Моделирование и анализ информационных систем. Ярославль: ЯрГУ. Т.**13**, №**1** (2006) 27–34

УДК 519.68/.69

Иерархическая модель автоматных программ

Кузьмин Е.В.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 22 03 2006

Аннотация

В работе описывается иерархическая модель программ, построенных на основе автоматного подхода к программированию. Приводится пример автоматной модели системы управления кофеваркой.

1. Введение

Статья посвящена описанию иерархической модели программ, построенных на основе автоматного подхода к программированию [2, 3, 4, 5]. Технология автоматного программирования является достаточно эффективной при построении программного обеспечения «реактивных» систем и систем логического управления. Эта технология, не исключая других методов построения программного обеспечения «без ошибок», существенно более конструктивна, так как позволяет начинать «борьбу с ошибками» еще на стадии алгоритмизации. Более того, автоматный подход к программированию с точки зрения моделирования и анализа программных систем имеет ряд преимуществ по сравнению с традиционным подходом. При построении модели для программы, написанной традиционным способом, возникает серьёзная проблема адекватности этой программной модели для исходной программы. Модель может не учитывать ряд программных свойств или порождать несуществующие свойства. При автоматном программировании такая проблема исключена, поскольку набор взаимодействующих автоматов, описывающий логику программы, уже является адекватной моделью, по которой формально и изоморфно строится программный модуль. И это является бесспорным плюсом автоматной технологии. К тому же свойства программной системы в виде автоматов формулируются и специфицируются естественным и понятным образом. Проверка свойств осуществляется в терминах, которые естественно вытекают из автоматной модели программы.

При автоматном подходе к проектированию и построению программ для задач логического управления выделяются две части: системно независимая и системно зависимая. Первая часть реализует логику программы и задаётся системой взаимодействующих автоматов Мура–Мили. Проектирование каждого автомата состоит в создании по словесному описанию (декларации о намерениях) схемы связей, описывающей его интерфейс, и графа переходов, определяющего его поведение. По этим двум документам формально и изоморфно может быть построен модуль программы (системно зависимая часть), соответствующий автомату.

2. Иерархическая модель автоматных программ «реактивных» систем и систем логического управления

Рассмотрим иерархическую систему взаимодействующих детерминированных конечных автоматов

$$\mathcal{A} = (A_0, A_{11}, \ldots, A_{1k_1}, \ldots, A_{n1}, \ldots, A_{nk_n}),$$

где n и k_i $(1 \le i \le n)$ — натуральные числа. Автомат A_0 назовём основным, остальные автоматы — вложенными. Между всеми автоматами установлена иерархия по вызываемости (вложенности). Автомат A_{ij} может вызывать (передавать управление) стоящему ниже по иерархии автомату A_{i+1k}. В таком случае автомат A_{ij} называется главным, а A_{i+1k} — вложенным (или вызываемым). У каждого вложенного автомата существует только один главный автомат, из которого он вызывается.

Система автоматов $\mathcal{A}$ рассматривается как система управления некоторым объектом. Система $\mathcal{A}$ получает от объекта управления события, характеризующие, например, изменение его состояния, а также сама запрашивает текущие параметры объекта, что также считается входным воздействием на систему $\mathcal{A}$. В то же время система управления, реагируя на поступающую информацию, сама оказывает воздействия на объект управления.

Кроме описанного взаимодействия с «внешней средой», происходит аналогичное взаимодействие автоматов между собой внутри системы, посредством передачи (со стороны главного автомата) управления с некоторым событием вложенным автоматам и отслеживанием их текущих состояний.

В рассматриваемой модели события от объекта управления принимает и реагирует на них только основной автомат A_0, вложенные же автоматы обращаются к объекту управления только с запросами состояния его параметров (причём вложенный автомат может сделать запрос лишь в том случае, если главный автомат передал ему управление).

