КОМП'ЮТЕРНІ ЗАСОБИ, МЕРЕЖІ ТА СИСТЕМИ

S.V. Zinchenko

HIGHLY PRODUCTIVE REAL-TIME SYSTEMS FOR THE LARGE-SCALE APPLICATIONS

The article discusses creation of highly productive real-time systems based on data-centric architecture. For performance issues, efficiency and maintainability of such systems it is proposed using technology of data distribution based on the data distribution service (DDS), with time synchronization based on IEEE 1588.
Key words: real time, data-centric system, highly productive system.

Розглянуті питання створення високопродуктивних систем реального часу на базі датацентричної архітектури з використанням технології розподілу даних (DDS), з підтримкою стандарту синхронізації часу IEEE 1588.
Ключові слова: реальний час, датацентрична система, високопродуктивна система.

Рассмотрены вопросы создания высокопродуктивных систем реального времени на базе датацентрической архитектуры с использованием технологии распределения данных (DDS), с поддержкой стандарта синхронизации времени IEEE 1588.
Ключевые слова: реальное время, датацентрическая система, высокопродуктивная система.

УДК:519.95: 518.0: 621.391: 681.325

С.В. ЗИНЧЕНКО

ВЫСОКОПРОДУКТИВНЫЕ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ ДЛЯ КРУПНОМАСШТАБНЫХ ПРИЛОЖЕНИЙ

Введение. Современные крупномасштабные приложения характеризуются тремя основными свойствами: сбор и обработка данных в реальном масштабе времени, причем, существенным является как количество передаваемых данных, так и объектов, вовлеченных в этот обмен данными, они различны и могут даже измениться в течение некоторого времени. Например, авиа диспетчерская служба, финансовая обработка транзакций, командование и управление боевыми действиями сухопутных или военно-морских сил, или системы автоматизации производства, являются примерами датацентрических систем [1].

Датацентрическая архитектура сглаживает информацию или шаблоны распределения данных, делая источники данных доступные для любого санкционированного узла в сети, который хочет использовать данные [2]. Она позволяет децентрализовать приложения на серверах и рассредоточить их на сетевых компьютерах. В децентрализованных или распределенных системах намного проще обеспечить отказоустойчивость и масштабируемость.

Для децентрализации систем в реальном времени необходимо использовать новые технологии при построении коммутируемых сетей передачи данных. Аппаратная часть коммутируемых сетей должна обеспечивать достаточную производительность по каналам передачи данных в реальном времени.

Сетевое программное обеспечение должно обеспечивать взаимодействие в реальном

времени программных компонентов и служб распределения данных и синхронизации времени. Служба распределения данных (DDS) определяет publisher-subscriber механизм, устанавливающий соответствие между топологией и возможностями коммутируемой сети (рис. 1) [3]. Избыточность и гибкость аппаратного уровня достигается, по необходимости, добавлением или удалением сетевых узлов и/или коммутированием для разворачивания или сворачивания сетевой структуры. Избыточность и гибкость программного уровня достигается, по необходимости, добавлением или удалением производителей (источников) и потребителей данных DDS. Комбинация коммутируемой сети и DDS позволяет аппаратным средствам и программному обеспечению адаптироваться к изменяющимся системным требованиям.

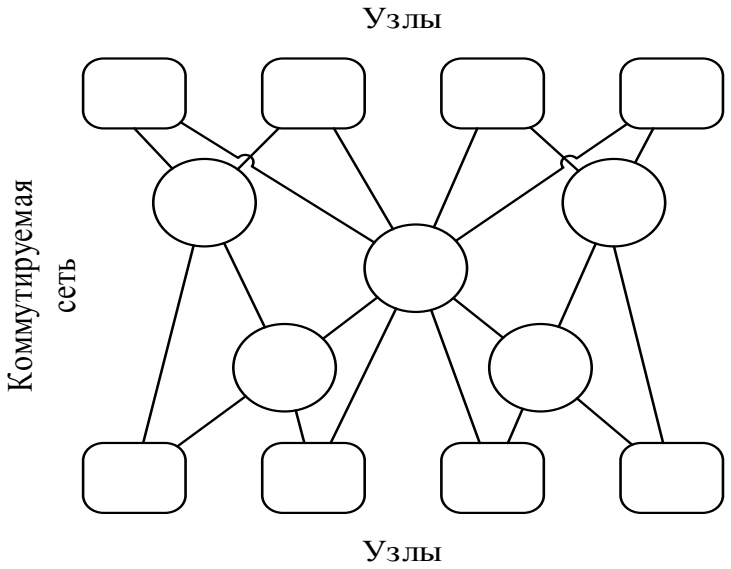

РИС. 1. DDS механизм устанавливающий соответствие между топологией и возможностями коммутируемой сети

