ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ

УДК 681.513
DOI 10.21685/2072-3059-2017-2-1

В. Н. Дубинин, И. В. Сенокосов, В. В. Вяткин, Л. П. Климкина

ПРЕОБРАЗОВАНИЕ АВТОМАТНЫХ СПЕЦИФИКАЦИЙ В ФУНКЦИОНАЛЬНО-БЛОЧНУЮ РЕАЛИЗАЦИЮ СИСТЕМЫ УПРАВЛЕНИЯ СОРТИРОВКОЙ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ДЕТАЛЕЙ

Аннотация.

Актуальность и цели. Усложнение выпускаемых изделий приводит к усложнению технологического процесса их изготовления. Важной составной частью производственных систем наряду с центрами обработки деталей становятся также центры сортировки и хранения деталей. Целью работы является разработка методов и средств описания и синтеза компонентно-базированных систем управления сортировкой деталей на основе выборки специфицированных последовательностей. Объект исследования – инфомехатронные производственные системы для сортировки деталей. Предмет исследования – методы и средства проектирования и реализации систем управления сортировкой последовательностей деталей на основе международного стандарта IEC 61499.

Материалы и методы. Исследования выполнены с использованием теории конечных автоматов и формальных языков, графовых трансформаций, логического программирования, а также методов разработки программного обеспечения управляющих систем на основе функциональных блоков (ФБ) стандарта IEC 61499.

*Результаты***.** Разработаны: 1) формальная модель селектирующего автомата для спецификации и выборки последовательностей деталей в промышленных системах сортировки; 2) подход к реализации селектирующих автоматов на основе ФБ с использованием механизма передачи маркеров; 3) система вывода графов, определяющая процесс порождения структур систем ФБ IEC 61499 из конечноавтоматной спецификации селектируемых последовательностей деталей; 4) методика трансляции автоматных спецификаций селектируемых последовательностей деталей в систему управления сортировкой на основе ФБ IEC 61499.

Выводы. Предложенный подход к построению систем сортировки последовательностей деталей является новым и еще не применялся в промышленной практике. Разработанные методы и средства описания и синтеза функционально-блочной системы управления сортировкой последовательностей деталей являются удобными и эффективными, что подтверждается опытом создания и эксплуатации соответствующих инструментальных программных средств.

Ключевые слова: сортировка, выборка, последовательность деталей, спецификация, конечный автомат, недетерминированный автомат, селектирующий автомат, передача маркеров, вывод графов, язык Prolog, функциональный блок, система управления, стандарт IEC 61499, система сортировки шаров, nxtStudio.

V. N. Dubinin, I. V. Senokosov, V. V. Vyatkin, L. P. Klimkina

TRANSFORMATION OF FINITE STATE MACHINE-BASED SPECIFICATIONS INTO FUNCTION BLOCK-BASED IMPLEMENTATION OF A CONTROL SYSTEM FOR WORKPIECE SEQUENCES SORTING

Abstract.

Background. Increasing product complexity complicates technological processes of manufacturing. Centers of workpiece sorting and storage are becoming an important part of production systems along with centers of workpieces processing. The goal of this paper is to develop methods and tools describe and synthesize component-based control systems for workpieces sorting on the basis of specified sequences selection. The research object is information mechatronic production systems for workpieces sorting. The research subject is methods and tools of design and implementation of control systems for workpieces sequences sorting on the basis of the standard IEC 61499.

Materials and methods. This research was carried out using the theory of finite state machines and formal languages, graph transformations, logic programming, as well as software engineering methods for control systems based on IEC 61499 function blocks (FB).

Results. The authors have developed as follows: 1) a formal model of selecting finite automata for specification and selection of workpiece sequences in industrial sorting systems; 2) an approach to implementation of FB-based selecting automata using a token transferring mechanism; 3) a graphs inference system defining the process of IEC 61499 FB system structure generation from finite state machine–based specifications of selectable workpiece sequences; 4) a method of automatic translation of specifications of selectable workpiece sequences into the IEC 61499 FB-based sorting control system.

Conclusions. The proposed approach to construction of workpiece sequences sorting systems is new and has not yet been used in industrial practice. The developed methods and tools for describing and synthesizing FB-based workpiece sequences sorting control systems are convenient and effective as evidenced by the experience of development and exploitation of the relevant software tools.

Key words: sorting, selection, workpiece sequence, specification, finite state machine, nondeterministic automaton, selecting automaton, token transfer, graphs inference, Prolog language, function blocks, control system, standard IEC 61499, ball sorting system, nxtStudio.

Введение

Важной составной частью производственных систем наряду с центрами обработки деталей становятся также центры сортировки, выборки и хранения деталей. Пример производственной системы для сортировки деталей (шаров), разработанный на симуляторе EasyVEEP, можно найти в работе [1]. В сборочном производстве или в производственной логистической системе в ряде случаев требуется выделение и выборка из общего потока разнообразных деталей, идущих по конвейеру, определенных последовательностей деталей. В интернете вещей [2] каждый материальный объект, включая изделия, имеет свой уникальный идентификатор, который может обрабатываться информационными процессами, в том числе процессами выборки и сортировки. Существующая тенденция слияния процессов обработки, передачи и хране-

ния информации с процессами обработки, транспортировки и складирования материалов и материальных объектов приводит к созданию нового класса систем – киберфизических систем промышленной автоматики.

В данной работе рассматриваются системы сортировки деталей на основе выборки определенных последовательностей деталей, специфицированных с помощью формальной модели (иными словами, системы сортировки последовательностей деталей), а не системы сортировки отдельных деталей в классическом понимании. Для спецификации и выборки последовательностей деталей предлагается использовать конечноавтоматную модель [3]. В этом случае система сортировки будет представлять собой совокупность взаимодействующих автоматов, каждый из которых ориентирован на выбор «своих» специфицированных последовательностей деталей. Следует отметить, что не удалось обнаружить исследования, напрямую связанные с данной проблематикой. Наиболее близкой является работа [4], в которой предлагается алгоритм для распознавания объектов на основе данных, поступающих из множества датчиков. При этом общая стратегия для извлечения свойств из различных источников реализуется как конечный автомат. Для распознавания используется дерево решений.

