Tool Support for the EKD Enterprise Modeling Method: Towards Managing Inter-View Consistency

Afef Awadid[1] and Selmin Nurcan[1]

[1] University of Paris 1 Pantheon-Sorbonne, Paris, France
{afef.awadid@malix.univ-paris1.fr,nurcan@univ-paris1.fr}

Abstract. Central to Multi-View Modeling Methods (MVMMs) is the structuration of an enterprise and its underlying systems into views and levels in order to curtail their complexities. However, efficient application of MVMMs considerably depends on the availability of corresponding modeling tools- the so-called Multi-View Modeling (MVM) tools. The latter are crucial for machine processing of models (views) and hence for managing inter-view consistency. This is the reason why the focus of this paper is geared towards developing a MVM tool for the Enterprise Knowledge Development (EKD) method - a representative of MVMMs. This tool is intended not only to fill a research gap, but also to implement insights and findings derived from our previous work on inter-view consistency. Two different ways of managing consistency are indeed stressed: consistency by construction and consistency checking. Compared to the latter, the former advocates preventing inconsistencies rather than handling/correcting them. The tool has been implemented using ADOxx platform and realized within the Open Models Laboratory (OMiLAB).

Keywords: EKD Enterprise Modeling Method, Multi-View Modeling Tool, Inter-View Consistency, ADOxx.

1 Introduction

At the core of Multi-View Modeling Methods (MVMMs) is Multi-View Modeling (MVM). MVM is a widely accepted technique to reduce the complexity of a system under study (e.g., a business process, enterprise, cyber-physical system). The key idea behind this technique is to capture different aspects of the modelled system (e.g. its structure and its behavior) by different views (models) [1]. Each view (i) sheds light on certain aspects of the system, and (ii) is specified by a viewpoint which depicts the concepts considered by the view and the valid combinations (e.g. specified by a meta-model) [2]. The usefulness of MVM "is obvious and objectively proven" [3]. This opinion has been ascertained by several studies e.g. the one conducted by [4], where the authors found that applying the viewpoint concept makes "it easier to cope with overall size of the problem domain" and leads "to a deeper understanding of the problem domain" [4].

Nonetheless, efficient application of MVMMs considerably depends on the availability of corresponding modeling tools [5] commonly known as MVM tools. Such tools become valuable assets in machine processing of models in general and in managing inter-view consistency in particular. It is, therefore, no surprise that modeling tools development gains quite a lot of attraction in academia and industry. A convincing proof is that a plethora of modeling tools underpinning MVMMs have been developed within the Open Model Initiative Laboratory (OMiLAB)[1] - “a physical and virtual research environment that supports meta-modeling research projects and communities" [6]. Following this direction, the paper at hand reports on the development of a MVM tool for the first version of the Enterprise Knowledge Development (EKD) method [7]. EKD stands for a framework for modeling enterprise knowledge that, grounding on the terms "dimension” and "perspective", it allows mastering not only the complexity of the enterprise as a whole, but also that of business processes.

In its current version, the tool focuses on the process related aspects of EKD since this method advocates a MVM approach for business processes. An EKD business process model is thereby depicted in terms of three complementary views (sub-models), where each view captures one or more perspectives (aspects). Based on this, these views are not independent from each other. There is hence a need for managing consistency among them. The proposed tool is intended not only to fill a research gap, but also to implement insights and findings derived from our previous work on inter-view consistency [8][9]. Two different ways of managing consistency are indeed stressed: consistency by construction and consistency checking. In contrast to the latter, the former advocates preventing inconsistencies rather than handling/correcting them. The foremost question raised in this paper is accordingly: How to implement an EKD modeling tool that accounts for both ways of managing inter-view consistency? The tool has been realized within the OMiLAB and implemented using the ADOxx platform- "a meta modelling-based development and configuration environment to create domain-specific modelling tools" [10].

This paper is organized as follows: Section 2 provides the theoretical foundations for this work and briefly presents related works. Section 3 puts emphasis on the conceptualization of the EKD modeling method. In Section 4, an overview of the EKD modeling tool along with a case study demonstrating the application of this tool are given. Lastly, section 5 concludes the paper with an outlook on future research.

