СБОРНИК **НАУЧНЫХ ТРУДОВ НГТУ. – 2010. – № 3(61). –** 61–70

УДК 62-50:519.216

ИСПОЛЬЗОВАНИЕ UML И ВРЕМЕННЫХ СЕТЕЙ ПЕТРИ ПРИ РАЗРАБОТКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ*

А.А. ВОЕВОДА, Д.О. РОМАННИКОВ

Рассматривается применение временных сетей Петри в методе разработки ПО с использованием диаграмм UML. Приведен следующий набор диаграмм: структурная диаграмма, диаграмма объектов, диаграмма вариантов использования и диаграмма последовательности. Для диаграмм выполнено преобразование в сети Петри и выполнен анализ сети. Рассмотрен пример нахождения логических ошибок в диаграммах и их исправление.

Ключевые слова: UML-диаграммы, сети Петри, временные сети Петри, разработка программного обеспечения.

ВВЕДЕНИЕ

Данная статья является первой частью рассматриваемого примера использования методики построения ПО с использованием UML-диаграмм и временных сетей Петри. Рассматриваемая методика была сформулирована [1] на основе методики [2], в которой использовались цветные сети Петри, а также разнообразных примеров [3–7] использования временных сетей Петри. Полученную методику можно применять для систем автоматики.

В качестве «опытной» системы выбрана система автоматического поддержания давления в трубе. Построение системы произведено согласно методике [1] – итерационно, каждый шаг снабжен комментариями и пояснениями.

1. ОПИСАНИЕ СИСТЕМЫ

Автоматизированная система поддержания давления представляет собой систему, включающую в себя следующий набор оборудования: насосы PU1, PU2; задвижки V1, V2; датчики давления P1, P2.

Расположение оборудования представлено на рис. 1. Насосы PU1, PU2 и задвижки V1, V2 управляются при помощи двух дискретных сигналов (включение/отключение). При включении оборудования подается сигнал о завершении действия, т.е. при закрытии задвижки ей подается сигнал об окончании закрытия, для остальных операций и оборудования сигналы о завершении действия подаются идентично.

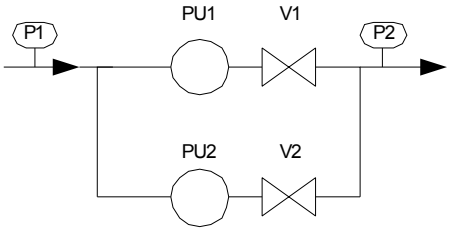

Рис. 1. Схема расположения оборудования

На выходе и входе системы присутствуют датчики давления. Управление системой осуществляется оператором, который может включать/отключать выбранные насосные агрегаты, квитировать аварийное состояние системы, просматривать состояние датчиков и оборудования системы. Вторым участником системы является администратор, который может менять настроечные значения системы, получать отчет о работе оборудования, добавлять операторов.

2. ПРИМЕНИЕ МЕТОДИКИ: UML-ДИАГРАММЫ

Согласно рассматриваемой методике разработки **первым** шагом является: *составление диаграммы прецедентов на основе требований к системе, составленных в форме технического задания или словесного описания*.

На рис. 2 представлена диаграмма прецедентов. На ней выделено два актера: оператор системы и администратор. Прецеденты ON_PU1, OFF_PU1 (ON_PU2, OFF_PU2) обозначают возможность выполнения оператором включения и отключения HA1 (HA2) соответственно. Данные прецеденты расширены прецедентами ON_V1, OFF_V1 (ON_V2, OFF_V2), которые обозначают открытие, закрытие задвижки V1 (V2) соответственно. Прецедент Acknowledgement дает возможность оператору выполнить квитирование аварийного состояния. Прецедент ViewRegistrationInfo обозначает возможность выполнения оператором просмотра информации о состоянии системы (оборудования, датчиков).

Второй актер, изображенный на диаграмме прецедентов, – администратор системы, который наследует все прецеденты оператора, а также может выполнять дополнительные. GetReport – прецедент, обозначающий возможность просмотра администратором отчета о работе системы. SetNewSettings – прецедент, обозначающий возможность изменения администратором настроеч-

ных параметров системы. AddOperator – прецедент, обозначающий возможность добавления администратором операторов в систему.

