А.В. Крючков
(ФГКУ "В/ч 44239"; e-mail: hook66@list.ru)

ИЕРАРХИЧЕСКИЕ ТРЕБОВАНИЯ К СПЕЦИАЛЬНОМУ ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ПРЕДПРИЯТИЕМ

Проведён анализ требований к специальному программному обеспечению автоматизированной системы управления предприятием, включая решение проблем безопасности. Предлагаются концептуальные основы АСУП.

Ключевые слова: программное обеспечение, автоматизированная система управления предприятием.

A.V. Kruchkov

HIERATICAL REQUIREMENTS TO SPECIAL SOFTWARE OF ENTERPRISE AUTOMATION MANAGEMENT SYSTEMS

Analysis of requirements to special software of enterprise automation management systems including solving problems of security. Proposed conceptual basis of enterprise automation management systems.

Key words: software, enterprise automation management systems.

Статья поступила 20 декабря 2014 г.

Для обеспечения техносферной безопасности на производстве необходимо создавать большое количество программ учёта информации по эффективности проводимых в этой области мероприятий.

Специальное программное обеспечение (СПО) является составной частью ***программного обеспечения (ПО) автоматизированной системы управления предприятием (АСУП)*** и требует для своего синтеза определённых информационных технологий, в состав которых входят ***системы программирования (СП)*** и ***языки программирования (ЯП)***. Учитывая, что СПО для крупной АСУП состоит из СПО для отдельных ***автоматизированных рабочих мест (АРМ)***, будем называть СПО, созданное для одного АРМ, ***единичной программной системой (ЕПС)***.

Многие методы синтеза СПО выстраивались вокруг программиста в виде поиска множества ***базовых элементов программ (БЭП)***. Ориентация на универсальность применения СП и ЯП привела к тому, что по мере универсализации компонент и библиотек ПО происходил процесс усложнения создаваемых с их помощью ЕПС, а также увеличивалось количество классов, их свойств и методов ([1] с.73).

Поэтому, прежде чем приступать к определению класса задач автоматизации в АСУП, рассмотрим несколько цитат из технической литературы, касающихся понятий программы и процесса программирования, которые могут быть полезны при формализации процесса.

"Поиски систематических процедур перевода записей алгоритмов в программы и извлечения программ из условия задачи и дополнительной информации составляют предмет автоматизации программирования". "Программа – описание некоторой задачи, рассчитанная на воспроизведение автоматом" ([2], с. 15). "Программа – символьное описание способа взаимодействия с символьным окружением" ([2], с. 16). "Чтобы разработать программу нужно специфицировать задачу автоматизации" ([3]).

"Диаграммы потоков данных (Data Flow Diagrams – DFD) – представляют собой иерархию функциональных процессов, связанных потоками данных. Цель такого представления – продемонстрировать, как каждый процесс преобразует входные данные в выходные…" ([5]). "Спецификация является конечной вершиной иерархии DFD… Фактически спецификации представляют собой описания алгоритмов задач, выполняемых процессами" ([4], с. 145, 146).

"Автоматизация – это процесс обмена информацией между процессами в операционной системе …, с помощью которого одна прикладная программа может управлять другой" ([6], с. 553). "Программная архитектура – это описание подсистем и компонентов программной системы, а также отношений между ними… Она являет собой результат процесса разработки программного обеспечения" ([1], с. 39). "Предметная область – область знаний или деятельности, характеризующаяся определённым набором понятий и терминологией, которыми пользуются специалисты в данной области" ([1], с. 45).