2 Motivation and Related Works

2.1 Motivation

Before presenting the motivation for this work, key terms like viewpoint, view, concern and consistency need to be defined. By viewpoint, we refer to the modeling language used to specify a view which is represented by a conceptual model. Each view allows capturing perspective(s). The latter is used to denote certain aspect(s) from which the system under study can be viewed (e.g., its behavioral and structural aspects). A perspective can then be a synonym to the term 'concern' defined by ISO/IEC/IEEE 42010

[1] http://omilab.org

as "any interest in the system" [11] and tied to the notion of 'separation of concerns'. A separation of concerns is of type "horizontal" if the considered concerns belong to the same level of abstraction/phase of development, or otherwise of type "vertical" [12]. By inter-view consistency, we mean the extent to which information contained in multiple views is not contradicting [13].
Inter-view consistency is an important aspect on which depends the quality of the created views and, hence their capability to be involved in the “design by reuse”. Inconsistent views are indeed not only hard for their users to grasp but also hamper the tasks of process analysis, redesign, reuse, and automation [14]. This may explain why our research work has been directed toward inter-view consistency.

Following the findings of a systematic literature review of inter-view consistency [8], we derived two different ways of managing inter-view consistency: consistency by construction and consistency checking. The idea behind the former is to manage inter-view consistency by preventing inconsistencies. By contrast, the aim of the latter is to manage inter-view consistency by handling/correcting inconsistencies once detected. These two ways of inter-view consistency are respectively in line with two principles of MVM originally defined in [15]: 1) system-oriented MVM, where any change in one view is automatically propagated to the other affected views so that all views are kept consistent. 2) In diagram-oriented MVM, the effects of a change in one view cannot automatically be seen in the other views. This thereby gives rise to temporary inconsistencies.

Two reasons constitute the motivation behind this work. First, the support of the EKD method by a tool to foster its efficient application. Second the implementation of the two diverging ways of managing consistency. Such implementation is needed to justify the preference of one way over the other.

2.2 An Overview of the EKD Modeling Method

The development of EKD method started in late nineties within a research project called ELECTRA (ELectrical Enterprise Knowledge for TRansforming Applications), in which several researchers have been involved including the co-author of this paper. EKD stands for a framework for modeling enterprise knowledge that, grounding on the terms "dimension” and "perspective", it allows mastering not only the complexity of the enterprise as a whole, but also that of business processes. As exhibited in Fig. 1, the EKD method structures the enterprise in three key dimensions: Goals, Business processes and Information systems. In its current version, the tool targets the dimension "business processes" which is organized into three perspectives (see Fig. 1): Actors/ Roles, Roles/Activities and Business Objects. Each perspective is captured by a view (i.e., sub-model). Consequently, according to the EKD framework, a business process model is divided into three views: actors/roles view, roles/activities view and business objects view. A horizontal separation of concerns is therefore applied by this method.

To facilitate the implementation of the EKD modeling method in a modeling tool, we perceived this method as a specialized instantiation of the generic modelling method framework as introduced in [16]. This framework considers a modeling method as a composite of three building blocks: (i) a modelling language that defines: the syntactic concepts and their valid relationships; the semantics, explicit properties in concept schemata; and the notation, the graphical representation of the concepts, (ii) a modelling procedure that focuses on how the modelling language is used to achieve results, and (iii) mechanisms & algorithms that process the knowledge captured in diagrammatic models. Design decisions adhering to these buildings blocks have to be taken during the implementation of a modelling method in a modelling tool. Such "decisions directly correspond to the intended use of the modelling method" [17]. Based on this and given that our aim behind using EKD modeling method is the MVM of business processes with a focus on inter-view consistency, our design decisions include the definition of the modeling language for EKD business process models, and the specification of mechanisms and algorithms to ensure machine processing of models in general and inter-view consistency management in particular.

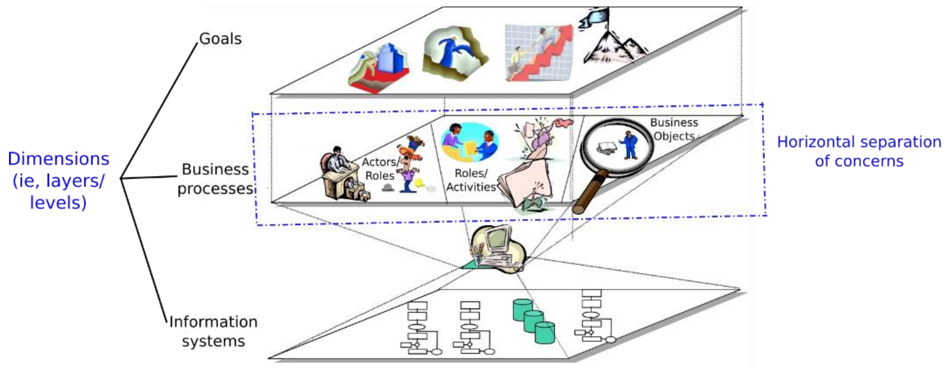


Fig. 1. The EKD enterprise framework.

