На конкурс на лучшую студенческую работу в разделе «Техническая кибернетика, вычислительная, микропроцессорная техника и информатика»

Студенческая научная работа по теме:

ПРОЕКТИРВАНИЕ И РЕАЛИЗАЦИЯ ВИРТУАЛЬНЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Девиз: «Лучшее враг хорошего»

Донецк – 2005 год

СВЕДЕНЬЯ
об авторе и научном руководителе научно-исследовательской работы
по теме:
ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ВИРТУАЛЬНЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Которая направляется на _____ конкурс, который проводится в 2005 году на лучшую студенческую работу по техническим, гуманитарным и общественным наукам.

1. Фамилия: Гриценко
2. Имя: Антон
3. Отчество: Александрович
4. Высшее учебное заведение, в котором автор получает образование: Донецкий национальный технический университет (ДонНТУ)
5. Факультет: Вычислительной техники и информатики (ВТИ)
6. Направление: 6.0915 «Компьютерная инженерия»
7. Курс, группа: 5, ВТ-01м
8. Домашний адрес: 83010, г. Донецк, ул. Алябьева, д. 16, кв. 9
9. Гражданство: Украина

Научный руководитель:

1. Фамилия: Шевченко
2. Имя: Ольга
3. Отчество: Георгиевна
4. Кафедра, должность: Электронных вычислительных машин, старший преподаватель
5. Научная степень:
6. Научное звание:

Работа рекомендована к участию в конкурсе 2005-06 учебного года по результатам вузовского конкурса 2005-го года

Председатель совета НТТМ _____

РЕФЕРАТ
Студенческая научная работа
25 рисунков, 70 страниц

Объектом исследования данной работы является проектирование, разработка, возможности и область применения виртуальных систем реального времени. Определяется понятие виртуальных систем реального времени и их характеристики. Проводится полноценная разработка архитектуры таких систем и их проектирование, включающие исследование аспектов применения современных методов проектирования и аспектов проектирования виртуальных систем. Рассматриваются вопросы, связанные с реализацией виртуальных систем, определяются факторы и условия их использования, методы оптимизации при разработке таких систем.

СИСТЕМА РЕАЛЬНОГО ВРЕМЕНИ, ГИБКАЯ СИСТЕМА$ РЕАЛЬНОГО ВРЕМЕНИ, ВИРТУАЛЬНАЯ СИСТЕМА РЕАЛЬНОГО ВРЕМЕНИ, БАЗОВАЯ СИСТЕМА, АБСТРАКЦИЯ, АРХИТЕКТУРА, МОДЕЛЬ ПРЕЦЕДЕНТОВ, МОДЕЛЬ СОСТОЯНИЙ, СТАТИЧЕСКАЯ МОДЕЛЬ, ИНТЕРФЕЙС, СПЕЦИАЛИЗАЦИЯ, ПРОЦЕСС, СУПЕРВИЗОР, ПЛАНИРОВЩИК, ДИСПЕТЧЕР, ФАКТОР, МНОГОЗАДАЧНОСТЬ, ОДНОЗАДАЧНОСТЬ, УНИВЕРСАЛЬНОСТЬ, ПРИОРИТЕТ, ДРАЙВЕР, ЗАХВАТ РЕСУРСА, СЕРВИС, ЯДРО

Содержание

ОТЗЫВ

научного руководителя на студенческую работу на тему:

«Проектирование и реализация виртуальных систем реального времени»

Разработка современного программного обеспечения требует разделения этапов проектирования и реализации. Проектирование сложных систем является более важным аспектом, нежели их реализация, причем важность качественного проектирования растет пропорционально величине решаемой задачи. Существует набор современных методологий (ООП, UML и т.д.), которые позволяют получать эффективные решения при проектировании сложных систем.

С развитием аппаратно-программной базы возник вопрос о возможности построения операционных систем на базе других систем. Этот подход является дальнейшим развитием методологий проектирования, в частности методологий ООП. В работе рассмотрен наиболее актуальный подход к этой проблеме – построение платформенно-независимых систем.

В работе рассмотрено проектирование систем реального времени, разработано понятие виртуальной системы и связанные с ним аспекты.

В работе в полной мере рассмотрены вопросы проектирования и реализации виртуальных систем реального времени.

Вопросы проектирования рассмотрены в следующем объеме:

- Разработка архитектуры виртуальных систем.
- Декомпозиция архитектуры на подсистемы.
- Декомпозиция подсистем.
- Проектирование статического и динамического поведения отдельных подсистем с разработкой динамических моделей (модель прецедентов, модель конечных автоматов и модель активности) и статических моделей (статическое представление и структурная модель).

Проектирование проводится с использованием современных методологий данной области.

Вопросы реализации рассмотрены в следующем объеме:

- Факторы построения виртуальных систем.
- Аспекты выбора и использования базовых систем.
- Методы оптимизации реализации виртуальных систем на многозадачных системах.

Продолжением теоретической части данной работы является оценка методов оптимизации реализации виртуальных систем реального времени на базе многозадачных систем.

Продолжением практической части данной работы является детальное проектирование и реализация виртуальной системы.

Работа может использоваться в качестве базовой для дальнейших исследований в области построения виртуальных систем.

Вступление.

В работе определяется понятие виртуальной системы реального времени, возможности для создания и использования таких систем. Рассматриваются вопросы разработки архитектуры и проектирования виртуальных систем реального времени и возможности реализации таких систем.

Разрабатывается архитектура виртуальной системы реального времени, проводится ее декомпозиция и проектирование всех ее подсистем. Проектирование каждой подсистемы включает в свою очередь ее декомпозицию, разработку модели прецедентов и модели состояний. Результатом является статическое моделирование каждой подсистемы. Наибольшее внимание отдается описанию моделей прецедентов, как моделей описывающих взаимодействие подсистемы с окружающими и модели состояний, как модели описывающей поведение подсистемы.

В качестве базиса проектирования выбран следующий ряд современных методологий, которые являются наиболее эффективными и предпочтительными:

- Нисходящее функционально-ориентированное проектирование с выделением критически-ориентированных частей системы
- Использование UML как основного средства описания проекта – все этапы проектирования проиллюстрированы подробными диаграммами
- Объектно-ориентированный подход к проектированию
    1. Повышение уровня абстракции компонент.
    2. Разработка эффективных интерфейсов.
    3. Инкапсуляция реализации компонент.
    4. Использование иерархий специализации.
- Высокая факторизация проекта
    1. Факторизация крупных систем на подсистемы.
    2. Разбиение подсистем на отдельные модули.

Разработка архитектуры и проектирования системы затрагивает следующий ряд вопросов:

- Проектирование общей архитектуры системы.
- Декомпозиция архитектуры системы.
- Проектирование общей архитектуры подсистем.
- Декомпозиция архитектуры подсистем.

- Исследование взаимодействия компонента и проектирование модели прецедентов для каждого из компонента подсистемы, разработка интерфейсов компонента.
- Исследование поведения компонента и проектирование модели состояний для каждого из компонентов системы, рассмотрение реализации компонента.
- Объектно-ориентированное исследование моделей и построение статистической модели для каждой подсистем.

Аспекты детального проектирования и кодирования связанные с полноценной реализацией виртуальной системы реального времени в данной работе не рассматриваются.

Возможности реализации виртуальных систем реального времени определяются исследованием следующего ряда вопросов:

- Факторы и условия использования виртуальной системы реального времени, преимущества и недостатки таких систем.
- Обзор базовых систем и выбор базовой системы, преимущества и недостатки базовых систем.

По результатам возможностей реализации виртуальных систем реального времени сделаны выводы и определены последующие направления работы.

1. Системы реального времени

1.1. Введение в системы реального времени

Система реального времени – система, для которой правильность работы определяется, выполнением задач в определенные, жестко факторизованные промежутки времени.

Неправильное распределение времени в таких системах вызывает серьезные ошибки, которые ведут к аварийному завершению работы всей системы, в случае невозможности сверхоперативного исправления ошибки. Одним из характерных факторов, определяющих принадлежность системы к системам реального времени является поведение системы в результате неправильного распределения времени, другим фактором, определяющим такие системы, являются повышенные требования к стабильности работы и надежности. Этот фактор обусловлен областью применения систем реального времени: промышленные системы управления технологическими процессами, встроенные и бортовые (мобильные) системы, системы распределенной обработки данных, системы обработки потоковых данных и т.д.

