УДК 681.58.011.56
В.Г. Сухоребрый, А.С. Гристан, Д.В. Джулгаков

Базы данных сигналов в среде СИНТАР 2007

Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ»

СИНТАР 2007 [1] – это среда разработки контроллерного ПО, ориентированная на АСУ крупных промышленных объектов. Примером применения СИНТАР 2007 могут служить АСУ атомных электростанций.

Для АСУ такого класса характерно использование большого количества шкафов управления (ШУ), операторских рабочих станций разных уровней, двойное и тройное резервирование на разных уровнях. Каждый ШУ может включать в себя один или несколько контроллеров и несколько десятков устройств связи с датчиками и исполнительными механизмами. Все это аппаратное многообразие связывает воедино разветвленная локальная сеть с весьма интенсивным трафиком. Данные, передаваемые этой сетью, называются *сигналами*.

Любой сигнал, помимо текущего значения, характеризуется рядом *свойств*, которые используются алгоритмами обработки и отображения сигналов. Например, входной аналоговый сигнал может описываться такими группами свойств, как идентификация, аварийные и предупредительные границы значения, характеристики датчика, атрибуты воспроизведения системами отображения, параметры резервирования, связь с внешними системами и т.п. Сигналы с одинаковым набором свойств относятся к одному *типу*.

Потребителями сигналов (и их свойств) являются многочисленные компоненты ПО АСУ – программы первичной обработки сигналов, автоматического регулирования и управления, размещенные на контроллерах ШУ различных функциональных подсистем, программы отображения и операторского управления рабочих станций, программы накопления и обработки архивов технологических данных, журналов событий, статобработки и т.д. Причем сигналы используюто на этапе не только эксплуатации, но и проектирования и разработки АСУ. Поэтому АСУ рассматриваемого класса хранит описания всех сигналов, применяемых в *базе данных сигналов.* Как правило, используется база данных (БД) с клиент-серверной архитектурой, обеспечивающей цивилизованный доступ к информации всем потребителям.

Специфические требования прикладной области

1. **Большой объем данных**. Представление об объеме информации, которую необходимо ввести в БД, дают следующие цифры. Количество ШУ может достигать 100, количество сигналов, используемых одним ШУ, - 1000 - 2000. Объем свойств сигнала зависит от его типа и колеблется от 4-5 до 20-30 числовых или строковых значений. Ясно, что наряду с обычными приемами ускорения ввода данных (копирование, итеративное заполнение по шаблону, значения умолчания) необходимы более эффективные средства автоматизации.
2. **Обеспечение коллективной разработки.** Большой объем работы и желание выполнить ее в разумные сроки приводят к распараллеливанию разработки между несколькими коллективами или организациями разработчиков.

Чтобы разделить работы и разграничить ответственность, АСУ разбивается на части. База данных сигналов также разбивается на части, как правило, соответствующие отдельным ШУ. Среда разработки озадачена организацией взаимодействия частей и объединения их в полную АСУ на заключительных этапах разработки.

