Opus Framework: Issues, Challenges, and Assessment

Nahla Haddar *, Mohamed Tmar † and Faiez Gargouri ‡

* Computer Science Department, Faculty of Economics Sciences and Management
Sfax, St. Airport Km 4, 3018 Tunisia
nhaddar@ymail.com

†‡ Computer Science Department, Higher Institute of Computer and Multimedia
Sfax, St. Tunis Km 10 B.P. 242, 3021 Tunisia
{mohamed.tmar, faiez.gargouri}@isims.rnu.tn

Abstract

After an exhaustive study in data-driven workflow management researches, we found that many drawbacks of existing approaches are related to the insufficient integration of data and processes. As a result, most of the existing Workflow Management Systems (WfMSs) present data in an abstract way, without being able to manipulate them. Opus system project aims to provide a comprehensive association between data and processes in order to overcome the fundamental limitations of the existing systems. So, in this paper, we are going to motivate the issues and the challenges addressed in our research, and then to evaluate our Opus system relatively to the most emerged data-driven WfMSs.

keywords: Data-driven workflow management, process modeling, process analysis and verification.

1 Introduction

In real world, there are many processes that are heavily related to knowledge (knowledge-intensive) and dynamism. These are processes whose participants decide about the processing and the order of their activities, depending on the available data. Data in processes are manipulated as objects. These objects present an abstraction of real entities of specific domains (e.g., orders and bills in a business process, applications and interviews in a human resource process, etc.). Each object is represented by a set of attributes (e.g., the attribute describing the status of a bill payment, the name of an applicant in an application object), and plays a fundamental role in the deployment of the process (e.g., after the assessing of an application, the applicant is accepted for making an interview or not). Therefore, the synchronization between objects and the process is necessary. Indeed, it is not sufficient to model the process only in terms of atomic activities.

The identification of this need has guided the interests of researchers to data-driven processes. Unlike the modeling approaches centered activities, data-driven approaches allow a particular step in a process that is not directly depending on the completion of the previous steps, but rather changes in some attribute values related to an object.

Many authors, such as [2, 16, 19, 27], have proposed approaches to allow a close integration of processes and data. However, most of them often provide default activities (called atomic or black box activities, e.g., a web service or an executable artifact) which are running non-specific tasks to individual needs of an organization [7]. As a result, the resulting processes reflect only a macroscopic view on the real work, and there is a contradiction between the way these processes can be defined and the preferred work practice [1]. Because of these limitations, most of business organizations use professional applications (e.g., Enterprise Resource Planning systems) instead of WfMSs. However, the resulting applications are both complex to design and expensive to deploy, in addition, the simplest changes in processes require hard code adaptation and expensive tests effort. The challenge ahead is to describe activities at the lowest level of granularity, taking into account their impact on the data transformation. In this paper, we locate our Opus system relatively to the most popular data-driven WfMSs. As well, we study the role of data in the routing logic and the definition of control flow at a level of granularity that allows a WfMS to manipulate data, which provides a data-driven execution customized to the management rules of an organization.

The remaining of the paper is organized as follows: first, Section 2 discusses the key challenges for Opus process management system. Section 3 presents the most emerged WfMSs to discuss how the existing systems support the integration of data and control flow. Then, Section 4 elucidates our Opus system and
the approach behind its implementation. Section 5 investigates the related work along a well defined evaluation schema. Section 6 concludes.

2 Problems, Challenges, and Requirements

In an exhaustive literature study, we have evaluated the properties of data-driven WfMSs. This section summarizes the problems found from this study. This leads us to number a set of fundamental requirements to be met by our Opus system.

2.1 Challenge 1: Integration of Data

Problems. Most of the approaches that result from research in the management of data-driven WfMSs, are inspired by the P-nets formalism [2, 4, 19]. This is due both to its graphical nature and its formal semantics allowing for processes verification. Despite the emergence of these approaches, their semantics influence the ability of workflow engines to manipulate data during the workflow execution. Indeed, the WfMSs based on these approaches, focus on the control flow perspective, i.e., they include generic functions for assigning tasks to actors (i.e., creating worklists for each actor), notifying actors on the most urgent tasks, ensuring the data routing between actors (e.g., presenting data as forms), or mapping data so that they can be used by invoked applications. However, the detail of the execution of activities (i.e., tasks / elementary actions of activities) and data manipulations, in turn, are out of the control of existing WfMSs, which implies the use of specific programming and the spending of more time and efforts as the modeling of the corresponding processes. Moreover, most of WfMSs often provide atomic activities which are running non-specific tasks to individual needs of an organization [7, 25]; i.e., data are managed by the invoked applications themselves. Thus, the operational perspective of a process is limited to the invocation of applications and the mapping of their input/output data.

