МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ

А.А. Абидов

ФУНКЦИОНАЛЬНЫЕ АСПЕКТЫ АДАПТАЦИИ, МОДЕЛИРОВАНИЯ И АЛГОРИТМИЗАЦИИ НАДЕЖНОГО ФУНКЦИОНИРОВАНИЯ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Монография

Ташкент – 2022

Абидов А.А., Функциональные аспекты адаптации, моделирования и алгоритмизации надежного функционирования систем реального времени. Монография. – Т.: ТДИУ, 2022. – 166 с.

В условиях формирования цифровой экономики одним из важных факторов успешного внедрения инновационных технологий в деятельность экономических объектов является бесперебойная работа компьютерных сетей. В этой связи, вопросы обеспечения высокой помехоустойчивости, надежности и эффективности работы всех компонентов компьютерных сетей приобретают первостепенное значение.

Масштаб современных информационных потоков требует четкой работы множества технических устройств и в частности узлов коммутации. В этой связи настоящее исследование является актуальным.

В монографии в логической последовательности представлены методы, средства, необходимые для эффективного функционирования коммуникационных систем. Кроме того, проанализированы возможные причины ошибок и отказов работы систем. По результатам анализа автором разработаны модели, алгоритмы, способствующие надежности функционирования сложных коммуникационных систем в режиме реального времени.

Особую ценность монографического исследования имеют авторские имитационные модели процесса функционирования систем реального времени в условиях возмущающего воздействия среды.

Настоящее издание снабжено интересным графическим и табличным материалом.

Монография может быть рекомендована специалистам в области применения информационных систем и технологий, разработки и эксплуатации компьютерных сетей, студентам, магистрантам и докторантам, изучающим вопросы внедрения и эксплуатации современных информационных систем и технологий в деятельность экономических объектов, а также принципы и методы информационной безопасности систем передачи данных в условиях цифровизации экономики.

Ответ. редактор: д.э.н., проф. Жуковская И.Е.

Реценценты: д.э.н., проф. Кучкаров Т.С.
д.э.н., проф. Шермухамедов А.Т.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В настоящее время одной из основных задач при создании вычислительных сетей является обеспечение высокой помехоустойчивости, надежности, живучести как сети в целом, так и ее отдельных элементов в условиях сбоев и отказов в системах передачи и обработки информации.

В узлах коммутации (УК), являющихся одним из основных элементов передачи, хранения и обработки информации в вычислительных сетях, обеспечение надежности приобретает особо важное значение.

До настоящего времени, вопросы повышения эксплуатационной надежности для узлов коммутации изучены в недостаточной степени, а для узлов адаптивной коммутации (АК) не рассмотрены вообще. Узел АК, обеспечивающий работу в режиме коммутации каналов (КК) и коммутации пакетов (КП), состоит из сложного программного обеспечения, функционирующего в системе реального времени в условиях сбоев и отказов отдельных блоков. В этой связи вопросы контроля и восстановления нормального состояния узла являются жизненно важными.

Исходя из вышеизложенного, актуальность темы данной работы заключается в решении практически важной задачи эффективного управления узлом адаптивной коммутации путем использования методов повышения надежности программного обеспечения.

Целью данной работы является разработка эффективных алгоритмов управления структурами данных узла адаптивной коммутации в условиях неопределенности влияния возмущающего воздействия среды на объект управления, программная реализация разработанных алгоритмов в виде стандартных программ и выработка рекомендаций по их практическому применению.

Исходя из этого, ставятся следующие задачи исследования:

- анализ специфических особенностей узла АК как объекта управления со сложной структурой;

- определение типов критических ошибок, возникающих в области данных узла АН из-за отказов технических средств ЭВМ;

- разработка аффективных методов обнаружения и устранения критических ошибок в области данных узла АК;

- проведение экспериментальной работы для исследования эффективности предлагаемых средств контроля и восстановления структур данных узла АК.

Методологическую основу работы составляют методы теории надежности программного обеспечения, теории вычислительных сетей, теории адаптивного управления. В работе также использованы методы теории вероятностей, имитационного моделирования, теории массового обслуживания и системного программирования.

Инновационный подход данной работы заключается в следующем:

- решена задача контроля и управления структур данных узла АК, основанная на условном разбиении данных на группы в зависимости от характера их изменения, в условиях отказов технических средств ЭВМ, влияющих на готовность и эффективность работы узла при функционировании последнего в системе реального времени;

- определены типы критических ошибок, приводящие к снижению пропускной способности узла или его отказу, являющиеся одной из причин процедурных ошибок вычислительных сетей.

- разработана модель адаптации, выполняющая автоматическое изменение параметров управления области данных узла адаптивной коммутации в зависимости от условий среды;

- создана имитационная модель по определению степени влияния возмущений (неисправностей технических средств вычислительной техники) на эффективность управления потоками данных в узле АК;

- создана аналитическая модель оценки совместной работы узла АК со средствами контроля и восстановления в условиях различной интенсивности

сбоев и отказов при разных загрузках сети ЭВМ.

Практическая ценность данной работы.

Применение предлагаемого адаптивного алгоритма управления обработкой данных позволяет повысить эффективность узла АК в условиях возмущений среды. Вместе с тем, разработанный комплекс программ на основе методов повышения надежности узла представлены в стандартной форме и может быть использован:

- для уменьшения числа отказов в узле и увеличении времени наработки на отказ;
- для повышения вероятности безотказной работы узла АК;
- для определения готовности и пропускной способности работы вычислительной системы узла.

Основные результаты работы обсуждались на республиканской конференции: "Методологические и прикладные аспекты систем автоматизированного проектирования и управления в отраслях народного хозяйства" (Ташкент,1985,1987), на всесоюзной школе семинаре по вычислительным сетям (Одесса, 1987, Алма-Ата, 1988), на научных семинарах лабораторий "Вычислительные системы", "Информационные системы", "Большие системы" ИК с ВЦ НПО "Кибернетика" АН УзССР (1985,1988,1989), на научном семинаре лаборатории "Информационные проблемы вычислительных сетей" Научного совета "Кибернетика" АН СССР.

По теме данной работы опубликовано девять научных работ. В том числе I пакет программ сдан в ГОСФАП АН СССР (№ 5086000627 от 07.02.86).

СОДЕРЖАНИЕ РАБОТЫ

Во введении обосновывается актуальность темы, формулируется цель работы и защищаемые положения.

В первой главе монографии дана общая характеристика состояния проблемы, рассмотрены особенности и специфические свойства узла адаптивной коммутации как объекта управления, функционирующего в вычислительной системе (ВС) в реальном времени (р.в.) в условиях возмущающего воздействия среды на его работу.

Анализируются возможные причины ошибок в таких системах. Произведен обзор методов повышения надежности в системах реального времени. Для обеспечения эффективного управления узлом АК поставлена задача создания модели адаптации структур данных к различным возмущениям.

Во второй главе приведены результаты исследований возможности повышения надежной работы узла АК путем эффективного управления структурами данных. Выявляются и классифицируются типы критических ошибок, приводящие к снижению пропускной способности узла и к отказу программного обеспечения. Для обнаружения и устранения критических ошибок разрабатываются методы контроля и восстановления, на основе которых составлены алгоритмы и программы. Управление средствами контроля и восстановления осуществляется моделью адаптации структур данных.

В третьей главе приводится аналитическая модель оценки надежной работы узла АК в ВС р.в. в условиях сбоев и отказов технических средств. Модель построена на базе непрерывной цепи Маркова для СМО с ограниченной очередью. Приводятся алгоритмы и описание программы расчета характеристик СМО типа M/M/1 на базе предложенной аналитической модели.

В четвертой главе описываются экспериментальная работа по определению степени влияния возмущений (неисправностей технических средств вычислительной техники) на эффективность управления потоками

данных в узле АК. Проверяется целесообразность данных узла АК и использования методов повышения надежности ПО ВС. Конкретно имитирован процесс функционирования блока БКП с общей областью памяти структур данных узла АК в условиях сбоев и отказов.

ГЛАВА I. ОСНОВНЫЕ НАПРАВЛЕНИЯ ОБЕСПЕЧЕНИЯ НАДЕЖНОГО ФУНКЦИОНИРОВАНИЯ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

В данном разделе описывается принцип работы узла с одним из методов гибридной коммутации — адаптивной коммутации, как техническая система управления со сложной структурой. Показана логическая связь элементов физической и программной структуры узла. Дается структура программного обеспечения (ПО), функционирующего в системе реального времени (р.в.). На основе возможных причин ошибок, возникающих в вычислительной системе, определяются основные факторы изменения внешней среды, влияющие на эффективность функционирования узла АК. Приводится анализ моделей надежности ПО оценки работы узла АК. Для качественного управления потоками данных в узле АК формализуется задача адаптации структур данных. Выбираются показатели оценки эффективности работы узла АК.

1.1. Актуальность проблемы контроля и эффективного управления систем реального времени

Вычислительные сети считаются сложными информационными системами [5,71], состоящие из: базовой сети передачи данных (СПД), узлов коммутации (УК), каналов (трактов) связи, терминальных комплексов, средств вычислительной техники и др. Каждая из которых представляет отдельную непростую по сложности систему.

Для анализа и синтеза таких систем есть только один путь — расчленение их на разнообразные элементы и исследования множества структур их взаимодействия [71]. Эти проблемы отражены в новом научном направлении, получившем название архитектуры систем управления. И в зависимости от объекта исследования она рассматривает архитектуру вычислительной сети, вычислительной машины либо полупроводникового кристалла (интегральной схемы).

Архитектура системы является емким понятием, включающим три

важнейших вида взаимосвязанных структур: физическую, логическую и программную. Каждая из этих структур определяется набором элементов и характером их взаимодействия. Связь структур друг с другом образует архитектуру рассматриваемой системы. Элементами физической структуры являются технические объекты, элементами логической структуры (логическими модулями) служат операционная система ЭВМ, терминальный модуль, группа коммуникационных программ. Их назначение сводится к функциям определяющим основные операции ввода, хранения, передачи, обработки или выдачи массивов информации.

В большинстве случаев система характеризуется чрезвычайно сложным многоцелевым программным обеспечением, осуществляющим различные виды информационных работ. Поэтому очень важной характеристикой архитектуры системы является также ее программная структура. Эту структуру образуют взаимосвязанные программы процесса обработки информации, доступ к этому процессу, диагностика неисправностей оборудования, передача информации, управление каналом и т.д.

Таким образом, архитектура системы является концепцией взаимосвязи большого числа различного типа элементов. Она в основном характеризуется переплетением физической, логической и программной структур этой системы. Это переплетение определяется размещением логических модулей в вычислительных машинах, синтезом этих модулей из наборов взаимосвязанных программ, реализацией программ в полупроводниковых кристаллах, контроллерах, аппаратуре передачи данных и т.д.

Перейдем к описанию узла адаптивной коммутации (АК).

Многие ВС строятся на базе метода коммутации пакетов [11,14,71,72]. Однако в последние годы уделяется весьма пристальное внимание созданию сетей с гибридными методами коммутации, в которых объединяются возможности нескольких методов коммутации: каналов, сообщений, пакетов. При этом гибридная коммутации, совмещающие методы коммутации каналов

(КК) и пакетов (КП), позволяют эффективно передавать смешанный трафик. К указанным методам относятся: гибридная коммутация с фиксированным порогом, гибридная коммутация с плавающим порогом и адаптивная коммутация.

Метод адаптивной коммутации, позволяющий динамически перераспределять пропускную способность тракта между КК и КП в зависимости от сетевого трафика, является наиболее эффективным по сравнению с остальными методами гибридной коммутации [57,59].

В отличие от гибридной коммутации с фиксированным порогом при адаптивной коммутации жесткий порог не устанавливается [56,60,62]. При этом пороги, в отличие от гибридной коммутации с плавающим порогом, когда смешение порога происходит в сторону КК, могут смещаться в обе стороны, двустороннее смешение границы зон КК и КП обеспечивается введением трех режимов передачи информации: КК высокого приоритета, КК низкого приоритета и КП. Кадр адаптивной коммутации условно разделен на три зоны: зона КК, в пределах которой располагаются все существующие в данный момент КК; зона кадра, временно занимаемая пакетами при отсутствии КК; зона кадра, используемая всегда в режиме КП [58].

На основе вышеизложенных суждений, архитектуру узла АК можно представить элементами физической, логической и программной структуры. Элементами физической структуры для узла АК служат: мультиплексоры, модемы, адаптеры, средства вычислительной машины (оперативная память, процессор, каналы ввод-вывода и др.). Логическую и программную структуру (здесь и в дальнейшем элементы этих двух структур рассматриваются вместе) составляют в основном программные изделия: операционная система узла АК, программы управления потоками данных, программы логической связи процесса обработки, передачи информации с физическим устройством, структуры данных хранения и логической связи между ними и др.

Для установления связи между элементами физической и программной

структуры достаточно рассмотреть общий алгоритм функционирования узла АК для обработки одного кадра (см.рис.1.1.). Этот алгоритм заключается в следующем. Кадры принимаются и разбираются блоком разборки кадров (БРК), данные КК, содержащиеся в кадре, обрабатываются в этом же блоке, а пакеты, содержащиеся в кадре, передаются в БКП. БКП сортирует эти пакеты и либо передает их в БКК, если пакет служебный и предназначен для организации или ликвидации каналов КК, либо обрабатывает их сам, если пакет информационный.

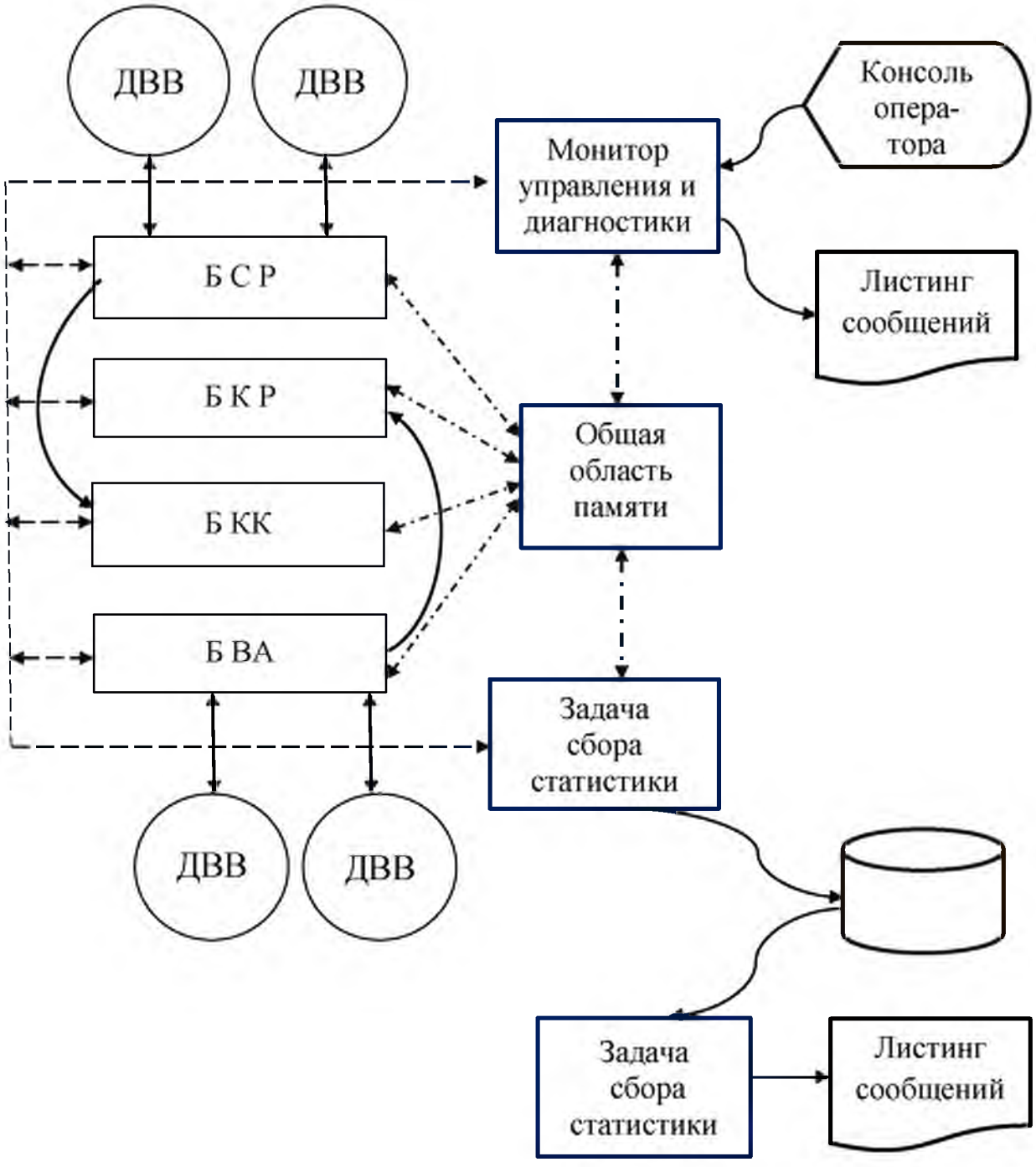
БКК после организации или ликвидации КК извещает об этом БРК и блок сборки кадров (БСК). Блоки БСК и БРК реализуются в виде отдельной задачи БСР. Если абонент находится в данном узле БКП отправляет пакет к БВА. Если пакет транзитный, блок БКП передает пакет в БСК. Блок БСК формирует кадры из транзитных пакетов и передает далее по сети. Данные КК или пакеты от абонентов принимает БВА. Данные КК затем направляются от БВА к блоку БСК, который формирует из этих данных кадры для передачи к следующему узлу. Пакеты БВА сразу направляет к БКП. Дальнейшие действия блока БКП по обработке этих пакетов описаны выше.

Прием и передача кадров, пакетов и данных КК осуществляются через драйверы ввода-вывода (ДВВ). ДВВ осуществляют логическую связь между адаптером и оперативной памятью ЭВМ.

В узле АК каждый тракт и абонентская линия дуплексные, прием и передача проводятся одновременно.

В общем алгоритме функционирования узла АК, выполнение каждой функции каким-либо блоком осуществляется с использованием данных узла АК. Различаются два типа данных: внешние, внутренние.

Внешними данными ПО узла АК служат пакеты, кадри, данные КК.



- передачи управления с помощью флагов событий и директив ввода-вывода
- доступ к общей области памяти
- обмен данными с помощью директив ОС РВ
- использование файловой системы ОС РВ

Рис.1.1. Структура ПО узла АК

К внутренним данным ПО узла АК относятся следующие структуры (таблицы, контексты, пула, управляющие элементы очередей, компенсационные буфера):

- управляющая таблица MCT;
- таблица основных данных TABPARAM;
- таблица блоков узла АК;
- контексты трактов, линий TCT, LCT;
- таблицы адресов контекстов трактов, линий TRUT, LINT;
- маршрутные таблицы режима КП, КК PSRT,CSRT;
- контексты входных, выходных КК в трактах ZICSTB, ZOCSTB;
- контексты КК в абонентских линиях ACSTB;
- управляющие элементы очередей, построенные по дисциплине FIFO: PNPQE, PLPQE - очереди к блоку БКП от трактов, линий; ZOTPQE, ZITPQE - в контекстах тракта; APQE - в контекстах линий;
- управляющие элементы пулов, построенные по дисциплине LIFO: PNPQE, PLPQE - пулов свободных длинных и коротких буферов.

В работах [58,61] описаны взаимосвязь основных блоков узла АК с внутренними данными. В табл.1.1 приведены блоки ПО узла АК и указаны данные с которыми они взаимодействуют (взаимосвязь блоков узла АК с внешними данными изложена выше). Как видно из табл.1.1, каждый блок узла АК взаимодействует не менее чем с 6-ю структурами внутренних данных узла. Например, блок БКП взаимодействует с 7-ю структурами. Это означает, что при пропускной способности 40 пак/с БКП обращается к 7 структурам внутренних данных узла АК 40 раз в секунду. Такое число может составить более 140 тыс. обращений в оперативную память ЭВМ (в течении одного часа работы блока БКП), так как все внутренние данные хранятся в общей области памяти (ООП) ЭВМ.

Таблица 1.1.

Взаимосвязь основных блоков с компонентами данных узла АК

№	Наименование блока	Выполняемые функции	Используемые компоненты ООП внутренных данных
1	БСК	Выборка знаков КК из КМРВ абоненских линий; изъятие пакетов из очередей TPQE выходного тракта; формирование(сборка) кадра АК; передача кадра АК в ДВВ	MCT, TRUT, KMPB, ZOTPQE, ZOCSTB, ZOKADR
2	БРК	Выделение из кадра данных транзитных КК и запись их буфера для кадров соответствующих трактов; выделение из кадра данных местных КК и запись их абонентские КМРВ; выделение из кадра пакетовб запись их в буфера в очередь к БКП или БСК	MCT, TRUT, KMPB, ZOKADR ZICSTB, PNPQE, CNPQE
3	БВА	Прием и передача кадров от/к абонентов КП; прием и передача данных КК от/к абонентов КК и БСК; прием и передача пакетов от/к БКП и БКК.	MCT, LINT, ACSTB ZICSTB, ZOCSTB
4	БКП	Прием пакетов от БВА и БРК; определение корректности данных в пакете; обработка пакета согласно выбранной процедуре протокола X. 25; определение маршрута пакета; постановка пакета в соответсву-ющую выходную очередь (к тракту или абонетской линии)	MCT, TRUT, LINT, PSRT, CSRT, TPQE, PULBIG, PLPQE TABPARAM, PULLTL, PNPQE
5	БКК	Организация и соединение в режиме КК; изъятие пакетов с очереди CPQE и вставление их после обработки либо LPQE или PNPQE	CPQE, LPQE, PNPQE, MCT, TRUT, LINT, ZICSTB, ZOCSTB

Все это показывает, что архитектура узла АК характеризуется переплетением элементов физической, логической и программной структуры. Эти переплетения характерны для оперативной памяти ЭВМ со структурами внутренних данных узла АК и программами процесса обработки, хранения, передачи данных, средств связи с драйвером ввод-вывода, программ управления сетью с установлением физической связи между узлом коммутации и трактами передачи данных и др.

Следственно, например, когда выполняются действия над структурами

внутренних данных узла АК задействована оперативная память и процессор ЭВМ.

Таким образом узел АК, как часть вычислительной сети, можно рассмотреть в виде системы управления, где все действия над процессами управляются программно. В связи с этим элементами узла АК могут служить операционная система узла АК, драйверы ввод-вывода, прикладные программы процесса обработки, хранения приема и передачи информации, программы управления сетью, область внутренних данных узла АК, внешние данные и т.д. Каждая из них в свою очередь может составить отдельную систему.

Узел АК можно отнести по изменению состояния — к динамическим системам (состояние изменяется во времени); по характеру функционирования — к стохастическим системам (т.к. нельзя однозначно судить о ее функционировании); по степени сложности структуры — к сложным системам; по виду элементов — к системам типа "процесс" (элементами служат операции, например обработка, хранение, прием, передача и др.); по связям с окружением — к открытым системам (имеются вход и выход).

Сеть с адаптивной коммутацией по сравнению с другими методами коммутации функционирует в р.в. с целью обеспечения коммутации каналов.

Как известно, в системах р.в. процессы происходят непрерывно, всякая неисправность (сбой, отказ) может существенно повлиять на готовность и пропускную способность исследуемого процесса. В целом, для информационных систем, к которым относятся сети передачи данных, вычислительные сети и др., рассматриваемое время для восстановления при отказах ограничено.

Учитывая особенности реализации ПО узла АК в р.в., необходимо:

- проанализировать влияние ошибок в области данных узла АК, вызванных сбоями и отказами устройств ЭВМ, на процесс функционирования узла;

- осуществить анализ существующих методов обнаружения и

восстановления с целью выявления наиболее приемлемых из них для качественного управления потоками данных узла АК;

- проанализировать существующие показатели эффективности ПО систем р.в., обосновать и выбрать их для оценки работы узла АК;
- оценить надежность ПО узла АК с учетом ошибок, возникающих в вычислительных системах.

1.2. Особенности функционирования систем реального времени при влиянии неисправности технических средств и модели оценки их надежности

К основным понятиям и определениям теории надежности [3,26,30] относятся:

Работоспособность – способность объекта, при котором он способен выполнять заданные функции с параметрами, установленными требованиями технической документации;

-отказ – событие, заключающееся в нарушении работоспособности;

-восстановление – событие перехода системы (объекта) из неработоспособного состояния в работоспособного в результате устранения отказа;

-самоустраняющий отказ (сбой) – событие, характеризующееся достаточно быстрым восстановлением работоспособности без внешнего вмешательства;

-безотказность – свойства объекта непрерывно сохранять работоспособность в течение некоторой наработки или в течении некоторого времени;

-неисправность – состояние объекта, при котором он не соответствует хотя бы одному из требований, установленных нормативно-технической документацией;

-надежность – свойство выполнять заданные функции, сохраняя во времени значения установленных эксплуатационных показателей в заданных пределах, соответствующих заданным режимам и условиям использования.

Приведенные определения и понятия заимствованы из теории надежности, и они в целом приняты в теории надежного программного обеспечения (НПО). Единственное отличие заключается в понятиях "сбой" и "отказ", которые в теории НПО классифицируются по длительности восстановления [31]. При этом устанавливается пороговое время разделения. Искажение считается сбоем в случае восстановления за время, меньшее порогового времени, в противном случае оно считается отказом. Выбор времени порога в общем случае зависит от максимально требуемого времени устранения одного отказа. Для каждой исследуемой системы выбирается свое время порога [31]. Например, для управления на борту самолета пороговое время выбирается менее 1с, то для управления информационно-справочными системами в диапазоне оно составляет 10–100с.

В работах [31,85,93,95] приведены результаты исследования с целью выявления причин ошибок, возникающих при функционировании ПО систем р.м.в. Результаты этих исследований сводятся к выделению трех видов факторов, влияющих на ПО: динамики исполнения программ, структуры построения программ, качества контроля и восстановления программ.

Рассмотрим причины неисправностей, связанные с I-ой группой факторов [31]:

- искажения исходной информации, поступающей от внешних абонентов;
- сбои в аппаратуре вычислительной системы;
- не выявленные ошибки в комплексе программ.

Искажение данных, поступающих от внешних абонентов, обусловлены:

- искажением данных на первичных носителях информации;
- сбоями и частичными отказами в аппаратуре ввода, приема и передачи

телекодовой информации;

– потерями или искажениями сообщений в ограниченных буферных накопителях вычислительной системы;

- ошибками в документах, используемых для подготовки данных.

Из-за сбоев в аппаратуре вычислительной системы средняя наработка на отказ для однопроцессорных ЭВМ измеряется сотнями часов, значительно чаще происходят сбои (в среднем один раз в час) [3I]. Большинство сбоев выявляются и устраняются средствами аппаратного контроля, но до обнаружения этих сбоев они могут искажать программное обеспечение систем р.м.в. При этом некоторая часть сбоев совсем не обнаруживается. Такие сбои проявляются в случайные моменты времени и практически невозможно наблюдать их повторение. Вследствие этого их регистрация и изучение не осуществляются. Эти сбои вызывают многие необоснованные зацикливания и остановы в программах. Необходимость учета сбоев констатируется в работе [99], где сбои составили 90% неисправностей вычислительной системы.

В целом, считается, что при среднем быстродействии ЭВМ (100 тыс.опер/с) произвольная операция искажается с вероятностью около$10^{-7} – 10^{-8}$.

На основе этих данных можно определить причины ошибок, влияющих на надежность функционирования ПО- узла АК:

- невыделенные ошибки в ПО узла АК;

- искажение внешних данных из-за неисправностей в трактах и линиях связи;

- искажения в программных кодах ПО узла АК;

-искажение внутренних данных ПО узла АК из-за сбоев в аппаратуре ВС (оперативной памяти, процессоре).

Первая причина сбоев и отказов выявляется и устраняется в стадии отладки или позже, в стадии опытной эксплуатации ПО узла АК. На этой стадии можно также выявлять ошибки в трактах и линиях с помощью тестирования. Последнее предусмотрено в ПО узла АК при начальной загрузке

и в случае уменьшения пропускной способности.

Контроль сохранности программ обеспечивает проверку соответствия записи программ в памяти исходному эталону[29]. Первичный контроль обычно проводится суммированием кодов программы и сравнением полученных сумм с заранее подготовленными контрольными значениями. Для суммирования может применятся различная логика использования команд сложения, однако необходимо, чтобы выявлялись типовые коррелированные ошибки, например, искажения одного и того же разряда и кодах текста программы. Секционирование суммирования по частям программы позволяет повысить достоверность проверки и при наличии искажения локализовать часть программы не соответствующую эталону. В ВС с внешней памятью возможна пословная проверка сохранности программ сравнением с дублями, содержащими те же программы на другом экземпляре внешнего носителя. Сравнение производится при обнаружении искажения контрольных сумм для локализации отклонения текста программы от эталона.

Однако контроль сохранности программ производят оперативно [31], на что требуется достаточное количество времени. Поэтому основу данного исследования составляет анализ ошибок, возникающих в области данных узла АК, хранящихся в оперативной памяти.

В связи с этим необходимо разработать средства, позволяющие устранять ошибки, возникающие во внутренних данных, причинами которых являются сбои в аппаратуре ВС.

В подразделе 1.3 приводится обзор методов обнаружения и устранения искажений в области данных ПО систем р.м.в. Далее приводятся результаты анализа моделей надежности программного обеспечения.

В настоящее время существует большое количество моделей для оценки надежности программного обеспечения. Эти модели позволяют оценить надежность на разных стадиях создания программ.

Модели надежности, используемые на стадии проектирования и отладки,

характеризуются как модели оценки, а на стадии эксплуатации как модели измерения и предсказания (прогнозирования) надежности.

Первые модели − оценки надежности ПО или, как их называют, модели временных интервалов [53,1] основывались на теории надежности и использовали определенные предположения о законе распределения и вероятности появления отказов ПО. Классическими примерами таких моделей служат модели Джелински-Моранды, Шик-Вельвертона, Гоэль-Окимото [30,77,92].

В этих моделях показатели надежности связаны с числом ошибок в программах, оставшихся до, в процессе и после тестирования. Такие показатели могут указать момент окончания тестирования, оценить уровень доверия к ПО и стоимость работ по сопровождению, но это относится лишь к технологии создания ПО, а не к эксплуатационным показателям его функционирования [84].

Модели измерения различаются от вышеуказанных моделей тем, что в них не заложены какие-либо предположения об интенсивности отказов и числа ошибок. Они строятся на твердом статистическом фундаменте. К таким моделям относятся модели Милсса и Нельсона.

Описание модели Милсса приведено в [33], а модели Нельсона с практической реализацией в [64].

В отличие от описанных моделей, Марковская модель [96,97] служит для предсказания надежности ПО [l6]. Марковская модель основана на предположении, что система проходит через последовательности "исправных" и "неисправных" состояний.

Надежность программ вычислительной машины должна быть гарантированной, однако в настоящее время эта надежность не может быть точна измерена [73].

Терриволи [98] указывает, что ни одна из моделей не дает удовлетворительных результатов, поскольку все они полностью игнорируют

функциональные различия между аппаратными средствами и программным обеспечением.

Подход, предложенный Шнейвиндом [91], показывает, что для каждой программы и конкретных условий ее исполнения должна быть своя модель надежности. Шнейвинд и установил, что ошибки в 19 программах не соответствовали какому-либо единственному распределению вероятностей.

Литллвуд считает, что все существующие модели могут служить мерой определения стадии отладки программы, однако пользователя интересует выполнение программы, а не степень ее отлаженности (т.е. число ошибок, оставшихся в этой программе [84]).

Кроме перечисленных моделей существуют и другие. Например, в работах [27,31], предложены модели оценки процесса функционирования на основе непрерывных и дискретных цепей Маркова.

Модель, основанная на непрерывной цепи Маркова, построена для случая, когда известны интенсивности переходов в состояние отказа, восстановления и перехода в нормальное состояние.

Другая модель построена на базе дискретных цепей Маркова и в отличие от предыдущей, требует задания вероятности вышеупомянутых переходов.

В целом, приведенные модели можно использовать для оценки надежности ПО систем р.м.в. на следующих стадиях: отладки ПО, проектирования и эксплуатации.

Однако эти модели неприемлемы для оценки ПО систем массового обслуживания, к которым относятся сетевые задачи. Неприемлемость объясняется тем, что в них не учитывается влияние очередей на показатели эффективности системы. В связи с этим возникает необходимость в разработке аналитической модели для оценки надежности ПО узла АК.

1.3. Анализ методов зашиты информации, обнаружения ошибок, диагностики состояния и восстановления работоспособности систем реального времени

Методы, предназначенные для повышения надежности программного обеспечения, разделяются на две группы: методы предотвращения ошибок и методы, используемые для восстановления ПО после появления ошибки [95]. К первой группе методов относятся: структурное программирование, доказательства правильности программ, верификация критических модулей программ. Ко второй − методы обнаружения, диагностики и восстановления нормального состояния ПО.

В настоящей работе будут рассмотрены вопросы повышения эксплуатационной надежности ПО узла АК, ввиду чего будут проанализированы работы, относящиеся к методам, отнесенным ко второй группе.

Для обеспечения надежного функционирования ПО систем р.м.в. используется избыточность. Существует три вида избыточности: временная, информационная, программная [26,28,85]. Временная избыточность, заключается в наличии резервного времени, превышающего минимально необходимое для выполнения программы с контролем и восстановлением вычислительного процесса в случае возможной ошибки. Как утверждается в [31], резервное время в системе р.м.в. для контроля и восстановления процесса или данных заранее не устанавливается. Это время обеспечивается за счет резерва, либо за счет сокращения времени решения функциональных задач [26,20].

Другой вид избыточности – информационная, состоит в дублировании данных и программ. При этом избыточность вносится как в память, так и во внешнюю память (магнитный диск − МД). В оперативной памяти содержатся копии часто изменяемых данных. Во внешней памяти содержатся копии программ и параметры, которые нельзя дублировать из-за частности изменений

или же большого объема.

Еще один вид избыточности − программная, используется для контроля и обеспечения достоверности наиболее важных решений по управлению и обработке информации. Она заключается в применении нескольких вариантов программ, различающихся методами решения некоторой задачи или программной реализацией одного и того же метода.

Выявление ошибок в ПО обеспечивается за счет контроля. В работах [85,96] описываются три основных подхода к реализации программного контроля.

1. Наблюдение сохранности структур данных в процессе функционирования.

2. Контроль состояния и динамики исполнения систем.

3. Использование некоторых показателей и характеристик для определения правильности функционирования системы р.м.в. Например, когда время отклика, производительность или время выполнения выходят за установленные пределы, считается, что в системе имеет место ошибка. Эти три вида наблюдений можно проводить непрерывно, периодически или в случае возникновения подозрения о наличии ошибки. Средства, позволяющие провести такие наблюдения для повышения надежности ПО систем р.м.в., основываются на методах обнаружения, диагностики и восстановления, которые реализуются в три этапа.

На первом этапе реализуются методы обнаружения ошибок, служащие для повышения надежности ПО систем р.м.в., которые описаны в [29,31,70,85,94, 95]. Эти методы позволяют контролировать:

− состояние и сохранность программ во внутренней и внешней памяти ВС;

−динамику процесса исполнении программ и сохранения логической связи между компонентами комплекса программ при рабочем функционировании;

–состояние и изменения данных в памяти ВС [31].

Контроль состояния и динамики исполнения программ, сохранения логической связи между компонентами комплекса программ при рабочем функционировании производится путем проверки по времени и по предшествованию. Метод проверки по времени основан на установке таймеров для подачи аварийного сигнала по истечении времени, достаточного для выполнения системой своих функций в случае правильного функционирования. Метод контроля по предшествованию (“эстафетных гонок") обеспечивает обнаружение непредусмотренных между подпрограммами переходов из-за ошибок в программах или сбоев аппаратуры. Для этого проверяемая программа в начале своего исполнения записывает в специальную ячейку код, характеризующий ее включение ("эстафетная палочка"). Перед завершением программы проверяется правильность этого кода.

В ПО узла адаптивной коммутации предусмотрено [61] использование метода проверки по времени для передачи и управления пакетами по сети. Метод "эстафетных гонок" можно использовать для контроля последовательности выполнения задач – блоков ПО узла адаптивной коммутации.

Методы обнаружения, служащие для контроля состояния и изменения данных в памяти ВС с целью повышения надежности ПО систем р.м.в. составляют проверки [28,31,85,93];

- по контрольной сумме (МОКС);
- путем сравнения с копией (МОСК);
- по предельным значениям (МОПР);
- по смыканию списка по номенклатуре (МОСС);
- путем поэлементного сравнения текущих и начальных значении (МОЭСР);
- по смыслу (МОСМ);
- по модулю (МОМД).

При реализации метода обнаружения по контрольной сумме каждая программа, изменяющая структуру данных, изменяет и контрольную сумму этой структуры. Затем с помощью программы, осуществляющей проверки, производится сравнение контрольных сумм (КС).

Этот метод позволяет фиксировать ошибку в поле данных, однако он не дает возможности обнаруживать местонахождения ошибки.

Для определения этого параметра более эффективным является метод сравнения с копией, который базируется на информационной избыточности.

В работе [85] указывается ценность метода предельных значений для защиты адресаций поля данных и значений величин, если известен их диапазон изменения.

Метод по смыканию списка заключается в проверке текущего значения параметра со списком возможных его значений. Он не подходит для сетевых задач из-за большого времени, требуемого для сравнения, и достаточно большого объема информационной избыточности. Например, для проверки одного параметра требуется хранить все возможные его значения и время для сравнения с каждым из них.

Метод элементного сравнения текущих и начальных значений, в целом, не отличается от метода МОСК. Разница заключается в типе контролируемого данного и доступа к нему: в одном случае в МОСК контролируется параметр с доступом к нему по адресу; в другом случае массив в МОЭСР с доступом как по адресу, так и по индексу.

Также возможно использование метода МОСМ, заключающего в проверке значения параметров по смыслу.

Еще один метод обнаружения: контроль по модулю, позволяющий производить сравнение адреса поля памяти на кратность размером модуля.

Обобщая возможность методов обнаружения для контроля состояния и изменения данных в памяти можно заметить, что все они используют как информационную, так и временную избыточность.

Следующим этапом повышения надежности в системах р.м.в. можно указать диагностику программного обеспечения, которая определяет степень любого повреждения и вероятную причину сбоя для проведения соответствующих исправлений и полного восстановления [74,85,93].

В основном, в стадии диагностики, пользуются словарем (таблицей) диагностики [85], которая состоит из возможных ошибок, причин их и способов восстановления, применяемых в зависимости от степени сбоя.

В целом диагностику проводят в три этапа [85].

1. Грубая оценка серьезности ошибки и степени ее влияния на элементы системы и обрабатываемые данные.

2. Детальное изучение данных и точная оценка с целью определения полного объема повреждения и выявления причин ошибок,

3. Использования диагностического словаря.

При этом в системах с пакетами прикладных программ [13], когда время останова и исправления ошибок не имеет жесткого ограничения используются первые 2 этапа, а в системах р.м.в, – третий [85].