3. **Достоверность и актуальность данных**. Необходимы средства контроля вводимых данных, индивидуальные для отдельных свойств и управляемые пользователем. Актуальность обычно обеспечивается запретом дублирования информации, однако это вступает в противоречие с требованием обеспечить коллективную разработку (создание надежной общей базы данных для нескольких организаций, особенно, если они расположены в разных городах, - довольно дорогое удовольствие). Дублирование информации возникает также из-за специализации ШУ, которая влечет за собой увеличение обмена сигналами между ними.
4. **Оперативное изменение типов и свойств сигналов.** Поскольку базой данных сигналов пользуются многочисленные компоненты ПО АСУ, выполняющие разные функции, на практике оказывается, что ее структура подвержена наиболее частым изменениям. В идеале эти изменения должны стать прерогативой пользователя, а фиксированная структура БД противоречит этому требованию.
5. **Наличие баз данных с разной структурой.** Предыдущее требование приводит к необходимости обслуживания одной версией среды баз данных различной структуры. Они неизбежно будут появляться, во-первых, благодаря специализации частей АСУ и, во-вторых, из-за необходимости сопровождать АСУ, находящиеся на разных стадиях развития.
6. **Массовый выбор сигналов для групповых операций.** Существует ряд операций, в которых участвуют большие группы сигналов, например, импорт сигналов в программный проект, экспорт в другие базы данных, размещение на сетевых линиях связи. Наряду с индивидуальным «посигнальным» выбором необходимо автоматизированное средство массового выбора сигналов.
7. **Ограничение доступа и регистрация изменений.** АСУ рассматриваемого класса управляют системами с критическими требованиями к безопасности. Наряду со структурными, алгоритмическими, организационными и другими мерами безопасности это накладывает отпечаток и на среду разработки: она должна защитить данные от несанкционированного доступа и регистрировать любые изменения, проводимые в базе данных сигналов.

Определение структуры БД

Очевидным следствием требования связанного с определением и изменением структуры БД пользователем, стало разбиение пользовательского интерфейса на две части: определение структуры (конструктор типов) и ввод/редактирование сигналов (редактор сигналов).

Конструктор типов выполняет такие функции:

- создание/удаление типов сигналов;
- определение свойств сигналов каждого типа;
- определение контролируемых ограничений;

- определение автогенерируемых сигналов для данного типа.

На рис. 1 показано окно конструктора типов. Левая панель содержит типы сигналов, средняя – свойства выбранного типа, правая – ограничения выбранного типа.