Часто возникает потребность в построении систем реального времени, для которых распределение времени остается наиболее важным фактором, но допускает некоторые отклонения. Формально это отклонение называется временем отзыва системы (либо временем отклика системы). Время отзыва системы реального времени – промежуток времени, который определяет время между возникновением системного события и началом его обработки. Например, для распределенной базы данных время отзыва может быть сопоставимо с максимально допустимым временем передачи данных от клиента к серверу.

Наиболее важным аспектом проектирования и реализации таких систем является то, что время отзыва такой системы можно использовать для создания унифицированной среды взаимодействия с оборудованием, что позволит поднять систему на более высокий уровень абстракции, сделав ее аппаратно независимой. Такой уровень абстракции принято называть уровнем абстракции от оборудования. Большинство современных систем реального времени включают уровень абстракции от оборудования.

Следующим уровнем абстракции становится надстройка систем реального времени над другими системами: однозадачными и многозадачными системами реального времени и универсальными многозадачными системами. Такой уровень абстракции принято называть уровнем абстракции от платформы. Этот уровень абстракции нельзя реализовать в системах реального времени в их чистом виде.

Разработка систем реального времени уровня абстракции от платформы является следующей ступенью, после абстракции от оборудования, в развитии систем реального времени.

Системы, соответствующие уровню абстракции от платформы, в дальнейшем будут называться виртуальными системами реального времени. На примере иерархии управления технологическим процессом, на рисунке 1.1 показано положение виртуальных систем реального времени.

Возникновение возможности проектирования и эффективного использования таких систем связано с постоянной эволюцией аппаратно-платформенных средств. Наращивание мощности современной аппаратуры и эффективности платформ снижает время, требуемое ими для обработки информации, в частности за счет предоставления сервисов управления ресурсами. Виртуальная система реального времени использует предоставленное ей время отзыва для вызова и выполнения сервисов базовой системы, что позволяет отказаться от проектирования и реализации таких сервисов и сконцентрироваться собственно на виртуальной системе.

1.2. Виды систем реального времени

Все системы реального времени факторизуются по признаку времени отзыва. В зависимости от времени отзыва выделяется два фундаментальных типа систем реального времени:

Жесткие системы реального времени – системы реального времени, для которых время отзыва должно быть нулевым или стремящимся к нулю, а временные факторы для каждой задачи должны выполняться неукоснительно, исход из чего эти системы не могут проектироваться как виртуальные. Примером жесткой системы реального времени может служить система управления кардиостимулятором. Такая система взаимодействует с ресурсами ниже уровня абстракции от оборудования, что позволяет использовать их наиболее эффективно. Важность правильного распределения времени в такой системе ярко выражается ее назначением. Такие системы обычно обладают рядом недостатков:

- Зависимость от оборудования – нулевое время отзыва требует максимально эффективного использования оборудования, что ведет к проектированию и разработке узко специализированных систем.
- Плохая переносимость – большинство таких систем требуют повторного проектирования и реализации при изменении аппаратной базы.
- Плохая факторизация – такие системы ограничены предметной областью их применения, то есть в большинстве случаев задача разработки системы реального времени вторична, по отношению к задаче предметной области и не может быть отделена от нее.

Гибкие системы реального времени – системы, для которых устанавливается время отзыва, размер которого варьируется в соответствии с типом каждой конкретной системы. В зависимости от времени отзыва изменяется и поведение системы в результате возникновения различных ошибок, так, если время отзыва превышает время выполнения определенного процесса, то такой процесс, в конечном счете, может быть проигнорирован. Например, гибкая система реального времени, которая отслеживает состояние датчика, может игнорировать изменение состояния датчика на время меньшее времени отзыва. Такая система неприемлема для управления температурой реактора, но приемлема для управления температурой двигателя внутреннего сгорания.

1.3. Виртуальные системы реального времени

Под виртуальной системой реального времени будем понимать гибкую систему реального времени, которая построена на базе другой системы. В свою очередь систему, на базе которой построена виртуальная система, будем называть базовой системой. Виртуальная система реального времени использует время отзыва гибкой системы реального времени для достижения максимальной аппаратной и платформенной независимости. Виртуальная система реального времени обладает следующими качествами:

- Абстрактность – виртуальная система находится на наиболее высоком из доступных уровней абстракции, что позволяет проектировать ее без учета влияния аппаратных или платформенных средств, кроме ее критически-ориентированных частей, которые проектируются с учетом только платформенных средств.
- Адаптируемость – виртуальная система полностью переносима на уровне проекта, вплоть до его детального уровня, единственное требуемое изменение – повторная реализация критически-ориентированных частей проекта.
- Факторизация – проектирование виртуальной системы реального времени может проходить как отдельный процесс, выполняемый независимо от проектирования концептуальных задач разрабатываемой предметной области, кроме того, разработка архитектуры и проектирование, исключая его детальный уровень, является совершенно отдельной задачей и может выполняться один раз с последующей адаптацией.
- Надежность – правильное проектирование виртуальной системы, позволяет избежать некоторых недостатков базовых систем или акцентировать на них достаточное внимание.
- Концептуальность – виртуальная система проектируется независимо от оборудования и платформы, что исключает влияние конкретных аппаратно-программных решений на общую концепцию системы, позволяя максимально точно отобразить концепции, положенные в основу виртуальной системы.
- Стоимость – виртуальная система стоит дешевле, потому что использует взаимодействие с ясно описанными, высокоуровневыми сервисами базовых систем. Использование этих интерфейсов снижает количество ошибок в системе и облегчает реализацию и тестирование таких систем. Стоимость повторной реализации таких сервисов может превысить стоимость реализации конкретных прикладных задач.

Абстрактность и Адаптируемость можно охарактеризовать как универсальность системы. В современных условиях универсальность является одним из наиболее важных факторов, в том понимании, что данная универсальность полностью абстрагирует предметную область от ее реализации.

Для виртуальных систем реального времени наиболее важным этапом является проектирование таких систем. Выработка правильной методологии проектирования в данной области важна потому, что она обеспечит возможность создания некоторой платформы виртуальных систем реального времени. Как было указано, для таких систем высокая факторизация позволяет разработать универсальный проект системы и проводить лишь детальное проектирование и реализацию в каждом конкретном случае.

В современных условиях проектирование крупных задач является наиболее важной частью их решения. Ошибка концептуальной модели обходится в среднем в 10 раз дороже ошибки высокоуровневого проектирования, в 100 раз дороже ошибки детального проектирования и в 1000 раз дороже ошибки реализации, что показывает важность проектирования крупных систем. Для не факторизованных систем ошибка в проектировании системы реального времени обходится не меньше чем ошибка проектирования концептуальной области.

Кроме того, с увеличением размера задачи доля проектирования увеличивается, а доля реализации уменьшается, при критически-ориентированном подходе реализация постепенно превращается во второстепенную задачу. Это замечание важно, т.к. проектирование системы реального времени, и ее реализация, может требовать больших усилий и ресурсов, чем реализация отдельных задач предметной области. Разработка и исследование методологий в данной области облегчает задачу проектирования таких систем и освобождает от необходимости уделять им чрезмерное влияние.

2. Проектирование виртуальных систем реального времени

2.1. Проектирование архитектуры виртуальной системы

Архитектура виртуальной системы адаптирует существующие архитектурные решения для систем реального времени и универсальных систем. Архитектура виртуальной системы приведена на рисунке 2.1.

Базовой подсистемой виртуальной системы является супервизор. В рамках виртуальной системы супервизор – специализированный процесс, который координирует работу всех остальных процессов системы. Процесс системы – контекст выполнения некоторой прикладной (задача, которая решает некоторую проблему предметной области) или системной (задача, которая не решает проблему предметной области самостоятельно, но используется при решении одной или нескольких проблем) задачи.

Виртуальная система не оперирует отдельными задачами, т.к. они не относятся к ее предметной области. Все прикладные и системные задачи абстрагируются процессами, которые инкапсулируют собственно задачи, предоставляя только развитой интерфейс управления ими для подсистем супервизора.

Разработка архитектуры и проектирование супервизора является отдельным этапом. Супервизор не может интерпретироваться как обычный процесс, потому что он не решает ни прикладных, ни системных задач, он только координирует их порядок их решения.