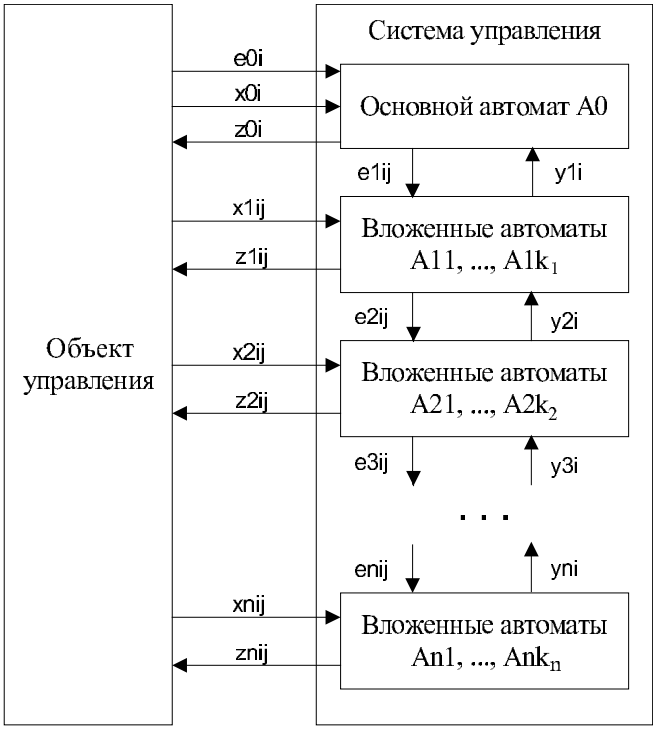

Рис. 1. Модель взаимодействия системы $\mathcal{A}$ с объектом управления

Описываемая модель взаимодействия системы управления $\mathcal{A}$ с объектом управления показана на рис. 1. На рисунке события обозначаются символом **e**, запросы к объекту управления (входные воздействия) — **x**, текущее состояние некоторого автомата A_{ij} хранится в переменной $\mathbf{y}_{ij}$, а выходные воздействия обозначаются **z**.

Поведение любого автомата A системы $\mathcal{A}$ аналогично поведению всей системы в том смысле, что автомат A реагирует на произошедшие события и в зависимости от своего состояния, состояния вложенных автоматов и состояния объекта управления оказывает воздействие на объект управления или же на вложенные автоматы, передавая им управление с некоторым событием.

Обозначим $E_A = \{e_1, \ldots, e_k\}$ множество всех имён событий, на которые реагирует автомат A. Определим на множестве E_A переменную событий e, в которую будет помещаться имя произошедшего события для автомата A. Текущее значение переменной e будем обозначать $E_A(e)$.

Также введём множество $X_A = \{x_1, \ldots, x_n\}$ запросов автомата A к объекту управления. Каждый запрос расценивается как некоторый предикат, истинность которого зависит от состояния параметров объекта управления.

Наконец, введём $Z_A = \{z_1, \ldots, z_r\}$ множество выходных воздействий автомата A. Причём воздействия z_i рассматриваются двух видов. Первая группа воздействий — это воздействия непосредственно на объект управления. Воздействия второго типа — передача управления вложенному автомату с некоторым событием (генерация события для вложенного автомата). В этом случае выходное воздействие z_i имеет вид $A'(e'_j)$, где A' — вложенный автомат, а e'_j — сгенерированное событие автоматом A для вложенного автомата A', которому передаётся управление для обработки этого события.

Тогда автомат A управляющей системы автоматов $\mathcal{A}$ представляет собой набор $(\Sigma, Q, q_0, E, X, Z, \delta)$, где

- $Q = \{q_0, q_1, \ldots, q_n\}$ — конечное множество состояний автомата A;
- q_0 — начальное состояние;
- $\Sigma = \{a_1, a_2, \ldots, a_k\}$ — конечный алфавит пометок дуг переходов;
- δ: $Q \times \Sigma \to Q$ — функция переходов из одного состояния в другое.

Каждый переход срабатывает по определённому правилу. Прежде чем описывать правила переходов, введём ещё ряд обозначений.

Для метки перехода $a \in \Sigma$ обозначим $E(a)$ событие, на которое автомат A реагирует при срабатывании перехода с меткой a.

Будем обозначать $X(a)$ множество запросов к объекту управления, истинность которых требуется для срабатывания перехода с пометкой a.

