$T_{ans}$ - средняя продолжительность вызовов, переадресованных на голосовую почту, с последующим оставлением сообщений на ней;

$T_{rel}$ - средняя продолжительность вызовов, переадресованных на голосовую почту, без оставления сообщений.

Предположим следующее: $T_{suc}$ = 120 секунд; $T_{ans}$ = 20 сек; $T_{rel}$ = 6сек.

Исходя из этого получаем:

1. Нагрузка, поступившая из внешней сети и закончившаяся разговором
$$A_{ext.conv.} = (16.8 * N_{calls} / T_{meas})$$
2. Нагрузка, поступившая из внешней сети, закончившаяся переадресацией на голосовую почту с оставлением сообщения
$$A_{ext.vm.mes.} = (0.12 * N_{calls} / T_{meas})$$
3. Нагрузка, поступившая из внешней сети, закончившаяся переадресацией на голосовую почту без оставления сообщения
$$A_{ext.vm.rel.} = (0.32 * N_{calls} / T_{meas})$$
4. Нагрузка на голосовую почту, поступившая из внешней сети, закончившаяся переадресацией на голосовую почту с оставлением сообщения
$$A_{int.vm.mes.} = (0.48 * N_{calls} / T_{meas})$$
5. Нагрузка на голосовую почту, поступившая из внешней сети, закончившаяся переадресацией на голосовую почту без оставления сообщения
$$A_{int.vm.rel.} = (1.3 * N_{calls} / T_{meas})$$

Сравнив нагрузку, получаемую на участке MSC11 – MSC12 при применение метода с задержкой установления соединения и без него, мы получим, что нагрузка на участке MSC11 – MSC12 при централизованном методе управления без задержки установления соединения больше нагрузки на том же участке при централизованном методе управления с задержкой установления соединения в 1.1 раза. Т.е., избавившись от паразитной нагрузки с помощью задержки установления соединения, мы можем увеличить полезную нагрузку на участке MSC11 – MSC12 на 10% без какого-либо влияния на качественные характеристики направления!

Таким образом, при использовании централизованного метода управления услугой голосовой почты метод с задержкой установления соединения, основанный на вероятностно-игровом методе, может принести хороший результат значительно увеличив эффективность используемых разговорных направлений и существенно уменьшив паразитную нагрузку, создаваемую пользователями сервера голосовой почты.

# РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ УПРАВЛЕНИЯ СОДЕРЖАНИЕМ ИНФОРМАЦИОННОГО СЕРВЕРА

Стрикелев Д.А.

По мере развития Интернета и включения его в структуры реального бизнеса растёт количество людей, в прямые служебные обязанности которых входит работа с содержанием информационного сервера. Квалификация большинства из них находится на уровне офисного работника, не всегда владеющего достаточными знаниями в области computer science. Это ставит задачу по максимальному упрощению процесса управления содержанием информационного сервера. Чем же плоха схема размещения информации, принятая на обычных статических информационных серверах? В основном тем, что она, при достаточно высокой стоимости ее поддержки, может удовлетворить только простейшие запросы пользователей. Это обусловлено следующими факторами.

- Смешение оформления и содержания: в тексте страницы вперемешку идут информация и визуальная разметка, что затрудняет публикацию новых и модификацию старых документов. При любом изменении внешнего вида информационного сервера необходимо привлекать профессиональных дизайнеров и программистов.
- Невозможность автоматизировать типичные операции: например, при добавлении нового документа невозможно сразу добавлять его на страницу с картой информационного сервера и в список новых документов.
- Затрудненная поддержка онлайновых сообществ (аутентификация посетителей, управление личными настройками, разделение доступа по имеющимся правам, обратная связь и т.д.): статическая схема размещения материала не позволяет динамически настраивать внешний вид страницы в зависимости от нужных параметров.
- Невозможность поддержки информационных бизнес-процессов: реальные процессы,



http://edoc.bseu.by

имеющие место в бизнес-деятельности, требуют обеспечения постоянного доступа к одной и той же информации для многочисленных его участников, что по своей сути не укладывается в жесткую схему «подготовил – опубликовал», единственно возможную в рамках статического сервера.

