
**МАТЕМАТИЧЕСКИЕ МЕТОДЫ
И МОДЕЛИРОВАНИЕ В ПРИБОРОСТРОЕНИИ**

УДК 681.3.068

© А. И. Петров

**ПЕРЕХОД ОТ ПРОБЛЕМНО-ОРИЕНТИРОВАННОГО ЯЗЫКА
К АВТОМАТНОМУ ПРИ ПРОЕКТИРОВАНИИ
АНАЛИТИЧЕСКИХ ПРИБОРОВ**

В статье рассматриваются основные положения автоматного программирования и обосновываются преимущества его использования при разработке программного обеспечения. Описывается методология построения проблемно-ориентированного языка на основе поведенческих, функциональных, информационных и структурных моделей, выполненных на стадии эскизного проекта при разработке аналитических приборов, и приводятся примеры успешного применения предлагаемого подхода на практике.

Кл. сл.: проблемно-ориентированный язык программирования, автоматное программирование, автоматный язык, конечный автомат, аналитические приборы

ВВЕДЕНИЕ

Современные аналитические приборы проще всего рассматривать как аппаратно-программные комплексы (АПК), призванные решать задачи аналитических измерений, зачастую представляющих комплекс сложных исследований, включая адаптацию способов и методов проведения измерений к новым задачам. С другой стороны, развитие микроэлектроники, программного обеспечения, каналов связи — всего того, что принято называть "информационными технологиями", дает возможность разрабатывать гибкие инструменты, возможности которых зачастую проявляются и осознаются во время эксплуатации и дают возможность решать все более сложные задачи.

Анализ трудоемкости решения всех задач при разработке АПК показывает, что произошло смещение объемов трудозатрат в область "информационных технологий", в первую очередь разработки программного обеспечения. Именно на программное обеспечение переносится ответственность за корректность работы и достоверность результатов работы АПК. Разработка программного обеспечения требует объединения усилий специалистов в различных предметных областях (эксперты-методисты предметной области, инженеры-конструкторы — разработчики аппаратуры, программисты). И как следствие необходимы специалисты, обеспечивающие синхронизацию работ при проектировании АПК.

В этой статье описывается подход к созданию программного обеспечения, который был использован при разработке прибора АНК-96.

АНК-96 является анализатором нуклеиновых

кислот на основе полимеразной цепной реакции с наблюдением в реальном времени, призванным продолжить серию приборов АНК, выпускаемых Институтом аналитического приборостроения РАН [1].

**АВТОМАТНЫЙ ПОДХОД
ПРИ ПРОЕКТИРОВАНИИ АПК**

Анализируя опыт по созданию АПК для различных применений, в которых автору приходилось участвовать, наиболее перспективным оказалось использование подхода описания алгоритмов, использующихся в АПК, на основе теории конечных автоматов. А.А. Шалыто удачно назвал такой подход автоматным программированием [2, 3].

В настоящее время в различных областях программирования все шире применяются конечные автоматы, которые в течение многих десятилетий использовались в аппаратуре в основном при аппаратных реализациях. В настоящей статье на основе обзора методов алгоритмизации и программирования для систем управления АПК сформулированы основные положения технологии алгоритмизации и программирования при разработке ПО. При этом проектируемые программы рассматриваются как взаимосвязанные конечные автоматы.

**КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ
ПРОГРАММНЫХ СИСТЕМ**

В России действует государственный стандарт на стадии создания автоматизированных систем

ГОСТ 34.601-90. Существует и международный стандарт на стадии жизненного цикла программной продукции (ISO 12207:1995).

Будем рассматривать АПК как автоматизированную систему (АС), в которой наряду с аппаратной частью все программные части образуют единую информационную систему.

При концептуальном проектировании автоматизированных систем применяют ряд спецификаций, среди которых центральное место занимают модели преобразования, хранения и передачи информации.

Модели, полученные в процессе обследования АС, являются моделями ее функционирования. Различают функциональные, информационные, поведенческие и структурные модели.

Функциональная модель системы описывает совокупность выполняемых системой функций.

Информационные модели отражают структуры данных — их состав и взаимосвязи.

Поведенческие модели описывают информационные процессы (динамику функционирования), в них фигурируют такие категории, как состояние системы, событие, переход из одного состояния в другое, условия перехода, последовательность событий, осуществляется привязка ко времени.

Структурные модели характеризуют морфологию системы (ее построение) — состав подсистем, их взаимосвязи.

Выделим основные, ключевые этапы проектирования АПК.

1. Анализ технических требований.
2. Разработка функциональных, информационных, поведенческих и структурных моделей.
3. Разработка спецификации проблемно-ориентированного языка на основе принятых поведенческих моделей.
4. Разработка автоматного языка на основе функциональных моделей.

Конкретизируем эти этапы на основе автоматного подхода.

ПРОБЛЕМНО-ОРИЕНТИРОВАННЫЙ ЯЗЫК

Проблемно-ориентированный язык (ПОЯ, *DSL — Domain Specific Language*) — язык программирования, сфокусированный на определенной предметной области и содержащий элементы (слова), отражающие абстракции именно этой предметной области [4]. По набору слов ПОЯ соответствует моделям (функциональным, информационным, поведенческим и структурным), полученным на стадии эскизного проекта в рамках концептуального проектирования [5, 6]. К проблемно-ориентированному набору слов могут быть добавлены структуры управления (циклы, усло-

вия, переходы). Иначе говоря, разработчик может выразить предметную область в самой лаконичной и пригодной для чтения и редактирования форме и, самое главное, понятной для экспертов форме.

