СПОСОБ ХРАНЕНИЯ АЛГОРИТМОВ РАБОТЫ ТЕХНОЛОГИЧЕСКОГО ОБЪЕКТА В РЕЛЯЦИОННОЙ БАЗЕ ДАННЫХ

Стрябков А.Н.
Донецкий национальный технический университет, г.Донецк
кафедра электронной техники

Abstract
Stryabkov A. The* method *storage of work algorisms of* technological object *in a relational database. *In work the way of storage of work algorithms of a dosing out complex in a relational database is offered. Tabulated representation of algorithms is developed. The estimation of equivalence of transition from a traditional kind of algorithms representation to tabulate is given. The choice of a relational database is made, being based on adjustment flexibility and speed.*

Постановка проблемы и ее актуальность. Разработанный ДонНТУ многодвигательный дозирующий комплекс (МДК) предназначен для приготовления высокооднородных многокомпонентных рецептурных смесей, в частности, пралиновых и конфетных масс. В зависимости от номенклатуры конкретного кондитерского предприятия, МДК может комплектоваться любым набором и количеством единиц оборудования для отработки рецептур, что предъявляет особые требования к системе контроля и управления. К ним относятся:

– наличие алгоритмов самотестирования (self-test) модулей системы контроля и управления,

– возможность замены аппаратно и программно совместимых модулей,

– возможность централизованной замены алгоритмов адаптации к конкретному решению.

Одним из основных требований является возможность быстро и понятно для пользователя (системы управления) заменить алгоритм работы всего МДК или отдельных его частей, в случае отказа отдельных модулей системы контроля и управления, изменения технологических условий эксплуатации, комплектации МДК или с целью модернизации. Актуальность организации такой системы контроля и управления технологическим оборудованием состоит в том, что ее внедрение сократит затраты на профилактические и ремонтные мероприятия, что в значительной степени повысит ее эффективность и универсальность для использования в аналогичных процессах.

Постановка задачи исследования. Система контроля и управления комплексом состоит из следующих составных частей:

1. Микроконтроллерных устройств (МУ), которыми оснащена каждая единица оборудования МДК. Эти МУ задают режимы работы, и контролируют основные параметры оборудования комплекса.
2. Центрального микроконтроллерного устройства (ЦМУ), которое задает режимы работы МУ, производит сбор и анализ собранной информации. Все микроконтроллеры объединены общей вычислительной сетью, построенной на основе протокола витой пары и протокола передачи данных CAN2.0B.
3. Компьютерной системы управления (КСУ), подключенной к ЦМУ. В процессе работы МДК функциями КСУ являются дублирование и координация основных режимов работы, среди которых дополнительно могут быть названы:

– режим хранения в реляционной базе данных и инициализации требуемых алгоритмов работы системы в реальном времени;

– режим тестирования централизованный;

– режим тестирования местный;

– режим экспертной оценки правильности работы устройств;

– режим эмулятора и идентификации повреждений;
– режим диагностирования возможных неисправностей;
– режим сбора информации и хранения ее в реляционной базе данных,
– режим организации анализа процессов в реальном времени;
– режим "диспетчер" для визуализации хода технологического процесса и речевых сообщений.

Как видно из перечисленных функций ЦМУ и все МУ необходимо оснастить перезаписываемой энергонезависимой постоянной памятью (ЭСППЗУ). Современная ЭСППЗУ, например последовательная энергонезависимая память 24FC256 фирмы MicroCHIP, имеет объем памяти 256 килобайт, гарантирует 1000000 циклов записи/стирания памяти данных, а период хранения таких данных составляет более 200 лет, при этом максимальная длина цикла записи составляет не более 5мс, блоками данных по 64 байта. Микросхема выпускается в промышленном исполнении, и имеет защиту от статики более 4000В. В ЭСППЗУ МУ будет храниться алгоритм работы контролируемой им единицы оборудования, а в ЭСППЗУ ЦМУ алгоритм системы управления МДК. Алгоритм работы задается/изменяется оператором на КСУ, после чего он передается в ЦМУ и далее передается на соответствующее МУ. Так же должна быть предусмотрена обратная «передача» алгоритма от МУ в КСУ, для проверки сохраненной в ЭСППЗУ информации, которая может проводиться в требуемом цикле резервирования на протяжении рабочей смены либо в процессе ремонта одного из микропроцессорных устройств.