Всё это привело к осознанию необходимости «динамичности» информационного сервера. Содержание его хранится в базе данных, а страницы генерируются динамически, «на лету», согласно визуальному представлению и алгоритмам, заданным при его создании. Функциональные возможности такого сервера намного шире, однако, и ему присущ ряд весомых недостатков:

- По-прежнему не устранена зависимость содержания и оформления, пусть и на другом уровне: структурно содержание отделено от оформления, но при возможных переделках все еще нужна помощь специалистов.
- Из-за необходимости «конструирования» страницы каждый раз при её показе, велики вычислительные затраты, приходящиеся на каждый запрос пользователя. Как следствие, при высокой посещаемости информационного ресурса, сервер может попросту не справиться с объёмом необходимых вычислений.
- Для управления содержанием сервера необходимо постоянное подключение к Интернету на этапе администрирования, что по ряду причин (в первую очередь, временных) не всегда целесообразно.

Компромиссом между двумя вышеперечисленными решениями являются системы Web-паблишинга или управления контентом. За счёт ограничений, накладываемых на логическую структуру содержания, внешний вид и функциональные возможности информационного сервера, такие системы позволяют радикально снизить трудоёмкость управления содержанием. Они представляют собой некоторое программное обеспечение, устанавливаемое на сервере, хранящее содержание в базах данных и управляемое через Web-интерфейс. Все подобные системы в том или ином объёме отделяют оформление от содержания, автоматизируют работу с сообществами и поддерживают бизнес-процессы. В то же время, даже им не удаётся избавиться от всех названных недостатков старых систем.

При рассмотрении существующих решений в области систем Web-паблишинга, в первую очередь следует отметить имеющееся разделение предлагаемых программных продуктов на три группы: high-end (ориентированные на крупные фирмы и корпорации), medium-end (ориентированные на фирмы среднего размера) и low-end (ориентированные на низкобюджетного потребителя).

На рынке high-end внимание в первую очередь привлекают разработки: компании Lotus (Lotus Domino) на западном рынке и, например, фирмы Individ (Saitistika) в России. Логика построения указанных систем (прегенерация статических страниц из исходных данных, хранящихся в базе данных) позволяет им с лёгкостью избежать вычислительной «перегрузки» сервера, а необходимость постоянного подключения к Интернет не играет ключевой роли для корпоративного пользователя. Основным сдерживающим фактором для применения подобных разработок является стоимость программного обеспечения и высокие запросы к аппаратной части сервера (так, например, для ПО Individ'а необходимо наличие двух выделенных серверов).

Среди продуктов класса medium-end можно отметить Lotus Quickplace и Zope. К серьёзным недостаткам Quickplace можно отнести очень жёсткие ограничения на структуру и дизайн информационного сервера, а также сложность расширения его функциональности при выходе за «запланированные» разработчиками Quickplace границы. К сильным сторонам программного продукта относятся богатые возможности публикации информации (мощный редактор текстов, выполненный в виде ActiveX-компонента) и возможность работы в оффлайновом режиме с последующим «сливанием» данных на сервер. Минусами Zope являются необходимость постоянной работы через Web-интерфейс и отсутствие адаптации пакета на отечественный рынок, а к существенными плюсам можно отнести гибкую модель авторизации и разграничения прав доступа (схема «пользователи-роли-права») и возможность обновлять содержимое сервера по команде.

Что же касается наличия подобных программных средств класса low-end, то здесь нет признанных лидеров. Имеющиеся решения, как правило, не доведены до коммерческого продукта и используются только внутри фирмы-разработчика. Поэтому остается актуальной задача создания программного продукта, предназначенного для автоматизации процесса разработки и поддержания низкобюджетных информационных серверов и управления их содержанием.

Рассматриваемый комплекс программ Missionary (Windows-клиент) и Web-Missionary (веб-приложение) и является попыткой решения данной задачи. Missionary представляет собой оболочку (однопользовательское рабочее место администратора), предназначенную для создания и поддержки информационной структуры (скелета) информационного сервера. Внешний вид главного окна оболочки представлен на Рис. 1.