Пусть Z^* — множество конечных последовательностей выходных воздействий, тогда для $a \in \Sigma$ обозначим $Z^*(a) \in Z^*$ последовательность выходных воздействий, которая происходит при срабатывании перехода с пометкой a.

Также для произвольного состояния q автомата A обозначим $Z^*(q) \in Z^*$ последовательность выходных воздействий, которая должна выполниться при попадании (переходе) автоматом в состояние q.

И, наконец, пусть $Y(a)$ — предикат, зависящий от состояний вложенных автоматов, истинность которого необходима для срабатывания перехода с пометкой a.

Правило перехода из состояния q в состояние q' по метке a имеет следующий вид.

q, a: if $e = E(a)$ and $(x = \text{true}, \forall x \in X(a))$ and $Y(a) = \text{true}$
then $Z^*(a)$; $Z^*(q')$; goto q'.

Таким образом, правило перехода в новое состояние можно описать так. В общем случае, после получения события автомат в зависимости от своего текущего состояния реагирует (или никак не реагирует) на событие, опрашивает параметры объекта управления, учитывает состояния вложенных автоматов, затем производит последовательность выходных воздействий, включая и те выходные воздействия, которые необходимо совершить при попадании в новое состояние, и только после этого переводится в новое состояние (которое может быть тем же самым в случае петли).

Выходное воздействие первого типа, направленное на объект управления, считается сразу же осуществлённым после его применения. Выходное воздействие второго типа, представляющее собой передачу управления с событием вложенному автомату, считается выполненным только лишь после реакции вложенного автомата на это событие, которая заключается в том, что либо автомат переходит в новое состояние (срабатывает один из переходов вложенного автомата), либо событие игнорируется вложенным автоматом (ни один из переходов сработать не может). До тех пор пока выходное воздействие второго типа не осуществится, работа (процесс перехода в новое состояние) главного автомата приостанавливается.

Важно отметить, что правила переходов, а точнее условия срабатывания переходов, должны удовлетворять условию детерминированности (или ортогональности), т. е. при наступлении некоторого события может быть готов к срабатыванию не более чем один переход. Если ни один переход в текущем состоянии при наступлении некоторого события сработать не может (условия переходов не выполняются), то событие игнорируется (переменная событий e для этого автомата обнуляется).

Далее рассмотрим иерархическую модель автоматных программ на примере системы управления кофеваркой.

3. Автоматная модель системы управления кофеваркой

Рассмотрим системно независимую часть автоматной программы, которая отвечает за *логику* управления кофеваркой.

Модель кофеварки представлена на рис. 2. Кофеварка позволяет выбирать количество порций кофе для варки с помощью кнопок «+» (увеличить количество порций на единицу) и «−» (уменьшить количество порций на единицу).

В кофеварке предусмотрена возможность индикации отсутствия воды и основных неисправностей (например, не работает нагреватель или не исправен один из клапанов). Количество порций изменяется от 1 до 5. При попытке увеличить максимальное значение порций нажатием кнопки «+» ничего не происходит

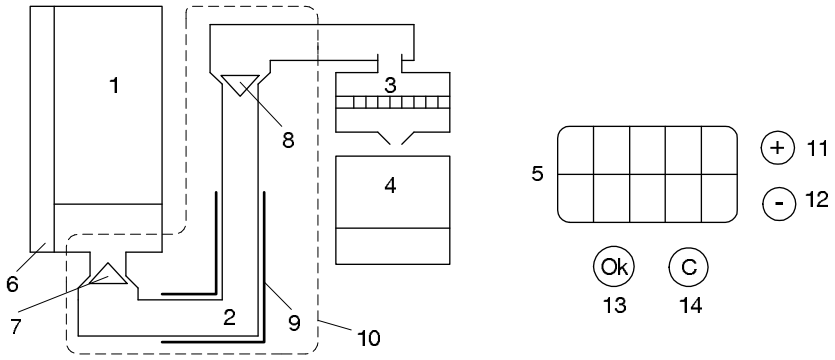

Рис. 2. Кофеварка: 1 — емкость для воды, 2 — емкость для кипячения, 3 — фильтр с кофе, 4 — колба для готового напитка, 5 — дисплей, 6 — датчик воды, 7 — входной клапан (клапан 1), 8 — выходной клапан (клапан 2), 9 — нагреватель кипячения воды, 10 — бойлер, 11 — кнопка увеличения «+», 12 — кнопка уменьшения «−», 13 — кнопка «Ok», 14 — кнопка «C» (отмена).

