Лабораторная работа № 3

Создание модели поведения системы при помощи State Diagram

Назначение диаграммы

Создание модели поведения объектов позволяет полостью описать все состояния системы и переходы в эти состояния во время работы, а также некоторые алгоритмы взаимодействия между объектами.

Единственный недостаток — это то, что пока содержание данной диаграммы никак не отражается на получаемом коде программы и это может создать у программиста ошибочное мнение, что без модели поведения можно обойтись. Но это не так. Модель поведения позволяет взглянуть на получаемый программный объект со стороны, ведь основное назначение объектно-ориентированного программирования - создавать объекты, наделенные определенным поведением, которые в дальнейшем и будут производить работу в программном коде. А данный тип диаграмм позволяет четко представить все поведение полученного программного объекта в виде графических значков состояний.

Создание заготовок классов

Для того чтобы начать создавать модель поведения, необходимо добавить в модель классы, на основе которых эти объекты впоследствии будут создаваться.

Исходя из рассмотренных ранее устройств, получаем следующих кандидатов для создания классов:

- EnvironmentalController - контроллер управления исполнительными устройствами;
- TemperatureSensor - датчик температуры;
- pHSensor — датчик кислотности;
- Heater — нагреватель;
- Cooler — вентилятор для снижения температуры;
- Light - осветитель;
- WaterTank — хранилище для воды;
- NutrientTank - хранилище для удобрений.

Использование английских наименований позволяет не опасаться, что при генерации исходного кода на выбранном языке программирования процесс может окончиться неудачей в случае конфликта с русскими именами.

Для добавления класса в модель необходимо перейти в окно Browser и на строке Logical View проделать RClick=>New=>Class, как показано на рис. 13.

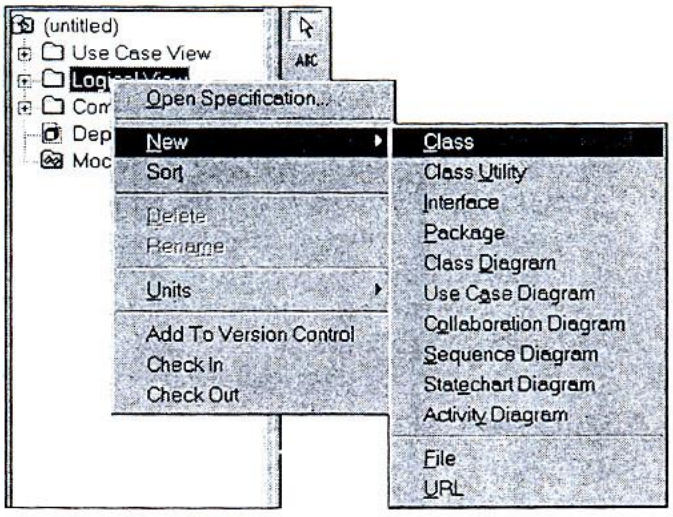

Рис. 13. Создание нового класса в окне Browser

Измените название NewClass на EnvironmentalController, и первый класс готов. Аналогично добавьте остальные классы, и можно начинать создавать модель поведения.

Создание Statechart диаграммы

В нашей системе самым сложным поведением обладает класс EnvironmentalController, поэтому будем разбирать создание модели поведения на его примере.

Для создания диаграммы есть несколько путей.

1. Создать посредством меню Menu=>Browse=>State Machine Diagram.
2. Создать посредством строки инструментов при помощи значка Statechart.
3. Выбрать пункт New Statechart Diagram из контекстного меню класса.

Версия 2000 лишена таких сложностей, по причине возможности создавать любые диаграммы в любом месте, для этого активизируется окно выбора диаграммы (рис.14).

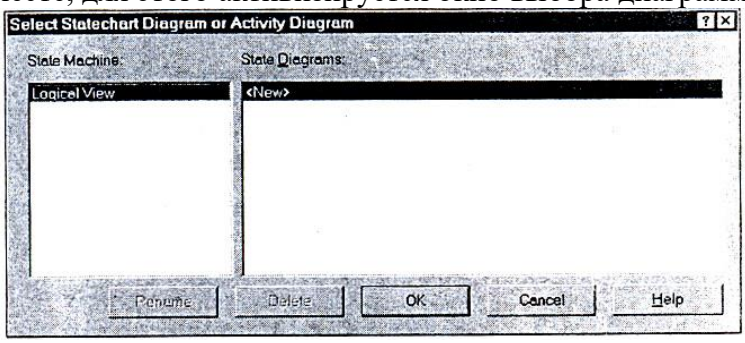

Рис.14.Выбор диаграммы Statechart или Activity

При помощи этого диалогового окна пользователь может выбрать диаграмму для назначения ее текущей, создать новую, удалить или переименовать уже существующую. При создании новой диаграммы будет предложено выбрать из двух возможных (рис. 15). Для создания диаграммы выберем тип Statechart.

