Object-centered Process Modeling

Rui Henriques

Center for Organization and Engineering (CODE) – INESC-ID
Instituto Superior Técnico (IST-UTL), Lisboa, Portugal
rui.henriques@inov.pt

Abstract. New modeling approaches appeared in the last decade centered on the premise that process structures in data-intensive landscapes are pushed by dependencies observed at the data level. However, as these approaches answer to reduced subsets of all data-related needs, they have a limited practical impact [12]. This work defines a set of data-based requirements to model responsive systems, studies emerging object-centered approaches to retrieve a set of principles and, finally, develops an event-driven solution that integrates such principles. Mapping of object-centered models to YAWL and case-applicability reveal the executability and adequacy of the new modeling approach to evolve data-intensive processes.

Key words: process modeling, data-intensive system, object-orientation

1 Introduction

The increasingly uncertain, dynamic and data-intensive landscape where some systems operate triggers challenges to process models *evolution*, either when processes need to be modeled from scratch or, as this work focuses, to organically adapt through local improvements. Since *data-intensive* systems highly depend on how their passive participants – system entities subjected to transformation – to prescribe its elements coordination, their evolution is pushed by changes at the data level. Exemplifying, a health-care system that relies on the state and mediation of patients, exams, reports and historics to deliver a diagnosis, *evolves* by changing the way these participants are constrained through process models in order to abstract and prescribe the new desired operation [28].

In traditional approaches, the modeling of processes is independent from the system data, historically hidden behind applications [22], disabling synergies between informational and functional views required for data-intensive scenarios.

Process models can create an environment for the evolution of data-intensive systems if they foster: *integration* by prescribing the relationships among system elements while bridging functional, informational, organizational, technological and contextual views [32], and *adaptability* and *agility* by promoting flexible and data-centered models with changes performed in a timely manner [18].

Motivation. Barriers for the evolution of data-intensive process models are pointed in [18][12]. Their resolution is particularly important for scientific workflow systems [1], manufacturing systems [21][17], government systems [7] or insurance systems [30]. Common *problems* of traditional approaches include the context-isolated enactment of activities causing data-access challenges and a loss of the process global view, the absence of criteria for the activities granularity and the rigidity required to specify networks of activities when loosely-coupled, dynamic and data-based constraints foster models flexibility and expressivity.

Results from research [6][17] support the fact that a system modeling integrating the data and process perspectives reveals opportunities that disrupt traditional modeling discipline. The natural outcome orientation of many administrative and operational processes [31] turns the progress of single process instances not directly dependent on the execution of activities but reactive on data changes [12]. Contrasting to traditional approaches that force the system modeling into monolithic processes, objects seem promising to model data-based processes as they capture the system operation as a collection of production-oriented intertwined loosely-coupled life-cycles running at different speeds at different levels of abstraction [28][12].

Since data-intensive processes rely on the premise that relations between the passive participants' components implicitly define sub-process dependencies [12], new ways of dynamically support processes' evolution can be exploited.

Contribution. This work proposes an analysis of the potentialities of emergent object-centered approaches to develop a solution basis of an approach where retrieved lessons coexist to foster the evolution of data-intensive systems.

Although research exist in the scope of process modeling centered on objects [12] and on process evolution [23], since existing approaches were developed to face small and specific sets of concerns [12], their practical applicability coverage and impact is limited [12]. This seems unaccountable in the contemporary era where, for instance, AI systems, several enterprises and many of their subsystems are truly data-intensive systems. This observation fosters the need to re-look to them from scratch in order to understand how potentialities can be combined.

This work systematizes the object-centered universe and it serves as a meta-guider for principles integration on tacit and data-intensive process modeling.

Structure. This work is divided into four logical sections. First, *Conceptual Foundations* provides a structured context for the universe of discourse. Second, *Related Work* identifies a set of requirements based on the limitations of traditional modeling approaches, and studies how emergent approaches answer them. Third, *Solution* presents a set of principles that restrict the solution space, derives a structure for their coexistence. Fourth, *Validation* assesses the approach executability and practical applicability. Finally, *Concluding Remarks* presents the resultant theorems and lines of thought for future research.

2 Conceptual Foundation

The Systemic Context. Inserted in the context of increasing data-intensive landscapes, this work adopts the process perspective to introduce new postulates on the modeling of system elements coordination that fosters evolution. Concepts are introduced below to structure the object-centered modeling.

Def.1 A **system** is a tuple $<R, C, E, G>$, where R is the structure, set of relationships among a composition of system elements C and external elements E, that satisfies a purpose G grounded on exchanges with its environment [8].

The system composition, C, is a set of *subsystems* or, from an elementary perspective, a set of participants P. *Participants* can either be *passive* ($P_P \subset P$) if subjected to transformation by a set of system actions, or *active* or agents ($P_A \subset P$) if performing actions aimed at changing passive participants [5].