(аналогично для значения порций 1 нажатие «−» игнорируется). Отдельно выделяется объект «бойлер», который состоит из клапанов 1 и 2, нагревателя и емкости для кипячения. Дисплей кофеварки разбит на части, использующиеся для индикации состояний клапанов, нагревателя и собственно кофеварки.

В кофеварке выделяются чётко выраженные подобъекты управления (бойлер, клапаны и нагреватель), которые имеют свои собственные подсистемы управления. Каждая подсистема управления представлена в виде конечных автоматов Мура–Мили, выстроенных в иерархию. В результате логическая часть системы управления кофеваркой имеет вид системы взаимодействующих автоматов. Автомат, находящийся выше по иерархии, управляет своими вложенными автоматами путем генерации события и передачи им управления с этим событием. Кроме того, автомат следит за состояниями вложенных автоматов, так как от них могут зависеть его собственные переходы по состояниям.

Диаграмма взаимодействия автоматов представлена на рис. 3. Автомат $A0$, называемый основным, получает от панели управления (панели кнопок) события $e0i$, на которые он реагирует. Специальное событие $e0$, генерируемое постоянно системным таймером, используется для проверки условий переходов (условий на дугах автоматов), которые не предусматривают реакцию ни на одно из событий $e0i$. Автомат $A0$ взаимодействует с автоматом $A1$, передавая ему управление с событиями $e1i$ и $e0$. Также автомат $A0$ отслеживает состояния $A1$ через переменную $y1$. При таком взаимодействии автомат $A0$ считается главным, а $A1$ — вложенным. Аналогичным образом происходит взаимодействие автомата $A1$ с вложенными автоматами $A2$, $A31$ и $A32$.

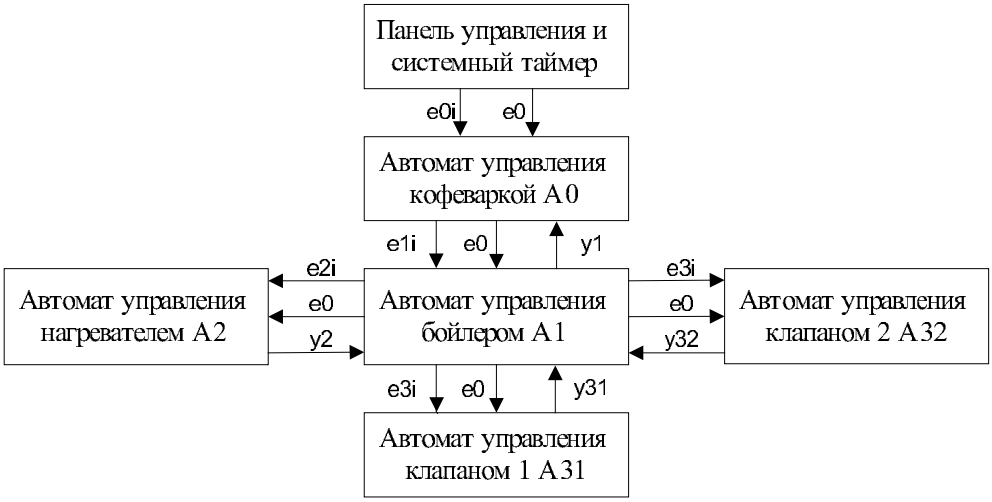

Рис. 3. Диаграмма взаимодействия автоматов управления

Автомат $A0$ реализует логику управления кофеваркой. Он реагирует на нажатие кнопок на панели управления (при задании количества порций, запуске процесса варки, отмене варки, сбросе ошибки и т. д.), передавая управление на время варки кофе взаимодействующему с ним автомату управления бойлером $A1$. Информация о текущих состояниях кофеварки отображается на дисплее.

Схема связей и граф переходов автомата управления кофеваркой $A0$ представлены на рис. 4.