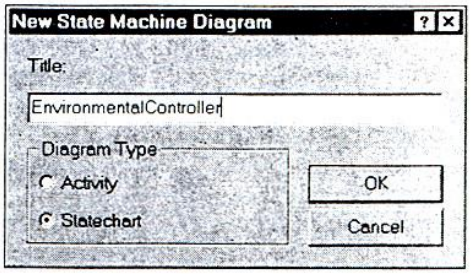

Рис. 15. Выбор типа диаграммы

Инструменты диаграммы Statechart

После активизации диаграммы становятся доступны следующие инструменты (рис.16).

Рис.16. Инструменты Statechart Diagram

State (состояние)

Инструмент State позволяет отразить состояние или ситуацию в течение жизни объекта, которая отвечает некоторому положению объекта или ожиданию им некоторого события. Каждое состояние представляет собой совокупную историю поведения объекта. Его имя должно быть уникально внутри класса, так как состояния с одинаковыми именами считаются представлением одних и тех же состояний. В текущем значке State

могут быть отражены действия по входу, выходу из состояния, действия не связанные с событиями или реакция на события.

Start State (начало)

Инструмент Start State позволяет создать значок начала работы. Для диаграммы Statechart он обозначает событие, которое переводит объект в первое состояние на диаграмме.

End State (завершение)

Инструмент End State позволяет создать значок окончания работы. Направление перехода может быть установлено только в данный значок, однако, никаких ограничений на количество переходов в End State, а также на количество таких элементов на диаграмме, не налагается.

State Transition (состояние перехода)

Инструмент State Transition позволяет создать значок состояния перехода, который означает, что объект переходит из одного состояния в другое в случае наступления определенного события или по изменению определенных условий.

Пользователь может указать несколько переходов из одного состояния в другое, в случае если каждый такой переход осуществляется при наступлении разных событий или при соблюдении разных условий.

Transition To Self (переход на себя)

Инструмент Transition To Self позволяет создать значок перехода в то же состояние, из которого осуществляется переход.

Данный переход похож на State Transition, однако, он не осуществляет переход в другое состояние при наступлении некоторого события. Таким образом, при наступлении события оно обрабатывается, и после обработки объект возвращается в то состояние, в котором он находился до наступления события.

Первые шаги в создании диаграммы

Первым шагом для создания диаграммы будет создание точки начала работы. Создайте ее при помощи кнопки Start State.

Обычно первым состоянием системы после начала работы будет ожидание наступления событий. И в нашем случае не будем делать исключений. Создадим новое состояние (State) и соединим его стрелкой State Transmission с начальной точкой.

Каждое состояние или событие должно иметь свое имя, поэтому присвоим имя состоянию ожидания RClick=>OpenSpecification=>General=>Name=>Idle, что в переводе означает «ожидание» (рис.17).

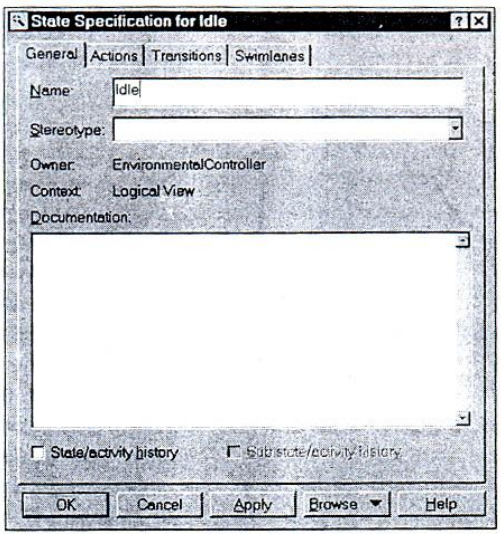

Рис.17 Вкладка General для спецификаций состояния

Для того чтобы назвать событие, которое переводит систему из начального состояния в состояние ожидания, выделим стрелку события и проделаем следующее RClick=>OpenSpecification=>General=>Event=>New Planting в переводе означает «посадка семян». Должно получиться состояние, показанное на рис. 18.