Супервизор может управлять двумя базовыми типами процессов (синхронными и асинхронными), причем в виртуальной системе должен выполняться хотя бы один синхронный процесс. Наличие хотя бы одного синхронного процесса – базовое условие построения системы реального времени. Если в системе используются только асинхронные процессы, то целесообразность реализации такой системы, как системы реального времени должна рассматриваться более тщательно.

Реализация супервизора независима от реализации отдельных процессов, что предоставляет возможность реализовывать новые версии отдельных процессов при сохранности базовой версии супервизора. Процесс абстрагирует задачу от виртуальной системы, что позволяет изменять решения любых задач, при этом в системе изменится только реализация одного процесса.

Например, при изменении обработчика сигнала датчика, достаточно заново реализовать только соответствующий процесс и его характеристики, не затрагивая систему в целом.

2.2. Проектирование процессов виртуальной системы

2.2.1. Процессы виртуальной системы

Процесс виртуальной системы определяет контекст выполнения одной из прикладных или системных задач виртуальной системы и интерфейс управления выполнением такой задачи. Контекст выполнения полностью отделен от интерфейса, что обеспечивает инкапсуляцию решения задачи, вся ответственность по правильной реализации интерфейса возлагается на каждый отдельный процесс.

В виртуальной системе используются два базовых типа процессов:

Синхронные (статически планируемые) процессы – набор процессов, временные характеристики которых известны системе до ее инициализации, что позволяет полностью определить алгоритм управления поведением таких процессов и взаимодействия между ними в процессе инициализации или выполнить статическое планирование. Если система использует только синхронные процессы, то ее состояние в любой момент времени детерминировано с учетом времени отзыва системы. Назовем такую систему статически планируемой виртуальной системой. Синхронные процессы инкапсулируют прикладные задачи.

Асинхронные (динамически планируемые) процессы – набор процессов, временные характеристики, которых только частично известны до инициализации системы, частично же определяются в процессе работы виртуальной системы. Планирование таких процессов называется динамическим. При использовании асинхронных процессов состояние системы недетерминировано в конкретный момент времени, но может быть определено вероятностно с учетом данных статистического и динамического планирования. Система, использующая динамическое планирование называется динамически планируемой виртуальной системой. Асинхронные процессы инкапсулируют системные задачи.

Динамически планируемые виртуальные системы обладают следующими преимуществами:

1. Вынесение системных задач, которые обычно являются небольшими по объему вычислений и используемых ресурсов, в динамические процессы позволяет акцентировать внимание на более важных и тяжеловесных синхронных задачах.
2. Выделение в сложных синхронных процессах небольших асинхронных процессов, которые могут вызываться несколько раз для выполнения одних и тех же действия над разными данными.

3. Использование динамических процессов позволяет эффективно оформить небольшие событийные процессы, которые возникают на короткий промежуток времени для обработки небольшого события, например интерактивного ввода или вывода.

Второй фактор активно применяется в современных многозадачных операционных системах, где тяжеловесные статические процессы делятся на ряд легковесных потоков.

Третий фактор используется в большинстве систем реального времени, где обработка прерываний выполняется небольшими асинхронными процессами.

Основным недостатком динамически планируемых систем является то, что они недетерминированы. Поведение детерминированных систем реального времени легче контролировать.

Статически планируемые виртуальные системы используют однотипные процессы, что позволяет упростить их интерфейс и, следовательно, облегчить его реализацию.

2.2.2. Прецеденты управления и модели состояний процессов

Модель прецедентов управления определяет возможность управления поведением процесса со стороны подсистем супервизора. Прецеденты описывают все возможные события, на которые должен реагировать процесс. Возможность реакции на событие и результат реакции на событие определяется текущим состоянием процесса, которое определяется моделью состояний процесса.

Простой процесс – это процесс, который обеспечивает минимальный интерфейс для управления своим поведением со стороны супервизора. Такие процессы могут использоваться в виртуальных однозадачных системах со статическим планированием и являются базой для всех остальных процессов. Модель прецедентов управления простым процессом приведена на рисунке 2.2.

Прецеденты делятся на два типа:

- Внешние – прецеденты, которые являются не планируемыми с точки зрения выполняемой процессом задачи, и всегда генерируются подсистемами супервизора, внешними по отношению к процессу.
- Внутренние – прецеденты, которые являются запланированными с точки зрения выполняемой процессом задачи, и всегда генерируются процессом.

Описание прецедентов управления простым процессом:

Внешний прецедент «Создание»

Сводка. Создание объекта процесса для требуемой задачи и выделение ему ресурсов

Предусловия. Для одной задачи используется только один процесс, для одного процесса – один объект.

Постусловия. Создан объект процесса, который получил контроль над требуемыми ресурсами.

Описание:

1. Проверить существование требуемого процесса.
2. Если процесс существует, проигнорировать и завершить.
3. Создать объект процесса.
4. Подготовить ресурсы, требуемые процессу.
5. Передать управление ресурсами процессу.

Альтернативы:

1. Если объект процесса не может быть создан, выполнить обработку ошибки.
2. Если ресурсы не могут быть подготовлены, выполнить обработку ошибки.

Внешний прецедент «Инициализация»

Сводка. Подготовка контекста выполнения задачи.

Предусловия. Процесс создан.

Постусловия. Контекст очищен и процесс готов к выполнению.

Описание:

1. Проверить существование требуемого процесса.
2. Если контекст процесса создан, очистить его.
3. Если контекст процесса не создан, создать его.

Альтернативы:

1. Если контекст выполнения задачи не может быть повторно использован, выполнить обработку ошибки.

Внешний прецедент «Запуск»

Сводка. Начать выполнение задачи процесса.

Предусловия. Процесс инициализирован.

Постусловия. Начато выполнение задачи процесса.

Описание:

1. Подготовить контекст процесса к выполнению.
2. Начать выполнение задачи.

Альтернативы:

1. Если выполнение не может быть начато, аварийно завершить процесс.

Внешний прецедент «Останов»

Сводка. Прервать выполнение задачи и завершить выполнение процесса.

Предусловия. Процесс выполняется.

Постусловия. Процесс прекратил выполнение и аварийно завершен.

Описание:

1. Аварийно завершить выполнение задачи.
2. Оповестить супервизор.

Внешний прецедент «Уничтожение»

Сводка. Уничтожает процесс и освобождает занятые ресурсы.

Предусловия. Процесс завершил выполнение.

Постусловия. Процесс уничтожен, ресурсы освобождены.

Описание:

1. Очистить контекст выполнения.
2. Передать контроль над используемыми ресурсами супервизору.

Внутренний прецедент «Аварийное завершение»

Сводка. Выполнение задачи прервано супервизором или из-за возникновения внутренней ошибки.

Предусловия. Возникновение ошибки выполнения, прецедент «Останов».

Постусловия. Процесс аварийно завершен.

Описание:

1. Завершить выполнение задачи.
2. Оповестить супервизор.

Внутренний прецедент «Нормальное завершение»

Сводка. Выполнение задачи завершено успешно.

Предусловия. Выполнение завершено.

Постусловия. Процесс нормально завершен.

Описание:

1. Завершить выполнение задачи.
2. Оповестить супервизор.

При возникновении ошибки обработки прецедента должна проводится обработка такой ошибки. Некоторые ошибки могут быть исправлены супервизором (например, он может попытаться перераспределить ресурсы и исправить ошибку создания процесса), некоторые процессом (например, для очистки контекста может быть выполнено несколько

попыток). В случае минимизации времени отзыва все ошибки могут обрабатываться как критические и вести к завершению работы виртуальной системы.

Процесс не участвует в захвате и освобождении ресурсов, он может их получить от супервизора и вернуть ему. Такой подход позволяет изменять алгоритмы управления ресурсами и сконцентрировать их в супервизоре. Результатом такого подхода является выделение прецедентов «Создание» и «Уничтожение», основной задачей которых, является правильное получение и возврат ресурсов.

Модель состояний процесса приведена на рисунке 2.3.

Существует специальное состояние «Приостановлен», управление которым для простых процессов невозможно со стороны супервизора. Это состояние используется при реализации механизма статического распределения ресурсов.

