ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 16100-2— 2010

Системы промышленной автоматизации и интеграция

ПРОФИЛИРОВАНИЕ ВОЗМОЖНОСТИ ИНТЕРОПЕРАБЕЛЬНОСТИ ПРОМЫШЛЕННЫХ ПРОГРАММНЫХ СРЕДСТВ

Часть 2

Методология профилирования

ISO 16100-2:2003
Industrial automation systems and integration —
Manufacturing software capability profiling for interoperability —
Part 2: Profiling methodology
(IDT)

Издание официальное

Москва
Стандартинформ
2014

Предисловие

1 ПОДГОТОВЛЕН Научно-техническим центром ИНТЕК на основе собственного аутентичного перевода на русский язык международного стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 100 «Стратегический и инновационный менеджмент»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 21 декабря 2010 г. № 858-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 16100-2:2003 «Системы промышленной автоматизации и интеграция. Профилирование возможности интероперабельности промышленных программных средств. Часть 2. Методология профилирования» (ISO 16100-2:2003 «Industrial automation systems and integration — Manufacturing software capability profiling for interoperability — Part 2: Profiling methodology»).

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов соответствующие им национальные стандарты Российской Федерации, сведения о которых приведены в дополнительном приложении ДА

5 ВВЕДЕН ВПЕРВЫЕ

Правила применения настоящего стандарта установлены в ГОСТ Р 1.0—2012 (раздел 8). Информация об изменениях к настоящему стандарту публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок — в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (gost.ru)

Содержание

Введение

Разработка комплекса стандартов ИСО 16100 обусловлена необходимостью решения следующих проблем, связанных с:

a) постоянно увеличивающейся базой решений, зависящих от поставщиков;

b) трудностями, возникающими у пользователей при применении стандартов;

c) необходимостью перехода к модульным наборам инструментальных средств интеграции системы;

d) признанием того, что прикладное программное обеспечение и практический опыт его применения являются интеллектуальным капиталом предприятия.

Комплекс стандартов ИСО 16100 определяет формат профиля возможностей программного обеспечения, интерпретируемого компьютером в электронно-цифровой форме и не вызывающего трудностей при его чтении человеком, а также устанавливает метод, отражающий основные возможности программного обеспечения на производстве в соответствии с ролями, определенными жизненным циклом производственного приложения, независимо от архитектуры определенной системы или платформы реализации.

Настоящий стандарт разработан Техническим комитетом ИСО/ТК 184 «Системы промышленной автоматизации и интеграция», Подкомитетом ПК 5 «Архитектура, коммуникации и структуры интеграции».

Комплекс стандартов ИСО 16100 имеет общее наименование «Системы промышленной автоматизации и интеграция. Профилирование возможности интероперабельности промышленных программных средств» и включает в себя следующие части:

- часть 1. Структура;
- часть 2. Методология профилирования;
- часть 3. Службы интерфейса, протоколы и шаблоны возможностей;
- часть 4. Методы аттестационных испытаний, критерии и отчеты;
- часть 5. Методология согласования конфигураций профилей с помощью многоцелевых структур классов.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Системы промышленной автоматизации и интеграция

ПРОФИЛИРОВАНИЕ ВОЗМОЖНОСТИ ИНТЕРОПЕРАБЕЛЬНОСТИ ПРОМЫШЛЕННЫХ ПРОГРАММНЫХ СРЕДСТВ

Часть 2

Методология профилирования

Industrial automation systems and integration.
Manufacturing software capability profiling for interoperability.
Part 2. Profiling methodology

Дата введения — 2011—09—01

1 Область применения

Настоящий стандарт устанавливает методы конструирования профилей возможностей производственного программного обеспечения. Стандарт распространяется на программные изделия, используемые в производственном домене.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты, которые необходимо учитывать при использовании настоящего стандарта. В случае ссылок на документы, у которых указана дата утверждения, необходимо пользоваться только указанной редакцией. В случае, когда дата утверждения не приведена, следует пользоваться последней редакцией ссылочных документов, включая любые поправки и изменения к ним.

ИСО 16100 (все части) Системы промышленной автоматизации и интеграция. Профилирование возможности интероперабельности промышленных программных средств (ISO 16100 (all parts), Industrial automation systems and integration — Manufacturing software capability profiling for interoperability)

REC-xmlschema-1-20010502 Схема языка XML. Часть 1. Структура. Рекомендации W3C от 2 мая 2001 г. (REC-xmlschema-1-20010502, XML Schema Part 1: Structures — W3C Recommendation 02 May 2001)

REC-xmlschema-2-20010502 Схема языка XML. Часть 2. Типы данных. Рекомендации W3C от 2 мая 2001 г. (REC-xmlschema-2-20010502, XML Schema Part 2: Datatypes — W3C Recommendation 02 May 2001)

3 Термины и определения

В настоящем стандарте применены термины по ИСО 16100-1, а также следующие термины с соответствующими определениями:

3.1 **ассоциация** (association): Семантическое взаимоотношение между двумя или более классификаторами, определяющими связи между их экземплярами.

[ИСО/МЭК 19501-1]

3.2 **основная спецификация** (base specification): Основной стандарт или широко применяемая и доступная спецификация.

3.3 **класс возможности** (capability class): Элемент метода профилирования возможности, представляющий функциональность и поведение единицы программного обеспечения в отношении программного обеспечения для производственной деятельности.

Издание официальное

3.4 **интеграция профиля возможности** (capability profile integration): Процесс, в котором две или более единицы программного обеспечения взаимодействуют с помощью эквивалентных интерфейсов, конфигурируемых в совместимом виде, на что указывают их профили возможностей.

3.5 **классификатор** (classifier): Механизм, характеризующий поведенческие и структурные свойства.
[ИСО/МЭК 19501-1]

Примечание — Классификатор включает в себя интерфейсы, классы, типы данных и компоненты.