"Рамки предметной области задаются в процессе определения предметной области (domain scoping), который в свою очередь является одной из составляющих предметной области" ([1], с. 49)."Линейка программных продуктов – группа продуктов с общим регулируемым набором характеристик, отвечающих требованиям сегмента рынка" ([7]). "Ведение линейки продуктов подразумевает сопровождение фонда базовых средств, обеспечивающее возможность регенерации любого существующего члена линейки" ([8], с. 451). "Предельным случаем автоматизации был бы, возможно, такой, когда программисту достаточно было сформировать некоторый начальный … замысел задачи, который он хотел бы запрограммировать" ([2], с. 13). "Требование – это условие, которому должна удовлетворять система, или свойство, которым она должна обладать, чтобы удовлетворить потребность пользователя в решении некоторой задачи и удовлетворить требования контракта, стандарта или спецификации" ([9]).

Итак, если следовать логике цитат, решение задачи поиска классов задач автоматизации в процессе синтеза СПО АСУП лежит в плоскости определения библиотеки БЭП и простых правил построения ЕПС из её элементов. А структура спецификаций требований к ЕПС в виде дерева должна предлагать однозначно интерпретируемый в любой предметной области метод такого построения. Каждая ЕПС содержит дерево требований, вид которого зависит от предметной области и не зависит от инструментального средства и техники реализации ЕПС программистом.

Для уточнения возможности построения единой концептуальной схемы семейства ЕПС (линейки программных продуктов) необходимо определить ***базовый класс задач автоматизации (БКЗА)***. Это позволит определить горизонтальную предметную область ([1]), для которой будет построена близкая к универсальной концептуальная доменная модель многократного применения. Такая предметная область будет по сути близка к понятию идеальной абстрактной предметной области.

Так как ни один из указанных технических источников не даёт такого определения, уточним его самостоятельно. Класс задач автоматизации – это набор требований к ЕПС, содержащий наборы требований, как к предметным областям, так и к программам, реализующим в рамках АРМ ЕПС с нужной функциональностью.

Соответственно БКЗА – это класс задач автоматизации, который содержит общие для множества различных предметных областей, охватываемых СПО крупной АСУП, наборы требований, как к предметным областям (прежде всего к структурам их данных), так и к программам, реализующим ЕПС в рамках АРМ АСУП.

Иерархия задач и функций в рамках крупной АСУП и уточнение функциональности АРМ в АСУП позволяют говорить о классе задач автоматизации в ней как о БКЗА. Будем использовать терминологию ***объектно-ориентированного программирования (ООП)*** для точного определения БКЗА.

В соответствии с [5], ***класс объектов*** – логическая группа объектов, обладающих сходными свойствами и методами. ***Тип объектов*** – реализация класса на конкретном ЯП. Используемый в программе конкретный объект – экземпляр класса. Совокупность всех экземпляров класса – экстент объекта. Группа классов – это библиотека.

Сам БКЗА – абстрактный класс. Этим обобщением может быть как ЕПС, так и её информационная схема. Всякий класс, в соответствии с этой терминологией, обладает набором свойств (атрибутов), методов (вызываемых подпрограмм и функций), связей (иерархии классов) и состояний (исключительных ситуаций). При определении БКЗА в рамках стандарта IDEF1X ([10]) необходимо говорить о сущностях, атрибутах и связях.

Ещё одним важным свойством БКЗА является то, что он является приложением БД и поэтому имеет свою концептуальную схему, что позволяет говорить о возможном наличии в ней одной центральной (главной, основной) таблицы БД. Важным (но необязательным) его свойством является то, что ЕПС на основе БКЗА является в подавляющем большинстве случаев локально-ориентированной системой. Во всяком случае, на первом этапе формулирования требований к БКЗА удобно использовать такое ограничение. Впоследствии, при развитии исследований в данном направлении, оно может быть снято. Поэтому наиболее предпочтительным инструментальным средством реализа-

ции ЕПС на основе БКЗА является ***система управления базами данных (СУБД)***. Учитывая ограничение локальности ЕПС, можно сказать, что БКЗА не использует технологию "клиент – сервер" и хранит необходимые таблицы БД на ПЭВМ локально.

