ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 18629-13 — 2011

Системы промышленной автоматизации и интеграция

ЯЗЫК СПЕЦИФИКАЦИЙ ПРОЦЕССА

Часть 13

Теории продолжительности и упорядочения операций

ISO 18629-13:2006

Industrial automation systems and integration —

Process specification language —

Part 13:

Duration and ordering theories

(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. № 1068-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 18629-13:2006 «Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 13. Теории продолжительности и упорядочения операций» (ISO 18629-13:2006 «Industrial automation systems and integration — Process specification language — Part 13: Duration and ordering theories»).

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 13
Теории продолжительности и упорядочения операций
Industrial automation systems and integration.
Process specification language.
Part 13. Duration and ordering theories

Дата введения – 2012 – 09 – 01

1 Область применения

В настоящем стандарте дано описание элементарных принципов, связанных с ограничениями продолжительности и упорядочения операций, а также рассмотрены следующие вопросы:

- упорядочение элементов субопераций;
- продолжительность операций;
- итерированное упорядочение операций;
- эндоморфизмы дерева операций;
- оболочки операций.

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 Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 1. Обзор и основные принципы (ISO 18629-1, 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 18629-12:2005 Системы промышленной автоматизации и интеграция. Язык спецификаций процесса. Часть 12. Внешнее ядро (ISO 18629-12:2005, Industrial automation systems and integration — Process specification language — Part 12: Outer core)

3 Термины, определения и сокращения

3.1 Термины и определения

В настоящем стандарте применены следующие термины с соответствующими и определениями:

3.1.1 **автоморфизм** (automorphism): Отображение «один к одному» элементов на множество, сохраняющее соотношения и функции в некоторой модели.

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

[ИСО 18629-1]

3.1.3 **коммутативная группа** (commutative group): Алгебраическая

структура с внутренней бинарной операцией (OP), по отношению к которой справедливо соотношение вида a OP b = b OP a.

3.1.4 **консервативное определение** (conservative definition): Определение, которое устанавливает необходимые и достаточные условия для полного соответствия термина, а также не позволяет выводить новые умозаключения из теории.

[ИСО 18629-1]

3.1.5 **теория ядра** (core theory): Набор аксиом для реляционных и функциональных символов, обозначающих примитивные понятия.

[ИСО 18629-1]

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

Примечание – Установленная лексика состоит из констант, функций и отношений.

Пример – Термины с консервативным определением.

[ИСО 18629-1]

3.1.7 **эндоморфизм** (endomorphism): Отображение множества на некоторое подмножество, сохраняющее соотношения и функции в некоторой модели.

3.1.8 **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]

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

[ИСО 18629-1]

3.1.10 **гомоморфизм** (homomorphism): Отображение между множествами, сохраняющее некоторые соотношения на элементах множества.

3.1.11 **интерпретация** (interpretation): Совокупность дискурсов и присвоение значений истинности (TRUE или FALSE) всем положениям теории.

Примечание – Пример интерпретации см. в приложении В.

[ИСО 18629-11]

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

[ИСО 18629-1]

3.1.13 **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]

3.1.14 **модель** (model): Сочетание набора элементов и истинного назначения, удовлетворяющее всем правильно построенным формулировкам в теории.

Примечание 1 – В настоящем стандарте определение термина «модель» отличается от используемого в научной и другой литературе: если предложение является верным в определенной интерпретации, то можно сказать, что интерпретация

– это модель предложения. Виды семантик, представленных в настоящем стандарте, часто называют теоретически смоделированными семантиками.

Примечание 2 – Модель обычно представляют в виде совокупности дополнительных структур (частично упорядоченных, в качестве структурного или векторного пространства). В этом случае модель определяет значения для терминологии и понятия истины для предложений языка в условиях данной модели. Задавая модель, основной набор аксиом математических структур, используемый в наборе аксиом, используют как основу для определения понятий, представленных в терминах языка, и их логических взаимосвязей, в результате чего набор моделей создает формальные семантики онтологии.

[ИСО 18629-1]

3.1.15 **мономорфизм** (monomorphism): Отображение «один к одному» между множествами, сохраняющее некоторые соотношения на элементах множества.

3.1.16 **онтология** (ontology): Лексика специализированной терминологии, дополненная необходимой спецификацией значений терминов.

Примечание 1 – Структурированный набор относительных терминов, представленный с описанием значений терминов на формальном языке. Описание значения объясняет, как и почему термины соотносятся, и определяет условия сегментирования и структурирования набора терминов.

Примечание 2 – Основополагающим компонентом языка технологических спецификаций ИСО 18629 является онтология. Примитивные концепции в онтологии, соответствующей определению ИСО 18629, достаточны для описания основных производственных, инженерных и бизнес - процессов.

Примечание 3 – Основное внимание онтологии направлено не только на термины, но и на их значения. Произвольный набор терминов включен в онтологию, но эти термины могут приниматься только в том случае, если их значения согласованы. Такие предполагаемые семантики терминов могут быть утверждены и использованы.

Примечание 4 – Любой термин, используемый без точного определения, может быть причиной неясности и путаницы. Сложность для онтологии в том, что структура нуждается в создании терминов, имеющих точное значение. Для онтологии, соответствующей определению ИСО 18629, необходимо предоставить математически строгую характеристику информационного процесса, а также четкое выражение основных логических свойств этой информации на языке, указанном в ИСО 18629.

[ИСО 18629-1]

3.1.17 **внешнее ядро** (outer Core): Набор теорий ядра, которые являются расширениями ядра PSL и настолько обобщены и распространены в своем применении, что каждая теория может быть представлена отдельно.

Примечание – На практике расширения включают в себя аксиомы внешнего ядра.

[ИСО 18629-1]

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

[ИСО 18629-1]

3.1.19 **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]

3.1.20 **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]

3.1.21 **теория доказательств** (proof theory): Совокупность теорий и лексических элементов, необходимых для интерпретации семантики языка.

Примечание – Теория доказательств состоит из трех компонентов: ядра PSL, внешнего ядра и расширений.

[ИСО 18629-1]

3.1.22 **ядро PSL** (PSL-Core): Набор аксиом для понятий деятельности, события деятельности, момента времени и объекта.

Примечание – Мотивацией для ядра PSL является наличие любых двух приложений, имеющих отношение к процессу, которые должны совместно использовать упомянутые аксиомы в целях обмена информацией о процессе. Поэтому ядро PSL является адекватным для описания основных концепций производственных процессов. Следовательно, эта характеристика основных процессов имеет несколько допущений в отношении их характеристик, за исключением тех, которые необходимы для описания процессов. Поэтому ядро PSL ограничено с точки зрения выражения логической возможности. При этом ядро PSL обеспечивает определение многих вспомогательных понятий, которые необходимы для описания всех интуитивных понятий в производственном процессе.

