ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 72217— 2025

Цифровая промышленность

УНИФИЦИРОВАННАЯ АРХИТЕКТУРА OPC

Часть 10

Программы

(IEC 62541-10:2020, NEQ)

Издание официальное

Москва
Российский институт стандартизации
2025

gost.e-ecolog.ru

Предисловие

1 РАЗРАБОТАН Ассоциацией «Цифровые инновации в машиностроении» (АЦИМ) и Федеральным государственным бюджетным учреждением «Российский институт стандартизации» (ФГБУ «Институт стандартизации»)

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 306 «Измерения, управление и автоматизация в промышленных процессах»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 28 октября 2025 г. № 1289-ст

4 Настоящий стандарт разработан с учетом основных нормативных положений международного стандарта МЭК 62541-10:2020 «Унифицированная архитектура OPC. Часть 10. Программы» (IEC 62541-10:2020 «OPC Unified Architecture — Part 10: Programs», NEQ)

5 ВВЕДЕН ВПЕРВЫЕ

*Правила применения настоящего стандарта установлены в статье 26 Федерального закона от 29 июня 2015 г. № 162-ФЗ «О стандартизации в Российской Федерации». 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.e-ecolog.ru

© Оформление. ФГБУ «Институт стандартизации», 2025

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

Содержание

1 Область применения 1
2 Нормативные ссылки 1
3 Термины, определения и сокращения 2
4 Концептуальные положения 2
5 Модель 8

Введение

Система стандартов в области цифровой промышленности имеет важное значение для управления процессами цифровой трансформации и создания умных (интеллектуальных) производств, основанных на интеграции и интероперабельности различных автоматизированных систем управления, технологического оборудования, промышленных роботов, средств оснащения, контроллеров и датчиков.

В настоящем стандарте определена информационная модель, связанная с описанием и управлением программами в открытой платформе взаимодействия OPC, а также основные понятия в этой области.

Настоящий стандарт является десятой частью серии стандартов под общим наименованием «Цифровая промышленность. Унифицированная архитектура OPC».

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Цифровая промышленность

УНИФИЦИРОВАННАЯ АРХИТЕКТУРА OPC

Часть 10

Программы

Digital industry. OPC Unified Architecture. Part 10. Programs

Дата введения — 2026—01—30

1 Область применения

Настоящий стандарт определяет информационную модель для описания и управления программами в соответствии с унифицированной архитектуры открытой платформы взаимодействия OPC. В стандарте приведены термины и определения, относящиеся к программным задачам, выполняемым серверами и устройствами путем реализации программного кода.

Настоящий стандарт является десятой частью серии стандартов под общим наименованием «Цифровая промышленность. Унифицированная архитектура OPC» и его необходимо применять совместно с другими частями указанной серии. Стандарт входит в систему стандартов в цифровой промышленности.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты:

ГОСТ Р 71808 Цифровая промышленность. Унифицированная архитектура OPC. Часть 3. Модель адресного пространства

ГОСТ Р 71809 Цифровая промышленность. Унифицированная архитектура OPC. Часть 4. Сервисы

ГОСТ Р 71810 Цифровая промышленность. Унифицированная архитектура OPC. Часть 5. Информационная модель

ГОСТ Р 72214 Цифровая промышленность. Унифицированная архитектура OPC. Часть 7. Профили

Примечание — При пользовании настоящим стандартом целесообразно проверить действие ссылочных стандартов в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет или по ежегодному информационному указателю «Национальные стандарты», который опубликован по состоянию на 1 января текущего года, и по выпускам ежемесячного информационного указателя «Национальные стандарты» за текущий год. Если заменен ссылочный стандарт, на который дана недатированная ссылка, то рекомендуется использовать действующую версию этого стандарта с учетом всех внесенных в данную версию изменений. Если заменен ссылочный стандарт, на который дана датированная ссылка, то рекомендуется использовать версию этого стандарта с указанным выше годом утверждения (принятия). Если после утверждения настоящего стандарта в ссылочный стандарт, на который дана датированная ссылка, внесено изменение, затрагивающее положение, на которое дана ссылка, то это положение рекомендуется применять без учета данного изменения. Если ссылочный стандарт отменен без замены, то положение, в котором дана ссылка на него, рекомендуется применять в части, не затрагивающей эту ссылку.

Издание официальное

3 Термины, определения и сокращения

3.1 Термины и определения

В настоящем стандарте применены следующие термины с соответствующими определениями:

3.1.1 **функция** (function): Программная задача, выполняемая сервером или устройством, обычно реализуемая путем выполнения компьютерного кода.

3.1.2 **конечный автомат** (finite state machine): Последовательность состояний и допустимых переходов между состояниями, а также причин и следствий этих переходов, которые определяют действия программы в терминах дискретных этапов.

3.1.3 **тип программы автомата состояний** (program state machine type): Определение типа программы, является подтипом конечного автомата.

3.1.4 **метод управления программой** (program control method): Метод, имеющий специфическую семантику, предназначенный для управления программой путем изменения ее состояния.

3.1.5 **вызов программы** (program invocation): Уникальный экземпляр объекта программы, существующий на сервере.

Примечание — Вызов программы отличается от других экземпляров объекта того же типа конечного автомата программы уникальным путем просмотра узла объекта.

3.2 Сокращения

В настоящем стандарте применены следующие сокращения:

FSM — конечный автомат (Finite state machine);

HMI — человеко—машинный интерфейс (Human—machine interface);

Int32 — тип данных, который представляет целое число со знаком, занимает 4 байта (32 бита) памяти;

OPC — открытая коммуникационная платформа программных технологий, обеспечивающих единый интерфейс для управления объектами автоматизации и технологическими процессами (Open Platform Communications);

UA — унифицированная архитектура (Unified Architecture);

UInt32 — тип данных, который представляет целое число без знака, занимает 4 байта (32 бита) памяти.

4 Концептуальные положения

4.1 Общие положения

В условиях функционирования цифрового производства интегрированные средства автоматизации управляют операциями посредством обмена данными и скоординированного вызова системных функций, как показано на рисунке 1.

Службы необходимы для осуществления обмена данными и вызова функций, составляющих работу системы. Эти функции могут быть задействованы через HMI, контроллеры сотовой связи или другие системы диспетчерского управления и сбора данных. OPC UA определяет методы и программы как совместимый способ объявления, обнаружения и запроса этих функций. Они обеспечивают механизм нормализации семантического описания, вызова и представления отчетов о результатах выполнения этих функций. Методы и программы совместно дополняют другие службы и типы объектов OPC UA, упрощая работу среды автоматизации с использованием иерархии клиент-сервер.

Методы и программы, моделирующие функции, имеют разные области применения, поведение, время жизни и сложности в OPC-серверах и базовых системах. Эти функции обычно не связаны с чтением или записью данных, которые выполняются с помощью набора служб атрибутов OPC UA.

Методы представляют собой базовые функции сервера, которые могут быть вызваны клиентом. Программы моделируют более сложные функции системы с отслеживанием состояния. Например, вызов метода может использоваться для выполнения вычислений или сброса счетчика. Программа используется для запуска и управления пакетным процессом, выполнения программы обработки деталей станка или управления загрузкой домена. Методы и механизмы их вызова описаны в ГОСТ Р 71808 и ГОСТ Р 71809.