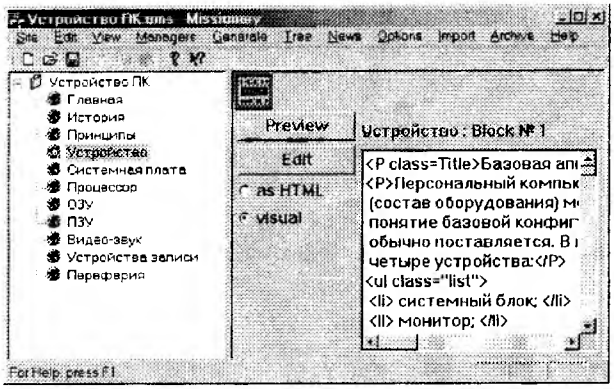


*Рисунок* 1. Главное окно приложения Missionary

Web-Missionary представляет собой веб-интерфейс для управления такими динамическими объектами сервера как: рубрики, новости, объявления, форумы, списки рассылки, голосования, опросы и т.д. (внешний вид типичного рабочего окна приложения изображён на Рис. 2). При этом полагается, что любой сервер делится на две части – квазистатическую и динамическую (термин «квазистатическая» употреблён в связи с тем, что указанная часть не является статической, но выглядит именно таковой).

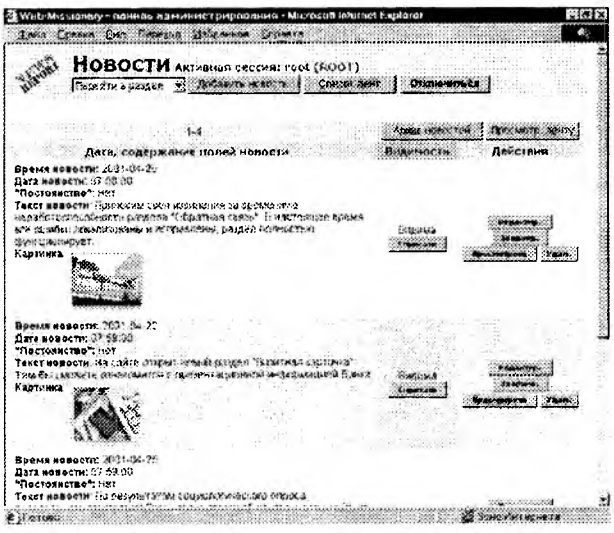


*Рисунок* 2. Типичное рабочее окно Web-Missionary

Возможности ПО Missionary позволяют создавать структуру информационного сервера, проводить наполнение и редактирование его разделов, как с помощью встроенного редактора, так и внешних модулей. Ключевыми требованиями к программному продукту были максимальное удобство пользования и универсальность применения.

Схема функционирования Missionary представлена на Рис. 3.

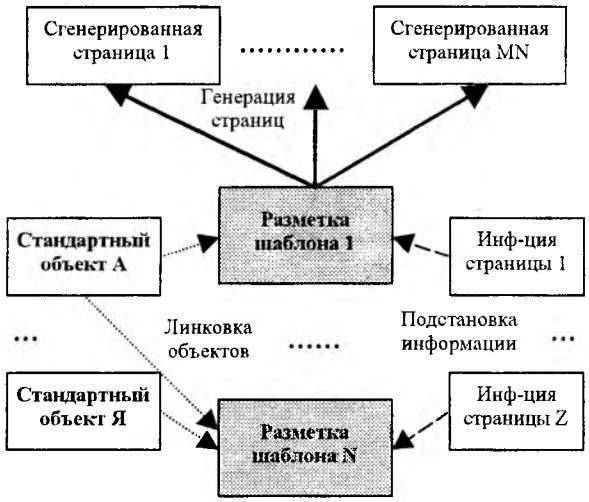


*Рисунок* 3. Структурная схема работы программы Missionary

Сервер, создаваемый средствами Missionary, можно условно разбить на следующие структурные единицы: шаблоны, объекты, страницы, файлы и каталоги.

Страница – это файл HTML, созданный как встроенными средствами Missionary, так и внешними редакторами. Все страницы разрабатываются на основе шаблонов, созданных в проекте, причем использование фреймов (frames), как основы для построения шаблона заранее исключается. В шаблоне, при помощи внутренней разметки, отмечаются места для информационных блоков страницы – изменяемой информации. А сам шаблон, в первую очередь, определяет дизайн и расположение информационных блоков страницы – является неизменяемой частью.