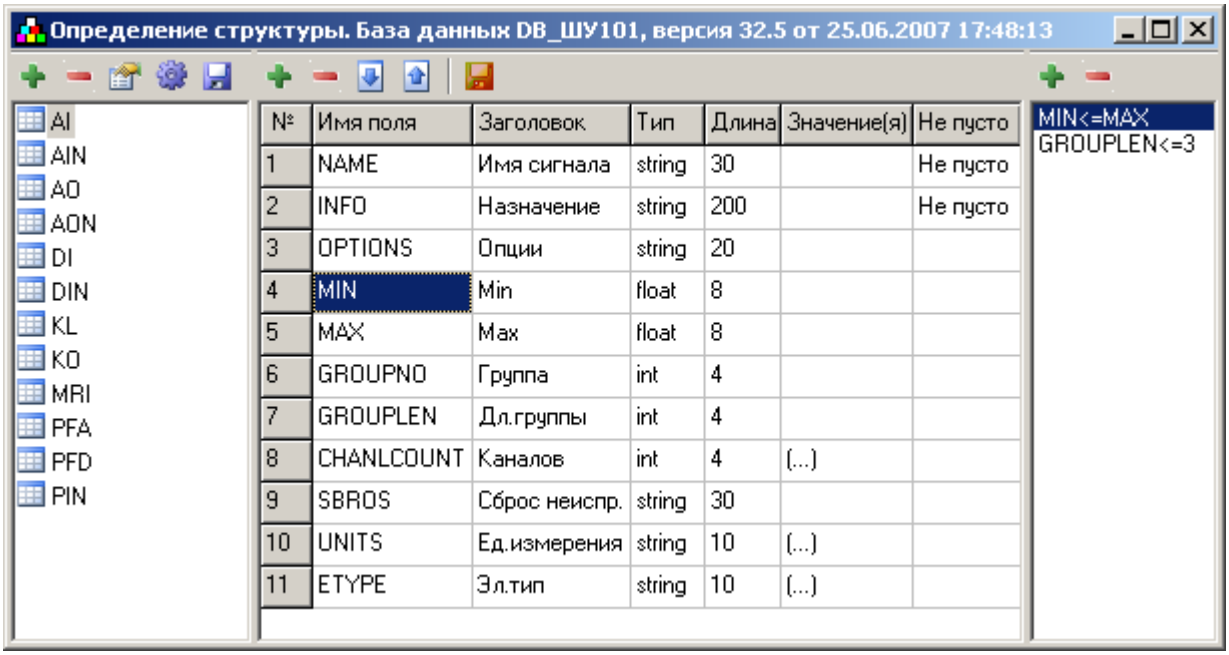

Рис. 1. Окно конструктора типов

Каждое свойство имеет ряд атрибутов и требует заполнения строки средней панели. Атрибут «Значение(я)» позволяет ввести одно значение умолчания или список допустимых значений (возможно, древовидный), который в дальнейшем будет предъявляться редактором для выбора при вводе данного свойства.

Поскольку многие свойства повторяются у разных типов сигналов, для сокращения объема ввода существует возможность в любой момент сохранить уже набранные свойства, и они будут появляться автоматически при создании нового типа сигналов; лишние можно будет удалить.

Введенные данные конструктор преобразует в запросы на языке DDL, выполнение которых формирует БД с заданной структурой и свойствами.

Управление автогенерацией

Автоматическая генерация сигналов [2] основана на том, что сигналы некоторых типов сопровождаются набором *подчиненных* сигналов, которые несут дополнительную информацию о *главном* сигнале. Например, с одним входным аналоговым сигналом может быть связано до 25 подчиненных сигналов различных типов. Причем эта связь настолько устойчива, что отсутствие главного сигнала при наличии подчиненных или наоборот может фиксироваться как нарушение ссылочной целостности БД.

Модель автогенерации подчиненных сигналов имеет древовидную структуру и хорошо видна на рис. 2. Поскольку структура модели фиксирована, дерево достраивается автоматически при добавлении пользователем нового подчиненного сигнала и каждого предопределенного свойства.

Предопределенное свойство (поле) генерируемого сигнала может быть *наследуемым* или *вычисляемым*. В качестве значения наследуемого поля использу-

