Serdica J. Computing **11** (2017), No 2, 137–164

Serdica
Journal of Computing

Bulgarian Academy of Sciences
Institute of Mathematics and Informatics

WORKFLOWS IN DYNAMIC SOFTWARE SYSTEMS

Hristo Yonchev, Boyan Bontchev

ABSTRACT. Business software systems and applications use business process management concepts to organize and automate processes. Incorporating a business process management system in existing business applications is a pretty complex task and, as well, is very expensive in terms of time and human resources. The article presents a newly conceived Framework for Dynamic Business Applications—a software platform that allows business users (domain experts) to define at run time all business objects that they will work with, their properties, relationships, and restrictions of their business/domain model. This is very similar to what can be done with business ontologies regarding representation of entities, ideas, and events, along with their properties and relations. After defining them, business users can use them to populate data in the application and later apply this data for data extraction, analysis, visualization, and reporting.

1. Introduction. Today business applications are not just screens for editing or viewing business data, they require complex logic to be executed

ACM Computing Classification System (1998): D.2.2, D.4.1, H.4.1.
Key words: workflows, management, BPM, FDBA.

against this data, which is often called business logic and can be defined as a set of business rules. Business Process Management (BPM) is defined as a support of "business processes using methods, techniques and software to design, enact, control and analyze operational processes involving humans, organizations, applications, documents and other sources of information" [1]. As well, BPM serves for optimization of business processes in a way which does not depend on the related technological support [2]. Thus, BPM represents a structured approach, which serves for discovery, design, execution, monitoring, and documentation of business workflows [3]. These workflows can be of both automated and non-automated types and have to be consistent and well-targeted and aligned with the organizational strategic goals [4]. BPM is a discipline that combines knowledge from information technology and knowledge from management sciences and applies this to operational business processes [5, 6]). On the other side, workflows may be regarded as "one fundamental building block to be combined with other parts of an organization's structure and information technology (which include business software), teams, projects and hierarchies" [7]. This makes workflows crucially important because they help BPM processes to be optimized and aligned with the business strategy of an organization in order to bring improvement of its overall performance and to reduce production and management costs [8].

Process-Aware Information Systems (PAIS) comprise traditional Workflow Management (WFM) systems and modern BPM systems, together with more enhanced, powerful and flexible platforms for management of specific processes [9, 10]. Thus, PAIS represent a large family of various systems such as platforms for Enterprise Resource Planning (ERP), e. g., SAP [11] and Oracle [12], Customer Relationship Management (CRM) systems [13], custom case-handling and rule-based systems [14]. All these types of systems can be regarded as *process-aware*, because, to varying degrees, they are oriented to description, execution, control, and monitoring of specific business processes. In many cases, PAIS may use other systems to support some steps of process execution, for example updating database records or sending e-mails through an e-mail server. However, in these cases, the database management system and the e-mail server serve as auxiliary systems, which are somehow involved in the management and orchestration of the business processes but

are not aware of the whole process [15]. Another example of BPM techniques that are not aware of the processes is process mining [16] that is used to discover and analyze newly emerging processes supported by the system [15].

BPM applies many languages, standards, and notations, which can be divided into four main groups according to their main purpose: for execution, interchange, graphical, and diagnosis purposes [17]. These languages, standards, and notations allow viewing BPM as an extension of WFM. Traditional WFM primarily focuses on the coordination and automation of business processes [1, 18, 19]. On the other hand, BPM has a rather broader scope: "from process automation and process analysis to operations management and the organization of work" [15]. There are two main types of processes in any organization: workflow and information flow. Workflows are not industry-specific and can be easily reused [9]. Therefore, they appear exceedingly helpful in defining processes within any company and reusing them among organizations.

Modern organizations tend to use various software products for BPM. However, in many cases, essential business requirements are not taken into account during software design and development especially when using standard software tools [20]. On the other side, processes modeled by business analysts cannot be implemented directly in a software system because of lack of sufficient destabilization of the process descriptions [21]. A recent study about trends in the use of such products and services [22] asked more than 300 companies: "What BPM products and services is your organization currently using?". The answers reveal (Fig. 1) that preferences of BPM suites that can manage the runtime execution of a business process are going steadily up, unlike the other types of BPM products (graphics modeling tools, repository-based systems, tools for managing rule-based applications, and process monitoring dashboard tools), whose usage remains the same or goes down. The explanation of this trend lies in the fact that BPM products managing the runtime execution of business processes are *dynamic business applications*. Dynamic business applications are defined as "applications that can quickly adapt to changing business needs, competitive pressure, and market opportunity" [23]. Such software systems are highly appreciated by domain users due to their runtime customization, configuration, and on-the-fly

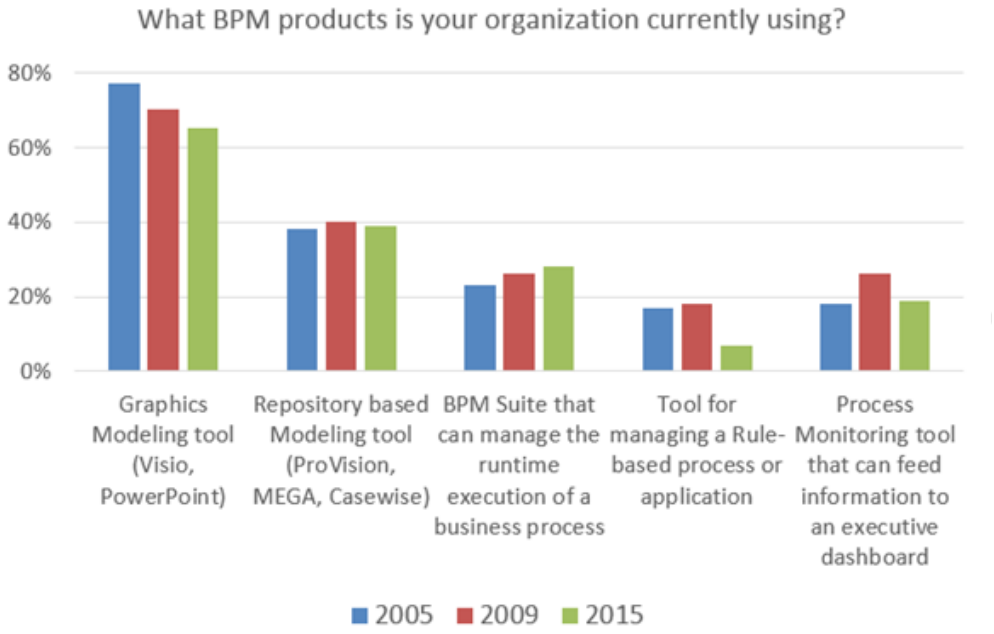

Fig. 1. Usage of BPM products in organizations (after [22])