Проблемно-ориентированный язык разрабатывают эксперты-методисты как разработчики поведенческих моделей АПК.

Можно выделить следующие достоинства ПОЯ.

- Использование труда экспертов — это возможность исправить то состояние дел, которое наблюдается сейчас в области взаимодействия программистов и экспертов в предметных областях. Последнее является главным камнем преткновения на пути успешной работы над проектами.
- Смена контекста выполнения дает разработчику широкие возможности по оптимизации продукта.
- Использование альтернативных моделей вычислений в рамках одного проекта позволяет гибко комбинировать парадигмы программирования, например, объектно-ориентированные и функциональные.
- Резкое снижение уровня "семантического шума" позволяет разработчику сконцентрироваться на решении проблемы предметной области, а не на особенностях путей конкретной реализации.

По семантике получившийся язык наиболее полно описывает жизнедеятельность разрабатываемого аппаратно-программного комплекса. Следовательно применение ПОЯ позволяет экспертам-методистам в предметной области разрабатывать алгоритмы работы АПК не привлекая технологов-программистов.

Для реализации ПОЯ возможно несколько подходов.

- Построение интерпретатора.
- Промежуточная компиляция в семантику классических языков программирования C++, C#, Java и т. п.
- Подход с точки зрения теории формальных языков.

Подход с точки зрения теории формальных языков является наиболее перспективным, т. к. позволяет использовать всю мощь теоретических исследований в этой области.

Сформулируем подход к реализации ПОЯ

При помощи разрабатываемого языка будем описывать алгоритмы работы АПК. Получившийся набор выражений (описаний алгоритмов) образует регулярное множество. Согласно теореме Клини (теорема синтеза), по регулярному выражению может быть построен автомат, представляющий регулярное множество [7].

Т. к. регулярное множество является конечным

в силу построения, то получившийся автомат будет конечным в силу определения. Таким образом, при описании реализации ПОЯ будем рассматривать конечные автоматы (КА) и все, что с ними связано [8].

Получившийся язык программирования отлично показывает, что делаем, но не конкретизирует как делаем. Это связано с семантическим разрывом, который все-таки получается при попытке реализовать ПОЯ только силами программистов.

Для преодоления этого разрыва проведем декомпозицию АПК на основе структурной модели. Обособленные, функционально законченные элементы АПК требуют дополнительной детализации в описании спецификаций — моделей, что в свою очередь приводит к созданию ПОЯ для данного элемента АПК. Назовем такой язык автоматным. При разработке моделей участвуют специалисты-эксперты в области конструирования аппаратуры и, следовательно, получившийся язык может сильно отличаться по семантике от целевого ПОЯ.

При реальной работе АПК будут отрабатываться алгоритмы, описываемые автоматным языком, следовательно, такой язык учитывает особенности аппаратуры и лучше всего подходит для управления аппаратной частью комплекса.

АВТОМАТНЫЙ ЯЗЫК

Автоматный язык (АЯ) — язык программирования, сфокусированный на определенный элемент АПК и содержащий слова, соответствующие функциональным и информационным моделям, полученным на стадии эскизного проекта в рамках концептуального проектирования данного АПК. АЯ реализуется конечным автоматом, распознающим разрабатываемый автоматный язык.

Автоматный язык призваны разрабатывать инженеры-конструкторы и технологи-программисты,

участвующие в разработке аппаратной части комплекса, а также функциональных и информационных моделей АПК.

В связи с описанием различных моделей разрабатываемого АПК различными группами разработчиков сразу получаем семантический разрыв, связанный с тем, что зачастую в различных предметных областях одни и те же термины трактуются по-разному, а одни и те же сущности могут называться по-разному. В этом семантическом разрыве кроются основные проблемы при разработке программного обеспечения.

Семантический разрыв в описании алгоритма работы аппаратуры и в описании работы всего АПК является естественным в силу построения поведенческих моделей АПК.

Разработку конечной аппаратуры ведет группа специалистов, далеких от предметной области исследований, но являющихся экспертами в конструировании аппаратуры и физических принципах работы аппаратуры. Поэтому поведенческая модель аппаратуры содержит свой набор алгоритмов, а, следовательно, порождает свой проблемно-ориентированный язык. Этот язык является автоматным в силу конечного множества использованных слов, а, следовательно, может быть распознан конечным автоматом. На рис. 1 показана взаимосвязь проблемно-ориентированного языка программирования, автоматного языка и конечного автомата.

В связи со множественностью задач, решаемых АПК следует рассматривать совокупность автоматных языков, а, следовательно, совокупность конечных автоматов (рис. 2). В данном подходе взаимосвязи между отдельными автоматами (автоматными языками) осуществляется на верхнем уровне силами ПОЯ.

