

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

СОГЛАСОВАНО

Генеральный директор
ЗАО «АйТи»



Вакнев О.Р.

2011 г.

УТВЕРЖДАЮ

Ректор НИУ ИТМО



Васильев В.Н.

2011 г.

МНОГОПРОФИЛЬНАЯ ИНСТРУМЕНТАЛЬНО-
ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА СОЗДАНИЯ
И УПРАВЛЕНИЯ РАСПРЕДЕЛЕННОЙ СРЕДОЙ
ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ CLAVIRE

ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА СОЗДАНИЯ РАСПРЕДЕЛЕННЫХ
КОРПОРАТИВНЫХ ПРОГРАММНЫХ ПРИЛОЖЕНИЙ
ДЛЯ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ СЛОЖНЫХ СИСТЕМ
И ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ ПО УПРАВЛЕНИЮ
КОМПЛЕКСНЫМИ ОБЪЕКТАМИ (МИП-К)

Руководство программиста

ЛИСТ УТВЕРЖДЕНИЯ

RU.СНАБ.80066-06 33 02-ЛУ

Ине.№ подл.	Подп. и дата
Взам.ине.№	Подп. и дата
Ине.№ дубл.	Подп. и дата

Представители
Организации-разработчика

Руководитель разработки,
профессор НИУ ИТМО

Бухановский А.В.

“29” декабря 2011 г.

Ответственный исполнитель,
с.п.е. НИУ ИТМО

Луценко А.Е.

“19” декабря 2011 г.

Нормоконтролер
ведущий инженер НИУ ИТМО

Позднякова Л.Г.

“29” декабря 2011 г.

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

УТВЕРЖДЕН
RU.СНАБ.80066-06 33 02-ЛУ

**МНОГОПРОФИЛЬНАЯ ИНСТРУМЕНТАЛЬНО-
ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА СОЗДАНИЯ
И УПРАВЛЕНИЯ РАСПРЕДЕЛЕННОЙ СРЕДОЙ
ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ CLAVIRE**

**ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА СОЗДАНИЯ РАСПРЕДЕЛЕННЫХ
КОРПОРАТИВНЫХ ПРОГРАММНЫХ ПРИЛОЖЕНИЙ
ДЛЯ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ СЛОЖНЫХ СИСТЕМ
И ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ ПО УПРАВЛЕНИЮ
КОМПЛЕКСНЫМИ ОБЪЕКТАМИ (МИП-К)**

Руководство программиста

RU.СНАБ.80066-06 33 02

Листов 36

Ине.№ подл.	
Подп. и дата	
Взам.ине.№	
Ине.№ дубл.	
Подп. и дата	

АННОТАЦИЯ

Документ содержит руководство программиста технологической платформы создания распределенных корпоративных программных приложений для компьютерного моделирования сложных систем и поддержки принятия решений по управлению комплексными объектами (МИТП-К) RU.СНАБ.80066-06 01 41. Технологическая платформа МИТП-К входит в состав многопрофильной инструментально-технологической среды (МИТП) CLAVIRE (Cloud Applications Virtual Environment) RU.СНАБ.80066-06. Она предназначена для создания предметно-ориентированных высокопроизводительных композитных приложений корпоративного уровня в неоднородной распределенной вычислительной среде с обеспечением эффективного использования вычислительных ресурсов организации потребителя. МИТП-К разработана в ходе выполнения проекта «Создание распределенной вычислительной среды на базе облачной архитектуры для построения и эксплуатации высокопроизводительных композитных приложений» (Договор № 21057 от 15 июля 2010 г., шифр 2010-218-01-209) в рамках реализации постановления Правительства РФ № 218 «О мерах государственной поддержки развития кооперации российских высших учебных заведений и организаций, реализующих комплексные проекты по созданию высокотехнологичного производства».

СОДЕРЖАНИЕ

1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММЫ	4
1.1. Назначение программы	4
1.2. Условия применения программы	6
2. ХАРАКТЕРИСТИКИ ПРОГРАММЫ	7
2.1. Технологические функции программы	7
2.2. Порядок использования программы	7
2.3. Режимы взаимодействия пользователя с программой	9
2.4. Общая архитектура МИТП-К	9
2.5. Основные компоненты МИТП-К	11
3. ОБРАЩЕНИЕ К ПРОГРАММЕ	12
3.1. Алгоритм построения композитного приложения	12
3.2. Описание композитных приложений на языке EasyFlow	15
3.2.1. Основные элементы языка	15
3.2.2. Построение workflow	22
3.2.3. Способы структурирования программы	26
3.3. Специализированные элементы языка для платформы МИТП-К	29
3.3.1. Явное задание ресурса	29
3.3.2. Определение приоритета workflow и шагов	30
4. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ	30
5. СООБЩЕНИЯ	31
ПЕРЕЧЕНЬ СОКРАЩЕНИЙ	35

1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММЫ

1.1. Назначение программы

Технологическая платформа создания распределенных корпоративных программных приложений для компьютерного моделирования сложных систем и поддержки принятия решений по управлению комплексными объектами (МИТП-К) RU.СНАБ.80066-06 01 41 входит в состав многопрофильной инструментально-технологической среды (МИТП) CLAVIRE (Cloud Applications Virtual Environment) RU.СНАБ.80066-06. Она предназначена для создания предметно-ориентированных высокопроизводительных композитных приложений корпоративного уровня в неоднородной распределенной вычислительной среде с обеспечением эффективного использования вычислительных ресурсов организации потребителя.

МИТП-К представляет собой комплекс программного обеспечения для разработки, настройки и эксплуатации сред распределенных вычислений корпоративного уровня, предназначенный для:

- 1) эффективного управления вычислительными, информационными и программными ресурсами неоднородной корпоративной среды в рамках модели облачных вычислений (частного облака);
- 2) создания, исполнения, управления и предоставления сервисов доступа к предметно-ориентированным высокопроизводительным композитным приложениям, функционирующим на основе облака распределенных прикладных сервисов в корпоративной среде¹;
- 3) обеспечения функционирования программно-аппаратного комплекса (ПАК) поддержки инфраструктуры предметно-ориентированных облачных вычислений в различных предметных областях на основе корпоративных ресурсов организации потребителя.

В соответствии с ТТ к основным функциям технологической платформы МИТП-К, обеспечивающим решение поставленных задач, относятся:

¹ Прикладным сервисом называется программа, входные и выходные данные которой интерпретируются в терминах конкретной предметной области и которая в распределенной среде через публичную либо корпоративную сеть передачи данных доступна для выполнения без использования вычислительных и программных ресурсов на стороне пользователя.

1. Поддержка разработки и исполнения композитных приложений на основе ресурсов корпоративной распределенной среды – набора программно-аппаратных комплексов, принадлежащих одному владельцу, связанных локальной вычислительной сетью и предназначенных для решения задач, координируемых единым набором требований.
2. Динамическое управление (мониторинг состояния, запуск приложений, передача данных, распределение нагрузки) в автоматическом режиме набором распределенных ресурсов, доступных в корпоративной распределенной среде.
3. Автоматическая оптимизация по времени процесса использования доступных вычислительных ресурсов и прикладных сервисов в корпоративной распределенной среде.
4. Представление описания композитных приложений в корпоративной распределенной среде на основе цепочек заданий (workflow), обеспечивающих запуск, выполнение, остановку и возобновление работы цепочки заданий в ручном и автоматическом режимах.
5. Унифицированный доступ к вычислительным ресурсам корпоративной распределенной среды на основе интерфейсов доступа в составе программно-аппаратной архитектуры SMP, MPP, GPGPU, СВЕА.
6. Поддержка процесса установки и первоначальной конфигурации технологической платформы и ее составных частей на ресурсах корпоративной распределенной среды.
7. Поддержка многопользовательского режима в корпоративной распределенной среде.
8. Квотирование использования ресурсов корпоративной распределенной среды.
9. Каталогизация входных данных пользователей на основе метаданных.
10. Администрирование и контроль работы с дифференцированными правами администраторов в рамках единой политики доступа к ресурсам корпоративной распределенной среды.
11. Модификация знаний, используемых системой в ручном и автоматическом режимах.
12. Функционирование сервисов резервирования и отката исправлений для пользовательских данных в удаленном хранилище в составе корпоративной распределенной среды.

13. Функционирование механизмов конвертирования данных между различными прикладными сервисами, по заданию пользователя.

14. Функционирование сервисов, обеспечивающих интерфейс с многофункциональными и предметно-ориентированными системами визуализации результатов расчетов с использованием прикладных сервисов и композитных приложений на их основе.

В данном документе рассматриваются способы обращения к МИТП-К с целью описания композитных приложений, построенных на основе прикладных сервисов (пакетов), доступных МИТП в ходе ее функционирования. Для описания композитных приложений применяется предметно-ориентированный язык описания EasyPackage.