extensibility according to the business requirements [20]. Modern business is very dynamic and fast-growing and, therefore, requires software that can serve its rapidly changing needs. At the same time, design, development, and support of dynamic business applications are very expensive and time-consuming. Moreover, it is not even possible for organizations without IT department—they have to either order a custom BPM system handling their specific business processes, or use an available PAIS and external consultancy services for building a customized solution [24]. The great challenge from scientific and applied point of view is to provide companies with a framework for straightforward, easy and cost-effective definition, managing and optimizing business processes by implementing workflows within dynamic software applications being adaptive to evolving business needs. The Framework for Dynamic Business Applications (FDBA) outlined here includes 'tools' to design, model, monitor, execute, manage and analyze workflows in order to fulfill the requirements for managing and optimizing business processes. The

article is focused mainly on features and means for modeling and monitoring workflows, including visualization and execution of workflows/processes, using an extensive library of predefined activities to allow 'programing' any kind of business process and, as well, integrating with existing systems by developing 'integrating' activities.

The rest of the article is organized as follows. The next section provides an explanation of our motivation behind this work, together with the most important related projects. Section 3 provides a description of the workflows in FDBA describing dynamic business applications, workflow activity types, predefined workflow activities, variables (objects that contain the data specific for a workflow), and workflow execution and extensibility. Section 4 is dedicated to a discussion of the important benefits offered by the described framework for workflow modeling and execution of business processes, together with an example of a business workflow for contract approval. As well, this section presents a comparison of FDBA with the most popular existing workflow framework—jBPM. Finally, the conclusion summarizes the functionalities of the FDBA workflow tools and provides some directions for future work in the area.

2. Background.

2.1. Motivation. The idea of this work is to define an appropriate framework for creating/modeling, executing, monitoring and changing business processes that can be used in existing business software systems and also will be part of FDBA and can 'handle' the dynamic nature of the business applications. This framework should be generic enough in order to be used in other existing/legacy business applications. Another requirement for it is to create/change business processes with no or minimum amount of software development (i. e., no programming required).

Workflows represent "computerized models of the business process, which specify all the parameters involved in the completion of these processes" and can be of administrative, collaborative, productive, and ad-hoc type [25]. Workflow is a set of activities which have an order of execution and which define a specific business short or long running process. Also, this can be described as an "ordered series of steps that accomplish some defined purpose

according to a set of rules" [26]. The next diagram shows a very simplified conceptual sample workflow with some activities (Fig. 2).

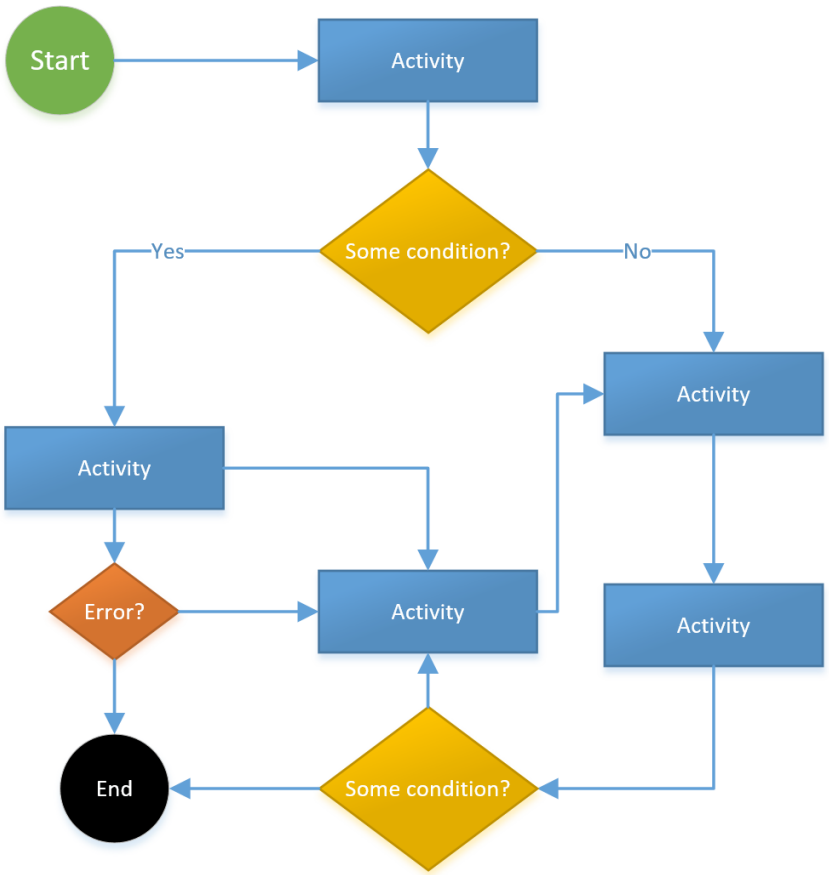

Fig. 2. Sample workflow with some activities

2.2. Related work. The section provides a short description of three existing frameworks/engines that allow workflow creation, execution, and monitoring.

2.2.1. Activiti. Activiti[1] is a light-weight workflow and BPM platform targeted at business people, developers and system admins. Its core is a super-

[1] http://www.activiti.org/

fast and rock-solid BPMN 2.0 [27] process engine for Java. It is open-source and distributed under the Apache license. Activiti runs in any Java application, on a server, on a cluster or in the cloud. It integrates perfectly with Spring while being extremely lightweight and based on simple concepts. It has a Designer tool that works with Eclipse. The representation of workflows is in XML.

The engine API is the most common way of interacting with Activiti. The figure below presents a sample activity diagram developed in Activiti.