Такие словари создаются двумя способами. Во-первых, перед запуском системы создают последовательность возможных сбоев ВС и их признаки заносят в эти словари. Во-вторых, после запуска системы возникающие сбои, их признаки и действия по их устранению добавляются в словарь.

На этапе диагностирования в ряде работ [10,18,32,34,42,70,83,92,98] предлагается тестирование (тестовое диагностирование [42]) ПО.

Известны две стратегии тестирования: структурное и функциональное. В первом случае [78,82] не принимается в расчет спецификация программ. Программа рассматривается в качестве "белого ящика" [34], и базируется на логической структуре программы[32]. Согласно второй стратегии программа рассматривается в качестве "черного ящика" [78,79], когда в расчет принимаются исключительно внешняя спецификация программы и сопряжения между модулями. В работе [34] указано о сложности составления тестов для

прохождения по всем путям по первой стратегами, и для всех входных-выходных данных по второй стратегии. Более полный обзор с классификацией стадий, методов, способов, подходов тестирования ПО вычислительных систем (см.рис .1.2) приводится в [32,34,42 ,47,78,82,98].

При двух известных стратегиях тестирования – структурном и функциональном, последний является наиболее предпочтительным для тестирования ПО вычислительных сетей. Функциональное тестирование обеспечивает правильность работы ПО ВС, когда известны данные входных и выходных трактов, при этом по сравнению со структурным тестированием не требуется знание логической структуры программ.

Анализ методов диагностики показывает, что сложность реализации диагностического словаря по сравнению с тестированием намного меньше. Это объясняется тем, что при тестировании на диагностику одной ошибки приходится количество тестов измеряемые десятками, тогда как причины одной неисправности в таблице диагностики будут занимать одну отроку.

На заключительном этапе осуществляется восстановления работоспособности программ. Если в системах разделения времени можно обойтись обнаружением и локализацией (выявления места) ошибок, то такой подход может быть неприемлем в системах р.м.в., в которых реакция системы на сбои должны происходить за минимально установленное время. Таким образом, этап восстановления является жизненно важной частью системы р.м.в.

Для проведения восстановительных работ в системах р.м.в. используются следующие методы (способы):

- дублирования данных (МВДБ) с контрольной суммой;
- троирование данных (МВТР);
- контрольных точек (контрольный перезапуск) (МВКТ);
- переинициализации данных (МВПИ);
- повторного счета (МВПС);
- восстановления за счет программной избыточности (МВПРИЗ).

При этом первые четыре метода: МВДБ, МВТР, МВКТ, МВПИ служат для восстановления состоянии данных в памяти ВС, а остальные два метода МВПС, МВПРИЗ для восстановления программ, связей программных модулей ВС. Рассмотрим назначение, условия применения, преимущества и недостатки этих методов.

Метод МВДБ с КС предназначается для восстановления значений данных за счет копии. Для использования этого метода создают структуры данных. Вычисляют КС структуры и копии. Метод МВДБ с КС обычно реализуется сравнением копий и контрольных сумм.

Другой метод восстановления МВТР, требует хранения дополнительно двух копий контролируемой структуры данных. Восстановление в случае ошибок осуществляется за счет I-ой или 2-ой копии. Достоверность копии и оригинала определяется сравнением оригинала с копиями. При адекватности двух проверяемых, искаженным считается третий.

Некоторые системы р.м.в. периодически заносят состояние системы в память (часто, во внешнюю намять) для того, чтобы иметь возможность перезапуска одного из таких состояний, в случае возникновения неисправности. Эта процедура известна как перезапуск от контрольной точки [12,20,24,77].

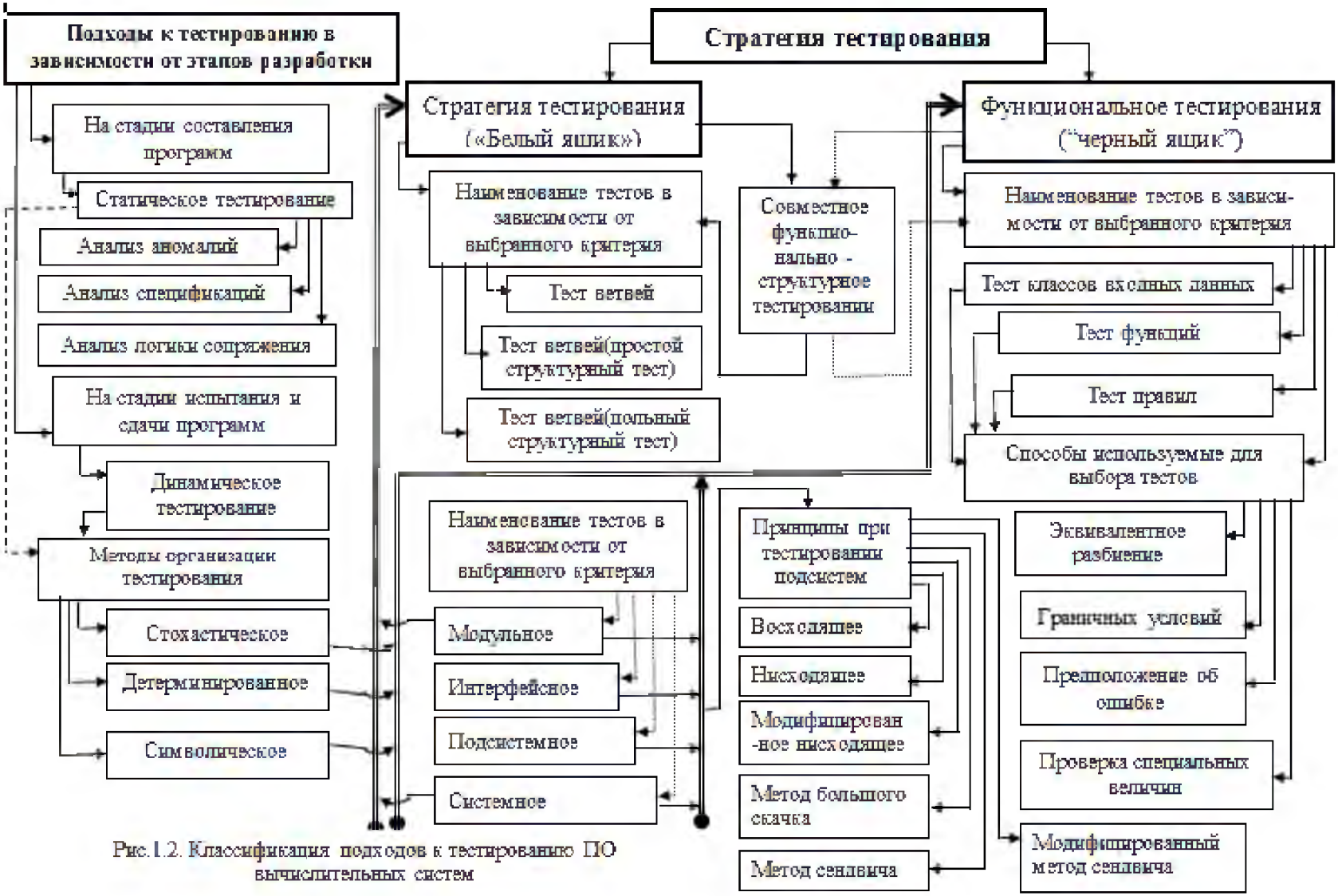


Рис.1.2. Классификация подходов к тестированию ПО вычислительных систем

Основной принцип контрольной точки заключается в снятии копий с последнего состояния системы через заданный интервал времени. В случае если обнаружена ошибка, система может быть восстановлена по последнему нормальному состоянию и может быть, осуществлен повторный пуск. Даже в том случае, когда выбираются оптимальные интервалы между контрольными точками, расходы, связанные с реализацией классического метода контрольных точек велики [l6]. Поэтому Ренделом [90] был предложен подход к методу контрольной точки, при котором значительно снижаются расходы на реализацию метода, так-как при этом запоминаются состояние системы только в те моменты, когда они модифицируются.

Но этот подход был реализован для частного случая.

Еще один способ восстановления состояния и сохранности данных в памяти ВС, заключается в использовании переинициализации исходного состояния ПО систем р.м.в. Время восстановления данным способом требует больше времени чем все перечисленные методы.

Менее эффективным являются методы повторного счета программной избыточности, предназначенные для восстановления работоспособности программ. Так метод МВПС эффективен в случае самоустраняющихся сбоев, когда потеря разряда, или синхронизации не влечет больших неприятностей [85].

Другой метод МВПРИЗ, существенно увеличивает время одной операции. Так как при реализации этого метода разрабатывается два блока основной и резервный, а также управляющая программа. После выполнения основного блока управляющая программа сверяет данные с ожидаемыми и в случае несоответствия запускает резервный.

Рассмотренные методы обнаружения, диагностики и восстановления ПО систем р.м.в. реализуются в виде оперативного и периодического контроля [30,83,85]. Использование того или другого вида контроля связано с дополнительными затратами машинного времени и оперативной памяти.

Оперативный контроль (ОК) позволяет проводить наблюдение за состоянием системы непрерывно, но требует относительно большие временные затраты.

При периодическом контроле(ПК) может произойти большая потеря информации, которая ведет к снижению эффективности ПО. При этом временные затраты для ПК требуются существенно меньше чем при ОК. Затраты на контроль и восстановление в пределах 3-5% от полезного времени решения задачи считаются допустимыми. И эти затраты должны обеспечивать вероятность обнаружения искажений не менее 0.9-0.95, а ошибки приводящие к отказу, должны обнаруживаться практически достоверно [28].

В работе [65] в качестве средств повышения надежности программ системы р.м.в. используется ОК и ПК. Однако в данной работе не приведены количественные оценки эффективности и сложности разработанных программных средств, реализованных на ЭВМ- М-6000.

Перейдем к рассмотрению работ, где использованы перечисленные или другие методы обнаружения, диагностики и восстановления в ПО сетей ЭВМ.

В работе [95] указывается, что проблемы надежности вычислительной сети такие, как переход ошибки от одного узла к другим узлам сети; сложность алгоритмом управлении сетью; сложные взаимозависимости участков сети; повышенная стоимость методов обнаружения, диагностики и восстановления сети при использовании их в сравнении с изолированной ЭВМ требуют специального решения. Надежность вычислительной сети должна обеспечиваться за счет надежности ЭВМ, ПО и средств связи. Наряду с анализом использования методов повышения надежности аппаратурной части ЭВМ в [95] исследуются методы повышения надежности ПО, которые подразделены на две группы: методы предотвращения появления ошибки и методы, используемые при появлении ошибок. К последним относятся методы обнаружения, диагностики и восстановления. Оценки по сложности реализации и эффективности предлагаемых методов не приводится.

В работе [80] описываются средства повышения достоверности информации, передаваемой между различными узлами распределенной сети. Эти средства составили:

1. Контроль по четности (или нечетности), при котором каждое слово памяти содержит один контрольный разряд, позволяющий обнаруживать ошибки в информационных разрядах;

2. Контрольное суммирование данных.

3. Избыточный контроль, при котором один и тот же контролируемый признак получается различными способами.

С целью повышения готовности сети организован профилактический контроль с применением диагностических процедур, проверяющих как программные функции системы, так и ее отдельные аппаратные блоки.

Некоторые работы посвящены [75,87] методам и средствам повышения надежности при управлении потоками и сетях передачи данных. В [87] отмечается, что для надежности управления важное значение имеет возможность обнаружения ошибок в управляющей информации, и, в частности, в протоколах.

Анализ работ по использованию методов обнаружения, диагностики и восстановления показывает, что в вычислительных сетях эти методы не использованы или использованы для проверки средств связи [4], для управления потоками в средствах связи [21,80]; в протоколах сетей ЭВМ [14,25,87]. Исключение составляют работы, где освещены проблемы, связанные с вопросами надежности ПО вычислительных сетей. Однако не приведены количественные оценки сложности и эффективности для выбора методов повышения надежности ПО.

1.4. Теоретические основы адаптации систем реального времени

Управление, как способ и средство достижения заданных целей в объекте, используется повсеместно и для объектов самой разнообразной природы –

технических, биологических, социальных систем [46]. Всякая система управления является, по сути дела, преобразователем целей в новое состояние объекта, которое удовлетворяет этим целям.

Целей в процессе управления обычно бывает много, и они могут изменяться во времени. Но среди них почти всегда присутствуют относительно стабильные цели типа: "Сделать это в предельном смысле наилучшем образом", т.е. с максимальной точностью или при минимальном нарушении каких-то заданных ограничений. Ввиду стабильности целей такого рода их достижение удобно поручить специального вида управлению, которое и получило название адаптации.

Адаптация также должна поддерживать объект в состоянии, определяемом целью. И в этом смысле адаптация является управлением. Но цель адаптации обычно одна, или, во всяком случае, таких целей немного. В этом и состоит специфика адаптации как управления. При этом адаптаций является вторичной — по отношению к основному контуру управления. Например, если управление выполняет основные цели, реализация которых обеспечивает функционирование объекта, то адаптация обеспечивает качество процесса функционирования объекта. Именно поэтому ресурс управления распределяется обычно неравномерно − основная часть выделяется на работу главного контура управления, обеспечивающего выполнение основных целей, а остальное идет на адаптацию, повышающую эффективность работы основного контура (или системы управления).

При этом необходимо подчеркнуть, что в адаптивных системах управления управляющая часть автоматически изменяет свою структуру и (или) свои параметры в зависимости от изменения внешних условий и свойств объекта управления.

Очевидно для управления объектом необходимо иметь о нем какие-то представления, требуется создать модель объекта. Под моделью подразумевается любой способ, позволяющий определить, каким образом будет

вести себя объект, в той или иной ситуации. Моделью могут быть логические рассуждения, с помощью которых состояние объекта "выводится" [43]. Часто модель представляется в виде графиков, связывающих состояние среды, управляющее воздействие и состояние объекта.

Но чаще всего модель объекта представляется в виде алгоритма, т.е. правил, с помощью которого можно осуществить оценку состояния объекта по известному состоянию среды и управляющему воздействию. В простейшем случае модель объекта представляется в виде формулы (для статического объекта) или оператора, например, в виде системы дифференциальных уравнений (для динамического объекта). Модель объекта дает возможность алгоритму управления определить управляющее воздействие, которое переведет объект в заданное целью состояние. Этому процессу управления предшествует процесс создания модели объекта (процесс идентификации) [45,46,63]. Он состоит в том, что по наблюдениям за функционированием объекта строится модель, удовлетворяющая этим наблюдениям.

В зависимости от того, есть модель объекта или нет, различают два очень важных случая: адаптацию с моделью и поисковую адаптацию (без модели). В первом случае в процессе синтеза адаптирующего (управляющего) воздействия объект можно заменить его моделью, что значительно облегчает сам процесс адаптации, а во втором этого делать нельзя.

Адаптацию с моделью разделяют на априорную адаптацию и апостиорную адаптацию [43]. Остановимся на первом из них.

Смысл которой заключается в следующем. При ограниченном числе возможных ситуаций, которые могут сложиться в процессе адаптации, для этих ситуаций можно заранее решить задачу адаптации и заготовить в памяти информацию о необходимых адаптирующих воздействиях в виде таблицы оптимальных решений. В этом случае процесс адаптации сводится к оценке ситуаций, выбору из таблицы решений информации об оптимальном адаптирующем воздействии и реализации этого воздействия на объект.

Такого рода адаптацию называют априорной [43], так как все здесь заготавливается заранее (априори) в виде решающей таблицы: ситуация (X) – необходимое оптимальное адаптирующее воздействие (U); причем адаптирующее устройство работает в режиме преобразователя: ситуация – воздействие на основе указанной таблицы. Блок схема такой априорной адаптации показана на рис.1.3. Это самая распространенная и очень удобная схема адаптации, т.е. приспособления к изменяющимся условиям функционирования.

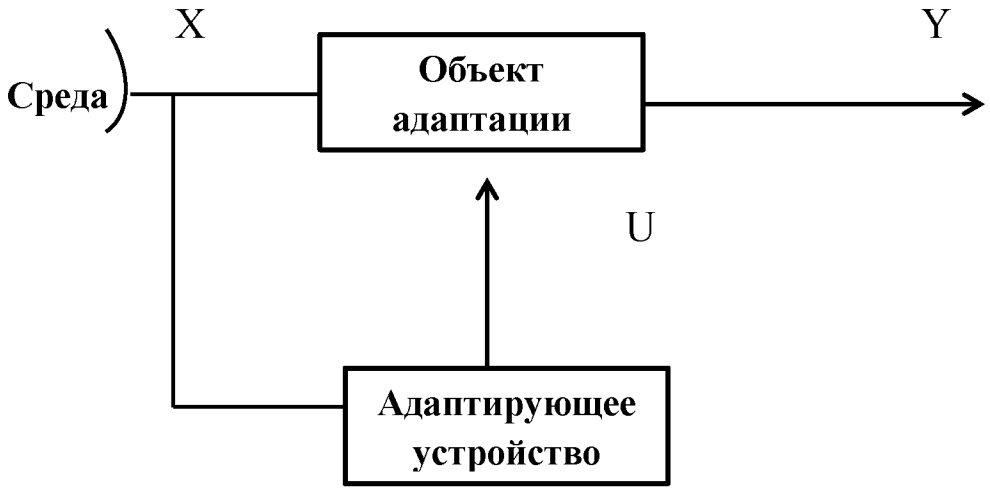


Рис.1.3. Схема системы адаптации при неизменном объекте (априорная адаптация)

При апостериорной адаптации заранее нельзя предвидеть в какой ситуации окажется объект адаптации, синтез адаптирующего воздействия следует производить после определения нового состояния среды. Она отличается от априорной адаптации, так же тем что оптимальное решение определяется не заранее, а лишь после того, как выявлена ситуация, в которую попал объект адаптации.

В случаях, когда для объекта нельзя построить модель, используется поисковая адаптация. При поисковой адаптации методом решения задачи адаптации является поиск, в процессе которого определяется влияние адаптирующего воздействия на эффективность объекта, и эта информация

используется для повышения эффективности объекта.

Сам по себе процесс поисковой адаптации имеет последовательный многоэтапный характер ─ на каждом этапе принимаются меры по повышению эффективности объекта (в отличие от адаптации с моделью, при которой возможна адаптация за один этап).

В целом, трудности адаптации с моделью сводятся к синтезу объекта, т.е. к идентификации, а сама адаптация сводится к решении оптимизационной задачи выбора такого адаптирующего воздействия, которое удовлетворило бы целям адаптации. При поисковой адаптации трудности иного роди − нужно одновременно и экспериментировать с объектом, и адаптировать его. Эта задача значительно труднее [46].

Цели адаптации. Адаптация, как сказано выше, является частным случаем управления со стабильными целями. Цели адаптации имеют двоякий характер: цели первого рода и второго рода. Для целей первого рода характерны жесткие требования предъявляемые к состоянию объекта. Такие как, время обработки или вероятность функционирования, или потребляемая мощность в заданных пределах и т.д.

Цели второго рода связаны с экстремальными требованиями, предъявляемыми к объекту адаптации в виде максимизации эффективности его функционирования. К таким требованиям можно отнести: максимальную надежность или живучесть, минимальные энергетические затраты и др. Этих показателей эффективности следует добиваться лишь при обязательном выполнении целей первого рода.

Перейдем к рассмотрению специфики узла АК как объекта адаптации. Можно выделить несколько черт узла АК, которые определяют целесообразность введения адаптации в процесс функционирования узла АК.

1. Существенная зависимость узла АК от среды. Узел АК является типичной системой массового обслуживания, эффективность которой определяется качеством обслуживания потока данных.

2. Значительная априорная неопределенность среды узла АК (внешняя и внутренняя).

Здесь внутренняя неопределенность связана с нестационарностью случайных помех, ошибок и неисправностей в узле АК.

Эти обстоятельства приводят к тому, что поведения среды узла АК (внешней и внутренней) имеют ярко выраженный нестационарный характер, что усугубляет неизбежность введения адаптации не только для реализации оптимального режима функционирования, но и для поддержания ее в рабочем режиме при изменении свойств среды.

Основные причины − факторы изменения свойств среды, их влияния на функционирование узла АК были проанализированы в предыдущих разделах (конкретно для узла АК в качестве сред рассмотрена оперативная память, процессор и другие технические средства ЭВМ). Этот анализ показал, что при функционировании узла АК может происходить потеря эффективности из-за сбоев и отказов технических средств ЭВМ (возмущающего воздействия среды).

При этом одним из уязвимых мест в узле АК являются структуры данных ПО. Для обнаружения и устранения возмущений в этом поле были рассмотрены в п.1.2,2 методы контроля и восстановления нормального состояния данных. Согласно этому анализу выбор и использование наиболее приемлемых из них должен производиться в процессе создания модели адаптации узла АК. Поэтому в рассматриваемой для узла АК проблеме обеспечения качественного функционирования, отвечающей основной цели адаптации − увеличению эффективности и поддержанию узла АК в рабочем режиме необходимо решить следующие задачи:

I) исследование структур данных узла АК и выявление основных видов возмущений влияющих на эффективность узла;

2) создания модели адаптации узла АК для случая работы его в вычислительной системе реального времени с возможными сбоями и отказами

технических средств ЭВМ;

3) разработка адаптирующего средства, осуществляющего качественное управление потоками данных в узле АК в условиях возмущающего воздействия среды на объект управления;

4) исследование эффективности использования средств адаптации в узле АК;

5) оценка сложности разрабатываемых средств адаптации структур данных узла АК к возмущениям среди.

Для решения этих задач требуется рассмотреть показатели эффективности систем, которые оцениваются надежностью.

1.4. Анализ и выбор показателей оценки надежности функционирования систем реального времени

В теории надежности одной из важных задач является выбор показателей [1,2]. Для выбора основных показателей оценки работы ПО узла адаптивной коммутации, будем учитывать следующие ее особенности:

- ПО узла адаптивной коммутации функционирует в вычислительной системе р.м.в., в которой происходят сбои и отказы;

-сбои и отказы в ПО узла адаптивной коммутации приводят к уменьшению пропускной способности узла, влияют на готовность системы.

Использование средств контроля и восстановления в составе ПО узла адаптивной коммутации влияет на эффективность и производительность ВС в р.м.в. и приводит согласно [31], к средней потере эффективности за единицу времени определяемой по формуле:

$$C_{эп} = (Tb_1 / t' + t' b_2 + (1 - t'/2T) b_3) / (T + b_3 (1 - t'/2T) \quad (1.1)$$

где, b_1 – затраты па проведение процедуры контроля; b_2 – потеря эффективности функционирования комплекса программ из-за запаздывания на единицу времени в обнаружении искажения данных или вычислительного

процесса ; b_3– потери из-за времени восстановления; t' -оптимальный период запусков процедур контроля; T -средняя наработка на отказ. Кроме этого, формула (I.I) верна в предположении что неисправности образуют пуассоновский поток событий. Значение t' определяется по формуле

$$t' = \sqrt{\frac{2b_1T}{b_2+2C_3}} \qquad (1.2)$$

где C_3 – потеря эффективности за единицу времени восстановления.

В случае, когда потери и затраты связаны только с изменением полезного времени функционирования программ и уменьшением производительности ВС, то b_1 можно интерпретировать как время проведения однократного контроля (t_1), а b_3 как время восстановления (t_3). При b_2=1 считается затраты равны t'/2 [31]. Тогда t' определяется по формуле

$$t' = \sqrt{\frac{2Tt1}{1+2t3}} \qquad (1.3)$$

Коэффициент простоя Kn комплекса программ в системах р.м.в. определяется:

$$Kn = \frac{\frac{Tt1}{t'}+\frac{t'}{2}+\left(1-\frac{t'}{2t}\right)t3}{T+t3(1-\frac{t'}{2T})} \qquad (1.4)$$

что сответствует уменьшению готовности системы

$$Krq = 1 - Kn \qquad (1.5)$$

В предположении экспоненциональности распределения наработки между отказами $\mathbf{F(t) = 1 - e^{-\lambda t}}$ и времени восстановления $\mathbf{G(t) = 1 - e^{-\mu t}}$ [3], готовность можно определить по формуле

$$Krq = \frac{T}{T+\tau} = \frac{\mu}{\lambda+\mu} \qquad (1.6)$$

Где τ – среднее время восстановления, определяемое как $\tau = 1/\mu$; μ – интенсивность восстановления; λ – интенсивность отказов.

Надежность программного обеспечения, обозначаемая через R(t), характеризуется вероятностью того, что за время t отказа не произойдет [3]. Отсюда F(t) = 1- R(t) – вероятность отсутствия отказа за это же время.

Интенсивность отказов $\lambda(t)$ определяется распределением отказов во времени, т.е, это интенсивность отказов при условии, что до данного момента система или ее часть работала нормально. Таким образом

$$\lambda(t) = P'(t)/R(t) \qquad \text{и} \qquad R(t)=\exp\left[-\int_0^{\infty} \lambda(x)d(x)\right]$$

В частном случае при λ= const надежность есть

$R(t)=\exp(-\lambda t)$

Средняя наработка на отказ определяется как математическое ожидание временного интервала между последовательными отказами. Ее можно связать с надежностью соотношением

$$T = \int_0^{\infty} R(t)dt$$

Если интенсивность отказов постоянная (т.е. когда длительность интервалов между последовательными отказами имеет экспоненциальное распределение), то средняя наработка на отказ обратна интенсивности отказов

$$\mathbf{T = 1/\lambda} \qquad \mathbf{(1.7)}$$

При заданных n наработках на отказ средняя наработка статистически определяется в виде

$$\mathbf{T} = (\mathbf{T}_1 + \mathbf{T}_2 + \ldots + \mathbf{T}n)/n = \sum_{i=1}^{n} Ti / n \qquad (1.8)$$

Еще одним показателем надежности ПО служит среднее время восстановления его τ, статистически определяемое как

$$\tau = (\tau_1 + \tau_2 + \ldots + \tau_n) \qquad (1.9)$$

где τ_I ($i=1,n$) – время, затрачиваемое на i -ое восстановление.

Рассмотрим далее, широко используемое в теории надежности понятие работоспособность [3].

Работоспособность или неработоспособность в общем случае, могут быть полными или частичными. Полностью работоспособный объект обеспечивает в определенных условиях максимальную эффективность. Эффективность применения в тех условиях частично работоспособного объекта меньше возможной, но значения ее показателей при этом еще в пределах, установленных для такого функционирования, которое считается нормальным. Частично неработоспособный объект может функционировать, но уровень эффективности при этом ниже допускаемого. Поэтому основным показателем работоспособности служит эффективность функционирования системы (объекта) [3], под которой понимается некоторая количественная характеристика качества и объема выполняемой системой работы.

Процесс восстановления ПО может формализовать СМО. Эффективность функционирования системы массового обслуживания характеризуется средним числом обслуженных требований (в случае отказов системы и потерь из-за нехватки мест ожидания) средним временем ожидания до окончания обслуживания и т.п. Эффективность функционирования информационных систем может характеризоваться объемом и достоверностью переданной информации. В целом задачи вычислительных сетей рассматривают как задачи, относящиеся к СМО. При этом для оценки эффективности узла адаптивной коммутации предлагается использовать следующий показатель

$$\textbf{Кэф=Сок/Спк} \qquad \textbf{(1.10)}$$

где **Сок** – число обработанных кадров; **Спк** – число принятых кадров.

При выборе показателей надежности следует руководствоваться следующими рекомендациями [3]:

1) общее число показателей надежности должно быть по возможности

минимальным;

2) следует избегать сложных показателей, получаемых в виде каких-либо сверток критериев;

3) выбранные показатели надежности должны иметь простой физический смысл;

4) выбранные показатели надежности должны допускать возможность проведения (проверочных) оценок на этапе проектирования;

5) выбранные показатели надежности должны допускать, возможность статистической (опытной) оценки при проведении специальных испытаний или по результатам эксплуатации;

6) выбранные показатели должны допускать задание норм надежности в количественной форме.

Согласно этим рекомендациям для оценки работы ПО узла адаптивной коммутации, функционирующего в вычислительной системе с неисправностями как при использовании средств контроля и восстановления, так и без них выберем следующие показатели:

- средняя наработка на отказ (**T**) ;
- среднее время восстановления ($\boldsymbol{\tau}$);
- коэффициент эффективности ПО узла АК (**Кэф**);
- вероятность безотказной работы (**R(t)**);
- готовность ПО узла адаптивной коммутации (**Krq**):
- вероятность обнаружения и устранения ошибок (**Po, Pb**).

Эти показатели достаточно полно могут характеризовать работу узла адаптивной коммутации с выполнением двух основных требований адаптации: максимизации эффективности управления и минимизации затрат на повышения надежности объекта управления.

ВЫВОДЫ

1. Узел адаптивной коммутации как система управления имеет сложную архитектуру, состоящую из физической, логической и программной и др. взаимосвязанных между собой структур.

2. Исследование основных причин появления ошибок в программной структуре показывает, что снижение пропускной способности и готовности узла АК зачастую происходит из-за неисправностей (возмущений) элементов физической структуры.

3. Проанализирована возможность использования априорной адаптации для структур данных узла АК, модель которой можно представить в виде таблицы решений.

4. Проведенный анализ методов повышения надежности систем реального времени показывает, что для использования их необходимо выполнить основные требования адаптации: минимизация дополнительных ресурсов и максимизация эффективности разрабатываемых средств.

5. Для оценки эффективности управления работой узла АК показана целесообразность применения следующих показателей: средняя наработка на отказ, среднее время восстановления, коэффициент готовности и пропускной способности, вероятность безотказной работы и вероятность обнаружения и устранения ошибки.

6. Анализ моделей систем реального времени указывает на необходимость разработки аналитической модели, оценивающую работу узла АК в условиях возмущений.

ГЛАВА II. АДАПТАЦИЯ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ ФУНКЦИОНИРУЮЩИХ В УСЛОВИЯХ ВОЗМУЩАЮЩЕГО ВОЗДЕЙСТВИЯ СРЕДЫ

В данном разделе будут приведены результаты исследований ПО повышению надежной работы узла АК с целью аффективного управления потоками данных. В априори определяется влияние возмущающего воздействия среды (технических средств вычислительной техники) на общую область памяти (ООП) узли АК. Производится анализ и классификации ООП узла АК в зависимости от характера изменения структур данных. Определяются группы структур данных наиболее уязвимые к неисправностям (возмущениям) [54]. Выявляются и классифицируются критические ошибки [50]. Осуществляется выбор и разработка методов обнаружения и восстановления. Строится таблица решений. Определяются функции и состав адаптирующего средства. Производится программная реализация. Приводится оценка их реализации [49,52,53].

2.1. Анализ и классификация структур данных и потока информации обеспечивающих работу систем реального времени

Для качественного управления структур данных ООП ПО узла АК необходимо определить вид структур, установить их типы, анализировать порядок и частоту изменения данных, что позволяет уменьшить трудности, связанные с разработкой методов контроля и восстановления. Данный параграф посвящен этой задаче.

Для эффективного взаимодействия блоков ПО узла АК, их внутренние данные, используемые каждым блоком, формируются в общей области памяти данных (рис.1.1).

Символические имена, присвоенные структурам данных (такие как MCT, TRUT, LINT, TABPARAM и др.), собираются в файле символических имен задач. Впоследствии этот файл (район) используется при

построении остальных задач ПО узла адаптивной коммутации (каждый блок узла реализуется в виде отдельных задач), представляя таким образом возможность доступа к структурам данных по символическим именам, либо с помощью указателя [61].

Приведенные структуры данных охватывают в основном все данные ООП, реально реализованные в данный момент в ПО узла АК. В этих таблицах графы 5,6 содержат номера групп данных ПО узла адаптивной коммутации, классифицированные из следующих соображений. В ООП внутренние данные связаны древовидной структурой. При начальной инициализации в главную таблицу узла **MCT** заносятся адреса всех отдельных глобальных компонентов узла. Затем осуществляется доступ к нужному компоненту (рис.2.1).

Для доступа к контекстам трактов или линий одного указателя **MCT** недостаточно (рис.2.1). Доступ, например, к данным контекста **i** -го тракта осуществляется с помощью двух указателей: первого − от **MCT** до **TRUT**; второго − от **TRUT** до (адрес **i** -го контекста тракта). В связи с этим до контекста КК таблицы **CSTB** входного-выходного КК) используется три, а до компенсационного буфера − четыре указателя.

В целом, в зависимости от интенсивности изменения данных ООП, можно предложить их классификацию с разбиением на следующие группы.

1. Данные, определяемые и изменяемые в стадии инициализации или переинициализации (группа постоянных данных - @1).

К группе постоянных данных относятся значения в указателях главной таблицы узла **MCT**; входных таблицах передач трактов или линий; таблицах основных данных; маршрутизаций КП и КК, адресов контекстов трактов и линий; таблицах основных блоков БКП, БВА, БРК, БСК, БКК.

2. Данные, определяемые и изменяемые при каждой организации или ликвидации КК (группа квазипеременных данных- @2). Эту группу составят данные, содержащиеся в контекстах трактов и линий.

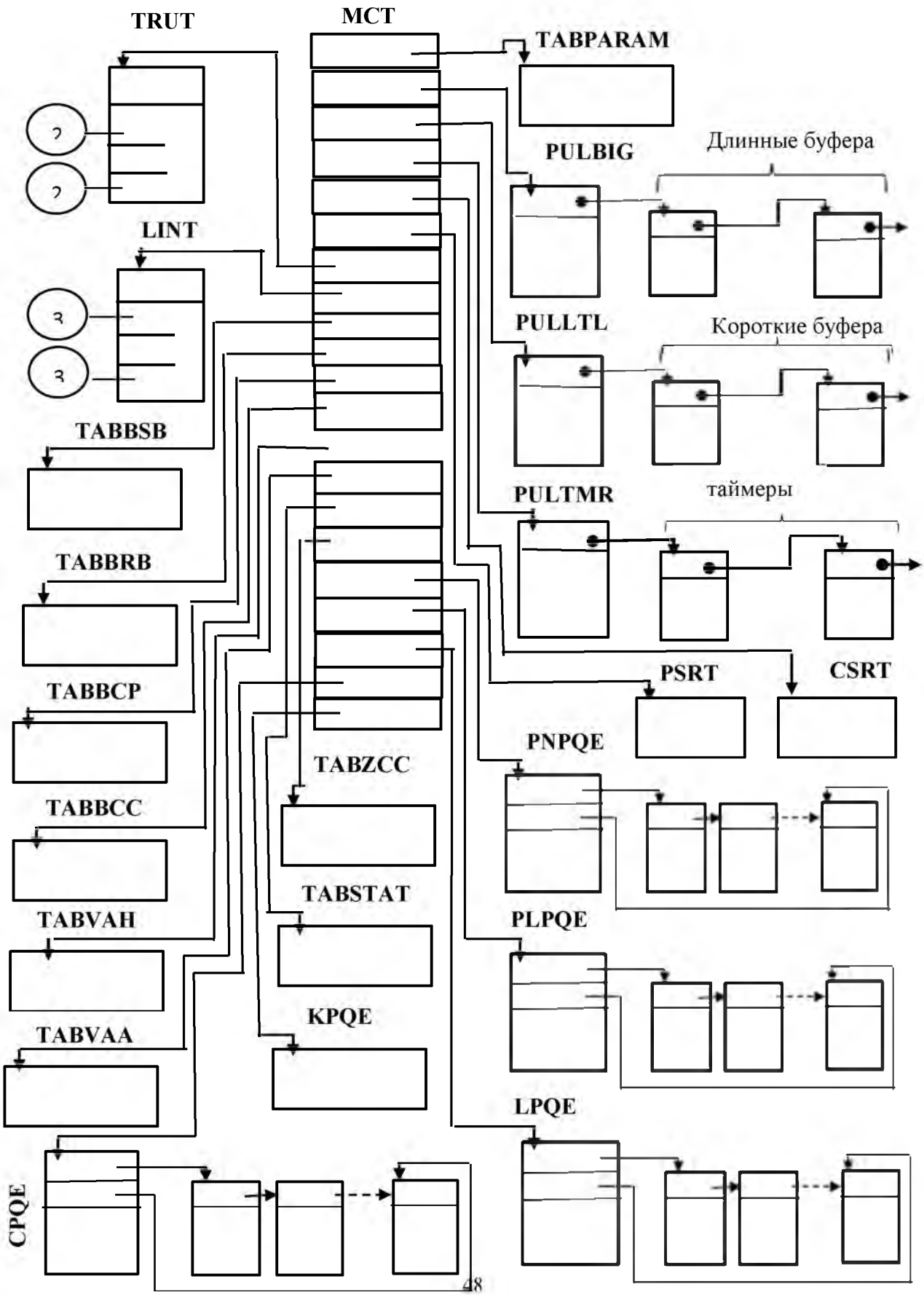
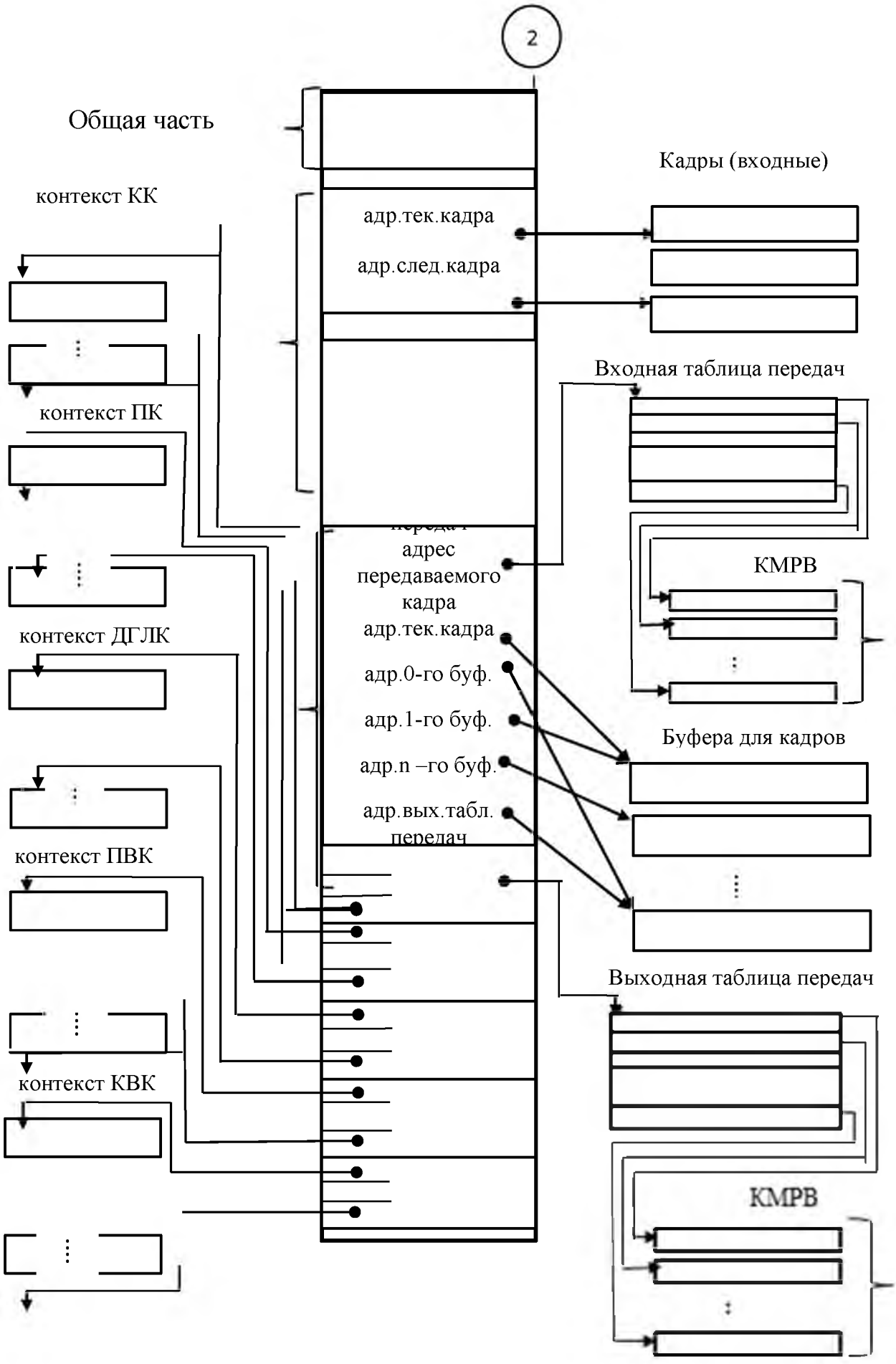
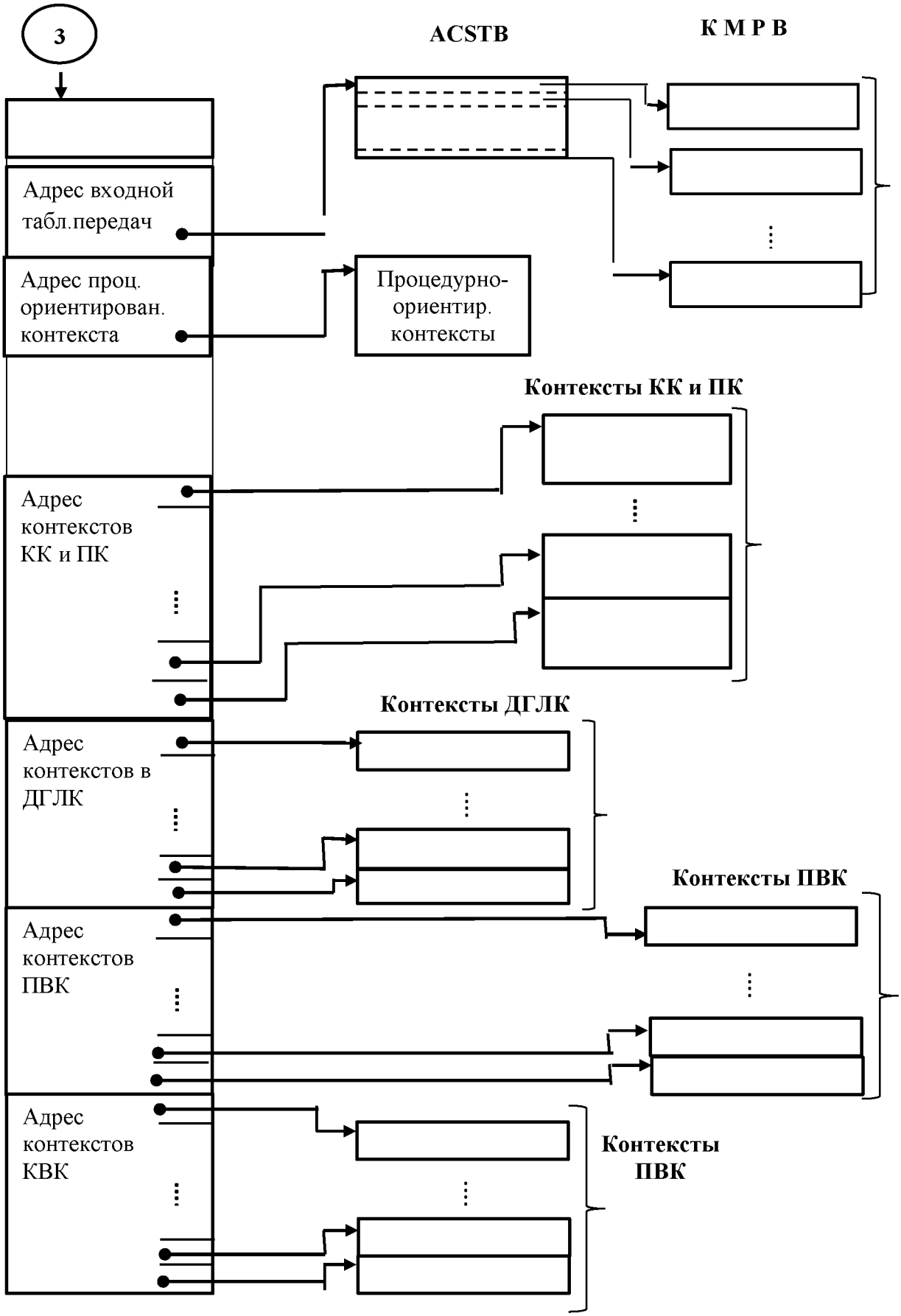


Рис. 2.1. Общая область памяти. Связь таблиц через указатели с управляющей таблицей



Прод.рис.2.1. Общая область памяти. Контекст тракта, таблиц передач, кадры и буфера для кадров, адреса контекстов КК, компенсационные буфера



Прод. рис.2.1. Общая область памяти. Контексты абонентской линии и связанные с ним объекты: процедурно-ориентированные контексты и контексты КК, ПК, ДГЛК, КВК, таблицы передач

ликвидации КК блоком БКК.