[ИСО 18629-1]

3.1.23 **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]

3.1.24 **область обсуждения** (universe of discourse): Совокупность конкретных или абстрактных вещей, относящихся к области реального мира, которые выбраны в соответствии с интересом, который они представляют для системы, подлежащей моделированию, и ее окружением.

[ИСО 15531-1]

3.2 Сокращения

В настоящем стандарте применены следующие сокращения:

- FOL – логика первого порядка (First-Order Logic);
- BNF – формализм Бэкуса – Наура (Backus-Naur Formalism);

- KIF — формат обмена знаниями (Knowledge Interchange Format);

- PSL — язык спецификаций процесса (Process Specification Language).

4 Информация, общая для всех частей стандарта ИСО 18629

В ИСО 18629-1 определен язык представления производственной информации, который является языком спецификаций процесса (PSL). Он состоит из лексики, онтологии и грамматики, предназначенных для описания процессов и указанных в частях 11 — 19 и 41 — 49 комплекса международных стандартов ИСО 18629[1)].

[1)] Некоторые части ИСО 18629 находятся в стадии разработки.

Примечание – PSL – это язык для описания производственных процессов, основанный на математически строго определенном лексическом наборе и грамматике. Этот язык существенно отличается от других языков, например от языка EXPRESS (определенного в ИСО 10303-11), и используется, например, в ИСО 10303-41, ИСО 10303-42, ИСО 10303-49, ИСО 13584, ИСО 15531 и ИСО 15926, являясь признанным языком моделирования. В области обмена информацией между двумя процессами PSL-язык позволяет описывать каждый процесс независимо от его характера. Например, объект, рассматриваемый как ресурс в одном процессе, может считаться аналогичным объектом даже несмотря на то, что в другом процессе он будет рассматриваться как продукт (изделие). PSL-язык основан на математической теории множеств и ситуационном исчислении (см. ИСО 18629-11: 2005, приложение D).

В частях 11 — 19 ИСО 18629 определены базовые теории, необходимые для получения четких определений и соответствующих аксиом для элементарных понятий ИСО 18629, что позволяет осуществлять точный семантический переход между различными логическими структурами.

Применение частей 11 — 19 ИСО 18629 обеспечивает:

- представление основных элементов языка;

- стандартизованный набор аксиом, которые отвечают интуитивным семантическим понятиям, достаточным для описания основных производственных процессов;

- набор правил для разработки в соответствии с PSL-Core других базовых теорий или расширений, например расширений, рассмотренных в частях 41 — 49 комплекса международных стандартов ИСО 18629.

5 Структура настоящего стандарта

В настоящем стандарте рассмотрены следующие базовые теории:

- теория упорядочения элементов субопераций (функциональной подгруппы) (soo.th);

- теория продолжительности операций (duration.th);

- теория автоморфизма дерева элементов (preserve.th);

- теория оболочки операций (envelope.th).

Все теории, представленные в настоящем стандарте, являются расширениями теорий, изложенных в ИСО 18629-12.

6 Теория упорядочения элементов субопераций

В теории упорядочения элементов субопераций вводятся понятия, необходимые для представления интуитивно воспринимаемых понятий, связанных с последовательностью выполнения технологических операций и частичного упорядочения элементов субопераций, из которых состоит комплексная операция.

6.1 Примитивные отношения в теории упорядочения элементов субопераций

Нелогическая лексика теории упорядочения элементов субопераций содержит два следующих примитивных символа отношений, а именно:

- soo;

- soo_precedes.

Нелогическая лексика теории упорядочения элементов субопераций содержит одну примитивную функцию отношений, а именно:

- soomap.

6.2 Определяющие отношения в теории упорядочения элементов субопераций

Нелогическая лексика теории упорядочения элементов субопераций содержит следующие определяющие символы отношений:

- root_soo;

- leaf_soo;

- next_subactivity.

6.3 Связь с другими группами аксиом

Теория упорядочения элементов субопераций требует использования следующих теорий:

- psl_core.th;
- occtree.th;
- atomic.th;
- complex.th;
- actocc.th.

Никаких дефинициональных расширений в теории упорядочения элементов субопераций не требуется.

6.4 Неформальная семантика теории упорядочения элементов субопераций

6.4.1 Примитивный символ отношения soo

KIF-формат обозначения примитивного символа отношения soo таков:

(soo ?s ?a).

Неформальная семантика для примитивного символа отношения soo такова:

(soo ?s ?a) в интерпретации теории упорядочения элементов субопераций принимает значение TRUE тогда и только тогда, когда элемент операции ?s является элементом субоперации, упорядоченным для операции ?a.

6.4.2 Примитивный символ отношения soo_precedes

KIF-формат обозначения примитивного символа отношения soo_precedes таков:

(soo_precedes ?s1 ?s2 ?a).

Неформальная семантика для примитивного символа отношения soo_precedes такова:

(soo_precedes ?s1 ?s2 ?a) в интерпретации теории упорядочения элементов субопераций принимает значение TRUE тогда и только тогда, когда элемент операции ?s1 предшествует элементу операции ?s2 в элементе субоперации, упорядоченном для операции ?a. Это отношение определяет частичное упорядочение по элементам субопераций для комплексной операции.

6.4.3 Примитивная функция отношения soomap

KIF-формат обозначения примитивной функции отношения soomap таков:

(soomap ?s).

Неформальная семантика для примитивной функции отношения soomap такова:

(= ?s1 (soomap ?s)) в интерпретации теории упорядочения элементов субопераций принимает значение TRUE тогда и только тогда, когда элемент операции ?s1 является элементом субоперации, упорядоченным так, чтобы он соответствовал элементу ?s.

6.5 Определения теории упорядочения элементов субопераций

6.5.1 Определение 1 (связанное с символом отношения root_soo)

Элемент операции является корневым узлом упорядоченной структуры элементов субоперации для операции ?a тогда и только тогда, когда он является элементом упорядочения и отсутствует элемент операции, который предшествует ему при упорядочении.

```
(forall (?s ?a) (iff (root_soo ?s ?a)
(and (soo ?s ?a)
        (not (exists (?s1)
                (soo_precedes ?s1 ?s ?a)))))
```

6.5.2 Определение 2 (связанное с символом отношения leaf_soo)

Элемент операции является конечным узлом упорядоченной структуры элементов субоперации для операции ?a тогда и только тогда, когда он является элементом упорядочения и отсутствует элемент операции, который следует за ним при упорядочении.

```
(forall (?s ?a) (iff (leaf_soo ?s ?a)
        (and (soo ?s ?a)
        (not (exists (?s1)
                (soo_precedes ?s ?s1 ?a)))))
```

6.5.3 Определение 3 (связанное с символом отношения next_soo)

Элемент операции ?s2 является следующим элементом субоперации после элемента ?s1 в упорядочении элементов субопераций для операции ?a

