

М.В. Заморёнов, С.Е. Голиков

ИМИТАЦИОННАЯ МОДЕЛЬ ФУНКЦИОНИРОВАНИЯ АВТОМАТИЗИРОВАННОЙ БАНКОВСКОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ С РЕЗЕРВИРОВАНИЕМ

Аннотация. Показана роль операционных рисков в общей структуре банковских рисков. С целью снижения информационно-технологических рисков описана адаптивная программная инфраструктура банковской автоматизированной системы. Предложена имитационная модель функционирования сервера приложений, для которого реализовано резервирование замещением, т.е. при отказе одного сервера обслуживание передается на следующий сервер.

Ключевые слова: автоматизированная банковская система, адаптивная программная инфраструктура, сервер приложений, имитационная модель, операционные риски.

Введение

В настоящее время невозможно представить проведение каких-либо банковских операций без автоматизированной информационной системы. В то же время сбои в системе автоматизации банков вносят значительный вклад в величину операционного [6] или операционно-технологического риска по классификации НБУ [4]. Исследования последних лет [1] говорят о том, что основной причиной потерь финансовых организаций значительных денежных сумм, становятся операционные риски. Таким образом, защита информационных систем от влияния операционных рисков выходит на первый план. Следовательно, разработка и внедрение механизмов, позволяющих минимизировать операционные риски финансовых учреждений, является актуальным направлением для исследований в условиях глобализации финансовых услуг наряду с возрастающей сложностью применяемых в них информационных технологий.

Под архитектурной устойчивостью к сбоям понимается создание такой архитектуры приложения, которая отвечает всем техническим и операционным требованиям и обеспечивает оптимальные про-

изводительность, доступность, надежность и безопасность. Отказоустойчивость является одним из важнейших требований к информационной системе. Хорошая архитектура существенно снижает операционные риски, связанные с созданием и функционированием программной системы. Архитектура программного обеспечения систем автоматизации банков представляет собой комплекс, состоящий из набора компонентов, выполняющих определенную функцию или набор функций. Современные банковские системы представляют собой совокупность различных архитектурных стилей. Наиболее часто используемая архитектурная парадигма – клиент-серверная (N-уровневая) с элементами компонентной и многоуровневой. Клиент-серверная (N-уровневая) архитектура позволяет разнести инфраструктурные элементы на разные физические компьютеры, повышая отказоустойчивость всей системы, обеспечивая централизованный доступ к данным и простоту обслуживания, а компонентная архитектура упрощает функциональное наращивание путем повторного использования логических компонентов. Многослойность позволяет разделить логику представления от бизнес-логики и логики доступа к данным, что обусловлено требованиями банковской безопасности. В качестве модели взаимодействия на уровне представления в основном применяется шаблон представления с разделением (модель – представление – контроллер). В отечественных банковских системах, как правило, применяется трехуровневая архитектура, в которой программные сегменты физически размещены на разных компьютерах. Для связи между элементами программной инфраструктуры используются сообщения. Характеристиками такой архитектуры являются функциональная декомпозиция программной системы, распределенное развертывание, что обеспечивает повышенную масштабируемость, доступность, управляемость и эффективность использования ресурсов. Каждый уровень функционально изолирован от других, кроме тех, с которыми он непосредственно соседствует. Бизнес-слой в данной архитектуре развернут на сервере приложений, слой представления – на клиентских компьютерах, а сами данные – в базе данных, что повышает конфиденциальность и безопасность. Размещение бизнес-логики на отдельном сервере гарантирует доступность изменений для всех пользователей системы, изменения в настройках производятся централизованно, ответственность за аутентификацию пользователей пе-

реносится с потенциально небезопасного клиентского уровня на уровень сервера приложений, скрывая уровень базы данных.

Основной проблемой централизованной обработки является гарантированность сервера приложений, так как в случае его выхода из строя клиенты не смогут получить доступ к приложению. Обычно, заданная величина отказоустойчивости достигается либо средствами операционной системы, либо средствами СУБД, либо применением аппаратных решений, что требует достаточно высокой квалификации персонала и значительных денежных затрат.