Всего существует два механизма распределения ресурсов:

Статическое распределение ресурсов – все процессы создаются во время инициализации супервизора, каждый процесс получает необходимые ресурсы. После завершения выполнения задачи процесс не уничтожается, а приостанавливается. При повторном выполнении задачи процесс заново инициализируется. Уничтожение процесса происходит при завершении работы виртуальной системы. Пересоздание процесса во время работы виртуальной системы происходит только после его аварийного завершения, после которого нельзя провести повторную инициализацию. Преимущества статического распределения: при вызове процесса система не тратит время на создание процесса, по завершении выполнения задачи система не тратит время на уничтожение процесса. Недостатки: чем больше процессов используется в системе, тем больше ресурсов потребляет система во время своей работы, время инициализации и завершения работы системы увеличивается пропорционально числу и сложности процессов. Статическое распределение можно применять только для синхронных процессов.

Динамическое распределение ресурсов – все процессы создаются только по факту запроса на выполнение соответствующей им задачи. После окончания выполнения задачи процессы сразу же уничтожаются. Пересоздание процесса происходит при каждом обращении к процессу. Каждому созданию процесса соответствует одна инициализация, повторная инициализация процесса недопустима. Преимущества системы: время инициализации и завершения работы системы постоянны и минимальны, количество ресурсов требуемых системе определяется пиковым значением их использования. Недостатки системы: при обращении к задаче система тратит время на создание процесса, по завершении задачи – на его уничтожение. Динамическое распределение можно применять как для синхронных, так и для асинхронных процессов.

При использовании динамического планирования простой процесс специализируется, образуя три типа процессов:

- Синхронный процесс – сложный процесс, реализующий решение прикладной задачи предметной области. Имеет расширенный интерфейс управления, т.к. может требовать использования асинхронных процессов для своей работы, может выполняться значительное время, в течении которого возникнет потребность в выполнении независимого от данного асинхронного процесса; в многозадачных системах может быть прерван более приоритетным синхронным процессом.
- Простой асинхронный процесс – простой процесс, реализующий выполнение простой системной задачи. Время выполнения такого процесса стремится к нулю. Такой процесс не может быть прерван, т.к. время затраченное на его прерывание может превысить время его выполнения; не может быть приостановлен, т.к. затраты на его повторную инициализацию больше или равны затратам на его повторное создание.
- Сложный асинхронный процесс – процесс, реализующий выполнение сложной системной задачи или простой прикладной задачи. Такой процесс может быть прерван только другим асинхронным процессом и время его выполнения больше, чем время выполнения простого асинхронного процесса, но все равно стремится к нулю. Процесс не может бать приостановлен, т.к. затраты на приостановление и восстановление превышают или равны затратам на уничтожение и повторное создание.

Модель прецедентов управления простым асинхронным процессом приведена на рисунке 2.4. Прецеденты этой модели не отличаются от прецедентов управления простым процессом.

Модель состояний простого асинхронного процесса приведена на рисунке 2.5. для простого асинхронного процесса состояния «Создан» и «Инициализирован» совмещены, т.к. повторная инициализация невозможна. Состояние «Приостановлен» отсутствует, т.к. простой асинхронный процесс всегда уничтожается по завершению выполнения. Для простых асинхронных процессов всегда используется динамическое распределение ресурсов.

Прецеденты управления сложным асинхронным процессом приведены на рисунке 2.6. Сложный асинхронный процесс может быть прерван другим асинхронным процессом, поэтому имеет два дополнительных прецедента для управления:

Внешний прецедент «Прерывание»

Сводка. Прерывает процесс для выполнения другого процесса.

Предусловия. Процесс выполняется.

Постусловия. Процесс прерван и ожидает возобновления выполнения.

Описание:

1. Сохранить контекст выполнения.
2. Прекратить выполнение задачи.

Альтернативы:

1. Если контекст не может быть сохранен или задача прервана, аварийно завершить выполнение процесса.

Внешний прецедент «Возобновление»

Сводка. Возобновляет прерванный процесс.

Предусловия. Процесс прерван и ожидает возобновления.

Постусловия. Процесс выполняется.

Описание:

1. Загрузить сохраненный контекст выполнения.
2. Продолжить выполнение задачи.

Альтернативы:

1. Если контекст не может быть загружен или задача возобновлена, аварийно завершить выполнение процесса.

Сложный асинхронный процесс может применять системный вызов. Системный вызов – запрос со стороны процесса к супервизору на выполнение асинхронного процесса, при этом процесс, который генерирует системный вызов, прерывается до завершения его выполнения. Системный вызов описывается внутренним прецедентом:

Внутренний прецедент «Системный вызов»

Сводка. Вызывает асинхронный процесс для внутренних целей.

Предусловия. Процесс выполняется.

Постусловия. Процесс прерван и ожидает возобновления выполнения по окончании обработки системного вызова.

Описание:

1. Сохранить контекст выполнения.
2. Прекратить выполнение задачи.
3. Сформировать системный вызов.
4. Передать вызов супервизору.

Альтернативы:

1. Если контекст не может быть сохранен или задача прервана, аварийно завершить выполнение процесса.

Модель состояний сложного асинхронного процесса приведена на рисунке 2.7. Как и простой асинхронный процесс, сложный не может быть приостановлен и должен использовать динамическое распределение ресурсов. Состояния «Создан» и «Инициализирован» в нем также совмещены.

Прецеденты управления синхронным процессом приведены на рисунке 2.8. синхронный процесс использует все прецеденты сложного асинхронного процесса, но, кроме того, может быть приостановлен и возобновлен, не только по завершении выполнения, но и сразу после инициализации или во время выполнения. Прецеденты, описывающие внешнее приостановление и возобновление процесса:

Внешний прецедент «Приостанов»

Сводка. Приостанавливает процесс и делает его неактивным.

Предусловия. Процесс инициализирован.

Постусловия. Процесс приостановлен и неактивен.

Описание:

1. Если задача выполняется, прервать выполнение задачи.
2. Очистить контекст выполнения.
3. Блокировать процесс.

Внешний прецедент «Восстановление»

Сводка. Активизирует приостановленный процесс.

Предусловия. Процесс приостановлен.

Постусловия. Процесс инициализирован и готов к выполнению.

Описание:

1. Повторно инициализировать процесс.

2. Разблокировать процесс.

Приостанов и восстановление можно называть блокировкой и разблокировкой процесса. Блокированный процесс не может выполняться, пока не будет разблокирован. Блокирование может использоваться при статическом распределении ресурсов, потому что приостанов может подразумевать освобождение части ресурсов, занятых, например, под контекст выполнения задачи. Для сложных процессов ресурсы, используемые контекстом выполнения, могут быть довольно велики, и их освобождение позволит перераспределить эти ресурсы между другими процессами. Данный механизм расширяет возможности контроля над процессами – все статически загруженные процессы, не использующиеся в данное время и не запланированные на некоторый ближайший промежуток времени, должны блокироваться.

Синхронные процессы могут использовать статическое распределение ресурсов и могут приостанавливаться после нормального завершения выполнения.

Модель состояний синхронного процесса приведена на рисунке 2.9.

2.2.3. Статическая модель процессов

Статическая модель разрабатывается в соответствии с определенным описанием процессов и их моделями прецедентов и состояний.

Базовым классом статической модели служит класс простого процесса. Данный класс является полностью виртуальным и реализовывается тремя специализированными классами, каждый из которых соответствует одному из типов процессов, расширяющих простой процесс.

Интерфейс класса определяется внешними прецедентами управления процессом и соответствует прецедентной модели. Реализация интерфейса предусматривает соответствие модели состояний.

Все специализирующие классы реализуют дополнительный интерфейс, связанный с их специальными прецедентами.

Для связи с супервизором используется специальный сигнальный класс, который специализируется в виде классов, описывающих внутренние прецеденты процессов. Процессы используют объекты этих классов для оповещения супервизора о возникновении внутренних прецедентов.

В виртуальной системе для каждой задачи определяется соответствующий ей класс, после чего реализуется предметная часть задачи и объединяется с реализацией интерфейса класса.

Статическая диаграмма процессов приведена на рисунке 2.10.

2.3. Проектирование супервизора виртуальной системы

2.3.1. Проектирование архитектуры супервизора

Супервизор – есть подсистема виртуальной системы, которая осуществляет координирование работы всей виртуальной системы. Архитектура супервизора определяется возложенными на него задачами:

- Планирование – статическое и динамическое планирование выполнения задач системы. Определение действий по управлению процессами.
- Диспетчеризация – управление ресурсами и поведением процессов виртуальной системы.

