Процессный подход к организации структур данных

А.В. Синельников,
к.т.н., доц., sinelnikov@corp.nstu.ru,
А.В. Бачурин,
инж. 2 кат., a.v.bachurin@corp.nstu.ru,
НГТУ, г. Новосибирск

В статье приведен подход к созданию структур данных промышленных систем автоматизации на основе моделей процессов. Недостатком специализированных структур данных является отсутствие гибкости, создание же универсальных структур неизбежно приводит к снижению эффективности обработки сложных запросов при значительных объемах данных. Выход видится в создании гибридных структур, позволяющих как повысить эффективность базовых функций системы, так и добиться универсальности при адаптации системы к условиям реального производства.

The article presents an approach to creating data structures of industrial automation systems based on process models. The disadvantage of specialized data structures is the lack of flexibility, creating the same universal structures inevitably leads to a decrease in the efficiency of processing complex queries with significant amounts of data. The way out is seen in the creation of hybrid structures that allow both to increase the efficiency of the basic functions of the system and to achieve universality in adapting the system to the conditions of real production.

Занимаясь внедрением комплексных систем технической подготовки и управления производством, разработчик всегда сталкивается с определенными трудностями. Основные из них заключаются в том, что на предприятии как правило используется большая номенклатура программных средств, связанная с недостаточной функциональностью какого-либо одного выбранного продукта или продуктов одной компании-производителя. Для разработчика это означает необходимость постоянно наращивать функциональность своих продуктов, а если ее недостаточно, то максимально использовать уже имеющиеся решения других производителей, в том числе и системы, зачастую разработанные самим предприятием, для решения его специфических задач.

Конечно, если на предприятии полностью отсутствуют программные средства (отметим, что такого практически не бывает) или требуемая функциональность полностью перекрывается одной системой (например TechnologiCS [1] для задач конструкторско-технологической подготовки производства (КТПП), см. рис. 1), то, как правило, проблем не возникает и внедрение представляет собой создание новых или конвертацию имеющихся баз данных, а также настройку системы на специфику предприятия. Если же имеющиеся программные средства выходят за рамки функциональности внедряемой системы или попутно с основными решают какие-либо специфические задачи, учитывающие особенности конкретного предприятия (например, PDM [2] и ERP [3], см. рис. 1), здесь внедрение сталкивается с существенными трудностями, связанными с кропотливой разработкой различного рода регламентов, программных интерфейсов или специальных структур для конвертирования и передачи данных из внешних систем и обратно.

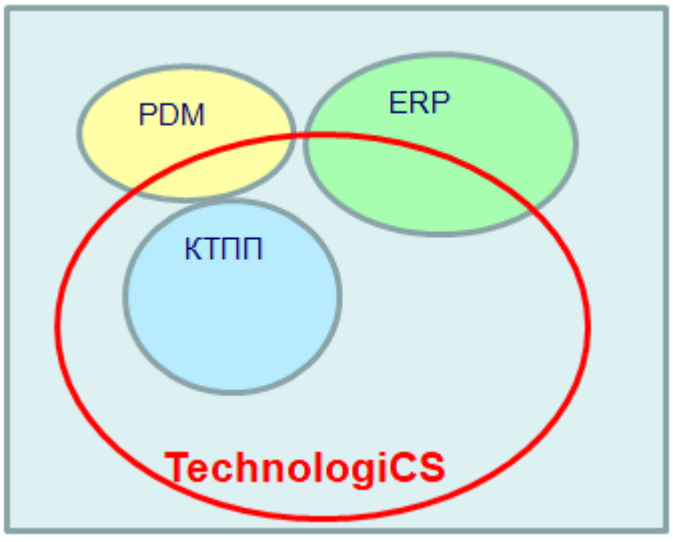

рис.1 – Задачи автоматизации

Преодоление этих трудностей видится в создании информационных систем, использующих процессный подход и позволяющих как развивать собственные решения, так и использовать решения других производителей программного обеспечения.