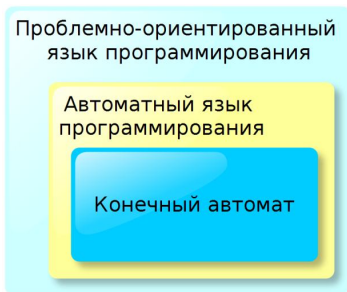


Рис. 1. Взаимосвязь ПОЯ—АЯ—КА

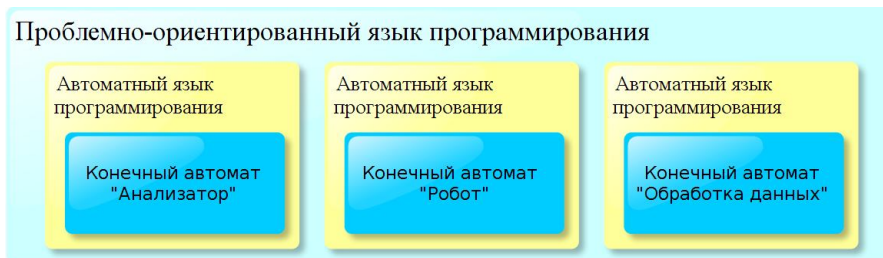


Рис. 2. Взаимосвязь ПОЯ с несколькими АЯ—КА

ПЕРЕХОД ОТ ПРОБЛЕМНО-ОРИЕНТИРОВАННОГО ЯЗЫКА К АВТОМАТНОМУ

Можно привести аналогию с классическими определениями языков программирования: Проблемно-ориентированный язык относится к языкам высокого уровня — аналог C++ или C#, а автоматный язык рассматриваем как язык низкого уровня — аналог языка ассемблера для данной вычислительной платформы. По аналогии переход от проблемно-ориентированного языка к автоматному будем называть трансляцией.

Построение транслятора ПОЯ является нетривиальной задачей, но если не гнаться за универсальностью и не пытаться построить "универсальный решатель", то решение оказывается простым и наглядным.

Алгоритм перехода от ПОЯ к автоматному языку может и должен быть описан с помощью конечного автомата — транслятора ПОЯ—КА.

Полностью определенный автоматный язык позволяет достаточно просто синтезировать конечный автомат для реализации, что в свою очередь может породить рекомендации к выбору или разработке аппаратной платформы.

Конечный автомат в силу построения полностью соответствует аппаратной части АПК. Реализация автомата, имитирующая работу аппаратуры, позволяет отлаживать программное обеспечение, не дожидаясь окончания разработки приборной части, выполняя такую важную часть разработки, как верификацию ПО.

АПРОБАЦИЯ МЕТОДА

Рассмотрим пример разработки программного обеспечения АПК генетического анализатора на основе предложенного подхода.

В руководстве [9] "Растение / 35S+FMV / NOS скрининг" (Тест-система для обнаружения ГМО растительного происхождения) (ЗАО Синтол / ВНИИСБ, Москва), кат. № GM-415, приведен сценарий работы анализатора АНК-32М — протокол "Раст/35S/NOS" с табл. 1 и 2.

Этапы проектирования ПО АПК

Резюмируя, можно выделить этапы проектирования ПО АПК (табл. 3).

Первый этап — общепринятый подход концептуального проектирования, выполняемый в рамках эскизного проекта. Построение поведенческих, структурных, функциональных и информационных моделей.

Второй этап — разработка ПОЯ на основе моделей, полученных на первой стадии.

Третий этап — выделение в рамках ПОЯ автоматных языков, соответствующих реальным алгоритмам работы аппаратуры.

Четвертый этап — разработка конечных автоматов, реализующих автоматные языки и конечный автомат-транслятор ПОЯ в автоматные языки.

Табл. 1. Циклограмма для теплового блока*

Номер ступени	Температура, °C	Время, с	Кол-во циклов	Метки циклов
1	95	300	1	—
2	60	40	50	Начало
3	72	15	50	—
4	95	15	50	Конец

* — по сравнению с [9] здесь в описание циклограммы добавлена температурная ступень 3, чтобы привести к классической схеме: денатурация (95 °C), отжиг (60 °C), элонгация (72 °C) [10].

Табл. 2. Реакции и цветовые каналы для оптического блока

Номер канала	ДНК-мишень	Канал регистрации
1	Растение	R6G / HEX / JOE / Yellow
2	Промотор 35S+FMV	ROX / TxRed / Orange
3	Терминатор NOS	FAM / Green
4	ВПК	Cy5 / Red

Табл. 3. Этапы разработки

Этапы проектирования		Модели АПК			
		функциональная	информационные	поведенческие	структурные
1	Построение моделей	Состав функций АПК	Входные данные, выходные данные	Динамика функционирования	Состав подсистем АПК, их взаимосвязи
2	Проблемно-ориентированный язык	Словарный состав языка	Переменные — структуры данных ПОЯ	Алгоритмы работы	Переменные — структуры данных ПОЯ
3	Автоматный язык	Словарный состав языка	Переменные — структуры данных АЯ	Алгоритмы работы	Переменные — структуры данных АЯ
4	Конечный автомат	Функции перехода, функции выхода	Состояния	Состояния, входной алфавит	Состояния

Построение моделей АПК

Поведенческая модель системы

Для исходных требований при разработке алгоритмов решения задач программного обеспечения были использованы методические пособия и инструкции по применению наборов реагентов, выпускаемых научно-производственной компанией СИНТОЛ и ООО "ИнтерЛабСервис". Были проанализированы алгоритмы работы и способы описания алгоритмов работы существующих анализаторов: АНК-32, DTprime, iQ4/5, CFX-96, Rotor-Gene 6000, ABIPrism 7500.

