УДК 681.3

М.Ю. КРУКОВСКИЙ

АВТОМАТНО-ГРАФОВАЯ ФОРМАЛЬНАЯ МОДЕЛЬ КОМПОЗИТНОГО ДОКУМЕНТООБОРОТА

Abstract: *This paper describes approach of usage finite states machine for creating a system of of FSMs that are connected by graphs. Document management system implemented as a SM that operates with FSMs as a behavior models. In the paper covered issues of document management implementation based on hierarchical finite state machine.*
Key words: *electronic document management, composite docflow, workflow, formal model, automat model, graph model, hierarchical finite state machine.*

Анотація: *У статті розглянуто застосування скінчених автоматів для створення системи автоматів, зв'язаної графами. Документообіг представлено у вигляді автомата, який оброблює автомати, кожен із них моделює поведінку одиниці системи документообігу. У статті розглянуто моделювання складних систем документообігу за допомогою ієрархічного скінченого автомата.*
Ключові слова: *електронний документообіг, композитний документообіг, процесне керування, графова модель, формальна модель документообігу, автоматна модель, графова модель, ієрархічний скінчений автомат.*

Аннотация: *В статье рассмотрено применение конечных автоматов для создания системы автоматов, связанных графами. Документооборот представляется в виде автомата, обрабатывающего автоматы, каждый из которых моделирует поведенческую единицу системы документооборота. В статье рассмотрено моделирование сложных систем документооборота с помощью иерархического конечного автомата.*
Ключевые слова: *электронный документооборот, композитный документооборот, процессное управление, формальная модель документооборота, автоматная модель, графовая модель, иерархический конечный автомат.*

1. Введение

В современном обществе идет процесс интенсификации вычислительных и информационных технологий во всех отраслях деятельности [1]. Внедрение электронного документооборота является актуальной задачей современного общества. Внедрение систем электронного документооборота позволяет сделать процесс движения документов управляемым и контролируемым, что обеспечивает более качественные услуги управления.

При решении задач электронного документооборота большое значение имеет использование формальных моделей в проектировании, разработке и внедрении СЭД. Применение формальных моделей позволяет специалистам оперировать измеримыми объектами, к которым может быть применен апробированный математический аппарат.

В отечественной и зарубежной науке существуют примеры применения теории графов и теории автоматов как самостоятельных формальных моделей документооборота. Одной из первых модель документооборота с применением теории графов была предложена в работе [2]. Зарубежные ученые рассматривали применение автоматов для моделирования информационного взаимодействия рабочих групп [3]. Другими авторами автоматы были использованы для моделирования связной системы взаимодействующих процессов при решении задач процессного управления [4]. Автор статьи предложил формальную модель документооборота, построенную на графах [5] и автоматах [6].

2. Постановка задачи

Целью настоящей статьи является обобщение полученных автором в работах [5, 6] результатов и создание автоматно-графовой формальной модели электронного документооборота, использующей аппарат теории графов и теории автоматов.

Теория графов позволяет отразить связность формальной модели, наглядным образом представить установленные связи между участниками документооборота. Аппарат графов дает возможность описывать взаимодействия участников и документов с помощью матрицы инцидентности.

В свою очередь, теория автоматов позволяет реализовать логику ветвления движения документов между участниками процессов документооборота. Автомат позволяет установить реакцию элементов системы документооборота на изменения в системе. С помощью автоматов процессы документооборота могут быть представлены в виде элементов с предсказуемым поведением и описанными интерфейсами.

Таким образом, при применении теории графов и теории автоматов к формальной модели документооборота появляется возможность представить документооборот в виде автоматов, связанных графами. Каждый автомат представляет собой поведенческую единицу системы документооборота, которая обладает своим поведением. Результаты работы автоматов поступают на вход другим поведенческим элементам системы документооборота, которые также реализованы автоматами. Связность и направленность передачи выходных результатов в качестве входных алфавитов определяется графом документооборота.

Приведенное выше описание модели представляет собой конечный автомат (КА), который, в свою очередь, оперирует автоматами. Входной и выходной алфавиты представляют собой КА, который реализует поведенческие единицы системы документооборота. Функция переходов задается с помощью графа, определяющего связность поведенческих единиц. Такой автомат в литературе [7] называется иерархическим конечным автоматом (Hierarchical Finite State Machine).