Интерфейс БКЗА является базовым интерфейсом АРМ СПО в АСУП. В его состав входят система меню, система сообщений, система помощи, группы экранных форм, группы отчётов. Перечисленные системы составляют группу интерфейсных свойств БКЗА. Все эти свойства являются множественными. Каждое из них в отдельности представляет собой многоуровневый граф, вершинами которого являются как сами элементы, входящие в данные системы, так и их свойства.

Методами БКЗА являются функции СПО, которые будут вызываться через систему меню.

Таким образом, имея БКЗА и зная его свойства и методы, можно выделить среди них те, которые не будут зависеть от предметной области, ЯП или СУБД, а также от взглядов и квалификации программиста.

В терминах ООП, каждый класс должен иметь свою структуру данных, интерфейс и реализацию. В условиях данного исследования, ЕПС в СПО конкретного АРМ – экземпляр абстрактного класса БКЗА. Все СПО АСУП является коллекцией объектов. Структура данных абстрактного класса БКЗА соответствует концептуальной схеме данных в ДИСП ЕПС. Набор систем, описанных выше, составляет интерфейс БКЗА. То, как объект поддерживает интерфейс (программный код на ЯП), принято называть ***реализацией***.

При конкретизации БКЗА в условиях конкретного АРМ для ЕПС необходимо:

- установить возможность необходимой и достаточной унификации внутри БКЗА объектов СПО (создать структуру для хранения информации);
- сформулировать правила расширения функциональности и модификации структур данных внутри БКЗА при изменении экземпляра объекта СПО (порядок её наполнения);
- определить уровни соответствия конкретных объектов СПО АСУП предлагаемому унифицированному абстрактному классу.

Подобно известным объектным моделям, рассматриваемый БКЗА не зависит от реализации. Он может быть реализован на любом из доступных ныне инструментальных средств. Вместе с ним могут быть реализованы дополнительные или сервисные программы, обеспечивающие настройку его конфигурации на конкретную предметную область. Причём при определённом уровне реализации данных средств такая настройка станет возможна специалистом, уровень квалификации которого значительно ниже той квалификации, которая требуется для программировании на ЯВУ.

На рис. 1 приведена концептуальная схема ЕПС без учёта системы сообщений, системы помощи. Структура информации представлена с использованием нотации IDEF1X. Атрибуты сущностей в данной схеме не выделены.