На сегодняшний день большинство промышленных систем имеют структуры данных, оптимизированные под свой интерфейс пользователя, API [4] и реализацию собственной базовой функциональности. Несомненным преимуществом такого «структурного» подхода является высокая скорость обработки данных при решении задач обеспечивающих базовую функциональность системы. Но во время внедрения зачастую необходима поддержка процессов, функциональность и объем которых, как правило, заранее не известны, и зачастую возникает ситуация, когда для автоматизации процесса необходимо выполнение обработки данных, не заложенное разработчиком в базовую

функциональность системы. Построение системы, выполняющей запросы и обработку данных «в обход» заложенной разработчиком структуры, всегда будет менее эффективно как с точки зрения затрат машинных ресурсов, так и трудоемкости разработки дополнительного программного кода.

Основной трудностью внедрения таких систем является их встраивание в информационную структуру предприятия, что неизбежно приводит к необходимости:

- создания специальных программных конверторов и структур данных для обмена (это могут быть и известные структуры, построенные в соответствии со стандартами ODMA [5], STEP [6], IGES [7] и т.д.);
- разработки специальных сложных регламентов передачи данных, позволяющих оперативно отслеживать изменения, в то же время, поддерживая общую структуру данных в актуальном и непротиворечивом состоянии.

Другой подход заключается в разработке универсальных структур данных и создании на их основе конфигураций для решения базовых задач, как например, это сделано в широко распространенной системе бухгалтерского учета 1С [8]. Данный подход позволяет получить максимальную гибкость при настройке базового функционала обработки данных и существенную универсальность для интеграции с внешними системами за счет развитого низкоуровнего API. Но проблема такого подхода всегда заключалась в скорости работы универсальной структуры со значительными объемами данных, особенно когда необходима их сложная обработка. Например, конструкторское разузлование большого изделия, построение сложного отчета или загрузка/выгрузка данных из внешних систем и обратно.

Каждый из этих подходов имеет свои преимущества, и определенные недостатки. Однако отметим, что для внедрения систем, данные которых построенных на основе «структурного» подхода, практически всегда необходимо участие разработчика.

Выход видится в создании структур данных, позволяющих как повысить эффективность базовых функций системы за счет оптимизации структуры данных и программного кода для их обработки, так и добиться универсальности при адаптации системы к условиям реального производства. Здесь и далее будем называть такие структуры данных «гибридными».

Современные тенденции развития программных средств показывают, что одним из лучших способов решения задач автоматизации является процессный подход, когда задачи формализуются в виде процессов, представляющих собой, как сказано в стандартах на системы менеджмента качества ISO 9000:2001 [9] «совокупность взаимосвязанных или взаимодействующих видов деятельности, которые преобразуют входы в выходы». Будем также помнить, что процессы состоят из процедур, представляющих собой повторяющиеся последовательности действий, приводящих к определенному результату. Таким образом, основные задачи автоматизации можно представить в виде набора «кубиков» (процессов), у которых информационные входы одних связаны с информационными выходами других (рис. 2).

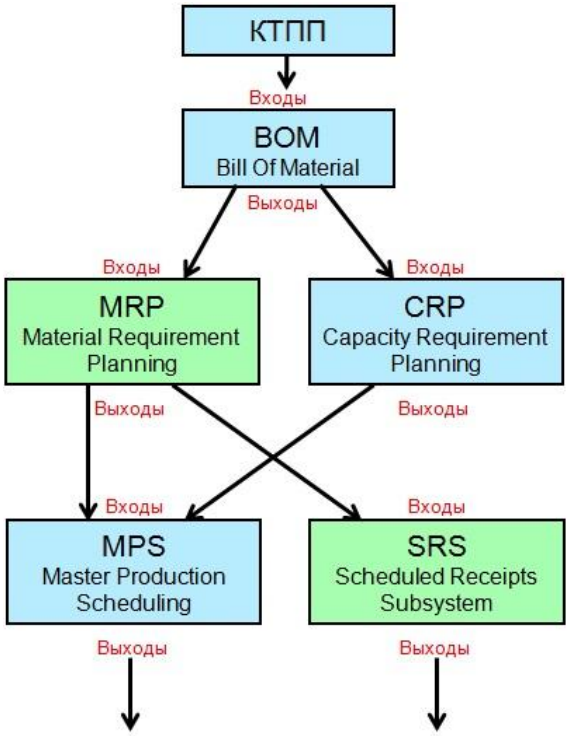

рис.2 – Взаимодействие процессов

Так, например, систему ERP [3] обычно представляют набором, состоящим из следующих процессов:

- Sales and Operation Planning (SOP);
- Master Production Scheduling (MPS);
- Inventory Transaction Subsystem (ITS Управление складом);
- Scheduled Receipts Subsystem (SRS Плановые поставки);
- Material Requirement Planning (MRP);
- Bill of Materials (BOM);
- Capacity Requirement Planning (CRP) и т.д.