Цель исследований. Разработать модель хранения алгоритмов работы в базе данных. Убедиться в адекватности такой модели, и целесообразности ее использования.

Основной материал и результаты исследования. Самым удобным и надежным способом хранения данных являются реляционные базы данных или СУБД — системы управления реляционными базами данных. Поэтому целесообразно хранение алгоритмов работы на стороне КСУ организовать с помощью СУБД. Структурная схема включения компонентов системы управления показана на рис. 1. В качестве базы данных была выбрана Oracle DB 9i. Так как — это надежная, хорошо зарекомендовавшая себя база данных, имеющая несколько средств защиты данных от потери (undo сегмент, архивация логов проводимых по базе операций изменения данных, и мощный инструмент восстановления поврежденных баз — RMAN). А так же большое быстродействие, которое осуществляется за счет поддержке **OLAP** (on-line analytical processing) — аналитическая обработка в реальном времени, и **DSS (**Decision Support System) — система поддержки принятия решений.

Реляционная модель данных — это такая модель, при которой данные представляются в виде таблиц, поэтому для того, что бы сохранить алгоритм в базе данных нужно перейти от традиционного представления в виде блок-схемы к табличному виду. Поля таблицы, в которую будет заноситься алгоритм работы, должны содержать такую же информацию, что и блок-схема для того, чтобы, имея табличное представление, можно было перейти к традиционному виду. С помощью такого двойного перехода блок-схема — таблица — блок-схема можно доказать эквивалентность двух представлений алгоритмов. Самое первое и основное, что представляется с помощью блок-схемы — это номер текущего шага, и действие, выполняющееся на этом шаге. Самым простым вариантом является линейная блок-схема, т.е. не имеющая ветвлений (рис. 2). Тогда в таблице, описывающей такой алгоритм, будет всего два поля: номер шага, и действие, выполняемое на этом шаге (табл. 1). Связанно это с тем, что все шаги исполняются последовательно.

Но такие простые блок-схемы, как на рис. 2 встречаются крайне редко, и не представляют интереса. Полноценный алгоритм — это алгоритм, в котором присутствуют ветвления, такие как условия и циклы, что подразумевает под собой изменение последовательности исполнения действий в зависимости от условий, например, для циклов: завершен цикл или нет. Для описания таких ветвлений нужно добавить в таблицу дополнительные поля:

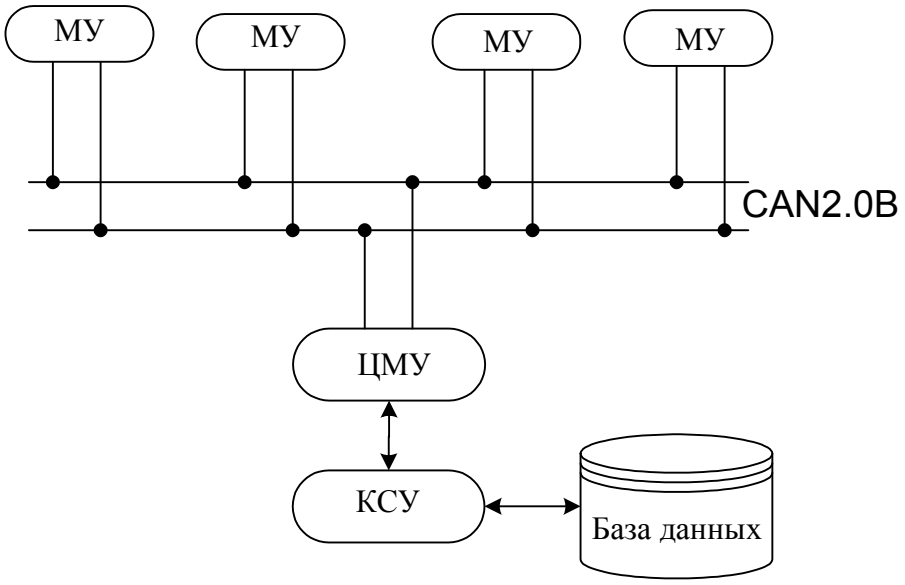

Рисунок 1 — Структурная схема включения компонентов системы управления

Таблица 1

Шаг №	**Действие**
1	Старт
2	Включить блок питания
3	Установка флага
4	Начало работы

Рисунок 2 — Линейная блок-схема

1. Признак — поле, в котором будет описан результат проверки условия, например, двигатель включен. Если действие линейное, то в поле «признак» ставится null.