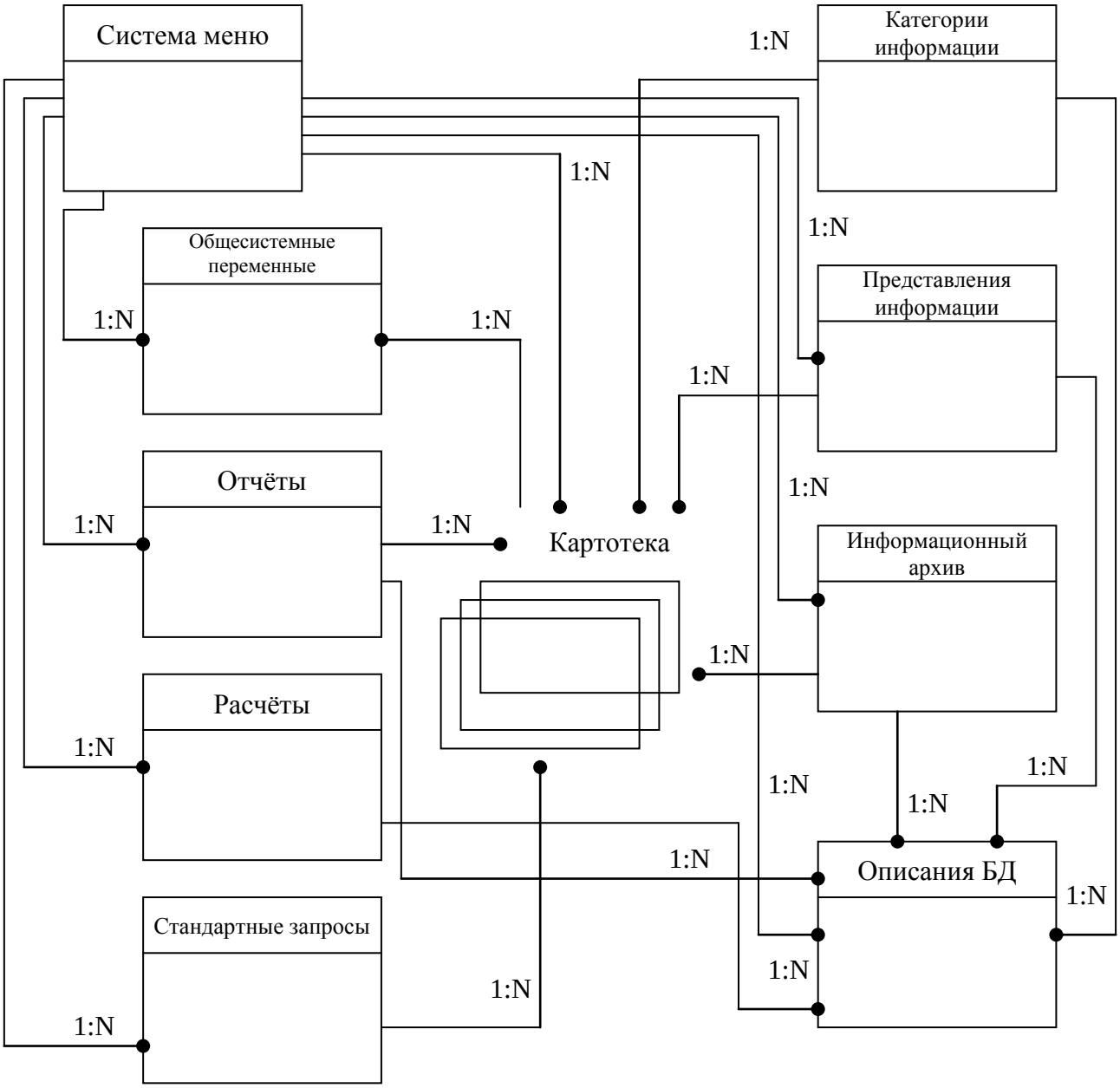

Рис. 1. Концептуальная схема ЕПС в СПО АСУП

Сущность "карточка учёта" рассмотрена более подробно на рис. 2. На данном рисунке выделены структурные компоненты сущности ИЕХ "карточка учёта" концептуальной схемы ЕПС. Сущность "классификаторы" представляет собой набор таблиц БД со структурой: № п/п, наименование. Она представляет собой набор доменов. Это наиболее простой способ их описания. В реальных системах, реализующих данную ДИСП, для сущности "классификатор" следует предусмотреть возможность введения дополнительных полей в структуре (например, для построения иерархий).