В условиях, когда заказчик свои процессы уже разработал, утвердил и запустил в промышленную эксплуатацию, система, спроектированная с учетом процессного подхода, должна обеспечивать гибкую настройку на входы и выходы имеющихся процессов, которые она либо не покрывает функционально, либо по каким-то причинам принято решение поддерживать отдельные процессы уже имеющимся на предприятии программным обеспечением (рис. 3).

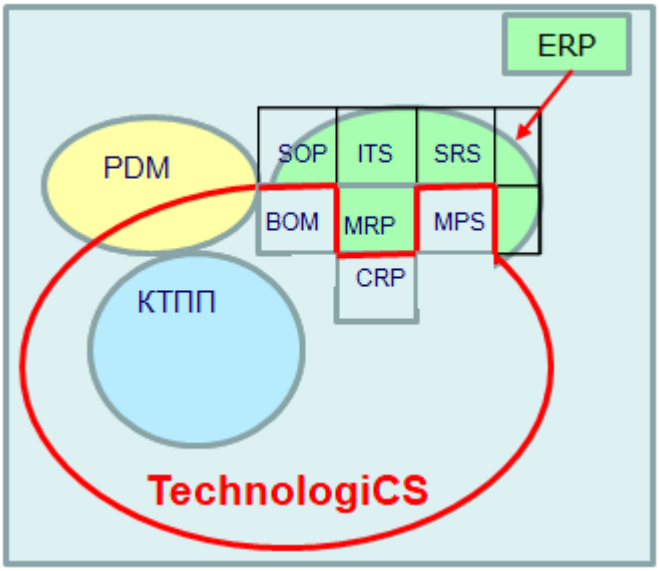

рис.3 – Процессный подход к автоматизации

В качестве примера рассмотрим гибридную организацию данных в системе TechnologiCS. Здесь помимо структур данных системы обеспечивающих базовую функциональность и поддерживающих встроенную обработку данных, пользователь может создавать собственные структуры, пользуясь механизмом «Наборы данных» с помощью специального инструмента «Визуальный построитель запросов» (рис. 4).