Использование коммутируемых сетей передачи данных и DDS при создании комплексных, распределенных систем позволяет решить ряд сложнейших проблем, возникающих при использовании централизованного подхода. В таблице показаны общие требования и проблемы, с которыми сталкиваются разработчики при использовании централизованного подхода. Сложность разрабатываемого ПО, в этом случае, увеличивает время разработки системы и усложняет последующее ее обслуживание. В некоторых случаях сложность реализации проекта настолько высока, что возникает вопрос о целесообразности его создания.

ТАБЛИЦА. Общие требования, существующие решения и проблемы производительности, работоспособности и обслуживаемости

Требования	Необходимость	Существующие решения	Проблемы
Производительность	– увеличение функциональных требований для новых систем; – увеличение требований для развернутых систем	– различные схемы соединения параллельной шины; – последовательные соединения типа Ethernet	– ограниченное физическое разделение плат (обычно <1м); – ограниченная полоса пропускания, высокие протокольные потери, недетерминированная производительность, если не использовать специализированных протоколов
Работоспособность	– общедоступные или отказоустойчивые системы	– избыточные шины, платы или полностью избыточные системы	– сложная логика обхода отказа и дорогое аппаратное дублирование; – если решено, при наличии избыточной «резервной» системы, то, в дополнение к расходам, решение занимает дополнительное физическое пространство
Обслуживаемость	– уменьшение стоимости жизненного цикла системы; – упрощение обновления системы	– проектирование с учетом резервирования слотов для будущих процессоров	– требуется изменение архитектуры ПО для использования на новых процессоров

Шина коммутирующих устройств уникальна тем, что позволяет всем узлам на шине "логически" связываться со всеми другими узлами (рис. 2). Каждый узел физически подключен к одному или нескольким коммутаторам. Коммутаторы могут быть подключены друг к другу. Эта топология избыточной или "коммутируемой" сети, в которой могут быть один или более избыточных физических трактов между любыми двумя узлами. Узел может быть логически под-

ключен с любым другим узлом через коммутатор(ы). Логический тракт – временный и может быть реконфигурирован или переключен между доступными физическими соединениями. Коммутируемые сети, среди других особенностей, могут использоваться для обеспечения отказоустойчивости и масштабируемости без непредсказуемого ухудшения производительности.

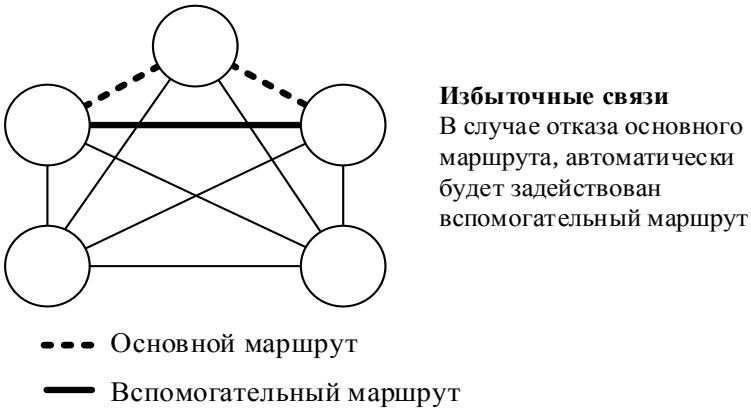

РИС. 2. Интегрированные и внешние коммутаторы обеспечивают избыточный путь коммутируемой сети

Служба распределения данных (DDS). Она используется для распределения данных в реальном времени. DDS, характеризуется рядом Издателей, и Подписчиков (publisher-subscriber) (рис. 3), обменивающихся данными с помощью топиков. Топик идентифицируется названием и типом данных. Издатель данных объявляет намерение опубликовать данные в топик; подписчик данных регистрирует свой интерес в получении данных, изданных в топике. Управляющее ПО поставляет данные, опубликованные в топике издателем подписчикам, подписавшимся на этот топик. Издатель может публиковать множество топиков, а потребитель может подписаться на их множество.

Уровень управляющего ПО изолирует производителей данных от потребителей, тем самым разделяет издателей от подписчиков; они не знают физические адреса друг друга, и являются анонимными. Такой подход позволяет создавать сложные шаблоны распределения данных при минимальных затратах ресурсов и времени. Анонимность упрощает определение избыточных издателей для отказоустойчивых систем. Управляющее ПО также упрощает подключение и удаление узлов из сети, приложений, перемещаемых от узла к узлу как требуется в сбалансированных системах.