Шаблон представляет собой определённый пользователем HTML-код, в указанные места которого подставляется присутствующая на странице информация, и обращения к стандартным объектам, определённым в программе. Собственно, достичь удобства пользования и универсальности программного продукта удалось именно благодаря применению при опре-

делении информационных страниц сервера концепции «шаблона».

Все файлы: графика, файлы для скачивания или программы, включаемые в проект, копируются в папку для файлов, находящуюся внутри корневого каталога сайта. Для управления подключаемыми файлами, программный комплекс содержит специально разработанную подсистему - «Менеджер файлов».

Работа с каталогами мало чем отличается от общепринятых стандартов: директории создаются, переименовываются, копируются или удаляются. Особенностью является то, что программа управляет деревом каталогов только внутри корневой папки информационного сервера, которая создается одновременно с учетной записью для нового сервера.

Мощным инструментом пакета Missionary является использование объектов. Объект – это хранящийся отдельно HTML код, описывающий то или иное представление информации. Примером может служить меню, имеющее активные элементы (когда элемент меню ссылается на загруженную страницу). Описав однажды такой объект и установив ссылку на него в шаблоне страницы, мы обеспечиваем все страницы, созданные на основе такого шаблона, полнофункциональным меню. При необходимости изменить представление меню будет достаточно подкорректировать объект и обновить проект, вместо редактирования каждой страницы в отдельности.

Объекты могут иметь вложенные объекты, т.е. код объекта содержит ссылку на другой объект или файл, что позволяет гибко организовывать структуру информационного сервера. Реализовать подобную особенность позволила «интеллектуальная вложенная обработка», заключающаяся в том, что программный комплекс производит итерационную обработку страниц до тех пор, пока на очередном шаге итерации не останется изменённых объектов.

Программа Missionary поддерживает ряд встроенных объектов, обеспечивая механизм полной обратной совместимости, т.е. добавление новых поддерживаемых объектов при расширении программного комплекса никак не влияет на уже созданные объекты. Для формирования списка объектов был проведен тщательный анализ наиболее часто используемых и наиболее сложных с точки зрения внесения изменений блоков страницы.

Встроенными объектами являются:

- Default – данный объект представляет собою обычный код HTML, введенный пользователем. Содержимое такого объекта будет подставляться в код страницы или шаблона в неизменном виде. Использование данного объекта имеет смысл при наличии неизменных блоков, присутствующих на большом количестве страниц сервера.
- Menu – представляет собой связанные в рамках определенной структуры элементы, являющиеся ссылками на страницы проекта, страницы Internet или ссылки другого типа. Данный объект позволяет очень гибко настраивать внешний вид навигационного меню, расположенного на страницах сервера, в зависимости от того, является ли страница активной, есть ли вложенное подменю и т.п.
- News – данный объект представляет собой шаблон для представления блока новостей. В нем описывается лишь структура вывода сообщений новостей. При разработке программного обеспечения крупных серверов данный объект может не использоваться и быть заменён модулем Web-Missionary.News, имеющим более широкие возможности, как с точки зрения удобства использования многими пользователями одновременно, так и с точки зрения политики безопасности.
- Location bar – данный объект предназначен для создания панели навигации данной страницы, т.е. иерархии ссылок по разделам от заглавной страницы до текущей. Использование данного объекта особенно полезно при создании сложных иерархических структур, т.к. позволяет легко структурировать информационные страницы по степени соподчинённости «Предок-потомок».
- Site map - данный объект предназначен для автоматического построения карты информационного сервера или его части, создаваемой на основе наименований существующих областей. Его использование снимает необходимость вносить изменения в страницу, содержащую иерархически структурированное множество всех страниц сервера.
- Styles - данный объект содержит описание CSS-стилей разметки, используемых в проекте или в какой-либо его части. Определённый стили становятся доступными на всех страницах, к которым подключен объект, что исключает необходимость многократного редактирования одной и той же информации в разных файлах.
- PHP Script - данный объект содержит код программы PHP. Удобство его использова-

ния заключается в том, что при генерации локального проекта вызовы PHP-функций автоматически помечаются как комментарии, что позволяет избежать сообщений броузера о многочисленных ошибках. При генерации же удалённого объекта, все вызовы функций остаются незакомментированными.

- User Interpreted – объект интерпретируется по алгоритму, заданному пользователем. Является наиболее общим типом объекта, по сути «прародителем» всех остальных.