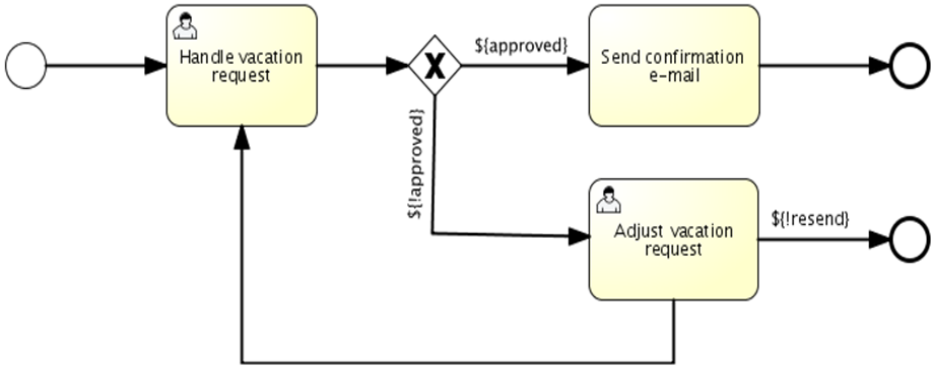

Fig. 3. Sample workflow with some activities

2.2.2. Camunda. Camunda[2] is an open source platform for modeling and executing workflows. It has a basic desktop designer (Fig. 4). Design documents can be saved as XML and later loaded and run by an execution engine. It is very easy to use, which allows business analysts to use it as well as developers, working on the same diagrams. Figure 5 presents a simple workflow developed with Camunda.

The main benefit of the platform consists in its performance: the Camunda engine is designed to process a high number of transactions (process instances) in a short time. Camunda achieves this in various ways, including caching, by reducing the interaction with the database to the necessary minimum and separating runtime data and history data.

[2] `http://camunda.com/`

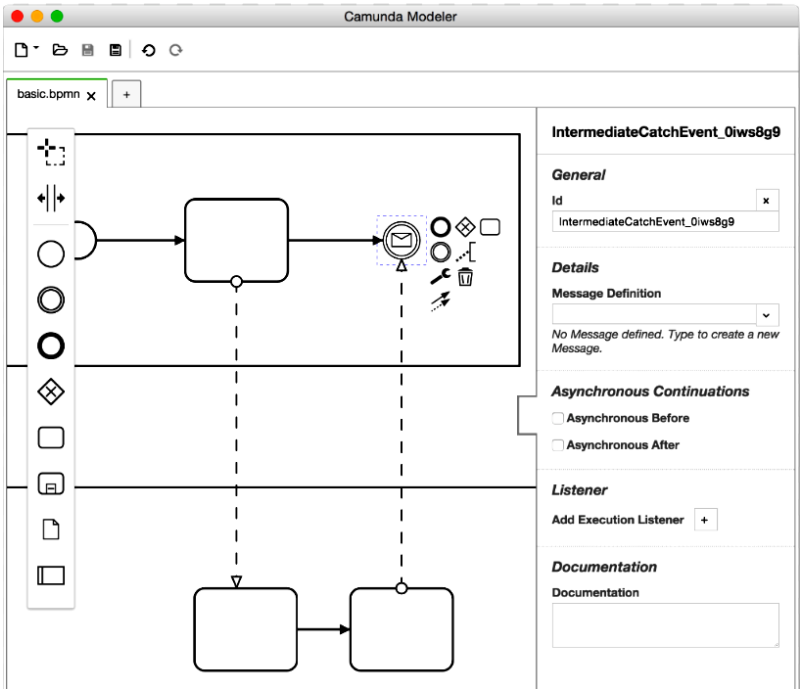

Fig. 4. View of the Camunda designer

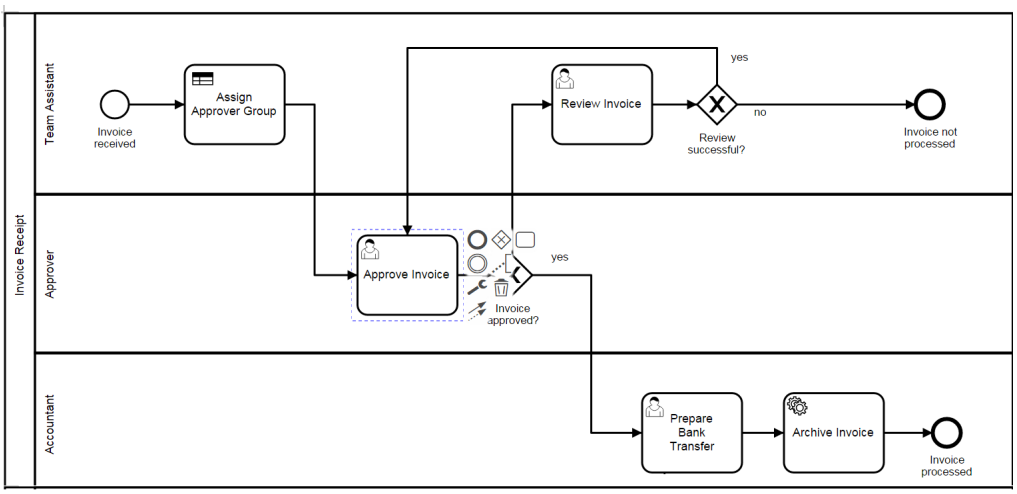

Fig. 5. Simple workflow with Camunda

2.2.3. jBPM. jBPM[3] is a flexible BPM Suite. It makes the bridge between business analysts and developers. Traditional BPM engines have a focus that is limited to non-technical people only. The core of jBPM is a light-weight, extensible workflow engine written in pure Java that allows you to execute business processes using the latest BPMN 2.0 specification. It can run in any Java environment, embedded in your application or as a service. On top of the core engine, many features and tools (Fig. 6) are offered to support business processes throughout their entire life cycle and also support the graphical creation of your business processes (drag & drop).

