УДК 681.3

А.А. Стецко

СТРУКТУРНО-ФУНКЦИОНАЛЬНОЕ ОПИСАНИЕ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ И МОДЕЛИРОВАНИЯ ВЫЧИСЛИТЕЛЬНОЙ СЕТИ НА ПРИКЛАДНОМ УРОВНЕ

Стецко Александр Алексеевич, *доктор технических наук, окончил Саратовский политехнический институт по специальности «Электроснабжение промышленных предприятий и городов». Заместитель главного инженера по производству ФНПЦ ОАО «НПО «Марс». Имеет монографию, статьи, изобретения в области электроники, электротехники и радиотехники. [e-mail: mars@mv.ru].*

Аннотация

Описаны имитационные модели «тонких» и «толстых» клиентов вычислительной сети (ВС), имитационные модели серверов. Приведены результаты моделирования сервера в системе GPSS. Представлены функциональные модели клиентских станций серверов на основе сетей Петри, формализованное описание процессов проектирования на основе DFD-диаграмм. Предложен алгоритм байесовской оптимизации вычислительной сети на прикладном уровне.

Ключевые слова: САПР, сеть Петри, вычислительная сеть, байесовские сети доверия, имитационная модель.

Abstract

The article describes simulation models of thin and thick clients of computer network, simulation models of servers. It gives an account of results of server modeling in the GPSS system and cites operational models of client server stations on basis of Petri nets, formalized description of design processes on basis of DFD-diagrams. The article suggests an algorithm of Bayesian computer-network optimization at application level.

Key words: CAD system, Petri net, computer network, Bayesian trust net, simulation model.

1 Имитационные модели «толстых» и «тонких» клиентов ВС

Формализованная модель «тонкого» клиента

«Тонкий» клиент реализует только презентационную логику – прикладной интерфейс для пользователя. Модель взаимодействия такого клиента с конечным пользователем и сервером представлена на рисунке 1.

На диаграмме видно, что основной функцией для «тонкого» клиента является генерация запроса, а, следовательно, основной характеристикой – частота генерации запроса.

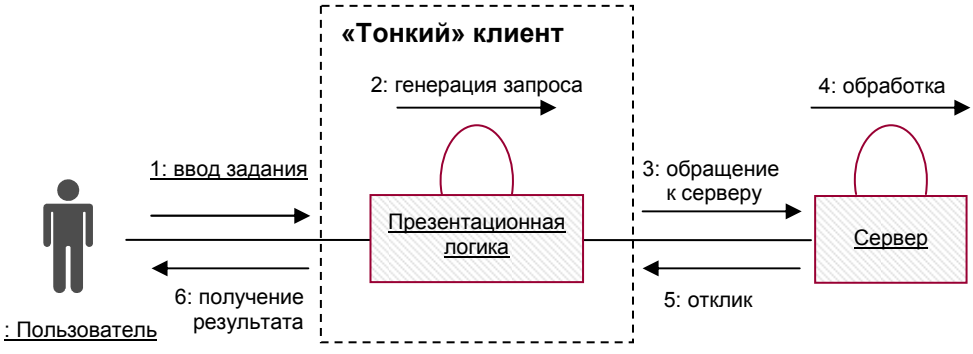

Рис. 1. Модель «тонкого» клиента

Формализованная модель «толстого» клиента

«Толстый» клиент объединяет в себе презентационную логику и логику выполнения и представляет собой обычную архитектуру рабочей станции. Модель взаимодействия компонентов «толстого» клиента с конечным пользователем и сервером приведена на рисунке 2.

Основными функциями «толстого» клиента являются генерация, анализ и обработка запроса. Основные характеристики: частота генерации запроса, время обработки запроса, занятость (выходная переменная).

Свойство «занятость» является характеристикой компоненты логики выполнения, поэтому в модели «тонкого»

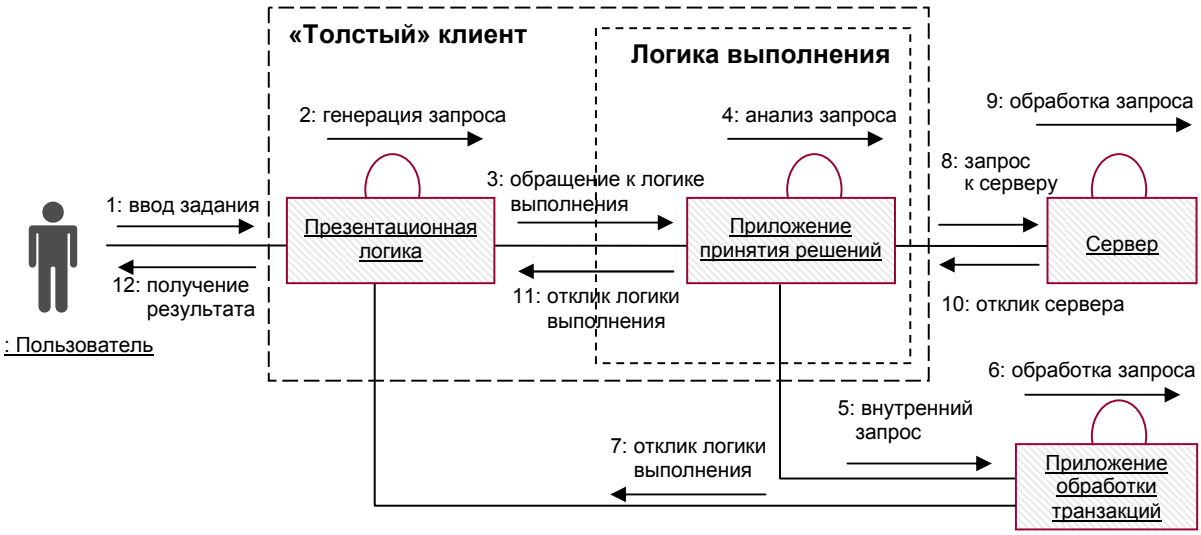

Рис. 2. Модель «толстого» клиента

клиента данное свойство не рассматривается. Данная переменная показывает загруженность объекта («толстого» клиента, сервера) при существующем потоке запросов (наличии определенных производственных процессов) и запас производственных процессов в сети.

2 Имитационные модели файлового сервера, сервера данных и proxy-сервера

2.1 Обобщенная модель серверов

Любой сервер характеризуется типом ресурса, к которому он предоставляет доступ. В общем виде сценарий основного варианта использования можно представить в виде диаграммы сотрудничества (рис. 3).

