УДК 681.3

М.Ю. КРУКОВСКИЙ

ПРОГРАММНЫЙ КОМПЛЕКС ПОДДЕРЖКИ СИСТЕМЫ КОМПОЗИТНОГО ДОКУМЕНТООБОРОТА НА ОСНОВЕ МОДЕЛЕЙ ПРОЦЕССОВ

Abstract: *This paper describes software package that is based on the author's theoretical base of composite document management. In this package there were used formal models based on graphs and automates and language based on JAVA. Implemented system allows to solve complex modern tasks of composite document management.*
Key words: *document management, docflow, workflow, graph model, automat model, document oriented language.*

Анотація: *У статті розглянуто програмний комплекс, побудований на основі авторської теоретичної бази композитного документообігу. Під час реалізації системи використовувались формальні моделі, основані на графах і автоматах, та мова програмування на основі JAVA. Реалізована система дозволяє вирішувати складні сучасні задачі композитного документообігу.*
Ключові слова: *електронний документообіг, процесне керування, графова модель, автоматна модель, документно-орієнтована мова.*

Аннотация: *В статье рассмотрен реализованный программный комплекс, основанный на разработанной авторской теоретической базе композитного документооборота. При реализации системы использовались формальные модели, основанные на графах и автоматах, и язык программирования на основе JAVA. Реализованная система позволяет решать сложные современные задачи композитного документооборота.*
Ключевые слова: *электронный документооборот, композитный документооборот, процессное управление, формальная модель документооборота, документно-ориентированный язык.*

1. Введение

В настоящее время актуальность задачи электронного документооборота обусловлена возросшим интересом пользователей информационных систем и значительно расширившейся компьютерной базой. Пользователи приходят к решению о необходимости реализации компьютеризированного документооборота под давлением постоянно усиливающихся информационных потоков. В то же время широкое распространение в организациях средств вычислительной техники становится удобной базой для решения таких распределенных многопользовательских задач, как документооборот.

По оценкам ведущей исследовательской компании Gartner Group, от 80% до 95% корпоративной информации хранится в документах. Та же компания отмечает, что до 25% корпоративных документов теряется либо приходит в такое состояние, что их дальнейшее использование невозможно. Таким образом, задача внедрения систем электронного документооборота является для организаций не только актуальной и своевременной задачей, но и экономически эффективной.

2. Постановка задачи

Целью настоящей статьи является описание системы документооборота, реализованной автором на основе предварительной теоретической разработки. В процесс работы были получены следующие теоретические результаты: концепция создания систем документооборота, методология построения формальной модели документооборота, определены и описаны модели документооборота на графах и автоматах, определены критерии эффективности. Кроме того, результатом теоретической работы явилось понимание того факта, что полноценную систему

документооборота, реализующую современные запросы организаций, можно сделать только на основе композитного подхода.

Композитный подход предполагает восприятие документооборота в виде двух составляющих: бумажной и электронной. Бумажный документооборот, являющийся обычным элементом организации человеческого общества, продолжает доминировать в решении формализованных управленческих задач, где требуется подтвержденная аутентификация решений. Электронный документооборот динамически развивается, значительно опережает бумажный документооборот по информативности и занимает все новые ниши, особо в сферах, где необходимо быстрое принятие решений. На пересечении использования электронной и бумажной технологий находятся все современные решения документооборота организаций. Понимание важности этой задачи решения привело к пониманию необходимости реализации программного комплекса, который бы отвечал такой постановке задачи.

3. Теоретические основы построения формальной модели процессов

3.1. Методология создания системы

При создании системы были использованы концепция и методология, изложенная автором в работе [1]. Основу этой концепции и методологии составляет следующая совокупность принципов автоматизации систем организационного управления [2], объектно-ориентированного проектирования [3] и управления проектами [4].

В соответствии с принципом автоматизации документооборота, являющимся продолжением принципов, предложенных В.М. Глушковым в [2], система спроектирована и разработана как активный элемент информационной системы. Это реализовано за счет адаптивности к требованиям пользователей системы, то есть система построена таким образом, что работает «на результат». В практическом смысле это значит, что во время движения документа происходят модификации маршрутов, форм документов и ролевых сценариев участников документооборота. Таким образом, система во время своего жизненного цикла сама себя модифицирует, адаптируясь к требованиям текущих пользователей.