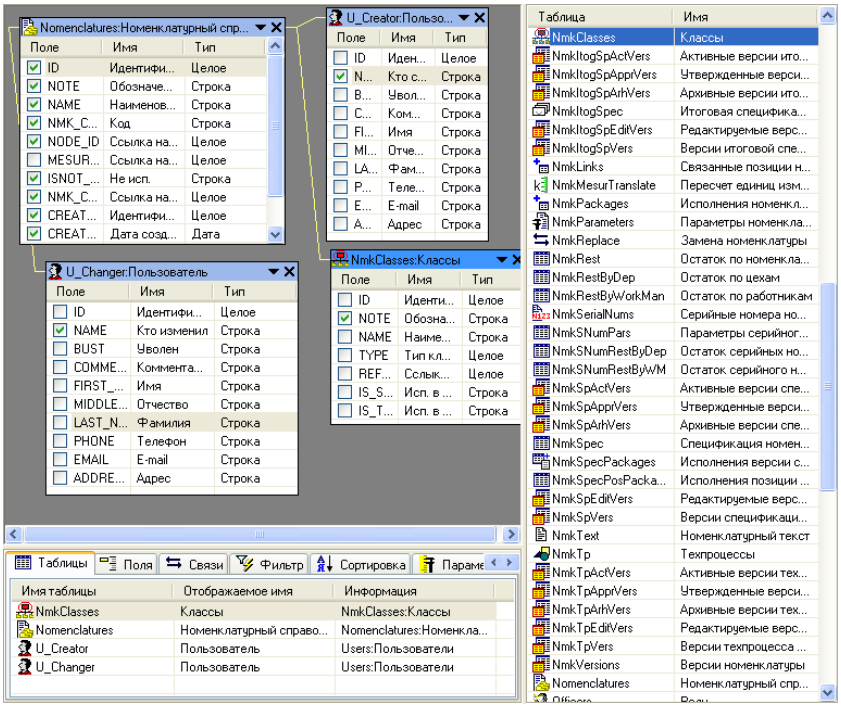

рис.4 - Визуальный построитель запросов TechnologiCS

Причем, в целях обеспечения целостности таких гибридных данных и реализации прав доступа, это не механизм прямого доступа к таблицам SQL-сервера, а механизм доступа к данным с помощью самих объектов API системы. «Визуальный построитель запросов» предназначен для построения пользователем запросов к объектам TechnologiCS в удобном и понятном виде, как это принято в большинстве информационных систем. Конечно, SQL-запрос виртуальный и состоит из объектов TechnologiCS, тем не менее, пользователь всегда может создавать собственные наборы данных и использовать их в визуальных формах, скриптах, отчетах и т.д., расширяя тем самым базовую функциональность системы.

Подобно собственным структурам данных любая программная система содержит также набор окон и встроенных алгоритмов обработки данных, которые вместе образуют интерфейс пользователя. Естественно этот интерфейс является для системы базовым и оптимизирован для работы со встроенными в систему структурами данных. Как правило, степень детализации процессов в таком интерфейсе максимальная, да и встроенные функции сгруппированы не всегда так, как это требуется конечному пользователю. Отсюда и возникают трудности, связанные с чрезмерно

перегруженным интерфейсом и необходимостью открывать большое количество окон для реализации, казалось бы, самых простых функций.

Для преодоления этих трудностей в TechnologiCS разработан специальный механизм, позволяющий пользователю создавать для работы собственные экранные формы, задавая им с помощью TechnologiCS API собственные алгоритмы обработки данных. Этот механизм называется «Редактор форм» и состоит из «Редактора кода» (рис. 5) и «Дизайнера форм» (рис. 6).