2.3 Related Works

A closer look at the literature on MVM tools supporting modeling methods discloses an increasing interest in these tools. Most of them have been developed within the OMiLAB. Bearing in mind the two ways of managing inter-view consistency (the motivation subsection), these tools can be broadly classified into two categories. The first category encompasses tools opting for consistency by construction, whereas the second category of tools opts rather for consistency checking.

The Semantic Object Model (SOM) tool [18] is one example of tools fitting in the first category. The SOM tool allows modeling of SOM business process models and resource models. This tool advocates the consistency preservation (ie how to keep all the views consistent even when one or more of them undergo changes). It hence implements techniques that prevent inconsistencies from occurring. Moreover, MEMO4ADO [19] is an ADOxx based tool that is devoted to the Multi-Perspective Enterprise Modelling (MEMO) method. MEMO4ADO "implements a subset of

MEMO languages specifically tailored for educational purposes". Similarly, to the developers of the SOM tool, developers of MEMO4ADO are proponents of the consistency by construction. Indeed, to achieve consistency among views using this tool, a view utilizes concepts that reference elements defined in other views. This fosters an automatic update of the views and hence their consistency preservation. In contrast to the two aforementioned MVM tools, the SemCheck tool [20] belongs to the second category. It thereby promotes consistency checking by enabling an analysis of interrelated views in order to detect errors/ inconsistencies and to extract information from those views. The systematic analysis of queries is the main technique implemented by the tool that enables detecting inconsistencies. Consistency management in multi-view enterprise models was also discussed in [21] where different techniques have been inventoried on a technical level.

Compared to existing MVM tools, the aim pursued in this paper is to develop a MVM tool for the EKD method that implements these two diverging ways of managing consistency. Such implementation is needed for evaluating them with respect to quality attributes like time and consistency, and therefore for providing a reasonable reason to prefer one over the other.

3 Conceptualization of the EKD Modeling Method

This section is devoted to defining the language for EKD business process models. The language is specified by a metamodel. As shown in Fig. 2, the EKD business process metamodel highlights concepts and their relationships that are used for modeling business processes. It is specified using UML class diagrams. Concepts are linked by an association relationship along with cardinalities to denote how many instances of the one concept can be connected to one instance of the other concept. The metamodel encompasses three interrelated viewpoints: actor/role, role/activity, and business objects. An EKD business process model is then represented in the form of three complementary views (sub-models). Each view is specified by its corresponding viewpoint (a metamodel part).

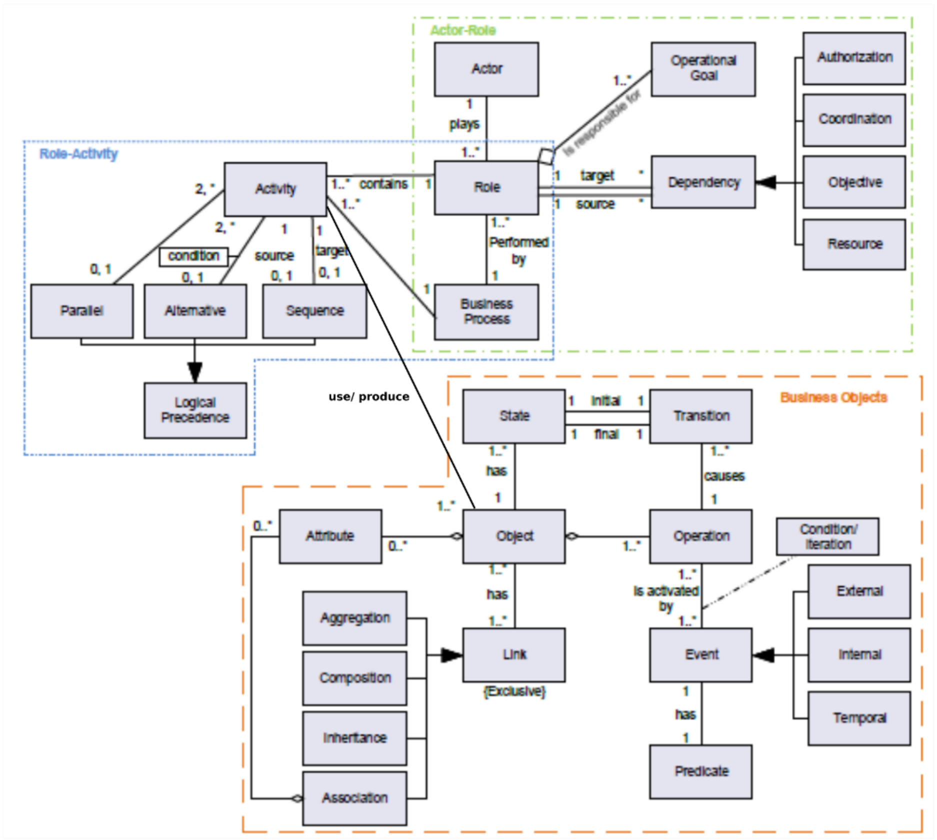


Fig. 2. The EKD business process metamodel [22].