3. Данные, отличающиеся наиболее высокой интенсивностью изменения (переменные данные –@3). В эту группу входят данные, фиксирующие текущее число и адреса буферов для пакетов в очередях и пулах, номеров и адресов буферов для кадров.

4. Данные, содержащиеся в пакетах и кадров компенсационных буферов; очередях трактов и линии (группа часто меняющихся данных.

Из всех групп данных узла адаптивной коммутации контролю и восстановлению будут подвергаться поля данных из групп @1, @2, @3 размер которых 500-600 байтов. При этом область данных ООП узла адаптивной коммутации составляет более 4000байтов. Основной причиной отсутствия контроля поля @4, содержащего информационные и служебные сообщения вычислительной сети, является частота изменения данных.

Перейдем к рассмотрению особенности неисправностей ПО узла АК, их влияния на нормальный ход работы блоков и данных всего ПО узла. Как показано в рис.2.1, наиболее уязвимым местом в поле данных ООП являются ссылки от главной таблицы узла − **МСТ** до требуемой компоненты. Так как доступ к каждому компоненту ПО узла АК происходит через указатели, содержащиеся в **МСТ**, и в случае искажения в поле **МСТ**, теряется возможность доступа в какой-нибудь компоненте (рис.2.2), Кроме этого, искаженное значение указателя может содержать адреса неизвестной области памяти. При этом, если эта область системно защищена, операционная система снимает задачу и выдает отказ. Точно к таким же последствиям приводят искажения ссылки в очередях к адресу первого буфера (рис.2.3). Отказы могут произойти при искажении адресов буферов для кадров, указателей адресов входных и выходных передач трактов и линий, компенсационных буферов.

Искажения в других данных ООП, не вызывая отказа ПО, влияют на эффективность функционирования узла. При этом возможна блокировка из-за искажений в управляющих элементах очередей как типа FIFO, так и LIFO.

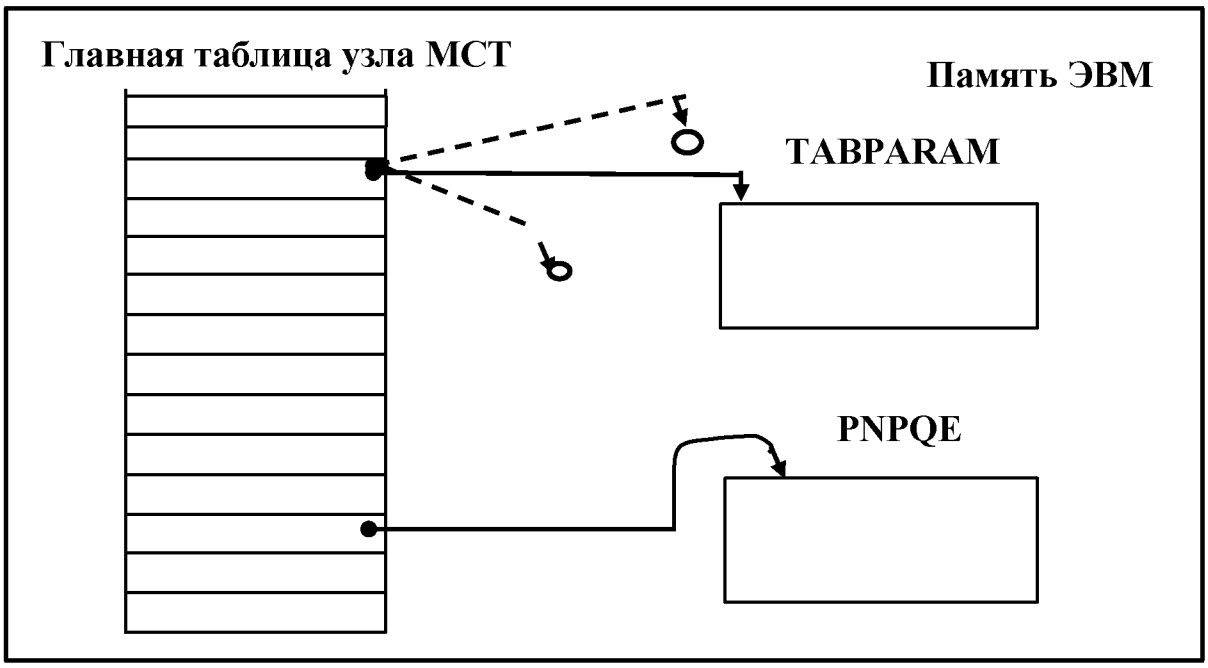


Рис.2.2. Физическая интерпретация ошибки от главной таблицы MCT до таблицы TABPARAM (стрелки со штрихами указывают на ошибку)

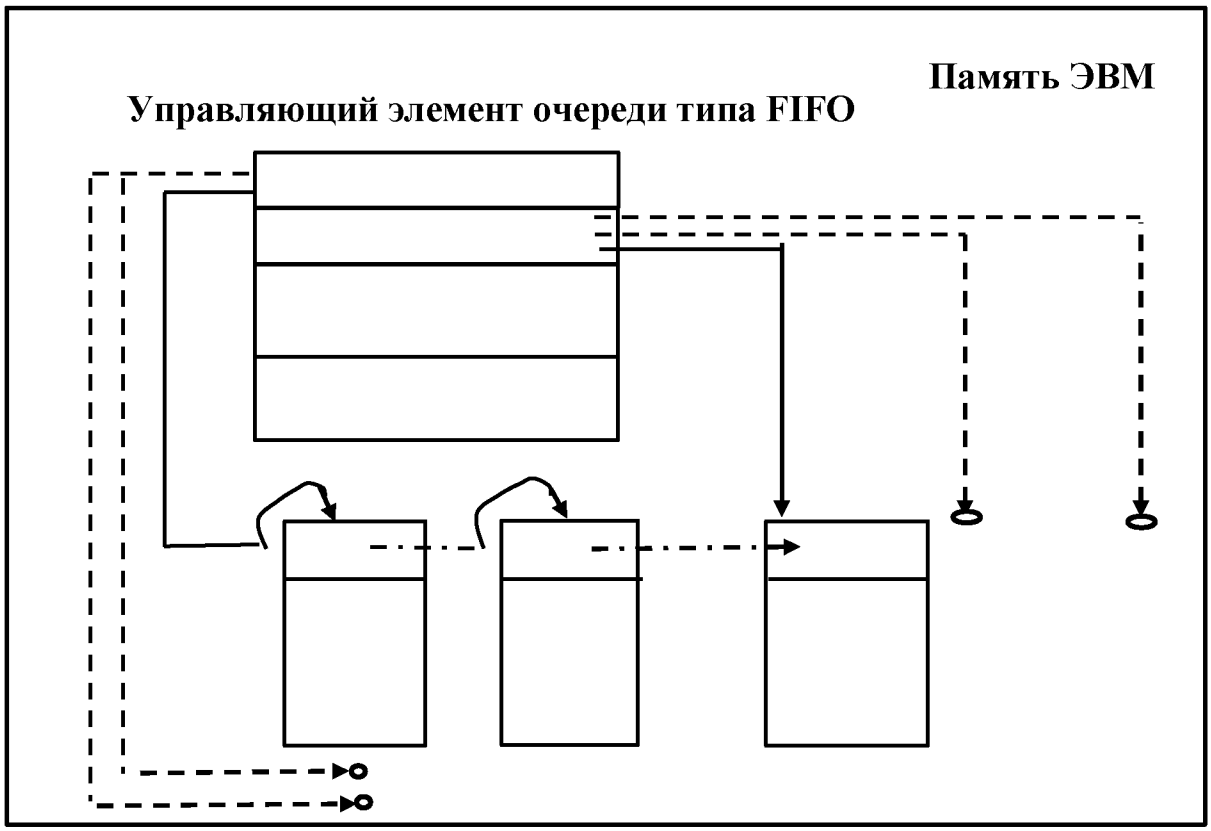


Рис.2.3. Физическая интерпретация ошибки в ссылке на очереди типа FIFO (стрелки со штрихами указывают на ошибку)

Возможны неисправности в контекстах КК, когда позиции, отведенные одному каналу, заняты другим каналом, либо в контексте тракта, когда из-за

искажения теряется адрес, и тем самым весь контекст КК и т.д. Также можно ожидать неисправностей в маршрутных таблицах как режима КК, так и режима КП.

Определение. Критическими ошибками ПО узла адаптивной коммутации являются ошибки, приводящие к снижению пропускной способности и отказу ПО узла АК.

В ходе анализа искажений значений структур данных узла адаптивной коммутации были установлены критические ошибки с однотипными последствиями: блокировка и потеря буферов в очередях из-за ошибок в управляющих элементах последних; разрушение ссылок в древовидной иерархии данных ООП и потеря доступа к контекстам, и таблицам из-за ошибок в указателях. Перечень критических ошибок, их источники и типы приводятся в табл.2.1.

Таблица 2.1.

Типы и источники критических ошибок (кр.ош.) ПО узла адаптивной коммутации

n n	Н а з в а н и е	Обозначение	Источники критических ошибок
1	Потеря доступа к требуемой компоненте: таблице основных данных, к таблицам всех блоков узла, к таблицам контекстов трактов и линий, к маршрутным таблицам режима КК и КП, к входным очередям блока БКП PNPQE и PLPQE блока БКК CPQE, к пулам свободных длинных и коротких буферов	кр.ош.1	Управляющая таблица МСТ
2	Потеря доступа от таблицы контекстов тракта TRUT (линий LINT) до требуемого контекста тракта(линии)	кр.ош.2	Таблица контекстов трактов(линий)
3	Потеря доступа от контекста тракта(линии) до входной(выходной) таблицы КК	кр.ош.3	Контекст тракта (линии)
4	Потеря доступа от входной(выходной) таблицы КК до компенсационного буфера	кр.ош.4	Таблица КК в контекстах
5	Потеря доступа к буферам очередей типа	кр.ош.5	Параметр управляющего

	FIF0: к этим очередям относятся входные очереди к блоку БКП − PNPQE и PLPQE, к блоку БКК - CNPQE и CLPQE, очереди в выходных трактах TPQE, очереди в выходных линиях		элемента очередей
6	Потеря доступа к буферам пулов типа LIFO: к пулам свободных длинных и коротких буферов	кр.ош.6	Параметр управляющего элемента пулов ADRFBUF
7	Текущее суммарное число буферов не равно заданному при инициализации. Происходит потеря буфера.	кр.ош.7	Параметр NUMBERB в очереди или в NUMBUF пулах
8	Блокировка буферов в очередях и пулах	кр.ош.8	Параметр MAXNB в очер. или в MAXBUF пулах
9	Основные параметры узла искажены	кр.ош.9	Поля памяти таблицы TABPARAM
10	Маршрутная таблица режима КП или КК неверна	кр.ош.10	Поля памяти таблицы RSRT(CSRT)
11	В таблицах блоков БКП, БКК, БСК, БРК, БВА; контекстах трактов и линий, а также в других данных группы @2 произошли искажения	кр.ош.11	Поля памяти соответствующих данных
12	Пулы контекстов трактов(линий) содержат искаженную (неверную) информацию	кр.ош.12	Контекст трактов TCT (линий LCT)
13	Пулы таблиц КК трактов искажены	кр.ош.13	Входная-выходная таблица КК тракта
14	Пулы таблиц КК абонентских линий искажены	кр.ош.14	Таблица ACSTB контекста линии
15	Блокировка очереди типа **FIFO**	кр.ош.15	Адрес первого буфера в очереди
16	Задержка буфера в очереди/пула	кр.ош.16	Текущее число буферов в очереди/пуле
17	Потеря буфера в пуле	кр.ош.17	Массив хранения буферов длинных NBIG; коротких NLTL

Таким образом, задача повышения эффективности работы узла АК сводится к качественному управлению структурами данных в условиях возмущающего воздействия неисправностей ЭВМ на нормальный ход работы узла. Решения, которые будут осуществляться адаптацией структур данных к возмущениям. Для чего в следующем параграфе произведем выбор и разработку методов повышения надежности ПО. На основе которых составим

алгоритм проверок ситуаций и таблицу решений.

2.2. Формализация модели адаптации, разработка проверок ситуаций и таблицы решений

Как выше отмечалось, основное назначение адаптации сводится к выработке адаптирующего воздействия на возмущения среды.

Из двух существующих классов адаптации с моделью и без нее, наименьшей сложностью отличается первое. При этом в самой адаптации с моделью из-за довольно удобной схемы реализации и большой эффективности использования − априорная адаптация в сравнении с апостиорной адаптацией более целесообразна. Действительно, трудно придумать что-либо проще и эффективнее, чем сразу и заранее заготовленным образом реагировать на изменение состояния среды [43]. Поэтому для эффективного управления в узле АК выберем и решим задачу априорной адаптации, позволяющей улучшить качество работы узла.

Модель априорной адаптации для области данных узла АК можно представить в виде рис.2.4. Модель составляют:

- структуры данных узла АК

$\mathbf{C = (C_1 , C_2 , \dots , C_{m1})}$

- методы обнаружения ошибок

$\mathbf{Мо = (Мо_1, Мо_2, \dots , Мо_{m2})}$

- проверки оценки ситуаций

$\mathbf{П = (П_1 , П_2 , \dots , П_{м3})}$

– критические ошибки происходящие из-за возмущений среды

$\mathbf{Кр=(Кр_1, Кр_2, \dots , Кр_{м4})}$

- методы восстановления нормального состояния работы узла

Рис. 2.4. Схема работы модели адаптации структур данных узла адаптивной коммутации

Обозначение: X – ситуация – состояние среды; Y – состояние объекта; U – адаптирующее воздействие.

- методы восстановления нормального состояния работы узла адаптивной коммутации

$$\mathbf{B} = (b_1, b_2, \dots, b_{M5})$$

– действия выполняемые на основе таблицы решений

Рш =(Рш$_1$, Рш$_2$, …, Рш$_{м6}$)

Последовательность работы механизма (обобщенного алгоритма) модели априорной адаптации для области данных узла АК заключается в следующем.

1. Опознание наименования структуры данных (**С**) адаптирующим средством.

2. Контроль состояния выбранной структуры с использованием методов обнаружения (**Мо**).

3. Проведение проверки оценки ситуаций (**П**). Определение: существуют критические ошибки или не существуют.

4. Выполнение операций, когда критические ошибки существуют

4.1. Выбор действий **Рш** из таблицы решений,

4.2. Выполнение действий по восстановлению (**В**) нормального состояния.

4.3. Передача управления в адаптирующее средство.

5. Выполнение операций, когда ошибки не существуют

5.1. Передача управления в адаптирующее средство,

6. Осуществление адаптирующего воздействия (**U**) на структуры данных узла.

Согласно этому алгоритму далее будут изложены следующие вопросы:

- выбор и разработка методов обнаружения ошибок;
- составление таблицы типов критических ошибок и методы их обнаружения;
- разработка проверок оценки ситуаций;
- выбор и разработка методов восстановления нормального хода работы узла АК;
- построение таблицы решений и описание работы с ней.

Рассмотренные ранее методы контроля и восстановления предназначались для повышения надежности вычислительных систем.

Применительно к узлам АК эти методы не были исследованы. В данном контексте рассматриваются существующие методы повышения надежности с учетом их доработки для узла АК.

Для многих методов обнаружения ошибок по контрольной сумме, сравнению с копией, сравнению со списком, по смысловому значению вероятность обнаружения равна единице. Для метода сравнения по предельным знамениям (МОПР) аналогичной оценки не имеется. Для ПО узла адаптивной коммутации такая оценка получена нами. Исследования показали, что метод МОПР с вероятностью 0.33 может обнаружить одиночные ошибки, с вероятностью 0.55 −двоичные, с вероятностью 0.7, 0.8 −трехкратные и четырехкратные ошибки и т.д. (результаты получены с использованием специально разработанной процедуры). Как видно из полученных результатов, вероятность обнаружения ошибки имеет весьма малые значения при наличии однократных ошибок. Использование метода предельных значений в сочетании с методами сравнения по модулю позволяет увеличить вероятность обнаружения ошибки до единицы.

Программная реализация методов обнаружения ошибок показала, что по сложности реализации эти методы можно расположить в следующей последовательности: МОПР, МОСК.МОЭСР, МОКС, МОМД (см. табл.2.5).

Оставшиеся 2 метода − проверки по смыслу и по номенклатуре не представляют практического интереса из-за дополнительных затрат, требуемых для их применения.

Сравнение методов обнаружения по критериям сложности и вероятности обнаружения ошибок позволяет рекомендовать для обнаружения критических ошибок ПО узла АК следующие методы: сравнения с копией, проверка на предельное значение, поэлементное сравнение и сравнение по модулю. В табл. 2.2. описывается предикат выполнения каждого из этих методов и выявляемые при этом критические ошибки ПО узла АК.

Таблица 2.2.

Методы обнаружения ошибок, используемые для ПО узла адаптивной коммутации и выявляемые ими критические ошибки

№	Метод обнаружения	Что контролируется и в чем заключается сущность метода	Обозначение	Выявляемые критические ошибки
1	Метод сравнения с копией	Сравнивается текущее α и предыдущее (запомненное) значение β некоторого параметра Y. Предикат успешного выполнения α= β	МОСК	кр.ош.1-4, кр.ош.7, кр.ош.17
2	Метод предельных значений	Проверяется значение некоторого параметра Y, на предельное значение. Предикат успешного выполнения α<ε< β, где α, β -минимальное и максимальное допустимые значения Y.; ε -текущее значение	МОПР	кр.ош.5-6, кр.ош.8, кр.ош.15-16
3	Метод поэлементного сравнения	Сравнивается значение элементов некоторого компонента Q с копией элементов G. Предикат успешного выполнения Q j = G j для всех j, где j=1,n; n − число элементов	МОЭСР	кр.ош.9-14
4	Метод сравнения	Сравнивается адрес j-го элемента(Ai) на кратность с адресом первого элемента (A_1). Предикат успешного выполнения Ai=(A_1+r),j , где j=2,n; n –число элементов, r - размер элемента	МОМД	кр.ош.5-6, кр.ош.8, кр.ош.15-16

Своевременная локализация ошибок и области данных осуществляется контролем элемента или байта (таблицы, контекста, пула, очереди ПО узла адаптивной коммутации). Для этого составлены программные проверки ситуаций. Перечень условных обозначений этих проверок, область их применения, причины ошибок и номер устранения критической ошибки и соответствующие методы восстановления приводятся в таблице решений (табл.2.3). В этой таблице используются следующие условные обозначения: **П(k,w),** где **П**- проверка, **k** -номер критической ошибки; **w**-число проверок, ис-

пользуемых для выявления **k**-ой критической ошибки; МВТР -метод восстановления по троированию данных; МВЧИ -частичная инициализация; МВЧПИ-частичная переинициализация.

Эксперименты для определения эффективности восстановления с помощью методов дублирования с КС и троирования показали, что вероятность устранения ошибок у них одинакова, однако время, затрачиваемое на метод троирования меньше.

Как видно, из табл.2.3 для восстановления нормального состояния ПО узла адаптивной коммутации выбраны методы МВТР, МВЧПИ, МВЧИ. Из них, первый метод был описан в п.1.3. Остальные два метода, учитывающие

специфику работы с очередями типа **FIFO** и **LIFO**, необходимо разработать. Для этого рассмотрим их значение и особенности.

Метод МВЧПИ служит для восстановления неисправной очереди путем

задания управляющим элементам начальных нулевых значений. При этом все ссылки (в их числе и неисправная ссылка) между буферами удаляются.

Потеря пакетов, за счет переинициализации неисправной очереди незначительна и в среднем равна ***n*** пакетам, тогда как при использовании контрольной точки это число составляет ***mxn*** пакетов, где ***n*** -среднее число пакетов в одной очереди, ***m***-число очередей в узле. В целом функции метода переинициализии не отличаются от метода контрольной точки. Разница заключается в объеме восстанавливаемых данных.

При использовании метода контрольной точки восстанавливается последнее состояние всех данных, тогда как использование метода частичной переинициализации позволяет восстанавливать именно ту часть области данных, где произошла ошибка.

Таблица 2.3

Диагностическая таблица

п п	Условное обозначение проверок	Причины ошибок	Восстанавливаемые компоненты	Методв восстанов-ления	Номер критичес-кой ошибки
1	2	3	4	5	6
1	П(1,1)	Слово поля управляющей таблицы	Значение указателя на компоненты	МВДБ	кр.ош.1
2	П(2,1)	Слово поля контекста тракта	Адрес контекста тракта	МВТР	кр.ош.2
3	П(2,2)	Слово поля контекста линии	Адрес контекста линии	МВТР	кр.ош.2
4	П(3,1)	Параметр ZOCSTB контекста тракта	Адрес контекста выходной таблицы КК	МВТР	кр.ош.3
5	П(3,2)	Параметр ZICSTB контекста тракта	Адрес контекста входной таблицы КК	МВТР	кр.ош.3
6	П(3,3)	Параметр ACSTB контекста линии	Адрес таблицы КК в контексте линии	МВТР	кр.ош.3
7	П(4,1)	Параметр KMPB входной/выходной таблицы контекста тракта	Адрес компенцнонного буфера входной/выходной таблицы КК	МВТР	кр.ош.4
8	П(4,2)	Параметр KMPB таблицы контекста линии	Адрес компенцнонного буфера в таблице КК	МВТР	кр.ош.4
9	П(5,1)	Адрес первого буфера очереди к блоку БКП	Очередь PNPQE/PLPQE	МВЧИ	кр.ош.5
10	П(5,2)	Адрес первого буфера очереди контекста выходного тракта	Очередь TPQEj, j-го тракта	МВЧИ	кр.ош.5
11	П(5,3)	Адрес первого буфера очереди контекста выходной линии	Очередь APQEj, j-го линии	МВЧИ	кр.ош.5
12	П(6,1)	Адрес первого буфера в пуле свободных длинных буферов	Пул PULBIG	МВЧИ	кр.ош.6

Продолжение табл.2.3

1	2	3	4	5	6
13	П(6,2)	Адрес первого буфера в пуле свободных коротких буферов	Пул PULLTL	МВЧИ	кр.ош.6
14	П(7,1)	Параметр текущее число буферов в очередях и пулах	Один из пулов или очередей	МВЧИ	кр.ош.7
15	П(8,1)	Параметр максимальное число буферов в очередях и пулах	Один из очередей или пулов	МВЧИ	кр.ош.8
16	П(9,1)	Слово в поле памяти таблицы	Значение основных данных	МВТР	кр.ош.9
17	П(10,1)	Слово в поле памяти маршрутной таблицы режима КП/КК	Значение в таблице CSRT/PSRT	МВТР	кр.ош.10
18	П(11,1)	Слово в поле памяти таблицах блоков БКП, БКК, БСК, БРК, ББА	Значение поля соответствующих данных	МВТР	кр.ош.11
19	П(12,1)	Поле памяти контекстов трактов/линий размером слово	Контекст тракта/линии	МВТР	кр.ош.12
20	П(13,1)	Поле памяти входной/выходной таблицы КК контекста тракта	Входная/выходная таблица КК ZICSTB/ZOCSTB контекста тракта	МВТР	кр.ош.13
21	П(14,1)	Поле памяти таблицы КК контекста линии	Таблица КК ACSTB контекста линии	МВТР	кр.ош.13
22	П(15,1)	Адрес первого буфера очереди	Одна из очередей типа FIFO	МВЧИ	кр.ош.15
23	П(16,1)	Текущее число буферов очереди тракта/линии	Одна из очередей типа FIFO	МВЧИ	кр.ош.16
24	П(17,1)	Адрес первого буфера в пуле	Одна из очередей типа LIFO	МВЧПИ	кр.ош.17

Другой метод восстановления частичной инициализацией предоставляет возможность восстановления потерянных буферов. Назначение этого метода описывается ниже.

Проведенный анализ функций основных блоков ПО (табл.1.1) узла адаптивной коммутации и их взаимосвязь с ООП показывает, что обращение к внутренним данным происходит при каждой сборке или разборке кадра. Если учесть тот факт, что данные ООП образуют древовидную структуру (см.рис.2.1), и доступ к требуемой компоненте производится через управляющую таблицу с одним указателем, к контексту трактов или линий – с двумя указателями, к таблицам входных или выходных КК – с тремя указателями, к компенсационным буферам – с четырьмя указателями, то без труда можно установить сложность работы со структурами данных узла АК. Так неверная каждая ссылка от указателей к структурам данных приводит к критическим последствиям. Возникает необходимость контроля значений указателей перед каждым ее использованием. В этом случае надо производить, проверки сохранности в виде оперативного контроля.

Оперативный контроль составляют:

– проверки П(1,1), П(2,1) ,.., П(3,1) ,.., П(4,1), П(4.2), служащие для контроля и восстановления сохранности указателей от таблицы МСТ до требуемой компоненты (1-уровень указателей), от таблицы TRUT или LINT до контекста тракта или линии (2-уровень указателей) от контекста тракта или линии до входных или выходных таблиц КК (3-уровень указателей), от входных или выходных таблиц КК до компенсационных буферов (4-уровень указателей)

– проверки П(5,1),…,П(6,1),…,П(6,2), служащие дли контроля сохранности адресов буферов очередей тина FIFO, пулов типа LIFO и восстановления этих очередей или пулов в случае существования в них неисправностей.

Остальные проверки составляют периодический контроль.

2.3. Исследование и описание организационно - функциональной структуры ПО систем реального времени

Рассмотрим функции, выполняемые проверками оперативного и периодического контроля.

Проверка П (1,1). Физически, ссылка от управляющей таблицы до требуемой компоненты ООП отождествляется значением адреса ячейки искомой компоненты. Этот адрес компоненты определяется в стадии инициализации [61] и заносится в МСТ. В связи с этим после инициализации можно отдублировать МСТ и создать ее копию СМСТ. Тогда при проверке П(1,1) достаточно сравнить методом МОЭСР значение ссылки от МСТ до требуемой компоненты с копией и при несоответствии этих данных считать, что существует ошибка. Устранение ошибки искаженного значения копии с СМСТ (метод МВДБ).

В целом функции проверок П(2,1,…,П(3,1),..,П(4,2) идентичны с функцией проверки П(1,1) − используются одни и те же методы обнаружения и восстановления, а также дополнительная информационная избыточность в виде копий, контролируемых указателей. Единственная разница заключается в назначении этих проверок, если П(1,1) служит для контроля указателей I-уровпя, то П(2,1), П(2,2) − 2 уровня и т.д.

Проверки П(5,1), П(5,2), П(5,3) контролируют адрес буфера в очередях типа FIFO. Для использования в составе этих проверок определяются параметры MINADROS и MAXADROS после инициализации (переинициализации), которые равны соответственно

MINADROS = <адрес 1-го буфера компоненты BBIG> и MAXADROS = <адрес последнего буфера компоненты BLTL>, где BBIG − свободные длинные буфера, BLTL −свободные короткие буфера. Сравнивается адрес буфера – ADRB1c параметрами MINADROS и MAXADROS. В случае, когда ADRB1 меньше MINADROS или ADRB1 больше MAXADROS (фиксируется ошибка (МОПР), и в контролируемой очереди уничтожаются ссылки на следующие буфера путем инициализации (метод МВЧИ) этой очереди. Параметрам

ADRB1, ADRB2 и NUMBERB присваиваются нулевые значения, а параметру MAXNB – максимальное число буферов MAXLFIFO. Для повышения достоверности (до единицы) обнаружения ошибок любой кратности в составе этих процедур реализован метод сравнения по модулю для определения кратности адреса длине буфера (короткого или длинного) в пределах MINADROS и MAXADROS. Восстановление неисправностей очереди в случае обнаружения ошибки производится тем же методом МВЧИ.

Аналогичные функции выполняют проверки П(6,1), П(б,2) для контроля буферов в пулах PULBIG, PULLTL, построенные по дисциплине LIFO.

Проверка П(9,1). Искажение значения в таблице основных данных TABPARAM может привести к различным последствиям для ПО узла адаптивной коммутации. Например, размер длинного буфера- SZ280 , определенный при инициализации, равен 280 байт. Если из-за искажения он в действительности имеет значение > 280, или < 280 байт, то блок БКП будет обрабатывать неверное число позиций буфера. При другой ошибке в таблице TABPARAM, когда время ожидания ответа от удаленного абонента искажено на меньшее значение, узел не дождется ответа и будет постоянно посылать повторную информацию и т.д.

Таблица TABPARAM строится при стадиях инициализации переинициализации. В этих стадиях необходимо отдублировать TABPARAM, создать копии CTABPARAM, DTABPARAM. Тогда метод позволяет при сравнении элементов указанной таблицы с элементами копии CTABPARAM и DTABPARAM обнаружить ошибку при несоответствии их значений. Методом восстановления служит МВТР выполняющий присвоение значения искаженному элементу таблицы TABPARAM.

Проверки П(10,1), П(11,1) по типу выполняемых функций идентичны проверке П(9,1) и контролируют сохранность значений данных группы @1: маршрутных таблиц, КП или КК – PSRT или CSRT; таблиц адресов контекстов трактов или линий TRUT или LINT; таблиц блока сборки кадров -TABBSB, разборки кадров – TABBRB; коммутации пакетов – TABBCP, коммутации

каналов − TABBCC, взаимодействия с абонентами – TABVAH, TABVAA, монитора управления –TABMDG, статистики и отображения статистики −TABSTAT, TABZSS, управляющей таблицы – МСТ.

Копии перечисленных компонент создаются после этапов инициализации или переинициализации блоком инициализации (БИ).

Проверки П(7,1), П(8,1), П(15,1),..., П(17,1) контролируют текущее число буферов в ПО узла адаптивной коммутации на двух уровнях. На первом уровне (выполняется лишь проверка П(7,1)) определяется во всех очередях и пулах текущее число буферов, затем они суммируются и сравниваются с параметром число буферов очередей и пулов - TEKBUF, определенным на стадии инициализации. В случае совпадения этих значений проверки завершают свою функцию, а в случае неравенства этих значений считается, что в какой-то очереди (пуле) произошла ошибка и осуществляется переход ко второму уровню.

На втором уровне осуществляется более детальный анализ каждой очереди (пула) с целью определения, какая из них содержит ошибку. Этот анализ заключается в следующем. Значение, содержащееся в параметре ADRB1 управляющего элемента очереди типа LIFO, должно бить равно одному из значений адресов с свободных буферов компонент BBIG или BLTL. Так, как при начальной инициализации под BBIG резервируется NBIGBUF, а для BLTL – NLTLBUF буферов, где NBIGBUF, NLTLBUF число длинных и коротких буферов. В дальнейшем блок БИ адреса длинных буферов связывает в пул PULBIG, а коротких − в пул PULLTL по дисциплине LIFO. Когда некоторый блок выбирает буфер с PULBIG (PULLTL), управление этим буфером переходит к этому блоку и адрес этого буфера заносится в ADRB1. Тогда проверка − действительно ли ADRB1 указывает на адрес первого буфера из BBIG или BLTL является смыслом контроля второго уровня.

Для этого составляются копии адресов буферов BBIG и BLTL в массиве ADRFBUF размером NBIGBUF + NLTLBUF в стадии инициализации.

Проверка П (5,1), которая выполняется на втором уровне, фиксирует

ошибку в очереди в случае, когда ADRB1=$ADRFBUF_j$, j=1,TEKBUF. Однако, в случае ADRB1=**0**, NUMBERB=**0** ошибка не фиксируется. Восстановление неисправностей очереди производится с частичной переинициализацией (метод МВЧПИ) этой очереди, аналогичной описанной в проверке П(5,1).

На этом функции проверки П(15,1) не завершаются. Далее эта проверка позволяет пройти по всем ссылкам управляющего элемента очереди. Если ADRB1 указывает на первый буфер, то адрес следующего буфера содержится в параметре ADRESBUFER буфера BUF (BUF определенный по типу TYPBUF [51]). Тогда операцией BUF=BUF->ADRESBUFER (операция команды языка программирования СИ) определим адрес 2-го буфера, еще одно повторение той же операции укажет на адрес 3-го буфера и т.д. В случае BUF= $ADRFBUF_j$ фиксируется ошибка в других ссылках очереди.

Работа по восстановлению в этом случае содержит следующие действия. Все связки между буферами, которые имеют адреса равные какому-нибудь буферу из ADRFBUF, сохраняются. Ошибочный буфер удаляется из списка очереди соответственно, в NUMBERB устанавливается реальное число буферов в очереди.

Как бы продолжением проверки П(16,1) является проверка П(17,1). Оиа анализирует все очереди, проходя по всем ссылкам, и фиксирует адрес имеющихся буферов. Затем эта проверка производит восстановление затерянных буферов после предыдущей проверки, связывая их в пул свободных длинных или коротких буферов. Для этого проверка П(17,1) сравнивает адреса имеющихся буферов в очередях с массивом и вставляют в указанные пулы не найденные в ADRFBUF буфера. При этом не найденные адреса буфера в элементе, индекс которого меньше NBIGBUF + I, заносятся в пул длинных буферов, в случае когда индекс больше NBIGBUF − в пул коротких буферов. Использованный при этом метод восстановления именуется частичной переинициализацией (МВЧПИ).

Другие проверки второго этапа контроля очередей П (8,1), П(16,1) служат для контроля частных параметров: < текущее число буферов > − NUMBERB,

NUMBUF, <макс.число буферов> – MAXNB, MAXBUF, сравнивая их на предельные возможные значения методом МОПР, и сохраненного значения (MOCK):

$0 \leq NUMBERB \leq MAXLFIFO \quad MAXNB(MAXBUF) = MAXLFIFO$

В случае несоответствия фиксируется неисправность и этим параметрам присваиваются их начальные значения, определенные при инициализации.