В следующем разделе рассмотрим описанную выше модель более подробно.

3. Синтез автоматно–графовой формальной модели

Адаптируем описанный выше математический аппарат для создания формальной модели композитного документооборота. Для решения этой задачи представим документооборот в виде связанной последовательности процессов, протекающих в дискретном временном пространстве.

3.1. Формальная модель документооборота

Для моделирования документооборота будем использовать формальную модель композитного документооборота, предложенную автором в работе [8]. В этой модели процесс документооборота может быть представлен в виде трех конечных множеств и связей элементов этих множеств между собой. Математическая нотация представлена в виде тройки $Д_T = \{У, Д, Ф\}$,

где $Д_T$ – формальная модель документооборота;

$У$ – множество участников;

$Д$ – множество действий;

$Ф$ – множество состояний документов.

Нотация означает следующее: «Документооборот – это множество действий, производимых множеством участников над множеством состояний документов». 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 форм $Ф$ – конечное множество состояний, которые могут принимать документы после произведения над ними действий из множества $Д$ участником из множества $У$.

Таким образом, поведение участника документооборота может быть представлено в виде последовательности состояний документов. Совокупность всех состояний документов представляет конечное множество, которое полностью описывает все возможные сценарии поведения участников.

3.2. Модель связности поведенческих единиц на графах

Для построения графа модели документооборота предлагается использовать способ отображения документооборота графами, предложенный автором в работе [5]. Для задания множества вершин графа будем использовать множество возможных состояний документов $Ф$. Ребра графа зададим с помощью множества действий $Д$. Установим это соответствие таким образом, чтобы 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.

Такое тождественное отображение множеств состояний $Ф$ в множество вершин v и множества состояний $Д$ в множество ребер e можно математически определить следующим образом: для любого i справедливо утверждение $v(i) = Ф(i)$ и $e(i) = Д(i)$, где i $i \in I, I = 1,2,3...n$. То есть определяются две парных грамматики – первая грамматика для установления перевода $Ф$ в v, вторая грамматика – для установления перевода $Д$ в e.

Таким образом, связи между вершинами тождественно соответствуют связям состояний моделируемого документооборота. В графе документооборота вершины графа соединяют ребра в том и только в том случае, если соответствующие вершинам состояния связаны действием, соответствующим ребру, то есть $e = \begin{Bmatrix} e(i), если\_ребро\_существует \\ 0, если\_ребро\_отсутствует \end{Bmatrix}$.

Направленность ребер устанавливается таким образом, чтобы отображать логику последовательности смены состояний документооборота. Вершина i является входящей вершиной для вершины j через ребро k в том и только в том случае, если состояние i сменяется на состояние j после совершения действия k. Таким образом, состояниям y_1, y_2,...y_n

сопоставляются вершины графа $v_1, v_2, ... v_n$, и каждая пара вершин v_i и v_j соединена дугой e_{ij}, идущей от v_i к v_j в том и только в том случае, когда состояние v_i является входным состоянием для v_j.

3.3. Модель поведенческой единицы на автоматах

В предложенной автором автоматной модели документооборота [6] система представляется в виде детерминированного конечного автомата. Автомат, представляющий поведенческую единицу документооборота, задается общепринятой нотацией конечного автомата. В соответствии с нотацией автомат представляется следующим образом:

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

где Q – конечное множество состояний документов, которое тождественно множеству Φ, что следует из определения композитного документооборота;

Σ – конечное множество входных символов, образующих входной алфавит и представляющих 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 документов) из множества Q;

F – множество завершающих или допускающих состояний из множества Q.

Множество состояний автоматов Q получается из множества состояний документов. 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 завершающему.

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 на основании 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 алфавита обозначены ролевые участники

производственных сценариев. Из этих символов образуются последовательности, обрабатываемые автоматом формальной модели. Последовательности символов, которые обрабатываются автоматом документооборота, являются допустимыми для модели. Такие последовательности символов приводят к корректному выполнению процесса документооборота и получению конечного результирующего документа. Последовательности символов, которые не принимаются автоматом, реализующим модель документооборота, являются недопустимыми для моделируемого процесса. Последовательности символов, принимаемые автоматом модели документооборота, являются его языком. Слова в этом языке 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 $\{A \equiv У$, $S \equiv Ф$, $s_0 = ф_0$, $S \equiv Ф_k$, $F \equiv Д\}$. При этом на каждом шаге автомата, множества соответствуют множествам, описывающим жизненный цикл модели документооборота. На каждом шаге существует соответствие A_1^i и $Дm_1^j : \forall i, j \subset N \cap i = j \Rightarrow \{\{s_0 = ф_0), (s_i = ф_j), A_i = У_j$, $F_i = Д_j\}$.