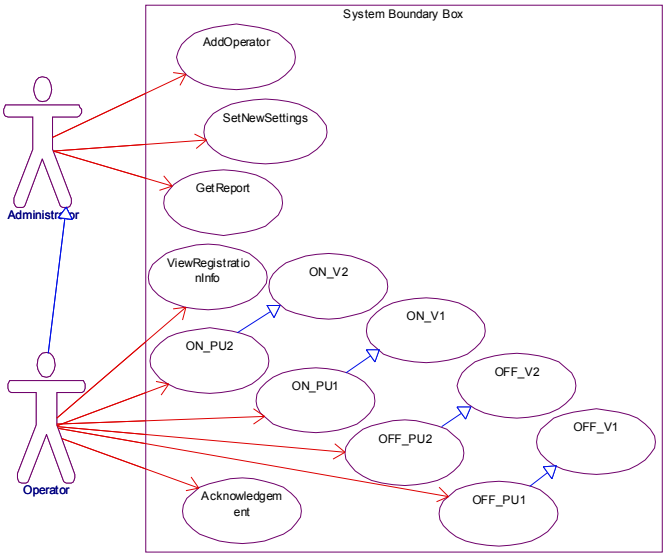

Рис. 2. Диаграмма прецедентов

В примере рассмотрена не вся функциональность, часть функций, таких как получение отчета или добавление оператора в систему, не рассмотрено по причине их реализации в части системы, отвечающей за визуализацию.

Вторым шагом рассматриваемой методики разработки является *выделение прецедентов, требующих более детального рассмотрения при помощи диаграмм последовательности.*

Рассмотрим прецедент ON_PU1 на диаграммах последовательности для успешного и ошибочного сценария. Остальные прецеденты являются либо более простыми с точки зрения реализации, либо идентичные рассмотренным.

На рис. 3 представлена диаграмма последовательности для прецедента успешного включения HA1. Из диаграммы видно, что включение представляет собой последовательность из следующих событий:

1 onPu1(). Команда от оператора на включение HA1;

2 vOn(1). Событие, генерируемое объектом SYSTEM и запускаемое открытие задвижки V1;

3 vOnAckSys(1). Событие, получаемое в результате подтверждения открытия задвижки за время, не превышающее установленное;

4 vPu(1). Событие, генерируемое объектом SYSTEM и запускаемое включение насоса PU1;

5 puOnAckSys(1). Событие, получаемое в результате подтверждения включения насоса за время, не превышающее установленное.

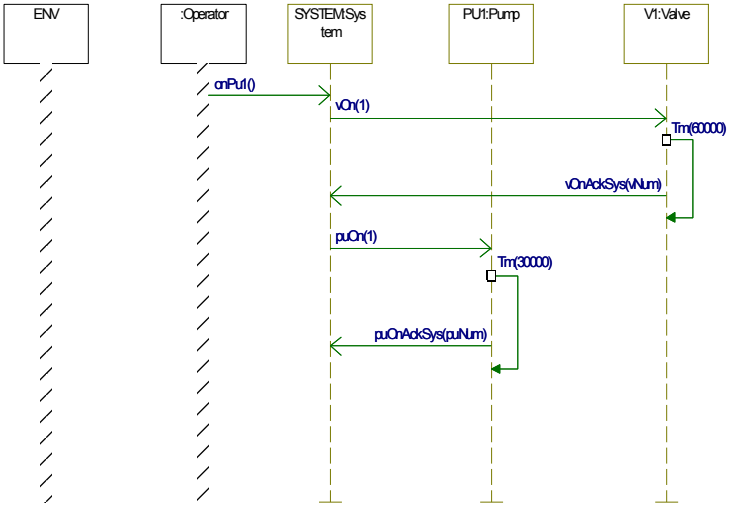

Рис. 3. Диаграмма последовательности для прецедента включения НА1 (успешный сценарий)

Рассмотренная диаграмма позволяет рассмотреть прецедент включения НА1 подробнее. Тем не менее на рассмотренной диаграмме представлен не весь набор возможных вариантов развития событий. Далее рассматривается диаграмма последовательности (рис. 4), где представлен вариант ошибочного включения НА1 и действия системы при аварии.

Ошибочное включение представляет собой последовательность из следующих событий:

6 onPu1(). Команда от оператора на включение НА1;

7 vOn(1). Событие, генерируемое объектом SYSTEM, и запускаемое открытие задвижки V1;

8 vOnAckSys(1). Событие, получаемое в результате подтверждения открытия задвижки за время, не превышающее установленное;

9 vPu(1). Событие, генерируемое объектом SYSTEM, и запускаемое включение насоса PU1;

10 puSetAlarm(1). Событие, получаемое объектом SYSTEM после завершения тайм-аута на включение насоса;

11 vOff(1). Событие, генерируемое объектом SYSTEM, и запускаемое закрытие задвижки V1;

12 vPu(1). Событие, генерируемое объектом SYSTEM, и запускаемое отключение насоса PU1;

13 ackAlarm(). Квитирование оператором аварии в системе.