Проверки П (13,1), П(14,1), П(12,1) служат исключительно для контроля данных группы квазипеременных (@2) и контролируют сохранность данных: таблицы КК контекста тракта – TCSTB, и линии – LCSTB, контекстов трактов и линий – TRUT и линий – LINT. При этом эти таблицы сравниваются, соответственно со своими копиями CTCSTB, DTCSTB, CLCSTB, DLCSTB, CTCT, DTCT, CLCT, DLCT, которые создаются блоком БКК, после организации или ликвидации КК. В остальном работы этих проверок аналогичен проверкам П(9,1) - П(11,1).

На основе выше разработанных проверок ситуаций и таблицы решений можно составить алгоритм адаптирующего блока узла АК. Данный алгоритм будет включать следующие действия: запуск той или иной проверки ситуаций из большой совокупности проверок П(1,1), П(2,1),..., П(17,1), анализ конкретной ситуации после проверки, выбор решения из таблицы диагностики, осуществление адаптирующего воздействия. Следующий параграф посвящен программной реализации данного алгоритма.

2.4. Алгоритмизация адаптирующих средств контроля и восстановления

В данном параграфе разрабатываются программные средства повышения надежности узла адаптивной коммутации, которые включают в себя функции и назначения проверок ситуаций, таблицы решения и адаптирующего средства.

Для исследования разработанных методов обнаружения и устранения ошибок, приведенные в табл.2.1 проверки ситуаций, должны быть реализованы программно. В табл.2.4 приводится наименование программного средства, код,

назначение и связь этого средства с проверками ситуаций, приведенными в предыдущем подразделе.

Таблица 2.4.

Программные модули контроля и восстановления для повышения надежности ПО узла адаптивной коммутации

пп	Код идентификатора модуля	Назначение программных модулей	Перечень проверок входяющий в состав модуля
1	2	3	4
1.	ОК1	Контроль сохранности и восстановления указателей 1-уровня	П(1,1)
2.	ОК2	Контроль сохранности и восстановления указателей 2-уровня	П(2,1), П(2,2)
3.	ОК3	Контроль сохранности и восстановления указателей 3-уровня	П(3,1), П(3,2), П(3,3)
4.	ОК4	Контроль сохранности и восстановления указателей 4-уровня	П(4,1), П(4,2)
5.	ОК5	Контроль доступа к адресам буферов очередей типа FIFO и устранения критической ошибки	П(5,1), П(5,2), П(5,3)
6.	ОК6	Контроль доступа к адресам буферов очередей типа LIFO и устранения критической ошибки	П(6,1), П(6,2)
7.	KONTAB	Контроль сохранности и восстановления значений группы @ (постоянных данных)	П(9,1) − П(11,1)
8.	KONKBAZI	Контроль сохранности и восстановления значений группы @**2** (квазипеременных данных)	П(12,1) − П(14,1)
9.	PKI	Запуск модулей KONTAB , KONKBAZI	—
10.	PROW3	Сравнение текущего суммарного числа буферов во всех очередях и пулов с начальным числом буферов	П(7,1)
11.	ANALOS, OSANAL	Контроль правильности сохранения значений в управляющих элементах очередей типа FIFO	П(8,1), П(15,1), П(16,1)
12.	SHIBOS	Контроль и восстановления буферов в пулах типа LIFO в составе ANALOS	П(17,1)
13.	PERKONTR	Запуск процедур периодического контроля PKI , PR0W3, ANAL0S	-
14.	KBAZITRL, COPTRLIN	Снятие копий контекста тракта/линии после организации или ликвидации КК	-
15.	KBAZIKK, COPTKK	Снятие копий входной/выходной таблицы КК контекста тракта/линии	-
16.	KBAZID	Запуск процедур KBAZITRL, KBAZIKK	-
17.	COPTAB	Снятие копий со значений группы постоянных	-

		данных после начальной инициализации	
18.	COPTABUD	Снятие копий со значений группы квазипеременных данных после начальной инициализации	-
19.	SHICOP	Снятие копий с адресов длинных и коротких буферов	-
20.	PKCOP	Запуск процедур COPTAB, COPTABUD, SHICOP	-

Все проверки оперативного контроля реализуются в виде отдельных процедур. Выполнение каждой из них будет происходить автономно. В функции этих процедур входит: выбор действий из таблицы решений (табл.2.3) для восстановления нормального состояния.

В табл. 2.4. приводится также все процедуры, которые составляют периодический контроль – PERKONTR. Процедура PERKONTR, разработана в виде единого программного блока (блок БКП, БСК и др.) и входит в состав ПО узла адаптивной коммутации, как отдельная задача со своим флагом, приоритетом [40], В дальнейшем будем называть ее блоком контроля и диагностики (БКД).

Процедура копирования контекстов трактов или линий таблиц КК будет запускаться блоком БКК после обслуживания каждой заявки на организацию или ликвидацию КК. Для этого достаточно включить в блок БКК одну команду. После стадии инициализации вызовом процедуры РКСОР, блок БИ предоставляет возможность снять копии начальных значений постоянных и квазипеременных данных ООП узла.

Файлы IZBDAT, MONDIAG служат для этих целой и будут содержать, копии этих данных. На рисунке 2.5 показана связь разработанных процедур с данными, содержащими информационную избыточность.

Все процедуры программного комплекса повышения надежности ПО узла адаптивной коммутации реализованы на языке СИ [22] для СМ ЭВМ.

В соответствии с приведенной структурой программного комплекса и логической связью процедур с файлами, созданными при разработке

комплекса, можно определить место предлагаемых процедур оперативного контроля, процедур копирования данных после организации или ликвидации КК (KVAZID) или в стадии инициализации (РКСОР) и блока БКД в ПО узла адаптивной коммутации (рис.2.6).

Схема функционирования блока БКД и процедур оперативного контроля в составе основных блоков узла АК показана на рисунке 2.6. В этой схеме блок БКД будет запускаться с низким приоритетом, что позволяет минимизировать временные затраты связанные с потерей эффективности узла.

2.5. Систематизация программной реализации разработанных средств

Для своевременного обнаружения ошибок, приводящих к отказу ПО узла АК, часть разработанных программных средств восстановления реализованы в виде процедур оперативного контроля. В данном параграфе рассматриваются основные операции, осуществляющие контроль.

Данные процедуры оперативного контроля: ОК1, ОК2, ОК3, ОК4, ОК5, ОК6, выполненные в виде отдельных подпрограмм (программных модулей), вызов которых осуществляется командами:

– ОК1(С_МСТ),

где, С_МСТ – номер параметра таблицы МСТ. В файле IZBDAT.C, для всех данных МСТ определены символические параметры.

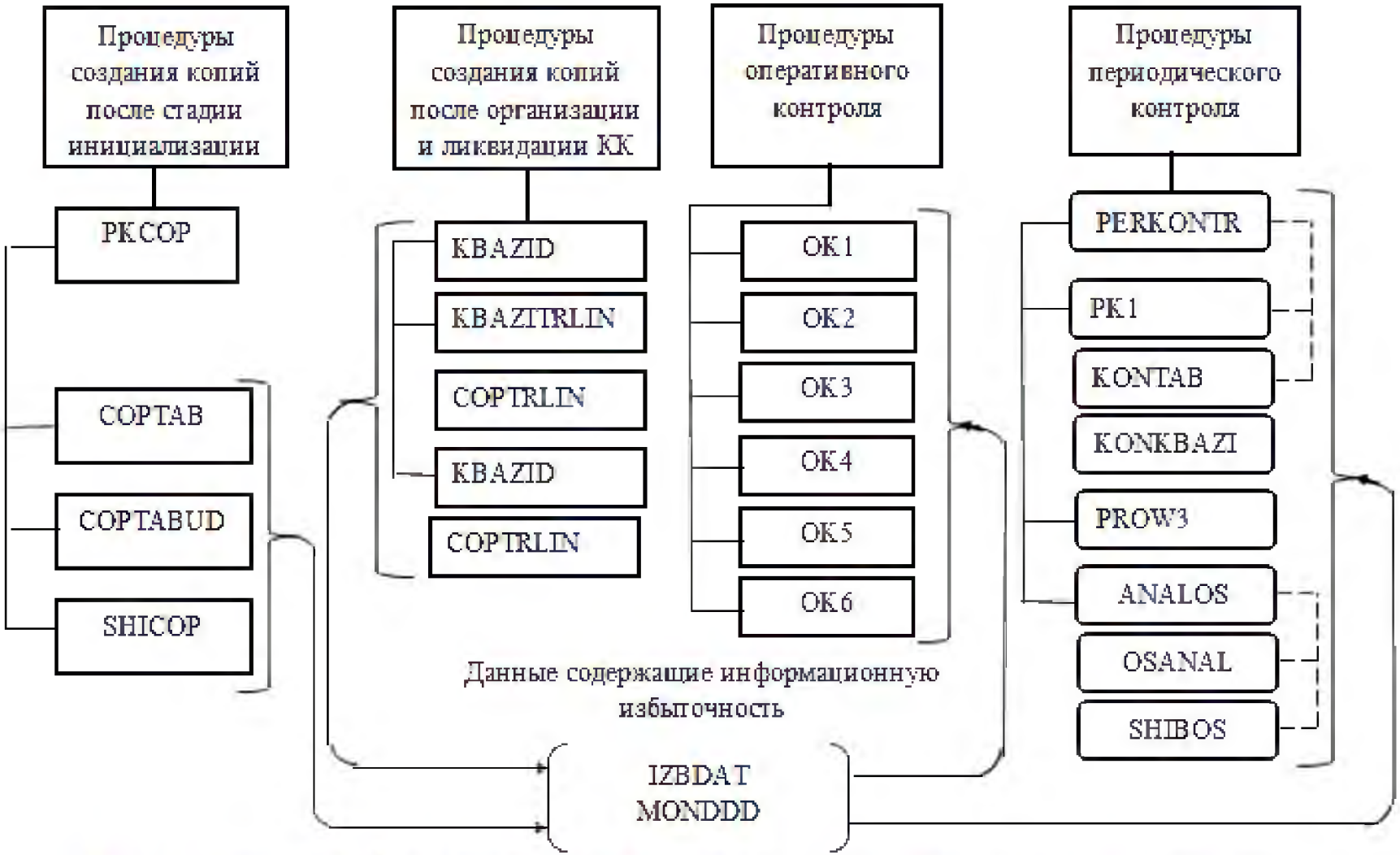


Рис.2.5. Связь модулей повышения надежности ПО узла адаптивной коммутации с данными, содержащими информационную избыточность

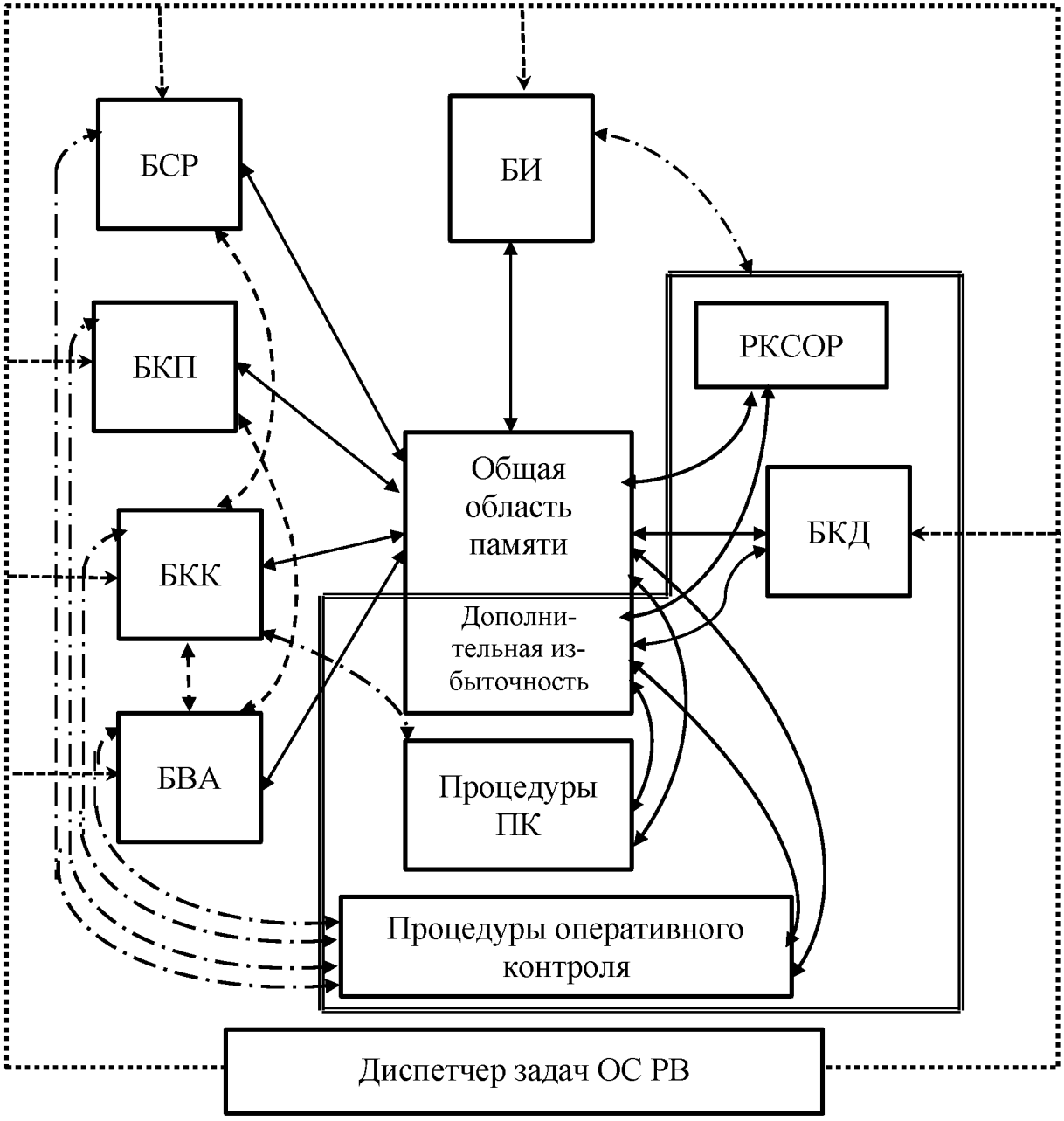


Рис. 2.6. Местонахождение разработанных процедур повышения надежности в ПО узла адаптивной коммутации.

Условные обозначения:

- ·····> – передача управления диспетчером задач ОС РВ;
- ◄- - - -► – передача управления с помощью флагов событий и директив ввода-вывода;
- ◄-·-·-► – запуск процедур повышения надежности(оперативный контроль) и снятие копий;
- ◄——► – доступ к общей области памяти;
- ═══ – зона разработанных средств контроля и восстановления.

Например, для проверки адреса таблицы TRUT вызывается следующая команда ОК1(TYP_TRUT), а для таблицы PSRT - 0К1(TYP_PSRT);

-ОК2(PILTRLIA, NOM_TRLIN),

где: PR_TRLIN−признак тракта или линии, NOM_TRLIN − номер тракта линии. Признаком тракта служит символическое обозначение PR_TRACT, а линии – PR_LINE, которым в IZBDAT.C присвоены, соответственно 1 и 0;

-ОК3(PR_TRLIN, NOM_TRLIN, PR_BB, NOM_КК),

где: PR_BB − признак входного (PR-BX) или выходного (PR_BIX) КК; NOM_КК − номер КК. В файле IZBDAT.C PR_BX присвоена 1, а PR_BTX 0. Например, вызов ОК3 (PR_TRACT, 0, PR_BX,**2**) означает произвести контроль адреса входной таблицы 2-го КК 0-го контекста тракта;

− ОК4 (PR_TRLIN, NOM_TRLIN, PR_BB, NOM_КК)

Обозначения параметров см.выше;

− Ок5(BUF, ϒOSER, “NAME”),

где BUF - буфер, определенный как TYPBUF, ϒOSER -управляющий элемент очереди типа FIFO (TYPFIF0), “NAME” - имя очереди, ϒ − знак, указывающий на адрес. Пример вызова ОК5 BUF, MCT. PNPQE, "PNPQE") означающий "произвести контроль входной очереди тракта к блоку БКП" ;

− ОК6 (BUF, ϒPUL , “NAME”),

где: PUL - управляющий элемент очереди пула типа LIFO. Пример вызова ОК6 (BUF, MCT.PULLTL , "PULLTL") означающий "произвести контроль пула свободных коротких буферов".

Порядок выполнения процедур оперативного контроля произвольный. Разработка в виде отдельных процедур (подпрограмм) позволяет пользователю подключить необходимую в каждой задаче ПО узла адаптивной коммутации проверку. Необходимость той или иной проверки устанавливается из таблицы.1.1, беглый анализ которой показывает во всех основных блоках ПО узла, кроме блока БКП, целесообразность использования всех процедур

оперативного контроля. К блоку БКП процедуры ОК3, ОК4 не подключаются, так как БКП с внутренними данными входных, выходных таблиц КК, с компенсационными буферами не взаимодействует.

Подключение этих процедур к блокам ПО узла адаптивной коммутации производится добавлением в исходный текст программы этих блоков команд вызова вышеприведенных процедур оперативного контроля. Затем осуществляется трансляция и компоновка блока (задачи).

В данной части рассматривается алгоритм работы средств периодического контроля. Средства периодического контроля полезны для обнаружения и устранения критических ошибок приводящие к уменьшению эффективности работы ПО узла АК. Основными процедурами периодического контроля являются процедуры РК1, PROW3, ANALOS, которые выполняются под управлением блока БКД. Укрупненная схема работы блока БКД сводится к следующему.

1. Начало работы блока БКД
2. Запуск процедуры РК1.
3. Запуск процедуры PROW3.
4. Анализ работы процедуры PROW3.

4.1.Если неисправность существует (процедура PROW3 служит для обнаружения критической ошибки кр.ош.7, когда текущее суммарное число буферов во всех очередях не равно запомненному при инициализации), то переход к п.5.

4.2. Если неисправность не существует, то переход к п. 6.

5. Запуск процедуры ANALOS.
6. Конец работы блока БКД.

Алгоритм процедур, входящих в состав блока БКД, программно реализовано.

Из приведенной выше укрупненной схемы работы блока БКД, видно, что контроль постоянных данных ООП (процедура РК1 будет производиться во всех случаях, когда процессорное время занимает блок БКД. Сравнение

текущего суммарного числа буферов всех очередей с определенным числом выполняется при каждом запуске периодического контроля. Анализ состояния очередей и пулов с целью обнаружения в них неверных ссылок, искажения параметров, блокировок будет выполняться в реже (запуском процедуры ANALOS), когда PROW3 определит, что потеря буфера в ПО узла адаптивной коммутации имела место.

Основными задачами процедур копирования являются создания копий постоянных и квазипеременных данных.

После начальной инициализации и переинициализации (эти функции выполняет блок БИ), необходимо произвести троирование данных (снятие двух копий). Троироваиие данных производит процедура РКСОР, для запуска которой достаточно включить в исходный текст блока БИ команду РКСОР (переменные);

При обслуживании заявки на организацию или ликвидацию КК блоком БКК, происходят изменения в таблицах КК, контекстах тракта или линии. Эти изменения фиксирует процедура KBAZID, которая осуществляет троирование данных. Для запуска процедуры в блоке БКК достаточно включить в исходный текст программы команду

KBAZID (PR_TRLIN, NOM_TRLIN, PR_BB, NOM_KK),

где: PR-TRLIN - признак тракта или линии, NOM_TRLIN - номер тракта или линии, в котором было произведено изменение, PR_BB - признак входной или выходной таблицы КК, NOM_KK - номер таблицы КК.

2.6. Критерии и показатели оценки сложности

Работа программных средств контроля и восстановления накладывает временную и информационную избыточность. Целесообразность использования этих средств определяется по вводимой ими избыточности как по времени, так и по памяти ЭВМ.

Для оценки сложности разработанных процедур контроля были разработаны более 10 программ с тестовыми примерами на языке СИ для

операционной системы ОС РВ, и контрольные запуски выполнялись в количестве 2000 прогонов с каждой программой на ЭВМ СМ-1600 и СМ-1420. Полученные результаты характеризует таблица 2.5.

Табл. 2.5

Временные затраты процедур оперативного контроля, блока БКД и средств копирования

№	п п	Идентификатор процедуры	Временные затраты, в сек.	
			СМ-1600	СМ-1420
I		Оперативный контроль	0,00145	0.00078
	I.	ОК1	0,00020	0 ,00015
	2.	ОК2	0,00017	0,00009
	3.	ОК3	0,00035	0,00019
	4.	ОК4	0,00036	0,00019
	5.	ОК5	0,00015	0,00008
	6.	ОК6	0,00014	0,00008
II		Периодический контроль(БКД)	0,03698	0,01993
	1.	PKI	0,03 631	0,01962
	2.	PROW3	0,00067	0,00031
II		Процедуры копирования		
	1.	PKCOP	0,04315	0,0233
	2.	KBAZID	0,0363	0,0157

Приведенный нами эксперимент на СМ-1600 и СМ-1420 позволил собрать большой статистический материал. Обработка этого статистического материала показала, что при использовании всех процедур оперативного контроля в одном блоке, когда этот блок занимает процессорное время минимум на $A^{(1)}$=0,2с, максимум на $A^{(2)}$= 1с, временная избыточность для средств контроля и восстановления составит для $A^{(1)}$ – 0,7 % или 0,39 % и для $A^{(2)}$ – 0,14 % или 0,07 % соответственно на ЭВМ СМ-1600 и СМ-1420.

Информационная избыточность, вводимая в ПО узла адаптивной коммутации с использованием средств контроля и восстановления, составляет не более 20 % на общую область данных ООП узла АК.

Программная избыточность, вводимая в ПО узла АК с использованием средств контроля и восстановления, определяется следующим образом. Для узла АК функциональными программами, скомпонованными в виде отдельных задач, являются программы БКП, БИ, БКК, БСК, БРК, БВА, сбора статистики (ЗСС), отображения статистики (ЗОС), управления (ПУ), драйвера ввод-вывода (ПДВВ). С разработкой программных средств контроля и восстановления (ПСКВ) число программ узла АК увеличится на одну программу, и в предположении, что каждый блок занимает одинаковую размерность, ПСКВ вводит избыточность в среднем порядка 12 %.

ВЫВОДЫ

1. Анализ и классификация структур данных общей области памяти узла адаптивной коммутации позволяет установить местонахождение критических ошибок, происходящих из-за возмущающего воздействия среды на объект управления. Часть критических ошибок приводит к отказу в узле (к ним относятся и контексты, имеющие ссылки на другие структуры, указатели в буферах, в очередях (пулах)), а другая часть − к уменьшению пропускной способности.

2. Разработанные проверки ситуаций, таблица решений, определенные адаптирующие воздействия, организованные в виде отдельных программных модулей на каждый тип возмущений, что позволяет их рассматривать как автономные адаптирующие средства. Другими словами автономное адаптирующее средство есть отдельная программа, предназначенная для повышения надежности в узле АК.

3. Разработанные методы обнаружения и устранения критических ошибок позволяют вести контроль и управление надежностью узла АК и эффективны при их реализации в виде:

а) оперативного контроля для критических ошибок, приводящих к уменьшению готовности системы;

б) периодического контроля для критических ошибок, снижающих пропускную способность узла.

4. Экспериментально установленная оценка сложности реализации программных средств повышения надежности с целью увеличения эффективности управления потоками данных в узле АК находится около нижнего предела времени, рекомендуемого для систем реального времени.

ГЛАВА III. АНАЛИТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ОТКАЗОУСТОЙЧИВЫХ СИСТЕМ В УСЛОВИЯХ АПРИОРНОЙ АДАПТАЦИИ

В данном разделе предложена аналитическая модель функционирования ПО узла адаптивной коммутации вычислительной системы, в которой происходят сбои и отказы.

Разработка аналитической модели вызывается необходимостью исследования влияния сбоев и отказов на пропускную способность и готовность ПО узла адаптивной коммутации; управления интервалом включения периодического контроля при различных коэффициентах загрузки; определения средней потери эффективности из-за оперативного и периодического контроля и восстановления.

При разработке модели принимаются допущения о пуассоновском потоке всех событий, происходящих в ПО узла адаптивной коммутации.

3.1. Обоснование выбора непрерывной цепи Маркова для моделирования работы адаптивных систем

Марковские случайные процессы называют марковской цепью, когда состояние и время в них дискретны, и непрерывной цепью Маркова – в случае непрерывности времени [6].

На практике значительно чаще сталкиваются со случайными процессами с непрерывным временем. Например, выход из строя (отказ) любого элемента аппаратуры может произойти в любой момент времени; окончание ремонта (восстановление) этого элемента также нельзя заранее зафиксировать и т.д.

В такой цепи переход системы (S) из состояния в состояние представляется происходящим под влиянием некоторых потоков событий с интенсивностью λij . Если все эти потоки пуассоновские (т.е. ординарные и без последствия, с постоянной или зависящей от времени интенсивностью), то процесс, протекающий в системе S, будет Марковским [39]. Кроме того, если все потоки событий, переводящие систему из состояния в состояние, являются

простейшими (т.е. стационарными пуассоновскими с постоянными интенсивностями λij ,то в некоторых случаях существуют финальные (или предельные) вероятности состояний, не зависящие от того, в каком состоянии система S находилась в начальный момент.

Система, для которой существуют финальные вероятности, называется эргодической, а соответствующий случайный процесс ─ эргодическим.

Для существования финальных вероятностей состояний одного условия **λij =*const*** недостаточно. Должно выполняться еще следующее условие: если система S имеет конечное число состояний S1,S2, • • •, Sn , то для существования финальных вероятностей достаточно, чтобы из любого состояния системы можно было (за какое-то число шагов) перейти в любое другое.

Для получения финальных вероятностей состояний (ФВС) пользуются следующим мнемоническим правилом: для каждого состояния суммарное значение выходящего потока вероятности равен суммарному входящему. Другим способом получения ФВС является решение алгебраических уравнений получаемых из дифференциальных уравнений Колмогорова при равенстве левой части (производных) нулю [7,29].

Очень многие системы, предназначенные для обслуживания каких-либо заявок (требований), поступающих на них в случайные моменты времени, исследуются аппаратом теории массового обслуживания (ТМО), а сами исследуемые системы именуются системами массового обслуживания (СМО).

Аналитическое исследование СМО является наиболее простым, если все потоки событий, переводящие ее из состояния в состояние простейшие (стационарные пуассоновские). Для СМО это допущение означает, что поток заявок, так и поток обслуживаний – простейшие.

В силу утверждения[8], если все потоки событий простейшие, то процесс, протекающий в СМО, представляет собой марковский случайный

процесс с дискертными состояними и непрерывным временем (непрерывная цепь Маркова), для которой финальные вероятности

состояний существуют при соблюдении вышеупомянутых условий.

Известно [35,67], что в сетях передачи данных предположение о пуассоновском распределении потоков поступления и обслуживания заявок, является единственным предположением, близким к действительности. В работе [67] обосновывается допустимость марковских цепей для анализа работ узлов ВС.

В связи с этим и в силу вышеупомянутого утверждения строится аналитическая модель в предположении, что все события, происходящие в узле АК, составляют простой пуассоновский поток.

3.2. Исследование и разработка аналитической модели

Процесс функционирования ПО узла адаптивной коммутации в вычислительной системе с неисправностями без использования процедур контроля и восстановления (процесс А) и с ними (процесс Б) можно представить в виде размеченного графа (рис.3.1)

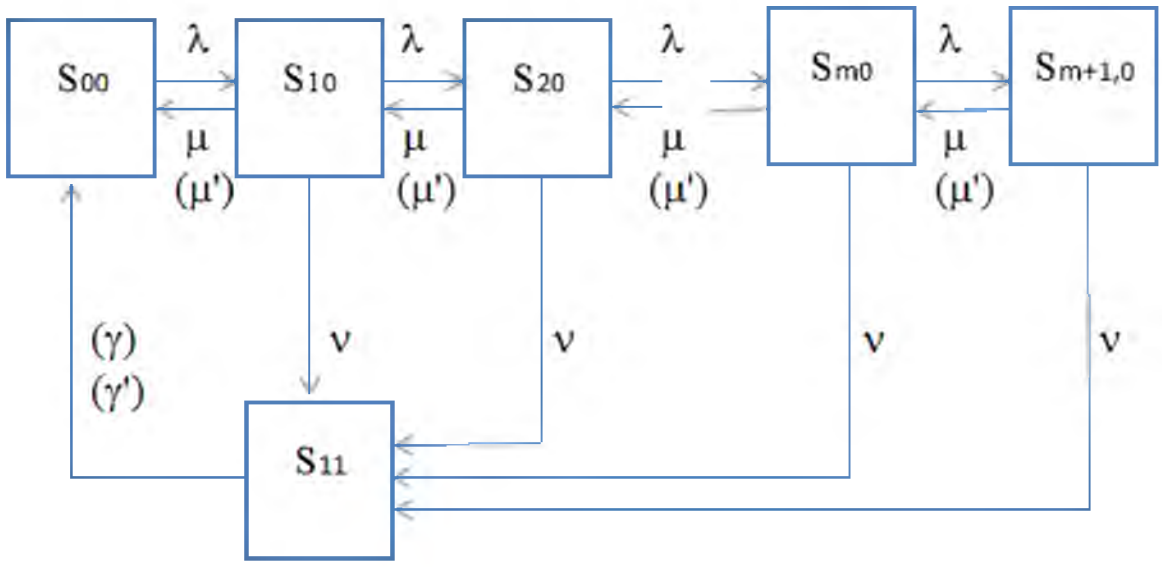


Рис. 3.1. Аналитическая модель процесса функционирования ПО узла адаптивной коммутации в вычислительной системе с неисправностями

где: Si_0 (i=0, m+1), S_{11} состояния СМО:

S_{00} – СМО свободен и исправен;

S_{10} – канал занят и исправен, очереди нет;

S_{20} – канал занят и исправен, в очереди одна заявка;

Sm_0 – канал занят и исправен, в очереди m-1 одна заявка;

$Sm_{+1,0}$ – канал занят и исправен, в очереди m плюс одна заявка;

S_{11} – канал неисправен, восстанавливается;

λ – интенсивность поступления заявок;

μ – интенсивность обслуживания заявок без использования оперативного контроля;

μ' – интенсивность обслуживания заявок с использованием оперативного контроля;

ν – интенсивность перехода в состояние отказа;

γ – интенсивность восстановления оператором;

γ' – интенсивность (программного) восстановления;

Pij – вероятность перехода в состояния Sij , где i=0,m+1, j=0,1.

Алгебраическое уравнение для ФБС процесса А выглядит следующим образом:

$$\lambda P_{00} = \mu P_{10} + \gamma P_{11}$$

$$zP_{10} = \lambda P_{00} + \mu P_{20}$$

. . .

$$zPm_0 = \lambda Pm_{-1,0} + \mu Pm_{+1,0} \qquad (3.1.)$$

$$(\mu+\nu)Pm_{+1,0} = \lambda Pm_0$$

$$\gamma P11 = \sum_{i=1}^{m+1} Pio$$

где $z = \lambda+\mu+\nu$

С учетом нормировочного условия

$P_{00} + P_{10} + \cdots + Pm_{+1,0} + P_{11} = 1$

отбрасывая первое уравнение и выражая через $Pm_{+1,0}$ вероятности, получим рекурентные формулы расчета для коэффициентов

$$Ak = \begin{cases} 1 & \text{, при } k = m+1 \\ (\mu+\nu)/\lambda & \text{, при } k = m \\ (zAk+1 - \mu Ak+2)/\lambda & \text{, при } 0 \le k \le m-1 \end{cases} \qquad (3.2)$$

Коэффициент Ak позволяет вычислить Pk0 через $Pm_{+1,0}$, т.е.

$$Pk_0 = Ak\ Pm_{+1,0} \qquad (3.3)$$

$$P_{11} = A_{11}\ Pm_{+1,0} \qquad (3.4)$$

где,

$$A_{11} = \frac{\nu \sum_{i=1}^{m+1} Ai}{\gamma} \qquad (3.5)$$

С учетом нормировочного условия имеем

$$Pm_{+1,0} = \frac{1}{A11+\sum_{i=1}^{m+1} Ai+Ao} \qquad (3.6)$$

или

$$Pm_{+1,0} = \frac{1}{\frac{\nu}{\gamma}(1+\frac{\gamma}{\nu})\sum_{i=1}^{m+1} Ai+Ao} \qquad (3.7)$$

Вычисление коэффициентов следует осуществлять в следующей последовательности $Am_{+1,0}, \cdots, A_0, A_{11}$

Подставляя Ak в (3.7) или Ak , A_{11} в (3.6) находим Pm_{+10}. Вероятности Pk (k=1,m) и P11 находятся по формулам (3.3), (3.4).

Данная модель строилась на базе модели, приведенной в работе [6], и является более обобщенной. Так отбросив m = 2,3…,n можно прийти к базовой модели.

Согласно [6] относительная пропускная способность вычисляется по формуле

$$Q = (1\ Ротк)\ (1- P11)p\ p \qquad (3.8)$$

где: Ротк - вероятность отказа в обслуживании, (1-P11) вероятность нахождения в работоспособном состоянии, p - вероятность того, что канал не откажет за время обслуживания заявок

($p = \mu/(\mu+\nu)$), p - вероятность безошибочного обслуживания при неисправности, когда сбой за собор не влечет отказ ВС(где p -отношение числа, правильно обработанных заявок к общему числу принятых заявок).

Данная схема позволяет определить коэффициент потери эффективности Knom, которая характеризуется вероятностью нахождения в состоянии восстановления после ошибок — P11. Тогда можно вычислить готовность ВС процесса А по формуле

$$Krq = 1 - Knom \qquad (3.9)$$

Для процесса функционирования ПО узла адаптивной коммутации в ВС с неисправностями с использованием процедур контроля и восстановления (процесс Б) аналитическая модель аналогична модели процесса А. Разница лишь в том, что вместо параметров μ и γ, используются μ' и γ'. Параметр σ' учитывает временную избыточность внесенную оперативным контролем в ПО узла АК. Если $\mu = 1/Тобс$, то

$$\mu' = \frac{1}{T'обс} = \frac{1}{Тобс + \sigma'} \qquad (3.10)$$

где Тобс, Т'обс - время обслуживания одной заявки без оперативного контроля и с ним.

Получена формула для определения числа обработанных заявок, после которых периодический контроль не будет вносить временную избыточность в работу По узла АК. Эта формула выглядит следующим образом

$$n = \frac{Тпк}{\frac{\beta з - \beta р}{\lambda} - Ток}, \quad \frac{\beta з - \beta р}{\lambda} > Ток \qquad (3.11)$$

где: λ − интенсивность поступления заявок; Тпк − время выполнения периодического контроля, реализованного в БКД, Ток − суммарное время оперативного контроля; $\beta з$ −заданное значение коэффициента загрузки, $\beta р$ −значение его реальное.

При получении формулы (3.11) предполагалось, что в процессе работы узла адаптивной коммутации из-за нестационарности потока заявок реальная загруженность сети может иметь значение βр, меньшее βз, т.е.

$$\beta p < \beta з \qquad (3.12)$$

а значение βз (коэффициент загрузки ПО узла АК с использованием оперативного и периодического контроля) меньше

$$\beta' p \leq \beta p + \lambda \sigma' \qquad (3.13)$$

Используя правую часть неравенства (3.12) и формулу (3.13), определим средние временные ограничения на разовое выполнение средств контроля и восстановления

$$\beta p + \lambda \sigma' \leq \beta з \;, \qquad \sigma' \leq \frac{\beta з - \beta p}{\lambda}$$

В связи с этим σ' суммируется из затрат оперативного контроля (Ток) и остаточного времени (σпк)

$$Ток + \sigma пк \leq (\beta з - \beta p) / \lambda \qquad \text{или}$$

$$\sigma пк \leq (\beta з - \beta p) / \lambda - Ток \qquad (3.14)$$

Время выполнения периодического контроля (Тпк) может превышать σпк в n раз. Поэтому

$$Тпк/\sigma пк = (\beta з - \beta p) / \lambda - Ток \qquad (3.15)$$

Параметр n в формуле (3.15) является средним числом обслуженных заявок, после которых запуск периодического контроля, не налагает временную избыточность на ПО узла АК.

Пример. Пусть: Тпк = 0.003с, Ток = 0.0009с, βр = 0.60, βз = 0.65, λ = 40, μ = 66.66, тогда

n = (0.003/ (0.65-0.6)/40 – 0.0009) = 9.23

Полученное *n* показывает, что после 10 обслуженных заявок можно производить периодический контроль с минимальными временными затратами.

В другом примере можно построить зависимость интервала периодического контроля от интенсивности поступления заявок при $\beta з = 0.8$, μ = 50, Тпк = 0.003с, Ток = 0.00078с и различных λ: 30 ; 31 ; 32 ; 33 ; 34; 35 ; 36 ; 37 ; 38 ; 39 Соответственно получено: 5,6 ; 7; 9; II ; 14;21 ; 36 ; 110; 2803 (см. рис .3 .2) .

В формуле (3.11) n зависит от λ, при $\lambda \to 0$, значение $n \to 0$. Поэтому

$$Tmin = n/\lambda \qquad (3.16)$$

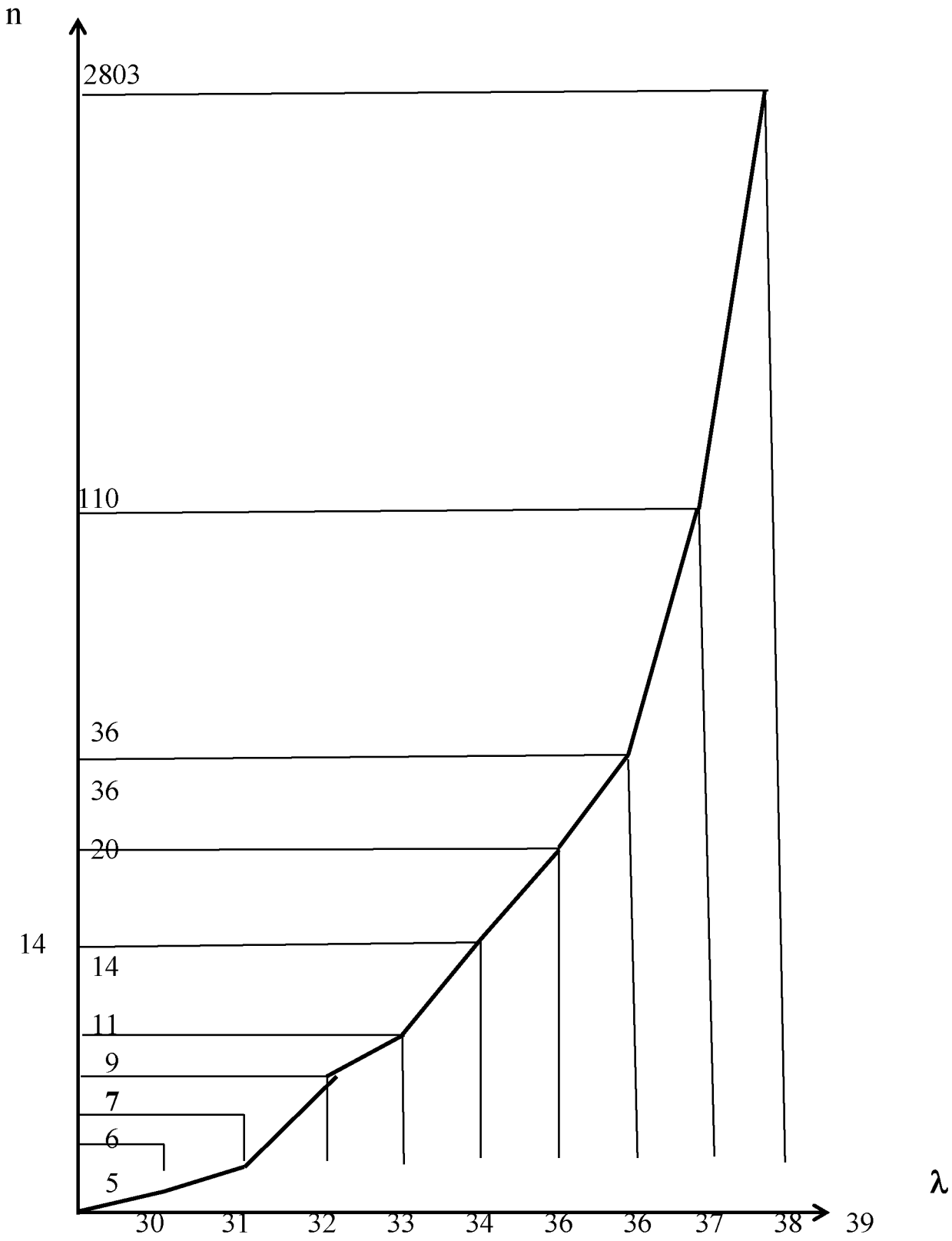


Рис.3.2. Зависимость интервала периодического контроля от интенсивности поступления заявок при заданной загруженности β=0.8 и μ=50

Интервал времени запуска периодического контроля ***Tmin*** равен отношению числа обслуженных заявок (n) к интенсивности

поступления заявок (λ).