Понятия планирования и диспетчеризации используются во всех современных системах реального времени и универсальных системах.

Планировщик может быть разделен на две составляющие, в соответствии с выполняемыми им функциями:

- Статический планировщик, который осуществляет планирование синхронных процессов.
- Динамических планировщик, который осуществляет планирование асинхронных процессов.

Такая декомпозиция обусловлена тем, что алгоритмы статического и динамического планирования обычно сильно различаются, кроме того, планирование обычно выполняется в два этапа: для асинхронных и синхронных процессов отдельно. Выделение динамического планировщика также дает возможность быстрых асинхронных вызовов части супервизора для обработки фактов возникновения запросов на выполнение асинхронных задач. Позволяет проводить диспетчеризацию асинхронных вызовов до синхронного планирования.

Диспетчер процессов факторизуется на две основные части, которые определяются по его назначению:

- Диспетчер объектов процессов – диспетчер базы данных процессов оперирует данными о текущем состоянии процессов и их временных характеристиках, управляет поведением процессов.
- Диспетчер ресурсов – унифицирует распределение ресурсов для различных платформ. Это критически-ориентированная зависимая от платформы часть виртуальной системы.

Архитектура супервизора приведена на рисунке 2.11.

2.3.2. Прецеденты управления и модель состояний супервизора, активация супервизора, прецеденты активации

Прецеденты управления супервизором определяют возможности его взаимодействия с виртуальной системой. Управление супервизором должно быть минимальным. Модель прецедентов управления супервизором приведена на рисунке 2.12.

Внешний прецедент «Создание»

Сводка. Создание супервизора и его подсистем.

Предусловия. Отсутствуют.

Постусловия. Супервизор и его подсистемы созданы.

Описание:

1. Выделить ресурсы подсистемам супервизора.
2. Создать подсистемы.
3. Выделить ресурсы супервизору.
4. Создать супервизор.
5. Передать супервизору управление подсистемами.

Альтернативы:

1. В случае ошибки выделения ресурсов, освободить уже распределенные ресурсы и завершить работу.

Внешний прецедент «Инициализация»

Сводка. Инициализировать подсистемы и подготовить их к работе.

Предусловия. Супервизор создан.

Постусловия. Супервизор готов к работе.

Описание:

1. Инициализировать диспетчер процессов.
2. Инициализировать планировщик.
3. Выполнить активацию.

Внешний прецедент «Уничтожение»

Сводка. Уничтожение супервизора и его подсистем.

Предусловия. Супервизор создан.

Постусловия. Супервизор уничтожен.

Описание:

1. Уничтожить диспетчер процессов.
2. Уничтожить планировщик.
3. Уничтожить супервизор.

Внешний прецедент «Синхронная активация», прецедент «Асинхронная активация»

Сводка. Активация супервизора.

Предусловия. Супервизор готов к работе или выполняется.

Постусловия. Активация выполнена.

Описание:

1. Подготовить данные активации.
2. Выполнить активацию.

Активация супервизора – вызов супервизора для выполнения планирования и диспетчеризации.

Синхронная активация – активация, выполняемая через определенные промежутки времени. Синхронная активация может запрашиваться только виртуальной системой. Период синхронной активации называется периодом квантования.

Асинхронная активация – активация, выполняемая в результате реакции на какое либо событие. Асинхронная активация может запрашиваться как виртуальной системой, так и процессами, выполняемыми в системе. Например, системный вызов из процесса является асинхронной активацией. Активация после инициализации является асинхронной и инициируется самим супервизором.

В соответствии с типом активации, супервизор может быть: синхронным, асинхронным, смешанным.

Каждая подсистема супервизора инициализируется независимо и должна предоставлять соответствующий интерфейс. Для управления супервизором используется специальная подсистема «Координатор супервизора», которая обрабатывает все запросы, связанные с прецедентами супервизора.

Прецеденты активации супервизора показаны на рисунке 2.13. во время активации координатор супервизора выполняет набор действий, которые определяют создание, выполнение и завершение активации. Выполнение активации осуществляется вызовом системного планировщика, при этом никакой связи между координатором супервизора и диспетчером процессов нет. Прецеденты активации супервизора:

Прецедент «Инициализация»

Сводка. Подготовить данные активации.

Предусловия. Отсутствуют.

Постусловия. Супервизор готов к выполнению активации, контекст активации создан.

Описание:

1. Создать контекст активации.
2. Подготовить данные активации.

Прецедент «Планирование»

Сводка. Выполнить планирование.

Предусловие. Активация инициализирована.

Постусловия. Активация выполнена и готова к завершению.

Описание:

1. Передать данные активации планировщику.
2. Ожидать завершения работы планировщика.

Прецедент «Завершение»

Сводка. Завершение активации.

Предусловия. Планировщик завершил работу.

Постусловия. Контекст активации уничтожен.

Описание:

1. Уничтожить контекст активации.
2. Завершить активацию.

Прецедент «Диспетчеризация»

Сводка. Обновить состояние процессов.

Предусловие. Выполнено планирование.

Постусловие. Состояние процессов обновлено.

Описание:

1. Выполнить планирование.
2. Подготовить контекст диспетчеризации.
3. Выполнить диспетчеризацию.
4. Уничтожить контекст диспетчеризации.

Модель состояний супервизора приведена на рисунке 2.14. В случае критической ошибки при выполнении активации происходит аварийное завершение работы супервизора. Такая ошибка является фатальной для всей виртуальной системы и вызовет ее завершение после уничтожения супервизора.

2.3.3. Проектирование планировщиков супервизора

Планирование является одной из наиболее важных частей системы реального времени. Для виртуальных систем правильное и эффективное планирование является основополагающим фактором возможности их работы.

Системный планировщик – подсистема супервизора, которая осуществляет статическое и динамическое планирование работы системы и управляет диспетчеризацией процессов системы.

Системный планировщик функционально делится на два планировщика более низкого уровня:

- Синхронный (статический) планировщик – планирует синхронные процессы, для многозадачной системы включает планирование взаимодействий синхронных процессов.
- Асинхронный (динамический) планировщик – планирует простые и сложные асинхронные процессы, может подразделяться на два планировщика для каждого типа асинхронных процессов в отдельности.

Такая декомпозиция позволяет более ясно разделить различные алгоритмы планирования. Асинхронное планирование более критично по времени, т.к. сами асинхронные процессы выполняются значительно быстрее синхронных. Асинхронное планирование более приоритетно, чем синхронное планирование, т.к. любой процесс, кроме простого асинхронного процесса, может быть прерван асинхронным процессом. В то же время синхронный процесс может быть только прерван, но сам не может прервать работу никакого другого процесса (кроме синхронного процесса с более низким приоритетом в многозадачных виртуальных системах).

Прецеденты системного планировщика приведены на рисунке 2.15. Для супервизора планирование является нераздельным процессом, поэтому существует только один прецедент планирования. Управление отдельными планировщиками входит в компетенцию системного планировщика и не выносится на более высокий уровень абстракции. Описание прецедентов:

Внешний прецедент «Создание»

Сводка. Создание системного планировщика и его составных планировщиков.

Предусловия. Отсутствуют.

Постусловия. Планировщики созданы.

Описание:

1. Выделить ресурсы составным планировщикам.
2. Создать планировщики.
3. Выделить ресурсы системному планировщику.
4. Создать системный планировщик.

Альтернативы:

1. В случае ошибки выделения ресурсов, освободить уже распределенные ресурсы и оповестить супервизор.

Внешний прецедент «Инициализация»

Сводка. Инициализировать системный планировщик.

Предусловия. Системный планировщик создан.

Постусловия. Системный планировщик готов к работе.

Описание:

1. Подготовить базу данных планирования для асинхронного планировщика.
2. Подготовить контекст выполнения алгоритма асинхронного планирования.
3. Подготовить базу данных планирования для синхронного планировщика.
4. Выполнить создание процессов со статическим распределением ресурсов.
5. Подготовить контекст выполнения алгоритма синхронного планирования.

Внешний прецедент «Планирование»

Сводка. Выполнение планирования и изменения состояния процессов.

Предусловия. Системный планировщик инициализирован.

Постусловия. Планирование выполнено.