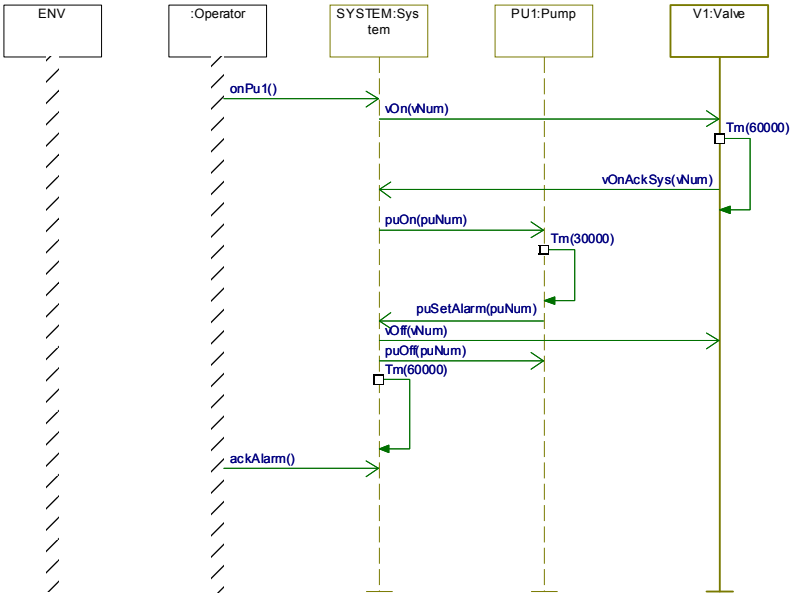

Рис. 4. Диаграмма последовательности для прецедента включения НА1 (ошибочный сценарий)

Остальные прецеденты включения и отключения НА являются идентичными рассмотренным.

Третьим шагом рассматриваемой методики разработки является: *составление диаграммы компонентов, которая позволяет сформировать иерархическую структуру системы, состоящую из файлов, папок, библиотек, исполняемых файлов и т.д.*

Данный шаг позволяет сформировать структуру проекта, выделить основные части и является желательным при программной реализации, однако его можно пропустить, так как вся его направленность – сформировать эстетическую сторону проекта, внести в него порядок, с функциональной точки зрения он не несет в себе никакой нагрузки.

Четвертым шагом является: *построение диаграммы объектов и классов.* Одной из ключевых особенностей методики разработки является определение списка сообщений (включая объект-источник, объект назначения, тело сообщения), используемых для взаимодействия между объектами системы. Определенные на данном этапе сообщения будут использоваться в дальнейшем для изменения состояний объектов в диаграмме состояний.

Также на данном этапе должны быть определены атрибуты классов.

На диаграммах, представленных на рис. 5 и 6, изображены диаграммы классов и объектов соответственно. На диаграмме классов представлены классы Pump, Valve и System.

Объявленные классы и объекты данных классов представляют собой каркас для дальнейшего построения диаграмм состояний для классов – **пятый** шаг методики.