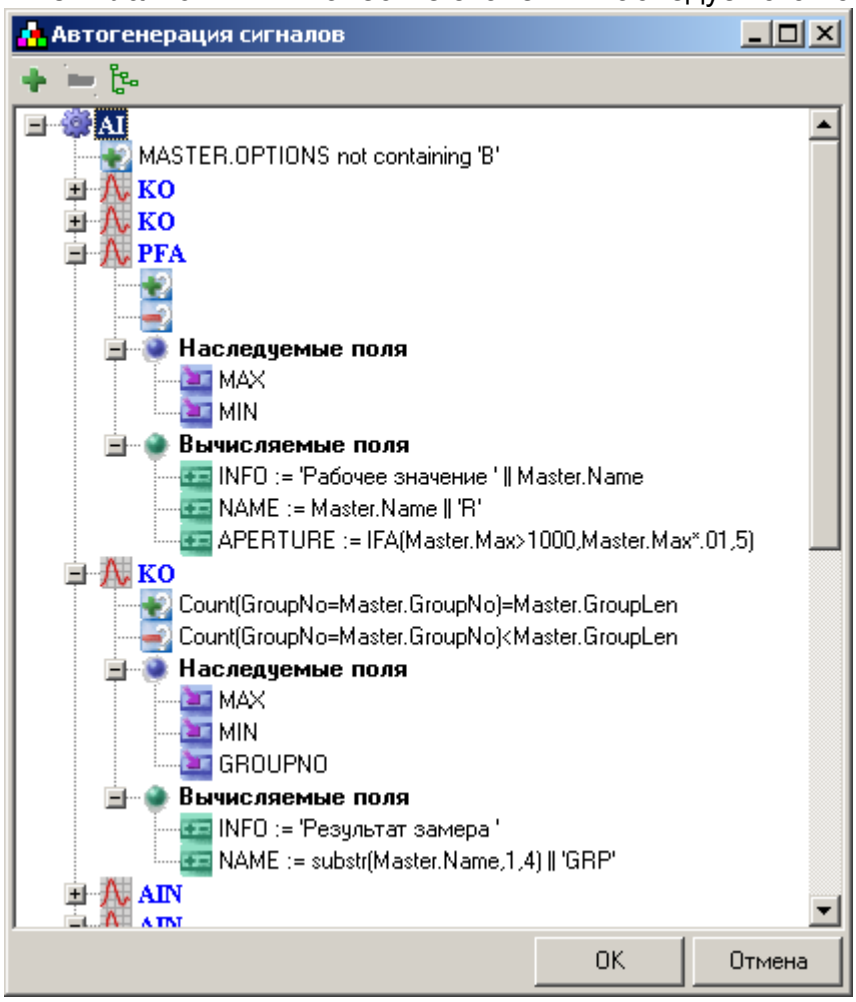

Рис. 2. Пример диалога определения автогенерации

ется значение одноименного поля главного сигнала, а в качестве значения вычисляемого поля – выражение, которое может включать в себя константы, поля главного (Master) сигнала и функции, встроенные либо определяемые пользователем.

Модель автогенерации также включает условия генерации и удаления как для всего набора, так и индивидуально для каждого подчиненного сигнала. При отсутствии этих условий вставка и удаление подчиненных сигналов осуществляются безусловно в момент выполнения аналогичных операций для главного сигнала.

В приведенном примере описана автогенерация для сигналов типа AI. В набор генерируемых сигналов входят подчиненные сигналы следующих типов: KO (три сигнала), PFA (один сигнал), AIN (два сигнала). Условия генерации и удаления указаны перед наследуемыми полями.

В качестве языка для условий и выражений используется упрощенный вариант SQL. Из полученных описаний конструктор генерирует тексты триггеров, которые автоматически срабатывают на сервере в момент вставки, удаления или изменения главного сигнала и, соответственно, вставляют, удаляют или изменяют указанные подчиненные сигналы по заданным правилам.

Редактирование сигналов

Редактор сигналов выполняет такие функции:

- ввод и редактирование свойств сигналов выбранного типа;
- удаление одного, группы или всех сигналов данного типа;
- копирование/перемещение строк;
- итеративное заполнение столбцов по заданному шаблону;
- поиск в таблице сигналов;
- экспорт сигналов в другие БД в режиме использования или копирования;
- показ скрытых свойств сигнала;
- классификация и фильтрация сигналов.

Пример окна редактора сигналов показан на рис. 3.