В работе [2] предложена динамически настраиваемая инфраструктура системы автоматизации банков. Суть предлагаемого решения состоит в применении резервирования для сервера приложений, осуществляющего функции обработки бизнес - логики и доступа к данным, хранящихся в базе данных. При выходе из строя основного сервера приложений, клиентское приложение автоматически подключается к одному из резервных серверов приложений, обеспечивая доступность данных. Повышенная живучесть архитектуры обеспечивается возможностью транслировать клиентские запросы в случае выхода из строя всех серверов приложений непосредственно к используемой СУБД. Логическая схема сервера приложений приведена на рисунке 1.

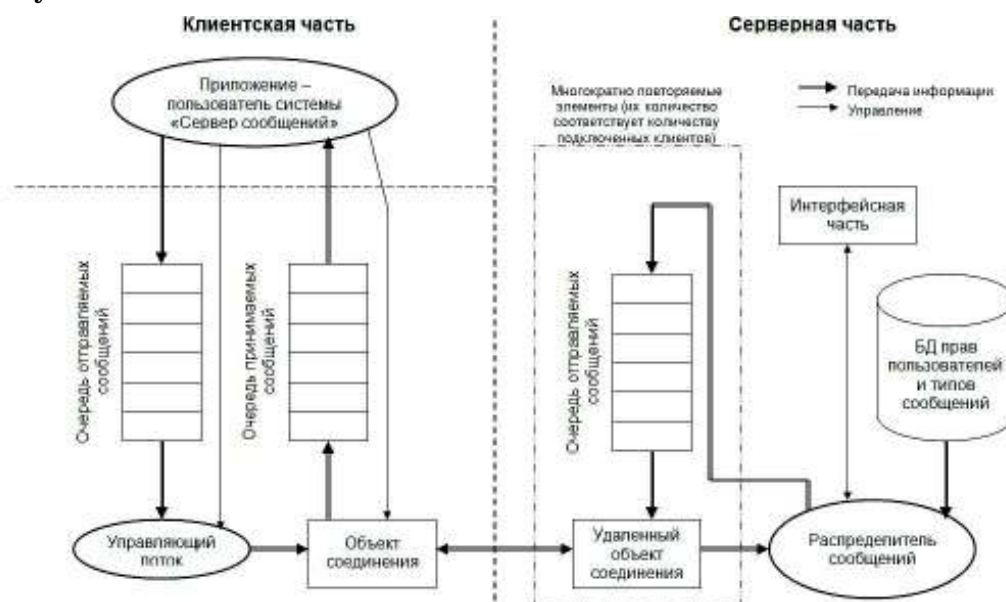


Рисунок 1 – Логическая схема сервера приложений

В данной архитектуре используется резервирование замещением, при котором резервный сервер приложений включается в работу

автоматически при перенаправлении на него запросов от клиентских приложений.

Постановка и решение задачи

Необходимо построить имитационную модель адаптивной банковской системы, наиболее уязвимым звеном которой является сервер приложений. Модель подразумевает, что отказавшие элементы заменяются (ремонтируются), а устройство обеспечения резервирования имеет нулевую вероятность отказа (надежность устройство обеспечения резервирования включена в надежность элементов резервирования), также принято, что математическое ожидание времени обработки запросов, наработки на отказ и восстановления серверов одинаковое.

Целью построения имитационной модели является определение коэффициента использования серверов приложений различных уровней резервирования при заданных параметрах надежности и производительности.

Сервер приложений можно рассматривать как дискретную стохастическую систему с постоянной структурой. Любая подобная система неизбежно испытывает различные возмущения, источниками которых могут быть либо внешние воздействия, обусловленные случайными или систематическими изменениями окружающих условий, либо внутренние флуктуации, возникающие в самой системе в результате взаимодействия элементов. При исследовании эти системы обычно представляются в виде стохастических моделей дискретных процессов. Несмотря на успешное применение методов аналитического моделирования, достаточно эффективным методом исследования таких систем остается имитационное моделирование на ЭВМ с применением специализированных систем.

Система GPSS предоставляет пользователю законченную высокоуровневую информационную технологию создания имитационных моделей [3, 5]. В этой системе имеются средства формализованного описания параллельных дискретных процессов в виде условных графических изображений или с помощью операторов собственного языка. Координация процессов осуществляется автоматически в едином модельном времени. Пользователь в случае необходимости может ввести свои правила синхронизации событий. Имеются средства управления моделью, динамической отладки и автоматизации обработки

результатов. В языке GPSS предусмотрены различные виды объектов, используемых при моделировании дискретных систем: сообщения, устройства, многоканальные устройства, очереди и т.д.

