ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 13374-2— 2011

Контроль состояния и диагностика машин

ОБРАБОТКА, ПЕРЕДАЧА И ПРЕДСТАВЛЕНИЕ ДАННЫХ

Часть 2

Обработка данных

ISO 13374-2:2007
Condition monitoring and diagnostics of machines — Data processing, communication and presentation — Part 2: Data processing
(IDT)

Издание официальное

Москва
Стандартинформ
2012

Предисловие

Цели и принципы стандартизации в Российской Федерации установлены Федеральным законом от 27 декабря 2002 г. № 184-ФЗ «О техническом регулировании», а правила применения национальных стандартов Российской Федерации — ГОСТ Р 1.0—2004 «Стандартизация в Российской Федерации. Основные положения»

Сведения о стандарте

1 ПОДГОТОВЛЕН Автономной некоммерческой организацией «Научно-исследовательский центр контроля и диагностики технических систем» (АНО «НИЦ КД») на основе собственного аутентичного перевода на русский язык международного стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 183 «Вибрация, удар и контроль технического состояния»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 16 ноября 2011 г. № 550-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 13374-2:2007 «Контроль состояния и диагностика машин. Обработка, передача и представление данных. Часть 2. Обработка данных» (ISO 13374-2:2007 «Condition monitoring and diagnostics of machines — Data processing, communication and presentation — Part 2: Data processing»).

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

5 ВВЕДЕН ВПЕРВЫЕ

Информация об изменениях к настоящему стандарту публикуется в ежегодно издаваемом указателе «Национальные стандарты», а текст изменений и поправок — в ежемесячно издаваемых информационных указателях «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ежемесячно издаваемом информационном указателе «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

Содержание

Введение

Разнообразные реализации программного обеспечения, предназначенного для работы с данными в программах контроля состояния и диагностики машин, зачастую не обеспечивают простых и удобных способов обмена данными, сложны в установке и требуют больших усилий по их интегрированию в системы мониторинга. Это затрудняет выработку у пользователей единого взгляда на процедуры мониторинга. Настоящий стандарт устанавливает общие требования к спецификации открытого программного обеспечения, применяемого в целях контроля состояния, в части обработки данных безотносительно к используемым операционным средам и аппаратным средствам.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Контроль состояния и диагностика машин

ОБРАБОТКА, ПЕРЕДАЧА И ПРЕДСТАВЛЕНИЕ ДАННЫХ

Часть 2

Обработка данных

Condition monitoring and diagnostics of machines. Data processing, communication and presentation.
Part 2. Data processing

Дата введения — 2012—12—01

1 Область применения

Настоящий стандарт устанавливает требования к информационной модели и к модели обработки информации, которым должна соответствовать открытая архитектура систем контроля состояния и диагностики машин в целях обеспечения их совместимости.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты:

ИСО 13374-1:2003 Контроль состояния и диагностика машин. Обработка, передача и представление данных. Часть 1. Общее руководство (ISO 13374-1:2003, Condition monitoring and diagnostics of machines — Data processing, communication and presentation — Part 1: General guidelines)

ИСО/МЭК 14750:1999 Информационная технология. Взаимосвязь открытых систем. Язык описания интерфейсов (ISO/IEC 14750:1999, Information technology — Open Distributed Processing — Interface Definition Language)

3 Требования к информационной архитектуре систем контроля состояния и диагностики

3.1 Общие положения

Информационная архитектура описывает все объекты и их свойства (атрибуты), типы данных, отношения между объектами и информационные документы для заданной системы или приложения. Открытая спецификация информационной архитектуры системы контроля состояния и диагностики машин должна описывать каждый из пяти уровней, показанных на рисунке 1.

Издание официальное

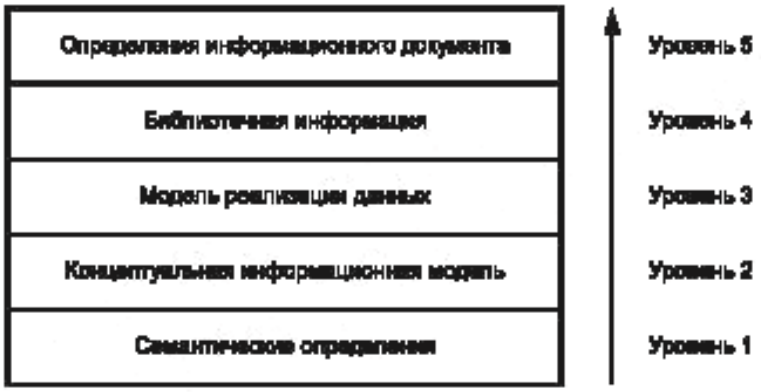

Рисунок 1 — Уровни информационной архитектуры системы контроля состояния и диагностики

3.2 Требования к семантическому определению

Для облегчения понимания между сторонами, использующими информационную архитектуру, ее открытая спецификация должна предоставлять набор семантических определений для всех основных объектов системы. При этом могут быть использованы как неформальные описания, например определения объектов посредством естественного языка, так и формальные описания, использующие онтологические схемы, как, например, предложенный Консорциумом Всемирной паутины [World Wide Web Consortium (W3C)] формат описания ресурсов [Resource Definition Format (RDF)].

3.3 Требования к концептуальной информационной модели

Концептуальная информационная модель — это обобщенное определение всех ключевых объектов, относящихся к системе контроля состояния и диагностики, а также их главных свойств (также называемых «атрибутами»); типов данных; взаимных отношений между объектами и ограничений, накладываемых на объекты и отношения между ними. Открытая спецификация должна предоставлять некоммерческую концептуальную модель данных, также именуемую «схемой данных», абстрагированную от способа хранения и осуществления доступа на физическом уровне. Данная концептуальная информационная схема представляет собой план местоположения различных информационных элементов, служащий для обеспечения системной интеграции и целостности данных. С помощью концептуальной схемы может быть проверена модель реализации данных.

Унифицированный язык моделирования [Unified Modeling Language (UML)], получивший наибольшее распространение среди языков моделирования в области разработки программного обеспечения, в настоящее время стандартизован на международном уровне (см. [19]). UML включает в себя стандартное представление диаграммы классов для информационного моделирования (дополнительная информация об UML содержится в приложении В).

3.4 Требования к модели реализации данных

Модель реализации данных, основанная на концептуальной информационной модели, обеспечивает точное представление сохраняемых или передаваемых информационных элементов. Открытая спецификация информационной архитектуры должна предоставлять открытую модель реализации данных, согласующуюся с концептуальной информационной моделью.

Открытая модель реализации данных для систем контроля состояния и диагностики должна предусматривать возможность объединения множества источников данных о состоянии машин, поддерживать пиринговые базы данных, пользовательские поисковые критерии и использовать стандартизированные временные метки и единицы измерения. Информационная схема должна поддерживать уникальную идентификацию предприятий, их местоположений, баз данных и источников данных, для того чтобы различать информацию, полученную из территориально удаленных источников. Также она должна поддерживать уникальные на уровне системы идентификаторы заводских участков с машинами (местоположения обслуживаемых участков) согласно древовидной иерархии. Для обеспечения наблюдения и слежения за компонентами системы схема должна поддерживать уникальные идентификаторы ресурсов.

Спецификацией схемы должны быть определены принципы работы с информацией о предприятии, местах сбора данных и базами данных, с информацией о производственных участках (сегментах), наблюдаемых машинах и их узлах (ориентации в пространстве, типе привода, способе установки, устройстве сопряжения валов и т. п.), с данными изготовителя (такими как номинальная частота вращения, номинальная мощность, тип и модель машины, подшипников и других узлов), с информацией о точках измерений, источниках данных измерений, преобразователях и их ориентации, единицах измерений, преобразовании полученных данных, порядке сбора данных и выходных сигналах системы мониторинга. Схема должна поддерживать форматы для передачи регистрируемых числовых данных, временных реализаций, Фурье-преобразований, спектров с постоянной относительной шириной полосы, результатов анализа образцов, термографических изображений и двоичных больших объектов. Схема должна поддерживать функцию указания даты (времени) в формате согласно [1].

Данная спецификация может быть определена с использованием различных языков описания. Файловое описание используется научным сообществом в течение многих лет. Оно может быть реализовано в виде текстового или двоичного формата информационных файлов, загружаемых в информационную систему или выгружаемых из нее. Опубликовано полное описание формата записи, содержащее: поля данных в файле; их точное местоположение относительно других полей данных; идентификатор текстового/двоичного поля; точное описание типа данных (вещественное число с плавающей запятой, целое число, символ, символьная строка переменной длины) для каждого поля.

Формат реляционной информационной схемы представляет собой язык описания для реляционных систем управления базами данных. Реляционная модель данных отвечает сути концептуального проектирования и определяет следующие элементы: различные отношения или таблицы, в которых хранится информация; столбцы данных в таблицах; типы данных для каждого столбца (вещественное число с плавающей запятой, целое число, строка и т. д.); указания, может ли столбец содержать только пустые значения; уникальные идентификаторы строк («первичные ключи»). Таблицы могут быть связаны друг с другом через «внешние ключи».

Рекомендуемым языком для описания для информационной схемы является расширяемый язык разметки [Extensible Markup Language (XML)]. Разработка спецификаций XML контролируется рабочими группами W3C. XML — открытый формат, написанный на обобщенном языке разметки SGML (Standard Generalized Markup Language) с применением стандарта [2] для описания структур различных типов электронных документов. Версия спецификации 1.0 была принята W3C в качестве рекомендуемой 10 февраля 1998 года. 3 мая 2001 года W3C ввел схемы XML в качестве рекомендации. Схемы XML определяют разделяемые словари разметок, структуру XML-документов, использующих данные словари, а также предоставляют набор процедур для назначения семантики. Путем привнесения типизации данных в XML схемы увеличивают использование настоящего стандарта разработчиками систем обмена данными. Схемы XML позволяют определить, какие части документа могут быть проверены на применимость схемы или установить части документа, к которым схема может быть применена. Кроме того, поскольку схемы сами по себе также являются XML-документами, они могут обрабатываться обычными средствами редактирования XML (дополнительная информация об XML содержится в приложении В).

Независимо от выбора формата информационной схемы модель реализации данных должна определять минимальный набор элементов, подлежащих включению в схему для соответствия ей. В дополнение к обязательным элементам может быть включен набор необязательных (опциональных) элементов.

3.5 Требования к библиотечной информации

Для эффективного использования модели реализации данных стандартные элементы должны храниться в библиографической базе данных. Открытая спецификация информационной архитектуры системы контроля состояния и диагностики должна предусматривать открытую библиографическую базу данных, согласующуюся с моделью реализации данных. Данная спецификация должна обеспечивать наполнение и поддержку классификаций, характерных для данной отрасли, а также кодов ссылок на данные и позволять добавление отраслевых и пользовательских элементов в библиографическую базу данных, используя уникальные значения из базы данных, как поставщикам, так и конечным пользователям. Также, по необходимости, спецификация должна поддерживать создание стандартных наборов кодов для разных языков.

Библиографическая база данных определяет все таблицы кодов для типов машин и их узлов, значимых объектов, точек измерений, единиц измерений, выборок данных, событий (связанных с процедурами диагностирования и прогнозирования), видов технических состояний, отказов и их основных причин.

3.6 Требования к определениям информационного документа

Открытая спецификация информационной архитектуры систем контроля состояния и диагностики должна также определять формат публикации (печати) информационного документа. Данная спецификация обеспечивает стандартный способ чтения или печати библиотечной информации и поддерживает приложения, необходимые для обеспечения импорта или экспорта этой информации.

Спецификации, использующие для модели реализации данных формата схемы описания файлов, будут, скорее всего, использовать тот же формат для публикации документов в двоичном формате или в формате ASCII. Должно быть опубликовано полное описание формата печати, определяющего поля данных в файле и их взаимное расположение по отношению друг к другу, а также точный тип данных для каждого поля (вещественное число с плавающей запятой, целое число, символ, символьная строка переменной длины).

Спецификации, использующие для модели реализации данных формат схемы XML, будут, скорее всего, использовать тот же формат для публикации XML-документов. Схема XML обеспечивает определение XML-документа, инструменты для его синтаксического анализа и подтверждения применения XML-схемы.

3.7 Совместимые спецификации

Открытая спецификация информационной архитектуры систем контроля состояния и диагностики должна реализовывать информационную архитектуру, описанную в 3.1—3.6. Спецификация, совместимая с указанными требованиями, разработана MIMOSA, некоммерческим объединением, разрабатывающим открытые информационные спецификации для систем контроля состояния и диагностики, и известна как «архитектура открытых систем MIMOSA для интеграции приложений предприятия (OSA-EAI)». Описание такой спецификации приведено в приложении А.

4 Требования к архитектуре обработки данных систем контроля состояния и диагностики

4.1 Общие положения

Архитектура обработки данных описывает все процедуры обмена данными между модулями самой системы, между системой и конечным пользователем и с программными продуктами других систем. Открытая спецификация архитектуры обработки данных систем контроля состояния и диагностики должна обеспечить реализацию процедур обмена информацией, показанной на рисунке 2 в виде блоков различного назначения. Каждый блок системы должен быть соответствующим образом конфигурирован.