Издатели отправляют данные непосредственно подписчикам, без посредников. Так как подписчики получают данные, базируясь только на топике (название и тип), а не на абсолютном адресе, мы можем резервировать изда-

телей для создания устойчивых сценариев восстановления после отказа, как показано на рис 4.

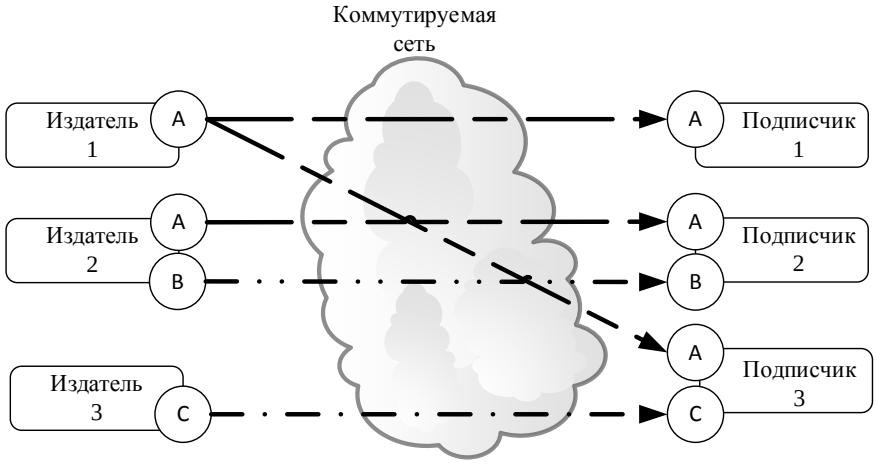

РИС. 3. DDS обеспечивает прямую анонимную связь между издателями и потребителями данных. У топика A есть первичный издатель 1, и резервный издатель 2. Отметим, что номинально, когда издатель 1 является активным, потребители не получают данные от резервного издатель 2

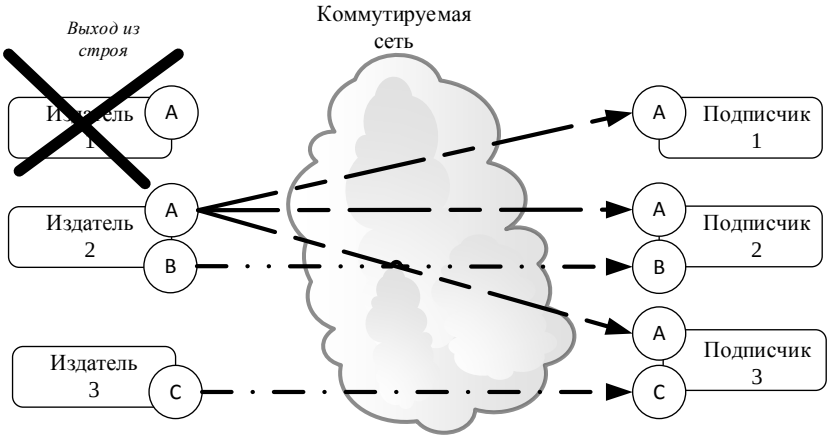

РИС. 4. DDS предусматривает автоматическое восстановление после отказа. Когда первичный издатель 1 топика A выходит из строя, управляющее ПО автоматически переключается на резервный издатель 2 топика A. Подписчики получают непрерывные данные

Важнейшее свойство DDS – это своя способность к гибкому и точному определению требований производительности между всеми частями системы. Оно определяется использованием параметров QoS. Параметры QoS конфигурируют систему, и согласуют издателей и подписчиков, точно определяя, как информация должна распространяться между узлами. Согласованность QoS обеспечивает предсказываемость производительности и контроль за системными ресурсами, в реальном масштабе времени, сохраняя модульность, масштабируемость и живучесть publisher-subscriber модели.

Параметры QoS управляют фактически каждым аспектом модели DDS и основных связующих механизмов. Управляющее ПО DDS отвечает за предоставление подписчику издателем необходимых данных по запросу, при этом, устанавливая связь или указывая ошибку несовместимости. При этом обеспечивая гарантии того, что участники обмена данными получают необходимый, для систем реального времени, уровень обслуживания.