3.6 **элемент** (element): Элементарная составляющая модели.
[ИСО/МЭК 19501-1]

3.7 **сущность** (entity): Любая конкретная или абстрактная вещь, представляющая интерес.
[ИСО/МЭК 10746-2]

3.8 **интерфейс** (interface): Абстракция поведения объекта, состоящего из подмножества взаимодействий этого объекта, с учетом накладываемых ограничений при их возможном появлении.
[ИСО/МЭК 10746-2]

3.9 **объект** (object): Модель сущности.
[ИСО/МЭК 10746-2]

Примечание — Объект характеризуется поведением и состоянием и отличается от любого другого объекта. Объект инкапсулирован, т. е. любое изменение его состояния может произойти только в результате внутреннего действия или взаимодействия с его окружением. Объект взаимодействует с окружением в определенных точках взаимодействия. Основное внимание может быть направлено на поведение или состояние объекта. При акценте на поведение объекта он выполняет определенные функции и предлагает услуги, делая функцию доступной. При моделировании эти функции и услуги указывают в описании поведения объекта и его интерфейсе. Функция может выполняться как одним, так и несколькими взаимодействующими объектами одновременно.

3.10 **профиль** (profile): Совокупность одной или более основных спецификаций и/или подпрофилей, и в приемлемых случаях идентификация выбранных классов, согласующихся подмножеств, опций и параметров основных спецификаций или подпрофилей, необходимых для выполнения конкретной функции, деятельности или взаимосвязи.

Примечание — Определение данного термина см. ИСО/МЭК ТО 10000-1.

3.11 **роль** (role): Специфическое поведение сущности, указанное в определенном контексте и имеющее имя.
[ИСО/МЭК 19501-1]

Примечание — Роль может быть статической (например, конец соединения) или динамической (например, коллективная роль).

3.12 **таксономия** (taxonomy): Схема классификации профилей ссылок или набора профилей.
[ИСО/МЭК ТО 10000-1]

4 Сокращения

В настоящем стандарте применены следующие сокращения:

CORBA — технология построения распределенных объектных предложений (Common Object Request Broker Architecture);

IDL — язык описания интерфейсов (Interface Definition Language);

OMG — рабочая группа по развитию стандартов объектного программирования (Object Management Group);

PSL — язык спецификации процесса (Process Specification Language);

UML — унифицированный язык моделирования (Unified Modeling Language);

XML — язык (расширяемый) гипертекстовой разметки (eXtensible Markup Language).

5 Метод профилирования возможности

5.1 Концепция профилирования возможности

Основной целью разработки комплекса стандартов ИСО 16100 является обеспечение возможности интероперабельности производственных программных средств разных поставщиков. Концепция профиля возможности для интеграции интероперабельного программного обеспечения изображена на рисунке 1.

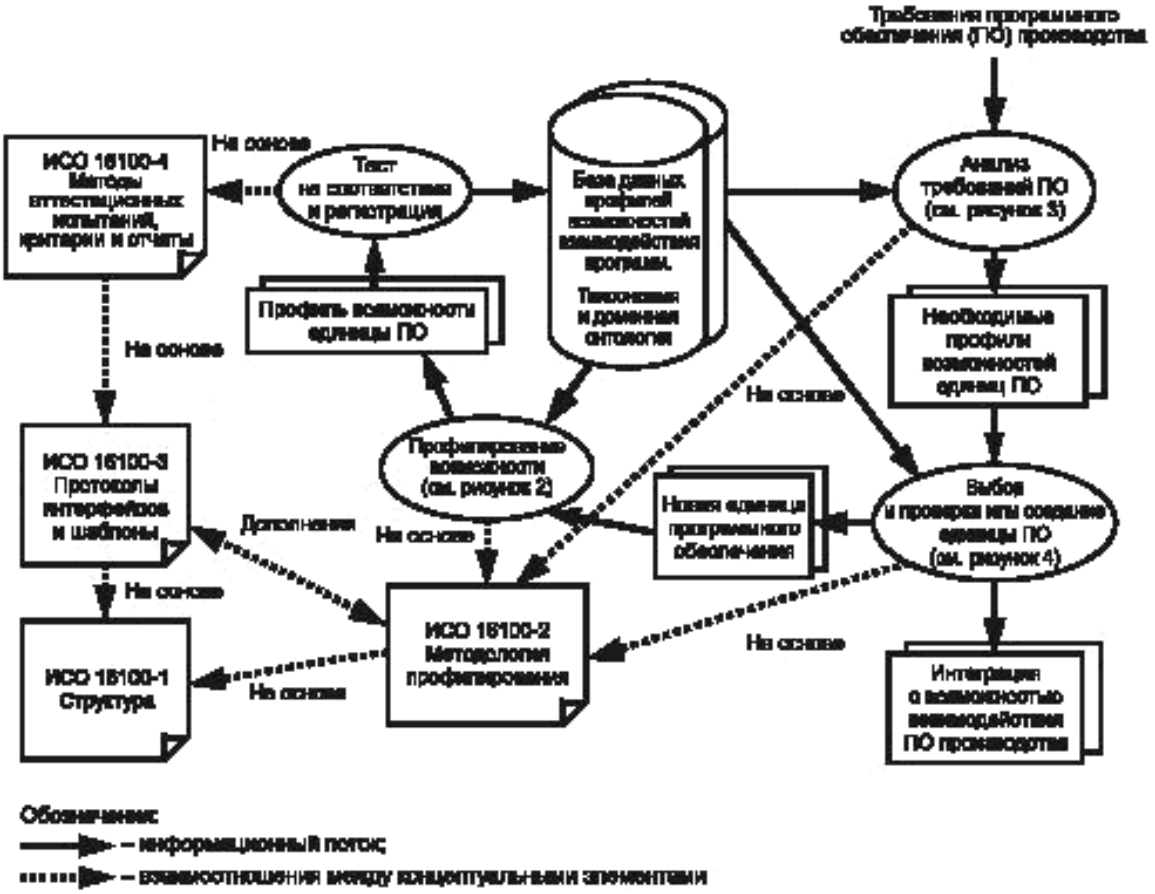

Рисунок 1 — Концепция профиля возможности интероперабельности программного обеспечения

Интероперабельность программных продуктов разных поставщиков описывают с точки зрения их возможностей, которые ассоциированы с аспектами функциональности, интерфейса и структуры. Эти аспекты на основе структуры модели системы специфического применения домена, приведенной в ИСО 16100-1, определены в разделах 5 и 6 настоящего стандарта, а подробное описание приведено в ИСО 16100-3.