2. Номер следующего действия — действие, которое исполняется в случае, когда сравнение признака с результатом проверки условия дает «истину», либо при выполнении линейного действия.

3. Номер альтернативного действия — действие, выполняемое в том случае, если сравнение признака с результатом проверки условия дает «ложь». Если действие линейное, то в поле «номер альтернативного действия» ставится null.

Приведем пример более сложной блок-схемы, имеющей ветвления по условиям и циклам (рис. 3).

Теперь представим данный алгоритм в виде таблицы, добавив в неё еще три дополнительных поля, назначение которых было описано выше (табл. 2).

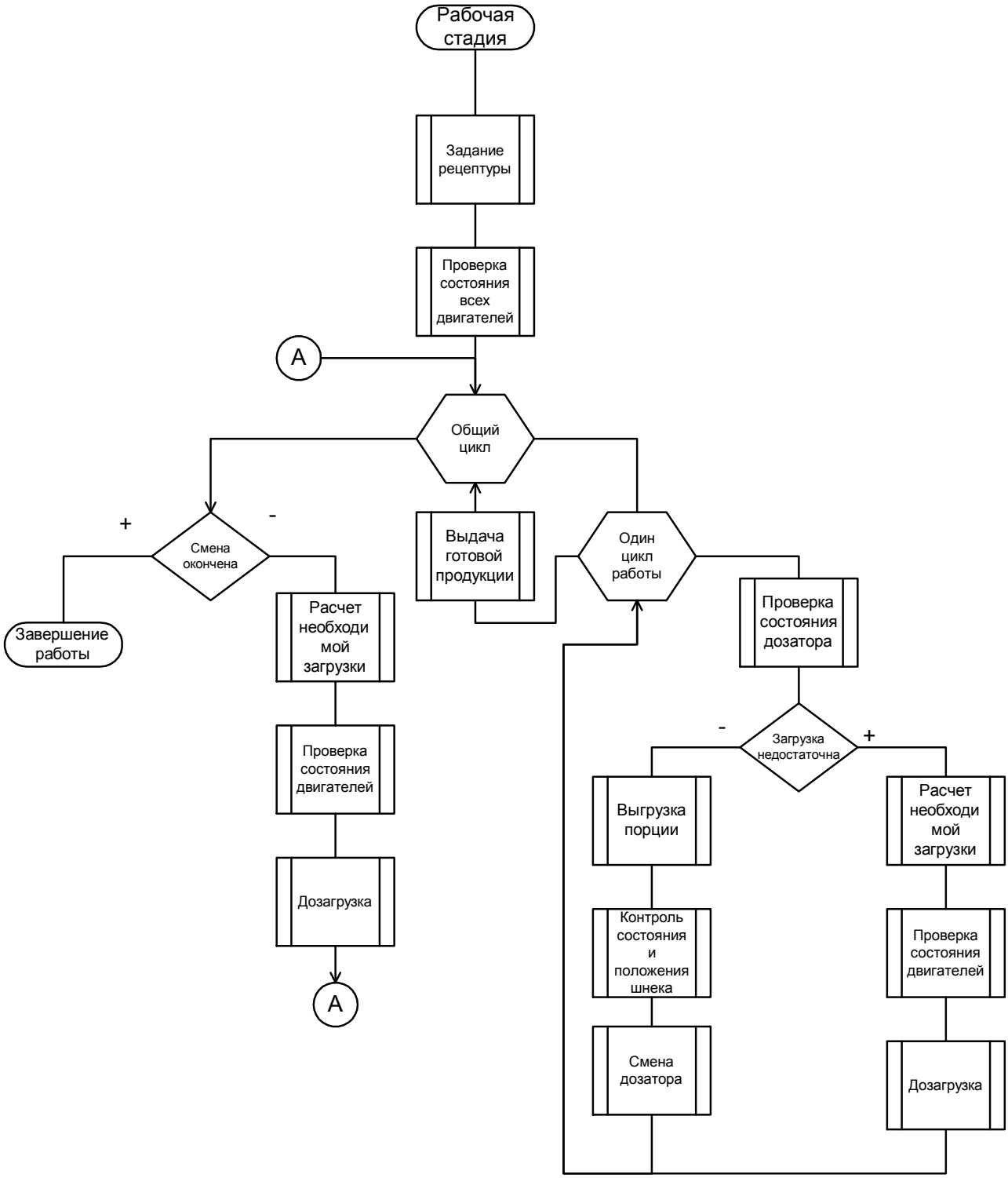

Рисунок 3 — Блок-схема работы МДК

Создадим SQL-запрос, для описания таблицы алгоритмов в базе данных.

```
create table algoritms (
id number(5),
algoritm_id number (3) constraint alg_alg_id_nn not null,
step_id number (2) constraint alg_stp_id_nn not null,
action varchar2(50),
flag varchar2(25),
```

```
next_step number (2),
alt_step number (2),
constraint algoritms_id_pk primary key (id),
constraint alg_alg_id_step_id_uk unique (algoritm_id, step_id));
```

Таблица 2

Шаг №	Действие	Признак	№ следующего действия	№ альтернативного действия
1	Рабочая стадия	Null	2	Null
2	Задание рецептуры	Null	3	Null
3	Проверка состояния всех двигателей	Null	4	Null
4	Общий цикл	Не завершен	5	15
5	Один цикл работы	Не завершен	6	14
6	Проверка состояния дозатора	Null	7	Null
7	Загрузка недостаточна	Да	8	11