3.4. Иерархический конечный автомат

Для представления иерархического конечного автомата документооборота будем использовать описание ИКА в символах работы [9]. В соответствии с этим описанием ИКА представляется пятеркой (I, O, S, T, r), где I и O описывают множества входных и выходных алфавитов, S представляет множество состояний, функции переходов $T : S \times B^{|I|} \times B^{|O|} \times S \to B \mid B = \{0,1\}$, r задает начальное состояние. Детерминированный ИКА может быть представлен шестеркой (I, O, S, λ, δ, r), где $\lambda : S \times B^{|I|} \to B^{|O|}$ описывает выходную функцию, $\delta : S \times B^{|I|} \to S$ описывает функции переходов.

О данной паре входных и выходных последовательностей (σ_i, σ_o), где $\sigma_i = (i_o, ..., i_{t-1})$ и $\sigma_0 = (o_o, ..., o_{t-1})$,говорят, что она принимается ИКА $T = (I, O, S, T, r)$, если существует последовательность состояний $(s_0, ..., s_t)$ такая, что $T(s_j, i_j, o_j, s_{j+1}) = 1$для всех $j = 0, ..., t-1$ и $s_0 = r$.

В настоящей статье будем рассматривать поведение ИКА, определенного отношением элементов входа и выхода. Иными словами, отношение между алфавитами входа и выхода есть набор пар входов и выходов, которые определяют состояния детерминированного ИКА. Для заданного автомата $T = (I, O, S, T, r)$ поведение между входом I и выходом O содержится в функциях переходов T, если каждая пара последовательности входов и выходов реализуется в T. Рассмотрим реализацию ИКА, управляющего конечными автоматами, на примере на рис. 1.

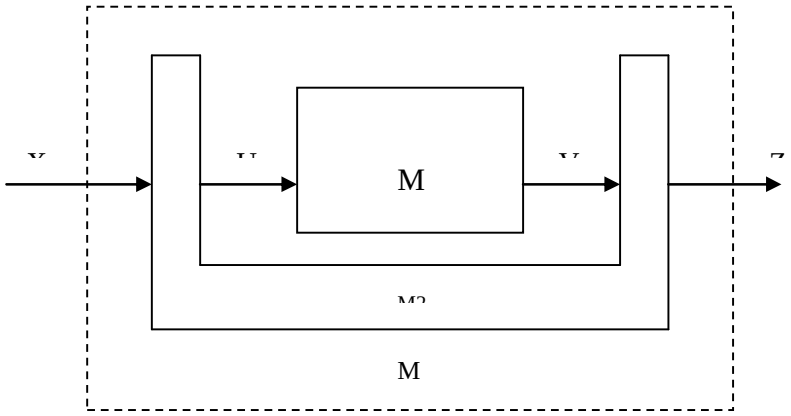

Рис. 1. Пример иерархического конечного автомата

Итак, заданные автоматы $M = (I, O, S, T_M, r)$ и $M2 = (X \cup V, U \cup Z, S_2, (\lambda_2^{(u)}, \lambda_2^{(z)}), \delta_2, r_2))$. Предполагается, что система документооборота может принимать сразу несколько состояний, в то время как один исполнитель производит смену состояния только на одно из возможных. Таким образом, автомат M может быть НДКА, в то время как автомат $M2$ может быть только ДКА. Выходная функция λ_2 автомата $M2$ состоит из $\lambda_2^{(u)} u \lambda_2^{(z)}$, которые соответственно определяются выходными функциями U и Z.

Заданы подмножество s^* из множества состояний S и вход x автомата M, зададим $\Lambda(s^*, x)$ как множества всех возможных выходов. То есть $z \in \Lambda(s^*, x)$ в том и только том случае, если существует $(\tilde{s}, s) \in s^* \times S$ такое, что $T_M(\tilde{s}, x, z, s) = 1$. Аналогично, $\Delta(\tilde{s}, x)$ будет множеством всех возможных состояний, то есть $s \in \Delta(\tilde{s}, x)$ в том и только том случае, если $\exists(\tilde{s}, z) \in s^* \times B^{|z|}$ такое, что $T_M(\tilde{s}, x, z, s) = 1$.