Производственный процесс имеет структуру, которая является как вложенной, так и иерархической. На каждом уровне производственного процесса требования к производственному программному обеспечению могут быть смоделированы в виде совокупности классов возможностей, организованных в подобную структуру. Требования к производственному программному обеспечению удовлетворяют путем интеграции в него нескольких единиц производственного программного обеспечения.

Требования производственного программного обеспечения должны быть выражены в показателях профилей возможностей программных единиц. Для профилирования единиц программного обеспечения необходимо подготовить краткое заявление (декларацию) о производственных возможностях, обеспечиваемых единицей программного обеспечения в показателях выполняемых функций, предоставленных интерфейсов и поддерживаемых протоколов в соответствии с требованиями производства.

Методология профилирования возможности должна быть определена в соответствии с требованиями раздела 6 путем использования атрибутов, относящихся к специальному домену, и методов, ассоциированных с каждой специфической единицей программного обеспечения для описания профилей возможностей программных единиц в показателях имени, производственных функций и других необходимых свойств класса.

Требуемые профили сравнивают с уже существующими в базе данных. При совпадении профилей профилируемая единица программного обеспечения должна быть рассмотрена на возможность интеграции. Если профили не совпадают, то новая единица программного обеспечения с требуемыми возможностями должна быть сначала разработана, а затем профилирована и зарегистрирована в базе данных профилей возможностей.

Определение профиля возможностей единиц программного обеспечения должно быть зарегистрировано в соответствующей базе данных после проведения аттестационного испытания в соответствии с методом его проведения с использованием абстрактных испытательных комплектов, указанных в ИСО 16100-4.

В базе данных профиля должен быть приведен набор таксономий, используемый для описания профилей возможностей.

5.2 Процесс профилирования возможности

Часть концепции профиля возможности интероперабельности программного обеспечения (см. рисунок 1), имеющая отношение к процессу профилирования возможности, представлена на рисунке 2.

Единицу программного обеспечения, подлежащую профилированию, анализируют в рамках возможных поддерживаемых вариантов структуры класса возможностей. Описание концепции этой структуры приведено в 6.2.1 настоящего стандарта, а сама структура определена в ИСО 16100-3.

В этом случае возможные варианты должны быть использованы для поиска совпадающих шаблонов в базе данных. При обнаружении совпадающего шаблона его поля заполняют с целью создания профиля. Если совпадающий профиль не найден, то новый шаблон должен быть сформирован с помощью набора классов возможностей.

Рисунок 2 — Процесс профилирования возможности интероперабельности

5.3 Процесс анализа требований программного обеспечения

Часть концепции профиля возможности интероперабельности программного обеспечения (см. рисунок 1), имеющая отношение к процессу анализа требований программного обеспечения, представлена на рисунке 3.

Профили возможностей каждой единицы производственного программного обеспечения являются производными от требований производственного программного обеспечения в процессе их анализа. Сначала требования производственного программного обеспечения разбивают на несколько простых требований, которые выполняются классами возможностей, отобранными из базы данных. В случае, если шаблон, соответствующий классу, существует, то его заполняют специфическими требованиями для создания необходимого профиля возможностей. В случае, если такого шаблона нет, то новый шаблон создают с применением правил создания шаблона, приведенных в 6.3.

Рисунок 3 — Процесс анализа требований программного обеспечения

5.4 Выбор и проверка единицы программного обеспечения или процесс ее создания

Часть интеграции профиля возможности (см. рисунок 1), имеющая отношение к выбору и проверке (верификации) единицы программного обеспечения или к процессу ее создания, представлена на рисунке 4.

Рисунок 4 — Процесс выбора, проверки или создания единицы программного обеспечения

Для каждого профиля возможности выполняют поиск совпадающих профилей возможностей, которые определяют существующее программное обеспечение. Поиск совпадений проводят в соответствии с правилами, приведенными в 6.6. Если совпадение существует, то единицу программного обеспечения добавляют в список кандидатов. При отсутствии совпадений необходимо:

a) разработать новую единицу программного обеспечения, соответствующую требованиям профиля;

b) разложить требуемый профиль на комбинацию нескольких профилей;

c) пересмотреть требования, предъявляемые к существующим профилям.

Профиль новой единицы программного обеспечения должен быть зарегистрирован в базе данных в соответствии с процессом профилирования, указанным в 5.2. Выбранные единицы программного обеспечения верифицируют на соответствие требованиям производственного программного обеспечения согласно критериям возможности интероперабельности программных средств.

6 Элементы и правила профилирования возможности

6.1 Таксономия

Ключевым аспектом таксономии профилей возможностей является их способность идентифицировать содержание, которое составляет определение класса возможности. Таксономия должна обеспечивать необходимые средства для взаимного обмена информацией о возможности.

Таксономия должна описывать частичную совокупность действий, выполняемых в пределах жизненного цикла производственных предприятий. При необходимости добавления в таксономию новой деятельности класс возможностей, ассоциированный с этой деятельностью, должен соответствовать требованиям 6.2.