Def.2 **System evolution** is the process of increasing the system responsiveness to its environment by continually optimizing the efficiency to pursue its purpose

under changing conditions. It depends on its ability to behave as an integrated, adaptable and agile system, i.e., to timely improve its structure ($R \rightarrow R'$) when internal, external or purposeful changes occur ($\{C, E, G\} \rightarrow \{C, E, G\}'$).

Functional decomposition of a system defines hierarchies of abstractions needed for the modeling of systems operation [32]. *Activities*, units of work, are its nodes. In open ($E \neq \emptyset$) and dynamic or multi-state systems, a system act or an *event* (implying a system action) produces a change to the system state. System activities coherently and consistently relate system acts.

Def.3 A system **process**, $<A_{\wp}, P_{\wp}, G_{\wp}, R_{\wp}>$, structures a set of system activities ($A_{\wp} \subseteq A$) performed in a constrained manner ($R_{\wp} \subseteq R$) based on the coordination of a set of system participants ($P_{\wp} \subseteq P$) to realize a set of system goals ($G_{\wp} \subseteq G$). A **process model** is a model for system processes, an abstraction $m(A_{\wp}, P_{\wp}, G_{\wp}, R_{\wp})$, which *describes* and *prescribes* the operation of an object system by its interacting systems based on its functional composition.

The Role of Data. Different data taxonomies for process modeling can be found in [33][24]. Weske [32] depicts the role of data within processes according to data visibility, interaction, transfer and support to routing logic. Aalst [29] distinguishes two main types of data: case and non-case. *Case data* is the data used by system applications to support activities. Non-case data can be divided into *support data*, if it affects the process routing logic, and *management data*, if it is produced by the process execution environment (e.g. audit trails).

This work simplifies this taxonomy into data generated and consumed by processes [33]. The reason behind this simplification – the increasingly blurry boundary between data exposed for process routing decisions and pure application data – lead us, finally, to the notion of data-intensive system.

In data-intensive systems, the data consumed and generated by some system elements is related with the production of other system elements [33].

Def.4 A **data-intensive system** is a system with a structure R that relates its elements C based on data mediation and transformation. Thus, its operation is constrained by the passive participants' state and relations or, more broadly, by the way the system productions to the environment realize the system goals.

For instance, automotive industrial systems rely on the entanglement of data components, the passive participants, to deliver a physical production. Claim-processing systems use claimer's information, regulatory and financial reviews and claim state to audit and deliver a decision.

Def.5 A process within a data-intensive landscape is referred as **data-intensive process**. Scientific research has been focusing on two main types of data-intensive processes: *i)* processes driven by the state of passive participants [6][30] and *ii)* collaborative and tacit processes that use passive participants as record objects to capture the system operation [25].

Research on data-intensive vertents has been adopting different terms as *documents*, *products* and *artifacts*, here all generalized and captured as objects, either simple or compound (encapsulating a set of related objects). Objects, either representing logical or physical elements, can be seen as building blocks that bridge the functional and informational perspectives [6].

Def.6 A system **object** represents any relevant state-based system element, either a simple participant or a composition of system participants. Its state is defined by the object content and it is modified during its life-cycle as the result of an invocation of a set of activities that act upon its data-attributes.

Informational decomposition of a system defines hierarchies of object models.

Modeling Orientation. Process modeling approaches can be divided according to their main focus of modeling: activity-flow, agent-coordination or data-needs focus lead, respectively, to *activity-centered*, *agent-centered* or *object-centered* languages. For instance, agent-centered modeling is the natural choice for processes with strict distribution of responsibilities owned by specialized agents, since activities precedences are implicitly derived from agent-interaction constraints [16]. This work also distinguishes *multi-paradigm* approaches where process models have not a clear orientation [5], and *hybrid* approaches where different approaches co-exist exposing different views for different users.

Since objects are generically used to model simple and compound structures of system participants, the dependencies among activities in object-centered approaches is derived from constraints on these participants' state and relations.

Def.7-1 An **object-centered process model** is a process model that uses the knowledge of the participants composition $P_{\wp}$ to derive the system structure under modeling $R_{\wp}$. Modeling of participants must be expressive, so changes at the system data models dynamically affect the system process models.

If we recover the system definition, we detect four main perspectives: functional, informational, instrumental and contextual, or, respectively, the activity-based, object-based, agent-based and goal-based views (see Fig.1 and Fig.2). All these models are integrated by a process model, the governance model, which constrains their interaction to capture and prescribe the operation of a target system.