8	Расчет необходимой загрузки	Null	9	Null
9	Проверка состояния двигателей	Null	10	Null
10	Дозагрузка	Null	5	Null
11	Выгрузка порции	Null	12	Null
12	Контроль состояния и положения шнека	Null	13	Null
13	Смена дозатора	Null	5	Null
14	Выдача готовой продукции	Null	4	Null
15	Смена окончена	Нет	16	19
16	Расчет необходимой загрузки	Null	17	Null
17	Проверка состояния двигателя	Null	18	Null
18	Дозагрузка	Null	4	Null
19	Завершение работы	Null	Null	Null

В таблице, хранящей информацию в базе данных, описано еще одно дополнительное поле algoritm_id — «Номер алгоритма». Это поле необходимо, так как каждый дозатор имеет свой собственный алгоритм работы дозатора, и все эти алгоритмы нужно хранить в базе данных. Создавать для каждого алгоритма свою собственную таблицу не целесообразно, поэтому их можно хранить в одной таблице, указав порядковый номер алгоритма. На уровне базы данных создано ограничение целостности, которое требует уникальности комбинации полей algoritm_id и step_id, то есть предот-

вратить ошибочный ввод двух действий с одинаковым номером шага для одного и того же алгоритма. Таким образом, мы защитили дозирующий комплекс от ошибок оператора, которые могу привести к критическим сбоям при разборе алгоритмов.

Пример вызова алгоритма №1 из программы связи информационных средств с базой данных показан на рис. 4. Для отображения первого алгоритма программа сделала запрос “SELECT * from Algoritms WHERE algoritm_id = 1 ORDER BY step_id” к базе данных.