The actor/role viewpoint is about representing the organizational perspective of a business process. The latter focuses on where and by whom activities are performed. The concepts "Actor" and "Role" are consequently at the core of the actor/role viewpoint. Here, it is worth noting the distinction between these two concepts. An “Actor” refers to a physical enterprise entity (e.g., customer, business unit manager, financial director), whereas the “Role” defines the responsibility assigned to an actor within a business process (e.g., loan request validator). An actor plays one or more roles. Each "Role" is responsible of one or more operational goals. Moreover, there is a "Dependency" between roles. Such dependency can be of type "Authorization", "Coordination", "Objective" or "Resource". Fig. 3 exhibits the representation of an actor/role view.

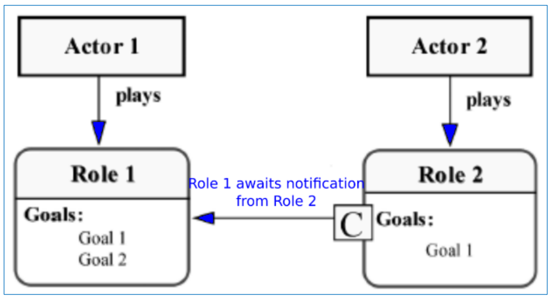


Fig. 3. Representation of an actor/ role view.

By contrast to the actor/role viewpoint, the role/activity viewpoint puts emphasis on the functional and behavioral perspectives of a business process. Functional perspective represents what activities should be performed. While, behavioral perspective represents when and how activities are performed. Central to role/ activity viewpoint are "Activity" and the control flow: "Sequence", "Parallel" and "Alternative". A representation of parallel and alternative executions is given in Fig. 4.

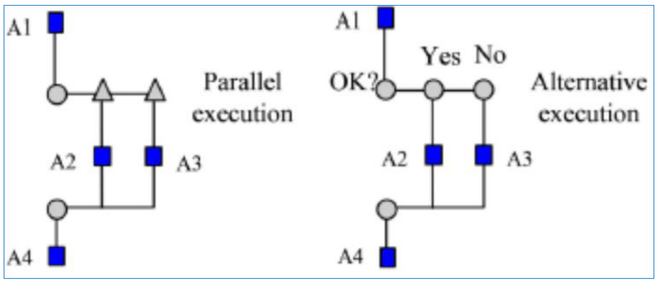


Fig. 4. Representation of parallel and alternative executions.

Last but not least, the business objects viewpoint that is articulated around the concept "Object". Each business activity of a given business process requires business objects as generations of input and output data. A business activity therefore use/produce business objects. Business objects refer to physical and informational elements of an enterprise that are required for its functioning. This viewpoint highlights then the informational perspective depicting the business objects handled by a process, their structures, and the links between them. As outlined in the metamodel (Fig. 2), an "Object" has one or more "State" and is composed of one or more "Operation". An "Operation" causes one or more "Transition". Accordingly, a business objects viewpoint allows describing not only a static aspect of business objects (i.e. business objects and their links), but also a dynamic one enabling representing business objects lifecycles according to activities carried out within business processes. Note that the concept "Business process" appearing in the metamodel denotes a compound building block within a business process model. It is therefore not a basic concept of the language.

4 The EKD Modeling Tool

This section discusses the conceptualization of the theoretical foundation introduced above towards a modeling tool for the EKD method. The tool is based on the ADOxx metamodeling platform. In this section, an overview of the tool functionalities as well as a case study demonstrating the use of the tool are given.

4.1 EKD Multi-View Modeling Functionality

The EKD tool enables modeling of the EKD business process models (the second dimension of the EKD enterprise framework on Fig. 1). Moreover, the tool allows to carry out the two principles of MVM (cf. section 2.1). Views can therefore be created following either the system-oriented or the diagram-oriented MVM. The modeler is the one who decide which principle to follow. Table 1 brings an insight on how the EKD viewpoints have been implemented in ADOxx.

Table 1. MVM principles, EKD viewpoints and their ADOxx counterparts.