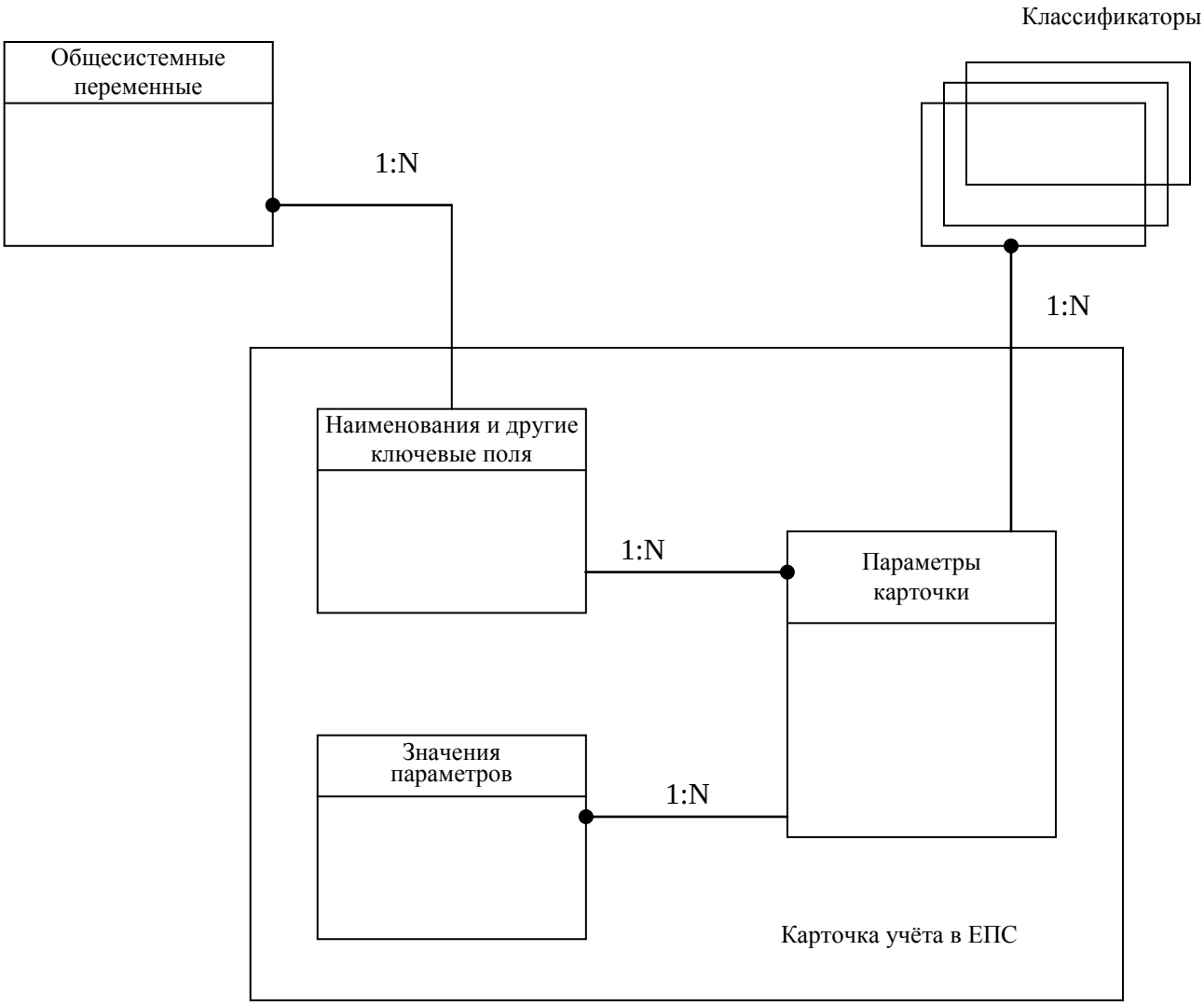

Рис. 2. Структура карточки учёта (ИЕХ) в картотеке ЕПС СПО

Что же касается сущности "карточка учёта в ЕПС", то её детальное описание в терминах нотации IDEF1X приводится на рис. 3. На этом рисунке данная сущность изображена как отношение "2 к 1", причём сущность "значения параметров" показана как слабая с двумя внешними ключами.

В сущности "карточка" на этом рисунке в качестве первичного ключа указан "код карточки", а в качестве атрибутов указаны ключи-кандидаты. Это могут быть, например, фамилии сотрудников в качестве первого из ключей и их имена и отчества в качестве частей второго ключа. Это могут быть также номера войсковых частей – в качестве первого из ключей и их сокращённые наименования – в качестве второго. Это могут быть первичные номера документов в качестве первого из ключей и их тип, адресат и дата отправления в качестве второго. Эти ключи могут быть связаны с доменами.