Основные функции сервера: анализ запроса, обработка отклика запрашиваемого ресурса и занятие ресурса (обработка запроса). Основные характеристики: время отклика ресурса, время обработки запроса и занятость.

Время отклика ресурса зависит от вида сервера (ftp-сервер проверяет логин, proxy-сервер – информацию в кэш-памяти или в сети). Время обработки запроса зависит от типа запрашиваемого ресурса, т.е. от вида сервера.

В реальных системах оперативной обработки данных используется квантовая технология. Для обслуживания отдельной заявки отводится постоянный квант времени q, достаточный для выполнения нескольких тысяч операций. Если работа была выполнена за время q, она покидает систему. В противном случае она вновь поступает в конец очереди и ожидает предоставления ей очередного кванта процессорного времени.

Квантовая обработка данных будет осуществляться и на «толстом» клиенте, т.к. это относится к элементу концептуального уровня вычислительной сети — логике выполнения (или бизнес-логике, как в некоторых источниках литературы). «Толстый» клиент будет осуществлять обработку внутренних запросов, а также обработку ответов сервера на заданные запросы.

2.2 Событийное моделирование

Рассмотрим само требование, т.е. запрос. Он характеризируется следующими свойствами:

- новый запрос или повторный (флаг);
- категория запроса (формальный или неформальный);
- объем запроса, высчитывается по категории;
- только для «толстого» клиента – параметр, хранящий номер отославшего данный запрос клиента.

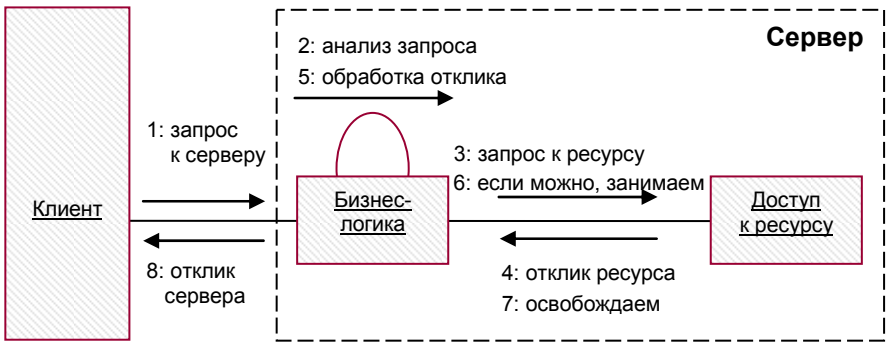

Рис. 3. Модель сервера

«Тонкий» клиент

Для облегчения построения имитационной модели изобразим графически основной процесс функционирования модели «тонкого» клиента (рис. 4).

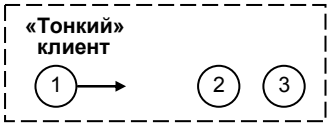

Рис. 4. Функционирование «тонкого» клиента

В данном случае все происходящие события следующие:

1. Генерирование запроса — требования (транзакта);
2. Определение категории запроса;
3. Определение параметров требования.

Сервер

Теперь рассмотрим более сложную структуру работы сервера (рис. 5).

1. Обращение к серверу (проверка занятости сервера);
2. Занятие места оперативной памяти (многоканального устройства);
3. Вход требований в очередь (ресурс — процессор);
4. Проверка занятости ресурса (доступ к ресурсу);
5. Выход требования из очереди;
6. Принятие решения — в зависимости от вида сервера будет разный характер поведения требования;
7. Обслуживание требования — запроса;
8. Освобождение канала обслуживания — процессора сервера;
9. Выход требования из оперативной памяти;
10. Выход требований из системы (в модели «тонкого» клиента, в модели «толстого» клиента происходит обращение к клиенту с ответом).

В модели используются элементы: требование, очередь и канал обслуживания.

С события № 6 развиваются разные сценарии работы серверов. Затем все сходятся в дальнейших пунктах.

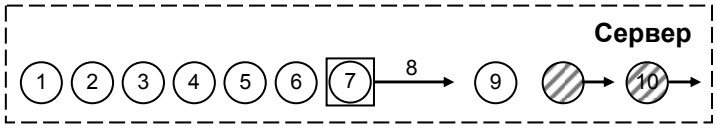

Рис. 5. Модель работы сервера

Сервер приложений и ftp-сервер

Сервер приложений и ftp-сервер аналогично будут обрабатывать запрос, отличаться будут логика доступа к ресурсу и определение быстродействия сервера, следовательно, по первому отличию задержка в принятии решения у ftp-сервера будет выше, по второму — быстродействие будет определяться у сервера приложений быстродействием процессора, а ftp-сервера — скоростью жесткого диска. Данные параметры будут задаваться пользователем.

При работе данных серверов используется квантовая обработка заданий. Для обслуживания отдельной заявки отводится постоянный квант времени q, достаточный для выполнения нескольких тысяч операций. Если работа была выполнена за время q, она покидает систему. В противном случае она вновь поступает в конец очереди и ожидает предоставления ей очередного кванта процессорного времени.

На рисунке 6 представлена схема обработки запроса сервером приложений и ftp-сервером.

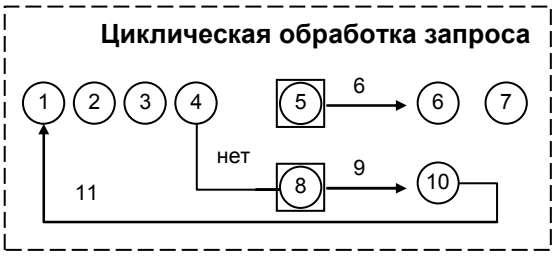

Рис. 6. Обработка запроса сервером

Возможны следующие события:

1. Обращение к ресурсу (вход требования в очередь);
2. Проверка занятости ресурса (канала обслуживания — процессора);
3. Выход требования из очереди;
4. Запрос меньше или равен отведенному кванту времени;
5. Да, обслуживание требования полностью;
6. Освобождение канала обслуживания;
7. Освобождение памяти на обслуженный запрос;
8. Нет, обработка кванта времени;
9. Освобождение канала обслуживания;
10. Освобождение памяти на объем, определенный квантом обработки;
11. Переход в конец очереди.