По результатам опроса лаборантов-методистов центра коллективного пользования научным оборудованием ВНИИСБ "Биотехнология" получили словесное описание действий лаборанта и работы анализатора в составе аппаратно-программного комплекса.

- Ввод параметров работы теплового блока: описание циклограммы работы (табл. 1).
- Ввод параметров работы оптического блока: описание каналов регистрации (табл. 2).
- Запуск измерений:
 - установка плашки;
 - передача данных в анализатор;
 - запуск анализатора.
- Проведение измерений:
 - выполнение циклограммы работы теплового блока;
 - получение сигналов флуоресценции.
- Наблюдение в реальном времени:
 - получение данных из анализатора;
 - визуализация данных.

Результатом работы анализатора являются данные измерений сигналов флуоресценции, полученные в заданном канале измерений при заданной температуре.

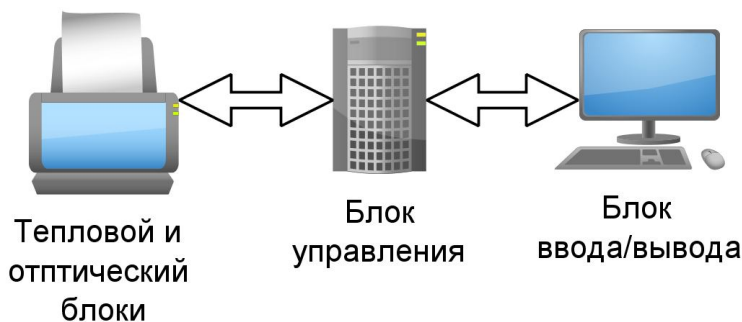


Рис. 3. Структурная схема анализатора нуклеиновых кислот

Структурная модель АПК

Структурная модель программного обеспечения АПК повторяет структуру анализатора (рис. 3).

В состав АПК входят:

- подсистема "тепловой блок";
- подсистема "оптический блок";
- подсистема "блок управления":
 - модуль обмена данными;
 - модуль хранения данных;
- подсистема "блок ввода/вывода данных".

Функциональная модель системы

Функциональная модель системы получается обобщением поведенческих моделей при выделении описаний функциональных законченных блоков поведенческих моделей в отдельные модули-функции:

- задать температуру теплового блока;
- задать канал регистрации оптического блока;
- запустить измерения;
- сохранить данные.

Информационные модели системы

Информационная модель системы получается обобщением поведенческих моделей при выделе-

нии описаний типов и структур данных функциональных компонент.

Поведенческие модели блоков анализатора

Построим поведенческую модель работы теплового блока:

- включить тепловой блок;
- задать температуру;
- ждать стабилизации температуры;
- выключить тепловой блок.

Построим поведенческую модель работы оптического блока:

- включить оптический блок;
- установить канал регистрации;
- запустить регистрацию данных;
- получить данные по образцам;
- выключить оптический блок.

Основное утверждение

Любой алгоритм работы анализатора будет описываться приведенными выше функциями.

Реализация ПОЯ

С помощью проблемно-ориентированного языка опишем сценарий работы анализатора [10].

Задание на анализ GM-415

// Определение сценария ПОЯ

Циклограмма:

Денатурация: температура — 95 °C, время — 300 с

40 циклов:

Отжиг: температура — 60 °C, время — 40 с

Элонгация: температура — 72 °C, время — 15 с

Денатурация: температура — 95 °C, время — 15 с

Хранение: 25 °C

Каналы регистрации: R6G, ROX, FAM, Cy5

Регистрация данных: Все

Завершить задание

// Закончить сценарий

На первый взгляд можно построить простой автомат, непосредственно обрабатывающий сценарий на ПОЯ.

Перепишем сценарий на ПОЯ с алгоритмической точки зрения:

Ступень 1: $T = 95\text{ }^{\circ}\text{C}$, $t = 300\text{ с}$,
измерить каналы: R6G, ROX, FAM, Cy5.

Цикл 40 повторов:

Ступень 1: отжиг, $T = 60\text{ }^{\circ}\text{C}$, $t = 40\text{ с}$,
измерить каналы: R6G, ROX, FAM, Cy5.

Ступень 2: элонгация, $T = 72\text{ }^{\circ}\text{C}$, $t = 15\text{ с}$,
измерить каналы: R6G, ROX, FAM, Cy5.

Ступень 3: денатурация, $T = 95\text{ }^{\circ}\text{C}$, $t = 15\text{ с}$,
измерить каналы: R6G, ROX, FAM, Cy5.

Конец цикла.

С учетом того, что в реальной циклограмме может быть несколько циклов, а начало работы может состоять из двух ступеней — отжиг и денатурация, — отобразим алгоритм работы распознавателя ПОЯ в виде блок-схемы рис. 4.

В предлагаемом подходе при реализации автоматного языка в первую очередь избавляемся от циклов, которые обычно являются причиной неадекватного поведения аппаратуры в виде зависаний ПО. Тем более что реальное функционирование АПК растянуто по времени, и проще всего рассматривать его как последовательность действий в одном направлении без "возвратов в прошлое". Таким образом, реальная поведенческая модель АПК будет соответствовать табл. 4.