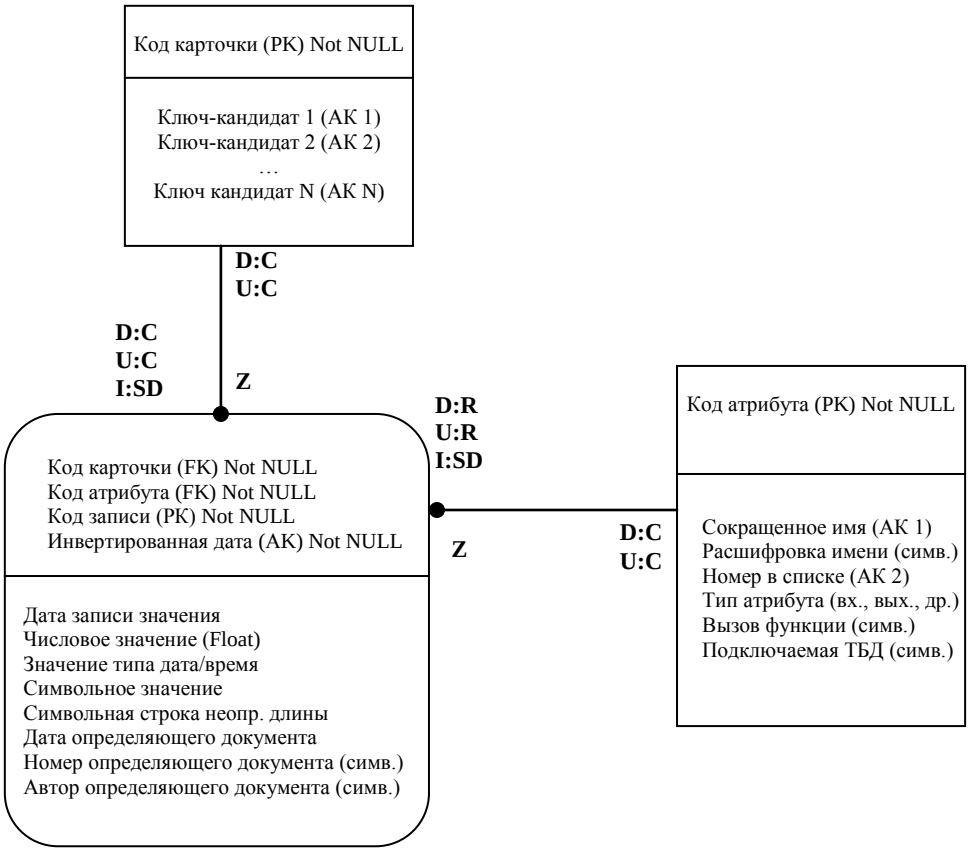

Рис. 3. Модель (схема) отношения 2:1 сущности "карточка учёта в ЕПС" ИС в терминах нотации IDEF1X с процедурами ограничения ссылочной целостности (отношение для БЗКА)

Связь данных ключей с доменами может быть интерпретирована в пользу существования доменно-ключевой нормальной формы [10] данного отношения, то есть формы отношения, имеющей наибольшую степень нормализации. Однако вывод об этом невозможно сделать ввиду того, что строгое определение принадлежности какого-либо отношения к данному типу формы отсутствует. В каждом конкретном случае применения отношения к предметной области в рамках БКЗА прежде, чем сделать вывод о его состоянии, необходимо проанализировать, являются ли условия, наложенные на данное отношение, следствием доменов и ключей. К доменам будут принадлежать те атрибуты базовой сущности "карточка учёта", которые необходимо вводить из списка заранее определённых или определяемых в процессе работы значений.

Для сущности "параметр карточки" первичным ключом является код атрибута (параметра) карточки. Помимо него, в данной сущности в качестве альтернативных ключей указываются: сокращённое имя параметра и номер в списке. Номер в списке служит для определения порядка сортировки уже введённых в список параметров карточки атрибутов. Вызываемые процедуры и подключаемая ТБД используются для определения связи отдельных парамет-

ров отношения "карточка" с классификаторами. Тип атрибута указывает, в каких целях в процессе работы программы данный атрибут должен использоваться. Он может быть входным, выходным или промежуточным.