Событие № 5 и событие № 8 — обслуживание требования, создают задержку на определенное время. Это время рассчитывается с помощью нормального распределения, но среднее значение высчитывается для серверов по-разному.

Скорость обрабатываемой информации ftp-сервером определяется скоростью дисковой подсистемы.

Среднее значение обработки = объем запроса/скорость диска.

Реакция сервера приложений зависит от быстродействия процессора. Среднее значение обработки = объем запроса/быстродействие процессора.

Прокси-сервер

У прокси-сервера не выделяем циклическую обработку запроса, поскольку задержка на принятие решения слишком мала, а обрабатываемые запросы самим сервером будут содержаться в кэш-памяти (малым объемом), следовательно, отклик будет в несколько раз быстрее.

Среднее значение обработки в кэш-памяти = (объем запроса/быстродействие процессора/коэффициент).

Данная переменная распределяется по экспоненциальному закону, поскольку за день информация в кэш-памяти будет накапливаться до определенного порога, затем неиспользуемые данные будут удаляться из кэш-памяти.

Прокси-сервер может отослать запрос к внешней сети или внутренней, где логика доступа к ресурсу и обработка запроса будут разными.

Схематично работа прокси-сервера представлена на рисунке 7.

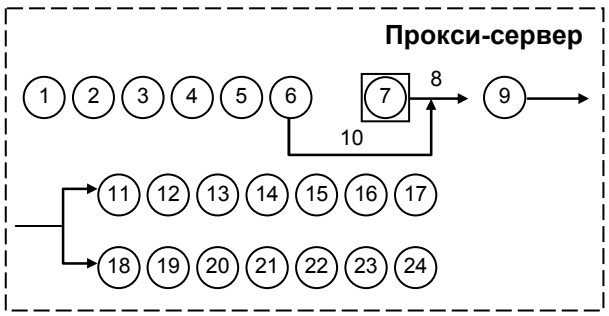

Рис. 7. Работа прокси-сервера

Возможны следующие события:

1. Обращение к серверу (проверка занятости сервера);
2. Занятие места оперативной памяти (многоканального устройства);
3. Вход требований в очередь (ресурс — процессор);
4. Проверка занятости ресурса (доступ к ресурсу);
5. Выход требования из очереди;
6. Принятие решения — есть ли запрос в кэш-памяти;
7. Да, обслуживание требования — запроса;
8. Освобождение канала обслуживания — процессора сервера;
9. Выход требования из оперативной памяти;
10. Нет, запрос в сети, переход на событие № 8;
11. Запрос во внутренней сети;
12. Да, вход требований в очередь (ресурс — устройство);
13. Проверка занятости ресурса (доступ к ресурсу), ресурс — одноканальное устройство обслуживания;
14. Выход требования из очереди;
15. Обслуживание требования — запроса;
16. Освобождение канала обслуживания — устройства;
17. Формирование отклика устройства;
18. Определение потерянных запросов во внешней сети;
19. Вход требований в очередь (ресурс — внешняя сеть);
20. Проверка занятости ресурса (доступ к ресурсу), ресурс — многоканальное устройство обслуживания;
21. Выход требования из очереди;
22. Обслуживание требования — запроса;
23. Освобождение канала обслуживания — устройства;
24. Формирование отклика сети.

Формирование отклика устройства можно определить как удаление запроса из вычислительной сети, например, печать с принтера.

«Толстый» клиент

В модели «толстого» клиента рассмотрим основные события работы клиента, т.к. последовательность событий, описывающая логику обработки данных (на основе квантования), описана выше при рассмотрении работы сервера. Событийная схема представлена на рисунке 8.

1. Генерирование запроса — требования;
2. Определение категории запроса;
3. Определение параметров требования;
4. Обращение к клиенту (проверка наличия оперативной памяти);
5. Занятие места оперативной памяти (многоканального устройства);
6. Вход требований в очередь (ресурс — процессор);
7. Проверка занятости ресурса (доступ к ресурсу);
8. Выход требования из очереди;
9. Принятие решения: требование — ответ с сервера;
10. Да, обслуживание запроса (циклической обработкой с квантованием времени);
11. Освобождение канала обслуживания — процессора сервера;
12. Выход требования из оперативной памяти;
13. Удаление требования;
14. Принятие решения: требование — отказ с сервера;
15. Да, определение вероятности повторного запроса;
16. Переход на освобождение ресурса;
17. Фиксирование повторного запроса с задержкой времени;

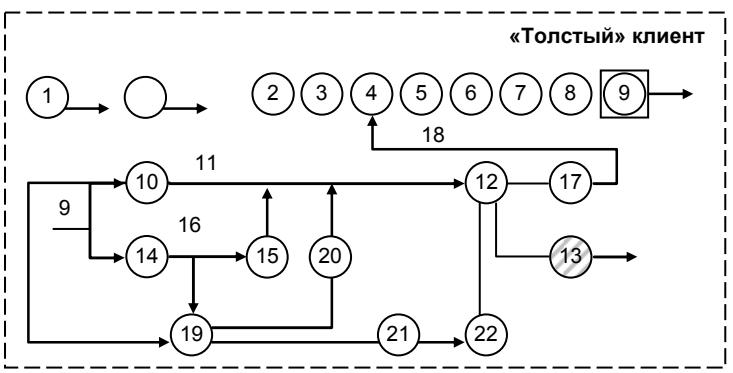

Рис. 8. Событийная схема работы «толстого» клиента

18. Переход в захват оперативной памяти;
19. Принятие решения: первая попытка к клиенту или серверу;
20. Да, к клиенту — переход на циклическую обработку;
21. Нет, к серверу — переход на освобождение ресурсов клиента;
22. Обращение к серверу.

2.3 Результаты моделирования работы серверов в системе GPSS

Модель «тонкий клиент»

Моделировалась ВС, содержащая в своем составе один прокси-сервер и 20 "тонких" клиентов. Результаты моделирования, соответствующие пяти экспериментам, сведены в таблицу 1, в которой представлены только основные характеристики модели.

Ниже представлены условия экспериментов и результирующие рекомендации.

средним значением обработки запроса в кэш-памяти, во внутренней и внешней сетях.

Во внешней сети время обработки самое высокое, поскольку доступ к устройству и тем более к кэш-памяти намного быстрее.

Эксперимент № 2 характеризуется увеличением нагрузки на внешнюю сеть, уменьшением мощностей сервера (оперативной памяти и быстродействия). Однако занятость сервера не сильно повысилась из-за увеличения времени между возникновением запросов.