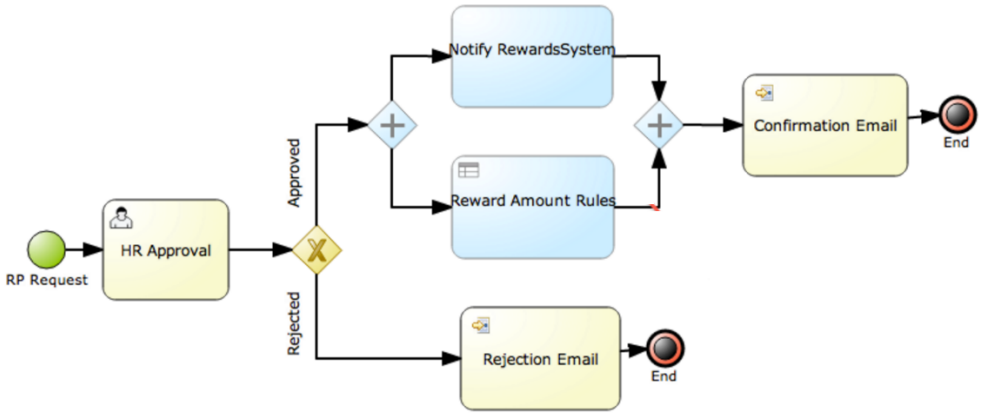

Fig. 6. Screenshot of jBPM designer

The most important features of jBMP (that are also a part of FDBA Workflows) are as follows:

- pluggable persistence and transactions based on persistence APIs;
- pluggable human tasks that need to be performed by human actors;
- management console supporting process instance management, task lists, and task form management, and reporting;
- optional process repository to deploy your process (and other related knowledge);

[3] https://www.jbpm.org/

- history logging (for querying/monitoring/analysis);
- integration with other systems.

2.3. The challenge. All three workflow tools described above in brief provide features for graphical edition of workflows, persisting workflows (mostly in XML and BPMN2), and execution and monitoring of defined workflows. However, they miss some functionalities, which appear crucially important for dynamic business applications, such as the following:

- predefined activities—extensive library of predefined activities to allow programming any kind of business process;
- a zero-coding approach—the ability to create fully-fledged business applications based on workflow diagrams, rules, and user interface descriptions;
- need of new activity both to load dynamic data and to import/export data;
- easy integration with any kind of business software systems—for example, by providing 'integrating' activities.

Thus, the main challenge of the present work lies in the research question: "Can we design and develop a framework for straightforward, easy and cost-effective defining, managing and optimizing business processes by implementing workflows as dynamic software applications, which are adaptive to evolving business needs?". The Framework for Dynamic Business Applications, presented in the next sections, provides tools to design, model, monitor, execute, manage and analyze workflows in order to fulfill the requirements for managing and optimizing business processes, together with features and means for modeling, visualization, execution, and monitoring business processes. It offers an extensive library of predefined activities to allow programming any type of business processes and, on the other hand, integrating with existing systems.

3. Workflows in FDBA. We are interested in a special type of workflows—these that will be used in a FDBA and, as well, can be reused in

legacy software systems that will benefit on adding automated workflows. FDBA is a framework that allows business users (domain experts) to define all business objects that they will work with and their relationships, and we need to have a 'tool' to create 'programs' and processes that will use the data in the business objects.

3.1. Workflow describing dynamic business applications. Workflows as part of the FDBA will serve as "Business Logic" or "Business Processes". They are implemented as a set of discrete, testable components that are assembled into workflows like building blocks. In FDBA, the users can create any kind of Business Object Types (BOTs) and, later, can create instances/objects of these BOTs. Here, business applications are not just applications executing CRUD (Create, Read, Update and Delete) operations against the objects in the application, but also various business processes (long, short, simple, complex) that work with the business data. To create low to high level 'programs' and processes in FDBA, we will use workflows, which will be used by the business to define their operational business processes.

In FDBA, a business process is a collection of related, structured activities or tasks that produce a specific service or product (serving a particular goal) for a particular customer or customers. We can divide business processes into three main categories:

- **Management processes** that govern the operation of a system. Typical management processes include corporate governance and strategic management.
- **Operational processes** that constitute the core business and create the primary value stream. Typical operational processes are purchasing, manufacturing, marketing, and sales.
- **Supporting processes** that support the core processes. Examples include accounting, recruitment, and technical support.

A business process can be decomposed into several sub-processes, which have their own attributes, but also contribute to achieving the goal of the super-process.

We can make an analogy to the traditional programming model, where the developer implements any task using source code in some specific programming language—for example, Java, C# or another general-purpose, object-oriented programming (OOP) language. In the source code, the developer specifies the instructions in some sequence (flow of control) and also code-decisions statements like `if/else`, `while` and so on. These are all based on some variables/data. In this way, any kind of business process can be programmed, but the process is written in source code and it cannot be changed without recompilation and requires software developers to change it.

Unlike the general-purpose OOP languages as Java or C#, workflows can be considered as languages for creating rules/processes for a specific problem domain. Domain experts can change the workflows because they are understandable and more user-friendly than the source code. As well, they can be changed by the domain experts unlike the source code, which once compiled is not changeable. However, in order to be fully completed and expressive as a programming language, FDBA workflows will have all needed activities that correspond to instructions of programming languages, namely:

- declare a variable and set a value to a variable;
- update object fields/properties;
- create methods and pass parameters;
- execute a loop.

Other benefits of using workflows are as follows:

- all the business processes in one business application are created in a consistent way;
- the set of activities can be extended with specific or custom activities for some business and thus provide reusability and extensibility of the workflows;
- a workflow can be stopped/paused in the middle and persisted for later continuation.

Creating workflows follows a declarative way of programming and consists of four tasks:

1. Specify which activities will be used in one workflow.
2. Specify the order/sequence of these activities.
3. Set variable values and activity's properties values.
4. Pass variables to activities.