Данные, полученные из блока сбора данных (DA) в цифровом формате, после соответствующих преобразований приобретают вид соответствующих рекомендаций на выходе блока составления рекомендаций (AG). По мере продвижения от блока DA к блоку AG данные поступают на очередной блок преобразования вместе с дополнительной информацией от внешних систем, а с выхода этого блока также могут быть посланы внешним системам. При этом данные, вовлекаемые в информационный поток, нуждаются в соответствующем стандартном отображении и простом графическом представлении. Многие приложения в целях сохранения результатов преобразования информации каждым блоком системы требуют, чтобы соответствующие данные были архивированы. Блоки DA, DM и SD отвечают за оценку качества данных, которое может быть высоким, низким или неопределенным.

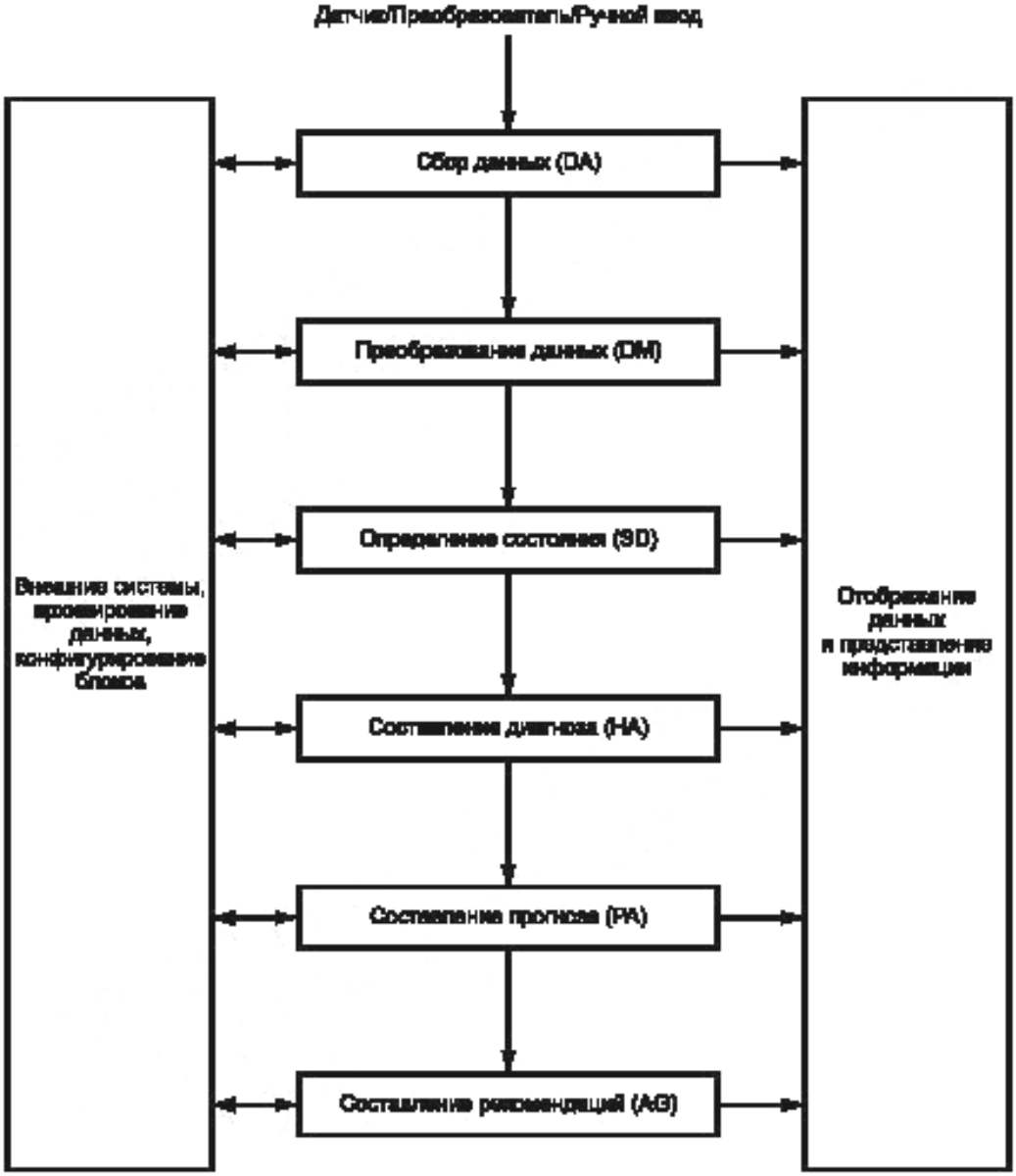

Рисунок 2 — Блок-схема этапов обработки данных

4.2 Блок сбора данных (DA)

Как показано на рисунке 3, блок DA представляет собой обобщенный программный модуль, обеспечивающий поступление в систему оцифрованных данных в автоматизированном режиме или в режиме ручного ввода. Блок может быть сконфигурирован специальным образом для приема аналогового сигнала с датчика или для приема информации по шине данных, собирающей сигналы с разных датчиков. В более современных системах программный интерфейс блока может обеспечить поступление данных с микропроцессорного датчика. В любом случае блок DA можно рассматривать как сервер, собирающий оцифрованные измеримые данные.

Выходами блока DA являются:

- оцифрованные данные;
- временные отметки, обычно синхронизованные с местным временем;
- индикатор качества данных (с градациями: «высокое качество», «низкое качество», «не определено», «в стадии определения» и т. п.).

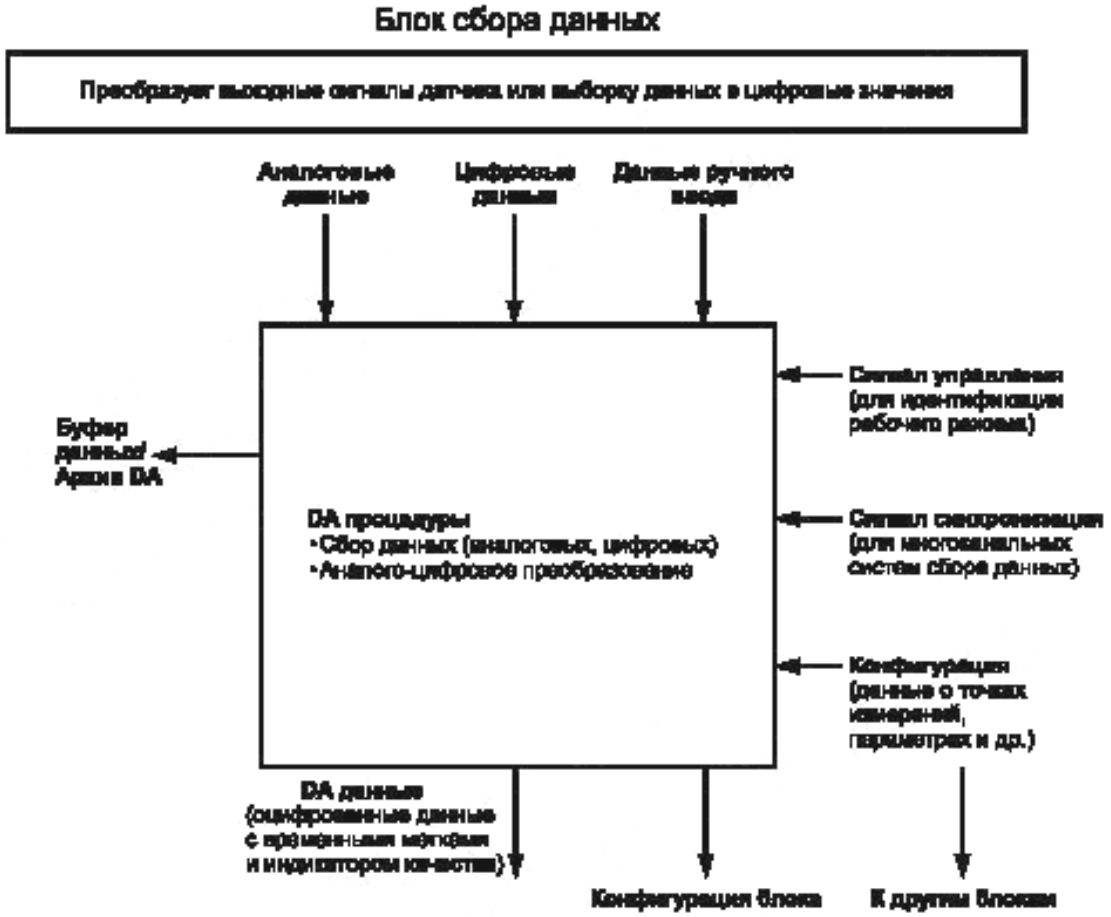

Рисунок 3 — Блок сбора данных

Примерами оцифрованных данных могут быть:

- скалярные значения в виде вещественного числа с плавающей запятой;
- выборочные значения временного сигнала;
- данные теплового излучения в виде оцифрованного изображения (термограммы);
- данные результатов контроля проб масла, воздуха, воды.

4.3 Блок преобразования данных (DM)

Как показано на рисунке 4, блок DM обрабатывает цифровые данные, полученные с блока DA, чтобы преобразовать их в параметры (признаки), используемые в процедурах контроля состояния и диагностирования. Зачастую в этом блоке используются несколько специализированных алгоритмов обработки сигнала. К таким алгоритмам относятся, например, быстрое преобразование Фурье (БПФ), вейвлет-анализ, усреднение данных на заданном интервале времени.

Примерами результатов работы блока DM являются:

- выделенные диагностические признаки;
- представление данных в частотной области преобразованием временного сигнала и, наоборот, во временной области преобразованием спектра сигнала;
- расчетные или статистические значения;
- сигнал виртуального датчика (например, разность давлений на входе и выходе);
- сигналы скорости и перемещения (полученные интегрированием сигнала ускорения);
- отфильтрованный сигнал;
- нормализованный сигнал;
- временной ряд с указанием частоты выборки.

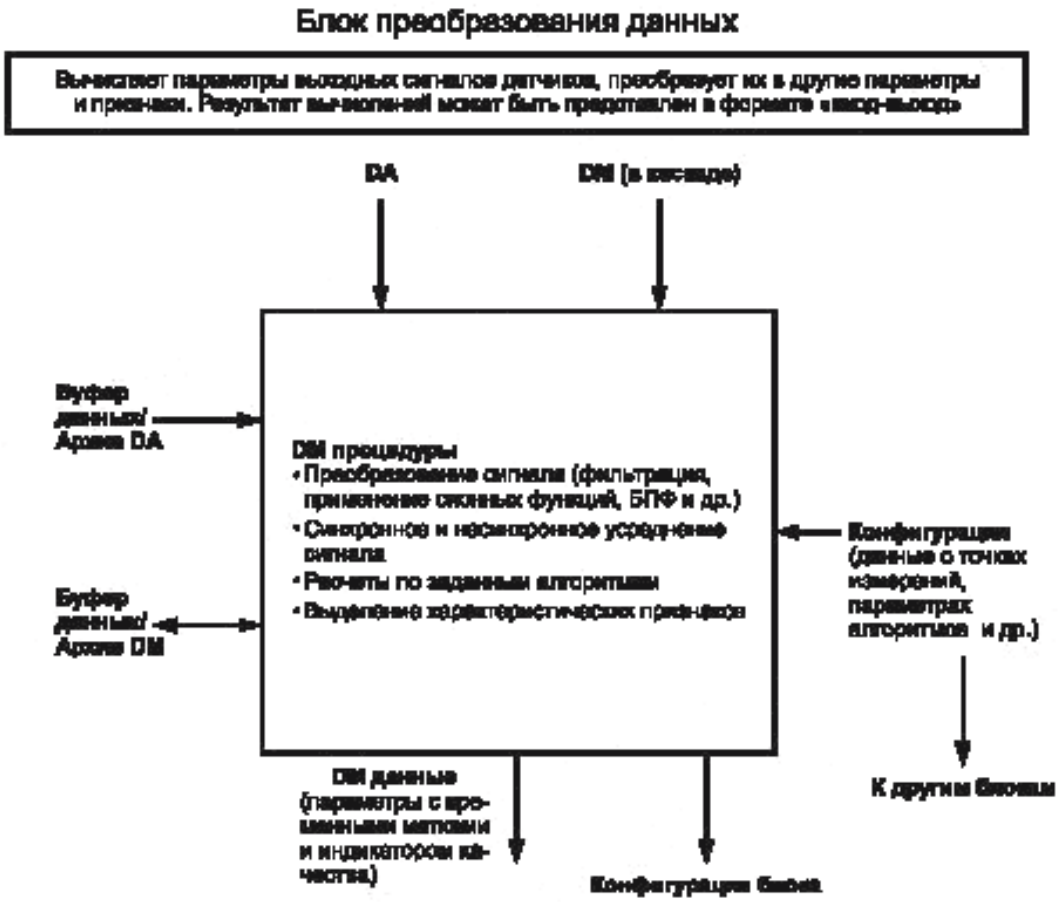

Рисунок 4 — Блок преобразования данных

4.4 Блок определения состояния (SD)

Как показано на рисунке 5, основной функцией блока SD является сравнение данных с входов блоков DM и (или) DA со значениями базовой линии или предельными значениями эксплуатационных параметров, чтобы сформировать сигнал оповещения при превышении соответствующих границ. Выходные данные блока SD могут быть использованы блоком HA для формирования сигналов уведомления и предупреждения. Оценочные значения, формируемые блоком SD, должны быть получены с учетом предшествующей истории работы машины, текущего рабочего режима и условий эксплуатации.

