ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 18629-44 — 2011

Системы промышленной автоматизации и интеграция

ЯЗЫК СПЕЦИФИКАЦИЙ ПРОЦЕССА

Часть 44

Дефинициональные расширения: расширения ресурсов

ISO 18629-44:2006

Industrial automation systems and integration — Process specification language —

Part 44:

Definitional extension: Resource extensions

(IDT)

Издание официальное

Москва
Стандартинформ
2014

gost.e-ecolog.ru

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 Научно-техническим центром «ИНТЕК» на основе собственного аутентичного перевода на русский язык стандарта, указанного в пункте 4

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

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 22.12.2011 г. № 1613-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 18629-44:2006 «Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 44. Дефинициональные расширения: расширения ресурсов» (ISO 18629-44:2006 «Industrial automation systems and integration — Process specification language — Part 44: Definitional extension: Resource extensions»)

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

5 ВВЕДЕН ВПЕРВЫЕ

Информация об изменениях к настоящему стандарту публикуется в ежегодно издаваемом информационном 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

© Стандартинформ, 2014

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

Введение

ИСО 18629 – это комплекс стандартов на компьютерно-интерпретируемый обмен данными обеспечения технологического процесса. Все части комплекса стандартов ИСО 18629 устанавливают групповой язык программирования для описания конкретного технологического процесса, рассматриваемого как часть всего процесса изготовления изделия либо внутри одной промышленной компании, либо сразу в нескольких промышленных секторах (компаниях) вне его связи с какой-либо моделью компьютерного представления. Природа данного языка программирования такова, что он обеспечивает доступ к спецификациям технологического процесса и технологическим данным изделия на всех стадиях процесса его изготовления.

В настоящем стандарте установлены описания дефинициональных расширений языка программирования, относящихся к расширениям действий в соответствии с комплексом стандартов ИСО 18629.

Все части комплекса ИСО 18629 не связаны с какой-либо конкретной моделью компьютерного представления технологического процесса в рассматриваемом техническом приложении. Все вместе указанные части ИСО 18629 обеспечивают структурную технологическую взаимосвязь процессов производства для улучшения оперативной совместимости рассматриваемых технических приложений.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Системы промышленной автоматизации и интеграция
ЯЗЫК СПЕЦИФИКАЦИЙ ПРОЦЕССА
Часть 44
Дефинициональные расширения: расширения ресурсов

Industrial automation systems and integration. Process specification language. Part 44. Definitional extension. Resource extensions

Дата введения – 2012 – 09 – 01

1 Область применения

Настоящий стандарт устанавливает спецификацию непримитивных понятий языка программирования. При этом используется набор определений, написанных на языке, установленном в ИСО 18629. Данные определения устанавливают аксиомы для терминологии в соответствии с ИСО 18629.

Настоящий стандарт распространяется на:

- определения понятий, установленных в ИСО 18629-11, ИСО 18629-12 и ИСО 18629-14 и связанных с ресурсами, множествами ресурсов и соотношениями между ресурсами и действиями;

- определения понятий, установленных в ИСО 18629-11, ИСО 18629-12 и ИСО 18629-14, характеризующих соотношения между ресурсами и действиями.

2 Нормативные ссылки

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

ИСО/МЭК 8824-1 Информационные технологии. Нотация абстрактного синтаксиса версии 1 (ASN.1). Часть 1. Спецификация базовой нотации (ISO/IEC 8824-1, Information technology - Abstract Syntax Notation One (ASN.1) – Part 1: Specification of basic notation)

ИСО 15531-1 Системы промышленной автоматизации и интеграция. Управляющая информация промышленным производством. Часть 1. Общий обзор (ISO 15531-1, Industrial automation systems and integration – Industrial manufacturing management data – Part 1: General overview)

ИСО 18629-1: 2004 Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 1. Обзор и основные принципы (ISO 18629-1:2004, Industrial automation systems and integration – Process specification language – Part 1: Overview and basic principles)

ИСО 18629-11:2005 Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 11. Ядро PSL (ISO 18629-11:2005, Industrial automation systems and integration – Process specification language – Part 11: PSL core)

ИСО 18629-12 Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 12. Внешнее ядро (ISO 18629-12, Industrial automation systems and integration – Process specification language – Part 12: Outer core)

ИСО 18629-14 Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 14. Теории ресурсов (ISO 18629-14, Industrial automation systems and integration – Process specification language – Part 14: Resource theories)

3 Термины, определения и сокращения

3.1 Термины и определения

В настоящем стандарте применены следующие термины с соответствующими определениями:

3.1.1 **аксиома** (axiom): Точно сформулированное аналитическое выражение на формальном языке, устанавливающее ограничения к интерпретации символов в словаре языка.

[ИСО 18629-1]

3.1.2 **установленная лексика** (defined lexicon): Набор символов в нелогической лексике, обозначающих установленные понятия.

Примечание – Описываемая лексика включает константы, функции и символы соотношений.

Пример – Термины с консервативными определениями [ИСО 18629-1].

[ИСО 18629-1]

3.1.3 **дефинициональное расширение** (definitional extension): Расширение ядра PSL, представляющее новые лингвистические понятия, которые могут быть определены с помощью терминов ядра PSL.

Примечание – Дефинициональные расширения не добавляют выразительную силу ядру PSL и используются для подробного описания семантики и терминологии в области применения.

[ИСО 18629-1]

3.1.4 **расширение** (extension): Расширение ядра PSL, содержащее дополнительные аксиомы.

Примечание 1 – Ядро PSL представляет собой относительно простой набор аксиом, достаточный для представления широкого круга основных процессов. Однако для представления более сложных процессов требуются дополнительные ресурсы, отсутствующие в ядре PSL. Ядро PSL с каждым понятием следует использовать для описаний того или иного процесса, а для описания разнообразных модульных расширений следует использовать расширение и дополнения ядра PSL. В этом случае пользователь может использовать такой язык, который соответствует требованиям к выразительности.

Примечание 2 – Все расширения являются теориями ядра или дефинициональными расширениями.

[ИСО 18629-1]

3.1.5 **грамматика** (grammar): Правила совместного использования логических символов и словарных терминов для составления точно сформулированных аналитических выражений.

[ИСО 18629-1]

3.1.6 **язык** (language): Сочетание лексики и грамматики.

[ИСО 18629-1]

3.1.7 **лексика** (lexicon): Набор символов и терминов.

Примечание – Лексика состоит из логических (например, Булевы выражения и квантификаторы) и нелогических символов. В комплексе стандартов ИСО 18629 нелогическая часть лексики состоит из выражений (констант, функциональных символов и реляционных символов), необходимых для представления основных понятий онтологии.

[ИСО 18629-1]

3.1.8 **производство** (manufacturing): Функция или действие, предусматривающие перевод или превращение материала из сырья или заготовки в завершенное состояние.

[ИСО 15531-1]

3.1.9 **производственный процесс** (manufacturing process): Структурированный комплекс видов деятельности или работ, выполняемых с материалом для перевода его из сырья или заготовки в завершенное состояние.

Примечание – Производственные процессы могут быть представлены в виде технологической схемы процесса, схемы движения продукта, в виде табличной схемы или схемы фиксированного расположения. К планируемым производственным процессам могут относиться изготовление продукта для складирования, на заказ и для сборки на заказ и т.д., основанным на стратегическом использовании и размещении материально-производственных запасов.

[ИСО 15531-1]

3.1.10 **примитивная концепция** (primitive concept): Лексический термин, не имеющий консервативного определения.

[ИСО 18629-1]

3.1.11 **примитивная лексика** (primitive lexicon): Набор символов в нелогическом словаре, обозначающих элементарные понятия.

Примечание – Примитивная лексика включает в себя постоянные, функциональные и реляционные символы.

[ИСО 18629-1]

3.1.12 **процесс** (process): Структурированный ряд видов деятельности,

включающий в себя различные сущности предприятия, предназначенный и организованный для достижения конкретной цели.

Примечание – Данное определение аналогично определению, приведенному в ИСО 10303-49. Тем не менее ИСО 15531 нуждается в понятии структурированного набора деятельностей, без какого-либо предопределенного отношения ко времени или этапам. С точки зрения управления потоком некоторые свободные процессы могут требовать синхронизации в отношении цели, хотя в действительности они ничего не выполняют (задачи-призраки).

[ИСО 15531-1]

3.1.13 **ресурс** (resource): Любые устройство, инструмент и средства, за исключением сырья и компонентов конечной продукции, имеющиеся в расположении предприятия для производства товаров и услуг.

Примечание 1 – Рассматриваемое понятие ресурса адаптировано по отношению к ИСО 15531-1. Понятие ресурса, введенное в ИСО 15531-1, не включает сырьевые материалы, продукты и компоненты, являющиеся (с точки зрения системной теории) элементами окружающей среды и, таким образом, не являющиеся частью системы. В настоящем стандарте данное допущение снято. Более того, определение, принятое в ИСО 15531-1, во многом использует определение, принятое в ИСО 10303-49, при этом оно включается в определение, принятое в настоящем стандарте. В дополнение к понятию ресурса, принятому в ИСО 15531, понятие ресурса, принятое в настоящем стандарте, включает сырьевые и расходуемые материалы в соответствии с ИСО 18629-14.

Примечание 2 – Ресурсы в соответствии с приведенным выше определением включают также рабочую силу, рассматриваемую как особое средство с заданными возможностями и заданной производительности. Указанные средства рассматриваются как целесообразные для использования в процессе производства на основании технического задания. Данное определение не включает какого-либо моделирования индивидуального или группового поведения человеческого ресурса, за исключением его способности выполнять заданную работу в процессе производства (например, преобразование сырого материала или полуфабриката, обеспечение логистических услуг и т.п.). Это означает, что человеческие ресурсы, как и другие, рассматриваются с точки зрения их функций, их возможностей и их состояния (например, занят, свободен). При этом исключается какое-либо моделирование или представление какого-либо аспекта индивидуального или группового социального поведения.

[ИСО 15531-1]

3.1.14 **теория** (theory): Набор аксиом и определений, относящийся к данному понятию или набору понятий.

Примечание – Данное определение отражает подход искусственного интеллекта, где теория – это набор предположений, на которых основано значение соответствующего понятия.

[ИСО 18629-1]

3.2 Сокращения

- KIF (Knowledge Interchange Format – формат обмена знаниями).

4 Общая информация об ИСО 18629

Части с 41 по 49 комплекса международных стандартов ИСО 18629 определяют дефинициональные расширения, необходимые для формулировки точных определений и родственных аксиом непримитивных понятий ИСО 18629. Дефинициональные расширения определены ИСО 18629-11 и ИСО 18629-12, где вводятся новые элементы лексики. Данные элементы дефинициональных расширений могут быть полностью определены в соответствии с ИСО 18629-11 и ИСО 18629-12. Дефинициональные расширения дают точные семантические определения элементов, используемых в спецификациях индивидуальных технических приложений или типов технических приложений, при обеспечении совместных работ. Дефинициональные расширения существуют в следующих категориях:

- расширения действий;
- временные расширения и расширения, основанные на состоянии;
- упорядочивание действий и расширение продолжительности;
- назначения ресурса;
- наборы ресурсов;
- расширения действий обрабатывающей программы (процессора).

Индивидуальным (групповым) пользователям комплекса международных стандартов ИСО 18629 может потребоваться расширение ИСО 18629 для спецификации понятий, которое отсутствует в настоящее время в частях 41 – 49 комплекса международных стандартов ИСО 18629. Для этих целей они должны

использовать элементы, определенные в других частях комплекса международных стандартов ИСО 18629. Пользовательские расширения и их определения устанавливают дефинициональные расширения, которые не должны быть включены в части 41 – 49 комплекса международных стандартов ИСО 18629.

Примечание – Пользовательские расширения должны удовлетворять требованиям ИСО 18629 в соответствии с ИСО 18629-1:2004 (подразделы 5.1 и 5.2).

Части 41 – 49 комплекса международных стандартов ИСО 18629 распространяются на:

- семантические определения (на основе понятий, установленных в ИСО 18629-11 и ИСО 18629-12), элементы которых являются характерными для шести понятий, определенных выше;

- набор аксиом, ограничивающих использование элементов в дефинициональных расширениях.

Части 41 – 49 комплекса международных стандартов ИСО 18629 не распространяются на:

- определения и аксиомы для понятий, определенных ИСО 18629-11 и ИСО 18629-12;

- элементы, не определенные в соответствии с ИСО 18629-11 и ИСО 18629-12;

- пользовательские расширения.

5 Структура настоящего стандарта

В настоящем стандарте рассмотрены следующие дефинициональные расширения:

- назначение ресурса;
- согласованность действий, основанная на емкости ресурса;
- совместное использование ресурса;
- действия, обусловленные набором ресурсов;
- взаимозаменяемые ресурсы;
- гомогенные множества;
- набор материально-производственных ресурсов;
- объединенные ресурсы;
- действия обрабатывающей программы (процессора);
- пути ресурса.

Все требования, приведенные в настоящем стандарте, являются расширениями ИСО 18629-14, а также расширениями ИСО 18629-12 и ИСО 18629-11.

6 Назначение ресурса

Данный раздел характеризует определения, обусловленные назначением ресурса.

В контексте ИСО 18629 объектом рассмотрения являются ресурс и соответствующие действия (процессы), которые используют данный ресурс. Поэтому никакая аксиоматизация свойств ресурсов (например, рассмотрение дискретных и непрерывных ресурсов) не рассматривается. Назначение ресурса является одним из путей формализации, когда действию ставится в соответствие ресурс. Интуитивно ясно, что в основе аксиоматизации назначения ресурса лежит классификация взаимовлияния действий по отношению к ресурсу, который они делят. В особенности множество назначений ресурса, определенное в данном разделе, задает классификацию интерферирующих действий (когда эффекты одного действия искажают входные условия для другого действия).

Примечание – В контексте системной теории, соответствующей требованиям стандартов ИСО 15531, ресурсы являются средствами (находящимися в распоряжении системы) преобразования входного сигнала в выходной. Следовательно, их нельзя рассматривать как вход или выход системы. Они являются подмножеством ресурсов в соответствии с рассматриваемым здесь определением. Соответственно в стандартах типа ИСО 15531 они не ассоциированы с некоторым заданным действием. Ресурс может быть пустым и ассоциированным с действием А в процессе 1 или с действием В в процессе 2. Это очевидно для человеческих ресурсов, но также верно, например, и для какого-нибудь токарного станка, который может быть ассоциирован с действием «обточка цилиндра» в процессе 1 или с действием «сверление отверстия» в процессе 2. Тем не менее все определения в настоящем стандарте также могут быть использованы для ресурсов указанного типа.

6.1 Примитивная лексика назначения ресурса

Лексика назначения ресурса не требует никаких примитивных соотношений.

6.2 Определяемая лексика понятий для назначения ресурса

В данном разделе определены следующие соотношения:

- (reusable ?r ?a);

- (possibly_reusable ?r ?a);
- (renewable ?r ?a);
- (weakly_reusable ?r ?a);
- (consumable ?a);
- (possibly_consumable ?r ?a);
- (weakly consumable ?r ?a);
- (wearable ?r ?a).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

6.3 Теории ядра, обусловленные назначением ресурса

Для данного расширения необходимы:

- additive .th;
- requires.th;
- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

6.4 Дефинициональные расширения, обусловленные назначением ресурса

Назначения ресурса не требуют никаких дефинициональных расширений.

6.5 Определения понятий для назначения ресурса

Для назначения ресурса определены нижеследующие понятия.

6.5.1 reusable

Какой-либо ресурс ?r может быть использован повторно действием ?a, если какое-либо другое действие, также требующее ?r, может выполниться после завершения выполнения ?a в любом случае в будущем.

Пример – Повторно используемым ресурсом может быть станок, не требующий повторного запуска между действиями. Как только одно действие закончено, всегда

возможно выполнить и другое действие.

```
(forall (r ?a1 ?a2 ?a ?occ1 ?occ2) (iff (reusable ?r ?a1)
  (implies (and (common ?a1 ?a2 ?r)
      (subactivity ?a1 ?a)
      (subactivity ?a2 ?a)
      (occurrence_of ?occ2 ?a1))
    (forall (?b)
      (implies (forall (?occ3)
          (implies (and (subactivity_occurrence ?occ3 ?b)
              (occurrence_of ?b ?a)
              (precedes ?occ2 ?occ3))
            (poss ?a2 ?occ3)))))))
```

6.5.2 possibly_reusable

Какой-либо ресурс ?r может быть использован повторно действием ?a, если какое-либо другое действие, также требующее ?r, может быть выполнено после завершения действия ?a в некотором случае в будущем.

Пример – Повторно используемым ресурсом может быть станок, требующий некоторой загрузки между различными действиями. После выполнения одного действия возможно выполнение другого действия, но только после загрузки.

```
(forall (?r ?a1) (iff (possibly_reusable ?r ?a1)
(forall (?a2 ?occ1 ?occ2)
    (implies (and (common ?a1 ?a2 ?r)
        (subactivity ?a1 ?a)
        (subactivity ?a2 ?a)
        (occurrence_of ?occ2 ?a1))
      (exists (?b)
        (and (exists (?occ3)
            (and    (subactivity_occurrence ?occ3 ?b)
                (occurrence_of ?b ?a)
                (precedes ?occ2 ?occ3))
                (poss ?a2 ?occ3)))))))
```

6.5.3 renewable

Какой-либо ресурс ?r является обновляемым по отношению к некоторому действию ?a, если какое-либо другое действие, также требующее ?r, может быть выполнено после завершения действия ?a в любом случае в будущем при отсутствии помех.

Пример – Обновляемым ресурсом является солнечная батарея. Если ее заряд истощен, то всегда есть возможность в будущем перезарядить батарею (с помощью солнца), тогда ею снова можно будет пользоваться.

```
(forall (?r ?a1) (iff (renewable ?r ?a1)
(forall (?a2 ?occ1 ?occ2)
        (implies (and (common ?a1 ?a2 ?r)
                      (subactivity ?a1 ?a)
                      (subactivity ?a2 ?a)
                      (occurrence_of ?occ2 ?a1))
                 (forall (?b)
                    (implies (exists (?occ3)
                       (and      (subactivity_occurrence ?occ3 ?b)
                             (occurrence_of ?b ?a)
                             (precedes ?occ2 ?occ3))
                             (poss ?a2 ?occ3))))))
```

6.5.4 weakly_reusable

Какой-либо ресурс ?r является слабо используемым повторно действием ?a, если какое-либо другое действие, также требующее ?r, может быть выполнено после завершения ?a в любом случае в будущем, но при отсутствии помех.

Пример – Слабо используемый повторно ресурс появляется там, где имеются помехи для обновления ресурса. Например, кистью для рисования можно пользоваться повторно только после отмачивания ее в растворителе. В противном случае она становится непригодной.

```
(forall (?r ?a1) (iff (weakly_reusable ?r ?a1)
(forall (?a2 ?occ1 ?occ2)
        (implies (and (common ?a1 ?a2 ?r)
                      (subactivity ?a1 ?a)
                      (subactivity ?a2 ?a)
                      (occurrence_of ?occ2 ?a1))
```

```
(exists (?b)
    (and    (forall (?occ3)
                (implies (and (subactivity_occurrence ?occ3 ?b)
                              (occurrence_of ?b ?a)
                              (precedes ?occ2 ?occ3))
                         (poss ?a2 ?occ3))))))
```

6.5.5 consumable

Какой-либо ресурс ?r является потребляемым действием ?a, если какое-либо другое действие, также требующее ?r, не может быть выполнено после завершения ?a.

Пример – Потребляемым ресурсом являются дрова в костре или сырьевые материалы в производственном процессе.

```
(forall (?r ?a1) (iff (consumable ?r ?a1)
(forall (?a2 ?occ1 ?occ2)
        (implies (and (common ?a1 ?a2 ?r)
                      (subactivity ?a1 ?a)
                      (subactivity ?a2 ?a)
                      (occurrence_of ?occ2 ?a1))
                 (forall (?b)
                     (implies (forall (?occ3)
                                  (implies (and (subactivity_occurrence ?occ3 ?b)
                                                (occurrence_of ?b ?a)
                                                (precedes ?occ2 ?occ3))
                                           (not (poss ?a2 ?occ3))))))))))
```

6.5.6 possibly_consumable

Какой-либо ресурс ?r является возможно потребляемым по отношению к некоторому действию ?a1, если после завершения ?a1 возникает ситуация, в которой любое действие, требующее ?r, становится неосуществимым.

```
(forall (?r ?a1) (iff (possibly_consumable ?r ?a1)
(forall (?a2 ?occ1 ?occ2)
        (implies (and (common ?a1 ?a2 ?r)
                      (subactivity ?a1 ?a)
                      (subactivity ?a2 ?a)
                      (occurrence_of ?occ2 ?a1))
                 (exists (?b)
```

```
(and (exists (?occ3)
        (and (subactivity_occurrence ?occ3 ?b)
            (occurrence_of ?b ?a)
            (precedes ?occ2 ?occ3))
            (not (poss ?a2 ?occ3)))))))
```

6.5.7 weakly_consumable

Какой-либо ресурс ?r является слабо потребляемым по отношению к некоторому действию ?a1, если после завершения ?a1 всегда возможна ситуация в будущем, когда какое-либо другое действие, требующее ?r, становится невозможным.

Пример – Слабо потребляемым ресурсом является кисть для рисования. Если не положить ее в растворитель после использования, то любое действие в будущем, требующее данной кисти, становится невозможным.

```
(forall (?r ?a1) (iff (weakly_consumable ?r ?a1)
(forall (?a2 ?occ1 ?occ2)
        (implies (and (common ?a1 ?a2 ?r)
                (subactivity ?a1 ?a)
                (subactivity ?a2 ?a)
                (occurrence_of ?occ2 ?a1))
            (exists (?b)
                (and (forall (?occ3)
                        (implies (and (subactivity_occurrence ?occ3 ?b)
                                (occurrence_of ?b ?a)
                                (precedes ?occ2 ?occ3))
                            (not (poss ?a2 ?occ3)))))))
```

6.5.8 wearable

Какой-либо ресурс ?r является изнашиваемым по отношению к некоторому действию ?a1 тогда и только тогда, когда после завершения ?a1 всегда возможны ситуации в будущем, когда какое-либо другое действие, требующее ?r, становится невозможным.

Пример – Изнашиваемым ресурсом является бита дрели. В любом случае в будущем всегда возможна ситуация, когда бита будет изношена до предела и далее не может быть использована.

```
(forall (?r ?a1) (iff (wearable ?r ?a1)
```

```
(forall (?a2 ?occ1 ?occ2)
        (implies (and (common ?a1 ?a2 ?r)
                      (subactivity ?a1 ?a)
                      (subactivity ?a2 ?a)
                      (occurrence_of ?occ2 ?a1))
                 (forall (?b
                  (implies (exists (?occ3)
                                   (and (subactivity_occurrence ?occ3 ?b)
                                        (occurrence_of ?b ?a)
                                        (precedes ?occ2 ?occ3))
                                        (not (poss ?a2 ?occ3)))))))
```

7 Согласованность действий, основанная на емкости ресурса

Данный раздел характеризует все определения, обусловленные согласованностью действий, основанной на емкости ресурса.

7.1 Примитивная лексика согласованности действий, основанной на емкости ресурса

Лексика согласованности действий, основанной на емкости ресурса, не требует никаких примитивных соотношений.

7.2 Определяемая лексика понятий для согласованности действий, основанной на емкости ресурса

В данном подразделе определены следующие соотношения:

- (exclusive_use ?a ?r);
- (capacity_based ?a ?r);
- (unary_resource ?r);
- (capacitated ?r);
- (uniform_demand ?r ?q);
- (layput ?r ?a).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

7.3 Теории, обусловленные согласованностью действий, основанной на емкости ресурса

Для данной теории необходимы:

- additive .th;
- requires.th;
- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

7.4 Дефинициональные расширения, обусловленные согласованностью действий, основанной на емкости ресурса

Согласованность действий, основанная на емкости ресурса, не требует никаких дефинициональных расширений.

7.5 Определения согласованности действий, основанной на емкости ресурса

Для согласованности действий, основанной на емкости ресурса, определены нижеследующие понятия.

7.5.1 exclusive_use

Ресурс является эксклюзивным для некоторого действия тогда и только тогда, когда запрос ресурса эквивалентен его предоставлению.

```
(forall (?a ?r) (iff (exclusive_use ?a ?r)
(forall (?q1 ?q2 ?occ ?occp)
        (implies (and (do ?a ?occ ?occp)
                      (holds (demand ?a ?r ?q1) ?occ)
                      (holds (resource_point ?r ?q2) ?occ))
                 (= ?q1 ?q2))))))
```

Примечание – Если какой-либо ресурс является эксклюзивным по отношению к двум действиям, то эти два действия не могут быть совместными.

7.5.2 capacity_based

Ресурс определяется емкостью некоторого действия тогда и только тогда, когда запрос ресурса меньше его предоставления.

```
(forall (?a ?r) (iff (capacity_based ?a ?r)
(forall (?q1 ?q2 ?occ ?occp)
        (implies (and (Do ?a ?occ ?occp)
                      (holds (demand ?a ?r ?q1) ?occ)
                      (holds (resource_point ?r ?q2) ?occ))
                 (lesser ?q1 ?q2))))))
```

Примечание – Ресурс, определяемый емкостью, можно разделить между несколькими действиями.

7.5.3 unary_resource

Какой-либо ресурс является унарным тогда и только тогда, когда для всех действий, требующих данный ресурс, он является эксклюзивным.

```
(forall (?r) (iff (unary_resource ?r)
(forall (?a)
        (implies (res_requires ?a ?r)
                (exclusive_use ?a ?r))))))
```

7.5.4 capacitated_resource

Какой-либо ресурс имеет ограниченную пропускную способность тогда и только тогда, когда все действия, требующие данный ресурс, имеют ограниченную емкость.

```
(forall (?r) (iff (capacitated_resource ?r)
(forall (?a)
        (implies (res_requires ?a ?r)
                (capacity_based ?a ?r))))))
```

7.5.5 uniform_demand

Некоторое действие выполняет унифицированный запрос на какой-либо ресурс тогда и только тогда, когда запрос данного ресурса один и тот же во всех случаях.

```
(forall (?a ?r ?q) (iff (uniform_demand ?a ?r ?q)
(forall (?occ)
        (holds (demand ?a ?r ?q) ?occ)))))
```

7.5.6 layout

Некоторое действие является раскладочным для некоторого ресурса тогда и только тогда, когда его запрос зависит от эффектов других действий.

```
(forall (iff (layout ?r ?a)
(forall (?q ?occ1 ?occ2)
        (not (iff (holds (demand ?r ?a ?q) ?occ2)
(holds (demand ?r ?a ?q) ?occ1)))))))
```

Пример – Два действия при выпечке кексов могут быть совместными, если кексы размещаются по краю печи. Действия не могут быть совместными, если кексы находятся в середине печи.

8 Совместное использование ресурса

Данный раздел характеризует все определения, обусловленные совместным использованием ресурса.

Важно, что понятие количества, связанное с данным расширением, не зависит от понятия существования ресурса или понятия идентичности, также как и от различия между дискретным и непрерывным ресурсом. Главное – это доступность ресурса для будущих действий. Понятие предоставления ресурса ограничивает его доступность. Количество определяет совместное использование ресурса по отношению к нескольким совместным действиям. Предоставление ресурса определяет максимальный набор совместных действий, требующих данный ресурс. Таким образом, если количество равно нулю, то ни одно из действий, требующих данный ресурс, не представляется возможным.

8.1 Примитивная лексика совместного использования ресурса

Лексика совместного использования ресурса не требует никаких примитивных соотношений.

8.2 Определяемая лексика совместного использования ресурса

В данном подразделе определены следующие соотношения:

- (consumes_quantity ?s1 ?s2 ?a);
- (strict_consumes_quantity ?s1 ?s2 ?a);
- (produces_quantity ?occ1 ?occ2);
- (strict_produces_quantity ?occ);
- (uses_quantity ?occ);
- (creates ?occ);
- (destroys ?occ);
- (fixed_quantity ?r);
- (nonreplenishable ?r);
- (uses ?a ?r);
- (consumes ?a ?r);
- (strict_consumes ?a ?r);
- (produces ?a ?r);
- (strict_produces ?a ?r);
- (provides ?a ?r);

- (provides_quantity ?a ?r).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

8.3 Теории, обусловленные совместным использованием ресурса

Для данной теории необходимы нижеследующие расширения:

- additive.th;
- requires.th;
- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

8.4 Дефинициональные расширения, обусловленные совместным использованием ресурса

Совместное использование ресурса не требует никаких дефинициональных расширений.

8.5 Определения для совместного использования ресурса

Для совместного использования ресурса определены нижеследующие понятия.

8.5.1 consumes_quantity

Некоторое действие потребляет некоторое количество ?q какого-либо ресурса тогда и только тогда, когда запрос ресурса равен ?q и предоставление ресурса уменьшается на ?q после завершения действия.

```
(forall (?a ?r ?q) (iff (consumes_quantity ?a ?r ?q)
(forall (?q1 ?occ ?occ1 ?occ2)
        (implies (and (do ?a ?occ1 ?occ2)
                      (holds (demand ?a ?r ?q) ?occ1)
                      (holds (resource_point ?r ?q1) ?occ1))
                 (holds (resource_point ?r (- ?q1 ?q)) ?occ2)))))
```

8.5.2 strict_consumes_quantity

Некоторое действие строго потребляет некоторое количество какого-либо ресурса тогда и только тогда, когда данный ресурс является непополняемым.

```
(forall (?a ?r ?q) (iff (strict_consumes_quantity ?a ?r ?q)
(and (consumes_quantity ?a ?r ?q)
        (nonreplenishable ?r))))))
```

8.5.3 produces_quantity

Некоторое действие производит некоторое количество ?q какого-либо ресурса тогда и только тогда, когда предоставление ресурса увеличивается на ?q после завершения действия.

```
(forall (?a ?r ?q) (iff (produces_quantity ?a ?r ?q)
(forall (?q1 ?q2 ?occ ?occ1 ?occ2)
        (implies (and (do ?a ?occ1 ?occ2)
                (holds (resource_point ?r ?q1) ?occ1)
                (= ?q2 (plus ?q1 ?q)))
            (holds (resource_point ?r ?q2) ?occ2))))))
```

8.5.4 strict_produces_quantity

Некоторое действие строго производит некоторое количество какого-либо ресурса тогда и только тогда, когда не существует других действий, потребляющих какое-либо его количество.

```
(forall (?a ?r ?q) (iff (strict_produces_quantity ?a ?r ?q)
(exists (?q)
        (and (produces_quantity ?a ?r ?q)
                (not (exists (?a2 ?q2)
                        (and (subactivity ?a2 ?a)
                                (consumes_quantity ?a2 ?r ?q2)))))))))
```

8.5.5 uses_quantity

Действие использует некоторое количество ?q какого-либо ресурса тогда и только тогда, когда запрос ресурса равен ?q и предоставление ресурса не меняется при выполнении действия.

Пример – Например, в случае, когда для строительства нужно пять рабочих, так как число рабочих не меняется в процессе работы.

```
(forall (?a ?r ?q) (iff (uses_quantity ?a ?r ?q)
(forall (?q1 ?q2 ?q3 ?occ1 ?occ2)
        (implies (and (do ?a ?occ1 ?occ2)
                (holds (demand ?a ?r ?q) ?occ1))
                (holds (resource_point ?r ?q1) ?occ1)
                (holds (resource_point ?r ?q2) ?occ2)
            (= ?q2 ?q1)))
```

8.5.6 creates

Некоторое действие создает какой-либо ресурс тогда и только тогда, когда оно производит некоторое количество данного ресурса, и количество ресурса

до начала выполнения данного действия равнялось нулю.

```
(forall (?a ?r) (iff (creates ?a ?r)
(exists (?q1)
        (and (produces_quantity ?a ?r ?q1)
                (forall (?q2 ?occ)
                        (implies (and (occurrence_of ?occ ?a)
                                (prior (resource_point ?r ?q2) ?occ)
                        (= ?q2 zero_quantity)))))))
```

8.5.7 destroys

Некоторое действие уничтожает какой-либо ресурс тогда и только тогда, когда оно потребляет некоторое количество ресурса, и количество данного ресурса после завершения данного действия равно нулю.

```
(forall (?a ?r) (iff (destroys ?a ?r)
(exists (?q1)
        (and (consumes_quantity ?a ?r ?q1)
                (forall (?q2 ?occ)
                        (implies (and (occurrence ?occ ?a)
                                (prior (resource_point ?r ?q2) ?occp)
                        (= ?q2 zero_quantity)))))))
```

8.5.8 fixed_quantity

Какой-либо ресурс имеет фиксированное количество ?q тогда и только тогда, когда предоставление ресурса неизменно и одно и то же во всех случаях.

```
(forall (?r ?q) (iff (fixed_quantity ?r ?q)
(forall (?occ)
        (holds (resource_point ?r ?q) ?occ)))))
```

8.5.9 nonreplenishable

Какой-либо ресурс является непополняемым тогда и только тогда, когда предоставление ресурса не может увеличиваться после завершения действия, потребляющего данный ресурс.

```
(forall (?r) (iff (nonreplenishable ?r)
(forall (?a ?q1 ?q2 ?occ1 ?occ2 ?occ3)
        (implies (and (implies          (do ?a ?occ1 ?occ2)
                                        (holds (resource_point ?r ?q1) ?occ2))
        (precedes ?occ2 ?occ3)
        (holds (resource_point ?r ?q2) ?occ3)))
        (or (greater ?q1 ?q2)
            (= ?q1 ?q2)))))
```

8.5.10 uses

Некоторое действие использует какой-либо ресурс тогда и только тогда, когда оно использует некоторое количество данного ресурса.

```
(forall (?a ?r) (iff (uses ?a ?r)
(exists (?q)
        (uses_quantity ?a ?r ?q)))))
```

8.5.11 consumes

Некоторое действие потребляет какой-либо ресурс тогда и только тогда, когда оно потребляет некоторое количество данного ресурса.

```
(forall (?a ?r) (iff (consumes ?a ?r)
(exists (?q)
        (consumes_quantity ?a ?r ?q)))))
```

8.5.12 strict_consumes

Некоторое действие строго потребляет какой-либо ресурс тогда и только тогда, когда оно строго потребляет некоторое количество данного ресурса.

```
(forall (?a ?r) (iff (strict_consumes ?a ?r)
(exists (?q)
        (strict_consumes_quantity ?a ?r ?q))))
```

8.5.13 produces

Некоторое действие производит какой-либо ресурс тогда и только тогда, когда оно производит некоторое количество данного ресурса.

```
(forall (?a ?r) (iff (produces ?a ?r)
(exists (?q)
        (produces_quantity ?a ?r ?q))))
```

8.5.14 strict_produces

Некоторое действие строго производит какой-либо ресурс тогда и только тогда, когда оно строго производит некоторое количество данного ресурса.

```
(forall (?a ?r) (iff (strict_produces ?a ?r)
(exists (?q)
        (strict_produces_quantity ?a ?r ?q))))
```

8.5.15 provides_quantity

Некоторое действие обеспечивает некоторое количество какого-либо ресурса тогда и только тогда, когда существует поддействие, которое производит некоторое количество данного ресурса, и другое поддействие, которое потребляет некоторое количество данного ресурса.

```
(forall (?a ?r ?q) (iff (provides_quantity ?a ?r ?q)
(and (exists (?a1)
        (and (subactivity ?a1 ?a)
                (produces_quantity ?a1 ?r ?q)))
        (exists (?a2)
```

```
(and (subactivity ?a2 ?a)
     (consumes_quantity ?a2 ?r ?q))))))
```

8.5.16 provides

Некоторое действие обеспечивает какой-либо ресурс тогда и только тогда, когда оно обеспечивает некоторое количество какого-либо ресурса.

```
(forall (?a ?r) (iff (provides ?a ?r)
(exists (?q)
     (provides_quantity ?a ?r ?q))))
```

9 Действия, обусловленные набором ресурсов

Данный раздел характеризует все определения для действий, обусловленных набором ресурсов.

9.1 Примитивная лексика действий, обусловленных набором ресурсов

Лексика действий, обусловленных набором ресурсов, не требует никаких примитивных соотношений.

9.2 Определяемая лексика действий, обусловленных набором ресурсов

В данном подразделе определены следующие соотношения:

- (nondet_select ?a);
- (nondet_set_select ?a);
- (nondet_quantity_select ?a);
- (res_requires_set ?a);
- (res_requires_full_set ?a);
- (nondet_res_activity ?a).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

9.3 Теории для действий, обусловленных набором ресурсов

Для данной теории необходимы:

- res_set.th;
- additive .th;
- requires.th;
- act_occ.th;

- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

9.4 Дефинициональные расширения для действий, обусловленных набором ресурсов

Для данного расширения необходимо нижеследующее дефинициональное расширение:

- strong_poset.def.

9.5 Определения для действий, обусловленных набором ресурсов

Для действий, обусловленных набором ресурсов, определены нижеследующие понятия.

9.5.1 nondet_select

Некоторое действие является действием с недетерминированным выбором по отношению к какому-либо набору ресурсов ?r1 тогда и только тогда, когда выполнение данного действия эквивалентно выполнению поддействия, требующего какой-либо ресурс, являющийся элементом набора, ассоциированного с набором ?r1.

```
(forall (?a1 ?r1) (iff (nondet_select ?a1 ?r1)
(forall (?occ)
        (iff (occurrence_of ?occ ?a1)
                (exists (?r2 ?i ?a2 ?occ2)
                        (and (subactivity ?a2 ?a1)
                                (holds (resource_set ?i ?r1) (root_occ ?occ))
                                (holds (in ?r2 ?i) (root_occ ?occ))
                                (res_requires ?a2 ?r2)
                                (occurrence_of ?occ2 ?a2)
                                (subactivity_occurrence ?occ2 ?occ)))))))
```

9.5.2 nondet_set_select

Некоторое действие является действием с недетерминированным выбором набора по отношению к какому-либо набору ресурсов ?r1 тогда и только тогда, когда выполнение данного действия эквивалентно выполнению поддействия, требующего поднабор ресурсов, включающий элементы набора,

ассоциированного с ?r1.

```
(forall (?a1 ?r1) (iff (nondet_set_select ?a1 ?r1)
(forall (?occ)
        (iff (occurrence_of ?occ ?a1)
                (exists (?r2 ?a2)
                        (and (subactivity ?a2 ?a1)
                                (holds (resource_subset ?r2 ?r1) (root_occ ?occ))
                                (res_requires ?a2 ?r2)
                                (occurrence_of ?occ2 ?a2)
                                (subactivity_occurrence ?occ2 ?occ)))))))
```

9.5.3 nondet_quantity_select

Некоторое действие является действием с недетерминированным выбором количества по отношению к какому-либо набору ресурсов ?r тогда и только тогда, когда оно является действием с недетерминированным выбором набора ресурсов и количество выбранных элементов поднабора равно ?q.

```
(forall (?a ?r ?q) (iff (nondet_quantity_select ?a ?r ?q)
(and (nondet_set_select ?a ?r)
        (= ?q (cardinality ?r)))))
```

9.5.4 requires_set

Некоторое действие требует набор ресурсов ?r тогда и только тогда, когда каждое поддействие требует некоторый ресурс, являющийся элементом набора, ассоциированного с ?r.

```
(forall (?a ?r) (iff (res_requires_set ?a ?r)
(forall (?occ1)
        (iff (occurrence_of ?occ1 ?a)
                (forall (?a1 ?i)
                        (implies (and (holds (resource_set ?i ?r) (root_occ ?occ))
                                        (subactivity ?a1 ?a))
                                (exists (?r1 ?occ2 ?s2)
                                        (and (occurrence_of ?occ2 ?a1)
                                                (holds (in ?r1 ?i) (root_occ ?occ2))
                                                (res_requires ?a1 ?r1)
                                                (subactivity_occurrence ?occ1 ?occ1)))))))))
```

9.5.5 requires_full_set

Некоторое действие требует полный набор ресурсов ?r тогда и только тогда, когда каждый ресурс, являющийся элементом набора, ассоциированного с ?r, обусловлен некоторым поддействием.

```
(forall (?a ?r) (iff (res_requires_full_set ?a ?r)
(forall (?occ1)
        (iff (occurrence_of ?occ1 ?a)
                (forall (?r1)
                        (implies (holds (in_resource_set ?r1 ?r) (root_occ ?occ1))
                                (exists (?a1 ?occ2)
                                        (and (subactivity ?a1 ?a)
```

```
(res_requires ?a1 ?r1)
(occurrence_of ?occ2 ?a1)
(subactivity_occurrence ?occ2 ?occ1)))))))))
```

9.5.6 nondet_res_activity

Некоторое действие является действием с недетерминированным ресурсом тогда и только тогда, когда причина, по которой действие является недетерминированным, — это недетерминированный выбор по отношению к некоторому набору ресурсов.

```
(forall (?a) (iff (nondet_res_activity ?a)
(implies (choice_poset ?a)
        (exists (?r1)
                (nondet_select ?a ?r1)))))
```

10 Взаимозаменяемые ресурсы

Данный раздел характеризует все определения, обусловленные взаимозаменяемыми ресурсами.

10.1 Примитивная лексика взаимозаменяемых ресурсов

Лексика взаимозаменяемых ресурсов не требует никаких примитивных соотношений.

10.2 Определяемая лексика взаимозаменяемых ресурсов

В данном подразделе определены следующие соотношения:

- (superpose_select ?a);
- (homogeneous_set ?a);
- (set_contention ?a).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

10.3 Теории, обусловленные взаимозаменяемыми ресурсами

Для данной теории необходимы:

- res_set.th;
- additive .th;
- requires.th;
- act_occ.th;
- complex.th;
- subactivity.th;

- occtree.th;

-disc_state.th;

- psl_core.th.

10.4 Дефинициональные расширения, обусловленные взаимозаменяемыми ресурсами

Для данного расширения необходимы: set_action.def.

10.5 Определения для взаимозаменяемых ресурсов

Для взаимозаменяемых ресурсов определены нижеследующие понятия.

10.5.1 superpose_select

Некоторое действие ?a осуществляет выбор с суперпозицией по отношению к какому-либо набору ресурсов ?r тогда и только тогда, когда каждое поддействие недетерминировано выбирает поднабор ресурсов из ?r.

Пример – Некоторое действие, требующее и рабочего, и токарный станок, может выбрать нужный токарный станок из некоторого набора станков. Действия, выбирающие каждый токарный станок, осуществляют выбор с суперпозицией.

```
(forall (?a ?r) (iff (superpose_select ?a ?r)
(forall (?a1 ?occ1)
        (implies (and (occurrence_of ?occ1 ?a)
                      (subactivity ?a1 ?a)
                      (primitive ?a1))
                 (exists (?a2 ?r1 ?occ2)
                         (and (subactivity ?a1 ?a2)
                              (subactivity ?a2 ?a)
                              (occurrence_of ?occ2 ?a2)
                              (holds (resource_subset ?r1 ?r) (root_occ ?occ2))
                              (nondet_select ?a2 ?r1)))))))
```

10.5.2 homogeneous_set

Какой-либо набор ресурсов является гомогенным тогда и только тогда, когда он обусловлен действием, осуществляющим выбор с суперпозицией.

Пример – Для некоторого действия, требующего одного рабочего и один токарный станок, набор из трех рабочих является гомогенным, набор из четырех станков также является гомогенным. Вместе с тем набор из одного рабочего и одного станка не является гомогенным.

```
(forall (?a ?r) (iff (homogeneous_set ?r ?a)
(exists (?a2)
        (and (superpose_select ?a2 ?r)
             (subactivity ?a ?a2)))))
```

10.5.3 set_contention

Некоторое действие является конкурентным по отношению к какому-либо ресурсу ?r тогда и только тогда, когда входные условия для данного действия требуют, чтобы ?r был гомогенным набором и чтобы ?r был доступным.

```
(forall (?r ?s)
        (implies (poss (set_contention ?r) ?s)
                 (and (forall (?a)
                              (implies (subactivity ?a (set_contention ?a))
                                       (prior (homogeneous_set ?r ?a) ?s)))
                      (prior (available ?r (set_contention ?r)) ?s)))))
```

11 Гомогенные множества

Данный раздел характеризует все определения, обусловленные гомогенными множествами.

11.1 Примитивная лексика гомогенных множеств

Лексика гомогенных множеств не требует никаких примитивных соотношений.

11.2 Определяемые соотношения для гомогенных множеств

В данном подразделе определены следующие соотношения:

- (pile?a);
- (stock ?a);
- (pool?a);
- (pool_demand ?a);
- (uses_pile ?a ?r);
- (consumes_pile ?a ?r);
- (produces_pile ?a ?r).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

11.3 Теории ядра, обусловленные гомогенными множествами

Для данной теории необходимы следующие теории ядра:

- res_set.th;
- additive .th;
- requires.th;

- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

11.4 Дефинициональные расширения, обусловленные гомогенными множествами

Для данного расширения необходимы следующие дефинициональные расширения:

- subst_res.def;
- res_set_action.def;
- res_divisible.def.

11.5 Определения для гомогенных множеств

Для гомогенных множеств определены нижеследующие понятия.

11.5.1 pile

Какой-либо набор ресурсов является «множественным» по отношению к некоторому действию тогда и только тогда, когда он является гомогенным набором по отношению к данному действию, а предоставление ресурса эквивалентно количеству элементов набора.

```
(forall (?a ?r) (iff (pile ?r ?a)
(and (homogeneous_set ?r ?a)
        (forall (?q ?occ)
                (iff (prior (resource_point ?r ?q) ?occ)
                        (= ?q (cardinality ?i)))))
```

11.5.2 stock

Какой-либо набор ресурсов является «резервным» по отношению к некоторому действию тогда и только тогда, когда он является гомогенным набором по отношению к данному действию, существует какой-либо набор ресурсов с количеством элементов, равным запросу, а совокупный запрос равен количеству элементов набора.

```
(forall (?r ?a) (iff (stock ?r ?a)
(and (homogeneous_set ?r ?a)
        (forall (?q ?occ)
                (implies (prior (demand ?a ?r ?q) ?occ)
                        (exists (?i1 ?r1)
                                (and (= ?q (cardinality ?i1))
                                    (prior (resource_set ?i1 ?r1) ?occ)
                                    (res_requires_set ?a ?r1)))))
                        (forall (?q3)
                                (implies (prior (agg_demand ?r ?q3) ?occ)
                                        (= ?q3 (cardinality ?i)))))
```

11.5.3 pool

Какой-либо набор ресурсов является объединенным по отношению к некоторому действию тогда и только тогда, когда он является гомогенным набором по отношению к данному действию, а запрос объединенного ресурса равен количеству элементов поднабора.

```
(forall (?r ?a) (iff (pool ?r ?a)
(and (homogeneous_set ?r ?a)
        (forall (?q ?occ)
                (implies (prior (demand ?a ?r ?q) ?occ)
                        (exists (?i1 ?r1)
                                (and (subset ?i1 ?i)
                                        (= ?q (cardinality ?i1))
                                        (prior (resource_set ?i1 ?r1) ?occ)
                                        (res_requires_set ?a ?r1))))))
```

11.5.4 pool_demand

Объединенный ресурс используется некоторым действием тогда и только тогда, когда данное действие использует некоторое количество данного объединенного ресурса (необходимо помнить, что объединенные ресурсы сами являются ресурсами).

```
(forall (?a ?r ?q) (iff (pool_demand ?a ?r ?q)
(and (pool ?r ?a)
        (forall (?q1 ?occ)
                (implies (holds (demand ?a ?r ?q1) ?occ)
                        (= ?q ?q1))))
```

11.5.5 uses_pile

«Множество» используется некоторым действием тогда и только тогда, когда указанное действие использует некоторое количество данного объединенного ресурса (необходимо помнить, что объединенные ресурсы сами являются ресурсами).

```
(forall (?a ?r ?q) (if (uses_pile ?a ?r ?q)
(and (pile ?r ?a)
        (uses_quantity ?a ?r ?q))))
```

11.5.6 consumes_pile

«Множество» потребляется по отношению к некоторому действию тогда и только тогда, когда данное действие потребляет некоторое количество данного ресурса в объединенном ресурсе (необходимо помнить, что объединенные ресурсы сами являются ресурсами).

```
(forall (?a ?r ?q) (iff (consumes_pile ?a ?r ?q)
(and (pile ?r ?a)
        (consumes_quantity ?a ?r ?q)))))
```

11.5.7 produces_pile

«Множество» производится по отношению к некоторому действию тогда и только тогда, когда данное действие производит некоторое количество данного ресурса в объединенном ресурсе (необходимо помнить, что объединенные ресурсы сами являются ресурсами).

```
(forall (?a ?r ?q) (iff (produces_pile ?a ?r ?q)
(and (pile ?r ?a)
        (produces_quantity ?a ?r ?q)))))
```

12 Объединенные ресурсы

Данный раздел характеризует все определения, обусловленные объединенными ресурсами.

12.1 Примитивная лексика объединенных ресурсов

Лексика объединенных ресурсов не требует никаких примитивных соотношений.

12.2 Определяемая лексика объединенных ресурсов

В данном подразделе определены следующие соотношения:

- (resource_pool ?a);
- (conservative _pool ?a);
- (material_pool ?a).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

12.3 Теории, обусловленные объединенными ресурсами

Для данной теории необходимы:

- res_set.th;

- additive .th;
- requires.th;
- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

12.4 Дефинициональные расширения, обусловленные объединенными ресурсами

Нижеследующие дефинициональные расширения обусловлены объединенными ресурсами:

- homogeneous_setdef;
- subst_res.def;
- res_set_action.def.

12.5 Определения для объединенных ресурсов

Для объединенных ресурсов определены нижеследующие понятия.

12.5.1 resource_pool

Объединение ресурсов представляет собой какой-либо набор ресурсов, гомогенный по отношению к некоторому действию ?a. Его емкостные ограничения удовлетворяют следующим условиям:

- предоставление ресурса для объединения ресурсов равно количеству элементов ассоциированного набора;

- запрос объединения ресурсов некоторым действием равен количеству элементов поднабора ресурсов, требуемых данным действием;

- минимальная емкость объединения ресурсов эквивалентна минимальному количеству элементов ассоциированного набора.

Пример – При выполнении типовых производственных действий объединенные ресурсы являются множеством ресурсов повторного использования, таких, как набор токарных станков или набор отверток, которые могут быть выбраны для выполнения заданного действия.

```
(forall (?r ?a) (iff (resource_pool ?r ?a)
(forall (?i ?occ)
        (implies (holds (resource_set ?i ?r) ?occ)
                 (and (pile ?r ?a)
                      (pool ?r ?a)
                      (forall (?q3)
                              (implies (holds (min_capacity ?a ?r ?q3) ?occ)
                                       (lesser ?q3 (cardinality ?i)))))))))
```

12.5.2 conservative_pool

Объединение ресурсов является консервативным по отношению к некоторому действию тогда и только тогда, когда запрос на объединенный ресурс равен количеству, которое используется или потребляется.

```
(forall (?a ?r) (iff (conservative_pool ?a ?r)
(and (resource_pool ?r ?a)
        (forall (?q)
                (iff (pool_demand ?a ?r ?q)
                        (or (uses_pile ?a ?r ?q)
                                (consumes_pile ?a ?r ?q)))))))
```

12.5.3 material_pool

Объединение материалов представляет собой объединение ресурсов по отношению к некоторому действию тогда и только тогда, когда объединение ресурсов обусловливает количество по отношению к заданному действию.

```
(forall (?r ?a) (iff (material_pool ?r ?a)
(and (resource_pool ?r ?a)
        (exists (?q)
                (provides_quantity ?a ?r ?q))))
```

13 Набор материально-производственных ресурсов

Данный раздел характеризует все определения, обусловленные набором материально-производственных ресурсов.

13.1 Примитивная лексика набора материально-производственных ресурсов

Лексика набора материально-производственных ресурсов не требует никаких примитивных соотношений.

13.2 Определяемая лексика набора материально-производственных ресурсов

В данном подразделе определены следующие соотношения:

- (inventory_resource ?s1 ?s2 ?a);

- (inventory_pool ?s1 ?s2 ?a);

- (inventory_contains ?s1 ?s2 ?a).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

13.3 Теории, обусловленные набором материально-производственных ресурсов

Для данной теории необходимы:

- res_set.th;
- additive .th;
- requires.th;
- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

13.4 Дефинициональные расширения, обусловленные набором материально-производственных ресурсов

Нижеследующие дефинициональные расширения обусловлены набором материально-производственных ресурсов:

- homogeneous_set.def;
- subst_res.def;
- res_set_action.def;
- processor.def.

13.5 Определения для набора материально-производственных ресурсов

Для набора материально-производственных ресурсов определены нижеследующие понятия.

13.5.1 inventory_resource

Материально-производственный ресурс – это либо входной материал, либо выходной материал для некоторого действия.

```
        (forall (?r) (iff (inventory_resource ?r)
(exists (?a)
        (or (input_material ?r ?a)
          (output_material ?r ?a)))))
```

13.5.2 inventory_pool

Материально-производственный объединенный ресурс — это какой-либо набор ресурсов, являющийся гомогенным по отношению к некоторому действию ?a. Его емкостные ограничения удовлетворяют следующим условиям:

- предоставление объединенного ресурса равно максимальному количеству элементов ассоциированного набора;

- совокупный запрос объединенного ресурса перед выполнением некоторого действия равен количеству элементов набора, ассоциированного с ресурсом в данном состоянии;

- запрос объединенного ресурса некоторым действием равен количеству элементов набора ресурсов, обусловленного данным действием;

- минимальная емкость объединенного ресурса эквивалентна минимальному количеству элементов ассоциированного набора.

Пример – В типовом производственном случае объединенные материально-производственные ресурсы представляют собой резервные запасы материалов и прочие наборы входных и выходных материалов.

```
(forall (?r ?a) (iff (inventory_pool ?r ?a)
(forall (?i ?occ)
        (implies (holds (resource_set ?i ?r) ?occ)
                (and (inventory_resource ?r)
                        (homogeneous_set ?r ?a)
                        (forall (?q1)
                                (implies (holds (resource_point ?r ?q1) ?occ)
                                        (greaterEq ?q1 (cardinality ?i))))
                        (stock ?r ?a)
                        (forall (?a ?q4)
                                (implies (holds (min_capacity ?a ?r ?q4) ?occ)
                                        (lesserEq ?q4 (cardinality ?i)))))))))))
```

13.5.3 inventory_contains

Ресурс является содержащимся в объединенном материально-производственном ресурсе, если он входит в набор ресурсов, ассоциированный с данным материально-производственным ресурсом.

```
(forall (?r1 ?r2 ?occ)
        (iff (state (inventory_contains ?r1 ?r2) ?occ)
```

```
(exists (?a ?i)
        (and (inventory_pool ?r2 ?a)
             (holds (resource_set ?i ?r2) ?occ)
             (holds (in ?r1 ?i) ?occ))))))
```

14 Действия обрабатывающей программы (процессора)

Данный раздел характеризует все определения, обусловленные действиями процессора.

14.1 Примитивная лексика действий процессора

Лексика действий процессора не требует никаких примитивных соотношений.

14.2 Определяемая лексика действий процессора

В данном подразделе определены следующие соотношения:

- (processor_activity ?occ);
- (processor_resource ?r ?a);
- (input_material ?r ?a);
- (output_material ?r ?a).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

14.3 Теории, обусловленные действиями процессора

Для данной теории необходимы:

- additive .th;
- requires.th;
- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

14.4 Дефинициональные расширения, обусловленные действиями процессора

Для данного расширения необходимы: res_role.def.

14.5 Определения для действий процессора

Определены нижеследующие понятия для действий процессора, которые образуют подкласс действий, определенных по отношению к назначению ресурса.

14.5.1 processor_activity

Действие процессора представляет собой класс действий, использующих некоторый набор ресурсов, потребляющих некоторый другой набор ресурсов и производящий набор объектов.

Пример – Действия процессора являются типичными производственными процессами, особенно это относится к поддействиям планов технологического процесса и моделей логистических цепочек.

```
(forall (?a) (iff (processor_activity ?a)
(exists (?r1 ?r2 ?r3)
        (and (or (reusable ?r1 ?a)
                 (possibly_reusable ?r1 ?a))
        (or (consumable ?r2 ?a)
                 (possibly_consumable ?r2 ?a)
        (or (consumable ?r3 ?a)
                 (possibly_consumable ?r3 ?a)
(creates ?a ?r3)))))
```

14.5.2 processor_resource

Объект ?r является ресурсом процессора для некоторого действия ?a, если ?a является действием процессора, использующим ?r.

Пример – В типовом производственном случае ресурсом процессора является станок или инструмент.

```
(forall (?r ?a) (iff (processor_resource ?r ?a)
(and (processor_activity ?a)
        (reusable ?r1 ?a)
        (possibly_reusable ?r1 ?a))))
```

14.5.3 input_material

Объект ?r является входным материалом для некоторого действия ?a тогда и только тогда, когда ?a является действием процессора, потребляющим или возможно потребляющим ?r.

Пример – Ресурсы входных материалов определяются строго по назначению ресурса. В типовом производственном случае потребляемые ресурсы (например, сырьевые материалы) следует рассматривать как входные материалы.

```
(forall (?r ?a) (iff (input_material ?r ?a)
```

```
(and (processor_activity ?a)
        (or (consumable ?r ?a)
                (possibly_consumable ?r ?a))))))
```

14.5.4 output_material

Объект ?r является выходным материалом для некоторого действия ?a, если ?a – это действие процессора, производящее, потребляющее или возможно потребляющее ?r.

Пример – В типовом производственном случае выходной материал –это либо нечто созданное (например, печатная плата), либо нечто возможно потребляемое (например, некоторое сборочное приспособление, на которое устанавливаются вспомогательные детали).

```
(forall (?r ?a) (iff (output_material ?r ?a)
(and (processor_activity ?a)
        (or (creates ?a ?r)
(consumable ?r ?a)
        (possibly_consumable ?r ?a))))))
```

15 Пути ресурса

Данный раздел характеризует все определения, обусловленные путями ресурса.

15.1 Примитивная лексика путей ресурса

Лексика путей ресурса не требует никаких примитивных соотношений.

15.2 Определяемая лексика путей ресурса

В данном подразделе определены следующие соотношения:

- (next_processor_path ?a ?s);
- (pro_precedes ?a ?s);
- (resource_path ?a ?s)
- (initial_resource_path ?a ?s);
- (final_resource_path ?a ?s).

Каждое понятие обусловлено неформальной семантикой и некоторой аксиомой KIF.

15.3 Теории, обусловленные путями ресурса

Для данной теории необходимы:

- additive .th;
- requires.th;

- soo.th;
- act_occ.th;
- complex.th;
- subactivity.th;
- occtree.th;
- disc_state.th;
- psl_core.th.

15.4 Дефинициональные расширения, обусловленные путями ресурса

Для путей ресурса необходимы нижеследующие дефинициональные расширения:

- processor.def;
- res_role.def.

15.5 Определения для путей ресурса

Для путей ресурса определены нижеследующие понятия.

15.5.1 next_processor_path

Событие ?occ2, заключающееся в выполнении некоторого действия следует за событием ?occ1, заключающимся в выполнении поддействия процессора в некотором действии ?a тогда и только тогда, когда выходной материал для ?a1 является входным материалом для ?a2, и не существует поддействия процессора для ?a, потребляющего выходной материал из ?a1 и имеющего место между ?a1 и ?a2.

```
(forall (?occ1 ?occ2 ?a) (iff (next_processor_path ?occ1 ?occ2 ?a)
(and (next_subactivity ?occ1 ?occ2 ?a)
        (exists (?a1 ?a2 ?r)
                (and (occurrence_of ?occ1 ?a1)
                    (occurrence_of ?occ2 ?a2)
                    (processor_activity ?a1)
                    (processor_activity ?a2)
                    (output_material ?r ?a1)
                    (input_material ?r ?a2))))))
```

15.5.2 pro_precedes

pro_precedes — оператор частичного упорядочивания событий, заключающихся в выполнении поддействий процессора для ?a по отношению к потоку ресурсов.

```
(forall (?occ1 ?occ2 ?a) (iff (pro_precedes ?occ1 ?occ2 ?a)
(and (soo_precedes ?occ1 ?occ2 ?a)
        (forall (?occ3)
                (implies (and (soo_precedes ?occ1 ?occ3 ?a)
                        (soo_precedes ?occ3 ?occ2 ?a))
                    (exists (?occ4 ?occ5)
                      (and (next_processor_path ?occ4 ?occ3 ?a)
                              (next_processor_path ?occ3 ?occ5 ?a))))))))
```

15.5.3 resource_path

Некоторое действие является каким-либо путем ресурса тогда и только тогда, когда упорядочивание событий, заключающихся в выполнении поддействий эквивалентно упорядочиванию потока.

```
(forall (?a) (iff (resource_path ?a)
(forall (?occ1 ?occ2)
        (iff (soo_precedes ?occ1 ?occ2 ?a)
                (pro_precedes ?occ1 ?occ2 ?a))))
```

Пример – Пути ресурса включают понятия планов технологического процесса, маршрутизацию и потоки в логистических цепочках.

15.5.4 initial_resource_path

Рассматриваемым событием является инициирующее событие для некоторого действия ?a, если ?a есть какой-либо путь ресурса, а данное событие является начальным для упорядочивания событий.

```
(forall (?occ ?a) (iff (initial_processor_path ?occ ?a)
(and (resource_path ?a)
        (root_soo ?occ ?a))))
```

15.5.5 final_resource_path

Некоторое событие является заключительным событием процессора для некоторого действия ?a, если ?a есть какой-либо путь ресурса, а данное событие является конечным для упорядочивания событий.

```
(forall (?occ ?a) (iff (final_processor_path ?occ ?a)
(and (resource_path ?a)
        (leaf_soo ?occ ?a))))
```

Приложение А
(справочное)
ASN. 1 Идентификатор настоящего стандарта

Для однозначной идентификации информационного объекта в открытой системе настоящему стандарту присвоен следующий идентификатор:

iso standard 18629 part 44 version 1

Значение данного идентификатора определено в ИСО/МЭК 8824-1 и детально описано в ИСО 18629-1.

Приложение В
(справочное)
Пример описания технологического процесса в соответствии с настоящим стандартов

В данном приложении рассмотрен подробный сценарий использования языка спецификаций процесса PSL (Process Specification Language) в соответствии с ИСО 18629, а также программное описание конкретного технологического процесса.

Данный сценарий включает совместное выполнение нескольких операций. Целью является применение языка программирования PSL для повышения эффективности использования данных о технологическом процессе при изготовлении изделия. Подчеркнем, что данный язык программирования прежде всего позволяет повысить эффективность компьютерного обмена данными между сотрудниками планового отдела и руководством производственного подразделения.

В данном приложении рассмотрено расширение примера, использованного в ИСО 18629-11: 2005 (приложение Е). Пример иллюстрирует технические приложения понятий дефинициональных расширений для спецификации процесса изготовления изделия GT-350.

В.1 Процесс изготовления изделия GT-350

В данном разделе различные производственные процессы объединены в набор действий высокого уровня, необходимых для создания изделия GT-350. В соответствии с технологической картой изделия GT-350 (см. ИСО 18629-11: 2005, приложение D, таблица D.1) компоненты данного изделия либо покупают по контракту, либо изготовляют внутри самого предприятия. Рассматриваемые описания технологических процессов связаны с конкретными действиями, выполняемыми внутри предприятия для изготовления компонентов изделия. Данное рассмотрение технологического процесса в направлении «сверху —

вниз» дает общую картину происходящего, описание комплексного действия по изготовлению изделия GT-350, состоящего из составляющих действий, выполняемых на уровне более мелких подразделений предприятия.

В соответствии с рисунком В.1, данным ниже, весь процесс изготовления изделия GT-350 организован в шести основных секторах. В первых пяти из них (изготовление интерьера, изготовление привода, изготовление кузова, изготовление двигателя и изготовление шасси) работы могут быть выполнены независимо друг на друга. Одно условие: они должны быть закончены к моменту начала общей сборки изделия.

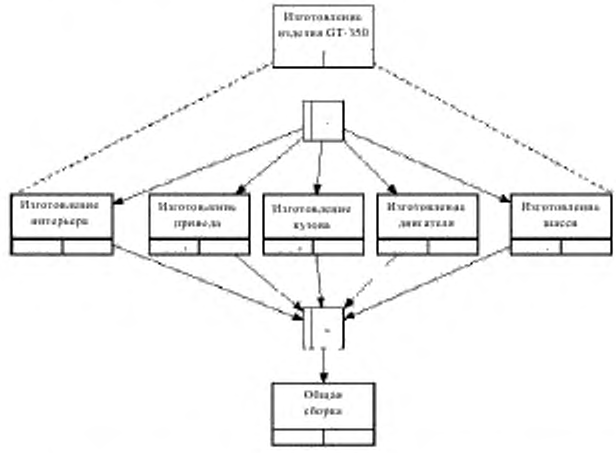

Рисунок В.1 – Верхний уровень процесса изготовления изделия GT-350 [4]

Представление верхнего уровня технологического процесса с помощью расширения языка программирования PSL имеет вид:

```
(resource_path make_gt350)

(subactivity make-chassis make_gt350)
(subactivity make-interior make_gt350)
(subactivity make-drive make_gt350)
(subactivity make-trim make_gt350)
(subactivity make-engine make_gt350)
(subactivity final-assembly make_gt350)

(forall (?occ)
        (implies (occurrence_of ?occ make_gt350)
                 (exists (?occ1 ?occ2 ?occ3 ?occ4 ?occ5 ?occ6)
                         (and (occurrence_of ?occ1 make_chassis)
                              (occurrence_of ?occ2 make_interior)
                              (occurrence_of ?occ3 make_drive)
                              (occurrence_of ?occ4 make_trim)
                              (occurrence_of ?occ5 make_engine)
                              (occurrence_of ?occ6 final_assembly)
```

```
(subactivity_occurrence ?occ1 ?occ)
(subactivity_occurrence ?occ2 ?occ)
(subactivity_occurrence ?occ3 ?occ)
(subactivity_occurrence ?occ4 ?occ)
(subactivity_occurrence ?occ5 ?occ)
(subactivity_occurrence ?occ6 ?occ)
```

Каждое из указанных абстрактных действий может быть рассмотрено в деталях, однако в данном примере это сделано только в отношении некоторых из них.

На базе представления IDEF3 (в терминах представления технологического процесса) для краткого описания действий, встречающихся на различных стадиях процесса изготовления изделия, в настоящем стандарте приведены некоторые примеры использования языка программирования PSL-Outercore и дефинициональных расширений в соответствии с ИСО 18629-12.

B.2 Абстрактное действие «make_engine» (изготовление двигателя)

Двигатель изделия GT-350 собирается из агрегатов, изготовленных в нескольких подразделениях предприятия. Схема процесса изготовления дана на рисунке B.2. Агрегат состоит из двигательного блока, жгутов и кабелей. Составляющие процессы детально рассмотрены в подразделах ниже. Двигатель изделия GT-350 собирается на сборочном стенде A004. Сборка одного двигателя занимает 5 мин.