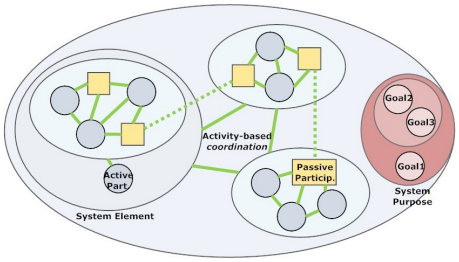

Fig. 1. Abstract data-intensive system

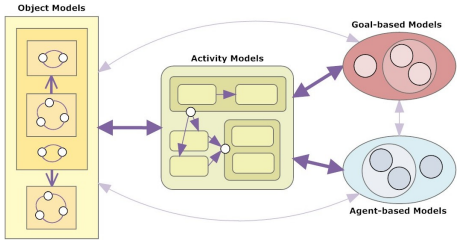

Fig. 2. Abstract process model

3 Related Work

In this section the research problem is reduced to *five* requirements and emergent approaches are evaluated according to their ability to answer them.

Research Problem. The pushing of passive participants to the processes background as a result of an historical hidden of data behind service and application layers is explained in [32][33]. Seven reasons are pointed in [22]. Nevertheless, our work aims to develop an environment for the integrated management of system data and system activities since their independence in data-intensive landscapes blocks benefits obtained from their coupled evolution.

IBM's *Global Financing* case, a system specialized in financing assets is partially depicted in Fig.3. The process modeling layer was based on a typical activity-centered approach. A synthesized description of the traditional approaches limitations when evolving data-intensive process models is presented in Table 1.

This study led this analysis to a point where the initial problem was breakdown into five pieces. Table 2 structures the five requirements triggered by the limitations of traditional approaches when modeling data-intensive processes.

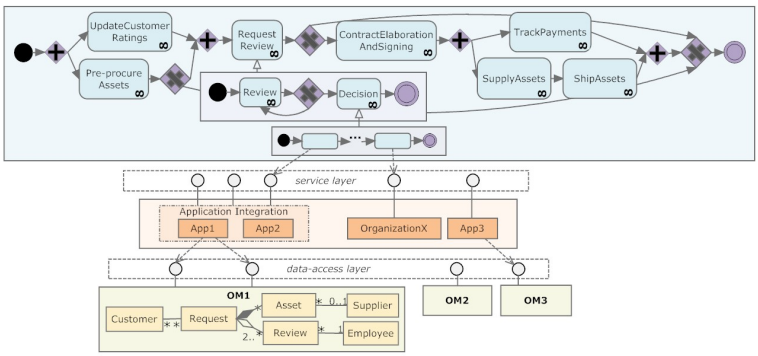

Fig. 3. Traditional Process Modeling Landscape

Problems	*GF* Challenges (Fig.3 upgrades)	Area of Concern
Process data redundantly created with application data poses consistency problems. Isolated execution of activities causes a loss of process contextual view. Data-access specification must be explicit for every activity and is not maintained among them, leading to non-usable models. There is no support for an integrated access to old or non-related process data;	Reviews or contract negotiations become: affected by the state of assets procurement (if running in parallel) or other request's reviews, their progress depends on customer's past and similar requests' information, and their contextual data continues accessible without the need to specify all attributes as input parameters for each activity;	*Data Access*
Activities have to be related in a net of ordering dependencies, turning difficult a spontaneously repetition of process instances, their stoppage and caught up at a later point in time and a dynamic reaction on data conditions;	Contract negotiation becomes reactively available on a condition satisfaction over reviews' data, independently from executing reviews' progress. Assets and reviews are dynamically instantiated based, respectively, on request and customer's data;	*Data-state Reaction*
Process instances (from the same or different process types) are executed in isolation to each other, hampering the support for expressive communication patterns among processes;	Decoupling of process segments in a modular way (e.g. Reviews and Assets). Aggregation of related segments: collective reviewal, collective asset supplying and shipping;	*Data-based Synchron.*
Service layers turn impossible the definition of criteria for the processes granularity since client application activities may reside at different granular levels;	Activities for accessing and changing customer or request attributes (e.g. customer address) must be available, and their composition must follow concrete criteria;	*Data-based Granularity*
Since data-centered and activity-centered models are separated by an application layer, modeling of data objects is roughly done at the process modeling level.	A request becomes handled by multiple customers (the process modeling environment detects the change of an association multiplicity on an underlying data model).	*Data Modeling*

Table 1. Problems of Traditional Modeling Approaches in Data-Intensive Landscapes

Introductory remarks for the understanding of data-awareness requirements can be found in [12][13][19]. Table 2 reviews and structures them under a taxonomy oriented to incremental steps for process models' evolution.