В рамках определения иерархического конечного автомата, который реализует комплексную систему документооборота, рассмотрим реализуемость и допустимость возможных моделей документооборота. Рассмотрим возможные поведения ДКА, которые будут допустимы на автомате $M1$. Кроме того, рассмотрим реализации сочетаний поведенческих единиц КА, которые будут реализуемы с помощью ИКА.

При заданном автомате $M2 = (X \cup V, U \cup Z, S_2, \lambda_2, \delta_2, r_2)$ детерминированный конечный автомат $M1 = (V, U, S_1, \lambda_1, \delta_1, r_1)$ считается реализуемым на $M1$, если существует хотя бы одна пара цикличных реализаций $M1$ и $M2$, таких, что их соединение не вызывает цикла между U и V.

При заданных автоматах $M = (X, Z, S, T_M, r)$ и $M2 = (X \cup V, U \cup Z, S_2, \lambda_2, \delta_2, r_2)$ ДКА $M1 = (V, U, S_1, \lambda_1, \delta_1, r_1)$ является допустимым автоматом, если автомат $M1$ реализуем и поведение $M1 \times M2$ содержится в M, где $M1 \times M2$ является выходным результатом M. Поведение, которое реализуется допустимым автоматом, является допустимым поведением.

3.4.1. Свойства ИКА

Рассмотрим применение НДКА для реализации ИКА. Пусть при данном $S^0 = \{(r_2, \{r\})\}$ надо получить T^{t+1} и S^{s+1} для заданного $S^{(t)} \subseteq S_2 \times 2^{|S|}$, где $2^{|S|}$ означает мощность множества S.

Пусть при этом Σ_p и Σ_n являются подмножествами $S_2 \times 2^{|S|}$, а $u \subseteq B^{|U|}$ и $v \subseteq B^{|V|}$. Функция перехода существует, т.е $T^{(t+1)}(\Sigma_p, u, v, \Sigma_n) = 1$, если выполняются три следующих условия:

1) $\Sigma_p \in S^{(t)}$;

2) $\forall x \in B^{|X|}, \forall (\tilde{s}_2, \tilde{s}^*) \in \Sigma_p : u = \lambda_2^{(u)}(\tilde{s}_2, xv) \Rightarrow \begin{cases} \lambda_2^z(\tilde{s}_2, xv) \in \Lambda(\tilde{s}^*, x) \\ (\delta_2(\tilde{s}_2, xv), \Delta(\tilde{s}^*, x)) \in \Sigma_n \end{cases}$;

3) $\forall (s_2, s^*) \in \Sigma_n : \exists (x, \tilde{s}_2, \tilde{s}^*) \in B^{|X|} \times S_2 \times 2^{|S|} \Rightarrow \begin{cases} (\tilde{s}_2, \tilde{s}^*) \in \Sigma_p \\ u = \lambda_2^{(u)}(\tilde{s}_2, xv) \\ s_2 = \delta_2(\tilde{s}_2, xv) \\ s^* = \Delta(\tilde{s}^*, x) \end{cases}$.

В каждом из вычислений $S^{(t)}$ есть множеством подмножества $S_2 \times 2^{|S|}$. Причем пустое подмножество $\{0\}$ тоже может входить во множество $S^{(t)}$. При заданном $T^{(t+1)}$ множество $S^{(t+1)}$ вычисляется следующим образом: $S^{(t+1)}(\Sigma_p) = 1$ в том и только том случае, если также $S^{(t)}(\Sigma_p) = 1$ или $\exists \tilde{\Sigma}_p \in S^{(t)}, u \in B^{|U|}, v \in B^{|V|} : T^{(t+1)}\left(\tilde{\Sigma}_p, u, v, \Sigma_p\right) = 1$.

Пусть K будет позитивным целым, таким, что $S^{(K)}\left(\Sigma_p\right) = S^{(K-1)}\left(\Sigma_p\right)$. Такое K всегда существует, если количество элементов множества $S^{(t)}$ не возрастает во время вычислений и количество подмножеств $S_2 \times 2^{|S|}$ конечно.