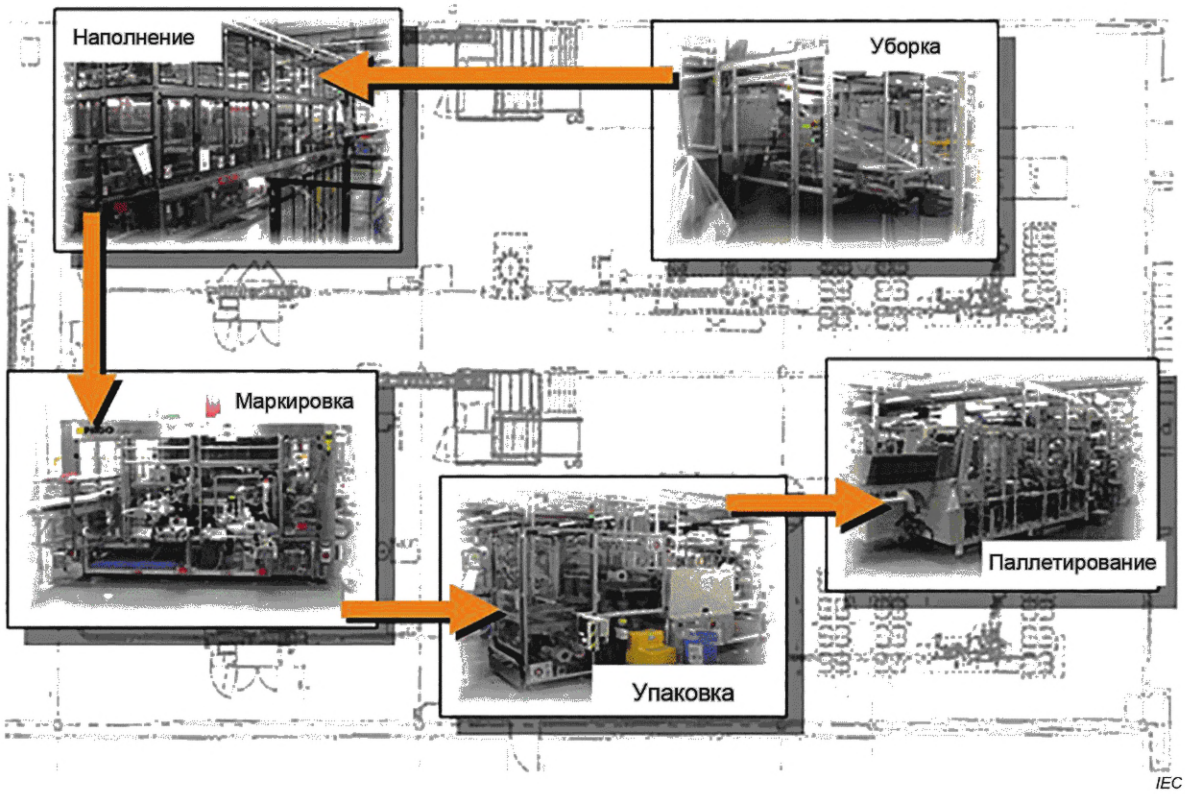


Рисунок 1 — Управление средствами автоматизации

В настоящем стандарте описываются расширения и конкретное использование основных возможностей, определенных в ГОСТ Р 71810, которые требуются для программ.

4.2 Программы

4.2.1 Общие положения

Программы — это сложные функции сервера или базовой системы, которые могут вызываться и управляться клиентом. Программы могут представлять любой уровень функциональности в системе или процессе, в котором требуется контроль или вмешательство клиента, а также мониторинг хода выполнения. Модель показана на рисунке 2.

Программы отслеживают состояние и переходят в заданную последовательность состояний по мере их выполнения. Их поведение определяется программным FSM. Элементы программного FSM описывают фазы выполнения программы в терминах допустимых переходов между набором состояний, стимулов или причин этих переходов и результирующих эффектов этих переходов.

4.2.2 Обеспечение безопасности

Программы могут использоваться для выполнения сложных алгоритмов управления или других действий, их использование должно ограничиваться персоналом с соответствующими правами доступа. Рекомендуется генерировать события метода обновления аудита, чтобы можно было отслеживать количество запущенных программ в дополнение к частоте их выполнения.

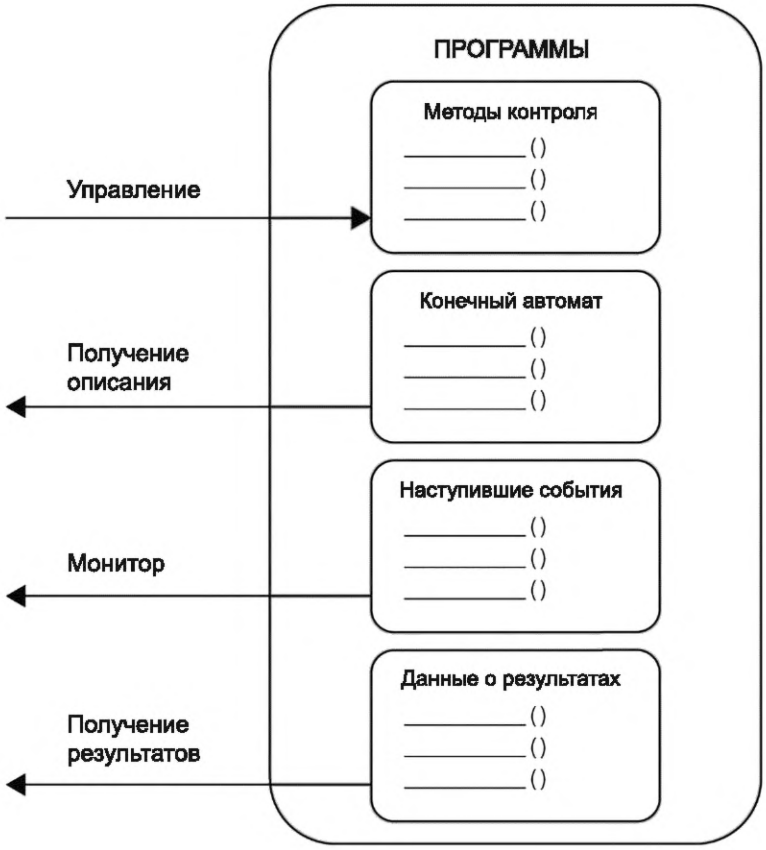


Рисунок 2 — Иллюстрация программы

4.2.3 Программный конечный автомат

В таблице 1 и на рисунке 3 представлены состояния, переходы, причины и следствия, из которых состоит программный FSM.

Таблица 1 — Описание переходов состояний программного FSM

Номер перехода	Название перехода	Причина	Исходное состояние	Конечное состояние	Эффект
1	Из «Остановлено» в «Готово»	Сброс	Остановлено	Готово	Отчет о переходе: 1 событие/результат
2	Из «Готово» в «Выполняется»	Старт	Готово	Выполняется	Отчет о переходе: 2 событие/результат
3	Из «Выполняется» в «Остановлено»	Проверка состояния или внутренняя причина (ошибка)	Выполняется	Остановлено	Отчет о переходе: 3 событие/результат
4	Из «Выполняется» в «Готово»	Внутренняя причина	Выполняется	Готово	Отчет о переходе: 4 событие/результат
5	Из «Выполняется» в «Приостановлено»	Приостановка	Выполняется	Приостановлено	Отчет о переходе: 5 событие/результат
6	Из «Приостановлено» в «Выполняется»	Возобновление	Приостановлено	Выполняется	Отчет о переходе: 6 событие/результат
7	Из «Приостановлено» в «Остановлено»	Остановка	Приостановлено	Остановлено	Отчет о переходе: 7 событие/результат

Окончание таблицы 1

Номер перехода	Название перехода	Причина	Исходное состояние	Конечное состояние	Эффект
8	Из «Приостановлено» в «Готово»	Внутренняя причина	Приостановлено	Готово	Отчет о переходе: 8 событие/результат
9	Из «Готово» в «Остановлено»	Метод остановки	Готово	Остановлено	Отчет о переходе: 9 событие/результат