тогда и только тогда, когда элемент ?s1 предшествует элементу ?s2 при упорядочении, а между этими элементами при упорядочении отсутствует какой-либо элемент субоперации.

```
(forall (?s1 ?s2 ?a) (iff (next_soo ?s1 ?s2 ?a)
(and (soo_precedes ?s1 ?s2 ?a)
        (not (exists (?s3)
          (and (soo_precedes ?s1 ?s3 ?a)
                (soo_precedes ?s3 ?s2 ?a)))))))
```

6.6 Аксиомы теории упорядочения элементов субопераций

6.6.1 Аксиома 1

Примитивный символ отношения soo_precedes упорядочивает элементы субопераций.

```
(forall (?s1 ?s2 ?a)
        (implies (soo_precedes ?s1 ?s2 ?a)
                (and (soo ?s1 ?a)
                        (soo ?s2 ?a))))
```

6.6.2 Аксиома 2

Элементы упорядочения элементов субоперации являются элементами дерева операций.

```
(forall (?a ?s)
        (implies (soo ?s ?a)
                (or (root ?s ?a)
                        (exists (?s1)
                                (min_precedes ?s1 ?s ?a)))))
```

6.6.3 Аксиома 3

Примитивная функция отношения soomap отображает дерево операций на упорядоченную структуру элементов субопераций.

```
(forall (?s ?a)
        (soo (soomap ?s) ?a))
```

6.6.4 Аксиома 4

Элементы упорядоченной структуры элементов субопераций фиксируются с помощью примитивной функции отношения soomap.

```
(forall (?s ?a)
        (implies (soo ?s ?a)
                (= ?s (soomap ?s))))
```

6.6.5 Аксиома 5

Примитивная функция отношения soomap является ветвлением мономорфизма.

```
(forall (?s ?a)
        (or    (mono (?s (soomap ?s) ?a)
                  (= ?s (soomap ?s))))
```

6.6.6 Аксиома 6

Дерево операций является порядково-гомоморфичным в отношении упорядоченной структуры элементов субопераций.

```
(forall (?a ?s1 ?s2)
        (implies (min_precedes ?s1 ?s2 ?a)
                 (iff (soo_precedes (soomap ?s1) (soomap ?s2) ?a)
                          (not (exists (?s3 ?s4)
                                   (and (min_precedes ?s4 ?s3 ?a)
                                        (= (soomap ?s3) (soomap ?s1))
                                        (= (soomap ?s4) (soomap ?s2))))))))
```

6.6.7 Аксиома 7

Примитивный символ отношения soo_precedes не является симметричным.

```
(forall (?a ?s1 ?s2)
        (implies (soo_precedes ?s1 ?s2 ?a)
                 (not (soo_precedes ?s2 ?s1 ?a))))
```

6.6.8 Аксиома 8

Примитивный символ отношения soo_precedes является транзитивным.

```
(forall (?a ?s1 ?s2 ?s3)
        (implies (and (soo_precedes ?s1 ?s2 ?a)
                      (soo_precedes ?s2 ?s3 ?a))
                 (soo_precedes ?s1 ?s3 ?a)))
```

7 Теория продолжительности операций

В этой теории вводится понятие «продолжительность операции», в которой путем отображения каждой пары точек на новый подкласс объектов, называемых «временные интервалы» (timeduration), на временной шкале вводится метрика, причем эти интервалы в векторном пространстве удовлетворяют определенным аксиомам.

7.1 Примитивные отношения в теории продолжительности операций

В нелогической лексике теории продолжительности операций содержатся два примитивных символа отношений, а именно:

- timeduration;
- lesser.

7.2 Примитивные функции и константы

В нелогической лексике теории продолжительности операций содержатся три примитивных символа функций, а именно:

- duration (продолжительность);
- add (сложение);
- mult (умножение).

В нелогической лексике теории продолжительности операций содержатся четыре символа констант, а именно:

- zero (нуль);
- one (единица);
- max+;
- max-.

7.3 Определяющие отношения в теории продолжительности операций

Нелогическая лексика теории продолжительности операций содержит один определяющий символ отношения, а именно:

- time_add.

7.4 Связь с другими группами аксиом

Требуемая базовая теория — это:

- psl_core.th.

Никаких дефинициональных расширений в теории продолжительности операций не требуется.

7.5 Неформальная семантика теории продолжительности операций

7.5.1 Примитивный символ отношения timeduration

KIF-формат обозначения примитивного символа отношения timeduration таков:

(timeduration ?d)

Неформальная семантика для примитивного символа отношения timeduration такова:

(timeduration ?d) в интерпретации теории продолжительности операций принимает значение TRUE тогда и только тогда, когда ?d является элементом множества временных интервалов в общей совокупности дискурсов интерпретации. Временные интервалы являются подкатегорией объекта.

7.5.2 Примитивный символ отношения lesser

KIF-формат обозначения примитивного символа отношения lesser таков:

(lesser ?d1 ?d2)

Неформальная семантика для примитивного символа отношения lesser такова:

Все временные интервалы линейно упорядочены.

(lesser ?d1 ?d2) в интерпретации теории продолжительности операций принимает значение TRUE тогда и только тогда, когда значение временного интервала ?d1 меньше другого значения временного интервала ?d2.

7.5.3 Примитивный символ функции duration

KIF-формат обозначения примитивного символа функции duration таков:

(duration ?t1 ?t2)

Неформальная семантика для примитивного символа функции duration такова:

(= (duration ?t1 ?t2) ?d) в интерпретации теории продолжительности операций принимает значение TRUE тогда и только тогда, когда ?d означает временной интервал, чье значение является расстоянием между двумя временными точками ?t1 и ?t2 на временной оси.

7.5.4 Определяющий символ отношения time_add

KIF-формат обозначения определяющего символа отношения time_add таков:

(time_add ?t ?d)

Неформальная семантика для определяющего символа функции time_add такова:

(= (time_add ?t ?d) ?t2) в интерпретации теории продолжительности операций принимает значение TRUE тогда и только тогда, когда ?t2 означает временную точку, отстоящую на расстоянии ?d от временных точек ?t на временной оси.

7.5.5 Примитивный символ функции add

KIF-формат обозначения примитивного символа функции add таков:

(add ?d1 ?d2)

Неформальная семантика для примитивного символа функции add такова:

(= (add ?d1 ?d2) ?d3) в интерпретации теории продолжительности операций принимает значение TRUE тогда и только тогда, когда ?d3 означает временной интервал, чье значение является суммой значений временных интервалов ?d1 и ?d2.

7.5.6 Примитивный символ функции mult

KIF-формат обозначения символа функции mult таков:

(mult ?x ?d)

Неформальная семантика для символа функции mult такова:

(= (mult ?x ?d) ?d2) в интерпретации теории продолжительности операций принимает значение TRUE тогда и только тогда, когда ?d2 означает временной интервал, чье значение является произведением значений временных интервалов ?d и ?x, где ?x является элементом коммутативной группы.

