Создание пакета прикладных программ синтеза систем реального времени на базе UML и сопряжение ее с лучшими мировыми образцами САПР предоставит возможность автоматизации всех стадий проектирования ВС и других технических систем, начиная с эскизного проекта и заканчивая выпуском технической документации, а также оптимального планирования технологических процессов.

Литература

1. Буч, Рамбо, Якобсон. UML.
2. Bruce Powell Douglass. Real-Time UML.
3. Скляров В.А. Синтез автоматов на матричных БИС. – Минск: Наука и техника. – 1984.
4. Кобайло А.С. Основы теории синтеза вычислительных структур реального времени. – Мн.: БГУИР, 2001.
5. Анкудинов Г.И. Синтез структуры сложных объектов: Логико-комбинаторный подход. – Л.: Изд-во Ленинград. Ун-та, 1986.
6. Кобайло А.С. Единичные функции для математических моделей вычислительных процессов: Деп. в Бел ИСА. - Мн.: РСНР, 1997.

АВТОМАТИЗАЦИЯ НА ОАО ”САВУШКИН ПРОДУКТ”

Иванюк Д. С.
Брестский государственный технический университет, г. Брест

В конце 90-х было осознано, что дальнейшее развитие предприятия (в отношении производства) невозможно без автоматизации, поэтому встал вопрос - что дальше? Либо сторонние предложения, либо свои. Было принято решение разработать свою SCADA-систему. SCADA – Supervisory for Control And Data Acquision - диспетчерское управление и сбор данных; SCADA-система – название класса ПО для создания систем автоматизации промышленного производства (АСУ ТП). В результате работы трех человек в течение 4-х лет была разработана программа Monitor и EasyServer. Один из первых проектов – контроль температурных режимов, в дальнейшем автоматизация приёмки молока и т.п. Сейчас – более 20-ти проектов, охватывающих различные области производства (начиная от учёта воды и электричества, различные цеха и участки), проекты в филиалах. Система имеет клиент-серверную структуру: на выделенном компьютере (сервере) выполняется серверная часть проекта, к нему подключаются клиенты (рабочие станции).

Общая схема системы управления приведена на рисунке 1.

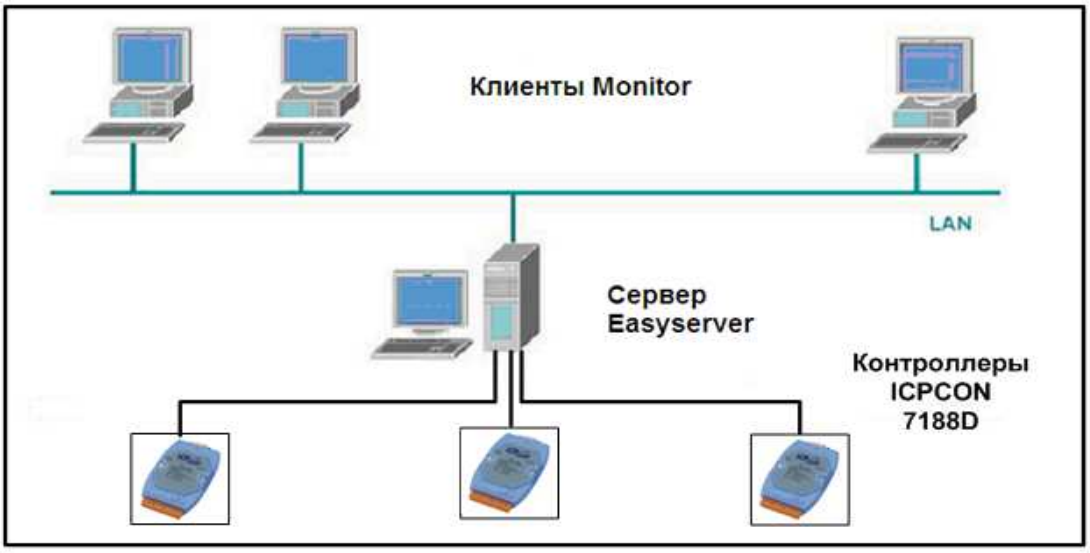

Рисунок 1

Нижний уровень составляют:

- Контроллер – содержит программу, управляющую конкретным технологическим процессом;
- Модули ввода-вывода. Они уже непосредственно управляют оборудованием (клапанами, насосами и т.д.), получают данные с датчиков (температура, давление, концентрация и т.д.);
- Различные операторские панели – автономное управление процессом на рабочем месте.

Для разработки использовались следующие языки программирования: уровень контроллера – C++ (Turbo C 3.0), монитора и сервера – Object Pascal (Delphi).

Monitor – является модульной масштабируемой системой визуализации процесса для приложений различного уровня, начиная с простых однопользовательских приложений в промышленности и до сложных многопользовательских систем.

Monitor обладает удобным пользовательским интерфейсом для создания промышленных приложений и гарантирует стабильную и надежную работу. Пример пользовательского интерфейса программы Monitor приведен на рисунке 2.

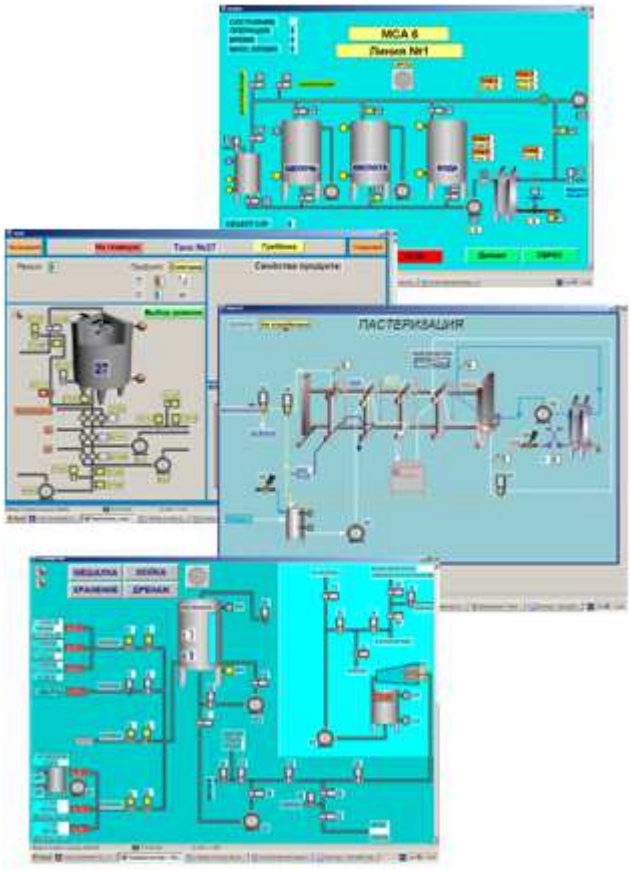

Рисунок 2

Monitor позволяет инженерам-проектировщикам создавать индивидуальные стандарты путем разработки специфических меню, ориентированных на конкретную отрасль промышленности и на конкретный проект.

Monitor также осуществляет архивирование измеряемых значений процесса, предоставляя в дальнейшем возможность отображения их в виде трендов или использования в других отчетах.

Проект, сделанный в программе Monitor, загружается как на сервере, так и локально в качестве клиента. Часть проекта служит как визуализация технологического процесса и управления (клиент), другая часть выполняет функции обработки данных. При создании проекта тэговые свойства объектов привязываются к тэгу базы каналов. В данном случае тэг является указателем на область памяти (ячейку) области данных. База каналов позволяет связать сервер и устройства измерения и управления (смотри рис. 3). Для ее создания служит программа DbEditor. Перед созданием проекта автоматизации в программе Монитор необходимо создать базу каналов. Результатом будет создание файла, описывающего область адресов устройств (датчики, клапана, насосы и т.д.), которые будут использоваться в проекте.