1.2. Условия применения программы

Для развертывания компонентов МИТП-К необходима вычислительная система под управлением ОС Windows (XP и выше) с установленной средой Silverlight 4.0, или Linux (с ядром 2.6.22 и выше), с установленной средой Mono Framework с поддержкой библиотек .NET 2.0 и выше (рекомендуется версия Mono Framework 2.6 или выше). Для корректного функционирования необходимо наличие установленного web-сервера с поддержкой технологии ASP .NET WebServices, WCF, Silverlight и удаленного развертывания сервисов (с использованием технологии WebDeploy). Примером web-сервера, соответствующего предъявленным требованиям, может служить Microsoft IIS версии 7.0 или выше.

Дополнительно для функционирования МИТП-К должен быть установлен сервер баз данных MongoDB версии 1.6.5. В ходе установки и настройки используются стандартные конфигурации указанных программных средств, не требующие специфической модификации. После установки необходимо осуществить запуск сервера баз данных для локального использования (localhost).

Компоненты МИТП-К функционируют на вычислительной системе – серверной ЭВМ со следующими минимальными характеристиками:

- тип процессоров: Intel-совместимый;
- число ядер – не менее 4;
- число процессоров – не менее 2;
- тактовая частота каждого процессора – не ниже 2.0 ГГц;
- оперативная память (на ядро) – не менее 2.0 ГБ;

- дисковая подсистема – не менее 5×250 ГБ RAID5;
- пропускная способность сетевых интерфейсов – не менее 1 Гбит/с.

Для взаимодействия с другими модулями системы требуется наличие выхода в Интернет или локальную сеть (если web-сервисы других подсистем доступны из локальной сети) с соответствующей поддержкой со стороны оборудования.

В целях увеличения производительности и реактивности МИТП-К отдельные компоненты могут функционировать на разных вычислительных системах в рамках общей локальной сети.

2. ХАРАКТЕРИСТИКИ ПРОГРАММЫ

2.1. Технологические функции программы

МИТП-К призвана обеспечить выполнение следующих технологических функций:

- 1) использование вычислительных мощностей неоднородной корпоративной среды в рамках модели облачных вычислений (частного облака);
- 2) использование доступных прикладных пакетов, установленных на ресурсах корпоративной среды, для решения задач избранной предметной области;
- 3) формирование композитных приложений, использующих в своем составе вызовы предметно-ориентированных сервисов, реализованных с использованием доступных системе прикладных пакетов;
- 4) подготовка файлов входных данных и расчетных параметров, необходимых для выполнения композитных приложений;
- 5) автоматизированное определение целевых ресурсов, доступных для исполнения сервисов в составе композитного приложения;
- 6) запуск процесса выполнения сформированных композитных приложений в распределенной вычислительной среде;
- 7) мониторинг процесса исполнения пользовательского задания в распределенной вычислительной среде;
- 8) работа с пользовательскими данными в удаленном хранилище.

2.2. Порядок использования программы

Порядок использования МИТП CLAVIRE следующий.

1. Пользователь авторизуется через графический интерфейс платформы (представленный отдельным web-приложением или информационным web-порталом), что дает ему возможность доступа к соответствующим сервисам МИТП.
2. Пользователь через соответствующий web-интерфейс формирует композитное приложение на предметно-ориентированом языке EasyFlow в виде описания абстрактного WF.
3. При этом может использоваться как графическая, так и текстовая форма представления композитных приложений. В ряде случаев пользователь может не конструировать приложение, а использовать готовые WF, разработанные другими пользователями.
4. Для подготовленного описания композитного приложения пользователь конфигурирует условия вычислений: определяет требуемые параметры WF, редактирует (при необходимости) его описание, готовит и загружает в хранилище входные данные для расчетов.
5. Пользователь запускает задачу на выполнение. Автоматически определяется наилучший набор сервисов, соответствующий вызовам в данном абстрактном WF (планирование). При необходимости процесс планирования может повторяться несколько раз в ходе выполнения композитного приложения.
6. Пользователь (при необходимости) осуществляет мониторинг процесса исполнения (в форме динамического отображения конкретного WF). Пользователь может на время расчетов завершить рабочую сессию с МИТП и начать ее только при необходимости использования результатов (необязательно в тот момент, когда расчеты завершены).
7. Когда расчет задачи завершен, результаты перемещаются из распределенной среды в соответствующее хранилище данных МИТП. Пользователь может получить доступ к результатам расчетов (в форме выдачи результатов соответствующих прикладных пакетов) через web-интерфейс МИТП.

Более детально порядок применения МИТП CLAVIRE рассмотрен в документе «Описание применения» RU.СНАБ.80066-06 31 01.

2.3. Режимы взаимодействия пользователя с программой

Для взаимодействия программиста (разработчика композитных приложений) с платформой МИТП предусмотрены три основных режима.

- Режим использования шаблонов, соответствующих типовому применению вызовов отдельных прикладных пакетов. В этом случае пользователь выбирает только пакет (или вид пакета по предметному описанию) и отвечает за подготовку входных данных. Выбор условий выполнения, настройка виртуального вычислительного ресурса и запуск пакета осуществляются автоматически.
- Режим ручного создания композитных приложений подразумевает описание композитного приложения в окне редактирования на языке EasyFlow, с непосредственным определением режимов вызова прикладных пакетов, необходимых пользователю.
- Режим использования сценариев позволяет применять заранее подготовленные описания композитных приложений с ассоциированными входными данными в форме проектов МИТП. При этом пользователь по необходимости может изменять только отдельные значения параметров запуска.

Для практической реализации перечисленных выше режимов используется предметно-ориентированное описание композитных приложений на языке EasyFlow.

2.4. Общая архитектура МИТП-К

В отличие от ядра МИТП, архитектура МИТП-К (рис. 2.1) не содержит компонента информационного портала. Это связано с тем, что платформа МИТП-К ориентирована на использование в рамках одной организации. К тому же данный вариант МИТП не подразумевает работы пользователя с системами интеллектуальной поддержки при составлении WF, поэтому в МИТП-К не используются компоненты диалога поддержки принятия решений и хранения знаний. Специфика МИТП-К предполагает использование упрощенного процесса задания WF, заключающегося в выборе WF из типовых вариантов или конструировании простых WF вручную.