Requirements. Our approach must consider that data are located above the definition of control flow [1], i.e., the definition of activities and routing rules must depend on data which ensures a data-driven execution of processes. Even the definition of actors performing the work must be around the data manipulated by their activities. In fact, each actor needs data to perform his activities which produce other data needed by other actors. As we manage data driven processes, we choose to decompose a process according to the data needed by an actor to perform his work. These actors are either individuals or groups (e.g., commercial service). The concept of role results from this classification [11]. To meet these requirements, the proposed formalism must be able to automatically manipulate data and integrate them with the existing Information System (IS).

2.2 Challenge 2: Verification of Process models

Problem. The verification of processes has been widely studied using P-nets. These researches have led to the definition of several verification properties, the most dominant is the soundness property. We are going to demonstrate in Section 5 that despite the abundance of the soundness verification techniques, those defined for data-driven processes are still incomplete.

Requirement. Our verification method must grant all the properties ensured by the classical soundness (i.e., livelocks, deadlocks, and dead tasks).

We assume that a comprehensive integration between processes and data promotes the efficacy and efficiency of the resulting workflows. This means, not only a generic process support but also generic activity details become possible.

3 Data-driven Workflow Management Systems

We present in this section the most popular workflow systems intended to provide a comprehensive integration between processes and data.

FlexConnect is proposed by [26, 27] to organize processes as a set of interactions of Object Life Cycles (OLCs), rather than flows of activities. These life cycles are defined by means of colored P-nets. Indeed, the object states correspond to the activities that must be carried out during the whole life cycle of an object. Nonetheless, FlexConnect only manages worklists of users, but the manipulation of data is carried through the invocation of external applications [16]. Besides, FlexConnect focuses on the coordination and the creative flexibility of unstructured processes, but it ignores the data manipulation [27]. Another drawback of FlexConnect is that it could not support the parallel execution of activities modifying the same object in the same process instance, while maintaining the data consistency [10].

PHILharmonicFlows [8, 16, 18, 19] manages data as inter-connected object types. Each type of object has a set of inter-related attributes and its behavior is defined in terms of states and transitions. Moreover, PHILharmonicFlows provides a set of consistency rules. The latter assume that a process can be completed successfully if it does not contain deadlocks, and if the deadlock occurs during execution, the system
guides the concerned user to its recovery. In addition, the system allows for the automatic generation of forms from data objects, and during the execution of a particular activity, the authorized users can manipulate the forms corresponding to the same object instance simultaneously. In this context, a mechanism for controlling the simultaneous access to data is applied. Nevertheless, the activities with advanced features, such as complex calculations, are classified by the system as atomic activities that must implemented by programmers or executed by Web services.

**Case Handling Paradigm WfMSs** (such as FLOWer [2], Staffware [29], and COSA Activity Manager [6]): the Case Handling Pradigm (CaseHP) [2] is centered on the concept of Case. The Case is the manufactured product, e.g., the evaluation of a job application. Each Case is a collection of data objects and activities. Each activity has a life cycle based on finite state machines, and is interconnected with other activities by a directed acyclic graph. Besides, WfMSs based on the CaseHP can handle two types of activities: The first type presents the activities interacting with the IS (a relational database or XML documents) to import / save the data that are manipulated by authorized users by means of forms; the second type presents activities as Web services. Moreover, as the CaseHP is originally based on Business Artifacts (BAs) [24], the method proposed in [12] to verify the termination (i.e., the process is deadlocks and livelocks free) property of BAs models is adopted by the CaseHP WfMSs. The main drawback of the CaseHP is that the definition of a Case can be associated with multiple roles involved in its processing, however, the approach does not define the necessary rules for managing the competition access to data, i.e., users can modify data of the same Case simultaneously without applying any data access controls, which affects the data consistency [10, 18].

