Система автоматизированного проектирования нижнего уровня автоматизированной системы управления технологическими процессами (САПР АСУ ТП).

А. С. ГРИСТАН , В. Г. СУХОРЕБРЫЙ , Л. А. ШЕТКИНА

Рассмотрены назначение, область применения и особенности, поддерживаемая методика проектирования, компоненты (подсистемы) и их назначение, оболочка, интерфейс пользователя САПР АСУ ТП, получившей название СИНТАР.

САПР СИНТАР относится к системам класса SCADA и представляет собой интегрированную среду автоматизированного проектирования нижнего уровня АСУ ТП. Под нижним здесь понимается уровень программируемых контроллеров, работающих без непосредственного присутствия оператора.

Система может использоваться как при строительстве новых производственных объектов, так и при реконструкции существующих в разных отраслях промышленности и обладает следующими особенностями:

- комплексный охват этапов нисходящего проектирования: описание задач и связей между ними, конфигурирование вычислительной сети, программирование задач;
- интеграция разноплановых инструментальных средств в единой среде;
- ориентация на сложные алгоритмы сбора и обработки данных, контроля и управления;
- альтернативный выбор средств представления алгоритмов: в визуальной либо текстовой форме;
- независимость от состава библиотек стандартных функций, который определяется задачами конкретной прикладной области.

Методика проектирования, которую обеспечивает СИНТАР, предполагает поэтапную разработку распределенного программно-аппаратного комплекса АСУ ТП. Первый, подготовительный этап, заключается в выборе стандартных элементов ПО и аппаратуры, вводе в базу дан-

ных спецификаций библиотечных функций, перечней и характеристик используемых аппаратных средств.

Результаты этого этапа:

- библиотеки стандартных функций графического и текстового языков;

- каталоги устройств связи с объектом (УСО), датчиков и исполнительных механизмов, драйверов - с настроенными под данные условия применения параметрами;

Второй этап, проектирование сети, включает:

- создание схемы потоков данных (каналов) между задачами;

- описание характеристик каждой задачи;

- спецификацию передаваемых по каналам данных;

- распределение задач по узлам вычислительной сети (процессорам);

- определение параметров операционной системы (ОС) каждого процессора;

- выбор сетевых драйверов;

- автоматическую генерацию настроечных данных для ОС и сетевого ПО в виде исходных файлов базового языка.

Результаты этого этапа сохраняются в виде файлов общих сетевых данных и БД для каждого узла вычислительной сети.

Третий этап, конфигурирование аппаратных средств, охватывает следующие работы:

- привязку оборудования к административно-территориальной структуре предприятия (центральной диспетчерской, цехам, участкам);

- описание иерархии аппаратных компонент и их составляющих (стоек, секций, кассет УСО);

- заполнение посадочных мест секций кассетами УСО;

- привязку сигналов к контактам УСО и описание характеристик сигналов;

- автоматическую генерацию настроечных данных для драйверов УСО.

При выполнении этих операций используются каталоги аппаратных средств, построенные на первом этапе.

В результате этапа в базе данных формируется полная информация об используемом оборудовании и его размещении на территории предприятия, секции стоек заполняются подходящими для размещаемых типов сигналов кассетами УСО, определяются аппаратные адреса абонентов сети и кассет УСО, распределяется память буферов обмена с УСО.

Четвертый этап, программирование задач, обеспечен двумя языковыми интерфейсами и включает:

- построение схем задач графического языка СПРУТ;
- ввод формульных зависимостей текстового языка ЛОГАР;
- компиляцию графических и текстовых модулей в исходные файлы базового языка;
- компиляцию и компоновку исходных файлов базового языка в исполняемую программу (выполняется инструментальными средствами базового языка).

В качестве базового языка взят ассемблер целевого микропроцессора BE-51.