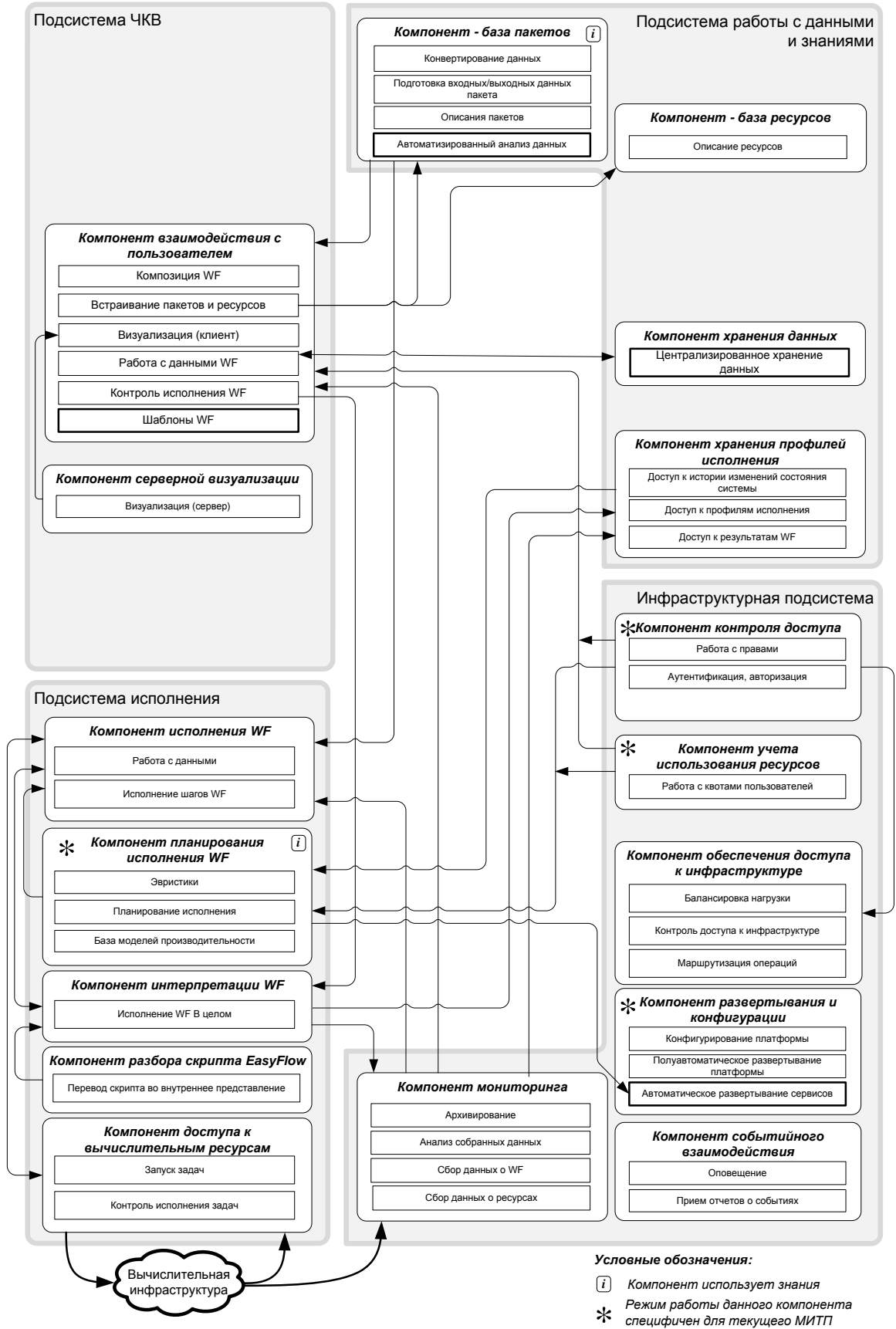


Рисунок 2.1 – Структура МИТП-К

МИТП-К выделяется среди других технологических платформ возможностью производить автоматическое развертывание сервисов на ресурсах (потому как ресурсы находятся в собственности одной организации, и к ним обеспечен административный доступ). Данная опция позволяет расширить количество вариантов при планировании, что приводит в итоге к более эффективному использованию доступных ресурсов. Для примера, при планировании может быть учтен вариант перемещения сервиса к данным наряду со стандартным сценарием перемещения данных к сервису. В таком варианте МИТП одним из основных оптимизируемых параметров при планировании становится максимально возможное использование доступных ресурсов.

Компонент учета использования ресурсов присутствует в вырожденном виде, так как используется для непосредственного учета ресурсов без тарификации. Для крупных организаций (когда организационная структура обладает сложной схемой деления и взаимодействия между отделами) предусмотрена опциональная поддержка функциональности по тарификации и квотированию для регулирования и распределения доступных мощностей.

Компонент контроля доступа использует единый сценарий доступа – «доступ к платформе». Компонент в данной платформе использует низкий уровень внутренней защиты и не использует защиту от внешних угроз, так как предполагается, что МИТП-К работает во внутренней защищенной корпоративной сети.

2.5. Основные компоненты МИТП-К

В состав МИТП-К входят все компоненты ядра МИТП (см. документ «Руководство системного программиста» МИТП RU.СНАБ.80066-06 32 01). Кроме того, в состав МИТП-К включены следующие компоненты (с указанием зависимостей между ними и компонентами ядра МИТП):

- RU.СНАБ.80066-06 01 22. Компонент серверной визуализации CLAVIRE/CSNV.
Зависимости: CLAVIRE/Storage.
- RU.СНАБ.80066-06 01 32. Компонент профилей исполнения CLAVIRE/Provenance.
Зависимости: CLAVIRE/Eventing, CLAVIRE/Monitoring.
- RU.СНАБ.80066-06 01 34. Компонент учета использования ресурсов CLAVIRE/Billing. *Зависимости:* CLAVIRE/Eventing, CLAVIRE/Monitoring.
- RU.СНАБ.80066-06 01 27. Компонент обеспечения доступа к инфраструктуре CLAVIRE/InfraAccess. *Зависимости:* нет.

- RU.СНАБ.80066-06 01 19. Компонент разбора скрипта EasyFlow CLAVIRE/EasyFlow. *Зависимости:* нет.
Кроме того, в состав МИТП-К включен ряд дополнительных компонентов:
- RU.СНАБ.80066-06 01 62. Компонент поддержки пользователя при разработке скриптов на EasyFlow CLAVIRE/UI/EasyFlowEditorServices, обеспечивающий поддержку пользователя в процессе ввода скриптов на языке EasyFlow.
- RU.СНАБ.80066-06 13 64. Компонент средств мониторинга и настройки основных компонентов платформы CLAVIRE/AdminTool, предназначенный для администрирования и контроля работы платформы. *Зависимости:* CLAVIRE/ResourceBase, CLAVIRE/PackageBase, CLAVIRE/Billing, CLAVIRE/GateKeeper, CLAVIRE/Portal.

3. ОБРАЩЕНИЕ К ПРОГРАММЕ

В ходе обращения к МИТП CLAVIRE программист формирует описание композитного приложения как абстрактный WF на языке EasyFlow. При оперировании характеристиками пакетов используется описание прикладных пакетов на языке EasyPackage (см. «Руководство программиста» для МИТП RU.СНАБ.80066-06 33 02). В данном разделе приводится общая характеристика языка EasyFlow, а также описываются операции, имеющие особое значение для технологической платформы МИТП-К.

Взаимодействие программиста с МИТП CLAVIRE осуществляется посредством интерактивной работы с интерфейсом компонента CLAVIRE/Ginger, обеспечивающего возможность построения композитного приложения, передачи пользовательских входных данных, запуска построенного приложения и получения результата. Обращение к МИТП, основные операции по работе с интерфейсом CLAVIRE/Ginger, способы запуска композитных приложений (передача управления МИТП) описаны в документе «Руководство оператора» для МИТП-К RU.СНАБ.80066-06 34 02.

3.1. Алгоритм построения композитного приложения

Типовым сценарием создания композитного приложения в МИТП-К является построение WF по шаблону. Схема соответствующего алгоритма приведена на рис. 3.1. Рассмотрев этот алгоритм, следует отметить ряд особенностей работы.

- 1) В процессе редактирования WF ядром МИТП используется информация из базы пакетов для поддержки пользователя в процессе задания параметров WF.

- 2) При задании входных данных WF пользователь имеет возможность выбрать данные, доступные в его персональном хранилище.
- 3) Для оценки предварительного планирования (с оценкой времени решения) используются: база моделей производительности (для построения априорной модельной оценки времени), база пакетов (для определения параметров пакетов, используемых в составе WF), база ресурсов (для определения актуальных характеристик вычислительных ресурсов).
- 4) Необходимо учесть, что ввиду наличия (и возможности автоматического расширения) множества экземпляров предметно-ориентированных сервисов особую значимость в задачах, решаемых с использованием МИТП-К, приобретает распараллеливание по данным. На уровне создания WF это приводит к активному использованию соответствующих специализированных блоков в языке EasyFlow, см. раздел 3.2 (перебор параметров, декомпозиция массивов данных и т.п.).

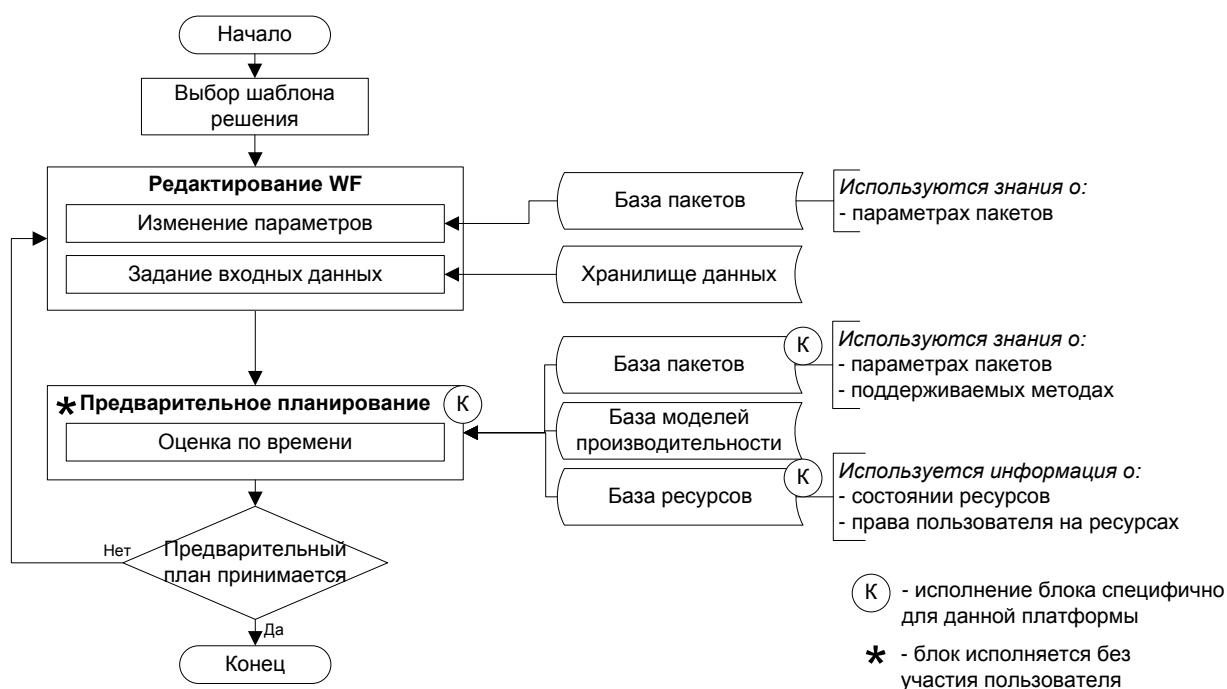


Рисунок 3.1 – Алгоритм создания типового композитного приложения в МИТП-К

Построенное композитное приложение может быть исполнено с применением базового алгоритма интерпретации и исполнения WF. При этом существует ряд особенностей, характерных в первую очередь для реализации данного процесса в рамках МИТП-К.