Эта возможность, а именно адаптивность системы, позволяет безболезненное ее внедрение, поскольку способствует преодолению сопротивления персонала. Опыт внедрения систем документооборота говорит о том, что персонал оказывает значительное сопротивление нововведению, поскольку опасается увеличения объема и интенсивности работы, а также увольнения. В процессе адаптации к требованиям персонала система приспосабливается к привычной работе организации, что снижает сопротивление и способствует привыканию персонала.

Система построена на основе промышленной платформы систем документооборота Lotus Domino производства компании IBM. Такой выбор позволил положить в основу системы апробированные решения, которые проверены в практической эксплуатации миллионами пользователей во всем мире. Использование платформы Lotus Domino в качестве основы системы позволило сделать ее открытой к информационному взаимодействию с другими системами. В частности, позволяет реализовать обмен данными и действиями с другими системами

электронного документооборота. Платформа оснащена большим количеством шлюзов и поддерживает практически все современные и прошлые стандарты информационного обмена систем.

Описанный выше подход позволил создать систему в виде набора взаимодействующих программных модулей, функционирующих вокруг операционного ядра. Это позволило системно подойти к решению задачи документооборота. При этом задача документооборота декомпозирована на функциональный набор задач, которые представлены сценариями. Эти сценарии обрабатываются модулями системы как последовательно, так и параллельно. Центральное ядро системы обеспечивает синхронизацию процессов обработки, корректность исполнения сценариев и протоколов взаимодействия.

3.2. Формальная модель

Система поддерживает весь жизненный цикл создания формальной модели по цепочке «Моделируемый объект» -> «Общая теоретическая модель» -> «Конкретная теоретическая модель» -> «Модель». Различные модули системы предоставляют поддержку на каждом шаге создания формальной модели от исследования объекта до запуска в эксплуатацию готовой системы.

Система обеспечивает реализацию двух подходов к реализации документооборота. Первый подход называется детерминированным. В рамках этого подхода все процессы, происходящие в организации, инвентаризуются, описываются по последовательности исполнения, определяется входящая и исходящая информация. Упорядоченная совокупность процессов с детерминированными входами и выходами представляют систему документооборота организации.

Второй подход принято называть адаптивным. В рамках этого подхода система представляется в виде последовательности действий, работающих «на результат». Таким образом, при старте процесса не определены условия и триггеры завершения. Во время движения процесса происходит определение форм документов и назначение исполнителей на роли. Процесс может быть остановлен в любой момент времени, если ответственный за процесс примет решение о том, что необходимый результат достигнут.

В соответствии с теоретическими результатами, полученными в работе [5], система представляет процессы документооборота в виде тройки $Д_Т = \{У, Д, Ф\}$, где

$Д_Т$ – формальная модель документооборота;

$У$ – множество участников;

$Д$ – множество действий;

$Ф$ – множество состояний документов.

В этой тройке множество $У$ определяется как конечное множество ролей, которые могут быть назначены фактическим участникам документооборота. Множество $Д$ определяется как конечное множество действий, выполнение которых допустимо в пределах рассматриваемой системы документооборота. Множество $Ф$ – конечное множество состояний, которые могут принимать документы после выполнения участником множества $У$ действий из множества $Д$.

Взаимодействие элементов этих множеств и их связей полностью задает производственные сценарии, реализуемые системой документооборота.

На основе предложенной декомпозиции тройки множеств строятся две составляющие формальной модели документооборота: логическая и функциональная. Логическая модель определяет событийную составляющую системы документооборота. При этом описываются действия, которые происходят в системе, и декларируются моменты времени или условия событий, после которых эти действия будут выполнены. Логическая модель описывается с помощью теории графов. Графы также являются основой представления процессов, которые хранятся и воспроизводятся описываемой системой документооборота.