При построении имитационной модели для сервера приложений предполагается, что время обслуживания единицы продукции, время безотказной работы, время восстановления являются независимыми СВ, имеющими конечные математические ожидания и дисперсии. При отказе сервера приложений обслуживание запроса прерывается, а следующий запрос обрабатывается на резервном сервере приложений. После восстановления работоспособности отказавшего сервера запросы снова начинают поступать к нему.

На рисунке 2а приведена блок-схема алгоритма функционирования первого сервера приложений, на рисунке 2б – алгоритм его наработки на отказ и восстановления.

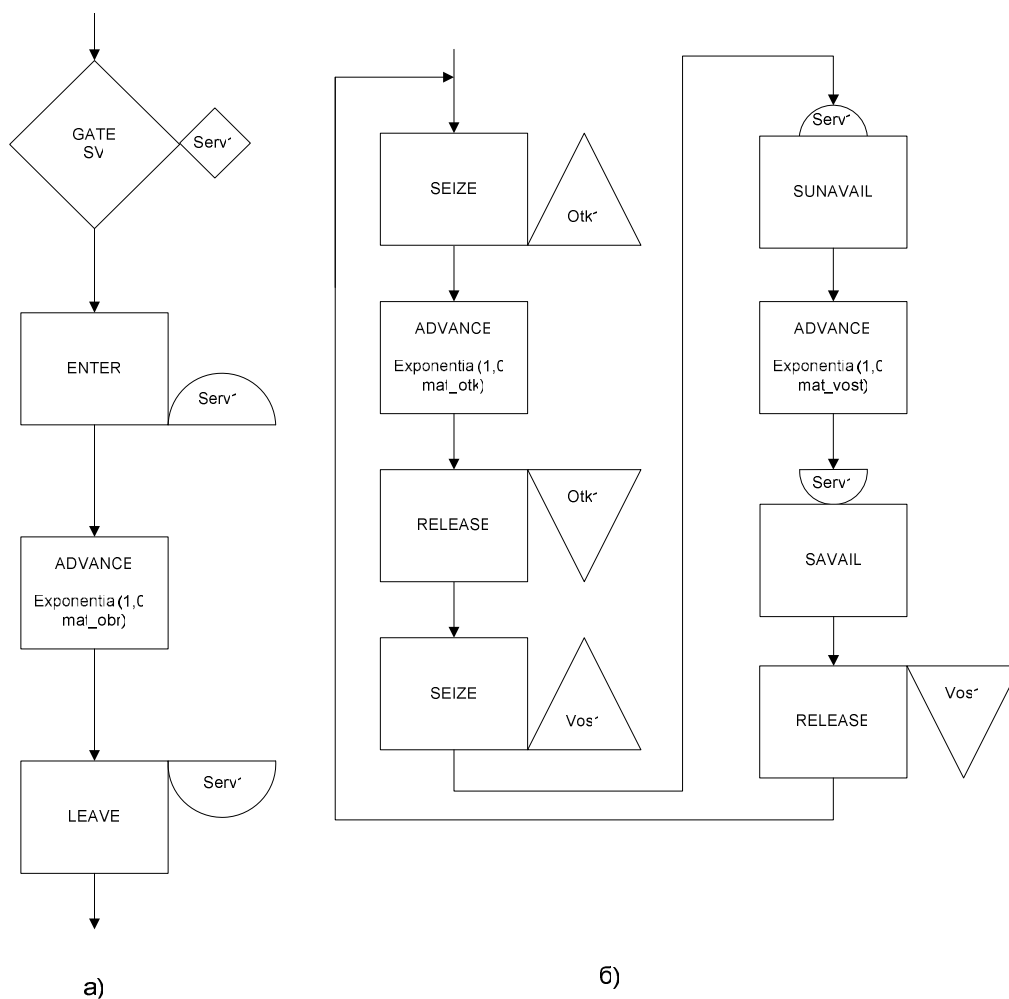


Рисунок 2 – блок-схема алгоритма: а) – функционирования сервера приложений №1; б) – наработки на отказ и восстановления сервера приложений №1

Законы распределения времени обслуживания единицы продукции, наработки на отказ и восстановления ячейки полагаются экспоненциальными.