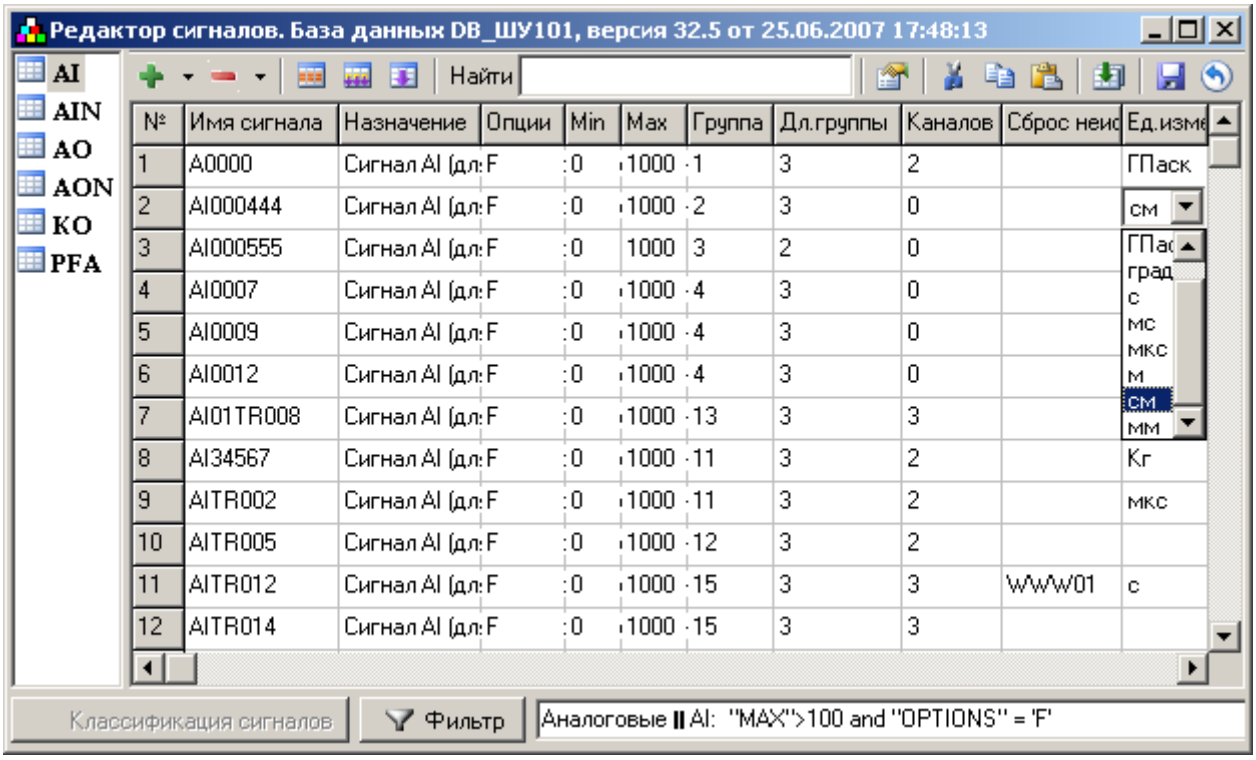

Рис. 3. Пример окна редактора сигналов

Экспорт сигналов поддерживает специализацию ШУ, например выделение ШУ для централизованного ввода входных сигналов с датчиков, и связанное с ней возрастание количества связей между ШУ. Сигналы, импортируемые из другого ШУ, а значит, из другой БД, считаются внешними и не могут редактироваться в данной БД. При каждом открытии БД ее внешние сигналы реэкспортируются из БД-источников. Это соответствует требованию сохранения актуальности данных и

поддерживает коллективную разработку в условиях отсутствия единой базы данных.

Классификация и фильтрация

Массовый выбор сигналов для выполнения групповых операций поддерживается механизмом фильтрации. Пользователь может определить фильтр, который выделит нужное подмножество сигналов БД с помощью SQL-запроса, сгенерированного редактором по выражению фильтрации. Выражение фильтрации может включать в себя классы сигналов из придуманной пользователем классификации и ограничения на значения полей сигналов нужных типов (см. рис. 3).

Классификация сигналов работает в трех режимах (см. рис. 4):

- редактирования ключей;
- связывания с ключами;
- связывания с сигналами;
- просмотра соответствия.