Описание:

1. Выполнить асинхронное планирование.
2. Если требуется диспетчеризация асинхронных процессов, выполнить ее.
3. Выполнить синхронное планирование.
4. Если требуется диспетчеризация синхронных процессов, выполнить ее.

Внешний прецедент «Уничтожение»

Сводка. Уничтожение системного планировщика.

Предусловия. Системный планировщик создан.

Постусловия. Системный планировщик уничтожен.

Описание:

1. Выполнить уничтожение процессов со статическим выделением ресурсов.
2. Уничтожить составные планировщики.
3. Уничтожить системный планировщик.

На рисунке 2.16 приведена модель состояний планировщика. Планировщик ожидает запроса на планирование, во время которого определяется потребность в диспетчеризации. Планировщик не выполняется диспетчеризацию самостоятельно, он готовит данные о необходимых действиях диспетчера процессов и оповещает системный планировщик о том, что требуется диспетчеризация. Системный планировщик осуществляет диспетчеризацию независимо от своих подсистем, что позволяет централизовать систему управления диспетчером процессов.

На рисунке 2.17 приведена модель состояний системного планировщика. Активация системного планировщика подразумевает запуск составных планировщиков в порядке их приоритета: от асинхронных к синхронным планировщикам и выполнения диспетчеризации по требованию составных планировщиков.

Архитектура системного планировщика имеет следующую особенность: планировщик более низкого приоритета не может быть запущен, пока не завершит работу планировщик более высокого приоритета. То есть сначала буде проведена обработка и диспетчеризация всех асинхронных вызовов и только потом будет запущен синхронный планировщик. Такой подход позволяет избежать конфликта между различными типами задач в случае их конкуренции (например, в многозадачной системе запрос на прерывание одновременно запросили синхронный и асинхронный процесс: запрос синхронного процесса будет отложен до окончания выполнения запроса асинхронного планировщика). Такая система позволяет прервать планирование на любом этапе, то есть появляется возможность проводить полное планирование только при синхронных активациях, а при асинхронных активациях выполнять частичное планирование – только для асинхронных процессов. Возможно проведение редуцированного синхронного планирования – если планировщик более высокого уровня уже выполнил диспетчеризацию, то нет смысла передавать управлении планировщику более низкого уровня, обрабатываемые им процессы заранее имеют более низкий приоритет исполнения.

На рисунке 2.18 приведена статическая модель системного планировщика. Системный планировщик включает один синхронный планировщик и максимум два асинхронных планировщика. Асинхронные планировщики могут отсутствовать, например, в статически планируемых системах. В таком случае статический планировщик может быть интегрирован с системным планировщиком. Асинхронный планировщик может быть один для всех типов асинхронных процессов, либо для каждого типа асинхронных процессов может быть выделен собственный планировщик.

Каждый составной планировщик реализует базовый интерфейс планировщика, который универсален для всех планировщиков, включая системный. Диспетчер процессов может использовать только системный планировщик.

2.3.4. Проектирование диспетчера процессов супервизора

Диспетчер процессов – подсистема супервизора, используемая системным планировщиком для управления состояниями процессов в виртуальной системе. Диспетчер процессов определяет способы управления поведением процессов, а также контролирует распределение ресурсов системы. Диспетчер процессов изолирует управление процессами от всех других подсистем.

Диспетчер процессов декомпозирован на следующие подсистемы:

- Диспетчер объектов процессов – состоит из базы данных объектов процессов и системы управления базой данных. Отвечает за хранение текущего состояния каждого процесса, его характеристик и описания ресурсов, выделенных процессу. Информация о ресурсах хранится до их принудительного освобождения по завершении процесса. Все ресурсы предоставляются процессам в пользование, но диспетчер объектов сохраняет право на управление этими ресурсами. То есть диспетчер делегирует процессам только часть прав на ресурсы – права пользователя ресурсов, оставляя за собой права супервизора ресурсов.
- Диспетчер ресурсов – критически-ориентированная подсистема, предоставляющая унифицированный интерфейс получения, распределения и контроля ресурсов. Диспетчер ресурсов использует сервисы базовой системы для получения и освобождения ресурсов. Этот диспетчер организовывает взаимодействие между диспетчером процессов и процессами.

На рисунке 2.19 представлены прецеденты управления диспетчером процессов. Эти прецеденты представляют собой более высокую абстракцию относительно прецедентов управления процессами, т.к. для системного планировщика отдельные прецеденты управления процессами представляют собой абстракции слишком низкого уровня. Прямое повторение прецедентов управления процессами для диспетчера процессов неприемлемо. Описание прецедентов диспетчера процессов:

Прецедент «Запустить»

Сводка. Запрос на запуск процесса.

Предусловия. Отсутствуют.

Постусловия. Процесс создан и выполняется.

Описание:

1. Подготовить ресурсы для запуска процесса.
2. Создать процесс.
3. Инициализировать процесс.
4. Запустить процесс.

Альтернативы:

1. Сообщить системному планировщику об ошибке.

Прецедент «Переключить»

Сводка. Прерывание текущего процесса и запуск нового.

Предусловия. Процесс выполняется, прерывание допустимо.

Постусловия. Текущий процесс прерван, выполняется новый процесс.

Описание:

1. Прервать текущий процесс.
2. Выделить ресурсы для запускаемого процесса.
3. Создать процесс.
4. Инициализировать процесс.
5. Запустить процесс.

Альтернативы:

1. При ошибке запуска возобновить прерванный процесс и сообщить системному планировщику об ошибке.
2. При ошибке прерывания сообщить системному планировщику об ошибке.

Прецедент «Приостановить»

Сводка. Блокировать процесс.

Предусловия. Процесс инициализирован, выполняется, или нормально завершен.

Постусловия. Процесс приостановлен.

Описание:

1. Приостановить процесс.

Альтернативы:

1. Сообщить системному планировщику об ошибке.

Прецедент «Восстановить»

Сводка. Восстановить процесс.

Предусловия. Процесс приостановлен.

Постусловия. Процесс инициализирован.

Описание:

1. Восстановить процесс.

Альтернативы:

1. Сообщить системному планировщику об ошибке.

Прецедент «Завершить»

Сводка. Завершить текущий процесс, если требуется восстановить прерванный процесс.

Предусловия. Процесс нормально завершил выполнение или выполняется.

Постусловия. Текущий процесс завершен, прерванный им процесс возобновлен.

Описание:

1. Если процесс выполняется, выполнить останов.
2. Если требуется уничтожение, уничтожить процесс.
3. Если имеется прерванный процесс, возобновить его.

Альтернативы:

1. При ошибке завершения сообщить системному планировщику об ошибке.
2. При ошибке возобновления сообщить системному планировщику об ошибке.

На рисунке 2.20 приведена модель состояний диспетчера процессов. При активации диспетчер процессов должен получить данные о требуемом действии диспетчеризации. После чего он выполняет одно из действий.

Модель состояний действия диспетчера процессов приведена на рисунке 2.21. Все действия диспетчера процессов начинаются с контроля входных данных, т.к. выполнение каждого из действий требует определенного состояния процесса, для которого выполняется диспетчеризация. После этого готовится контекст диспетчеризации (например, осуществляется выделение ресурсов или их освобождение). Если контекст подготовлен

удачно, то выполняется собственно действие. Такой подход обеспечивает высокий уровень защиты от сбоев при взаимодействии с процессами.

Для сложных действия, которые оперируют более чем одним процессам, разработаны отдельные модели состояний. На рисунке 2.22 приведена модель состояний для переключения процессов. При переключении кроме прерывания текущего процесса требуется выполнить создание и запуск прерывающего процесса. Поэтому осуществляется дополнительный контроль по правильности создания и запуска. В случае неудачи диспетчер процессов восстанавливает прерванный процесс. Системный планировщик должен оценить последствия неудачного прерывания и в случае, если они не критичны для системы, продолжить работу.

На рисунке 2.23 приведена модель состояний для завершения процесса. При завершении диспетчер контролирует возобновление прерванного процесса. Это позволяет упростить модель прецедентов и в тоже время поддерживать ее целостной. Операция восстановления прерванного процесса по завершении прервавшего его процесса является полностью логичной, поэтому описывается одним прецедентом.