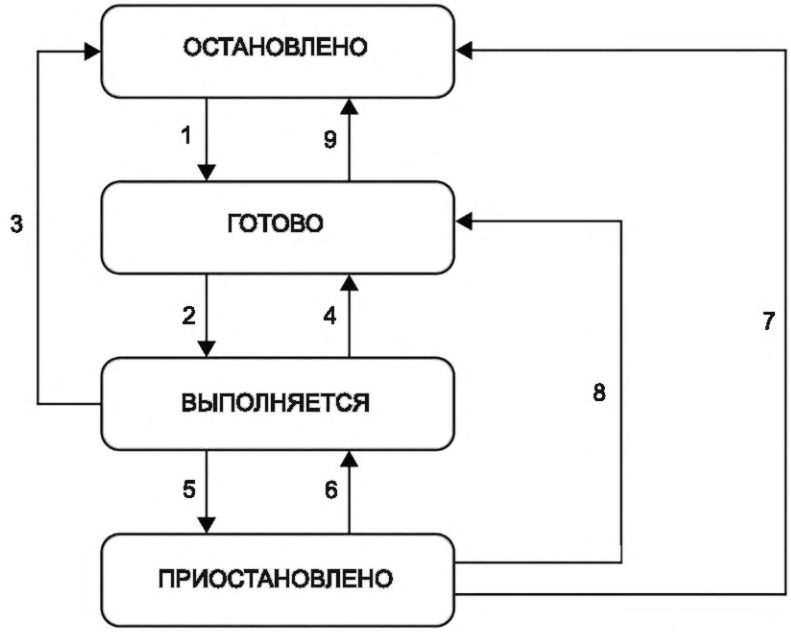


Рисунок 3 — Состояния и переходы программы

4.2.4 Состояния программы

Стандартный набор базовых состояний определен для программ как часть программного FSM. Эти состояния представляют этапы, на которых программа может существовать в данный момент времени, как их видит клиент. Данное состояние является текущим состоянием программы. Все программы должны поддерживать этот базовый набор. Программа может не требовать от клиента выполнения действий, вызывающих изменение ее состояния. Формально состояния программы определены в таблице 2.

Таблица 2 — Состояния программы

Состояние	Описание
Готово	Программа должным образом инициализирована и может быть запущена
Выполняется	Выполнение программы приближается к завершению
Приостановлено	Программа была остановлена до достижения конечного состояния, но может быть возобновлена
Остановлено	Программа находится в состоянии завершения или сбоя, и ее нельзя запустить или возобновить без сброса настроек

Набор состояний, определенных для описания программы, может быть расширен. Подсостояния программы могут быть определены для базовых состояний, чтобы обеспечить более полное описание процесса и описать причины и следствия дополнительных стимулов и переходов. Органы по стандартизации и отраслевые группы могут расширить базовую модель конечного состояния программы, чтобы она соответствовала различным отраслевым моделям. Например, состояние «Остановлено» мо-

жет включать в себя дополнительные параметры «Прервано» и «Завершено», чтобы указать, была ли функция успешно завершена до перехода в режим остановки. Например, переходные состояния, такие как «Запуск» или «Приостановка», также могут быть продолжением текущего состояния.

4.2.5 Переходы состояний

Для FSM программы определен стандартный набор переходов между состояниями. Эти переходы определяют допустимые изменения текущего состояния программы в терминах начального состояния и результирующего состояния. Переходы между состояниями определены в таблице 3.

Таблица 3 — Переходы между состояниями программы

Номер перехода	Название перехода	Начальное состояние	Результирующее состояние
1	Из «Остановлено» в «Готово»	Остановлено	Готово
2	Из «Готово» в «Выполняется»	Готово	Выполняется
3	Из «Выполняется» в «Остановлено»	Выполняется	Остановлено
4	Из «Выполняется» в «Готово»	Выполняется	Готово
5	Из «Выполняется» в «Приостановлено»	Выполняется	Приостановлено
6	Из «Приостановлено» в «Выполняется»	Приостановлено	Выполняется
7	Из «Приостановлено» в «Остановлено»	Приостановлено	Остановлено
8	Из «Приостановлено» в «Готово»	Приостановлено	Готово
9	Из «Готово» в «Остановлено»	Готово	Остановлено

4.2.6 Стимулы перехода в состояние программы

Стимулы или причины для изменения состояния программы могут быть внутренними или внешними для сервера. Завершение этапов обработки, обнаружение аварийного состояния или передача пакета данных являются примерами внутренних стимулов. Методы являются примером внешних стимулов. В настоящем стандарте определены стандартные методы, которые действуют как стимулы для управления программой.

4.2.7 Методы управления программой

Клиенты управляют программой, вызывая методы. Методы влияют на поведение программы, вызывая определенные переходы состояний. Переходы состояний определяют действия, выполняемые программой. Настоящий стандарт определяет набор стандартных методов управления программой. Эти методы предоставляют клиенту достаточные средства для запуска программы.

В таблице 4 перечислен набор определенных методов управления программой. Каждый метод вызывает переходы из заданных состояний и должен использоваться, когда программа находится в одном из этих состояний.

Отдельные программы могут дополнительно поддерживать любое подмножество методов управления программой. Например, некоторым программам может быть запрещено приостанавливать работу, и поэтому они не будут предоставлять методы приостановки и возобновления работы.

Программы могут поддерживать дополнительные пользовательские методы. Пользовательские методы не должны изменять поведение FSM базовой программы.

Таблица 4 — Методы программного управления

Название метода	Описание
Начало	Приводит к переходу программы из состояния готовности в состояние выполнения
Приостановка	Приводит к переходу программы из состояния выполнения в состояние приостановки
Возобновление	Приводит к переходу программы из состояния приостановки в состояние выполнения
Остановка	Приводит к переходу программы из состояния готовности, выполнения или приостановки в состояние остановки
Сброс	Приводит к переходу программы из состояния остановки в состояние готовности

Все методы управления программой определяются с их именем просмотра в типе FSM программы с помощью необязательного правила моделирования с использованием заполнителя. Как определено в ГОСТ Р 71808, это правило позволяет включать аргументы для этих методов в подтипы или экземпляры. Например, метод «Начало» может содержать аргумент «Опции», который определяет динамические параметры, используемые для определения поведения программы. Служба вызова метода, указанная в ГОСТ Р 71809, определяет статус возврата. Этот статус возврата указывает на успешное выполнение метода программного управления или причину его сбоя.

4.2.8 Эффекты перехода состояния программы

Изменение состояния программы имеет причину, а также приводит к эффекту. Эффект является побочным продуктом изменения состояния программы, который может использоваться клиентом для отслеживания хода выполнения программы. Эффекты могут быть внутренними или внешними. Внешним результатом перехода в другое состояние является генерация уведомления о событии. Каждый переход в другое состояние программы связан с уникальным событием. Эти события отражают прогрессию и траекторию программы через набор определенных состояний. Внутренними последствиями перехода состояния могут быть выполнение некоторых программных действий, таких как генерация данных.

4.2.9 Данные о результатах работы программы

4.2.9.1 Общие положения

Результирующие данные генерируются запущенной программой. Результирующие данные могут быть промежуточными или окончательными. Результирующие данные могут быть связаны с определенными переходами состояний программы.

4.2.9.2 Данные промежуточного результата

Данные промежуточного результата являются временными и генерируются программой в сочетании с нетерминальными переходами состояний. Элементы данных, которые составляют промежуточные результаты, определяются в соответствии с конкретными переходами состояний программы. Их значения актуальны только на уровне переходов.

Каждый переход в состояние программы может быть связан с различными элементами данных о результатах. В качестве альтернативы, набор переходов может совместно использовать элемент данных о результатах. Процент завершения является примером промежуточных данных о результатах. Значение процента завершения выводится, когда происходит переход в состояние, и доступно клиенту.

Клиенты получают промежуточные данные о результатах, подписываясь на события перехода состояния программы. События определяют элементы данных для каждого перехода. Когда происходит переход, сгенерированное событие передает полученные значения данных о результатах подписанным клиентам. Если ни один клиент не следит за работой программы, промежуточные данные о результатах могут быть удалены.