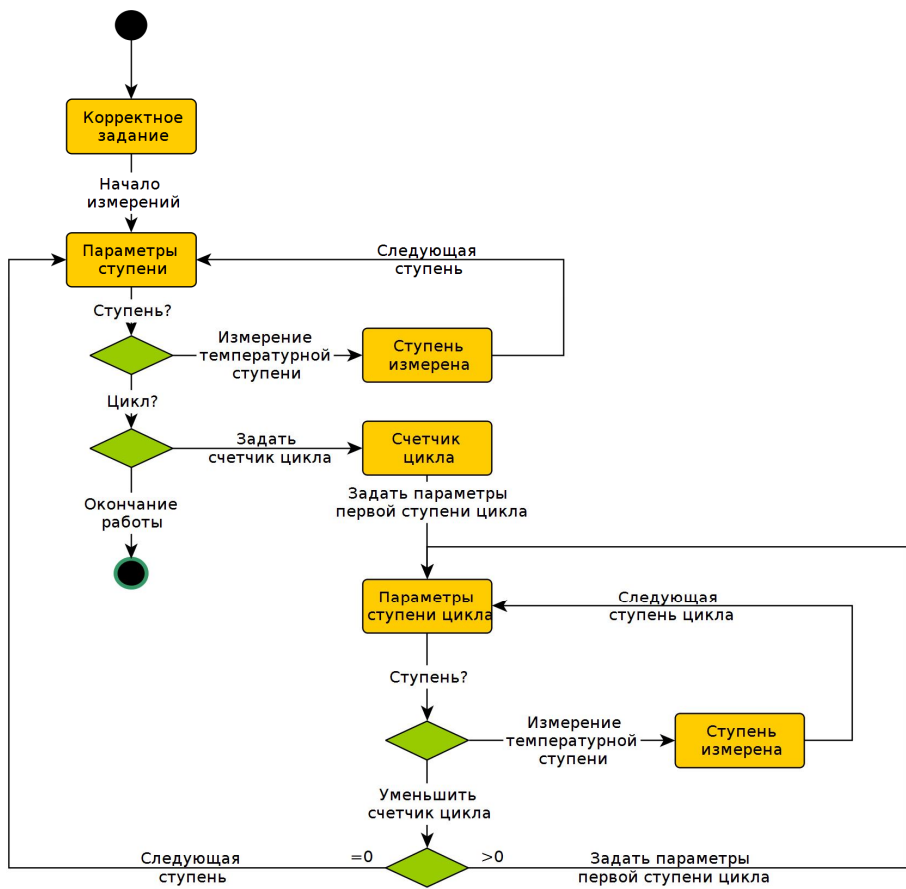


Рис. 4. "Очевидный" алгоритм работы анализатора

Табл. 4. Поведенческая модель анализатора в виде последовательного списка исполняемых функций

№	Описание функции алгоритма
1	Включить тепловой блок
2	Включить оптический блок
3	Задать температуру 95 °С
4	Ждать стабилизации температуры
5	Установить канал регистрации FAM
6	Запустить регистрацию данных
7	Получить данные по образцам
8	Установить канал регистрации R6G
9	Запустить регистрацию данных
10	Получить данные по образцам
11	Установить канал регистрации ROX
12	Запустить регистрацию данных
13	Получить данные по образцам
14	Установить канал регистрации Су5
15	Запустить регистрацию данных
16	Получить данные по образцам

...	...
161	Задать температуру 95 °С
162	Ждать стабилизации температуры
...	...
2116	Выключить оптический блок
2114	Выключить тепловой блок

Для реализации такой модели очень хорошо подходит простой автоматный язык, по семантике совпадающий с поведенческой моделью, и соответствующий АЯ конечный автомат рис. 5.

Сравнение алгоритмов работы конечного автомата для ПОЯ и для автоматного языка

Первое, что бросается в глаза при сравнении ПОЯ и АЯ, — это семантический разрыв в описании одного и того же алгоритма работы прибора:

– ПОЯ написан экспертом-методистом лаборатории ПЦР, т. е. конечным пользователем прибора.