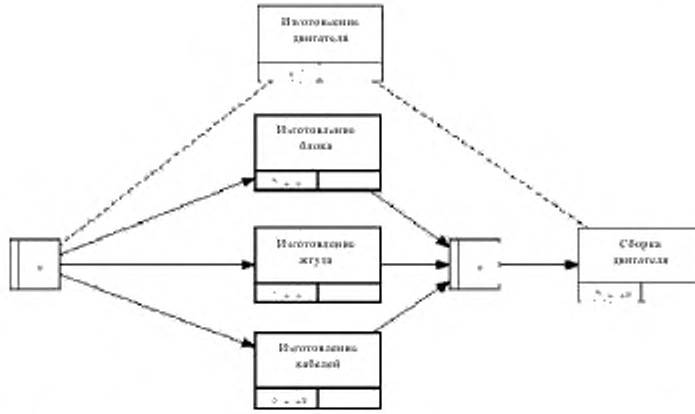

Рисунок B.2 – Процесс изготовления двигателя изделия GT-350 [4]

Представление некоторых действий и соответствующих данных

технологического процесса на языке PSL на этапе изготовления двигателя имеет вид:

```
(subactivity make_block make_engine)
(subactivity make-harness make_engine)
(subactivity make-wires make_engine)
(subactivity assemble_engine make_engine)
(processor_activity make_engine)
(exists (?r1 ?r2 ?r3 ?r4)
        (and (requires make_engine ?r1)
        (requires make_engine ?r2)
        (requires make_engine ?r3)
        (requires make_engine ?r4)
        (workcell ?r4)
        (reusable ?r4 make_engine)
        (processor_resource ?r4 make_engine)
        (engine_block ?r1)
        (input_material ?r1 make_engine)
        (uses_quantity make_engine ?r1 1)
        (output_material ?r1 make_engine)
        (possibly_consumable ?r1 make_engine)
        (harness ?r2)
        (input_material ?r2)
        (consumable ?r2 make_engine)
        (consumes_quantity make_engine ?r2 1)
        (wire ?r3)
        (input_material ?r3)
        (consumes_quantity make_engine ?r3 5)
        (wearable ?r3 make_engine)))

(forall (?r)
    (implies (engine_block ?r)
          (resource ?r)))

(forall (?r)
     (implies (harness ?r)
           (resource ?r))

(forall (?r ?s)
      (implies (wire ?r)
            (exists (?i)
                 (and (prior (resource_set ?i ?r) ?s)
                       (pile ?r make_engine)))))

(forall (?occ)
      (iff (occurrence_of ?occ make_engine)
           (exists (?occ1 ?occ2 ?occ3 ?occ4)

                (and (occurrence_of ?occ1 make_block)
                      (occurrence_of ?occ2 make_harness)
                      (occurrence_of ?occ3 make_wires)
                      (occurrence_of ?occ4 assemble_engine)
                      (subactivity_occurrence ?occ1 ?occ)
                      (subactivity_occurrence ?occ2 ?occ)
                      (subactivity_occurrence ?occ3 ?occ)
                      (subactivity_occurrence ?occ4 ?occ)
```

```
(forall (?s1 ?s2 ?s3 ?s4)
  (implies (and     (leaf_occ ?s1 ?occ1)
          (leaf_occ ?s2 ?occ2)
          (leaf_occ ?s3 ?occ3)

                    (root_occ ?s4 ?occ4))
          (and (min_precedes ?s1 ?s4 make_engine)
                    (min_precedes ?s2 ?s4 make_engine)
                    (min_precedes ?s3 ?s4 make_engine)))))
```

Данное представление формализует технологический процесс, представленный на рисунке В.2.

Процесс изготовления двигателя требует четыре ресурса: блок двигателя, жгут, набор проводов и производственный участок. Производственный участок является повторно используемым ресурсом. Ресурс «engine_block» (блок двигателя) является «possibly_consumable» (возможно потребляемым ресурсом), так как он может быть использован повторно в будущем при модификации данного блока. Однако данный ресурс не может быть использован всеми действиями (после того как некоторые изменения в него были внесены). Ресурс «harness» (жгут) является «consumable» (потребляемым ресурсом), так как никакие последующие действия не смогут использовать жгут повторно после его установки. Ресурс «wires» (набор проводов) – это изнашиваемый ресурс, так как в результате многократного совершенствования процесса «make_engine» (изготовление двигателя) количество проводов в наборе ресурсов уменьшается.

Все указанные ограничения показывают, что действие «make_engine» является действием процессора, в котором жгут и провод являются входными материалами. Объект «engine_block» (блок двигателя) является одновременно и входным материалом, и выходным материалом, так как он модифицируется в процессе изготовления двигателя «make_engine».

Провода представляют собой некоторый набор ресурсов, так как любой поднабор проводов может быть потреблен в процессе «make_engine». Данный набор ресурсов – это «pile» («куча»).

В.2.1 Изготовление блока двигателя «make_block»

Блок изделия GT-350 выполняется как агрегат для сборки двигателя изделия GT-350. Изготовление блока требует выполнения всех технологических операций, начиная от литья заготовки и ее механической обработки (см.

рисунок В.3).



Рисунок В.3 – Процесс изготовления блока изделия GT-350 [4]

Представление некоторых действий и технологических данных на языке программирования PSL:

```
(subactivity produce_molded_metal make_block)
(subactivity machine_block make_block)
(primitive machine_block)
(primitive produce_molded_metal)

(resource_path make_block)
(processor_activity produce_molded_metal)
(processor_activity machine_block)

(exists (?r1 ?r2 ?r3)
        (and (requires produce_molded_metal ?r1)
        (requires produce_molded_metal ?r2)
        (cast ?r1)
        (reusable ?r1produce_molded_metal)
        (processor_resource ?r1 produce_molded_metal)
        (metal_block ?r2)
        (input_material ?r2 make_block)
        (output_material ?r3 make_block)
        (molded_block ?r3)
        (produces produce_molded_metal ?r3)
        (consumable ?r2 produce_molded_metal)
        (consumes_quantity produce_molded_metal ?r2 1)
        (unary_resource ?r1)
        (exclusive_use ?r1 produce_molded_metal)))

(exists (?r1 ?r2)
        (and (requires machine_block ?r1)
        (requires machine_block ?r2)
        (reusable ?r2 machine_block)
        (metal_block ?r1)
        (input_material ?r1 machine_block)
        (uses_quantity machine_engine ?r1 1)
        (output_material ?r1 machine_block)
        (consumable ?r1 machine_engine)
        (milling_machine ?r2)
        (processor_resource ?r2 machine_block)
        (consumes_quantity machine_engine ?r1 1)
        (capacitated_resource ?r1 )))
(forall (?r)
```

```
        (implies (milling_machine ?r)
                (resource ?r)))

(forall (?r)
        (implies (molded_block ?r)
                (resource ?r))

(forall (?occ)
        (iff (occurrence_of ?occ make_block)
                (exists (?occ1 ?occ2)
                        (and (occurrence_of ?occ1 produce_molded_metal)
                                (occurrence_of ?occ2 machine_block)
                                (next_processor_path ?occ1 ?occ2 make_block)))))
```

Данное представление формализует технологический процесс, представленный на рисунке В.3.

Действие «make_block» (изготовление блока) – это действие процессора с путем, включающее последовательность двух поддействий процессора «produce_molded_metal» (прессование металлической заготовки и «machine_block» (механическая обработка блока).

Поддействие «produce_molded_metal» потребляет металлический блок, использует литье и изготовляет прессованный блок.

Поддействие «machine_block» потребляет прессованный блок и использует фрезерный станок.

Литье выполняется только однажды. Оно эксклюзивно используется действием «produce_molded_metal» и, таким образом, является «unary» (унарным) ресурсом.

Фрезерный станок может быть использован многократно, это ресурс с ограниченной пропускной способностью.

В.2.2 Изготовление жгута «make_harness»

Жгут изделия GT-350 (см. рисунок В.4) изготовляется как сборочный агрегат двигателя изделия GT-350. Данный технологический процесс организован в цехе кабелей и проводов. Жгут изделия GT-350 собирается на особом стенде из проводов и кабелей. Сборка одного жгута занимает 10 мин.



Рисунок В.4 – Процесс изготовления жгута изделия GT-350 [4]

Ниже дано представление некоторых действий и соответствующих технологических данных на языке программирования PSL-Outercore:

```
(resource_path make_harness)
(pro_precedes make_harness_wire assemble_harness make_harness)

(subactivity make_harness_wire make_harness)
(subactivity assemble_harness make_harness)
(primitive assemble_harness)

(forall (?r)
        (implies (harness ?r)
                (wearable ?r make_harness)))

(forall (?occ)
        (implies (occurrence_of ?occ make_harness)
                (exists (?occ1 ?occ2 ?occ3 ?r)
                        (and (occurrence_of ?occ1 make_harness_wire)
                                (harness ?r)
                                (requires make_harness_wire ?r)
                                (occurrence_of ?occ2 assemble_harness)
                                (requires assemble_harness ?r)
                                (leaf_occ ?occ3 ?occ1)
                                (min_precedes ?occ3 ?occ2 make_harness)))))

(forall (?occ ?r ?q)
        (implies (and (occurrence_of ?occ assemble_harness)
                (requires assemble_harness ?r)
                (prior (resource_point ?r ?q) ?occ))
            (holds (resource_point ?r ?q) ?occ)))
```

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/МЭК 8824-1	IDT	ГОСТ Р ИСО/МЭК 8824-1:2001 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 1. Спецификация основной нотации»
ИСО15531-1	IDT	ГОСТ Р ИСО 15531-1:2008 «Промышленные автоматизированные системы и интеграция. Данные по управлению промышленным производством. Часть 1. Общий обзор»
ИСО15531-42:2005	IDT	ГОСТ Р ИСО 15531-32:2010 «Системы промышленной автоматизации и интеграция. Управляющая информация промышленным производством. Управление использованием ресурсов. Часть 32. Концептуальная модель данных для управления использованием ресурсов»
ИСО 18629-1:2004	–	*

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО 18629-11:2005	–	*
ИСО 18629-12:2005	–	*
ИСО 18629-14:2006	–	*

* Соответствующий национальный стандарт отсутствует (в разработке). 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] ИСО 10303-1 Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 1. Обзор и основные принципы

[2] ИСО 10303-49 Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 49. Интегрированные родовые ресурсы: структура и свойства процесса

[3] ИСО 18629-13 Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 13. Теория длительности и упорядочения операций

[4] Federal Information Processing Standards Publication 184, Integration Definition for Information Modeling (IDEF3), FIPS PUB 184, National Institute of Standards and Technology, December 1993. IDEF3. Available from the Internet: <gost.e-ecolog.ru >

УДК 65.011:56.681.3 ОКС 25.040.40 Т 58

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

Подписано в печать 30.04.2014. Формат 60x84 1/8.

Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

ФГУП «СТАНДАРТИНФОРМ»

123995 Москва, Гранатный пер., 4.

info@gostinfo.ru

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