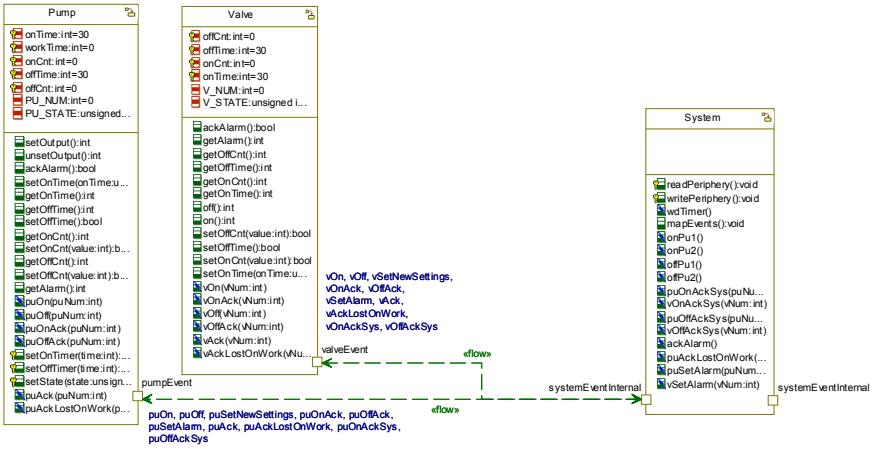

Рис. 5. Диаграмма классов для автоматизированной системы поддержания давления

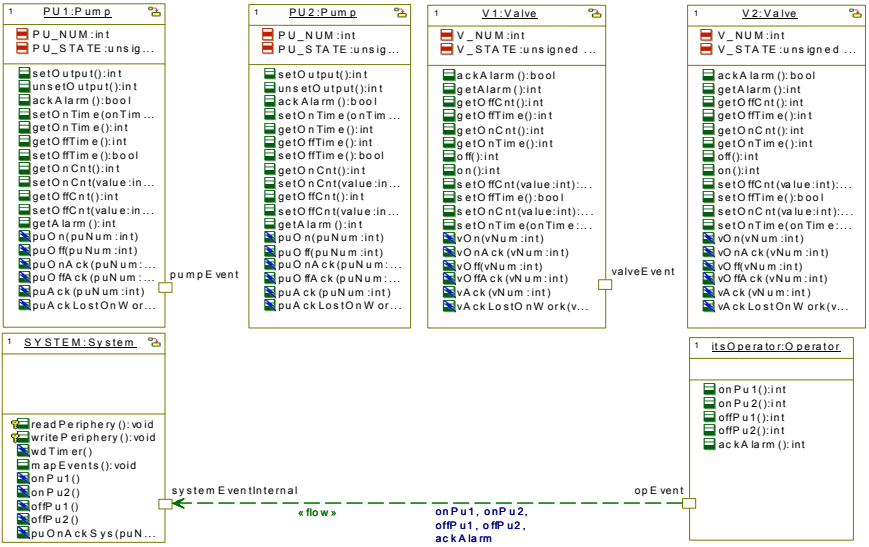

Рис. 6. Диаграмма объектов для автоматизированной системы поддержания давления

Представленная на рис. 7 диаграмма состояний определяет поведение всех объектов данного класса. Начальное состояние объекта – initialState. При появлении события vOn срабатывает переход системы в состояние waitingForOnAck – ожидание подтверждения включения. При срабатывании перехода vOnAck за время меньшее, чем getOnTime(), система переходит в состояние workState, иначе в состояние initialState по срабатыванию перехода tm(), при переходе генерируется событие vSetAlarm. Из состояния workState в состояние waitingForOffAck объект переходит либо при помощи перехода puOff, либо puAckLostOnWork. В случае второго варианта генерируется событие vSetAlarm. Аналогично – для перехода из состояния waitingForOffAck в состояние initialState.

Логика работа диаграммы состояний (рис. 8) класса Valve идентична логике работы вышерассмотренной диаграммы.