Заключительным этапом в работе над проектом является его генерация (т.е. генерация всех страниц информационного сервера), после чего готовые страницы «выкладываются» на сервер по протоколу FTP.

Web-Missionary представляет собой набор подсистем, собранных вместе по модульной схеме (наличие или отсутствие одной подсистемы не влияет на функционирование других). Конкретный набор модулей может изменяться в зависимости от требований к серверу. Web-Missionary по принципу своей работы является веб-приложением, т.е. взаимодействие с пользователем осуществляется посредством веб-броузера, а данные затем пересылаются по Интернету. Данный подход позволил снять требования на присутствие того или иного программного обеспечения на рабочем месте (достаточно наличия только веб-броузера, входящего в подавляющее большинство операционных систем) и позволил пользователям работать с программным комплексом с любого компьютера, подключенного к сети Интернет (вне зависимости от его географического положения и т.п.).

Для осуществления политики безопасности, система реализует многоуровневую схему доступа с назначением прав, позволяя также при необходимости производить шифрование по протоколу SSL (Secure Sockets Layer) всех передаваемых между броузером и сервером данных.

Структурная схема функционирования панели администрирования представлена на Рис. 4.

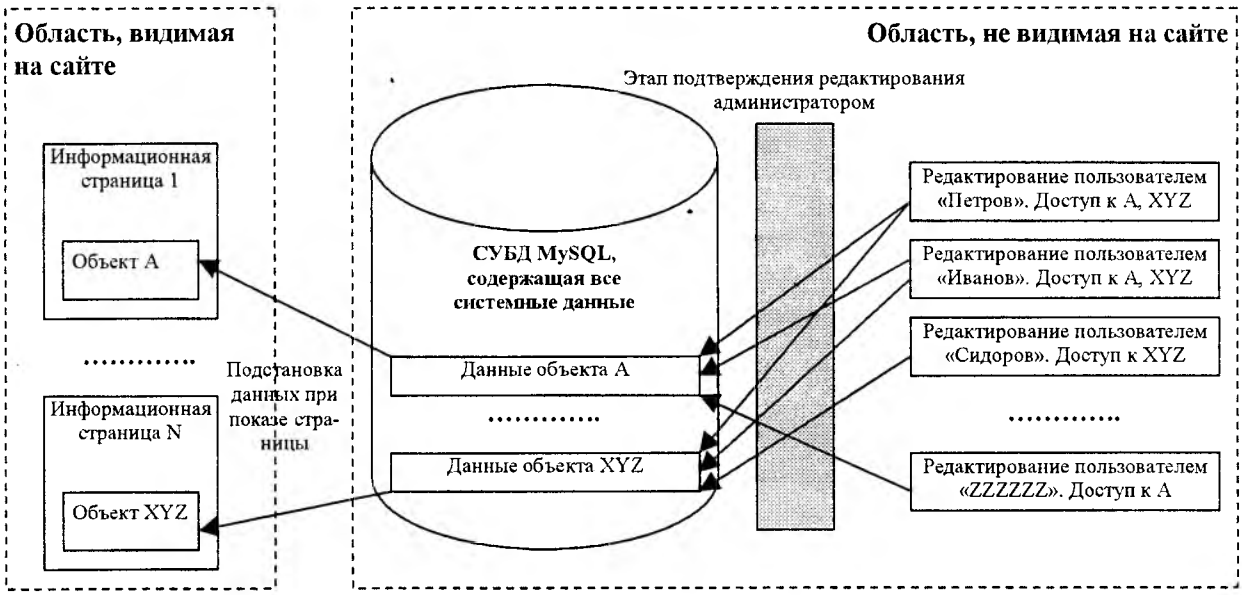


*Рисунок* 4. Структурная схема функционирования панели администрирования.

Информационные страницы могут содержать программный код объектов, поддерживаемых панелью администрирования. При получении запроса пользователя на просмотр страницы, этот программный код отвечает за подстановку необходимой информации из базы данных. В свою очередь, содержимое базы данных изменяется специально указанными «редакторами». Их задачей является работа с информационным наполнением сервера. После редактирования данных, система отсылает администратору запрос на подтверждение или отказ в подтверждении сделанных изменений. Сам же администратор не может производить редактирование информации. Таким образом осуществляется разделение обязанностей так, чтобы никто не имел исключительного права доступа.