6.2 Классы возможностей и их содержание

6.2.1 Содержание класса возможности единицы программного обеспечения

Возможность единицы производственного программного обеспечения должна быть выражена в показателях классов возможностей. Эти классы являются производными от производственных действий, указанных в ИСО 16100-1, рисунок 4. Классы определяют производственную функцию, ресурс и информацию, обрабатываемую единицей производственного программного обеспечения в соответствии с требованиями производственного процесса.

Содержание класса возможностей единицы программного обеспечения должно включать в себя следующее:

a) тип производственного домена;

b) тип производственной деятельности, являющейся частью процесса, ресурсы, необходимые для осуществления этой деятельности, и типы информации, которой обмениваются на протяжении деятельности;

c) тип вычислительной системы, соответствующий рабочему окружению, архитектуру программного обеспечения и использованную модель проектирования;

d) тип сервисов, протокол и типы данных, необходимые для запуска программного обеспечения;

e) наименование поставщика, версию программы и историю ее изменений;

f) выполнение контрольных задач;

g) индексы надежности;

h) сервисную и поддерживающую политику;

i) пользовательское соглашение и ценовую политику.

Описание дополнительных требований к содержанию класса возможностей приведено в ИСО 16100-3.

Концептуальная структура класса возможностей приведена на рисунке 5.

Имя класса
Атрибуты: Тип производственного домена Тип производственной деятельности Список типов ресурсов Список типов информации Список дополнительных возможностей Тип функции
Методы Список поддерживаемых сервисов (см. ИСО 16100-3)

Рисунок 5 — Концептуальная структура класса возможностей

6.2.2 Домен применения на производстве

6.2.2.1 Модель деятельности по внедрению в производство

В модели деятельности по внедрению в производство специфического домена и трех ее ассоциативных моделях информации, процессов и ресурсов, приведенных на рисунке 4 ИСО 16100-1, учтены общепринятые требования и структура интероперабельности программ для всех единиц программного обеспечения, предложенных в качестве примера основных эксплуатационных характеристик и возможной структуры.

Так как предложенная единица программного обеспечения может охватывать не все части приложения, целевая часть модели должна быть выделена и отмечена с применением зарегистрированной таксономии, последовательности и номера уровня моделей производственной деятельности.

Пример — Пример соответствует приведенному в ИСО 16100-1, приложение B:

(A) Производственная деятельность

(AA) Разработка изделия

(AA1) Проектирование изделия

(AA11) Разработка концептуального проекта

(AA111) Определение функции и ограничений изделия

(AA112) Описание поведения изделия

(AA113) Разбиение на части ограничений функции и поведения

(AA114) Задание конфигурации изделия

(AA12) Разработка рабочего проекта

(AA121) Проектирование системы/компонента

(AA122) Анализ системы/компонента

(AA123) Оценка проекта системы/компонента
(AA124) Оптимизация проекта
(AA125) Завершение проекта системы/компонента
(AA126) Разработка сборочных чертежей
(AA2) Технологический процесс
(AA21) Разработка концептуального плана процесса
(AA211) Выбор производственного процесса
(AA212) Выбор производственных ресурсов
(AA2121) Выбор станков
(AA2122) Выбор инструментов/зажимных приспособлений
(AA2123) Выбор квалифицированных рабочих
(AA213) Оценка производственных расходов/времени
(AA22) Разработка подробного плана технологического процесса
(AA221) Формирование последовательности процесса
(AA222) Создание операции
(AA2221) Определение возможности промежуточной механической обработки
(AA2222) Спецификация установки компонентов изделия и ресурсов механической обработки
(AA2223) Вычисление допуска промежуточной механической обработки
(AA2224) Разработка инструкции по механической обработке
(AA223) Определение параметров производства
(AA224) Создание программы управления
(AA2241) Создание траектории перемещения инструментов
(AA2242) Задание параметров управления процессом
(AA2243) Создание программы управления станком
(AA225) Создание маршрута перемещения в цехе
(AA2251) Определение конфигурации цеха
(AA2252) Определение средств транспортирования
(AA2253) Определение согласованности по времени
(AA3) Планирование ресурсов предприятия
(AA4) Закупка ресурсов
(AA5) Выполнение производственных заказов
(AA51) Разработка последовательности операций и подробного графика
(AA52) Отправка изделий производства
(AA53) Прокладка маршрута изделий производства и ресурсов
(AA54) Руководство данными/документами цеха
(AA6) Управление оборудованием и технологическим процессом

6.2.2.2 Модель производственного процесса и ее профиль

Класс модели производственного процесса должен включать в себя модель требуемых специфических действий приложения с соответствующей таксономией и номером индекса (см. 6.2.2.1) в системе с особыми ресурсами и информационным потоком.

В модели деятельности с помощью языка моделирования, например IDEF0, описывают требования и поток данных прикладной системы.

В модели процесса определены автоматизированные функции с выбранными ресурсами и информационный поток между ними. Модель процесса должна иметь наименование и метку в соответствии с зарегистрированной таксономией, последовательностью и номерами уровней связанных (целевых) моделей деятельности.

Каждая модель ресурсов должна быть представлена в виде соответствующего профиля.

6.2.2.3 Модель производственных ресурсов и ее профиль

Модель производственных ресурсов должна включать в себя выбранные ресурсы, например устройства, оборудование, коммуникационные сети, людей и материалы. Ресурсы используют в модели технологического процесса для выполнения требований информационной модели, определяющей информационный поток между ресурсами.

Информационная модель должна иметь наименование и метку в соответствии с зарегистрированной таксономией, последовательностью и номерами уровней связанных (целевых) моделей деятельности.

Каждая модель ресурсов должна быть представлена в виде соответствующего профиля.

6.2.2.4 Информационная модель производства и ее профиль

Информационная модель производства представляет собой типы данных событий, а также данные, которыми обмениваются между собой ресурсы в модели процесса, определяющей специфическую область деятельности в модели действия прикладных программ.

Информационная модель должна иметь наименование и метку в соответствии с зарегистрированной таксономией, последовательностью и номерами уровней связанных (целевых) моделей деятельности.