7.5.7 Символ константы zero

Неформальная семантика для символа константы zero такова:

символ zero определяет константу для временного интервала, которая является аддитивной единицей для функции add.

7.5.8 Символ константы one

Неформальная семантика для символа константы one такова:

символ one означает константу, которая является мультипликативной для функции mult.

7.5.9 Символ константы max+

Неформальная семантика для символа константы max+ такова:

max+ — максимальный временной интервал.

7.5.10 Символ константы max-

Неформальная семантика для символа константы max- такова:

max- — минимальный временной интервал.

7.6 Определения теории продолжительности операций

7.6.1 Определение 1

Функция time_add отображает временную точку ?t1 и временной интервал ?d на временную точку ?t2 так, что временной интервал между ?t1 и ?t2 составляет ?d.

```
(forall (?t1 ?t2 ?d)
(iff (= ?t2 (time_add ?t1 ?d))
        (and (timepoint ?t1)
          (timepoint ?t2)
          (timeduration ?d)
          (= ?d (duration ?t2 ?t1)))))
```

7.7 Аксиомы теории продолжительности операций

Ниже приведен полный набор аксиом для теории продолжительности операций.

7.7.1 Аксиома 1

zero, max+ и max- — временные интервалы.

```
(and (timeduration zero)
        (timeduration max+)
        (timeduration max-))
```

7.7.2 Аксиома 2

Результат сложения двух временных интервалов также является временным интервалом.

```
(forall (?d1 ?d2)
(implies (and (timeduration ?d1)
        (timeduration ?d2))
        (timeduration (add ?d1 ?d2))))
```

7.7.3 Аксиома 3

Функция add является ассоциативной.

```
(forall (?d1 ?d2 ?d3)
(implies (and (timeduration ?d1)
        (timeduration ?d2)
        (timeduration ?d3))
          (= (add (add ?d1 ?d2) ?d3) (add ?d1 (add ?d2 ?d3)))))
```

7.7.4 Аксиома 4

Символ константы zero является аддитивной единицей.

```
(forall (?d)
(implies (timeduration ?d)
        (= (add ?d zero) ?d)))
```

7.7.5 Аксиома 5

Временной интервал, взятый с противоположным знаком, также является временным интервалом.

```
(forall (?d1)
(implies (timeduration ?d1)
          (exists (?d2)
        (and (timeduration ?d2)
            (= (add ?d1 ?d2) zero)))))
```

7.7.6 Аксиома 6

Функция add является коммутативной.

```
(forall (?d1 ?d2)
(implies (and (timeduration ?d1)
        (timeduration ?d2))
          (= (add ?d1 ?d2) (add ?d2 ?d1)))))
```

7.7.7 Аксиома 7

Результат умножения временного интервала на скалярную величину также является временным интервалом.

```
(forall (?d ?r)
(implies (timeduration ?d)
        (timeduration (mult ?r ?d))))
```

7.7.8 Аксиома 8

Временные интервалы могут умножаться на скаляры:

```
(forall (?d1 ?d2 ?r)
        (= (mult ?r (add ?d1 ?d2)) (add (mult ?r ?d1) (mult ?r ?d2))))
```

7.7.9 Аксиома 9

Операция умножения временных интервалов на скаляры является дистрибутивной:

```
(forall (?d ?r ?s)
        (= (mult (add ?r ?s) ?d) (add (mult ?r ?d) (mult ?s ?d))))
```

7.7.10 Аксиома 10

Операция умножения временных интервалов на скаляры является ассоциативной:

```
(forall (?d ?r ?s)
        (= (mult (mult ?r ?s) ?d) (mult ?r (mult ?s ?d))))
```

7.7.11 Аксиома 11

Символ константы one является мультипликативной единицей.

```
(forall (?d)
        (= ?d (mult one ?d)))
```

7.7.12 Аксиома 12

Функция add на временных интервалах сохраняет структуру упорядочения lesser.

```
(forall (?d1 ?d2 ?d3)
  (and (timeduration ?d1)
        (timeduration ?d2)
        (timeduration ?d3))
        (iff (lesser ?d1 ?d2)
            (lesser (add ?d1 ?d3) (add ?d2 ?d3))))
```

7.7.13 Аксиома 13

Если два временных интервала равны, то результат их сложения также будет тем же.

```
(forall (?d1 ?y ?z)
        (and (timeduration ?d1)
          (timeduration ?d2)
          (timeduration ?d3))
          (iff (= ?d1 ?d2)
                (= (add ?d1 ?d3) (add ?d2 ?d3)))))
```

7.7.14 Аксиома 14

Константа max- меньше любого временного интервала, а константа max+ больше любого временного интервала.

```
(forall (?d)
        (implies (timeduration ?d)
                 (and (lesser ?d max+)
                      (lesser max- ?d))))
```

7.7.15 Аксиома 15

Результат сложения любого временного интервала (отличного от константы max+) с константой max- равен константе max-, и наоборот; а сумма констант max- и max+ равна нулю.

```
(forall (?d)
(implies (timeduration ?d)
        (and (implies (not (= ?d max-)) (= max+ (add ?d max+)))
        (implies (not (= ?d max+)) (= max- (add ?d max-)))
        (= zero (add max+ max-)))))
```

7.7.16 Аксиома 16

Функция продолжительности операции присваивает временной интервал каждой паре временных точек.

```
(forall (?t1 ?t2)
        (implies (and (timepoint ?t1)
                      (timepoint ?t2))
                 (timeduration (duration ?t1 ?t2))))
```

7.7.17 Аксиома 17

Каждый временной интервал равен значению функции продолжительности операции для любой пары временных точек.

```
(forall (?d)
        (implies (timeduration ?d)
                 (exists (?t1 ?t2)
                         (and (timepoint ?t1)
                              (timepoint ?t2)
                              (= ?d (duration ?t1 ?t2))))))
```

7.7.18 Аксиома 18

Значение функции продолжительности операции равно нулю тогда и только тогда, когда две временные точки совпадают.

```
(forall (?t1 ?t2)
(implies (and (timepoint ?t1)
        (timepoint ?t2))
        (iff (= zero (duration ?t1 ?t2))
        (= ?t1 ?t2)))
```

7.7.19 Аксиома 19

Значение функции продолжительности операции между временными точками ?t1 и ?t2 является аддитивной инверсией этой функции между временными точками ?t2 и ?t1.

```
(forall (?t1 ?t2)
(implies (and (timepoint ?t1)
        (timepoint ?t2))
        (= zero (add (duration ?t1 ?t2) (duration ?t2 ?t1)))))
```