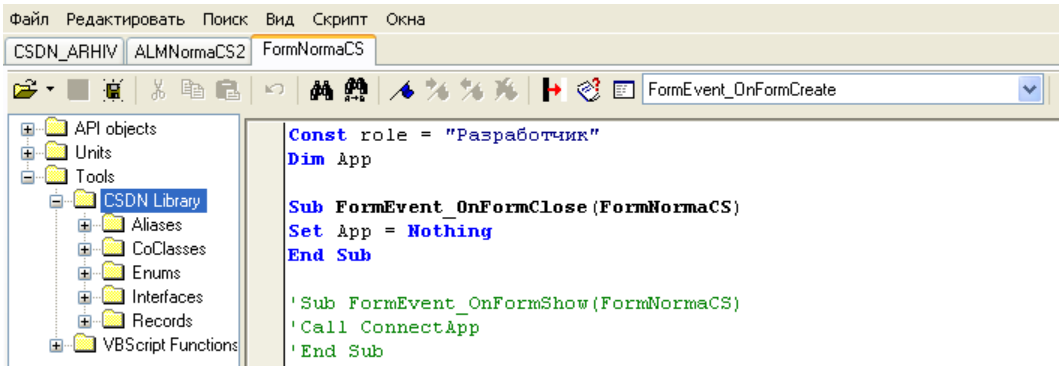

рис. 5 - Редактор кода TechnologiCS

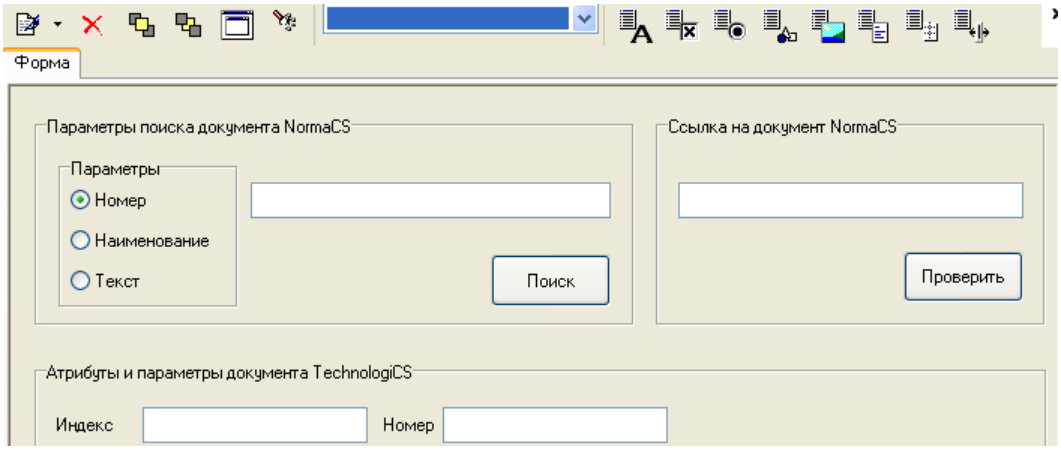

рис. 6 - Дизайнер форм TechnologiCS

«Редактор кода» представляет собой среду разработки, предназначенную для создания и редактирования макросов на языке VBScript, и является полнофункциональным редактором текста. При работе с редактором доступны операции с блоками текста, функции поиска и замены, цветовое выделение синтаксических элементов программных модулей и т.д. «Дизайнер форм», в свою очередь, предназначен для визуального проектирования элементов формы. Таким образом, пользователь может самостоятельно не только разрабатывать собственные структуры данных, но и создавать свои формы для ввода и отображения информации.

Для модификации стандартного и создания собственного интерфейса пользователя существует механизм, называемый «Дизайнер интерфейсов» (рис. 7).

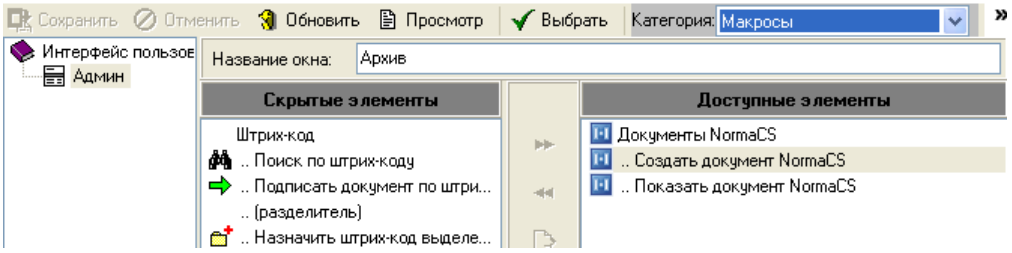

рис. 7 - Дизайнер интерфейсов TechnologiCS

Этот механизм позволяет сделать систему более гибкой, предоставляя пользователю возможность настраивать конкретные рабочие места, значительно упрощать интерфейс, заданный разработчиком, создавать свои кнопки, экранные формы и макросы.

Таким образом, пользователь имеет возможность самостоятельно разрабатывать и поддерживать собственные процессы, формируя автоматизированные рабочие места различной функциональности, и что очень важно – не прибегая при этом к услугам разработчика.

Данный подход имеет ряд преимуществ:

- пользователь может разрабатывать собственные структуры данных отсутствующие в системе, оставаясь в рамках единой базы данных, что автоматически будет обеспечивать их целостность и непротиворечивость;
- поскольку такие структуры основаны не на таблицах базы данных а на объектах системы, они будут поддерживаться даже если разработчик будет трансформировать внутреннюю структуру данных;

- в будущем, в случае изменения объёма автоматизируемых процессов, пользователь всегда сможет адаптировать свои структуры данных, не прибегая к услугам разработчика.

Литература

1. TechnologiCS [Электронный ресурс]. URL: http://www.technologics.ru/
2. Н. Дубова, И. Островская. Словарь терминов по PDM. «Открытые системы», 1997, №3
3. Leon, Alexis. Enterprise Resource Planning. — 2nd. — New Dehli: McGraw-Hill, 2008. —500 с. — ISBN 978-0-07065680-2.P
4. API [Электронный ресурс]. URL: http://ru-wiki.org/wiki/API
5. ODMA. Open Document Management API [Электронный ресурс]. URL: http://odma.info/
6. The STEP Standard - ISO 10303 [Электронный ресурс]. URL: http://www.steptools.com/stds/step/
7. IGES 5.3 (ANSI-1996) in PDF from US Product Data Association (USPRO)
8. 1С:Бухгалтерия 8. Учебная версия.-5-е изд.-М.: ООО «1С Паблишинг», 2010.-594с: ил.+1CD
9. ГОСТ Р ИСО 9000-2001 Системы менеджмента качества. Основные положения и словарь. М.: Изд-во стандартов, 2001.