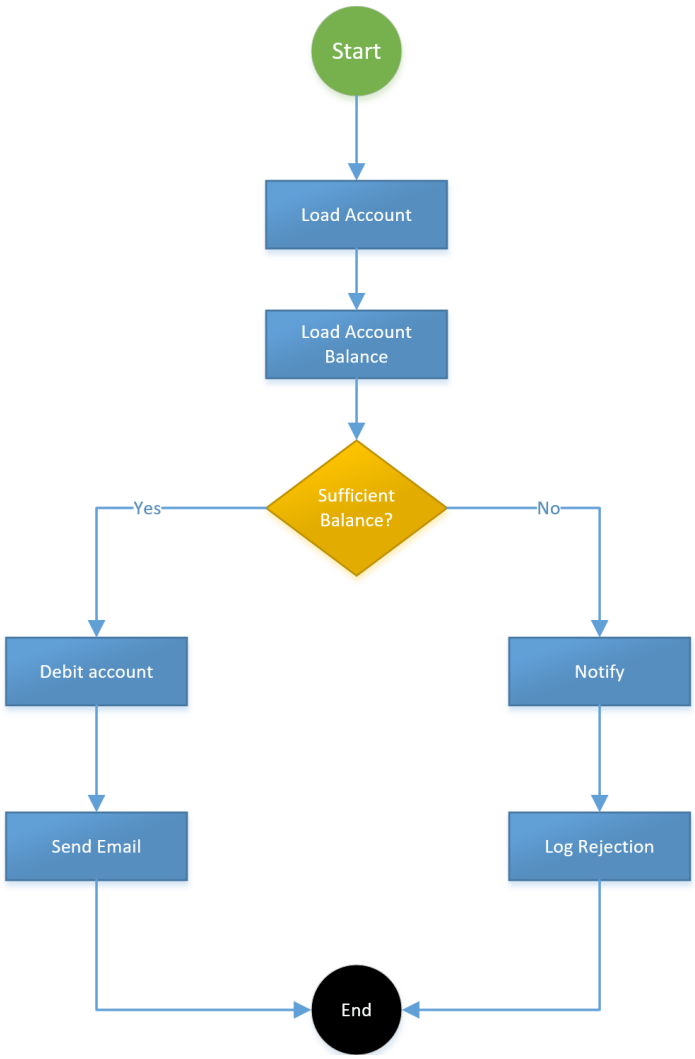

Fig. 7. Sample workflow for "Debit Account"

The ultimate purpose of the workflows as part of FDBA is to enable business users to 'implement' the business logic/business processes, which are part of the business applications. In order to successfully define workflows for business logic/business processes as a complete working solution, the following issues should be implemented:

1. existence of a predefined set of activities;
2. ability to inherit the predefined activities;
3. ability to order activity in diagram in specific sequences—modeling workflow diagrams;
4. runtime where workflows are executed;
5. ability to see/edit all defined workflows.

In order to have working workflows which can be used by the business we need also:

- execution of the modeled workflows;
- interaction with a human who can make a decision as part of one workflow, so the workflows must have activities that can be used to interact with the users;
- monitoring the running and executed business process.

Fig. 7 gives an example of a workflow for the process "Debit Account".

3.2. Workflow activity types. One workflow is a set of activities; they are building blocks of workflows. The activities can have a user-friendly name in order to be more readable when the workflow is displayed; so all activities have the `DisplayName` property. Activities can be of several types, as follows:

- **Control flow**:
    - `If`—Checks/tests a 'condition' and continue with one or another activity depending on the result;
    - `While`—Loops until a 'condition' value is met.

- **Messaging**:
  - sends message to web service;
  - sends email;
  - sends message to message queue.
- **Simple primitives**:
  - `Wait`—waits for specific time;
  - `Set`—waits for variable/property.
- **Data transactions**:
  - `Load/Delete/Update` objects.
- **Error handling**:
  - logging—logs information/error messages to system.
- **UI (user interface)**:
  - shows message to the users once open the system;
  - shows information/error message as notification.
- **Start workflow**—start a workflow at some event, for example:
  - after another workflow completion;
  - before another workflow;
  - on some event in the system;
  - on some trigger (data change) in the system.
- **Data**—activities to export/import/transport data/objects.
- **PL level**—activities, which correspond to programming languages instructions; extensive number (around 200) of activities like:
  - assign value to variable or array;
  - invoke method;
  - pass parameters;
  - sort list;
  - create objects;
  - others.

3.3. Predefined workflow activities. For it to be easier to create complex workflows, we provide an extensive number of predefined workflow activities. Any sequence of activities can be considered also as an activity in order to be able to create a library of new activities. There is a special type of activity, called `ExecuteWorkflow`, which can be used to nest one workflow in another. One workflow can be composed of activities and workflows and some of the activities are 'container' activities. In this way, we can define any kind of process, no matter how complex it is.

Table 1 explains some of the activities, which in workflows are connected with 'links'. Only some basic activities are covered; the entire list features around 400 prebuilt ones that can be extended, reused or grouped in 'container' activities.

Table 1. List of some of the predefined activities in FDBA

Activity	Description
If Else	Test a condition and execute one of the next activities
While	Loop until a condition is met
DoWhile	Loop until a condition is met
For	Loop specific number of times—N times. The Index property is changed from 1 to N for each iteration
Parallel	Execute activities in parallel
ForEach	Loop in a collection of objects and set property `CurrentItem` for each object
Sequence	Serve as container for other activities that will be executed in sequence. Also known as Composite activity
Assign Variable	Assign value to variable. Variables have scope—activity or workflow
Workflow	Activity to start another workflow
Send email	Sends email to specific email address. Email template can be used. Also objects can be exported into email content or attached as files
Load object	Loads specific object by Id or another filter condition (where clause). This activity is specific for FDBA, it will load object of specific BOT
Load objects	Loads many objects that met filter condition (where clause). This activity is specific for FDBA, it will load objects of specific BOT

Activity	Description
Delete object	Deletes specific object. This activity is specific for FDBA, it will delete objects of specific BOT
Update object	Sets properties of an object. This activity can be inherited in order to set specific property and have better name, for example, "Sign Document" will update the property `IsSigned` for BOT "Document"
Create object	Creates object of specific BOT
Create temp object	Creates non-persistent object that live only temporary in the workflow
Send message to queue	Sends message to message queue
Read message from queue	Reads message from message queue
Event Handler	Executes the next activity when specific even in the system is raised
Raise event	Raises event that later can be handled by another workflow or another component in the system
Trigger Handler	Executed when specific object or object property is changed
Switch	Evaluates a specified expression and executes the activity from a collection of activities whose associated key matches the value obtained from the evaluation
Send message	Sends message to web service
Read message	Reads message from web service
Delay	Waits specific time
Write in Log	Writes information/error message in the log
Write in Console	Write information/error message in the console
Transaction	Container for activities executed in Transaction
Rollback Transaction	Rollback transaction
Commit Transaction	Commits currently executing transaction
Add To Collection	Adds object to specific collection
Remove From Collection	Removes object from collection of objects
Catch Error	Catches error executed by some of the previous activities in a workflow
Throw Error	Logs and throws an error
Save Changes	Saves changes (if any) made by previous activities in a workflow execution

Activity	Description
Export Objects	Exports some objects to file/database (DB)
Import Objects	Imports some objects from file/DB
Convert Objects	Converts objects from one BOT to another BOT
Aggregate Objects	Aggregates objects to other BOT using aggregate functions
Merge Objects	Merges two collection of objects
Join Objects	Joins two collection of objects using join condition. The result is one collection of different BOT. Joins can be FULL, LEFT, or RIGHT
DB Script	Executes SQL script/script file
Sort Objects	Sorts objects
Extract from XML	Extracts objects from XML
Export Object(s) to Word, Excel, Others	Exports objects to word, excel or other template
Timer	Executes the specified activity when timer ticks; can be recurrent timer (for example "Every Monday" in January)

3.4. Variables. Variables represent objects that contain the data specific for the workflow or specific activity in the workflow during its execution. Variables have data in them only during execution; after the workflow completion, they lose their values, so they are not persisted after the workflow completion, they are only persisted if we persist the workflow in the middle of its execution.