MVM principle (ADOxx model type group)	EKD viewpoint (ADOxx model type)
System-oriented MVM	Actor/ Role
	Role/ Activity
	Business Objects
Diagram-oriented MVM	Actor/ Role
	Role/ Activity
	Business Objects

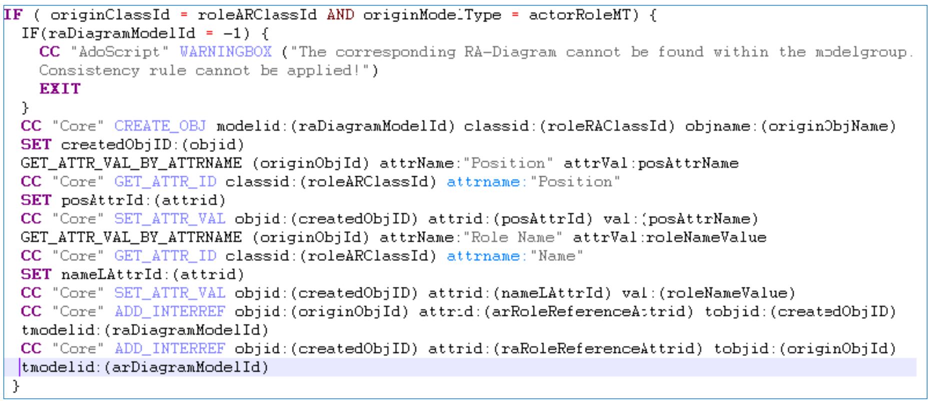
4.2 EKD Consistency Management Functionality

The implementation of this functionality requires as prerequisite the definition of the corresponding consistency rules. The latter have been defined in a previous work [10]. The tool implements the two ways of managing inter-view consistency: consistency by construction and consistency checking. As indicated in section 2.1, these two ways dovetail respectively with the two principles of MVM defined in [15]: system-oriented MVM and diagram-oriented. Accordingly, each way of managing consistency refers to a tool functionality that is aligned with the corresponding principle (see Table 2) and that is implemented using AdoScript language.

Table 2. MVM principle realized as ADOxx model type group and corresponding tool functionality.

MVM principle (ADOxx model type group)	Tool functionality
System-oriented MVM	Consistency by construction
Diagram-oriented MVM	Consistency checking

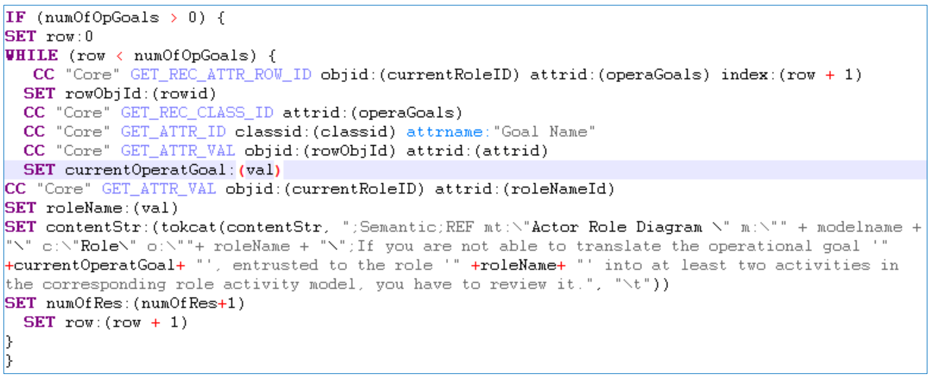
In the system-oriented MVM, the system (i.e. the tool) plays a key role in managing inter-view consistency. It somehow constructs consistency and hence relieves the modeler from the burden of handling/correcting inconsistencies. By contrast, in the diagram-oriented MVM, it is the modeler who plays a key role in managing inter-view consistency as it is up to him to resolve inconsistencies once detected by the system. The consistency by construction functionality is supported by the event handlers in ADOxx along with the attribute type "interRef" which together promote an automatic consistency preservation. Event Handlers are AdoScripts that are executed when certain events occur (e.g. create a node, create a connector, edit an attribute value,etc). "InterRef" is an attribute type that stands for a reference on a model or an instance. One example of EKD consistency rules that have been implemented in this way is rule 1 (roles in the actor role view and those in the role activity view have to be the same). The corresponding AdoScript implementing this rule is shown in Fig. 5.



```
IF ( originClassId = roleARClassId AND originModelType = actorRoleMT) {
  IF(raDiagramModelId = -1) {
    CC "AdoScript" WARNINGBOX ("The corresponding RA-Diagram cannot be found within the modelgroup.
    Consistency rule cannot be applied!")
    EXIT
  }
  CC "Core" CREATE_OBJ modelid:(raDiagramModelId) classid:(roleRAClassId) objname:(originObjName)
  SET createdObjID:(objid)
  GET_ATTR_VAL_BY_ATTRNAME (originObjId) attrName:"Position" attrVal:posAttrName
  CC "Core" GET_ATTR_ID classid:(roleARClassId) attrname:"Position"
  SET posAttrId:(attrid)
  CC "Core" SET_ATTR_VAL objid:(createdObjID) attrid:(posAttrId) val:(posAttrName)
  GET_ATTR_VAL_BY_ATTRNAME (originObjId) attrName:"Role Name" attrVal:roleNameValue
  CC "Core" GET_ATTR_ID classid:(roleARClassId) attrname:"Name"
  SET nameLAttrId:(attrid)
  CC "Core" SET_ATTR_VAL objid:(createdObjID) attrid:(nameLAttrId) val:(roleNameValue)
  CC "Core" ADD_INTERREF objid:(originObjId) attrid:(arRoleReferenceAttrid) tobjid:(createdObjID)
  tmodelid:(raDiagramModelId)
  CC "Core" ADD_INTERREF objid:(createdObjID) attrid:(raRoleReferenceAttrid) tobjid:(originObjId)
  tmodelid:(arDiagramModelId)
}
```

