УДК 004.4'22

РАЗРАБОТКА ПРОГРАММНОЙ СИСТЕМЫ АВТОМАТИЗАЦИИ АППАРАТНОГО КОМПЛЕКСА ЛАБОРАТОРНЫХ СТЕНДОВ

О.С. Большаков, А.В. Петров
Рыбинская государственная авиационная технологическая академия имени П.А.Соловьева, г. Рыбинск

При создании новых установок, предназначенных для выполнения некоторого технологического процесса разработчику необходимо иметь удобную систему разработки программ управления данной установкой. Нередко такие установки создают с учебными целями, например, для проведения лабораторных работ для студентов той или иной специальности (направления). Установки, которые сейчас представлены на рынке, требуют немалых денежных средств, что ограничивает их использование в учебном процессе. Существующие программные системы автоматизации рассчитаны на использование особых автономных устройств автоматизации – промышленных логических контроллеров (ПЛК) и их программирование, что усложняет процесс создания управляющих программ и увеличивает затраты на дополнительную закупку и поддержку оборудования.

На базе имеющегося оборудования РГАТА им. П.А. Соловьева СКБ ИТ было создано три лабораторных стенда для трех кафедр: кафедры резания материалов, станков, инструментов (РМСИ); кафедры материаловедения, литья, сварки (МЛС); кафедры технологии авиационных двигателей и общего машиностроения (ТАДиОМ);

Стенды предназначаются для обучения студентов, проходящих курс автоматизации в промышленности, а также для демонстрации некоторых технологических процессов (изготовление, литье деталей). Оборудование стендов состоит из роботов-манипуляторов МП-11 и МП-9С, приобретенных в 90-ые годы на “Автовазе”, а также другого пневматического и позиционного оборудования. Стандартная схема управления роботами МП-11, МП-9с предполагала использование блоков управления МПЦУ. Необходимо было заменить устаревшие блоки управления новыми с возможностью управления с персонального компьютера [1]. Новые блоки управления должны выполнять функции силовых модулей, т.е. быть посредниками между программной частью системы, находящейся на компьютере, и оборудованием установок.

Системные решения

В связи с тем, что технологические процессы, моделируемые каждым стендом требовали одновременной работы множества устройств возникла необходимость разработки системы параллельного управления устройствами с возможностью их синхронизации. Кроме того, для обучения студентов была необходима разработка интуитивно понятых представлений программ, языка

программирования, модуля моделирования работы стенда, отладки управляющих программ с визуализацией стенда трехмерной модели.

Общая схема работы системы представлена на рисунке 1.