The variables can be set by using Assign activity or their value can be set during workflow design and during the workflow execution. If a variable is specific for the workflow, its value can be read/written by any activity in this workflow. If a variable has scope concrete activity, only this and its sub-activities can read/write this value. For example, if we have four activities in sequence: Load Object, Update Object, Save Changes and Send Email, and we also have one variable of BOT "Product" with name "product", the sequence of actions that will be done during workflow creation/design will be as follows:

1. Create a variable "product" of type "Product" and set its scope as 'workflow scope' ('workflow scope' is the default scope, the other possible scope is 'activity scope').

2. In the 'Load Object' activity, load the required object and set it to "product" variable.
3. In the 'Update Object' activity, set some properties of "product" object.
4. `SaveChanges()` detects if there are some changes of business objects.
5. In 'Send Email' activity, "product" can be used to send this data to some email.
6. The "product" variable is available in the next activities in the workflow.

Note that this is just a sample approach to do this task; this can be done in many different ways.

3.5. Workflow execution. The execution of workflows will be done by a so-called "Workflow Processing Engine" (WPE). It allows both synchronous and asynchronous workflow execution.

An instance of the workflow is created before it is executed. It is called "Workflow Instance". This instance can be persisted/serialized if needed. The persisted workflow is saved in the DB. Workflows are executed in the same process where the business logic layer is executed in FDBA, not in a separate process. The Workflow Instance can be 'paused' and persisted between activities, but not in the middle of an activity (which is not a container like `Sequence` or `Workflow`).

Different events are raised during workflow executions. Some of them are as follows:

- `WorkflowStarted`
- `WorkflowTerminated`
- `WorkflowCompleted`
- `WorkflowPaused`
- `WorkflowPersisted`
- `WorkflowLoaded`
- `WorkflowResumed`

Note the difference when saving Workflows and Workflow Instances. When we save Workflow we save its definition—what activities it has, what is

their order and sequence, what variables are used and so on. When we save a Workflow Instance, we persist the current state of the executing Workflow Instance. One Workflow can have multiple Workflow Instances. The relation is the same as that of class to object in object-oriented programming.

3.6. Workflow extensibility. Any activity type can be inherited (by inheriting the abstract class `Activity`), so new activities can be defined, e. g., `ValidateAccountActivity`, `ValidateProductActivity`, `ChangeSystemSettingsActivity` and so on. The newly defined activities are later available in the graphical workflow designer where they can be used as activities in the new workflows.

Any activity is implemented by a class that has a name similar to its own. For example, the "If Else" activity class is called `IfElseActivity`, the Timer activity has a class `TimerActivity` that implements it. All activity classes implement the interface/abstract class `Activity`. The main attributes and properties of this class are as follows:

- `DisplayName`—the name displayed in the graphical tool for designing workflows;
- `Id`—identifier of activity instance unique in the application;
- `Serialize()` method;
- `Deserialize()` method;
- `Invoke()`—the actual implementation of the activity;
- `Variables`—list of variables specific for this activity;
- `HasError`—a Boolean property indicating the occurrence of an error;
- `Error`—description of the error (if such has occurred).

Workflows also implement the abstract class `Workflow`, which implements the abstract class `Activity`. The activities can be grouped in a `Sequence` activity and this activity can be saved and later reused. This new activity is later available in the graphical workflow designer, where it can be used as an activity in the new workflows (note that activities newly 'programmed' by software developers are not restricted to using the convention `SomeActivity`, they only need to inherit the abstract class `Activity`).

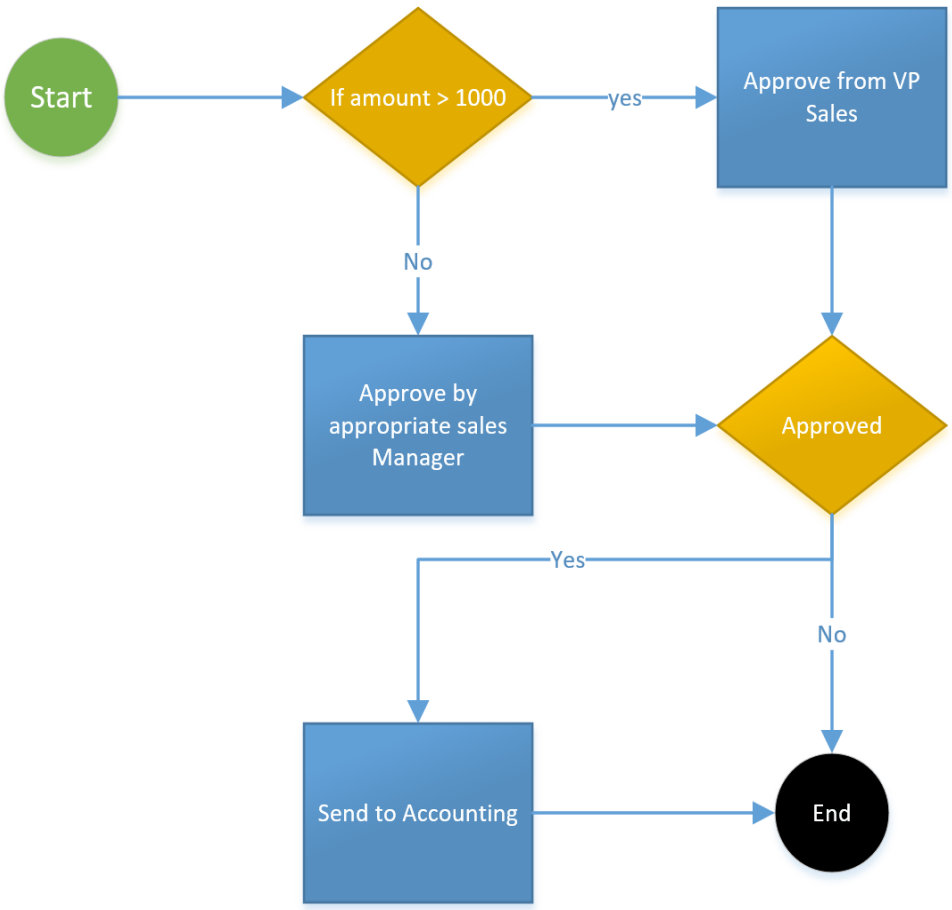

Fig. 8. Business workflow for contract approval

4. Discussion. The framework for workflow modeling and execution described in the previous section possesses several important advantages: it

- creates business processes according to the business needs;
- creates/manages processes which with high complexity;
- provides a list of predefined activities to create other activities and workflows.