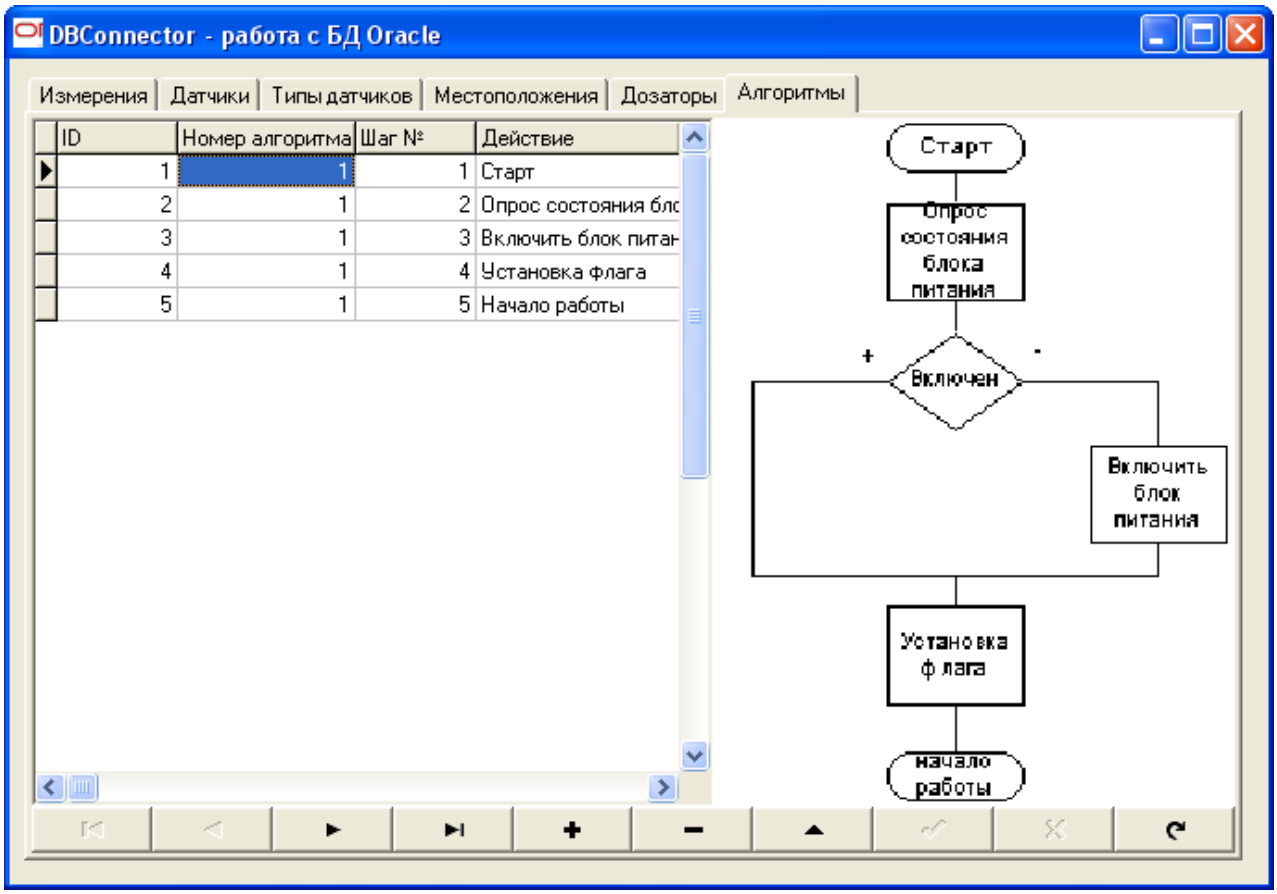

Рисунок 4 — Пример работы программы

Выводы

1. В статье впервые показана возможность эквивалентного представления алгоритмов и хранения их в реляционной базе данных.

2. Такое представление значительно упрощает процесс смены алгоритмов работы сложной технологической системы при необходимости резервирования, модернизации или наличии неисправностей.

3. Переход к табличному представлению алгоритмов приводит к возникновению дополнительных возможностей работы системы и повышению ее эффективности.

Литература

1. Introduction to Oracle9i: SQL, Nancy Greenberg, Priya Nathan: — Oracle corp., June 2001.
2. Oracle9i database administrator fundamentals 1, Sarath Chandran, Marie St. Gelais, S Matt Taylor Jr: — Oracle corp., May 2001.
3. Oracle9i database administrator fundamentals 2, Donna Keesling, James Womack: — Oracle corp., May 2001.
4. Сайт компании MicroCHIP: www.microchip.ru.
5. Поль Дюбуа, MySQL: Пер. с англ .: — М.: Издательский дом «Вильямс», 2001. — 816 с.