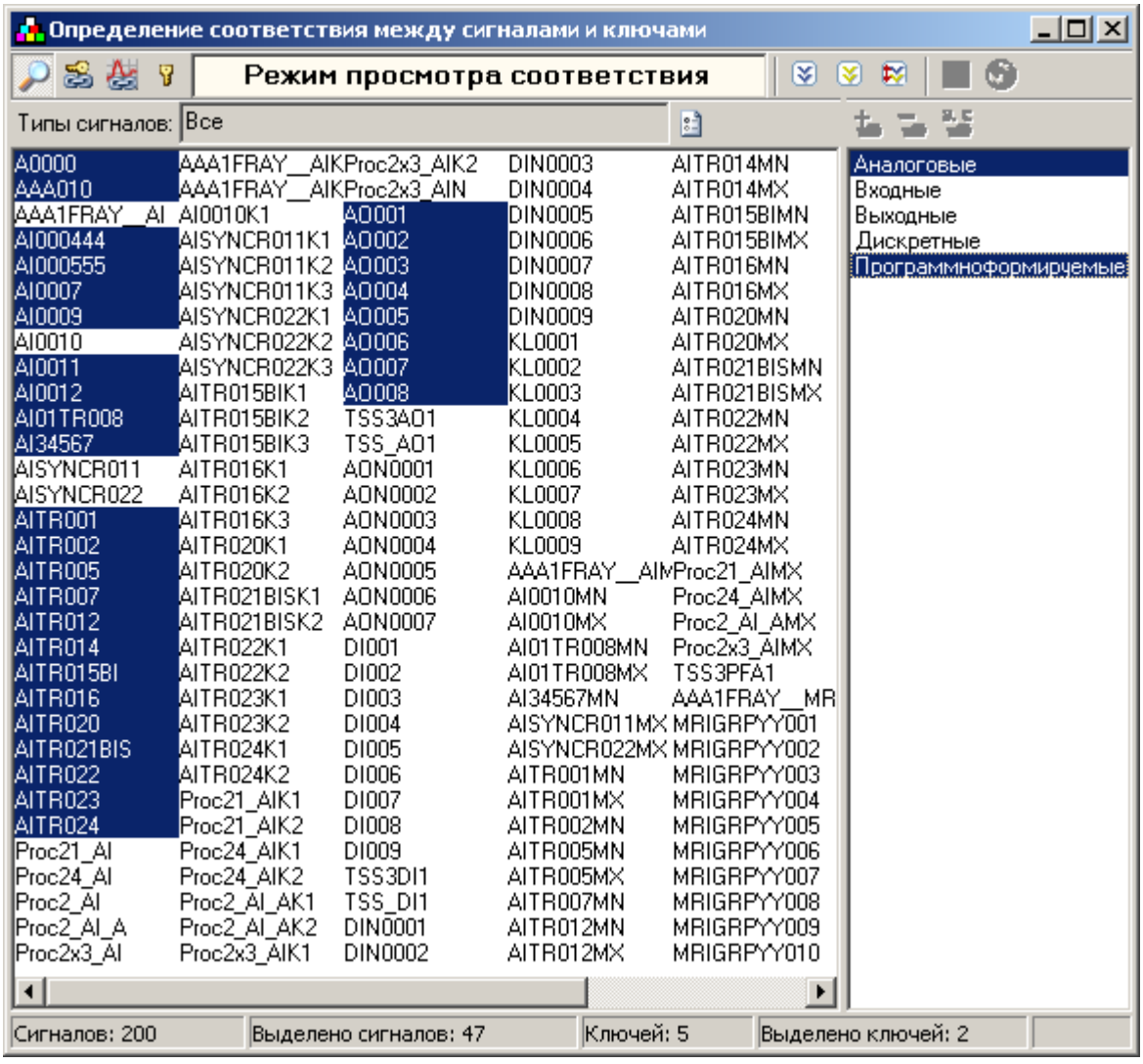

Рис. 4. Окно классификации в режиме просмотра соответствия

В режиме редактирования ключей можно добавлять, удалять или изменять необходимые названия классов (ключи). При этом одновременно могут сосуществовать различные классификации. В приведенном примере две классификации:

- аналоговые/дискретные,
- входные/выходные/программно-формируемые.

В режиме связывания с ключами выбранный набор ключей фиксируется, а выбираемые на левой панели сигналы считаются принадлежащими выбранной конъюнкции классов.

В режиме связывания с сигналами, наоборот, фиксируется выделенное подмножество сигналов, а выбираемые на правой панели ключи устанавливают соответствие этого подмножества конъюнкции желаемых классов.

В режиме просмотра соответствия доступны обе панели: выбор элементов на одной из них вызывает выделение соответствующих элементов на другой.

