УДК 004.031.6

СИСТЕМА ОБНОВЛЕНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ КОМПЛЕКСА ТЕХНИЧЕСКИХ СРЕДСТВ ЛУЧ–2

А.О. Ключев, Е.В. Петров

Статья посвящена проблеме обновления программного обеспечения в распределенных управляющих системах (РУС). Проводится краткий обзор существующих средств программирования РУС. Предлагается способ программирования, использующий понятие целей, объединяющих сходные по своим свойствам ресурсы РУС, загрузочные модули, содержащие программируемые данные и управляющую программу, и простую виртуальную машину LVM, интерпретирующую загрузочные модули.
Ключевые слова: обновление программного обеспечения, виртуальная машина, встраиваемая система, интерпретатор, загрузочный модуль.

Введение

Задача обновления программного обеспечения возникает в любой системе, имеющей программную составляющую, независимо от того, является ли эта система распределенной, управляющей или встроенной, так как даже после сдачи готовой системы в эксплуатацию возникает необходимость замены программного обеспечения элементов системы. Это может происходить по нескольким причинам: исправление обнаруженных в программе ошибок, добавление новых функций в систему в связи с ее развитием и т.д. Если элементов системы, нуждающихся в обновлении программного обеспечения (ПО), немного (несколько единиц), то замену ПО можно провести стандартными (штатными) средствами. Данный способ требует наличия квалифицированных специалистов, способных выполнить действия по перепрограммированию контроллера узла системы и дополнительного оборудования – компьютера, источника питания и т.д., необходимого для этой операции. Если же система состоит из множества (десятков, сотен) элементов, появляется ряд дополнительных проблем: узлы системы могут находиться в различных районах города или даже за его пределами, а доступ к узлам системы может быть затруднен.

На сегодняшний день, с появлением новой элементной базы, все большее развитие получают системы автоматизации и телемеханики на основе проводных и беспроводных каналов связи, таких как Ethernet, CAN, Zigbee, Bluetooth, GSM/GPRS с использованием протоколов TCP, ModBus и других. Такие системы применяются на стационарных и мобильных объектах, используются для сбора телеметрических данных и выполнения функций автоматического управления различными объектами. При этом вопрос замены программного обеспечения в подобных системах в целом не решен. Сегодня для описанного класса систем отсутствуют готовые средства обновления программного обеспечения, и, следовательно, существует потребность в их разработке. Таким образом, задача создания систем обновления программного обеспечения для распределенных управляющих систем на базе смешанных (проводных и беспроводных) каналов связи является актуальной и востребованной. Решению этой задачи, на примере комплекса технических средств (КТС) Луч–2, и посвящена данная статья.

Комплекс технических средств Луч–2

КТС Луч–2 представляет собой распределенную управляющую систему на базе сетей Интернет и GPRS. Данная система предназначена для управления наружным освещением населенных пунктов, а также для выполнения некоторых функций АСКУЭ. Система включает в себя сервер сбора данных, автоматизированное рабочее место диспетчера и периферийные контроллеры, осуществляющие непосредственное управление освещением.

Основными причинами сложности процесса обновления программного обеспечения большинства распределенных управляющих систем, в том числе КТС Луч–2, являются:

- многообразие и постоянное изменение программируемой элементной базы;
- несовместимость версий ПО элементов системы;
- большое количество элементов системы, требующих обновления ПО;
- повышенные требования к надежности и безопасности процедуры обновления [1].

Главными из них являются многообразие программируемой элементной базы и конфликты версий программного обеспечения. В мире существует большое количество внутрифирменных стандартов загрузчиков, семейство JTAG-стандартов. Большая работа проводится в сегменте массовых, универсальных, офисных вычислительных систем. Но в силу специфики используемой элементной базы они не могут быть применены в полном объеме в КТС Луч–2.