Время моделирования принимается 4 месяца, единица модельного времени – 1 мин.

При реализации данной задачи была использована система имитационного моделирования GPSS World Student version 5.2.2.

Текст программы, приведенный ниже, снабжен комментариями и дополнительных пояснений не требует.

```

*****
GPSS World File –Модель функционирования сервера приложений.
Единица модел. времени – 1 мин
*****
mat_zapr      EQU  0.01 ;
Математическое ожидание времени запроса
mat_obr EQU  1 ; Математическое ожидание времени обработки
mat_otk EQU  1000;Математическое ожидание времени наработки на
отказ
mat_vost EQU  100 ;
Математическое ожидание времени восстановления
schetchik EQU  1 ; Номер генератора случайных чисел
serv1 storage 100 ; Ёмкость первого сервера приложений
serv2 storage 100 ; Ёмкость второго сервера приложений
serv3 storage 100 ; Ёмкость третьего сервера приложений
serv4 storage 100 ; Ёмкость четвертого сервера приложений
serv5 storage 100 ; Ёмкость пятого сервера приложений
generate      (Exponential(schetchik,0,mat_zapr)) ;
Генерация запросов
queue och1 ; Постановка запросов в очередь
;первый сервер
met1gate sv serv1,met2 ; проверка работоспособности первого сервера
enter serv1 ; поступления запроса на первый сервер
depart och1 ; освобождение очереди
advance (Exponential(schetchik,0,mat_obr)) ; обработка запроса
leave serv1 ; запрос покидает сервер
transfer ,mett ; окончание запроса (удаление запроса из системы)
;второй сервер
met2gate sv serv2,met3 ; проверка работоспособности второго сервера
enter serv2 ; поступления запроса на второй сервер
depart och1 ; освобождение очереди
advance (Exponential(schetchik,0,mat_obr)) ; обработка запроса
leave serv2 ; запрос покидает сервер
transfer ,mett ; окончание запроса (удаление запроса из системы)
;третий сервер
met3gate sv serv3,met4 ; проверка работоспособности третьего сервера
enter serv3 ; поступления запроса на третий сервер

```

```

depart och1 ; освобождение очереди
advance (Exponential(schetchik,0,mat_obr)) ; обработка запроса
leave serv3 ; запрос покидает сервер
transfer ,mett ; окончание запроса (удаление запроса из системы)
;четвертый сервер
met4gate sv serv4,met5 ; проверка работоспособности четвертого сервера
enter serv4 ; поступления запроса на четвертый сервер
depart och1 ; освобождение очереди
advance (Exponential(schetchik,0,mat_obr)) ; обработка запроса
leave serv4 ; запрос покидает сервер
transfer ,mett ; окончание запроса (удаление запроса из системы)
;пятый сервер
met5gate sv serv5,mett ; проверка работоспособности пятого сервера
enter serv5 ; поступления запроса на пятый сервер
depart och1 ; освобождение очереди
advance (Exponential(schetchik,0,mat_obr)) ; обработка запроса
leave serv5 ; запрос покидает сервер
transfer ,mett ; окончание запроса (удаление запроса из системы)
mett terminate
;отказ первого сервера
generate ,,1
mot1 seize otk1 ; в данном устройстве
advance (Exponential(1,0,mat_otk)) ; имитируется наработка
release otk1 ; на отказ первого сервера
seize vos1 ; в данном устройстве имитируется восстановление
sunavail serv1 ; первый сервер становится недоступен
advance (Exponential(1,0,mat_vost)) ; восстановление
savail serv1 ; первый сервер становится доступен
release vos1 ; окончание восстановления
transfer ,mot1 ; начало наработки на новый отказ
;отказ второго сервера
generate ,,1
mot2 seize otk2 ; в данном устройстве
advance (Exponential(1,0,mat_otk)) ; имитируется наработка
release otk2 ; на отказ второго сервера
seize vos2 ; в данном устройстве имитируется восстановление
sunavail serv2 ; второй сервер становится недоступен
advance (Exponential(1,0,mat_vost)) ; восстановление
savail serv2 ; второй сервер становится доступен
release vos2 ; окончание восстановления
transfer ,mot2 ; начало наработки на новый отказ
;отказ третьего сервера
generate ,,1
mot3 seize otk3 ; в данном устройстве
advance (Exponential(1,0,mat_otk)) ; имитируется наработка
release otk3 ; на отказ третьего сервера
seize vos3 ; в данном устройстве имитируется восстановление
sunavail serv3 ; третий сервер становится недоступен
advance (Exponential(1,0,mat_vost)) ; восстановление