Ограниченный доступ и регистрация изменений

Для защиты от несанкционированного доступа используются 2 метода:

1) вход в среду разработки, разрешенный только зарегистрированным пользователям;
2) дополнительная защита паролем отдельных БД.

Предусмотрены два метода регистрации изменений. Первый обеспечивает сравнение структуры или содержимого двух баз данных. В результате формируется протокол в виде древовидной структуры обнаруженных отличий. Кроме того, возможна синхронизация структуры и данных обеих БД, при которой пользователь может управлять внесением изменений на уровне отдельных отличий.

Второй метод заключается в установлении пользователем режима регистрации изменений на определенном этапе жизненного цикла АСУ. В этом режиме любое изменение БД (модификация структуры, добавление/удаление сигналов или изменение их свойств) фиксируется в специальных таблицах, называемых журналом коррекций. Диалог просмотра журнала коррекций позволяет сортировать и фильтровать данные по номеру коррекции, дате, имени пользователя, типу элемента (структура, ШУ, тип сигналов, сигнал), имени элемента, действию пользователя (добавление, удаление, изменение), экспортировать полученные данные в документ Word.

Структура базы данных

В качестве базовой СУБД была выбрана система Borland Interbase v.7.0 [3]. Это сетевая СУБД с клиент-серверной архитектурой и поддержкой языка SQL. Ее широко применяют для разработки крупных баз данных, она пригодна для обеспечения всех перечисленных требований и относительно недорога. Реляционная структура Interbase хорошо «ложится» на структуру описаний сигналов: типу сигналов соответствует таблица БД, а свойству – поле таблицы.

База данных Interbase содержит всю информацию о пользовательских таблицах (или *метаданные*) в системных таблицах. Поэтому для построения БД со структурой, определяемой пользователем, можно было бы воспользоваться этими метаданными. Однако такое решение ведет к зависимости от версионных изменений Interbase и, кроме того, вызывает проблемы с хранением дополнительной информации, которая связана со специфическими требованиями и в метаданных не содержится. Поэтому была выбрана структура с тремя группами таблиц (пользовательских с точки зрения Interbase):

1) таблицы для метаданных, т.е. информации о типах сигналов и их свойствах, ограничениях, автогенерируемых сигналах и полях;
2) таблицы для идентификации и классификации сигналов, вводимых пользователем;
3) таблицы, создаваемые конечным пользователем (по одной на каждый тип сигналов) для свойств конкретных сигналов.

Первые две группы образуют фиксированную часть БД сигналов, третья - переменную часть (см. рис. 5).