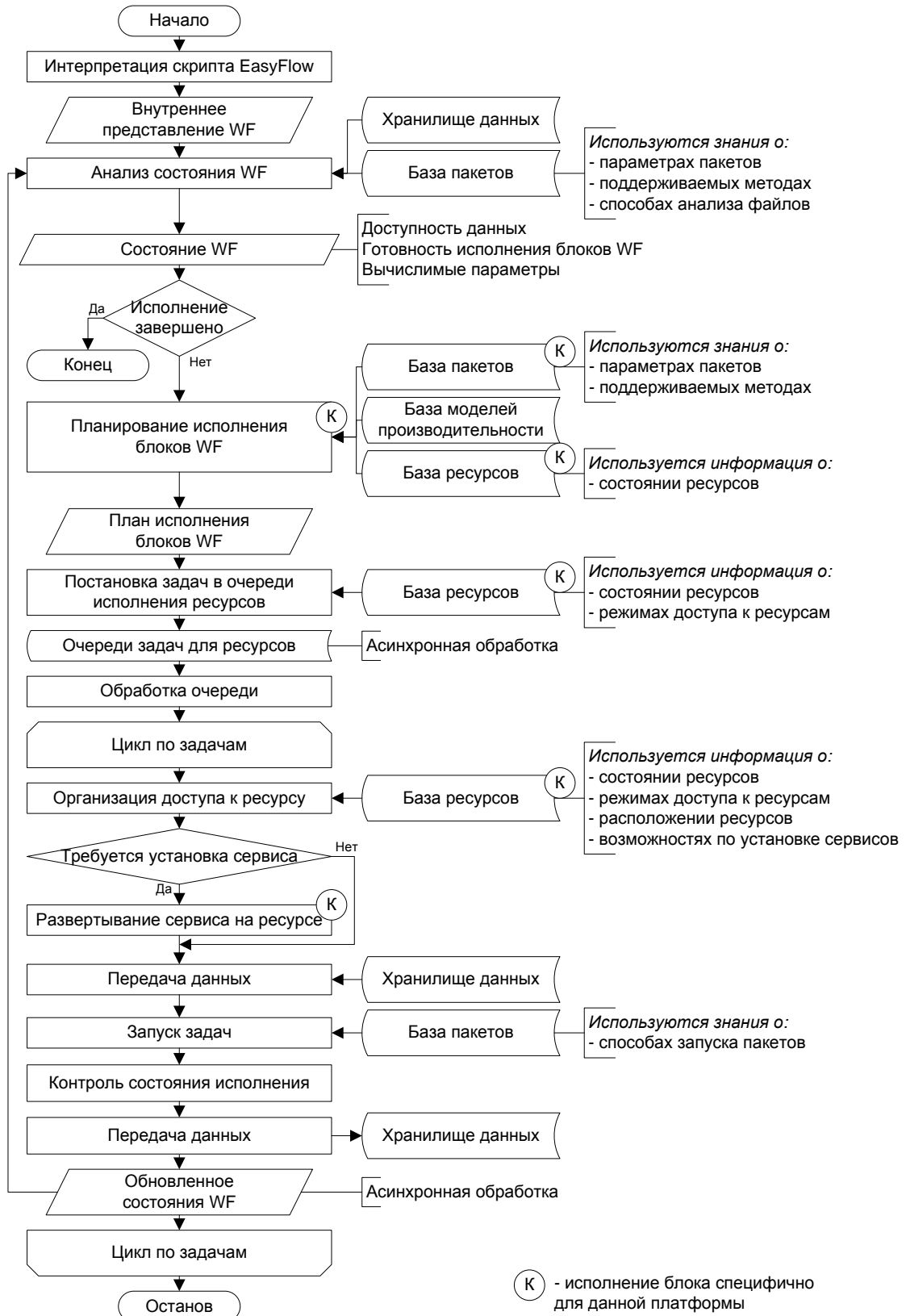


Рисунок 3.2 – Алгоритм исполнения композитного приложения в МИП-К

- 1) Использование специализированного режима планирования. В рамках данной платформы предполагается возможность доступа к ресурсам в режиме

повышенного приоритета. Таким образом, в ходе планирования необходимо учитывать: а) возможность монопольного доступа к ряду ресурсов; б) возможность установки вычислительных сервисов на ресурсы.

- 2) Возможность установки вычислительных сервисов на ресурсы приводит к появлению дополнительного шага в ходе обработки задач модулем исполнения WF.
- 3) Активное применение распараллеливания приводит к необходимости применения специализированных эвристик планирования при планировании WF, призванных обеспечить оптимальность процедуры распараллеливания с учетом набора доступных ресурсов.
- 4) В процессе планирования не учитывается профильная информация о пользователе ввиду отсутствия индивидуальных тарификации и квотирования.

3.2. Описание композитных приложений на языке EasyFlow

Язык описания композитных приложений в форме WF EasyFlow (далее – Язык или EasyFlow) предназначен для описания инструкций с целью выполнения композитных приложений в распределенной вычислительной среде. Он предоставляет конечному пользователю гибкие возможности по заданию различных форм потоков вычислений (WF), в рамках которых происходят выполнение различных прикладных пакетов, генерация выходных данных, их получение, конвертация и обработка.

Основной целью Языка является полное абстрагирование от особенностей распределенной вычислительной среды, в которой работает пользователь. Это делает описание задания полностью независимым от той вычислительной среды, которая доступна на данный момент. Такой подход можно кратко выразить так: EasyFlow – это высокоуровневый язык описания *абстрактных workflow* (AWF). Такой подход позволяет описывать саму решаемую задачу, а не способ ее исполнения на конкретной вычислительной архитектуре.

3.2.1. Основные элементы языка

В данном разделе описаны элементы языка EasyFlow с использованием нотации ANTLR.

Язык EasyFlow имеет следующие основные характеристики:

- скрипт представляет собой текстовый файл (с расширением .flow) в кодировке UTF-8;
- EasyFlow является языком, зависящим от регистра (т. е. myVariable и MyVariable представляют собой различные идентификаторы);
- типизация переменных – динамическая, слабая, с ограниченными возможностями по указанию типов;
- EasyFlow является платформо-независимым (с точки зрения различия операционных систем и вычислительных платформ);
- EasyFlow является интерпретируемым.

Переводы строк и пробелы. В EasyFlow переводы строк и пробелы являются разделителями различных конструкций языка. Для определения работы с переводами строк и пробелами в грамматику языка внесены правила, приведенные в листинге 3.1.

Листинг 3.1. Правила для обработки пробельных символов

```
wsn
: WS
| NEWLINE
;

wsnp
: wsn+
;

wsns
: wsn*
;

WS
: ' ' | '\t' ;

NEWLINE
: '\r'? '\n'
;
```

Пробельными символами считаются: перевод строки, пробел и знак табуляции. Переводы строк могут быть следующими: «\r\n» (Windows-style) и «\n» (Unix-style).

Для удобства определены правила wsns (любое количество пробельных символов) и wsnp (один и более пробельных символов), которые активно используются во всей грамматике языка.

Комментарии. В тексте скрипта могут встречаться однострочные комментарии после символов «//» и до конца строки, которые игнорируются парсером. Многострочные комментарии заключаются внутри символов «/*» и «*/».

Служебные символы: «{», «}», «[», «]», «(», «)», «=», «;» (точка с запятой), «:» (двоеточие), «.» (точка), «,» (запятая), «~» (тильда), «<-» (стрелка влево). Определение терминалов для этих символов приведено в листинге 3.2.

Листинг 3.2. Определение терминалов для служебных символов

```
LCUR
  : '{'
  ;

RCUR
  : '}'
  ;

LSQ
  : '['
  ;

RSQ
  : ']'
  ;

LPAREN
  : '('
  ;

RPAREN
  : ')'
  ;

ASSIGN
  : '='
  ;

SEMICOLON
  : ';'
  ;

COLON
  : ':'
  ;

DOT
  : '.'
  ;

COMMA
  : ','
  ;

TILDA
  : '~'
  ;

ARROW
  : '<-'
  ;
```

Идентификаторы языка EasyFlow определяются правилами, приведенными в листинге 3.3. Они могут начинаться на букву английского алфавита или знак подчеркивания и содержать любое количество букв английского алфавита, цифр и знаков подчеркивания.

Листинг 3.3. Определение идентификаторов

```
ID
  : ID_BASE
  ;

fragment ID_BASE
  : ('a'..'z'|'A'..'Z'|'_' ) ('a'..'z'|'A'..'Z'|'0'..'9'|'_' )*
```

Идентификаторы могут быть простыми (ID) и для указания констант (ID_CONST), которые префиксируются символом «@».

Строки в EasyFlow – это последовательности символов, заключенные в двойные кавычки вида «"». Внутри кавычек могут располагаться как обычные символы, так и escape-последовательности символов для ввода специальных символов, список которых приведен ниже:

- «\"» – символ «"» (двойные кавычки);
- «\b» – символ backspace;
- «\t» – символ табуляции;
- «\n» – символ перевода строки;
- «\f» – символ разрыва раздела;
- «\r» – символ возврата каретки;
- «\|» – символ «|»;
- «\'» – символ «'» (одинарные кавычки).

Строки могут содержать коды символов в шестнадцатеричном (вида «\uXXXX», где X – шестнадцатеричная цифра) и восьмеричном (вида «\XXX», где X – восьмеричная цифра) представлениях.

Листинг 3.4. Определение терминалов для строк

```
STRING
  : '"' ( (~('\\" | '"') | ESC_SEQ) * '"'
  ;

fragment HEX_DIGIT : ('0'..'9'|'a'..'f'|'A'..'F') ;

fragment ESC_SEQ
```

```

: '\\ ' ('b'|'t'|'n'|'f'|'r'|'\"'|'\''|'\\')
|  UNICODE_ESC
|  OCTAL_ESC
;

fragment OCTAL_ESC
: '\\ ' ('0'..'3') ('0'..'7') ('0'..'7')
|  '\\ ' ('0'..'7') ('0'..'7')
|  '\\ ' ('0'..'7')
;

fragment UNICODE_ESC
: '\\ ' 'u' HEX_DIGIT HEX_DIGIT HEX_DIGIT HEX_DIGIT
;

fragment RULETTER
: '\u0400'..' \u04FF'

```

Целые числа в EasyFlow представляются стандартным образом в десятичной системе счисления и могут иметь знак. Определение целых чисел приведено в листинге 3.5.

Листинг 3.5. Определение терминалов для целых чисел

```

INT : ('+' | '-')? DEC_DIGIT+
;

fragment DEC_DIGIT : ('0'..'9') ;

```

Дробные числа в EasyFlow представляются стандартным образом в десятичной системе счисления и могут иметь знак, целую часть, мантиссу и показатель степени (примеры: 0.14, 0.15e+14, .34e10, 5.e-14). Определение дробных чисел приведено в листинге 3.6.

Листинг 3.6. Определение терминалов для дробных чисел

```

DOUBLE
:  DEC_DIGIT+ '.' DEC_DIGIT* EXPONENT?
|  '.' DEC_DIGIT+ EXPONENT?
|  DEC_DIGIT+ EXPONENT
;

fragment DEC_DIGIT : ('0'..'9') ;

fragment EXPONENT : ('e'|'E') ('+'|'-')? ('0'..'9')+ ;

```

Булевы константы в EasyFlow представляют собой ключевые слова: true (истина) и false (ложь). Определение булевых констант приведено в листинге 3.7.

Листинг 3.7. Определение терминалов булевых констант

```

BOOL
  : ('true') | ('false')
  ;

```

Выражения в EasyFlow в текущей версии являются простыми и могут представлять собой:

строку;

список (см. ниже);

целое число;

дробное число;

булеву константу;

константный идентификатор (ID_CONST, см. выше);

выражение доступа (см. ниже).

Определение правила для выражения приведено в листинге 3.8.

Листинг 3.8. Определение правила для выражения

```

expression
  : STRING
  | INT
  | DOUBLE
  | BOOL
  | list
  | ID_CONST
  | varIdentifier
  ;

```

Выражения доступа. В EasyFlow для описания доступа к различным сложным объектам (именам пакетов, структурам) используются выражения доступа (varIdentifier), которые представляют собой идентификаторы, разделенные точками и, возможно, имеющие индексы, заключенные в квадратные скобки (примеры: object.Field, object.Array[13], ORCA.List[“Hello”].str). Определение правил для выражений доступа приведено в листинге 3.9.

Листинг 3.9. Определение правил для выражения доступа

```

varIdentifier
  : simpleVarIdentifier( wsns DOT wsns simpleVarIdentifier ) *
  ;

simpleVarIdentifier

```

```

: ID (wsns varIndexer)?
;

varIndexer
: LSQ wsns expression wsns RSQ
;

```

Списки в EasyFlow представляют собой последовательности выражений, разделенных запятыми и заключенных в квадратные скобки (примеры: [1, 2, 3], [a, [c, d]]). Списки могут быть вложенными и содержать значения различных типов. Определение правил для списков приведено в листинге 3.10.

Листинг 3.10. Определение правил для списков

```

list
: LSQ wsns expressionList? RSQ
;

expressionList
: expression wsns
(COMMA wsns expression wsns)*
;

```

Ключевые слова. EasyFlow для служебных целей резервирует следующие ключевые слова:

- flow** – для префиксирования атрибутов WF (см. ниже);
- require** – для определения требуемых входных файлов;
- step** – описание начала шага;
- ~step** — описание долгоживущего шага;
- sweep** – оператор перебора по значениям;
- runs** – дескриптор запускаемого пакета;
- after** – указатель явной передачи управления между шагами;
- on** – указатель передачи управления между шагами по событиям;
- true / false** – истина/ложь;
- pre, code, post** — определение секций пре- и постобработки.

Определение терминалов для ключевых слов приведено в листинге 3.11.

Листинг 3.11. Определение терминалов для ключевых слов

```

FLOW
: 'flow'
;

```

```

REQUIRE
  : 'require'
  ;

STEP
  : 'step'
  ;

SWEEP
  : 'sweep'
  ;

RUNS
  : 'runs'
  ;

AFTER
  : 'after'
  ;

ON
  : 'on'
  ;

BOOL
  : ('true') | ('false')
  ;

POST
  : 'post';

fragment CODE
  : 'code';

PRE
  : 'pre';

fragment END
  : 'end';

```

Предопределенные константы. EasyFlow может использовать предопределенные средой константы. Для этого используются идентификаторы с префиксом в виде символа «@».

Листинг 3.12. Определение терминала для предопределенных констант

```

ID_CONST
  : '@' ID_BASE
  ;

```

3.2.2. Построение workflow

Строение скрипта. Тело скрипта (правило program) представляет собой набор выражений верхнего уровня (правило topClause) – см. листинг 3.13.

Листинг 3.13. Правила program и topClause

```

wsn
  : WS
  | NEWLINE
  ;

wsnp
  : wsn+
  ;

wsns
  : wsn*
  ;

program
  : wsns ( topClause wsns )*
  ;

topClause
  : flowAttributeClause
  | stepClause
  | requireClause
  ;

```

Как видно из листинга, выражением верхнего уровня могут быть: атрибут WF (flowAttributeClause), определение шага (stepClause) и определение требования файла (requireClause). Об этих элементах см. ниже.

Атрибуты WF представляют собой задание настройки для исполнения WF и имеют синтаксис, приведенный в листинге 3.14. Пример задания настройки для WF приведен в листинге 3.15.

Листинг 3.14. Определение правила для атрибутов WF

```

flowAttributeClause
  : LSQ wsns FLOW wsns COLON wsns ID wsns ASSIGN wsns expression RSQ
  ;

```

Листинг 3.15. Примеры атрибутов WF

```

[flow:priority = @urgent]
[flow:author = "Paul McCartney"]
[flow:name = "Molecular geometry optimization"]
[flow:mode = @raw]

```

Тело скрипта может содержать любое количество атрибутов WF, заданных в форме, приведенной выше. Эти атрибуты задают либо некую информацию о WF

(например, данные об авторстве или названии WF), либо системные настройки, связанные с запуском.

Директива требования файлов (require). Чтобы указать, какие файлы требуются WF, в EasyFlow введена директива *require*. Она представляет собой ключевое слово «require» и следующий за ним список разделенных запятыми идентификаторов, обозначающих файлы. В дальнейшем эти идентификаторы будут ассоциированы с конкретными файлами, использующимися в запусках. В рамках одного скрипта директива требования файлов может появляться неограниченное число раз.

Определение правила для директивы *require* приведено в листинге 3.16. Примеры директив приведены в листинге 3.17.

Листинг 3.16. Определение директивы require

```
requireClause
: REQUIRE wsnp idList wsns SEMICOLON
;
idList
: ID (wsns COMMA wsns ID)*
;
```

Листинг 3.17. Примеры использования директивы require

```
require file1;
require jpg, png, gif;
```

Директива определения шага (step). Одной из основных функций EasyFlow является описание запусков вычислительных пакетов безотносительно к распределенной вычислительной архитектуре. Для этого используется директива определения шага *step*, задание правил для которой приведено в листинге 3.18.

Листинг 3.18. Определение правил для директивы step

```
stepClause
:
  attributes
  (TILDA)? STEP wsnp id
  RUNS wsnp compoundName
  (wsnp AFTER wsnp idList)? // Run conditions end
  wsns LPAREN
    wsns (APP wsns COLON wsns)? namedParametersList?
    ((EXEC wsns COLON wsns) namedParametersList)?
  RPAREN
  (wsns 'pre' WS+ CODE_BLOCK)?
  (wsns 'post' WS+ CODE_BLOCK)?
```

```

(wsns SEMICOLON)?
;

compoundName
: ID ( wsns DOT wsns ID ) *

attributes
: (attribute wsns)*
;

attribute
: LSQ wsns ID wsns ASSIGN wsns expression wsns RSQ
;

namedParametersList
: namedParameter wsns
(
(
( COMMA wsns namedParameter wsns )+
( COMMA wsns )? // comma at the end
)
| (COMMA wsns)?
)
;

namedParameter
: COLON? ID wsns ASSIGN wsns (SWEEP wsns)? expression
;

```

Директива *step* задает параметры и связи с другими шагами для выполняемого вычислительного пакета. Полный формат описания шага включает: атрибуты шага, имя шага, запускаемый пакет, условия получения управления (директива *after*), список входных параметров и файлов и описание секции постобработки. Пример полного определения шага приведен в листинге 3.19.

Листинг 3.19. Пример полного определения выполнения шага

```

require file1, file2;

step AnotherStep runs EmptyPackage ();

[priority = @high]
step StepName runs Package.Method after AnotherStep
(
  inFile1 = file1,
  inFile2 = file2,
  stringInput = "some string here",
  intInput = 100,
  doubleInput = 3.14,
  sweepParam = sweep [1, 2, 3],
  listParam = [AnotherStep.outs["out.txt"]]
)
post code ruby
i = 1
list = StepName.Result.outs

```

```
list.reverse
code end

~step LongRunningStep runs LRPackage
(
  inStream <- StepName.Result.outs["output.txt"]
);
```

3.2.3. Способы структурирования программы

В предыдущем разделе был описан синтаксис языка EasyFlow. Данный раздел посвящен особенностям создания WF с его помощью, а также технике выстраивания выполнения шагов относительно друг друга. В следующих пунктах подробно разбираются различные возможности описательной стороны EasyFlow.

Номенклатура названий вычислительных пакетов. Для определения шага требуется указать тот пакет, который предполагается запустить. Для этого используется нотация составного имени пакета, которая позволяет указывать не только имя запускаемого пакета, но и конкретный модуль, метод или режим, который требуется запустить. Для этого используется запись т.н. объекта запуска, которая представляет собой строку идентификаторов, разделенных точками. Ниже приведены примеры названий объектов запуска и их описания:

ORCA.DFT – пакет ORCA в режиме работы по методу DFT;

ScienceVis.ThreeDGraph.Mesh – запуск научного визуализатора, вызов его модуля ThreeDGraph и построение графика методом Mesh.

Такая нотация позволяет очень гибко описывать встраиваемые в комплекс пакеты и делать вызов отдельных часто используемых режимов и модулей более прозрачным.

Порядок передачи управления между шагами. Так как WF представляет собой ориентированный граф, в EasyFlow введены возможности по организации его структуры, а именно механизмы определения порядка выполнения шагов. Они делятся на два вида: зависимости по управлению и зависимости по данным.

Зависимости по управлению представляют собой явные указания на то, что шаг А должен начать свое исполнение после шага В с помощью директивы *after*, как это показано в листинге 3.20.

Листинг 3.20. Пример зависимостей по управлению

```
step A1 runs Pkg0 ();
step A2 runs Pkg1 ();
```

```
step B runs Pkg2 after A2 ();  
step C runs Pkg3 after A2 ();  
step D runs Pkg4 after B, C, A1 ();
```

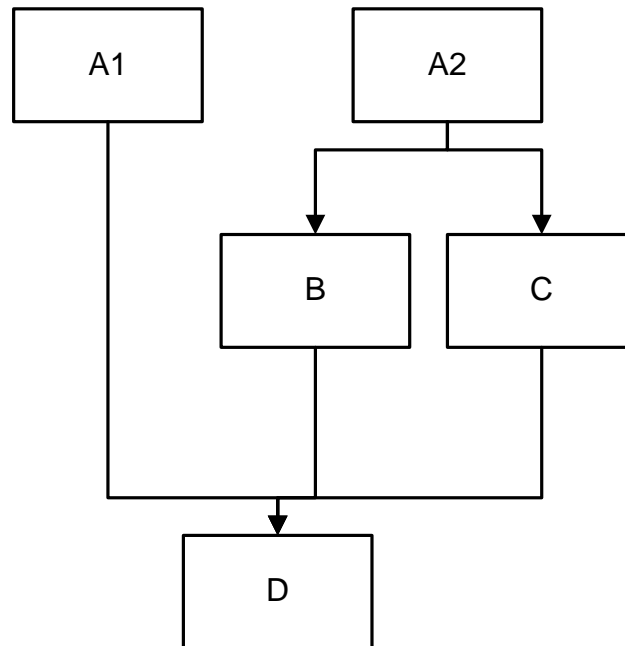


Рисунок 3.3 – Иллюстрация к листингу 3.20

Как видно из листинга, шаги A1 и A2 выполняются первыми, так как не зависят ни от каких других шагов. Шаги B и C выполняются после шага A1, а шаг D выполняется по завершении шагов B, C и A2. Эта запись может быть представлена в виде графа на рис. 3.3.

Зависимости по данным представляют собой неявные указания на зависимости между шагами, которые анализируются при интерпретации скрипта EasyFlow. Данный тип зависимости определяет порядок выполнения шагов: зависимые шаги могут быть запущены только после готовности всех требуемых для их работы данных. Такие зависимости могут присутствовать в описываемом WF одновременно с зависимостями по управлению, что позволяет очень гибко настраивать порядок выполнения шагов. Пример зависимостей по данным приведен в листинге 3.21, а соответствующий этому описанию граф – на рис. 3.4.

Листинг 3.21. Пример зависимостей по данным и управлению

```
step A1 runs Pkg0 ();
```

```
step A2 runs Pkg1 ();  
  
step B runs Pkg2  
(  
  inFile = A1.outs["out.txt"]  
);  
  
step C runs Pkg3 after A2 ();  
  
step D runs Pkg4 after C, A2 ();
```

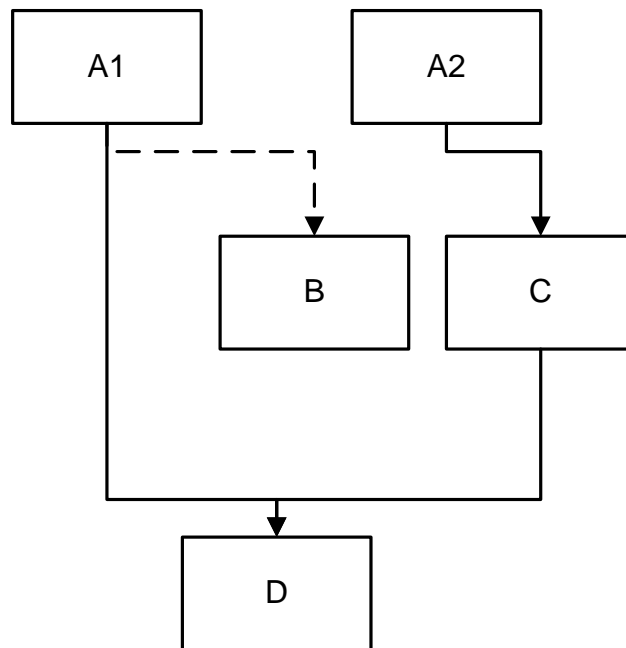


Рисунок 3.4 – Иллюстрация к листингу 3.21

Данный пример отличается от листинга 3.20 тем, что в описании скрипта шаг В зависит от шага А1 по данным, т.к. явно использует его результаты (A1.outs["out.txt"]).

Автоматический перебор параметров. Для удобства пользователя при проведении экспериментов в EasyFlow введена возможность автоматического варьирования параметров (parameter sweep). Такая задача часто возникает, когда необходимо запустить один и тот же вычислительный пакет, закрепив одни и варьируя другие параметры. Для этого в язык введена директива *sweep*, которая принимает список параметров для варьирования и из одного шага делает N шагов. N здесь – количество элементов в декартовом произведении списков варьирования для различных параметров.

Пример варьирования по двум параметрам (*iterations* и *precision*) представлен в листинге 3.22.

Листинг 3.22. Пример варьирования параметров

```

step SweepExample runs SomePackage
(
  width = 100,
  height = 200,
  precision = [0.1, 0.01]
  iterations = sweep [100, 200, 300]
);

```

В данном примере требуется запустить пакет `SomePackage`, варьируя два параметра (`iterations` и `precision`) и закрепив два других (`width` и `height`). При запуске будет создано шесть шагов с наборами пар (`precision`, `iterations`): (0.1, 100), (0.1, 200), (0.1, 300), (0.01, 100), (0.01, 200), (0.01, 300).

3.3. Специализированные элементы языка для платформы МИТП-К

В данном разделе перечислены элементы, обладающие особой значимостью в рамках задач, соответствующих платформе МИТП-К

3.3.1. Явное задание ресурса

В ситуациях, когда пользователь достаточно опытен и необходимо выбрать фиксированный ресурс, используемый платформой, для расчетов конкретного пакета, существует возможность явного задания ресурсов через атрибут «`resource`». Данный атрибут может быть использован двумя способами.

- 1) Если известна информация об идентификаторе группы ресурсов (например, кластер или блейд-серверы), то пользователь может определить у очередного шага WF желаемую группу ресурсов (листинг 3.23).

Листинг 3.23. Использование атрибута «`resource`» для группы ресурсов

```

[Resource = "b4"]
step ResourceUseExample runs SomePackage
(
  mainParament = "mainValue";
);

```

- 2) Если известна информация об идентификаторе группы ресурсов и самом ресурсе (например, конкретный узел кластера или блейд-сервер), то пользователь может определить у очередного шага WF желаемый ресурс.

Листинг 3.24. Использование атрибута «`resource`» для конкретного ресурса

```

[Resource = "b4.b4-130"]
step ResourceUseExample runs SomePackage
(

```

```
mainParament = "mainValue";
);
```

3.3.2. *Определение приоритета workflow и шагов*

В зависимости от потребностей пользователя в EasyFlow возможно задать приоритет выполнения целого WF или отдельных его шагов, используя атрибут «priority». Данный атрибут позволяет внутреннему планировщику платформы распределить выполнение заданий с учетом желаемых приоритетов пользователей системы. Атрибут принимает следующие значения: «low»– низкая значимость, «normal» – нормальная значимость, «high»– высокая значимость.

По умолчанию атрибут принимает значение «normal». Примеры использования атрибута представлены в листингах 3.25, 3.26. В первом случае шаг «PriorityExmple» выполняется с приоритетом «high», а «SimpleExmple» с нормальным. Во втором примере весь WF выполняется с приоритетом «low».

Листинг 3.25. Пример использования атрибута «priority» для шага WF

```
[priority = "high"]
step PriorityExmple runs SomePackage
(
    mainParament = "mainValue";
);
step SimpleExmple runs SomePackage
(
    mainParament = "mainValue";
);
```

Листинг 3.26. Пример использования атрибута «priority» для WF

```
[flow:priority = "low"]
step LowPriorityExmple1 runs SomePackage
(
    mainParament = "mainValue";
);
step LowPriorityExmple2 runs SomePackage
(
    mainParament = "mainValue";
);
```

4. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Для работы с МИТП не требуется специальных видов входных данных. В ходе работы входными данными могут являться любые входные файлы, соответствующие по формату запускаемым прикладным сервисам (в текстовом, цифровом, графическом виде), а также данные, вводимые пользователем с клавиатуры по запросу сервиса. В случае

несоответствия данных условиям их использования будет выдано системное сообщение. Для их приведения к общему формату используется технология описания на основе языка EasyPackage.

В качестве выходных данных МИТП предоставляет результаты расчетов, загруженные с удаленного хранилища CLAVIRE/Storage RU.СНАБ.80066-06 01 25 (в форме текстового, графического или цифрового файла, размещаемого в директории, указываемой пользователем через соответствующее диалоговое окно). Формат файла соответствует тому сервису, посредством которого был произведен расчет. Для обеспечения единого формата в целях унификации процесса передачи данных между сервисами используется технология описания на основе языка EasyPackage.

5. СООБЩЕНИЯ

В данном разделе определяются основные сообщения программисту, информирующие об ошибке, возникшей в процессе работы с МИТП. Сообщения программисту могут выдаваться двумя способами:

1. *Посредством пользовательского интерфейса.* В этом случае программист получает сообщение в виде всплывающего окна, информирующего о возникновении исключительной ситуации (рис. 5.1).

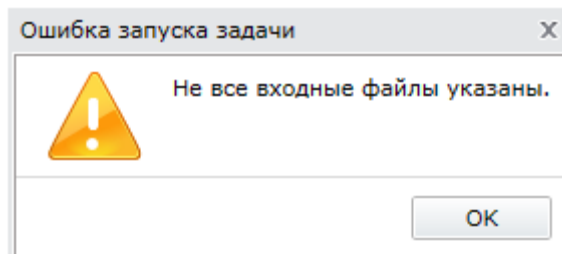


Рисунок 5.1 – Сообщение об исключительной ситуации (пример)

2. *В журнал событий МИТП.* В журнале событий приводится развернутая информация, необходимая для устранения ошибки. Доступ к журналу сообщений МИТП и ее компонентов программист может получить с использованием возможностей компонента средств мониторинга и настройки основных компонентов платформы CLAVIRE/AdminTool. Действуя в соответствии с документом «Методика использования компонента» мониторинга и настройки основных компонентов платформы CLAVIRE/AdminTool» RU.СНАБ.80066-06 ИЗ

64, программист может получить доступ к просмотру внутренних журналов платформы и ее компонентов (рис. 5.2).

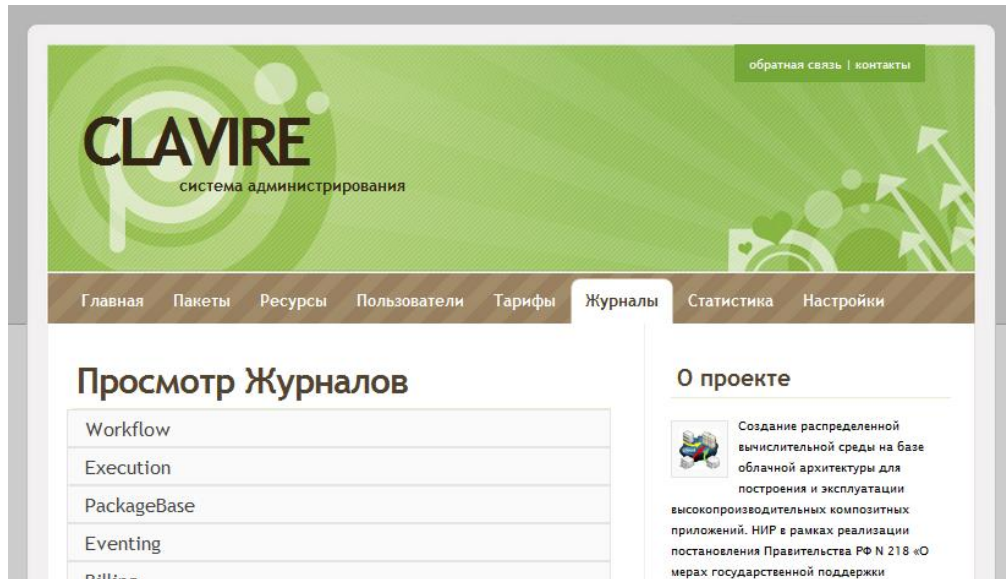


Рисунок 5.2 – Журналы работы компонентов платформы

Выбрав компонент, можно увидеть окно просмотра журнала этого компонента (рис. 5.3). Особое внимание стоит обращать на строчки, выделенные красным, – это сообщения об ошибках в работе комплекса.

```

2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.DeclarativeInterpreter Internal event enqueued @block_started(WF#02fc0e8c-0c23-459a-87d1-4778764fde2.1 +nul
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.GlobalDataScope Shared variable 'Data_base.Result' in global data scope
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.NodeBase Node#1 state changed state_started -> state_pre_section
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.NodeBase Step:Data_base#1(state_pre_section) Pre section is NULL ignoring
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.NodeBase Node#1 state changed state_pre_section -> state_run_start
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.NodeBase RunMode was set to Meta
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.NodeBase [Ignoring temporary] Error while checking package run signature.
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.NodeBase Ignoring parameter list
2011-12-27 19:08:37.4785 Easis.Wfs.Interpreting.NodeBase [Ignoring temporary] Error while forming outputs
2011-12-27 19:08:37.4785 Easis.Wfs.FlowSystemService.DryExecutionStepStarter Defining step using Execution.
2011-12-27 19:08:37.4785 Easis.Wfs.FlowSystemService.DryExecutionStepStarter Staring step using Execution.
2011-12-27 19:08:39.5126 Easis.Wfs.FlowSystemService.ExecutionEventConverter Found event from Execution for StepId 21909. Trying to find accordance in id di
2011-12-27 19:08:40.5584 Easis.Wfs.FlowSystemService.ExecutionEventConverter Found event from Execution for StepId 21910. Trying to find accordance in id di
2011-12-27 19:08:41.5252 Easis.Wfs.FlowSystemService.ExecutionEventConverter SequenceGetInfo returns valid object
2011-12-27 19:08:41.5252 Easis.Wfs.FlowSystemService.ExecutionEventConverter { "t": "Task", "ExtensionData": {}, "ExecParams": {}, "InputFiles": {
2011-12-27 19:08:41.5252 Easis.Wfs.FlowSystemService.ExecutionEventConverter RunInfo has been successfully fetched
2011-12-27 19:08:41.5252 Easis.Wfs.FlowSystemService.ExecutionEventConverter Event Eventing.EventReport converted with Easis.Wfs.FlowSystemService.Executio
2011-12-27 19:08:41.5252 Easis.Wfs.FlowSystemService.JobExecutor Got PushEvent command for WF#02fc0e8c-0c23-459a-87d1-4778764fde2. @run_started(WF#02fc0e8c
2011-12-27 19:08:41.5252 Easis.Wfs.Interpreting.DeclarativeInterpreter External event enqueued @run_started(WF#02fc0e8c-0c23-459a-87d1-4778764fde2.0 +Easis
2011-12-27 19:08:41.5252 Easis.Wfs.Interpreting.NodeBase Node convert action action_set_run_info was called with arg Easis.Wfs.Interpreting.StepRunInfo
2011-12-27 19:08:41.5252 Easis.Wfs.Interpreting.NodeBase Node#0 state changed state_run_start -> state_wait_results
2011-12-27 19:08:42.5710 Easis.Wfs.FlowSystemService.ExecutionEventConverter SequenceGetInfo returns valid object
2011-12-27 19:08:42.5710 Easis.Wfs.FlowSystemService.ExecutionEventConverter { "t": "Task", "ExtensionData": {}, "ExecParams": {}, "InputFiles": [
2011-12-27 19:08:42.5710 Easis.Wfs.FlowSystemService.ExecutionEventConverter RunInfo has been successfully fetched
2011-12-27 19:08:42.5710 Easis.Wfs.FlowSystemService.ExecutionEventConverter Event Eventing.EventReport converted with Easis.Wfs.FlowSystemService.Executio
2011-12-27 19:08:42.5710 Easis.Wfs.FlowSystemService.JobExecutor Got PushEvent command for WF#02fc0e8c-0c23-459a-87d1-4778764fde2. @run_started(WF#02fc0e8c
2011-12-27 19:08:42.5710 Easis.Wfs.Interpreting.DeclarativeInterpreter External event enqueued @run_started(WF#02fc0e8c-0c23-459a-87d1-4778764fde2.1 +Easis
2011-12-27 19:08:42.5710 Easis.Wfs.Interpreting.NodeBase Node Data_base action action_set_run_info was called with arg Easis.Wfs.Interpreting.StepRunInfo
2011-12-27 19:08:42.5710 Easis.Wfs.Interpreting.NodeBase Node#1 state changed state_run_start -> state_wait_results
2011-12-27 19:08:44.7516 Easis.Wfs.FlowSystemService.ExecutionEventConverter Found event from Execution for StepId 21910. Trying to find accordance in id di
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter SequenceGetInfo returns valid object
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter { "t": "Task", "ExtensionData": {}, "ExecParams": {}, "InputFiles": [
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter Found output file name:'test_mol.pdbqt' slot:'none' storageid:G80X2C1050X747Y
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter Found output file name:'test_2.pdbqt' slot:'none' storageid:88BX9YI2LN4W0468K
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter Found output file name:'test_3.pdbqt' slot:'none' storageid:CGV9JNSJSH7F007658
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter Found output file name:'test_4.pdbqt' slot:'none' storageid:CGV9JNSJSH7F007658
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter Storage service returned 4 ids for 4 data entries
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter Registered new files in storage
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter StepResult has been successfully fetched
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.ExecutionEventConverter Event Eventing.EventReport converted with Easis.Wfs.FlowSystemService.Executio
2011-12-27 19:08:46.8041 Easis.Wfs.FlowSystemService.JobExecutor Got PushEvent command for WF#02fc0e8c-0c23-459a-87d1-4778764fde2. @run_finished(WF#02fc0e8c
2011-12-27 19:08:46.8041 Easis.Wfs.Interpreting.DeclarativeInterpreter External event enqueued @run_finished(WF#02fc0e8c-0c23-459a-87d1-4778764fde2.1 +Easi
2011-12-27 19:08:46.8041 Easis.Wfs.Interpreting.NodeBase Node Data_base action action_set_run_results was called with arg Easis.Wfs.Interpreting.StepRunResu
2011-12-27 19:08:46.8041 Easis.Wfs.Interpreting.NodeBase Node#1 state changed state_wait_results -> state_run_finish
2011-12-27 19:08:46.8041 Easis.Wfs.Interpreting.NodeBase Exception while converting output param 'db1' with value 'null'. Ignoring.
2011-12-27 19:08:46.8041 Easis.Wfs.Interpreting.GlobalDataScope Shared variable 'Data_base.Result' in global data scope
2011-12-27 19:08:46.8041 Easis.Wfs.Interpreting.NodeBase Node#1 state changed state_run_finish -> state_post_section
2011-12-27 19:08:46.8041 Easis.Wfs.Interpreting.NodeBase Step:Data_base#1(state_post_section) Post section is NULL ignoring
2011-12-27 19:08:46.8197 Easis.Wfs.Interpreting.NodeBase Node#1 state changed state_post_section -> state_finished
2011-12-27 19:08:46.8197 Easis.Wfs.Interpreting.NodeBase Node#1 generated event BLOCK_FINISHED
2011-12-27 19:08:46.8197 Easis.Wfs.Interpreting.DeclarativeInterpreter Internal event enqueued @block_finished(WF#02fc0e8c-0c23-459a-87d1-4778764fde2.1 +nul

```

<</logs

Рисунок 5.3 – Журнал системных сообщений МИТП

Перечень наиболее важных сообщений, выдаваемых компонентами МИТП, приведен в табл. 5.1.

Таблица 5.1

Основные сообщения программисту и оператору

Сообщение	Типовые действия по выявлению и устранению ошибки
Ошибки входа в систему	
Введенная пара логин / пароль неверна	Проверить введенные логин и пароль на корректность, проверить состояние клавиши CapsLock
Ваша учетная запись отключена	Обратиться к администратору комплекса по поводу отключения учетной записи
Система находится на профилактике, попробуйте позднее	Повторить попытку входа через 3–5 минут
Сервер возвратил ошибку: NotFound	Обратиться в службу поддержки комплекса за устранением ошибки
Нет соединения с сетью	Проверить наличие соединения с Интернетом и попробовать снова выполнить операцию
Истекло время ожидания ответа от сервера	Проверить наличие соединения с Интернетом и попробовать снова выполнить операцию. В случае повторения обратиться в службу поддержки комплекса
Соединение принудительно прервано сервером	Попробовать снова выполнить операцию. В случае повторения обратиться в службу поддержки комплекса
Ошибки работы с проектами и задачами	
Не удалось сохранить проект	Проверить наличие соединения с Интернетом и попробовать снова выполнить операцию. В случае повторения обратиться в службу поддержки комплекса
Не все входные данные задачи указаны	Указать все требуемые для задачи файлы и повторить операции
Невозможно запустить задачу: недостаточно свободных ресурсов	Подождать 3–5 минут и попробовать выполнить операцию снова. В случае повторения обратиться в службу поддержки комплекса
Системные ошибки	
Нехватка места на жестком диске	Проверить корректность работы программных компонентов, установленных на указанной ЭВМ, на предмет бесконтрольного заполнения свободного пространства на жестком диске, осуществить переконфигурацию программных модулей, принять меры к увеличению свободного пространства на жестком диске
Нехватка оперативной памяти	Проверить корректность работы программных компонентов, установленных на указанной ЭВМ, на предмет

	бесконтрольного заполнения пространства оперативной памяти, осуществить переконфигурацию программных модулей, принять меры к увеличению объема оперативной памяти
<i>Ошибки при разработке программ для корпоративной среды</i>	
Ошибка явного задания ресурса	Проверить корректность указания ресурса в скрипте EasyFlow
Ошибка определения приоритетов в WF	Проверить корректность задания приоритетов WF и отдельных его блоков. Проверить наличие прав на задание повышенного приоритета.
Ошибка базы шаблонов	Проверить корректность содержимого базы шаблонов
Ошибка работы с виртуальными машинами	Проверить корректность настройки используемой инфраструктуры виртуальных ЭВМ.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

WF	Поток заданий, workflow
AWF	Абстрактный WF
CWF	Конкретный WF
ННС	Национальная нанотехнологическая сеть
MWF	Мета-WF