4.2.9.3 Данные о результатах работы терминала

Данные о результатах работы терминала — это окончательные данные, генерируемые программой при завершении ее выполнения. Общее время выполнения, количество созданных виджетов и обнаруженная неисправность являются примерами данных о результатах работы терминала. Когда программа переходит в конечное состояние, эти данные о результатах могут быть переданы клиенту с помощью события перехода. Данные о результатах работы терминала также доступны в программе для чтения клиентом после остановки программы. Эти данные сохраняются до тех пор, пока экземпляр программы не будет повторно запущен или удален.

4.2.9.4 Программы мониторинга

Клиенты могут отслеживать действия, связанные с выполнением программы. Эти действия включают в себя вызов методов управления, генерацию данных о результатах и прохождение программы через все ее состояния. Для вызовов методов и переходов между состояниями предусмотрены события аудита. Эти события позволяют вести учет клиентов, которые взаимодействовали с любой программой, и изменений состояния программы, возникших в результате этого взаимодействия.

4.2.10 Время жизни программы

4.2.10.1 Общие положения

У программ может быть разный жизненный цикл. Некоторые программы могут всегда присутствовать на сервере, в то время как другие создаются и удаляются. Создание и удаление могут контролироваться клиентом или могут быть ограничены локальными средствами.

Программа может быть создана клиентом. Если программа создана клиентом, то он может добавить программу на сервер. Для создания экземпляра программы используется метод создания объекта, определенный в ГОСТ Р 71808. Начальное состояние программы может быть «Остановлено» или «Готово». Например, для некоторых программ может потребоваться, чтобы ресурс стал доступен после его создания и до того, как он будет готов к запуску. В этом случае он будет инициализирован в состоянии остановки и перейдет в состояние готовности, когда ресурс будет доставлен.

Программа может быть удалена клиентом. Если программа удалена клиентом, то он может удалить экземпляр программы с сервера. Для удаления экземпляра программы используется служба удаления узлов, определенная в ГОСТ Р 71809. Программа должна находиться в состоянии остановки для удаления. Программа также может быть автоматически удалена. Автоматически удаляемая программа удаляет сама себя после завершения выполнения.

4.2.10.2 Экземпляры программы

Программы могут быть как с несколькими экземплярами, так и с одним экземпляром. Сервер может поддерживать несколько экземпляров программы, если эти экземпляры могут выполняться параллельно. Например, программа может определять метод запуска, который имеет входной аргумент, указывающий, с каким ресурсом работают ее функции. Затем запускается каждый экземпляр программы, определяющий использование различных ресурсов. Клиент может обнаружить все экземпляры программы, запущенные на сервере. Каждый экземпляр программы уникально идентифицируется на сервере и управляется клиентом независимо.

4.2.10.3 Повторное использование программ

Программы могут быть запущены один раз или многократно (повторно). Программа, запущенная один раз, будет оставаться в состоянии остановки на неопределенный срок после ее запуска. Обычным способом было бы удалить ее после проверки конечных результатов.

Программы, пригодные для повторного использования, могут иметь ограниченное или неограниченное количество циклов. Для этих программ может потребоваться этап сброса для перехода из состояния остановки в состояние готовности. Это позволяет пополнить ресурсы или повторно инициализировать параметры перед перезапуском программы. Метод программного управления «Сброс» запускает переход в это состояние и любые связанные с ним действия или эффекты.

5 Модель

5.1 Общие положения

Программная модель расширяет модели типа FSM и базового типа объекта, представленные в ГОСТ Р 71810. Каждая программа имеет определение типа, которое является подтипом типа FSM. Тип FSM программы описывает модель FSM, поддерживаемую любым программным вызовом этого типа. Тип FSM программы также определяет набор свойств, который характеризует конкретные аспекты поведения этой программы, такие как время жизни и повторное использование, а также определяет данные о результатах, которые выдаются программой.

Тип FSM базовой программы определяет стандартный FSM, указанный для всех программ. Сюда входят состояния, переходы, а также причины (методы) и следствия (события) перехода. Подтипы типа FSM базовой программы могут быть определены для расширения или более конкретной характеристики поведения отдельной программы, как показано на рисунке 4.

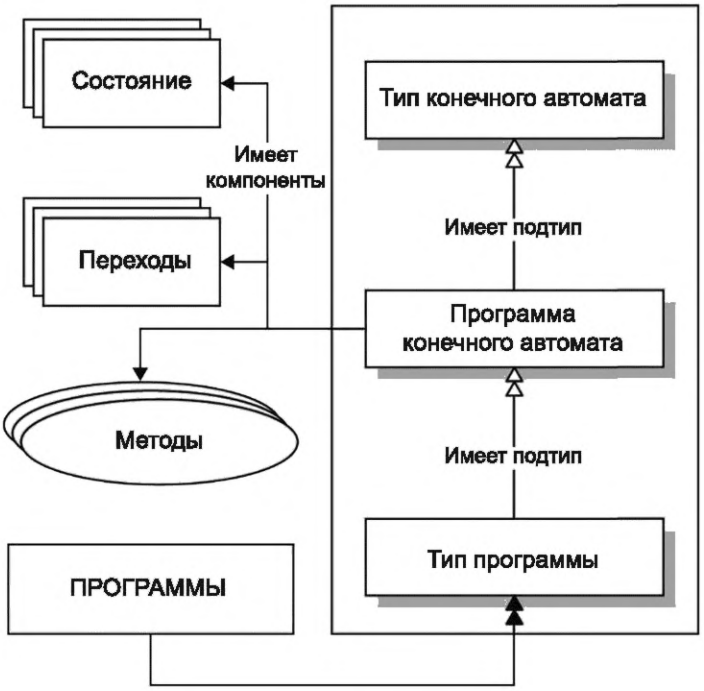


Рисунок 4 — Тип программы

5.2 Тип программы автомата состояний

5.2.1 Общие положения

Дополнительные свойства и компоненты, из которых состоит тип FSM программы, перечислены в таблице 5. Другим атрибутам или свойствам базового типа объекта или типа FSM не присваивается семантика, специфичная для типа FSM программы.

Таблица 5 — Тип программы автомата состояний

Атрибут	Значение				
	Включает все атрибуты, указанные для типа FSM				
Имя просмотра	Тип программы автомата состояний				
Абстрактный	Ложь				
—	—				
Рекомендации	Класс узла	Имя просмотра	Тип данных	Определение типа	Правило моделирования
Имеет свойство	Переменная	Поддающийся созданию	Boolean	Тип собственности	—
Имеет свойство	Переменная	Восхитительный	Boolean	Тип собственности	Обязательный
Имеет свойство	Переменная	Автоматическое удаление	Boolean	Тип собственности	Обязательный
Имеет свойство	Переменная	Количество повторных обращений	Int32	Тип собственности	Обязательный
Имеет свойство	Переменная	Количество экземпляров	UInt32	Тип собственности	—

Продолжение таблицы 5

Рекомендации	Класс узла	Имя просмотра	Тип данных	Определение типа	Правило моделирования
Имеет свойство	Переменная	Максимальное количество экземпляров	UInt32	Тип собственности	—
Имеет свойство	Переменная	Максимальное количество обращений	UInt32	Тип собственности	—
—	—	—	—	—	—
Имеет компонент	Переменная	Диагностика программы	Программная диагностика типа данных 2	Программная диагностика типа 2	Необязательный
—	—	—	—	—	—
Имеет компонент	Объект	Остановлено	—	Тип состояния	—
Имеет компонент	Объект	Готово	—	Тип состояния	—
Имеет компонент	Объект	Выполняется	—	Тип состояния	—
Имеет компонент	Объект	Приостановлено	—	Тип состояния	—
—	—	—	—	—	—
Имеет компонент	Объект	Из «Остановлено» в «Готово»	—	Переходный тип	—
Имеет компонент	Объект	Из «Готово» в «Выполняется»	—	Переходный тип	—
Имеет компонент	Объект	Из «Выполняется» в «Остановлено»	—	Переходный тип	—
Имеет компонент	Объект	Из «Выполняется» в «Готово»	—	Переходный тип	—
Имеет компонент	Объект	Из «Выполняется» в «Приостановлено»	—	Переходный тип	—
Имеет компонент	Объект	Из «Приостановлено» в «Выполняется»	—	Переходный тип	—
Имеет компонент	Объект	Из «Приостановлено» в «Остановлено»	—	Переходный тип	—
Имеет компонент	Объект	Из «Приостановлено» в «Готово»	—	Переходный тип	—
Имеет компонент	Объект	Из «Готово» в «Остановлено»	—	Переходный тип	—
—	—	—	—	—	—
Имеет компонент	Метод	Начало	—	—	Необязательно для заполнения
Имеет компонент	Метод	Приостановка	—	—	Необязательно для заполнения
Имеет компонент	Метод	Сброс	—	—	Необязательно для заполнения