Каждая модель ресурсов должна быть представлена в виде соответствующего профиля.

6.2.3 Вычислительная модель и ее ассоциативный класс

Вычислительная модель представляет собой картографическое распределение модели процесса, модели ресурсов и информационной модели, описания которых приведены в 6.2.2.

6.2.3.1 Представление класса единицы программного обеспечения

6.2.3.1.1 Имя класса

Информационная модель должна иметь наименование и метку в соответствии с зарегистрированной таксономией, номером последовательности и уровнем связанных (целевых) моделей деятельности.

6.2.3.1.2 Атрибуты класса

Атрибуты класса должны быть указаны вместе с типом данных и возможностью внешнего доступа.

6.2.3.1.3 Операции класса

Операции класса должны быть указаны с сигнатурой и возможностью внешнего сервиса.

Единица программного обеспечения может включать в себя множество последующих классов. В этом случае должны быть указаны все включенные классы.

6.2.3.2 Архитектура программного обеспечения, используемый класс шаблонов программного обеспечения для проектирования

Характерным свойством архитектуры программного обеспечения, а также шаблона для проектирования программного обеспечения является использование структуры единицы программного обеспечения и необходимость указания роли, которую выполняет это программное обеспечение.

Образцы проектирования архитектуры и примеры[1)] их структуры и ролей:

a) Архитектура иерархического представления

Пример — Структура: прикладные программы, которые могут быть разделены на группы подзадач, в которых каждая группа находится на определенном уровне абстракции. Роль: логический объект на уровне N, предоставляющий услугу для логического объекта на уровне N+1.

b) Посредническая архитектура

Пример — Структура: распределенные системы программного обеспечения с несвязанными компонентами, которые взаимодействуют путем вызова удаленного сервиса. Роль: клиенты, серверы, брокеры, мосты, программы — посредники на стороне клиента, программы — посредники на стороне сервера.

c) Архитектура «Модель — представление — контроллер»

Пример — Структура: модель содержит функциональные возможности ядра операционной системы и данные, контролирует информацию, выводимую на дисплей для пользователя, управляет вводом абстрактного идентификатора пользователя. Роль: модель, наблюдатель, представление, контроллер.

d) Схема «Master — Slave»

Пример — Структура: главный компонент распределяет работу между идентичными подчиненными компонентами и вычисляет конечный результат по результатам, возвращаемым этими подчиненными. Роль: клиент, главный, 1-й подчиненный, 2-й подчиненный,..., n-й подчиненный.

e) Программа-посредник «Proxy»

Пример – Структура: позволяет клиентам компонента поддерживать связь с представителем, а не с самим компонентом. Роль: клиент, посредник, оригинал.

---

1) F. Buschmann et al «Pattern Oriented Software Architecture», John Wiley & Sons, june 2000.

f) Издатель — подписчик

Пример — Структура: один издатель уведомляет любое число подписчиков об изменениях его состояния. Роль: издатель, подписчик.

6.2.3.3 Класс сервиса или протокола

Интерфейсы единицы программного обеспечения должны иметь описание, представленное в качестве сервиса (например, в архитектуре иерархического представления логический объект на уровне *N* должен обслуживать объект на уровне *N* + 1) или в виде протокола с указанием его типа данных (например, в архитектуре «Клиент — сервер» должны быть указаны интерфейсы клиентов со специальным протоколом доступа к серверу).

6.2.4 Нефункциональные свойства единицы программного обеспечения

В настоящем пункте приведены свойства единицы программного обеспечения нефункционального представления программы, не соответствующие требованиям 6.2.2 и 6.2.3.

6.2.4.1 Поставщик, версия и история программного продукта

Профиль возможности единицы программного обеспечения должен включать в себя наименование поставщика, его контактный адрес, а также новейшую версию программы и историю ее пересмотра.

6.2.4.2 Вычислительные средства, планируемые для применения

Профиль возможности единицы программного обеспечения должен включать в себя следующие данные:

a) процессор — тип процессора является важной характеристикой, но также следует принимать во внимание его производительность, так как слабая производительность может серьезно препятствовать эффективной и своевременной работе программного обеспечения;

b) операционная система и требуемые опции — подходящая операционная система и необходимая рабочая версия для работы программной части (компонента). Любое свойство, обеспечивающее поддерживание совместимости снизу вверх, может быть включено в эту информацию;

c) язык — источник языка для компонента программного обеспечения, включая версии редактора, компилятор, редактор связей и отладчик, используемый для создания программной части. Любое свойство, обеспечивающее поддерживание совместимости снизу вверх, может быть включено в эту информацию;

d) память для запуска программы — тип и объем памяти, необходимые для работы программной части вместе с любой другой поддержкой запуска программы;

e) пространство на диске — тип и объем информационных средств, необходимых для хранения форм — источников исполняемой программной части. Эта информация должна включать в себя хранение данных, например запоминающее устройство на диске, необходимое для операционных переменных, результатов обработки и механизмов восстановления после повреждений;

f) многопользовательская поддержка — способность программного обеспечения одновременно взаимодействовать со многими пользователями, клиентами или абонентами;

g) удаленный доступ — способность программного обеспечения, а также любых других компонентов программного обеспечения, которые загружаются по каналу связи, поддерживать удаленный доступ, управление и контроль;

h) дополнительные средства и сменные платы — расширения программного обеспечения, необходимые для временного поддерживания работы программного обеспечения в течение периода его выполнения, например интерпретацию импортированных изображений или фильтрацию входящего потока данных в другом формате.

6.2.4.3 Измеренная рабочая характеристика программного изделия

Профиль возможности единицы программного обеспечения должен включать в себя следующие рабочие характеристики специального вычислительного средства, необходимого для использования программного изделия с жестким временным режимом в реальном времени:

a) время работы (время выполнения) с особыми входными данными и ограничения к ним (основные данные функционирования);

b) число специальных транзакций в единицу времени (объединенные данные функционирования).