В эксперименте № 3 запросы очень часто возникают из-за большого объема информации, даже достаточно мощный сервер не справляется с их потоком. Как следствие — очень большой коэффициент повторных запросов, перегрузка процессора и оперативной памяти, доля информации, находящейся в кэш-памяти, мала, информация ищется больше по внешней сети, поэтому внутренняя сеть свободна. Но внешняя

Таблица 1

Результаты моделирования системы прокси-сервер + 20 «тонких» клиентов

Переменные	Эксперименты				
	1	2	3	4	5
Входящие					
Частота генерации запроса (с)	240	360	24	60	890
Процент повторного запроса (%)	85	80	90	80	100
Средний объем запроса категории 1 (байт)	50	20	50	100	50
Средний объем запроса категории 2 (байт)	500	300	500	400	200
Вероятность выбора категории 1 запроса (%)	35	40	25	65	50
Быстродействие процессора (байт/с)	600	200	1500	1500	600
Оперативная память (байт)	131 072	3000	131 072	131 072	32 768
Вероятность нахождения в кэш-памяти, среднее значение (%)	60	50	20	40	60
Вероятность нахождения во внутренней сети (%)	60	30	10	65	40
Выходящие					
Занятость процессора сервера	0.577	0.556	0.999	0.876	0.234
Среднее время ожидания в очереди к процессору	1.090	0.650	1.979	1.85	0
Занятость операционной памяти	0.001	0.055	0.953	0.446	0.002
Занятость устройства внутренней сети	0.328	0.332	0.146	0.926	0.172
Среднее время ожидания в очереди к устройству	2.241	6.42	15.260	8.42	0.112
Занятость внешней сети	0.102	0.259	0.247	0.196	0.102
Процент запросов категории 1	0.624	0.568	0.154	0.654	0.500
Процент повторов	0	0	4.190	0	0

Эксперимент № 1 характеризует ситуацию с небольшой нагрузкой на сеть. Редкая генерация запросов, небольшие объемы запрашиваемой информации, хорошие характеристики сервера приводят к небольшой загруженности оперативной памяти и средней загруженности процессора; загруженность внутреннего устройства получилась выше, т.к. вероятность попадания на него больше, чем во внешнюю сеть.

Обработка запроса генерируется по нормальному распределению с рассчитываемым

сеть тоже свободна, так как сервер не успевает пересылать к ней запросы. Необходимо увеличение мощности компьютера.

Эксперимент № 4 характеризуется высокой частотой генерации запросов, но преимущественно небольшого объема. Поэтому у оперативной памяти запас есть, а у процессора — нет. При этом большой коэффициент занятости у устройства во внутренней сети, поскольку устройство одноканальное и справляется с данным потоком запросов с трудом.

Эксперимент № 5 характеризуется очень редким возникновением запросов среднего объема, не очень мощным сервером, свободным состоянием внутреннего устройства и внешней сети. У такой сети большой запас в дополнительных производственных процессах.

Следующие условия моделирования предполагали наличие сервера приложений и 20 «тонких» клиентов. Поскольку сервер приложений будет отличаться от ftp-сервера только скоростью доступа к ресурсу и скоростью обработки запроса — характеристиками, которые устанавливаются экспериментатором, то вычислительные эксперименты проводятся с использованием характеристик одного из них, а именно сервера приложений.

Результаты моделирования представлены в таблице 2.

Эксперимент № 1 характеризуется средней загруженностью сети: при средних значениях входных параметров получаем средние выходные параметры. Рассмотрим циклическую обработку запросов с выделением кванта времени (рис. 9).

На графике изображены оставшиеся части запроса для обработки. Из графика видно, что сервер обрабатывает отведенным квантом времени по циклу с законом LIFO «первый пришел — первый ушел».

В эксперименте № 2 наблюдается средний запас производственных процессов по сети, в эксперименте № 5 большой запас, что определяется редкой генерацией запросов среднего объема, мощным сервером.

Эксперименты № 3, № 4 характеризуют загруженность сети: эксперимент № 3 — загруженность оперативной памяти, так как запросы большого объема поступают, а объем памяти небольшой; эксперимент № 4 — загруженность процессора сервера, не справляющегося с частым потоком запросов.

Таблица 2

Результаты моделирования системы "Сервер приложений + 20 «тонких» клиентов"

Переменные	Эксперименты				
	1	2	3	4	5
Входящие					
Частота генерации запроса (с)	240	360	24	60	890
Процент повторного запроса (%)	85	80	90	80	100
Средний объем запроса категории 1 (байт)	50	20	50	100	50
Средний объем запроса категории 2 (байт)	500	300	500	400	200
Вероятность выбора категории 1 запроса (%)	35	60	25	65	50
Быстродействие процессора (байт/с)	200	300	100	200	600
Оперативная память (байт)	131 072	3000	3000	31 072	32 768
Квант времени обработки (с)	0.5	0.2	0.6	0.5	0.3
Выходящие					
Занятость процессора сервера	0.700	0.412	0.600	0.996	0.047
Среднее время ожидания в очереди к процессору	0.547	0.285	32.600	2.63	0
Занятость оперативной памяти	0.006	0.021	0.993	0.378	0.000
Процент запросов категории 1	0.343	0.650	0.141	0.637	0.634
Процент повторов	0	0	0.198	0	0

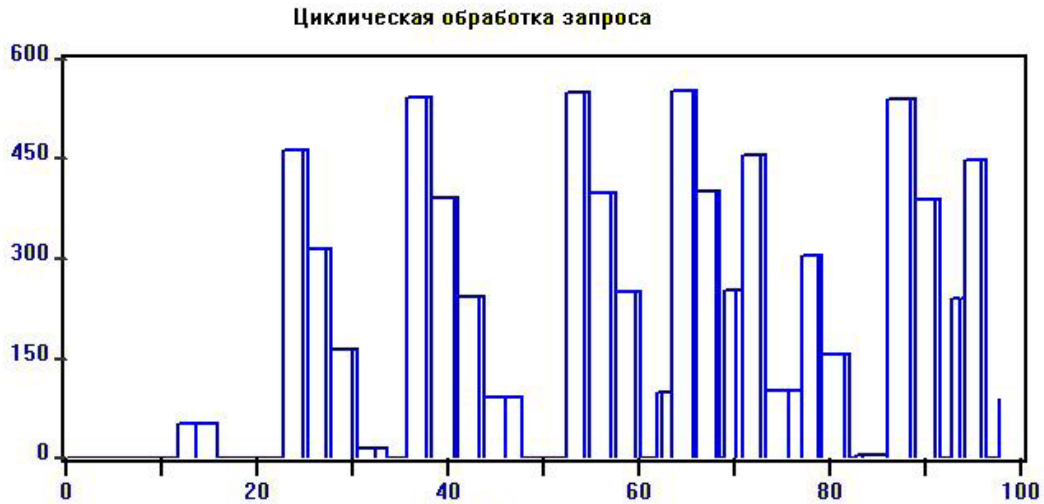

Рис. 9. Обработка запросов с выделением кванта времени

Модель «толстый» клиент

Следующие условия моделирования предполагают наличие одного ftp-сервера и двух «толстых» клиентов. Результаты моделирования сведены в таблицу 3.