Окончание таблицы 5

Рекомендации	Класс узла	Имя просмотра	Тип данных	Определение типа	Правило моделирования
Имеет компонент	Метод	Остановка	—	—	Необязательно для заполнения
Имеет компонент	Метод	Возобновление	—	—	Необязательно для заполнения
Имеет компонент	—	—	—	—	—
Имеет компонент	Объект	Данные об окончательном результате	—	Базовый тип объекта	Необязательный

5.2.2 Свойства типа программы автомата состояний

Свойство создания — это логическое значение, которое указывает, могут ли вызовы программы этого типа FSM создаваться клиентом. Если значение «Ложь», то эти вызовы программы являются постоянными или могут быть созданы только сервером.

Свойство удаления — это логическое значение, которое указывает, может ли клиент удалить вызов программы этого типа FSM. Если значение «Ложь», эти вызовы программы могут быть удалены только сервером.

Свойство автоматического удаления — это логическое значение, которое указывает, будут ли вызовы программы этого типа автоматически удаляться сервером после завершения выполнения. Если значение «Ложь», эти вызовы программы сохраняются на сервере до тех пор, пока они не будут удалены клиентом. Когда вызов программы удаляется, все результирующие данные, связанные с экземпляром, также удаляются.

Свойство подсчета переработки — это целое число без знака, которое указывает, сколько раз вызов программы такого типа был повторно выполнен или перезапущен с начальной точки (не возобновлялся). Обратите внимание, что для подготовки программы к перезапуску может потребоваться метод сброса.

Свойство максимального количества переработок — это целое число, которое определяет максимальное количество раз, когда вызов программы этого типа может быть повторно запущен или перезапущен с начальной точки (без возобновления). Если значение меньше 0, то количество перезапусков не ограничено. Если значение равно нулю, то программа не может быть повторно запущена или перезапущена повторно.

Свойство счетчика экземпляров — это целое число без знака, которое определяет количество вызовов программы этого типа, которые существуют в данный момент.

Свойство максимального количества экземпляров — это целое число, которое определяет максимальное количество вызовов программы данного типа, которые могут одновременно выполняться на данном сервере. Если значение меньше 0, то ограничений нет.

5.2.3 Компоненты типа программы автомата состояний

5.2.3.1 Общие положения

Компоненты типа машины состояний программы состоят из набора ссылок на объектные экземпляры типов состояний, типов переходов, типов событий и методов, которые в совокупности определяют конечную машину состояний программы.

На рисунке 5 показаны компонентные ссылки, определяющие ассоциации между двумя состояниями типа программы автомата состояний — «Готово» и «Выполняется». Дополнительные ссылочные типы опущены для упрощения иллюстрации.

5.2.3.2 Состояния типа программы автомата состояний

В таблице 6 указаны объекты состояний типа программы автомата состояний. Эти объекты являются экземплярами типа состояния, определенного в ГОСТ Р 71810. Каждому состоянию присваивается уникальное значение номера состояния. Подтипы типа машины состояния программы могут добавлять ссылки из любого состояния на подчиненный или вложенный объект машины состояния для расширения конечной машины состояния.

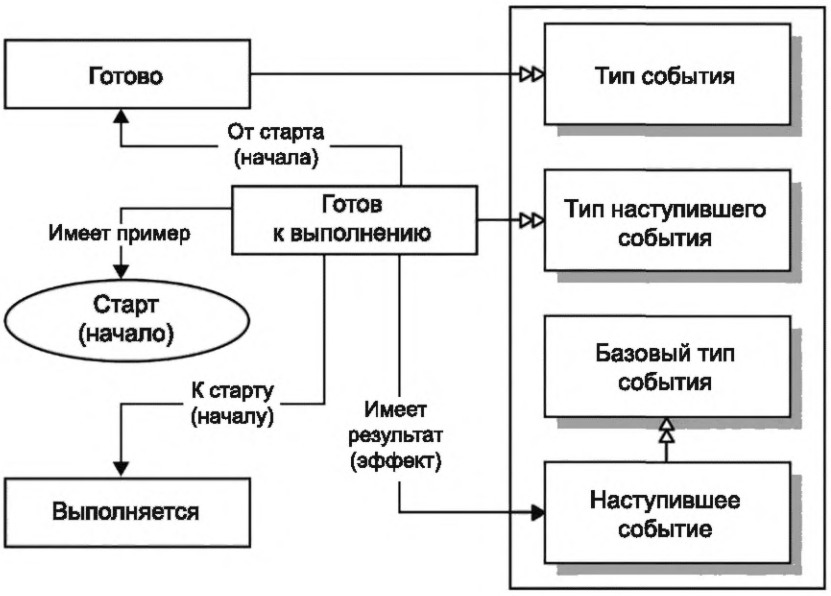


Рисунок 5 — Ссылки на программный FSM

Таблица 6 — Состояния программы

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Состояния					
Остановлено	Имеет свойство	Номер состояния	11	Тип свойства	—
—	К переходу	Из «Остановлено» в «Готово»	—	Тип перехода	—
—	Из перехода	Из «Выполняется» в «Остановлено»	—	Тип перехода	—
—	Из перехода	Из «Приостановлено» в «Остановлено»	—	Тип перехода	—
—	Из перехода	Из «Готово» в «Остановлено»	—	Тип перехода	—
—	—	—	—	—	—
Готово	Имеет свойство	Номер состояния	12	Тип свойства	—
—	Из перехода	Из «Остановлено» в «Готово»	—	Тип перехода	—
—	К переходу	Из «Готово» в «Выполняется»	—	Тип перехода	—
—	Из перехода	Из «Выполняется» в «Готово»	—	Тип перехода	—
—	К переходу	Из «Готово» в «Остановлено»	—	Тип перехода	—
—	—	—	—	—	—
Выполняется	Имеет свойство	Номер состояния	13	Тип свойства	—
—	К переходу	Из «Выполняется» в «Остановлено»	—	Тип перехода	—
—	К переходу	Из «Выполняется» в «Готово»	—	Тип перехода	—
—	К переходу	Из «Выполняется» в «Приостановлено»	—	Тип перехода	—
—	Из перехода	Из «Готово» в «Выполняется»	—	Тип перехода	—
—	Из перехода	Из «Приостановлено» в «Выполняется»	—	Тип перехода	—
—	—	—	—	—	—

Окончание таблицы 6

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Состояния					
Приостановлено	Имеет свойство	Номер состояния	14	Тип свойства	—
—	К переходу	Из «Приостановлено» в «Выполняется»	—	Тип перехода	—
—	К переходу	Из «Приостановлено» в «Остановлено»	—	Тип перехода	—
—	К переходу	Из «Приостановлено» в «Готово»	—	Тип перехода	—
—	Из перехода	Из «Выполняется» в «Приостановлено»	—	Тип перехода	—

Остановленное состояние — это состояние ожидания для программы. Это может быть начальное состояние или конечное состояние. В качестве начального состояния вызов программы не может начаться из-за условий на сервере. В качестве конечного состояния «Остановлено» может указывать либо на сбой, либо на завершение программы. Для определения характера завершения можно использовать подчиненное состояние или результат. Состояние «Приостановлено» ссылается на четыре объекта перехода, которые определяют допустимые переходы из состояния «Готово» в состояние «Готово» и из состояний «Запущено» и «Приостановлено».

Состояние готовности указывает на то, что программа готова к началу выполнения. Программы, которые готовы к запуску после их создания, могут немедленно перейти в состояние готовности. Состояние готовности ссылается на четыре объекта перехода, которые идентифицируют разрешенные переходы из состояния «Запущено» и «Остановлено», а также из состояний «Остановлено» и «Готово».

Запущенное состояние указывает на то, что программа активно выполняет свою функцию. Запущенное состояние ссылается на пять объектов перехода, которые определяют допустимые переходы из состояния «Остановлено», «Готово» и «Приостановлено», а также из состояния «Готово» и «Приостановлено».

Приостановленное состояние указывает на то, что программа прекратила выполнение своей функции, но сохраняет возможность возобновить выполнение функции с того момента, в котором она выполнялась при приостановке. Приостановленное состояние ссылается на четыре объекта перехода, которые идентифицируют разрешенные переходы из состояния готовности в состояние готовности, запущенное состояние, остановленное состояние и из состояния готовности.

5.2.3.3 Переходы типа конечного автомата программы

Переходы типа FSM программы являются экземплярами типа перехода, определенного в ГОСТ Р 71810, который также включает определения перехода в состояние, из состояния, имеет причину и имеет следствие, используемые ссылки. В таблице 7 указаны переходы, определенные для типа FSM программы. Каждому переходу присваивается уникальный номер перехода. В столбце «Примечания» указано, когда причина ссылается на методы, а когда последствия являются необязательными.

Таблица 7 — Переходы между программами

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Переходы					
Из «Остановлено» в «Готово»	Имеет свойство	Номер перехода	1	Тип свойства	—
—	К состоянию	Готово	—	Тип состояния	—
—	Из состояния	Остановлено	—	Тип состояния	—
—	Имеет причину	Перезагрузка	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—

Продолжение таблицы 7

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Переходы					
—	—	—	—	—	—
Из «Готово» в «Выполняется»	Имеет свойство	Номер перехода	2	Тип свойства	—
—	К состоянию	Выполняется	—	Тип состояния	—
—	Из состояния	Готово	—	Тип состояния	—
—	Имеет причину	Старт	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—
—	—	—	—	—	—
Из «Выполняется» в «Остановлено»	Имеет свойство	Номер перехода	3	Тип свойства	—
—	К состоянию	Остановлено	—	Тип состояния	—
—	Из состояния	Выполняется	—	Тип состояния	—
—	Имеет причину	Остановка	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—
—	—	—	—	—	—
Из «Выполняется» в «Готово»	Имеет свойство	Номер перехода	4	Тип свойства	—
—	К состоянию	Готово	—	Тип состояния	—
—	Из состояния	Выполняется	—	Тип состояния	—
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—
—	—	—	—	—	—
Из «Выполняется» в «Приостановлено»	Имеет свойство	Номер перехода	5	Тип свойства	—
—	К состоянию	Приостановлено	—	Тип состояния	—
—	Из состояния	Выполняется	—	Тип состояния	—
—	Имеет причину	Приостановка	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—
—	—	—	—	—	—

Окончание таблицы 7

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Переходы					
Из «Приостановлено» в «Выполняется»	Имеет свойство	Номер перехода	6	Тип свойства	—
—	К состоянию	Выполняется	—	Тип состояния	—
—	Из состояния	Приостановлено	—	Тип состояния	—
—	Имеет причину	Возобновление	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—
—	—	—	—	—	—
Из «Приостановлено» в «Остановлено»	Имеет свойство	Номер перехода	7	Тип свойства	—
—	К состоянию	Остановлено	—	Тип состояния	—
—	Из состояния	Приостановлено	—	Тип состояния	—
—	Имеет причину	Остановка	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—
—	—	—	—	—	—
Из «Приостановлено» в «Готово»	Имеет свойство	Номер перехода	8	Тип свойства	—
—	К состоянию	Готово	—	Тип состояния	—
—	Из состояния	Приостановлено	—	Тип состояния	—
—	Имеет причину	Перезагрузка	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—
—	—	—	—	—	—
Из «Готово» в «Остановлено»	Имеет свойство	Номер перехода	9	Тип свойства	—
—	К состоянию	Остановлено	—	Тип состояния	—
—	Из состояния	Готово	—	Тип состояния	—
—	Имеет причину	Остановка	—	—	Метод
—	Имеет эффект	Тип события «Переход программы»	—	—	—
—	Имеет эффект	Тип события «Переход программы аудита»	—	—	—

Переход от состояния «Остановлено» к состоянию «Готово» определяет переход из состояния «Остановлено» в состояние «Готово». Это может быть вызвано методом сброса.

Переход «Готово к работе» определяет переход из состояния «Готово» в состояние «Запущено». Это вызвано методом начала.

Переход из «Выполняется» в «Остановлено» определяет переход из состояния «Выполняется» в состояние «Остановлено». Он вызывается методом остановки.

Переход из «Выполняется» в «Готово» определяет переход из состояния «Выполняется» в состояние «Готово».

Переход из «Выполняется» в «Приостановлено» определяет переход из состояния «Выполняется» в состояние «Приостановлено». Он вызывается методом приостановки.

Переход из «Приостановлено» в «Выполняется» определяет переход из состояния «Приостановлено» в состояние «Выполняется». Он вызывается методом возобновления.

Переход из «Приостановлено» в «Остановлено» определяет переход из состояния «Приостановлено» в состояние «Остановлено». Он вызывается методом остановки.

Переход из «Приостановлено» в «Готово» определяет переход из состояния «Приостановлено» в состояние «Готово». Это вызвано внутренними причинами.

Переход из «Готово» в «Остановлено» определяет переход из состояния «Готово» в состояние «Остановлено». Он вызывается методом остановки.

Для каждого перехода по программе указаны две ссылки на имеющийся эффект. Эти эффекты представляют собой тип события «Переход программы» и тип события «Переход программы аудита», определенные в 5.2.5. Тип события «Переход программы» уведомляет клиентов о переходе программы и передает данные о результатах. Тип события «Переход программы аудита» используется для аудита переходов, которые являются результатом методов управления программой. Профиль «Аспект сервера аудита» в соответствии с ГОСТ Р 72214 требует поддержки типа события «Переход программы аудита».

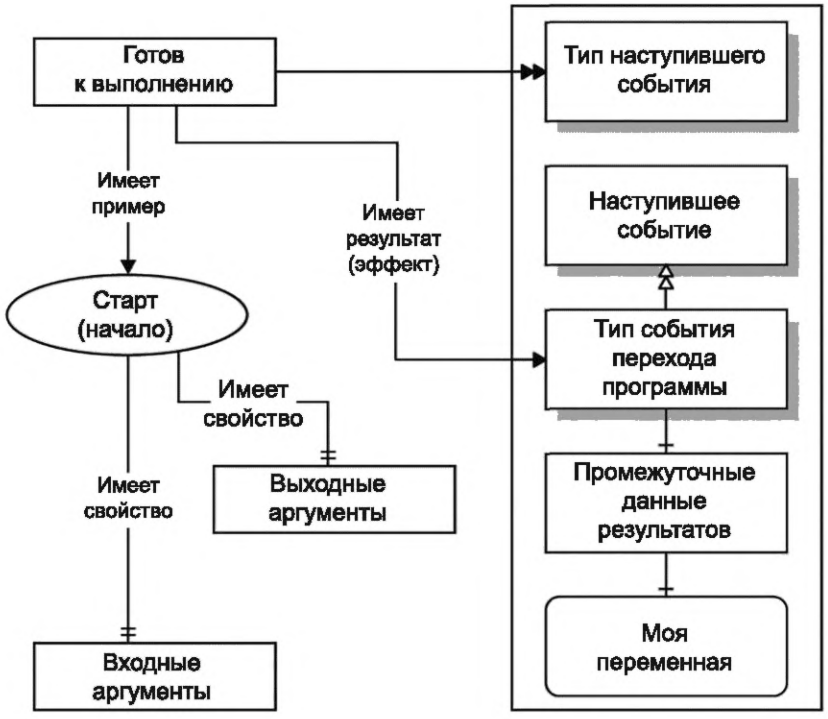


Рисунок 6 — Причины и следствия типа FSM программы

5.2.4 Причины (методы) типа конечного автомата программы

5.2.4.1 Общие положения

Тип FSM программы содержит ссылки на причины переходов между состояниями конкретной программы. Эти причины относятся к экземплярам методов. Программы, которые не поддерживают метод программного управления, будут исключать ссылку на причины, указывающие на этот метод, из ссылок на тип FSM программы. Если ссылка на причины метода не указана в типе FSM программы, клиент не сможет вызвать соответствующий переход состояния. Экземпляры метода, на которые ссылается тип FSM программы, идентифицируют входные аргументы и выводит аргументы, необходимые для вызо-

вов методов, для программных вызовов этого типа FSM программы. В таблице 8 указаны методы, определенные в качестве причин для типов FSM программы. На рисунке 6 показаны ссылки, связывающие компоненты и свойства методов и событий с программными переходами.

Таблица 8 — Причины типа FSM программы

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Причины					
Старт	Имеет свойство	Входные аргументы	—	Тип свойства	Необязательно
—	Имеет свойство	Выходные аргументы	—	Тип свойства	Необязательно
—	—	—	—	—	—
Приостановка	Имеет свойство	Входные аргументы	—	Тип свойства	Необязательно
—	Имеет свойство	Выходные аргументы	—	Тип свойства	Необязательно
—	—	—	—	—	—
Возобновление	Имеет свойство	Входные аргументы	—	Тип свойства	Необязательно
—	Имеет свойство	Выходные аргументы	—	Тип свойства	Необязательно
—	—	—	—	—	—
Остановка	Имеет свойство	Входные аргументы	—	Тип свойства	Необязательно
—	Имеет свойство	Выходные аргументы	—	Тип свойства	Необязательно
—	—	—	—	—	—
Сброс	Имеет свойство	Входные аргументы	—	Тип свойства	Необязательно
—	Имеет свойство	Выходные аргументы	—	Тип свойства	Необязательно
—	—	—	—	—	—

Метод «Старт» приводит к переходу от программы, готовой к выполнению.

Метод «Приостановка» приводит к переходу от запущенной программы к приостановленной. Метод «Возобновление» приводит к переходу от приостановленной программы к запущенной.

Метод «Остановка» приводит к остановке выполнения, метод «Приостановка» — к остановке или готовности к остановке программы в зависимости от текущего состояния программы.

Метод «Сброс» приводит к остановке и готовности программы к переходу.

5.2.4.2 Стандартные атрибуты

Атрибут исполняемого метода указывает, может ли метод выполняться в данный момент. Для методов управления программой это означает, что программа-владелец находится в текущем состоянии, которое поддерживает переход, вызванный методом.

5.2.4.3 Стандартные свойства

Методы могут ссылаться на набор входных аргументов. Для каждого типа FSM программы может быть определен набор входных аргументов для поддерживаемых методов управления программой. Данные, передаваемые в аргументах, дополняют информацию, необходимую программе для выполнения ее функции. Все вызовы метода программного управления для каждого вызова программы этого типа FSM программы должны передавать указанные аргументы.

Методы могут ссылаться на набор выходных аргументов. Для каждого типа FSM программы определен набор выходных аргументов для поддерживаемых методов управления программой. Все вызовы метода управления программой для каждого вызова этого типа FSM программы должны передавать указанные аргументы.

5.2.5 Эффекты (события) типа конечного автомата программы

5.2.5.1 Общие положения

Тип FSM программы содержит ссылки на компоненты, указывающие на эффекты каждого из переходов состояния программы. Эти эффекты являются событиями. Каждый переход должен иметь ссылку

на эффекты типа FSM программы и может иметь тип события перехода программы аудита. Когда происходит переход, для подписанных клиентов генерируются уведомления о событиях указанного типа. Вызов программы может служить средством оповещения об этих событиях, или объект-владелец или объект-сервер могут предоставлять уведомления.

Типы FSM программы предоставляют средства для предоставления данных о результатах и подтверждения переходов состояний для подписанных клиентов при каждом определенном переходе состояния программы. Тип события перехода программы аудита позволяет выполнять аудит изменений в состоянии программы в сочетании с вызовами клиентских методов.

5.2.5.2 Тип события перехода программы

Тип события перехода программы является подтипом типа события перехода. Он используется в программах для получения промежуточных или конечных результатов или других данных, связанных с переходом состояния. Программа может иметь уникальное определение типа события перехода программы для любого перехода. Каждый тип события перехода программы определяет данные промежуточного результата, относящиеся к определенному переходу состояния в этом типе FSM программы. Каждый переход может приводить к различным данным промежуточного результата. В таблице 9 указан тип события перехода программы.

Таблица 9 — Тип события перехода программы

Атрибут	Значение				
Имя для просмотра	Тип события перехода программы				
Аннотация	Правда				
Ссылки	Класс узла	Имя обзора	Тип данных	Определение типа	Правило моделирования
Подтип базового типа события перехода программы определен в ГОСТ Р 71810					
Имеет компонент	Объект	Промежуточные результаты	—	Базовый тип объекта	Необязательно

Номер перехода определяет переход программы, который вызвал событие. Номер состояния «От» определяет состояние до перехода программы.

Номер состояния «К» определяет состояние после перехода программы.

Промежуточный результат — это объект, который объединяет набор переменных, значения которых актуальны для программы в момент соответствующего перехода. Тип объекта для промежуточного результата определяет набор переменных с использованием набора ссылок на имеющийся компонент.

В таблице 10 указаны типы событий перехода программы, которые указаны для типов FSM программ.

Таблица 10 — События перехода программы

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Эффекты					
Переход события из «Остановлено» в «Готово»	—	—	—	—	—
—	Имеет свойство	Переходный номер	1	Тип свойства	—
—	Имеет свойство	От номера состояния	1	Тип свойства	—
—	Имеет свойство	К номеру состояния	2	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—

Продолжение таблицы 10

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Эффекты					
Переход события из «Готово» в «Выполняется»	—	—	—	—	—
—	Имеет свойство	Переходный номер	2	Тип свойства	—
—	Имеет свойство	От номера состояния	2	Тип свойства	—
—	Имеет свойство	К номеру состояния	3	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—
Переход события из «Выполняется» в «Остановлено»	—	—	—	—	—
—	Имеет свойство	Переходный номер	3	Тип свойства	—
—	Имеет свойство	От номера состояния	3	Тип свойства	—
—	Имеет свойство	К номеру состояния	1	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—
Переход события из «Выполняется» в «Готово»	—	—	—	—	—
—	Имеет свойство	Переходный номер	4	Тип свойства	—
—	Имеет свойство	От номера состояния	3	Тип свойства	—
—	Имеет свойство	К номеру состояния	2	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—
Переход события из «Выполняется» в «Приостановлено»	—	—	—	—	—
—	Имеет свойство	Переходный номер	5	Тип свойства	—
—	Имеет свойство	От номера состояния	3	Тип свойства	—
—	Имеет свойство	К номеру состояния	4	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—
Переход события из «приостановлено» в «Выполняется»	—	—	—	—	—
—	Имеет свойство	Переходный номер	6	Тип свойства	—