Def.8 An **object-centered system** is, thus, a data-intensive system with a structure R that satisfies the introduced requirements, i.e., a system where its passive participants are visible to every system agent, dynamically affect activities

Data Access	Process and application data must be coherently and consistently integrated, meaning that system's process and object models must be bridged and evolve in a coupled way according to a well-defined set of relations. Process models must avoid data-context tunneling (causing the loss of a broader view on the process) when executing isolated or groups of activities, and additionally must expressively hold data-access contexts from single to multiple activities [30]. Data-scope specification must additionally be usable, seizing benefits of using data models expressivity (e.g. accessing compound or sets of data-objects). Authorized users must access data at any time regardless of the process status [30][13];
Data-state Reaction	Processes must dynamically react on object state constraints, i.e., it is optional to define of network of activities' precedences. Since activities are related to objects, they must dynamically adapt their behavior (e.g. availability) based on objects' state (horizontal dynamic granularity) [12] and provide a natural method to deduce omissive path localization, minimizing sequentiality and, thus, fostering process flexibility;
Data-based Synchro.	Processes must use object models constructors to express advanced patterns of synchronization, including the definition of: *i)* multiple related instances aggregation or vertical aggregation to reduce execution effort (e.g. grouping related requests) [3], *ii)* asynchronous points of coordination to minimize processes coupling (e.g. synchronize the progress of a set of instances responsible for the assets procurement with their related request) [12], and *iii)* expressive transition's rule-sets;
Data-based Granularity	Atomicity and composition of activities must be based on the underlying process data [12] to, respectively, safeguard the availability of fine-grained activities and of a criterion to compound processes using multiple levels of modeling abstractions;
Data Modeling	There must be possible to model and adapt object models at the process modeling level [12]. Evolution of processes is, thus, fostered by the previous requirements, which assure that execution constraints are dynamically derived from the dependencies of object models in a usable manner, with this one, which enables modeling flexibility.

Table 2. Data-related Requirements for Object-centered Approaches

progress and composition, and are adequately accessed and expressively captured at the process modeling level.

State-of-the-Art Analysis. Six mature object-centered approaches were evaluated according to their ability to answer to the introduced *five* requirements. Their selection was based on the practical maturity, data-orientation and novelty of aspects. Results were collapsed in Table 3 and their analysis done in Table 4.

Approach	1.Data Access	2.Data-state Reaction	3.Data-based Synchronization	4.Data-based Granularity	5.Data Modeling
Document-based Modeling	+/–	+	+/–	–	+/–
Artifact-centric Modeling	–	+	+/–	+/–	+/–
Product-based Modeling	–	–	–	+/–	+/–
Data-driven Coordination	–	–	+	+/–	+/–
Case Handling	+	+	–	–	+/–
Proclets	–	–	+	–	–
Object-aware Modeling	+	+	+	+/–	+/–

Table 3. Approaches Evaluation [subtitles: + answers; +/– partially answers; – not answers]

Data-based multiple instantiation [11], batch-orientation [3], objects and activities connection through procedural links [9], data-based clustering for objects definition [15] or the attachment of operational semantics within objects [6] are some other interesting directions on this field of knowledge.

Discussion reveals that each approach answers differentially to the introduced requirements. Although object-aware direction [14] can be considered the most promising, it does not yet provided a concrete solution. No other approach satisfies more than two requirements, which may justify their limited case applicability coverage and impact. However, lessons can be used to retrieve principles to derive a more mature modeling approach.

Document-based Modeling [20][2]	It fosters simplicity by modeling constraints recurring to data-dependencies (2), guides enactement (3), supports authorized data-access using knowledge-bases (1) and captures ad-hoc forms of collaboration. However, it limits data modeling to plain structures (4,5), does not support advanced relations among documents (3), and instances are defined statically (3);
Artifact-centric Modeling [10][4][6]	It is oriented to business needs and execution constraints are automatically driven from artifacts modeling (2,5). However, activity data-access is limited to the related artifact (1), life-cycles synchronization only supports few patterns (3) and, finally, composition of artifacts is not possible (4);
Product-based Modeling [21][31]	It is good for process models that periodically require a clean-sheet, uses quality attributes for dynamic path choice (3). However, data-access is restricted to operations' input components, there must exist an explicit precedence network of operations (2), and applicability relies on the ability to specify productions using composition relationships that can be assembled into a single product (3,4,5);
Data-driven Coordination [18][17]	It is indicated for large and numerous concurrently executing processes. It adds advanced communication patterns, as the definition of synchronization points among instances belonging to the same or different process type (3), based on objects relationship types, and allows for complex structures specification that can guide functional decomposition (5). However it disregards simplicity, data content that leads to data reaction and access problems (1,2), and atomicity of activities (4);
Case Handling [30][27]	It is unique in providing a global view of the process to its users, data-access is expressive and users can surpass the activities by accessing data for which they have access levels (1). It allows for horizontal aggregation (2) and, since it is fully state-based, it is easy to conceptualize. However, processes hierarchies and data object-oriented patterns for synchronization purposes are poorly exploited (3,4,5);
Proclets [26][28]	It promotes a shift from control to communication emphasis, where processes interact according to an agreed level of reliability, security, closure and formality (3). It supports multiple messages-exchange patterns and batch-oriented tasks, thus, enabling vertical dynamic aggregation of proclets instances (3). Although proclets are decoupled process fragments, composition is not supported (4) and each fragment can still be considered an activity-centered model, thus, suffering from same data-access and data-state reaction limitations (1,2,5);
Object-aware Modeling [12][13][14]	It integrates case handling's data-access (1), enriching authorization with state-based and instance-based levels, and form principles (1). It supports artifact-centric patterns of coordination. It introduces collective enactment of batch activities (3), on-the-fly form changes (2), multiple instantiation using cardinalities (3) and re-execution of submitted activities (1). However, it does not support vertical aggregation for instances with different higher-level object instances (3), sound criteria and object-oriented modeling patterns as inheritance and composition (4,5).