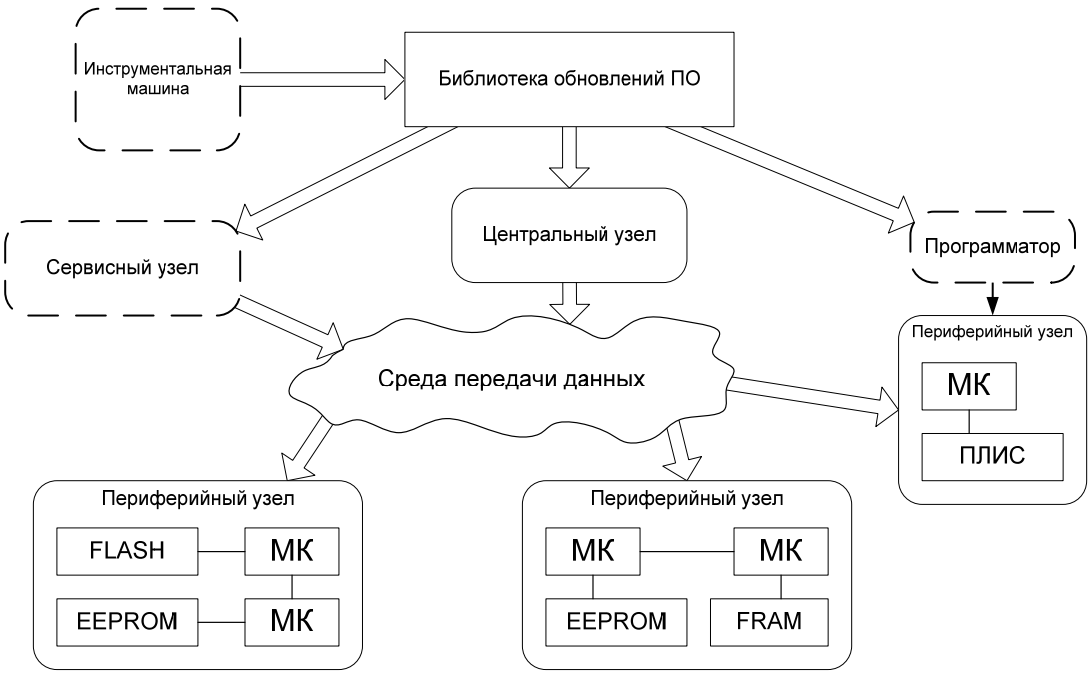

Рис. 1. Схема распространения ПО в КТС Луч–2

На рис. 1 представлена схема процесса обновления программного обеспечения КТС Луч–2. Система обновления включает инструментальную машину, центральный и сервисный узлы, периферийные узлы и среду передачи данных. В качестве среды передачи данных могут выступать как проводные, так и беспроводные каналы связи. Инструментальная машина выполняет функции хранилища программ и данных. Центральный или сервисный узлы осуществляют управление процессом доставки обновлений. Подлежащие обновлению ресурсы находятся в периферийных узлах. Это внутренняя память микроконтроллеров, внешние запоминающие устройства, периферийные контроллеры.

В принципе программирование ресурсов контроллеров может осуществляться локально, с помощью подключаемого внешнего программатора, как показано на рисунке. Задача системы обновления программного обеспечения состоит в доставке данных от инструментальной машины до ресурсов периферийных узлов.

Рассмотрим существующие на сегодняшний день способы замены программного обеспечения, применимые в КТС Луч–2. Средства программирования можно разделить на используемые в режиме разработки и на этапе эксплуатации. В режиме разработки

системы чаще всего используются программатор микросхем и ISP программирование – программирование логических устройств, установленных в законченную систему. Для этого могут использоваться самые различные интерфейсы, как стандартные, так и специализированные. В рабочем режиме используется IAP – процедура программирования логических устройств на этапе эксплуатации. Программирование ведется процессором, исполняющим резидентную программу. При этом данные для программирования могут получаться как «на лету», так и из специального хранилища в контроллере.

Концепция цели

Впервые данная концепция была использована при создании инструментальной системы М3Р для программирования многопроцессорной системы М3М, используемой для управления железнодорожными стрелками, в рамках системы КТС ТРАКТ в 1997 г.

Как было сказано выше, одной из главных трудностей замены программного обеспечения элементов распределенных управляющих систем является большое разнообразие используемых программируемых ресурсов. Различные ресурсы имеют различные типы памяти, интерфейсы подключения, быстродействие, адресные пространства и т.д. Для упрощения и унификации работы со всеми ресурсами системы предлагается способ программирования, основанный на понятии цели. Цель – это виртуальное устройство, объединяющее сходные по своим свойствам реальные устройства. Назначение цели – унификация интерфейсов разнообразных устройств. В качестве примеров целей можно назвать цель EEPROM, обеспечивающую доступ к внутреннему EEPROM микроконтроллера и внешней микросхеме EEPROM, цель RAM, обеспечивающую доступ к оперативной памяти микроконтроллера, и т.д. Понятие цели позволяет организовать работу с различными программируемыми ресурсами контроллеров одинаково.