Функциональная модель представляет собой описание системы на языке выполняемых ею функций. Это представление формальной модели позволяет описать и воспроизвести работу системы с точки зрения последовательности производимых действий и получаемых результатов. Результаты, представленные состояниями документов, выстраиваются в последовательность изменяемых состояний. Это позволяет представить документооборот в виде конечного автомата, который оперирует документами в виде алфавита и действиями участников, представленными в виде функций перехода автомата.

3.2.1. Графовая модель системы

Логический уровень формальной модели документооборота реализован с использованием аппарата теории графов. Теоретические исследования, позволившие реализовать данный подход в системе документооборота, были проведены автором в работе [6]. Предложенный подход представления модели в виде совокупности графов был практически реализован в программном комплексе и является основой для хранения данных об архитектуре моделируемой системы документооборота.

Таким образом, система использует существующий аппарат теории графов для решения практических задач документооборота. Модуль системы, отвечающий за проектирование процессов (архитектор процессов), позволяет проектировать и хранить последовательности процессов в виде графов. Двигатель процессов воспринимает представленную библиотеку графов и обеспечивает исполнение процессов в соответствии с определенными графами. Визуализатор, модуль системы, отвечающий за визуальное отображение динамики процессов, отслеживает модификацию активных процессов и отображает их в виде графов.

Такое представление и использование процессов на языке графов позволяет не только проектировать, исполнять и хранить процессы, но и осуществлять операции над ними как операции над графами. Например, с помощью таких графовых операций можно проектировать более сложные процессы как совокупность простых процессов, выделять необходимый участок процесса в отдельный процесс и аналогичные операции.

В использованной формальной модели для представления графа документооборота принято описание вида $G = (V, E, \Gamma)$, где V – множество вершин графа; E – множество ребер графа; Γ – множество отношений инцидентности. Граф G состоит из непустого множества элементов, называемых вершинами; множества связанных пар из множества вершин, называемых

ребрами; множества признаков направленности ребер. Множество, состоящее из вершин графа G, называется множеством вершин графа и обозначается $V(G)$. Аналогично, множество, состоящее из ребер, называется множеством ребер и обозначается $E(G)$. Если v и w являются вершинами графа G, тогда ребром vw называется связь, которая соединяет v и w.

Формальная модель документооборота, актуализированная множествами $\{У\}$, $\{Д\}$ и $\{Ф\}$, отображается на графе документооборота таким образом, чтобы при отождествлении выполнялись следующие правила:

– одной вершине графа соответствует один и только один элемент множества $Ф$;

– одному ребру графа соответствует один и только один элемент множества $Д$;

– одному элементу множества $Ф$ соответствует одна и только одна вершина графа;

– одному элементу множества $Д$ соответствует одно и только одно ребро графа.

Математически такое тождественное отображение множества состояний $\{Ф\}$ в множество вершин $\{V\}$ и множества состояний $\{Д\}$ в множество ребер $\{E\}$ описывается следующим образом:

для любого $i \in I$, i=1,2,3.. n справедливо утверждение $v(i) = Ф(i)$ и $e(i) = Д(i)$ при $i \in I$, i=1,2,3.. n.

Для описания систем документооборота используются как неориентированные, так и ориентированные графы. Выбор вида графа определяется моментом технологического процесса, на котором происходит его использование. На разных этапах жизненного цикла создания документооборота эффективны различные типы графов, что и определяет их выбор.

Неориентированные графы используются на этапах анализа и проектирования для наглядного отображения данных, полученных при обследовании. После опроса пользователей, выявления критических требований участников и декомпозиции документов до приемлемого уровня происходит констатация связности полученных данных. Эти первичные данные сохраняются в системе в виде множества неориентированных графов, которые соединяют полученные элементы множеств. Система сохраняет эту предварительно выбранную архитектуру системы в библиотеке шаблонов и позволяет не только дорабатывать процессы на последующих этапах, но и повторно использовать полученные шаблоны.