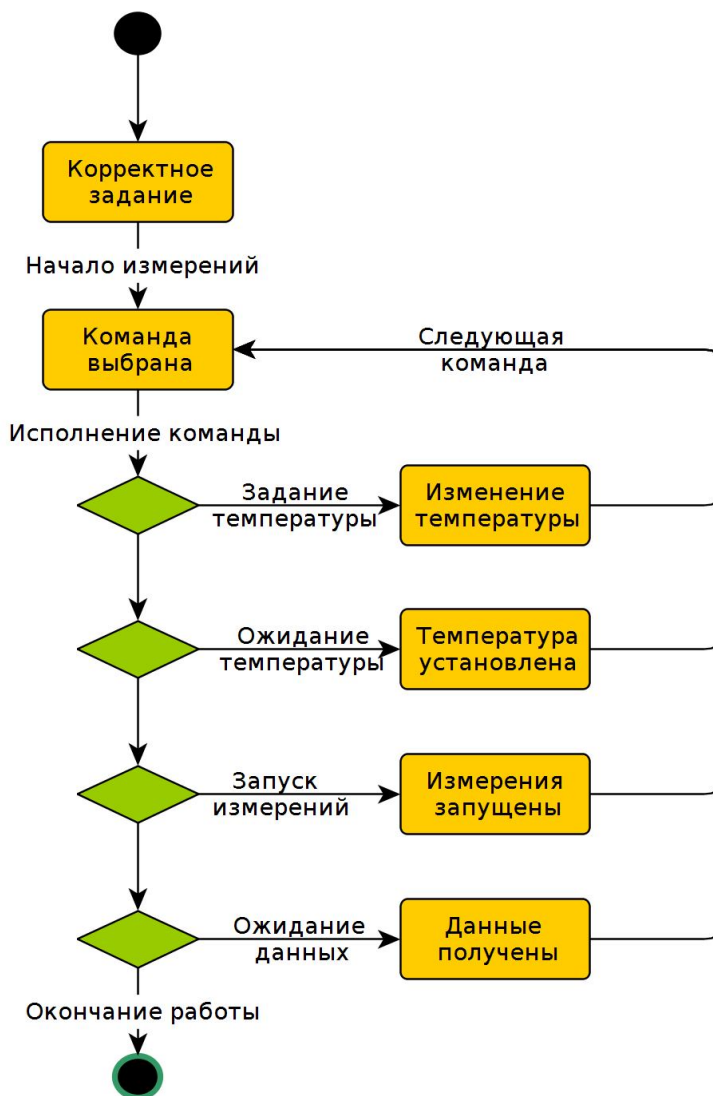


Рис. 5. КА для реализации автоматного языка

– АЯ написан инженером-разработчиком с точки зрения работы составных блоков анализатора.

Во-вторых, очевидно, что конечный автомат АЯ имеет более простую структуру, меньшее число состояний и переходов. К тому же в нем отсутствуют переходы с условиями, что упрощает алгоритм работы, повышает надежность работы алгоритма.

Главное отличие реализации АЯ в том, что команды управления аппаратурой можно чередовать в произвольном порядке и менять параметры (например, температуру) по любому правилу. Все эти изменения будут касаться только входного алфа-

вита предложенного автомата, но не самого автомата.

Попытка изменить параметры в случае ПОЯ потребует изменения входного алфавита и, что самое существенное, реализации конечного автомата, что приведет к еще большему усложнению алгоритма работы КА-распознавателя ПОЯ, а, следовательно, и аппаратуры, если делать аппаратную реализацию данного КА.

Таким образом:

- ПОЯ: "простой" алфавит — "сложный" автомат;
- АЯ: "сложный" алфавит — "простой" автомат.

Предлагаемое решение позволяет совместить оба подхода, взяв простоту входного алфавита ПОЯ и простоту конечного автомата АЯ:

- ПОЯ является естественным для конечного пользователя;
- АЯ является естественным для аппаратуры.

– КА распознавателя ПОЯ осуществляет преобразование одного заданного алфавита (ПОЯ) в другой (АЯ), также заранее определенный.

В конечном итоге очень эффективной оказалась схема предлагаемого подхода (рис. 6).

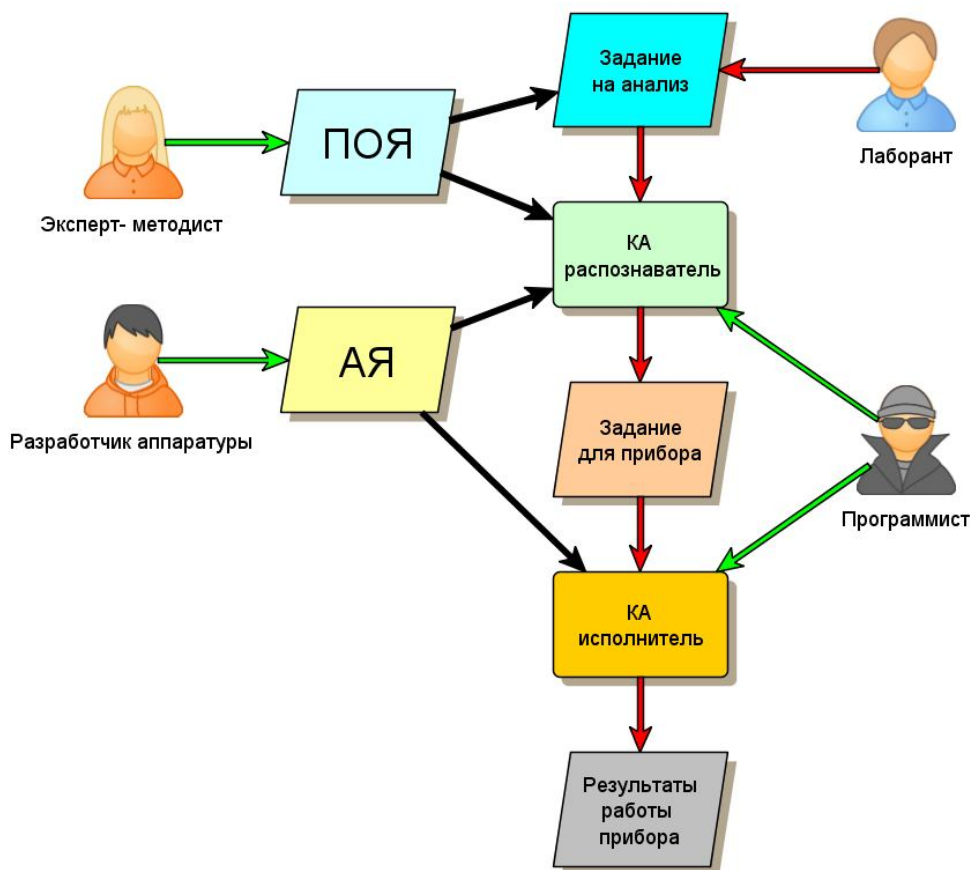


Рис. 6. Структурная схема предлагаемого подхода построения ПО аналитических приборов

Эксперт-методист формирует входной алфавит ПОЯ.