Достаточно близкими (но гораздо более примитивными) аналогиями цели можно считать BSD сокеты (1983 г.) и концепцию виртуальной файловой системы, принятой на вооружение фирмой Sun Microsystems еще в 1985 г. [2]. С сокетами цель объединяет понятие порта – связующего звена между потоком данных и процессом. Виртуальная файловая система предоставляет унифицированный способ доступа к неким сущностям, имеющим адресное пространство с возможностью проведения операций позиционирования, чтения и записи. Если посмотреть, на каких моделях вычислений базируется каждая из концепций, нетрудно понять, что сокеты ограничивают наши возможности законами, заложенными в модель вычислений, называемую «сети процессов», а виртуальная файловая система ограничивает нас законами машины фон-Неймана, с ее пассивным адресным пространством.

Наиболее близко к концепции целей подошли в системе моделирования Ptolemy. В этом проекте, разрабатываемом в институте Беркли [3], предлагается актор-ориентированная концепция, позволяющая представить широкий спектр моделей вычислений единообразным способом. Семантика моделей определяется текущей моделью вычислений (в системе моделирования ее задает так называемый директор). В системе Ptolemy у каждой цели может быть своя модель вычислений.

В системе М3Р взаимодействие с целью осуществляется прикладными пакетами, являющимися аналогами токенов системы Ptolemy. Для каждой цели может быть определен уникальный набор команд, позволяющий реализовывать большое количество вычислительных моделей.

Порядок обновления ПО

Программное обеспечение целевого контроллера включает как стандартные загрузчики, так и загрузчики собственной разработки. Необходимость разработки собст-

венных загрузчиков возникает в связи с тем, что использование только фирменных средств приводит к сильному удорожанию аппаратуры контроллеров, а также делает процесс программирования медленным и неудобным. В результате при не очень высокой стоимости реализации собственного загрузчика такое решение оказывается эффективным и дает высокий выигрыш в стоимости и удобстве системы программирования.