Графический редактор языка СПРУТ предоставляет следующие возможности:

- выбор из структурированного меню и размещение на схеме блоков всех видов; меню алгоблоков структурируется по функциональным группам; меню сетевых данных - по задачам, связанным с текущей;
- построение связей между блоками (ориентированных гипердуг) обоих видов с контролем соответствия типов данных связываемых выходов и входов;
- настройку значений параметров-констант и значений умолчания для неподсоединенных входов алгоблоков и модулей;
- связывание сигналов УСО с конкретными входами/выходами алгоблоков путем выбора из меню сигналов соответствующего типа;
- копирование, перемещение и удаление элементов схемы;
- переход к просмотру/редактированию вложенной или охватывающей схемы;
- печать схемы и сохранение ее в БД.

Альтернативный текстовый язык ЛОГАР - это простейший язык,

так же, как СПРУТ, ориентированный на непрограммиста: в нем можно записывать только арифметические и логические формулы, которые выражают зависимости между выходами и входами ЛОГАР-модуля; никакие управляющие структуры традиционных языков программирования недоступны. Формулы могут иметь произвольную скобочную структуру, включать библиотечные функции. Учитывается старшинство арифметических и логических операций. При необходимости компилятор выполняет неявное преобразование типов.

Текстовый редактор языка ЛОГАР позволяет выбирать из меню входы и выходы модуля, а также библиотечные функции, фиксирует синтаксические ошибки. Компилятор при обнаружении ошибки указывает на нее в окне с исходным текстом, позволяет исправить и возобновить компиляцию.

Результатом этапа являются исполняемые файлы программ для каждой контроллерной секции (узла нижнего уровня).

Важной функцией поддержки всех этапов проектирования является получение проектной документации. Ее выполняет управляемый пользователем генератор отчетов, который позволяет:

- настраиваться на исходный файл базы данных;
- отфильтровывать необходимые записи и сортировать их по нескольким (до 4) полям;
- определять форму документа и содержащиеся в нем данные (названия колонок, их содержание в виде произвольных выражений над полями с использованием стандартных функций обработки);
- генерировать текст документа, редактировать его и записывать в выбранный файл.

Выбор данных более чем из одного файла БД осуществляется с помощью функций пользователя. При задании обработки значений полей файла в меню отображается список полей с указанием их типов.

СИНТАР - это система, управляемая данными. Ее скелет представлен в виде "дерева" меню (на самом деле, это, скорее, сетевой граф с циклами и петлями) и хранится в БД. Вершинами дерева являются меню, а дугами - команды. К свойствам вершин относятся форма графического изображения, функция, возвращающая список элементов меню, заголовок, подсказки и т.п., к свойствам дуг - функция, ре

ализующая команду, ссылка на следующую вершину.

Таким образом, полная информация о структуре и поведении системы находится в БД. С помощью специального интерфейса, доступного только разработчику СИНТАР, эту информацию можно воспроизводить на экране и редактировать, что позволяет в некоторых случаях модифицировать систему даже без рекомпиляции и компоновки.

В заключение отметим ряд преимуществ СИНТАР по сравнению с наиболее используемыми у нас САПР "КВАРЦ" и САПР "Trace Mode".

Средства проектирования сети и конфигурирования аппаратуры не только обеспечивают компьютерную поддержку этих этапов проектирования, но и позволяют построить "разумный" интерфейс, ограничивающий выбор пользователя допустимыми значениями (например, сигналами требуемого типа, или сетевыми данными, передаваемыми от выбранной задачи), а также автоматизировать адресацию (аппаратную и программную) сетевых данных и сигналов УСО.

Поддержка этапов спецификации элементов оборудования и стандартных библиотек (строительных блоков проектируемого комплекса программно-аппаратных средств) обеспечивает относительную инвариантность среды и облегчает ее перенос на новую аппаратуру и другой класс задач.

Явное отображение потока управления в схемах задач обеспечивает надежность и однозначность представления сложных алгоритмов с ветвлением и циклами: игнорирование потока управления (или отождествление его с потоком данных) допустимо только в относительно простых схемах.

Наличие альтернативного текстового языка дает возможность избежать излишней громоздкости, присущей графическим средствам, а также компенсирует неадекватность библиотеки алгоблоков в нестандартных ситуациях.