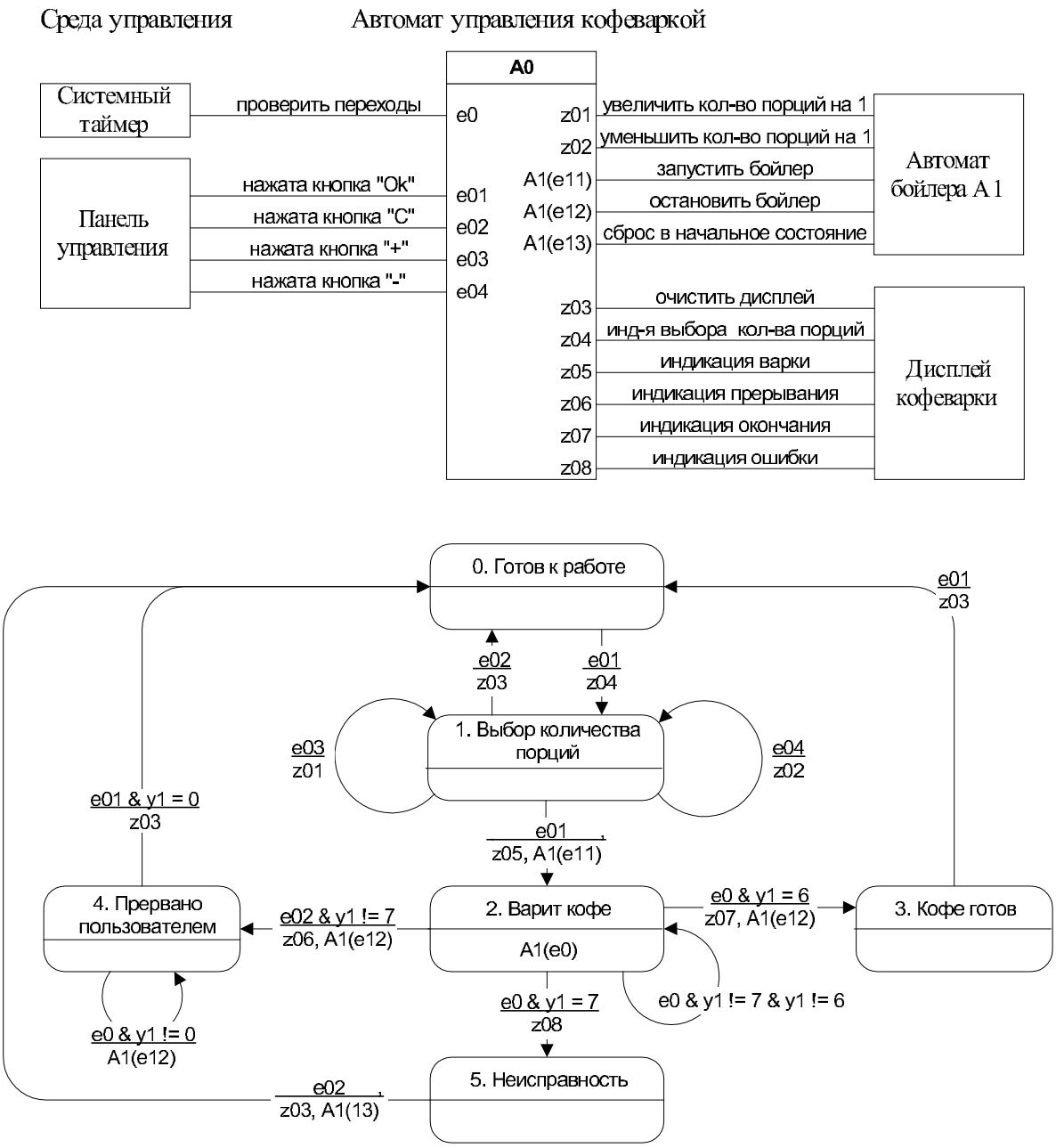

Рис. 4. Схема связей и граф переходов автомата управления кофеваркой $A0$

Автомат $A1$ реализует логику управления бойлером, который непосредственно отвечает за процесс варки кофе.