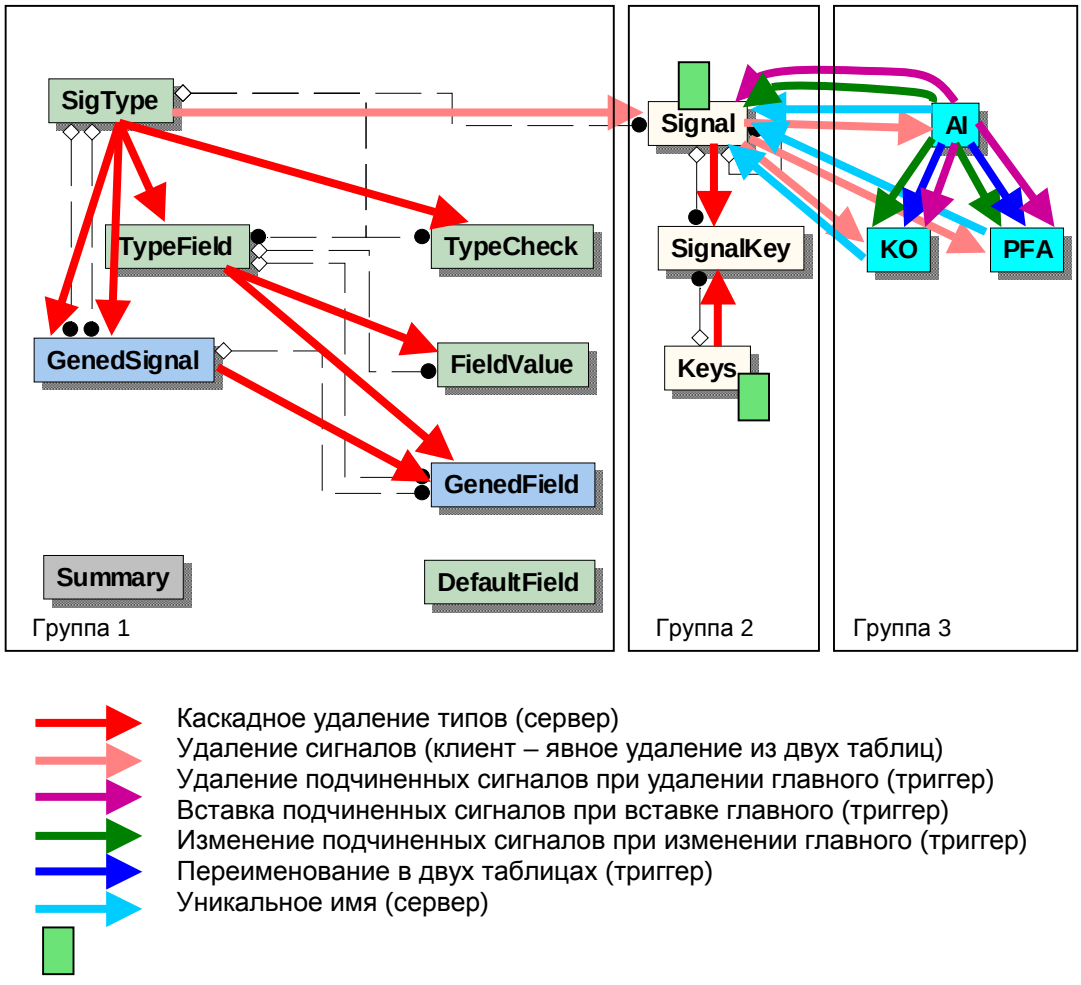

Рис. 5. Зависимости в БД сигналов (таблицы группы 3 показаны как пример)

Характерной особенностью реализации базы данных сигналов является широкое использование средств сервера Interbase: триггеров, внешних ключей,

ограничений ссылочной целостности, контролируемых ограничений (см. рис. 5). Это не только разгружает и упрощает клиентские программы, но и обеспечивает независимость заданных пользователем свойств БД от источника ввода данных.

Выводы

Анализ требований прикладной области и опыт аналогичных разработок позволили построить средства разработки баз данных сигналов, обладающих необходимыми качествами. Они ориентированы на большие объемы данных, присущие крупным промышленным АСУ, поддерживают коллективную разработку, дают возможность пользователю самостоятельно определять и изменять структуру БД без потери данных, обеспечивают необходимые средства автоматизации ввода и выбора данных, защищают данные от несанкционированного доступа и регистрируют изменения.

Список литературы

1. Сухоребрый В.Г. СИНТАР-3 и СИНТАР 2007: сравнительный анализ систем разработки контроллерного ПО / В.Г.Сухоребрый, А.С.Гристан, Д.В.Джулгаков. // Открытые информационные и компьютерные интегрированные технологии. - Х.: НАКУ «ХАИ». – 2007. – Вып.37 . - С. 113 - 118.

2. Сухоребрый В.Г. Методы автоматизации в среде СИНТАР / В.Г.Сухоребрый, А.С.Гристан, Д.В.Джулгаков. // Открытые информационные и компьютерные интегрированные технологии. - Х.: НАКУ «ХАИ». – 2007. – Вып. 38. - С. 91 - 97.

3. Interbase 7.0