Обычно информация с выхода этого блока используется для составления диагноза в блоке HA. Для оценки технического состояния блок SD может использовать как текущие, так и предшествующие данные с выходов блоков DA и DM. В нем может также осуществляться преобразование данных и формирование команд управления сбором данных (например, порядком и временем опроса датчиков).

Примерами результатов работы блока SD являются:

- числовое значение индикатора технического состояния;
- оповещение о достижении пороговых значений;
- оценка степени превышения порогового значения или приближения к нему;
- оповещение о скорости изменения контролируемых параметров;
- оценка степени отклонения от нормы;
- результаты статистического параметрического или непараметрического анализа (например, оценок распределений Вейбулла или Гаусса).

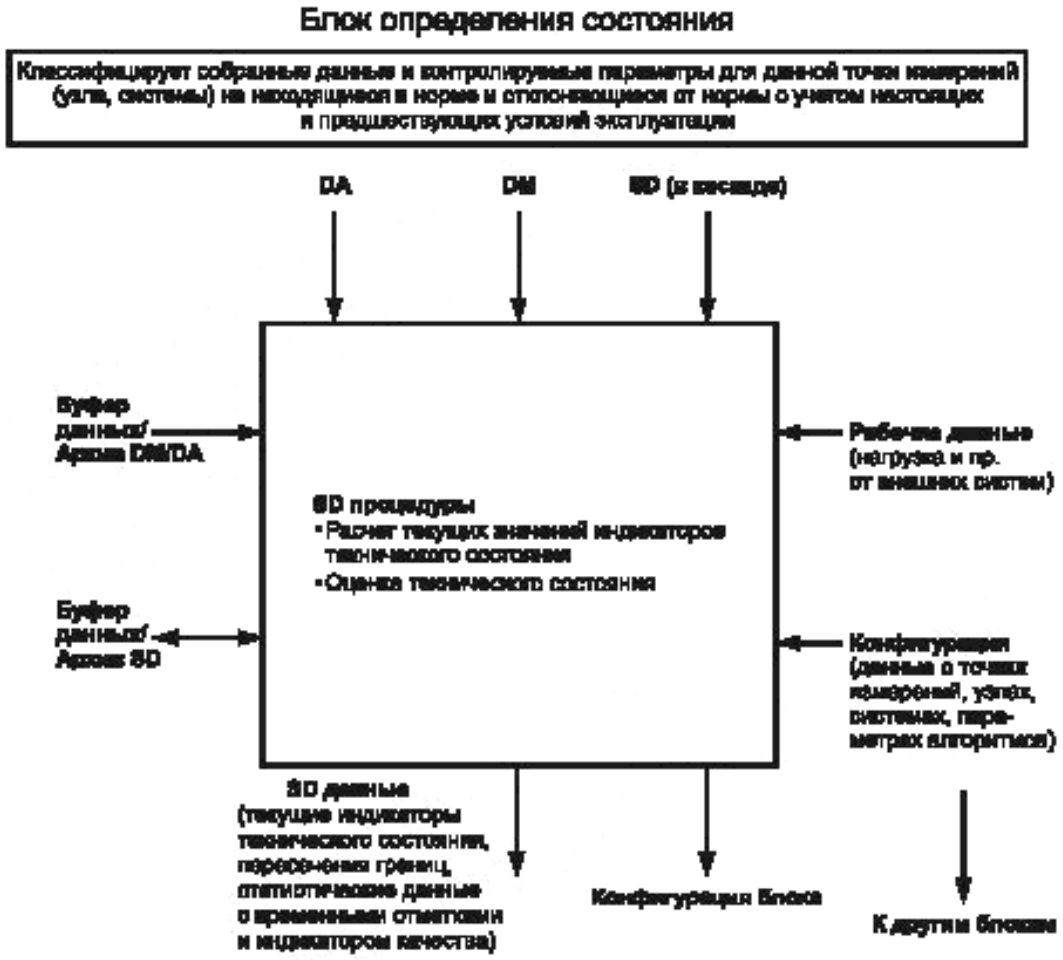

Рисунок 5 — Блок определения состояния

4.5 Блок составления диагноза (HA)

Как показано на рисунке 6, блок HA является информационным блоком, использующим экспертные правила или вычислительные процедуры для определения текущего состояния машин и выявления возможных неисправностей. Заключение о текущем состоянии и возможных неисправностях формируется на основе данных с блоков DA, DM, SD и других блоков HA, работающих в каскаде.

Результатом работы блока HA являются оценка технического состояния узлов (систем) и выявленные неисправности и отказы с указанием доверительного уровня диагноза. Выходная информация может также содержать уровень приоритета риска отказа. Диагноз может включать в себя альтернативные версии. Выходная информация может также содержать пояснения, почему был составлен данный диагноз или получена данная оценка технического состояния.

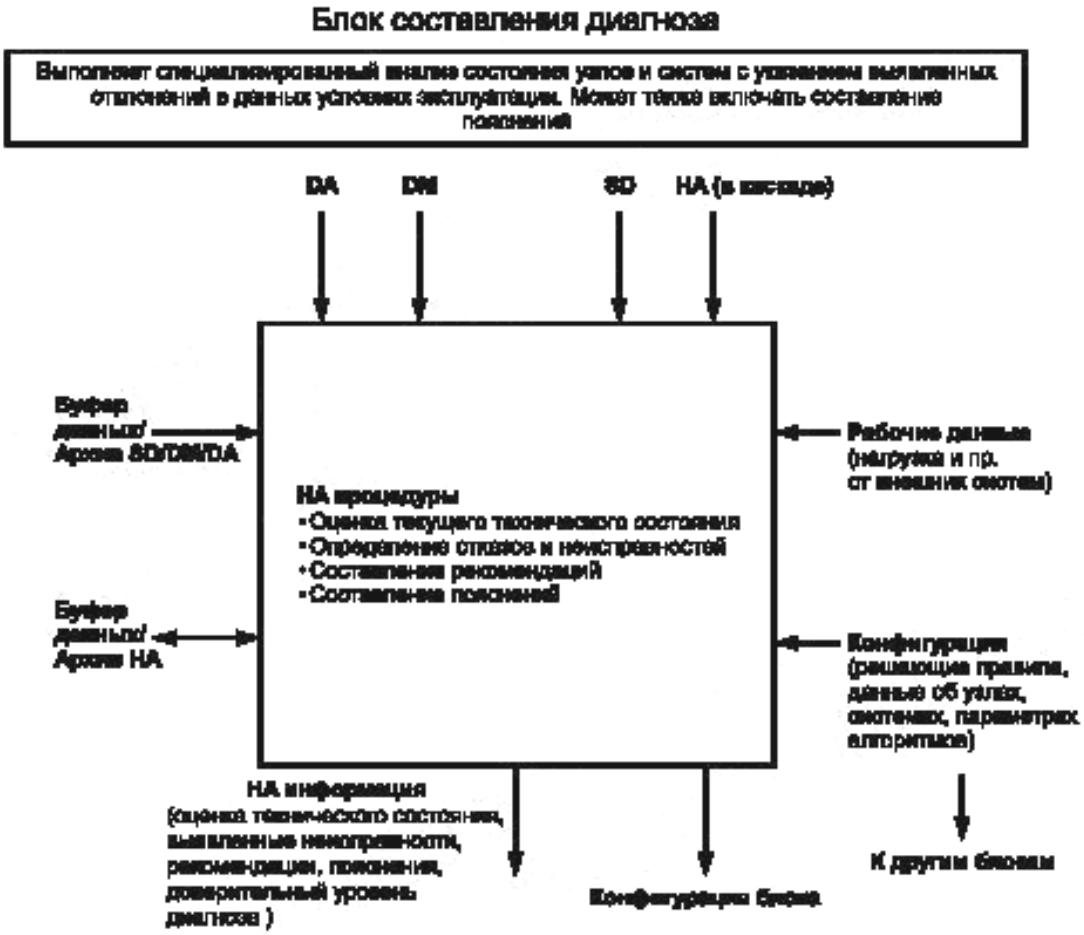

Рисунок 6 — Блок составления диагноза

4.6 Блок составления прогноза (PA)

Как показано на рисунке 7, основной функцией блока PA является определение технического состояния контролируемой машины в будущие моменты времени с использованием разнообразных прогностических моделей и алгоритмов, включая модели развития будущих отказов. Заключение о вероятном техническом состоянии в будущие моменты времени и возможном развитии неисправностей конкретных видов формируется на основе данных с блоков DA, DM, SD, HA и других блоков PA с учетом планируемого использования машины. Алгоритмы и модели блока могут использовать накопленные данные об истории эксплуатации машины и имевших место отказах, а также информацию о расчетной частоте отказов в конкретных условиях эксплуатации.

На этапе прогнозирования помимо оценки технического состояния в будущие моменты времени может быть дана также оценка остаточного ресурса машины для данного режима работы. Эти оценки могут сопровождаться прогнозом будущих отказов и указанием уровня приоритета риска отказа. Выходная информация блока содержит оценку технического состояния узлов (систем) в будущие моменты времени и прогнозируемые отказы вместе с оценкой достоверности прогноза. Прогноз может включать в себя альтернативные версии. Выходная информация может также содержать пояснения, почему был составлен данный прогноз отказов или получена данная оценка технического состояния в будущие моменты времени.

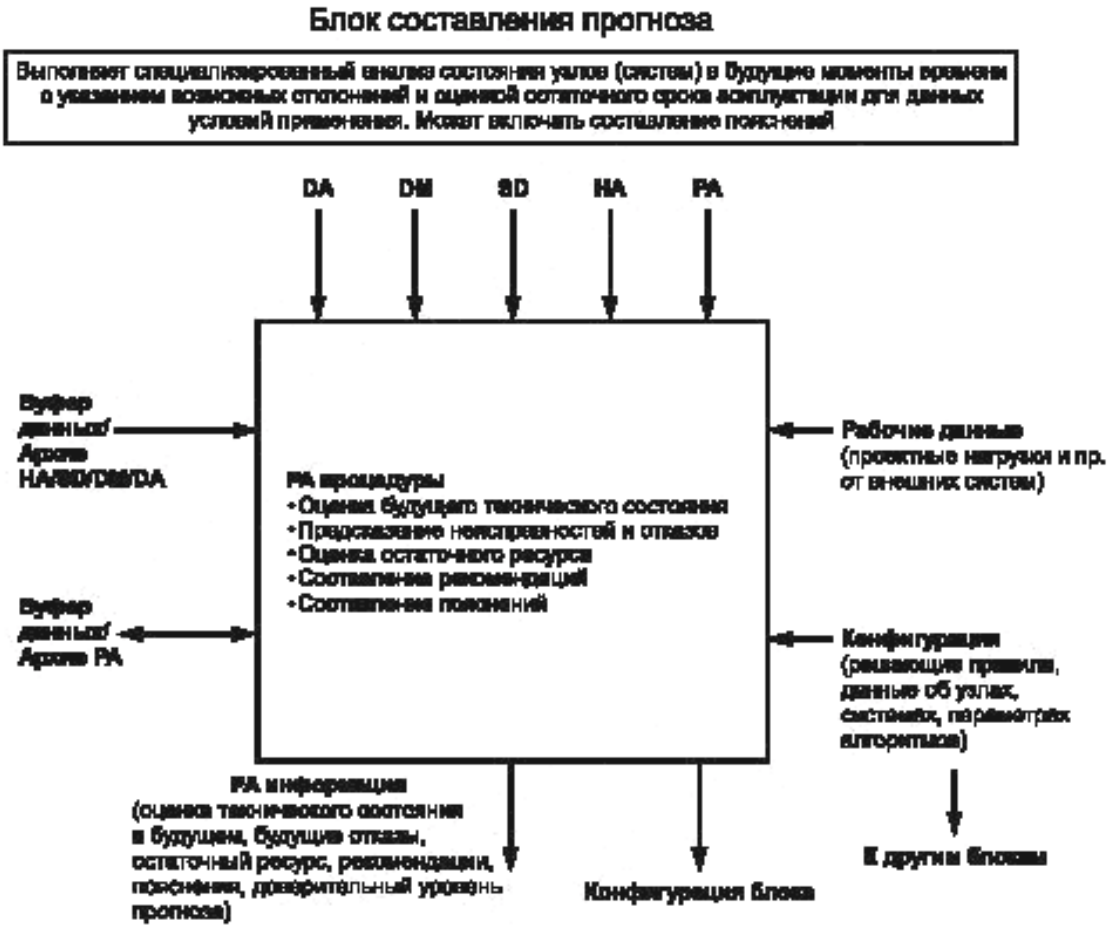

Рисунок 7 — Блок составления прогноза

4.7 Блок составления рекомендаций (AG)

Как показано на рисунке 8, основной функцией блока AG является объединение информации, поступающей от блоков DA, DM, SD, HA, PA и других блоков AG, внешних ограничений (требований безопасности, возможностей бюджета и пр.) для выработки оптимальных рекомендаций и возможных альтернатив для персонала и внешних систем. Рекомендации могут указывать приоритетность возможных действий по управлению машинами и их обслуживанию, а также включать в себя предложения по изменению условий эксплуатации. При выработке рекомендаций необходимо иметь данные об истории работы машин (включая записи об их использовании и выполненные операции технического обслуживания), их назначение в настоящий момент и в будущем, а также ограничения на ресурсы.