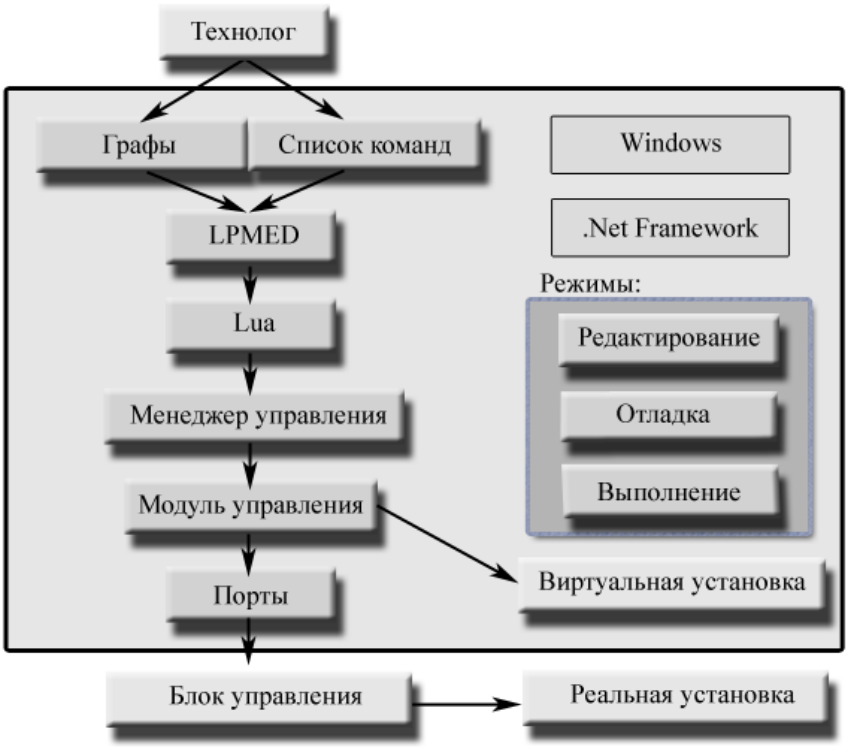

Рисунок 1 - Общая схема работы системы

В первую очередь, возникла необходимость выбора языка, с помощью которого наиболее удобно осуществлять координацию работы оборудования стенда. При этом необходимо было учитывать, что важным свойством такого языка должна являться возможность параллельного исполнения действий (разделения программы по логическим потокам управления), а также синхронизация по выполнению действий. В соответствии с данными требованиями был разработан язык LPMED. Транслятор, встроенный в системе IronHand был создан при помощи инструмента “ANTLR 3.0” [2]. Язык LPMED (Language of Parallel Management of External Devices) позволяет естественным образом описать последовательность выполнения асинхронных вызовов действий и ожидания их завершения. Данная концепция оказалась удобной для запуска и остановки параллельно выполняющихся процессов, что находит свое отражение при реализации алгоритмов автоматизации работы роботов-манипуляторов, различного рода конвейеров и тому подобного оборудования.

Реализация LPMED в системе “IronHand” предполагает использование описания оборудования, структурированного по иерархической модели. Родительскими узлами в такой иерархической модели являются группы узлов оборудования (удобно их объединять по принадлежности к одному устройству). Каждый из узлов может содержать список вложенных узлов и так далее. Обращение к вложенным узлам некоторого узла из программы осуществляется в стиле объектно-ориентированного программирования. В

описании оборудования для каждого узла содержится список его возможных состояний и методов, которые он может исполнять. В данной реализации для метода узла прописывается виртуальный пин и значение, которое необходимо на данный пин послать для асинхронного запуска процесса (аналогично для прекращения выполнения метода).

Пользователю предлагается разрабатывать программу на графических языках (Графы и Списки команд), каждый из которых транслируется в LPMED. Далее с языка LPMED при данном описании оборудования программа транслируется на язык Lua, на котором и происходит исполнение пользовательской программы виртуальной машиной Lua на уровне посыла управляющих сигналов на виртуальные пины и считывания с них информационных сигналов, приходящих с датчиков. Именно на уровне Lua и производится реализация логического параллелизма посредством использования со-процедур (coroutine).

Внутренняя организация системы скрыта от пользователя - работая в Списках команд и Графах, технолог описывает параллельные процессы естественным образом (разветвление потоков управления и их слияние). Данные графические языки, транслирующиеся в базовый текстовый язык LPMED, в наибольшей степени подходят для автоматизации управления оборудованием заказчика в рамках данной работы и максимально интуитивны для восприятия человеком, не имеющим навыков программирования, что идеально укладывается в рамки предназначения данного аппаратно-программного комплекса.

Альтернативные подходы

При рассмотрении языков для автоматизации стоит в первую очередь рассмотреть наиболее распространенные языки, применяющиеся сейчас в автоматизации промышленности. Речь идет о пяти языках стандарта МЭК-61131-3 (IL, ST, LD, FBD и SFC) и их прямых аналогах (CFC, упрощенный SFC) [3]. Языки данного стандарта имеют ряд недостатков [4,5]. Очевидно, что стандарта из пяти языков принятых более 15-ти лет назад не может быть достаточно для эффективного применения в разных предметных областях автоматизации. Так выходит и в данной ситуации: намного эффективнее оказывается язык, изначально заточенный под данную предметную область, каким и является LPMED. Из приведенного выше стандарта подходящим под задачи пользователя языком мог бы стать SFC, имеющий необходимые конструкции для распараллеливания логических потоков управления. Похожим на SFC является графический язык Графы, основанный на параллельных асинхронных блок-схемах (ПАБС) [6], встроенный в IronHand, однако имеющий некоторые существенные отличия.

Существует альтернативный подход к построению языка параллельного управления – текстовый язык Рефлекс (Reflex) [7]. При таком подходе предполагается предварительная декомпозиция пользовательской программы на совокупность процессов, которые могут исполняться параллельно, менять

свои состояния в результате изменения состояний других процессов и/или изменения непосредственно входных данных, поступающих от оборудования. Другими словами, Рефлекс предполагает написание программы как совокупности реакций процессов, находящихся в определенных состояниях, на происходящие изменения. Такой подход является в некотором роде модификацией автоматного программирования, предполагающего использование графа состояний, или Switch-технологии [8], при том отличии, что Рефлекс скрывает от пользователя реализацию логического параллелизма. Сравнительные примеры программ на языках LPMED и Рефлекс для использующегося оборудования приведены на рисунках 2 – 3, из которых видно, что программа на LPMED является все же более простой и наглядной, поскольку описывает управляющую программу примерно так, как ее в текстовом виде записывал бы заказчик на неформальном языке (более удобные графические представления, как уже было сказано, в системе предоставлены на языках более высокого уровня: “Списки команд” и “Графы”).

```
MODULE Program
{
 mp9.Arm.Forward.go(); // выдвинуть руку
 while (mp9.Arm.State ~ = mp9.Arm.Forward) // дождаться выдвижения
 {}
mp11.UpperArm.Forward.go();// выдвинуть руку
while (mp11.UpperArm.State ~ = mp11.Upper Arm.Forward) // дождаться выдвижения
 {}
}
```