Fig. 5. Consistency by construction: implementation of rule 1 using an event handler (after_create_modeling_node).

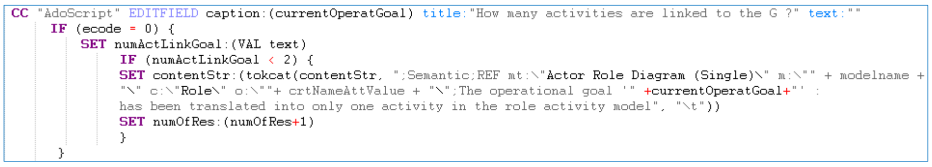
The implementation of other rules has taken the form of guidelines for the modeler. Example of these rules is rule 2 (an operational goal has not to be atomic). An excerpt AdoScript implementing this rule is presented in Fig. 6. The aim always is to prevent inconsistencies as much as possible.



```
IF (numOfOpGoals > 0) {
SET row:0
WHILE (row < numOfOpGoals) {
   CC "Core" GET_REC_ATTR_ROW_ID objid:(currentRoleID) attrid:(operaGoals) index:(row + 1)
  SET rowObjId:(rowid)
  CC "Core" GET_REC_CLASS_ID attrid:(operaGoals)
  CC "Core" GET_ATTR_ID classid:(classid) attrname:"Goal Name"
  CC "Core" GET_ATTR_VAL objid:(rowObjId) attrid:(attrid)
  SET currentOperatGoal:(val)
CC "Core" GET_ATTR_VAL objid:(currentRoleID) attrid:(roleNameId)
SET roleName:(val)
SET contentStr:(tokcat(contentStr, ";Semantic;REF mt:\"Actor Role Diagram \" m:\"" + modelname +
"\" c:\"Role\" o:\""+ roleName + "\";If you are not able to translate the operational goal '"
+currentOperatGoal+ "', entrusted to the role '" +roleName+ "' into at least two activities in
the corresponding role activity model, you have to review it.", "\t"))
SET numOfRes:(numOfRes+1)
  SET row:(row + 1)
}
}
```

Fig. 6. Consistency by construction: implementation of rule 2 as a guideline using AdoScript.

As to the consistency checking functionality, it refers to an AdoScript code that allows comparing the created views based on the defined consistency rules. As a result of executing such functionality, a list of consistency issues is displayed to the modeler who has to handle them. For some EKD consistency rules, the execution of this functionality requires a system/modeler interaction using "EDITFIELD" in AdoScript as is the case for rule 2 (see Fig. 7).

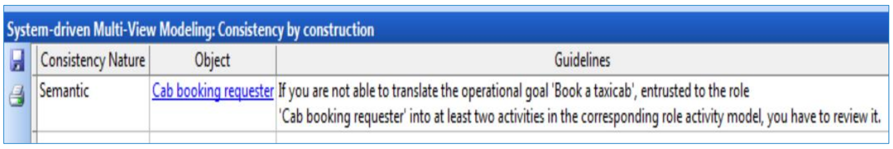


```
CC "AdoScript" EDITFIELD caption:(currentOperatGoal) title:"How many activities are linked to the G ?" text:""
    IF (ecode = 0) {
        SET numActLinkGoal:(VAL text)
            IF (numActLinkGoal < 2) {
            SET contentStr:(tokcat(contentStr, ";Semantic;REF mt:\"Actor Role Diagram (Single)\" m:\"" + modelname +
            "\" c:\"Role\" o:\""+ crtNameAttValue + "\";The operational goal '" +currentOperatGoal+"' :
            has been translated into only one activity in the role activity model", "\t"))
            SET numOfRes:(numOfRes+1)
            }
     }
```

Fig. 7. Consistency checking of rule 2 based on a system/ modeler interaction.