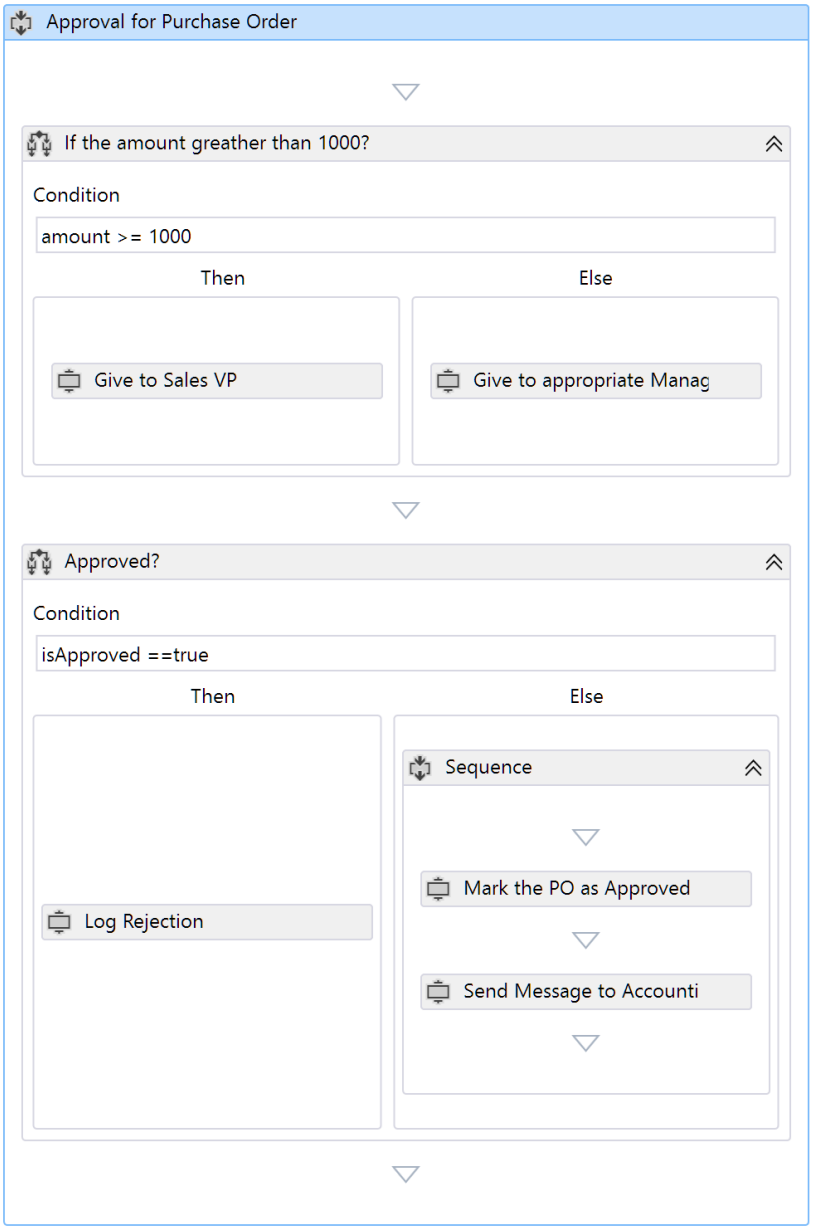

Fig. 9. Partial view of activities of the business workflow for contract approval

Fig. 8 presents a business workflow described by the framework for contract approval, namely Approving Purchase Order. Here, we see a sample workflow, which can be executed when a Purchase Order (PO) is requested. This is a simple version that shows a need for approval for PO above a quantity of 1000. The next Fig. 9 provides a view of some of the activities of this business workflow.

Table 2 offers a comparison of FDBA with the most popular existing workflow framework—jBPM. There are compared features important for dynamic workflows creation. The main advantages for business processes developers are as follows:

- persistence—using both DB records and BPMN2;
- zero-coding—no need of new development (programming) of new activities;
- graphical user interface—ability to create/view/edit the workflow in a graphical tool. Also, an ability to see graphically currently executing workflows—in order to answer the questions: "What is the current execution status of the workflow?", "Which step/activity is currently executing?", "Why did the workflow finish with an error?", and others;

Table 2. Feature comparison between FDBA workflows and jBPM

Criteria	FDBA Workflows	jBPM
Zero-coding approach	Yes	No
Graphical Workflow Editor	Yes	Yes
Web Graphical Workflow Editor	Yes	Yes
Platform	.Net/Java	Java
Persist workflow as	Database records or BPMN2	BPMN2
Easy to use	Yes	No
Need new activity to load dynamic data	No	Yes
Need new activities to import/export data	No	Yes
Easy integrate with any kind of business software systems	Yes	No
Extensive list of predefined activities	Yes	No

- work with dynamic model—because the model is dynamic, the workflows can contain activities to work with this dynamic model—for example to load data from new business objects defined by the users of the system;
- import/export activities provide 'integration point' with other systems.

5. Conclusions. Modern software platforms need automated, extensible workflows for BPM systems. Adding a BPM system in organizations is very expensive and time-consuming, especially when the company already uses software and business data that need to be integrated with the BPM system. With FDBA workflow tools, business users and software developers have the liberty to use plenty of functionalities, such as:

- design, model, and monitor workflows;
- visualize (using diagrams) workflows/processes;
- execute, persist and resume workflows;
- use an extensive library of predefined activities to allow 'programing' any kind of business process (high or low level, long-running or short-running process;
- extend, reuse, group existing activities or create new activities (done by software developers);
- create new activities from existing or by grouping activities (done by business/domain experts);
- integrate with existing systems by developing 'integrating' activities;
- interact with the humans when decisions need to be made.