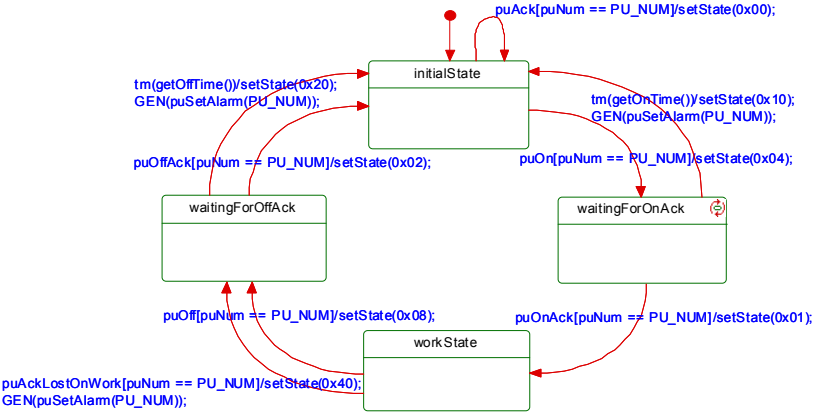

Рис. 7. Диаграмма состояний для класса Pump

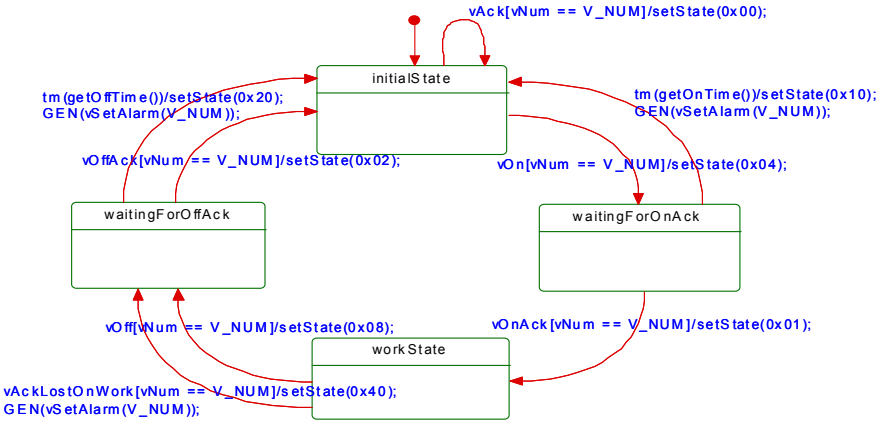

Рис. 8. Диаграмма состояний для класса Valve

Представленная на рис. 9 диаграмма состояний определяет последовательность включения/отключения оборудования, квитирование. Логика «чтения» диаграммы совпадает с логикой «чтения» диаграмм, изображенных на рис. 7 и 8.

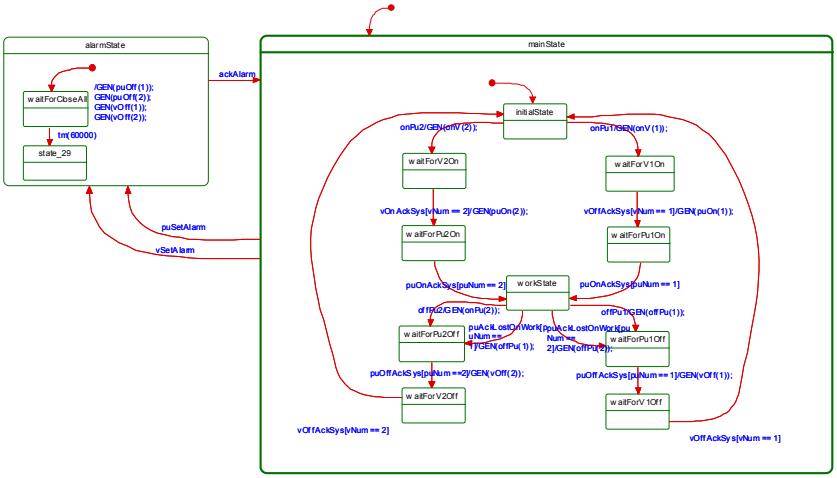

Рис. 9. Диаграмма состояний для класса SYSTEM

ЗАКЛЮЧЕНИЕ

В данной статье рассмотрен пример создания набора UML-диграмм для автоматизированной системы согласно методике построения ПО с использованием UML и сетей Петри. Составлена диаграмма прецедентов, диаграмма последовательности, диаграммы объектов и классов, а также структурные диаграммы для созданных объектов.

[1] *Романников Д.О.* Использование UML-диаграмм и сетей Петри в разработке программного обеспечения систем распределенной обработки информации, критических ко времени исполнения: магистерская диссертация, 2010.

[2] *Коротиков С.В.* Применение сетей Петри в разработке программного обеспечения центров дистанционного контроля и управления: дис. … канд.техн. наук. – Новосибирск: Изд-во НГТУ, 2007.

[3] *Воевода А.А. Романников Д.О.* Применение UML-диаграмм и сетей Петри при разработке встраиваемого программного обеспечения // Научный вестник НГТУ. – 2009. – № 3(36)

[4] *Воевода А.А. Романников Д.О.* Моделирование сетей Петри в CPN Tools // Сб. науч. тр. НГТУ. – № 3(53). – 2008.

[5] *Воевода А.А. Романников Д.О.* О моделировании систем реального времени с использованием UML и сетей Петри // Сб. науч. тр. НГТУ. – № 1(55). – 2009.

[6] *Воевода А.А. Романников Д.О.* О проектировании программного обеспечения для МК с использованием UML // Сб. науч. тр. НГТУ. – № 3(57). – 2009.

[7] *Воевода А.А. Романников Д.О.* О компактном представлении языков раскрашенных сетей Петри // Сб. науч. тр. НГТУ. – № 3(53). – 2008.

Воевода Александр Александрович **–** профессор кафедры автоматики Новосибирского государственного технического университета, д.т.н., профессор. E-mail: ucit@ucit.ru

Романников Дмитрий Олегович – аспирант кафедры автоматики Новосибирского государственного технического университета. E-mail: rom2006@gmail.com

Voevoda A.A, Romannikov D.O.
UML and Timed Petri nets using in the approach of software design

Article considers an approach of using UML diagrams and Timed Petri nets for designing software. Set of UML diagram (object diagram, statechart diagrams, and structure diagrams) is recommended to use for designing software systems. Rule of modeling of changing information between objects suggested. Considered an example of logic errors in the diagrams.

Key words: diagram, linear system, stability.