В соответствии с архитектурой, прецедентами управления и моделью состояний спроектирована статическая модель диспетчера процессов, которая приведена на рисунке 2.24. Как и системный планировщик, диспетчер процессов предоставляет интерфейс создания, инициализации и уничтожения. Рассмотрение этих прецедентов нерелевантно после рассмотрения идентичных прецедентов планировщиков и супервизора. Диспетчер процессов предоставляет интерфейс в соответствии с прецедентами управления диспетчеризацией. Диспетчер процессов включает две подсистемы и изолирует их от других подсистем супервизора. Интерфейс этих компонент может изменяться и зависеть от конкретных условий, поэтому спроектирован только минимальный интерфейс для этих подсистем.

3. Реализация виртуальных систем реального времени

3.1. Факторы использования виртуальной системы реального времени

Факторы проектирования и реализации полноценных систем реального времени определяются областью их применения и конкретизируют причины их использования:

1) Экономический фактор – существует большое количество универсальных систем и систем реального времени, которые в полной мере реализуют уровни абстракции от оборудования и дают возможность реализации платформо-независимых программных продуктов. Современные варианты этих систем используют эффективные системы распределения ресурсов и управления ими. Ядра таких систем разработаны с использованием наиболее качественных алгоритмов и представляют собой законченный продукт. Повторная разработка таких систем выгодна только в случае специализированной разработки крупного промышленного продукта. В большинстве случаев использование уже разработанных продуктов гораздо более эффективно. Технология повторного использования является одной из наиболее применяемых в современных продуктах.
2) Фактор документирования – использование существующих систем не только снимает вопрос о повторной разработке, но и вопрос проектирования и документации таких систем. Объем документации возрастает с понижением уровня абстракции, то есть низкоуровневые части продукта, связанные со взаимодействием с оборудованием и управлением ресурсами, потребуют наибольшее количество документации. Может возникнуть ситуация, когда документирование предметной части продукта будет превышать документирование базовой части продукта.
3) Фактор проектирования и реализации – для проектирования и реализации зависимость между уровнем абстракции и затратами сохраняется. Кроме того этап проектирования будет заметно осложнен низкоуровневым проектированием, которое является критически-ориентированной частью системы. Разработка и тестирование этой части поглотит больше времени, чем проектирование и реализация высокоуровневых частей системы. Вопросы переносимости системы в таком случае будут наиболее критичны, потому что создание универсального ядра еще более усложнит низкоуровневую часть системы.

4) Фактор поддержки и распространения – для самих разработчиков на первой ступени разработка и реализация конкретных задач предметной области с использованием собственных систем управления и распределения ресурсов может быть облегчена. Однако дальнейшая поддержка специализированной системы требует все больших затрат. Кроме того, такие системы сложно адаптировать к новым условиям. Распространение таких систем ограничено областью их применения, которая обычно достаточна узка, чтобы реализация системы реального времени не превысила времени потраченного на реализацию прикладных задач предметной области.

Эти факторы показывают основные недостатки использования полноценных систем реального времени для решения задач, допускающих построение виртуальных систем реального времени.

Виртуальные системы должны применяться при наличии следующих условий:

1) Предметная область допускает применение гибкой системы реального времени – это условие, которое должно выполняться в любом случае, варьируется только время отзыва системы.
2) Предметная область не использует специализированного оборудования для систем управления – аспект этого случая состоит в том, что специализация управляемого оборудования значения не имеет, важен только базис самой системы управления, а не чем она управляет.
3) Имеется один или несколько вариантов базовых систем – аспект состоит в наличии систем управления ресурсами, зачастую этот аспект описывается станциями (аппаратно-программное обеспечение, используемое для управления технологическим процессом), использующимися для управления и операционными системами этих станций.

Выполнение всех условий свидетельствует о том, что использование виртуальной системы, скорее всего, будет иметь высокую эффективность по всем факторам. Невыполнение хотя бы одного условия обычно влечет невозможность проектирования и реализации виртуальной системы реального времени для решения задач предметной области.

Исходя из факторов проектирования и реализации полноценных систем и условий использования виртуальных систем, можно выделить факторы эффективности виртуальных систем:

1) Экономическая эффективность таких систем связана с уменьшением объема проектируемого и используемого материала для таких систем. Большинство станции уже будут иметь универсальные или реального времени системы,

которые можно использовать в качестве базовых. Покупка таких систем будет стоить дешевле, чем проектировка с нуля.

2) Документирование – документация виртуальной системы описывает только саму систему, а потому вся документация будет релевантной в рамках всего проекта, что недостижимо при полноценном проектировании. Например, документирование низкоуровневых аппаратных интерфейсов редко является релевантным к предметной области.
3) Проектирование и реализация – виртуальные системы могут проектироваться с разделением предметных задач и системы. При этом предметные задачи могут проектироваться и реализовываться в рамках целевой базовой системы, что способствует их более качественной реализации.
4) Поддержка и распространение – обучение работы с виртуальной системой не требует обучения работы с базовой системой, полноценные системы требуют гораздо более качественного и количественного материала связанного с тематикой самой системы, но не предметной области. Поддержка систем облегчается, т.к. требуется поддержка системы, как таковой, но не новых аппаратных решений.
5) Фактор повышения уровня абстракции систем – отделение виртуальной системы, как детали реализации, от решения задач предметной области. Современные программные и аппаратные решения постоянно совершенствуются, пропорционально чему и усложняются, поэтому технический императив по уменьшению сложности становится все более важным.

Все приведенные факторы показывают, что область применения виртуальных систем достаточно широка. Условия для использования таких систем в большинстве случаев не критичны, а в некоторых случаях наиболее приемлемы.

3.2. Обзор базовых систем для виртуальных систем реального времени

Все базовые системы можно условно поделить на три типа:

1) Однозадачные системы реального времени.
2) Многозадачные системы реального времени.
3) Универсальные многозадачные системы.

Каждый из типов систем имеет свои достоинства и недостатки как базовые системы для виртуальных систем. Основные критерии оценки определяются по факторам использования и составляют следующий контрольный список:

1) Граница времени отзыва, предоставляемая базовой системой.
2) Уровень качественности сервисов по распределению и управлению ресурсами.
3) Уровень абстракции от оборудования.
4) Распространенность системы.

Контрольный список может быть увеличен, но этих факторов достаточно для получения предварительной оценки.

3.2.1. Однозадачные системы реального времени

Однозадачные системы реального времени близки к полноценным системам реального времени и близки по концепции к виртуальным системам реального времени. Основные отличия виртуальных систем реального времени:

1) Улучшенные механизмы планирования и диспетчеризации процессов – виртуальные системы не ориентируются на ресурсы системы при работе супервизора, правильность управления для них может быть более критичной чем правильность исполнения процессов. Эта закономерность усиливается с увеличением времени отзыва виртуальной системы. Данное отличие взаимосвязано с третьим отличием.
2) Улучшенные механизмы контроля выполнения процессов и распределения ресурсов – контроль выполнения процессов и ресурсов выполняется более жестко. Виртуальная система не может допускать прямого управления ресурсами, т.к. все процессы используют одну подсистему управления ресурсов и полностью абстрагированы от их физического и низкоуровневого представления. В полноценной системе реального времени процессы выполняются на уровне ядра системы, в виртуальной системе такой подход не

используется. Ядро системы, которое может взаимодействовать с ресурсами, всегда изолированно.

3) Наличие механизмов коррекции ошибок – для полноценных систем реального времени ошибки работы обычно обозначают критические ошибки оборудования, обработка таких ошибок затруднена, т.к. требует взаимодействия с оборудованием. В виртуальных системах все механизмы абстрагированы и для многих ошибок могут быть предусмотрены обработчики. Обработка ошибок может занимать время, которое допустимо временем отзыва системы.

Однозадачные системы реального времени очень близки к оборудования и зачастую основные компоненты таких систем, включая подсистемы супервизора могут быть редуцированы и частичная их имплементация сопряжена с используемым оборудованием. Примером могут служить асинхронные обработчики прерываний.

Основные достоинства однозадачных систем реального времени:

1) Минимальное время отзыва уровня базовой системы – время требуемое такой системой для обработки запросов виртуальной системы будет очень малым, причины этого в большое специализации таких систем и сильной связности с оборудованием.
2) Простота в использовании – облегчает реализацию критически-ориентированных частей виртуальной системы.