В процессе варки контролируются работоспособность клапанов, нагревателя, а также проверяется наличие воды. В случае неисправности одного из клапанов или нагревателя варка прерывается и высвечивается на дисплее соответствующее предупреждение. В случае отсутствия воды в емкости варка приостанавливается, пока вода не будет долита либо пользователь не прервет варку.

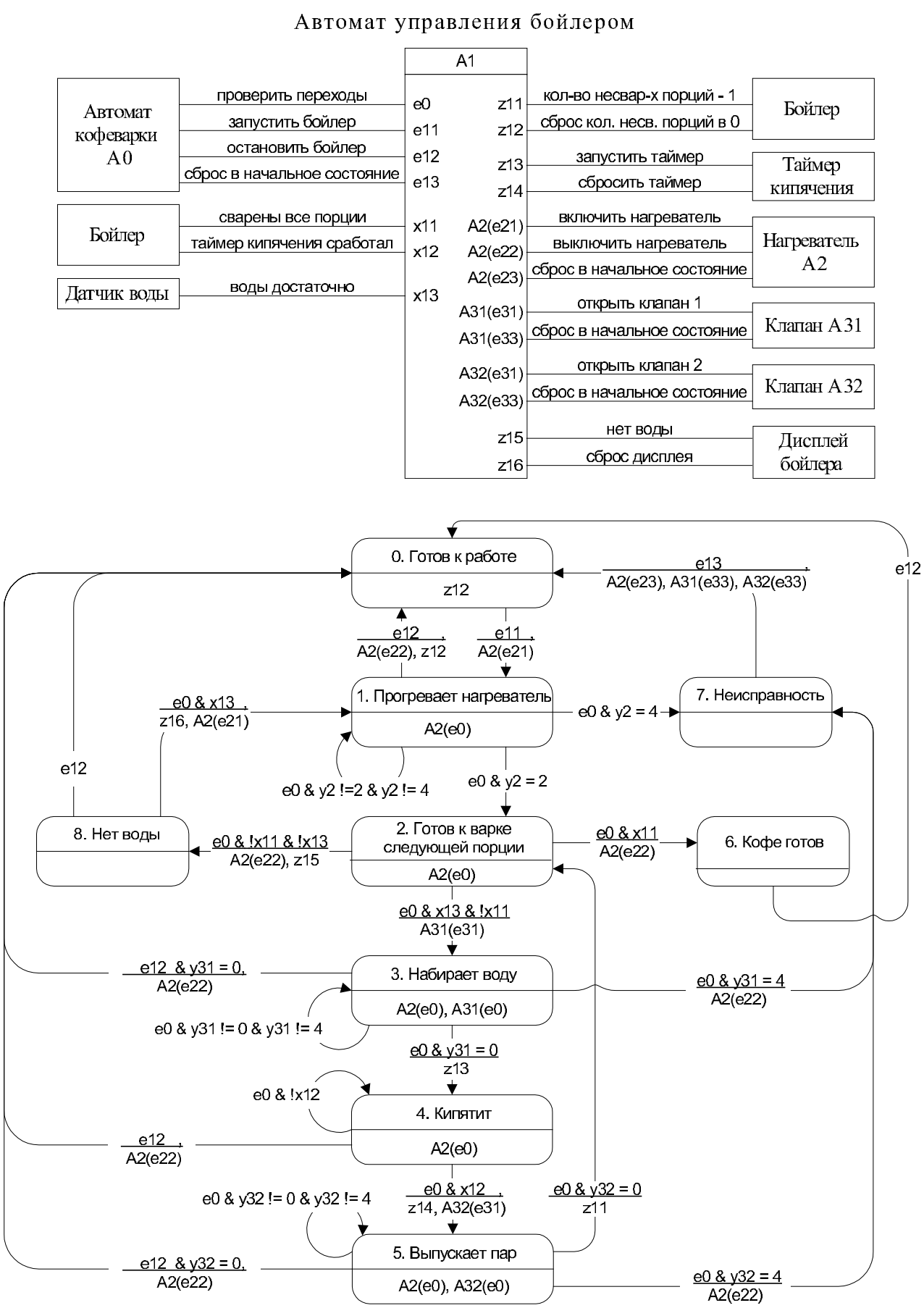