Table 4. Brief Review of Emergent Approaches' [subtitles: (x) referes to requirement(s) x]

4 Solution Basis

This section uses the understanding of how emergent approaches answer data requirements to retrieve principles and to define a structure for their coexistence.

A process is here captured as a set of state-based and synchronized entities – objects, activities, goals and time. Exemplifying, a transition between states of an entity A can trigger an event for another transition occurrence in an entity B if B has some sort of dependency with A.

Two *axioms* must be introduced. *First*, process state is a function of time and of its activity, object, and goal models. *Second*, synchronization among entities is defined through event-driven state transitions recurring to rule-set models.

Object models and *activity models* in the modeling landscape of data-intensive systems are integrated by a *process model* (the governance model) that establishes relationships and constrains their interaction through *rule-set models*.

Note that the notion of system applications that frequently intertwined data and process models are in data-intensive landscapes pushed back and here seen as workflow systems' additions to implement non-trivial system rules.

Def.7-2 An **object-centered process model**, $<OM, AM, GM, RM>$, is a model derived from a set of state-based object (or participant) models OM, activity models AM and goal models GM. These governed models are synchronized through rule-set models RM that constraint the availability to invoke activities based on

state restrictions or concessions. It can generically be defined as a pair $<N, E>$ where $N = N_{OM} \cup N_{AM} \cup N_{GM} \cup N_{RM}$, and $E \subseteq N \times N$.

Principles. The solution space constraints that support the satisfaction of the introduced requirements are synthesized in Table 5. Note that since, execution constraints for data-intensive processes mainly derive from objects' state and relations, object models adaptation becomes the great source to evolve processes.

Data Access	***First***, all system data is captured (by modeling either tacit or non-tacit applications and using their trace to feed their objects), standardized (by using a widely-accepted data modeling notation) and accessible (by not hidding data behind applications). ***Second***, activity models prescribe objects' data access by specifying labeled associations, both imperative and declarative, at any granular level of activities and of objects. ***Third***, related running activities are presented together using forms. Forms fields change dynamically since aggregate activity-related attributes can be alternately submitted and may turn new attributes available (soundness must solve form's deadlocks). ***Fourth***, there is a default criteria for the automatic definition of the activities' data scope based on the object models. An activity has not only access to its related object or attribute but may access related/internal/super objects and attributes if these associations declare *public* visibility. ***Fifth***, authorization is separated from distribution using an object-based structure to manage agents' privilege access levels. Vertical (instance-based) and state-based authorization also define access levels.
Data-state Reaction	***First***, the object-centered models' interplay assures that activities react on objects state in a traceable manner. Even activity's completion, failure or cancellation behaviour is, by omission (although editable), retrieved from related object specification. ***Second***, it is possible to specify dependencies of different types (e.g. start-to-start, finish-to-start, start-to-finish, finish-to-finish) among data-attributes or any granular level of objects, which generates expressive dependencies on the functional level and fosters an usable parallelization of attributed-based activities and the ability of process models to evolve through object models adaptation.
Data-based Synchro.	***First***, object state transitions may depend on conditions over the state of the related object instances and also affect their markings. Communication among objects, always mediated by a third object, is derived from object models to process models and it enables the definition of asynchronous points of synchronization between processes. ***Second***, the skeleton for the rule-set models is automatically generated on the basis of their input and output states and of expressive constructors. ***Third***, rule-set models placed in objects' state transitions can comprise advanced formulas based on aggregation constructors, data-scope settings, time conditions and executable code additions.
Data-based Granularity	***First***, each compound activity is a composition of fine-grained activities and for each data-field exists an activity that triggers system acts to access and modify its content. ***Second***, object model's relations of encapsulation constitute a criterion to derive the processes' composition, enabling zoom operations through different operational levels.
Data Modeling	Definition and continuously adaptation of data-intensive system is possible through dynamical creation, edition and removal of objects at the process modeling level. Soundness and migration coherency of the affected object-centered instances is assured.