Сущность "значения параметров" определена как слабая идентификационно-зависимая сущность, поэтому она имеет два внешних ключа: код карточки и код атрибута. В качестве ключей используются также код значения как первичный ключ и инвертированная дата как альтернативный ключ. Инвертированная дата представляет собой символьную строку, записанную так:

ГГГГММДД,

где ГГГГ – год;
ММ – месяц;
ДД – день.

Она используется для поиска и сортировки. Такая форма записи не зависит от типов данных, используемых в конкретном инструментальном средстве (СУБД, ЯП, СП). Необходимость в ней возникает в связи с тем, что многие атрибуты карточек должны иметь более одного значения. Её дополняет дата записи, обработанный вариант которой и представляет инвертированная дата.

В состав данной сущности включены также возможные значения параметра карточки различных типов:

- числовое;
- символьное заданной и неопределённой длины;
- несколько полей типа "дата" или "дата + время" (для задания временных ограничений значения);
- ряд атрибутов, обосновавших включение данного атрибута в таблицы БД (дата, номер и автор определившего данное значение документа).

Если не вдаваться в детализацию предметной области до уровня объектов, можно увидеть, что многие сходные характеристики в характеристических моделях разных АРМ АСУП реализуются при помощи сходных групп компонентов. Развитие этого процесса в сторону укрупнения компонентов путём их группировки привело к формированию нового направления в развитии инструментальных средств разработки приложений – ***аспектно-ориентированному программированию (АОП)***. Оно основано на понятии "аспекта", трактуемого, условно говоря, как группа компонентов, обладающих собственным поведением и собственным интерфейсом. Реализованный в СПО "аспект", созданный для дальнейшего использования, как кубик из конструктора, конечно же, представляет собой предварительно написанный программный код. Но совокупность требований к его реализации уже больше походит на математическую модель, на которой отображается предметная область. Наличие большого числа реализаций и разных версий "аспектов", также как и компонентов, на которых отображается предметная область, её функциональные характеристики и элементы, существенно затрудняют их повторное использование.

Помимо этого, при расширении "аспекта" как понятия АОП до АРМ, возникает необходимость вводить разветвлённую конфигурацию приложения при его использовании в различных конкретных условиях. Поэтому для БКЗА потребуется создать математическую модель укрупнённой характеристической модели предметной области и связать её с обобщённой информационной моделью данных приложения. В дальнейшем для информационной схемы приложения необходимо будет определить паттерн (образец применения) реализации для среднего пользователя, содержащий необходимые семантические элементы интерфейса приложения. Средним пользователем [11] принято считать человека, которому, согласно способу реализации его полномочий в СПО, уже не нужна разветвлённая система помощи и поддержки его действий при работе с СПО АРМ в ИС, но, вместе с тем, он ещё недостаточно квалифицирован для работы с ним совсем без неё.

Информационную модель СПО можно представить в виде последовательности ***информационных единиц хранения (ИЕХ)***, которые составляют ***дерево информационной схемы приложения (ДИСП)***, являющееся математическим отображением характеристических моделей предметных областей на пространство их реализаций средствами различных систем управления базами данных. Как показывает практика, последовательность ИЕХ может иметь несколько явно выделяющихся уровней.

Первый уровень – уровень сортировки ИЕХ по группам на основе принятой в данной предметной области классификации. Этот уровень сложно формализовать из-за того, что разные предметные области изначально имеют различную структуру. Будем называть его ***групповым***.

Далее идёт ***уровень единичной группы***. Он соответствует порядку поиска пользователем ИЕХ в данной группе. На этом уровне последовательность ИЕХ должна иметь один или несколько ***поисковых идентификаторов***, позволяющих получить доступ к элементам дерева ИЕХ. Как правило, это названия организаций, фамилии и имена сотрудников, коды единиц хранения и т.п. смысловые термины. Будем называть его ***идентификационным***.