Отдельной проблемой является преобразование описаний исходных спецификаций в модель реализации. В данном случае может использоваться современный подход к проектированию программного обеспечения на основе управления моделями (*Model Driven Engineering – MDE*) и его расширение для доменно-специфических языков, получившее название интегрированных модельных вычислений (*Model-Integrated Computing – MIC*) [5]. Отправным пунктом проектирования является начальная модель системы, а конечным результатом – целевая модель системы. Основой данных технологий является трансформация моделей.

Для проектирования и реализации систем управления промышленной автоматики все чаще используется международный стандарт IEC 61499 [6]. Данный стандарт имеет большое значение в построении информационно-управляющих систем нижнего уровня нового поколения. Основными артефактами проектирования при этом являются функциональные блоки (ФБ).

Ниже рассматриваются вопросы проектирования и реализации систем управления выборкой последовательностей деталей на основе стандарта IEC 61499 с использованием конечноавтоматного подхода.

1. Использование конечных автоматов для спецификации и выборки последовательностей деталей

Конечные автоматы могут использоваться как для задания автоматных языков, так и для распознавания входных последовательностей данных языков. Традиционная область применения подобных автоматов-распознавателей – системы лексического анализа при построении трансляторов и компиляторов [3].

Предлагается использовать конечный автомат для спецификации и выборки последовательностей деталей в соответствии с их типом (цветом). В дальнейшем будем называть такой конечный автомат селектирующим конечным автоматом. В отличие от автомата-распознавателя, селектирующий автомат может игнорировать поступающие на его вход входные символы, если они не вызывают срабатывание ни одного из его переходов.

Определим селектирующий конечный автомат A (далее – просто автомат) как кортеж следующего вида:

$$A = (Q, T, C, \delta, \varphi, Q_0, F),$$

где Q – конечное множество состояний автомата; $T \subseteq Q \times Q$ – конечное множество переходов автомата (множество дуг); $C = \{c_1, c_2, ..., c_m\}$ – конечное множество типов (цветов) деталей; $\delta : T \to C$ – функция, назначающая переходам цвета; $\varphi : T' \to N^+$ – функция, назначающая петлям из $T' \subseteq T$ максимальное число допустимых деталей. По петле $t \in T$ нельзя принять больше $\varphi(t)$ деталей цвета $\delta(t)$; Q_0 – множество начальных состояний автомата; $F \subseteq T$ – множество конечных переходов автомата.

При конечном переходе автомата считается, что предшествующая последовательность деталей принята и начинается сборка новой последовательности. Как правило, конечные переходы автомата оканчиваются в начальном состоянии. Это имитирует сброс автомата в начальное состояние. Вместе с тем представляет интерес и альтернативный вариант. В этом случае можно считать, что происходит динамическое изменение начального состояния автомата при сборке новой последовательности деталей. Данное свойство можно отнести к классу самомодификации формальной модели. Следует также отметить, что приведенное формальное определение селектирующего автомата может быть расширено под определенный класс решаемых задач сортировки.

Селектирующие конечные автоматы делятся на детерминированные и недетерминированные. Если выполняется условие

$$\forall (q_a, q_b), (q_a, q_c) \in T \, [(q_b \neq q_c) \to \delta(q_a, q_b) \neq \delta(q_a, q_c)],$$

то автомат детерминированный, иначе – недетерминированный. В детерминированном автомате существует только одно начальное состояние, $|Q_0|=1$, в то время как в недетерминированном автомате их может быть несколько.

На рис. 1 приведен пример селектирующего детерминированного конечного автомата (СДКА), где O (orange), G (grey) и B (blue) – это цвета деталей. Конечный переход обозначен пунктирной линией.

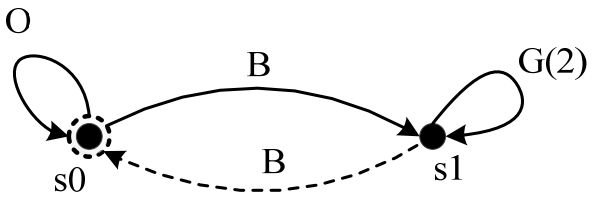
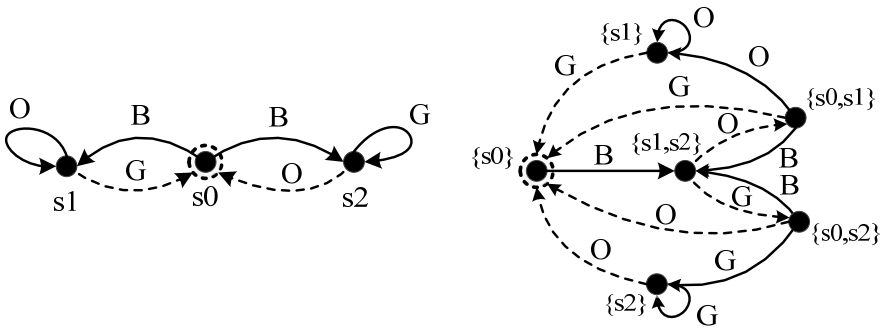


Рис. 1. Пример селектирующего детерминированного конечного автомата

Приведенный на рис. 1 автомат выбирает, например, следующие последовательности деталей: BB, BGB, OOBGB, OOOBBBBGGB. Если на вход автомата будут поданы цепочки BG, OOGGGG, OOOBGG, то он не сможет выбрать из них ни одну из специфицированных последовательностей.

Пример селектирующего недетерминированного конечного автомата (СНДКА) представлен на рис. 2,*а*. Эквивалентный ему СДКА, полученный в результате процедуры детерминизации, приведен на рис. 2,*б*. Преимуществом СНДКА перед СДКА является компактность описания, недостатком – более сложная реализация.



а) *б*)

Рис. 2. Пример селектирующего недетерминированного конечного автомата: ***а*** – СНДКА; ***б*** – результат детерминизации СНДКА

Как можно заметить, автомат на рис. 2 может изменить свое начальное состояние {s0} в ходе своей работы на состояния {s0,s1}, {s0,s2}, (s1}, {s2}. В результате этого он может принимать цепочки, начинающиеся с O или G, что было невозможно в случае классического СНДКА.