Выработанные блоком рекомендации по техническому обслуживанию должны подробно указать последовательность будущих работ, которые могут включать в себя проверку данных мониторинга или проведение дополнительного мониторинга. Рекомендации должны быть выполнены в формате заявки на производство работ для передачи ее во внешнюю систему управления техническим обслуживанием. На основе этой заявки данная система сможет заблаговременно составить график работ и обеспечить наличие запасных частей и инструмента для их проведения.

Рекомендации по управлению машинами могут как иметь вид текущих сообщений оператору об изменении технического состояния и действиях, которые должны быть в связи с этим предприняты, так и быть связанными с более долговременными мероприятиями, например, через посылку извещения в систему планирования производства о высоком риске остановки технологической линии вследствие ожидаемого в ближайшее время отказа машин.

Блок AG может также давать ответы на запросы о вероятности надежного функционирования машин в рамках технологического процесса. Такая оценка весьма важна для систем прогнозирования производства при решении ими вопроса о возможных заказах и распределении работ, исходя из принципов оптимального проектирования.

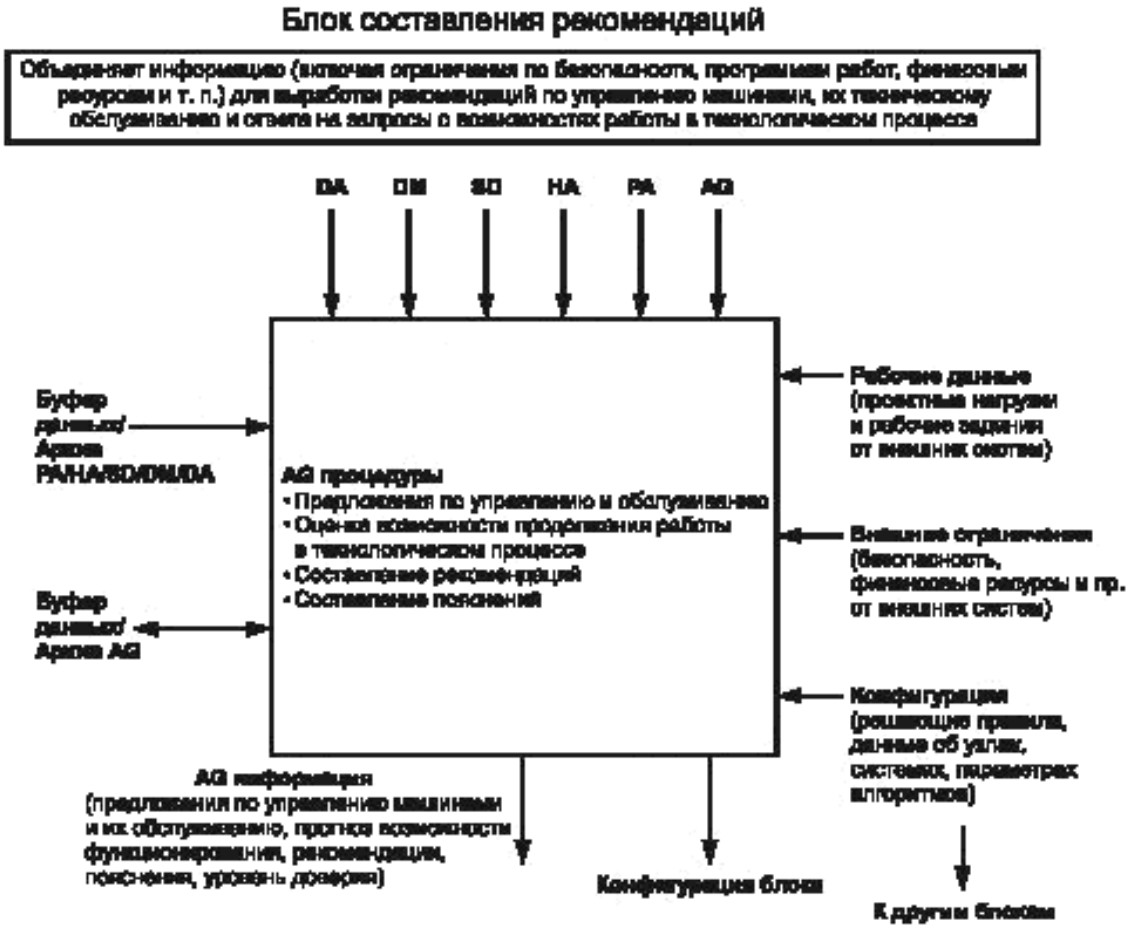

Рисунок 8 — Блок составления рекомендаций

4.8 Конфигурирование блоков

Для каждого блока обработки данных необходима информация о его конфигурации, которая может включать в себя как постоянные параметры, так и параметры, изменяющиеся в процессе работы системы мониторинга.

В качестве примера ниже приведен образец вопросов, на которые следует дать ответ при конфигурировании блока сбора данных:

a) описание точек измерений (таблица точек измерений):
1) ориентация и относительное расположение,
2) описание места измерений;
b) интервал между сбором данных:
1) непрерывный сбор данных,
2) периодический сбор данных:
- интервал между сбором данных по умолчанию,
- параметры сбора данных по умолчанию,
c) управление сбором данных (по сигналу):
1) уставка,
2) область нечувствительности;
d) синхронизация (отсутствие синхронизации) сбора данных;
e) информация о преобразователях и измерительных каналах:
1) коэффициент преобразования,
2) допуск,
3) перечень характеристик преобразователя,
4) число измерительных каналов;
f) сведения о калибровке.

4.9 Внешние системы

В оценке технического состояния машины важную роль играет хранимая в системе технического обслуживания информация об истории эксплуатации машин данного вида, о прошлых результатах измерений эксплуатационных параметров (периоды эксплуатации и применявшиеся нагрузки). После составления диагноза рекомендуемые решения по обслуживанию могут лежать в диапазоне от сокращения интервала между наблюдениями до проведения ремонтных работ и замены машины или ее узла. В отношении функционирования машины может быть принято решение от изменения режима работы до немедленного останова машины. Возможные решения предъявляют требования к скорости передачи сообщений в систему технического обслуживания и систему управления машиной, т. е. к программному интерфейсу.

4.10 Архивирование данных

Архивирование данных является важным элементом работы всех блоков системы контроля состояния и диагностики. Результаты текущих измерений могут быть подвергнуты анализу на статистическую связь с предшествующими данными. Система архивирования данных должна предусматривать правила в отношении частоты и объема архивирования. Сохраненные в архиве решения (рекомендации) системы по получении новой информации следует проверять на их точность (правильность). С поступлением новой информации уточняются данные о причинах отказов.

4.11 Отображение данных

С целью облегчения анализа данных они должны быть отображены в виде трендов и соответствующих зон состояния. Это даст возможность персоналу идентифицировать или подтвердить отклонение в состоянии и понять природу этого отклонения.

4.12 Представление информации

Блоки HA, PA и AG должны соответствующим образом представлять выходную информацию. Важно, чтобы данные были преобразованы к виду, из которого легко извлечь информацию, необходимую для принятия решений о корректирующих действиях. В некоторых случаях после получения информации об обнаруженных отклонениях в техническом состоянии машин пользователю необходимо иметь возможность исследовать более подробно, как исходные данные были отображены блоками SD, DM и DA.

4.13 Совместимые спецификации

Открытая спецификация архитектуры обработки данных систем контроля состояния и диагностики должна реализовывать архитектуру, описанную в настоящем разделе. Спецификации должны определять модель (схему) обработки данных и интерфейс программирования приложений [Application Programming Interface (API)], описание которого должно удовлетворять требованиям ИСО/МЭК 14750. Это позволит блоки обработки данных от различных поставщиков интегрировать в единую функциональную систему. Спецификация, совместимая с указанными требованиями, опубликована MIMOSA и известна как «архитектура открытых систем MIMOSA для технического обслуживания по состоянию (OSA-CBM)». Описание этой спецификации приведено в приложении А.

На схеме рисунка 9 показано, как блоки могут взаимодействовать друг с другом для образования единой интегрированной системы. Схема имеет вид «колеса», где «втулка» представляет собой средство коммуникаций между блоками, которое может быть реализовано с использованием широко применяемых коммуникационных и подпрограммных технологий (понятие подпрограммного или, иначе говоря, межплатформенного программного обеспечения раскрывается в приложении В). Как следствие, отсутствует необходимость размещать все модули на одном компьютере. Вместо этого они могут быть рассредоточены в разных местах, соединенных через локальную сеть или Интернет. Конструкция архитектуры открытых систем дает возможность интеграции улучшенных прогностических возможностей в новую или существующую систему, обеспечивая максимальную гибкость в ее обновлении.

Один модуль может объединять в себе функции одного или нескольких блоков обработки данных описанной архитектуры. В примере на рисунке 10 показано, как модуль А осуществляет функции блока определения состояния, в то время как модуль В объединяет функции двух блоков: блока составления диагноза и блока составления прогноза. В одном модуле может быть применен один или несколько API. В примере на рисунке 10 модуль D объединяет функции двух блоков: блока преобразования данных и блока определения состояния. В этом случае данный модуль может предоставлять информацию об обработанных данных и об оценке технических состояний.

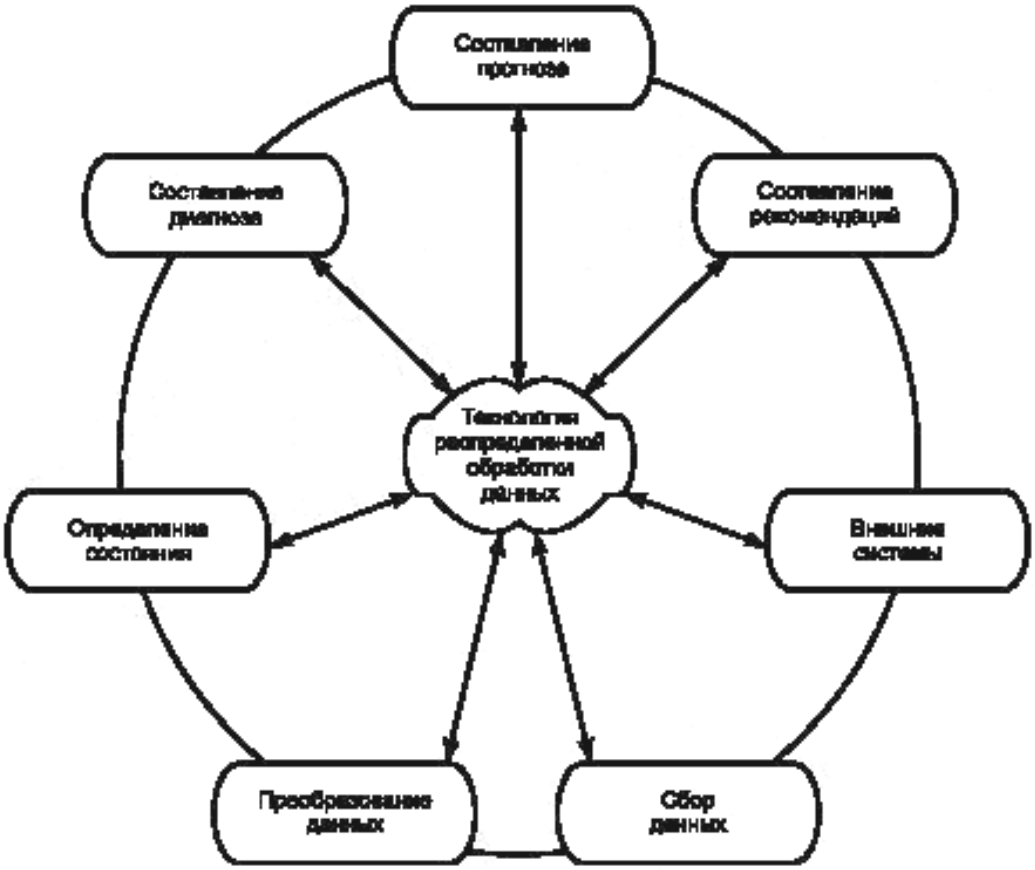

Рисунок 9 — Поток обработки данных в рамках стандартной архитектуры

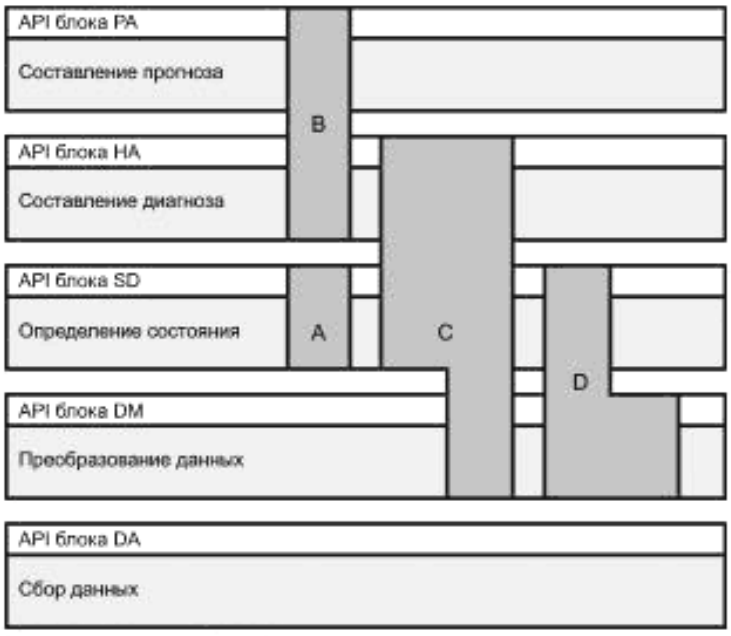

Рисунок 10 — Пример объединения в модуле функций нескольких блоков

Пример совместимой системы показан на рисунке 11. Эта система имеет большое число блоков сбора данных (DA), эти собранные данные поступают в блоки, выполняющие разнообразные функции, пока, наконец, единый блок составления рекомендаций (AG) не выдаст рекомендации по обслуживанию и управлению на внешний дисплей пользователя.