Ориентированные графы используются на этапах проектирования, реализации, внедрения и разработки. Во время разработки системы документооборота на этих этапах происходит установление направленности связей, выявленных на предыдущих этапах. То есть, если во время анализа был просто установлен факт наличия связи, то на этапе реализации происходит определение направленности связности. На практике это означает, что архитектор системы, используя предварительные («сырые») данные, детерминирует направленность движения документов и последовательность ролевых сценариев. При этом действия, необходимые для продвижения документа от начального состояния к конечному упорядочиваются, и у каждого из них выходные данные являются входными данными другого действия.

Данные формальной модели, представленные тремя множествами, хранятся в виде матриц, которые описывают связность состояний документов, действий участников и ролевых сценариев. Матричная форма хранения данных полностью отображает представление системы документооборота в виде множеств $\{У\}\{Д\}\{Ф\}$.

Для хранения используется множество плоских прямоугольных матриц документооборота, упорядоченных в коллекции. Каждая из матриц представляет состояние системы документооборота на некотором дискретном шаге. Столбцы матрицы документооборота ставятся в соответствие состояниям документов таким образом, чтобы первый столбец соответствовал первому элементу множества $\{Ф\}$, второй столбец – второму элементу и так далее до последнего элемента множества $\{Ф\}$. Строки матрицы документооборота ставятся в соответствие действиям, произведение которых приводит к изменению состояния хотя бы одного документа. Первая строка соответствует первому элементу множества $\{Д\}$, вторая строка – второму элементу множества и так далее для всего множества $\{Д\}$.

Таким образом, получается множество прямоугольных матриц со столбцами, количество которых равно размерности множества $\{Ф\}$ и строкам матрицы $\{Д\}$. Заполняется данная матрица элементами множества участников $\{У\}$. Элемент заполняется как элемент матрицы в том случае, если соответствующий участник производит действие, соответствующее элементу строки. В том случае, если на этом шаге документооборота действие строки не изменяет состояние столбца, то элемент матрицы заполняется нулевыми значениями.

Формальная модель, которая хранит данные в виде графов, позволяет производить операции над данными. А именно – использовать следующие операции: объединение, пересечение, разность и произведение. Эта возможность системы позволяет архитектору системы документооборота создавать новые системы при помощи существующих апробированных решений.

3.2.2. Автоматная модель системы

Автоматная модель композитного документооборота была получена автором при работе над теоретическими основами документооборота и подробно описана в работе [7]. Полученные теоретические результаты были положены в основу описываемой системы документооборота и стали основой разработанного программного обеспечения. Практические результаты показали не только пригодность автоматной модели для реализации практических задач документооборота, но и дали возможность сделать вывод о целесообразности расширения применения теории автоматов при решении задач документооборота.

С помощью разработанной модели представлено движение документов по процессам, а также взаимодействие между собой моделей документооборота различных организаций. Такой подход позволяет программно реализовать архитектурный, компонентный и семантический уровни формальной модели.

В реализованной автоматной модели моделируемая система документооборота представляется в виде детерминированного конечного автомата, заданного общепринятой нотацией конечного автомата. В соответствии с автоматной нотацией документооборота, предложенной автором в работе [7], автомат представляется следующим образом:

$$Д_T = (Q, \Sigma, \delta, q_0, F),$$

где Q – конечное множество состояний документов, тождественное множеству $\{\Phi\}$ из определения композитного документооборота;

Σ – конечное множество входных символов, образующих входной алфавит и представляющее собой данные, которые поступают на вход системы документооборота;

δ – функция переходов, аргументами которой являются текущее состояние и входной символ, а значением – новое состояние;

q_0 – начальное состояние (или множество начальных состояний) из множества Q;

F – множество заключительных или допустимых состояний из множества Q.

Множество состояний автоматов $\{Q\}$ получается из множества состояний документов, полученное при проектировании формальной модели. Завершающие состояния выделяются из общего множества состояний путем анализа каждого из состояний. Если при анализе выявляется состояние, которое имеет одну или несколько входящих связей, но не имеет ни одной исходящей, то оно помечается как завершающее. Состояния моделирующего автомата упорядочиваются таким образом, чтобы документооборот был представлен состояниями документа в порядке от начального состояния документа к конечному состоянию.