Готовность **ПО** узла **АК** с использованием средств контроля и восстановления определяется по формуле:

$$Krq = (1 - P11)(1 - Kпок) \qquad (3.17)$$

где **Кпок** – коэффициент потери эффективности из-за оперативного контроля.

Схему-граф, приведенный на рис.3.1, можно использовать для анализа функционирования как всего ПО утло ЛК, так и отдельно взятого блока.

Для случая функционирования всего ПО узла АК, когда загруженность ПО узла β суммируется из загруженности каждого блока βi, (i=1,k где **k** - число блоков узла)

$$\beta = \beta_1 + \beta 2 + \ldots + \beta k \qquad (3.18)$$

или

$$\frac{\lambda}{\mu} = \frac{\lambda 1}{\mu 2} + \frac{\lambda 2}{\mu 2} + \ldots + \frac{\lambda k}{\mu k}$$

и

$$Tобс = \frac{\sum_{i=1}^{k} \frac{\lambda i}{\mu i}}{\lambda} \qquad (3.20)$$

Время обработки заявок *l* -го блока можно рассчитать по формуле

$$T'обс = 1 / \mu i \qquad (3.21)$$

3.3. Алгоритмизация и программная реализация аналитической модели расчета показателей надежности ПО

Разработанная аналитическая модель процесса функционирования ПО узла АК в ВС с неисправностями при использовании средств контроля и восстановления, а также без них программно реализована (MODEL) на языке Ф0РТРАН-IY для СМ ЭВМ, объемом около 200 команд. Время выполнения измеряется секундами.

Алгоритм разработанной программы MODEL аналитической модели

сводится (рис. 3.3.) к следующему:

1. Выбор режима работы.

2. Функционирование с использованием средств контроля и восстановления.

3. Функционирование без использования средств контроля и восстановления.

II. Ввод исходных данных с внешнего носителя (магнитный диск).

III. Рассчет требуемых коэффициентов.

IY. Запись результатов расчета на внешний носитель.

Y. Выдача результатов на АЦПУ.

YI. Завершение работы.

Функциональное назначение MODEL заключается в следующем:

-вычисление коэффициентов Ak (k=1,m) и A_{11} по формулам (3.2) и (3.5);

-определение вероятности нахождения в состояниях $Sm_{+1,0}$ и вероятностей Pk, k=1,m и P_{11} по формулам (3.б), (3.3), (3.4).

- расчет относительной пропускной способности, коэффициента готовности (Krq) ПО узла АК по формулам 0.8), (3.9), а также (3.17).

Входной и выходной информацией для работы MODEL служат файлы. В файле MOD1.DAT подготавливаются данные для работы без средств контроля и восстановления, а в файле MOD2.DAT для работ со средствами контроля и восстановления. Структура файла MOD2.DAT содержащая данные процесса Б, идентична структуре аайла MOD1.DAT .

Порядок и состав данных для работы MODEL соответствующих одной строке файла MOD1.DAT следующие: задаются значения программных идентификаторов ***LAMBDA , MQ, NQ, GAMMA, M, P***, параметров СМО ***λ, μ, ν, γ, m, p.***

В результате работы MODEL создаются два файла: BER.DAT, XAR.DAT. В файле BER.DAT содержатся

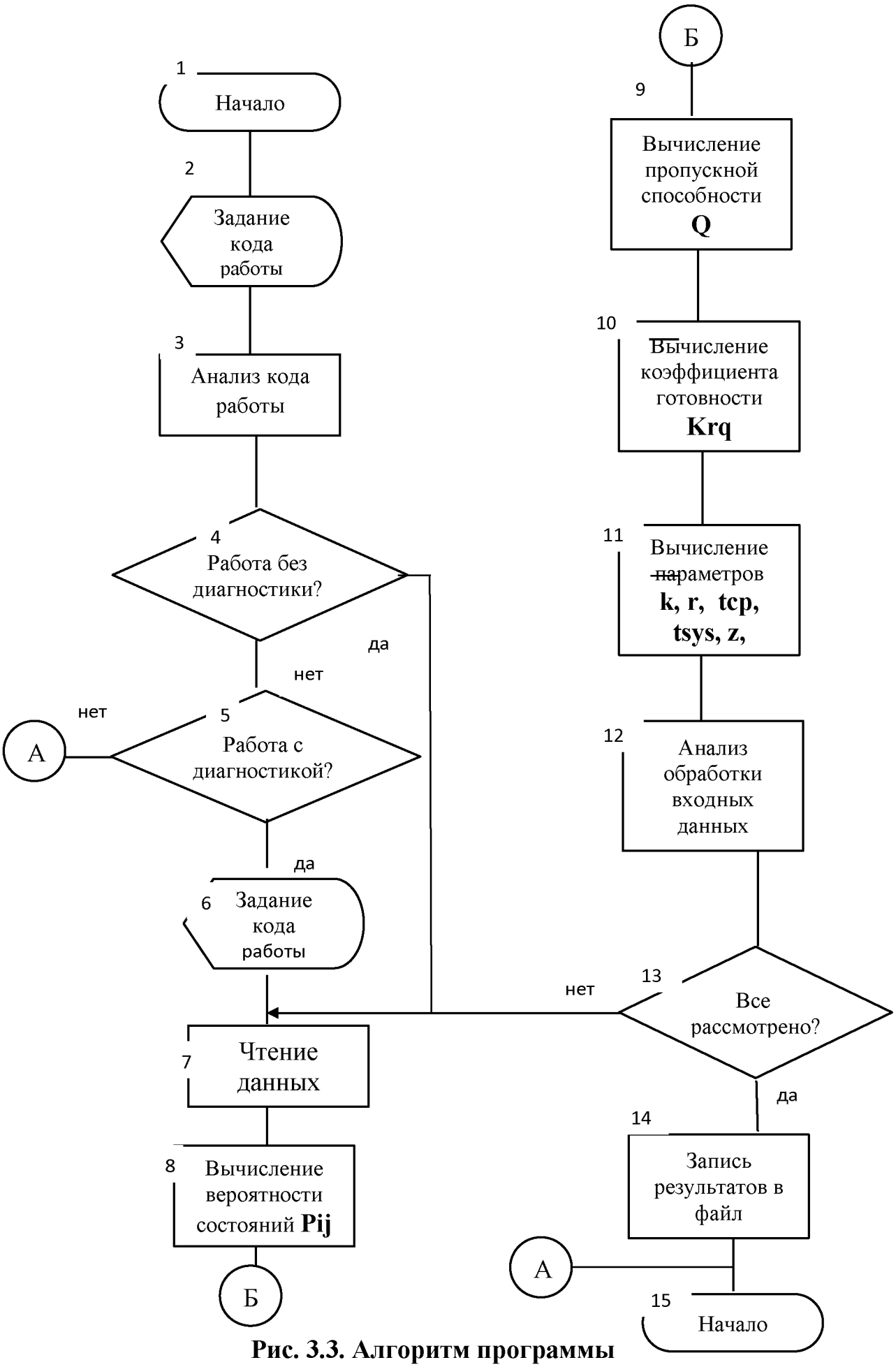


Рис. 3.3. Алгоритм программы

вероятности финальных состояний – Pk, k=1,m. в XAR.DAT файле основные характеристики CMQ, Krq и др.

Кроме перечисленных функций, программу MODEL можно использовать для получения: (žřťǩ)

- среднего числа заявок в очереди ***ř*** (RSH - идентификатор);
- среднего числа заявок в системе ***ž*** (ZSH);
- среднего числа занятых каналов ***ǩ*** (KSH);
- среднего времени пребывания заявок в ПО узла АК ***t'sys*** (Tsvs);
- среднего времени пребывания заявок в очереди ***t'or*** (Toch);
- зависимости пропускной способности Q от количества буферов (***m***) и загруженности системы (β).

Вычисление параметров ***ř***, ***ž***, ***ǩ***, ***t'sys***, ***t'or*** производится согласно формулам, приведенным в [7].

$$\check{k} = 1 - Po\,,$$

$$\check{r} = (\beta^2[1-P^m(m+1-mp)])/((1-\beta^{m+2})(1-\beta),$$

$$\check{z} = \check{k} + \check{r}\,,$$

$$t'sys = \frac{\check{z}}{\lambda}\,,$$

$$t'or = \frac{\check{r}}{\lambda}.$$

Эти формулы использованы при разработке аналитической модели и при ее программной реализации.

3.4. Рассчет эффективности разработанных средств контроля и восстановления

Результаты расчета аналитической модели оценки работы блока БКП узла АК в ВС с неисправностями как без использования контроля и восстановления (случая А), так и с ними (случай Б) были проведены по программе MODEL.

Для случая А были заданы 32 набора данных, согласно структуре файла MOD1.DAT. Первые 19 наборов служили для получения зависимости пропускной способности (Q) от количество (**m**) по аналитической модели, когда **λ** равно 30 и 40, а ***β*** 0,6 и 0.8 при **μ** =50, ν=0.00002 (во всех наборах данных предполагается что за 12.86 ч. происходит отказ).

Анализ результатов показывает, что при задании большого **m**, пропускная способность увеличивается значительно с 0.8163 (***m***=1) до 0.9765) (***m***=9) как при ***β =0.6,*** **так при** ***β =0.8*** **составляют 0.8163 до 0.9765)** (рис.3.4). Однако при ***β*** , близких к единице, увеличение числа, ***m*** будет мало влиять на пропускную способность. Но в любом случае увеличение ***m*** приводит к уменьшению вероятности ожидания заявок **Poo** (рис.3.5).

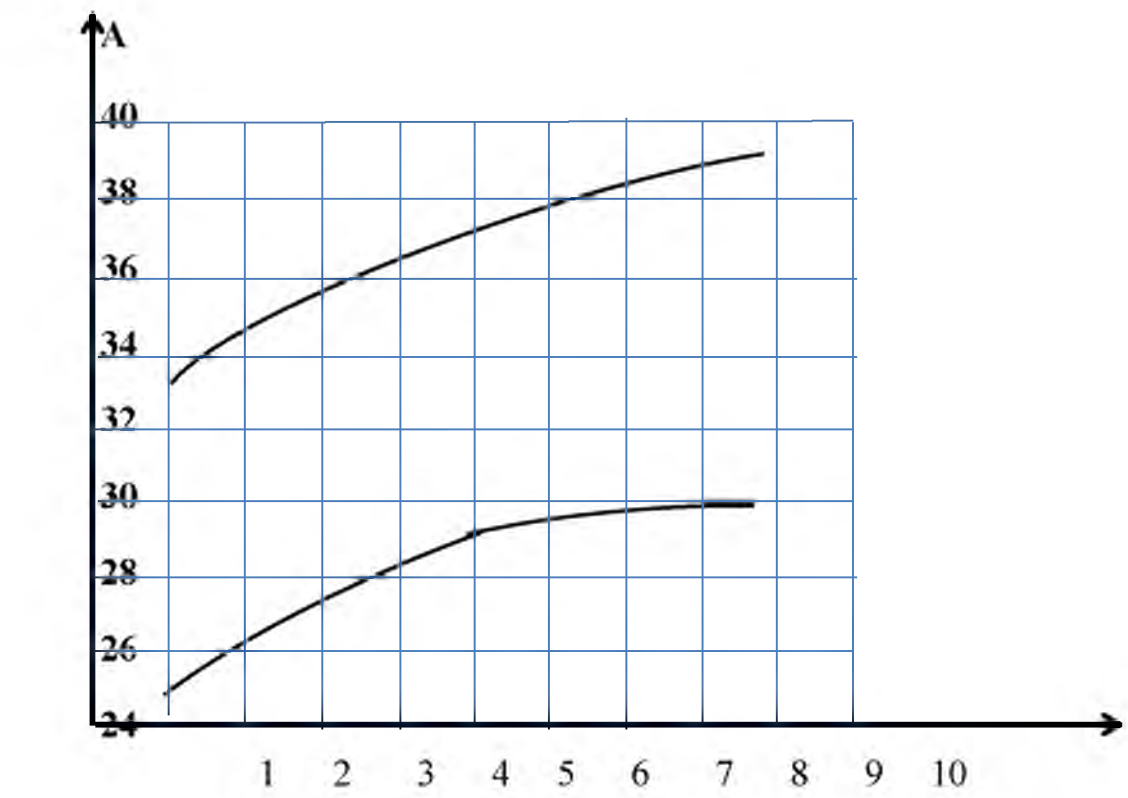


Рис.3.4. Зависимость абсолютной пропускной способности (**A**) от количества мест (**m**) для а). **λ**=30, **β**=0.6; б). **λ**=40, **β**=0.8 при довольно быстром восстановлении (**Твост**=1с)

Наборы 20,21 служили для вычисления основных характеристик работы блока БКП. В результате расчета в MODEL с этими данными получено Krq = 0.9923, Q=0.6327 при P =0.702 и Krq =0.9928 ; Q =0.9737 при P=1.

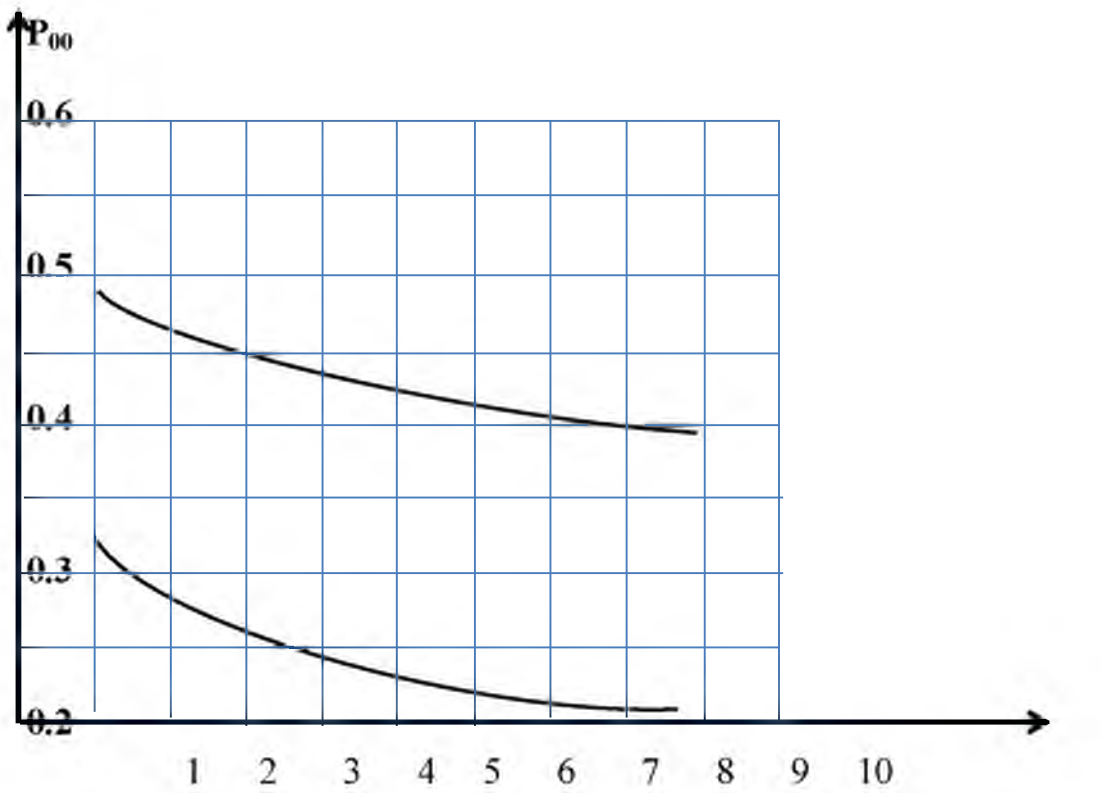


Рис.3.5. Зависимость вероятности ожидания заявок ($\mathbf{P_{00}}$) от количества мест (**m**) для а). λ=30, β=0.6; б). λ=40, β=0.8 при **Твост** =1с.

Как видно из этого расчета недостоверное обслуживание влияет в основном на Q и не изменяет Krq (рис.3.6). Остальные наборы данных использовались для получения характеристик других блоков ПО узла АК. Полученные результаты практически подтверждают возможность оценки по разработанной модели процесса функционирования основных блоков узла АК.

Расчет для случая Б также проводился для 32 наборов и получены результаты. Данные этих наборов в целом адекватны набору данных случая А. Разница заключается в следующем. В первом случае для расчета интенсивность обслуживания заявок без средств контроля (μ) и восстановления, без средств повышения надежности ПО (γ). Во втором случае расчет производился при: интенсивности обслуживания заявок с средствами оперативного контроля (μ') и восстановления с средствами программного восстановления (γ'). В качестве интенсивностей были выбраны μ'=49.65 (из-за оперативного контроля вносится временная избыточность γ' =271).

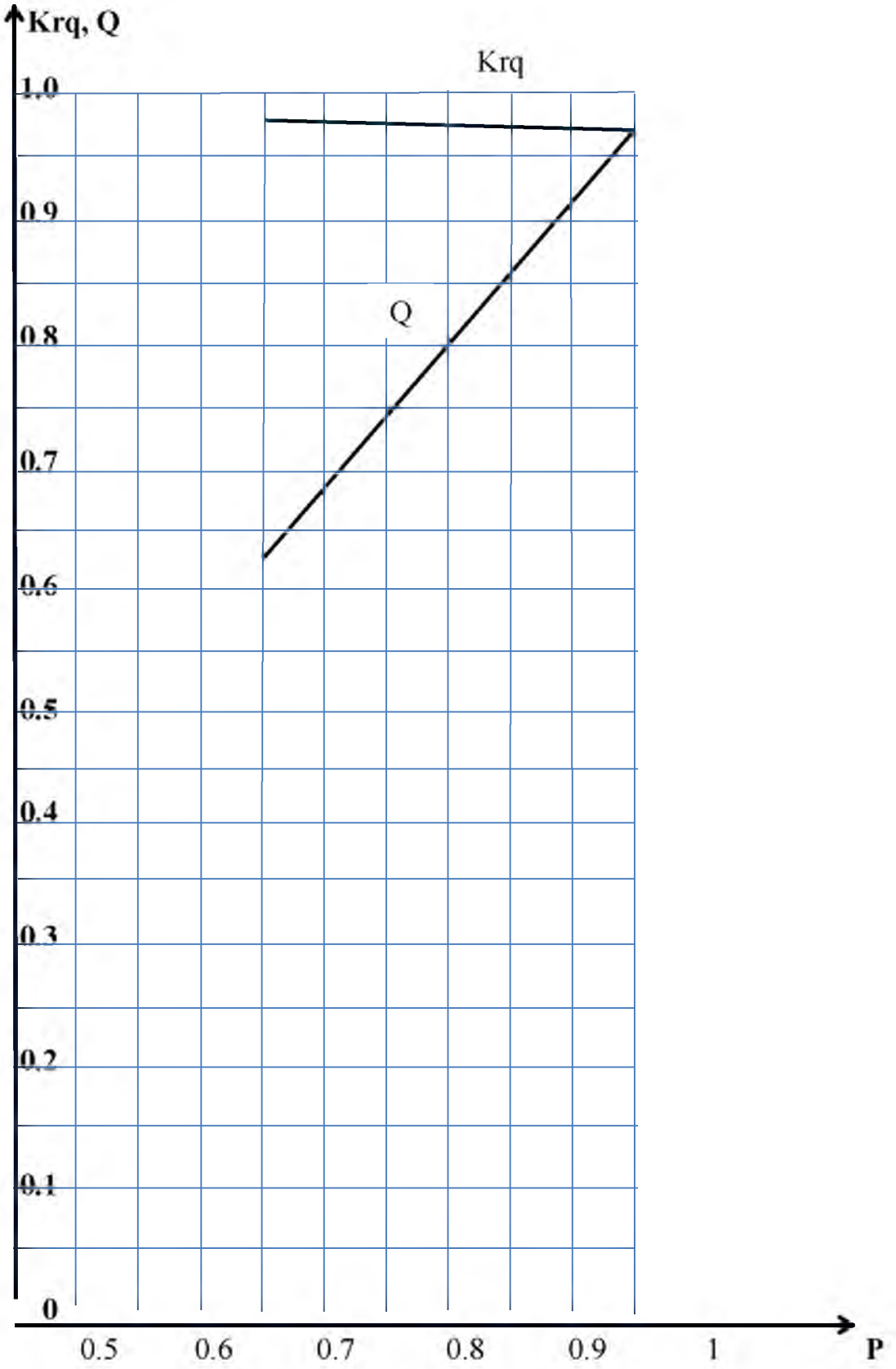


Рис. 3.6. Зависимость пропускной способности (Q) и готовности (Krq) ПО узла адаптивной коммутации от достоверного обслуживания (P)

Анализ результатов случая Б показывает, что за счет разработанных средств контроля и восстановления пропускная способность блока БКП и коэффициент готовности ПО улучшены с Q =0.6827 и Krq = 0.9928 до Q= 0.9779 и Krq = 0.9986. При этом вероятность нахождения в состоянии отказа **P** уменьшена с 0.0075 на 0.0000001.

Кроме того, за счет средств контроля и восстановления улучшены такие характеристики как среднее число занятых каналов ($\check{k}$), уменьшенное с 0.591 до 0.588; среднее время пребывания заявок *в* СМ0 (***t'sys***), сниженная соответственно с 0.4338 до 0.4328с, среднее число заявок в системе ($\check{z}$), ожидающих обслуживания сокращенное с I.30I до 1.298.

Эти цифры более весомы, если рассмотреть обслуживание 10000 заявок, поступающих в течение одного часа. Экономия времени составит около 1с, в течении которой можно обработать на узле коммутации в среднем 40 пакетов.

Другие результаты, полученные по аналитической модели приводятся в разделе 4 для сравнения с результатами имитационной модели.

ВЫВОДЫ

1. Предложена аналитическая модель, позволяющая оценить полезность функционирования как автономно выполняемых блоков узла адаптивной коммутации, так и всего ПО узла при использовании средств контроля и восстановления и без них.

2. Разработанная программа расчета показателей надежности ПО узла АК позволяет получать на ЭВМ различные зависимости пропускной способности, и готовности ПО узла адаптивной коммутации.

3. Полученные результаты показывают, что использование средств контроля и восстановления существенно увеличивает пропускную способность при уменьшении вероятности пребывания в состоянии отказа более чем в десять раз.

4. Выражение для оценки ***n*** числа обслуженных заявок позволяет определить минимальный интервал запуска периодического контроля.

ГЛАВА IV. ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ПРОЦЕССА ФУНКЦИОНИРОВАНИЯ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ В УСЛОВИЯХ ВОЗМУЩАЮЩЕГО ВОЗДЕЙСТВИЯ СРЕДЫ

В данном разделе будет описана экспериментальная работа по определению степени влияния возмущений (неисправностей технических средств вычислительной техники) на эффективность управления потоками данных узла АК. Конкретно будет имитирован процесс функционирования блока БКП с общей областью памяти структур данных узла АК в условиях отказов и сбоев, составляющий пуассоновский поток. Будет проверена целесообразность введения адаптации в структурах данных узла АК, получены оценки эффективности (или не эффективности) применения адаптации в узле АК согласно использованию методов повышения надежности ПО и выработка рекомендаций по их пользованию.

4.1. Формализация модели и имитация параметров работы сложной иерархической структуры ПО систем реального времени

В теории надежности, когда представляет интерес появление некоторого числа одинаковых событий (такими событиями являются отказы), используется пуассоновсое распределение[16,68,69]. Появление каждого числа события соответствует некоторая точка на временной шкале.

Плотность пуассоновского распределения определяется как

$$f(n) = \frac{(\nu t)^n}{n!}\exp(-\nu t),\ n = 0,1,\ldots \qquad (4.1)$$

где: t – время, а ν - постоянная интенсивность отказов или интенсивность входящего потока событий.

Функция распределения имеет вид

$$F = \sum_{i=1}^{n} \frac{(\nu t)^i}{i!}\exp(-\nu t) \qquad (4.2)$$

Согласно исследования структуры простейшего потока событий [6,34], вероятность появления **n** событий в заданный промежуток времени соотвествует пуассоновскому распределению, а вероятность того, что временные интервалы между событиями будут меньше некоторого наперед заданного числа **m**, соотвествует экспоненциональному распределению.

$$P(t) = e^{-\nu t} \qquad (4.3)$$

Функция распределения величины **T** определяется как

$$F(t) = 1 - e^{-\nu t} \qquad (4.4)$$

Дифференцируя выражение (4.4) находим плотность распределения величины

$$f(t) = \lambda e^{-\nu t} \qquad (4.5)$$

Числовые характеристики величины **T** – математическое ожидание (среднее значение) **mt ,** дисперция **δt** и среднее квадратическое отклонение **σt** определяются по формулам

$$T = \int_0^\infty t f(t) dt = \frac{1}{\nu} \qquad (4.6)$$

$$\delta t = \int_0^\infty t^2 f(t) dt = \frac{1}{\nu^2} \qquad (4.7)$$

$$\sigma t = \sqrt{\delta t} = \frac{1}{\nu}$$

Разработанная имитационная модель, основанная на применении (4.1) - (4.9), служит для оценки работоспособности блока узла АК в ВС как при использовании средств контроля и восстановления, так и без них [47, 49,52].

Имитационная модель учитывает:

1. Работу блока узла АК в СМО типа M/M/1 с интенсивностью поступления (λ) и обслуживания (μ) пакетов для коэффициента загрузки $\beta < 1$, $\beta = \lambda/\mu$

2. Отказы, составляющие простой пуассоновский поток, вероятность попадания на заданный отрезок отказов равна

$$Pm = \frac{(\nu t)^m}{m!} e^{-\nu t} \qquad (4.8)$$

где ν - интенсивность отказов.

3. Интервалы времени возникновения неисправностей, распределенные по экспоненциальному закону и определенные по формуле

$$t_i = -\frac{1}{\nu} \ln \xi_i \qquad (4.9)$$

где $i=1,m$; ξ_i - псевдослучайные числа в интервале 0,1.

4. Случайные адреса искаженных ячеек памяти поля данных узла

$$Ppi = (Ppmax - Ppmin)\,\xi_i + Ppmin \qquad (4.10)$$

где **Ppmax, Ppmin** - максимальные и минимальные адреса ячеек памяти.

5. Случайные значения неисправностей в поле данных узла АК

$$ZHi = (ZHmax - ZHmin)\,\xi_i + ZHmin \qquad (4.11)$$

где **ZHmax, ZHmin** - максимально и минимально возможные значение неисправностей, равные соответственно 32767, -32768 при длине ячейки памяти, равной одному машинному слову.

6. Моменты времени возникновения неисправностей, определяемые по формуле:

$$t_k = t_{k-1} + t_k \qquad (4.12)$$

при $k=1$, $t_0 = 0$

7. Интервалы периодов запуска программ периодического контроля, вычисляемые по формуле (3.11)

$$n = \frac{Tnk}{\frac{\beta z - \beta p}{\lambda} - Tok} \qquad (4.13)$$

где, n — минимальное число обработанных пакетов, после которой целесообразен запуск периодического контроля.

8. Критерии:

а) наработка на отказ

$$\mathbf{T} = \frac{\mathbf{1}}{\boldsymbol{\nu}} \qquad (4.14)$$

б) время восстановления $\mathbf{t_z}$ (4.15)

в) вероятность безотказной работы

$$\mathbf{R(t) = exp(-\nu t)} \qquad (4.16)$$

г) готовность блока узла АК

$$\mathbf{Krq} = \frac{\boldsymbol{T}}{\boldsymbol{T+\tau}} \qquad (4.17)$$

д) коэффициент эффективности

$$\mathbf{Kef} = \frac{\boldsymbol{Co}}{\boldsymbol{Cn}} \qquad (4.18)$$

е) вероятность обнаружения и устранения ошибок **Po, Pb**

где, **Co, Cn** – число обработанных и принятых пакетов.

На основе вышеизложенного материала была разработана имитационная модель процесса функционирования блока БКП узла адаптивной коммутации в ВС с неисправностями. Основное назначение этой модели сводится к следующему:

- исследование работы блока БКП, к входным очередям которого подаются извне (поочередно) пакеты как длинные, так и короткие;
- анализу работы блока после обработки каждого пакета;
- ведению статистики поступивших и обработанных пакетов;
- генерации неисправностей в области данных ООП, с которой функционирует блок БКП;
- оценки эффективности использованных средств контроля и восстановления.

Разработанная имитационная модель состоит из следующих компонент:

- имитатора блока БКП для совместной работы этого блока с процедурами оперативного контроля и без них;
- имитататора работы блока БСР;
- модели ошибок (генератора неисправностей);
- монитора для управления всеми процессами имитационной модели.

Перейдем к описанию функций составных частей.

Имитатор блока БКП выполняет следующие действия:

- извлекает пакет из очереди входных трактов ***PNPQE***, или очереди входной линии к блоку БКП ***PLPQE***;

- производит маршрутизацию пакета;

- вставляет поочередно пакеты во входные очереди трактов ***TPQE*** или линий ***APQE***;

- подключает к работе блока БКП процедуры оперативного контроля.

Функции имитатора блока БСР включает:

- вставку пакетов поочередно во входную очередь трактов или линий пакетов; при этом вставляются как длинные, так и короткие пакеты;

- ведение счета пакетов поданных к входным очередям блока БКП;

- анализ выходных очередей трактов и линий с целью установления факта обработки или наработки блока БКП поданных пакетов;

- ведение счета обработанных пакетов с выходных очередей тракта и линии.

Модель ошибок предназначена для генерации:

- времени возникновения неисправностей t_k;

- адреса ячейки памяти поля данных узла АК PPk;

- значения неисправности ZHk .

Выбор экспоненциальности закона распределения появления неисправностей объясняется отсутствием экспериментальных данных, которые позволяли бы сделать более сложные предположения о потоке отказовых ситуаций в ВС при исполнении программ в режиме реального масштаба времени [2,30,31]. При этом для выбора использовано допущение о возникновении сбоя в среднем одного сбоя в 1 час.

Разработанная головная программа (монитор) управляет очередностью выполнения всеми составными частями имитационной модели.

4.2. Особенности алгоритмизации и программной реализации имитационной модели

Разработанная имитационная модель процесса функционирования блока БКП узла АК в ВС с неисправностями при использовании средств контроля и восстановления, а также без них реализована в виде программных модулей [49]. Алгоритмы имитационной модели приводятся в приложении 4. Общее число программных модулей имитационной модели, управляемой головной программой MDGMAK, составляет более 20 процедур.

Все программы имитационной модели разработаны на языке СИ [22] для ОС РВ СМ ЭВМ. Исключение составляет программа GENFOR, реализованная на языке ФОРТРАН – IY и предназначенная для автономной работы.

Функциональное назначение программ имитационного моделирования следующее:

1. MDGMAK – управляет всеми программами, осуществляет ввод исходных данных, завершает эксперимент;

2. LENBUF – имитирует работу блока БСР поочередно вставляя длинные и короткие пакеты на входные очереди от трактов и линий;

3. BCP - имитирует работу блока БКП как с процедурами оперативного контроля, так и без них;

4. WRCTAT – осуществляет сбор статистики работы блока БКП, ведет счет поданных и успешно обработанных пакетов;

5. SERWNAT, SERWRT – производят вспомогательные действия при вставки и изъятии пакетов из очередей и пулов;

6. ICTAC – осуществляет обработку собранной статистики;

7. PROWDAT – служит для хранения копий контролируемых данных постоянного и квазипеременного характера общей области памяти узла адаптивной коммутации;

8. MKDOO – осуществляет запуск процедур периодического контроля (назначение процедур периодического контроля PK1, PROW3, ANALOS было проведено выше в п.2.3).

9. PIKONT – выдает результаты эксперимента в требуемой форме;

10. MONDIAG – производит наблюдение за последовательностью процедур LENBUF, BCP, WRCTAT, GENFAULT ; осуществляет прогон БКП в соответствии с заданным значением;

11. GENFOR (запускается автономно) – выполняет часть функций модели ошибок, определяет внесения неисправностей, значения неисправностей, код идентифицирующий, куда должна вносится неисправность.

В программе GENFOR использована версия ДСЧ (датчика случайных чисел) - RAN, реализованная на СМ ЭВМ и входящая в библиотеку стандартных функций ФОРТРАН – IY.

12. GENFAULT – определяет физический адрес ячейки памяти; заносит в определенную ячейку памяти ошибку.

Рассмотрим укрипненный алгоритм работы программ (рис.4.1.).

Началом цикла работ служит запуск MONDIAG , который ведет отчет времени выполнения блока БКП (программы BCP) (в данном эксперименте время заменено числом прогона). После этого работает модуль LENBUF, который вставляет пакет (буфер) во входную очередь тракта или линии. Затем производится проверка времени запуска GENFAULT. Если время запуска истекло, GENFAULT вводит ошибку в поле внутренних данных ООП узла адаптивной коммутации, с которыми взаимодействует блок БКП.

В случае истечении очередного интервала времени запускается периодический контроль MKDOO.

Программа BCP (в случае работы со средствами контроля и восстановления, запускается вариант блока БКП с оперативным контролем) выбирает из входной очереди пакет, определяет маршрут и вставляет обработанный пакет в выходные очереди тракта TPQE или линии APQE. Для одного выходного тракта и двух линий количество этих очередей было 3.

После каждого выполнения программы BCP запускается процедура WRCTAT, предназначенная для сбора статистики обработанных и поданных

пакетов.

В MDGMAK производится проверка: “ Конец эксперимента”.

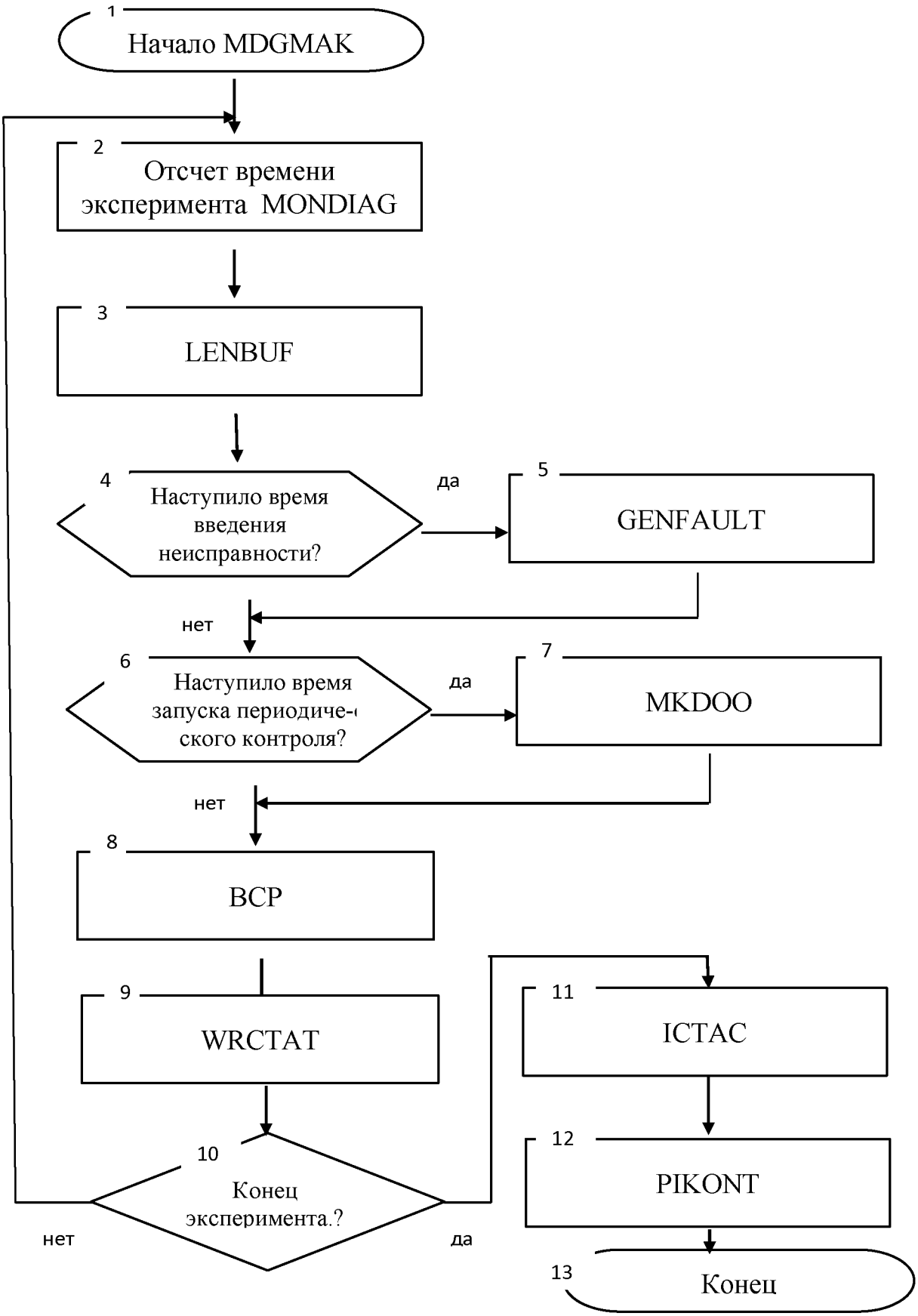


Рис.4.1. Блок-схема функционирования имитационной модели

Текущее время эксперимента подлежит проверке и сравнивается с заданным временем. Если время эксперимента не истекло, то начинается новый цикл.

В PIKONT выдаются результаты. Результатом эксперимента являются число поданных и обработанных пакетов, виды введенных ошибок, адреса искаженных полей памяти и соответствующие им идентификаторы программ, наработка на отказ, устанавливаемые экспериментом.