ArtiFact system [10, 14] is developed to provide a graphical modeling and execution environment for Guard-Stage-Milestone (GSM) models. The GSM meta-model is recently proposed to specify the lifecycle of artifacts, by a more declarative way, with an operational semantics based on Event-Condition-Action rules. The schema of artifacts are saved in modeling step as XML schemas and their instances can be entered by the authorized users, or imported at run-time from a relational database. The management of the artifacts life cycles is ensured by the invocation of Web services (WSDL and REST services), and the routing conditions (guards) must be written by the process designers in JEXL or OCL languages. Accordingly, like most the other WfMSs, ArtiFact is intended to support the management of activities, but not the details of their execution [10]. Besides, ArtiFact is not using control mechanisms for concurrent access to data. But data consistency is ensured. Indeed, the system assumes that two activities that are writing in the same data attribute can be executed in parallel, and values written by the first activity can be replaced when the second activity is finished. Finally, to verify the correctness of ArtiFact process models, two formal verification techniques have been proposed by [13, 28], to verify, respectively, their termination and boundedness properties.

**Yet Another Workflow Language** (YAWL) [4, 30] extends the P-nets to provide a comprehensive support that is able to present all the control flow patterns defined in [3, 15]. Furthermore, it suitably supports data by treating them through XML standards [9]. In fact, YAWL defines data types as XML schemas. Besides, at run-time, when a process requires data from an external environment, either a web form is generated to enter data by the authorized users, or a Web service is invoked to provide the required data. Besides, YAWL can coordinate activities as Web services, or activities that read, write, apply a calculation expression, or compare data from XML documents. The conditional and parallel routings are specified by designers as XPath expressions. Moreover, the latest version of YAWL [31] simplifies the definition of data types by introducing a new language based on Pascal syntax, allowing for easier definition of data types. Then, the new syntax will be automatically transformed to/from XML schemas [5]. This version also enable the automatic generation of XQuery codes for querying XML data [5]. Finally, to verify the correctness of YAWL processes, Wynn et al. [30] propose a technique to verify the weak soundness property for YAWL models that contain OR-Join blocks or cancellation regions. For the other YAWL models the soundness property is decidable.

## 4 Opus Framework

Opus framework ¹ is implemented according to the approach described in [20–23]. The latter assumes that the decomposition of a process is related to the organizational criterion of an organization, so that an activity is performed by procedural roles. Each role defines a list of tasks related to the production of particular data. It can be human or system. Accordingly, a role is defined as a sub-process belonging to an overall workflow process.

---

¹All the features of Opus system, described in this section, are illustrated by the example of the Order management process provided in the following link: https://sites.google.com/site/wfmsopus/workflow-management-demo
The data handled by the workflow can be entered by a role, imported from the IS (relational database or XML documents), or produced by other tasks. In particular, each role work is presented as a P-net in which its places are relational entities, called Data Structures (DSs), dedicated to support structured tokens (i.e., data tuples), and its transitions are data operations some of which are inspired from the relational algebra [20, 21]. The use of the relational algebra is due to its theoretical basis allowing for the automation of the data management. The other proposed operations have various utilities, such as writing the values of an attribute, performing complex calculations, presenting conditional and parallel routing, presenting iterative routing, invoking Web services, mapping of heterogeneous data, and integrating the process with the IS. Thanks to these operations our approach can present the details of most activities manipulating data, which leads to create workflow applications that really reflect the preferred work practice.

In order to maintain the data consistency during parallel writing in the same data element, our framework ensures that when one of the parallel operations wrote in an attribute of a DS, the other operations are blocked until the termination of the first one. Furthermore, we assured the modeling of a well-formed process through the definition of a method to verify a relaxed soundness property [22]. Particularly, our verification method ensures that there are no livelocks, deadlocks, or dead tasks in process models that contain many initial and final states. Finally, in [23], we extended our meta-model by adding a multidimensional aspect to the relational data handled by our workflow models. Indeed, the extension allows generating multidimensional data models (OLAP hypercubes) from the DS instances.

All the theoretical aspects described above have been implemented into Opus system. The system consists of a number of components including a modeling editor, a workflow engine, a matching tool, a verification module, and a reporting tool. The modeling editor is equipped with a set of graphical interfaces to model workflow processes. Using this editor, the designer is not intended to know the formal aspect of the data operations. In fact, the editor provides all the necessary assistants to handle the designer during the modeling step. Besides, in case of changes in a role sub-process, the designer can easily update its sub-process model without damaging the other sub-processes. He can also use the verification tool to analyze his designed process models and verify their well-formedness [22] before their deployment.