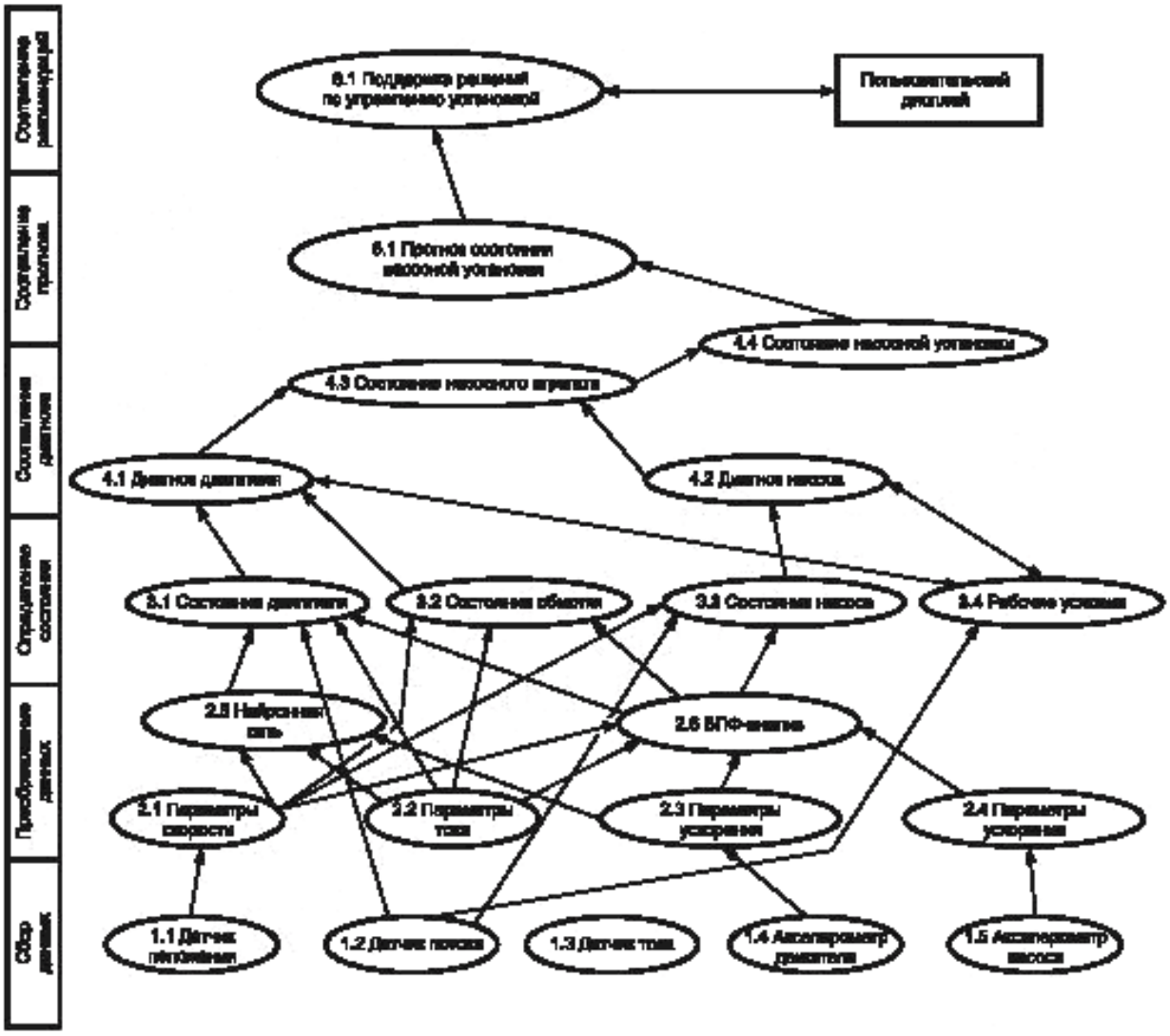

Рисунок 11 — Пример совместимой модульной системы

Приложение А
(справочное)

Совместимые спецификации

А.1 Спецификация информационной архитектуры MIMOSA OSA-EAI для систем контроля состояния и диагностики

А.1.1 Общие положения

MIMOSA представляет собой некоммерческое профессиональное объединение, разрабатывающее и продвигающее открытые информационные стандарты в области производственной деятельности и технического обслуживания. MIMOSA объединяет в своем составе провайдеров систем управления промышленными объектами и конечных пользователей этих систем, которые участвуют в разработке открытых спецификаций интегрирования информации для управления физическими объектами.

Спецификация архитектуры открытых систем MIMOSA для интеграции приложений предприятия (OSA-EAI), находящаяся в свободном доступе на сайте http://www.mimosa.org/, совместима со спецификацией информационной архитектуры для систем контроля состояния и диагностики по ИСО 13374-1 и настоящему стандарту и облегчает интегрирование управления объектами и информацией по территориально распределенным производствам. Спецификации MIMOSA OSA-EAI предназначены для специалистов систем обеспечения надежности и технического обслуживания на предприятии, а также для разработчиков и поставщиков производственных технологий.

Для первой группы пользователей принятие спецификаций MIMOSA OSA-EAI облегчает интегрирование информации об управлении объектами, обеспечивает свободу выбора из широкого спектра программных приложений и экономию средств за счет уменьшения трудоемкости интегрирования и поддержания программных средств.

Для поставщиков технологий принятие спецификаций MIMOSA OSA-EAI расширяет рынок сбыта их продукции, позволяет сосредоточить усилия на поиске эффективных технологических решений, не распыляя ресурсы на обеспечение совместимости платформ и пользовательских интерфейсов, и, в конечном итоге, сокращает издержки производства.

MIMOSA регулярно обновляет стандарт OSA-EAI, добавляя в него новые возможности. При ссылке на стандарт рекомендуется указывать сайт MIMOSA (http://www.mimosa.org/), где размещена последняя версия стандарта.

А.1.2 Архитектура

А.1.2.1 Схема архитектуры

Архитектура OSA-EAI, версия 3.1, показана на рисунке А.1.

<table>
<tr><td>Tech-Doc-File для импорта и экспорта XML-файлов</td><td>Tech-CDE-Services для программного интерфейса Tech-XML Clients & Servers</td><td>Tech-XML-Web для интерфейса Tech-XML Clients & Servers по протоколу HTTP</td><td>Tech-XML-Services для программного интерфейса Tech-XML Clients & Servers</td><td>Определение EAI-приложений</td></tr>
<tr><td>Tech-Doc, CRIS-совместимые схемы XML-документов для поставщика и потребителя данных</td><td>Tech-CDE-Aggregate, CRIS-совместимые XML-транзакции в схеме клиент — сервер</td><td colspan="2">Tech-XML Atomic, CRIS-совместимые XML-транзакции в схеме клиент — сервер</td><td>Определение XML-документа</td></tr>
<tr><td colspan="4">Библиографическая база данных CRIS</td><td>Классификация метаданных</td></tr>
<tr><td colspan="4">Общая реляционная информационная схема (CRIS)</td><td>Модель реализации данных</td></tr>
<tr><td colspan="4">Общая концептуальная объектная модель (CCOM) OSA-EAI</td><td>Концептуальная модель</td></tr>
<tr><td colspan="4">Терминологический словарь OSA-EAI</td><td>Семантические определения</td></tr>
</table>

Рисунок А.1 — Схема архитектуры MIMOSA OSA-EAI (Лист 1)

Технологические типы для интерфейсов *Tech*-Doc, *Tech*-CDE и *Tech*-XML:
REG — Управление объектным реестром;
WORK — Управление работой экспертов (интеллектуальных программ);
DIAG — Диагностирование, прогнозирование, оценка состояния;
TREND — Рабочие параметры и границы предупреждения;
DYN — Временные сигналы вибрации и звука и границы предупреждения;
SAMPLE — Состав масла (жидкости, газа, твердого вещества) и границы предупреждения;
BLOB — Бинарные объекты (термограммы) и границы предупреждения;
REL — RCM, FMECA, модели надежности;
TRACK — Пространственное расположение объектов

Рисунок А.1 — (Лист 2)

А.1.2.2 Пояснения к используемым терминам

А.1.2.2.1 Терминологический словарь OSA-EAI

Для облегчения взаимопонимания между сторонами, использующими спецификацию MIMOSA OSA-EAI, MIMOSA предоставляет стандартный набор терминов в Терминологическом словаре OSA-EAI, обеспечивая основные семантические описания, которые используются в спецификациях OSA-EAI.

А.1.2.2.2 Общая концептуальная объектная модель (CCOM)

Общая концептуальная объектная модель (CCOM) обеспечивает основу концептуальной модели для OSA-EAI. Разработчики программного обеспечения, знакомые с диаграммами классов в унифицированном языке моделирования (UML), могут использовать эту модель для лучшего понимания основных классов, основных атрибутов и взаимоотношений между классами в OSA-EAI. CCOM версии 3.0 доступна в виде pdf-документа, а также в видео-формате (VSD).

А.1.2.2.3 Общая реляционная информационная схема (CRIS)

Общая реляционная информационная схема (CRIS) MIMOSA обеспечивает общую схему реализации данных для передачи информации между разными системами и ее интегрирования. Реляционный формат схемы делает ее совместимой с большинством систем баз данных. Каждой таблице в CRIS присваивается уникальный ссылочный номер. CRIS версии 3.0 опубликована в виде pdf-документа, в формате Word, а также в формате XML.

Источники данных, содержащие информацию об объекте, не требуют физического перепроектирования для совместимости с CRIS, но они должны быть способны передавать информацию в столбцы таблиц CRIS. Чтобы облегчить эту передачу, а также для тех пользователей спецификаций MIMOSA, которые хотели бы применять CRIS-совместимые физические базы данных, MIMOSA предоставляет скрипты (сценарии) для создания таблиц в системах управления базами данных ORACLE и Microsoft SQLServer.

А.1.2.2.4 Общая концептуальная объектная модель и Общая реляционная информационная схема (CCOM/CRIS)

CCOM/CRIS содержит номенклатуру идентификаторов предприятий, их участков, сегментов, объектов и групп, типичных для структуры корпоративной организации. Каждому «предприятию» присвоен определенный «тип предприятия». Предприятие присваивает «уникальный код местонахождения» каждому новому «(производственному) участку» и может управлять одним или многими «участками» (в том числе теми, которые ранее могли управляться другими предприятиями). Чтобы предприятия могли обмениваться информацией, каждое предприятие должно затребовать и использовать собственный неизменяемый «уникальный код предприятия».

«Участок» — это то, что предприятие определяет как некую целостную структуру, которая может быть разбита на отдельные «сегменты», в свою очередь определяющие объекты, группы, базы данных и точки измерений. Одно предприятие может иметь много участков. Один участок может включать много сегментов. Применительно к производствам под «участком» обычно понимают построенное сооружение. В промышленности под целостной структурой обычно понимают цех. Для транспортной организации это может быть депо технического обслуживания, отвечающее за состояние определенных объектов. Некоторые очень большие объекты, например авианосец, сами могут рассматриваться в качестве «участка». Каждое «предприятие» присваивает каждому своему «участку» его собственный неизменяемый «уникальный код участка».

Поскольку CCOM и CRIS проектировались для управления жизненным циклом объектов на территориально разнесенных участках, почти все таблицы в CRIS (за исключением нескольких собственных ссылочных таблиц MIMOSA) включают в себя ссылки на предприятия, участки, базы данных. Чтобы минимизировать число столбцов первичных ключей, в CRIS 3-й версии индивидуальный код предприятия (4-битный) и индивидуальный код участка (тоже 4-битный) объединены в «код участка» фиксированной длины из 16 символов. Этот код получается преобразованием двух 4-битных неотрицательных целых чисел в шестнадцатеричный формат (по 8 символов на каждое целое число) с последующим их объединением в 16-символьную строку.

В CCOM и CRIS версий 3.1 также добавлена возможность разбивать функциональные участки на иерархию «сегментов», в рамках которых с течением времени могут определяться пронумерованные «объекты». Кроме того, CCOM/CRIS устанавливает способ стандартной идентификации точек измерений для различных технологий контроля состояния. CCOM/CRIS предусматривает моделирование изменяющихся скалярных данных, таких как рабочая температура, давление или нагрузка, а также поддерживает динамические данные в виде временного сигнала

или спектра, используемые при анализе вибрации и мониторинге акустического шума. Для передачи рисунков, отчетов, диаграмм и фотографий предусмотрена поддержка массивных двоичных объектов (BLOB). В CCOM/CRIS предусмотрено также управление результатами контроля образцов, осуществляемого, например, в целях анализа масла и качества воздуха. CCOM/CRIS обеспечивает возможность коммуникации для получения диагностической информации от интеллектуальных систем и облегчает составление рекомендаций. В специальных таблицах обслуживания и надежности определены поля событий (реальных, гипотетических, прогнозируемых), состояний, прогнозируемых сроков и рекомендаций. CCOM/CRIS поддерживает планирование запросов по техническому обслуживанию и управлению производством, а также отслеживание выполнения мероприятий по техническому обслуживанию и производственным работам в отношении отдельных объектов. Кроме того, CCOM/CRIS обеспечивает информационную интегрированную среду хранения данных по надежности для разных объектов.

