правил форматирования, индексирования и наименования файлов. Но оригиналы (Word-документы или CAD-чертежи со стандартными текстовыми полями) терпеливы, как бумага. С другой стороны, метаданные, которые удается экстрагировать из старых документов (например, из переписки, заметок-напоминаний, технических чертежей) часто представляют собой ценный клад для предприятия.

Таким образом, систематическое извлечение и "очистка" старых данных, включая проверку на достоверность различных источников, имеет смысл по двум причинам: обеспечение практически безошибочной обработки данных в новой системе и вскрытие до сих пор неиспользованного потенциала. В этой ситуации необходима процедура систематического вскрытия и протоколирования ошибок в данных, позволяющая позднее произвести очистку и изменение структуры данных.



Алгоритм миграции старых данных

Первый шаг к успеху в переносе старых данных состоит в том, что учет затрат на "ремонт" и оценка миграционного риска производятся уже на стадии планирования проекта. В случаях, когда миграция проводится в виде краткосрочной операции накануне сдачи новой системы, она способна взорвать все границы – как бюджетные, так и временные.

Процесс миграции можно наглядно сравнить со строительством моста, при этом модели данных старой системы и новой находятся на разных берегах. В то время как о цели (находящейся там модели данных и их форматах) имеется довольно четкое представление, на исходном "берегу" почва довольно неустойчивая: для понимания имеющейся модели данных часто требуются инвестиции в реконструкцию. Опросы пользователей и программистов ведут к успеху только после многократного повторения цикла: "допущение-тест-новое допущение".

Переводить старые данные непосредственно в формат новых не рекомендуется, так как при этом трудно оценить затраты, необходимые на "очистку". К тому же могут быть потеряны возможности унификации процессов и инструментов. Для решения этой задачи (в ассоциации с мостом) имеет смысл соорудить промежуточную опору: промежуточный формат, в котором при помощи стандартизированных процессов и инструментов аккуратно проведены шаги по очистке о реорганизации данных. От промежуточного формата уже легко перейти к целевому формату.

К промежуточному формату легко применить универсальные ИТ-инструменты (рисунок).

Наиболее важными критериями качества миграции и "ремонта" данных являются, наряду с их безошибочностью, также их воспроизводимость и изменяемость. Первой предпосылкой для этого является процесс документирования с точно определенными шагами и промежуточными результатами, которые можно подвергнуть перепроверке.

Следующее требование заключается в необходимости автоматизировать обработку больших объемов информации за достаточно короткое время. Обрабатывать большие объемы данных вручную, например, используя команду "найти-заменить", означает подвергнуться риску нереконструируемых ошибок, даже без учета того, что скорость таких процедур и их функциональность в части распознавания и замены текстовых шаблонов явно недостаточны.

Таким образом, для последовательной автоматизации работы с данными требуются программируемые, параметризируемые и конфигурируемые инструменты. Достаточно эффективными показали себя скриптовые языки, содержащие все функции обработки текстов и объемный интерфейс ко всем современным ОС. Это сегодня Perl и Python, являющиеся полноценными языками программирования с усложненными типами данных (например, string, список, словарь), а также функциями для распознавания и замены образцов текста.

Язык Python характеризуется доступностью, мобильностью, интегрируемостью с языками C/C++, хорошим качеством проектирования ПО, надежной обработкой текстов и легкостью в освоении. Python доступен для ОС Windows, различных производных ОС UNIX и Linux, для систем, базируемых на ПК BSD, IBM, а также для Embedded Systems с ОС QNX и Vx Works. К этому добавляется поддержка компьютеров Apple и Handheld, управляемых ОС Palm, Epoc или Windows CE. Однотипные интерфейсы к системным функциям позволяют использовать идентичные Python-программы, например, как в окрестности Unix, так и Windows. Python можно интегрировать с приложениями и библиотеками, написанными в C/C++ или Java. Вызов функций (методов) возможен в обоих направлениях.

Язык Python имеет четкую концепцию проектирования качественного ПО, включающую: комбинацию простых семантических понятий, логичный синтакс, разделение кода на семантические и синтактические единицы, образующие ступенчатую структуру, а также мощные стандартные типы данных. Все это позволяет для относительно сложных задач в короткое время создавать сжатые по форме, легко читаемые программы.

Что касается надежной обработки текстов, то Python позволяет так организовать регулярные выражения для анализа и манипуляции текстов (и в Unicode), что написание и доработка кода, работающего с этими выражениями, значительно упрощается. Помимо этого исключаются типичные ошибки, возникающие при изменениях в регулярных выражениях.

Наконец, весомый аргумент: небольшое число и простота правил, прозрачность кода позволяют овладеть Python в значительно более короткое время, чем этого требуют конкурирующие языки программирования. Python по этой причине является не только языком для "одноразовых" скриптов, но и для универсальных инструментов, как, например, конфигурируемый фильтр для таблиц данных, который применим для очистки от устаревшей или разнородной системы обозначений в старых данных. В этом случае необходимые замены данных в каждом поле определяются с помощью таблиц или регулярных выражений.

Фишбах Райнер, Лахманн Ингрид, Виннемут Марио *– сотрудники фирмы Engineering Consulting & Solutions GmbH (г. Ноймаркт, Германия). Контактный телефон (49) 9181 4764-10.*

НОВАЯ КНИГА

Суранова А.Я. LabVIEW 8.20: справочник по функциям

Справочник содержит описание более 800 функций среды проектирования виртуальных приборов (ВП) LabVIEW 8.20. Большое внимание уделено функциям программирования, математики, обработки сигналов, коммуникации, взаимодействия приложений, управления приборами и обмена данными по стандартным интерфейсам. Для большинства Экспресс-ВП приведены окна конфигурирования с переводом их содержимого. Рассмотрены также новые элементы LabVIEW 8.20 – проект, разделяемая переменная, элементы объектно-ориентированного программирования и язык MathScript. Описание функций включает, как правило, их изображения с надписями терминалов на английском и русском языке, а также описание назначения каждого входа/выхода функции. Описание выполнения функций сопровождается примерами их использования.

Справочник может быть полезен широкому кругу специалистов, решающих задачи измерения, обработки или моделирования сигналов.

Москва. Изд-во "ДМК Пресс".
Стр. 536 с иллюстр.

Книгу можно приобрести в Internet-магазине Http:// www.abook.ru Контактный телефон (495) 258-91-94.