```

```

savail serv3 ; третий сервер становится доступен
release vos3 ; окончание восстановления
transfer ,mot3 ; начало наработки на новый отказ
;отказ четвертого сервера
generate ,,1
mot4 seize otk4 ; в данном устройстве
advance (Exponential(1,0,mat_otk)) ; имитируется наработка
release otk4 ; на отказ четвертого сервера
seize vos4 ; в данном устройстве имитируется восстановление
sunavail serv4 ; четвертый сервер становится недоступен
advance (Exponential(1,0,mat_vost)) ; восстановление
savail serv4 ; четвертый сервер становится доступен
release vos4 ; окончание восстановления
transfer ,mot4 ; начало наработки на новый отказ
;отказ пятого сервера
generate ,,1
mot5 seize otk5 ; в данном устройстве
advance (Exponential(1,0,mat_otk)) ; имитируется наработка
release otk5 ; на отказ пятого сервера
seize vos5 ; в данном устройстве имитируется восстановление
sunavail serv5 ; пятый сервер становится недоступен
advance (Exponential(1,0,mat_vost)) ; восстановление
savail serv5 ; пятый сервер становится доступен
release vos5 ; окончание восстановления
transfer ,mot5 ; начало наработки на новый отказ
generate 172800 ; счетчик времени (4 месяца непрерывной ра-
боты)
terminate 1

```

Результаты моделирования представлены в таблице 1.

Таблица 1

Результаты моделирования

Исходные данные				Результаты моделирования				
По- ступл. запро- са, мин	Обра- ботка запроса, мин	Отказ серве- ров, мин	Вос- стан. серве- ра, мин	Кэфф. исп. сер- вера №1	Кэфф. исп. сер- вера №2	Кэфф. исп. сер- вера №3	Кэфф. исп. сер- вера №4	Кэфф. исп. серве- ра №5
0,01	1	100	1000	0,918	0,075	0,007	0,001	0
0,025	5	100	1000	0,914	0,064	0,016	0,001	0
0,1	10	100	1000	0,892	0,097	0,009	0,001	0

Выводы

Построенная имитационная модель функционирования адаптивной автоматизированной банковской информационной системы с резервированием позволяет получить коэффициенты использования серверов приложений, при помощи которых планируется определять целесообразность использования определенного количества резервных серверов, исходя из возможных потерь при отказе одновременно

всей структуры. Как видно из таблицы 1, при использовании одного сервера приложений без резерва приводит при первом варианте исходных данных к потерям 8,2% запросов. В тоже время, если использовать хотя бы один из них, потери снижаются и составляют 0,8%, что может составлять достаточно большую сумму, в зависимости от стоимости одного запроса. В дальнейшем планируется построение аналитической модели функционирования такой системы.

ЛИТЕРАТУРА

1. 2012 ORX Report on Operational Risk Loss Data. /Электронный ресурс. <http://www.orx.org/Pages/orxdata.aspx>
2. Голиков С.Е. Применение динамически настраиваемой инфраструктуры системы автоматизации банков с целью минимизации операционных рисков. /С.Е. Голиков//Системы обработки информации № 12 2012.
3. Томашевский В.Н. Имитационное моделирование в среде GPSS [Текст] / В.Н. Томашевский, Е.Г. Жданова. – М.: Бестселлер, 2003. — 416 с.
4. Филатов Б.Г. Определение операционного риска в банковской деятельности. Принципы управления./Б.Г. Филатов//Электронный ресурс.
http://www.nbu.gov.ua/portal/soc_gum/pprbbsu/texts/2009_27/09_27_11.pdf
5. Шрайбер Т.Дж. Моделирование на GPSS [Текст] / Т.Дж. Шрайбер ; [пер. с англ. В.И. Гарчера, И.Л. Шмуйловича]; под ред. М.А. Файнберг. – М.: Машиностроение, 1980. – 592 с.
6. The New Basel Capital Accord. Basel Committee on Banking Supervision. April 2003.
/Электронный ресурс.
http://www.oenb.at/de/img/cp3_overview_tcm14-13383.pdf