А.1.2.2.5 Спецификация библиографической базы данных CRIS

CCOM/CRIS содержит много «типов» классов (таблиц), позволяющих пользователям присваивать типы предприятиям, участкам, сегментам, объектам, группам, точкам измерений, единицам измерений и т. д. в виде стандартных числовых кодов, общих для всех систем. MIMOSA разрабатывает и поддерживает промышленно-ориентированные классификации и коды для большинства таких таблиц, однако CCOM/CRIS допускает возможность как для поставщиков, так и для конечных пользователей добавлять к таблицам собственные специфические элементы. Эксперты MIMOSA разработали большую базу библиографических данных, ее CRIS-совместимую спецификацию в формате XML, а также в форматах нескольких широко применяемых реляционных баз данных. Версия 3.1 спецификации этой базы данных содержит множество полезных кодов, что обеспечивает их унификацию не только для самых разнообразных систем, но и для разных стран. Например, таблица типов объектов обеспечивает стандартные запросы для объектов общеупотребительных типов, таких как «асинхронный электродвигатель переменного тока», который имеет неизменяемый трехцифровой уникальный идентификатор. Существуют также таблицы стандартных кодов для производственных сегментов, точек измерений, единиц измерений, результатов контроля проб, диагностических событий, технических состояний, видов и причин отказов. Это позволяет осуществлять поиск по разным системам для отказов общих видов и для конкретных типов машин.

А.1.2.2.6 Схемы *Tech*-Doc интерфейса для поставщика и потребителя данных

Для систем, основанных на документах в формате XML, MIMOSA предлагает схему интерфейса между поставщиком и потребителем данных *Tech*-Doc, которая позволяет преобразовывать данные CRIS в XML-документ. Интерфейсы *Tech*-Doc определяют содержание XML-документа, но не устанавливают способ его физического хранения и передачи. Приложения *Tech*-Doc Consumer Read-only используют XML-документы, созданные с помощью *Tech*-Doc, но не изменяют их постоянные данные. Приложения *Tech*-Doc Consumer Write-only изменяют их постоянные данные после успешного импорта файла.

А.1.2.2.7 Схемы *Tech*-CDE-Aggregate интерфейса CRIS-совместимых XML-транзакций в схеме клиент — сервер

Схемы интерфейса для обмена составными документами [Compound Document Exchange (CDE)] между сервером и клиентом *Tech*-CDE являются еще одним элементом OSA-EAI. Эти XML-схемы обеспечивают единый набор определений интерфейса обмена XML-документами между сервером и клиентом для запроса и передачи больших массивов данных, обычно хранимых в CRIS-формате на сервере системы. *Tech*-CDE-интерфейс определяет содержание обмена данными между клиентом и сервером, но не физический способ их передачи.

А.1.2.2.8 Схемы *Tech*-XML-Atomic интерфейса CRIS-совместимых XML-транзакций в схеме клиент — сервер

Схемы интерфейса между клиентом и сервером *Tech*-XML являются ключевым элементом OSA-EAI. Эти схемы обеспечивают единый набор определений интерфейса обмена XML-документами между сервером и клиентом для различных протоколов обмена данными, давая возможность специализированным системам предоставлять ответ в CRIS-формате на дискретные транзакции. *Tech*-XML-интерфейс определяет содержание обмена данными между клиентом и сервером, но не физический способ их передачи.

А.1.2.2.9 Технологические типы интерфейсов *Tech*-Doc, *Tech*-CDE и *Tech*-XML

Каждый из трех интерфейсов (*Tech*-Doc, *Tech*-CDE и *Tech*-XML) разделен на 10 технологических типов, перечисленных на рисунке А.1. Это позволяет разработчикам сосредоточиться на поддержке обмена информацией в одной конкретной предметной области, такой, например, как анализ вибрации или управление техническим обслуживанием. Для ссылки на полный набор из 10 технологий используется набранный курсивом префикс «*Tech*-». Каждой вертикальной технологии (см. рисунок А.1) приложения соответствуют определенные CRIS-таблицы. Это позволяет конструкторам уменьшить общий объем спецификации CRIS до рассмотрения только тех таблиц, которые необходимы для конкретного приложения.

А.1.2.2.10 Спецификации *Tech*-XML-Services и *Tech*-CDE-Services

Спецификации *Tech*-XML-Services и *Tech*-CDE-Services используются для построения сервера или клиента сервисно-ориентированных приложений [Service Oriented Application (SOA)], позволяющими обмениваться информацией между системами контроля состояния, системами диагностирования, системами управления надежностью, системами управления реестрами и системами управления производством работ посредством простого протокола доступа к объектам [Simple Object Access Protocol (SOAP)], использующего XML-сообщения. Интерфейсы системы клиент — сервер определены через схемы XML.

А.2 Спецификация архитектуры обработки данных MIMOSA OSA-CBM для систем контроля состояния и диагностики

Спецификация архитектуры открытых систем MIMOSA для технического обслуживания по состоянию (OSA-CBM), находящаяся в свободном доступе на сайте http://www.mimosa.org/, совместима со спецификацией архитектуры обработки данных для систем контроля состояния и диагностики по настоящему стандарту. Обработка информации начинается со сбора данных и проходит ряд функциональных уровней, прежде чем достигнет уровня составления рекомендаций. На каждом уровне существует возможность запросить данные с любого другого уровня, однако основной поток информации идет между соседними уровнями. Ниже указаны функции, выполняемые на каждом уровне обработки данных:

a) *Уровень 1 — Сбор данных*. Модуль сбора данных представлен в обобщенном виде как программный модуль, обеспечивающий поступление в систему оцифрованных данных с преобразователя. Модуль может быть конфигурирован специальным образом для приема аналогового сигнала с датчика или для приема информации по шине данных, собирающей сигналы с разных датчиков. Как вариант, он может представлять собой программный интерфейс для микропроцессорного (интеллектуального) датчика (например, совместимый с [13]). По сути, модуль сбора данных представляет собой сервер для записей оцифрованных данных с датчиков.

b) *Уровень 2 — Преобразование данных*. Модуль преобразования данных может быть выполнен в виде одноканальной или многоканальной схемы преобразования сигналов в вид, пригодный для их дальнейших преобразований алгоритмами СВМ.

c) *Уровень 3 — Определение технического состояния*. Основной функцией модуля определения состояния является сравнение основных параметров данных с ожидаемыми значениями или предельными значениями рабочих параметров и формирование упорядоченных индикаторов технического состояния (например, «низкий уровень», «нормальный уровень», «высокий уровень» и т. п.). Модуль определения состояния может также генерировать сигнал оповещения при превышении предельных значений. При наличии соответствующих данных монитор состояния может выводить информацию о рабочих условиях (текущем режиме работы или условиях эксплуатации).

d) *Уровень 4 — Составление диагноза*. Основной функцией модуля составления диагноза является определение того, не ухудшилось ли техническое состояние контролируемой машины, подсистемы или элементов оборудования. Если ухудшение произошло, то модуль может сформировать диагностическое сообщение с указанием одного или нескольких возможных неисправных состояний с соответствующим доверительным уровнем диагноза. При составлении диагноза должны учитываться тренды данных, рабочий режим и нагрузка, а также проведенные мероприятия по техническому обслуживанию.

e) *Уровень 5 — Составление прогноза*. Основной функцией модуля составления прогноза является определение на основе текущего технического состояния того, каким оно будет в будущие моменты времени с учетом планируемого применения машины. Сообщение модуля составления прогноза может иметь вид оценки технического состояния в будущие моменты времени или оценки остаточного ресурса объекта в предполагаемых условиях применения. Эти оценки могут сопровождаться также прогнозом будущих отказов.

f) *Уровень 6 — Составление рекомендаций*. Основной функцией модуля составления рекомендаций является выработка предложений по рекомендуемым действиям и возможным альтернативам, а также предложений по реализации данных рекомендаций. Рекомендации включают в себя планируемые действия по техническому обслуживанию, изменение условий работы машины с учетом выполняемых функций или изменение этих функций, исходя из производственной необходимости. При выработке рекомендаций должны учитываться данные об истории работы машины (включая стоявшие перед нею задачи и выполненные операции технического обслуживания), ее назначение в настоящий момент и в будущем, а также ограничения на ресурсы.

Составление задания на разработку спецификаций модулей СВМ, разработка этих спецификаций и конструирование на основе разработанных спецификаций модулей СВМ обеспечивает полную функциональность СВМ-систем. Разработка открытой архитектуры системы дает возможность интегрирования улучшенных прогностических возможностей вновь создаваемых или существующих систем, обеспечивает максимальную гибкость и возможность обновления системы.

А.3 Интегрированная среда OSA-CBM

Интегрированная среда OSA-CBM была разработана на основе описания функциональных уровней обработки данных, а также существующих и разрабатываемых стандартов мониторинга и технического обслуживания, таких как информационная схема MIMOSA OSA-EAI, AI-ESTATE[1], [13]. В настоящее время интегрированная среда OSA-CBM исключает уровень принятия решений, поскольку эта функция является специфической для конкретных приложений. Описанные уровни являются клиентскими уровнями, и они открыты для любой технологии пользовательского интерфейса, поэтому в настоящем стандарте никаких требований к интерфейсу не установлено. Первым шагом было определение объектно-ориентированной модели данных на унифицированном языке моделирования

[1] См. IEC 62243:2005 «Artificial intelligence exchange and service tie to all test environments (AI-ESTATE)» (МЭК 62243:2005 «Обмен информацией и сервисные программы в системах искусственного интеллекта для разных сред тестирования (AI-ESTATE)»).

(UML) для каждого уровня, чтобы затем преобразовать ее в абстрактную спецификацию интерфейса. Такая абстрактная спецификация может быть впоследствии преобразована для межплатформенного программного обеспечения конкретного приложения.

Объектная модель UML определяет только интерфейсы. Модель не определяет классы объектов (внутри каждого уровня архитектуры), которые требовали бы применения соответствующих программных средств. Основное внимание уделяется описанию структуры информации на каждом уровне, которая могла бы представлять интерес для пользователя. OSA-CBM не налагает никаких ограничений на внутреннюю структуру соответствующих программных модулей. Все ограничения архитектуры относятся только к структуре открытого интерфейса и к функциональности модулей. Такой подход позволяет использовать собственные алгоритмы пользователя и применяемые им подходы к проектированию программных средств в рамках программного модуля.

Приложение В
(справочное)

Понятия UML, XML, межплатформенного программного обеспечения и связанные с ними термины

В.1 Общие положения

Настоящее приложение содержит перечень определений и терминов, относящихся к унифицированному языку моделирования UML, расширяемому языку разметки XML и межплатформенного (промежуточного) программного обеспечения. Дополнительно приведены сведения о справочниках и подробных руководствах по данным вопросам.

В.2 Унифицированный язык моделирования (Unified Modeling Language, UML)

В.2.1 Определение языка UML

UML — графический язык для создания и документирования деталей программной системы. UML предоставляет стандартный способ описания проекта системы, включая концептуальные составляющие — такие как бизнес-процессы и системные функции — и конкретные составляющие — например, конструкции языка программирования, схемы баз данных и повторно используемые программные компоненты (см. [27]).

В.2.2 Термины

Термины и их определения взяты из разных источников, в частности [20].

Операция (activity) — шаг или действие в рамках диаграммы операций, представляющее действие, предпринятое системой или участником.

Диаграмма операций (activity diagram) — схема, отображающая шаги, решения и параллельные операции внутри какого-либо процесса (например, алгоритма или бизнес-процесса).

Участник (actor) — человек или внешняя компьютерная система, взаимодействующая с разрабатываемой системой.

Ассоциация (association) — соединение между двумя элементами модели. Ассоциация может представлять собой член класса в коде, связь между объектом и записями о нем, взаимосвязь между двумя рабочими процессами или любую другую связь подобного вида. По умолчанию оба элемента ассоциации равноправны и получают информацию друг о друге через ассоциацию. Также ассоциация может быть направленной. В этом случае только один из элементов получает информацию о другом элементе ассоциации.

Класс-ассоциация (association class) — класс, предоставляющий информацию об ассоциации между двумя другими классами.

Атрибут (attribute) — поле с данными о классификаторе.

Базовый класс (base class) — класс, определяющий атрибуты и операции, наследуемые подклассом через отношение обобщения.

Ветвление (branch) — точка принятия решения в диаграмме операций. Каждый из нескольких переходов из данной точки характеризуется своим условием, только одно из которых может принять истинное значение. Переход с истинным значением определяет дальнейшее течение процесса.

Класс (class) — категория похожих объектов, описываемая одним набором атрибутов и операций. Такие объекты совместимы с точки зрения присваивания.

Диаграмма классов (class diagram) – диаграмма, демонстрирующая отношения между классами и интерфейсами.

Классификатор (classifier) — элемент UML, содержащий атрибуты и операции, в частности, такие как участник, класс или интерфейс.

Взаимодействие (collaboration) — отношение между двумя объектами в диаграмме отношений, обозначающее возможность двунаправленной передачи сообщений между объектами.