Для формирования списка поддерживаемых объектов был проведен тщательный анализ наиболее часто используемых и важных динамических компонентов информационного сервера (результаты анализа были проверены эмпирически). В данной версии, программное

обеспечение Web-Missionary обеспечивает работу со следующими объектами (модулями): Пользователи, Файлы, Рассылка, Объявления, Голосование, Новости, Форумы, Анкеты, Рубрики, Архивирование.

Служебная панель администрирования Web-Missionary является многопользовательским программным обеспечением. Оно работает по многоуровневой схеме прав доступа с подтверждением вносимых изменений. Программой поддерживаются 4 типа пользователей:

- Редактор (USER). Данный тип отвечает за наполнение соответствующих разделов информацией.
- Администратор системы (SYSA). Данный тип отвечает за подготовку к использованию тех или иных модулей, разметку и определение параметров используемых объектов.
- Администратор безопасности (SECA). Данный тип отвечает только за подтверждение всех операций, произведенных редакторами и администраторами системы. Вносить какие-либо изменения в содержание он не может.
- ROOT. Супер-пользователь, работающий в привилегированном режиме. Может выполнять все операции первых 3-х типов, причём его действия применяются напрямую к центральной копии данных, видимой на сервере без необходимого этапа подтверждения.

Для каждого типа пользователей, кроме ROOT, указывается набор модулей, для работы с которыми он имеет надлежащие права. Число пользователей неограниченно, однако в системе может существовать только 1 супер-пользователь (добавление большего количества супер-пользователей достигается только прямым редактированием пользовательской таблицы в системной базе данных).

В общих чертах, логика работы системы отражена на схеме (Рис. 5):

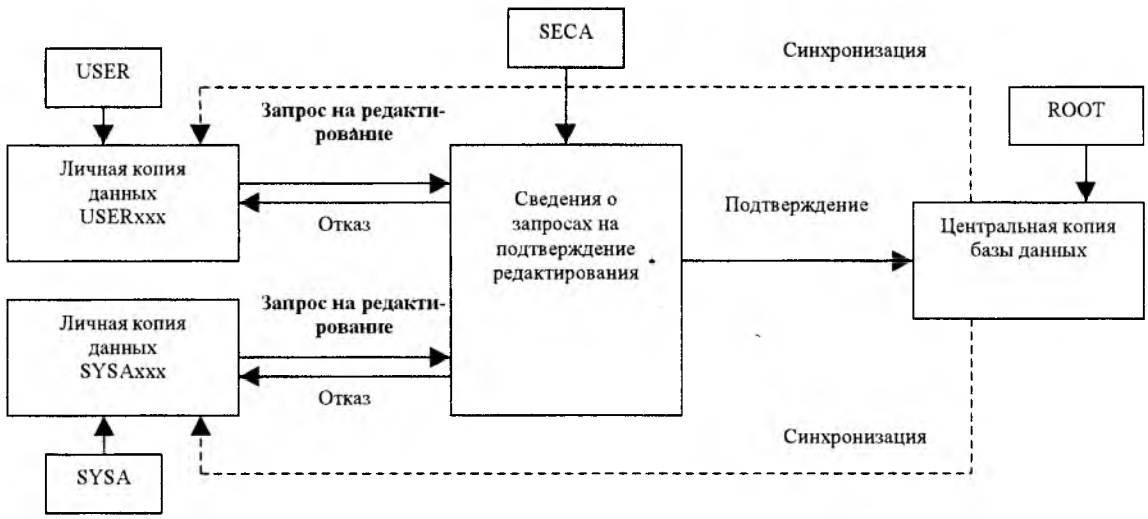


*Рисунок* 5. Схема работы подсистемы безопасности панели Web-Missionary

Каждый редактор и системный администратор имеет свою собственную копию системных данных, с которой он работает (т.е. все изменения затрагивают только его собственную копию, не мешая другим). После внесения всех запланированных изменений, пользователь формирует запрос администратору безопасности на подтверждение редактирования. При положительном ответе, сделанные изменения реплицируются на все копии данных (включая и корневую, видимую в Интернет). При отказе, личная копия пользователя «откатывается» к виду центральной копии (эталона), видимой в Интернет.