Рисунок 2 – Пример программы на языке LPMED

```
ПРОЦ Программа {
СОСТ Сост1{
      СТАРТ ПРОЦ mp9_Arm_Forward_go;
      ЕСЛИ (!(ПРОЦ mp9_Arm_Forward_go В СОСТ АКТИВНОЕ))
            В СЛЕДУЮЩЕЕ;
}
СОСТ Сост2{
      СТАРТ ПРОЦ mp11_UpperArm_Forward_go;
      ЕСЛИ (!(ПРОЦ mp11_UpperArm_Forward_go В СОСТ АКТИВНОЕ))
            В СЛЕДУЮЩЕЕ;
}
ПРОЦ mp9_Arm_Forward_go {/* Описание процесса выдвижения руки робота МП-9с*/
СОСТ Начало {
      <Подать сигнал движения mp9.Arm.Forward>;
      В СЛЕДУЮЩЕЕ;
}
СОСТ Движение {
      ЕСЛИ (<доехали в mp9.Arm.Forward>) СТОП;
 }
}
ПРОЦ mp11_UpperArm_Forward_go {/* Описание процесса выдвижения руки робота МП-11*/
СОСТ Начало {
      <Подать сигнал движения mp11.UpperArm.Forward>;
      В СЛЕДУЮЩЕЕ;
}
СОСТ Движение {
      ЕСЛИ (<доехали в mp11.Upper Arm.Forward>) СТОП;
 }
}
```

Рисунок 3 – Возможная реализация программы на языке Рефлекс

Однако неверно было бы недооценить и подход, который реализуется с помощью Рефлекса. Основные отличия подходов к построению программы на LPMED и Рефлексе. Во-первых, программа на Рефлексе выглядит более структурированной, более декларативна в отличие от LPMED (за счет циклического выполнения программы в рабочем цикле так, что у каждого процесса выполняется код только одного состояния), лучше поддается анализу формальными средствами (например, средствами верификации). Во-вторых, в отличие от LPMED, у Рефлекса нет привязки к оборудованию, пользователь работает со скалярными переменными, а в LPMED пользователь работает с оборудованием, представленным в иерархической структуре, причем методы инкапсулированы в узлы. Состояния узлов на LPMED могут изменять только сигналы от датчиков. Состояния процессов на Рефлексе программируются и изменяются самим пользователем. В-третьих, программа на Рефлексе исполняется в рабочем цикле, что с определенной вероятностью гарантирует время реакции системы на событие, что может оказаться важным для реализации реактивных систем.

Результат

В результате программная система автоматизации аппаратного комплекса лабораторных стендов “IronHand” включает в себя следующие элементы: два графических языка программирования (Списки команд и Графы), транслирующиеся в LPMED; интегрированную среду разработки, исполнения и отладки пользовательских программ; встроенный модуль трехмерной визуализации (в качестве библиотеки визуализации используется Axiom [9]).

В ближайшее время предполагается встраивание в систему модуля верификации, который позволит проверять соответствие пользовательской программы предварительно написанной для нее спецификации (правил, условий, ограничений) до непосредственного запуска самой программы.

Список использованных источников

1 Петров А.В., Кудрявцев И.А., Виноградов И.С. Прототип цикловой системы управления роботом МП-11 Сборник статей Всероссийской конференции «Мавлютовские чтения», УГАТУ, Уфа, 2008

2 http://antlr.org

3 http://www.iec.ch/

4 Wagner F. Going beyond the limitations of IEC 61131-3 // StateWORKS, 2005.

5 Татарчевский В. А. Проблемы применения языков стандарта IEC 61131-3 и возможные пути решения // «Информационно-математические технологии в экономике, технике и образовании», Екатеринбург, УГТУ–УПИ, 2007.

6 Viktor Il’ich Varshavskiĭ, Mikhail Aleksandrovich Kishinevskiĭ, Alexandre V. Yakovlev. Self-timed control of concurrent processes: the design of aperiodic logical circuits in computers and discrete systems Mathematics and its applications // Springer Science & Business, 1990

7 Зюбин В.Е. Язык «рефлекс» – диалект си для Программируемых логических контроллеров // "Средства и системы автоматизации " CSAF’06, Томск, 1-3 ноября 2005.

8 А.А. Шалыто Автоматное программирвание // Научно-технический вестник Санкт-Петербургского государственного Университета информационных технологий, механики и оптики, Выпуск 53, Санкт-Петербург, 2008

9 http://axiomengine.sf.net