6.2.4.4 Данные надежности программного изделия

Профиль возможности интероперабельности единиц программного обеспечения должен содержать историю применения, число поставок программного изделия, его планируемый безопасный уровень целостности и информацию о том, был ли этот уровень установлен самостоятельно или третьей стороной.

6.2.4.5 Компетенция

Профиль возможности интероперабельности единицы программного обеспечения должен включать в себя информацию о компетенции каждого поставщика, например его полномочия, обязательства и политику применения единицы программного обеспечения, лицензионные требования, внутреннюю политику предприятия и требования к обучению операторов.

6.2.4.6 Данные о цене

Профиль возможности интероперабельности единицы программного обеспечения должен включать в себя данные о начальных и текущих расходах на программное изделие.

6.3 Шаблоны возможностей и правила

Единица программного обеспечения, обеспечивающая или поддерживающая деятельность с ассоциированным классом возможностей, должна быть кратко описана в шаблоне возможности. Структура шаблона возможности интероперабельности программного обеспечения должна быть аналогична структуре класса производственных возможностей.

Примечание — В иерархической структуре шаблон возможности ассоциируется с каждой возможностью, определенной на каждом уровне структуры. Во вложенной структуре подобная ассоциация существует между каждым классом возможностей и шаблоном на каждом уровне структуры.

Пример концептуальной структуры шаблона приведен на рисунке 6. Концептуальная структура шаблона должна состоять из части, общей для всех шаблонов, и части, специфичной для класса возможностей. Формальная структура шаблона определена в ИСО 16100-3.

Common part	Общая часть
Template ID	Идентификация шаблона
Capability Class Name	Имя класса возможностей
Software Unit ID	Идентификация единицы ПО
Vendor Name	Наименование поставщика
Version Number & History	Номер версии и история
Computing Facilities Required	Требуемые вычислительные средства
Processor	Процессор
OperatingSystem & Options	Операционные системы и опции
Language	Язык
Runtime Memory	Запоминающее устройство времени прогона программы
Disk Space	Пространство на диске
MultiUserSupport	Многопользовательская поддержка
RemoteAccess	Дистанционная выборка
AddUns&PlugIns	Дополнения и сменные платы
Measured Performance of the Unit	Измеренная характеристика единицы ПО
ElapsedTime	Время работы (истекшее время)
NumberOfTransactionsPerUnitTime	Число транзакций в единицу времени
Reliability Data of the Unit	Данные о надежности единицы ПО
UsageHistory	История применения
NumberOfShipments	Число поставок
IntendedSafetyIntegrityLevel	Планируемый безопасный уровень целостности
Certification Body	Орган по сертификации
Support Policy	Политика поддержки
Price Data	Данные о цене
Reference Dictionary Name	Название словаря-справочника
NumberOfMethods	Число методов
...	...
Part Specific to Capability Class	Часть, специальная для класса возможностей
....	...

Рисунок 6 — Пример структуры шаблона

В общую часть должны быть включены следующие элементы:

a) идентификация шаблона — опознаватель шаблона объекта;

b) идентификация единицы программного обеспечения — опознаватель единицы производственного программного обеспечения, обеспечивающей выполнение производственной функции;

c) имя справочного словаря — название словаря, содержащего определения классов возможностей;

d) имя класса возможностей — название класса эталонных возможностей;

e) число атрибутов профилей — число атрибутов, унаследованных из соответствующего класса возможностей;

f) число методов — число атрибутов, предусмотренных соответствующим классом возможностей;

g) количество ресурсов — количество ресурсов, необходимых для программной среды;

h) число ограничений — число условий, необходимых для выполнения единицы программного обеспечения;

i) число расширений — число аспектов других единиц программного обеспечения, предоставленных поставщиком;

j) число нижних уровней — число уровней вложенной структуры или самый нижний уровень иерархии структуры класса эталонных возможностей;

k) число субшаблонов на следующем нижнем уровне — число шаблонов, связанных с подвозможностями, составляющими целевую способность, ассоциированную с шаблоном, расположенным на один уровень ниже текущего уровня в иерархии или вложенной структуры.

В соответствующую часть для класса возможностей должны быть включены следующие перечни:

a) атрибутов;

b) методов;

c) ресурсов, например тип операционной системы;

в) ограничений, например тип архитектуры, образец проекта;

e) расширений;

f) нижних уровней;

g) субшаблонов.

Шаблоны способностей должны быть указаны с использованием условных обозначений XML для создания схем XML (см. REC-xmlschema-1-20010502 и REC-xmlschema-2-20010502). Взаимоотношения между шаблонами возможностей должны быть указаны с помощью условных обозначений на языке XML для трансформации схем XML и файлов XML. Если класс возможностей указан в шаблоне и на его основе будут создаваться экземпляры, то созданный экземпляр класса представляет собой объект. Два шаблона возможностей являются идентичными, если их соответствующие атрибуты и операции идентичны. Если атрибуты одного шаблона являются подмножеством атрибутов другого шаблона и операции одного шаблона являются подмножеством операций другого шаблона, то два шаблона возможностей считают совместимыми и совпадающими.

6.4 Профили возможностей и правила

Профили возможностей являются шаблонами возможностей, если они содержат, как минимум, экземпляр имени единицы профильного программного обеспечения. Другие элементы заполняют в соответствии с заданным уровнем спецификаций.

Профили возможностей взаимодействий должны быть определены с помощью условных обозначений на языке XML для создания файлов на языке XML. Взаимоотношения между профилями возможностей должны быть обозначены с использованием условных обозначений на языке XML для трансформации файлов на языке XML. Если на шаблон возможности есть ссылки в профиле возможности и данный шаблон заполнен, то заполненный шаблон представляет собой объект профиля.