Table 5. Principles for the FIVE Requirements

The *Process* of Modeling Data-intensive Processes. The modeling of the target process models begins with the objects' specification by enriching the system data models with synchronized life-cycles and data dependencies. Rule-set models are used to specify advanced behaviour. Activity models, partially derived from object models, contain editable functional aspects and data-scope. Process models are dynamically derived from the previously defined models.

Object-centered modeling steps are synthesized by the following algorithm:

1. (Manual) Definition of a data model for the target system using UML;
2. (Automatic) Generation of the object models skeleton;
3. (Manual) Edition of object models to capture the system semantics;
4. (Automatic) Test of object models' soundness. Generation of activity models;
5. (Manual) Activity models' constraints and data-scope edition;

6. (Automatic) Test of activity models' soundness. Derivation of the object-centered process models net. Generation of default rule-set models;
foreach *true* **do**
    7. (Manual) Adaptation of one of the system object-centered models;
    8. (Automatic) Test of models' soundness. Change of the affected models;
end

Fig.4 illustrates an enriched data model. Data-dependencies, object inheritance and composition, and objects life-cycles synchronization are presented. Formal rules are generated on-the-fly, but can be edited to include advanced behavioral patterns. Fig.5 presents the dynamically derived process model from the previous models.

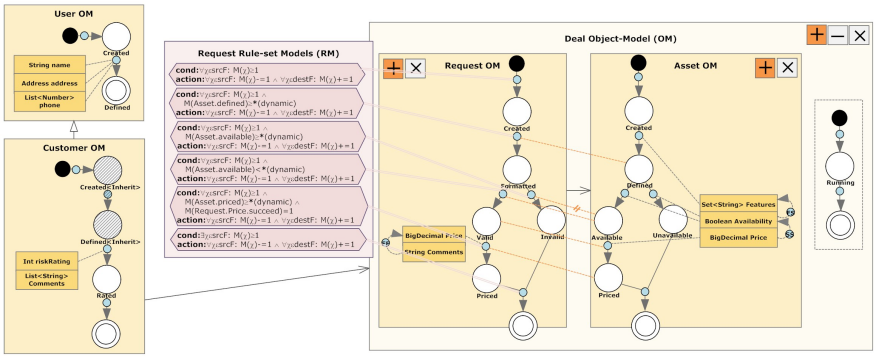

Fig. 4. Object Models and default Rule-set Models

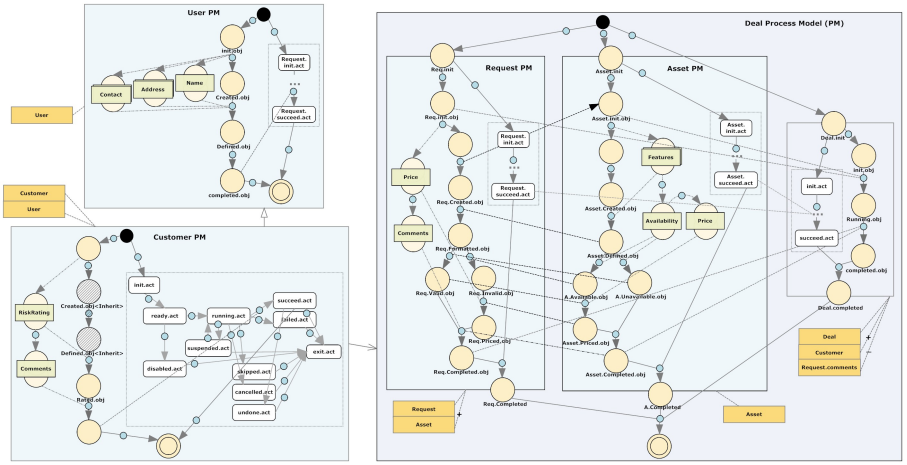

Fig. 5. Derived object-centered Process Models

Beside the illustrated data-reaction, loosely-coupled coordination, default behavior and composition support, additional features were defined for the integration of data, for the dynamic derivation and change of activity-centered aspects (as activities suspension, skipping, cancelling, re-execution and their non-local effects), for the capture of advanced behavior (as vertical aggregation constructs) and for the generation and edition of data-contexts.

Event-orientation. Every system entity dynamically publishes and subscribes a set of events, corresponding either to values-changes or state-changes. When an entity instance's marking change, an event is triggered, and conditions referring that marking places are notified to evaluate their conditions. Event-orientation increases the system responsiveness, agility and flexibility (agents can be added, removed or upgraded as they do not hide data and interact with process modeling landscape via events either through adaptors or by manual filling of forms).

An event-driven solution applied to the object-centered modeling enables a natural support of advanced behavior based on complex event processing techniques, the aggregation of low-level events into relevant compositions fostering a system modeling at an arbitrary level of abstraction, and, finally, an increased auditability as the source of data-change events becomes traceable and their occurrence verified. Finally, encapsulation limits the scope of entities synchronized through events to increase the usability and locality of changes when modeling.