7.7.20 Аксиома 20

Для заданной временной точки ?t1, отличной от inf- или inf+, продолжительность операции между временной точкой ?t1 и любой другой временной точкой является однозначно определяемой.

```
(forall (?t1 ?t2 ?t3)
        (implies (and (timepoint ?t1)
                (timepoint ?t2)
                (timepoint ?t3)
                (not (= ?t1 inf-))
                (not (= ?t2 inf+))
                (= (duration ?t1 ?t2) (duration ?t1 ?t3)))
           (= ?t3 ?t2)))
```

7.7.21 Аксиома 21

Продолжительность операции от любой точки, отличной от inf- до inf-, является константой max-, а продолжительность операции от любой точки, отличной от inf+, до inf+ является константой max+.

```
(forall (?t)
(and       (implies (and       (timepoint ?t)
              (not (= ?t inf-)))
          (= max+ (duration inf- ?t)))
     (implies (and (timepoint ?t)
              (not (= ?t inf+)))
          (= max- (duration inf+ ?t)))
```

7.7.22 Аксиома 22

Продолжительность операции от inf- до любой точки, отличной от inf-, является константой max+ , а продолжительность операции от inf+ до любой точки, отличной от inf+, является константой max-.

```
(forall (?t)
(and       (implies (and       (timepoint ?t)
              (not (= ?t inf-)))
          (= max- (duration ?t inf-)))
     (implies (and (timepoint ?t)
              (not (= ?t inf+)))
```

(= max+ (duration ?t inf+)))

8 Теория автоморфизма дерева элементов

Теория автоморфизма дерева элементов делает аксиоматичной интуитивные представления относительно ограничений, в которых допустимые элементы одной операции зависят от элементов других операций.

8.1 Примитивные отношения в теории автоморфизма дерева элементов

В нелогической лексике теории автоморфизма дерева элементов один примитивный символ отношения, а именно:

— ubiquitous (повсеместный).

8.2 Определяющие отношения в теории автоморфизма дерева элементов

Нелогическая лексика теории автоморфизма дерева элементов содержит три определяющих символа отношений, а именно:

- end_ИСО;
- legal_map;
- tree_map.

8.3 Связь с другими группами аксиом

Требуемые базовые теории — это:

- pslcore.th;
- occtree.th;
- atomic .th;
- complex.th;
- actocc.th.

Требуемое дефинициональное расширение теории — это:

- occ_precond.def.

8.4 Неформальная семантика теории автоморфизма дерева элементов

8.4.1 Примитивный символ отношения ubiquitous

KIF-формат обозначения примитивного символа отношения ubiquitous таков:

(ubiquitous ??a1 ?a2)

Неформальная семантика для примитивного символа отношения ubiquitous такова:

(ubiquitous ?a1 ?a2) в интерпретации теории автоморфизма дерева элементов принимает значение TRUE тогда и только тогда, когда элементы операции, которые сохраняются с помощью автоморфизма дерева элементов для операции ?a1, являются элементами субоперации ?a2.

8.4.2 Определяющий символ отношения end_iso

KIF-формат обозначения определяющего символа отношения end_iso таков:

(end_iso ?s1 ?s2 ?s3 ?s4)

Неформальная семантика для определяющего символа отношения end_ИСО такова:

(end_iso ?s1 ?s2 ?s3 ?s4) в интерпретации теории автоморфизма дерева элементов принимает значение TRUE тогда и только тогда, когда существует один и тот же автоморфизм дерева элементов, который отображает элемент ?s1 в ?s2, а также элемент ?s3 в ?s4.

8.4.3 Определяющий символ отношения legal_map

KIF-формат обозначения определяющего символа отношения legal_map таков:

(legal_map ?s3 ?s4 ?a)

Неформальная семантика для определяющего символа отношения legal_map такова:

(legal_map ?s3 ?s4 ?a) в интерпретации теории автоморфизма дерева элементов принимает значение TRUE тогда и только тогда, когда существует автоморфизм дерева элементов, который отображает элемент ?s3 в ?s4, сохраняя допустимые элементы для операции ?a.

8.4.4 Определяющий символ отношения tree_map

KIF-формат обозначения определяющего символа отношения tree_map таков:

(tree_map ?s1 ?s2 ?a1 ?a2)

Неформальная семантика для определяющего символа отношения tree_map такова:

(end_iso ?s1 ?s2 ?a1 ?a2) в интерпретации теории автоморфизма дерева элементов принимает значение TRUE тогда и только тогда, когда любое отображение на допустимое дерево элементов, которое сохраняет допустимые элементы ?a1 и отображает элемент ?s1 в ?s2, сохраняя дерево операций для ?a2.

8.5 Определения теории автоморфизма дерева элементов

Определения, введенные теорией автоморфизма дерева элементов, таковы:

8.5.1 Определение 1

Один и тот же автоморфизм, который отображает элемент ?s1 в ?s2, а также отображает элемент ?s3 в ?s4.

```
(forall (?s1 ?s2 ?s3 ?s4) (iff (end_iso ?s1 ?s2 ?s3 ?s4)
(exists (?s5 ?s6)
        (and (precedes ?s5 ?s1)
             (precedes ?s5 ?s3)
             (precedes ?s6 ?s2)
             (precedes ?s6 ?s4)
             (tree_equiv ?s1 ?s2)
             (tree_equiv ?s3 ?s4)
             (tree_equiv ?s5 ?s6))))
```

8.5.2 Определение 2

Автоморфизм дерева элементов, который отображает элемент ?s3 в ?s4 и сохраняет допустимые элементы операции ?a.

```
(forall (?s ?s3 ?s4) (iff (legal_map ?s3 ?s4 ?a)
(and (occurrence_of ?s3 ?a)
     (occurrence_of ?s4 ?a)
     (tree_equiv ?s3 ?s4)
     (forall (?s1)
          (implies (occurrence_of ?s1 ?a)
               (exists (?s2)
```

```
                (and (occurrence_of ?s2 ?a)
                     (legal ?s2)
                     (end_iso ?s1 ?s2 ?s3 ?s4))))))))
```

8.5.3 Определение 3

Допустимый автоморфизм дерева элементов сохраняет допустимые элементы ?a1 и отображает элемент ?s1 в ?s2, а также сохраняет дерево операций для ?a2.

```
(forall (?a1 ?a2 ?s1 ?s2) (iff (tree_map ?s1 ?s2 ?a1 ?a2)
(and (legal_map ?s1 ?s2 ?a1)
     (forall (?s3 ?s4)
          (implies (min_precedes ?s3 ?s4 ?a)
               (exists (?s5 ?s6)
                    (and (min_precedes ?s5 ?s6 ?a2)
                         (end_iso ?s3 ?s5 ?s1 ?s2)
                         (end_iso ?s4 ?s6 ?s1 ?s2))))))))
```

8.6 Аксиомы теории автоморфизма дерева элементов

Ниже приведен полный набор аксиом для теории автоморфизма дерева элементов.