6.5 База данных профиля единицы программного обеспечения

В базах данных хранятся следующие элементы, отличающиеся словарными именами и описанные в 6.1—6.4:

a) таксономии;

b) классы возможностей;

c) шаблоны возможностей;

d) профили возможностей.

Базы данных могут быть структурированы в качестве свободной комбинации четырех вышеуказанных элементов для предоставления необходимых сервисов.

Таксономия должна быть уникальной в таксономическом словаре. Класс возможностей должен быть уникальным в словаре классов. Шаблон возможности должен быть уникальным в словаре шаблонов. Профиль возможности должен быть уникальным в словаре профилей.

6.6 Правила приведения профилей в соответствие

Совпадение профилей возможностей используется в следующих процессах:

a) анализ единицы программного обеспечения в процессе профилирования возможности (рисунок 2);

b) декомпозиция требований в процессе анализа требований программного обеспечения (рисунок 3);

c) поиск базы данных для каждого профиля в процессе выбора и верификации или создания единицы программного обеспечения (рисунок 4).

В этих процессах и профилях возможностей была предпринята попытка найти совпадения описаний единиц программного обеспечения, требований производственного программного обеспечения или требуемых профилей возможностей единиц программного обеспечения. Единицу программного обеспечения и требования программного обеспечения производства следует описывать с помощью условных обозначений на языке XML путем ссылки на таксономию, используемую в базе данных, содержимого классов возможностей и существующих шаблонов. Описания созданных единиц программного обеспечения, требований программного обеспечения производства или требуемых профилей способностей единиц программного обеспечения сравнивают с содержимым классов способностей в базе данных по 6.2. Если найдено, по меньшей мере, частичное совпадение, то описание класса возможностей в базе данных является результатом этого совпадения. Поиск шаблонов в базе данных осуществляют путем использования контента на выходе классов возможностей.

6.7 Критерии интероперабельности

Критерии интероперабельности программных изделий разных поставщиков используют для выполнения процесса верификации по 5.4.

7 Соответствие

Требования к соответствию установлены в ИСО 16100-4.

Приложение А
(справочное)

Методы ссылок

А.1 Язык XML

Язык расширенной (гипертекстовой) разметки XML обладает свойствами, которые могут быть использованы прямо или косвенно при профилировании возможности программного обеспечения. Язык XML является языком, используемым для выражения лексических элементов документа, представляемого в виде ориентированного графа, в частности, для размещения на web-узле. Лексические элементы могут быть определены пользователем. Язык XML является практическим подмножеством стандартного обобщенного языка разметки SGML (Standard Generalized Markup Language) и обеспечивает разметку страницы тегами подобно языку HTML. Любой документ на языке XML может быть также верифицирован на достоверность XML. Для использования языка XML в профилировании возможности программного обеспечения следует отметить, что XML имеет пространство присваиваемых имен (XML Namespaces) для согласования или регистрации пространства имен.

А.2 Словари, определения и форматы обмена для комплектов программного обеспечения: описание открытого программного обеспечения и формата определения канала

Описание открытого программного обеспечения[21] (OSD) представляет собой словарь, созданный на основе языка XML для описания комплектов программного обеспечения и их зависимости друг от друга. OSD используют в окружающих средах распределения программного обеспечения либо путем скачивания, инициированного пользователем, либо методом автоматической рассылки. OSD может использоваться для распределения программного обеспечения на Web-узле в одном из двух вышеуказанных методов.

Распределение программного обеспечения методом скачивания требует участия пользователя для нахождения, загрузки и обновления программного обеспечения. Несмотря на то что OSD облегчает автоматизацию загрузки и инсталляции требуемых компонентов программного обеспечения, пользователи Web-узла должны просматривать страницу HTML, которая инициирует процесс инсталляции программного обеспечения. Тег «OBJECT» из спецификации HTML 4.0 используется для рекламы нового программного обеспечения в Web. При обнаружении тега объекта OBJECT в ресурсе OSD агент пользователя, осведомленный об OSD, может автоматически загружать и обновлять необходимые компоненты программного обеспечения.

Формат определения канала CDF также создан на основе языка XML и предоставляет словарь метаданных, необходимых для описания взаимоотношений между страницами HTML и другими web-ресурсами. Клиенты, осведомленные о CDF, могут использовать методы интеллектуального опроса (smart-pull) для автоматической загрузки web-содержания (контента), а серверы, осведомленные о CDF, могут реализовать механизмы web-вещания (true — push) для автоматической передачи содержания (контента) от клиента к серверу. Таким образом, CDF предоставляет язык передачи содержания (push), тем самым обеспечивая идеальное средство для приведения в действие принудительной рассылки (push) или автоматического распределения программного обеспечения. Для того чтобы CDF активировал программное обеспечение (push), файл CDF должен содержать ссылки на комплекты программного обеспечения на основе OSD.

OSD включает в себя обширный словарь, с помощью которого приводят описания элементов программного обеспечения. Это описание включает в себя следующее:

SOFTPKG:	определяет общий пакет программного обеспечения;
IMPLEMENTATION:	используется для описания реализации пакета программного обеспечения;
DEPENDANCY:	указывает зависимости между распределениями программного обеспечения или его компонентов;
TITLE:	представляет заголовок или интуитивно понятное «дружественное» имя пакета программ;
ABSTRACT:	представляет краткое описание, резюмирующее характер и цель распределения программного обеспечения;
LICENSE:	определяет местоположение лицензионного соглашения или уведомления об авторском праве;
CODEBASE:	определяет местоположение архива распространения программного обеспечения;
OS:	указывает требуемую операционную систему;
OSVERSION:	определяет необходимую версию операционной системы;
PROCESSOR:	определяет необходимый естественный язык пользовательского интерфейса;
LANGUAGE:	представляет заголовок или интуитивно понятное «дружественное» имя пакета программного обеспечения;

[21] Van Hoff и др., 1997.