Из проведенных экспериментов можно сделать следующие выводы. Основные факторы, больше всего влияющие на выходные характеристики, — это частота генерации запросов, объем запрашиваемой информации, а также характеристики аппаратного обеспечения (оперативная память и быстродействие). На распределение производственных процессов по сети влияет вероятность обращения клиентов к серверу. Объем данных зависит от преобладания категории запроса 1 (малый объем) или 2 (большой объем).

При достаточно частой генерации запросов клиенты, имеющие хорошее аппаратное обеспечение компьютеров, справляются со своим потоком запросов, если же к серверу запросов по сети идет меньше 50%, то сервер простаивает.

С другой стороны, если оба клиента ориентированы в основном на работу с сервером, то сервер получается основным нагруженным элементом в сети, а клиентские компьютеры простаивают, только обрабатывают ответы сервера; если же обработку переместить на сервер, то клиентские приложения будут бездействовать. Возможно, в этом случае необходимо заменить «толстые» клиенты на «тонкие» и промоделировать их работу в тех же условиях.

В основном эксперименты показали достаточно большой запас производственных процессов сервера, поскольку взаимодействуют с сервером всего два клиента. Если увеличить число посылающих запросы клиентов, то и нагрузка на сервер увеличится.

Работа сервера приложений и прокси-сервера рассмотрена при вычислительных экспериментах на «тонком» клиенте. Выделенные закономерности, алгоритмы функционирования сохранятся и при работе с «толстыми» клиентами, поскольку работу сервера не затрагивает ничего, кроме входящих потоков запросов, следовательно, как ведут себя остальные элементы, не влияет на функциональность сервера.

3 Функциональные модели клиентских станций и серверов на основе сетей Петри

Сеть Петри — это инструмент для моделирования динамических систем [1,2]. Данный инструмент позволяет моделировать систему математическим представлением ее в виде сети Петри, проанализировав которую можно получить информацию о структуре и динамическом поведении моделируемой системы.

Информационную систему предприятия можно представить как соединенные в одно целое через компьютерную сеть технические средства с функционирующими производственными приложениями. Производственные приложения (программные продукты, автоматизированные системы) позволяют автоматизировать различные производственные процессы организации и являются генераторами трафика.

На общее быстродействие информационной системы в целом влияет небольшое количество параметров конечных устройств и линий связи. Например, для сервера — это быстродействие процессора, дисковой подсистемы, размер оперативной памяти. Каждый тип устройств, линий связи может быть смоделирован как система массового обслуживания с помощью «раскрашенных» сетей Петри. Каждое приложение может быть представлено как генератор пакетов, обрабатываемых системой массового обслуживания.

Пакет, генерируемый приложением (например, CAD под операционной системой Linux) по определенному расписанию, характеризуется следующими параметрами:

- размер запроса, разброс запроса, частота запроса;
- размер ответа, разброс ответа;
- коэффициенты загрузки у клиента оперативной памяти, процессора, жесткого диска, видеопамяти.

Для сервера приложения характеризуются следующими параметрами:

- тип сервера: FTP, приложений;
- базовая операционная система: Windows Server 2003, Linux;
- коэффициент загрузки памяти, жесткого диска, процессора, видеопамяти.

В реальной ВС на одном аппаратном сервере могут функционировать несколько программных серверов (например, ftp-сервер и web-сервер), а рабочая станция пользуется услугами, предоставляемыми разными автоматизированными системами.

В модуле САПР ВС, работающем на основе «раскрашенных» сетей Петри, для каждого типа вычислительной техники создается отдельная модель и заносится в библиотеку. Сетевой трафик представлен в виде фреймов, которые генерируются клиентами сетевых автоматизированных систем по нормальному закону распределения.

В сети Петри используются 3 простых цвета: mac, data, load и один составной — frame.

Фрейм (frame) содержит:

- адрес источника, адрес-получателя (mac);
- данные запроса, данные ответа (data);
- коэффициенты загрузки запросом процессора, оперативной памяти, жесткого диска, видеопамяти (load);
- коэффициенты загрузки ответом процессора, оперативной памяти, жесткого диска, видеопамяти (load).

В модели возможно реализовать достаточно сложную логику обработки внутри рабочей

Таблица 3

Результаты моделирования системы «ftp-сервер + 2 «толстых» клиента»

Переменные	Эксперименты				
	1	2	3	4	5
Входящие					
Клиент 1					
Частота генерации запроса (с)	24	2	4	60	8
Процент повторного запроса (%)	85	80	100	80	100
Средний объем запроса категории 1 (байт)	50	20	100	100	50
Средний объем запроса категории 2 (байт)	500	300	1000	400	200
Вероятность выбора категории 1 запроса (%)	55	40	10	65	50
Быстродействие процессора (байт/с)	300	200	100	300	200
Оперативная память (байт)	13102	3000	13072	3268	32608
Вероятность обращения к серверу (%)	70	20	80	40	90
Клиент 2					
Частота генерации запроса (с)	24	3	4	6	8
Процент повторного запроса (%)	75	80	100	30	90
Средний объем запроса категории 1 (байт)	30	60	100	40	50
Средний объем запроса категории 2 (байт)	300	700	900	300	800
Вероятность выбора категории 1 запроса (%)	55	30	10	25	40
Быстродействие процессора (байт/с)	300	200	100	300	200
Оперативная память (байт)	32768	13072	13072	3268	13720
Вероятность обращения к серверу (%)	30	30	80	70	50
Сервер					
Быстродействие (байт/с) — скорость винта	80	40	80	80	90
Оперативная память (байт)	32768	13072	3000	3268	13072
Квант времени (с)	0.5	0.4	0.2	0.4	0.5
Выходящие					
Клиент 1					
Занятость оперативной памяти	0.621	0.991	0.931	0.839	0.165
Занятость процессора	0.048	0.019	0.430	0.009	0.019
Среднее время ожидания в очереди к процессору	0.000	0.140	124	0.0	0.247
Клиент 2					
Занятость оперативной памяти	0.449	0.972	0.931	0.896	0.119
Занятость процессора	0.083	0.009	0.450	0.007	0.006
Среднее время ожидания в очереди к процессору	0.039	0.050	132	0.0	0.145
Сервер					
Занятость оперативной памяти	0.001	0.000	0.575	0.023	0.300
Занятость процессора	0.083	0.043	0.299	0.110	0.280
Среднее время ожидания в очереди к процессору	0.030	0.450	67	30	0.161

