Открытые системы, процессы стандартизации и профили стандартов

Источник: http://citforum.ru/database/articles/art_19.shtml

Сергей Кузнецов

1. Введение: понятие подхода Открытых Систем

Применение подхода открытых систем в настоящее время является основной тенденцией в области информационных технологий и средств вычислительной техники, поддерживающих эти технологии. Идеологию открытых систем реализуют в своих последних разработках все ведущие фирмы - поставщики средств вычислительной техники, передачи информации, программного обеспечения и разработки прикладных информационных систем. Их результативность на рынке информационных технологий и систем определяется согласованной (в пред конкурентной фазе) научно-технической политикой и реализацией стандартов открытых систем.

Что понимается под открытыми системами?

Для рассмотрения этого вопроса воспользуемся определениями открытых систем, которые приведены в руководстве, изданном Французской ассоциацией пользователей UNIX (AFUU) в 1992 году.

"Открытая система - это система, которая состоит из компонентов, взаимодействующих друг с другом через стандартные интерфейсы". Это определение, данное одним из авторов упомянутого руководства Жаном-Мишелем Корну, подчеркивает системный аспект (структуру открытой системы).

"Исчерпывающий и согласованный набор международных стандартов информационных технологий и профилей функциональных стандартов, которые специфицируют интерфейсы, службы и поддерживающие форматы, чтобы обеспечить интероперабельность и мобильность приложений, данных и персонала". Это определение, данное специалистами IEEE, подчеркивает аспект среды, которую предоставляет открытая система для ее использования (внешнее описание открытой системы).

Вероятно, одно достаточно полное и общепринятое определение открытых систем еще не сформировалось. Однако сказанного выше уже достаточно, чтобы можно было рассмотреть общие свойства открытых систем и выяснить существо связанных с ними проблем.

Общие свойства открытых систем обычно формируются следующим образом:

- расширяемость/масштабируемость -extensibility/scalability,
- мобильность (переносимость) - portalility,
- интероперабельность (способность к взаимодействию с другими системами) - interoperability,
- дружественность к пользователю, в т.ч. - легкая управляемость - driveability.

Эти свойства, взятые по отдельности, были свойственны и предыдущим поколениям информационных систем и средств вычислительной техники. Новый взгляд на открытые системы определяется тем, что эти черты рассматриваются в совокупности, как взаимосвязанные, и реализуются в комплексе.

2. Архитектура Открытых Систем

Понятие "система" носит двоякий характер. С одной стороны, по общему определению, система - это совокупность взаимодействующих элементов (компонентов), аппаратных и/или программных. С другой стороны, система может выступать в качестве компонента другой, более сложной системы, которая в свою очередь может быть компонентом системы следующего уровня.

В связи с этим нужно уточнить представление об архитектуре систем и средств, как внешнем их описании (reference model) с точки зрения того, кто ими пользуется. Архитектура открытой системы, таким образом, оказывается иерархическим описанием ее внешнего облика и каждого компонента с точки зрения:

- пользователя (пользовательский интерфейс),
- проектировщика системы (среда проектирования),
- прикладного программиста (системы и инструментальные средства /среды программирования),
- системного программиста (архитектура ЭВМ),
- разработчика аппаратуры (интерфейсы оборудования).

Предлагаемый взгляд на архитектуру открытых систем вытекает из указанной выше необходимости комплексной реализации общих свойств открытости и является расширением принятого понятия об архитектуре ЭВМ по Г.Майерсу.

Для примера рассмотрим архитектурное представление системы обработки данных, состоящей из компонентов четырех областей: пользовательского интерфейса (соответственно точкам зрения всех указанных выше групп), средств обработки данных, средств представления и хранения данных, средств коммуникаций. Для этого представления требуется использовать три уровня описаний: среды, которая представляется системой, операционной среды (системы), на которую опираются прикладные компоненты, и оборудования. Каждый из этих уровней разделен для удобства на два подуровня (см.табл.1).

Уровень среды для конечного пользователя (user environment) характеризуется входными и выходными описаниями (генераторы форм и отчетов), языками проектирования информационной модели предметной области (языки 4GL), функциями утилит и библиотечных программ и прикладным уровнем среды коммуникаций, когда требуются услуги дистанционного обмена информацией. На этом же уровне определена среда (инструментарий) прикладного программирования (application environment): языки и системы программирования, командные языки (оболочки операционных систем), языки запросов СУБД, уровни сессий и представительный среды коммуникаций.