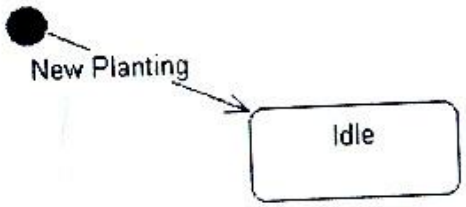

Рис.18. Состояние ожидания

Состояние тестирования датчиков

Задача нашего контроллера состоит в поддержании заданных значений параметров среды теплицы: температуры, освещенности, концентрации pH. Для того чтобы эти параметры поддерживать согласно плану, необходимо считывать текущее состояние среды с помощью датчиков.

В нашем случае не будем рассчитывать на большую интеллектуальность датчиков. Датчики не будут иметь своего процессора и будут только выдавать измененные параметры по запросу контроллера. Таким образом, следующим состоянием будет опрос датчиков. Добавим это состояние на диаграмму, назовем Testing Environment и соединим с состоянием Idle. Это событие назовем Timer. Имеется в виду, что у контроллера есть встроенные часы, которые через заданное время инициализируют это событие.

Теперь разберем подробнее, что необходимо сделать для тестирования параметров среды.

Активизировавшись контроллер опрашивает датчики температуры и pH. Для того чтобы отразить, что опрос датчиков происходит в течение данного состояния, необходимо добавить новое действие (Action). Для этого во вкладке Actions из контекстного меню выбираем Insert.

Выберем полученное действие двойным нажатием мыши и попадем в диалоговое окно, представленное на рис.19.

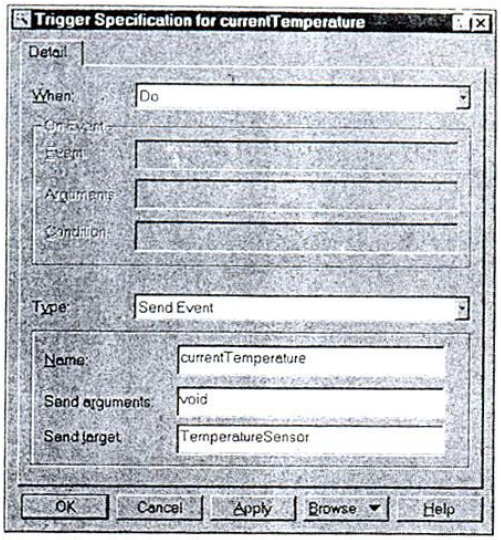

Рис.19. Настройка параметров действия

Здесь мы можем переключаться между действием (Action) и посылкой сообщения (Send Event). Разница между этими пунктами в том, что действие осуществляется самим классом, для которого мы создаем диаграмму состояния, то есть здесь вызывается метод

этого класса, а посылка сообщения направлена на объект другого класса, метод которого вызывается при помощи этого сообщения. Этот объект задается в строке Send target. Также можно задать имя вызываемого метода класса в строке Send event и аргументы в строке Send arguments.

Можно настроить момент, в который происходит отмеченное действие. Здесь предоставляется выбор из четырех состояний:

- On entry показывает что указанное действие необходимо производить При входе в состояние до выполнения остальных действий.
- On exit показывает, что действие должно быть выполнено перед самым выходом из указанного состояния.
- Do — действия, производимые в течение состояния до выхода. Если таких действий набирается несколько, то почти наверняка их можно выделить в отдельную диаграмму состояния.
- On Event — действия в ответ на определенные события, обрабатываемые в текущем состоянии. При выборе этого пункта открывается возможность заполнить события (Event), параметры (Arguments) и условия, когда это действие обрабатывается (Condition), то есть может случиться так, что событие произошло, а условие его обработки не наступило. Этот пункт удобно использовать при обработке события, которое не приводит к переходу в другие состояния и отражается значком Transition To Self.

Нужно заметить, что при изменении этих параметров изменяется и надпись на значке состояния.

События отражаются при помощи символа ^ перед ним. Действия: при входе: entry:, при выходе: exit:, в течение работы: do:, действия по сообщению: on:. Условие обработки показывается выражением в квадратных скобках. Данная нотация довольно удобна и позволяет, не активизируя окно свойств, одним взглядом оценить сделанное.