Для описания и обработки селектирующих автоматов был выбран язык *Prolog* [7]. Описание автомата, представленного на рис. 1, в виде базы фактов на языке *Prolog*:

```
state(s0).
state(s1).
transition(t1,s0,s1,black,_,ordinary).
transition(t2,s0,s0,orange,999,ordinary).
transition(t3,s1,s1,gray,2,ordinary).
transition(t4,s1,s0,black,_,final).
initial_state(s0).
```

В приведенном описании используются следующие предикаты:

`state` – унарный предикат для представления состояний автомата;

`transition` – 6-арный предикат для представления переходов автомата, имеющий следующие аргументы: первый аргумент – идентификатор перехода; второй аргумент – идентификатор состояния-источника; третий аргумент – идентификатор состояния-приемника; четвертый аргумент – цвет детали; пятый аргумент – максимальное число допустимых деталей (для петли); шестой аргумент – тип перехода (обычный или конечный);

`initial_state` – унарный предикат, обозначающий начальное состояние автомата.

2. Подходы к реализации селектирующих автоматов на основе функциональных блоков

Ниже рассматриваются подходы к реализации селектирующих автоматов на основе функциональных блоков международного стандарта IEC 61499.

В свою очередь функционально-блочная реализация автомата может быть легко интегрирована в общую систему управления сортировкой на основе того же стандарта.

СДКА может быть напрямую реализован с использованием диаграммы управления выполнением (диаграммы ECC) базисного ФБ. Однако реализация СНДКА требует более сложных подходов. В работе [8] недетерминированный автомат (НДА) трактуется как параллельная модель с собственными правилами функционирования. С другой стороны, функционирование НДА из данной работы можно представить как неявную детерминизацию НДА в режиме реального времени. В этом случае состояния детерминированного конечного автомата (ДКА) получаются в процессе (параллельного) функционирования НДА как комбинации его активных состояний. Существенным моментом при этом является синхронный характер функционирования НДА. В дальнейшем примем такой подход за основу. Пример двухфазной реализации НДА на языках программируемых логических контроллеров стандарта IEC 61131-3 можно найти в работе [9].

Можно выделить следующие подходы к реализации СНДКА на ФБ: когда переходы (а), или состояния (б) СНДКА представляются в виде ФБ. Данные подходы по своей сути эквиваленты, но ориентация на переходы удобнее для представления петель, и, кроме того, она ограничивает число основных типов ФБ до числа используемых типов деталей, поэтому в дальнейшем используем данный подход. Ввиду синхронного функционирования СНДКА общей проблемой этих двух подходов является проблема конфликтов при установке/сбросе активных состояний [9]. Эта ситуация возникает, когда в одно и то же состояние входят и выходят дуги, помеченные одним и тем же входным сигналом, например: $S_1 \xrightarrow[t1]{x} S_2 \xrightarrow[t2]{x} S_3$. При активном состоянии $\{S_1, S_2\}$ правильной последовательностью срабатывания переходов является ($t2$, $t1$), приводящая к смене состояния $\{S_1, S_2\} \xrightarrow[t2,t1]{x} \{S_2, S_3\}$. В противном случае будет получено неверное результирующее состояние: $\{S_1, S_2\} \xrightarrow[t1,t2]{x} \{S_3\}$. Ввиду ограниченности объема статьи конфликтные ситуации в дальнейшем не рассматриваются. Можно только отметить, что в реализациях на ФБ для разрешения конфликтов на основе приоритетов для задания явного порядка срабатывания переходов можно использовать дейзи-цепочки из ФБ. Далее для простоты изложения будем рассматривать реализацию преимущественно СДКА. Нюансы реализации СНДКА будут отмечены отдельно.

Для моделирования функционирования селектирующего конечного автомата предлагается использовать механизм передачи маркеров. Это универсальный подход по отношению к СДКА и СНДКА. Маркер является динамическим объектом, который может быть передан из одного ФБ в другой ФБ. Если ФБ имеет маркер, то он может активно контролировать входящие детали, и в случае, когда деталь и собственно ФБ имеют одинаковые цвета, ФБ инициирует действие для захвата этой детали. Возможно наличие маркеров в один и тот же момент времени в разных ФБ. Из совокупности всех ФБ с маркером принять текущую деталь могут только те ФБ, которые настроены на прием деталей данного типа.

На рис. 3 представлена структура функционально-блочной реализации автомата, приведенного на рис. 1. Предполагается, что информация в данную систему ФБ поступает через сервисный интерфейсный ФБ (СИФБ) с датчиков цвета детали. Выходным сигналом системы является сигнал *EOS* «Конец выбранной последовательности».

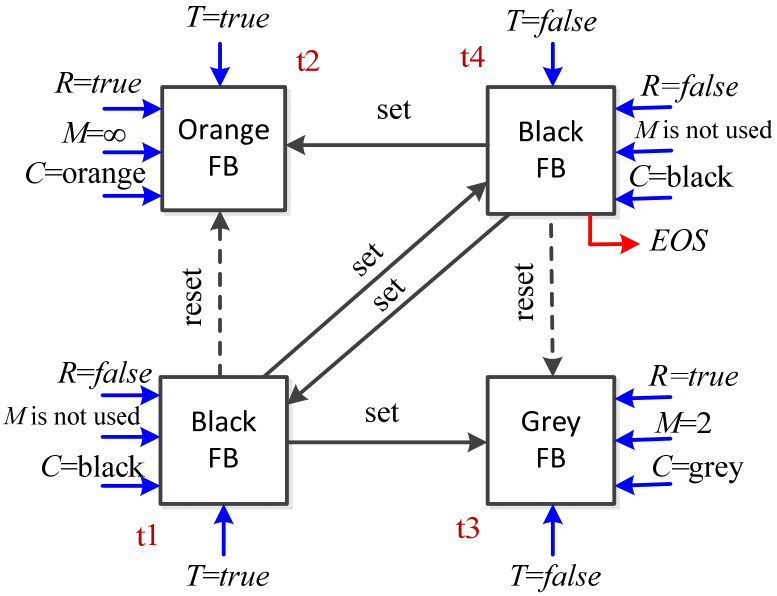


Рис. 3. Структура функционально-блочной реализации автомата из рис. 1

Для представления структуры функционально-блочной реализации автомата используются следующие элементы:

• дуга *set* для передачи маркера из исходного ФБ в целевой ФБ;

• дуга *reset* для удаления маркера из целевого ФБ;

• входные висячие дуги для установки начальных значений следующих параметров:

C – цвет детали, которую может принять ФБ. Условно это можно назвать «цветом ФБ». Данный параметр не может быть изменен во время функционирования системы и устанавливается изначально;

T – флаг наличия маркера в ФБ;

R – флаг петли (итерации). Если *R* = *true*, то ФБ осуществляет сбор последовательности деталей заданного цвета. Если *R* = *false*, то ФБ является «транзитным». После обработки одной детали он передает маркер другим ФБ;

M – максимальное количество деталей заданного цвета, которые могут быть обработаны данным ФБ перед передачей маркера на другой (другие) ФБ. Данный параметр действителен только для ФБ с *R* = *true*.