станции с задержками на переходах, ограничениями по скорости обработки при переполнении буферов.

Реализованная схема позволяет учитывать различные варианты устройств (например, рабочие станции с несколькими сетевыми картами, несколькими процессорами, коммутатор с различным количеством портов), дополнять библиотеку вновь появляющимися типами устройств.

Для вычислительных систем (сервер, рабочая станция) создана библиотека, позволяющая для каждого типа систем сохранять следующие параметры: производительность процессора, жесткого диска, оперативной памяти, видеопамяти, размер оперативной памяти. Заполнение параметров осуществляется на основе специализированной тестирующей программы SiSoft Sandra.

Полученные в процессе моделирования результаты отображаются на графике (выборочно), «выгружаются» в протокол. По пиковым значениям на графиках определяются наиболее загруженные узлы ВС и линии связи, требующие модернизации. Протокол экспериментов успешно импортируется в табличные редакторы (такие как MS Excel), где возможно провести более детальный анализ результатов.

4 Формализованное описание процессов проектирования на основе DFD-диаграмм

В соответствии с DFD-методологией (Data Flow Diagram) модель системы определяется как иерархия диаграмм потоков данных, описывающих процессы преобразования информации от момента ее ввода в систему до выдачи конечному пользователю [1,2]. Диаграммы верхних уровней иерархии — контекстные диаграммы — задают границы модели, определяя ее окружение (внешние входы и выходы) и основные рассматриваемые процессы. Контекстные диаграммы детализируются при помощи диаграмм следующих уровней.

Основными элементами диаграмм потоков данных являются:

- внешние сущности;
- процессы;
- накопители данных;
- потоки данных.

Внешние сущности

Под внешней сущностью (External Entity) понимается материальный объект, являющийся источником или приемником информации. В качестве внешней сущности на DFD-диаграмме могут выступать заказчики, поставщики, клиенты, склад, банк и другие. К сожалению, DFD-методология не оформлена как стандарт. По этой причине в диаграммах потоков данных используются различные условные обозначения. На рисунке 10 показаны символы внешних сущностей, используемые в нотациях «Yourdon and Coad Process Notation» и «Gane and Sarson Process Notation».

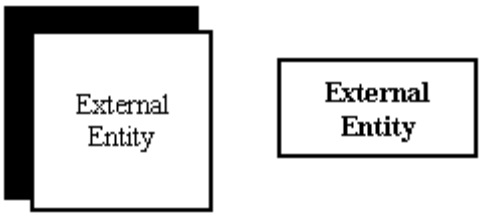

Рис. 10. Символы внешних сущностей

Определение некоторого объекта в качестве внешней сущности указывает на то, что он находится за пределами границ анализируемой информационной системы.

Процессы

Процессы представляют собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. В реальной жизни процесс может выполняться некоторым подразделением организации, выполняющим обработку входных документов и выпуск отчетов; отдельным сотрудником; программой, установленной на компьютере; специальным логическим устройством и тому подобное.

Процессы на диаграмме потоков данных изображаются, как показано на рисунке 11.

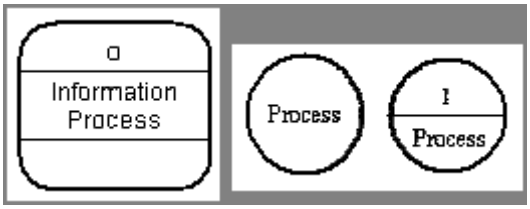

Рис. 11. Символы процессов

Номер процесса служит для его идентификации. В поле имени вводится наименование процесса в виде предложения с глаголом в неопределенной форме (вычислить, рассчитать, проверить, определить, создать, получить) и поясняющими существительными.

Информация в нижнем поле символа процесса указывает какое подразделение организации, сотрудник, программа или аппаратное устройство выполняет данный процесс. Если такое поле отсутствует, то подобная информация может быть указана в текстовом примечании.

Накопители данных

Накопители данных предназначены для изображения неких абстрактных устройств для хранения информации, которую можно туда в любой момент времени поместить или извлечь, безотносительно к их конкретной физической реализации. Накопители данных являются неким прообразом базы данных информационной системы организации. Наиболее часто употребляемые символы для их обозначения показаны на рисунке 12.

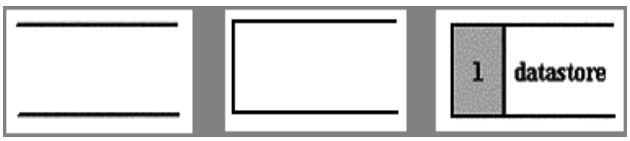

Рис. 12. Символы накопителей данных

Внутри символа указывается его уникальное в рамках данной модели имя, наиболее точно, с точки зрения аналитика, отражающее информационную сущность содержимого, например, «Поставщики», «Заказчики», «Счета-фактуры», «Накладные». Символы накопителей данных в качестве дополнительных элементов идентификации могут содержать порядковые номера.

Потоки данных

Поток данных определяет информацию, передаваемую через некоторое соединение (кабель, почтовая связь, курьер) от источника к приемнику. На DFD-диаграммах потоки данных изображаются линиями со стрелками, показывающими их направление. Каждому потоку данных присваивается имя, отражающее его содержание.

5 Модификация DFD-диаграмм для задач АП. Включение в формализм расписания задач

ВС представляет собой набор связанных генераторов трафика, где по связям двигаются потоки данных. В этой связи наиболее удобным средством моделирования представляется язык DFD.