Недостатков у таких систем значительно больше:

1) Низкий уровень абстракции от оборудования.
2) Высокая специализация.
3) Низкий уровень абстракции ядра системы.
4) Сильная вариация различных систем друг от друга, что снижает их распространенность, то есть несмотря на то, что однозадачных систем реального времени много, каждая из них, по сути, уникальна.
5) Сложность проектирования и реализации целевых прикладных задач.

Использование таких систем в качестве базовых допустимо при отсутствии других вариантов.

При проектировании виртуальных систем на базе таких систем требуется:

1) Проектирование и реализация собственного менеджера ресурсов, который будет минимизировать несанкционированное использование ресурсов.

2) Проектирование и реализация виртуального ядра, выделение отдельного концептуального слоя ядра виртуальной системы, которое обеспечивает все подсистемы виртуальной системы своими сервисами.
3) Использование механизмов базовой системы, а не аппаратных механизмов, несмотря на высокую консолидацию систем реального времени и оборудования следует использовать наиболее высокий из доступных уровней абстракции от оборудования.

3.2.2. Многозадачные системы реального времени

Эти системы являются наиболее приемлемыми кандидатами в качестве базовых систем. Такие системы имеют хорошую абстракцию от оборудования, удобные сервисы управления ресурсами и средства для проектирования и построения процессов прикладных задач.

Эти системы имеют собственные супервизоры, что облегчает взаимодействие с ними. Системы драйверов также помогают в реализации виртуальных систем.

Основные достоинства таких систем:

1) Среднее или низкое время отзыва – системы реального времени проектируются для задач, для которых факторизация времени и уменьшение времени отклика являются базовыми требования, а потому время отзыва всей системы максимально уменьшается. Но время отзыва в таких системах может превышать время отзыва в однозадачных системах реального времени.
2) Средний или высокий уровень абстракции от оборудования – наличие развитых сервисов контроля и управления ресурсами, систем драйверов и т.д. облегчает построение виртуальных систем.
3) Средний или высокий уровень абстракции ядра системы – дает возможность эффективной виртуализации ресурсов, предоставляемых системой, и облегчает проектирование подсистем контроля и распределения ресурсов.
4) Распространенность таких систем достаточно широка и гораздо шире, чем однозадачных систем, хотя они и остаются специализированными системами.

Основные недостатки таких систем:

1) Специализация системы – большинство таких систем предназначены для определенного, пусть и широкого, класса задач, например, мультимедийные системы.
2) Требование специальных навыков по проектированию прикладных задач – зачастую такие системы имеют собственные средства проектирования и реализации, что может усложнять реализацию предметной части проекта.

Из всего сказанного следует, что многозадачные системы реального времени является наиболее сбалансированными и оптимальными для использования их в качестве базовых.

3.2.3. Универсальные многозадачные системы

Самый первый фактор для их использования – экономическая целесообразность: системы широко распространены, активно поддерживаются, имеется большое количество квалифицированного персонала для работы с ними и т.д.

Основные преимущества таких систем:

1) Высокий уровень абстракции от оборудования.
2) Высокий уровень абстракции ядра.
3) Качественные сервисы для распределения и контроля ресурсов.
4) Высокая переносимость между различными платформами и универсальными системами.
5) Высокая распространенность.

Данный список можно продолжать, но все эти системы обладают существенным недостатком, который противоречит первому условию использования виртуальных систем: большое время отзыва, которые увеличивается при увеличении универсальности многозадачной системы.

Нейтрализация этого фактора делает универсальные системы лучшими кандидатами на качество базовой системы.

3.3. Основные методы минимизации времени отзыва для многозадачных систем

И универсальные и системы реального времени требуют минимизации времени отзыва даже в случае, если оно изначально удовлетворяет требованиям виртуальной системы. Запас времени отзыва будет оправдан при расширении виртуальной системы.

Описываются только методы минимизации, оценка их эффективности есть следующий этап данной работы:

1) Использование систем приоритетов – виртуальная система и ее процессы должны получать максимальные приоритеты в рамках базовой системы, что обеспечит максимальную скорость обработки их запросов к базовой системе. Обычно речь идет о приоритетах реального времени, которые предоставляют большинство многозадачных систем. Если базовая система предусматривает градацию приоритетов реального времени, то супервизор должен получать максимальный приоритет, асинхронные процессы средний приоритет, а синхронные процессы минимальный, но все приоритеты должны быть реального времени.
2) Использование систем драйверов – погружение всех простых асинхронных процессов в драйвера, или в специализированный драйвер. Драйвера обычно используют режим ядра и имеют максимальную скорость исполнения. Простые асинхронные процессы в таком случае будут максимально соответствовать своей роли, и оказывать минимальное влияние на работу виртуальной системы. Минусом такого подхода является высокая связность с базовой системой и выделение подсистемы управления асинхронными процессами, погруженными в драйвера. Эта подсистема может быть частью подсистемы управления ресурсами.
3) Использование невыгружаемых пулов памяти и захват ресурсов – виртуальная система должна использовать все средства базовой для захвата максимального количества ресурсов и их блокировки на время своей работы. Невыгружаемые пулы памяти позволяют сократить время работы с памятью и отключить механизмы кэширования данных на диск, которые замедляют работу виртуальной системы подкачками страниц с диска.
4) Блокирование процессов, не относящихся к виртуальной системе (разгрузка системы) – виртуальная система или внешний модуль контролируют работу

процессов в многозадачной системе и блокируют их, что позволяет освободить ресурсы для виртуальной системы.

5) Минимизация использования ядра – замена библиотек ядра на собственные, что снизит количество переключений контекста и может значительно повысить производительность виртуальной системы.
6) Распределение виртуальной системы – виртуальная система может быть распределенной, в частности, супервизор может занимать отдельный процессор или разделять его с асинхронными процессами. Выделение синхронным задачам отдельного вычислительного ресурса значительно уменьшит время отзыва.

Реализация всех этих методов максимально минимизирует время отзыва системы и максимизирует ее производительность. Реализация всех методик одновременно довольно дорогостоящий процесс, поэтому имеет смысл выбрать наиболее подходящие. В дальнейшем предполагается более тщательный анализ каждой из методик и оценка их.

Заключение.

В работе рассмотрены системы реального времени и области их применения. Введено понятие виртуальной системы реального времени и исследованы возможности применения таких систем, проектирования и реализации таких систем.

Исследование вопросов проектирования виртуальных систем реального времени проведено в полном объеме. Рассмотрена архитектура виртуальной системы и проведена ее декомпозиция. Полностью рассмотрено проектирование процессов для решения задач прикладной области. Рассмотрена архитектура супервизора и проведена ее декомпозиция. Проведено проектирование всех элементов супервизора с указанием связей компонент. Рассмотрены статические модели для каждого компонента виртуальной системе. Проектирование документировано с использованием UML нотации.

Рассмотрены факторы и условия применения виртуальных систем реального времени. Проведен анализ различных вариантов базовых систем. В частности экономический фактор в данном вопросе может быть вынесен на первый план, т.к. виртуализация в первую очередь предназначена для правильного распределения ресурсов по проектированию и реализации. Научный фактор (проектирование и реализация виртуальных систем, повышение уровня абстракции систем и их универсализация) интересен в первую очередь с точки зрения исследования процессов проектирования и использования таких систем. Вопросы абстрагирования и распределения систем выносятся сейчас на первый план в первую очередь для снижения сложности крупных проектов. Кроме того, существует теория о том, что любая проблема разрешима более эффективно при перенесении ее на более высокий уровень абстракции.

Приведены преимущества и недостатки выбора различных базовых систем. Проведен анализ методов оптимизации использования многозадачных систем в качестве базовых.

Работа в полной мере раскрывает поставленную задачу и может быть использована в качестве базовой для работ по реализации прикладных виртуальных систем реального времени и исследования различных методов построения виртуальных систем реального времени на базе многозадачных систем.

Список литературы:

1. Таненбаум Э. Современные операционные системы, 2-е издание. / СПб.: Питер, 2002, 1040 с.
2. Гома Х. UML. Проектирование систем реального времени, параллельных и распределенных приложений / ДМК Пресс, 2002, 704 с.
3. Буч Г., Якобсон А., Рамбо Дж. Справочник UML, 2-е издание. / СПб.: Питер, 2005, 736 с.
4. Макконнелл С. Совершенный код, 2-е издание. / СПб.: Питер, 2005, 896 с.