Therefore, FDBA workflows will help business with modeling, automation, execution, control, measurement and optimization of business activity flows, in support of enterprise goals, spanning systems, employees, customers, and partners within and beyond the enterprise boundaries. By applying FDBA, companies will improve business performance outcomes and operational agility beyond the currently existing levels.

The article presented only one core feature of FDBA—handling business workflows following the ever-changing functional and organizational

requirements. Further work will address other user functionalities of FDBA such as creation and maintenance of user interfaces for dynamic business applications. As well, the software architecture of the framework should be precisely formulated in accordance with a formalized conceptual model for the creation of such applications. Finally, we have to validate practically the framework concerning its effectiveness—meaning the expected results regarding their usage, i. e., how efficiently it serves for creation, execution, and monitoring of any business workflow embedded into a dynamic business application. As well, its efficiency will be validated—i. e., whether it is more efficient regarding both cost and time of software design and development when compared to other similar approaches and platforms.

REFERENCES

[1] Van der Aalst M. P., K. van Hee. Workflow Management: Models, Methods, and Systems, MIT press, Cambridge, Mass., USA, 2004.

[2] Grüne G., S. Lockemann, V. Kluy, S. Meinhardt. Business Process Management Within Chemical and Pharmaceutical Industries: Markets, BPM Methodology and Process Examples. Springer Science & Business Media, 2013.

[3] ABPMP Standards for Business Process Management (BPM), `http://www.abpmp.org/?page=BPM_Profession`, 18 February 2018.

[4] Jeston J., J. Nelis. Business Process Management, Routledge, 2014.

[5] Van der Aalst W. M. P. Business Process Management Demystified: A Tutorial on Models, Systems and Standards for Workflow Management. In: J. Desel, W. Reisig, G. Rozenberg (eds). Lectures on Concurrency and Petri Nets. Springer, Berlin, Heidelberg. *LNCS*, **3098** (2004), 1–65.

[6] Weske M. Business Process Management: Concepts, Languages, Architectures. Springer, Berlin, Heidelberg, 2007.

[7] NocSmart Services. Work Flow Automation, https://web.archive.org/web/20130907014418/http://nocsmart.com:80/index.php?option=com_content&view=article&id=17&Itemid=135, 18 February 2018.

[8] Freund J., B. Rücker. Real-Life BPMN: Using BPMN 2.0 to Analyze, Improve, and Automate Processes in Your Company, Camunda, 2012.

[9] Madison D. Process Mapping, Process Improvement and Process Management. Paton Professional, 2005.

[10] Dumas M., W. M. P. van der Aalst, A. H. M. ter Hofstede (eds). Process-Aware Information Systems: Bridging People and Software through Process Technology. John Wiley & Sons, Hoboken, NJ, 2005.

[11] Buciuman-Coman V., M. Chervenic. Beyond SOA: A New Enterprise Architecture Framework for Dynamic Business Applications, March 28, 2008. https://www.infoq.com/articles/dynamic-business-applications, 18 February 2018.

[12] Quaglini S., M. Stefanelli, G. Lanzola, V. Caporusso, S. Panzarasa. Flexible guideline-based patient careflow systems. *Artificial intelligence in medicine*, **22** (2001), No 1, 65–80.

[13] Buttle F. Customer relationship management: concepts and technologies. Routledge, 2009.

[14] Müller R., U. Greiner, E. Rahm. Agentwork: a workflow system supporting rule-based workflow adaptation. *Data & Knowledge Engineering*, **51** (2004), No 2, 223–256.

[15] Van der Aalst W. M. P. Business Process Management: A Comprehensive Survey. *ISRN Software Engineering*, Article ID 507984, 2013, 37 p. doi: 10.1155/2013/507984

[16] Van der Aalst W. M. P. Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Berlin, Heidelberg, 2011.

[17] Ko R. K. L., S. S. G. Lee, E. W. Lee. Business process management (BPM) standards: a survey. *Business Process Management Journal*, **15** (2009), No 5, 744–791. doi: 10.1108/14637150910987937

[18] Jablonski S., C. Bussler. Workflow Management: Modeling Concepts, Architecture, and Implementation. International Thomson Computer Press, London, UK, 1996.

[19] Leymann F., D. Roller. Production Workflow: Concepts and Techniques. Prentice-Hall, Upper Saddle River, NJ, USA, 1999.

[20] Scheer A.-W. ARIS — vom Geschäftsprozess zum Anwendungssystem. Springer-Verlag, 2013.

[21] Grüne G., S. Lockemann, V. Kluy, S. Meinhardt. Business Process Management Within Chemical and Pharmaceutical Industries: Markets, BPM Methodology and Process Examples. Springer Science & Business Media, 2013.

[22] Harmon P. The State of the BPM Market—2016. A BPTrends Report. Business Process Trends, 2016.

[23] BPM CBOK version 3.0: Guide to the Business Process Management Common Body of Knowledge. ABPMP Int., Saint Paul, MN, 2013.

[24] Rymer J. R., C. Moore. The Dynamic Business Applications Imperative. Forrester Research, 2007.

[25] Alonso G., D. Agrawal, A. El Abbadi, C. Mohan. Functionality and limitations of current workflow management systems. *IEEE expert*, **12** (1997), No 5, 105–111.

[26] Bukovics B. Pro WF: Windows Workflow in .NET 4. Apress, 2010.

[27] About the Business Process Model and Notation Specification Version 2.0. January, 2011. `http://www.omg.org/spec/BPMN/2.0`, 18 February 2018.

Hristo Yonchev
Department of Software Technologies
Faculty of Mathematics and Informatics
St. Kliment Ohridski University of Sofia
5, J. Baurchier Blv
1164 Sofia, Bulgaria
e-mail: `ristoky@fmi.uni-sofia.bg`

Boyan Bontchev
Department of Software Technologies
Faculty of Mathematics and Informatics
St. Kliment Ohridski University of Sofia
5, J. Baurchier Blv
1164 Sofia, Bulgaria
e-mail: `bbontchev@fmi.uni-sofia.bg`

Received October 17, 2017
Final Accepted November 15, 2017