Диаграмма взаимодействия (collaboration diagram) — диаграмма, показывающая взаимодействия между объектами, а также сообщения, отсылаемые в рамках данных взаимодействий для реализации некоторой модели поведения.

Компонент (component) — минимальная внедряемая единица кода программной системы.

Диаграмма компонентов (component diagram) – диаграмма, отображающая отношения между различными компонентами и интерфейсами.

Зависимость (dependence) — отношение, показывающее, что классификатор осведомлен об атрибутах и операциях другого классификатора, но не связан напрямую с его экземплярами.

Диаграмма внедрения (deployment diagram) — диаграмма, демонстрирующая отношения между различными процессорами.

Элемент (element) — любая составляющая модели.

Событие (event) — (применительно к диаграмме состояний) входной сигнал, явление или воздействие, приводящее к предпринимаемому системой действию или к изменению ее состояния.

Конечное состояние (final state) — обозначение завершающей точки для диаграмм состояний и операций.

Развилка (fork) — точка на диаграмме операций, в которой берут начало несколько параллельных потоков управления.

Обобщение (generalization) — отношение наследования, в котором подкласс наследует операции и атрибуты от базового класса и добавляет новые.

Начальное состояние (initial state) — обозначение начальной точки для диаграмм состояний и операций.

Интерфейс (interface) — классификатор, определяющий атрибуты и операции, формирующие соглашение о поведении. Обслуживающий класс или компонент может выбрать реализацию интерфейса (т. е. применение его операций и атрибутов). В этом случае обслуживаемый класс или компонент может зависеть не от самого обслуживающего класса или компонента, а от выбранной им реализации интерфейса.

Объединение (join) — точка на диаграмме операций, в которой несколько параллельных потоков управления синхронизируются и объединяются.

Линия жизни (lifeline) — линия, обозначающая период существования объекта на диаграмме последовательности.

Член (member) — атрибут или операция в рамках классификатора.

Слияние (merge) — точка на диаграмме операций, в которой сходятся несколько вариантов течения процесса.

Сообщение (message) — общение между объектами на диаграммах последовательности и взаимодействия, приводящее к доставке информации или запросу на обслуживание.

Модель (model) — центральный объект UML, состоящий из различных элементов и связей между ними, иерархически упорядоченных по пакетам.

Направленность (navigability) — указатель, какой из двух объектов, связанных отношением, осведомлен о другом. Осведомленность может быть как двунаправленной, так и однонаправленной.

Направленная ассоциация (navigable association) — ассоциация, характеризующаяся однонаправленностью.

Заметка (note) — текст, приводимый для более подробного объяснения диаграммы.

Выноска (note attachment) — пунктирная линия, соединяющая заметку с описываемым элементом.

Объект (object) — элемент, получающий или передающий информацию к операциям (в диаграмме операций) или участвующий в изображаемом на диаграмме сценарии (в диаграммах взаимодействия и последовательности). Объект является экземпляром заданного классификатора (класса, интерфейса или участника).

Операция (operation) — метод или функция, осуществляемая классификатором.

Пакет (package) — элемент, наделяющий модели иерархией.

Диаграмма пакетов (package diagram) — диаграмма классов, в которой отображены только пакеты и зависимости.

Параметр (parameter) — аргумент операции.

Частный (уровень видимости) (private) — уровень видимости, применяемый к атрибуту или операции, когда доступ к члену может быть осуществлен только из кода того же классификатора.

Процессор (processor) — компьютер или иное программируемое устройство на диаграмме внедрения, на котором может быть размещен код.

Защищенный (уровень видимости) (protected) — уровень видимости, применяемый к атрибуту или операции, когда доступ к члену может быть осуществлен только из кода того же классификатора или его подклассов.

Публичный (уровень видимости) (public) — уровень видимости, применяемый к атрибуту или операции, когда доступ к члену может быть осуществлен из любого кода.

Реализация (realization) — поддержка заданного интерфейса компонентом или классом.

Диаграмма последовательности (sequence diagram) — диаграмма, показывающая жизненный цикл объектов, включая сообщения, передаваемые для осуществления некоторой модели поведения.

Состояние (state) — представление отдельного состояния системы или подсистемы (что происходит; каковы значения данных) на диаграмме состояний.

Диаграмма состояний (state diagram) — диаграмма, описывающая состояния системы или подсистемы, переходы между ними, а также события, повлекшие эти переходы.

Статический (атрибут, операция) (static) — атрибут, существующий в единственной копии, разделяемой между всеми экземплярами классификатора или операция, осуществляемая над классификатором в целом, а не над его экземплярами.

Стереотип (stereotype) — модификатор элемента модели, иллюстрирующий некоторую невыразимую посредством стандартного UML характеристику. Стереотипы позволяют определить пользовательский диалект UML.

Подкласс (subclass) — класс, наследующий атрибуты и операции через отношение обобщения.

Дорожка (swimlane) — элемент диаграммы операций, показывающий какие части системы выполняют определенные операции. За все операции внутри дорожки ответственны объект, компонент или участник данной дорожки.

Переход (transition) — передача управления от одних операций, слияний или ветвлений другим в диаграмме операций или переход от одного состояния к другому в диаграмме состояний.

Пример использования (use case) — (в диаграмме примеров использования) действие, предпринимаемое системой в ответ на запрос от участника в диаграмме примеров использования.

Диаграмма примеров использования (use case diagram) — диаграмма, отображающая отношения между участниками и примерами использования.

Видимость (visibility) — модификатор атрибута или операции, показывающий каким участкам кода позволено осуществлять доступ к члену. Существуют следующие уровни видимости: публичный, защищенный и частный.

B.2.3 Общие руководства по UML

Рекомендуется обращаться к следующим популярным руководствам:

- Руководство по использования UML для построения реальных систем ([29], [30]);
- Что такое UML ([31]);
- Руководство по UML. Сложные преобразования ([32]);
- Вводное руководство по UML ([34]);
- Руководство по UML от компании Borland ([35]);

B.2.4 Подробные руководства по UML

Подробные руководства содержатся в публикациях:

- Моделирование объектов в UML. Введение в UML, поведенческое и углубленное моделирование ([35], [36], [37])

B.3 Расширяемый язык разметки (eXtensible Markup Language, XML)

B.3.1 Определение языка XML

XML — подмножество языка SGML, полностью описанное в настоящем стандарте. Главная задача данного языка — позволить обрабатывать в сети Интернет обобщенный SGML так же, как это сейчас имеет место в отношении HTML. XML был разработан для упрощения реализации и взаимодействия SGML и HTML (см. [28]).

B.3.2 Термины

Термины и их определения взяты из разных источников, в частности [21].

Приложение (application) — самостоятельная программа, выполняющая определенную функцию для пользователя. Примером может служить программа XMLwriter, позволяющая пользователю редактировать XML и прочие текстовые файлы.

Атрибут (attribute) — элемент разметки, вставляемый в начальные или пустые теги в форме «название_атрибута=значение_атрибута». Атрибуты служат для предоставления дополнительной информации об элементе, в дальнейшем интерпретируемой приложением.

Объявление списка атрибутов (ATTLIST) [attribute-list (ATTLIST) declaration] — раздел DTD, определяющий список атрибутов, доступных в XML-документе, каким элементам они принадлежат и каковы значения атрибута по умолчанию.

Браузер (browser) — приложение, позволяющее пользователю взаимодействовать с информацией, хранимой в разнообразных форматах в сети Интернет, частных сетях или локально. Большинство браузеров поддерживают множество форматов, включая XML, HTML и Java. Примером браузера является Microsoft Internet Explorer 5.

CDATA — символьная информация (текст), не нуждающаяся в разборе. Разметка с данной секцией не интерпретируется как разметка.

Символ (character) — (в XML) любой графический элемент из числа описанных как символы стандартами ISO/IEC 10646 или Юникод (эти стандарты идентичны). Кроме того, поддерживаемые символы могут включать в себя символы табуляции, возврата каретки и перевода на новую строку.

Дочерний элемент (child element) — элемент, вложенный в родительский элемент.

Каскадные таблицы стилей (CSS) (Cascading Style Sheets) — формальный язык описания внешнего вида документа, используемый для оформления HTML- и XML-документов.

Объявление типа документа (DOCTYPE) [document type (DOCTYPE) declaration] — инструкция, содержащая внутренний DTD или дающая ссылку на внешний DTD.

Определение типа документа (DTD) (Document Type Definition) — набор правил, описывающих структуру XML-документа. Документ должен соответствовать данным правилам, чтобы пройти проверку на соответствие схеме XML.

Пустой тег (empty tag) — элемент, не имеющий содержимого. В XML используется следующий синтаксис: «<имя></имя>» или «</имя>».

Завершающий тег (end tag) — тег, закрывающий элемент. Чтобы документ был составлен правильно, название закрывающего тега должно соответствовать названию открывающего. Синтаксис: «</имя>».

Тип элемента (element type) — название открывающего, закрывающего или пустого тегов.

Объявление типа ELEMENT (ELEMENT type declaration) — раздел DTD, определяющий какие элементы допустимо использовать внутри XML-документа и каким может быть содержание элементов.

Объявление ENTITY (ENTITY declaration) — раздел DTD, содержащий обозначения категорий и текст подстановки для каждого такого обозначения. Если текст или данные сохранены удаленно, то могут использовать внешние ссылки.

Внешнее DTD (external DTD) — DTD, описанное во внешнем файле, возможно, сохраненном удаленно.

Ссылка на обычную категорию (general entity reference) — категория, использованная в теле XML-документа. Синтаксис для обычных категорий: «&имя».

Язык гипертекстовой разметки (HTML) (HyperText Markup Language) — один из основных издательских языков сети Интернет. Состоит из набора предопределенных тегов, указывающих браузеру на способы отображения текстовой и графической информации конечному пользователю.

Внутреннее DTD (internal DTD) — DTD, описанное внутри XML-документа.

Java — платформенно-независимый объектно-ориентированный язык программирования с синтаксической структурой, аналогичной C++.

Разметка (markup) — специальный синтаксис, используемый внутри основного текста, обозначающий инструкции для форматирования для процессора или парсера. В XML начинается с символов «<» и «&».

Метаинформация (metadata) — определение или описание набора данных.

Название (name) — (в XML) имя элемента, атрибута или категории, начинающееся с буквы или одного из символов: «_», «:» и продолжающееся комбинацией букв, цифр и символов «.», «—», «_», «:».

Объявление NOTATION (NOTATION declaration) — раздел DTD, определяющий формат данных, отличных от XML, и указывающий на приложение, необходимое для интерпретации данных. Пример такого объявления: <!NOTATION gif PUBLIC "gif viewer">.

Ссылка на параметризованную категорию (parameter entity reference) — категория, использованная внутри DTD. Синтаксис для параметризированных категорий: «% имя».

Родительский элемент (parent element) — элемент, в который вложены другие (дочерние) элементы.

Парсер (parser) — приложение, обрабатывающее документ, отделяя текст от разметки. Интерпретирует разметку и определяет характеристики и содержимое документа. Пример парсера: программа проверки XML браузера Internet Explorer 5.

PCDATA — разобранные символьные данные. Такой текст не является разметкой, но разбирается парсером.

Инструкция по обработке (processing instruction) — инструкция, используемая в XML для внедрения информации, предназначенной для закрытого программного обеспечения.

Рекурсия (recursion) — способ определения объекта через ссылку на себя самого.

Корневой элемент (root element) — первый элемент в документе. Формирует основу дерева элементов документа.

Документ схемы (schema document) — XML-документ, определяющий структуру и содержимое других XML-документов похожим на DTD образом.

Начинающий тег (start tag) — тег, открывающий элемент. Чтобы документ был правильно составленным название открывающего тега должно соответствовать названию закрывающего. Синтаксис: «<имя>».

Тег (tag) — элемент разметки, содержащий имя документа. Теги бывают открывающими, закрывающими и пустыми.

Юникод (Unicode) — система для отображения, обмена и обработки текста, написанного на любом из множества поддерживаемых языков. Набор символов является многобайтовым и включает большое количество международных символов.

Неразобранный (элемент) (unparsed) — часть информационного объекта, содержащая информацию, которая не может быть распознана как текст или разметка.

Унифицированный индикатор ресурса (URI) (Uniform Resource Identifier) — общий набор всех имен и адресов, ссылающихся на некоторый ресурс.

Единый указатель ресурсов (URL) (Uniform Resource Locator) — URI, указывающий на ресурс в сети Интернет. URL бывают абсолютными («http://www.xmlwriter.net») и относительными («doc001.xml»).

Ограничение смысловой правильности (validity constraint) — правила, определенные в спецификации XML и применяемые в отношении XML-документа через DTD. Документ считается правильным, если он соответствует DTD и является правильно составленным.

W3C (World Wide Web Consortium) — Консорциум, ответственный за выпуск стандартов, относящихся к Интернет-технологиям.

Ограничение синтаксической правильности (well-formedness constraint) — правила, определенные в спецификации в отношении синтаксиса XML. Для существования документу достаточно быть правильно составленным (т. е. ему не обязательно обладать ограничениями смысловой правильности).

XML — язык расширяемой разметки. Является подмножеством SGML (языка стандартной обобщенной разметки), использующим структурированные смысловые теги. XML предлагает гибкий способ создания новых форматов данных и обмена данными и метаинформацией с другими пользователями или приложениями.