3. Автоматическая трансформация автоматных спецификаций в структуры функционально-блочной реализации

Для автоматического преобразования селектирующего автомата в структуры функционально-блочной реализации предлагается использовать

механизм графовых трансформаций [10]. Преимуществом такого подхода является наглядность, формальность и наличие инструментальных средств для проведения трансформации. Кроме того, в ряде случаев напрямую возможен логический вывод элементов целевой системы из элементов исходной системы. Набор правил для вывода элементов целевой системы ФБ из элементов исходного селектирующего автомата приведен ниже. Следует заметить, что ограничением в данном случае является класс используемой автоматной модели – это СДКА и СНДКА без конфликтов. Для последнего класса правило на рис. 4 должно быть немного модифицировано: не должны сбрасываться ФБ-переходы, имеющие пометку, равную текущему цвету детали.

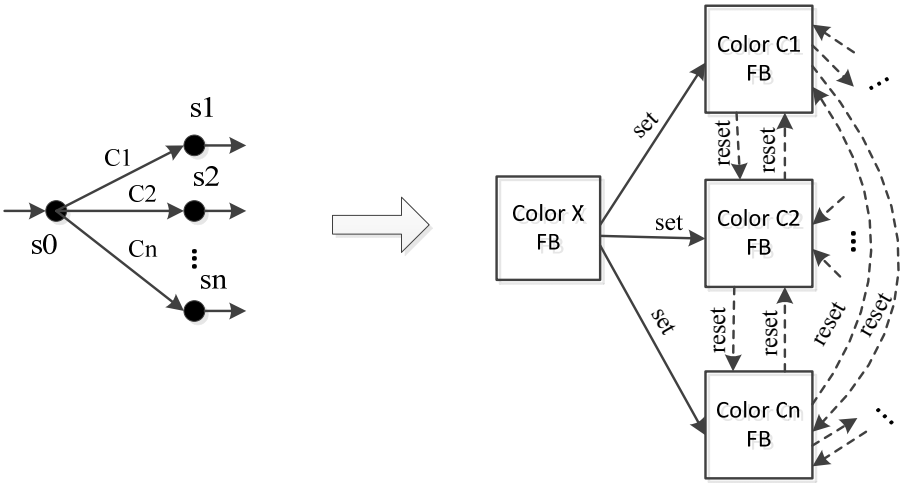


Рис. 4. Обобщенное правило преобразования автоматной конструкции «Выбор»

Как видно из рис. 5–8, параметры *C* и *M* для генерируемых ФБ выводятся из разметки автоматного перехода. Приведенные правила даны для случая обычных переходов и должны быть слегка модифицированы в случае использования конечных (*final*) переходов автомата. Модификации будут в основном касаться добавления в генерируемый ФБ выходного сигнала *EOS*. Для выполнения вывода структуры реализующей системы ФБ была разработана программа на языке логического программирования *Prolog* [7]. Язык *Prolog* имеет мощный встроенный механизм логического вывода. Фактически каждое правило из приведенных выше рисунков представляется правилом на языке *Prolog*.

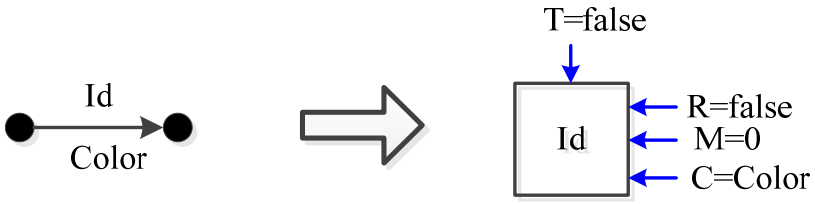


Рис. 5. Правило вывода ФБ, не имеющего начального маркера и итерации, использующее переход автомата, начинающийся не в начальном состоянии

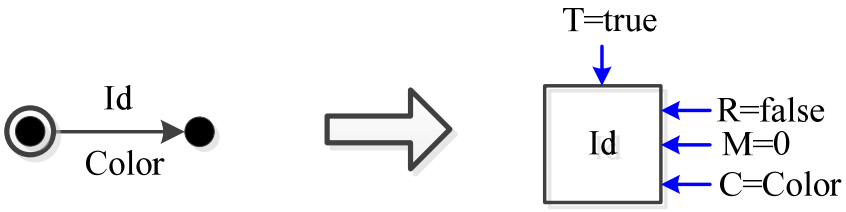


Рис. 6. Правило вывода ФБ, имеющего начальный маркер,
но не имеющего итерации, использующее переход автомата,
начинающийся в начальном состоянии

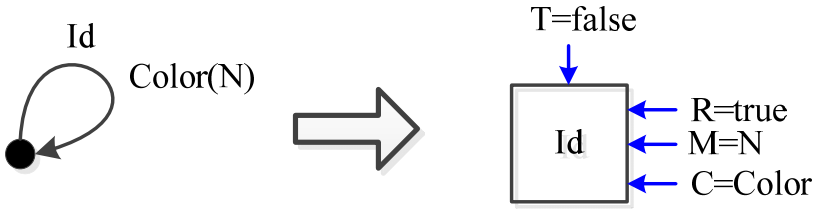


Рис. 7. Правило вывода ФБ, не имеющего начального маркера,
но имеющего итерацию, использующее переход автомата в виде петли,
начинающейся не в начальном состоянии