Заполним событие, как показано на рис. 19. По аналогии заполним действие для опроса датчика pH и получим рис. 20.

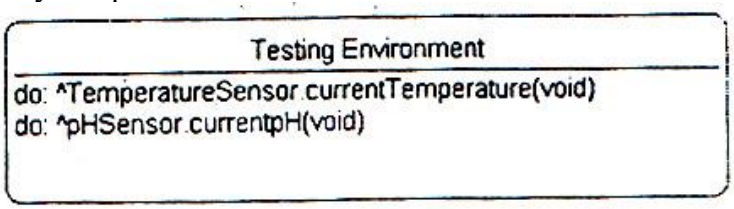

Рис. 20. Значок состояния при тестировании среды после изменении

Вывод диаграммы из цикла

После того как протестировано состояние среды, необходимо привести показатели к норме посредством включения соответствующих исполнительных устройств. Для отражения этого действия создадим состояние Adjusting Environment, которое соединим с состоянием Testing Environment.

После настройки среды система переходит в ожидание следующего момента времени. Отразим это при помощи стрелки, соединяющей состояния Adjusting Environment и Idle.

Однако у нас получилась система с бесконечным циклом. У нее нет выхода. Но мы знаем, система должна закончить работу в тот момент, когда урожай созрел. Ключом созревания будем считать истечение времени выращивания растения по плану выращивания.

Для отражения этого введем состояние Analysing Time, которое включим между состоянием Idle и Testing Environment.Перетащим конец стрелки Timer на Analysing Time и соединим его новой стрелкой Testing Environment. Добавим End State, соединим это состояние с Analysing Time, заполним условие RClick=>OpenSpecification=>Detail (рис.

21), говорящее о том, что переход осуществляет только тогда, когда истекло время роста растения. Это окно имеет практически те же заполняемые поля, что и рассмотренное ранее, поэтому не будем на нем подробно останавливаться. Единственное, что хотелось бы отметить, это то, что стрелку можно перенести прямо отсюда, изменив позиции From (откуда) и To (куда).

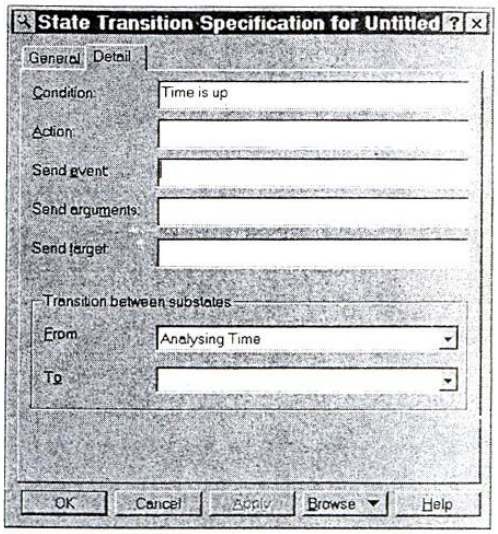

Рис.21.Заполнение условия выполнения перехода

Добавление замечания

Для добавления последнего, еще не использованного нами значка, введем в диаграмму комментарий, который соединим при помощи значка Anchor Note to Item с состоянием Analysing Time, и должно получиться примерно следующее (рис. 22).

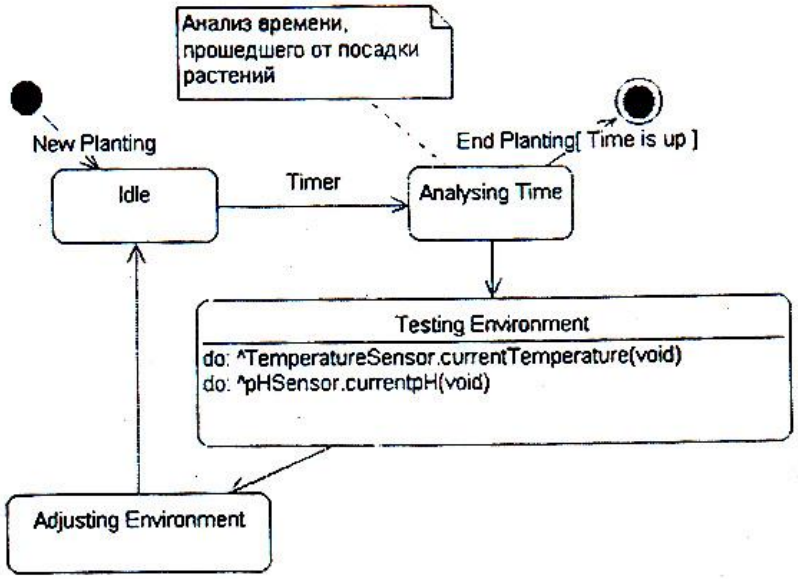

Рис.22. Диаграмма состояния после добавления замечания

Настройка среды

Распишем подробнее, какие состояния должна пройти система для настройки среды в соответствии с планом выращивания, при этом сделаем следующие допущения