На уровне операционной системы представлены компоненты операционной среды, реализующие функции организации процесса обработки, доступа к среде хранения данных, оконного интерфейса, а также транспортного уровня среды коммуникаций. Нижний подуровень операционной системы - это ее ядро, файловая система, драйверы управления оборудованием, сетевой уровень среды коммуникаций.

На уровне оборудования легко видеть привычные разработчикам ЭВМ составляющие архитектуры аппаратных средств:

- система команд процессора (процессоров),
- организация памяти,
- организация ввода-вывода и т.д.,

а также физическую реализацию в виде:

- системных шин,
- шин массовой памяти,
- интерфейсов периферийных устройств,
- уровня передачи данных,
- физического уровня среды хранения.

Представленный взгляд на архитектуру открытой системы обработки данных относится к одно-машинным реализациям, включенным в сеть передачи данных для обмена информацией. Понятно, что он может быть легко обобщен и на многопроцессорные системы с разделением функций, а также на системы распределенной обработки данных. Поскольку здесь явно выделены компоненты, составляющие систему, можно рассматривать как интерфейсы взаимодействия этих компонентов на каждом из указанных уровней, так и интерфейсы взаимодействия между уровнями.

Описания и реализации этих интерфейсов могут быть предметом рассмотрения только в пределах данной системы. Тогда свойства ее открытости проявляются только на внешнем уровне. Однако значение идеологии открытых систем состоит в том, что она открывает методологические пути к унификации интерфейсов в пределах родственных по функциям групп компонентов для всего класса систем данного назначения или всего множества открытых систем.

Стандарты интерфейсов этих компонент (де-факто или принятые официально) определяют лицо массовых продуктов на рынке. Область распространения этих стандартов являются предметом согласования интересов разных групп участников процесса информатизации - пользователей, проектировщиков систем, поставщиков программных продуктов и поставщиков оборудования.

Выше был рассмотрен пример архитектуры открытых систем, реализующих технологию обработки данных. Можно было бы представить аналогичным образом открытые системы для всех классов информационных технологий: обработки текстов, изображений, речи, машинной графики. Особенно актуально проработать подходы открытых систем для мультимедиа-технологий, сочетающих несколько разных представлений информации. Как известно, за рубежом эти работы проводятся различными ассоциациями и консорциумами заинтересованных фирм и академических организаций и международными организациями по стандартизации. К сожалению, российские специалисты в этих работах до сих пор в лучшем случае играют роль наблюдателей.

3. Преимущества идеологии открытых систем

Конечно, подход открытых систем пользуется успехом только потому, что обеспечивает преимущества для разного рода специалистов, связанных с областью компьютеров.

Для пользователя открытые системы обеспечивают следующее:

- новые возможности сохранения сделанных вложений благодаря свойствам эволюции, постепенного развития функций систем, замены отдельных компонентов без перестройки всей системы;
- освобождение от зависимости от одного поставщика аппаратных или программных средств, возможность выбора продуктов из предложенных на рынке при условии соблюдения поставщиком соответствующих стандартов открытых систем;

- дружественность среды, в которой работает пользователь, мобильность персонала в процессе эволюции системы;
- возможность использования информационных ресурсов, имеющихся в других системах (организациях).

Проектировщик информационных систем получает:

- возможность использования разных аппаратных платформ;
- возможность совместного использования прикладных программ, реализованных в разных операционных системах;
- развитые 4средства 0инструментальных сред, поддерживающих проектирование;
- возможности использования готовых программных продуктов и информационных ресурсов.

Разработчики общесистемных программных средств имеют:

- новые возможности разделения труда, благодаря повторному использованию программ(reusability);
- развитые инструментальные среды и системы программирования;
- возможности модульной организации программных комплексов благодаря стандартизации программных интерфейсов.

Это последнее свойство открытых систем позволяет пересмотреть традиционно сложившееся дублирование функций в разных программных продуктах, из-за чего системы, интегрирующие эти продукты, непомерно разрастаются по объему, теряют эффективность. Известно, что в той же области обработки данных и текстов многие продукты, предлагаемые на рынке (текстовые редакторы, настольные издательства, электронные таблицы, системы управления базами данных) по ряду функций дублируют друг друга, а иногда и подменяют функции операционных систем. Кроме того, замечено, что в каждой новой версии этих продуктов размеры их увеличиваются на 15%.

В распределенных системах, содержащих несколько рабочих мест на персональных компьютерах и серверов в локальной сети, избыточность программных кодов из-за дублирования возрастает многократно. Идеология и стандарты открытых систем позволяют по-новому взглянуть на распределение функций между программными компонентами систем и значительно повысить тем самым эффективность. Частично этот подход обеспечивает компенсацию затрат ресурсов, которые приходится платить за преимущества открытых систем относительно закрытых систем, ресурсы которых в точности соответствуют задаче, решаемой системой.

4. Открытые Системы и объектно-ориентированный подход

В связи с применением подхода открытых систем весьма перспективным направлением представляется объектно-ориентированный стиль проектирования и программирования.

Объектно-ориентированное программирование - это относительно новый подход к разработке программных систем. Этот подход строится на следующих основных принципах:

- данные и процедуры объединяются в программные объекты;
- для связи объектов используется механизм посылки сообщения;
- объекты с похожими свойствами объединяются в классы;
- объекты наследуют свойства других объектов через иерархию классов.

Объектно-ориентированные системы обладают следующими 4основными свойствами:

- Инкапсуляция (скрытие реализации) - данные и процедуры объекта скрываются от внешнего пользователя, и связь с объектом ограничивается набором сообщений, которые "понимает" объект.
- Полиморфизм (многозначность сообщений) - одинаковые сообщения по-разному понимаются разными объектами, в зависимости от их класса.
- Динамическое (позднее) связывание - значение имени (область памяти для данных или текст программы для процедур) становится известным только во время выполнения программы.
- Абстрактные типы данных - объединение данных и операций для описания новых типов, позволяющие использовать новые типы наравне с уже существующими.
- Наследование - позволяет при создании новых объектов использовать свойства уже существующих объектов, описывая заново только те свойства, которые отличаются.

Заметим, что основные свойства открытых систем хорошо поддерживаются объектно-ориентированным подходом к реализации системы (Табл. 2). Рассмотрим отдельные аспекты этой поддержки.

1. Мобильность
   Инкапсуляция позволяет хорошо скрыть машинно-зависимые части системы, которые должны быть реализованы заново при переходе на другую платформу. При этом гарантируется, что остальная часть системы не потребует изменений. При реализации новых машинно-зависимых частей многое может быть взято из уже существующей системы благодаря механизму наследования.
2. Расширяемость
   Наследование позволяет сэкономить значительные средства при расширении системы, поскольку многое не нужно создавать заново, а некоторые новые компоненты можно получить, лишь слегка изменив старые. Кроме повторного использования, увеличивается также надежность, поскольку используются уже отлаженные компоненты.
   Возможность конструирования абстрактных типов данных для создания новых средств - обеспечивается самим понятием класса, объединяющего похожие объекты с одинаковым набором операций.
3. Интероперабельность
   Способность системы взаимодействовать с другими системами хорошо поддерживается принципом посылки сообщения и соответствующими понятиями полиморфизма и динамического связывания. В сообщении объекту (возможно удаленному) передается имя действия, которое должно быть им выполнено, и некоторые дополнительные аргументы сообщения. Как это действие выполнять - знает и решает только сам объект - получатель сообщения. От него только требуется выдать в ответ результат. Совершенно очевидно, что разные объекты будут по-разному реагировать на одинаковые сообщения (полиморфизм). Кроме того, очень удобно выбирать способ реализации в последний момент - при ответе на сообщение, в зависимости от текущего состояния системы (динамическое связывание). Для того, чтобы разные системы могли обмениваться сообщениями, необходима либо единая трактовка всех типов данных, в том числе абстрактных, либо индивидуальная процедура преобразования сообщения для каждой пары неодинаковых взаимодействующих систем. Простота понятия абстрактных типов данных в объектно-ориентированных системах существенно облегчает разработку такой процедуры.
4. Дружественность
   Удобство взаимодействия человека с системой требует от последней наличия всех

трех вышеуказанных качеств. Мобильность необходима ввиду быстрой смены старых и появления новых устройств, в частности, средств мультимедиа. Расширяемость требуется для разработки программной поддержки новых парадигм общения человека с машиной. Интероперабельность просто рассматривает человека как другую систему, с которой открытая система должна уметь взаимодействовать.

5. Стандарты Открытых Систем

В настоящее время в мире существует несколько авторитетных сообществ, занимающихся выработкой стандартов открытых систем. Однако исторически и, по-видимому, до сих пор наиболее важной деятельностью в этой области является деятельность комитетов POSIX. В этом разделе мы приведем краткий обзор этой деятельности.

Первая рабочая группа POSIX (Portable Operating System Interface) была образована в IEEE в 1985 г. на основе UNIX-ориентированного комитета по стандартизации /usr/group (ныне UniForum). Отсюда видна первоначальная направленность работы POSIX на стандартизацию интерфейсов ОС UNIX. Однако постепенно тематика работы рабочих групп POSIX (а со временем их стало несколько) расширилась настолько, что стало возможным говорить не о стандартной ОС UNIX, а о POSIX-совместимых операционных средах, имея в виду любую операционную среду, интерфейсы которых соответствуют спецификациям POSIX.

Сейчас функционируют и регулярно выпускают документы следующие рабочие группы POSIX.

POSIX 1003.0. Рабочая группа, выпускающая "Руководство по POSIX-совместимым средам Открытых Систем". Это руководство содержит сводную информацию о работе и текущем состоянии документов всех других рабочих групп POSIX, а также других тематически связанных организаций, связанных со стандартизацией интерфейсов Открытых Систем.

POSIX 1003.1. Интерфейсы системного уровня и их привязка к языку Си. В документах этой рабочей группы определяются обязательные интерфейсы между прикладной программой и операционной системой. С выпуска первой версии этого документа началась работа POSIX, и он в наибольшей степени связан с ОС UNIX, хотя в настоящее время интерфейсы 1003.1 поддерживаются в любой операционной среде, претендующей на соответствие принципам Открытых Систем. Заметим, что несмотря на очевидную важность 1003.1, в документе отсутствуют спецификации многих важных интерфейсов, в частности, интерфейсы системных вызовов, обеспечивающих межпроцессные взаимодействия.

POSIX 1003.2. Shell и утилиты. Рабочая группа специфицирует стандартный командный язык shell, основанный главным образом на Bourne shell, но включающий некоторые черты Korn shell. Кроме того, в документах этой рабочей группы специфицировано около 80 утилит, которые можно вызывать из процедур shell или прямо из прикладных программ. В документах серии 1003.2a описываются дополнительные средства, позволяющие пользователям работать с системой с помощью только ASCII-терминалов.

POSIX 1003.3. Общие методы проверки совместимости с POSIX. Целью рабочей группы является разработка методологии проверки соответствия реализаций стандартам POSIX. Документы рабочей группы используются в различных организациях при разработке тестовых наборов.

POSIX 1003.4. Средства, предоставляемые системой для прикладных программ реального времени. В соответствии с определением 1003.4, системой реального времени считается система, обеспечивающая предсказуемое и ограниченное время реакции. Работа ведется в

трех секциях: файловые системы реального времени, согласованные многопотоковые (multithread) архитектуры, а также в секции, занимающейся такими вопросами, как семафоры и сигналы.

POSIX 1003.5. Привязка языка Ада к стандартам POSIX. В документах этой рабочей группы определяются правила привязки программ, написанных на языке Ада, к системным средствам, определенным в POSIX 1003.1.

POSIX 1003.6. Расширения POSIX, связанные с безопасностью. Разрабатываемый набор стандартов базируется на критериях министерства обороны США и будет определять безопасную среду POSIX.

POSIX 1003.7. Расширения, связанные с администрированием системы. Стандарт, разрабатываемый рабочей группой, будет определять общий интерфейс системного администрирования, в частности, разнородных сетей. Отправной точкой является модель OSI.

POSIX 1003.8. Прозрачный доступ к файлам. Будут обеспечены интерфейсы и семантика прозрачного доступа к файлам, распределенным в сети. Работа основывается на анализе существующих механизмов: NFS, RFS, AFS и FTAM.

POSIX 1003.9. Привязка языка Фортран. Определяются правила привязки прикладных программ, написанных на языке Фортран, к основным системным средствам.

POSIX 1003.10. Общие черты прикладной среды суперкомпьютеров (Application Environment Profile - AEP).

POSIX 1003.11. Общие черты прикладной среды обработки транзакций (On-line Transaction Processing Application Environment - OLTP).

POSIX 1003.12. Независимые от протоколов коммуникационные интерфейсы. Разрабатываются два стандартных набора интерфейсов для независимых от сетевых протоколов коммуникаций "процесс-процесс". Результаты должны обеспечивать единообразную работу с TCP/IP, OSI и другими системами коммуникаций.

POSIX 1003.13. Общие черты прикладных сред реального времени. POSIX 1003.14. Общие черты прикладных сред мультипроцессоров. Помимо прочего, должны быть предложены соответствующие расширения стандартов других рабочих групп.

POSIX 1003.15. Расширения, связанные с пакетной обработкой. Определяются интерфейсы пользователя и администратора и сетевые протоколы для пакетной обработки.

POSIX 1003.16. Привязка языка Си. Задача проекта, выполняемого реально рабочей группой 1003.1, состоит в выработке правил привязки международного стандарта языка Си (ISO 9989) к независимым от языка интерфейсам, определяемым POSIX 1003.1-1990 (ISO 9945-1).

POSIX 1003.17. Справочные услуги и пространство имен. Задачей рабочей группы является анализ и выработка рекомендаций по работе со справочниками и пространством имен в контексте X.500.

POSIX 1003.18. Общие черты среды POSIX-платформы. В одном документе должны быть специфицированы основные характеристики интерактивной многопользовательской

прикладной платформы, соответствующей стандартам POSIX. Работа выполняется группой 1003.1.

6. Профили стандартов Открытых Систем

Интеграция компонентов в открытой системе должна следовать профилям стандартов на интерфейсы этих компонент.

Профиль составляют набор согласованных стандартов интерфейсов компонентов на каждом уровне системы (как было показано выше на примере системы обработки данных) и обеспечивают их совместимость.

Для определенности рассмотрения интерфейсов компонент и проведения необходимых анализов их реализуемости можно использовать модель среды открытых систем MUSIC, разработанную центральным агентством по компьютерам и телекоммуникациям (CCTA) Великобритании. Эта модель используется в руководстве фирмы Digital Equipment по построению открытых систем. Модель MUSIC содержит пять групп компонентов, из которых строятся открытые системы:

- управление (Management) - функции системной администрации, безопасности, управления ресурсами, конфигурацией, сетевое управление;
- пользовательский интерфейс (User Interface) - интерфейс пользователя с прикладными программами и со средой разработки приложений;
- системные интерфейсы для программ (Service Interface for Programs) - интерфейсы между прикладными программами и между прикладными программами и операционной системой, в частности API (Application Programs Interface);
- форматы информации и данных;
- интерфейсы коммуникаций.

Европейская рабочая группа по открытым системам (EWOS) предложила шесть профилей стандартов составляющих среды открытых систем:

- среда рабочих станций,
- среда серверов процессов,
- среда серверов данных,
- среда транзакций,
- среда реального времени,
- среда суперкомпьютеров.

Кроме указанного набора профилей по классам аппаратно-программных средств существует необходимость формирования вертикальных профилей открытых систем, ориентированных на проблемно-ориентированные области применения. В качестве таких первоочередных областей применения открытых систем в России можно назвать:

- интегрированные производственные системы,
- информационные системы (системы информационного обслуживания) с удаленным доступом к ресурсам,
- системы автоматизации учреждений,
- системы автоматизации банков,
- системы автоматизации научных исследований,
- системы передачи данных.

7. Заключение

Подход открытых систем обеспечивает слишком много преимуществ, чтобы можно было игнорировать его в России. Однако до сих пор все, что делается по этому поводу, основывается главным образом на энтузиазме. Просматриваются, как минимум, два необходимых и безотлагательных действия.

Во-первых, необходимо выполнить ряд научных проектов, связанных с анализом реализуемости международных стандартов в наших условиях, выбором и разработкой профилей стандартов открытых систем по областям их применения, как технической основы информационной инфраструктуры общества.

Во-вторых, требуется выработать и согласовать стандарты интерфейсов на разработку или приобретение аппаратных и программных средств.

Таблица 1

Иерархия представления архитектуры системы обработки данных

Уровень архитектуры системы обработки данных	**Компоненты системы обработки данных**			
	Интерфейсы	Средства обработки данных	Представление и хранение данных	Коммуникации
Среда для конечного пользователя и инструментарий прикладного программиста	Генераторы форм и отчетов	Утилиты и библиотеки	Языки программирования 4GL	OSI. Прикладной уровень
	Языки программные и командные языки (оболочки)	Прикладные программы	Языки запросов СУБД	OSI. Уровни сессий и представительный
Операционная система	Средства оконного интерфейса	Верхний уровень ОС (организация процесса обработки)	Средства доступа к среде хранения	OSI. Транспортный уровень
	Драйверы	Ядро операционной системы	Файловая система	OSI. Сетевой уровень
Оборудование	Системные интерфейсы (в т.ч. организация ввода-вывода)	Процессоры (система команд)	Организация памяти	OSI. Уровень передачи данных
	Периферийные устройства	Системная шина	Шины (интерфейс) массовой памяти	OSI. Физический уровень

Таблица 2

Сопоставление свойств открытых систем и объектно-ориентированных систем программирования

Свойства открытых систем	Дружественность (пользователь)	Мобильность (платформы)	Расширяемость (новые функции и области применения)	Интероперабельность (другие системы, пользователь)
Свойства объектно-ориентированных систем программирования	Объектное представление предметной области, наиболее удобное человеку. Сочетание всех других свойств при конструировании пользовательского интерфейса	Инкапсуляция (скрытие реализации)	Наследование, абстрактные типы данных	Полиморфизм, динамическое связывание