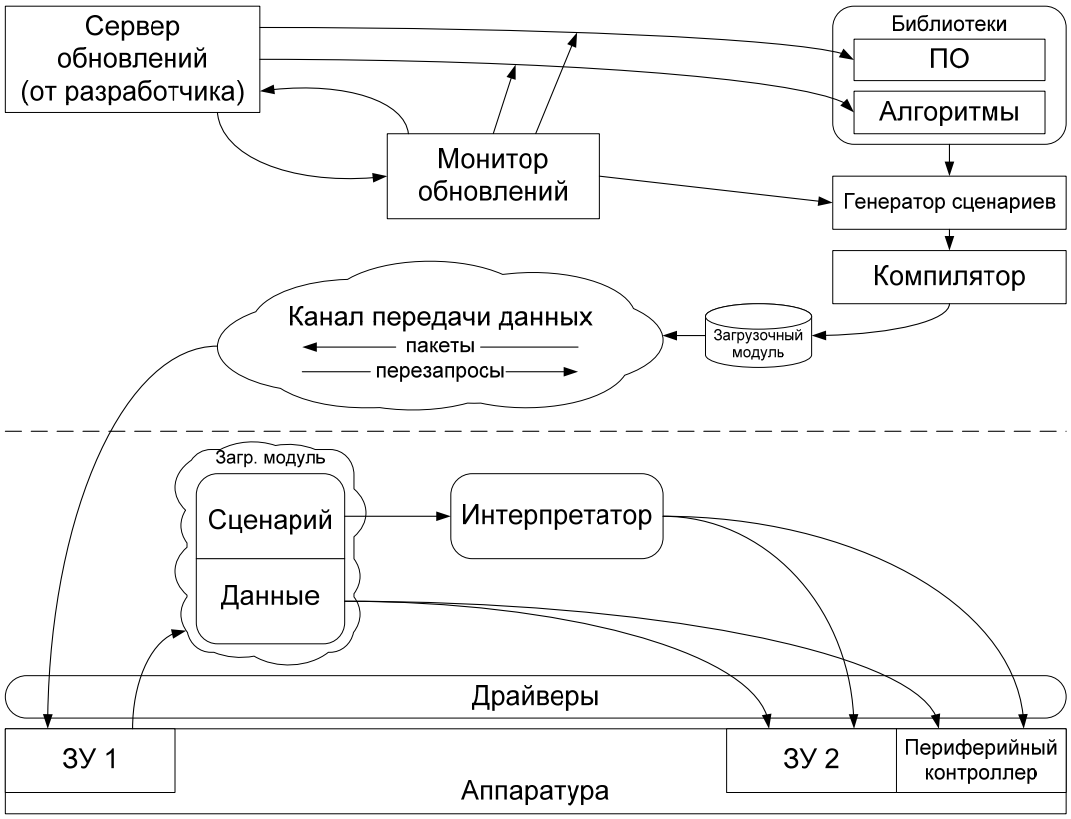

Рис. 2. Схема обновления ПО в режиме эксплуатации

В режиме разработки от системы требуется следующее: программировать целевое программное обеспечение, записывать тестовые данные, конфигурировать элементы системы, а также осуществлять вспомогательные действия для отладочных нужд. Для реализации этих функций удобнее всего воспользоваться описанными ранее целями.

Цели предназначены для решения проблемы разнородности программируемых ресурсов, и их использования достаточно для построения системы обновления программного обеспечения в режиме разработки. Но на этапе эксплуатации к системе обновления предъявляются другие требования – своевременное обнаружение новых версий программного обеспечения, его доставка до целевой системы и корректная установка. Для решения перечисленных задач предлагается решение, использующее монитор обновлений, сценарии программирования, загрузочные модули и интерпретатор сценариев. Схема процесса обновления программного обеспечения КТС Луч–2 в режиме эксплуатации представлена на рис. 2.

Функции монитора заключаются в обнаружении и доставке новых версий программного обеспечения до инструментальной машины. Чуть более подробно рассмотрим понятия загрузочного модуля и сценария программирования. Загрузочный модуль – совокупность данных и инструкций по их размещению. Сценарий программирования – последовательность действий, обеспечивающая программирование ресурсов контроллеров со сложной организацией.

Рассмотрим пример языка сценария, примененный в КТС Луч–2. Для реализации компилятора был выбран язык Форт, позволяющий достаточно легко создавать несложные языковые конструкции. Из них составляется сценарий программирования. Таким образом, на этапе компиляции загрузочного модуля работает Форт-система, а на этапе исполнения – виртуальная машина LVM.

Характеристики языка сценария КТС Луч–2:

- простая базовая система команд (стирание, копирование, исполнение);
- расширение системы команд;
- линейное исполнение;
- компиляция в Форт-системе;
- исполнение в виртуальной машине LVM.