Приведенная на рис.4.1. схема работы программ имитационного моделирования соответствует случаю функционирования процесса БКП в вычислительной системе с неисправностями со средствами контроля и восстановления. Для случая без контроля и восстановления этапы 6,7 выполняться не будут.

Входной информацией для MDGMAK (имитационного моделирования) служат:

— номер компоненты (очереди, пула, таблицы, контекста) в диапазоне (1,13) ;

— номер байта в диапазоне (0,37) для случая попадания неисправности в таблицу или контекст;

— значение неисправности элемента очереди длиной одномашинное слово в диапазоне (-32768, 32767);

— номер элемента в диапазоне (1,7), если неисправность попадает в очередь или пул;

— значение неисправности в ячейке памяти размером в байт в диапазоне (-128, 127);

— число прогонов с одной неисправностью в диапазоне (1, 9999) .

Эти данные порождаются генератором неисправностей (модель ошибок) GENFOR и выдаются на АЦПУ.

В таблицах содержатся моменты введения ошибок, номер компоненты элемента и значение ошибок.

Ввод этих данных в MDGMAK, производится с клавиатуры дисплея в

диалоговом режиме, и в дальнейшем процедура заносит их в ячейку памяти области ООП узла адаптивной коммутации. Таким образом, происходит генерация ошибки в области данных ООП узла АК.

Описание структуры пакета программ имитационной модели приведено в [52].

4.3. Определение статистической устойчивости оценки моделирования и длительности эксперимента

Для определения статистической устойчивости оценки моделирования вычисляется значение по большому количеству реализаций.

В работе [5] приводится формула определения количества реализаций, при которых в силу центральной предельной теоремы теории вероятностей (теорема Хинчина), отношения числа событий m к объему реализаций (N) m/N при достаточно больших N имеет распределение, близкое к нормальному [66]. Поэтому выбирая для заданной доверительной оценки (λ) из таблицы нормального распределения величину подсчитывают погрешность (ε) между теоретическим и экспериментальным параметром распределения:

$$\varepsilon = T_\lambda \sqrt{D\left(\frac{m}{N}\right)} \qquad (4.19)$$

где m – число событий в N реализациях.

Теоретическим и экспериментальным параметром распределения служат интенсивности ν и ν_i . Выполнение условия

$$| \nu - \nu_i | < \varepsilon$$

служит оценкой их отклонения [14].

Выражая дисперсию D(m/N) через P(1-p)/N получаем формулу для расчета N

$$N = \frac{P(1-p)}{\varepsilon^2} T\lambda^2 \qquad (4.20)$$

В практике моделирования P обычно не известно. Она определяется следующим образом. Выбирают $\mathbf{N_0}$ =50 ÷ 100 и фиксируют число ожидаемых событий (m) в $\mathbf{N_0}$ реализациях, в качестве **P** принимают [5]: и $P = m / N_0$.

Далее рассмотрим вопросы связанные с возможностью сокращения длительности эксперимента.

Пуассоновский поток неисправности, согласно данным статистики сбоев и отказов в реальных ВС [31], можно ожидать с интенсивностью сбоев ν=0,00027 при наработке T = 3600с.

Моделирование такого потока с экспоненцианальным законом распределения времени прохождения событий, как известно [1], требует большого объема испытаний и представляется сложным из-за больших материальных ресурсов и временных затрат ЭВМ.

В данном параграфе показывается возможность решения поставленной задачи.

Существующая формула нахождения следующего момента времени экспоненцианального распределения при моделировании имеет вид:

$$t = - 1/\nu \ln \xi_i \qquad (4.21)$$

где ξ_i – псевдослучайные числа.

При сжатии периода прохождения сбоя (T) в $\mathbf{К_ж}$ раз можно получить

$$\mathbf{T_ж = T / К_ж} \qquad (4.22)$$

для интенсивности прохождения сбоя в периоде

$$\boldsymbol{\nu_ж = 1/T_ж} \text{ или } \mathbf{\nu_ж = К_ж/T} \text{ или } \mathbf{\nu_ж = К_ж \nu} \qquad (4.23)$$

Тогда периоды прохождения сбоя для наработки на сбой определяются

$$\mathbf{Тж = -1 /\nu_ж \ln \xi_i = -1/(К_ж \nu) \ln \xi_I = t' / к_ж} \qquad (4.24)$$

Общее время эксперимента (Тэкс) составит сумма

$$\mathbf{Тэкс} = \sum_{k=1}^{N} t(k)$$

и после сжатия ее можно определить как

$$\mathbf{T'экс} = \frac{\text{Тэкс}}{\text{Кж}} = \frac{\sum_{k=1}^{N} t(k)}{\text{Кж}} = \sum_{k=1}^{N} t_{ж(k)} \qquad (4.25)$$

Экспериментальное подтверждение достоверности использование коэффициента сжатия для сокращения времени эксперимента приводится ниже.

4.4. Анализ экспериментальных результатов

Работа по имитационному моделированию проводилась в 2 этапа.

На первом этапе были проведены следующие работы:

а) сокращение длительности эксперимента. Для этого при $\mathbf{K_ж = 120}$ по формулам (4.23), (4.25) получены интенсивности прохождения сбоя (ошибки) $\boldsymbol{\lambda ж}$ = 0.0333 и наработка на сбой $\mathbf{T_ж}$ =30 с. Затем выбирая $\mathbf{N_0}$ = 100 [5] определена точность моделирования $\boldsymbol{\varepsilon}$ (формула 4.19). Вставляя в 4.20 для доверительной оценки $\boldsymbol{\lambda}$ = 0.95 из таблицы нормального распределения значение T_λ, выбрав p=m /N_0 , получено количество реализаций N. $\boldsymbol{\varepsilon}$ и N равны, соответственно 0,005599 и 3931, а **m** = 116.

б) анализ отклонения экспериментальной интенсивности появления ошибок ($\boldsymbol{\nu i}$) от теоретической ($\boldsymbol{\nu}$) в зависимости от числа сгенерированных ошибок (**m**) при выбранной погрешности ($\boldsymbol{\varepsilon}$) моделирования. Результаты анализа показывают (рис.4.2.) для заданной доверительной оценки ($\boldsymbol{\lambda}$=0.95) из 100 случаев в 95 разность $|\boldsymbol{\nu} - \boldsymbol{\nu}_i|$ не превышает заданную погрешность;

в) сгенерированы с помощью программы GENFOR интервалы происхождения сбоев ($\mathbf{t_k}$), адреса ячеек памяти поля данных узла адаптивной коммутации (**PPk)**, значение ошибок длиной машинного слова **ZHk**.

На втором этапе производились работы, связанные с имитацией процесса функционирования блока БКП в вычислительной системе с неисправностями.

Для этого были выполнены:

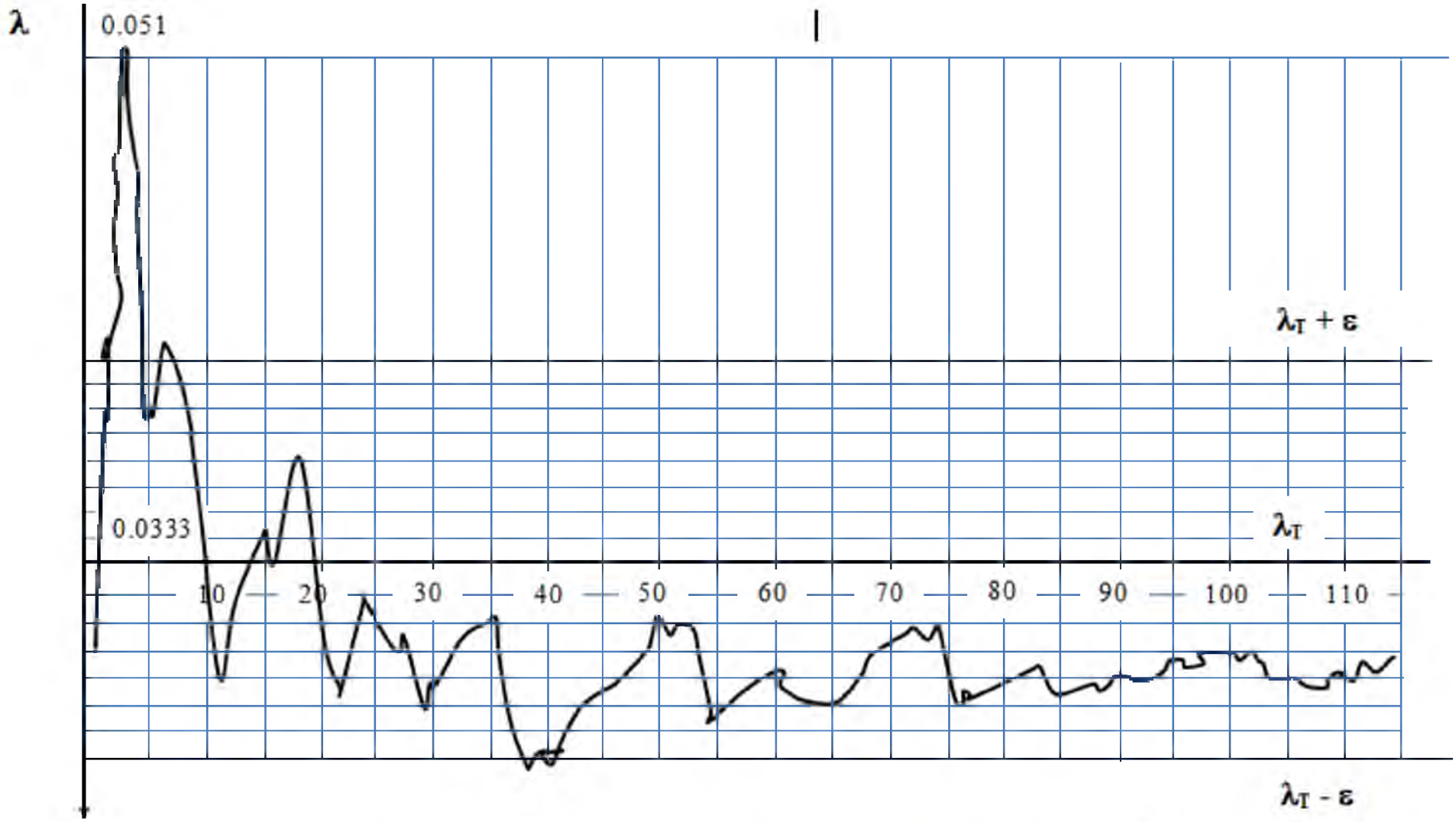


Рис. 4.2. График отклонения экспериментальной интенсивности появления ошибки (λ_T) от теоритической (λ_i) в зависимости от числа сгенирорированных ошибок (**m**), где **ε** - погрешность моделирования.

а) включение программ, входящих в состав MDGMAK согласно алгоритму описанному в 4.2 и схемы, приведенной на рис.4.1;

б) прогон блока БКП в течение времени **Т'экс**, определенный в предположении, что в 1с поступает I пакет;

в) ввод в диалоговом режиме ошибок в относительные адреса компонент узла адаптивной коммутации находящихся в общей области памяти. Этими компонентами являются MCT, PNPQE, TRUT и др.

г) периодический запуск средств контроля и восстановления. Период запуска, определенный по (4.23) для **n** при **βz**=0.8, **μ** = 50, $\mathbf{T_{пк}}$ =0.003с, **T**ok=0.0078с для **λ** =30 и **βp**=0.6 составил число 5 ;

д) ведение статистики поданных (**Сп**) и обработанных (**Со**) пакетов;

е) фиксация числа отказов ($\mathbf{N_{otk}}$).

В таблице 4.1 приводятся результаты, полученные по имитационной модели. Просмотр этой таблицы даст возможность сравнения основных показателей надежности работы блока БКП с неисправностями для случая использования средств контроля и восстановления и без них. Коэффициент эффективности $\mathbf{К_{эф}}$ улучшен на счет средств контроля и восстановления с 0.679 на 0.962. **Krq** с 0.991 на 0.993, а число отказов сокращено на порядок. График изменения коэффициент $\mathbf{К_{эф}}$ в зависимости от времени работы ПО при заданной наработке на ошибку **T**=3600с приведен на рисунке 4.3.

График показывает влияние ошибок на эффективность обработки пакетов для блока БКП при использовании средств контроля и восстановления (линия **б**), и без них (линия **а**). Существенное уменьшение коэффициента $\mathbf{К_{эф}}$ происходит для случая работы блока БКП, когда средства контроля и восстановления не использовались. Например, для происхождения 4-го отказа (линия a_4), когда ошибки не сразу приводили к отказу снижена $\mathbf{К_{эф}}$ до 0.14 и потеря достигла более 80 %, к исходу 10 часового эксперимента с десятью ошибками. В другом эксперименте безотказная работа блока БКП длилась около 67 часов.

Табл.4.1.

Результаты имитационной модели

пп	ПОЛУЧЕННЫЕ ХАРАКТЕРИСТИКИ											Время эксперимента Тэкс в ч.
	Число поданных пакетов, Сп	Число обработанных пакетов, Со	Исходная наработка на ошибку в области данных узла АК, Т		Число отказов	Наработка на отказ блока БКП		Время восстановления при наличии		Коэффициент эффективности	Коэффициент готовности	
			в с.	в ч.		в с.	и.ч.	ошибки	отказа			
1	2	3	4	5	6	7	8	9	10	11	12	13
1.Без средств контроля и восстановления												
1)	4919	2833	3600	1	9	65586	18.2	-	360	0.575	0.994	162
2)	1944	1829	3600	1	11	21207	5.89	-	360	0.94	0.983	65
Общее												
	6863	4662	3600	1	20	41178	11.43	-	360	0.679	0.9913	227
2. Со средствами контроля и восстановления												
1)	3487	3372	3600	1	1	418428	116.23	0.03	-	0.957	0.993	120
2)	2824	2704	3600	1	1	338879	94.13	0.03	-	0.967	0.993	100
Общее												
	6311	6076	3600	1	2	378648	105.18	0.03	-	0.962	0.993	220

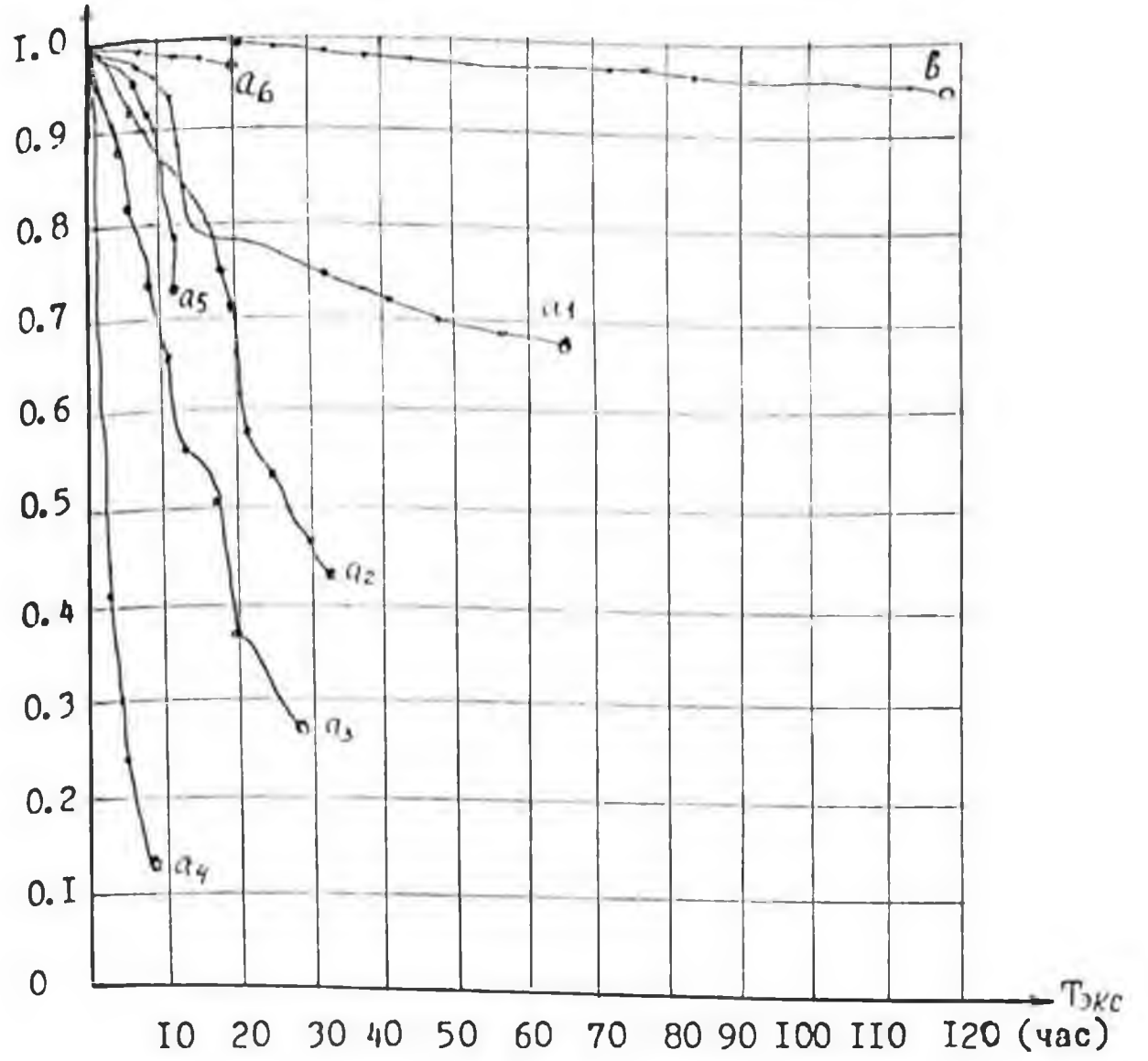


Рис. 4.3. График изменения коэффициента эффективности в зависимости от наработки на ошибку $T_{ош}$ =1(до происхождения отказа)

Пояснение к рисунку: • -моменты появления ошибок в ПО области данных ПО узла адаптивной коммутации; ° - моменты появления отказа в ПО узла АК (блока БКП);

$\mathbf{a_i}$ - график изменения коэффициента эффективности ($\mathbf{K_{эф}}$) во времени до **i** -го отказа программного блока БКП ПО узла АК без средств контроля и восстановления;

b -график изменения коэффициента $\mathbf{K_{эф}}$ во времени до 1-го отказа ПО узла АК со средствами контроля и восстановления.

Накопление около 67 ошибок в области данных ПО узла адаптивной коммутации привели к уменьшению $\mathbf{K_{эф}}$ на 0.32.

График зависимости $\mathbf{K_{эф}}$ от времени эксперимента для работы блока БКП со средствами контроля и восстановления показывает, что предложенные средства контроля и восстановления своевременными локализациями и устранениями ошибок позволяют поддерживать коэффициент $\mathbf{K_{эф}}$ в пределах 0.95-0.96.

В целом из полученных результатов следует: накопление критических ошибок приводит к резкому уменьшению пропускной способности. В одних случаях накопление критических ошибок приводит к отказу (в течении 1-4 часа) ; в некоторых случаях накопление критических ошибок не существенно повлияет на время безотказной работы (67 часов). Однако, среднее время до отказа составило 11,43 ч. для случая работы блока БКП без средств контроля и восстановления. Тогда как со средствами контроля и восстановления это число равно 105 ч.

Установленные экспериментом наработки на отказ позволили построить функцию надежности **R(t)** работы блока БКП ПО узла адаптивной коммутации (рис. 4.4).

В предложенной наработке на сбои ($\mathbf{T_н}$ =3600с) надежность ПО при длительности работы без средств контроля и восстановления более 10 ч существенно уменьшается, а при длительности работы($\mathbf{T_д}$) более 50 ч вероятность безотказной работы почти равно нулю. Когда как использование средств контроля и восстановления позволяет при $\mathbf{T_д}$ =10 часов сохранить надежность работы ПО узла адаптивной коммутации, близкой 0.95-0.97, а при $\mathbf{T_д}$=**50** ч **R(t)** более 0.6.

В случае предположенное ($\mathbf{T_н}$ =50ч) надежность уменьшается ненамного, и для очень большой длительности $\mathbf{T_д}$, больше 230 ч составляет 0.62. Для случая с контролем и восстановлением это число близко к единице.

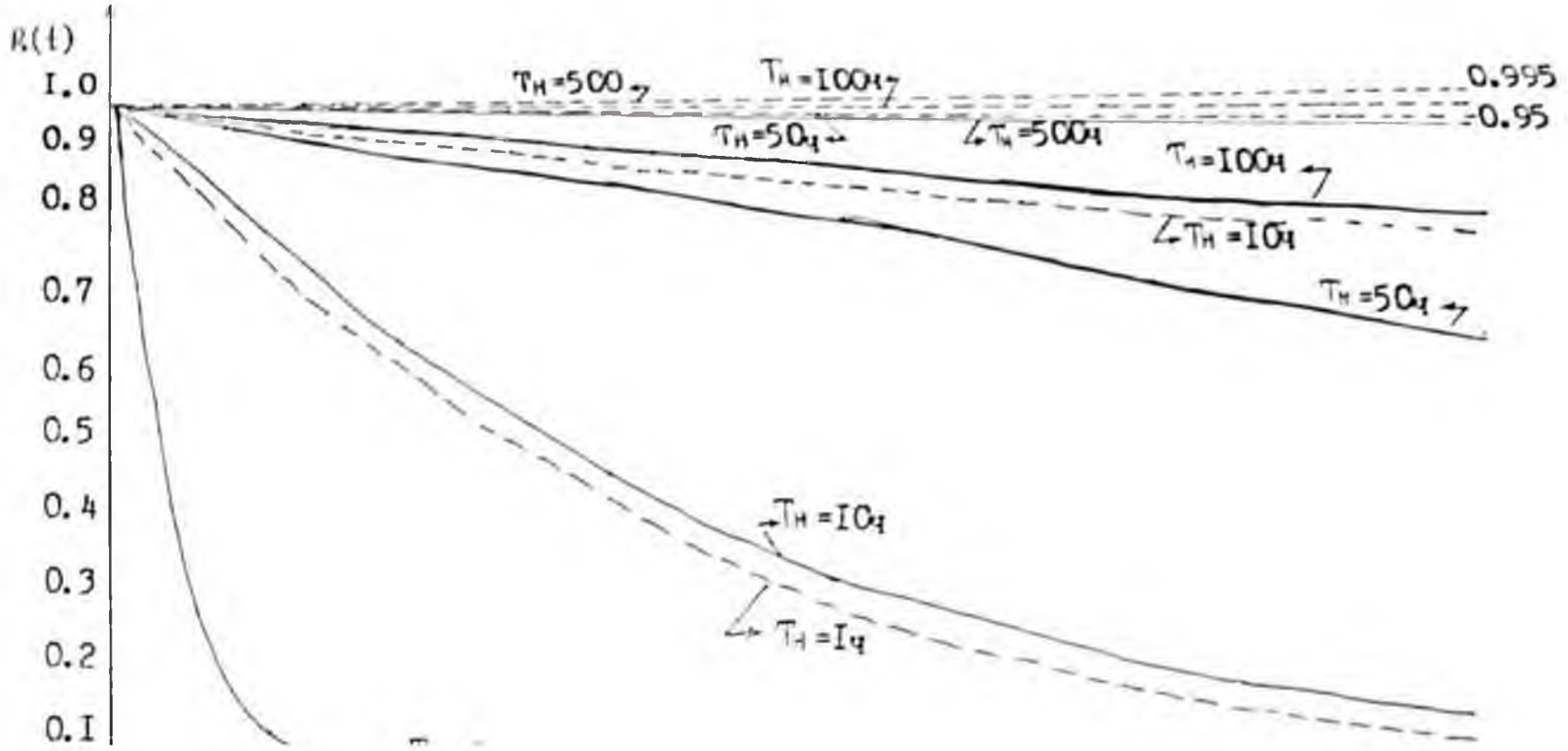


Рис. 4.4. Вероятность безотказной работы блока БКП узла адаптивной коммутации при различных наработках на отказ (Тн). Результаты имитационной модели: а) без средств контроля и восстановления (КВ) для Т =12.8 ч (обычные линии); б) со средствами КВ для Т =116.6 ч (линии со штрихами)

В целом, как показывает рис.4.4 при больших наработках на сбой $\mathbf{T_н} >> 50$ ч существенное уменьшение надежности не происходит и в этом случае, средства контроля и восстановления не используются. Однако, когда $\mathbf{T_н} < 50$ ч необходимость использования средств контроля и восстановления возрастает.

Далее, экспериментом была подтверждена достоверность использования коэффициента $\mathbf{K_ж}$ для сокращения времени эксперимента. Для участка моделирования между 4-ым и 5-ым отказами при 20 ошибках были проведены расчеты для различных $\mathbf{K_{эф}}$.

Результаты приводятся в таблице 4.2.

Полученные результаты показывают, что при наработках на отказ $\mathbf{T_н}$ при сжатии времени, периода эксперимента $\mathbf{T_{экс}}$ коэффициент эффективности меняется незначительно. Погрешность (погрешность из-за сжатия) составляет в среднем 9-10 пакетов для 1000 переданных пакетов, что равно около 1%.

Как видно из рисунка 4.5 погрешность моделирования из-за сжатия наработка на ошибку ($\mathbf{T_н}$), при достаточно большом сжатии $\boldsymbol{K_ж} > 120$ резко увеличивается и может составить более 1%. Поэтому эксперимент проводился для $\mathbf{K_ж} < 120$. При $\mathbf{K_ж} = 120$ погрешность составила не более 0,7 %.

Таблица 4.2.

№ п/ п		Данные				Результаты
	$\boldsymbol{\nu}$	$\mathbf{T_н}$ в ч:	$\mathbf{K_ж}$	**Сп**	**Со**	$\mathbf{K_{эф}}$
1	2	3	4	5	6	7
1.	0.098	0.0027	360	290	100	0.3448
2.	0.054	0.005	200	523	149	0 .2848
3.	0.033	0.0083	120	861	245	0.2845
4	0.027	0.01	100	1052	300	0.2843
5	0.0054	0.05	20	5299	1506	0.2843
6	0.0027	00.1	10	10608	3016	0.2843

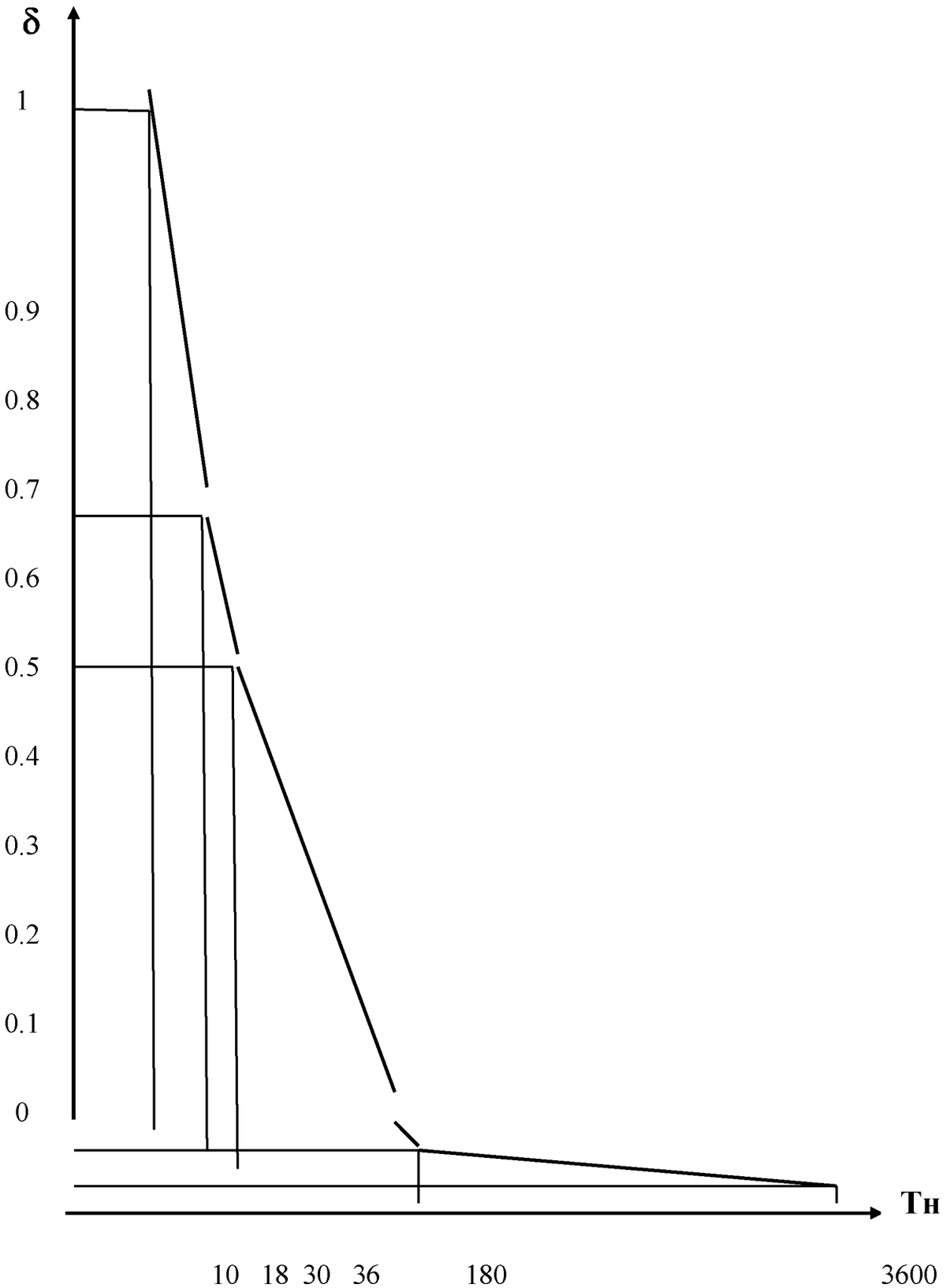


Рис. 4.5. Погрешность моделирования (δ) из-за сжатия

На основе полученных результатов имитационной модели процесса работы блока БКП (рис.4.6) ПО узла адаптивной коммутации функционирующей в вычислительной системе с неисправностями выявлен диапазон изменения

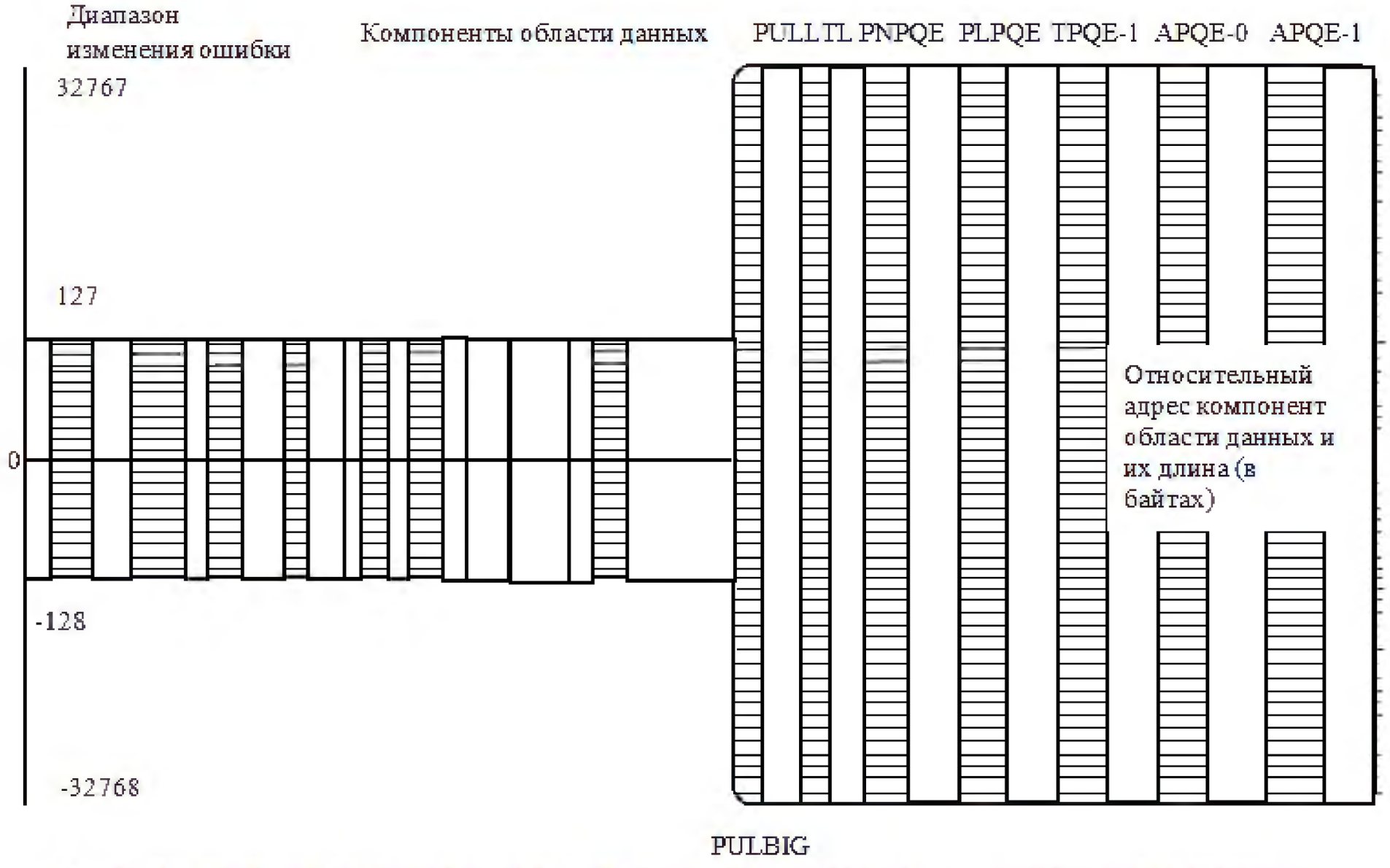


Рис. 4.6. Диапазон (заштрихованная область) изменения ошибки, при наложении которой на некоторые области структуры данных ПО узла адаптивной коммутации появляется отказ в блоке БКП

значений ошибки, при наложении которой на область данных появляется отказ в блоке БКП. На этом рисунке заштрихованные области являются источниками отказов ПО узла адаптивной коммутации. Источники отказов ПО узла АК составили: указатели в таблицах MCT, TRUT,LINT в контекстах тракта и линий; ссылки на буферы в очередях тракта и линий, выходных трактов (TPQE) и линий (APQE).

В ходе эксперимента подтверждено, что искажение в указателях приводит к отказу: из 20-ти отказов 14-составили ошибки в таблицах на MCT, TRUT,LINT; по одному отказу приходится на TABPARAM, PULBIG, APQE-1, TPQE-0, PLPQE и PNPQE.

Для эксперимента были выбраны 1 тракт с очередью и 2 линии с очередями APQE-0, APQE-1.

В целом из совокупности 223 критических ошибок в 208 случаях критические ошибки не порождали отказ, однако при этом $\mathbf{K_{эф}}$ существенно уменьшен и составил 0.69. А использование средств контроля и восстановления позволили улучшить $\mathbf{K_{эф}}$ до 0.962 при этом числа отказов уменьшено на порядок.

Анализ готовности для ПО узла адаптивной коммутации со средствами контроля и восстановления и без них показывает (табл.4.3), что разница составляет в тысячных долях 0.993 и 0.991. Из этого факта можно было бы утверждать, что использования средств контроля и восстановления не рационально. Однако, улучшение $\mathbf{K_{эф}}$ надежности R(t) ПО показывает эффективность и целесообразность использования последних для области ПО узла адаптивной коммутации.

4.5. Полученные показатели эффективности разработанных средств

Имитационное моделирование процесса функционирования блока БКП ПО узла адаптивной коммутации в ВС с неисправностями позволило экспериментально оценить эффективность разработанных средств контроля и восстановления (раздел 4.5), основанных на выбранных в разделе 2.1 методах обнаружения и устранения ошибок.

Таблица 4.3

Характеристики готовности и достоверного обслуживания ПО узла коммутации

n n	ВХОДНЫЕ ДАННЫЕ								Полученные результаты			
	Интенсивности			Наработка на отказ	Время восстановления		Коэффициент достоверного обслуживания	Количество мест, m	по имитационной модели		по аналитической модели	
	поступления, λ	обслуживания, μ	обслуживания, μ'		оператором, в с.	программно, в с.			Krq	Q	Krq	Q
1	40	50	-	50000	360	-	0.9142	1	0.993	0.907	0.993	0.671
2	40	50	-	50000	360	-	0.9142	5	0.993	0.907	0.993	0.848
3	40	50	-	50000	360	-	0.9142	9	0.993	0.907	0.993	0.890
4	40	50	-	50000	360	-	0.7002	1	0.993	0.695	0.993	0.514
5	40	50	-	50000	360	-	0.7002	5	0.993	0.695	0.993	0.649
6	40	50	-	50000	360	-	0.7002	9	0.993	0.695	0.993	0.679
7	40	50	-	5000	360	-	0.9142	1	0.993	0.853	0.993	0.644
8	40	50	-	5000	360	-	0.9142	5	0.993	0.853	0.993	0.800
9	40	50	-	5000	360	-	0.9142	9	0.993	0.853	0.993	0.834
10	40	-	49,56	50000	360	0.003	0.9142	1	0.993	0.907	0.993	0.672
11	40	-	49,56	50000	360	0.003	0.9142	5	0.993	0.907	0.993	0.852
12	40	-	49,56	50000	360	0.003	0.9142	9	0.993	0.907	0.993	0.891
13	40	-	49,56	5000	360	0.003	0.9142	1	0.993	0.907	0.993	0.672
14	40	-	49,56	5000	360	0.003	0.9142	5	0.993	0.907	0.993	0.852
15	40	-	49,56	5000	360	0.003	0.9142	9	0.993	0.907	0.993	0.891

Достаточно сказать, что использования процедур оперативного контроля ОК1, 0К2, 0К3, 0К4, ОК5, 0К6 позволили защититься от 90 % отказов, при этом контролировалась сохранность значения указателей в управляющей таблице данных ООП узла АК, ссылки к буферам как от пула, так и от очередей. Время восстановления за счет автоматического (программного) устранения ошибок заложенных в эти процедуры измеряются миллисекундами (см.2.4), в то время как на ручное восстановление потребовалось бы как минимум 2-3 мин [35].

Экспериментально показано, что для блока БКП со средствами контроля и восстановления временная избыточность составила на процедуры оперативного контроля не более 0.1 %.

Как показал эксперимент – использование метода сравнения с копией (МОСК) и метода предельных значений (МОПР) в сочетании с методом сравнения по модулю (МОМД) достаточно для успешной борьбы с критическими ошибками кр.ош.1-кр.ош.6.

Успешным можно считать выбор методов устранения критических ошибок: троирования данных (МВТР) и частичной переинициализации (МВЧПИ), использованные в процедурах оперативного контроля в качестве восстановления. Выбор метода МВТР объясняется его эффективностью: очень быстрое и достоверное восстановление при малых операциях (требуется всего одна команда на языке СИ). А выбор метода МВЧПИ обосновывается необходимостью учета спецификации работы с очередями, пулами, восстановление работоспособности состояния в очередях производится за счет ликвидации ссылок между буферами. Время, требуемое на выполнение данного метода также сравнительно маленькое, но при этом методе из-за искажений могут теряться до MAXLFIFO (MAXLIFIFO =5) буферов, что может повлиять на эффективность работы контролируемого блока.