Пусть $T : S \times B^{|U|} \times B^{|V|} \times S \to B$ будет такой связью, что $T\left(\Sigma_p, u, v, \Sigma_n\right) = 1$ в том и только в том случае, если $\Sigma_p = \Sigma_n = \{0\}$ или $T^{(K)}\left(\Sigma_p, u, v, \Sigma_n\right) = 1$.

Значит, мы можем определить ИКА документооборота пятеркой $T = \left(U, V, S, T, \Sigma_r\right)$, где $\Sigma_r = \{(r_2, \{r\})\}$. При этом каждое состояние ИКА представляет подмножество $S_2 \times S$.

3.4.2. Архитектура ИКА

В работе [10] показано, что рекурсивные алгоритмы могут быть построены из иерархических модулей, которые вызывают сами себя и рекурсивно передают входные и выходные данные. На рисунке 2 показана рекурсивная функция gcd. Локальные состояния $x1$ и $x2$ определяют соответствующую ветвь алгоритма. Микрооперации $y1$ и $y2$ осуществляют обработку данных и рекурсивную передачу данных между модулями алгоритма. Рекурсия организуется путем многократного вызова одного модуля, в нашем случае модуля Z.

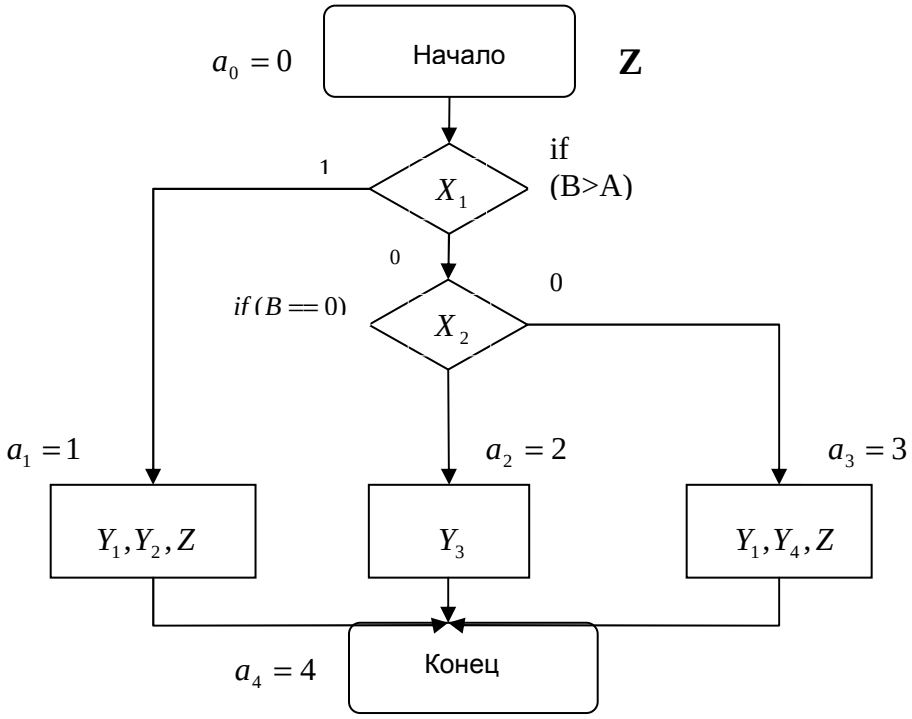

Рис. 2. Рекурсивная функция $\gcd$

Архитектура иерархического конечного автомата, который может рекурсивно обрабатывать конечные автоматы, приведена на рис. 3.

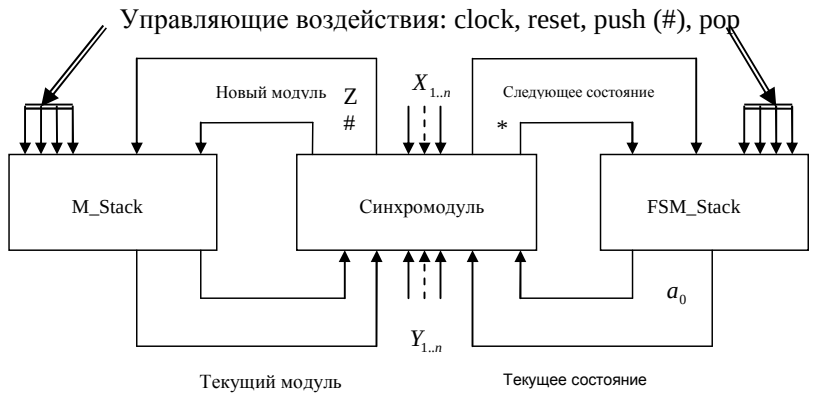

Рис. 3. Архитектура ИКА

Иерархический КА, который приведен на рис. 3, имеет два технологических стека. Один стек автомата, который обозначен FSM_stack, предназначен для обработки состояний. Второй стек, обозначенный M_stack, предназначен для обработки модулей, представляющих собой автоматы моделирования поведенческих единиц.