Автомат исполняет функции переходов для принятия решения о выборе следующего состояния. Функции переходов программируются с помощью анализа действий участников документооборота. Производимое действие определяет результирующее состояние, для которого входными данными для определения выбора являются текущее состояние документа и участник процесса. Автомат реализует документооборот, в котором на каждом шаге происходит действие, на основании процесса и анализа текущего состояния документа (исполнителя) принимается решение о следующем состоянии документа. Функция перехода F_i автоматной модели является i-м элементом множества действий $\{Д\}$ документооборота, после выполнения которого происходит смена состояния s_i на состояние s_{i+1}.

В реализованной автоматной модели документооборота в качестве алфавита автомата использован список участников. Символами обозначены ролевые участники производственных сценариев. Из этих символов образуются последовательности, которые обрабатываются автоматом формальной модели. Последовательности символов, обрабатываемые автоматом документооборота, являются допустимыми для модели. Такие последовательности символов приводят к корректному выполнению процесса документооборота и получению конечного результирующего документа. Последовательности символов документооборота, которые не принимаются автоматом, реализующим модель документооборота, являются недопустимыми для

моделируемого процесса. Последовательности, принимаемые автоматом модели документооборота, являются его языком. Слова этого языка являются возможными последовательностями участников документооборота, участвующих в процессе работы над документами.

Программирование автомата осуществляется путем установления соответствия между нотацией конечного автомата и композитного документооборота. То есть после проведения декомпозиции процессов и синтеза модели $\{У, Д, Ф\}$ строится автоматная модель документооборота, которая определяется пятеркой (A, S, s_0, T, F), где $\{\{A\} \equiv \{У\}, \{S\} \equiv \{Ф\}$, $s_0 = ф_0$, $\{S\} \equiv \{Ф_к\}\}$, $\{F\} \equiv \{Д\}\}$. При этом на каждом шаге автомата множества, определяющие его состояния, соответствуют множествам, описывающим жизненный цикл модели документооборота. На каждом шаге существует соответствие A_1^i и $Дт_1^j$: $\forall i, j \subset N \cap i = j \Rightarrow$ $\{(s_0 = ф_0), (s_i = ф_j), A_i = У_j, F_i = Д_j\}$.

3.2.3. Язык документооборота

Во время создания описываемой системы документооборота автор пришел к выводу о необходимости разработки специализированного языка [8]. Наличие такого языка позволило сузить задачи программирования до решения прикладных задач, что дало возможность уйти от общесистемных вопросов реализации и развития системы.

Для решения задачи создания документно-ориентированного языка использовалась модификация языка JAVA для реализации сложных приложений распределенных предприятий J2EE (JAVA 2 Enterprise Edition). Такой выбор связан с тем, что язык JAVA уже использовался при разработке системы документооборота и его расширение позволило реализовать язык в виде дополнения к уже созданному, как отмечалось выше, программному инструментарию. Язык назван GJE – Graph JAVA Extension.

Язык GJE сделан в виде расширения к существующему и широко используемому языку JAVA. При создании языка предполагалось, что сделанные на этом языке приложения будут использоваться во всех трех существующих реализациях конечных приложений на JAVA. А именно – локальных приложениях Application, дистанционно исполняемых приложениях Applet и тиражируемых сервером приложениях Servlet.

Язык GJE представляет собой внешнее расширение пакетов JAVA для решения задач документооборота и описан как package javax.workflow. Это делает язык открытой средой разработки и дает возможность внешним разработчикам подключать пакет к своим приложениям. Различные приложения на языке GJE позволят достичь распределенного использования и дадут развитие существующей библиотеке.

Язык GJE позволяет строить модели документооборота, основанные на графовой модели документооборота, которая описана в [7] п.3.2.1. Поэтому при описании классов дается не только общее назначение методов и данных с точки зрения графа как математического понятия, но и применение содержания классов, введенное в модели документооборота. В табл. 1 приведено описание основных классов языка GJE с описанием реализованных методов.