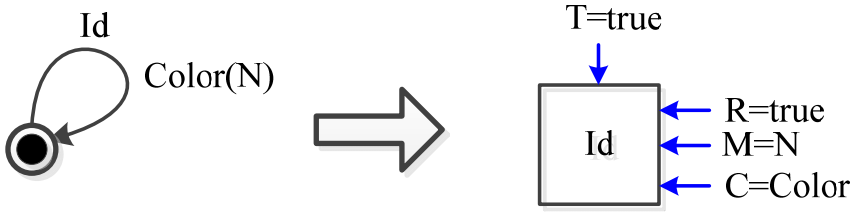


Рис. 8. Правило вывода ФБ, имеющего начальный маркер и итерацию, использующее
переход автомата в виде петли, начинающейся в начальном состоянии

В программе на языке *Prolog* ФБ представляется 6-местным предикатом `function_block`, имеющим следующие аргументы (в порядке следования): идентификатор ФБ, цвет, наличие итерации, наличие маркера, максимальное число допустимых деталей, наличие выхода *EOS*. Для представления связей между ФБ используется 3-местный предикат `connection` со следующими аргументами (в порядке следования): идентификатор ФБ-источника, идентификатор ФБ-приемника, тип дуги-соединения. На рис. 9, 10 для примера приведены два правила на языке *Prolog*.

Правило, кодирующее графическое правило из рис. 7:

```
function_block(Id,Color,true,true,NRep,false):-
transition(Id,State,State,Color,NRep,ordinary),
initial_state(State).
```

Правило, кодирующее верхнее графическое правило из рис. 9:

```
connection(T1,T2,set):-
transition(T1,S1,S2,_,_,_),
S1\==S2,
transition(T2,S2,S3,_,_,_),
```

```
function_block(T1,_,_,_,_,_),
function_block(T2,_,_,_,_,_).
```

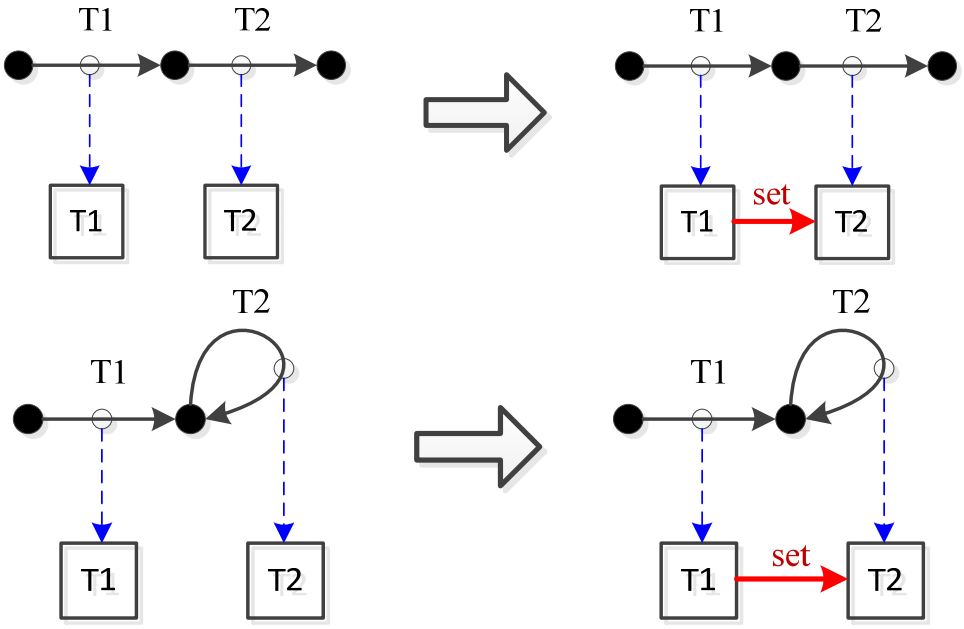


Рис. 9. Правила вывода дуги *set*, использующие смежные переходы автомата

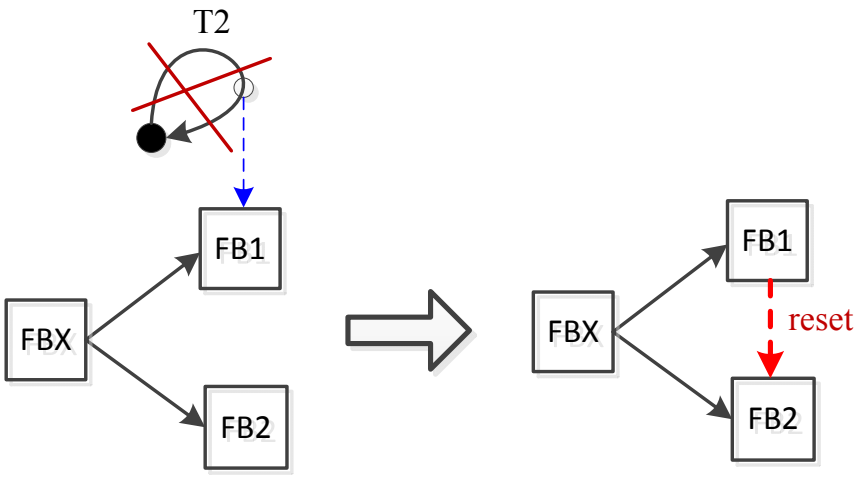


Рис. 10. Правила вывода дуги *reset*

4. Пример. Система сортировки последовательностей шаров

В качестве демонстрационного примера предлагается система сортировки шаров на основе выборки специфицированных последовательностей деталей. Система состоит из выталкивателя; трех датчиков для распознавания типа шаров (емкостной, индуктивный и оптический датчики); двух датчиков определения положения выталкивателя; емкостного датчика, определяющего, был ли вытолкнут шар в шахту; двух заслонок для забора шаров из шахты

в определенный накопитель. Человеко-машинный интерфейс (*HMI*) включает основные статические элементы системы сортировки, динамические объекты (шары), а также световые индикаторы для значений датчиков и сигнала *EOS* (рис. 11). В качества прототипа *HMI* использовался *HMI* симулятора *EasyV-EEP* для случая системы сортировки шаров [1].