Однако, эти потери неизбежны в любом случае, так как лишь дублирование ссылок при каждом изменении их (частость изменения велика и равносильна частости обработки буфера) можно минимизировать потерю. Но эта задача также невыполнима из-за больших временных затрат.

Экспериментально показано, поле наблюдаемое процедурами ОК1-ОК6 составило не более 2 % области данных узла адаптивной коммутации. Необходимость и целесообразность контроля этого поля в виде текущих проверок были обусловлены

с целью предотвращения от отказов из-за искажений при доступе к требуемой компоненте, к буферам в очередях или пулах, что было экспериментально доказано.

Критические ошибки (кр.ош.7 – кр.ош.17) приводящие к снижению пропускной способности обнаруживались и устранялись процедурами периодического контроля (РК1, PROW3, ANALOS).

Экспериментально установлено, что около 60 % обнаруженных ошибок приходится на долю процедур периодического контроля. При этом типы этих ошибок соответствуют кр.ош.7— кр.ош.14 и они согласуются с таблицей 2.1.

Временная затрата, требуемая па выполнение РК1 0.0367с, в целом, можно считать небольшой, если учесть тот факт, что при каждом выполнении, контролю подвергаются все данные постоянного и квазипеременного характера.

Метод поэлементного сравнения (МОСРЭ), использованный в РК1, для обнаружения неисправностей в компонентах ПО узла АК, выполняющий побайтное сравнение поля компонент с их копиями, метод МВДБ осуществляющий восстановление за счет копий, послужили для контроля данных более 20 компонент. К этим компонентам относятся: управляющая таблица, таблица основных данных, блоков БКП, БКК, БСК, БРК, маршрутизации КП и КК; адресов трактов и линий; контексты трактов и линий, таблицы входных и выходных КК и др. При этом, в зависимости от контролируемого поля: количества КК, тракта, линии — число которых может увеличится до некоторого числа **M**, из-за ограниченности объема памяти ЭВМ и пропускной способности узла. Точно таким образом увеличивается, с ростом числа трактов и линий, количество очередей в узле, подвергающиеся контролю со стороны процедур: PROW3, ANALOS.

Основное назначение возложенное на PROW3: проверка текущего числа буферов в узле предостерегает от больших неприятностей. Возможные блокировки, неверные ссылки, из-за которых очередь переходит в неисправное состояние, последствия чего приводят к большим потерям, к уменьшению пропускной способности, как показал эксперимент, можно не допустить, если использовать довольно не сложную, но эффективную проверку, реализованную в PROW3. Переход по всем очередям и пулом, с целью пересчета числа буферов и сравнение его с заданным числом, требует не более 0.0006 с. Эксперимент показал, что при блокировке какой-либо очереди, если во время не обнаружена критическая ошибка

(кр.ош.9), то при предположении о равномерном поступлении буферов ко всем очередям $1/K$ (K – число очередей) часть буферов будет потеряна.

Однако, в PROW3 лишь устанавливается факт о потери буфера, или о существовании неисправности в какой-либо очереди. Тогда ANALOS проводит тщательный анализ каждой очереди, устанавливает сохранность буферов в очереди, а затем восстанавливает буферы вставляя их в пулы PULBIG, PULLTL и на этом считается, что произведен полный контроль и восстановления в очередях и пулах узла адаптивной коммутации. Достоинство процедуры ANALOS заключается в анализе состояния очередей и приведения их в работоспособное состояние, в восстановлении потерянных буферов и обеспечения синхронной работы основных блоков ПО узла адаптивной коммутации с очередями. При этом на эти операции затрачивается 0.67 мс на ЭВМ СМ-1600 (см.раздел 2.4).

4.6. Параметрический анализ и сравнительная оценка результатов аналитического и имитационного моделирования

Сравнение результатов имитационной и аналитической модели процесса работы блока БКП ПО узла адаптивной коммутации функционирующей совместно со средствами контроля и восстановления и без них в ВС с неисправностями производилось как по коэффициенту готовности (**Krq**), так и по пропускной способности (**Q**).

Коэффициент **Krq**, полученный той и другой моделью для случая работы блока БКП без средств контроля и восстановления (табл.4.1) отличается в тысячных долях.

В отличие от коэффициента **Krq,** коэффициент **Q** полученный по аналитической модели значительно отличается от **Q** полученный по имитационной модели при ограниченном количестве мест (m) в очередях. Результаты таблицы 4.3 показывают, что при $m^{(1)}= 1$, $m^{(2)}= 5$, $m^{(3)}= 9$ Q_1 ($Q_1^{(1)}$ =0.6713, $Q_1^{(2)}$ =0.8479, $Q_1^{(3)}$ =0.8865) полученный по аналитической модели как для наработки на отказ $T^{(1)}$=50000, так и для $T^{(2)}$=5000с – Q_2 ($Q_2^{(1)}$ =0.6441, $Q_2^{(2)}$ =0.8, $Q_2^{(3)}$ =0.8341) возрастают и приближаются к Q (0.9076, 0.8527) полученной по имитационной модели. А более близкие Q той и другой модели можно получить при больших **m**

(**m**>10). При этом Q для имитационной модели вычисляется по формуле

$Q = К_{эф} \bullet Krq$

Близость полученных результатов аналитической и имитационной модели показывает на адекватность аналитической модели изучаемому процессу, что позволяет получать достоверные аналитические оценки для всего ПО узла адаптивной коммутации.

Основными особенностями аналитической модели является получение основных характеристик работы произвольно выбранного блока узла адаптивной коммутации. При этом представленная программа аналитической модели MODEL , позволяет за считанные секунды производить расчет независимо от количества задаваемой информации. Полученные на ЭВМ показатели: финальные вероятности состояний в СМО, пропускная способность (Q), готовность ПО узла адаптивной коммутации, среднее время пребывания заявки узла адаптивной коммутации в очереди и в системе, зависимость от количества мест (буферов) очереди и выбор максимального числа буферов в соответствии с информационной избыточностью, вносимой в ПО узла АК дают исчерпывающую характеристику об исследуемой системе.

В отличие от аналитической модели, имитационная модель трудно реализуема. Например, для реализации только одного блока БКП потребовалось создать более 20 программных модулей.

Однако в имитационной модели можно получить точное значение некоторых характеристик, которые нельзя получить при помощи аналитической модели. К этим данным относятся средняя наработка на отказ, вероятность достоверного обслуживания, которая характеризует работоспособность блока БКП, число отказов.

Кроме этого имитационная модель позволила экспериментально оценить эффективность разработанных средств контроля и восстановления (раздел 2.3) основанные на выбранных в разделе 2.1 методах обнаружения и устранения ошибок.

Также, по результатам имитационной модели процесса работы блока БКП функционирующего совместно с процедурами контроля и восстановления в ВС с неисправностями выявлен диапазон изменения значения ошибок, при наложении которых на область данных появляется отказ в общей области памяти (ООП) ПО узла адаптивной коммутации (рис.4.6).

4.7. Выработка рекомендаций

1.Узлы коммутации сложны при реализации. Создание ПО для них требует учета ряда требований. Одним из основных требований является выполнение в системе реального масштаба времени (р.м.в.).

2. Для обеспечения выполнения функций узла коммутации в системе р.м.в., общая область данных создается в оперативной памяти.

3. Создание общей области данных в оперативной памяти позволяет узлу коммутации обработать кадры (пакеты) наиболее эффективно с максимальным быстродействием.

4. Для повышения надежности сохранности данных общей области памяти рекомендуется использовать методы контроля и восстановления данных. Рекомендуемые методы программно реализованы и составляют программные средства контроля и восстановления (ПСКВ).

5. Для использования разработанных средств ПСКВ разработчику ПО узла коммутации (УК) необходимо выполнить следующие условия:

5.1. Производить условное разбиение области данных на группы данных a_{11}, a_{12}, a_{22}, a_{3L}, a_{32}. Где a_{11} – структура с постоянными данными: a_{12} – структуры с данными, часть которых не является постоянными; a_{21} – структуры с квазипеременными данными; a_{22} – структуры с данными, часть которых не являются квазипеременными; a_{33} – структуры с переменными данными; a_{32} – структуры содержащие информационные и служебные сообщения.

Под постоянными данными будем понимать данные, которые имеют значения присвоенные при инициализации ПО узла коммутации и не меняющиеся в течение всей работы УК. В отличие от постоянных данных — квазипеременные данные меняют свои значения. Частота изменения зависит от организации новых каналов или ликвидации существующих каналов. По средним оценкам продолжительность работы каналов составляет около 3-х минут.

Данные меняющиеся с наиболее большой интенсивностью (в один час это число может составить 144000) относится к переменным данным.

5.2. К данным типа $\mathbf{a_{31}}$ должны быть дополнительно определены: тип буфера (обозначение $\mathbf{a_{31т}}$), имя пула длинных буферов ($\mathbf{a_{31}рд}$), имя пула коротких буферов

($\mathbf{a_{31РК}}$), число длинных буферов ($\mathbf{a_{31кд}}$), число коротких буферов ($\mathbf{a_{31}кк)}$.

5.3. Запустить интерпретатор средств контроля и восстановления (ИН_КВ).

6. Назначение интерпретатор ИН_КВ сводится к автоматическому макрорасширению исходного текста ПО УК средствами контроля и восстановления (рис.4.7).

Функции интерпретатора ИН_КВ.

Основные функции интерпретатора ИН_КВ сводятся к макрорасширению исходного текста:

- программы описания общей области данных (ООД) УК;
- программы инициализация 00Д УК;
- **i** – ой программы выполнения функций УК (i = 1, n; n – число программ реализованных в конкретном УК).

Макрорасширение осуществляется за счет контроля и восстановления. Во время выполнения интерпретатора ИН_КВ формируется таблица связей структур с копиями, составляются программы проведения периодического контроля.

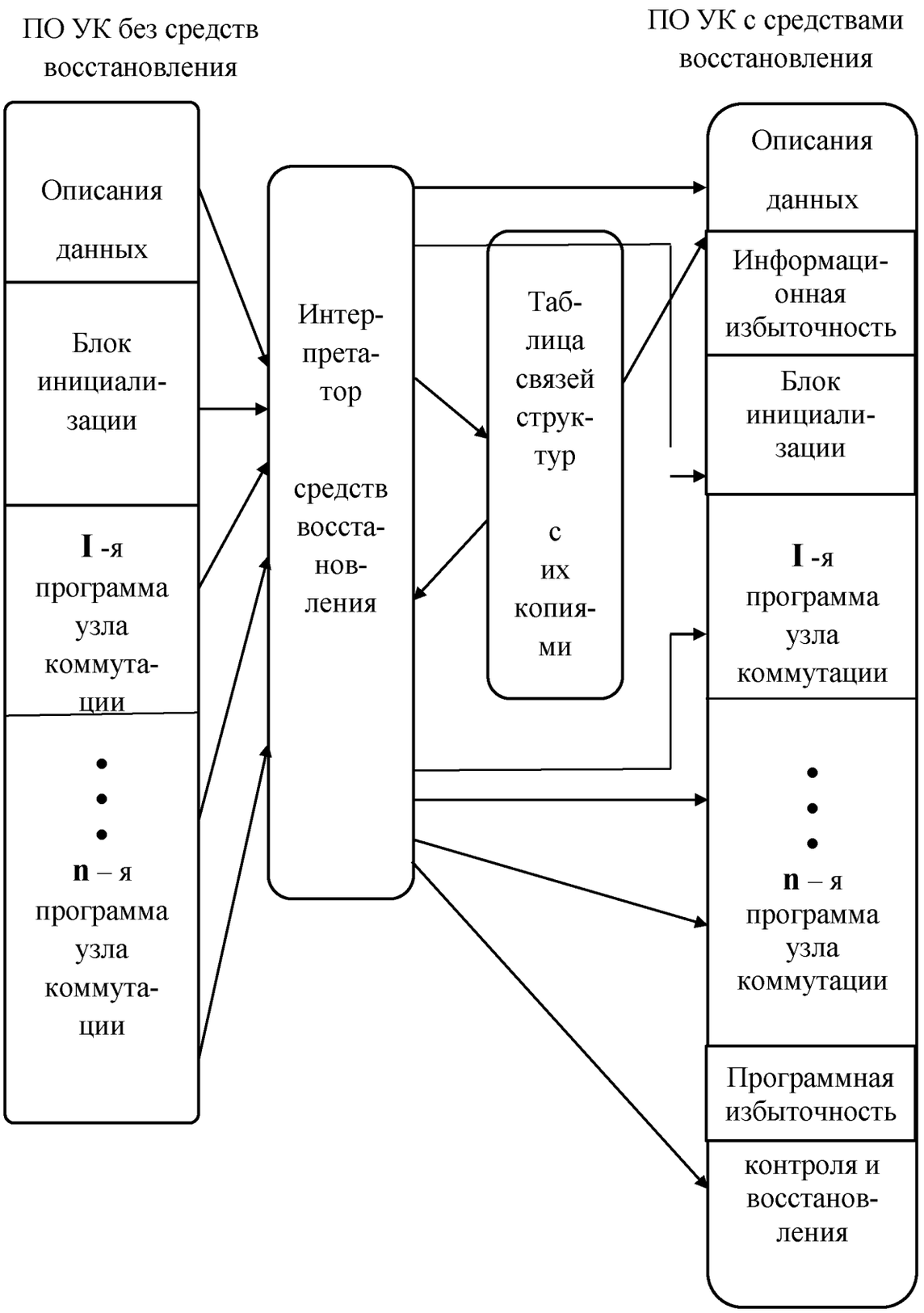


Рис. 4.7. Схема работы интерпретатора средств восстановления

Во время выполнения интерпретатора ИН_КВ формируется таблица связей структур с копиями, составляются программы проведения периодического контроля и восстановления, снятия копий.

Функции, выполняемые интерпретатором ИН_КВ для программ описания ООД УК (ПОД УК).

Основными функциями интерпретатора ИН_КВ для программы ПОД УК являются:

- анализ исходного текста и определение помеченных структур (помеченными структурами являются те структуры, для которых в поле комментарий указывается знак $\mathbf{a_{жж}}$, где $_{жж}$ - номер типа данных);

- формирование таблицы связей структур с копиями;

- макрорасширение исходного текста описания данных УК.

Для выполнения вышеперечисленных функций ИН_КВ (интерпретатор средств контроля и восстановления) происходит просмотр текста программы ПОД УК, в поле комментариев находит помеченные структуры. Действия, выполняемые ИН_КВ после нахождения помеченных структур следующие:

I. Для структур типы a_{11} и a_{21}.

а) составление одной копии путем выбора 8-символьного идентификатора. Наименование идентификатора формируется согласно следующего правила. Первым символом берется первая буква латинского алфавита, вторым, третьем — номер типа данных, остальные символы заполняются первыми, 5-ю символами наименования структуры.

Полученный идентификатор копии заносится в таблицу копий. Перед записью в таблицу этот идентификатор проверяется на условие «идентификатор с таким именем не встречается в данных УК и в таблице копий». Если такой идентификатор встречается в данных УК, то берется 2-я буква из латинского алвафита для 1-го символа идентификатор копий, если и при этом условие не выполняется, то берется следующая буква и т.д.

При условии, когда идентификатор встречается в таблице копий заменяется 8-я буква на одну из букв латинского алфавита до выполнения условия когда идентификатора с таким именем не будет в таблице копий.

б) Определение параметра для контрольной суммы (идентификатор которой **КСим.стр**., где **им.стр**. ― первые 6 букв имени структуры);

в). Запись в таблицу копия следующей информации: тип данных; имя структуры; тип структуры; имя копии; контрольная сумма.

2. Для структуры типа $\mathbf{a_{12}}$ и $\mathbf{a_{22}}$

а) составление 2-х копий путем выбора 8 – символьных идентификаторов.

Наименования идентификаторов формируются согласно порядка приведенного выше.

б) Запись в таблицу копий следующей информации: тип данных; имя структуры; тип структуры; ими 1-й копии; имя 2-й копии.

3. Действия интерпретатора ИН_КВ для структуры $\mathbf{a_{31}}$.

а) Определение типа структуры с обозначением $\mathbf{a_{31Т}}$ из поля комментарий.

б) Изменение типа структуры $\mathbf{a_{31}}$ путем добавления параметра «контр.адр.буфера».

в) Формирование идентификатора для параметра «контр.адр.буф» производится согласно вышеприведенного порядка для первых 3-х символов. Остальными символами можно выбрать следующие выражения ADRBF .

г) Запись в таблицу копий следующей информации: тип данных ($\mathbf{a_{31Т}}$); имя структуры; тип структуры; имя добавленного параметра.

д) Формирование копии путем выбора 8-символьных идентификаторов (см.выше).

е) Запись в таблицу копий следующей информации: тип данных ($\mathbf{a_{31}}$); имя структуры; имя копии структуры.

ё) Определение пула длинных буферов с обозначением $\mathbf{a_{313Д}}$ и коротких буферов с обозначением $\mathbf{a_{31РК}}$.

ж) Запись в таблицу копий следующей информации: тип данных (**$a_{31Р}$** или **$a_{31РК}$**); имя структуры; тип структуры.

з) Определение размера длинного буфера с обозначением **$a_{31КД}$** и короткого буфера с обозначением **$a_{31КК}$**.

и) Запись в таблицу копий следующей информации: тип данных (**$a_{31КД}$** или **$a_{31КК}$);** имя структуры, тип структуры.

й) Для хранения макс.и мин.адреса буферов запись в таблицу копий следующей информации: тип данных (**$a_{31ММ}$**); идент.макс.ад.буф., идент.мин.ад.буф. Где **$a_{31ММ}$** – указывает на допол**нительно вво**димые данные; идент.макс.адр.буф.**и** иден.мин.адр.буф. – идентификаторы максимального и минимального адреса буфера.

Действия, выполняемые ИН_КВ для макрорасширения исходного текста программы ПО УК сводятся к:

а) формированию команд для описания копии структур в виде**:** "тип структуры" – "имя копии структуры";

б) внесению команд определения копии в исходный текст программы ПОД УК;

в) формированию команд описания параметра контрольной суммы структур;

г) внесению команд определения параметра "конт.сумм" структур в исходный текст программы ПОД УК;

д) формированию команд описания идент.макс.буф. и идент.мин.буф. в виде

– "тип данных (целый) "имя идентификатора".

Функции интерпретатора ИН_КВ для программы инициализации (блока БИ) узла коммутации

Основные действия, выполняемые интерпретатором для программы БИ УК заключаются:

а) во включении в исходный текст программы инициализации команду вызова программного средства создания копий (РКСОР);

б) в разработке программного средства РКСОР.

Для включения в исходный текст БИ команду вызова программного средства РКСОР (в зависимости от исходного языка программирования УК) выбирается команда типа CALL РКСОР. Для системных языков: ассемблер, СИ CALL опускается.

Разработку программного средства РКСОР интерпретатор ИН_КВ осуществляет в автоматизированном режиме, путем выполнения следующих этапов:

а) определение с таблицы копии типов структуры, имен структуры, имен копий и др. параметров. Составление команд описания структур приведенных в таблице копий как общими (типа COMMON);

б) составление команды (процедуры) создания копий для структур типа $\mathbf{a_{11}}$, $\mathbf{a_{21}}$, $\mathbf{a_{31}}$;

в) составление команды (процедуры) создания 2-х копий для структур типа $\mathbf{a_{21}}$, $\mathbf{a_{22}}$;

г) составление команды определения макс. и мин.адреса буферов;

д) составление команд вычисления контрольных сумм для структур типа $\mathbf{a_{11}}$, $\mathbf{a_{21}}$.

Функции интерпретатора ИН-КВ для создания программы периодического контроля и восстановления (БКВ).

Основные действия, выполняемые интерпретатором ИН-КВ для разработки программы БКВ осуществляются в автоматизированном режиме, путем выполнения следующих этапов:

а) определение с таблицы копий типов структур, имен структур, имен копий и др. параметров, составление команд описания структур приведенных в таблице копий как общими (типа COMMON);

б) составление команды (процедуры) контроля и восстановления структуры и копий для типов $\mathbf{a_{11}}$, $\mathbf{a_{21}}$;

Функции интерпретатора ИН-КВ для i -ой программы выполнения функций УК (ПРОГ УК)

Основные действия, выполняемые интерпретатором ИН_КВ для **i**-ой программы УК сводятся к следующему:

1. Для структур $\mathbf{a_{11}}$ и $\mathbf{a_{21}}$:

а) определению по тексту программы ПРОГ УК структур с обозначением $\mathbf{a_{11}}$ и $\mathbf{a_{21}}$. Определению с таблицы копий имени копии для этой структуры.

б) определению в этой структуре имени элемента;

в) определению с таблицы копий идентификатора контрольной суммы (КС) для этой структуры;

г) составлению команды обращения к процедуре оперативного контроля ОК1 в виде:

имя проц. (имя струк., имя копии, имя элемента, идент.КС)

д) вставлению в исходный текст программы ПРОГ УК команды вызова процедуры ОК1;

е) вставлению в исходный текст программы ПРОГ УК команды определения копии структуры типа $\mathbf{a_{11}}$ или $\mathbf{a_{21}}$.

2. Для структур типа $\mathbf{a_{22}}$ или $\mathbf{a_{12}}$:

а) определению по тексту программы ПРОГ УК структур с обозначением $\mathbf{a_{22}}$ или $\mathbf{a_{12}}$;

б) определению с таблицы копий имени 2-х копий для этой структуры;

в) определению имени элемента в этой структуре;

г) составлению команды обращения к процедуре оперативного контроля ОК2, в виде:

имя проц. (имя стр., имя 1-й копии, имя 2-й копии, имя элемента)

д) вставлению в исходный текст программы ПРОГ УК команды вызова процедуры ОК2;

е) вставлению в исходный текст программы ПРОГ УК команды определения I-й копии и 2-й копии структур типа $\mathbf{a_{12}}$ или $\mathbf{a_{22}}$.

3. Для структур с обозначениями $\mathbf{a_{31}}$:

а) определению по тексту ПРОГ УК структур с обозначения $\mathbf{a_{31}}$;

б) определении имени копии для этой структуры и параметра "мах.адр.буф" и "мин.адр.буф";

в) описанию и добавлению в текст программы имени копий для структур $\mathbf{a_{31}}$ и параметров мин .и мах.адрес буферов;

г) составлению команды вызова процедуры ОКЗ (проверки очередей типов FIFO или LIFO в виде:

имя проц. (имя струк., имя копий, мин.адр.буф., мах.адр.буф., размеры длинных и коротких буферов);

д) определению команды записи (вставления) и чтения (изъятия) с очереди буфера и созданию аналогичной команды для копии очереди.

в) вставлению составленных команд в исходный текст.

ВЫВОДЫ

1. Разработана имитационная модель, позволяющая исследовать работу отдельно взятого блока узла АК в ВС р.м.в. с неисправностями, возникающие по вине цетрального процессора или оперативной памяти. Она служит для оценки работоспособности отдельно взятого блока при совместном использовании в его составе как средств контроля и восстановления, так и без них.

2. Получены формулы уменьшения объема испытаний за счет сжатия времени эксперимента. Экспериментально была подтверждена достоверность использования этих формул. Полученные результаты показывают, что при больших наработках на отказ при сжатии времени периода эксперимента, пропускная способность меняется незначительно. Погрешность из-за сжатия составляет в среднем число 1 %.

3. На основе полученных результатов имитационной модели процесса работы блока БКП узла АК, функционирующего совместно с процедурами контроля и восстановления в ВС с неисправностями, выявлен диапазон изменения значения ошибок, при наложении которых на область данных появляется отказ в ПО узла АК.

4. Экспериментальная работа, проведенная по имитационной модели функционирования блока БКП узла АК в ВС с неисправностями показывает на эффективность средств контроля и восстановления:

– работоспособность блока БКП узла АК увеличена за счет использования процедур повышения надежности ПО до 0,967 против 0.70;

– наработка на отказ в ПО узла АК при работе блока БКП увеличена более чем в 9 раз;

– готовность ПО узла АК при работе блока БКП равна 0.998 против 0.992.

5. Результаты аналитическом и имитационной модели работы блока БКП узла АК для коэффициента готовности и пропускной способности отличаются в тысячных долях, что показывает возможность замены имитационной модели аналитической для анализа других блоков узла при использовании средств контроля и восстановления.

6. Имитационное моделирование показало на эффективность использования разработанных средств контроля и восстановления, которые и решают задачу адаптации структур данных в условиях возмущающего воздействия среды на объект.

ГЛАВА V. АКТУАЛЬНЫЕ АСПЕКТЫ ОТКАЗОУСТОЙЧИВЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

5.1. Другие направления разработки надежного программного обеспечения

Встроенные электронные устройства могут работать в неопределенной среде. Электромагнитные помехи, колебания напряжения и высокая или низкая температура могут легко вызывать периодические или постоянные отказы в полупроводниковых устройствах, что потенциально может привести к ошибкам работы в системе реального времени. Чтобы справиться в такой ситуации системы реального времени обычно имеют отказоустойчивый механизм. Хотя существующий отказоустойчивый механизм допускает некоторые виды ошибок, ограничения. Он не может справиться с некоторыми фатальными ошибками или ситуациями с огромным количеством ошибок. В работе [100] в предположении, что досягаемость разлома является пуассоновским процессом приближенно предлагается отказоустойчивая модель встроенной системы. При анализе отказоустойчивого планирования используются методы обхода в ширину построить сценарии ошибок системы реального времени. Кроме того, предложенный в работе ключевой новый алгоритм предназначен для расчета надежности системы. Результаты эксперимента показывают, осуществимость предлагаемого метода оценки.

В настоящее время [100] встроенное программное обеспечение, работающее в критически важных с точки зрения безопасности системах, таких как авионика (бортовое радиоэлектронное оборудование), вычисление миссии и управление автомобилем, в основном используется для сбора информации от внешних раздражителей и своевременно реагировать на различные помехи среды. Для удовлетворения требований по его критическим свойствам, особенно надежности, необходимо точно оценить способность отказоустойчивости системы реального времени, в противном случае несоблюдение требований по времени может привести к системным сбоям, что

может привести к в катастрофических последствиях. Надежность [101] – это способность системы или компонента выполнять возложенные на него функции в указанных условиях в течение определенного периода времени.

Например, IEC61508 [102] предлагает набор методов оценки риска.

Анализ надежности был горячей темой исследований с начала 1960-х годов. Однако, большинство методов применялись на этапе внедрения программного обеспечения. Оценка надежность на ранней стадии проектирования системы является более быстрым формальным методом. Он создает вероятностную модель системы реального времени и анализ временного свойства на основе анализа расписания. Он заблаговременно выявляет потенциальные угрозы безопасности, поэтому дорогостоящие изменения вносятся на этапе проектирования системы. Много существует литературы, где в основном рассматривается взаимосвязь между приоритетом задачи и временем отклика в наихудшем случае при отказоустойчивом состоянии.

В связи с растущими требованиями безопасности встроенного программного обеспечения многие исследования возникли в области оценки отказоустойчивости и надежности. Традиционный подход в основном опирается на данные об ошибках тестирования программного обеспечения для прогнозирования поведения ошибок в будущем.

Метод, основанный на статистике, такой как моделирование Монте-Карло [103], был используемые для работы с переходными неисправностями, зависели от имитационного эксперимента с большими количествами образцов.

Из-за большого количества вычислений в статистическом подходе формальный анализ планирования отказоустойчивости и анализ надежности были рассмотрены некоторыми разработчиками в предыдущей исследовательской деятельности. Структура отказоустойчивости программного обеспечения в режиме реального времени системы [104] и метод контрольно-пропускного и временного резервирования [105] дают основные модели

прогнозирования надежности программного обеспечения. А. Бернс и др. [106] ввел вероятностную модель в анализе расписания в рамках вероятностной гарантии того, что все задачи всегда необходимо завершать к их срокам. И. Бростер и др. [107] расширил этот метод в сети CAN вычислять точные прогнозы вероятности отказа по распределению вероятностей время отклика. Однако эти подходы [106,107] имеют определенные ограничения и приводят к крайне пессимистичным результатам.

Критически важные с точки зрения безопасности приложения должны работать правильно и в установленные сроки ограничений даже при наличии ошибок. Г. Лима и др. [108] предложил наихудший случай анализ планирования времени отклика для отказоустойчивых систем жесткого реального времени и с учетом восстановления задач, запущенных с более высоким приоритетом, а затем введенных алгоритм назначения приоритетов [109] для повышения отказоустойчивости системы. Они обсудили взаимосвязь между использованием процессора и отказоустойчивостью. Ли и др. [110] предлагают новый отказоустойчивый алгоритм назначения приоритетов, основанный на анализе планирования времени отклика в наихудшем случае для отказоустойчивых задач жесткого реального времени с ограниченными уровнями приоритета [111] и произвольно большими сроками [112].

В работе [113] обсудили время отклика в наихудшем случае для задач в рамках группового упреждающего планирования и проверки временного свойства на раннем этапе проектирования системы. Однако эти методы в основном сосредоточены на взаимосвязи между планируемостью приоритетности задач и отсутствием вероятностной модели для оценки надежности систем, которая была включена в рассмотрение.

М. Себастьян и др. представили метод оценки надежности [114], который вычислял надежность системы в течение определенного периода времени. Сравнение со статистической ошибкой моделирование, метод численного анализа был быстрым и точным, но его использовали в сети CAN с

неупреждающей стратегией планирования. С. Гуи и др. [115] представлена модель надежности для анализа возможности планирования при возникновении сбоев и вычисления вероятности того, что задачи все еще могут быть запланированы в наихудшем сценарии выполнения, но фактор времени в надежности не учитывался.

В. Изосимов, Петру Элес и др. [116,117] предлагает несколько алгоритмов оптимизации для отображения в реальном времени. Она определяет критически важные для безопасности приложения на распределенных встроенных системах путем обработки повторного выполнения и репликации для допустимых временных сбоев.

Вышеупомянутый метод надежности [118] сосредоточен на переходных неисправностях, таких как изменение температуры и электромагнитные помехи, а не иметь дело с постоянными неисправностями. И представляет новую модель надежности, в которой применяется метод анализа планируемости на основе времени отклика в наихудшем случае для оценки надежности системы на этапе проектирования программного обеспечения реального времени.

В работе [118] приводится обзор многовариантного программирования (включая программирование с блоками восстановления), предназначенного для систем обработки информации и управления в реальном масштабе времени, операционных систем и ряда других. Рассматриваются методы повышения отказоустойчивости и уровня правильности программ, а также методы улучшения их функциональных характеристик. Анализируются результаты экспериментальных исследований. Вводится классификация отказоустойчивых вычислительных систем с их программным обеспечением по признакам идентичной и неидентичной избыточности разных типов.

В проблема надежности программного обеспечения, надежность обычно понимается как вероятность работы программного обеспечения без отказов в течение заданного времени, определяемая с учетом стоимости для пользователя

каждого отказа [34]. Понятие «надежное программное обеспечение» имеет два значения, различающихся по смыслу: правильность программ и устойчивость их функционирования.

Программа является правильной, если она удовлетворяет своим функциональным спецификациям. При этом полагается, что она вырабатывает ожидаемые результаты, пока входные данные удовлетворяют спецификациям.

Программа является устойчивой (отказоустойчивой), если она обеспечивает полезную работоспособность не ниже заданного минимального уровня в условиях неожиданного или неблагоприятного влияния на нее окружающей среды, включая искажения входных данных и неисправности аппаратуры [131]. К неблагоприятным факторам добавляется и влияние не выявленных остаточных программных ошибок, проявления которых могут быть самыми различными и неожиданными. Классификацию большого числа моделей и обзор методов анализа надежности программного обеспечения, в том числе связанных с темой настоящего обзора, можно найти в [130].

Особо остро проблема обеспечения надежности программ встает при создании больших программных систем многократного применения, таких, как крупные системы обработки информации и управления в реальном масштабе времени и операционные системы. Надежность такого рода систем должна быть очень высокой, нередко предельно возможной. Их отказы сопровождаются не просто ущербом, оцениваемым той или иной стоимостью, а могут вызывать катастрофические последствия.

По мере отладки и тестирования программ отыскание их ошибок становится все более трудным, и продолжение этого процесса, начиная с некоторого уровня надежности, приводит к существенному увеличению времени на создание программной системы и к резкому возрастанию ее стоимости. В этих условиях конкурентоспособным по общим затратам на весь жизненный цикл программ становится подход, связанный с независимой разработкой, вводом и сопровождением двух или более различных вариантов

(версий) программ, выполняющих одни и те же функции. При функционировании выходные данные этих вариантов сравниваются в автоматическом режиме и осуществляется соответствующий выбор результатов.

Такое программирование получило название многовариантного, при двух вариантах оно называется дуальным. Наличие нескольких вариантов программ позволяет улучшить надежность характеристики программной системы как за счет повышения уровня ее правильности при разработке, так и за счет повышения устойчивости функционирования при эксплуатации. Оно позволяет также улучшить функциональные характеристики за счет адаптивного автоматического выбора для исполнения тех или иных вариантов программ в зависимости от изменений внешней обстановки.

Существуют два основных подхода к решению проблемы надежности вычислительных систем, первый из которых направлен на достижение безотказной работы, а второй — работы, устойчивой к отказам, причем они оба относятся как к аппаратным, так и программным средствам. Эти подходы взаимно дополняют друг друга.

В целях обеспечения безотказности, т. е. предотвращения неисправностей, необходимы безотказные компоненты и такие методы и технологии разработки, сборки и отладки, которые не допускают проектных ошибок. Этих целей, однако, не удается достигнуть ни для аппаратных, ни для программных средств — для первых, главным образом, из-за неизбежных физических сбоев и отказов, а для вторых –из-за не выявленных при отладке ошибок, которые обычно остаются в сколько-нибудь сложных программах.

Такое положение стимулирует разработку и применение методов обеспечения устойчивости к отказам. Под отказоустойчивостью обычно понимается такое свойство вычислительной системы, которое обеспечивает ей, как логической машине, возможность автоматического продолжения полезных действий, заданных программой, с возвращением после возникновения

неисправностей из ошибочных состояний к правильной обработке информации [119].

Устойчивость к отказам определяется организацией структуры и функционирования системы с ее аппаратными и программными средствами и достигается при помощи временной, аппаратурной, программной и информационной избыточности, которая была бы излишней в случае безотказности.

В упомянутой выше работе указаны вычислительные системы, отказ и последующий простой которых приводят к ситуации, опасной для жизни человека, к тяжелым экономическим последствиям, к фактической потере системы (на необслуживаемых космических и подводных аппаратах и т. п.). В этих и многих других случаях требуется введение избыточности для обеспечения устойчивости к отказам.

Избыточность структурных и функциональных компонентов аппаратных и программных средств, которая лежит в основе обеспечения отказоустойчивости вычислительных систем, наиболее широко используется в форме двукратного или многократного копирования идентичных устройств или машин и выполняемых ими программных модулей или программ в целом. Такую избыточность естественно называть идентичной.

Типичными примерами такого рода являются многомашинные и многопроцессорные системы с продублированными программами. Напомним, что первоначальной целью построения систем с несколькими процессорами было достижение высокой готовности, и только после решения этой задачи возможности таких систем стали использоваться также и для повышения производительности [122]. Хорошо известно, что вычислительные системы с продублированной или многократно копированной аппаратурой и соответствующими копиями программ после их доводки обычно работают устойчиво, несмотря на сбои и отказы компонентов аппаратуры.

Глобальные отказы таких систем, приводящие к перерывам в их работе и требующие восстановления вручную, вызываются, как правило, упущениями в алгоритмах и ошибками в программах решаемых задач, непредвиденными сочетаниями входных и промежуточных данных. В этих случаях дублирование и многократное копирование программ оказываются неэффективными и никак не могут помочь в деле автоматического восстановления и продолжения вычислительного процесса, поскольку программные ошибки также копируются.

Типичный пример такого рода описан в [120]. Симметричная вычислительная система с тремя идентичными ЭВМ и копиями реализованных на них программ, осуществляющая управление железнодорожным движением в реальном масштабе времени, за три года работы отказала только 7 раз, из них 5 раз вследствие ошибок в программах.

Более широкие возможности дает другой подход к программной избыточности, а именно многовариантное программирование, основанное на применении двух или более независимо разработанных различных программ решения одних и тех же задач. При этом могут использоваться одни и те же языки программирования и алгоритмы решения задач, либо одни и те же языки, но разные алгоритмы, либо разные языки, но одни и те же алгоритмы, либо, наконец, различные языки и различные алгоритмы.

Многовариантные программы могут быть реализованы на однопроцессорных, многомашинных, многопроцессорных и других типах систем. Такую избыточность естественно называть неидентичной (она может быть отнесена и к аппаратуре). Подход к многовариантному программированию предложен в [126], такого рода предложения сделаны также в [123, 127, 128, 129], а его современная практически реализуемая форма дана в [124, 125] с использованием термина «многовариантное программирование» (TV-version programming).

5.2. Качество и отказоустойчивость реально функционирующих

систем

Системы реального времени – это системы, в которых компьютер обязан своевременно реагировать на внешние раздражители. Приложения реального времени должны работать корректно даже при наличии сбоев. Отказоустойчивость может быть обеспечена либо аппаратным, либо программным обеспечением, либо временным резервированием. Критические для безопасности приложения имеют строгие временные и стоимостные ограничения, а это означает, что необходимо не только допускать ошибки, но и соблюдать ограничения. Крайний срок планирования означает, что обрабатывается задача с самым ранним требуемым временем отклика. Наиболее распространенными алгоритмами планирования являются: монотонная скорость (RM) и самый ранний крайний срок вперед (EDF). В этой статье рассматривается взаимодействие между отказоустойчивой стратегией и реальной EDF [132].

Системы реального времени можно разделить на системы жесткого реального времени, в которых последствия нарушения сроков могут быть катастрофическими, и системы мягкого реального времени, в которых последствия относительно терпимы. В системах жесткого реального времени важно, чтобы задачи выполнялись в установленные сроки даже в случае сбоя [132]. Примерами систем жесткого реального времени являются системы управления на космических станциях, системы автопилота и системы наблюдения за пациентами в критических состояниях.

В системах мягкого реального времени более важно как можно быстрее обнаружить неисправность, чем замаскировать ее. Примерами мягких систем реального времени являются все виды приложений для бронирования авиабилетов, банковских операций и электронной коммерции.

Отказоустойчивость – это способность продолжать работу, несмотря на отказ ограниченного подмножества аппаратного или программного обеспечения. Таким образом, цель проектировщика системы состоит в том,

чтобы гарантировать, что вероятность отказа системы будет приемлемо малой. Может быть аппаратная или программная ошибка, которая мешает системам реального времени уложиться в сроки.

В случае сторожевых таймеров [133] ход программы или передаваемые данные периодически проверяются на наличие ошибок. Простейшая схема сторожевого таймера, сторожевой таймер, отслеживает время выполнения процессов, не превышает ли оно определенного предела.

Отказоустойчивая система должна продолжать работать, несмотря на выход из строя некоторых ее частей, у нее должны быть резервные мощности для запуска. Есть два способа сделать систему более устойчивой к сбоям [134]. Аппаратное обеспечение: этот метод основан на добавлении в систему дополнительного резервного оборудования, чтобы сделать ее отказоустойчивой. Программное обеспечение: этот метод основан на дублировании кода, процесса или даже сообщений, в зависимости от контекста.