Таблица 1. Описание основных классов языка документооборота GJE

Название	Описание класса	Реализованные методы
Node	Описывает состояния документов, которые отображаются вершинами графа	Object getValue(); void setValue(Object value)
Edge	Описывает действия участников документооборота, которые отображаются ребрами графа	Node getInPoint(); Node getOutPoint(); Object getDirection(); void setDirection(Object direction); void setValue(Object value

Graph	Объединяет функциональность классов и Node Edge. Обеспечивает хранение информации о графе в виде совокупности вершин и ребер. Реализует функцию депозитария процессов. Реализовано управление описанием процесса: добавление, удаление и корректировка данных о документах, действиях и участниках	Collection getNodes();
		Collection getEdges();
		Node createNode(Object value);
		Edge createEdge(Node in, Node out, Object direction, Object value);
		void deleteNode(Node node);
		void deleteEdge(Edge edge);
		String getName();
		Void setName(String name)

HyperGraph	Является высшим классом в иерархии GJE. Объединяет в себе другие классы и операции над ними. Реализует представление совокупности графов в коллекции в виде единого целого без потери свойств и характеристик. Является хранилищем процессов и инструментарием для их обработки. Инструментально реализована алгебра документооборота. На данный момент доступны следующие операции над процессами: добавление, удаление, объединение, пересечение, вычитание и умножение	Collection getGraphs();
		void addGraph(Graph graph);
		void deleteGraph(Graph graph);
		Graph unionGraph(Graph graph1, Graph graph2);
		Graph intersectionGraph(Graph graph1, Graph graph2);
		Graph differenceGraph(Graph graph1, Graph graph2);
		Graph cartesianGraph(Graph graph1, Graph graph2);
		Graph createGraph(Collection nodes, Collection edges)

Таким образом, язык GJE представляет собой набор расширений языка JAVA, которые использованы для создания системы проектирования и исполнения системы композитного документооборота. Язык позволяет обеспечить локальное и сетевое взаимодействие между процессами документооборота. Кроме того, является архитектурно открытым и позволяет разработчику развивать его в соответствии с текущими требованиями заказчика.

4. Практическая реализация системы поддержки композитного документооборота

Наличие развитой теоретической базы требовало ее апробации, вследствие чего был разработан программный комплекс системы композитного документооборота. Комплекс прошел испытания и на данный момент используется в ряде организаций Украины. Эксплуатация показала, что система достаточно эффективна как в малых, так и в больших организациях. Система оказалась эффективной как для реализаций локальных решений малых организаций, так и для распределенных решений больших организаций, оперирующих адаптивными сценариями.

4.1. Архитектура системы

Программный комплекс построен в виде комплекса взаимодействующих приложений. Каждое приложение имеет функциональное назначение. При создании конкретной системы поддержки документооборота происходит комплектация комплекса, компоновка модулей и адаптация в соответствии с техническим заданием к требованиям пользователя. Модули имеют интерфейсы, которые можно использовать для взаимодействия с другими модулями комплекса, а также для взаимодействия с внешней средой. Комплекс является гетерогенным, то есть в качестве внешних

приложений можно использовать как приложения базовой среды Windows, так и приложения других операционных сред, поддерживающих протоколы сетевого взаимодействия.

Комплектация комплекса является практически избыточной по отношению к большинству встречаемых на практике задач документооборота. Это сделано преднамеренно для того, чтобы покрыть максимальное количество пользовательских потребностей и сделать комплекс минимально требовательным при адаптации к конкретным требованиям пользователя.

На рис. 1 приведена архитектурная схема программного комплекса.