Основным источником сетевого трафика являются процессы, происходящие на узле сети. При составлении функциональной модели сети следует обозначить все эти процессы и потоки данных, генерируемые ими. Кроме того, каждый из процессов имеет свое расписание выполнения в течение дня и его необходимо учитывать при моделировании.

В связи с этим к части языка DFD, описывающей сущность «процесс», были сделаны дополнения для более корректного описания процессов, происходящих в ВС.

В свойства сущности «процесс» помимо названия и номера включены еще нечеткие прогнозные оценки трафика и вычислительной загрузки, а также расписание выполнения процесса (рисунке 13). При моделировании работы системы расписание имеет решающее значение, так как в определенные моменты времени процесс не функционирует, и генерации трафика в сеть не происходит. Кроме того, для дальнейшей взаимосвязи с физической структурой сети в свойства процесса добавлен IP-адрес того компьютера, на котором этот процесс выполняется.

Поток данных в DFD представляет собой связь между блоками диаграммы с собственным номером. Эта связь указывает на другие процессы, с которыми взаимодействует данный процесс (рис. 14). При вычислении суммарных оценок сетевой загрузки потоки данных указывают маршруты, которые загружаются генераторами трафика.

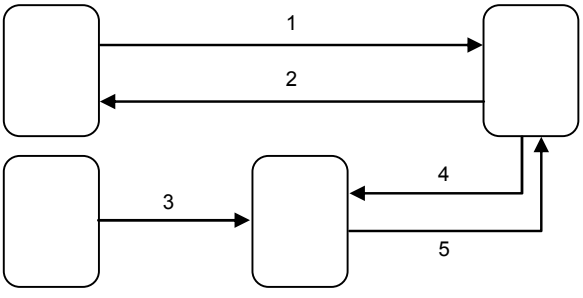

Рис. 14. Потоки данных DFD

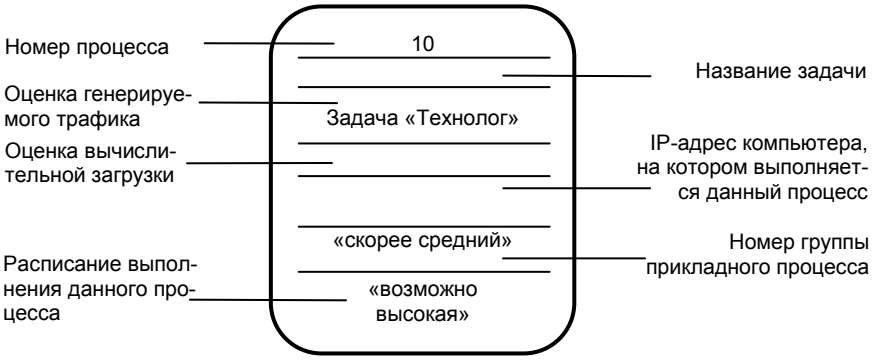

Рис. 13. Изображение процесса на потоковой диаграмме

6 Байесовская оптимизация ВС на прикладном уровне

6.1 Байесовские сети доверия (БСД) в оптимизации ВС

Для построения графа сети доверия необходима информация об интенсивностях взаимодействия компьютеров ВС между собой. Эта информация извлекается из дополненных DFD-диаграмм, описывающих автоматизируемые производственные процессы. В стандартные DFD-диаграммы были добавлены новые элементы.

Каждому пользователю ВС приписываются количество рабочих мест и выполняемые им задачи. Одним пользователем считается группа пользователей ВС, выполняющих одинаковые задачи.

Каждая задача представляется в виде последовательности элементов типа «Документ» или «Запрос», представляющих документы и запросы, передаваемые пользователем по сети. Различие между документами и запросами заключается в том, что документы имеют четкий числовой размер, а размер запросов является величиной нечеткой.

В разработанном прототипе САПР ВС механизм БСД используется для оптимизации следующих параметров ВС: состав коммуникационного оборудования и пропускная способность каналов. Преимуществом применения БСД является отсутствие необходимости в моделировании многочисленных промежуточных вариантов сети, среди недостатков следует отметить необходимость построения графа сети доверия для каждой ВС.

БСД для решения задачи оптимизации коммуникационного оборудования состоит из четырех слоев: слой взаимодействий, слой интенсивности трафика, слой внутрисегментных и межсегментных трафиков, слой определения состава оборудования. Рассмотрим функции каждого слоя БСД.

Слой 1 – слой взаимодействий. Данный слой вводит в БСД начальную информацию об интенсивностях взаимодействия компьютеров ВС между собой. Эта информация определяется

мастером перехода на основании информации об автоматизируемых производственных процессах, введенной в DFD-диаграмму. Величина интенсивности взаимодействия является нечеткой, так как передаваемый трафик состоит из документов, имеющих четкий размер, и запросов к базам данных, имеющих нечеткий размер.

Слой 2 — слой интенсивности трафика. На данном слое происходит группировка информации об интенсивностях взаимодействия компьютеров по каналам сети, через которые проходят взаимодействия. Основная трудность при построении данного слоя заключается в том, что у каждого узла имеется большое количество предков, что приводит к резкому возрастанию размера матрицы условных вероятностей (для трех предков размер матрицы равен 64 строкам, для пяти предков — 1024 строкам, для семи предков — 16384 строкам). Для преодоления данной трудности при построении графа БСД используется процедура группировки, принцип действия которой показан на рисунке 15.

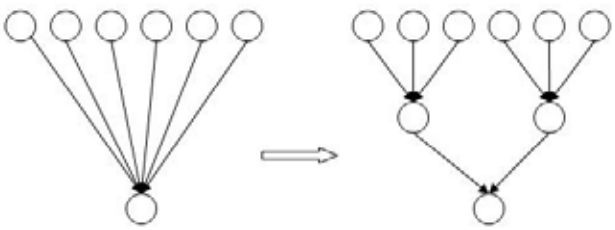

Рис. 15. Группировка узлов-предков в БСД

Основная идея работы процедуры заключается во введении новых узлов, на которые группируются узлы-предки, после чего введенные узлы становятся предками узлов второго слоя.

Слой 3 — слой внутрисегментных и межсегментных трафиков. На данном уровне определяется интенсивность внутрисегментных и межсегментных взаимодействий для каждого узла сети. Каждый узел сети образует сегмент. Для каждого узла сети в данном слое существуют два узла БСД — на одном формируется внутрисегментный трафик, на другом — межсегментный. Трафик формируется по принципу протекания через сегмент или внутри него. Таким образом, у каждого узла сети следующего слоя оказывается минимум один предок.