8.6.1 Аксиома 1

Если ?a2 является убиквитарной операцией для операции ?a1, то элементы субоперации на дереве операций для операции ?a2 сохраняются с помощью автоморфизмов дерева элементов, которое сохраняет допустимые элементы операции ?a1.

```
(forall (?a1 ?a2 ?s1 ?s2)
     (implies (and (ubiquitous ?a1 ?a2)
               (min_precedes ?s1 ?s2 ?a2))
          (exists (?s3 ?s4 ?s5 ?s6)
               (and (tree_equiv ?s1 ?s2)
                    (tree_equiv ?s3 ?s4)
                    (occurrence ?s5 ?a1)
                    (occurrence ?s6 ?a1)
                    (legal_equiv ?s5 ?s6)
                    (end_iso ?s3 ?s4 ?s5 ?s6)))))
```

8.6.2 Аксиома 2

Если ?a2 является убиквитарной операцией для операции ?a1, то любые автоморфизмы дерева элементов, которые сохраняют элементы субоперации на дереве операций для операции ?a2, также сохраняют допустимые элементы операции ?a1.

```
(forall (?a1 ?a2 ?s1 ?s2)
   (implies (and (ubiquitous ?a1 ?a2)
         (min_precedes ?s1 ?s2 ?a2))
      (not (exists (?s3 ?s4)
         (and (occurrence ?s3 ?a1)
               (occcurrence ?s4 ?a1)
               (legal_equiv ?s3 ?s4)
               (end_iso ?s1 ?s2 ?s3 ?s4))))))
```

8.6.3 Аксиома 3

Каждый элемент операции, сохраненный с помощью автоморфизмов дерева элементов, сохраняет допустимые элементы операции ?a1 и является элементом субоперации убиквитарной операции.

```
(forall ?a1 ?a2 ?s1 ?s2 ?s3 ?s4)
      (implies (and (ubiquitous ?a1 ?a2)
            (occurrence ?s3 ?a1)
            (occurrence ?s4 ?a1)
            (legal_equiv ?s3 ?s4)
            (end_iso ?s1 ?s2 ?s3 ?s4))
         (exists (?o1 ?o2)
            (and (occurrence ?o1 ?a1)
                  (occurrence ?o2 ?a1)
                  (subactivity_occurrence ?s1 ?o1)
                  (subactivity_occurrence ?s2 ?o1)))))
```

9 Теория оболочки операций

Базовая теория оболочки операций устанавливает аксиомы, относящиеся к взаимодействию между элементами внутренних и внешних операций. В частности, последними могут быть операции, которые обязательно должны возникать при появлении допустимых элементов операции, а также могут быть внешние операции, которые запрещают их появление при наличии некоторых элементов операции.

9.1 Примитивные отношения в теории оболочки операций

В нелогической лексике теории оболочки операций содержатся два примитивных символа отношений, а именно:

- envelope (оболочка);

- umbra (тень).

9.2 Определяющие отношения в теории оболочки операций

Отсутствуют.

9.3 Связь с другими группами аксиом

К базовым теориям, которые необходимы для теории оболочки операций, относятся:

- pslcore.th;
- occtree.th;
- subactivity.th;
- atomic .th;
- complex.th;
- actocc.th.

Дефиниционалное расширение, требуемое в теории оболочки операций, таково:

— embedding.def.

9.4 Неформальная семантика теории оболочки операций

9.4.1 Примитивный символ отношения envelope

KIF-формат обозначения символа отношения envelope таков:

(envelope ?a1 ?a2 ?s)

Неформальная семантика для символа отношения envelope такова:

(envelope ?a1 ?a2) в интерпретации теории оболочки операции принимает значение TRUE тогда и только тогда, когда дерево операций для операции ?a2 с корневым узлом ?s содержит все внешние элементы операции, возникающей для допускаемых элементов операции ?a1.

9.4.2 Примитивный символ отношения umbra

KIF-формат обозначения элементарного символа отношения umbra таков:

(umbra ?a1 ?a2)

Неформальная семантика для символа отношения umbra такова:

(umbra ?a1 ?a2) в интерпретации теории оболочки операции принимает значение TRUE тогда и только тогда, когда дерево операций для операции ?a2

с корневым узлом ?s содержит все внешние элементы операции, запрещенной в допускаемых элементах операции ?a1.

9.5 Определения теории оболочки операций

Отсутствуют.

9.6 Аксиомы теории оболочки операций

Ниже приведен полный набор аксиом для теории оболочки операций.

9.6.1 Аксиома 1

Каждое дерево операций имеет свою оболочку операций.

```
(forall (?a2 ?s)
        (implies (root ?s ?a2)
                 (exists (?a1)
                         (envelope ?a1 ?a2 ?s))))
```

9.6.2 Аксиома 2

Каждое дерево операций имеет свою «теневую» операцию.

```
(forall (?a2 ?s)
        (implies (root ?s ?a2)
                 (exists (?a1)
                         (umbra ?a1 ?a2 ?s))))
```

9.6.3 Аксиома 3

Любая активная область оболочки не является активной областью первоначальной (исходной) операции.

```
(forall (?a1 ?a2 ?s ?s1)
        (implies (and (envelope ?a1 ?a2 ?s)
                      (live_branch ?s1 ?s ?a1))
            (not (dead_branch ?s1 ?s ?a2))))
```

9.6.4 Аксиома 4

Любая активная «теневая» область не является активной областью первоначальной (исходной) операции.

```
(forall (?a1 ?a2 ?s ?s1)
        (implies (and (umbra ?a1 ?a2 ?s)
                      (live_branch ?s1 ?s ?a1))
            (not (live_branch ?s1 ?s ?a2))))
```

9.6.5 Аксиома 5

Любая активная часть операции является активной частью ее оболочки.

```
(forall (?a1 ?a2 ?s ?s1)
        (implies (and (envelope ?a1 ?a2 ?s)
                      (live_branch ?s1 ?s ?a2))
          (live_branch ?s1 ?s ?a1)))
```

9.6.6 Аксиома 6

Любая неактивная часть операции является активной «теневой» частью.

```
(forall (?a1 ?a2 ?s ?s1)
        (implies (and (umbra ?a1 ?a2 ?s)
                      (dead_branch ?s1 ?s ?a2))
          (live_branch ?s1 ?s ?a1)))
```

9.6.7 Аксиома 7

Оболочка для операции ?a1 является неограниченной в активных областях для операции ?a1.

```
(forall (?a1 ?a2 ?s ?o)
        (implies (and (envelope ?a1 ?a2 ?s)
                      (occurrence ?o ?a1)
                      (root_occ ?s ?o))
          (unconstrained ?o)))
```

9.6.8 Аксиома 8

«Теневая» область для операции ?a1 является неограниченной в неактивных областях для операции ?a1.

```
(forall (?a1 ?a2 ?s ?o)
        (implies (and (umbra ?a1 ?a2 ?s)
                      (occurrence ?o ?a1)
                      (root_occ ?s ?o))
          (unconstrained ?o)))
```

Приложение А
(обязательное)

Использование идентификаторов стандартного языка описания синтаксиса (ASN.1) в стандартах подкомитета ПК 4

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

iso standard 18629 part 13 version 1

Смысл этого обозначения указан в ИСО/МЭК 8824-1 и описан в ИСО 18629-1.