Due to the formal definitions of the data operations, Opus engine can follow up the data flow routing, simulates the processing of operations, and automatically instantiates the resulting DSs of each operation. It is also equipped with the IS Integration Module. The latter uses the matching tool to map the data imported, saved, or updated in the IS. The main drawback of the engine, is that it instantiates data to end-users using a simple JAVA tool, i.e., the JTable component that presents data as tuples and allows the authorized users to manipulate a DS attribute values. Thus, the implementation of the engine must be completed to ensure the generation of forms from the handled DS instances, which allows users to manipulate data through forms.

Finally, Opus Reporting tool 2 allows modeling complex business reports from the DS instances, as easily as possible, by providing drag and drop tools to organize the report objects. Data in the report are visualized as multidimensional tables or statistical graphs, and generated from the current DS instance, according to the dimensions and the fact defined graphically by business analysts. The main drawback of this tool is that the analyzed data source is limited to the DS instances generated by the workflow instances, so it must be improved to support more data sources (relational tables, XML documents, Excel tabular, etc.).

5 Related Work

To discover the drawbacks of the existing data-driven WfMSs, we performed a detailed investigation by an extensive literature study. We find that the integration of data with processes differs from a WfMS to another. Some WfMSs (e.g., FlexConnect and PHILharmonicFlows) present data as objects and integrate them in the control flow by exploiting their life cycles presented as finite state machines. Some other systems (e.g., ArtiFact, FLOWer, Staffware, and COSA) extend the OLC approach by the use of BAs, to combine the informational and functional aspects of a process in a global unit (i.e., the artifact). And finally, we studied YAWL as the most emerged WfMS that extends the P-nets formalism by abstracting from data and presenting them as tokens. By detailing, in Section 3, the features and the drawbacks of each of these systems, we extract a set of evaluation criteria that allows us to evaluate our system relatively to them. In summary, our systems analysis has revealed the evaluation illustrated in Figure 1.

The signs in Figure 1 denote the following meanings:

(+): supported criterion; (-): unsupported criterion; (±): criterion partially supported; (a): activities are presented as black boxes; (b): Opus system presents

2 An example illustrating the functioning of this tool is provided via the following link: https://sites.google.com/site/wfmsopus/download
data as relational tables or as business analytic reports in PDF format; (c) only for complex calculations or comparisons; (d) only for a specific type of IS (Relational database for ArtiFact, and XML documents for YAWL); (e) black boxes activities may require an implementation by a programmer [19]; (f) the conditions of routing must be defined at a specific language (XPath expression for YAWL [31], OCL or JEXL languages for ArtiFact [10]); (g) The designer must had the necessary competence to use the relational algebra; (h) Only for coordination [16]; (i) the execution is directed by control flow and data [17].

Figure 1: Evaluation Schema of Opus system and existing data-driven WfMSs

We can deduce from Figure 1 that although most of the WfMSs, such as PHILharmonicFlows, CaseHP WfMSs, and ArtiFact, ensure a data-driven execution of processes, they failed in verifying all the basic properties (i.e. livelocks + deadlocks + dead tasks = liveness) for the proper functioning of their workflow applications. Nonetheless, whatever the type of the handled data, Opus framework ensures the verification of the well-formedness (i.e. liveness + boundness) of the conceived workflow models. However, unlike most of the systems that support reading and writing in data instances by means of forms, Opus system can manipulate data only by means of JTable components. In fact, the system does not support the generation of forms from DSs. Nevertheless, unlike most of the studied systems, Opus system supports the application of complex calculations and transformations on data, which allows executing activities according to specific business rules without the need of programmers.

6 Conclusion

In this paper, we exposed the challenges we undertaken in Opus project, and the problems of existing data-driven WfMSs behind each challenge. Then, we defined the basic requirements that our system must meet to overcome these problems. Finally, based on a detailed comparison of the emerged data-driven WfMSs, we elucidated the assessment of our system. Currently, we are developing the missing module of forms generation. In future work, we will elaborate more detailed issues in the context of our system, and we will apply it to real world processes for evaluation purpose (i.e., health care, human resource management, etc.).

References