4.3 Case Study: A Cab Booking Process in EKD

This subsection demonstrates the use of the EKD tool for modeling a cab booking process. In this example, we give a brief idea of the execution of the two diverging functionalities of managing inter-view consistency: consistency by construction and consistency checking. The execution of the AdoScript implementing rule 2 according to the consistency by construction approach (see Fig. 6) for the cab booking process gives rise to a guideline (see Fig. 8).



System-driven Multi-View Modeling: Consistency by construction

Consistency Nature	Object	Guidelines
Semantic	Cab booking requester	If you are not able to translate the operational goal 'Book a taxicab', entrusted to the role 'Cab booking requester' into at least two activities in the corresponding role activity model, you have to review it.

Fig.8. Output of executing the consistency by construction functionality for rule 2.

Fig. 8 shows that the applied consistency rule (i) is of type semantic, (ii) concerns the role "Cab booking requester" and (iii) is implemented as a guideline. The latter insists on the necessity of translating the operational goal "Book a taxicab" into at least two activities in the corresponding role activity view.
The execution of the consistency checking functionality gives rise to the output presented in Fig. 9. As shown in this figure, such execution allows the detection of a set of consistency issues found in the created views. Each consistency issue is accompanied

with its nature (syntactic or semantic) and the modeling element that is source of this issue.

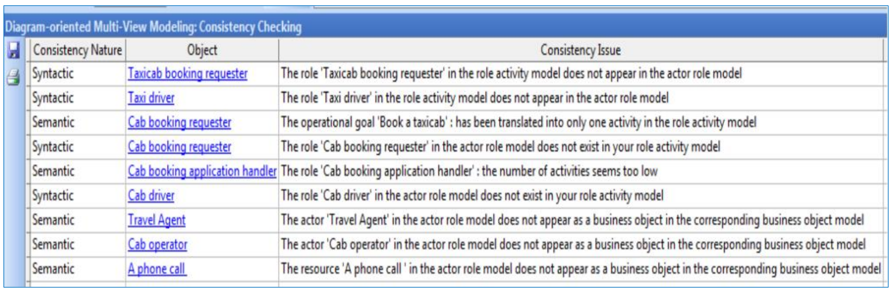


Diagram-oriented Multi-View Modeling: Consistency Checking

Consistency Nature	Object	Consistency Issue
Syntactic	Taxicab booking requester	The role 'Taxicab booking requester' in the role activity model does not appear in the actor role model
Syntactic	Taxi driver	The role 'Taxi driver' in the role activity model does not appear in the actor role model
Semantic	Cab booking requester	The operational goal 'Book a taxicab' : has been translated into only one activity in the role activity model
Syntactic	Cab booking requester	The role 'Cab booking requester' in the actor role model does not exist in your role activity model
Semantic	Cab booking application handler	The role 'Cab booking application handler' : the number of activities seems too low
Syntactic	Cab driver	The role 'Cab driver' in the actor role model does not exist in your role activity model
Semantic	Travel Agent	The actor 'Travel Agent' in the actor role model does not appear as a business object in the corresponding business object model
Semantic	Cab operator	The actor 'Cab operator' in the actor role model does not appear as a business object in the corresponding business object model
Semantic	A phone call	The resource 'A phone call ' in the actor role model does not appear as a business object in the corresponding business object model

Fig. 9. Output of executing the consistency checking functionality for several consistency rules.

5 Concluding Remarks

The paper at hand reports on a MVM tool for the EKD enterprise modeling method. In its current version, the tool accounts for two different ways of managing inter-view consistency: consistency by construction and consistency checking. In contrast to the latter, the former advocates preventing inconsistencies rather than correcting them.
The implementation of these two ways in one tool is needed for evaluating them with respect to quality attributes like time and consistency, and therefore for providing a reasonable reason to prefer one over the other. The aforementioned ways of managing inter-view consistency dovetail respectively with two different principles of MVM: system-oriented MVM and diagram-oriented MVM. The tool has been implemented using ADOxx platform and realized within the OMiLAB. The development of this tool is an ongoing work. Therefore, we intend to finish the development of the tool and evaluate it with students to determine its usefulness and to improve it accordingly.