Soundness Criteria. Object-centered process models assure *termination* [32] and *no dead transitions* [32] properties of soundness. To assure process models termination data-attributes' valid domains and rule-set models' conditions and actions are analyzed. Additionally, multiple instantion patterns force the termination of running instances that no longer affect their higher-level instance. To assure the absence of dead transitions, complete nets are derived from each compound object model. The modeling approach transgresses the *proper termination* criterion [32].

Contribution Reviewed. This work is centered on defining a solution for the coexistence of principles. It is not focused on the specificities of each principle but on providing useful constructors to support their definition and adaptability. It is also unique in presenting a direction for an approach that: *i)* fully derives and adapts process models from enriched data models, *ii)* defines expressive behavior based on formal rules, *iii)* comprises advanced traceability and soundness criteria, and *iv)* exploits object-oriented patterns for an expressive derivation of processes.

5 Validation

Proof-of-the-Concept. Algorithms were developed for the generation and completion of object and activity models, and to derive simple and compound process models. Object-centered models were formalized and their soundness criteria defined. To assure the execution, advanced verification, instances migration and the interoperability of the target models, a mapping to *YAWL* was defined. Its formal structure, composition and multiple-instance support, non-local behavior and notational convenience, turns *YAWL* the natural candidate. For a fully mapping, both *proclets* addition and the resource-view construct of *newYAWL* additions (based on multi-colored places) were used.

As the set of object-centered models can be considered a high-level domain-specific language to lower-level constructs provided by *YAWL*, syntax specification and model-to-model transformations were defined using the *ASF+SDF* engine.

Practical Applicability. The object-centered modeling was applied to three distinct domains – financing, manufacturing and healthcare. Their modeling showed that object-centered constructs successfully coverage general and domain-specific requirements when compared with its alternatives. Additionally, implications resulting from the adoption of this new modeling paradigm were retrieved and their role on fostering the evolution of data-intensive processes was studied.

6 Conclusions

Activity-centered approaches are limited in modeling data-intensive processes. Such limitations can be translated into a set of requirements that foster the ability to process models evolve in data-intensive landscapes. Emergent object-centered approaches do not successfully satisfy all requirements, which may be correlated with their limited practical applicability, but provide key principles.

Principles to deal with different requirements are not mutually exclusive in an event-driven solution basis centered on the state-based synchronization of entities. A more responsive and modular solution pushed by this solution can be achieved by decoupling the traditional process notion into loosely-coupled interacting objects, as it discourages a thinking on constrained paths of activities and fosters a natural parallelization of the derived activities.

The expressive enrichment of data models and their coupled alignment with activity models (dynamically deriving and encapsulating execution aspects), fosters an evolution of process models centered on the adaptation of object models.

Possible lines of though for future research comprise the development of an usable graphical layer on top of the existing textual models, the continuously systematization and enrichment of rule-set models expressivity, the conception of an hybrid approach for heterogeneous systems where object-centered modeling plays its part, and the attachment of semantics to objects so their composition and constraints can be dynamically derived to satisfy sets of system goals.

References

1. S. Abiteboul, L. Segoufin, and V. Vianu. Modeling and verifying active xml artifacts. *IEEE Data Eng. Bull.*, 32(3):10–15, 2009.
2. A. Abrahams and D. Eyers. Using annotated policy documents as a user interface for process management. In *ICAS '07*, page 64, DC, USA, 2007. IEEE CS.
3. P. Barthelmess and J. Wainer. Workflow systems: a few definitions and a few suggestions. In *COCS '95*, pages 138–147, New York, NY, USA, 1995. ACM.
4. K. Bhattacharya, C. Gerede, R. Hull, R. Liu, and J. Su. Towards formal analysis of artifact-centric business process models. In Alonso, Dadam, and Rosemann, editors, *BPM*, volume 4714 of *Lecture Notes in CS*, pages 288–304. Springer, 2007.
5. Ilia Bider. Choosing approach to business process modeling - practical perspective. In *Journal of Conceptual Modeling*, January 2005.
6. D. Cohn and R. Hull. Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.*, 32(3):3–9, 2009.
7. F. Corradini, A. Polzonetti, R. Pruno, and L. Forastieri. Document exchange methodology for collaborative work in e-government. In *DEXA '06*, pages 283–287, Washington, DC, USA, 2006. IEEE Computer Society.
8. Jan L. G. Dietz. *Architecture - Building strategy into design.* Academic Service, The Hague, Netherlands, 2008.
9. D. Dori. Object-process methodology as a bp modelling tool. In *ECIS '00*, 2000.
10. C. Fritz, R. Hull, and J. Su. Automatic construction of simple artifact-based business processes. In *ICDT '09*, pages 225–238, New York, NY, USA, 2009. ACM.
11. A. Guabtni and F. Charoy. Multiple instantiation in a dynamic workflow environment. *Advanced Information Systems Engineering*, pages 175–188, 2004.
12. V. Künzle and M. Reichert. Towards object-aware process management systems: Issues, challenges, benefits. In *Enterprise, BP and IS Modeling*, volume 29 of *Lecture Notes in BIP*, pages 197–210. Springer Berlin Heidelberg, 2008.