Рис. 5. Схема связей и граф переходов автомата управления бойлером $A1$

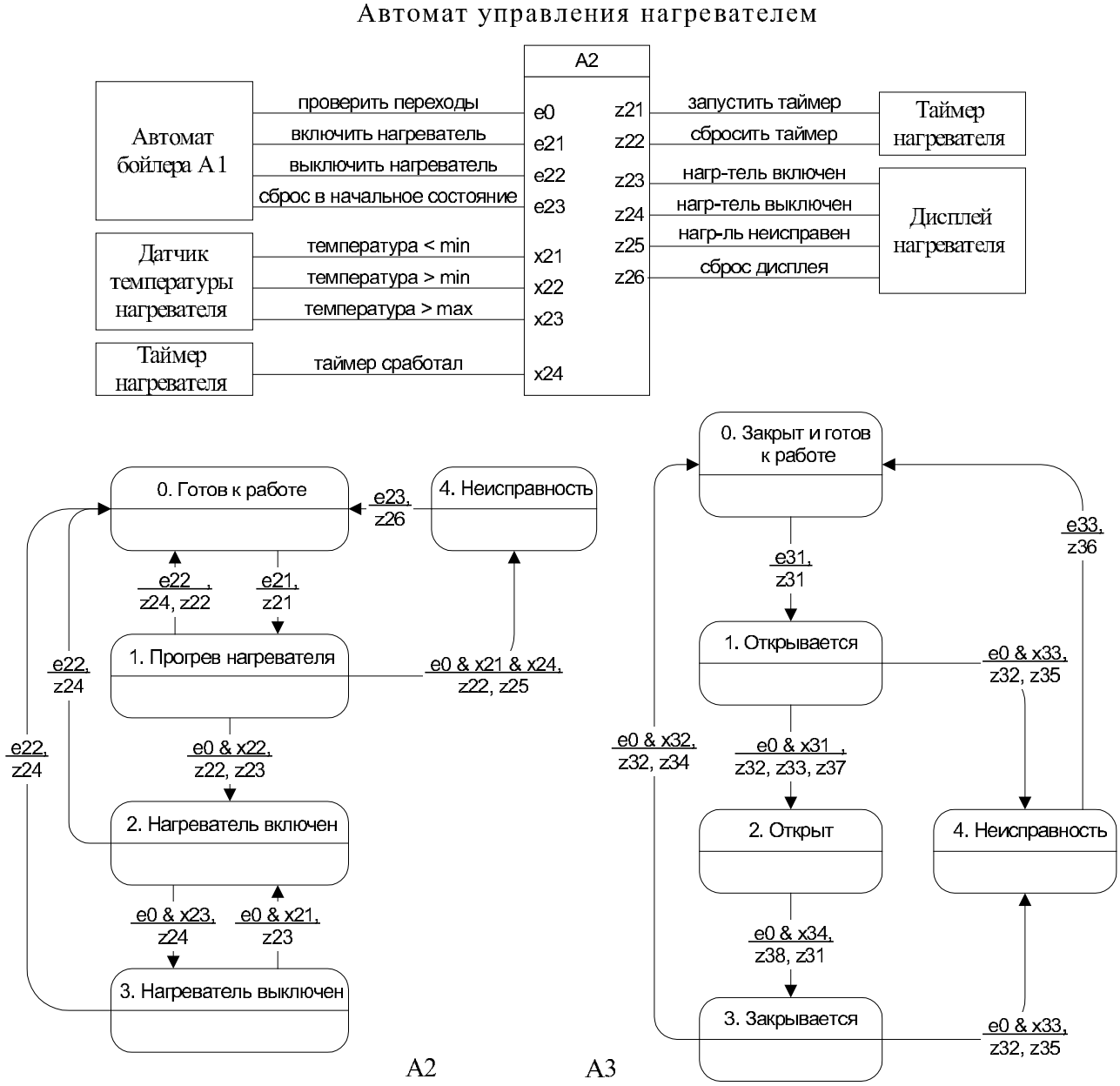

Рис. 6. Схемы связей и графы переходов автоматов управления нагревателем $A2$ и клапаном $A3$

Если не произошло поломок, процесс варки повторяется до тех пор, пока не будет сварено установленное количество порций кофе. Один цикл варки соответствует одной порции.