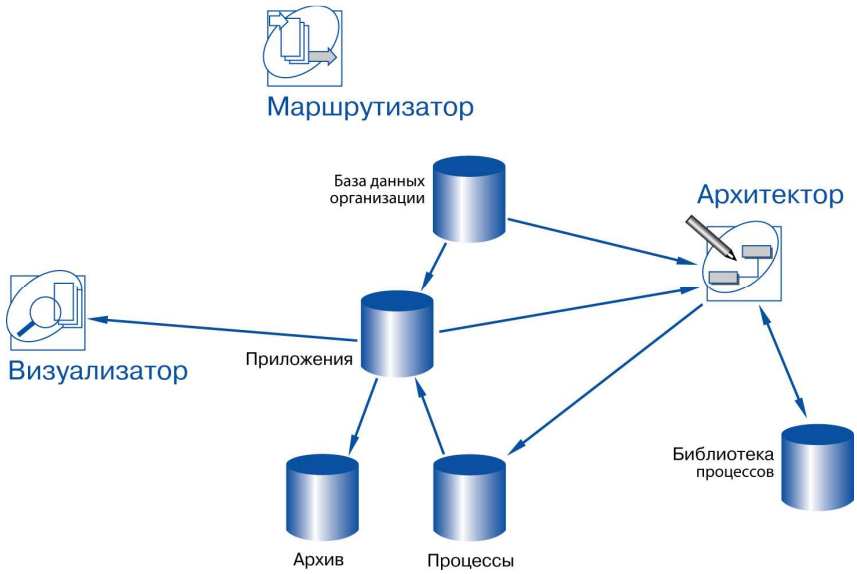

Рис. 1. Архитектура программного комплекса

На схеме отображены три основных модуля программного комплекса, которые выполняют функциональные задачи: маршрутизатор процессов, визуализатор движения документов и архитектор процессов и форм.

Маршрутизатор процессов предназначен для реализации действий, производимых участником документооборота над документами. После того, как в документе появляется отметка о том, что участник принял решение о произведении действия, документ переводится в рабочее состояние, в котором он доступен только маршрутизатору. Маршрутизатор постоянно производит ревизию активных процессов и выявляет процессы, которые надо обработать. После выявления такого процесса маршрутизатор производит задекларированное действие, переводит документ в результирующее состояние и делает его доступным следующему участнику.

Визуализатор движения документов обеспечивает доступ руководящих лиц организации к состоянию текущих активных процессов. Визуализатор позволяет увидеть актуальное состояние всего множества процессов и составить адекватное представление о состоянии исполнительской дисциплины в организации.

Архитектор процессов предназначен для проектирования сценариев движения документов и форм документов. Сценарии движения реализованы в виде набора графов в соответствии с описанием модели документооборота на графах, предложенной автором в работе [6]. Такая реализация позволяет задействовать одно из основных достоинств теории графов – наглядность представляемой информации. Формы документов получаются в результате декомпозиции жизненного цикла документа на набор состояний. Каждое состояние представляется в виде отдельной формы и кодируется набором полей, содержащих текстовую, аудио- и видеоинформацию. Между набором форм устанавливается связь, что обеспечивает полноту описания документа.

5. Апробация программного комплекса на практических задачах

Реализованный программный комплекс был применен во время практической реализации информационных систем организаций Украины. В зависимости от конкретной потребности в системах использовались различные модули комплекса. По приблизительной оценке на сегодняшний день системы, реализованные на основе комплекса композитного документооборота, обрабатывают около 40 миллионов документов в год.

5.1. Укрпочта

В УППС «Укрпочта» разработанный документооборот внедрен для реализации задач комплекса АСРК-РП. Программный комплекс АСРК-РП предназначен для отслеживания ценных почтовых отправлений, обращающихся в Украине. Каждое почтовое отправление, которое оценивается отправителем как ценное, проходит соответствующую технологическую обработку и получает сопровождающие реквизиты. По этим реквизитам происходит отслеживание отправления во время его следования до пункта получения или долговременного хранения в случае отсутствия адресата.

По результатам использования внедренной системы документооборота УППС «Укрпочта» приняла официальное решение о перспективности использования системы при развитии предприятия и реализации других задач.

5.2. НАК «Нефтегаз Украины»

При внедрении системы управления персоналом в НАК «Нефтегаз Украины» был использован ряд модулей разработанной системы документооборота: двигатель процессов и дизайнер маршрутов. Система предназначена для обработки задач управления более чем тридцатитысячным персоналом компании. В решаемые задачи входят планирование потребностей в персонале, подбор персонала, планирование отпусков, табелирование, учет отпусков, отсутствий и прочие задачи кадрового документооборота предприятия.

Использование системы показало высокое качество реализации и глубокую проработку специфики задач кадрового документооборота. По результатам внедрения руководство организации приняло решение об использовании системы документооборота в будущих решениях.