13. V. Künzle and M. Reichert. Integrating users in object-aware process management systems: Issues and challenges. In Rinderle-Ma, Sadiq, and Leymann, editors, *BPM Workshops*, volume 43 of *Lecture Notes in BIP*, pages 29–41. Springer, 2009.
14. Vera Künzle, Barbara Weber, and Manfred Reichert. Object-aware business processes: Properties, requirements, existing approaches. Technical report, University of Ulm, Germany, 2010.
15. R. Liu, K. Bhattacharya, and F. Y. Wu. Modeling business contexture and behavior using business artifacts. In J. Krogstie, A. Opdahl, and G. Sindre, editors, *CAiSE*, volume 4495 of *Lecture Notes in CS*, pages 324–339. Springer, 2007.
16. M. Merz, B. Liberman, and W. Lamersdorf. Using mobile agents to support interorganizational workflow-management. *Int. Journal on Applied AI*, 11(6), 1997.
17. D. Müller, M. Reichert, and J. Herbst. Data-driven modeling and coordination of large process structures. In *OTM Conferences (1)*, pages 131–149, 2007.
18. D. Müller, M. Reichert, and J. Herbst. A new paradigm for the enactment and dynamic adaptation of data-driven process structures. In *CAiSE '08*, pages 48–63, Berlin, Heidelberg, 2008. Springer-Verlag.
19. P. Nandi, D. König, S. Moser, R. Hull, V. Klicnik, S. Claussen, M. Kloppmann, and J. Vergo. Introducing business entities and the business entity definition language. Technical report, IBM, 2010.
20. M. Rahaman, Y. Roudier, and A. Schaad. Document-based dynamic workflows: Towards flexible and stateful services. *IEEE Congress on Services*, 0:87–94, 2009.
21. H. A. Reijers, S. Limam, and W. M. P. van der Aalst. Product-based workflow design. *J. Manage. Inf. Syst.*, 20(1):229–262, 2003.
22. W. Sadiq, K. Schulz, M. E. Orlowska, and S. Sadiq. When workflows will not deliver: The case of contradicting work practice. In Witold Abramowicz, editor, *BIS '05*, pages 69–84. Wydawnictwo Akademii Ekonomicznej w Poznaniu, 2005.
23. H. Schonenberg, R. Mans, N. Russell, N. Mulyar, and W. M. P. van der Aalst. Process flexibility: A survey of contemporary approaches. In J. Dietz, A. Albani, and J. Barjis, editors, *CIAO! EOMAS*, volume 10 of *Lecture Notes in Business Information Processing*, pages 16–30. Springer, 2008.
24. Hong Linh Truong and Schahram Dustdar. Integrating data for business process management. *IEEE Data Eng. Bull.*, 32(3):48–53, 2009.
25. G. van Bussel, F. Ector, G. van der Pijl, and P. Ribbers. Building the record keeping system: Process improvement triggered by management of archival documents. In *HICSS '01*, volume 8, page 8060, Washington, DC, USA, 2001. IEEE CSociety.
26. W. M. P. van der Aalst, P. Barthelmess, C. Ellis, and J. Wainer. Workflow modeling using proclets. In *CooplS '02*, pages 198–209, London, 2000. Springer-Verlag.
27. W. M. P. van der Aalst and P. Berens. Beyond workflow management: product-driven case handling. In *GROUP '01*, pages 42–51, NY, USA, 2001. ACM.
28. W. M. P. van der Aalst, R. S. Mans, and N. C. Russell. Workflow support using proclets: Divide, interact, and conquer. *IEEE Data Eng. Bull.*, 32(3):16–22, 2009.
29. W. M. P. van der Aalst and K. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT Press, 2002.
30. W. M. P. van der Aalst, M. Weske, and D. Grünbauer. Case handling: A new paradigm for business process support. *Data and Knowledge Eng.*, 53:2005, 2005.
31. I. Vanderfeesten, H. Reijers, and W. van der Aalst. Product based workflow support: Dynamic workflow execution. In Bellahsene and Léonard, editors, *CAiSE*, volume 5074 of *Lecture Notes in CS*, pages 571–574. Springer, 2008.
32. Mathias Weske. *Business Process Management: Concepts, Languages, Architectures*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
33. Michael zur Muehlen. Volume versus variance: Implications of data-intensive workflows. *IEEE Data Eng. Bull.*, 32(3):42–47, 2009.