Р10-2008-185

Н. Г. Мазный*

ПОДСИСТЕМА ПЕРЕДАЧИ ДАННЫХ МЕЖДУ ПРОЦЕССАМИ В РАСПРЕДЕЛЕННОЙ СИСТЕМЕ АВТОМАТИЗАЦИИ

*НПЦ «Аспект», г. Дубна

Мазный Н. Г. Р10-2008-185

Подсистема передачи данных между процессами
в распределенной системе автоматизации

Сложность систем автоматизации эксперимента (САЭ) растет одновременно с развитием средств вычислительной техники и систем программирования. В настоящее время обычными являются требования обеспечить работу САЭ на распределенной конфигурации, возможность использования горячего резерва оборудования, динамического конфигурирования и другие.

В любой распределенной системе существенную роль играют средства передачи данных между ее компонентами, и унификация этих средств ведет к экономии времени разработки САЭ и повышению ее надежности.

В статье описывается разработанная унифицированная подсистема передачи данных, предназначенная для использования в распределенных САЭ со смешанным составом аппаратных средств коммуникации.

Отличительной особенностью данной подсистемы передачи данных является использование унифицированного кода.

Работа выполнена в Лаборатории нейтронной физики им. И. М. Франка ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна, 2008

Maznyy N. G. P10-2008-185

Data Transfer Subsystem Between Processes
in the Distributed System of Automation

Complexity of systems for automation of experiment (SAE) is growing alongside with the development of means of computer technology and programming systems. At the present time requirements to ensure SAE functioning in distributed environment, availability of hot standby, dynamic configuring etc. are considered usual.

In any distributed system data transmission facilities play an important role, and unification of these facilities leads to economy of SAE engineering time and reliability growth.

The present article describes elaborated unified data transmission subsystem, intended for use in distributed SAE with mixed transmission hardware.

Characteristic feature of this data transmission subsystem consists in the usage of the unified code.

The investigation has been performed at the Frank Laboratory of Neutron Physics, JINR.

Communication of the Joint Institute for Nuclear Research. Dubna, 2008

1. ОБОБЩЕННАЯ СХЕМА РАСПРЕДЕЛЕННОЙ САУ

Можно выделить две основные задачи, в которых используется передача данных. Это — задачи управления и передачи данных между компонентами системы.

На рис. 1 представлена обобщенная схема распределенной САУ. В общем случае возможно наличие двух уровней:

- верхний уровень, на котором работают ЭВМ Оператора, Управляющая ЭВМ, Клиенты-наблюдатели; все это — ЭВМ типа IBM PC с ОС Windows, связанные друг с другом сетью Ethernet;

- драйверный уровень, компоненты которого, вообще говоря, могут соединяться с Контроллерами верхнего уровня другим способом, например, через COM-порт по протоколу MODBUS или комбинированным способом.

Целесообразно вопросы передачи информации между процессами для этих двух уровней рассмотреть раздельно.

2. СРЕДСТВА ПЕРЕДАЧИ ИНФОРМАЦИИ НА ВЕРХНЕМ УРОВНЕ

Средства передачи данных на верхнем уровне предназначены для управления и мониторинга САУ. Для обеспечения этих функций разработан WEB-интерфейс с использованием стандартных программных и аппаратных решений, что определяет требования к средствам передачи данных на верхнем уровне. Сеть верхнего уровня может быть локальной или глобальной, созданной с использованием любой существующей технологии передачи данных, но обязательно с поддержкой стека IP-протоколов.