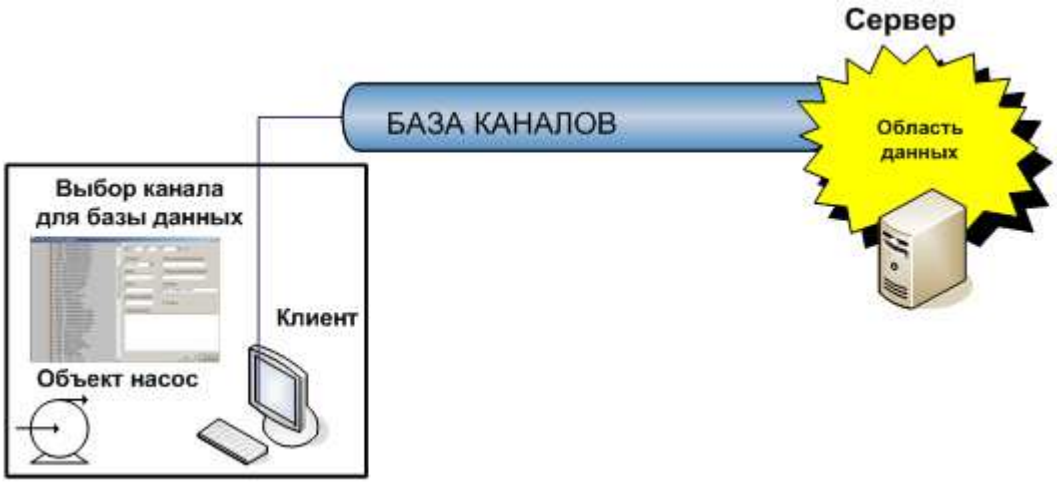

Рисунок 3

В настоящее время на основе разработанного инструментария ведётся не только разработка новых проектов, но и обновление уже существующих в соответствии со спецификой производства – сфера динамически развивается, поэтому это требует соответствующей модернизации оборудования и систем управления.

РЕКУРРЕНТНЫЙ АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЕРА С ВЕТВЛЕНИЕМ НА ЗАДАЧАХ О НАЗНАЧЕНИИ

Кишкевич А.П., Ревотюк М.П.
Белорусский государственный университет информатики и радиоэлектроники, г. Минск

Предмет рассмотрения – рекуррентный алгоритм решения задачи коммивояжера [1] с минимальными изменениями исходной матрицы при переходе от шага к шагу. Потребность учета изменения состояния в представлении задачи возникает в случае ее распараллеливания и синхронизации агентов распределенных вычислений [2,3]. Цель исследования – сокращение расхода памяти и, как следствие, объема передаваемых данных между узлами вычислительной сети .

Задача коммивояжера в классической постановке формулируется так: задана матрица расстояний (стоимости переезда) $C = \left\| c_{ij}, i, j = \overline{1,n} \right\|$ между любым из *n* городов, необходимо найти цикл минимальной длины однократного посещения каждого города.

Если решение задачи коммивояжера представить матрицей булевых переменных $X = \left\| x_{ij}, i, j = \overline{1,n} \right\|$, где единица означает включение в оптимальный цикл дуги $i \to j$, то формальная модель оптимизации имеет вид:

$$\begin{cases} Z = \sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} c_{ij} \cdot x_{ij} \Rightarrow \min; \\ \sum\limits_{i=1}^{n} x_{ij} = \sum\limits_{j=1}^{n} x_{ij} = 1; \\ x_{ij} \geq 0, i, j = \overline{1,n}; \\ u_i - u_j + n x_{ij} \leq n-1, \ i = \overline{2,n}, \ j = \overline{2,n}, \ i \neq j. \end{cases} \quad (1)$$