- для увеличения температуры в теплице необходимо включить нагреватель;
- для уменьшения температуры необходимо включить вентилятор;
- для переключения в режим «День» необходимо включить освещение;
- для переключения в режим «Ночь» необходимо выключить освещение;
- для уменьшения pH необходимо добавить в раствор воды;

- для увеличения pH необходимо добавить в раствор удобрений.

Причем возможно одновременное включение нагревателя, лампочек и водяной заслонки, но должно быть невозможно одновременное включение нагревателя и вентилятора, а также одновременное поступления воды и удобрений.

Произведя анализ необходимых действий, нетрудно заметить, что перечисленный набор можно разделить на три независимые части, это:

- настройка температуры;
- настройка освещения;
- настройка уровня pH.

Так как мы приняли допущение, что процессор у нас только один, то эти состояния будут пройдены последовательно. Но если для каждой части используется свой процессор, то можно было бы создать автономные классы для управления этими состояниями.

Rational Rose предоставляет возможность создания вложенных диаграмм состояния, что удобно для большей детализации каждого состояния. Воспользуемся этим в нашем случае.

Добавим три новых состояния Adjusting t (настройка температуры), Adjusting Light (настройка освещения) и Adjusting pH (настройка pH). Для этого добавим новые состояния прямо в значок Adjusting Environment. Перед добавлением значок будет выделен рамкой, а после добавления будет автоматически раздвинут, чтобы в него уместился добавленный элемент. Изменим направления входящей и выходящей стрелок так, чтобы входящая указывала на Adjusting pH, а выходящая исходила из Adjusting t, и соединим эти состояния последовательно. Теперь распишем, что происходит в этих состояниях (рис. 23).

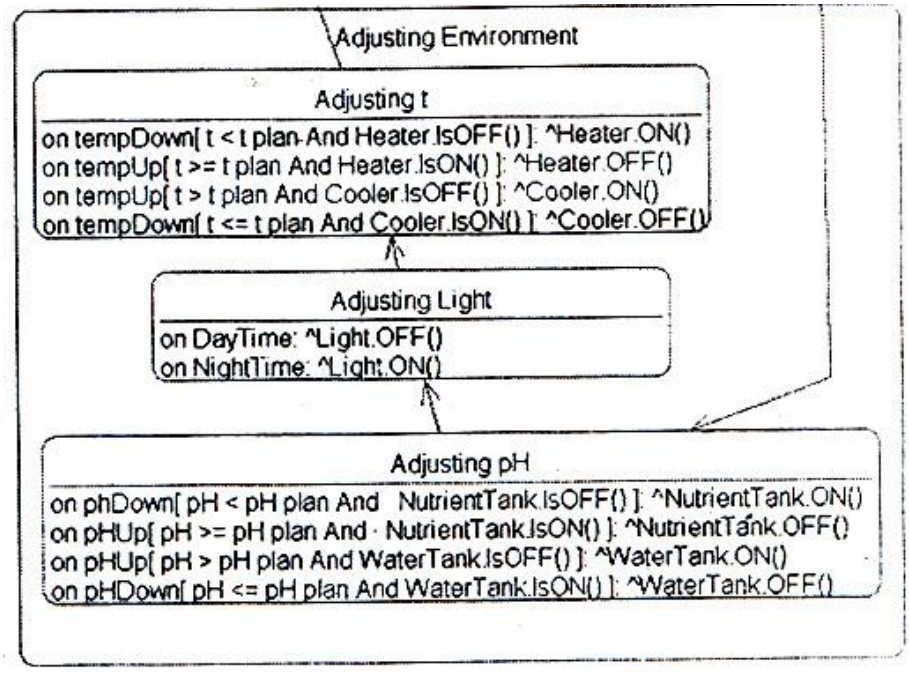

Рис.23. Настройка среды

Настройка температуры происходит только при ее изменении, что показывают функции tempDown и tempUp. Если произошло одно из этих событий, то температура сравнивается с той, которая должна быть по плану, и если она меньше, то включается нагреватель, а если больше — вентилятор.