Автомат $A1$ взаимодействует с автоматом управления $A0$, передавая последнему текущее состояние и получая от него события. Автомат $A1$ также взаимодействует с автоматами $A31$, $A32$ (отвечающими за управление клапанами 1 и 2) и $A2$ (управляющим нагревателем), получая их состояния и посылая им события.

Схема связей и граф переходов автомата управления бойлером $A1$ представлена на рис. 5.

Автомат $A2$ реализует логику управления нагревательным элементом. Он поддерживает температуру нагревателя в заранее заданном диапазоне (для температуры устанавливаются допустимые минимум и максимум). Предусмотрена возможность установления неисправности вида «нагреватель не работает».

После включения нагревателя происходит его прогрев до нижней границы рабочего диапазона. Если он не прогрелся до необходимой температуры за заданное время, то нагреватель считается неисправным. Нагреватель включается при достижении температурой нижней границы рабочего интервала и выключается при достижении верхней границы.

Автомат $A2$ взаимодействует только лишь с автоматом управления бойлером $A1$ и не имеет вложенных автоматов.

Автоматы $A31$ и $A32$ полностью идентичны. Под автоматом $A3$ понимается как автомат $A31$, так и автомат $A32$. Автомат $A3$ служит для реализации логики управления работой клапана в режиме «открыть–пауза–закрыть». Автомат имеет четыре состояния и, не имея вложенных автоматов, взаимодействует только с автоматом управления бойлером $A1$, передавая последнему свое состояние и получая от него события. Автомат $A3$ начинает цикл своей работы по событию $e31$ («открыть клапан»). Сначала клапан открывается, затем делается пауза для набора воды или выпуска пара, а после этого клапан закрывается. В случае если за определенное время клапан не откроется (либо не закроется), он считается неисправным.

Схемы связей и графы переходов автомата управления нагревателем $A2$ и автомата управления клапаном $A3$ представлены на рис. 6.

4. Заключение

При построении автоматной программы в рамках иерархической модели логика программы концентрируется на основном автомате, который распределяет управление вложенным автоматам в зависимости от поведения управляемого объекта. Каждый автомат программы взаимодействует только со своими главным и вложенными автоматами, что облегчает понимание программы. Во время проектирования или верификации такой автоматной программы возможно рассмотрение части или некоторого поддерева системы автоматов в зависимости от той функции, которая реализуется выделенной подсистемой автоматов. Любая подсистема взаимодействующих автоматов в иерархической модели представляет собой дерево автоматов, которое можно рассматривать как отдельную систему (как отдельную автоматную программу). Это позволяет менять масштаб всей системы, относя не интересующие проектировщика в данный момент времени автоматы к внешней среде, т. е. к внешнему объекту управления, и удерживать во внимании только анализируемую подсистему. И, наконец, при верификации спецификация и анализ свойств автоматной программы проводятся проще при понятной и не сложной по структуре верифицируемой модели.

Список литературы

1. *Кессель С. В.* Разработка системы управления кофеваркой на основе автоматного подхода. — http://is.ifmo.ru/projects/, 2003.

2. *Шалыто А. А.* Switch-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука, 1998. 628 с.

3. *Шалыто А. А.* Автоматное проектирование программ. Алгоритмизация и программирование задач логического управления // Известия академии наук. Теория и системы управления. 2000. №6. С. 63–81. (http://is.ifmo.ru, раздел «Статьи»).

4. *Шалыто А. А.* Алгоритмизация и программирование для задач логического управления и «реактивных» систем // Автоматика и телемеханика. Обзоры. 2001. №1. С. 3–39. (http://is.ifmo.ru, «Статьи»).

5. *Шалыто А. А., Туккель Н. И.* Программирование с явным выделением состояний // Мир ПК. 2001. №8. С. 116–121. №9. С. 132–138. (http://is.ifmo.ru, раздел «Статьи»).