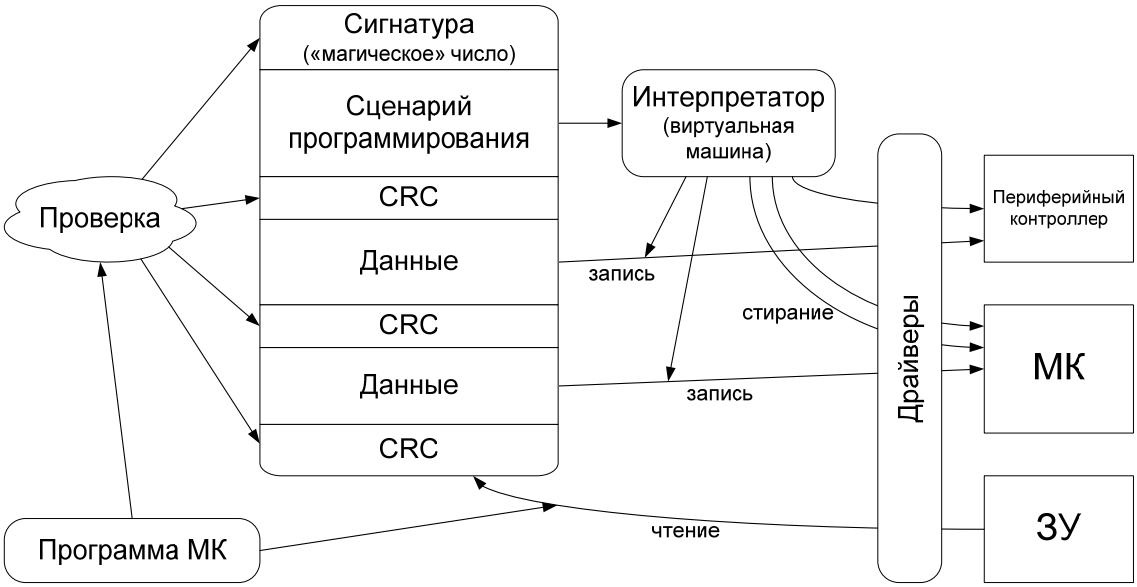

Рис. 3. Схема интерпретации загрузочного модуля в контроллере

На рис. 3 показана схема работы интерпретатора сценариев программирования. Интерпретатор представляет собой виртуальную машину, названную LVM, которая запускается загрузчиком контроллера в случае обнаружения признака необходимости обновления программного обеспечения. Виртуальная машина последовательно исполняет команды сценария программирования, определяемые типом цели. Например, в случае работы с FLASH это могут быть команды: стереть, скопировать, передать управление. Аргументами команд могут быть адреса, а также данные, хранящиеся в блоке данных загрузочного модуля. Данная реализация требует наличия в целевом контроллере промежуточного буфера для хранения загрузочного модуля.

Необходимость разворачивания цельного загрузочного модуля непосредственно на месте вызвана проблемами передачи данных по плохим каналам связи через достаточно большое количество узлов сети. Перечисленные проблемы естественным образом подталкивают нас к использованию принципа слабой связи. Близкими аналогами такой системы являются дистрибутивы Windows (setup.exe) или, к примеру, rpm в Linux. В отличие от LVM, разворачивание таких дистрибутивов происходит в рамках виртуальной файловой системы (Linux, Windows) и системного реестра (Windows). В LVM же взаимодействие происходит с различными целями, работающими в рамках различных моделей вычислений.

Одним из важнейших является комплекс вопросов информационной надежности и безопасности обновления программного обеспечения. Для обеспечения безопасности

процедуры замены программного обеспечения в КТС Луч–2 предусмотрен целый ряд механизмов, предотвращающих выход контроллеров из строя. Среди них – контроль целостности данных на этапе доставки загрузочного модуля в контроллер, контроль целостности загрузочного модуля, хранящегося в памяти контроллера, непосредственно перед началом процесса перепрограммирования, а также контроль совместимости версий загрузочного модуля и интерпретатора (виртуальной машины). Кроме того, весь обмен данными между сервером обновлений и контроллерами производится в зашифрованном виде, что делает невозможным подмену программируемого ПО злоумышленниками.

Заключение

Предложенная система обновления программного обеспечения, реализованная в КТС Луч–2, показала свою высокую надежность и эффективность, уменьшив в сотни раз время (а значит, и стоимость), затрачиваемое на замену программного обеспечения многочисленных контроллеров, рассредоточенных в пределах города. Рассмотренная система также значительно облегчила процесс отладки и тестирования КТС Луч–2 в целом благодаря возможности доступа к любым программируемым ресурсам контроллеров.

Данная система обновления может быть успешно использована как в проводных, так и в беспроводных системах, независимо от географической протяженности. Использование в системе обновления сценариев программирования, загрузочных модулей и интерпретатора сценариев делает ее легко переносимой в другие системы, подобные КТС Луч–2.

Литература

1. Петров Е.В. Обновление программного обеспечения в распределенных управляющих системах // Научно-технический вестник СПБГУ ИТМО. – 2007. – Выпуск 45. Информационные технологии. – С. 65–70.
2. Kleiman S.R. Vnodes: An Architecture for Multiple File System Types in Sun UNIX // Proc. USENIX, Summer, 1986.
3. Ptolemy Project [Электронный ресурс]. – Режим доступа: http://ptolemy.eecs.berkeley.edu/, свободный.

Ключев Аркадий Олегович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, кандидат технических наук, доцент, kluchev@d1.ifmo.ru

Петров Евгений Владимирович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, аспирант, sshhiiccoo@yandex.ru