Синхронизация времени. Существование системы не возможно без механизмов синхронизации времени ее компонентов. Стандарт IEEE 1588 позволяет осуществлять очень точную синхронизацию в сети Ethernet [4]. Протокол PTP первоначально был создан компанией Agilent для задач контроля и управления. Метод синхронизации базируется на работах Джона Эйдсона (John Eidson), он как председатель комитета по стандартизации был ответственным за одобрение стандарта в ноябре 2002 г. Протокол предназначен для использования в небольших однородных, а также неоднородных локальных сетях. Особое внимание его разработчики уделили минимизации необходимых вычислительных ресурсов, так что PTP может быть реализован в низкоуровневых и недорогих терминальных устройствах.

С помощью IEEE 1588 можно синхронизировать локальные часы в датчиках, исполнительных механизмах и других терминальных устройствах с точностью менее 1 мкс, используя при этом ту же сеть, по которой передаются данные. Подобно другим протоколам синхронизации, PTP выбирает наилучшее возможное согласование во времени между передаваемыми и получаемыми данными.

Основная функция протокола заключается в том, чтобы обеспечить с помощью наиболее точных в сети часов синхронизацию всех остальных устройств. Часы, у которых имеется только один сетевой порт, называются ординарными. Существует два типа часов: ведущие (Master) и подчиненные (Slave). В принципе, и первые, и вторые могут выполнять обе функции. Категория точности часов определяется протоколом, и наилучшие выбираются автоматически с помощью соответствующего алгоритма.

Точность синхронизации очень сильно зависит от сети и используемых компонентов. В этом смысле переход через наименее детерминированные устройства, к примеру, маршрутизаторы и коммутаторы, также может быть учтен протоколом с помощью так называемых граничных часов (Boundary clock).

Функции администрирования и конфигурирования часов входят в протокол управления.

PTP базируется на многоадресных IP-коммуникациях и не ограничивается только сетями Ethernet – он может применяться на любой шинной системе, которая поддерживает многоадресную рассылку.

Каждые подчиненные часы синхронизируются с ведущими часами посредством обмена сообщениями. Сам процесс синхронизации состоит из двух фаз. В первой корректируется расхождение между ведущими и подчиненными часами. Вторая фаза процесса синхронизации заключается в измерении значения времени задержки прохождения пакетов между ведущим и подчиненным устройствами. С этой целью подчиненные часы направляют ведущим пакет, который называется delay request. Ведущие часы фиксируют время приема пакета и посылают его значение подчиненным часам в специальном пакете delay response. Затем подчиненные часы вычисляют время задержки, используя соответствующие временные метки. В этом процессе особо важным является равнозначность задержки в каждом из направлений.

Измерения времени задержки выполняются нерегулярно и через большие отрезки времени (по умолчанию – через случайные значения между 4 и 60 с), чем измерения временного сдвига. Таким способом достигается разгрузка сети и терминальных устройств.

В отличие от ординарных, граничные часы имеют несколько портов. По отношению к каждому из них устройство ведет себя как ординарные часы для соответствующего сегмента сети. В типичном случае локальные часы в граничных устройствах синхронизируются главными ведущими часами через PTP-порт в режиме подчиненных. Остальные порты выполняют роль ведущих и синхронизируют другие устройства в сети. Граничные часы могут иметь только один подчиненный порт и любое количество ведущих. На рис. 5 показана типичная схема использования граничных часов.

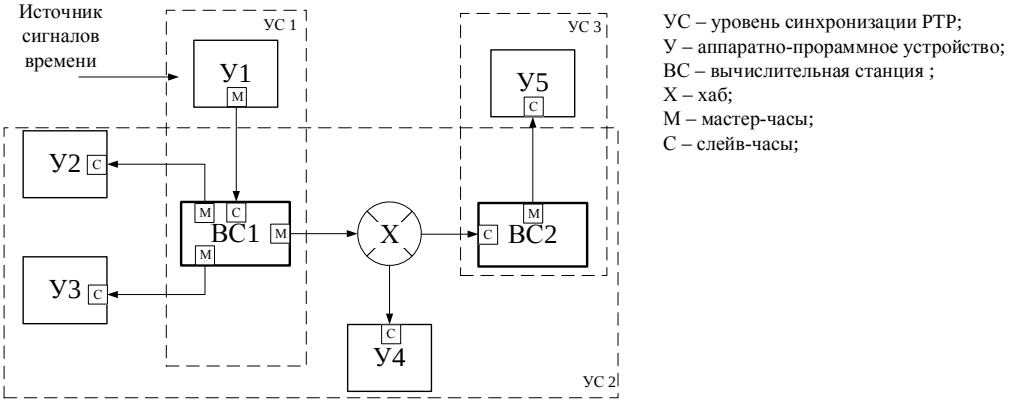

РИС. 5. Пример использования граничных часов

Подводя итоги, можно сказать, что PTP обеспечивает синхронизацию в субмикросекундном диапазоне и потенциально может предоставить еще большую точность.