Объявление XML (XML declaration) — инструкция для обработки, располагаемая в первой строчке XML-документа и обязательная для включения в этот документ.

Спецификация XML (XML specification) — рекомендация, выпущенная W3C 10 февраля 1998 года в качестве официального стандарта по написанию XML-документов.

В.3.3 Общие руководства по XML

Рекомендуется обращаться к следующим популярным руководствам:

- Создание документа XML ([38], глава 3);
- Руководство по XML от W3school.com ([39]);
- Бесплатный краткий курс по XML ([40]).

В.3.4 Подробные руководства по XML

Подробные руководства содержатся в публикациях:

- Книга по XML DTD ([22]);
- Обучающие примеры с использованием XML ([38], главы 27—30).

В.4 Межплатформенное программное обеспечение

В.4.1 Определение межплатформенного программного обеспечения

Межплатформенное программное обеспечение представляет собой крайне простое микроядро — расширяемый набор библиотечных служб и функций, необходимый множеству сетевых приложений для правильной работы (см. [24]). Данный вид программного обеспечения управляет взаимодействием клиента и сервера, используя многополосную архитектуру (см. [25]). Межплатформенное программное обеспечение — это набор сетевых системных программ, располагающийся между прикладным программным обеспечением, операционной системой и транспортным сетевым уровнем и служащий для совместной обработки данных (см. [26]).

В.4.2 Термины

Термины и их определения взяты из различных источников, в частности [23].

Сервер приложений (application server) — серверная программа, обеспечивающая установку характерных для приложения программных компонентов таким образом, что они могут быть удаленно вызваны (обычно одним из методов вызова удаленного объекта).

Поддержка сохранности состояния бином (bean-managed persistence) — способ обеспечения долговременного сохранения состояния EJB-системы.

Байт-код (bytecode) — (в Java) платформенно-независимый исполняемый программный код.

Кластеризация (clustering) — объединение серверов для обеспечения сервиса определенного вида, обычно выполняемое в целях резервирования или улучшения качества сервиса.

Стандарт на компоненты (component standard) — определение способов взаимодействия компонентов программного обеспечения (в том числе распределения функций и межкомпонентного интерфейса). Для Java-приложений межплатформенного программного обеспечения стандарт на компоненты обычно включает в себя спецификации интерфейса, предоставляемого межплатформенным программным обеспечением, и интерфейса компонента, требуемого межплатформенным программным обеспечением.

Поддержка сохранности состояния контейнером (container-managed persistence) — способ обеспечения долговременного сохранения состояния объектных бинов в EJB-системе.

CORBA — стандарт, поддерживаемый консорциумом OMG.

COS Naming — стандарт CORBA для каталогов объектов.

Источник данных (data source) — (в спецификациях Java API для транзакций и соединения с базами данных для Java-приложений) постоянное хранилище данных. Обычно представляет собой базу данных, но может также быть объектом, обеспечивающим соединение с базой данных (например, программой-драйвером).

DCOM — технологический стандарт компании Microsoft для распределенного взаимодействия компонентов программного обеспечения (COM-компонентов).

Enterprise JavaBeans (EJB) — технологический стандарт компании Sun Microsystems для написания и поддержки серверных компонентов.

Объектный бин (entity bean) — EJB-компонент, который поддерживает состояние EJB-системы между сессиями и который может быть обнаружен в каталоге объектов по своему индивидуальному идентификатору.

Восстановление после сбоя (failover) — способность реагировать на повреждение компонента переключением на другой компонент.

IDL — язык описания интерфейсов распределенных объектов в рамках обобщенной архитектуры CORBA.

IIOP — межброкерный протокол для Интернет в рамках архитектуры CORBA для передачи вызовов распределенным объектам.

ISAPI — разработанный компанией Microsoft на языке C++ API для создания Интернет-приложений для информационных Интернет-серверов этой компании.

Java-интерфейс доступа к сервисам имен и каталогов (JNDI) (Java Naming and Directory Interface) — Java-стандарт API для доступа клиентов к службам каталогов, таким как LDAP, COS Naming и др.

Java API для транзакций [Java Transaction API (JTA)] — спецификация для определения транзакций, управляемых клиентом (компонентом), и построения драйверов транзакции источника данных.

JTS — сервис транзакций Java в архитектуре CORBA. JTS обеспечивает поставщику программного обеспечения возможность разрабатывать средства межплатформенного управления транзакциями.

JVM — виртуальная машина Java, интерпретирующая и исполняющая байт-код.

LDAP — протокол прикладного уровня для доступа к службе каталогов X.500.

Межплатформенное программное обеспечение (middleware) — программное обеспечение, устанавливаемое на сервере и работающее как приложение, управляющее работой шлюза, или в качестве маршрутизирующего моста между клиентскими частями программного обеспечения и источниками данных или другими серверами.

NSAPI — разработанный компанией Netscape на языке C API для создания Интернет-приложений для Интернет-серверов этой компании.

OMG — консорциум, занимающийся разработкой и продвижением стандартов объектно-ориентированного программирования.

OODB — объектно-ориентированная база данных.

OODBMS — система управления OODB.

ORB — брокер объектных запросов, являющийся основным компонентом маршрутизации запросов в архитектуре CORBA.

Пассивация (passivate) — перевод объекта в неактивное состояние, когда тот недоступен, таким образом, чтобы сохранялась возможность его возврата в активное состояние.

Сохранность состояния (persistence) — поддержание состояния неизменным в течение длительного времени, включающего в себя перерывы между сессиями

Поддержка пула объектов (pooling) — поддержание объектов, серверов, соединений и других ресурсов в состоянии быстрого доступа, чтобы его не приходилось создавать заново каждый раз, когда в этом возникает необходимость.

RMI — стандарт Java программного интерфейса вызова удаленных методов, определяющий способ сетевого вызова методов, работающих на других распределенных объектах (виртуальных машинах Java).

RMI через IIOP — реализация интерфейса RMI через протокол IIOP.

Сервлет (servlet) — Java-приложение, выполняемое на сервере.

Сессионный бин (session bean) — EJB-компонент, который, как правило, существует только в пределах текущей сессии и который порождается во время сессии для связи с определенным клиентом.

Объект-скелет (skeleton) — серверный компонент программного обеспечения, служащий для перехвата удаленных вызовов клиента к методам, выполняемым на сервере. Обычно данный объект автоматически генерируется специальным компилятором.

SQLJ — расширенный синтаксис Java для встраивания SQL-подобных команд в Java-программы.

Объект-заглушка (stub) — клиентский компонент программного обеспечения, служащий для направления удаленных вызовов на удаленный сервер и получающий соответствующие ответы. Обычно данный объект автоматически генерируется специальным компилятором.

Трехзвенная архитектура (three-tier) — архитектура, в которой удаленный клиент получает доступ к удаленному источнику данных через сервер приложений.

Менеджер транзакций (transaction manager) — компонент программного обеспечения, координирующий отдельные транзакции к множественным источникам данных таким образом, чтобы они выглядели как единая транзакция. Менеджер транзакций требует, чтобы у каждого источника данных был свой драйвер, участвующий в координации транзакций. Менеджер транзакций обычно обеспечивает возможность отображения результатов транзакций и их статистику.

Транзакционная операция (transactional) — операция, которая выполняется целиком и успешно или, если ее завершение стало невозможно вследствие сбоя, либо не приводит к изменению состояния (данных), либо повторяется заново после того, как причины сбоя устранены.

В.4.3 Общие руководства по межплатформенному программному обеспечению

Рекомендуется обращаться к следующим популярным руководствам:

- Пример JDBC- приложения ([41], главы 5 и 7);
- Руководство по сокетам в Java ([43], глава 3);
- Руководство по RMI ([43], глава 4);
- Руководство по CORBA ([43], глава 5);
- Пример JDBC-приложения ([43], глава 8);
- Простой пример использования COM и CORBA ([44], глава 2).

В.4.4 Подробные руководства по межплатформенному программному обеспечению

Подробные руководства содержатся в публикациях:

- Построение приложений с использованием COM ([42]);
- Внедрение COM-CORBA ([44], главы 7—11);
- Примеры взаимодействия COM-CORBA в ПО ([45]).

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 13374-1:2003	IDT	ГОСТ Р ИСО 13374-1—2011 «Контроль состояния и диагностика машин. Обработка, передача и представление данных. Часть 1. Общее руководство»
ИСО/МЭК 14750:1999	—	*

* Соответствующий национальный стандарт отсутствует. 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.

Примечание — В настоящей таблице использовано следующее условное обозначение степени соответствия стандартов:
- IDT — идентичные стандарты.

Библиография

[1] ISO 8601:2000, Data elements and interchange formats - Information interchange — Representation of dates and times
[2] ISO 8879:1986, Information processing — Text and office systems — Standard Generalized Markup Language (SGML)
[3] ISO/IEC 9075 (all parts), Information technology — Database languages — SQL
[4] ISO/IEC 9506 (all parts), Industrial automation systems — Manufacturing Message Specification
[5] ISO/IEC 9579:2000 Information technology — Remote database access for SQL with security enhancement
[6] ISO/IEC 10646:2003, Information technology — Universal Multiple-Octet Coded Character Set (UCS)
[7] ISO/IEC 10746 (all parts), Information technology — Open Distributed Processing — Reference Model
[8] ISO 13372, Condition monitoring and diagnostics of machines — Vocabulary
[9] ISO 13373-1, Condition monitoring and diagnostics of machines — Vibration condition monitoring — Part 1: General procedures
[10] ISO 13379, Condition monitoring and diagnostics of machines — General guidelines on data interpretation and diagnostic techniques
[11] ISO 13380, Condition monitoring and diagnostics of machines — General guidelines on using performance parameters
[12] ISO 13381-1, Condition monitoring and diagnostics of machines — Prognostics — Part 1: General guidelines
[13] IEEE 1451.2:1997, Smart Transducer Interface for Sensors and Actuators — Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats
[14] ISO 14830-1, Condition monitoring and diagnostics of machines — Tribology-based monitoring and diagnostics — Part 1: General guidelines
[15] ISO 17359, Condition monitoring and diagnostics of machines – General guidelines
[16] ISO 18436-1, Condition monitoring and diagnostics of machines — Requirements for training and certification of personnel — Part 1: Requirements for certifying bodies and the certification process
[17] ISO 18436-2, Condition monitoring and diagnostics of machines — Requirements for training and certification of personnel — Part 2: Vibration condition monitoring and diagnostics
[18] ISO/IEC 19500-2, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)
[19] ISO/IEC 19501:2005, Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2
[20] http://www.tabletuml.com/Help/UMLGlossary.htm
[21] http://xmlwriter.net/xml_guide/glossary.shtml
[22] http://www.xmlxperts.com/xmlbookdtd.htm
[23] http://www.javaworld.com/javaworld/jw-04-1999/jw-04-middleware.html
[24] http://www.cra.org/Policy/NGI/draft/mid2.html
[25] http://www.prolifics.com/docs/panther/html/gl_html/glossary.htm
[26] http://iishelp.web.cern.ch/IISHelp/iis/htm/core/iigloss.htm#GlossaryM
[27] http://www.omg.org/uml/
[28] http://www.w3.org/TR/2004/REC-xml-20040204
[29] http://www.sparxsystems.com.au/UML_Tutorial.htm
[30] http://www.sparxsystems.com.au/WhitePapers/The_Business_Process_Model.pdf
[31] http://pigseye.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/what_is_uml.htm
[32] http://www.objectmentor.com/resources/articles/cplxtrns.pdf
[33] http://www.cragsystems.co.uk/ITMUML/
[34] http://bdn.borland.com/article/0,1410,31863,00.html
[35] ftp://ftp.omg.org/pub/docs/omg/99-11-04.pdf
[36] ftp://ftp.omg.org/pub/docs/omg/00-01-05.pdf
[37] ftp://ftp.omg.org/pub/docs/omg/00-03-03.pdf
[38] Navarro A., White C. and Burkman L. Mastering™ XML, Sybex, 2000
[39] http://www.w3schools.com/xml/default.asp
[40] http://www.spiderpro.com/ebooks/kickstart_tutorial_xml.pdf
[41] Chan H. et al. E-Commerce: Fundamentals and Applications, John Wiley & Sons, 2001
[42] Box D. Essential COM, Addison Wesley, 1998
[43] Boger M. Java™ in Distributed System, John Wiley & Sons, 1999
[44] Rosen M. and Curtis D. Migrating CORBA and COM Applications, John Wiley & Sons, 1998
[45] Geraghty R. et al. COM-CORBA Interoperability, Prentice Hall PTR, 1999

УДК 534.322.3.08:006.354 ОКС 17.160 35.240.99 Т34

Ключевые слова: контроль состояния, диагностика, обработка данных, информационная схема, открытая архитектура, спецификации

Редактор *Б.Н. Колесов*
Технический редактор *В.Н. Прусакова*
Корректор *Л.Я. Митрофанова*
Компьютерная верстка *И.А. Налейкиной*

Сдано в набор 13.08.2012. Подписано в печать 10.09.2012. Формат 60 × 84 1/8. Гарнитура Ариал.
Усл. печ. л. 3,72. Уч.-изд. л. 3,25. Тираж 106 экз. Зак. 766.

ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru
Набрано во ФГУП «СТАНДАРТИНФОРМ» на ПЭВМ.
Отпечатано в филиале ФГУП «СТАНДАРТИНФОРМ» — тип. «Московский печатник», 105062 Москва, Лялин пер., 6.