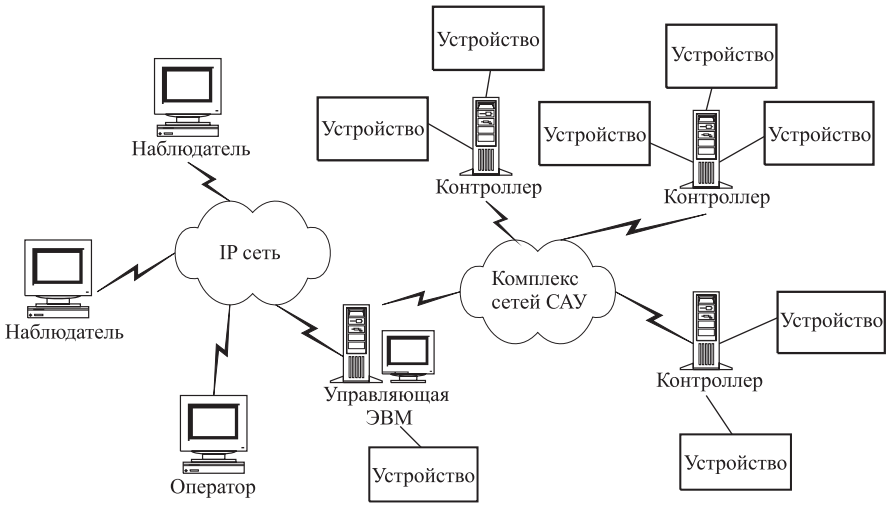

Рис. 1. Обобщенная схема распределенной САУ

2.1. Исследование используемых в САУ механизмов взаимодействия процессов. В разработанной структуре САУ программные компоненты представлены в виде отдельных задач (процессов), взаимодействующих по определенным правилам. Ввиду того, что САУ работает на распределенной конфигурации, потребовалось решить вопросы взаимодействия процессов в пределах одной ЭВМ, в однородной сети и в неоднородной. В работе [4] описан ряд способов передачи информации между процессами (IPC) в ОС Windows NT.

При детальном анализе нескольких доступных вариантов (см. данные таблицы) можно сделать следующие утверждения:

- наиболее простой и быстрый способ IPC в рамках локальной ЭВМ — сообщения WM_COPYDATA; его недостатки: 1) зависимость от ОС, 2) доступен только на локальной ЭВМ;
- несколько более сложный, но технологичный способ OLE 2.0; недостаток — платформенная зависимость; проекты фирм, сотрудничающих с Microsoft, по распространению данной технологии на другие платформы до сих пор не получили продолжения;
- использование сокетов позволяет проходить границы между процессами как на локальной ЭВМ, так и в сети; данный вариант прост в реализации и не зависит от платформы.

Сравнительные характеристики некоторых способов организации обмена данными между процессами

Способ обмена данными	Платформенная зависимость	Взаимодействие процессов на разных ЭВМ
Сообщение **WM_COPYDATA**	Да	Нет
DDE, OLE 2.0	Да	Да
Именованные каналы (**pipe**)	Частичная	Да
Сокеты	Нет	Да

В работе [5] с целью сравнения скоростных характеристик и надежности реализованы несколько вариантов интерфейсов драйверов с передачей информации:

- посредством сообщений Windows WM_COPDATA;
- исполняемый модуль с DCOM интерфейсом;
- исполняемый модуль с COM интерфейсом;
- COM интерфейс, реализованный в .dll;
- через сокеты, протокол TCP/IP.

Скоростные характеристики этих вариантов близки и не дают оснований для выбора. Вариант с интерфейсом на основе сообщений Windows вносит некоторое неудобство при использовании диалоговых управляющих программ, так как в случайный момент времени используемое оператором окно может временно стать неактивным. Из этих вариантов платформенную независимость может обеспечить только вариант использования протокола TCP/IP. На основании опытной эксплуатации автор пришел к выводам:

- для работы на локальной ЭВМ и на сети желательно и возможно использовать один и тот же механизм IPC;
- выбор варианта IPC следует сделать из соображений, диктуемых удобством организации работы и масштабирования распределенной САУ.

2.2. Выбор механизма IPC верхнего уровня распределенной САУ и разработка программ Выбор механизма IPC, по сути, сделан на материале предыдущего раздела — использование сокетов и протокола TCP/IP обеспечивает платформенную независимость и позволяет легко масштабировать распределенную САУ. Назначение используемых ЭВМ определяется списком загруженных базовых модулей, а конфигурация — назначенными сетевыми адресами в документации, легко редактируемой пользователями.

Задача свелась к разработке сервера **TCPserver** для Управляющей ЭВМ, а также **TCPclient** для ЭВМ Контроллера и Оператора. **TCPserver** на Управляющей ЭВМ выполняет:

- регистрацию подключившихся Контроллеров;
- прием и интерпретацию команд оператора.

На ЭВМ Контроллера, как было указано выше, программа **TCPclient** выполняет следующие функции:

- передает драйверам команды, поступающие от Управляющей ЭВМ;
- передает асинхронные сообщения от драйверов на Управляющую ЭВМ.

3. СРЕДСТВА ПЕРЕДАЧИ ДАННЫХ НА НИЖНЕМ УРОВНЕ

Передачу данных на нижнем уровне выполняет Подсистема Передачи Данных (ППД).

3.1. Требования, предъявляемые к ППД:

- обеспечение функций передачи событий и данных между отдельными компонентами системы;
- обеспечение независимости от средств, используемых для передачи данных;
- обеспечение функций диагностики коммуникаций;
- обеспечение принципов платформенной независимости при построении САУ;
- обеспечение принципов платформенной независимости на уровне исходного кода.

3.2. Функциональная схема ППД. Функциональная схема ППД представлена на рис. 2.

3.2.1. Маршрутизатор. Маршрутизатор выполняет функции преобразования форматов пакетов данных, а также объединения всех линий связи, используемых в системе, в единую логическую сеть передачи данных.

3.2.2. Ядро. Ядро является основным процессом, реализующим логику работы ППД. Собственно, ядро — инициатор передачи пакетов в ППД.

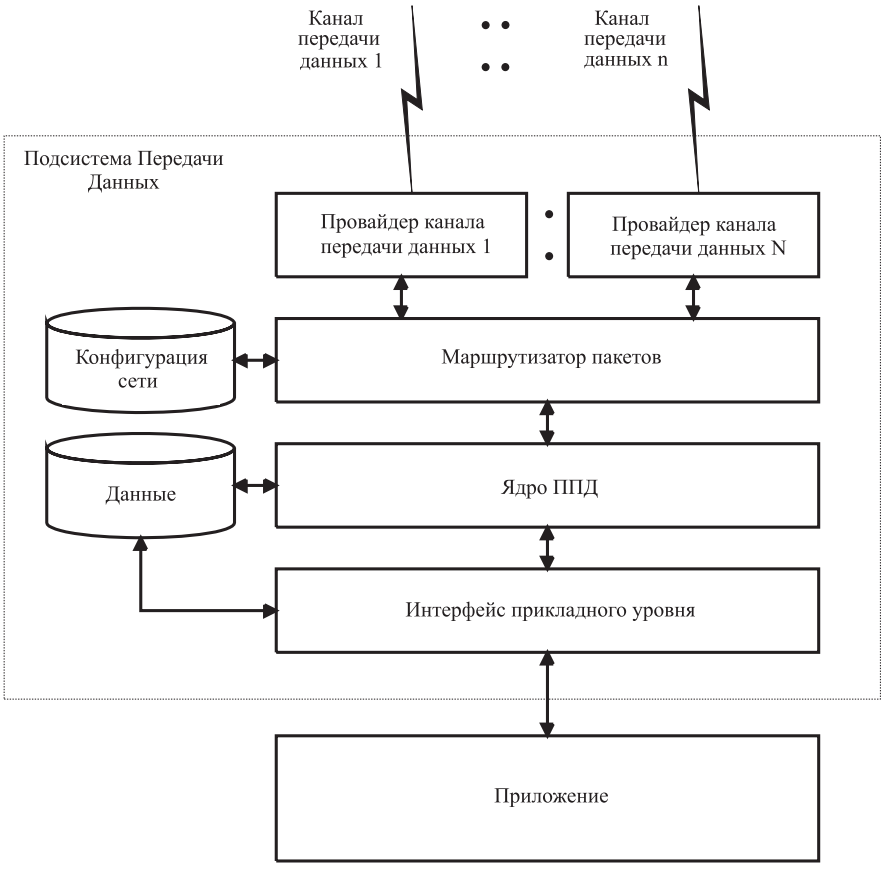

Рис. 2. Функциональная схема ППД

3.2.3. Интерфейс прикладного уровня. Интерфейс прикладного уровня обеспечивает взаимодействие приложения с ППД. Его спецификация унифицирована для любой программно-аппаратной платформы, чем обеспечивается платформенная независимость на уровне исходного кода.

3.2.4. Провайдер канала передачи данных. Провайдер канала передачи данных — это программы, предоставляющие единый интерфейс передачи пакетов по линиям связи, вне зависимости от реализации этих линий. Например, передача может быть осуществлена в локальных IP-сетях, через глобальную сеть интернет с использованием стандартных для нее протоколов передачи (IMAP, POP), посредством коммутируемых линий связи, средств радиосвязи, сотовой и спутниковой связи. Важнейшей функцией провайдера является

преобразование пакета данных из внутреннего представления в перемещаемый по линии связи формат. Форматы могут различаться для разных видов линий связи. Вместе с тем формат, определенный для данного вида линии связи, является для нее стандартом, что позволяет строить САУ на комплексе оборудования, состоящего из разных программно-аппаратных платформ, т. е. позволяет реализовать принцип платформенной независимости САУ.

3.3. Логическое представление системы с использованием ППД. Для рассмотрения логической архитектуры сети, построенной на базе ППД, определим некоторые используемые термины.

- **Управляющая сеть** — комплекс сетей, задачей которого является передача данных между компонентами САУ на нижнем уровне.
- **ЭВМ** — компьютер, составная часть управляющей сети.
- **Параметр** — элементарная именованная единица данных, представленных в системе.
- **Контроллер** — функционально законченный программный модуль, являющийся клиентом для ППД. Выполняет функции диспетчеризации как для реальных устройств в системе, так и для программных модулей, осуществляющих операции обработки данных. Это делает возможным осуществлять некоторые операции обработки и автоматизации локально, что, в свою очередь, позволит обеспечить больший уровень надежности системы и сократить трафик сети. Контроллер может экспортировать параметры в управляющую сеть и импортировать данные из управляющей сети.

На уровне приложения-клиента ППД все данные системы можно представить как дерево следующего вида:

$$< \text{ЭВМ} > — < \text{Контроллер} > — < \text{драйвер} > — < \text{параметр} >$$

Список машин, а также маршруты определяются оператором на стадии описания топологии сети и содержатся в специальной базе данных САУ.

Список Контроллеров ЭВМ определяется при регистрации их на Сервере приложения при запуске САУ и передается в ППД.

Список параметров САУ составляется двумя способами: 1) при определении конфигурации САУ пользователем или разработчиком; 2) динамически при запуске САУ. Первый способ составления этого списка (являющегося параметрической моделью САУ) описан в работе [6]. Второй способ выполняется по следующей схеме. Каждый драйвер экспортирует список своих параметров. При подключении драйверов к Контроллерам каждый из них импортирует эти списки параметров и составляет сводный список для своей

группы драйверов. При регистрации Контроллеров Сервер приложения составляет сводный рабочий список параметров САУ. Заметим, что рабочий список является подмножеством параметрической модели, используемым в конкретном сеансе работы САУ. При составлении задания (см. описание программы **PSJ** в работе [6]) используется список изменяемых параметров, который также является подмножеством параметрической модели. В момент составления задания рабочий список не известен, но при запуске САУ он может быть использован для проверки, достаточна ли используемая (например, урезанная) конфигурация САУ для выполнения составленного задания. Если все изменяемые параметры включены в рабочий список, то используемая конфигурация достаточна.

3.4. Передача данных.

3.4.1. Синхронизация. По способу инициации обмена данными системы сбора данных можно разделить на две основные группы, в которых

1) инициатором служит управляющая программа — синхронный обмен (или обмен по запросу);

2) инициатором является источник данных — асинхронный обмен (или обмен по событию).

Простота реализации синхронизации при синхронном обмене сделала такой способ передачи данных достаточно популярным. Примерами реализации синхронного обмена может быть сеть MODBUS, USB и другие.

Недостаток первого способа особенно сильно проявляется в случаях, когда в системе ожидается некоторое редкое событие, время реакции на которое должно быть минимальным. Для решения этой задачи требуется вести постоянный опрос источников данных с периодом меньшим, чем время реакции. В случае, если требуемое время реакции мало, сеть будет излишне загружена «бесполезными» сообщениями — запросами на передачу. Кроме того, если в сети количество устройств, требующих определенного времени реакции системы, достаточно велико, задача становится неразрешимой, что вызвано увеличением периода опроса всех этих устройств. Решение проблемы снижения трафика — в использовании систем сбора данных второй группы. Сеть IP является хорошим примером сети с асинхронным обменом. Однако и такой способ синхронизации не является универсальным для решения некоторых задач. Недостаток его может проявиться в случаях, когда время считывания данных определяется управляющей программой, а не источником данных. Как и в первом случае, можно запрограммировать источник на частую трансляцию данных, однако это снова приведет к излишней загрузки сети.