VM: определяет требуемую виртуальную вычислительную машину;
MEMSIZE: определяет необходимый объем памяти времени прогона программы;
DISKSIZE: определяет необходимый объем пространства на диске;
IMPLTYPE: определяет тип реализации.

А.3 Сервисы распространения программного обеспечения: распределенная обработка в открытой системе и технология построения распределенных объектных приложений

Эталонная модель распределения обработки в открытой системе ODP характеризует важные атрибуты информационной системы, для которой ограничения могут быть установлены для того, чтобы эта система была открытой и распределенной. Эталонная модель является основой серии стандартов, устанавливающих эти требования.

Рабочая группа по развитию стандартов объектного программирования (OMG) является некоммерческим консорциумом поставщиков программных изделий, разработчиков-программистов и конечных пользователей. Консорциум был образован в мае 1989 г. Его работа направлена на поиск архитектурной структуры для распределенных, ориентированных на объект приложений на основе широкодоступных спецификаций интерфейсов, например тех, которые предоставлены ODP. Архитектура объектного программирования (OMA) является основой деятельности рабочей группы OMG и состоит из эталонной модели (опубликованной в 1992 г.), которая идентифицирует и характеризует компоненты, интерфейсы и протоколы, составляющие OMA, но не предоставляет подробных деталей.

Технология построения распределенных объектных приложений CORBA, разработанная рабочей группой OMG, является архитектурой и протоколом для динамически связанных распределенных объектов. Эта связь является только синтаксической и лексической с ограниченными возможностями языка описания интерфейсов (IDL).

Эталонная модель включает в себя пять компонентов:

a) брокер (программа-посредник) запроса объекта предоставляет инфраструктуру, позволяющую объектам осуществлять преобразование, независимое от специальных платформ и технологий, используемых для реализации объекта;

b) объектные сервисы стандартизируют менеджмент жизненного цикла объектов. Интерфейсы предоставляют для создания объектов, а также для управления доступом к объектам и прослеживания перемещения объектов. Объектные сервисы предусматривают взаимосвязь приложений;

c) общие средства предоставляют набор родовидовых функций приложения, которые могут быть сконфигурированы в специфические требования частной конфигурации. Эти средства более легко распознаются конечным пользователем, например для печати и электронной почты;

d) доменные интерфейсы представляют вертикальные области, обеспечивающие функциональность и непосредственную заинтересованность конечных пользователей в доменных приложениях, например таких, как производство, финансы и здравоохранение;

e) экземпляры не являются частью деятельности OMG в области стандартизации, но в OMG признают, что эти вопросы являются очень важными для разработки успешных приложений.

Методы OMG определяют интерфейсы для распределенных объектов с помощью языка описания интерфейсов (IDL). Рабочая группа OMG выявила необходимость использования семантически богатого описательного языка. В результате этого планируется создание запросов на предложения (RFP), которые охватывают деловой объектный домен и, в частности, обращают внимание на необходимость использования семантически обогащенного языка. Рабочая группа OMG начала стандартизацию ряда деловых объектов, например, в области валютных операций, что в перспективе может вывести на поставщиков, создающих программное обеспечение с использованием некоторых основных стандартизованных объектов, и облегчить возможную интероперабельность программных изделий разных поставщиков.

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации

Таблица ДА.1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО 16100-1:2009	—	*
ИСО 16100-2:2003	—	*
ИСО 16100-3:2005	—	*
ИСО 16100-4:2006	—	*
ИСО 16100-5:2009	—	*
ИСО 16100-6:2011	—	*

* Соответствующий национальный стандарт находится в стадии разработки. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта. Перевод данного международного стандарта находится в Федеральном информационном фонде технических регламентов и стандартов.

Библиография

[1] ИСО/МЭК ТО 10000-1:1998 (ISO/IEC TR 10000-1:1998) Информационные технологии. Структура и таксономия международных стандартизованных профилей. Часть 1. Общие принципы и структура документации (Information technology — Framework and taxonomy of International Standardized Profiles — Part 1: General principles and documentation framework)

[2] ИСО/МЭК 10746-2:1996 (ISO/IEC 10746-2:1996) Информационные технологии. Распределенная обработка в открытой системе. Эталонная модель. Принципы (Information technology — Open Distributed Processing — Reference Model: Foundations)

[3] ИСО 18629-1 (ISO 18629-1) Системы промышленной автоматизации и интеграция. Язык спецификации процесса. Часть 1. Обзор и основные принципы (Industrial automation systems and integration — Process specification language — Part 1: Overview and basic principles)

[4] ИСО/МЭК 19501-1 (ISO/IEC 19501-1) Информационные технологии. Унифицированный язык моделирования (UML). Часть 1. Спецификация (Information technology — Unified Modeling Language (UML) — Part 1: Specification)

[5] ИИЭР 1320-1:1998 (IEEE 1320-1:1998) Стандарт языка функционального моделирования. Синтаксис и семантика IDEF0 (Standard for Functional Modeling Language — Syntax and Semantics for IDEF0)

[6] REC-xml-20001006 Расширяемая спецификация языка XML 1.0, второе издание — W3C рекомендации от 6 октября 2000 г. (REC-xml-20001006, Extensible Markup Language (XML) 1.0 Second Edition — W3C Recommendation 6 October 2000)

УДК 658.52.011.56:006.354 ОКС 25.040.01 Т58

Ключевые слова: автоматизированные промышленные системы, интеграция, жизненный цикл систем, управление производством

Редактор *Т.А. Леонова*
Технический редактор *В.Н. Прусакова*
Корректор *В.И. Варенцова*
Компьютерная верстка *В.И. Грищенко*

Сдано в набор 20.02.2014. Подписано в печать 27.03.2014. Формат 60x84 1/8. Гарнитура Ариал. Усл. печ. л. 2,32. Уч.-изд. л. 2,08. Тираж 78 экз. Зак. 380.

Издано и отпечатано во ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru