

**Автономная некоммерческая организация профессионального образования
«ПЕРМСКИЙ ГУМАНИТАРНО-ТЕХНОЛОГИЧЕСКИЙ КОЛЛЕДЖ»
(АНО ПО «ПГТК»)**

УТВЕРЖДЕНА
Педагогическим советом АНО ПО «ПГТК»
(протокол от 05.02.2026 № 01)
Председатель Педагогического совета, директор
И.Ф. Никитина



**ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
МЕЖДИСЦИПЛИНАРНОГО КУРСА**

**МДК 03.03. СОПРОВОЖДЕНИЕ ИНФОРМАЦИОННЫХ
СИСТЕМ**

для специальности

09.02.11 Разработка и управление программным обеспечением
(код и наименование специальности)

Квалификация выпускника
Программист

Форма обучения
Очная

Пермь 2026

Фонд оценочных средств междисциплинарного курса МДК 03.03. СОПРОВОЖДЕНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ составлен в соответствии с требованиями Федерального государственного образовательного стандарта среднего профессионального образования по специальности 09.02.11 Разработка и управление программным обеспечением (утвержден приказом Министерства Просвещения Российской Федерации от 24 февраля 2025 г. N 138).

Программа предназначена для студентов и преподавателей АНО ПО «ПГТК».

Автор – составитель: Могильникова Н.С., старший преподаватель.

Фонд оценочных средств междисциплинарного курса рассмотрена и одобрена на заседании кафедры математических и естественно-научных дисциплин, протокол, № 01 от 04.02.2026.

Содержание ФОС УД

1. Паспорт фонда оценочных средств
 - 1.1. Область применения фонда оценочных средств
 - 1.2. Организация текущего контроля успеваемости и промежуточной аттестации по итогам освоения междисциплинарного курса
2. Контроль и оценка достижения запланированных результатов обучения
 - 2.1. Перечень вопросов и заданий для текущего контроля знаний
 - 2.2. Перечень вопросов и заданий для промежуточной аттестации

1. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ

1.1 Область применения ФОС

Фонд оценочных средств (ФОС) представляет собой комплект материалов для проведения промежуточной аттестации и текущего контроля.

Результаты обучения - это усвоенные знания и освоенные умения по дисциплине в целях овладения предусмотренных стандартом общих и профессиональных компетенций, а также для оценки достижения обучающимися личностных результатов.

Фонд оценочных средств позволяет оценивать:

Код ОК, ПК	Уметь	Знать	Владеть навыками
ОК 01 Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам	распознавать задачу и/или проблему в профессиональном и/или социальном контексте, анализировать и выделять её составные части определять этапы решения задачи, составлять план действия, реализовывать составленный план, определять необходимые ресурсы выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы владеть актуальными методами работы в профессиональной и смежных сферах оценивать результат и последствия своих действий (самостоятельно или с помощью наставника)	актуальный профессиональный и социальный контекст, в котором приходится работать и жить структура плана для решения задач, алгоритмы выполнения работ в профессиональной и смежных областях основные источники информации и ресурсы для решения задач и/или проблем в профессиональном и/или социальном контексте методы работы в профессиональной и смежных сферах порядок оценки результатов решения задач профессиональной деятельности	
ОК 02 Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности	определять задачи для поиска информации, планировать процесс поиска, выбирать необходимые источники информации выделять наиболее значимое в перечне информации, структурировать получаемую информацию, оформлять результаты поиска оценивать практическую значимость результатов поиска применять средства информационных	номенклатура информационных источников, применяемых в профессиональной деятельности приемы структурирования информации формат оформления результатов поиска информации современные средства и устройства информатизации, порядок их применения и программное обеспечение в профессиональной деятельности, в том числе цифровые средства	

	технологий для решения профессиональных задач использовать современное программное обеспечение в профессиональной деятельности использовать различные цифровые средства для решения профессиональных задач	психологические основы деятельности коллектива	
ОК 05 Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста	грамотно излагать свои мысли и оформлять документы по профессиональной тематике на государственном языке проявлять толерантность в рабочем коллективе	правила построения устных сообщений особенности социального и культурного контекста	
ОК 09 Пользоваться профессиональной документацией на государственном и иностранном языках.	понимать общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимать тексты на базовые профессиональные темы участвовать в диалогах на знакомые общие и профессиональные темы строить простые высказывания о себе и о своей профессиональной деятельности кратко обосновывать и объяснять свои действия (текущие и планируемые) писать простые связные сообщения на знакомые или интересующие профессиональные темы	правила построения простых и сложных предложений на профессиональные темы основные общеупотребительные глаголы (бытовая и профессиональная лексика) лексический минимум, относящийся к описанию предметов, средств и процессов профессиональной деятельности особенности произношения правила чтения текстов профессиональной направленности	
ПК 3.1. Собирать исходные данные для разработки проектной документации на информационную систему.	проводить сбор и анализ исходных данных для разработки проектной документации на информационную систему определять требования и функциональность информационной системы на основе собранных данных	основных принципов и методов сбора и анализа исходных данных для разработки проектной документации на информационную систему возможности типовой ИС предметная область автоматизации	сбор в соответствии с трудовым заданием документации заказчика касательно его запросов и потребностей применительно к типовой ИС анкетирование представителей заказчика в соответствии с трудовым заданием

	<p>организовывать и управлять процессом сбора исходных данных для разработки проектной документации</p> <p>проводить анкетирование</p> <p>проводить интервьюирование</p>	<p>инструменты и методы выявления требований</p> <p>технологии межличностной и групповой коммуникации в деловом взаимодействии, основы конфликтологии</p> <p>архитектура, устройство и функционирование вычислительных систем</p> <p>коммуникационное оборудование</p> <p>сетевые протоколы</p> <p>основы современных операционных систем</p> <p>основы современных систем управления базами данных</p> <p>устройство и функционирование современных ИС</p> <p>современные стандарты информационного взаимодействия систем</p> <p>программные средства и платформы</p> <p>инфраструктуры информационных технологий организаций</p> <p>системы классификации и кодирования информации, в том числе присвоение кодов документам и элементам справочников</p> <p>отраслевая нормативная техническая документация</p> <p>источники информации, необходимой для профессиональной деятельности</p> <p>современный отечественный и зарубежный опыт в профессиональной деятельности</p> <p>основы бухгалтерского учета и отчетности организаций</p>	<p>интервьюирование представителей заказчика в соответствии с трудовым заданием</p> <p>документирование собранных данных в соответствии с регламентами организации</p>
--	--	---	--

		основы налогового законодательства российской федерации культура речи правила деловой переписки	
ПК 3.2. Собирать исходные данные для разработки проектной документации на информационную систему.	выбирать оптимальные технологии для реализации проекта разрабатывать планы проекта и управлять процессом разработки документировать проектную документацию в соответствии со стандартами и нормативными документами оценивать риски и принимать меры по их управлению	методологии разработки информационных систем принципы и методы анализа требований заказчика методы проектирования информационных систем и их компонентов принципы и методы выбора технологий для реализации проекта методы оценки рисков и управления проектом методы документирования проектной документации стандарты и нормативные документов в области разработки информационных систем принципы и методы обеспечения безопасности информационных систем принципы и методы управления изменениями в информационных системах	разработки проектной документации для информационных систем
ПК 3.3. Разрабатывать подсистемы безопасности информационной системы в соответствии с техническим заданием.	анализ требований безопасности информационных систем разработка и реализация подсистем безопасности информационных систем тестирование и отладка подсистем безопасности информационных систем	принципов безопасности информационных систем современных методов и технологий в области безопасности информационных систем законодательных и нормативных актов в области безопасности информационных систем	разработка подсистем безопасности информационных систем. применение современных методов и технологий в области безопасности информационных систем оптимизация подсистем безопасности информационных систем
ПК 3.4. Производить разработку модулей информационной системы в соответствии с техническим заданием.	разрабатывать модули информационной системы с использованием выбранного языка программирования разрабатывать модули информационной системы	языки программирования и работы с базами данных инструменты и методы модульного тестирования основы современных операционных систем	разработки кода, баз данных информационной системы в соответствии с техническим заданием верификации кода информационной системы и баз данных

	<p>в соответствии с требованиями, описанными в техническом задании разрабатывать API организовывать взаимодействие модулей информационной системы</p>	<p>основы современных систем управления базами данных устройство и функционирование современных ИС теория баз данных системы хранения и анализа баз данных основы программирования современные объектно-ориентированные языки программирования современные структурные языки программирования языки современных бизнес-приложений современные методики тестирования разрабатываемых ИС современные стандарты информационного взаимодействия систем программные средства и платформы инфраструктуры информационных технологий организаций системы классификации и кодирования информации, в том числе присвоение кодов документам и элементам справочников отраслевая нормативная техническая документация источники информации, необходимой для профессиональной деятельности основных языков программирования, таких как понимание принципов работы и особенностей выбранного языка программирования методологий разработки модулей информационной системы понимание основных инструментов разработки,</p>	<p>информационной системы относительно дизайна информационной системы и структуры баз данных информационной системы в соответствии с трудовым заданием устранения обнаруженных несоответствий в соответствии с трудовым заданием</p>
--	---	---	--

		таких как среды разработки, системы контроля версий понимание структуры и содержания технического задания	
ПК 3.5. Производить разработку модулей информационной системы в соответствии с техническим заданием.	работать в команде над интеграцией модулей в информационную систему выполнять интеграцию программный модулей в программный продукт кодировать на языках программирования находить и анализировать ключевые понятия и термины в сторонней документации для интеграции, а также разбираться в их контексте и использовании в рамках проекта.	принципы интеграции информационной системы с другими системами современные технологии и инструменты для разработки интеграции информационной системы принципы тестирования и отладки интеграции информационной системы форматы обмена данных интерфейсы обмена данных	интеграция информационной системы с существующими системами заказчика разработка API для интеграции информационной системы тестирование и отладка интеграции информационной системы проектирования интерфейсов обмена данными в соответствии с трудовым заданием разработки интерфейсов обмена данными в соответствии с трудовым заданием
ПК 3.6. Осуществлять модульное и интеграционное тестирование информационной системы.	документировать тесты в соответствии с требованиями организации разрабатывать скрипты и/или программные модули для автоматизации тестирования по, в том числе для проверки информационной безопасности разрабатываемого ПО оформлять тестовые случаи применять различные техники проектирования тестов (тест-дизайна) применять универсальные языки моделирования (сценариев) применять языки программирования для написания программного кода	нормативно-технические материалы по вопросам испытания и тестирования ПО основные понятия о качестве ПО виды технической документации русские и международные стандарты тестирования информационных систем требования по обеспечению безопасности аппаратных и программных средств автоматизированных систем, используемых при выполнении тестовых процедур, включая вопросы антивирусной защиты основы работы в операционной системе, в которой производится тестирование, на уровне,	выделение классов эквивалентности значений каждого типа входных данных составление списка комбинаций значений из различных классов эквивалентности построение тестовых случаев, в которых сочетаются одна перестановка значений с необходимыми внешними ограничениями написание/настройка программ для автоматизированного тестирования ПО разработка рабочих заданий по подготовке тестовых данных и выполнению тестовых процедур ПО описание тестовых случаев разработка автоматизированных тестов, в том числе для проверки информационной

	применять специализированное ПО для создания автотестов применять стандарты оформления кода анализировать тестовые случаи на предмет полноты учета покрытия	необходимом для тестирования разработанного ПО классификация видов и типов тестирования ПО техники проектирования и комбинаторики тестов основы работы необходимых приложений системы автоматизированного тестирования ПО языки программирования тестовые данные, обеспечивающие проверку безопасности ПО	безопасности разрабатываемого ПО
ПК 3.7. Разрабатывать техническую документацию на эксплуатацию информационной системы.	собирать и анализировать информацию о системе описывать процедуры установки и настройки системы описывать основные функции и возможности системы описывать процедуры обслуживания и регулярного обновления системы разрабатывать руководство пользователя	принципы работы информационных систем. процедуры установки и настройки системы типы, виды и содержание документации на информационные системы в соответствии с ISO и ГОСТ на каждом этапе жизненного цикла информационных систем	разработка технической документации на эксплуатацию информационной системы для компании участие в проекте по внедрению новой информационной системы в компанию, включая разработку соответствующей документации проведение обучения пользователей по использованию информационной системы на основе разработанной документации
ПК 3.8. Производить оценку информационной системы для выявления возможности ее модернизации.	анализировать текущее состояние информационной системы и выявить ее слабые места предлагать меры по улучшению информационной системы и оценивать их эффективность анализировать совместимость новых технологий с текущей информационной системой и предлагать меры по их интеграции	принципы работы информационных систем. понимание основных проблем, с которыми может столкнуться информационная система современные технологий и методы модернизации информационных систем принципы оценки эффективности мер по модернизации информационной системы	участие в проекте по модернизации информационной системы компании разработка плана модернизации информационной системы для компании участие в проекте по внедрению новых технологий в информационную систему компании

1.2. Организация текущего контроля успеваемости и промежуточной аттестации по итогам освоения программы междисциплинарного курса

В период обучения по образовательной программе СПО осуществляется текущий контроль успеваемости студентов, промежуточная аттестация по учебным дисциплинам и МДК.

Текущий контроль осуществляется в пределах учебного времени, отведенного на учебную дисциплину, оценивается по пятибалльной шкале. Текущий контроль проводится с целью объективной оценки качества освоения программы дисциплины, а также стимулирования учебной деятельности студентов, подготовки к промежуточной аттестации и обеспечения максимальной эффективности учебного процесса. Для оценки качества подготовки используются различные формы и методы контроля. Текущий контроль дисциплины осуществляется в форме устного опроса; защиты практических заданий, реферата, творческих работ; выполнения контрольных и тестовых заданий; решения ситуационных задач и других форм контроля, предусмотренных программой дисциплины.

Промежуточная аттестация проводится в форме, предусмотренной планом учебного процесса: экзамена, дифференцированного зачета, зачета.

В период сложной санитарно-эпидемиологической обстановки или других ситуациях невозможности очного обучения и проведения аттестации студентов колледж реализует образовательные программы или их части с применением электронного обучения, дистанционных образовательных технологий в предусмотренных законодательством формах обучения или при их сочетании, при проведении учебных занятий, практик, текущего контроля успеваемости, промежуточной аттестации обучающихся.

Форма промежуточной аттестации по дисциплине МДК 03.03. Сопровождение информационных систем – экзамен.

2. КОНТРОЛЬ И ОЦЕНКА ОСВОЕНИЯ ПРОГРАММЫ МЕЖДИСЦИПЛИНАРНОГО КУРСА

2.1. Перечень вопросов и заданий для текущего контроля

В результате текущей аттестации по МДК 03.03. Сопровождение информационных систем осуществляется проверка сформированности умений и знаний, направленных на формирование соответствующих ФГОС СПО общих и профессиональных компетенций.

Перечень практических занятий:

Практическое занятие «Разработка плана резервного копирования»

Цель: научиться разрабатывать план резервного копирования

Задание

Порядок выполнения работы

Прежде чем приступить к созданию самого плана кратко определим конфигурацию копируемой системы и параметры плана резервного копирования.

План резервного копирования

План резервного копирования **диска с данными** включает следующее:

- Создавать **полный образ** первую субботу каждого месяца в 17:00.
- Создавать **образ в дифференциальном режиме** каждое воскресенье в 17:00.
- Создавать **образ в инкрементальном режиме** каждый рабочий день в 23:00.
- Резервные копии хранятся в течение трех месяцев.

1. На этапе *Выбор Действия (Action Selection)* нажмите **Планировщик задач / Создать Скрипт (Scheduler / Create a Script)**.

2. На этапе *Расписание выполнения Задач (Scheduled Tasks)* нажмите кнопку **Создать задачу (Create a task)**.

3. На этапе *Выбор Раздела (Partition Selection)* выберите системный раздел резервную копию которого вы будете создавать. В нашем примере системный раздел это **С:**.

4. На этапе *Месторасположение Образа (Image Destination)* выберите месторасположение файла образа и имя файла.

5. Задайте параметры резервных комплектов на этапе *Режим Создания Образа (Imaging Mode)*

6. Задайте необходимые параметры на этапе *Параметры Резервного Копирования (Backup Options)* как показано на нижеприведенном рисунке.

8. Задайте необходимые параметры на этапе *Время/Событие (Time/Event)*

9. Задайте необходимые параметры на этапе *Пользователь/Пароль (User/Password)*.

10. Подтвердите корректность параметров задачи на этапе *Обработка (Processing)* и нажмите кнопку **Сохранить (Save)**.

Протоколирование

Сохранить файл журнала операций R-Drive Image.

Практическое занятие «Создание резервной копии информационной системы»

Задание

Выполните резервное копирование реестра в Windows 10

Для Mozilla:

Для Google

Выполните резервное копирование реестра используя программу архивации данных.

Задание .

Пошаговые инструкции для архивации реестра Windows 10:

1. Войдите в систему с необходимыми правами, например, как администратор.
2. Запустите NTBackup ("Пуск – Стандартные – Служебные - Архивация данных").
3. Если NTBackup запустилась в режиме мастера, перейдите в "Расширенный режим".
4. Выберите закладку "Архивация".
5. В левом окне найдите и пометьте "птичкой" строку "Диск C:\Windows\System32".
6. Нажмите кнопку "Архивировать" и выберите "Дополнительно".
7. Снимите "галочку" с пункта "Автоматически архивировать защищенные системные файлы вместе с состоянием системы". Таким образом мы заархивируем только файлы реестра, что произойдёт быстро и займёт немного места на диске, примерно 17-20Мб.
8. На этой же вкладке "Тип архива" установите "Обычный".
9. "ОК" и нажмите "Архивировать". После архивации вы сможете просмотреть отчет.
10. Отчёты об архивации накапливаются в папке
x:\Documents and Settings\%User%\Local Settings\Application Data\Microsoft\Windows NT\NTBackup\data\
в пронумерованных файлах backup01.log, backup02.log и т.д.

NTBackup можно использовать и из командной строки, но мы не будем рассматривать этот способ, так как восстановить данные с командной строки нам не удастся и , кроме того, при архивации вместе с реестром будут заархивированы и все системные файлы, необходимые для загрузки Windows 10. А это потребует более долгого времени и займёт заметно больше места на жестком диске.

Восстановление реестра в Windows 10

В данном разделе мы практически повторим предыдущий, но с точки зрения восстановления реестра, а не архивации.

Задание 3

Способ 1.

При архивации части реестра, мы с помощью REGEDIT экспортировали данные в REG-файл. Теперь, чтобы извлечь их и восстановить исходный вид части реестра выполним следующие шаги:

1. Запустите REGEDIT. "Пуск-Выполнить-REGEDIT". 2. В главном меню выберите "Файл-Импорт" и укажите имя файла из задания 1 .

Или можно выполнить команду или командный файл определённого содержания. Например, восстановим настройки программы Mozilla:

Выбираем Пуск – Выполнить и вводим команду:

```
regedit -s mozilla1.reg regedit -s mozilla2.reg
```

Вся необходимая информация будет взята из файлов MOZILLA1.REG и MOZILLA2.REG.

Способ 2.

Пошаговые инструкции для полного восстановления реестра Windows 10:

1. Войдите в систему с необходимыми правами, например, как администратор.
2. Запустите NTBackup.
3. Если NTBackup запустилась в режиме мастера, нажмите кнопку "Расширенный" в окне мастера архивации.
4. Перейдите на вкладку "Восстановление и управление носителем"
5. Установите в списке "Установите флажки для всех объектов, которые вы хотите восстановить" флажок для объекта "Состояние системы". Это позволит восстановить данные состояния системы вместе с остальными данными, отмеченными в текущем задании восстановления.

6. Отчёты о проделанной работе находятся в папке x:

\Documents and Settings\%User%\Local Settings\Application Data\Microsoft\Windows NT\NTBackup\data\ в пронумерованных файлах типа backup01.log, backup02.log и т.д.

Восстановление повреждённого реестра когда Windows 10 не загружается

А теперь мы посмотрим, что нужно делать, когда из-за ошибок в реестре Windows 10 не загружается.

Описываемая процедура не гарантирует полное восстановление системы к предыдущему состоянию; однако, мы сможем восстановить наши данные.

Разрушенные файлы системного реестра могут вызывать ряд различных сообщений об ошибках.

Попробуйте при загрузке Windows 10 нажать F8 и выбрать вариант "Загрузка последней удачной конфигурации" (Boot Using Last Known Good Configuration). При этом восстанавливаются только данные в разделе реестра HKLM\System\CurrentControlSet. Любые изменения в других разделах реестра сохраняются. Загрузка последней удачной конфигурации позволяет восстановить

реестр в случае неполадок, вызванных, например, новым, несовместимым с имеющимся оборудованием, драйвером. Неполадки, возникшие вследствие повреждения или ошибочного удаления драйверов или файлов, не могут быть устранены таким образом.

Итак, при попытке запуска Windows 10 вы получаете сообщение об ошибке, например, одно из указанных ниже:

*Windows 10 could not start because the following file is missing or corrupt:
\\WINDOWS\\SYSTEM32\\CONFIG\\SYSTEM*

*Windows 10 could not start because the following file is missing or corrupt:
\\WINDOWS\\SYSTEM32\\CONFIG\\SOFTWARE*

*Stop: c0000218 {Registry File Failure} The registry cannot load the hive (file):
\\SystemRoot\\System32\\Config\\SOFTWARE or its log or alternate*

Очень хорошо, теперь настала пора применить ваши знания на практике. Если вы когда-либо выполняли NTBACKUP и завершили системное копирование успешно, то вы можете сразу приступить к **4-ому шагу**.

Шаг 2.

Чтобы выполнить процедуру, описанную в этом разделе, вы должны войти как администратор, или как пользователь приравненный к администратору. Т.е. пользователь имеющий учетную запись в группе Администраторы.

Выполняем следующие действия:

1. Перегрузите компьютер.
2. При загрузке Windows 10 нажмите F8.
3. Выберите безопасный режим.

Если вы используете проводник в качестве файл-менеджера, то придётся выполнить несколько действий, чтобы сделать папку System Restore видимой:

1. Запускаем "Проводник".
2. В меню "Сервис" выбираем "Свойства папки" и далее закладку "Вид".
3. Раскрываем опцию "Скрытые файлы и папки" и щёлкаем на "Показывать скрытые файлы и папки".
4. Далее щёлкаем на "Применить" и "Ок".

Теперь:

1. Открываем раздел жёсткого диска где установлена Windows 10 и находим папку System Volume Information. Примечание: Это скрытая системная папка. Она содержит одну или более папок с именами вида _restore {GUID} , например, _restore{87BD3667-3246-476B-923F-F86E30B3E7F8}

2. Откройте папку, которая была создана НЕ в текущее время. Это может быть одна или больше папок, имена которых начинаются с "RP". Это - точки восстановления.

3. Откройте выбранную папку и затем папку с именем Snapshot. Например,
c:\System Volume Information_restore{DBB3294C-F5C9-43A9-9010-A75010CD2631}\RP2\snapshot

4. Из папки Snapshot в папку C:\Windows\Tmp, уже созданную на первом этапе, скопируйте следующие файлы:

- REGISTRY_USER_DEFAULT
- REGISTRY_MACHINE_SECURITY
- REGISTRY_MACHINE_SOFTWARE
- REGISTRY_MACHINE_SYSTEM
- REGISTRY_MACHINE_SAM

Эти файлы созданы службой восстановления системы - System Restore. Так как на предыдущем шаге мы использовали файлы системного реестра, созданные при начальной установке Windows 10, то этот "новый" системный реестр не знает, что "старые" точки восстановления существуют и доступны. При загрузке Windows 10 создана новая папка с новым GUID и с новым System Volume Information, и создана новая точка восстановления, которая включает копию файлов нового системного реестра. Вот почему важно не использовать самую новую папку, особенно, если время ее создания - текущее время.

Таким образом конфигурация существующей системы не знает о предыдущих точках восстановления. Нам нужна предыдущая, "старая" копия системного реестра от предыдущей, "старой" точки восстановления, чтобы сделать все предыдущие, "старые" точки восстановления доступными. Я надеюсь, что вы меня поняли.

Файлы системного реестра были скопированы из папки Snapshot в папку C:\Windows\Tmp чтобы сделать их доступными, когда мы будем находиться в Recovery Console. Мы будем использовать эти файлы, чтобы заменить ими файлы текущего системного реестра в папке C:\Windows\System32\Config. Дело в том, что в Recovery Console папка с System Volume Information в общем случае недоступна.

Практическое занятие «Создание резервной копии базы данных»

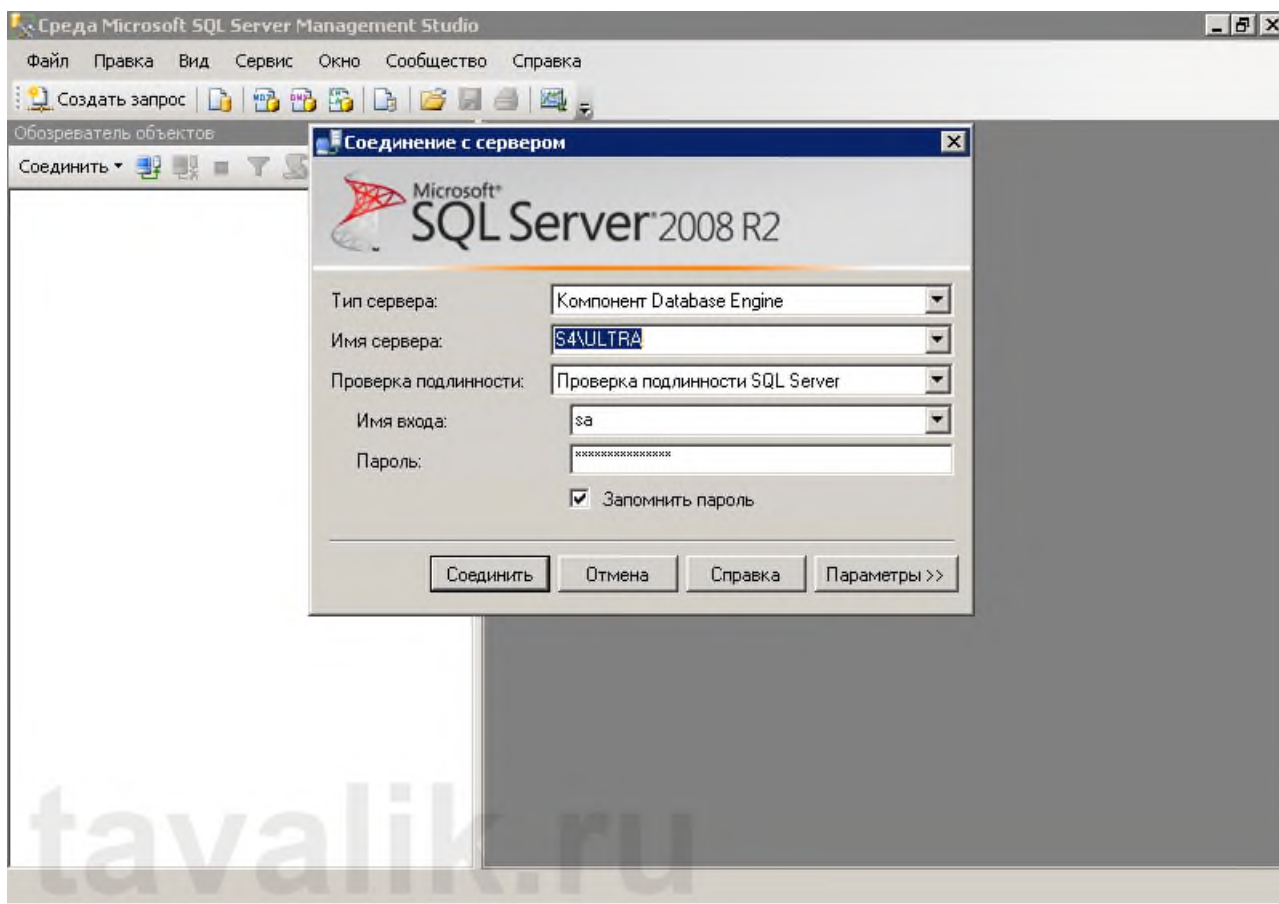
Цель: научиться создавать резервные копии базы данных

Задание

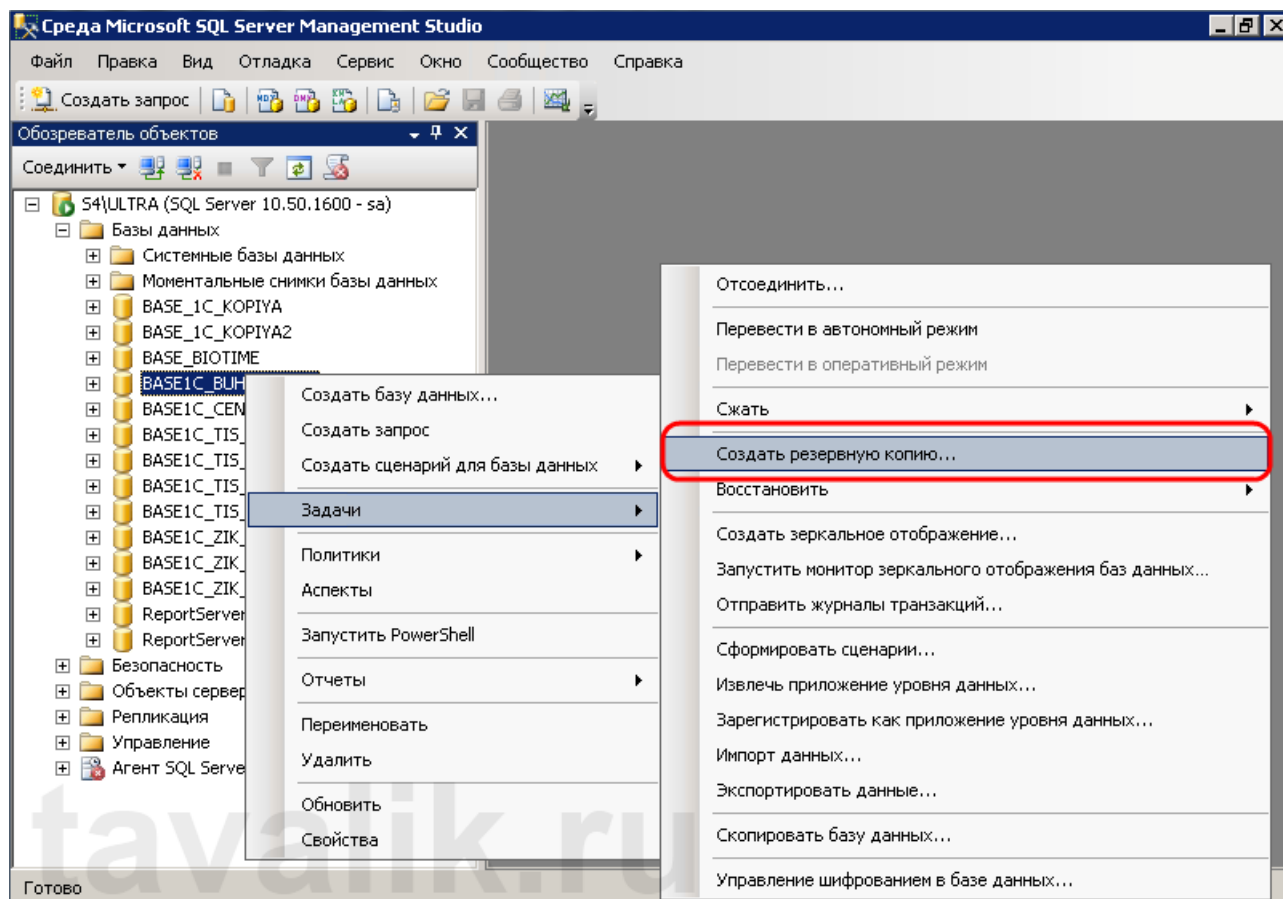
Как вручную сделать полную резервную копию базы данных в SQL Server 2008 R2 с помощью программы «Среда Microsoft SQL Server Management Studio».

1. Создание резервной копии

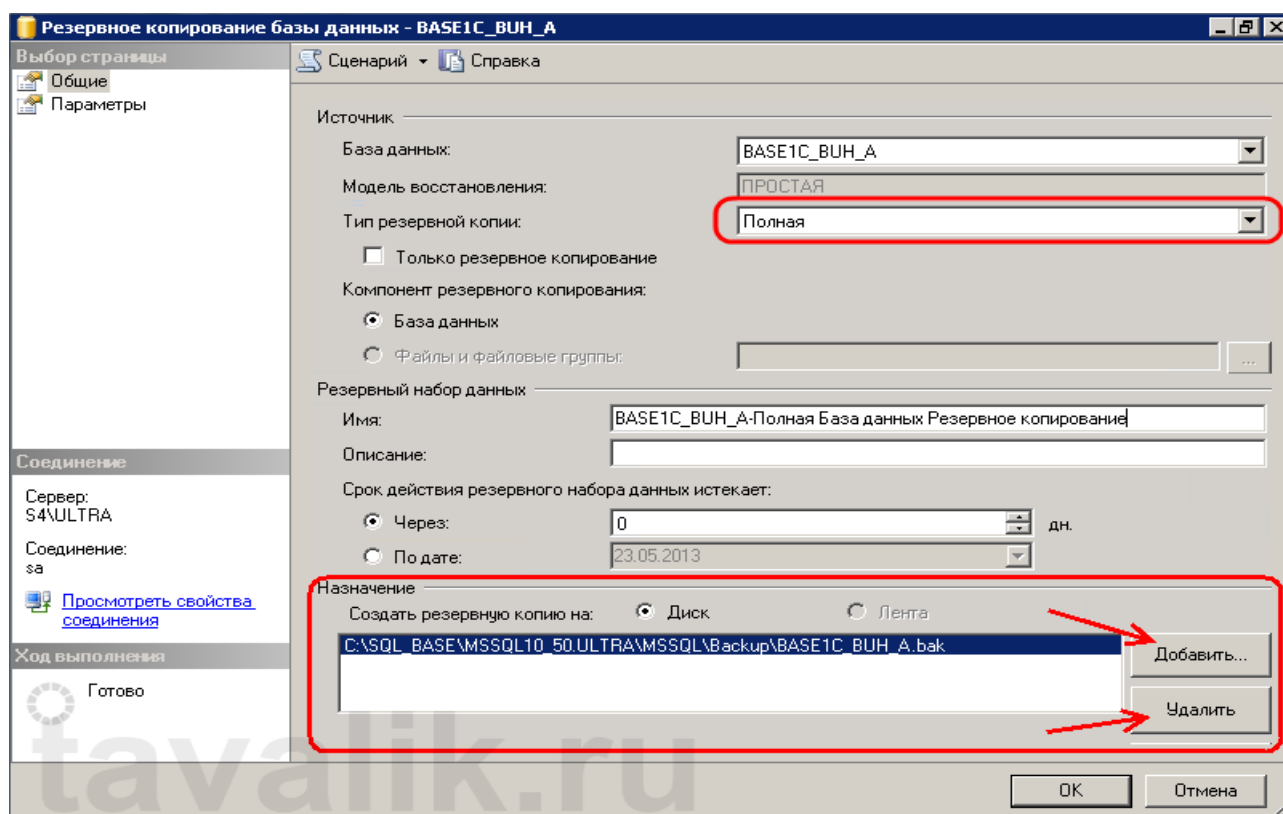
На самом деле все довольно просто. Запускаем оснастку «Среда Microsoft SQL Server Management Studio» («Пуск» — «Все программы» — «SQL Server 2008 R2» — «Среда Microsoft SQL Server Management Studio») и вводим данные для авторизации.



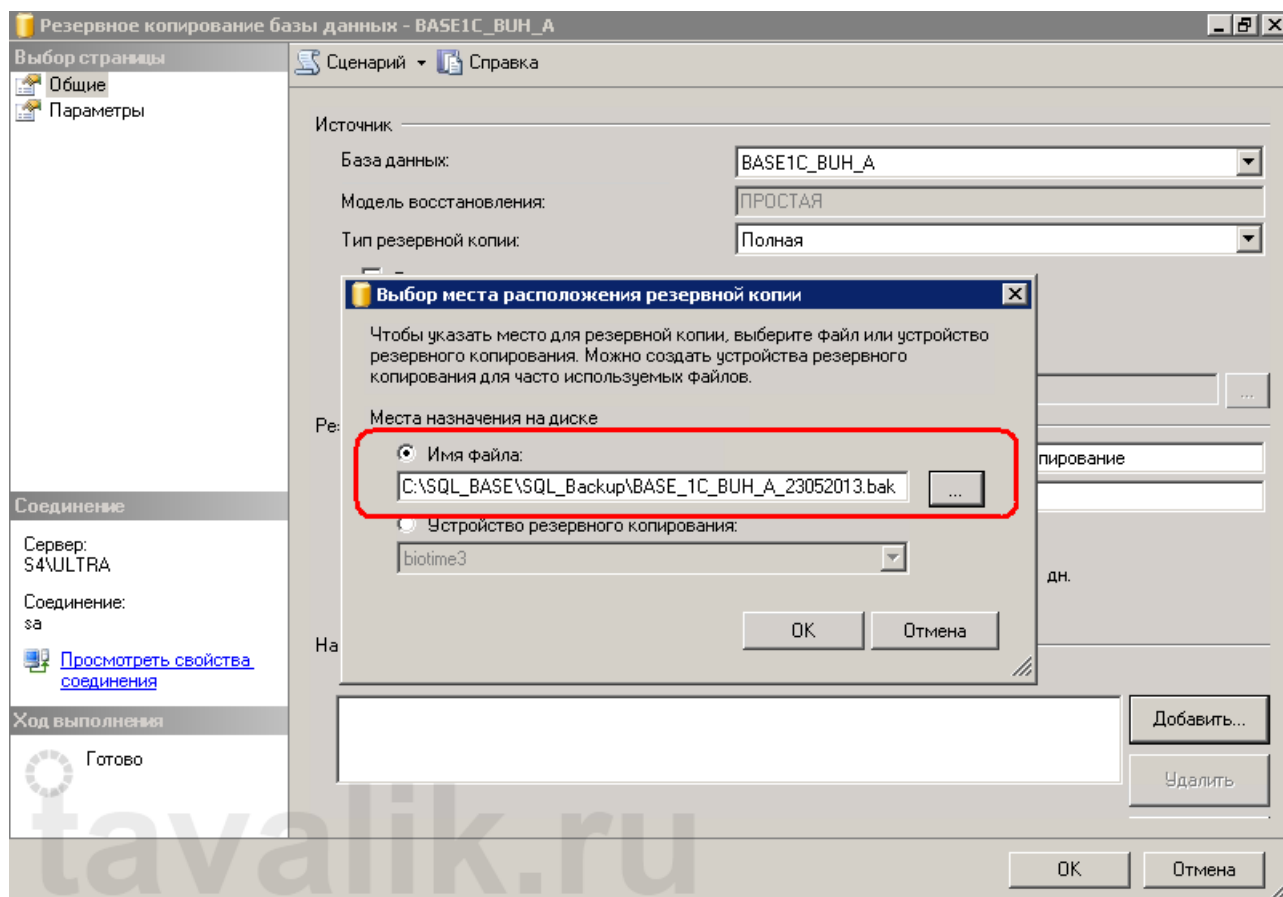
После чего в Обозревателе объектов раскрываем вкладку «Базы данных» и кликнем правой кнопкой мыши по той базе данных, для которой необходимо сделать резервную копию. В появившемся контекстном меню выберем «Задачи» (Tasks) — «Создать резервную копию» (Back Up...).



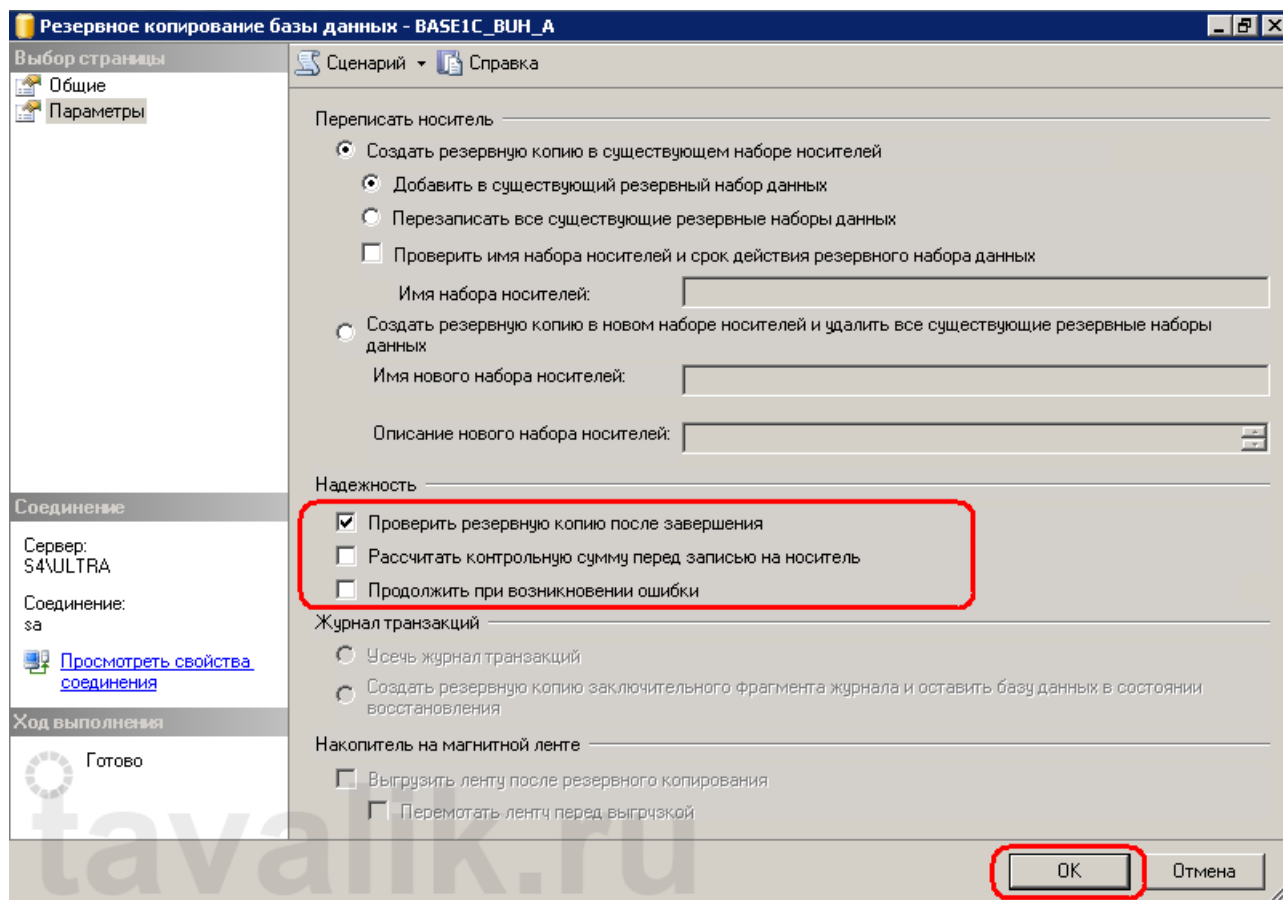
Запустится окно «Резервная копия базы данных» (*Back Up Data Base*) . Убедимся, что тип резервной копии стоит «Полная» (*Full*), при необходимости зададим имя и описание, а также укажем назначение резервной копии. По умолчанию выбран путь на жестком диске компьютера в папку Ваксир основного расположения баз SQL-сервера. Для того чтобы изменить место размещения копии, сначала нажмем «Удалить» (*Remove*), чтобы удалить существующее назначение, а затем «Добавить» (*Add...*) для добавления нового.



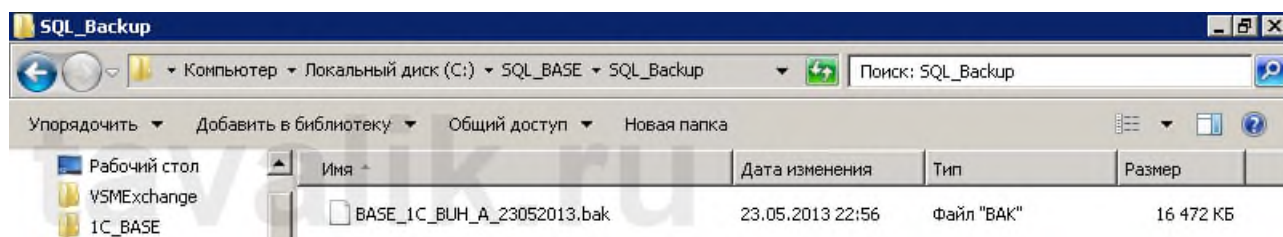
Здесь зададим расположение и имя файла резервной копии и нажмем «OK». Таких мест назначений можно задать несколько. В этом случае резервная копия будет разбита на равные части, каждая часть в указанном файле.



Далее перейдем на вкладку «*Параметры*» (*Options*), на которой можно указать, что резервную копию необходимо проверить после выполнения, а также задать другие параметры надежности.

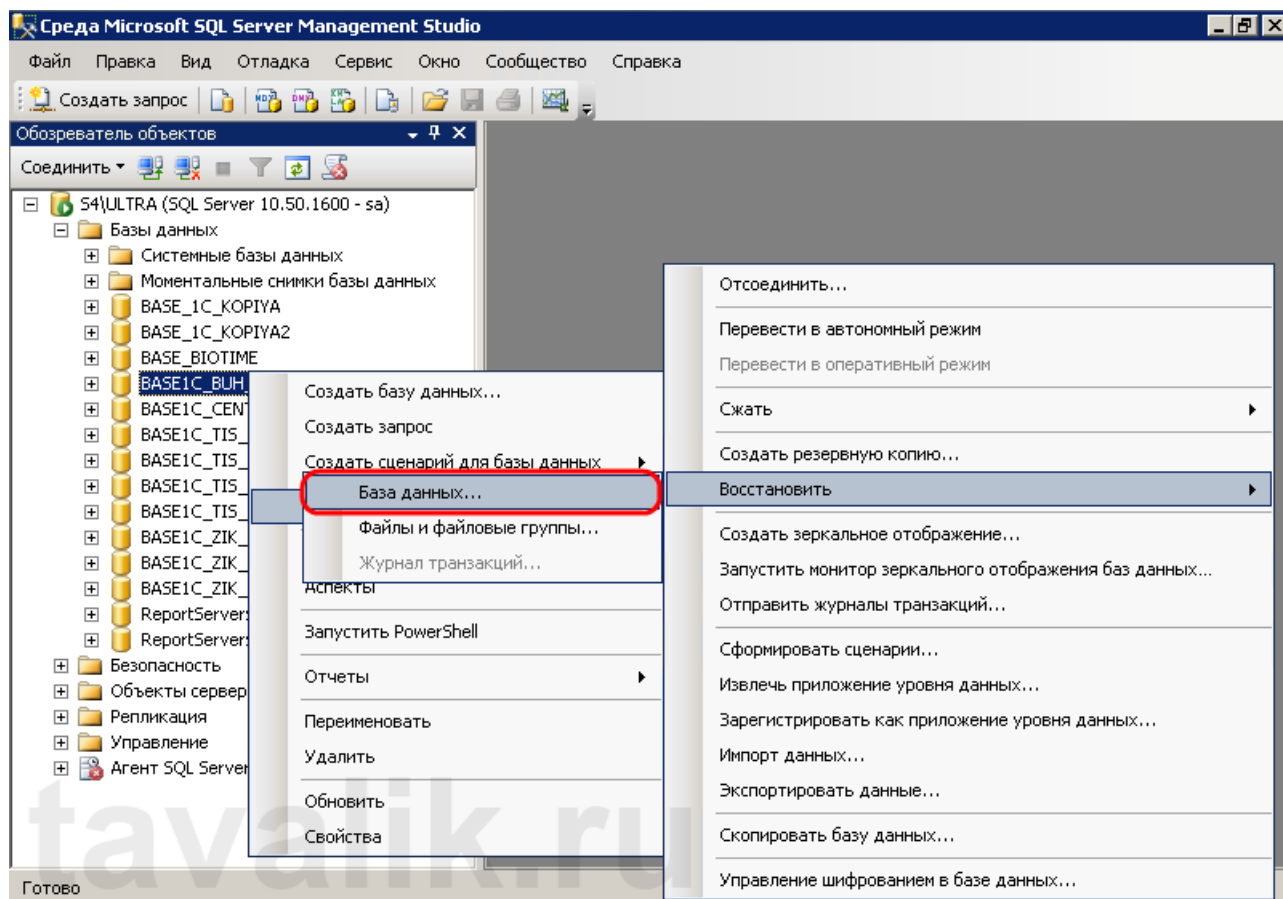


Когда все настройки установлены, нажимаем «*OK*» и ждем завершения задачи. Если все сделано правильно, в указанной директории мы найдем файл резервной копии базы данных SQL.

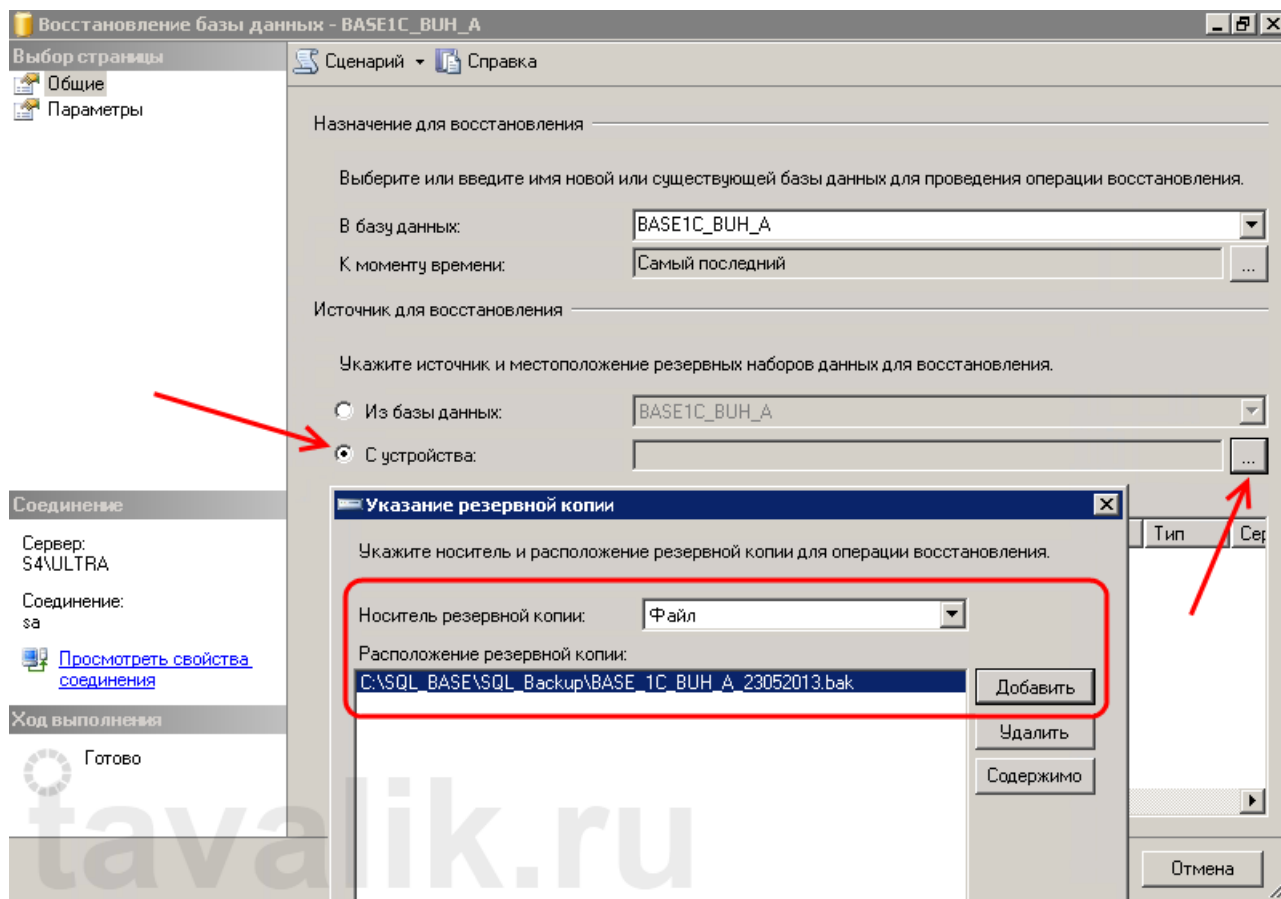


2. Восстановление базы данных из резервной копии

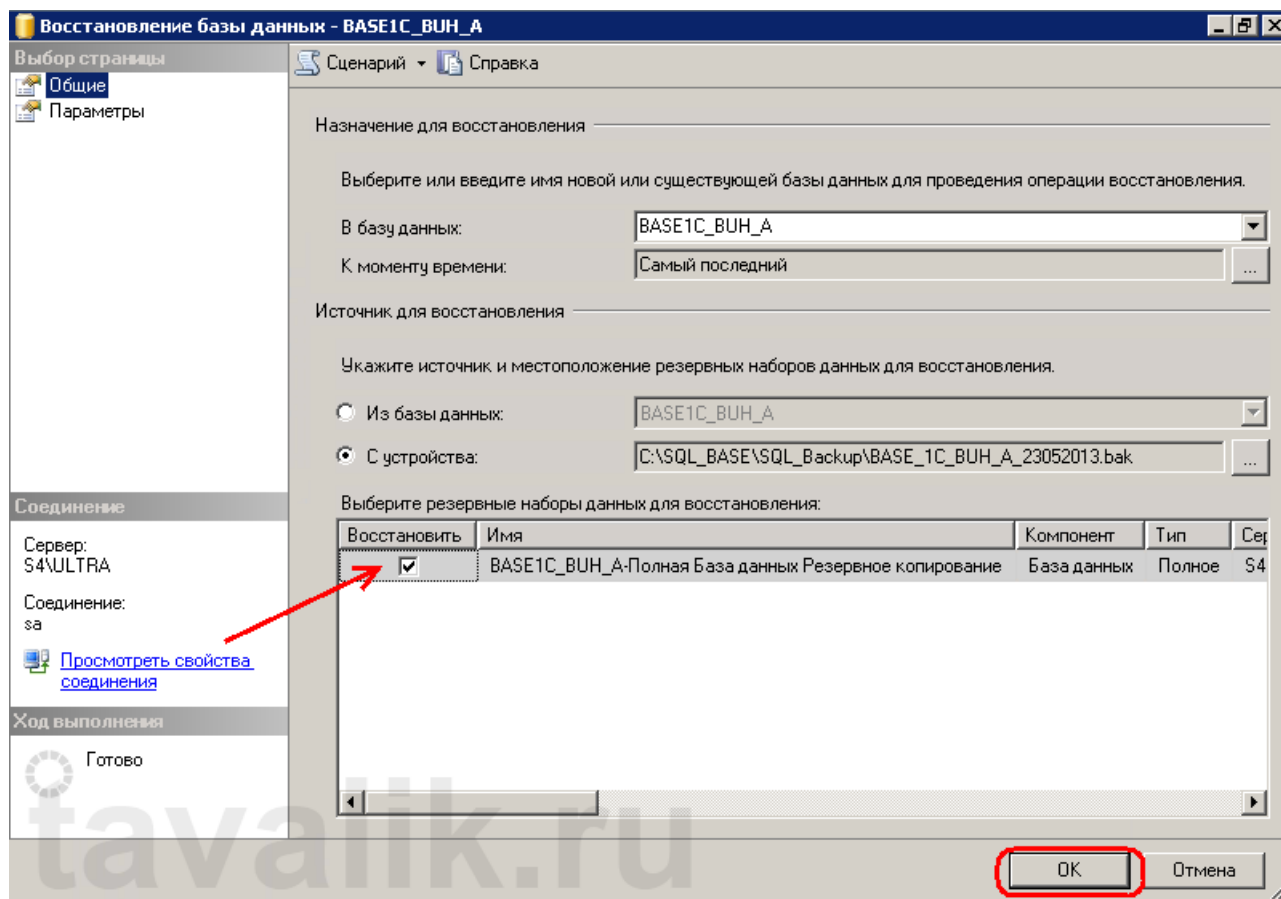
Восстановление происходит по аналогичной схеме. В «*Среде Microsoft SQL Server Management Studio*» выбираем базу **из которой сделана резервная копия**, кликаем по ней правой кнопкой мыши, в контекстном меню выбираем «*Задачи*» (*Tasks*) — «*Восстановить*» (*Restore*) — «*База данных...*» (*Database...*).



Откроется окно «Восстановление базы данных» (*Restore Database*). Здесь, в качестве источника укажем «С устройства» (*From device*) и выберем файл резервной копии (созданных в пункте 1).



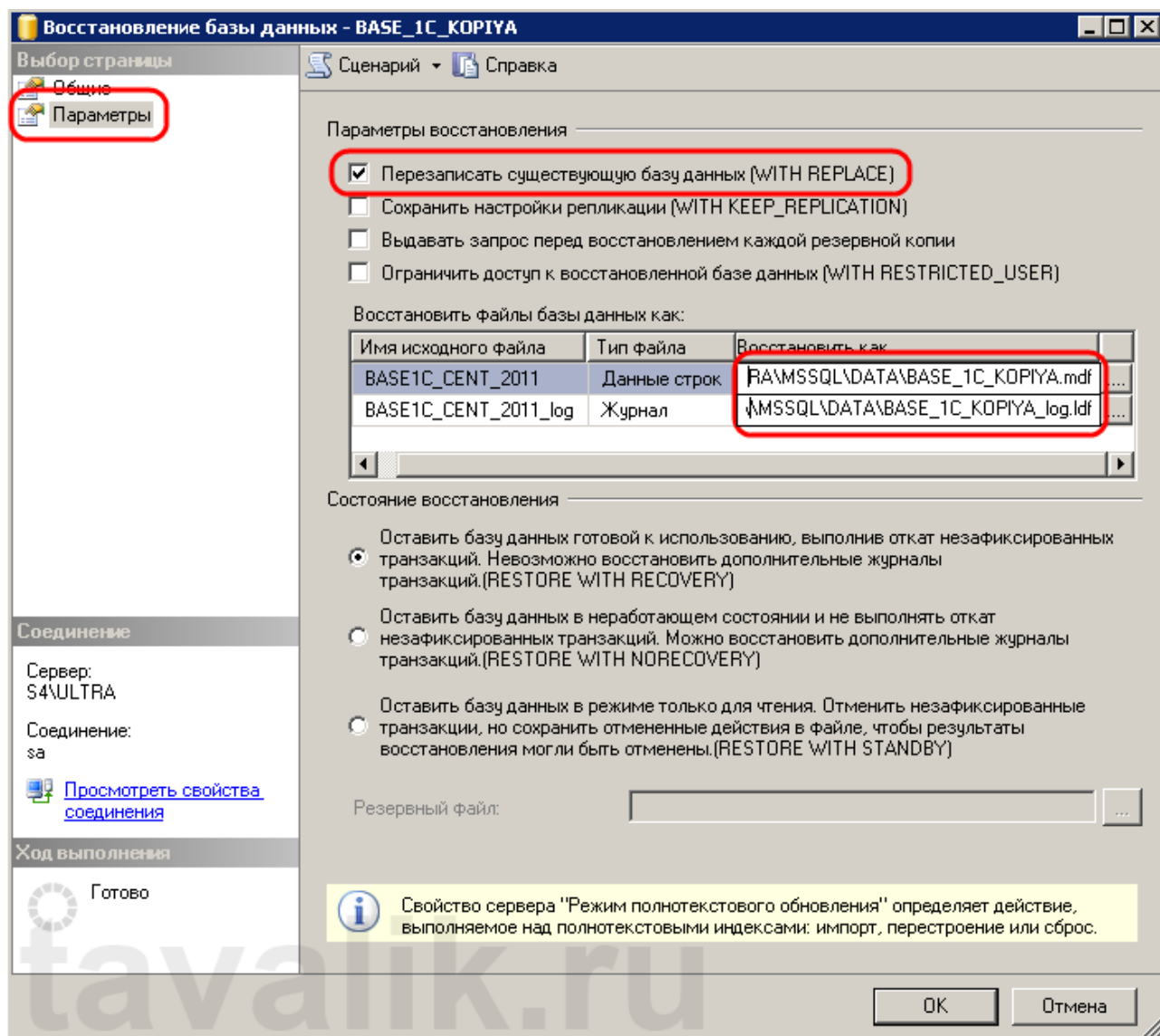
Установим флаг «Восстановить» (*Restore*) напротив выбранной резервной копии. При необходимости, на вкладке «Параметры» (*Options*), можно указать дополнительные параметры восстановления, о значении которых можно прочитать [здесь](#).



После того, как все настройки сделаны, жмем «OK» и ожидаем сообщения об успешном восстановлении базы данных.

3. Восстановление резервной копии в другую базу данных (копирование данных)

Если же необходимо загрузить данные в базу данных, **отличную от той из которой была сделана резервная копия**, то при загрузке помимо действий, описанных в пункте 2, необходимо на вкладке «*Параметры*» (Options) задать имена файлов этой базы данных и установить флаг «*Перезаписывать существующую базу данных*» (WITH REPLACE).



Практическое занятие «Восстановление данных»

Цель: научиться восстанавливать данные

Задание

"Восстановление Базы данных". Этот пункт служит для восстановления испорченной базы данных с помощью созданных ранее резервных копий. Открывающееся окно содержит следующие элементы: - "Восстановить" - эта группа переключателей предназначена для выбора типа устройства или архива, с которого предполагается восстановить резервную копию (база данных, группы файлов или файлы, устройство); - "Тип восстановления" - этот раскрывающийся список содержит список всех копий баз данных. Выбрав ту или иную копию и указав имя копии и устройство, на котором оно расположено, после нажатия кнопки ОК можно с помощью таблицы, расположенной в нижней части вкладки, ознакомиться со списком резервных копий, созданных ранее для этой базы данных. Если сбой произошел после создания копии и внесения дополнительных изменений в БД

1

АДМИНИСТРИРОВАНИЕ СЛУЖБ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ (выполнения транзакций), следует осуществить ввод транзакций (не сохраненных в копии) вручную, выбрав в меню "Файл" пункт "Ввод транзакций вручную". При этом вводятся все транзакции, включая и ту, на которой произошел сбой. Если после ручного ввода транзакций база данных примет вид, какой она имела на момент сбоя, то появится сообщение об успешном выполнении восстановления, иначе высветится сообщаящее о том, что база данных не восстановлена должным образом. Порядок выполнения работы

1. Запустить файл "SQLServer2000Emu.exe".
2. Создать исходную базу данных (в имени базы данных должны присутствовать номер группы и варианта задания, например, "Database035_02"). Номер варианта определяется номером бригады в списке журнала преподавателя.
3. Сформировать файлы на рабочих станциях для исходной базы данных.
4. Внести изменения в базу данных (количество транзакций для каждого из вариантов задать, как $15+N$, где N – номер варианта).
5. Создать новое устройство резервного копирования (в имени файла-копии также отразить номер группы и вариант, например, "Databackup0_035_2").
6. Создать полную резервную копию базы данных.
7. Просмотреть и зафиксировать содержимое исходной и резервной БД.
8. Внести новые изменения в базу данных (число транзакций - $10+N$).
9. Создать дифференцированную резервную копию базы данных с использованием мастера резервного копирования.

10. Запротоколировать изменения, внесенные в базу данных между полным и дифференцированным режимами копирования.

11. Внести изменения в базу данных (10+N транзакций).

12. Создать резервную копию журнала транзакций на новом устройстве резервного копирования.

13. Просмотреть и зафиксировать содержимое журнала транзакций.

АДМИНИСТРИРОВАНИЕ СЛУЖБ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ

14. Внести изменения в базу данных (5+N транзакций).

15. Просмотреть и зафиксировать параметры введенных транзакций (от начала последнего ввода транзакций до момента сбоя в системе включительно).

16. Выполнить восстановление базы данных, осуществляя последовательно восстановление с использованием полной, дифференциальной копий и копии журнала транзакций.

17. Ввести вручную зафиксированные транзакции, чтобы привести базу данных к виду, в котором она находилась на момент сбоя.

18. Проверить правильность восстановления информации. Содержание отчета Отчет о лабораторной работе должен содержать: 1) краткие теоретические сведения; 2) описание содержимого базы данных (исходной, после полного и дифференциального резервного копирования) и журнала транзакций (после сохранения и до момента сбоя в информационной системе);

17 АДМИНИСТРИРОВАНИЕ СЛУЖБ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ 3) описание условий, при которых производится копирование информации.

Практическое занятие «Восстановление работоспособности системы»

Цель: научиться восстанавливать работоспособность системы

Задание

Как запустить восстановление системы в Windows 10



Восстановление Windows 10 позволяет вернуть операционную систему к работоспособному или исходному состоянию из созданной автоматически или вручную точки отката системы или хранимого на винчестере полного образа системы. Также в наборе инструментов «десятки» числится средство выполнения сброса ОС, которое избавит от длительной переустановки Windows 10, и создание флешки восстановления, необходимой для возобновления функционирования операционной системы в критических ситуациях (когда Windows 10 не загружается и не предоставляет возможности попасть в среду восстановления).

Предложенная статья-инструкция рассказывает обо всех инструментах, из которых состоит среда восстановления системы в Windows 10, механизмах их функционирования, способах использования той или иной функции и эффективности методов восстановления в определенных ситуациях. Прежде чем приступить к изучению материала, рекомендуется ознакомиться с инструкциями на тему восстановления загрузчика операционной системы, проверки ее файлов на целостность и восстановления поврежденных файлов реестра.

- 1 Безопасный режим
- 2 Возвращаем компьютер/ноутбук в исходное состояние
- 3 Флешка восстановления Windows 10
- 4 Создаем полный образ реанимации системы
- 5 Точки отката Windows 10
- 6 История файлов

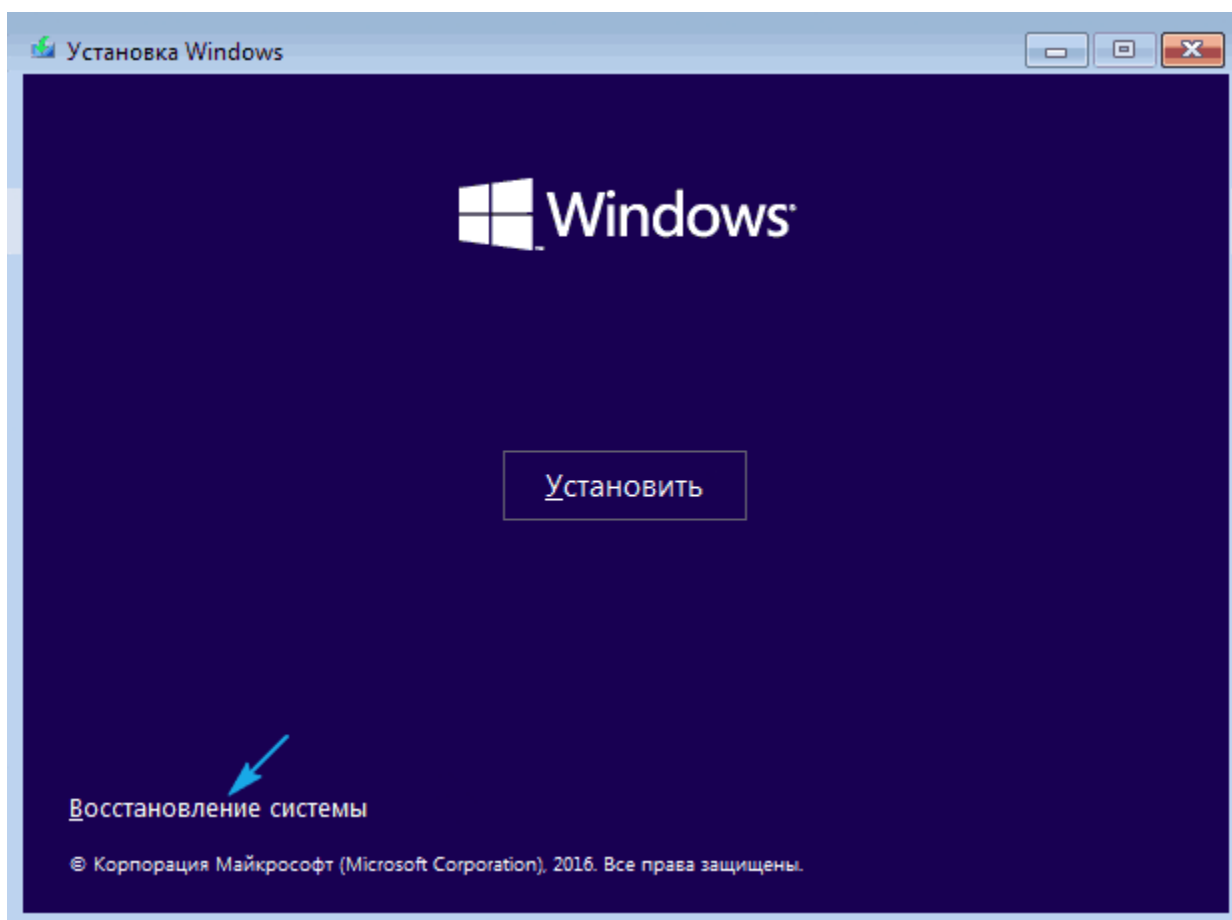
Безопасный режим

Первое, что следует попробовать при появлении неполадок, загрузиться в безопасном режиме. Рассмотрим ситуацию, когда «десятка» не загружается и не позволяет выполнить перезагрузку с соответствующими параметрами (попасть в этот режим через msconfig или особые варианты загрузки не получится).

1. Запускаемся из загрузочного носителя с дистрибутивом Windows 10, воспользовавшись Boot Menu.

2. Указываем «Русский» язык жмем «Далее».

3. В следующем окошке жмем по ссылке «Восстановление системы».



4. Выполняем команду «bcdedit /set safeboot minimal» для последующего запуска компьютера в безопасном режиме.

5. Перезагружаемся, закрыв все окна.

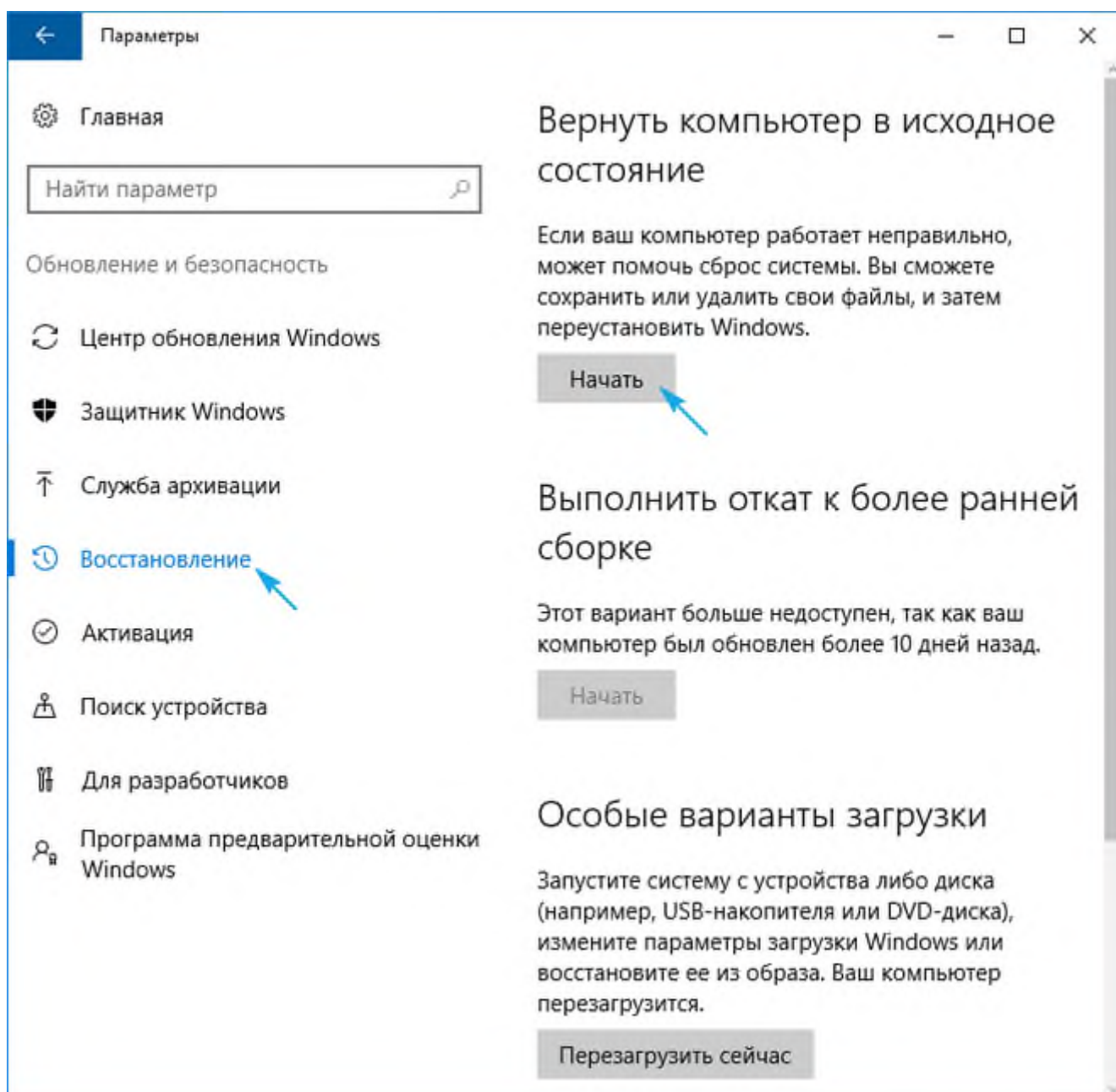
После запуска компьютера можно заняться решением проблемы, которая препятствует нормальному запуску/функционированию ПК.

Возвращаем компьютер/ноутбук в исходное состояние

Самая примечательная функция восстановления, которая появилась в Windows 10, — это возврат Виндовс к исходному состоянию. Воспользоваться ею можно через «Параметры».

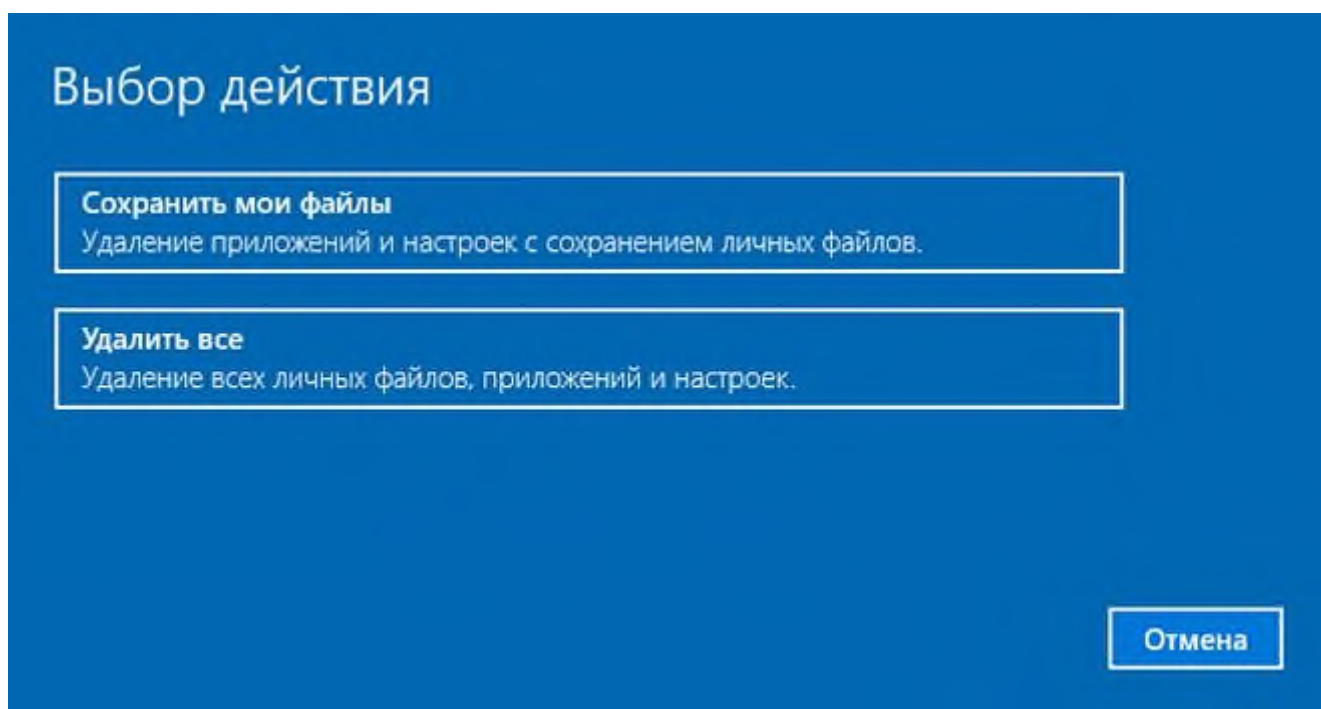
1. Вызываем меню при помощи Win→I.
2. Переходим в раздел «Обновление/безопасность».
3. Нажимаем по вкладке «Восстановление».

Пользоваться этой возможностью следует в самую последнюю очередь, когда приведенные ниже варианты не помогли решить проблему, ведь в итоге получите чистую только проинсталлированную операционную систему с личными данными или без них, в зависимости от указанных параметров.



4. Жмем «Начать», после чего появится диалог с предложением указать параметры сброса операционной системы.

Первый вариант — это полная очистка системного тома ото всех файлов и быстрая переустановка «десятки» без задействования установочного диска с дистрибутивом операционной системы, второй способ — быстрая инсталляция Windows 10 с сохранением файлов пользователя и настроек установленных на ПК приложений, сами же программы сохранены не будут.



Существует еще один путь вызвать диалог сброса операционной системы даже без авторизации в системе. Осуществляется все на экране авторизации. Для получения доступа к функции жмём по пиктограмме «Перезагрузка» с зажатой клавишей Shift. После перезапуска компьютера выполняем клик по пиктограмме «Диагностика», затем жмем по кнопке возврата системы в исходное состояние.

Преимуществами способа являются отсутствие необходимости иметь установочный диск/флешку и выполнение всех действий в автоматическом режиме без какого-либо вмешательства со стороны пользователя. Недостаток всего один — при удалении пользователем образа системы или расположении этого файла в поврежденных секторах жесткого диска совершить быструю переустановку не удастся, но здесь в арсенале «десятки» есть несколько дополнительных инструментов: использование диска восстановления системы при его наличии (очень редкое явление) и резервирование Windows 10 при помощи инструментов ОС на томе, отличающемся от системного.

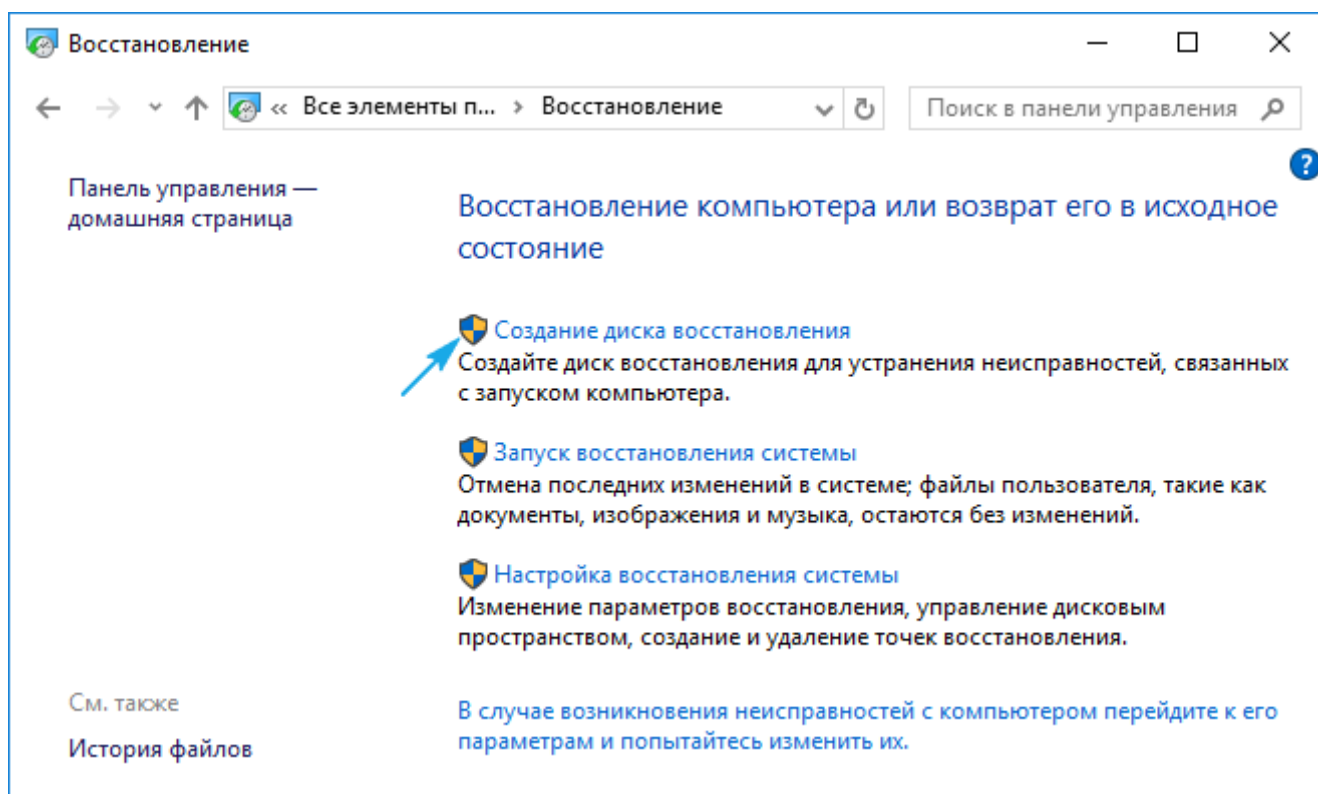
Флешка восстановления Windows 10

Инструмент называется диском восстановления Виндовс, но в «десятке» его следовало бы переименовать во флешку восстановления (будем пользоваться именно этим термином). Если ранее в ОС от Microsoft были утилиты для выполнения автоматической реанимации системы, которые в большинстве случаев только пытались что-то сделать, то в «десятке» присутствует опция создания образа системы для последующего возврата системного тома к запечатленному в этом образе состоянию посредством автоматической переустановки ОС, о чем говорилось разделом выше.

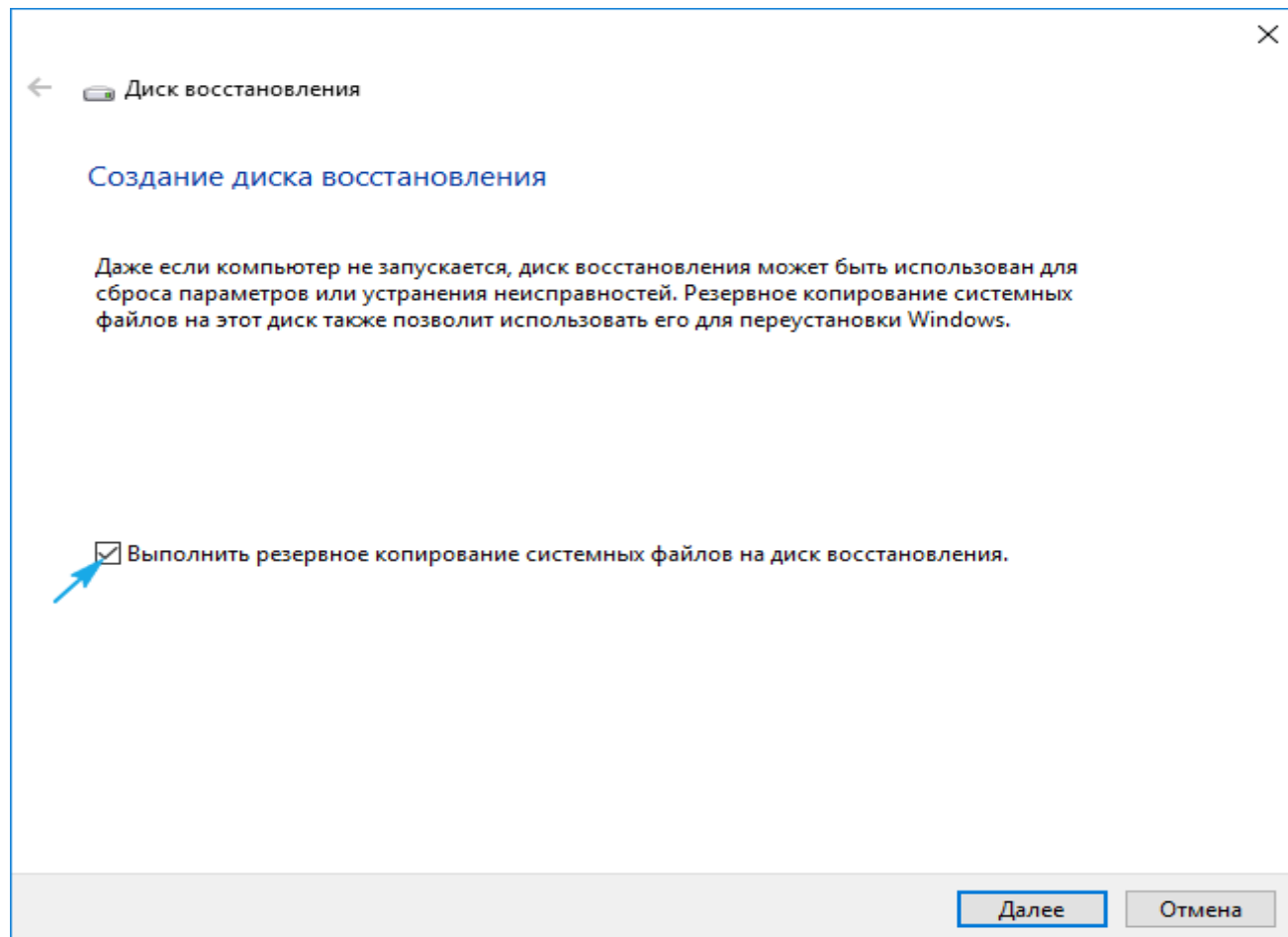
Создается подобный образ следующим путем:

1. Вызываем апплет Панели управления под названием «Восстановление».

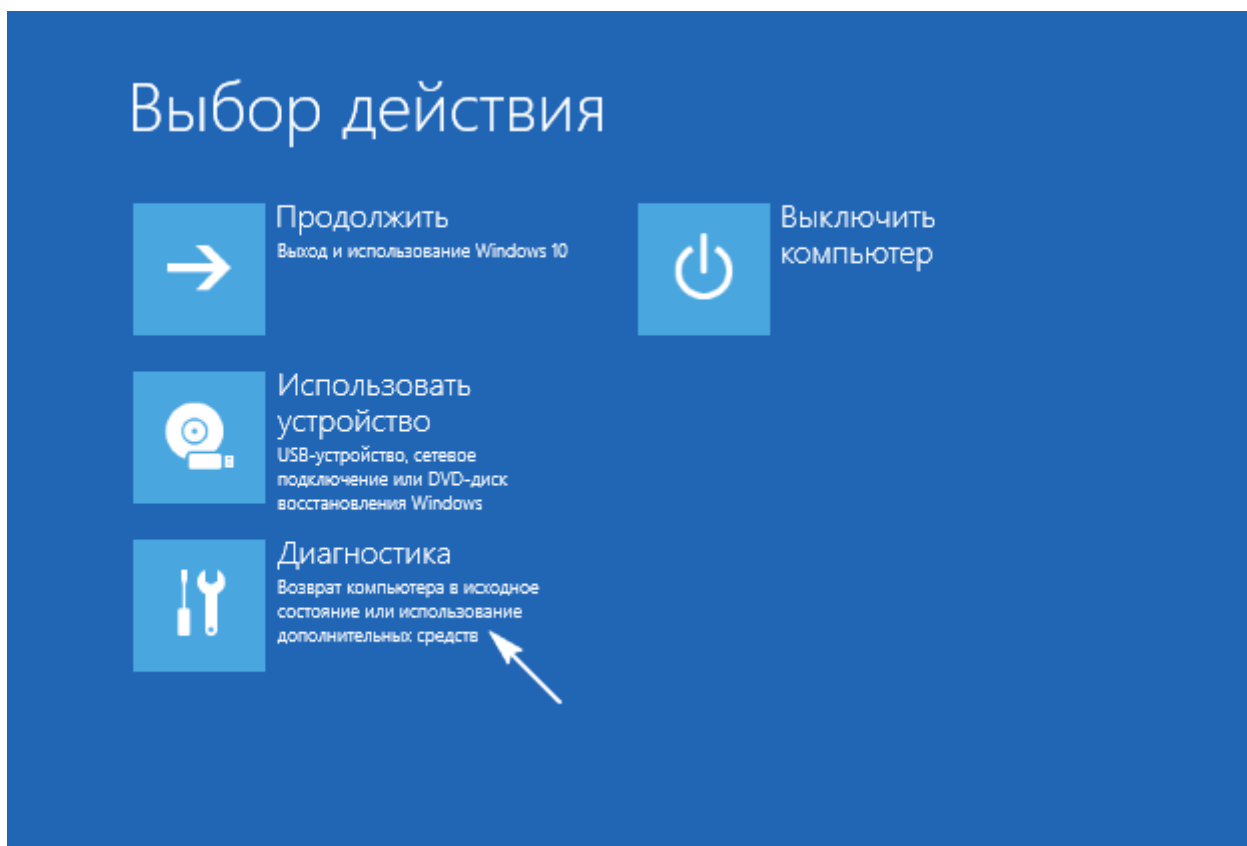
2. Находим ссылку «Создание диска восстановления» в вертикальном меню слева.



3. Отмечаем опцию резервирования системных файлов на флешку восстановления, чтобы получить возможность совершать мгновенную переустановку «десятки».



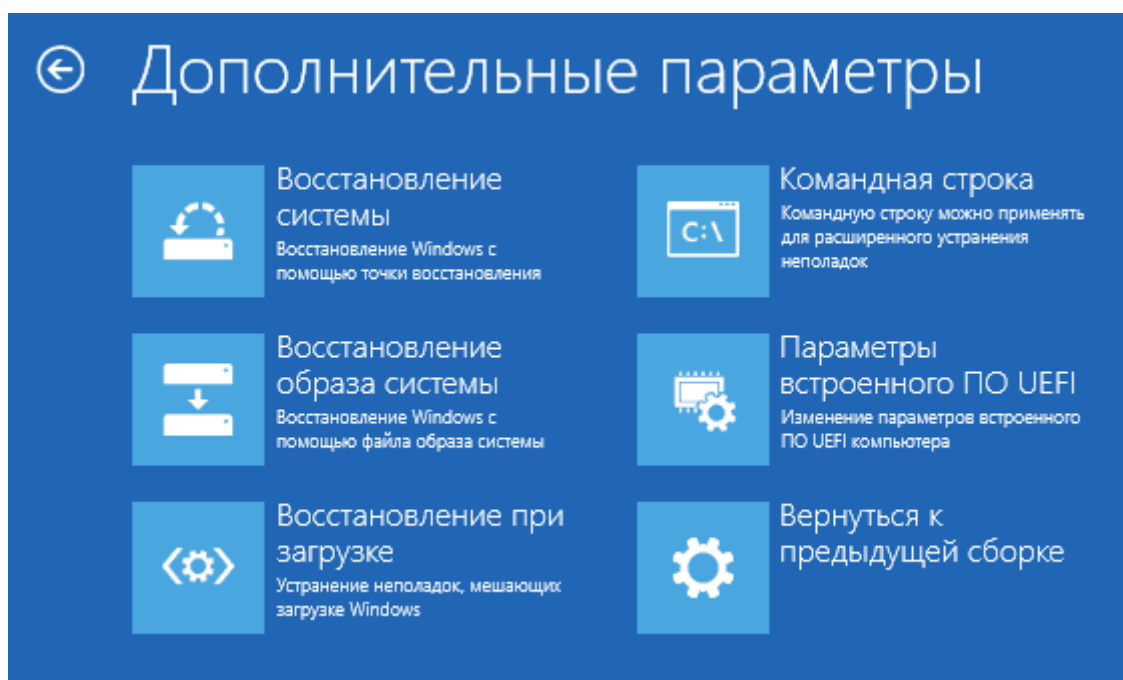
4. По окончании всех операций записи или в случае необходимости загружаемся с созданного накопителя, воспользовавшись функцией Boot Menu.



5. В окне выбора действия переходим в раздел «Диагностика».

Находясь в нем, откроем возможность выполнить следующие операции:

- воспользовавшись флешкой с образом, вернуть Windows 10 к прежнему состоянию;
- посетить параметры UEFI/BIOS;
- прибегнуть к реанимации «десятки» посредством точки отката;
- запустить через командную строку, например, для создания копии загрузчика на соответствующем томе;
- восстановить Windows 10 из полного образа ОС.

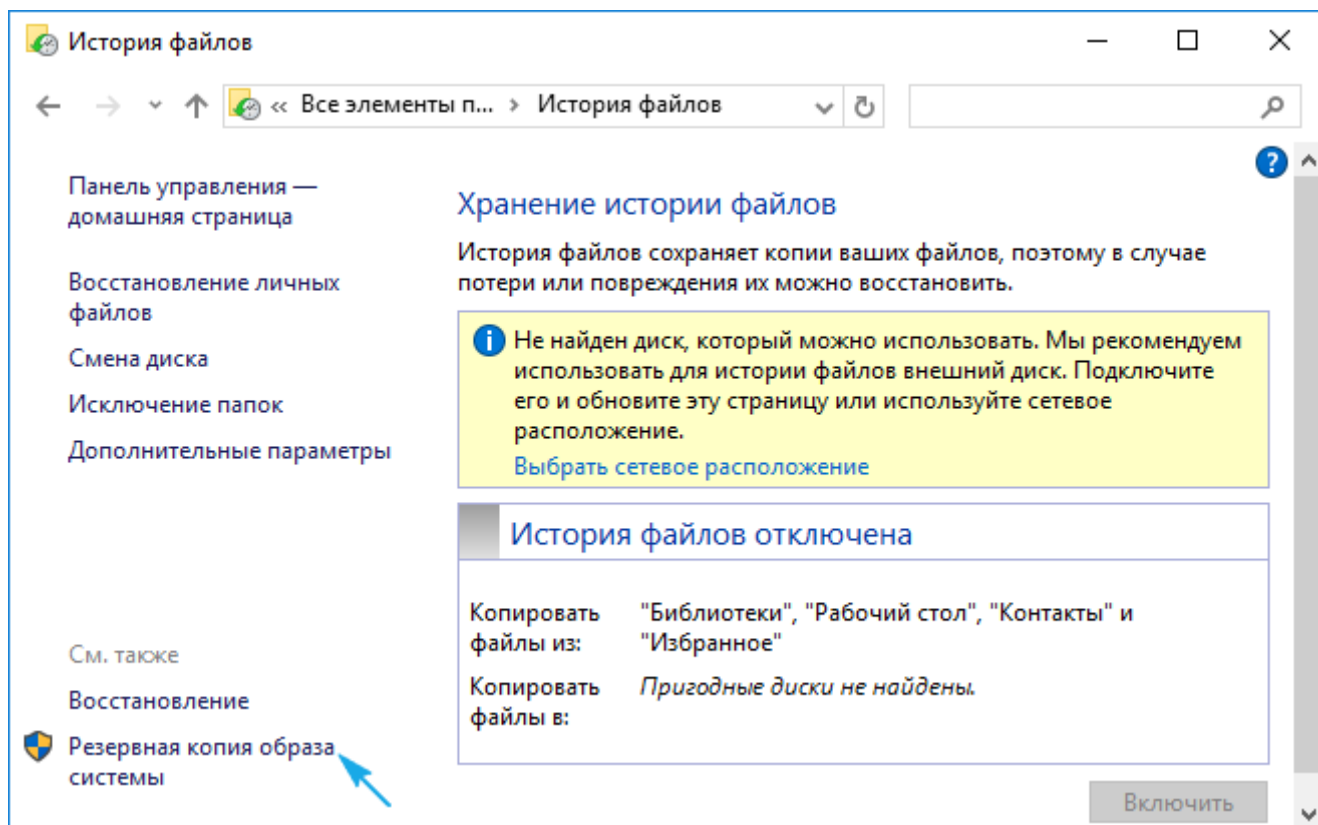


Наличие такой флешки в случае с «десяткой» намного полезнее, чем даже установочной, хотя и последняя позволяет запустить некоторые из операций восстановления операционной системы на экране с кнопкой «Установить» сразу после выбора языка.

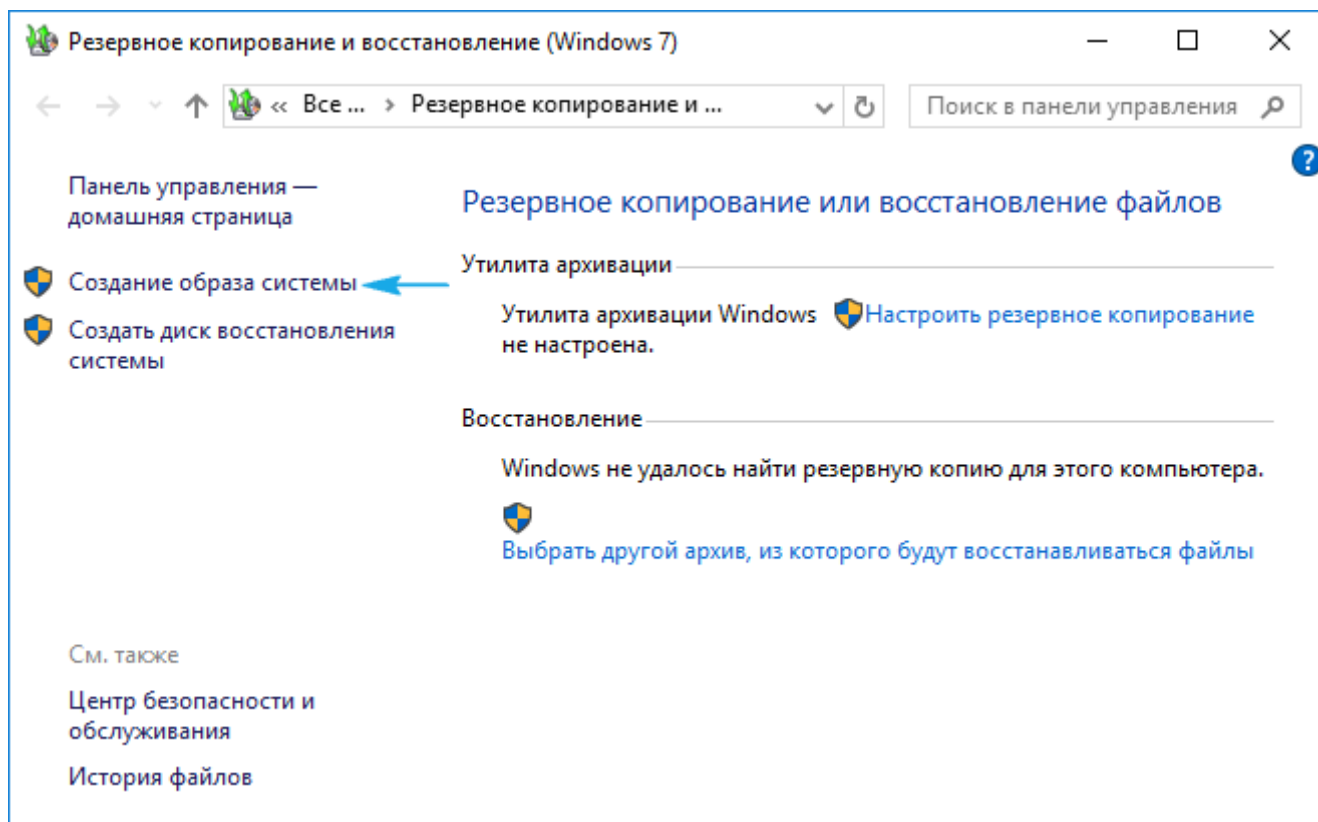
Создаем полный образ реанимации системы

Подготовка автоматического восстановления заключается в создании снимка Windows 10 на время ее нынешнего состояния. Лучше всего создавать такой образ сразу после инсталляции операционной системы со всеми драйверами и софтом, пока системный том не замусорен, как и реестр. **Не обязательно формировать снимок в первые часы функционирования новой ОС, это можно сделать спустя пару дней после ее переустановки, чтобы Windows притерлась и была доведена до нормального функционирующего состояния, но не успела обзавестись мусорными файлами и ключами реестра.**

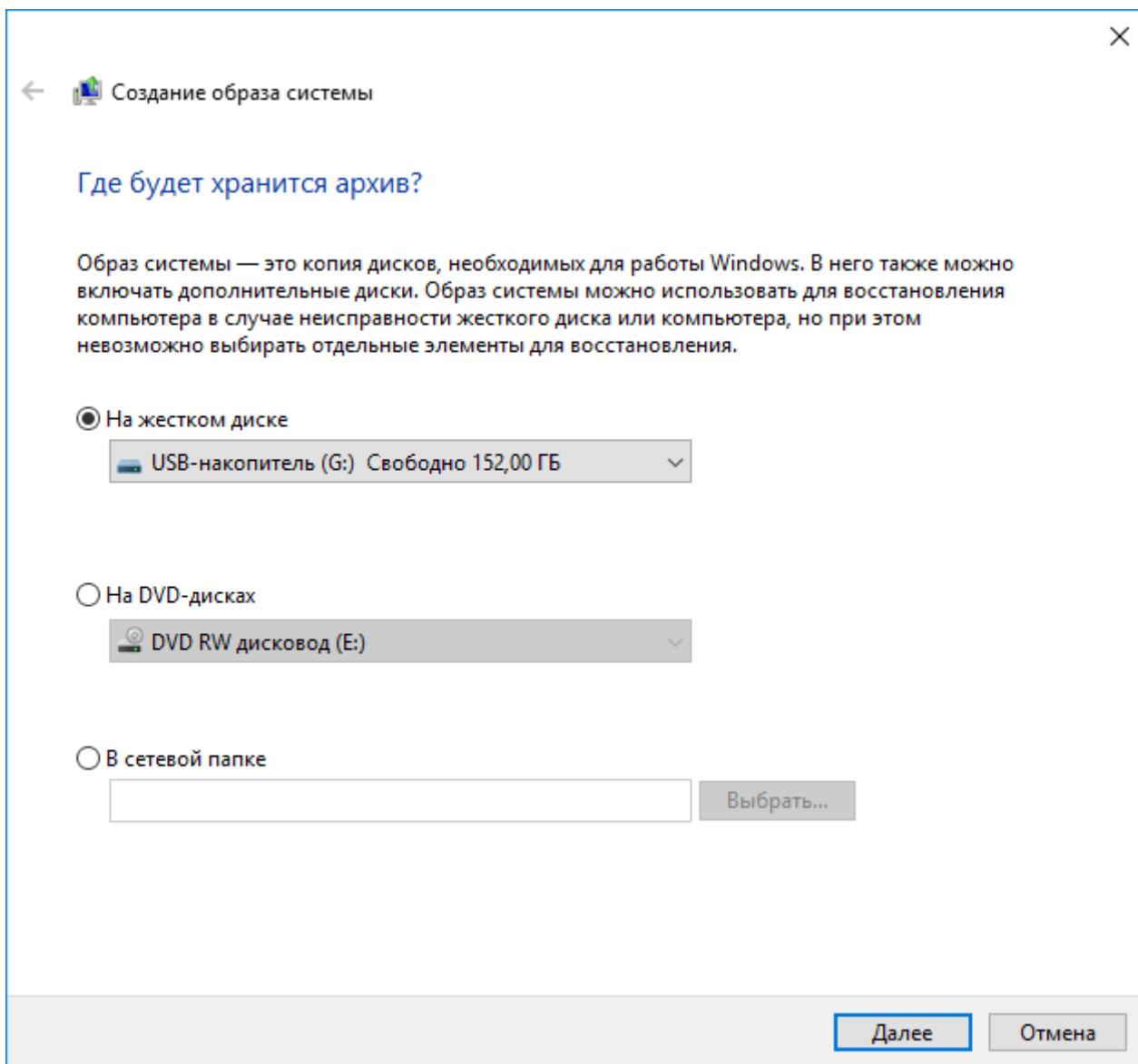
1. Процесс начинается с очистки от мусора диска C: системного реестра и деинсталляции программ, которые оказались ненужными.
2. Далее посещаем Панель управления.
3. Открываем апплет «История файлов», далее нажимаем «Резервная копия образа системы».



4. В вертикальном меню переходим по ссылке «Создание образа системы».



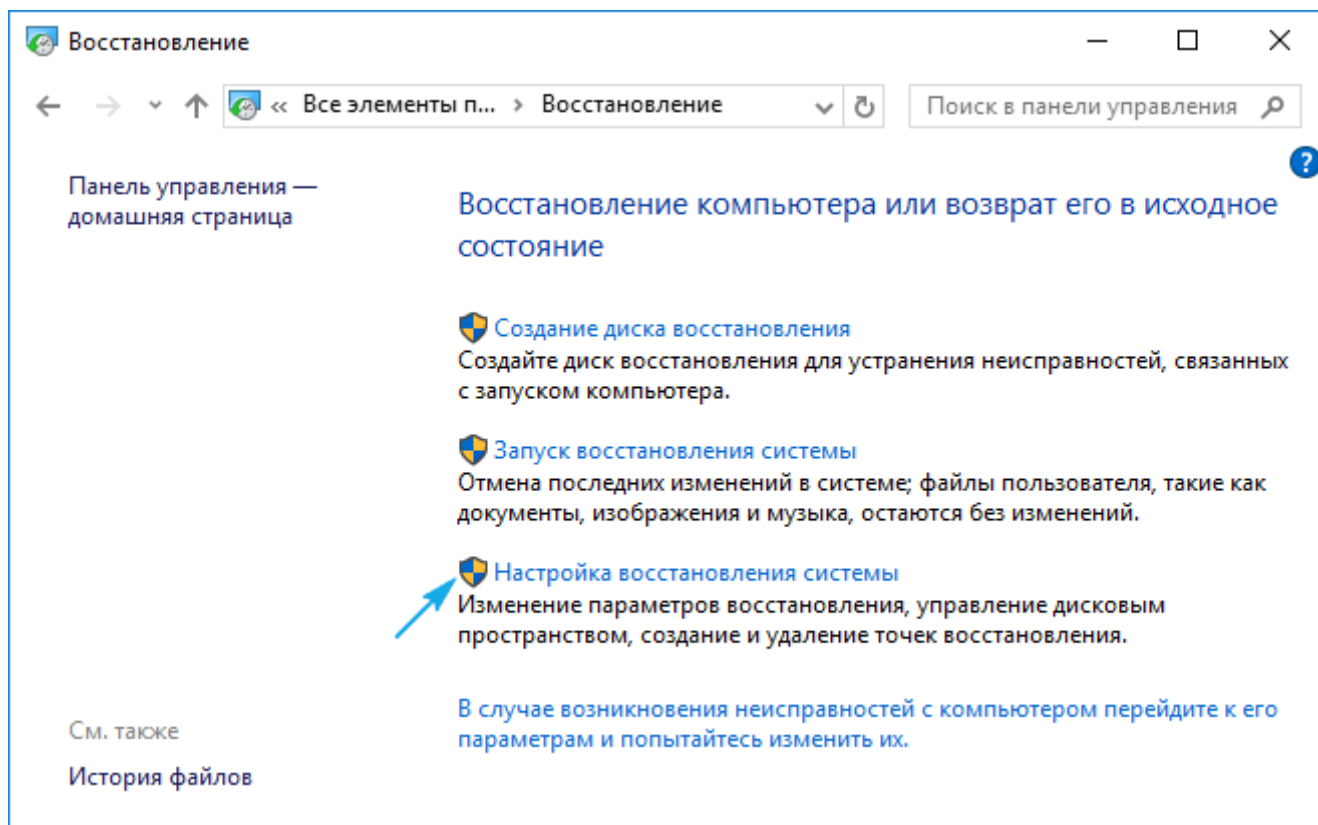
5. Определяемся с местом хранения снимка операционной системы и разделами, которые будут подвергаться резервированию (лучше всего указать съемный накопитель).



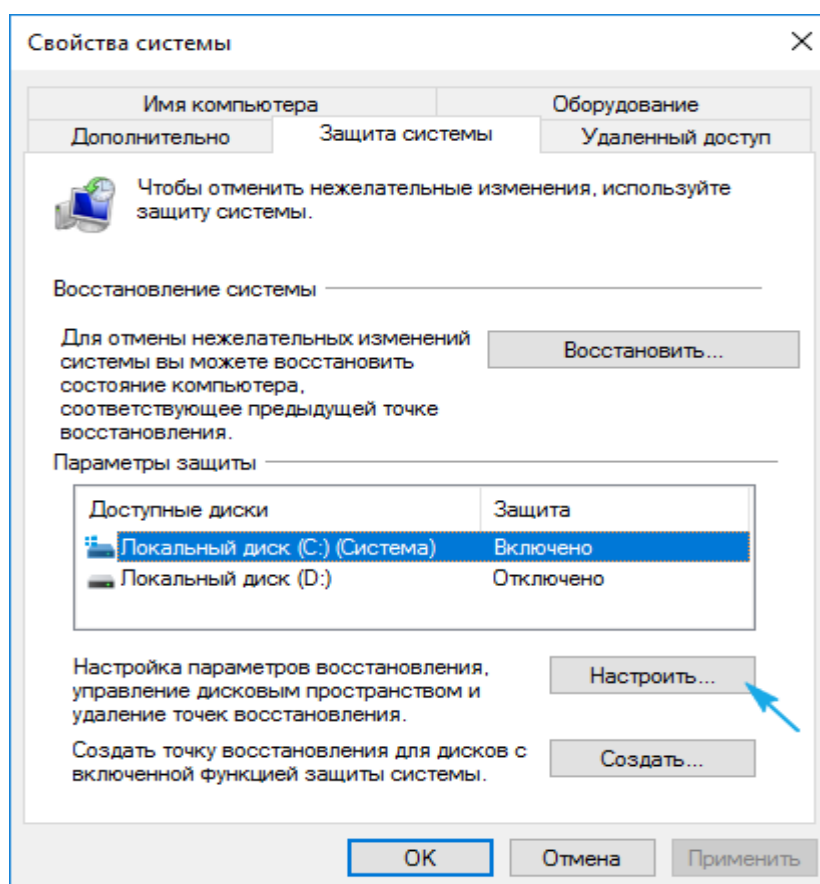
После завершения сжатия системных файлов и их перенесения на указанный цифровой носитель его можно будет использовать для быстрого возврата Windows 10 к запечатленному в образе состоянию. Для того чтобы запустить восстановление с образа необходимо выполнить загрузку компьютера с флешки, на которой файл хранится, или через инсталлятор Windows 10 («Диагностика» — «Расширенные параметры» — «Восстановление образа ОС»).

Точки отката Windows 10

С этой функцией нет никаких новшеств, все ее возможности работают, как в предыдущих версиях ОС. Она предоставляет шанс вернуть систему к одному из сохранившихся состояний через среду восстановления или в работающей операционной системе. Чтобы воспользоваться всеми преимуществами функции, она должна быть активированной. Проверить состояние можно через апплет Панели управления под именем «Восстановление». В окне жмем «Настройка восстановления системы».

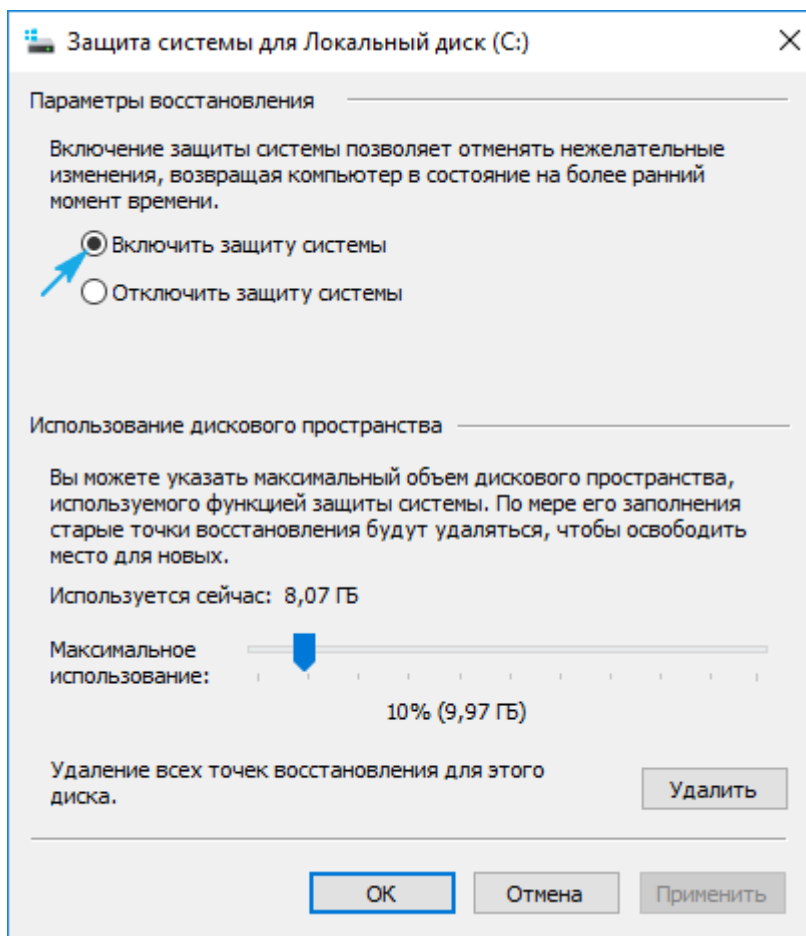


Для изменения параметров жмем «Настроить» и указываем выделяемое под хранение точек отката место на системном диске.



При использовании лицензионного образа эта функция активирована, но, если установили

Windows 10, скачанную с пиратских сайтов, возможно, автор сборки деактивировал эту функцию. Как включить восстановление системы? Выбираем системный раздел, жмем «Настроить» и перемещаем чекбокс к позиции «Включить защиту системы».



Обычно, точки отката формируются автоматически в случаях внесения со стороны пользователя или каких-либо приложений изменений, касающихся системных файлов, настроек, служб и параметров реестра. Также доступна возможность ручного создания точек восстановления. В окне «Свойства системы» нажимаем «Создать» и вводим название снимка, чтобы было проще идентифицировать его.

Для эксплуатации функции отката системы посредством одной из точек восстановления заходим в тот же апплет и жмем «Запуск восстановления системы». В случае когда Windows 10 не запускается, выполняем загрузку с диска восстановления или установочного дистрибутива и вызываем «Восстановление системы» через «Дополнительные параметры» в окне диагностики.

История файлов

Очередное новшество Windows 10, позволяющее делать и хранить резервные копии указанных файлов (зачастую текстовых документов и различных проектов) и извлекать из резерва нужную копию файла в случае необходимости.

Как можно увидеть, «десятка» обладает расширенным набором эффективных функций для возврата системы к работоспособному состоянию в любых случаях и без переустановки ОС. В

добавок ко всем перечисленным инструментам обязательно следует отнести функцию восстановления загрузчика средствами Windows 10.

Практическое занятие «Сбор информации об ошибках. Формирование отчетов об ошибках»

Цель: научиться собирать информацию об ошибках и формировать отчеты

Задание

Отчет об ошибках содержит следующую информацию:

Примечание: Для некоторых ошибок может быть получена не вся информация.

МЕТКА	Предопределенное название события.
ИД	Числовой идентификатор события.
Дата/Время	Дата и время события.
Порядковый номер	Уникальный номер события.
ИД системы	Идентификатор системного блока.
ИД узла	Мнемоническое имя системы.
Класс	Общий источник ошибки. Существуют следующие классы ошибок: N Аппаратного обеспечения. (При получении сообщения об ошибке аппаратного обеспечения обратитесь к руководству оператора системы за инструкциями по проведению диагностики отказавшего устройства или другого оборудования. Диагностическая программа определяет состояние устройства, проверяя устройство и анализируя связанные с ним записи протокола ошибок.) S Программного обеспечения. O Информационные сообщения. U Неопределенные (например, сбой сети).
Тип	Серьезность обнаруженной ошибки. Существует пять типов ошибок: PEND Устройство или компонент может стать недоступным. PERF Производительность устройства или компонента понизилась ниже допустимого уровня. PERM

Неисправимая ошибка. Этот тип относится к самым серьезным ошибкам и свидетельствует о неисправности устройства или модуля программного обеспечения. Ошибки всех типов, кроме PERM, обычно не означают неисправности, но записываются для анализа в диагностических программах.

TEMP

Ошибка, которая была исправлена после нескольких неудачных попыток. Этот тип ошибки также применяется для записи информационных сообщений, например, статистики передачи данных устройств DASD.

UNKN

Невозможно определить серьезность ошибки.

INFO

Запись протокола ошибок носит информационный характер и не свидетельствует об ошибке.

Имя ресурса	Имя ресурса, обнаружившего ошибку. В случае ошибки программного обеспечения, означает имя компонента программного обеспечения или программы. В случае ошибки аппаратного обеспечения - имя устройства или компонента системы. Это не означает, что компонент неисправен и требует замены. Это значение лишь определяет модуль диагностики, применяемый для анализа ошибки.
Класс ресурса	Общий класс ресурса, обнаружившего ошибку (например, класс устройства дисковый накопитель).
Тип ресурса	Тип ресурса, обнаружившего ошибку (например, тип устройства 355mb).
Код расположения	Путь к устройству. Может содержать до четырех полей, соответствующих корпусу, разъему, кабелю и порту.
VPD	Сведения о продукте. В этом поле может быть указана различная информация. Запись об устройстве в протоколе ошибок обычно содержит информацию о производителе устройства, серийном номере, Уровнях конструкторских изменений и уровнях ПЗУ.
Описание	Краткое описание ошибки.
Возможная причина	Список возможных источников ошибки.

Ошибки пользователя	Список возможных ошибок пользователя, вызвавших сбой. Примером таких ошибок являются неправильно вставленные диски или внешние устройства (такие как модемы и принтеры), питание которых отключено.
Рекомендуемые действия	Инструкции по устранению ошибок, вызванных пользователем.
Ошибка установки	Список возможных ошибок при установке и настройке, вызвавших сбой. Примерами такого типа ошибки являются несовместимость программного и аппаратного обеспечения, неправильное подключение кабелей или их отсоединение, а также неправильно настроенные системы.
Рекомендуемые действия	Инструкции по устранению ошибок, вызванных неправильной установкой.
Возможный сбой	<p>Список возможных неполадок программного и аппаратного обеспечения.</p> <p>Примечание: Раздел протокола ошибок "возможный сбой" обычно свидетельствует о неполадке программного обеспечения. Если же в протоколе есть записи об ошибке пользователя или установке, но нет записи о возможном сбое, то это обычно означает, что программное обеспечение не является причиной неполадки.</p> <p>Если вы считаете, что причиной является ошибка программного обеспечения или вам не удастся исправить ошибку пользователя или установки, сообщите о неполадке в отдел по обслуживанию программного обеспечения.</p>
Рекомендуемые действия	Инструкции по устранению сбоя. В случае ошибок аппаратного обеспечения список рекомендуемых действий содержит запись ВЫПОЛНИТЕ ПРОЦЕДУРЫ ЛОКАЛИЗАЦИИ НЕПОЛАДКИ . Это значит, что необходимо запустить диагностическую программу.
Подробные сведения	Уникальные для каждой записи протокола ошибок данные об ошибке, например, код ошибки устройства.

Некоторые ошибки можно исключить из отчета. Для просмотра ошибок, исключенных из отчета, введите команду:

```
errpt -t -F report=0 | pg
```

Если такие ошибки есть, включите в отчет все ошибки с помощью команды **errupdate**.

Некоторые ошибки могут не регистрироваться в протоколе. Для просмотра ошибок,

исключенных из протокола, введите команду:

```
errpt -t -F log=0 | pg
```

Если такие ошибки есть, включите регистрацию в протоколе для всех ошибок с помощью команды **errupdate**. Регистрация всех ошибок в протоколе необходима для воссоздания ошибки системы.

Примеры подробных отчетов об ошибках

Ниже приведен пример записей отчета об ошибках, созданного с помощью команды **errpt** -
a.

Класс ошибки **H** и тип ошибки **PERM** означают, что в системе была обнаружена ошибка устройства (драйвера адаптера SCSI), которую не удалось устранить.

С этим типом ошибки могут быть связаны данные диагностики.

Эта информация находится в конце сообщения об ошибке.

МЕТКА: SCSI_ERR1

ИД: 0502F666

Дата/Время: Jun 19 22:29:51

Порядковый номер: 95

ИД системы: 123456789012

ИД узла: host1

Класс: H

Тип: PERM

Имя ресурса: scsi0

Класс ресурса: adapter

Тип ресурса: hscsi

Расположение: 00-08

VPD:

Device Driver Level.....00

Diagnostic Level.....00

Displayable Message.....SCSI

EC Level.....C25928

FRU Number.....30F8834

Manufacturer.....IBM97F

Part Number.....59F4566

Serial Number.....00002849

ROS Level and ID.....24

Read/Write Register Ptr.....0120

Описание ADAPTER ERROR

Возможные причины

ADAPTER HARDWARE CABLE

CABLE TERMINATOR DEVICE

Возможные сбои ADAPTER

CABLE LOOSE OR DEFECTIVE

Рекомендуемые действия

PERFORM PROBLEM DETERMINATION PROCEDURES

CHECK CABLE AND ITS CONNECTIONS

Подробные сведения

SENSE DATA

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Порядковый номер протокола диагностики: 153

Проверенный ресурс: scsi0

Описание ресурса: SCSI I/O Controller

Расположение: 00-08

SRN: 889-191

Описание: Анализ

протокола ошибок указывает на неполадку аппаратного обеспечения.

Возможные FRU:

Шина SCSI FRU: нет 00-08

Вентилятор

SCSI2 FRU: 30F8834 00-08

Контроллер ввода-вывода SCSI

Класс ошибки **H** и тип ошибки **PEND** означают, что устройство (Token Ring) может в ближайшее время стать недоступным из-за большого количества ошибок, обнаруженных системой.

МЕТКА: TOK_ESERR

ИД: AF1621E8

Дата/Время: Jun 20 11:28:11

Порядковый номер: 17262

ИД системы: 123456789012

ИД узла: host1

Класс: H

Тип: PEND

Имя ресурса: TokenRing

Класс ресурса: tok0

Тип ресурса: Adapter

Расположение TokenRing

Описание

EXCESSIVE TOKEN-RING ERRORS

Возможные причины

TOKEN-RING FAULT DOMAIN

Возможные сбои TOKEN-RING FAULT DOMAIN

Рекомендуемые действия

REVIEW LINK CONFIGURATION DETAIL DATA

CONTACT TOKEN-RING ADMINISTRATOR RESPONSIBLE FOR THIS LAN

Подробные сведения

SENSE DATA

0ACA 0032 A440 0001 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 2080 0000 0000 0010 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 78CC 0000 0000 0005 C88F 0304 F4E0 0000 1000 5A4F 5685
1000 5A4F 5685 3030 3030 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000

Класс ошибки **S** и тип ошибки **PERM** означают, что в системе была обнаружена ошибка программного обеспечения, которую не удалось устранить.

МЕТКА: DSI_PROC

ИД: 20FAED7F

Дата/Время: Jun 28 23:40:14

Порядковый номер: 20136

ИД системы: 123456789012

ИД узла: 123456789012

Класс: S

Тип: PERM

Имя ресурса: SYSVMM

Описание

Data Storage Interrupt, Processor

Возможные причины

SOFTWARE PROGRAM

Возможные сбои SOFTWARE PROGRAM

Рекомендуемые действия

IF PROBLEM PERSISTS THEN DO THE FOLLOWING

CONTACT APPROPRIATE SERVICE REPRESENTATIVE

Подробные сведения

Data Storage Interrupt Status Register

4000 0000

Data Storage Interrupt Address Register

0000 9112

Segment Register, SEGREG

D000 1018

EXVAL

0000 0005

Класс ошибки **S** и тип ошибки **TEMP** означают, что в системе была обнаружена ошибка программного обеспечения. После нескольких попыток системе удалось устранить неполадку.

МЕТКА: SCSI_ERR6

ИД: 52DB7218

Дата/Время: Jun 28 23:21:11

Порядковый номер: 20114

ИД системы: 123456789012

ИД узла: host1

Класс: S

Тип: INFO

Имя ресурса: scsi0

Описание

SOFTWARE PROGRAM ERROR

Возможные причины

SOFTWARE PROGRAM

Возможные сбои SOFTWARE PROGRAM

Рекомендуемые действия

IF PROBLEM PERSISTS THEN DO THE FOLLOWING

CONTACT APPROPRIATE SERVICE REPRESENTATIVE

Подробные сведения

SENSE DATA

0000 0000 0000 0000 0000 0011 0000 0008 000E 0900 0000 0000 FFFF
FFFE 4000 1C1F 01A9 09C4 0000 000F 0000 0000 0000 0000 FFFF FFFF
0325 0018 0040 1500 0000 0000 0000 0000 0000 0000 0000 0000 0800
0000 0100 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000

Класс ошибки **O** означает информационное сообщение.

МЕТКА: OPMSG

ИД: AA8AB241

Дата/Время: Jul 16 03:02:02

Порядковый номер: 26042

ИД системы: 123456789012

ИД узла: host1

Класс: O

Тип: INFO

Имя ресурса: OPERATOR

Описание

OPERATOR NOTIFICATION

Ошибки пользователя

errlogger COMMAND

Рекомендуемые действия

REVIEW DETAILED DATA

Подробные сведения

MESSAGE FROM errlogger COMMAND

hdisk1: Анализ протокола ошибок указывает на неполадку аппаратного обеспечения.

Пример краткого отчета об ошибках

Ниже приведен пример краткого отчета об ошибках, созданного с помощью команды **errpt**.

Каждой записи об ошибке соответствует одна строка информации.

ERROR_

ИДЕНТИФИКАТОР СИСТЕМНОЕ_ВРЕМЯ Т КЛ ИМЯ_РЕСУРСА ОПИСАНИЕ_ОШИБКИ

192AC071 0101000070 I O errdemon Ведение протокола ошибок выключено

0E017ED1 0405131090 P H mem2 Сбой памяти

9DBCDFDEE 0101000070 I O errdemon Ведение протокола ошибок включено

038F2580 0405131090 U H scdisk0 НЕОПРЕДЕЛЕННАЯ ОШИБКА

AA8AB241 0405130990 I O OPERATOR ИЗВЕЩЕНИЕ ОПЕРАТОРА

Создание отчета об ошибках

Выполните следующие действия, чтобы создать отчет об ошибках программного обеспечения или неполадках аппаратного обеспечения.

1. Определите, включено ли ведение протокола ошибок. Для этого проверьте, содержит ли протокол ошибок записи:

2. `errpt -a`

Команда **errpt** создает отчет об ошибках из записей системного протокола ошибок.

Если протокол ошибок пуст, ведение протокола ошибок было отключено. Активизируйте средство ведения протокола ошибок с помощью следующей команды:

`/usr/lib/errdemon`

Примечание: Для запуска этой команды необходимы права доступа пользователя root.

Демон **errdemon** запускает ведение протокола ошибок. Если демон не работает, протокол ошибок не ведется.

3. Создайте отчет об ошибках с помощью команды **errpt**. Например, для просмотра всех ошибок дискового накопителя `hdisk1` введите команду:

4. `errpt -N hdisk1`
5. Создайте отчет об ошибках с помощью SMIT. Например, с помощью команды **smit errpt**:
6. `smit errpt`

Выберите **1**, чтобы направить отчет об ошибках в стандартный вывод, или **2**, чтобы отправить отчет на принтер.

Выберите **yes**, чтобы просматривать или распечатывать записи протокола ошибок по мере добавления, в противном случае выберите **no**.

Укажите нужное имя устройства в опции **Выбрать имена ресурсов** (например `hdisk1`).

Выберите **Do**.

Завершение ведения протокола ошибок

В данном разделе описано завершение работы средства ведения протокола ошибок. Как правило, нет необходимости отключать средство ведения протокола ошибок. Вместо этого следует удалить из протокола ошибок старые и ненужные записи. Инструкции по очистке протокола ошибок приведены в разделе Очистка протокола ошибок.

Средство ведения протокола ошибок следует отключать при установке или проверке нового программного или аппаратного обеспечения. В этом случае демон ведения протокола ошибок не будет отнимать время центрального процессора на регистрацию известных вам ошибок.

Примечание: Для запуска применяемой в этой процедуре команды у вас должны быть права доступа пользователя `root`.

Введите команду **errstop**, чтобы отключить ведение протокола ошибок:

`errstop`

Команда **errstop** завершает работу демона ведения протокола.

Очистка протокола ошибок

Этот раздел содержит информацию по удалению из протокола ошибок старых и ненужных записей. Обычно очистка протокола автоматически выполняется ежедневно с помощью команды **cron**.

Если эта процедура не выполняется автоматически, следует время от времени очищать протокол ошибок вручную, предварительно проверив его на наличие записей о серьезных неполадках.

Кроме того, можно удалить записи о конкретных ошибках. Например, после замены дискового накопителя можно удалить из протокола ошибок записи об ошибках старого дискового накопителя.

Для удаления всех записей протокола ошибок выполните одно из следующих действий:

- Вызовите команду **errclear-d**. Например, для удаления всех записей об ошибках программного обеспечения, введите команду:

- `errclear -d S 0`

Команда **errclear** удаляет из протокола ошибок записи, внесенные раньше определенного числа дней. В предыдущем для удаления всех записей указано значение 0.

- Введите команду **smit errclear**:
- `smit errclear`

Копирование протокола ошибок на дискету или магнитную ленту

Выполните следующие действия, чтобы скопировать протокол ошибок:

- С помощью команд **ls** и **backup** скопируйте протокол ошибок на дискету.

Вставьте отформатированную дискету в дисковод и введите команду:

- `ls /var/adm/ras/errlog | backup -ivp`
- Для копирования протокола ошибок на магнитную ленту вставьте магнитную

ленту в лентопротяжное устройство и введите команду:

- `ls /var/adm/ras/errlog | backup -ivpf/dev/rmt0`

ИЛИ

- С помощью команды **snap** соберите информацию о конфигурации системы в файл **tar** и скопируйте его на дискету. Вставьте отформатированную дискету в дисковод и введите команду:

Примечание: Для запуска команды **snap** у вас должны быть права доступа пользователя root.

```
snap -a -o
/dev/rfd0
```

В этом примере для сбора всей информации о конфигурации системы в команде **snap** указан флаг **-a**. Флаг **-o** позволяет скопировать сжатый файл **tar** на указанное устройство. `/dev/rfd0` указывает дисковод.

Введите следующую команду, чтобы собрать всю информацию о конфигурации в файле **tar** и скопировать его на магнитную ленту:

```
snap -a -o /dev/rmt0
/dev/rmt0 указывает лентопротяжное устройство.
```

Практическое занятие «Выявление и устранение ошибок программного кода информационных систем»

Цель: выявлять и устранять ошибки программного кода информационных систем

Задание

Системы моделирования обладают множеством достоинств. Обычно в их составе имеются отладчики и средства вывода информации на печать, однако системы моделирования это всего лишь имитаторы. Отлаживаемая программа может успешно исполняться в системе моделирования и быть полностью неработоспособной в реальных условиях. Так что системы моделирования это лишь частичное решение. Ошибки программного обеспечения вполне могут пройти мимо системы моделирования и всплыть в реальном оборудовании.

Именно в этом и скрыта главная проблема: как показано на рис. 1, исправление ошибок, которые выявляются не на этапе тестирования, а в процессе использования, обходится значительно дороже. Если ошибка найдена в программе для не-встраиваемых систем, то можно выпустить обновленную версию программы с исправлениями, стоимость таких обновлений, как правило, сравнительно невысокая. Если же ошибка найдена во встроенной системе, то для ее исправления необходим возврат и модификация самих устройств с этой системой. Стоимость такого возврата может достигать астрономических величин, и стать причиной разорения компаний.

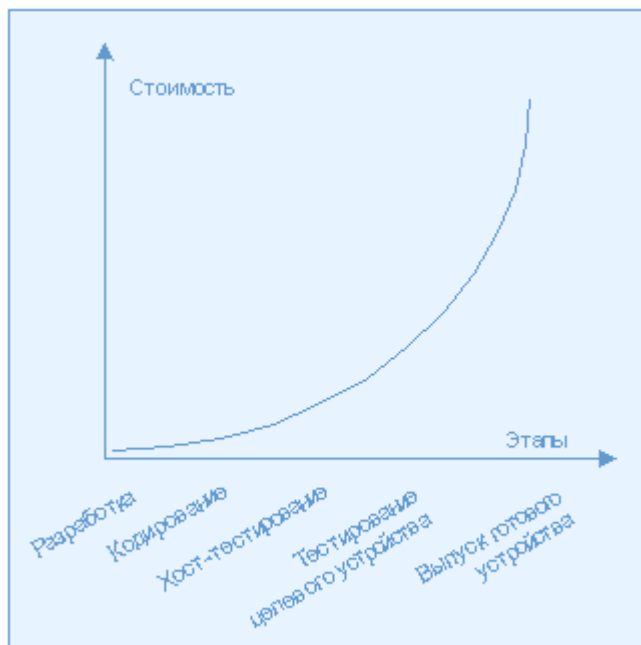


Рис. 1. Стоимость устранения ошибок во встраиваемых системах

На мой взгляд, сроки и затраты на выявление и устранение ошибок для встраиваемых систем приблизительно удваиваются (из-за описанных выше трудностей). В свете таких немислимых затрат любой метод, который изначально будет препятствовать появлению ошибок, имеет неоценимое значение. К счастью для разработчиков встраиваемых систем, для предотвращения ошибок можно использовать некоторые из новых технологий программной разработки. Наиболее рекомендуемые две из них: стандарты программирования и блочное тестирование.

Правда, оба этих метода сегодня не столько применяются, сколько прославляются. Практически каждый разработчик программного обеспечения согласен с их высокой ценностью, но пользуются ими единицы. Подобная непоследовательность объясняется в большинстве случаев двумя причинами. Прежде всего, многие считают следование стандартам программирования и блочное тестирование весьма утомительным делом. Учитывая, сколько времени и сил эти подходы позволяют сэкономить в будущем, разработчикам следовало бы немножко потерпеть и избежать огромных трудозатрат (и возможного отказа от проекта) впоследствии.

Разработчикам систем реального времени еще труднее они в дополнение ко всему должны решать проблемы, связанные с соблюдением различных временных зависимостей. В конце статьи мы рассмотрим трудности, возникающие при отладке систем реального времени, и познакомимся с некоторыми методами отладки, которые рассчитаны на преодоление этих трудностей и которые также могут быть использованы при разработке любого программного обеспечения.

Способы отладки программ Отладка программ заключается в проверке правильности работы программы и аппаратуры. Программа, не содержащая синтаксических ошибок тем не менее может содержать логические ошибки, не позволяющие программе выполнять заложенные в ней функции. Логические ошибки могут быть связаны с алгоритмом программы или с неправильным пониманием работы аппаратуры, подключённой к портам микроконтроллера. Встроенный в состав интегрированной среды программирования отладчик позволяет отладить те участки кода программы, которые не зависят от работы аппаратуры, не входящей в состав микросхемы микроконтроллера. Обычно это относится к вычислению математических выражений или преобразованию форматов представления данных. Для отладки программ обычно применяют три способа:

1. Пошаговая отладка программ с заходом в подпрограммы;

2. Пошаговая отладка программ с выполнением подпрограммы как одного оператора;
3. Выполнение программы до точки останова.

Пошаговая отладка программ заключается в том, что выполняется один оператор программы и, затем контролируются те переменные, на которые должен был воздействовать данный оператор. Если в программе имеются уже отлаженные подпрограммы, то подпрограмму можно рассматривать, как один оператор программы и воспользоваться вторым способом отладки программ. Если в программе существует достаточно большой участок программы, уже отлаженный ранее, то его можно выполнить, не контролируя переменные, на которые он воздействует. Использование точек останова позволяет пропускать уже отлаженную часть программы. Точка останова устанавливается в местах, где необходимо проверить содержимое переменных или просто проконтролировать, передаётся ли управление данному оператору. Практически во всех отладчиках поддерживается это свойство (а также выполнение программы до курсора и выход из подпрограммы). Затем отладка программы продолжается в пошаговом режиме с контролем локальных и глобальных переменных, а также внутренних регистров микроконтроллера и напряжений на выводах этой микросхемы.

Следуйте стандартам программирования!

Самый лучший способ повысить качество ПО это стараться не допускать ошибок в процессе ввода исходного текста.

Первый шаг на пути предотвращения ошибок это осознание того, что ошибки действительно можно предотвратить. Больше всего препятствует контролю над ошибками распространенное убеждение в том, что ошибки неизбежны. Это заблуждение! Ошибки сами по себе не появляются их вносит в текст разработчик. Человеку свойственно ошибаться, так что даже самые лучшие программисты время от времени допускают ошибки, если у них есть такая возможность. Поэтому чтобы уменьшить число ошибок, надо сократить возможности их появления. Один из лучших способов здесь следование стандартам программирования, что ликвидирует благодатную почву для возникновения ошибок на первых этапах.

Стандарты программирования — это специфичные для языка "правила", которые, если их соблюдать, значительно снижают вероятность внесения ошибок в процессе разработки приложения. Следовать стандартам программирования нужно на этапе

написания программ, до их переноса в целевые платформы, при этом стандартизация должна существовать для всех языков. Поскольку большая часть разработчиков встраиваемых систем пользуется языком C, больше внимания уделим именно стандартам программирования на C, хотя такие же стандарты существуют и для других языков, включая C++ и Java.

Как правило, стандарты программирования делятся на две категории:

- отраслевые стандарты программирования: правила, принятые всеми программистами на данном языке (например, запрет входа в цикл не через его заголовок).
- специальные стандарты программирования: правила, соблюдаемые конкретной группой разработчиков, в рамках конкретного проекта, или даже единственным программистом. Существует три типа специальных стандартов, которыми может воспользоваться разработчик встраиваемой программной системы: внутренние стандарты, персональные стандарты и стандарты, определяемые целевой платформой.

Внутренние стандарты программирования — это правила, которые специфичны для вашей организации или группы разработчиков. Так, уникальные для организации правила присвоения имен это пример внутренних стандартов программирования.

Персональные стандарты — это правила, которые помогут вам избежать ваших наиболее частых ошибок. Каждый раз при появлении какой-либо ошибки программист должен проанализировать причину ее появления и выработать собственное правило, препятствующее повторному ее возникновению. Например, если в операторе условия вы часто пишете знак присваивания вместо знака проверки на равенство (т.е. "if (a=b)" вместо "if (a==b)"), то вам необходимо создать для себя следующий стандарт: "Остерегаться применения знака присваивания в операторе проверки условия".

Стандарты, определяемые целевой платформой, это правила, нарушение которых в данной платформе может привести к появлению определенных проблем. Например, такими стандартами могут быть ограничения на использование памяти или размер переменных, налагаемые целевой платформой.

Чтобы лучше разобраться в том, что такое стандарты программирования и как они работают, познакомимся с ними на конкретных примерах. Рассмотрим следующую запись на языке C:

```
char *substring (char string[80], int start_pos, int length)
{
.
.
.
}
```

Здесь размер одномерного массива декларируется в списке аргументов функции. Это опасная конструкция, поскольку в языке С аргумент-массив передается как указатель на его первый элемент, и в разных обращениях к функции в числе ее фактических аргументов могут указываться массивы с разной размерностью. Создав такую конструкцию, вы предполагаете пользоваться буфером фиксированного размера на 80 элементов, считая, что именно такой буфер и будет передаваться функции, а это может привести к разрушению памяти. Если бы автор этого оператора следовал стандарту программирования "не объявлять размер одномерного массива в числе аргументов функции" (взятому из набора стандартов программирования на языке С одной из ведущих телекоммуникационных компаний), то этот текст выглядел бы следующим образом и проблем с разрушением памяти удалось бы избежать:

```
char *substring (char string[], int start_pos, int length)
{
.
.
.
}
```

Стандарты программирования позволяют также избегать проблем, которые до момента портирования кода на другую платформу могут не проявляться. Например, следующий кусок кода будет исправно работать на одних платформах и порождать ошибки после переноса его на другие платформы:

```
#include

void test(char c) {

if( a <= c && c <= z ) { // Неправильно
```

```

}
if(islower(c)) { // Правильно
}
while ( A <= c && c <= Z ) { // Неправильно
}
while (isupper(c)) { // Правильно
}
}

```

Проблемы портации могут быть связаны с символьными тестами, в которых не используется функции `ctype.h` (`isalnum`, `isalpha`, `isctrl`, `isdigit`, `isgraph`, `islower`, `isprint`, `ispunct`, `isspace`, `isupper`, `isxdigit`, `tolower`, `toupper`). Функции `ctype.h` для символьной проверки и преобразования прописных букв в строчные и наоборот работают с самыми разными наборами символов, обычно очень эффективны и гарантируют международную применимость программного продукта.

Лучший способ внедрить эти и другие стандарты программирования — это обеспечить их автоматическое применение в составе какой-либо технологии программирования, вместе с набором целенаправленных отраслевых стандартов и механизмами создания и поддержки стандартов программирования, ориентированных на конкретную систему. При выборе подобной технологии необходимо сначала найти ответы на вопросы, среди которых следующие:

- Применима ли она к данной программе и/или компилятору?
- Содержит ли она набор отраслевых стандартов программирования?
- Позволяет ли она создавать и поддерживать специальные стандарты программирования (включая стандарты, определяемые целевой платформой)?
- Легко ли структурировать отчеты в соответствии с вашими групповыми и проектными приоритетами?
- Насколько легко она интегрируется в существующий процесс разработки?

Блочное тестирование

Зачастую, слыша о блочном тестировании, разработчики воспринимают его как синоним модульного тестирования. Другими словами, проверяя отдельный модуль или подпрограмму более крупной программной единицы, разработчики считают, что выполняют блочное тестирование. Конечно, модульное тестирование имеет очень большое значение и, безусловно, должно проводиться, но это не тот метод, на котором я бы хотел остановиться. Говоря о "блочном тестировании", я имею в виду тестирование на еще более низком уровне: тестирование самых минимально возможных программных единиц, из которых состоит прикладная программа, всё ещё находясь в инструментальной среде (хост-системе) в случае языка С, это будут функции, которые проверяются сразу же после их компиляции.

Блочное тестирование значительно повышает качество программного обеспечения и эффективность процесса разработки. При тестировании на уровне объектов вы гораздо ближе к этой методике и обладаете гораздо большими возможностями построения входных наборов, выявляющих ошибки со стопроцентным покрытием (рис. 2). Кроме того, тестируя блок кода сразу после того, как он был написан, вы тем самым избегаете необходимости "продираться" через наложения ошибок, чтобы найти и исправить единственную исходную в данном случае вы сразу ее устраняете, и вся проблема решена. Это существенно ускоряет и облегчает процесс разработки, поскольку на поиск и устранение ошибок тратится значительно меньше материальных и временных ресурсов.

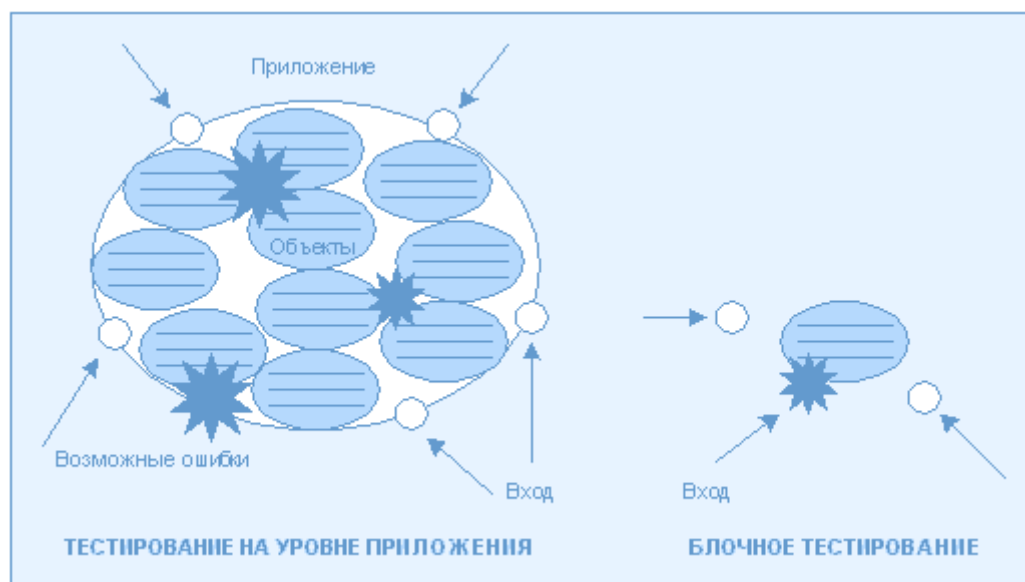


Рис. 2. Простота нахождения ошибок при блочном тестировании

Блочное тестирование можно разделить минимум на два отдельных процесса. Первый это тестирование "черного ящика", или процесс определения функциональных проблем. На уровне отдельных блоков тестирование "черного ящика" заключается в

проверке функциональных характеристик посредством определения степени соответствия параметров открытого интерфейса функции ее спецификации; подобная проверка выполняется без учета способов реализации. Результатом тестирования "черного ящика" на блочном уровне является уверенность в том, что данная функция ведет себя в полном соответствии с определением и что незначительная функциональная ошибка не приведет к лавине трудноразрешимых проблем.

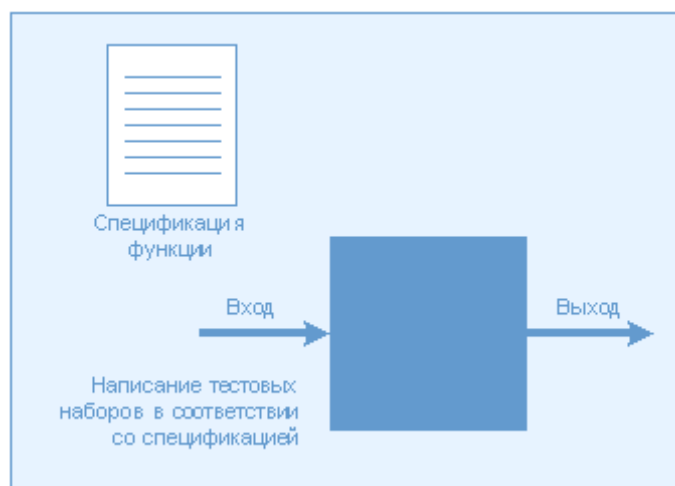


Рис. 3. Тестирование "черного ящика"

Второй процесс называется тестированием "белого ящика" и предназначен для выявления конструктивных недостатков. На уровне отдельных блоков проверяется, не произойдет ли крах всей программы при передаче в функцию неожиданных ею параметров. Этот вид тестирования должен проводиться специалистом, имеющим полное представление о способе реализации проверяемой функции. После такой проверки можно быть уверенным в том, что приводящих к краху системы ошибок нет и что функция будет устойчиво работать в любых условиях (т.е. выдавать предсказуемые результаты даже при вводе непредвиденных входных параметров).

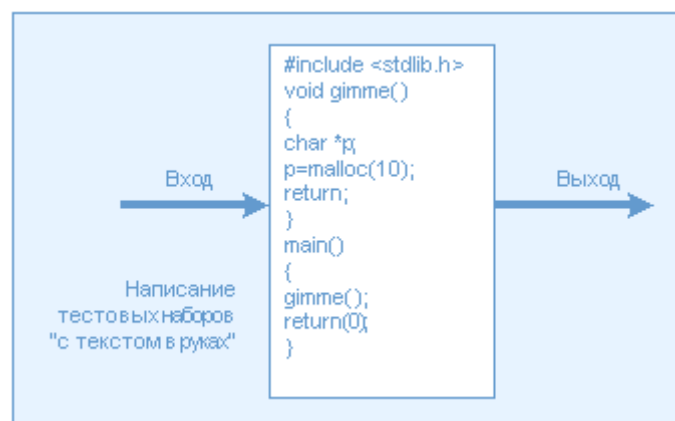


Рис. 4. Тестирование "белого ящика"

Оба вышеописанных процесса могут служить основой третьего, регрессивного тестирования. Сохранив тестовые наборы для "черного" и "белого" ящиков, можно использовать их для регрессивного тестирования на уровне блоков и контролировать целостность кода по мере того, как вы его модифицируете. Концепция регрессивного тестирования на этом уровне является новым оригинальным подходом. При выполнении регрессивного тестирования на уровне блоков можно сразу же после изменения вами текста определять, не появились ли новые проблемы, и устранять их немедленно после возникновения, тем самым препятствуя распространению ошибки по вышележащим уровням.

Главная проблема, связанная с блочным тестированием, заключается в том, что если не пользоваться технологиями автоматического блочного тестирования, проводить его трудно, утомительно и слишком долго. Рассмотрим вкратце причины, по которым включать неавтоматизированное блочное тестирование в сегодняшние процессы разработки трудно (если вообще возможно).

Первый этап блочного тестирования ПО встраиваемых систем заключается в создании такой среды, которая позволит запускать и тестировать интересующую функцию в хост-системе. Это требует выполнения следующих двух действий:

- разработка программного текста, который будет запускать функцию,
- написание фиктивных модулей, которые будут возвращать результаты вместо внешних ресурсов, к которым обращается функция и которые в текущий момент отсутствуют или недоступны.

Второй этап разработка тестовых наборов. Для полноты охвата конструктивных и функциональных особенностей функции необходимо создавать тестовые наборы двух типов: для "черного ящика" и для "белого ящика".

Основой разработки тестовых наборов для "черного ящика" должна стать спецификация функции. В частности, для каждой записи в спецификации должен быть создан хотя бы один тестовый набор, при этом желательно, чтобы эти наборы учитывали указанные в спецификации граничные условия. Проверять того, что некоторые входные параметры приводят к ожидаемым результатам, недостаточно; необходимо определить диапазон взаимосвязей входов и выходов, который позволит сделать вывод о корректной реализации указанных функциональных характеристик, и затем создавать тестовые наборы, полностью покрывающие данный диапазон. Можно также тестировать не указанные в спецификации и ошибочные условия.

Цель тестовых наборов для "белого ящика" обнаружить все скрытые дефекты путем всестороннего тестирования функции разнообразными входными параметрами. Эти наборы должны обладать следующими возможностями:

- обеспечивать максимально возможное (100%) покрытие функции: как уже говорилось, такая степень покрытия на уровне блоков возможна, поскольку создавать наборы для тестирования каждой характеристики функции вне приложения гораздо проще (стоцентное покрытие во всех случаях невозможно, но это цель, к которой надо стремиться);
- выявлять условия краха функции.

Следует заметить, что самостоятельно создание подобных наборов, не владея технологиями их построения, невероятно тяжелое занятие. Чтобы создать эффективные тестовые наборы для "белого ящика", необходимо сначала получить полное представление о внутренней структуре функции, написать наборы, обеспечивающие максимальное покрытие функции, и найти совокупность входов, приводящих к отказу функции. Получить спектр покрытия, необходимый для высокоэффективного тестирования "белого ящика", возможно лишь при исследовании значительного числа путей прохода по функции. Например, в обычной программе, состоящей из 10000 операторов, имеется приблизительно сто миллионов возможных путей прохода; вручную создать тестовые наборы для проверки всех этих путей невозможно.

После создания тестовых наборов необходимо провести тестирование функции в полном объеме и проанализировать результаты с целью выявления причин ошибок, крахов и слабых мест. Необходимо иметь способ прогона всех тестовых наборов и быстрого определения, какие из них приводят к возникновению проблем. Необходимо также иметь инструмент измерения степени покрытия для оценки полноты тестирования функции и определения необходимости в дополнительных тестовых наборах.

При любых изменениях функции следует проводить регрессивное тестирование, чтобы убедиться в отсутствии новых и/или устранении предыдущих ошибок. Включение блочного регрессивного тестирования в процесс разработки позволит защититься от многих ошибок они будут обнаружены сразу же после возникновения и не смогут стать причинами распространения ошибок в приложении.

Регрессивное тестирование можно проводить двумя способами. Первый заключается в том, что разработчик или испытатель анализирует каждый тестовый набор и определяет, на работе которого из них может сказаться измененный код. Этот подход

характеризуется экономией машинного времени за счет работы, проводимой человеком. Второй, более эффективный, заключается в автоматическом прогоне на компьютере всех тестовых наборов после каждого изменения текста. Данный подход гарантирует большую эффективность труда разработчика, поскольку он не должен тратить время на анализ всей совокупности тестовых наборов, для того чтобы определить, какие наборы следует прогонять, а какие нет.

Если вы сможете автоматизировать процесс блочного тестирования, то не только повысите качество тестирования, но и высвободите для себя значительно больше временных и материальных ресурсов, чем уйдет на этот процесс. Если вы пишете программы на языке С, то для автоматизации блочного тестирования можете воспользоваться существующими технологиями. Чем больше процессов вы сможете автоматизировать, тем больше пользы вы получите.

При выборе технологии блочного тестирования сначала следует ответить на следующие вопросы:

- Подходит ли эта технология для вашего текста и/или компилятора?
- Может ли она автоматически создавать тестовые схемы?
- Может ли она автоматически генерировать тестовые наборы?
- Позволяет ли она вводить создаваемые пользователем тестовые наборы и фиктивные модули?
- Автоматизировано ли регрессивное тестирование?
- Имеется ли в ее составе технология или связь с технологией автоматического распознавания ошибки в процессе прогона?

Средства отладки, не меняющие режим работы программ

Из-за того, что операционные системы реального времени должны выполнять определенные задачи в условиях заранее определенных временных ограничений, временные соотношения превращаются в важнейший параметр, который разработчики должны учитывать при установке тестового ПО. Обычно в процессе исполнения программ возникает множество различных прерываний, и чрезвычайно необходимо, чтобы в момент возникновения прерывания приложение реагировало корректно. Ситуация еще более усложняется, когда несколько прерываний возникает сразу или когда в системе

исполняется несколько приложений с несколькими взаимодействующими друг с другом тредами. По сути, это приложения с несколькими одновременными путями исполнения различные кодовые последовательности как бы исполняются в одно и то же время, даже если в системе всего один центральный процессор. Интересно заметить, что если бы эти приложения исполнялись в нескольких процессорах, то различные треды на практике были бы загружены в разные процессоры.

Если возникающие в приложениях реального времени ошибки проявляются во взаимодействиях между программой и прерываниями, то они будут в значительной мере чувствительны ко времени. В этом случае критически важно регистрировать порядок возникновения ошибок, поскольку это позволит разобраться в причинах и следствиях каждой ошибки. В этом как раз и кроется главная проблема отладки систем реального времени: существует достаточное количество трудновывяемых ошибок, которые проявляются только при определенных временных соотношениях.

Эта проблема осложняется тем, что подобные ошибки не так-то просто воспроизводятся. Очень трудно воссоздать ситуацию с такими же временными соотношениями, что и приведшие к возникновению ошибки в реальной программе. Механизм отладки таких приложений должен быть максимально возможно щадящим. Любое вмешательство в ход исполнения программ может привести к изменению ее временных характеристик и отсутствию условий возникновения ошибок. Конечно, создание условий, при которых ошибки не возникают, это хорошо, но в данном случае это является препятствием отладке программы.

Теоретической основой проблемы отладки систем реального времени может послужить известный всем из курса физики принцип неопределенности немецкого физика Вернера Гейзенберга, согласно которому одновременно определить скорость и местоположение движущейся частицы невозможно. Гейзенберг считал, что, определяя координаты частицы, экспериментатор тем самым изменяет её местоположение, что не позволяет определить её координаты точно. Операция измерения влияет на измеряемый объект и искажает результаты измерения. Принцип неопределенности это одна из аксиом квантовой механики.

Применительно же к нашей теме, этот принцип означает, что отладка системы требует сбора информации о ее состоянии. Однако сбор информации о состоянии системы меняет ее временные характеристики и существенно затрудняет надежное воспроизведение условий возникновения ошибки.

Таким образом, суть этой проблемы в том, что нужно найти способ обнаружения ошибок реального времени и анализа поведения программы без влияния на существующие временные соотношения. Наверное, вашим первым порывом было бы обращение к отладчику, но отладчики, как правило, прерывают исполнение программы и, соответственно, изменяют ее временные характеристики. Малопригодны и системы моделирования, поскольку они не могут воссоздать временные характеристики реальных технических средств. Еще никто не создал такую систему моделирования, которая могла бы смоделировать режим реального времени; временные параметры можно определить, только загрузив программу в само железо.

Последнее требует наличия специального механизма для упрощенной регистрации состояния системы. Один из возможных и подходящих механизмов запись информации в оперативную память, поскольку такая операция выполняется чрезвычайно быстро. Один из способов применения этого механизма организация где-нибудь в памяти специального буфера и использование в вашей программе указателя на этот буфер. Указатель всегда ссылается на начало буфера. В программу вставляются операции записи в ячейку, определяемую указателем. После каждой операции записи значение указателя меняется соответствующим образом. Иногда полезно пользоваться кольцевым буфером (т.е. когда после записи в последнюю ячейку буфера указатель начинает показывать на начало буфера), что позволяет отслеживать ситуации, приводящие к возникновению проблемы. Необходимо при этом предусмотреть способ сохранения содержимого буфера после нормального или аварийного завершения программы, чтобы впоследствии иметь к нему доступ и проводить так называемую "посмертную отладку". Способ реализации зависит от аппаратных средств, обычно это можно сделать, если не выполнять повторную инициализацию (reset) оборудования.

Теперь вам нужен механизм чтения этой памяти. Здесь можно использовать и отладчик, и другие средства извлечения информации из оперативной памяти. В частности, можно написать простенькую программу, которая будет пересылать эти данные в файл или на принтер. Каким бы средством вы не пользовались, конечным этапом, вероятнее всего, будет ручной анализ содержимого буфера. Если ваш буфер кольцевой, то вам необходимо иметь точные сведения о значении указателя; события, которые стали началом последовательности, будут непосредственно перед указателем, события, которые возникли непосредственно перед крахом, будут сразу же после указателя.



Рис. 5. Последовательность событий в кольцевом буфере

Теперь ваша главная задача попытаться разобраться в последовательности данных, записанных в буфере. Эта задача аналогична исследованию причин катастрофы самолета по показаниям приборов, зарегистрированных "черным ящиком" самолета. Анализ характеристик программы в этом случае проводится после свершившегося события, что, естественно, гораздо меньше влияет на ее исполнение, чем контроль в течение работы.

Иногда бывает очень трудно восстановить приведшие к краху события, и четкого понятия о моменте возникновения ошибки нет. Только на выяснение причины ошибки могут уходить многие месяцы. В таких случаях для поиска ошибочного оператора можно воспользоваться логарифмическим методом отладки. В разных местах отлаживаемого кода расставляются маркеры (например, операторы типа `exit`), а перед ними операторы записи в память. Затем запускаете программу и ожидаете момента краха. В случае краха вы знаете, между какими маркерами он произошел. Этот метод позволяет выявлять и проблемы согласования по времени, поскольку он позволяет находить сегменты кода, в которых возникают нарушения временных соотношений.

Ещё одно решение это применение в качестве технологии отладки так называемых брандмауэров. Брандмауэр это точка в логическом потоке программы, в которой доказывается справедливость предположений, на которые опирается последующий код. Проверка этих предположений отличается от обычного контроля ошибок. Срабатывание брандмауэра представляет собой сигнал разработчику о том, что внутреннее состояние системы неустойчиво. Это может произойти, например, если ожидающая строго положительного аргумента функция получает нулевую или отрицательную величину. Неискушённым разработчикам большинство брандмауэров кажутся тривиальными и ненужными. Однако опыт разработки крупных проектов показывает, что по мере развития и совершенствования программных систем неявные предположения в отношении среды исполнения нарушаются все чаще и чаще. Во многих случаях даже сам автор затрудняется сформулировать, что представляют собой надлежащие условия исполнения того или иного участка кода.

Реализуемые внутри встраиваемых систем брандмауэры нуждаются в специальных средствах связи для передачи сообщений во внешний мир; обсуждение способа установления таких каналов передачи выходит за рамки настоящей статьи.

Заключение

Рассмотренные выше методы предотвращения и обнаружения ошибок, а также технологии отладки могут значительно повысить качество программного обеспечения встраиваемых систем и уменьшить затрачиваемые на проведение отладки материальные и временные ресурсы. Вышеупомянутые методы без особого труда могут быть использованы в разработке самых разных проектов программного обеспечения встраиваемых систем, причем накопленный опыт полностью сохраняет свою ценность и при реализации иных проектов и целевых технологий. Кроме того, они позволяют гарантировать простоту сопровождения, модификации и портации созданных программ в устройства новых типов. И говоря коротко, рассматриваемые методы дают возможность не только совершенствовать существующие встроенные приложения и процессы разработки, но и гарантировать, что с распространением новых встраиваемых устройств у вас уже будет накоплен опыт, необходимый для разработки высокоэффективных приложений для этих технологий причем вовремя и в соответствии с выделенным бюджетом.

Практическое занятие «Выполнение обслуживания информационной системе в соответствии с пользовательской документацией»

Цель: выполнить обслуживание информационной системы в соответствии с пользовательской документацией

Задание

Документация информационного обеспечения АСУ предназначена для описания проектных решений по информационному обеспечению в документах:

- описание информационного обеспечения АСУ;
- описание организации информационной базы;
- описание системы классификации и кодирования;
- чертеж формы документа (видеограммы);
- описание массива информации;
- перечень входных сигналов и данных;
- перечень выходных сигналов (документов);
- описание технологического процесса обработки данных.

1.2. При разработке документов на части АСУ содержание разделов каждого документа ограничивают рамками соответствующей части.

1.3. В зависимости от назначения и специфических особенностей создаваемых АСУ допускается включать в документы дополнительные разделы и сведения, требования к содержанию которых не установлены настоящим стандартом.

1.4. Отсутствие проектных решений по разделу документа фиксируют в соответствующем разделе с необходимыми пояснениями.

ТРЕБОВАНИЯ К СОДЕРЖАНИЮ ДОКУМЕНТОВ

Описание информационного обеспечения АСУ

2.1.1. Документ должен состоять из следующих разделов:

- принципы организации информационного обеспечения;
- организация сбора и передачи информации;
- построение системы классификации и кодирования;
- организация внутримашинной информационной базы;
- организация внешнемашинной информационной базы.

2.1.2. Требования к содержанию разделов

2.1.2.1. В разделе «Принципы организации информационного обеспечения» должны быть приведены:

- состав, структура и принципы организации информационного обеспечения;
- обоснование выбора носителей данных и принципы распределения информации по типам данных;
- описание принятых видов и методов контроля в маршрутах обработки данных при создании и функционировании внешнимашинной и внутримашинной информационных баз с указанием требований, на соответствие которым проводят контроль;
- описание решений, обеспечивающих информационную совместимость АСУ с другими связанными с ней системами управления по источникам, потребителям информации, по сопряжению применяемых классификаторов (при необходимости), по использованию в АСУ унифицированных систем документации.

2.1.2.2. В разделе «Организация сбора и передачи информации» должны быть приведены перечни источников, носителей информации, оценка интенсивности и объема информации, описание общих требований к организации сбора и передачи информации.

2.1.2.3. В разделе «Построение системы классификации и кодирования» должны быть приведены:

- описание принятых систем классификации объектов;
- методы кодирования объектов классификации;
- перечень применяемых общесоюзных, отраслевых и других зарегистрированных классификаторов.

2.1.2.4. Раздел «Организация внутримашинной информационной базы» должен содержать описание принципов построения базы, характеристики ее состава и объема, структуры базы на уровне баз данных с описанием характера взаимосвязей баз данных и указанием функций АСУ, при реализации которых используют каждую базу данных, характеристики данных, содержащихся в каждой базе данных.

2.1.2.5. Раздел «Организация внешнимашинной информационной базы» должен содержать характеристики состава и объема, принципы построения базы, в том числе основные положения по организации и обслуживанию фонда нормативно-справочной информации во взаимосвязи с автоматизированными функциями управления.

2.1.2.6. В приложениях следует приводить справочные и другие вспомогательные материалы, и сведения (систематизированный перечень наименований структурных единиц информации с присвоенными им обозначениями и описаниями их сущности).

Описание системы классификации и кодирования

Документ должен содержать по каждому классифицируемому объекту описание метода кодирования, структуру и длину кода, указание о системе классификации и другие сведения по усмотрению разработчика.

Чертеж формы документа (видеограммы)

В документе должно быть приведено изображения формы документа или видеограммы в соответствии с требованиями государственных стандартов унифицированной системы документации и необходимые пояснения.

Описание массива информации

Документ должен содержать:

- наименование массива;
- обозначение массива;
- наименование носителя данных;
- перечень реквизитов в порядке их следования в записях массива с указанием по каждому реквизиту: обозначения алфавита, длины в знаках, диапазона изменения (при необходимости), логических и семантических связей с другими реквизитами данной записи и другими записями массива;
- оценку объема массива;
- другие характеристики массива (при необходимости).

Примечание. Если массив состоит из записей различных типов, то для записи каждого типа приводят все характеристики, перечисленные выше.

Техническая документация

При создании программы, одного лишь кода, как правило, недостаточно. Должен быть предоставлен некоторый текст, описывающий различные аспекты того, что именно делает код. Такая документация часто включается непосредственно в исходный код или предоставляется вместе с ним.

Подобная документация имеет сильно выраженный технический характер и в основном используется для определения и описания API, структур данных и алгоритмов.

Часто при составлении технической документации используются автоматизированные средства — генераторы документации, такие как Doxygen, javadoc, NDoc и другие. Они получают информацию из специальным образом оформленных комментариев в исходном коде, и создают справочные руководства в каком-либо формате, например, в виде текста или HTML.

Использование генераторов документации и документирующих комментариев многими программистами признаётся удобным средством, по различным причинам. В

частности, при таком подходе документация является частью исходного кода, и одни и те же инструменты могут использоваться для сборки программы и одновременной сборки документации к ней. Это также упрощает поддержку документации в актуальном состоянии.

Маркетинговая документация

Для многих приложений необходимо располагать рядом с ними рекламные материалы, с тем чтобы заинтересовать людей, обратив их внимание на продукт. Такая форма документации имеет целью:

- подогреть интерес к продукту у потенциальных пользователей
- информировать их о том, что именно делает продукт, с тем чтобы их ожидания совпадали с тем, что они получают
- объяснить положение продукта по сравнению с конкурирующими решениями

Одна из хороших маркетинговых практик — предоставление слогана — простой запоминающейся фразы, иллюстрирующей то, что мы хотим донести до пользователя, а также характеризующей *ощущение*, которое создаёт продукт.

Часто бывает так, что коробка продукта и другие маркетинговые материалы дают более ясную картину о возможностях и способах использования программы, чем всё остальное.

ОБЩИЕ ПОЛОЖЕНИЯ

1.1. Документация по техническому обеспечению АСУ предназначена для описания проектных решений по техническому обеспечению в документах:

- описание комплекса технических средств;
- структурная схема комплекса технических средств;
- план расположения;
- перечень заявок на разработку новых технических средств;
- перечень заданий заказчику АСУ (Генпроектировщику) на проектирование в смежных частях проекта объекта, связанное с созданием АСУ;
- ведомость оборудования и материалов;
- технические требования к технологическому объекту управления;
- задание на проектирование в смежной части проекта объекта, связанное с созданием АСУ;
- проектная оценка надежности комплекса технических средств;
- принципиальная схема;
- схема автоматизации;
- таблица соединений и подключений;
- схема соединений внешних проводов;
- чертеж общего вида;

- схема подключений внешних проводок;
- спецификация оборудования;
- чертеж установки технических средств;
- ведомость потребности в материалах.

(Измененная редакция, Изм. № 1)

1.2. При разработке документов на подсистему содержание разделов каждого документа ограничивают рамками данной подсистемы.

1.3. В зависимости от назначения и специфических особенностей создаваемых АСУ допускается включать в документы дополнительные разделы и сведения, требования к содержанию которых не установлены настоящим стандартом.

1.4. Отсутствие проектных решений по разделу документа фиксируют в соответствующем разделе с необходимыми пояснениями.

ТРЕБОВАНИЯ К СОДЕРЖАНИЮ ДОКУМЕНТОВ

СОСТАВ И СОДЕРЖАНИЕ

2.1. ТЗ содержит следующие разделы, которые могут быть разделены на подразделы:

- 1) общие сведения;
- 2) назначение и цели создания (развития) системы;
- 3) характеристика объектов;
- 4) требования к системе;
- 5) состав и содержание работ по созданию системы;
- 6) порядок контроля и приемки системы;
- 7) требования к составу и содержанию работ по подготовке объекта разработки

к вводу системы в действие;

- 8) требования к документированию;
- 9) источники разработки.

В ТЗ могут включаться приложения.

2.2. В зависимости от вида, назначения, специфических особенностей проекта и условий функционирования системы допускается оформлять разделы ТЗ в виде приложений, вводить дополнительные, исключать или объединять подразделы ТЗ.

В ТЗ на части системы не включают разделы, дублирующие содержание разделов ТЗ в целом.

2.3. В разделе «Общие сведения» указывают:

- 1) полное наименование системы и ее условное обозначение;

- 2) шифр темы или шифр (номер) договора;
- 3) наименование компаний разработчика и заказчика (пользователя) системы и их реквизиты;
- 4) перечень документов, на основании которых создается система, кем и когда утверждены эти документы;
- 5) плановые сроки начала и окончания работы по созданию системы;
- 6) сведения об источниках и порядке финансирования работ;
- 7) порядок оформления и предъявления заказчику результатов работ по созданию системы (ее частей), по изготовлению и наладке отдельных средств (технических, программных, информационных) и программно-технических (программно-методических) комплексов системы.

2.4. Раздел «Назначение и цели создания (развития) системы» состоит из подразделов:

- 1) назначение системы;
- 2) цели создания системы.

2.4.1. В подразделе «Назначение системы» указывают вид деятельности системы (управление, проектирование и т. п.) и перечень объектов информатизации (объектов), на которых предполагается ее использовать.

2.4.2. В подразделе «Цели создания системы» приводят наименования и требуемые значения технических, технологических, производственно-экономических или других показателей объекта информатизации, которые должны быть достигнуты в результате создания ИС, и указывают критерии оценки достижения целей создания системы.

2.5. В разделе «Характеристики объекта информатизации» приводят:

- 1) краткие сведения об объекте информатизации или ссылки на документы, содержащие такую информацию;
- 2) сведения об условиях эксплуатации объекта автоматизации.

2.6. Раздел «Требования к системе» состоит из следующих подразделов:

- 1) требования к системе в целом;
- 2) требования к функциям (задачам), выполняемым системой;
- 3) требования к видам обеспечения.

Состав требований к системе, включаемых в данный раздел ТЗ на ИС, устанавливают в зависимости от вида, назначения, специфических особенностей и условий функционирования конкретной системы.

2.6.1. В подразделе «Требования к системе в целом» указывают:

- требования к структуре и функционированию системы;
- требования к численности и квалификации персонала системы и режиму его работы;
- показатели назначения;
- требования к надежности;
- требования безопасности;
- требования к эргономике и технической эстетике;
- требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы;
- требования к защите информации от несанкционированного доступа;
- требования по сохранности информации при авариях;
- требования к защите от влияния внешних воздействий;
- требования к патентной чистоте;
- требования по стандартизации и унификации;
- дополнительные требования.

2.6.1.1. В требованиях к структуре и функционированию системы приводят:

- 1) перечень подсистем, их назначение и основные характеристики, требования к числу уровней иерархии и степени централизации системы;
- 2) требования к способам и средствам связи для информационного обмена между компонентами системы;
- 3) требования к характеристикам взаимосвязей создаваемой системы со смежными системами, требования к ее совместимости, в том числе указания о способах обмена информацией (автоматически, пересылкой документов, по телефону и т. п.);
- 4) требования к режимам функционирования системы;
- 5) требования по диагностированию системы;
- 6) перспективы развития, модернизации системы.

2.6.1.2. В требованиях к численности и квалификации персонала на ИС приводят:

- требования к численности персонала (пользователей) ИС;
- требования к квалификации персонала, порядку его подготовки и контроля знаний и навыков;
- требуемый режим работы персонала ИС.

2.6.1.3. В требованиях к показателям назначения ИС приводят значения параметров, характеризующие степень соответствия системы ее назначению.

2.6.1.4. В требования к надежности включают:

- 1) состав и количественные значения показателей надежности для системы в целом или ее подсистем;
- 2) перечень аварийных ситуаций, по которым должны быть регламентированы требования к надежности, и значения соответствующих показателей;
- 3) требования к надежности технических средств и программного обеспечения;
- 4) требования к методам оценки и контроля показателей надежности на разных стадиях создания системы в соответствии с действующими нормативно-техническими документами.

2.6.1.5. В требования по безопасности включают требования по обеспечению безопасности при поставке, наладке, эксплуатации и обслуживании системы.

2.6.1.6. В требования по эргономике и технической эстетике включают показатели ИС, задающие необходимое качество взаимодействия человека с машиной и комфортность условий работы персонала.

2.6.1.7. В требования к защите информации от несанкционированного доступа включают требования, установленные действующей в отрасли и информационной среде заказчика.

2.6.1.8. В требованиях по сохранности информации приводят перечень событий: аварий, отказов технических средств (в том числе - потеря питания) и т. п., при которых должна быть обеспечена сохранность информации в системе.

2.6.1.9. В требованиях по патентной чистоте указывают перечень стран, в отношении которых должна быть обеспечена патентная чистота системы и ее частей.

2.6.1.10. В дополнительные требования включают специальные требования по усмотрению разработчика или заказчика системы.

2.6.2. В подразделе «Требование к функциям (задачам)», выполняемым системой, приводят:

- по каждой подсистеме перечень функций, задач или их комплексов (в том числе обеспечивающих взаимодействие частей системы), подлежащих автоматизации;

- при создании системы в две или более очереди - перечень функциональных подсистем, отдельных функций или задач, вводимых в действие в 1-й и последующих очередях;
- временной регламент реализации каждой функции, задачи (или комплекса задач);
- требования к качеству реализации каждой функции (задачи или комплекса задач), к форме представления выходной информации, характеристики необходимой точности и времени выполнения, требования одновременности выполнения группы функций, достоверности выдачи результатов;
- перечень и критерии отказов для каждой функции, по которой задаются требования по надежности.

2.6.3. В подразделе «Требования к видам обеспечения» в зависимости от вида системы приводят требования к математическому, информационному, лингвистическому, программному, техническому, метрологическому, организационному, методическому и другие видам обеспечения системы.

2.6.3.2. Для информационного обеспечения системы приводят требования:

- 1) к составу, структуре и способам организации данных в системе;
- 2) к информационному обмену между компонентами системы;
- 3) к информационной совместимости со смежными системами;
- 4) по применению систем управления базами данных;
- 5) к структуре процесса сбора, обработки, передачи данных в системе и представлению данных;
- 6) к защите данных;
- 7) к контролю, хранению, обновлению и восстановлению данных;

2.6.3.3. Для лингвистического обеспечения системы приводят требования к применению в системе языков программирования высокого уровня, языков взаимодействия пользователей и технических средств системы, а также требования к кодированию и декодированию данных, к языкам ввода-вывода данных, языкам манипулирования данными, средствам описания предметной области, к способам организации диалога.

2.6.3.4. Для программного обеспечения системы приводят перечень покупных программных средств, а также требования:

- 1) к зависимости программных средств от операционной среды;

- 2) к качеству программных средств, а также к способам его обеспечения и контроля;

2.6.3.5. Для технического обеспечения системы приводят требования:

- 1) к видам технических средств, в том числе к видам комплексов технических средств, программно-технических комплексов и других комплектующих изделий, допустимых к использованию в системе;
- 2) к функциональным, конструктивным и эксплуатационным характеристикам средств технического обеспечения системы.

2.6.3.6. В требованиях к метрологическому обеспечению приводят:

- 1) предварительный перечень измерительных каналов;
- 2) требования к точности измерений параметров и (или) к метрологическим характеристикам измерительных каналов;
- 3) требования к метрологической совместимости технических средств системы;
- 4) перечень управляющих и вычислительных каналов системы, для которых необходимо оценивать точностные характеристики;
- 5) требования к метрологическому обеспечению технических и программных средств, входящих в состав измерительных каналов системы, средств, встроенного контроля, метрологической пригодности измерительных каналов и средств измерений, используемых при наладке и испытаниях системы;
- 6) вид метрологической аттестации (государственная или ведомственная) с указанием порядка ее выполнения и организаций, проводящих аттестацию.

2.6.3.7. Для организационного обеспечения приводят требования:

- 1) к структуре и функциям подразделений, участвующих в функционировании системы или обеспечивающих эксплуатацию;
- 2) к организации функционирования системы и порядку взаимодействия персонала ИС и персонала объекта информатизации;
- 3) к защите от ошибочных действий персонала системы.

2.7. Раздел «Состав и содержание работ по созданию (развитию) системы» должен содержать перечень стадий и этапов работ по созданию системы, сроки их выполнения, перечень организаций - исполнителей работ, ссылки на документы, подтверждающие согласие этих организаций на участие в создании системы, или запись, определяющую ответственного (заказчик или разработчик) за проведение этих работ.

В данном разделе также приводят:

- 1) перечень документов предъявляемых по окончании соответствующих стадий и этапов работ;
- 2) вид и порядок проведения экспертизы технической документации (стадия, этап, объем проверяемой документации, организация-эксперт);
- 3) программу работ, направленных на обеспечение требуемого уровня надежности разрабатываемой системы (при необходимости);
- 4) перечень работ по метрологическому обеспечению на всех стадиях создания системы с указанием их сроков выполнения и организаций-исполнителей (при необходимости).

2.8. В разделе «Порядок контроля и приемки системы» указывают:

- 1) виды, состав, объем и методы испытаний системы и ее составных частей;
- 2) общие требования к приемке работ по стадиям, порядок согласования и утверждения приемочной документации;

2.9. В разделе «Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие» необходимо привести перечень основных мероприятий и их исполнителей, которые следует выполнить при подготовке проекта к вводу ИС в действие.

В перечень основных мероприятий включают:

- 1) приведение поступающей в систему информации (в соответствии с требованиями к информационному и лингвистическому обеспечению);
- 2) создание условий функционирования проекта, при которых гарантируется соответствие создаваемой системы требованиям, содержащимся в ТЗ;
- 3) создание необходимых для функционирования системы подразделений и служб;
- 4) сроки и порядок комплектования штатов и обучения персонала.

2.10. В разделе «Требования к документированию» приводят:

- 1) согласованный разработчиком и Заказчиком системы перечень подлежащих разработке комплектов и видов документов; перечень документов, выпускаемых на машинных носителях;

- 2) при отсутствии государственных стандартов, определяющих требования к документированию элементов системы, дополнительно включают требования к составу и содержанию таких документов.

2.11. В разделе «Источники разработки» должны быть перечислены документы и информационные материалы, на основании которых разрабатывалось ТЗ и которые должны быть использованы при создании системы.

Сеть доступа

Сеть доступа представляет собой нижний уровень иерархии телекоммуникационной сети.

К этой сети подключаются конечные (терминальные) узлы - оборудование, установленное у пользователей (*абонентов*, клиентов) сети. В случае *компьютерной сети* конечными узлами являются компьютеры, телефонной - телефонные аппараты, а *телевизионной* или *радиосети* - соответствующие теле- и радиоприемники.

Основное назначение *сети доступа* - концентрация *информационных потоков*, поступающих по многочисленным каналам связи от оборудования пользователей, в сравнительно небольшом количестве узлов *магистральной сети*.

Сеть доступа, как и телекоммуникационная сеть в целом, может состоять из нескольких уровней (на рисунке показано два). *Коммутаторы*, установленные в узлах нижнего уровня, мультиплексируют информацию, поступающую по многочисленным абонентским каналам (называемым часто абонентскими окончаниями, *local loop*) и передают ее коммутаторам верхнего уровня, чтобы те в свою очередь передали ее коммутаторам магистрали. Количество уровней *сети доступа* зависит от ее размера; небольшая *сеть доступа* может состоять из одного уровня, а крупная - из двух-трех. Следующие уровни осуществляют дальнейшую концентрацию трафика, собирая его и мультиплексируя в более скоростные каналы.

Магистральная сеть

Магистральная сеть объединяет отдельные *сети доступа*, выполняя функции транзита трафика между ними по высокоскоростным каналам. *Коммутаторы магистрали* могут оперировать не только информационными соединениями между отдельными пользователями, но и *агрегированными информационными потоками*, переносящими данные большого количества пользовательских соединений. В результате информация с помощью *магистрали* попадает в *сеть доступа* получателей, демультиплексируется там и коммутируется таким образом, что на входной порт оборудования пользователя поступает только та информация, которая ему адресована.

В том случае, когда *абонент*-получатель подключен к тому же *коммутатору* доступа, что и *абонент*-отправитель (непосредственно или через подчиненные по иерархии связей *коммутаторы*), последний выполняет необходимую операцию *коммутации* самостоятельно.

Информационные центры

Информационные центры /центры управления сервисами - это собственные информационные ресурсы сети, на основе которых осуществляется обслуживание пользователей. В таких *центрах* может храниться информация двух типов:

- пользовательская информация, то есть те данные, которые непосредственно интересуют пользователей сети;
- вспомогательная служебная информация, позволяющая предоставлять пользователям некоторые услуги.

Примером информационных ресурсов первого типа могут служить Web-порталы, на которых расположена разнообразная справочная информация и новости, информация электронных магазинов и т.п. В *телефонных сетях* роль таких центров играют службы экстренного вызова (например, милиции, скорой помощи) и справочные службы различных организаций и предприятий - вокзалов, аэропортов, магазинов и т.п. В *телевизионных сетях* такими центрами являются телестудии, поставляющие "живую" картинку или же воспроизводящие ранее записанные сюжеты или фильмы.

К ресурсам второго типа относятся, например, различные *системы аутентификации* и *авторизации* пользователей, с помощью которых организация, владеющая сетью, проверяет права пользователей на получение тех или иных услуг; системы биллинга, которые в коммерческих сетях подсчитывают плату за предоставленные услуги; базы данных учетной информации пользователей, хранящие имена и пароли, а также перечни услуг, на которые подписан каждый пользователь. В *телефонных сетях* существуют центры управления сервисами (Services Control Point, *SCP*), где установлены компьютеры, на которых хранятся программы нестандартной обработки телефонных вызовов пользователей, например вызовов бесплатных справочных служб коммерческих предприятий (так называемые службы 800) или вызовов при проведении телеголосования. Еще одним из распространенных видов вспомогательного *информационного центра* является централизованная система управления сетью, которая представляет собой программное обеспечение, работающее на одном или нескольких компьютерах.

2.2. Перечень вопросов и заданий для промежуточной аттестации

Вопросы для подготовки к экзамену

Теоретическая часть.

1. Задачи сопровождения информационной системы.
2. Ролевые функции и организация процесса сопровождения.
3. Сценарий сопровождения.
4. Договор на сопровождение.
5. Анализ исходных программ и компонентов программного средства.
6. Программная инженерия и оценка качества.
7. Реинжиниринг.
8. Методы резервного копирования.
9. Восстановление информации в информационной системе.
10. Цели и регламенты резервного копирования.
11. Сохранение и откат рабочих версий системы.
12. Сохранение и восстановление баз данных.
13. Организация процесса обновления в информационной системе.
14. Регламенты обновления.
15. Техническое сопровождение информационной системы.
16. Регламенты по техническому сопровождению обслуживаемой информационной системы.
17. Обеспечение безопасности функционирования информационной системы.
18. Организация доступа пользователей к информационной системе.
19. Организация сбора данных об ошибках в информационных системах, источники сведений
20. Системы управления производительностью приложений.
21. Мониторинг сетевых ресурсов.
22. Схемы и алгоритмы анализа ошибок, использование баз знаний.
23. Отчет об ошибках системы: содержание, использование информации.
24. Методы и инструменты тестирования приложений

Практическая часть.

Для выбранного определенного объекта информации (номер варианта соответствует номеру студента по списку) необходимо описать поддержку сопровождения объекта, провести анализ сопровождения объекта информации по следующим разделам:

- 1 характер происхождения угроз;
- 2 классы каналов несанкционированного получения информации;
- 3 причины нарушения целостности информации;
- 4 возможные ошибки и проблемы совместимости.

Наименование объекта защиты информации:

Одиночно стоящий компьютер в бухгалтерии.
Сервер в бухгалтерии.
Почтовый сервер.
Веб-сервер.
Компьютерная сеть материальной группы.
Одноранговая локальная сеть без выхода в Интернет.
Одноранговая локальная сеть с выходом в Интернет.
Сеть с выделенным сервером без выхода в Интернет.
Сеть с выделенным сервером с выхода в Интернет.
Телефонная база данных (содержащая и информацию ограниченного пользования) в твердой копии и на электронных носителях.
Телефонная сеть.
Средства телекоммуникации (радиотелефоны, мобильные телефоны).
Банковские операции (внесение денег на счет и снятие).
Операции с банковскими пластиковыми карточками.
Компьютер, хранящий конфиденциальную информацию о сотрудниках предприятия.
Компьютер, хранящий конфиденциальную информацию о разработках предприятия.
Материалы для служебного пользования на твердых носителях и на электронных носителях в производстве.
Материалы для служебного пользования на твердых носителях и на электронных носителях на закрытом предприятии.
Материалы для служебного пользования на твердых носителях в архиве.
Материалы для служебного пользования на твердых носителях и на электронных носителях в налоговой инспекции.
Комната для переговоров по сделкам на охраняемой территории.
Комната для переговоров по сделкам на неохраняемой территории.
Сведения для средств массовой информации, цензура на различных носителях информации (твердая копия, фотографии, электронные носители и др.).
Судебные материалы (твердая копия и на электронных носителях).
Паспортный стол РОВД (твердая копия и на электронных носителях).

Критерии оценивания ответа

Оценка «отлично» выставляется студенту, если он глубоко и прочно усвоил программный материал, исчерпывающе, последовательно, четко и логически стройно его излагает, умеет тесно увязывать теорию с практикой, свободно справляется с задачами, вопросами и другими видами применения знаний, причем не затрудняется с ответом при видоизменении заданий.

Оценка «хорошо» выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос, правильно

применяет теоретические положения.

Оценка «удовлетворительно» выставляется студенту, если он имеет знания только основного материала, но не усвоил его деталей, допускает неточности, Недостаточно формулировки, нарушения логической последовательности в изложении программного материала.

Оценка «неудовлетворительно» выставляется студенту, который не знает значительной части программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы.

Критерии оценивания выполнения практического задания

- рациональное распределение времени по этапам выполнения задания
- обращение в ходе задания к информационным источникам
- знание терминологии
- скорость выполнение
- количество предложенных вариантов решения поставленной задачи.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

№ п.п.	Содержание изменения	Дата, номер протокола заседания кафедры, подпись зав.кафедрой
1	2	3
1		
2		
3		
4		