Приложение В
(справочное)
Пример описания производственного процесса в соответствии с настоящим стандартом

Целью данного приложения является предоставление подробного сценария использования языка спецификаций процесса (PSL) согласно ИСО 18629 в целях разделения информации в работах, связанных с выполнением множества производственных функций.

Указанный сценарий является функционально совместимым производственным сценарием. Последнее означает, что его целью является иллюстрация того, как PSL-язык можно использовать для облегчения обмена производственной информацией в промышленной среде. В основном в этом сценарии делается акцент на вопросах обмена информацией между инженером-технологом и диспетчером предприятия мелкосерийного производства.

В данном приложении более подробно рассмотрен контрольный пример, приведенный в ИСО 18629-11:2005 (приложение С) для иллюстрации некоторых положений теорий продолжительности и упорядочения операций в описании производственного процесса изготовления изделия, именуемого GT-350.

В.1 Производственные процессы изготовления изделия GT-350

В данном разделе различные внутрицеховые процессы объединены в совокупность операций высокого уровня, предписываемую для изготовления изделия GT-350. Как указано в структуре конечного изделия GT-350 (см. ИСО 18629-11:2005, таблица С.1), комплектующие этого изделия либо закупают, либо получают по договору с субподрядчиком, либо изготовляют на самом предприятии. При описании этих процессов рассматривают операции, выполняемые при внутрихозяйственном изготовлении комплектующих. Подобный «вертикальный» анализ производственного процесса формирует полную картину исходя из абстрактной операции «изготовление изделия GT-

350», которая расширена вниз до внутрицехового уровня.

Как показано на рисунке В.1, процесс изготовления изделия GT-350 разбит на шесть основных частей, первые пять из которых таковы: изготовление интерьера, изготовление привода, изготовление кузова, изготовление двигателя и изготовление шасси. Эти части процесса неупорядочены по отношению друг к другу, однако они должны быть завершены до начала окончательной сборки.

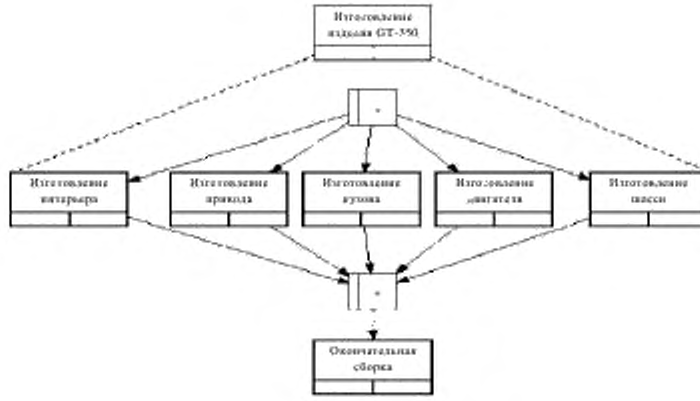


Рисунок В.1 – Блок-схема производственного процесса высокого уровня, применяемого для изготовления изделия GT-350 [8]

PSL-представление продолжительности и упорядочения операций в процессе высокого уровня расширяет основанное на внешнем ядре представление следующим образом:

```
(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)
        (<=> (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)
(soo_precedes (soomap ?occ1) (soomap ?occ6) make_gt350)
(soo_precedes (soomap ?occ2) (soomap ?occ6) make_gt350)
(soo_precedes (soomap ?occ3) (soomap ?occ6) make_gt350)
(soo_precedes (soomap ?occ4) (soomap ?occ6) make_gt350)
(soo_precedes (soomap ?occ5) (soomap ?occ6) make_gt350)))))))
```

В этом представлении символ отношения soo_precedes используется для указания ограничений при упорядочении элементов субопераций make_chassis, make_interior, make_drive, make_engine и final_assembly. Неформально каждая из стрелочек, изображенных на рисунке В.1, соответствует символу формулы soo_precedes. Символ формулы soomap используется для различения любых возможных многочисленных элементов субопераций.

Каждая из указанных абстрактных операций будет рассмотрена дополнительно, однако для предложенного в данном приложении примера все из них анализироваться не будут.

На основе IDEF3-представления процесса из абстрактных операций, встречающихся на различных этапах производственного процесса, будут взяты некоторые примеры описаний процессов, использующих PSL-представление продолжительности и упорядочения операций, которое рассмотрено в настоящем стандарте.

В.1.1 Абстрактная операция «Изготовление двигателя»

Сборку двигателя изделия GT-350 осуществляют на нескольких подразделениях независимых рабочих станций (CMW). Процесс изготовления двигателя показан на рисунке В.2. Двигатель состоит из блока цилиндров, передаточного механизма и приводных соединений. Подпроцессы сборки подробно описаны ниже. Сборку двигателя изделия GT-350 производили на сборочной линии, на что было затрачено 5 мин.

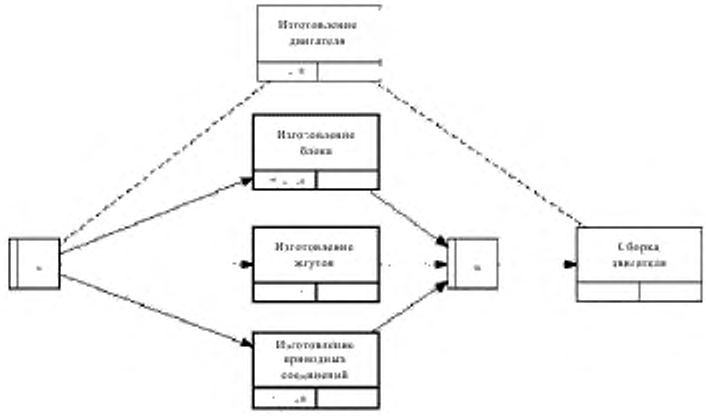


Рисунок В.2 – Блок-схема процесса изготовления двигателя изделия GT-350 [8]