Однако если температура достигла нормального уровня, то вентилятор и нагреватель отключаются. Аналогично происходит и с уровнем pH.

Для освещения лампочка просто включается при наступлении времени освещения и выключается при наступлении ночного времени суток.

Скрытие вложенных состояний

Теперь диаграмма стала достаточно трудна для обозрения. Представьте, что у вас таких состояний десяток. Общая картина теряется при большом количестве объемных состояний. Rational Rose позволяет скрыть ненужные в данный момент вложенные состояния. Для этого нужно выделить состояние, а затем выбрать Menu:View=>Hide Substates. При этом в правом нижнем углу состояния появится звездочка, а входящие и исходящие стрелки приобретут вид, показанный на рис. 24, в состоянии Adjusting Environment.

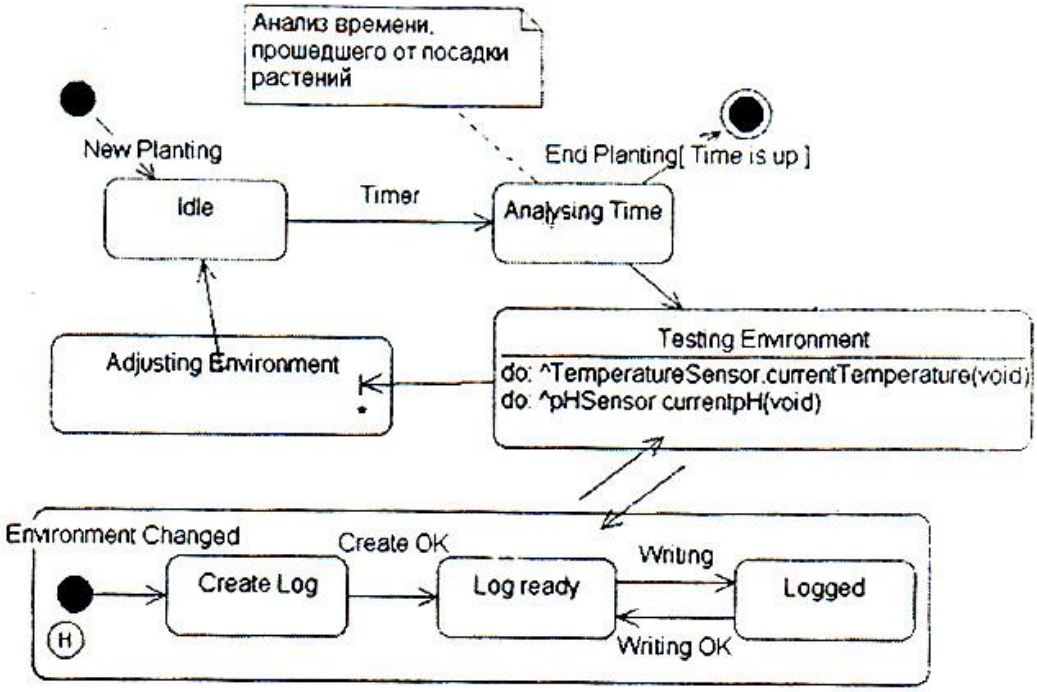

Рис.24.Окончательный вариант диаграммы состояний контроллера среды

State History (история состояний)

Единственное, что осталось не рассмотренным на этом типе диаграмм, это настройка State History.

Включение этой настройки позволяет показать, что в следующий раз, когда система попадает в указанное состояние, она должна не начинать с начала состояний, а сразу перейти на последнее состояние, из которого вышла, то есть при первом входе в некоторое состояние производятся единичные действия, которые при следующем входе проделывать уже не нужно. Например, при первом входе в режим протоколирования сообщений нужно создать файл протокола, который при последующих обращениях к этому режиму пересоздавать не нужно. На рис.24 видно, что установленная настройка State History отражается буквой H в кружке.

Протоколирование начинается при изменениях в условиях (Environment Changed). После этого создается Log файл (Create Log) и система переходит в состояние ожидания (Log ready). При необходимости записать изменения они записываются (Logged), и система снова переходит в состояние ожидания. При этом в следующий раз необходимо начинать с состояния Log ready, а не с создания протокола.