На ***третьем уровне*** определяется структура дерева ИЕХ посредством описания характеристик узлов и связей между ними. На этом уровне в ИЕХ определяется код и название узла графа (вершины дерева), связанные с его обработкой процедуры. Этот уровень уместно назвать ***параметрическим***.

На ***четвёртом уровне*** определяется формат физического хранения данных, связанных с узлом ИЕХ. На этом уровне могут быть использованы все допустимые в языке определения данных: типы, даты записи в базу данных информации и другие характеристики описательного характера, используемые обычно для подтверждения правильности записи. Будем называть его ***уровнем значений***.

На ***пятом уровне*** определяются связанные с ИЕХ кодификаторы и классификаторы. Они частично соответствуют понятию "домен" в теории баз данных и определяют, какая информация будет заноситься о конкретном узле в поле записи (обычно – числовое или символьное). Назовём его ***доменным уровнем,*** по аналогии с понятием домена в теории баз данных.

Такая структура ИЕХ позволяет сделать вывод о возможности обобщения ДИСП на втором, третьем и четвёртом уровнях ИЕХ. Обобщённая модель ИЕХ в данном случае будет содержать три элемента дерева: неизменяемую верхнюю, параметрическую и неизменяемую нижнюю. Верхняя часть служит для идентификации ИЕХ. При помощи параметрической части происходит настройка дерева ИЕХ и его вершин. Нижняя часть хранит данные.

Подобная структура даёт возможность, с одной стороны, настраивать готовые компоненты, содержащие, например, функциональные характеристики ввода, корректировки, поиска и просмотра на нужные предметные области, а с другой стороны, готовить компоненты с требуемой функциональностью отдельно от создания СПО.

Таким образом, рассмотренные иерархические требования и структуры данных "идеальной" предметной (не имеющей чёткого приложения, но имеющей чёткую структуру) области составляют костяк возможной конфигурации требований для любого из АРМ в крупной АСУП. Это позволяет заранее определять требования к ЕПС в составе АСУП, настраивая их с незначительными трудоёмкостями и финансовыми затратами на конкретные приложения. Для такой структуры данных возможна разработка диалогового приложения, позволяющего программировать приложения людям без квалификации программиста, а также разработка библиотек компонентов, содержащих реализации больших функционально-законченных блоков функций ЕПС.

Литература

1. ***Чарнецки К., Айзенкер У.*** Порождающее программирование. Методы, инструменты, применение. Пер. с англ. С.-Пб.: Питер, 2005.
2. ***Ильин В.Д.*** Система порождения программ. М.: Наука, 1989.
3. ***Агафонов В.Н.*** Спецификация программ – понятийные средства и их организация. Новосибирск: Наука, 1987.
4. ***Вендров А.М.*** Проектирование программного обеспечения экономических информационных систем. М.: Финансы и статистика, 2005.
5. ***Калашян А.Н., Кальянов Г.Н.*** Структурные модели бизнеса: DFD-технологии. М.: Финансы и статистика, 2003.
6. ***Избачков Ю., Петров В.*** Информационные системы: учебник для вузов. С.-Пб.: Питер, 2005.
7. ***Басс Л., Клементс П., Кацман Р.*** Архитектура программного обеспечения на практике. Пер. с англ. С.-Пб.: Питер, 2006.
8. ***Лефингуэлл Д., Уидриг Д.*** Принципы работы с требованиями к программному обеспечению. Унифицированный подход. Пер. с англ. М.: Вильямс, 2002.
9. ***Крёнке Д.*** Теория и практика построения баз данных. Пер. с англ. С.-Пб.: Питер, 2005.
10. ***Константайн Л., Локвуд Л.*** Разработка программного обеспечения. Пер. с англ. С.-Пб.: Питер, 2004 г.