5.3. НАСК «Оранта»

В НАСК «Оранта» была внедрена комплексная система управленческого и делового документооборота, которая позволяет решать задачи оперативного сопровождения деятельности компании и документарной поддержки деловых процессов. При реализации системы использовались модули разработанной системы документооборота, которые реализовывают адаптивную схему документооборота. Внедрение системы позволяет обрабатывать около 20 000 документов в год теми же сотрудниками и теми же средствами вычислительной техники, которыми обрабатывалось около 10 000 до внедрения СЭД. По оценкам компании, обработка одного документа стоит около 5 гривен, что дает экономический эффект до 50 000 гривен в год.

5.4. СК «Эталон»

В СК «Эталон» внедрена система оперативного документооборота для документальной поддержки урегулирования убытков. При возникновении страхового случая инспектор фотографирует ситуацию, добавляет свои комментарии и отправляет главному инспектору. В головной конторе задачи распределяются между главными инспекторами стохастическим образом в соответствии с

текущей нагрузкой. Главный инспектор принимает решение о выплате или отказе и через юридическую службу отправляет ответ клиенту.

При реализации системы был использован разработанный модуль обработки и движения предопределенных процессов. Этот модуль обрабатывает заявки, которые получаются из распределенной сети страховой компании, после чего обеспечивает доведение задачи до установленного участника.

5.5. ЗАО «УМС»

В ЗАО «УМС» внедрена система, реализующая оперативный документооборот, связанный с распределенной технической поддержкой пользователей. В системе используется модуль, обеспечивающий обработку и продвижение процессов поддержки. При возникновении обращения пользователя по поводу технической поддержки оператор, принимающий звонок, инициирует поиск технического специалиста с кратким описанием проблемы. Система реализует доставку документа и оповещение всем техническим специалистам, которые обладают достаточной квалификацией для решения проблемы. Оператор выбирает одного из доступных специалистов и передает ему более подробное описание проблемы. В процессе происходит обмен рекомендациями, вплоть до организации «общей доски» между специалистом и оператором и прямой голосовой связи между техническим специалистом и пользователем.

Внедрение системы позволило повысить исполнительскую дисциплину, минимизировать личностные конфликты при решении технических проблем, начать накопление базы знаний о типичных проблемах и методах их решения.

6. Выводы

1. Описанная система композитного электронного документооборота позволяет решать задачи значительного объема и сложности.

2. Система интегрирует теоретические результаты, полученные автором ранее [1, 5–8], и позволяет реализовать поддержку композитного документооборота, то есть такого, в котором есть и электронная, и бумажная составляющие процессов.

3. Использование системы позволяет комплексно решать задачи автоматизации документооборота. Реализованная система позволяет совместно использовать два ранее несовместимых решения, а именно: в системе можно реализовать как детерминированные решения на описанных процессах, так и адаптивные, принимающие стохастические неопределенные решения о движении документов на каждом шаге документооборота.

СПИСОК ЛИТЕРАТУРЫ

1. Круковский М.Ю. Методология построения композитных систем документооборота // Математичні машини і системи. – 2004. – № 1. – С. 101–114.
2. Глушков В.М. Введение в АСУ. – К.: Техніка, 1972. – 312 с.
3. Booch Grady. Object-Oriented Analysis and Design with Applications. – Third Edition.-NY.: Addison-Wesley, 1994. – 589 p.
4. Project Management Institute. A guide to the Project Management Body of Knowledge. – 2000.
5. Круковский М.Ю. Концепция построения моделей композитного документооборота // Математичні машини і системи. – 2004. – № 2. – С.149–163.
6. Круковский М.Ю. Графовая модель композитного документооборота // Математичні машини і системи. – 2005. – № 1. – С. 120–136.
7. Круковский М.Ю. Автоматная модель композитного документооборота // Математичні машини і системи. – 2004. – № 4. – С. 37–49.
8. Круковский М.Ю. Язык обработки графов на базе JAVA // Математичні машини і системи. – 2005. – № 2. – С. 45–53.