При проектировании ППД объединены положительные качества обоих способов синхронизации. При конфигурировании системы оператор в зави-

симости от поставленной задачи может выбрать тот или иной способ синхронизации для каждого из параметров. Вопросами реализации синхронизации данных тем или иным способом занимается ядро ППД.

3.4.2. Алгоритм работы ядра ППД — передача пакетов. Любой клиент ППД может получить доступ к данным системы. Данные в системе передаются пакетами. Алгоритм выработки решения о формировании и передаче пакета ядром ППД основан на анализе содержания следующих таблиц:

- Реестра Опубликованных Данных (РОД),
- Реестра Заявок (РЗ),
- Реестра Рассылок (РР).

На этапе инициации ППД каждый контроллер в обращении к ППД публикует список параметров подключенных к нему драйверов, их свойства и способ взаимодействия ППД с контроллером. Эти данные ППД заносит в РОД. Клиенты подают заявки на данные с указанием способа и параметров их синхронизации. ППД регистрирует эти данные в Реестре Заявок и очищает Реестр Рассылок.

В дальнейшем каждый контроллер по мере получения данных в асинхронном режиме обращается к ППД для регистрации в РОД текущих значений.

Логической структурой, обеспечивающей передачу пакета данных в системе, является виртуальный канал передачи данных (ВКПД). Виртуальный канал создается ППД после того, как клиент подаст заявку (подпишется) на необходимые ему данные. Каждый ВКПД представляет собой отдельную задачу, которая реализует доступ программы-клиента к опубликованным данным. Синхронизация данных обеспечивается ВКПД автоматически. Для одного ВКПД список данных может включать параметры нескольких контроллеров. При каждом изменении состояния ППД (т. е. при появлении новых данных или запроса) в зависимости от декларированного способа синхронизации выполняются следующие операции:

- в случае синхронного режима обмена данные читаются из устройства и регистрируются в РОД;
- производится проверка наличия запрашиваемых данных по записям в РОД;
- при наличии данных выполняется запись в таблицу РР;
- выполняется рассылка данных по списку РР.

Таким образом, виртуальный канал определяется соответствующими записями в двух таблицах: РЗ и РР.

ЗАКЛЮЧЕНИЕ

Отличительной особенностью данной подсистемы передачи данных является использование унифицированного кода, настраиваемого на конкретную конфигурацию без перетрансляции. Помимо этого, важными являются следующие свойства ППД:

- возможность работать в распределенной САУ;
- использование оборудования различных стандартов;
- наличие синхронного и асинхронного режимов работы.

В заключение автор пользуется случаем, чтобы поблагодарить руководство НПЦ «Аспект» и коллег по работе за предоставленные условия и помощь в работе.

ЛИТЕРАТУРА

1. *Kraimer M. et al.* EPICS Input/Output Controller Application Developer's Guide. Release 3.14.8, 2005. Argonne National Laboratory, http://www.aps.anl.gov/epics/
2. http://www.can-expo.ru/files/2006/
3. *Астахова Н. В. и др.* Распределенная беспроводная система регистрации с синхронизацией потоков данных. Препринт ОИЯИ Р13-2006-41. Дубна, 2006.
4. *Просис Дж.* Взаимодействие процессов в системе Windows NT // PC Magazine, June, 24, 1997.
5. *Астахова Н.В. и др.* Программный комплекс АС (автоматизация спектрометрии). 3. Методика управления окружением образца и применение ее в программе интерактивного управления спектрометром // ПТЭ. 2005. № 5. с. 36–43.
6. *Мазный Н. Г., Саламатин И. М., Саламатин К. М.* Генерация программ автоматизации экспериментов из модулей в формате загрузки. Препринт ОИЯИ Р13-2007-93. Дубна, 2007.

Получено 11 декабря 2008 г.

Редактор *М. И. Зарубина*

Подписано в печать 20.05.2009.
Формат 60 × 90/16. Бумага офсетная. Печать офсетная.
Усл. печ. л. 0,75. Уч.-изд. л. 0,88. Тираж 290 экз. Заказ № 56603.

Издательский отдел Объединенного института ядерных исследований
141980, г. Дубна, Московская обл., ул. Жолио-Кюри, 6.
E-mail: publish@jinr.ru
www.jinr.ru/publish/