Окончание таблицы 10

Имя для просмотра	Рекомендации	Целевое имя для просмотра	Значение	Определение целевого типа	Примечания
Эффекты					
—	Имеет свойство	От номера состояния	4	Тип свойства	—
—	Имеет свойство	К номеру состояния	3	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—
Переход события из «Приостановлено» в «Остановлено»	—	—	—	—	—
—	Имеет свойство	Переходный номер	7	Тип свойства	—
—	Имеет свойство	От номера состояния	4	Тип свойства	—
—	Имеет свойство	К номеру состояния	1	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—
Переход события из «Приостановлено» в «Готово»	—	—	—	—	—
—	Имеет свойство	Переходный номер	8	Тип свойства	—
—	Имеет свойство	От номера состояния	4	Тип свойства	—
—	Имеет свойство	К номеру состояния	2	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно
—	—	—	—	—	—
Переход события из «Готово» в «Приостановлено»	—	—	—	—	—
—	Имеет свойство	Переходный номер	9	Тип свойства	—
—	Имеет свойство	От номера состояния	2	Тип свойства	—
—	Имеет свойство	К номеру состояния	1	Тип свойства	—
—	Имеет компонент	Промежуточные результаты	—	Тип объекта	Необязательно

5.2.6 Тип события перехода программы аудита

Тип события перехода программы аудита является подтипом типа события обновления состояния аудита. Он используется в программах для обеспечения средств аудита изменений состояния программы, связанных с любым методом управления программой, вызываемым клиентом. Серверы должны генерировать события перехода программы аудита, если они поддерживают профиль фасета сервера аудита, согласно ГОСТ Р 72214.

В таблице 11 указано определение типа события перехода программы аудита.

Таблица 11 — Определение типа события перехода программы аудита

Атрибут	Значение				
Имя для просмотра	Тип события перехода к программе аудита				
Аннотация	Правда				
Ссылки	Класс узла	Имя обзора	Тип данных	Определение типа	Правило моделирования
Подтип типа события состояния обновления аудита, определенного в соответствии с ГОСТ Р 71810					
Имеет компонент	Объект	Промежуточные результаты	UInt32	Тип свойства	Обязательно

Этот тип события наследует все свойства типа события состояния обновления аудита, определенного в ГОСТ Р 71810, за исключением указанных ниже.

Статус, указанный в ГОСТ Р 71810, определяет, был ли переход в состояние результатом вызова метода программного управления (присвоено значение «Правда») или нет (присвоено значение «Ложь»).

Исходное имя, указанное в ГОСТ Р 71810, идентифицирует метод, вызывающий переход программы, когда он является результатом вызываемого клиентом метода программного управления. Имя источника имеет префикс «Метод/», а имя метода управления программой — «Метод/Старт».

Идентификатор клиента/пользователя, указанный в ГОСТ Р 71810, идентифицирует пользователя/клиента, который запустил метод программного управления, если он связан с этим переходом состояния программы.

Отметка времени действия, указанная в ГОСТ Р 71810, определяет, когда произошел переход состояния программы, который привел к созданию события.

Свойство номера перехода — это переменная, которая идентифицирует переход, вызвавший событие.

5.2.7 Данные конечного результата

Тип объекта данных конечного результата определяет типы переменных, которые сохраняются после завершения работы программы. Тип объекта включает в себя компонент для типа каждой переменной, который содержит данные конечного результата.

5.2.8 Тип данных «Диагностика программы 2»

Эта структура содержит элементы, которые фиксируют активность вызова программы и могут быть использованы для диагностики программных проблем.

Исходный тип данных «Диагностика программы» содержал недостатки. Чтобы избежать коллизий с существующими реализациями, была создана новая версия с названием типа данных «Диагностика программы 2». Ее состав определен в таблице 12.

Таблица 12 — Структура типа данных «Диагностика программы 2»

Имя	Тип	Описание
Создание идентификатора сеанса	Идентификатор узла	Содержит идентификатор сеанса, в котором выполнен метод создания вызова программы
Создание имени клиента	Строка	Имя клиента сеанса, в котором был создан вызов программы
Время создания вызова	Время отправления	Определяет время, когда был создан вызов программы
Время последнего перехода	Время отправления	Определяет время последнего произошедшего перехода состояния программы
Метод последнего вызова	Строка	Идентифицирует последний программный метод, вызванный при вызове программы
Метод идентификатор последнего сеанса	Идентификатор узла	Содержит идентификатор сеанса, в котором был выполнен последний вызов метода управления программой

Окончание таблицы 12

Имя	Тип	Описание
Метод последних входных аргументов	Аргумент[]	Последние входные аргументы метода содержат входные аргументы при последнем вызове программного метода
Метод последних выходных аргументов	Аргумент[]	Содержит выходные аргументы при последнем вызове программного метода
Метод последнего ввода значений	Базовый тип данных[]	Сохраняет значения входных аргументов при последнем вызове программного метода. Размер и порядок этого списка соответствуют размеру и порядку поля входных аргументов
Метод последнего вывода значений	Базовый тип данных[]	Сохраняет значения выходных аргументов при последнем вызове программного метода. Размер и порядок этого списка соответствуют размеру и порядку поля выходных аргументов
Метод времени последнего вызова	Время отправления	Определяет время последнего вызова программы
Метод статуса последнего возврата	Код статуса	Сохраняет значение статуса возврата для последнего метода управления программой

Представление в адресном пространстве определено в таблице 13.

Таблица 13 — Определение типа данных «Диагностика программы 2»

Атрибуты	Значение
Просмотр имени	Тип данных «Диагностика программы 2»

5.2.9 Тип переменной «Диагностика программы 2»

Этот тип переменной объединяет простые переменные, используя простые типы данных, которые отражают элементы структуры типов данных «Диагностика программы». Ее переменные данных имеют ту же семантику, что и в 5.2.8.

Тип переменной определен в таблице 14.

Таблица 14 — Тип переменной «Диагностика программы 2»

Атрибут	Значение			
Имя для просмотра	Тип «Диагностика программы 2»			
Аннотация	Ложь			
Рекомендации	Класс узла	Имя для просмотра	Тип данных/ определение типа	Правило моделирования
Подтип типа базовой переменной данных определен в соответствии с ГОСТ Р 71810				
Имеет компонент	Переменная	Создание идентификатора сеанса	Идентификатор узла	Обязательно
Имеет компонент	Переменная	Создание имени клиента	Строка	Обязательно
Имеет компонент	Переменная	Время создания вызова	Время отправления	Обязательно
Имеет компонент	Переменная	Время последнего перехода	Время отправления	Обязательно
Имеет компонент	Переменная	Метод последнего вызова	Строка	Обязательно
Имеет компонент	Переменная	Метод идентификатора последнего сеанса	Идентификатор узла	Обязательно

Окончание таблицы 14

Рекомендации	Класс узла	Имя для просмотра	Тип данных/ определение типа	Правило моделирования
Имеет компонент	Переменная	Метод последних входных аргументов	Аргумент[]	Обязательно
Имеет компонент	Переменная	Метод последних выходных аргументов	Аргумент[]	Обязательно
Имеет компонент	Переменная	Метод последнего ввода значений	Базовый тип данных[]	Обязательно
Имеет компонент	Переменная	Метод последнего вывода значений	Базовый тип данных[]	Обязательно
Имеет компонент	Переменная	Метод времени последнего вызова	Время отправления	Обязательно
Имеет компонент	Переменная	Метод статуса последнего возврата	Код статуса	Обязательно

УДК 004.03:006.354

ОКС 25.040.40
35.100.05

Ключевые слова: унифицированная архитектура OPC, программы, модель описания, управление программами

Редактор *Е.Ю. Митрофанова*
Технический редактор *И.Е. Черепкова*
Корректор *И.А. Королева*
Компьютерная верстка *Л.А. Круговой*

Сдано в набор 29.10.2025. Подписано в печать 26.11.2025. Формат 60×84⅛. Гарнитура Ариал.
Усл. печ. л. 3,26. Уч.-изд. л. 2,77.

Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

Создано в единичном исполнении в ФГБУ «Институт стандартизации»
для комплектования Федерального информационного фонда стандартов,
117418 Москва, Нахимовский пр-т, д. 31, к. 2.

gost.e-ecolog.ru gost.e-ecolog.ru

gost.e-ecolog.ru