Инженер-разработчик аппаратуры формирует входной алфавит для АЯ.

Программист реализует КА-распознаватель — транслятор ПОЯ → АЯ.

Программист реализует КА-исполнитель для распознавания АЯ.

Лаборант, используя ПОЯ, формирует задание на анализ, которое с помощью КА-распознавателя преобразуется на основе АЯ в задание для прибора. КА-исполнитель обрабатывает задание для прибора и формирует результаты работы прибора.

Наличие промежуточного автомата, связываю-

щего ПОЯ и АЯ, позволяет проводить верификацию алгоритма на основе ПОЯ без запуска реальных измерений, что повышает надежность измерений.

Простой в реализации КА-исполнитель и входной алфавит на основе АЯ позволяют ускорить разработку ПО для АПК и упростить процедуру тестирования аппаратуры.

РЕЗЮМЕ

Предлагаемый подход позволяет Заказчику, Методисту, Разработчику, Программисту, Пользо-

вателю однозначно понимать, что должно быть сделано, что делается и что сделано при разработке АПК.

Позволяет разделять работу, а, самое главное, ответственность между специалистами различных областей знаний, а при необходимости и между организациями.

Обеспечивает возможность уже на ранних стадиях проектирования учесть все детали технического задания и продемонстрировать Заказчику в удобной и наглядной для него форме (на основе ПОЯ), как оно понято.

Излагаемый подход позволяет снять с Программиста необходимость знания технологического процесса, с Разработчика — тонкостей программирования, а с Методиста — знаний тонкостей и программирования и особенностей аппаратуры. Подход дает возможность Программисту ничего не додумывать за Заказчика, Методиста и Разработчика, а только однозначно реализовать формализованную спецификацию, что позволяет резко снизить его трудозатраты, а в конечном счете перейти к автопрограммированию Методистом и Разработчиком.

Следствия применения описываемого подхода

1) Использование всех наработок в области теории конечных автоматов для алгоритмов управления и алгоритмов обработки АПК АНК:

- создание проблемно ориентированного языка управления экспериментом;
- преобразование КА с целью упрощения алгоритма, оптимизации работы оборудования;
- повышение надежности работы АПК.

2) Возможность работы с моделью.

3) Использование модели на этапе разработки алгоритмов управления, которую в таком случае можно совместить с разработкой самого оборудования.

4) Использование модели для создания обучающих программ, когда работа АПК сопряжена с большими затратами на расходные материалы или само оборудование дорогостоящее.

ВЫВОДЫ

1. Предложена новая методика проектирования АПК, состоящая из четырех этапов.

—Первый этап — общепринятый подход концептуального проектирования, выполняемый в рамках эскизного проекта. Построение поведенческих, структурных, функциональных и информационных моделей АПК.

—Второй этап — разработка ПОЯ на основе моделей, полученных на первой стадии.

—Третий этап — выделение в рамках ПОЯ автоматных языков, соответствующих реальным алгоритмам работы аппаратуры.

—Четвертый этап — разработка конечных автоматов, реализующих автоматные языки и конечный автомат-транслятор ПОЯ в автоматные языки.

2. Предложены оригинальные базовые решения по разработкам автоматных языков программирования, обеспечивающие упрощение алгоритмов работы с прибором, повышение гибкости работы с прибором, снижение трудозатрат программиста при реализации разнообразных алгоритмов работы с прибором.

3. Разработана оригинальная система вычислительных абстракций архитектурного уровня, определяющая стратегию проектирования аппаратно-программных комплексов (АПК АНК).

4. Разработана методика проектирования АПК АНК, базирующаяся на теории конечных автоматов. Разработаны методы проектирования АПК АНК различного типа.

5. Показаны преимущества предлагаемого подхода, позволяющие использовать всю мощь теории конечных автоматов при оптимизации КА.