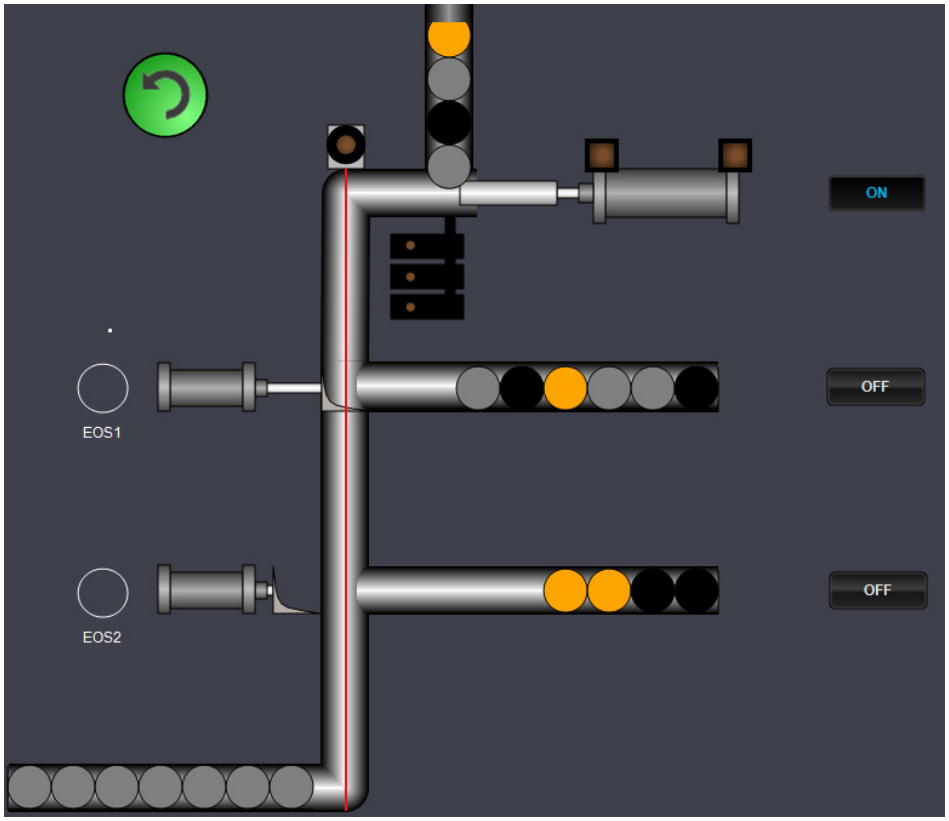


Рис.11. Система сортировки последовательностей шаров
(интерфейс пользователя в *nxtStudio*)

Система сортировки работает следующим образом: в первый накопитель попадают шары, выделенные СНДКА (рис. 2,*а*), а во второй накопитель – шары, отобранные с помощью СДКА (рис. 1). Шары, не попавшие в первые два накопителя, попадают в третий накопитель (для отбракованных деталей). Следует заметить, что в *HMI* на рис. 11 принятые последовательности деталей не удаляются.

5. Методика трансляции автоматных спецификаций в систему управления сортировкой последовательностей шаров на основе функциональных блоков стандарта IEC 61499

Структуры функционально-блочной реализации, используемые в разделе 3, не включают все детали реализации, поэтому возникает необходимость их дальнейшего уточнения в терминах стандарта IEC 61499. Функциональные блоки, используемые в данных структурах, имеют одинаковый настраиваемый функционал, но различаются только значениями входных па-

раметров. С их помощью происходит настройка ФБ под конкретные контекстные условия. Таким образом, для представления этих ФБ может использоваться один и тот же тип (базисного) ФБ (рис. 12).

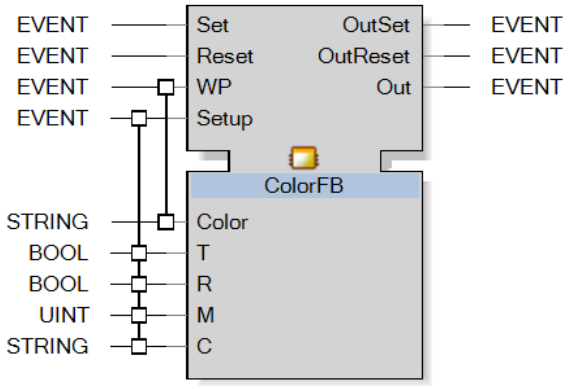


Рис. 12. Интерфейс базисного ФБ для моделирования переходов автомата

Интерфейс данного базисного ФБ состоит из следующих групп параметров:

Входные сигналы:

Set – сигнал установки маркера в данном ФБ;

Reset – сигнал удаления маркера из данного ФБ;

WP – сигнал поступления детали в систему сортировки;

Setup – сигнал, установки (сброса) начальных параметров ФБ.

Выходные сигналы:

OutSet – сигнал, устанавливающий маркеры в других ФБ;

OutReset – сигнал, сбрасывающий маркеры в других ФБ;

Out – сигнал, означающий, что ФБ обработал поступившую деталь.

Входные переменные соответствуют входным параметрам ФБ, описанным в разд. 3.

Диаграмма управления выполнением (иначе, диаграмма ECC) (рис. 13) данного базисного ФБ состоит из шести состояний:

1) состояние *Reset*, в котором выполняется сброс маркера ФБ;

2) состояние *Setup*, в котором устанавливаются начальные значения внутренних переменных;

3) состояние *Set*, в котором происходит установка маркера ФБ;

4) состояние *WP*, в которое автомат переходит при выполнении следующего условия: *WP & (T1 = TRUE) & (Color = C1)*. Переход в данное состояние произойдет, если в систему сортировки поступит деталь цвета, воспринимаемого этим ФБ, и данный ФБ имеет маркер;

5) состояние *WP1*, в котором происходит увеличение счетчика количества обработанных деталей. В данное состояние может попасть ФБ, имеющий *R = true*. Кроме того, в данном состоянии выдается выходные сигналы *Out* и *OutReset*;

6) состояние *WP2*, в котором происходит удаление маркера и выдача всех требуемых выходных сигналов. В данное состояние попадает ФБ, имеющий *R = false*, или ФБ, уже принявший *M* деталей заданного цвета.