В качестве дополнительных мер для реализации политики безопасности можно отметить:

- генерацию идентификатора пользовательского сеанса, содержащего информацию о времени активности пользователя, рабочей станции, с которой была произведена процедура авторизации;
- проверку прав пользователей на предмет доступа к конкретной подсистеме;
- протоколирование всех действий пользователей в специальной системной базе данных;
- возможность включения протокола шифрации SSL.

Комплекс программ успешно используется для поддержки таких информационных серверов как: ОАО «Белбизнесбанк», «Белорусские открытки», «Art-mission.com» и ряда других. В настоящее время идёт разработка корпоратив-

ной версии программного продукта, рассчитанной на работу на выделенном сервере. Среди особенностей новой программы: поддержка множества администраторов в режиме разметки структуры информационного сервера, возможность редактирования структуры сервера без специализированного программного обеспечения (через Интернет), прегенерация страниц непосредственно на Web-сервере (без необходимости пересылать по Интернету все изменённые страницы) и многое другое.

# ПРОЕКТ СИТУАЦИОННОГО ЦЕНТРА БЮДЖЕТИРОВАНИЯ КИС

Т.А. Ткалич

Кафедра информационных технологий, Белорусский государственный экономический университет, Партизанский пр., 26, Минск, 220672, БЕЛАРУСЬ, тел (80172) 249-19-81, tta@anitex. By

## АННОТАЦИЯ

Предлагается проект ситуационного центра по оценке затрат на функционирование корпоративной системы в соответствии с целями бизнеса и в условиях динамики рынка.

*Ключевые слова: бюджетирование КИС, учет затрат.*

## 1. ВВЕДЕНИЕ

Бурное развитие современных информационных технологий вовлекают современные предприятия разных масштабов в мир компьютерных информационных систем. Рынок информационных технологий (ИТ) и компьютерных систем является одним из наиболее интенсивно развивающихся рынков Как показывает проводимый известными компьютерными изданиями анализ рынка информационных технологий (газеты «Computer Weekly», «Компьютерные вести», «Мир ПК» и т.д.) рынок быстро насыщается, быстро обновляется, быстро изменяется цена технологий, средний срок обновления технических и программных средств составляет приблизительно 6-10 месяцев, а полное обновление рынка происходит в течение 5 лет.

Перед предприятиями встают вопросы, связанные с использованием множества разнородных приложений, программных средств, аппаратных платформ и сетевых протоколов. В таких условиях применение информационных технологий (ИТ) связано с использованием и заменой многочисленных разнородных и территориально распределенных вычислительных ресурсов, возникают проблемы с человеческим ресурсом, а также проблемы ведения бюджета информационной системы. Роль информационных технологий не всегда оценивается показателями повышения эффективности производства, это также могут быть неквалифицируемые показатели, отражающие цель приобретения компьютерной системы, которые оценивают повышение конкурентоспособности фирмы, goodwill, качество обслуживания и т.д.

В настоящее время для автоматизации деятельности предприятий создаются корпоративные информационные системы (КИС). Под КИС будем понимать информационную систему масштаба предприятия, главной задачей которой информационная поддержка производственных, административных и управленческих процессов, формирующих продукцию или услуги предприятия. Современные КИС разрабатываются на основе общепринятых западных методологий управления бизнесом (MRP,ERP, SCM) современных методологий управления предприятиями.

При плановой экономике в направлении исследования эффективности информационных технологий существовали целые направления и школы, которые занимались оценкой эффективности традиционных АСУ (до 1992г.), вопросы оптимизации АСУ рассматривались в работах В.М. Глушкова, А.А. Первозванского, Д.Б. Юдина, Е.Г. Гольштейна и других.

В настоящее время предлагаются новые направления оценка эффективности функционирования современных информационных систем. Ведущие западные фирмы-разработчики информационных систем разрабатывают стандарты (ESA, ISO 9000, IEC TC, IEEE 892) и специальные методологии оценки эффективности программных и технических средств (CMM, COCOMO и другие). Фирмы Oracle и SAB предлагают проводить оценку эффективности своих программных средств по совокупности разноплановых количественных и качественных показателей, характеризующих систему функционирования и эксплуатации по