СПИСОК ЛИТЕРАТУРЫ

1. Алексеев Я.И., Белов Ю.В. и др. Приборы для диагностики биологических объектов на основе метода полимеразной цепной реакции в реальном времени (ПЦР-РВ) // Научное приборостроение. 2006. Т. 16, № 3. С. 132–136. URL: [http:// 213.170.69.26/mag/2006/full3/Art14.pdf](http://213.170.69.26/mag/2006/full3/Art14.pdf).
2. Шалыто А.А. Автоматное проектирование программ. Алгоритмизация и программирование задач логического управления // Известия РАН. Теория и системы управления. 2000. № 6. С. 63–81.
3. Поликарпова Н.И., Шалыто А.А. Автоматное программирование. СПб.: Питер, 2009. 176 с.
4. Васильев С.С., Новосельцев В.Б. Об использовании в программировании проблемно-ориентированных языков // Известия Томского политехнического университета. 2008. Т. 313, № 5. С. 68–72.
5. Глушков В.М. Введение в АСУ. Изд. 2-е испр. и доп. Киев: Техника, 1974. 319 с.
6. Никаноров С.П., Никитина Н.К., Теслинов А.Г. Введение в концептуальное проектирование АСУ: Анализ и синтез структур. Изд. 2-е. М.: Концепт, 2007. 234 с.
7. Марченков С.С. Конечные автоматы. М.: Физматлит, 2008. 56 с.
8. Якубайтис Э.А., Васюкевич В.О., Гобземис А.Ю. и др. Теория автоматов // Итоги науки и техн. Сер.: Теор. вероятн. Мат. стат. Теор. кибернет. 1976. Т. 13. С. 109–188.
9. URL: <http://syntol.ru/bitrix/docs/GM-415-instrukciya.pdf>.
10. Ребриков Д.В., Саматов Г.А., Трофимов Д.Ю. и др. ПЦР в "реальном времени". БИНОМ, 2009. 223 с.

**Институт аналитического приборостроения РАН,
г. Санкт-Петербург**

Контакты: *Петров Александр Иванович*,
fataip@rambler.ru

Материал поступил в редакцию: 4.04.2016

THE TRANSITION FROM A DOMAIN SPECIFIC LANGUAGE TO THE FINITE STATE LANGUAGE IN THE DESIGN ANALYTICAL INSTRUMENTS

A. I. Petrov

Institute for Analytical Instrumentation of RAS, Saint-Petersburg, Russia

Automata-based programming basic provisions are considered and advantages of its using in software development are justified in the article. The method of construction of the domain specific language based on behavioral, functional, informational and structural models developed at the outline design stage when developing analytical instruments is described and examples of successful application of the proposed approach in practice are given.

Keywords: problem-oriented programming language, automatic programming, automatic language, finite-state machine, analytical devices

REFERENCES

1. Alekseev Ya.I., Belov Yu.V., Varlamov D.A., Konovalev S.V., Kurochkin V.E., Marakushin N.F., Petrov A.I., Petryakov A.O., Rumyantsev D.A., Skoblilov E.Yu., Sokolov V.N., Fesenko V.A., Chernyshev A.V. [Devices for diagnostics of biological objects based on the real-time polymerase chain reaction (RT-PCR) method]. *Nauchnoe Priborostroyeniye* [Scientific Instrumentation], 2006, vol. 16, no. 3, pp. 132–136. URL: [http:// 213.170.69.26/mag/2006/full3/Art14.pdf](http://213.170.69.26/mag/2006/full3/Art14.pdf) (In Russ.).
2. Shalyto A.A. [Automatic design of programs. Algorithmization and programming of problems of logical management]. *Izvestiya RAN. Teoriya i sistemy upravleniya* [News of the Russian Academy of Sciences. Theory and control systems], 2000, no. 6, pp. 63–81. (In Russ.).
3. Polikarpova N.I., Shalyto A.A. *Avtomatnoye programmirovaniye* [Automata-based programming]. Saint-Petersburg, Piter Publ., 2009. 176 p. (In Russ.).
4. Vasil'ev S.S., Novosel'cev V.B. [About use in programming of problem-oriented languages]. *Izvestiya Tomskogo politekhnicheskogo universiteta* [News of Tomsk polytechnical university], 2008, vol. 313, no. 5, pp. 68–72 (In Russ.).
5. Glushkov V.M. *Vvedeniye v ASU* [Introduction to Automated Control Systems]. Second edition. Kiev, Tekhnika Publ., 1974. 319 p. (In Russ.).
6. Nikanorov S.P., Nikitina N.K., Teslinov A.G. *Vvedeniye v konceptual'noye proektirovaniye ASU: Analiz i sintez struktur* [Introduction to conceptual design of ACS: Analysis and synthesis of structures]. Second edition. Moscow, Koncept Publ., 2007. 234 p. (In Russ.).
7. Marchenkov S.S. *Konechnyye avtomaty* [Finite state machine]. Moscow, Fizmatlit Publ., 2008. 56 p. (In Russ.).
8. Yakubajtis E.A., Vasyukevich V.O., Gobzemis A.Yu., Zaznova N.E., Kurmit A.A., Lorenc A.A., Petrenko A.F., Chapenko V.P. [Automata theory]. *Itogi nauki i tekhniki. Seriya: Teoriya veroyatnostey. Matematicheskaya statistika. Teoreticheskaya kibernetika* [Results of science and equipment. Series: Probability theory. Mathematical statistics. Theoretical cybernetics], 1976, vol. 13, pp. 109–188 (In Russ.).
9. "Plant / 35S+FMV/NOS screening". Test system for detection of GMO of a phytogenesis. URL: [http:// syntol.ru/bitrix/docs/GM-415-instrukciya.pdf](http://syntol.ru/bitrix/docs/GM-415-instrukciya.pdf) (In Russ.).
10. Rebrikov D.V., Samatov G.A., Trofimov D.Yu., Semyonov P.A., Savilova A.M., Kofiadi I.A., Abramov D.D. *PCR v "real'nom vremeni"* [PCR in "real time"]. Moscow, Binom Publ., 2009. 223 p. (In Russ.).

Contacts: *Petrov Aleksandr Ivanovich*,
fataip@rambler.ru

Article received in edition: 4.04.2016