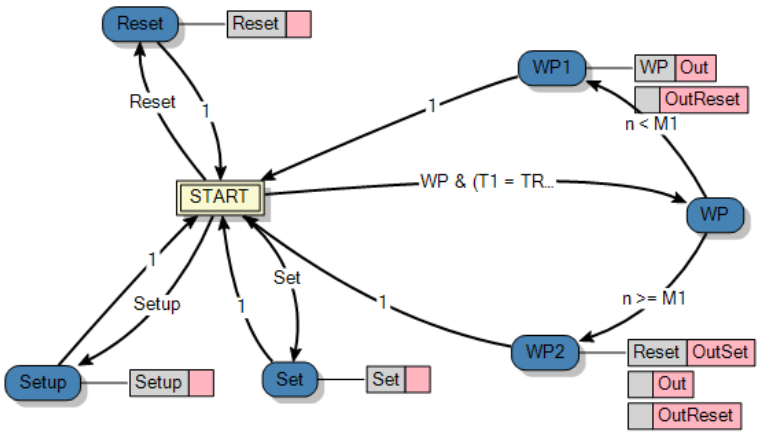


Рис. 13. Диаграмма ECC базисного ФБ для моделирования переходов автомата

Из базисных ФБ происходит построение составного ФБ, который представляет автомат, выбирающий последовательности деталей. Пример составного ФБ, соответствующий структуре, представленной на рис. 3, приведен на рис. 14.

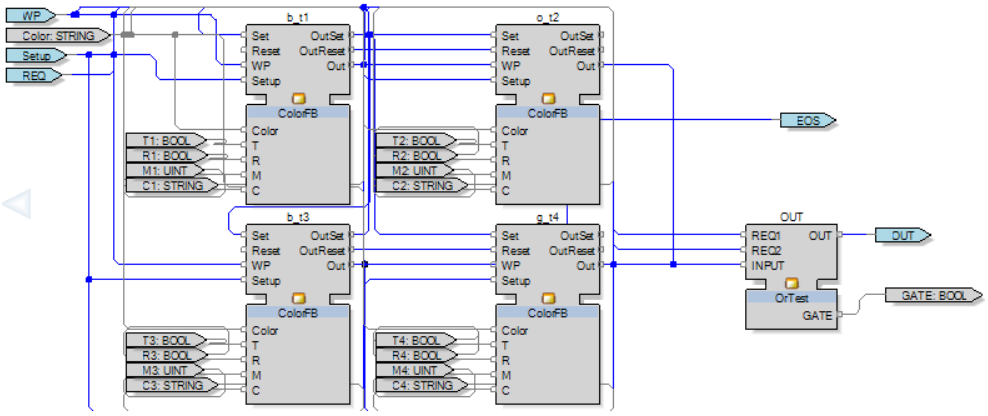


Рис. 14. Сеть ФБ стандарта IEC 61499, реализующая автомат из рис. 1

Реализация полной имитационной модели системы сортировки последовательностей шаров (рис. 15) состоит из четырех блоков: *BallSorting* – ФБ, представляющий визуальную (анимационную) модель, включающую генератор шаров; *FB3* – контроллер выталкивателя (управляет его возвратно-поступательным движением); *FB2* – первый селектирующий автомат; *FB4* – второй селектирующий автомат. Следует отметить, что ФБ *FB2*, *FB3* и *FB4* относятся к системе управления. Визуальная имитационная модель оборудования и *HMI* на основе ФБ разработана в университете *Aalto* (Финляндия).

Имитационная модель системы сортировки, включающая систему управления на основе ФБ и визуальную модель оборудования, работает следующим образом: сначала генерируется шар определенного цвета, который распознается с помощью датчиков и попадает на анализ первому селектирующему автомату. Если шар подходит, то сигнал «выдвинуть первую заслон-

ку» передается в модель сортировки, и, таким образом, шар попадает в первый накопитель. В противном случае сигнал передается на второй селектирующий автомат и проверяется им. Таким образом, первый накопитель является более приоритетным, чем второй. Если второму автомату шар «подходит», то в систему сортировки подается сигнал «выдвинуть вторую заслонку». Если и второму накопителю шар не подходит, то он попадает в третий накопитель, в котором собираются неиспользованные шары. Когда одна из двух заслонок выдвинута, но шар не подходит, то она задвигается и пропускает шар, летящий вниз по шахте.

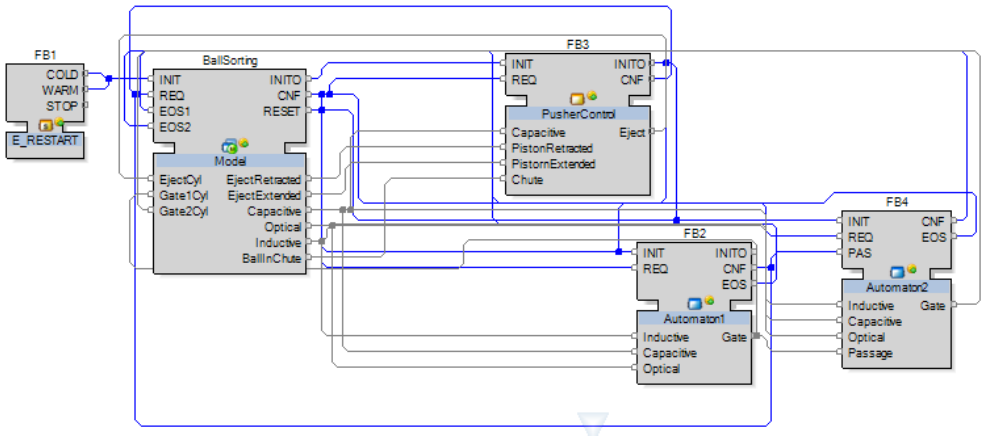


Рис.15. Реализация системы сортировки шаров

Программно реализован автоматический транслятор автоматных спецификаций на языке *Prolog* в *XML*-базированный проект системы *nxtStudio* [11], представляющий имитационную модель замкнутой системы сортировки последовательностей шаров.

Заключение

В представленной работе предложены принципы, модели, методы и средства для построения систем сортировки последовательности деталей. Выборка специфицированных последовательностей деталей задается с помощью формальной модели – конечного автомата. Направлением дальнейших исследований являются вопросы построения систем данного класса на основе селектирующих магазинных автоматов и селектирующих сетей Петри, а также решение задач интеграции и синтеза селектирующих автоматов на основе теории супервизорного управления.

Авторы статьи выражают благодарность разработчику Андрею Сандру (университет Аалто, Хельсинки) за предоставленную им визуальную имитационную модель аппаратной части системы сортировки шаров.