Особенность архитектуры PTP – это разделение критической ко времени части, реализованной в аппаратных средствах и описанной в протоколе, и программной части. Поэтому протокол выполняется в процессе с низким приоритетом, например, на процессоре с низкой производительностью. Модуль аппаратных средств состоит из таймера высокой точности и модуля временной метки (TSU) для генерации временной метки. Программный компонент реализует сам протокол IEEE 1588 и имеет связь с таймером реального времени и модулем временных меток реализованных на аппаратных средствах.

Важным для портирования реализации протокола PTP является программная часть компонента, которая должна быть максимально независимой от операционной системы. Для достижения данной цели используется три уровня с различным уровнем абстракции. Уровень реализации протокола операционной системой не зависит от PTP. Абстрактный уровень ОС формирует интерфейс между PTP и используемой операционной системой. Функции – задачи/процессы, семафоры, таймеры, сокеты и т. д. – предоставляются операционной системой.

Высший уровень, реализуемый PTP для синхронизации часов в сети и может быть реализован на различных элементах коммуникаций (компьютер, коммутатор, маршрутизатор и т. д.). Это тот уровень, где находятся фактические сведения для синхронизации различных элементов коммуникаций. В пределах этого уровня протокола используются только функции, описанные в протоколе ANSI/ISO C. Это гарантирует, что программная реализация может быть перенесена на различные платформы без значительного реинжиниринга и изменения принципов функционирования. Диспетчер протокола гарантирует атомное выполнение функций в единственном процессе. Связь между протоколом и абстрактным уровнем ОС реализована в виде очереди с тремя определенными интерфейсами.

Средний уровень инкапсулирует операционно-зависимые функции, которые должны быть адаптированы в случае необходимости.

Интерфейс Timestamp предоставляет PTP возможность определения временных меток для Sync и DelayRequest сообщений. В зависимости от требования точности, временная метка может быть непосредственно сгенерирована модулем аппаратных средств (TSU) или программным обеспечением. Программные временные метки лучше всего генерировать в операционно-зависимом NIC драйвере (RX-ISR, процесс отправки) как находящемся наиболее близко к передающей среде.

Локальные часы считываются и изменяются через Clock Interface. Эти функции, также требуются в переработке в зависимости от операционной системы. Приложения, которые не обеспечивают real-time часы, используют часы операционной системы для синхронизации или другие оптимизированные ре-

шения, типа nanokernel для ОС *nix семейства. В добавлении к управлению локальными часами, этот интерфейс также содержит алгоритмы управления, которые являются ответственными за качество (точность, стабильность, переходные характеристики и т. д.) синхронизации времени.

Port Interface используется, для отправки и получения сообщений PTP. IEEE 1588 использует исключительно UDP/IP Multicast пакеты и поэтому позволяют использовать интерфейс сокетов стека протокола IP для отправки и получения сообщений.

Требованиями к временным задержкам можно пренебречь, потому что временная метка генерируется непосредственно на транспортном уровне. Для управления протоколом (конфигурация, диагностика, пересылка сообщения PTP) используют API PTP.

Выводы. Улучшение рабочих характеристик разрабатываемых систем реального времени достигается с помощью комбинирования коммутируемых сетей передачи данных и связующего ПО, основанного на архитектуре DDS, с поддержкой механизмов синхронизации времени на основе протокола PTP (стандарт IEEE 1588). Рассмотренный подход, позволяет на базе коммутируемых сетей передачи данных создать гибкую прикладную среду, обеспечивающую потребности функционирования распределенных приложений через множество узлов сети.

1. *Nauman Arshad, Stewart Dewar and Ian Stalker*. Serial Switched Fabrics Enable New Military System Architectures // COTS Journal December 2005.
2. *Kaisler S.H.* Software paradigms. Wiley-Interscience,. Hoboken, N.J., 2005.
3. *OMG,* Data Distribution Services for Real-Time Systems, v1.1, Document formal/ 2005-12-04, http://www.omg.org, December 2005.
4. *Dlugy-Hegwer R., Huckeba H.* Designing and Testing IEEE 1588 Timing Netwotks // Symmetricom Timing, Test and Measurement Division. – January 2007.
5. *Eidson J.C.* Measurement, Control, and Communication Using 1588. – Springer, 2006.

Получено 14.03.2014