PSL-представление продолжительности и упорядочения некоторых операций и связанной с производством информации на этапе изготовления двигателя таково:

```
(subactivity make_block make_engine)
(subactivity make-harness make_engine)
(subactivity make-wires make_engine)
(subactivity assemble_engine make_engine)

(forall (?occ)
    (<=> (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)
                   (= (duration (beginof ?occ1) (endof ?occ1)) 10)
                   (= (duration (beginof ?occ2) (endof ?occ2)) 5)
                   (= (duration (beginof ?occ3) (endof ?occ3)) 12)
                   (= (beginof ?occ4) (time_add (endof ?occ3) 10))
                   (subactivity_occurrence ?occ1 ?occ)
                   (subactivity_occurrence ?occ2 ?occ)
                   (subactivity_occurrence ?occ3 ?occ)
                   (subactivity_occurrence ?occ4 ?occ)
                   (soo_precedes (soomap ?occ1) (soomap ?occ4) make_gt350)
                   (soo_precedes (soomap ?occ2) (soomap ?occ4) make_gt350)
                   (soo_precedes (soomap ?occ3) (soomap ?occ4) make_gt350)))))
```

В этом представлении символ отношения soo_precedes используют для указания ограничений при упорядочении элементов субопераций make_block, make_harness, make_wires и make_engine. Неформально каждая из стрелочек, изображенных на рисунке В.2, соответствует символу формулы soo_precedes. Символ формулы soomap используют для различения любых возможных

многочисленных элементов субопераций.

Кроме того, это представление указывает, что продолжительность элемента субоперации make_block составляет 10 единиц времени, продолжительность элемента субоперации make_harness – 5 единиц времени, а продолжительность элемента субоперации make_wires – 12 единиц времени. Кроме того, это представление использует символ функции time_add для указания того, что элемент субоперации assemble_engine начинается на 10 единиц времени позже элемента make_wires.

B.1.2 Абстрактная операция «Изготовление блока цилиндров»

Блок цилиндров изделия GT-350 изготовляют в виде подсборки двигателя этого изделия, что предусматривает работы литейного производства и цеха механообработки (см. рисунок B.3).



Рисунок B.3 – Блок-схема процесса изготовления блока цилиндров изделия GT-350 [8]

Далее приведено PSL-представление продолжительности и упорядочения некоторых операций и связанной с производством информации на этапе изготовления блока цилиндров:

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

(forall (?occ)
  (<=> (occurrence_of ?occ make_block)
       (exists (?occ1 ?occ2)
            (and (occurrence_of ?occ1 produce_molded_metal)
                 (occurrence_of ?occ2 machine_block)
                 (= (beginof ?occ2) (time_add (endof ?occ1) 12))
                 (soo_precedes (soomap ?occ1) (soomap ?occ2) make_block)))))
```

Данное представление использует символ функции time_add для указания того, что элемент субоперации machine_block subactivity начинается на 12 единиц времени позже окончания элемента субоперации produce_molded_metal.

В.1.3 Абстрактная операция «Изготовление жгута»

Передаточный механизм изделия GT-350 изготовляют в виде подсборки двигателя этого изделия, что предусматривает работы кабельной мастерской (см. рисунок В.4). Рисунок В.5 более подробно иллюстрирует процесс изготовления жгутов, сборку которых выполняет монтажник в кабельной мастерской, что занимает у него 10 мин на один комплект.



Рисунок В.4 – Блок-схема процесса изготовления жгута для изделия GT-350 [8]

PSL-представление продолжительности и упорядочения некоторых операций и связанной с производством информации на этапе изготовления жгутов таково:

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

(forall (?occ)
  (<=> (occurrence_of ?occ make_harness)
       (exists (?occ1 ?occ2 ?occ3)
            (and (occurrence_of ?occ1 make_harness_wire)
                 (occurrence_of ?occ2 assemble_harness)
                 (leaf_occ ?occ3 ?occ1)
                 (soo_precedes (soomap ?occ3) (soomap ?occ2) make_harness)))))
```

Данное представление формализует спецификацию процесса, изображенного на рисунке В.5, и использует символ отношения soo_precedes для указания ограничений элементов субопераций make_harness_wires и assemble_wires. Неформально каждая из стрелочек, изображенных на рисунке В.5, соответствует soo_precedes. Soomap используют для различения любых возможных многочисленных элементов субопераций.



Рисунок В.5 – Блок-схема процесса изготовления жгута проводки [8]

В.1.4 Изготовление жгутов проводки

Комплект жгутов проводки для изделия GT-350 изготовляют в виде подсборки двигателя этого изделия, что предусматривает работы кабельной мастерской.

PSL-представление продолжительности и упорядочения некоторых операций и связанной с производством информации на этапе изготовления проводов таково:

```
(subactivity extrude make_harness_wire)
(subactivity twist make_harness_wire)
(subactivity jacket make_harness_wire)
(primitive extrude)
(primitive twist)
(primitive jacket)

(forall (?occ)
  (<=> (occurrence_of ?occ make_harness_wire)
       (exists (?occ1 ?occ2 ?occ3 ?occ4)
          (and (occurrence_of ?occ1 extrude)
               (occurrence_of ?occ2 twist)
               (occurrence_of ?occ3 jacket)
               (occurrence_of ?occ4 assemble)
               (soo_precedes (soomap ?occ1) (soomap ?occ2) make_harness_wire)
               (soo_precedes (soomap ?occ3) (soomap ?occ4) make_harness_wire)
               (soo_precedes (soomap ?occ2) (soomap ?occ3) make_harness_wire)))))
```

Данное представление формализует спецификацию процесса, изображенного на рисунке В.6, а также использует soo_precedes для указания ограничений элементов субопераций экструзии оболочки, скрутки проводов, заключения проводов в оболочку и сборки. Неформально каждая из стрелочек, изображенных на рисунке В.6, соответствует soo_precedes. Soomap используют для различения любых возможных элементов субопераций.



Рисунок В.6 – Блок-схема процесса изготовления проводов для изделия GT-350 [8]

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»
ИСО 18629-1:2004	–	*
ИСО 18629-11:2005	–	*
ИСО 18629-12:2005	–	*

* 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 10303-11 — Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 11. Методы описания: справочное руководство по языку EXPRESS

[2] ИСО 10303-41 — Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 41. Интегрированные родовые ресурсы. Основы описания продукции и программного обеспечения

[3] ИСО 10303-42 — Системы промышленной автоматизации и интеграция. Представление данных о продукции и обмен данными. Часть 42. Интегрированные родовые ресурсы. Геометрическое и топологическое представление

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

[5] ИСО 13584 (все части) — Системы промышленной автоматизации и интеграция. Библиотека данных на детали

[6] ИСО 15531 (все части) — Системы промышленной автоматизации и интеграция. Управленческая информация промышленным производством

[7] ИСО 15926 (все части) — Системы промышленной автоматизации и интеграция. Интеграция данных о сроке службы нефтехимических установок, включая установки по добыче нефти и газа

[8] Федеральные стандарты на обработку информации, публикация 184, Описание интеграции для информационного моделирования (IDEF3), FIPS PUB 184, Национальный институт стандартов и технологии, декабрь 1993 г., IDEF3. Доступны в 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