Взаимодействие стеков обеспечивается синхронизационным модулем. Синхронизационный модуль отвечает за вызов новых модулей, передачу входных состояний активизированному модулю и получение выходных состояний из модуля, который заканчивает работу.

Состояние внутри ИКА соответствует состояниям $a0...a4$ на рис. 1. Поскольку каждый конкретный модуль имеет уникальный идентификатор, то обозначения одних и тех же состояний

могут использоваться в различных модулях. На рис. 2 показано, как ИКА исполняет алгоритм, приведенный на рис. 1.

Все неиерархические вызовы производятся путем смены кода модуля в верхнем состоянии регистра FSM_Stack. На рис. 2 такой пример обозначен знаком «*». Все иерархические вызовы изменяют состояния обоих стеков. При этом M_Stack сохраняет код нового модуля, а FSM_Stack устанавливается в начальное состояние инициализируемого модуля. Такой пример на рис. 2 обозначен символом «#».

Иерархический вызов активирует операцию «pop» без изменения стеков, на рисунке это обозначено «&». В результате завершения работы модуля ИКА перейдет в состояние, последующее за состоянием, на котором был произведен вызов модуля. Указатель стеков stack_ptr является общим для обоих стеков. Работа ИКА прекращается при достижении позиции «End» и состоянии указателя stack_ptr =0.

Таким образом, работают описанные выше автоматы, составляющие автоматную часть формальной модели композитного документооборота.

5. Выводы

Представление модели документооборота в виде ИКА позволяет применить к задачам документооборота апробированный аппарат. В результате сложные процессы документооборота могут быть формально представлены в виде единого автомата, которые, в свою очередь, оперируют автоматами, а каждый из них моделирует единицу поведения системы. Каждый из автоматов моделирует поведение участника, обрабатывающего изменения состояний документов.

Последовательность обработки автоматов центральным автоматом определяется связями, описанными графом, который устанавливает связность автоматов между собой. Применение графов позволяет использовать апробированный и развитый аппарат теории графов при описании связности обрабатываемый автоматов.

Такое представление дало возможность реализовать двигатель управленческих процессов, основанный на формальных моделях. Этот двигатель является центральной частью программного реализованного комплекса композитного документооборота.

СПИСОК ЛИТЕРАТУРЫ

1. Теслер Г.С. Новая кибернетика. – Киев: Логос, 2004. – 401с.
2. Алферова З.В. Математическое обеспечение экономических расчетов с использованием теории графов. – М.: Статистика, 1974. – 208 с.
3. Clarence Ellis Team Automata for Groupware Systems. – Arizona: ACM SIGGROUP, 2001. – P. 415–424.
4. Marc Hoffman, David Shute, Mike Ebbers Advanced Workflow Solutions. –New York: Redbooks IBM, 1999. –141 p.
5. Круковский М.Ю. Графовая модель композитного документооборота // Математичні машини і системи. – 2005. – № 1. – С. 120–136.
6. Круковский М.Ю. Автоматная модель композитного документооборота // Математичні машини і системи. – 2004. – № 4. – С. 37–50.
7. Inout Cardei, Rakesh Jha, Mihaela Cardei Hierarchical architecture for real-time adaptive resource management. – Secaucus, NJ, USA: Springer-Verlag, 2000. – 434 p.
8. Круковский М.Ю. Методология построения композитных систем документооборота // Математичні машини і системи. – 2004. – № 1. – С. 101–114.
9. Yosinori Watanabe, Robert Brayton The maximum set of permissible behaviors of FSM networks. – Los Alamitos CA USA // IEEE Computer society press. – 1993. – P. 316–420.
10. Sklyarov V.Hardware implementation of hierarchical FSMs. – Cape Town, South Africa: ACM, 2005. – P.148–153.