Слой 4 — слой определения состава оборудования. Данный слой является конечным слоем, и в его узлах формируются вероятности нахождения в определенном сегменте определенного типа коммуникационного оборудования. Для каждого узла ВС в данном слое существует один узел БСД, имеющий как минимум одного предка. Матрицы условных вероятностей для этого слоя задаются экспертом.

Для второго и третьего слоев матрицы условных вероятностей представляют собой таблицы сложения нечетких величин трафика.

Таким образом, оптимизация состава коммуникационного оборудования при помощи БСД позволяет получить результат, обусловленный автоматизируемыми производственными процессами и мнением эксперта.

В данной сети доверия выделяются три слоя: слой задания интенсивностей, слой определения трафика, слой определения пропускной способности. Рассмотрим функции каждого из слоев БСД.

Слой 1 — слой задания интенсивностей. Данный слой вводит в БСД начальную информацию об интенсивностях взаимодействия компьютеров ВС между собой. Эта информация определяется мастером перехода на основании сведений об автоматизируемых производственных процессах, введенных в DFD-диаграмму.

Слой 2 — слой определения трафика. В данном слое происходит группировка интенсивностей взаимодействий по каналам, через которые протекают взаимодействия.

Слой 3 — слой определения пропускной способности. В данном слое определяется пропускная способность каналов сети, необходимая для того, чтобы пропустить трафик, определенный для канала в предыдущем слое. Каждый узел данного слоя имеет только одного предка, несущего информацию о передаваемом по каналу трафике. Матрица условных вероятностей для данного слоя задается экспертом, для второго слоя она представляет собой таблицу сложения нечетких величин интенсивностей.

6.2 Сравнение байесовской и генетической оптимизации ВС

Наряду с использованием БСД в САПР ВС также могут использоваться генетические алгоритмы (ГА) для решения задач оптимизации, для которых сложно сформулировать набор правил или представить граф сети доверия. Это задачи оптимизации подключения рабочих станций (сложно сформулировать набор правил) и задача оптимизации размещения коммуникационного оборудования (сложно представить граф сети доверия).

Основная идея ГА, решающего задачу размещения коммуникационного оборудования, состоит в разбиении рабочего поля на 1024 части по обоим измерениям. Каждому узлу приписываются два коэффициента, описывающие местоположение узла. Таким образом, каждому узлу сети в хромосоме отвечает 20 генов (2 отрезка по 10 генов, каждый из отрезков дает 1024 варианта). Это справедливо для двумерной оптимизации, в трехмерном случае необходимо разбивать по трем измерениям и каждому узлу сети в хромосоме будет отвечать 30 генов.

Далее работает стандартный ГА. Функцией оптимальности является длина всех соедини-

тельных кабелей, алгоритм направлен на минимизацию функции оптимальности. В процессе работы алгоритм запоминает обработанные состояния, и в случае их повторения значения функции оптимальности получаются из сохраненных хромосом.

В алгоритме заложена возможность размещения оборудования в определенных пользователем областях. Для получения такого эффекта функция оптимальности каждой хромосомы удваивается при наличии факта нахождения какого-либо из узлов вне заданных областей (если промахнулся один узел, функция удваивается, если два узла — увеличивается в четыре раза и так далее). Таким образом, в процессе поиска минимума алгоритм автоматически загонит все узлы в рамки заданных ограничений (аналогично можно сделать для трехмерного случая, хотя сложность задания областей возрастет многократно).

Основная идея алгоритма для решения задачи оптимизации подключения рабочих станций заключена в том, что каждому компьютеру сети присваивается номер узла, к которому он подключен. Этот номер является геномом хромосомы алгоритма. Длина хромосомы равна количеству компьютеров в сети.

Основной недостаток алгоритма в том, что функцией оптимальности является максимальный трафик на каналах сети. Из этого следует, что для оценки оптимальности хромосомы необходимо производить статическое моделирование варианта сети, представленного хромосомой. С целью преодоления этого недостатка в алгоритм введен список промоделированных хромосом. Применение данного списка обусловлено частым повторением одних и тех же хромосом, особенно в конце процесса оптимизации.

7 Выводы по моделям прикладного уровня функционирования сети в крупной проектной организации

Модели прикладных процессов ВС определяют масштаб, а значит, в конечном счете, транспортную схему ВС. Понятие транспортной схемы включает в себя реальную топологию, состав коммуникационного оборудования сети. Моделирование прикладных процессов определяет параметры ВС. Задача автоматического проектирования (АП) ВС требует согласования прикладного и транспортного уровней описаний ВС.

Данный вывод и детали вышеприведенного исследования позволяют сформулировать следующим образом теоретический подход и методологию АП ВС.

АП ВС требует в качестве обязательной компоненты моделирование сети. Сеть представляет собой топологию узлов, каналов и коммуникационного оборудования. На прикладном уровне узлы делятся на серверы: данных, файлов, прокси и на «толстых» и «тонких» клиентов.

Взаимодействие узлов на прикладном уровне может быть описано на уровне производственных процессов, например, модифицированными DFD-диаграммами (дополнение расписанием работы ВС).

Для решения задач АП важны:

• варианты оптимизации:

1. Генетическая оптимизация выбора коммуникационного оборудования, типа пропускной способности каналов; (исследование и выбор типа ГА);

2. Выбор топологии сети — оптимизация переподключения узлов;

• принятие проектных решений — БСД используются для моделирования рассуждений проектировщика (сравнение по эффективности с ГА);

• функциональное моделирование узлов на прикладном уровне:

1. Имитационные модели серверов и клиентов на основе gpss;

2. Имитационные модели серверов и клиентов на основе сетей Петри;

• задача генерации транспортной схемы ВС, достаточной для организации работы прикладного уровня.

СПИСОК ЛИТЕРАТУРЫ

1. Батыршин И.З., Недосекин А.О., Стецко А.А., Тарасов В.Б., Язенин А.В., Ярушкина Н.Г. Нечеткие гибридные системы. Теория и практика / под ред. Н.Г. Ярушкиной. — М.: Физматлит, 2007.

2. Ярушкина Н.Г., Вельмисов А.П., Стецко А.А. Средства Data Mininq для нечетких реляционных серверов данных // Информационные технологии. — 2007. - № 6.