Библиографический список

1. Сайт EasyVeep. Ball sorting system. – URL: http://www.easyveep.com/modules.php?akt_modul=7&akt_Lang=2
2. **Rose, K.** The Internet of Things: An Overview / K. Rose, S. Eldridge, L. Chapin // Internet Society. – 2015. – October. – 51 p.

3. **Хопкрофт, Дж.** Введение в теорию автоматов, языков и вычислений / Дж. Хопкрофт, Р. Мотвани, Дж. Ульман. – М. : Вильямс, 2002. – 528 с.
4. **Hassibi, K. M.** A Multi-Sensor Robotics System For Object Recognition / K. M. Hassibi, K. A. Loparo, F. L. Merat // Proc. SPIE 1002, Intelligent Robots and Computer Vision VII, 1989.
5. Model-Driven Software Development / B. Sami, M. Book, V. Gruhn (eds.). – London : Springer, 2005. – 464 p.
6. **Vyatkin V.** IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design, second Edition / V. Vyatkin. – Instrumentation Society of America (ISA) and O^{3}neida, 2011. – 297 p.
7. **Клоксин, У.** Программирование на языке Пролог / У. Клоксин, К. Меллиш. – М. : Мир, 1987. – 336 с.
8. **Вашкевич, Н. П.** Недетерминированные автоматы и их использование для реализации систем параллельной обработки информации : моногр. / Н. П. Вашкевич, Р. А. Бикташев. – Пенза : Изд-во ПГУ, 2016. – 394 с.
9. **Дубинин, В. Н.** Проектирование и реализация систем управления дискретными событийными системами на основе иерархических модульных недетерминированных автоматов (Ч. 2. Методы и средства) / В. Н. Дубинин, Д. А. Будаговский, Д. Н. Дроздов, Д. В. Артамонов // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2016. – № 2 (38). – С. 18–32.
10. Graph Transformation for Practical Model Driven Software Engineering / L. Grunske, L. Geiger, A. Zuendorf, N. V. Eetvelde, P. V. Gorp, D. Varro // Model-driven Software Development. – Berlin ; Heidelberg : Springer Verlag, 2005. – P. 91–118.
11. Сайт nxtStudio (nxtControl). – URL: http://www.nxtcontrol.com/

References

1. *Web-site EasyVeep. Ball sorting system*. Available at: http://www.easyveep.com/modules.php?akt_modul=7&akt_Lang=2
2. Rose K., Eldridge S., Chapin L. *Internet Society*. 2015, October, 51 p.
3. Khopkroft Dzh., Motvani R., Ul'man Dzh. *Vvedenie v teoriyu avtomatov, yazykov i vychisleniy* [Introduction into the theory of automata, languages and computing]. Moscow: Vil'yams, 2002, 528 p.
4. Hassibi K. M., Loparo K. A., Merat F. L. *Proc. SPIE 1002, Intelligent Robots and Computer Vision VII*, 1989.
5. *Model-Driven Software Development*. B. Sami, M. Book, V. Gruhn (eds.). London: Springer, 2005, 464 p.
6. Vyatkin V. *IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design, second Edition*. Instrumentation Society of America (ISA) and O3neida, 2011, 297 p.
7. Kloksin U., Mellish K. *Programmirovanie na yazyke Prolog* [Prolog language programming]. Moscow: Mir, 1987, 336 p.
8. Vashkevich N. P., Biktashev R. A. *Nedeterminirovannye avtomaty i ikh ispol'zovanie dlya realizatsii sistem parallel'noy obrabotki informatsii: monogr.* [Nondeterministic automata and their application for implementation of parallel data processing systems: monograph]. Penza: Izd-vo PGU, 2016, 394 p.
9. Dubinin V. N., Budagovskiy D. A., Drozdov D. N., Artamonov D. V. *Izvestiya vysshikh uchebnykh zavedeniy. Povolzhskiy region. Tekhnicheskie nauki* [University proceedings. Volga region. Engineering sciences]. 2016, no. 2 (38), pp. 18–32.
10. Grunske L., Geiger L., Zuendorf A., Eetvelde N. V., Gorp P. V., Varro D. *Model-driven Software Development*. Berlin; Heidelberg: Springer Verlag, 2005, pp. 91–118.
11. *Web-site nxtStudio (nxtControl)*. Available at: http://www.nxtcontrol.com/

Дубинин Виктор Николаевич
доктор технических наук, профессор, кафедра вычислительной техники, Пензенский государственный университет (Россия, г. Пенза, ул. Красная, 40)

E-mail: dubinin.victor@gmail.com

Dubinin Viktor Nikolaevich
Doctor of engineering sciences, professor, sub-department of computer engineering, Penza State University (40 Krasnaya street, Penza, Russia)

Сенокосов Илья Владимирович
магистрант, кафедра вычислительной техники, Пензенский государственный университет (Россия, г. Пенза, ул. Красная, 40)

E-mail: senokosov.i@yandex.ru

Senokosov Il'ya Vladimirovich
Master's degree student, sub-department of computer engineering, Penza State University (40 Krasnaya street, Penza, Russia)

Вяткин Валерий Владимирович
доктор технических наук, профессор, кафедра «Ответственные коммуникации и вычисления», Технический университет Лулео (Швеция, г. Лулео, ул. Регнбогсаллен, корп. А)

E-mail: valeriy.vyatkin@ltu.se

Vyatkin Valeriy Vladimirovich
Doctor of engineering sciences, professor, sub-department of dependable communications and computations, Lulea University of Technology (building A, Regnbagallen street, Lulea, Sweden)

Климкина Людмила Петровна
старший преподаватель, кафедра организации и информатизации производства, Пензенский государственный аграрный университет (Россия, г. Пенза, ул. Ботаническая, 30)

E-mail: ludmila.klimkina@gmail.com

Klimkina Lyudmila Petrovna
Senior lecturer, sub-department of organization and informatization of manufacturing, Penza State Agricultural University (30 Botanicheskaya street, Penza, Russia)

УДК 681.513

Дубинин, В. Н.

Преобразование автоматных спецификаций в функционально-блочную реализацию системы управления сортировкой последовательностей деталей / В. Н. Дубинин, И. В. Сенокосов, В. В. Вяткин, Л. П. Климкина // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2017. – № 2 (42). – С. 3–18. DOI 10.21685/2072-3059-2017-2-1