References

1. Reineke, J., Tripakis, S.: Basic problems in multi-view modeling. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. Springer, Berlin, Heidelberg, pp. 217-232 (2014).
2. Bork, D., Karagiannis, D.: Model-driven development of multi-view modelling tools the muviemot approach. In: 9th International Conference on Software Paradigm Trends. IEEE, pp. IS-11 (2014).
3. Fischer, K., Panfilenko, D., Krumeich, J., Born, M., Desfray, P.: Based Modeling-Towards Defining the Viewpoint Concept and Implications for Supporting Modeling Tools. In EMISA, pp. 123-136 (2012).
4. Easterbrook, S., Yu, E., Aranda, J., Fan, Y., Horkoff, J., Leica, M., Qadir, R. A.:Do viewpoints lead to better conceptual models? An exploratory case study. In: 13th IEEE International Conference on Requirements Engineering. IEEE, pp. 199-208 (2005).
5. Bork, D.: Using Conceptual Modeling for Designing Multi-View Modeling Tools. In: 21st Americas Conference on Information Systems (AMCIS) (2015).

6. Karagiannis, D.: Agile modeling method engineering. In: Proceedings of the 19th Panhellenic Conference on Informatics. ACM, pp. 5-10 (2015)
7. Loucopoulos, P., Kavakli, V., Prekas, N., Rolland, C., Grosz, G. and Nurcan, S.: Using the EKD Approach: The Modelling Component, Paris (1997). Available at: https://hal.archives-ouvertes.fr/hal-00707997, last access: 19.05.2018.
8. Awadid, A., Nurcan, S.: A systematic literature review of consistency among business process models. In Enterprise, Business-Process and Information Systems Modeling. Springer, pp. 175-195 (2016).
9. Awadid, A., Nurcan, S.: Towards enhancing business process modeling formalisms of EKD with consistency consideration. In: IEEE Tenth International Conference on Research Challenges in Information Science (RCIS). IEEE, pp. 1-12 (2016).
10. Karagiannis, D.: Business process management: A holistic management approach. In International United Information Systems Conference. Springer, Berlin, Heidelberg. pp. 1-12 (2012).
11. Systems and software engineering- Architecture description: ISO/IEC/IEEE 42010. http://www.iso-architecture.org/ieee-1471/cm/, last access: 19.05.2018.
12. Solberg, A., Simmonds, D., Reddy, R., Ghosh, S., France, R.: Using aspect-oriented techniques to support separation of concerns in model driven development. In: 29th International Conference on Computer Software and Applications. IEEE, Vol. 1, pp. 121-126 (2005).
13. Persson, M., Törngren, M., Qamar, A., Westman, J., Biehl, M., Tripakis, S., Denil, J.: A characterization of integrated multi-view modeling in the context of embedded and cyber-physical systems. In: Proceedings of the Eleventh ACM International Conference on Embedded Software. IEEE Press p. 10 (2013).
14. Caetano, A. R. Silva, and J. Tribolet.: Business process decomposition-an approach based on the principle of separation of concerns, Enterprise Modelling and Information Systems Architectures, 5(1), 44-57 (2015).
15. Bork, D., Sinz, E. J.: Bridging the gap from a multi-view modelling method to the design of a multi-view modelling tool. Enterprise Modelling and Information Systems Architectures, 8(2), 25-41 (2013).
16. Karagiannis, D. and Kühn, H.: Metamodeling Platforms, Third International Conference EC-Web, Vol. LNCS 2455, p. 182 (2002).
17. Fill, H.-G. and Karagiannis, D.: On the Conceptualization of Modelling Methods Using the ADOxx Meta Modelling Platform. Enterprise Modelling and Information Systems Architectures, 8(1), pp. 4–25 (2013).
18. Ferstl, O. K., Sinz, E. J., Bork, D.: Tool support for the semantic object model. In Domain-Specific Conceptual Modeling. Springer, pp. 291-310 (2016).
19. Bock, A., Frank, U.: Multi-perspective Enterprise Modeling—Conceptual Foundation and Implementation with ADOxx. In Domain-Specific Conceptual Modeling. Springer, pp. 241-267 (2016).
20. Jeusfeld, M. A.: SemCheck: checking constraints for multi-perspective modeling languages. In Domain-Specific Conceptual Modeling. Springer, pp. 31-53 (2016).
21. Karagiannis, D., Buchmann, R. and Bork, D.: Managing Consistency in Multi-View Enterprise Models: An Approach based on Semantic Queries. In theTwenty-Fourth European Conference on Information Systems, p. Research paper 53 (2016).
22. Awadid, A., Bork, D., Karagiannis, D., Nurcan, S. Toward generic consistency patterns in multi-view enterprise modelling. In European Conference on Information Systems, Portsmouth (2018, in press).