Типичным примером применения вышеуказанных методов может быть система автопилота на борту крупногабаритного пассажирского самолета [135].

Пассажирский самолет обычно состоит из центральной системы автопилота с двумя другими резервными копиями. Это пример создания системы с двумя другими резервными копиями. Это пример обеспечения отказоустойчивости системы путем добавления резервного оборудования. Две дополнительные системы не будут использоваться, если основная система не будет полностью сломана.

В работе [136] представлены анализ трех основных классов оценки надежности программного обеспечения.

Анализ надежности «черного ящика»: оценка надежности программного обеспечения на основе наблюдения за неисправностями при испытаниях или эксплуатации. Эти подходы называются черным ящиком и коробка подходит, потому что внутренние детали программного обеспечения не учитываются.

Анализ надежности на основе метрик программного обеспечения, включает: оценку надежности на основе статического анализа программного обеспечения (например, строки кода, количество операторов, сложность) или процесса и условия его разработки (например, опыт разработчиков, прикладные методы тестирования).

Следующая, анализ надежности на основе архитектуры: оценка надежности программной системы от надежности программных компонентов и системной архитектуры (способ система состоит из компонентов). Эти подходы иногда называются оценкой надежности на основе компонентов (CBRE) или подходами серого или белого ящика.

Многие концепции проектирования надежности программного обеспечения адаптированы из старых и успешных методов аппаратной надежности. Применение методов аппаратной надежности к программному обеспечению должны быть сделаны с осторожностью, так как есть некоторые фундаментальные различия в характере аппаратного и программного обеспечения и процессах его отказа. Следовательно, хорошо зарекомендовавшие себя концепции надежности оборудования могут работать поразному (обычно не очень хорошо) для софта. Было даже предложено, чтобы «аппаратно-мотивированные меры такие как mttf, mtbf не должны использоваться для программного обеспечения без обоснования» [136].

Сегодня проектирование надежности программного обеспечения является отдельной областью. Ученые исследователи по измерению надежности программного обеспечения (например, работа Cheung, Littlewood , Musa и др.) обратились к характеристикам надежности программного обеспечения и адаптировал показатели надежности оборудования. Тем не менее, эмпирическая оценка важнее, чем надежность концепции, полученные из аппаратных подходов, могут быть применены к программному обеспечению. Например, такая эмпирическая оценка оценки надежности на основе компонентов была представлена Krishnamurthy и Mathur.

Несмотря на основные преимущества, оценка надежности программного обеспечения (с такими моделями, как модели роста надежности) недостаточно эффективны для удовлетворения очень высоких требований к надежности (например, 10-9 вероятности отказа в час).

Ошибки программного обеспечения – это ошибки дизайна.

Основное различие между «аппаратными» и «программными» сбоями заключается в лежащей в их основе неисправности.

Традиционно большая часть аппаратных отказов считается следствием физического изнашивания или износ. Рано или поздно эти естественные разломы, могут привести неисправности в аппаратных компонентах и, следовательно, привести к сбоям.

Опыт показал, что эти физические эффекты хорошо описываются экспоненциальной зависимостью уравнения по отношению ко времени. Использование обычно ускоряет снижение надежности, но даже неиспользуемое оборудование портится. Физическое разделение и изоляция неисправностей (например, электрические соединения с высоким импедансом и оптические соединители, например, применяемые Wensley и другие) позволили предположить (приблизительно) статистическую независимость процессов разрушения (природных разломов).

Тот факт, что это так называемое предположение о независимости справедливо для физических неисправностей, не только значительно снижает сложность модели надежности. Более того, это делает использование избыточности очень эффективным в контексте аппаратной отказоустойчивости. Такие концепции, как «горячее» резервирование в сочетании с голосованием или резервным резервированием (реконфигурация при обнаружении отказа), сделало его возможным проектирование систем с высокой аппаратной надежностью.

Ошибки проектирования являются другим источником сбоев. Они возникают в основном из-за человеческого фактора в процессе разработки или

обслуживания. Ошибки проектирования приведут к отказу при определенных обстоятельствах. Вероятность активизации конструктивной ошибки, как правило, зависит только от использования и не зависит от времени. По возрастанию сложности оборудования системы, ошибки проектирования становятся все более и более серьезной проблемой для измерения надежности оборудования, так что «разделение между аппаратной и программной надежностью несколько искусственный».

Программное обеспечение – это чистый дизайн считают авторы, на основе утверждения, и, следовательно, программные сбои вызваны разломами дизайна. Обратите внимание, что термин «дизайн» используется в широком смысле.

Дизайн находится в надежности программного обеспечения и относится ко всем этапам разработки программного обеспечения от требований к реализации [136]. Таким образом, неисправности, возникающие в процессе реализации также считаются конструктивными ошибками. В отличие от аппаратного обеспечения, программное обеспечение может быть совершенным (то есть безошибочным). К сожалению, обычно невозможно разработать сложное безотказное программное обеспечение, и даже в этом случае редко можно гарантировать, что программное обеспечение бесплатно неисправностей

Некоторые формальные методы могут доказать правильность программного обеспечения - это означает, что соответствует спецификации документа. Однако сегодняшние методы формальной проверки не предназначены для применения в больших программных системах, таких как потребительские операции системы или текстовые процессоры. Кроме того, правильность не гарантирует достоверности потому что сам документ спецификации уже может быть ошибочным. Так как это невыполнимо разрабатывать сложные программные системы без ошибок, а отсутствие ошибок не может быть гарантировано, надежность программного обеспечения должна быть оценена, чтобы выполнить высокие требования надежности.

Процесс разрушения проектных неисправностей отличается от процесса («аппаратных») естественных недостатков. Очевидно, что копии (обычного) программного обеспечения потерпят неудачу вместе, если будут выполняться с одинаковыми параметрами. Это показывает, что предположение о независимости не выполняется. И вероятность отказа копий программного обеспечения полностью зависит от текста оригинала. Это делает многие принципы отказоустойчивости оборудования неэффективными для программного обеспечения. Вместо того, чтобы использовать избыточных копий, надежность программного обеспечения может быть повышена за счет разнообразия дизайна. А распространенным подходом для этого является так называемое программирование N-версии (рассмотренное в Avižienis, введено Chen и Avižienis). Однако исследования Knight и Leveson указывает, что разнообразие дизайна, вероятно, будет менее эффективным для программного обеспечения чем N-модульная избыточность в проектировании надежности оборудования.

Некоторые исследования показали, что для сложных систем большинство отказов, как правило, вызванные программными сбоями (см., например, Gray). Хотя программные ошибки являются конструктивными ошибками, их поведение в надежных системах похоже на недостатки нестационарного оборудования. Это связано со стохастикой условий их активации.

Профили использования программного обеспечения Littlewood и Strigini утверждают, что надежность программного обеспечения должна быть вероятностной мерой. Потому что процесс отказа, т. е. то, как отказы становятся активными и вызывают отказы, зависит от входной последовательности и условий работы, и они не могут быть предсказаны с абсолютной уверенностью. Человеческое поведение вносит неопределенность и, следовательно, вероятность в надежность программного обеспечения, хотя программное обеспечение обычно дает сбой одинаковым образом для одних и тех же условий эксплуатации и одинаковым параметрам.

Дополнительный повод для утверждения вероятностной меры заключается в том, что обычно можно только приблизительно определить количество ошибок сложной программной системы. Чтобы выдать различные способы использования, концепции профилей пользователей и операционных профилией [136], которые являются общими для (черный ящик или белый ящик) измерения надежности программного обеспечения, необходимо создавать модели. Эти модели используют вероятности для взвешивания различных способов использования программного обеспечения.

Профили использования можно использовать и для оборудования. Для разработчиков программного обеспечения это легко (и часто практикуется) для включения «чрезмерной дополнительной функциональности». С этой точки зрения, взвешивание запросов на обслуживание кажется особенно важным для программного обеспечения.

Помимо использования программного обеспечения, может потребоваться включить другую контекстную информацию в оценка надежности. Это необходимо, потому что надежность программного обеспечения более чувствительна к различиям в операционных контекстах, чем надежность аппаратных средств. Другими словами, часть программного обеспечения, которая была надежной в одной среде, может быть очень ненадежной в немного другом.

Задача структурирования показателей качества ИВС (информационно-вычислительных систем), в том числе и на транспорте, сопряжена с рядом сложностей [137]. Весьма часто на практике предлагаются субъективные и неоднозначные структуры по оценке тех, либо иных показателей качества информационно-вычислительных систем. Кроме того, системы железнодорожного транспорта обладают своей спецификой [138]. Ряд показателей качества стандартизован, в том числе: показатели качества продуктов питания, ряда сферы услуг, программного обеспечения и пр.

Издаются учебники по управлению качеством, например [139]. Опубликован ряд статей, посвященный проблемам качества услуг [140, 141]. В области оценки качества информационных систем можно отметить работу Г.Н.Исаева [142]. Проблеме оценке качества информационных систем уделяется большое значение и в иностранных публикациях Своеобразный подход к анализу проблемы качества КИС (Корпоративной Информационной Системы) предложили английские исследователи М.Тэйлор и Дж.ДаКоста [143]. Интересна разработка ученых – финнов И.Тервонен и П.Керола [144], представивших развитие системы качества и методики создания КИС в виде спирали.

В [145] изложена методология информационной оценки качества. Статья [146] посвящена оценка информационного качества с помощь многокритериального анализа. В [147] разработана особая гомогенная структура для оценки качества данных. В настоящее время широко используется группа международных стандартов по управлению качеством и обеспечению качества серии ISO 9000[100]. В качестве аналога упомянутой группы для НАТО используется AQAP1. Если обратиться к группе Российской нормативно-технической документации, то следует отметить, что данная группа относит программное обеспечение скорее к некой материальной сфере, то есть к «продукции производственно-технического назначения».

Однако, это весьма спорно и, по мнению авторов, программное обеспечение скорее относится к некой нематериальной сфере. Наиболее известным из российских стандартов, позволяющих оценить качество программного средства, является ГОСТ 28195-89. «Оценка качества программных средств. Общие положения», который, всё же, в полной мере не охватывает всех аспектов и не учитывает современные тенденции и технологии разработки.

В работе [148] делается вывод, что существующая система общетехнических показателей надежности не позволяет учесть особенностей

вычислительных систем, работающих в реальном времени. Для таких систем при оценке надежности при требовании своевременного выполнения критических запросов необходимо определить вероятность выполнения запроса с задержкой, меньшей предельно допустимой величины t_0, с учетом готовности (работоспособности) системы при поступлении запроса (коэффициент готовности) и ее безотказности во время ожидания и обслуживания запроса.

Для систем реального времени как дополнительный показатель может оцениваться вероятность работоспособных состояний системы, для которых среднее время пребывания запросов меньше допустимых ограничений [149]. При этом считается, что восстановление системы во время выполнения критической задачи невозможно.

ЗАКЛЮЧЕНИЕ

Выполненные в данной работе исследования направлены на решение практических задач повышения качества и эффективности узла адаптивной коммутации сетей ЭВМ. Основные результаты монографии заключаются в следующем:

1. Проведен анализ специфических особенностей узла АК как объекта управления со сложной структурой. Определены факторы, влияющие на надежность его работы. Осуществлена постановка задачи обеспечения эффективного управления структурами данных узла АК.

2. Осуществлен выбор и обоснование целей и критериев эффективного управления узлом АК.

3. Определены типы критических ошибок, возникающих в области данных узла АК из-за возмущающего воздействия среды, являющиеся одними из причин процедурных ошибок в вычислительных сетях.

4. Разработаны эффективные методы обнаружения и устранения критических ошибок в области данных узла АК. Составлены алгоритмы для повышения надежности узла на основе этих методов. Выполнена программная реализация и экспериментальное исследование этих алгоритмов.

5. Разработаны и исследованы эффективные алгоритмы адаптации структур данных узла АК, в условиях стохастического поведения возмущений.

6. Разработана аналитическая модель оценки надежной работы узла АК, функционирующего в реальном времени в условиях сбоев и отказов ЗВМ для случаев использования средств повышения надежности программного обеспечения и без них.

7. Проведены исследования эффективности разработанных средств адаптации структур данных к различным возмущениям. Экспериментально получены их эффективность и сложности.

8. Методы и модели, предложенные в данной работе использованы при разработке пакета программ повышения надежности блока узла адаптивной

коммутации. Пакет сдан в ГОСФАП(№ 5086 000 0127 от 07.02.86г.)

ЛИТЕРАТУРА

1. Барзилович К.Ю., Беляев Ю.К., Каштанов В.А. и др. Вопросы математической теории падежности. /Под ред. Б.В. Гнеденко. – М.: Радио и связь, 1983.–376 с.

2. Барлоу Р., Прошан Ф. Математическая теория надежности. /Пер. с англ. – М.: Советское радио,1969. – 420 с.

3. Беляев Ю.К, Богатырев В.А., Болотин В.В. и др. /Под ред. И.А. Ушакова. Надежность технических систем. Справочник. – М.: Радио и связь,1985. – 600 с.

4. Букин И.М. Способ контроля каналов связи в вычислительных сетях. – Автоматика и вычислительная техника, 1986, №3. – С. 55-60.

5. Бусленко Н.П. Моделирование сложных систем. – М.: Наука, 1968. – 399 с.

6. Вентцель Е.С. Исследование операции. – М.: Советское радио, 1972. – 551с.

7. Вентцель Е.С. Исследование операций: принципы, методология. – М.: Наука, 1980. – 207 с.

8. Вентцель Е.С..Овчаров Л.А. Прикладные задачи теории вероятностей. – М.: Радио и связь, 1903. – 416 с.

9. Верлань А.Ф., Ефимов И.Е. Латышев А.В. Вычислительные процессы в системах управления и моделирования. – Л.: Судостроение, 1981. – 248 с.

10. Ю. Гласс Р. Руководство по надежному программированию. – М.: Финансы и статистика,1982. – 280 с.

11. Глушков В.М. и др. Сети ЭВМ. – М.: Связь,1977. – 280 с.

12. Геленбе Е. Модель восстановления информации методом кратных контрольных точек. // Автоматика и телемеханика,1979, № 4. – С.142–151.

13. Головкин Б.Л. Надежное программное обеспечение. // Зарубежная радиоэлектроника, 1978, № 12. – С.3-61.

14. Девис Д., Барбер Д., Пранс У, Соломинидес С. Вычислительные сети и сетевые протоколы. Перевод о английского. /Под ред. Самойленко С.И. – М.: Мир, 1982. – 563 с.

15. Джонсон Н., Лион Ф. Статистика и планирование экспериментов в технике и науке. Методы обработки данных. – М.: Мир,1980.

16. Диллон Б., Сингх Ч. Инженерные методы обеспечения надежности систем. – М.: Мир,1984. – 318 с.

17. Захаров Г.П. Методы исследования сетей передачи данных. – М.: Радио и связь, 1982.

18. Йодан Э. Структурное проектирование и конструирование программ. – М.: Мир, 1979. – 416 с.

19. Калниньш Л.А., Борзов Ю.В. Инвентаризация идей тестирования программ. ЛатГУ им.П.Стучки. – Рига, 1981. – 54 с.

20. Карась В.М. Повышение устойчивости вычислительных процессов к сбоям ЭВМ программным способом. //Управляющие системы и машины, 1982, № 3.

21. Келлон Р. Межсетевой протокол. – М.: Мир, ТИИЭР, т.71, № 12. – С.73–80.

22. Керниган Б., Ритчи Д., Фьюэр А. Язык программирования СИ. Задачи на языке СИ. – М.: Финансы и статистика,1985. –280 с.

23. Клейнрок Л. Вычислительные системы с очередями. – М.: Мир, 1979. – 600 с.

24. Козлик Г.А., Карась В.М., Кириллов И.А. Определение оптимального числа контрольных точек программ. //Управляющие системы и машины, 1977, № 2. – С.41– 42.

25. Конард Дж.У. Услуги и протоколы канального уровня. – М.: Мир, ТИИЗР, 1983, т.71, № 12. – С.61-68.

26. Креденцер Б.П. Прогнозирование надежности систем с временной избыточностью. – Киев: Науково думка, 1978. – 237с.

27. Липаев В.В., Гиганов П.Г., Просвирин В.М. Оперативный контроль функнционирования алгоритмов управляющих систем в реальном масштабе времени. – УСиМ: 1973, № 2. – С.62-68.

28. Липаев В.В., Фидловский Л.Л., Филиппович В.В., Шнейдер Б.Н. Отладка систем управляющих алгоритмов ЦВМ реального времени. – М.: Советское радио, 1974.

29. Липаев В.В. Проектирование математического обеспечения АСУ. – М.: Советское радио, 1977. – 400 с.

30. Липаев В.В. Основные понятия теории надежности при анализае функционирования комплексов программ управления. // Управляющие системы и машины, I980, № 5. – С. 3-7.

31. Липаев В.В. Надежность программного обеспечения АСУ. – М.: Энергоиздат, 1981. – 240 с.

32. Липаев В.В. Тестирование программ. – М.: радио и связь.1986. – 296 с.

33. Майерс Г. Надежность программного обеспечения. – М.: Мир,1980. – 360 с.

34. Майерс Г. Искусство тестирования программ. – М.: Финансы и статистика, 1982.

35. Мартин Дж. Системный анализ передачи данных. II том – М.: Мир,1975. – 431с.

36. Мартин Дж. Программирование для вычислительных систем реального времени. – М.: Наука, 1978.

37. Макклелланд Р.М. Услуги и протоколы физического уровня. – М.: Мир, ТИИЭР, т.71, № 12. – С.53-60.

38. Муса Дж.Д. Измерение и обеспечение надежности программных средств. – М.: Мир, ТИИЭР, 1980, № 10. – С.113-128.

39. Операционная система ОС РВ 2.1. Том YI. Языки программировании. Книга 2. Система программирования на языке ФОРТРАН IУ. Описание языка. Ч.072. 209–35-01.

40. Операционная система ОС РВ 2.1. Том I. Книга 1,2. Управляющая программа.

41. Овчаров Л.А. Прикладные задачи теории массового обслуживания. – М.: Машиностроение, 1969. – 324 с.

42. Пархоменко П.П. Правильщиков П.А. Диагностирования программного обеспечения (обзор).– Автоматика и телемеханика, 1980, № I, с 103-121.

43. Растригин Л.А. Вычислительные машины, системы, сети.... – М.: Наука, 1982. – 224 с.

44. Растригин Л.А. Адаптивные компьютерные системы. – М.: Знание, 1987. – 64 с. (Новое в жизни, науке, технике, Серия "Радиоэлектроника и связь", № 10).

45. Растригин Л.А., Маджаров Н.Е. Введение в идентификацию объектов управления. – М.: Энергия,1977. –412 с.

46. Растригин Л.А. Адаптация сложных систем. – Рига: Зинатне, 1981. – 375 с.

47. Рахматкаримов Э.У., Абидов А.А. Анализ подходов к тестированию программного обеспечения вычислительных систем. Известия АН УзССР. Серия техн.наук. – Ташкент: Деп.в ВИНИТИ I9.10.84, № 6791-84, – 21 с.

48. Рахматкариев Э.У., Абидов А.А. К использованию методов повышения надежности для программ, функционирующих в системе реального масштаба времени. – Методологические и прикладные аспекты систем автоматизированного проектирования и управления в отраслях народного хозяйства, II респ.конф. – Ташкент, 3-5 апр. 1985. Стр.13.

49. Рахматкариев Э.У., Абидов А.А. Процедуры контроля и диагностики блока коммутации пакетов (ПКДБКП). – М.: ВИНИТИ, 1986 (инв. № 508600001 от 7.0286).

50. Рахматкариев Э.У., Абидов А.А. Имитационная модель для оценки эффективности данных блока БКП узла АК и результаты моделирования.

Известия АН УзССР. Серия тех. наук. – Ташкент: (Деп.в ВИНИТИ № 8842-1386 от 24.12.86 – 23 с.).

51. Рахматкариев Э.У., Абидов А.А. Аналитическая модель совместного функционирования блока коммутации пакетов узла АК с блоком контроля и диагностики. Известия АН УзССР. Серия тех.наук. – Ташкент: (деп. в ВИНИТИ № 8043-1386 от 24.12.86.– 20 с).

52. Абидов А.А., Рахматкариев Э.У. Структура программного комплекса повышения надежности блока узла адаптивной коммутации. Алгоритмы. – Ташкент: РИСО АН УзССР,1986,вып.59. – С. 70-77.

53. Рахматкариев Э.У., Абидов А.А. Один из подходов к исследованию средств восстановления для ПО узла адаптивной коммутации. Методические и прикладные аспекты автоматизированного проектирования и управления в отраслях народного хозяйства III респ.конф. – Ташкент, 25-27 ноябрь,1987. – С.97.

54. Рахматкариев Э.У., Абидов А.А. Результаты исследования эффективности использования программных средств восстановления для ПО узла адаптивной коммутации. Двенадцатая Всесоюзная школа-семинар по вычислительным сетям. – Москва-Одесса: 1987, ч.1. С.223-227.

55. Рахматкариев Э.У., Абидов А.А. Программные средства контроля и восстановления структур данных узлов коммутатации. Тринадцатая Всесоюзная школа-семинар по вычислительным и сетям. – Москва-Алма-Ата: 1988, ч.1. С.359-364.

56. Самойленко С.И. Адаптивная коммутация в вычислительных сетях. 1978 (Научн.совет по компл.пробл ."Кибернетика" АН СССР). препр. – М.: ВИНИТИ. 1978.–78 с.

57. Самойленко С.И. Эффективность информационного обмена в сетях ЭВМ. Проблемы МСНТИ. МЦНТИ.– М.: 1981, № 2. С. 76-84.

58. Самойленко С.И. Протокол узла адаптивной коммутации. Препринт. – М.: ВИНИТИ, 1983.

59. Самойленко С.И. Оценка эффективности адаптивной коммутации в сетях интегрального обслуживания. Вопросы кибернетики. Проблемы теории вычислительных сетей / Под ред. Самойленко С.И, АН СССР. Научный совет по комплексной проблеме кибернетики, 1983.

60. Самойленко С.И., Давыдов А.А., Золотарев В.В., Третьякова Е.И. – Вычислительные сети (адаптивность, помехоустойчивость, надежность). – М.: Наука, 1983.– 277 с.

61. Самойленко С.И., Ващилин Э.П., Фомин С.С., Солонина Н.Б. Экспериментальная сеть адаптивной коммутации МИЭМ-АК. Девятая Всесоюзная школа-семинар по вычислительным сетям. В кн. Принципы построения, протоколы и реализация вычислительных сетей. – Москва– Пушкино: 1984, ч.2.1.

62. Самойленко С.И. Сети ЭВМ.– М.: 1986.

63. Срагович В.Г. Теория адаптивных систем. – М.: Наука, 1976. – 319 с.

64. Тейер Т., Липов М., Нельсон 3. Надежность программного обеспечения. – М.: Мир, 1981. – 325 с.

65. Тимофеев Б.Б., Ушаков В.А., Костенко В.С. Функциональные возможности пакета программ оперативного контроля и восстановления работоспособности систем реального времени.– Управляющие системы и машины. 1978, № 1. – С.17-25.

66. Феллер В. Введение в теорию вероятностей и ее приложение. В 2-х томах. Т.1. – М.: Мир, 1984. – 524 с.

67. Шарейко Л.А. Проблема эффективности вычислительных сетей и пути ее повышения. М.: АН СССР. – Научный совет по комплексной проблеме кибернетики, 1981. –72 с.

68. Шарейко Л.А., Белоус А.А. и др. Имитационная модель системы передачи информации и использование каналов коллективного пользования. Алгоритмы и программы, ВНЦТИ, 1983, № 3, Р00620І. –39 с.

69. Шарейко Л.Л. Оценка оптимальных технико-экономических

показателей систем передачи данных. – М.: Электросвязь, 1983, №9. – С. 45-48.

70. Шураков В.В. Надежность программного обеспечения систем обработки данных. – М.: Статистика,1981. – 216 с.

71. Якубайтис Э.А. Архитектура вычислительных сетей. – М.: Статистика, 1980. –279 с.

72. Якубайтис З.Я. Информационно-вычислительные сети. – М.: Финансы и статистика, 1984.

73. Anderson T., Kerr R. Recovery blocks in action a system supporting nigh reliabity. Inter Conf on software engineering, 2 nd. San-Fransicko, 1976, pp 447-457.

74. Blount Marlin L. Modeling of diagnosis in faultsoftly computer systems. The international conferens on fault–tolerant computing.1978, 21-23/Y, pros., pp.53-58.

75. Brand D., Joyner W.H. Verification of HDLC. IEEE trans.Commun.,1982, v30, №5, Part.2, pp. 1136-1142.

76. Dennng P.J. Fault tolerant operating systems. –Computing Surveys, 1976, v8, №4, pp.353-389.

77. Deswarte J., Lavictoire J. An intermittent faulure correctoin method. Paris, International sumposium un fault-tolerant. 5-th. 1975, pp.191-195.

78. Endres A., Glatthaar W. A complementary approach to program analysis and testing. –Lecture notes in computer science, 1978, v.65, pp. 380-401.

79. Fairly R.E. Tutorial: static analisis and dynamic testing of computer software, –Computer, 1978, v.11, №4, pp.14-23.

80. Holley C.Z., Reynolds K.Andit Concerns in online distributed computer network. “J.syst.Manag”, 1984, v.35, №6, pp. 32-36.

81. Hecht H. Fault-Tolerance software for real – time applications.– Computing surveys, 1976,v.8, №4, pp.391-407.

82. Howden W.E. Functional programm testing. –IEEE “Compac 78”. Computer software and applications conferensce 2id 1978. Proc., pp.321-325.

83. Kopets H. Systematic error treatment in real time software. Proc.JFAC world congrees 6th Boston-Combrige.1975. Proc. №4, 1975, pp.341/1-341/8.

84. Littlwood B. Theories of software reliability: how good are they and how can they be improved? –IEEE trans on software eng., v.5, №5,1980, pp.485-500.

85. Morgan D.E., Taylor D.J. A survey of methods of achiwing reliable sofware. "Computer", 1977, v.10, №2, pp.44-53.

86. Nelson E.C. Software reliability, Paris, Inter. Sump. ont fault-tolerent Computing, 5-th, Paris, 1975.

87. Pouzin L. Methods, Tools, and Observations on Flow Control in Packet-Switched Data Networks, "IEEE Trans. Commun.", 1981, Com–29, №4, pp.413-426.

88. Ramamoorthy C., Gansh S. Issnes in software reliability,–Sumposiumum reliability distributed sofware aude databass systems, Pittsburrgt, 1981,pp.113-116.

89. Ramamoorthy C., Ho S.F. testing large software with automatd softwqre evaluation systems.– IEEE transactions on software engineering, 1975, SE-1, №1, pp.45-58.

90. Randell B. System structure for software fault tolerance.– IEEE transactions on software Engineering, 1975, V.SE-1, №2, pp. 220-232.

91. Schneidewind N.F. An approach to software reliability prediction and guality control. – Proc. Of the 1972. Fall joint computer conf. montvall, N.Y.: AFIPS Press, 1972, pp.837-847.

92. Schick G.J. Walverton R.W. An analysis of competing software reliability models. –IEEE trans. on software eng. V.SE-4, N_o-2, march 1978.

93. Soi I.M., Gopal K. Some aspects of reliabl software packages. – "Microelectronics and reliability", 1979, v.19, №4, pp.376-386.

94. Soi I.M., Gopal K. Detection and diagnosis jf software makfunctions. – "Microelectronics and reliability", 1978, v.18, №14, pp.353-356.

95. Soi I.M. Aggarval K.K. Computer–cjvvunication network reliability: trends and issues. -"Microelectronics and reliability", 1981, v.21, №1, pp.75-79.

96. Shooman M.L., Trivedi A.K. A many–state Markov model for computer software performance parameters, IEEE Trans,Reliabl., R-25, pp. 66-68, 1976.

97. Shooman M.L., Trivedi A.K. A many-state Markov model for estimation and Predition of computer software performance parameters, Int. Con. on Reliable sofyware, 1975, Los Angelos, pp. 208-220.

98. Terrovoli G. Strument per il testing dei prodotti software. –Rivista di informatica, 1980, v.10, № 2, pp.145-176.

99. Wall J.F., Ferguson P.A. Progmatic software reliability prodiction. "Proc. 1977. Annuel Reliability on maintainability Sumposium", N.Y., 1977, pp.485-488.

100. Chen X., Hou W., Zhang Y. Reliability Evaluation of Embedded Real–time System based on Error Scenario. From the book Current Trends in Computer Science and Mechanical Automation Vol.2 Published by De Gruyter Open Poland 2022 https://doi.org/10.1515/ 9783110584998-056

101. IEEE (1990) 610.12–1990 – IEEE Standard Glossary of Software Engineering Terminology, pp 1-84

102. International Electrotechnical Commission (2010) Functional safety of electrical/electronic/ programmable electronic safety–related systems.

103. M. Sebastian, R. Ernst (2008) Modelling and Designing Reliable On-Chip-Communication Devices in MPSoCs with Real-Time Requirements. In: 13th IEEE International Conference on Emerging Technologies and Factory Automation, pp.1465-1472.

104. T. Anderson, J.C. Knight (1983) A Framework for Software Fault Tolerance in Real-Time Systems. IEEE Transactions on Software Engineering, SE-9(3): 355-364.

105. C. M. Krishna, A. D. Singh (1993) Reliability of Checkpointed real-time systems using time redundancy. IEEE Transactions on Reliability, 42(3): 427-435

106. A. Bums, S. Punnekkat, L. Strigini, D.R. Wright (1999) Probabilistic scheduling guarantees for fault-tolerant real-time systems. In: Dependable Computing for Critical Applications, pp.361-378

107. I. Broster, A. Burns, G. Rodriguez–Navas (2002) Probabilistic analysis of CAN with faults. In: 23rd IEEE Real-Time Systems Symposium, pp 269-278.

108. G. M. de A. Lima, A. Burns (2001) An Effective Schedulability Analysis for Fault-Tolerant Hard Real-Time Systems. In: 3th Euromicro Conference on Real-Time Systems, pp 209-216.

109. G. M. de A. Lima, A. Burns (2003) An Optimal Fixed-Priority Assignment Algorithm for Supporting Fault-Tolerant Hard Real-Time Systems. IEEE Transactions on Computers, 52(10): 1332-1346.

110. Li Jun, Yang Fumin, Lu Yansheng (2005) A Feasible Schedulability Analysis for Fault-Tolerant Hard Real-Time Systems. In: Proceeding of the 10th IEEE International Conference on Engineering of Complex Computer Systems, pp 176-183.

111. Li Jun, Yang Fumin, Tu Gang, Cao wanhua and Lu Yansheng (2007) Schedulability Analysis for Fault-Tolerant Hard Real-Time Tasks with Limited Priority Levels. In: The 4th International Conference on Autonomic and Trusted Computing, pp 529-538.

112. Li Jun, Yang Fumin and Lu Yansheng. (2007) Schedulability Analysis for Fault-Tolerant Hard Real-Time Tasks with Arbitrary Large Deadlines. In: the 6th International Symposium on Parallel and Distributed Computing, pp 149 – 156.

113. Z. Wu, L. Wang, G. Yang, Z. Zheng (2005) Schedulability Analysis For Fault-Tolerant Group-Based Preemptive Scheduling. Journal of Pervasive Computing and Communications, 1(3): 71-76.

114. M. Sebastian, R. Ernst (2009) Reliability Analysis of Single Bus Communication with Real-Time Requirements. In: 15th IEEE Pacific Rim International Symposium on Dependable Computing, Shanghai, pp.3-10.

115. S. Gui, L. Luo (2013) Reliability analysis of real-time fault-tolerant task models. Design Automation for Embedded Systems, 17(1): 87-107.

116. V. Izosimov, P. Pop, P. Eles, Zebo Peng (2005) Design optimization of time- and cost-constrained fault-tolerant distributed embedded systems. In: Proceedings of Design, Automation and Test in Europe, pp 864- 869.

117. Eles, V. Izosimov, P. Pop, Zebo Peng (2008) Synthesis of Fault-Tolerant Embedded Systems. In: Proceedings of the conference on design, automation and test in Europe, pp 1117–1122.

118. Б.А. Головкин, Многовариантное программирование и его применение. Автомат. и телемех, 1986, выпуск 7. С. 5-39.

119. Авиженис А. Отказоустойчивость – свойство, обеспечивающее постоянную работоспособность цифровых систем.–ТИИЭР, 1978, т. 66, № 10. С. 5–25.

120. Ихара Х., Фукуока К., Кубо Ю., Ёкота Ц. Отказоустойчивая вычислительная система с тремя симметричными вычислительными машинами.– ТИИЭР, 1978, т. 66, № 10. С. 68-89.

121. Абидов А.А. Статистика хизматида маълумотлар тузилмасини хавф-хатардан сақлаш моделини яратиш. «Raqamli iqtisodiyot va axborot texnologiyalari» elektron jurnali | 2021 йил, №2 (2).– 44-47 б.

122. Мультипроцессорные системы и параллельные вычисления/Под ред. Энслоу Ф.Г. – М.: Мир, 1976.

123. Avizienis A. Fault-tolerance and fault-intolerance: complementary approaches to reliable computing. - In: Proc. Int. Conf. Reliable Software. Los Angeles, 1975. N. Y. ACM, 1975, p. 458-464.

124. Avizienis A., Chen L. On the implementation of iV-version programming for software fault-tolerance during program execution.– In: Proc. 1977 COMPSAC. Int. Computer Software and Applications Conf. Chicago, 1977. p. 149–155.

125. Chen L., Avizienis A. Aversion programming: a fault-tolerance approach to reliability.

126. Elmendorf W. R. Fault-tolerant programming.–In: Int. Symp. Fault-Tolerant Computing, 1972, p. 79-83.

127. Fischler M. A., Firschein O., Drew D. L. Distinct software: an approach to reliable computing.–In: Proc. 2nd USA – Japan Computer Conf. Tokyo, 1975, p. 573–579.

128. Girard E., Rault J.-C. A programming technique for software reliability.–In: Proc. IEEE Symp. Computer Software Reliability, 1973. N. Y., 1973, p. 44-50.

129. Kopetz H. Software redundancy in real time systems. - In: Information Processing 74. Proc. IFIP Congr. 74, 1974, v. 2, p. 182-186.

130. Ramamoorthy C. V., Bastani F. B. Software reliability – status and perspectives. – IEEE Trans. Software Eng., 1982, v. SE-8, № 4, p. 354-371.

131. Yeh R. T. Guest Editorial-Computing Surveys, 1976, v. 8, № 3, p. 301-303. Also: Computing Surveys, 1976, v. 8, № 4, p. 355-357.

132. Christy Persya, T.R.Gopalakrishnan. Fault tolerant real time systems. International Conference on Managing Next Generation Software Application (MNGSA-08), Coimbatore, 2008.

133. Viacheslav izosimov. Scheduling and Optimization of Fault-Tolerant embedded Systems, Ph.D.Thesis,Linkopings university, November 2006.

134. Akash Kumar,"Scheduling for Fault-Tolerant Distributed Embedded Systems",IEEE Computer 2008.

135. Autopilot http://en.wikipedia.org/wiki/Autopilot,2007.

136. I. Eusgeld, F.C. Freiling, and R. Reussner (Eds.): Software Reliability. Dependability Metrics, LNCS 4909, pp. 104-125, 2008. –c Springer-Verlag Berlin Heidelberg 2008 Knight и Leveson.

137. Москат Н.А., Станкевич Е.А. Показатели качества информационно-вычислительных систем железнодорожного транспорта // Инженерный вестник Дона, 2013, №3. URL: ivdon.ru/ru/magazine/archive/n3y2013/1789.

138. Москат Н.А. Вероятностный анализ своевременности предоставления информации в автоматизированных системах управления

железнодорожным транспортом. [Текст] // Вестник Ростовского государственного университета путей сообщения. – Ростов н/Д, № 4, 2008. – С. 87-94.

139. Ильенкова, С.Д. Управление качеством. Учебник / С. Д. Ильенкова, Н.Д. Ильенкова, С.Ю. Ягудин и др.; Под ред. Доктора экономических наук, профессора Ильенковой С.Д. – М.: ЮНИТИ, 1998.–198 с.

140. Бокова Ф.М. Исследование эффективности и качества банковских услуг [Электронный ресурс] // «Инженерный вестник Дона», 2011, №1. – Режим доступа: http://ivdon.ru/magazine/archive/n1y2011/388 (доступ свободный) – Загл. с экрана. – Яз. рус.

141. Попова, Т.Д. Организационно-экономические условия и критерии анализа затрат на качество услуг и продукции [Электронный ресурс] // «Инженерный вестник Дона», 2012, №1. – Режим доступа: http://ivdon.ru/magazine/archive/n1y2012/814 (доступ свободный) – Загл. с экрана. – Яз. рус.

142. Исаев, Г. Н. Управление качеством информационных систем. Теоретико-методологические основания. [Текст] / Г.Н. Исаев. – М.: Наука, 2011. – 280 с.

143. M.J. Taylor, J.L. DaCosta, «Soft Issues in IS Projects: Lessons from an SME Case Study». Systems Research and Behavioral Science, vol. 16, No. 3, May-June 1999, 247-254.

144. I. Tervonen, P. Kerola, «Towards deeper co-understanding of software quality», Information and Software Technology, vol. 39, No 14-15 (1999).

145. Lee, Y. W., Strong, D. M., Kahn, B. K. & Wang, R. Y. (2002). «AIMQ: A Methodology for Information Quality Assessment», Information & Management 40, 133-146.

146. Michnik, J. & Lo, M.-C. (2009). «The Assessment of the Information Quality with the Aid of Multiple Criteria Analysis», European Journal of Operational Research 195, 850-856.

147. Mónica Bobrowski, Martina Marré, and Daniel Yankelevich. A homogeneous framework to measure data quality. In Proceedings of the International Conference on Information Quality (IQ), pages 115-124, Cambridge, MA, 1999.

148. Богатырев В.А., Богатырев А.В. Функциональная надежность систем реального времени. Научно-технический вестник информационных технологий, механики и оптики, 2013, № 4 (86). – С.151-152.

149. Богатырев В.А. Exchange of Duplicated Computing Complexes in Fault tolerant Systems // Automatic Control and Computer Sciences. – 2011. – V. 46. – № 5. – P. 268-276.

Абдужаббор Абдухамидович Абидов

ФУНКЦИОНАЛЬНЫЕ АСПЕКТЫ АДАПТАЦИИ, МОДЕЛИРОВАНИЯ И АЛГОРИТМИЗАЦИИ НАДЕЖНОГО ФУНКЦИОНИРОВАНИЯ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Монография

Редактор: Мадумарова Г.Э.
Корректор: Султанова Д.Х.
Технический редактор: Камилова Д.Д.

Подписано в печать 17.01.2021 г. Размер бумаги 80х60 1/16.
Условный печатный лист 10,3. Тираж 50 штук.
Заказ №
Выпущено в типографии ТГЭУ.
100003, г. Ташкент, проспект Ислама Каримова, 49.