Reflecting modeling languages regarding Wand and Weber's Decomposition Model

Florian Johannsen, Susanne Leist

Department of Management Information Systems
University of Regensburg
Universitaetsstraße 31
93053 Regensburg
Florian.Johannsen@wiwi.uni-regensburg.de
Susanne.Leist@wiwi.uni-regensburg.de

Abstract: The benefits of decomposing process models are widely recognized in literature. Nevertheless, the question of what actually constitutes a "good" decomposition of a business process model has not yet been dealt with in detail. Our starting point for obtaining a "good" decomposition is Wand and Weber's decomposition model for information systems which is specified for business process modeling. In the investigation at hand, we aim to explore in how far modeling languages support the user in fulfilling the decomposition conditions according to Wand and Weber. An important result of the investigation is that all investigated business process modeling languages (BPMN, eEPC, UML AD) can meet most of the requirements.

1 Introduction

Business process modeling is widely recognized as an important activity in a company [BWW09]. For instance, business process models can serve as a basis for decisions on IT-investments or the design and implementation of information systems [BWW09]. In view of its understandability the size of a business process model plays a central role [MRC07]. Depending on both the purpose of modeling and the target group considered, requirements on process models may differ. While a software engineer may be interested in details of a business process (e.g. complex control-flow mechanisms), another employee may only consider the more abstract model levels, giving him/her a basic understanding of the business process [BRB07]. For creating process models that are manageable and understandable in size, but also contain all the information needed (e.g. for software development, process improvement efforts etc.), they are decomposed "into simpler modules" [GL07]. In doing so, a process model is decomposed into several model levels that differ in detail [KKS04]. Nevertheless the characteristics that actually constitute a "good" decomposition [BM06, BM08] remain unclear. In practice, the decomposition of process models is usually done in an "ad hoc" fashion [RMD10]. Guidelines on how to decompose a model into subprocesses are missing [RMD10]. Our starting point is Wand and Weber's model for a good decomposition which was developed for information systems (see [WW89, We97]).

We specify this model for business process modeling giving business analysts a means to evaluate their decomposed models. As already mentioned in literature (see [Re09]), the potential of the Wand and Weber model seems promising for deriving criteria to judge whether the decomposition of a process model is "good" or "bad". As a first step in our investigation we evaluate the capabilities of common process modeling languages to enable Wand and Weber's decomposition model. It is our aim to explore how far these modeling languages support the user in fulfilling the defined conditions. Although, in fact, common modeling languages enable the decomposition e.g. by means of hierarchical functions in Event-driven Process Chains (EPCs) or subprocesses in the Business Process Modeling Notation (BPMN), the information given on a certain model level is not only dependent on the control-flow. Sometimes additional information is needed which becomes obvious by taking, for instance, a data-oriented view (e.g. focus on data elements). Since not all modeling languages support views that are not solely directed at the control-flow, the capabilities of a modeling language influence the quality of the decomposition.

This paper is structured as follows. In section two, we give a definition of the term decomposition, highlight the relevance of the Wand and Weber model, and describe the procedure for the research under study. Section three introduces the investigated business process modeling languages, and section four presents the decomposition model. Whether the process modeling languages are capable to support the decomposition model or not, is discussed in section five. Therefore requirements on process modeling languages are derived. Section six presents conclusions, a set of limitations, and potential directions for future research.

2 Conceptual Basics and Related Work

2.1 Decomposition and process model quality

Manifold metrics for judging the quality of a process model were recently developed (see [GL07, Va07, MRA10]). Moreover frameworks for evaluating conceptual models exist [SR98, KLS95]. In that context, decomposition is seen as a means to improve the understandability of a process model while reducing the likelihood of errors at the same time [MRA10]. The term decomposition is used in several publications, and many further publications (see e.g. [FS06, He09]) use terms with similar meanings (e.g. deconstruction, disaggregation, specialization). We define the decomposition of a process according to Weber [We97] as a set of subprocesses in such a way that the composition of the process equals the union of the compositions of the subprocesses in the set. Everything in the composition of the process is included in at least one subprocess in the set of subprocesses we chose. The decomposition of a process is represented in a level structure of subprocesses, and, on each level, the process or the subprocesses are displayed in a process model (see [We97]). Disaggregation and specialization are seen as special types of decomposition representing a part-of-relation respectively an is-a-relation. Most of the related work distinguishes heterogeneous types of decomposition for a given objective.

For example, vom Brocke [Br06] introduced design principles for reference modeling which aim to provide a greater flexibility in reference modeling. Malone et al. [Ma99] developed the "process compass" which differentiates between horizontal specialization by means of objects and vertical disaggregation into subprocesses. Heinrich et al. [He09] used disaggregation and specialization for decomposing a process landscape, aiming at identifying primarily functional similarities of the detailed subprocesses. Ferstl and Sinz [FS06] defined principles (the so-called decomposition rules) to recursively refine processes over several levels of detail which support disaggregation and specialisation. The principles were especially designed to be used within the framework of their SOM (semantic object model) methodology. Österle [Ös95] described a pragmatic procedure to decompose processes. The objective of the procedure is to detail macro processes into micro processes (see [Ös95]). Therefore he suggested four sources (services, business objects, process or activities of the customer process, existing activities) which help to derive activities from the macro process [Ös95]. While, based on their objectives, different principles of decomposition are defined in these publications, characteristics of a good decomposition are not investigated.

2.2 Relevance of Wand and Weber's decomposition model

As described above (section 2.1) various principles and suggestions to help practitioners achieve a good decomposition exist. But, to our knowledge, only one general theory of decomposition has so far been proposed in information systems (see [BM06]): Wand and Weber's good decomposition model (see [WW89, WW90, We97]). The decomposition model is part of the Bunge-Wand-Weber model (BWW model) [We97]. The BWW model is deeply rooted in the information system discipline [Re09] and considers the representational model, the state-tracking model, and the decomposition model as named above [We97]. The representational model has gained popularity as a means of the ontological analysis of modeling languages (see e.g. [RI07, Ro09, RRK07]). Therefore modeling languages are evaluated regarding ontological completeness and ontological clarity [Re09].

Both the state-tracking and the decomposition model are based on the concepts of the representational model [We97]. Details on the BWW model can be found in Weber [We97], for example. The decomposition model as it was originally developed by Wand and Weber comprises five conditions to judge the quality of a decomposition [We97]: *minimality, determinism, losslessness, minimum coupling,* and *strong cohesion*. These conditions help a user to decide whether an information system has been appropriately decomposed or not. Investigating these principles of good decomposition [We97] to support the creation of manageable business process models in large-scale initiatives has already been promoted by Recker et al. [Re09] as a promising field for research. If the decomposition model proves to be appropriate for that purpose, guidelines on how to decompose business process models may be derived in a subsequent step. This opinion is shared by Reijers and Mendling [RM08] as well. The positive effect of the decomposition conditions on the comprehensibility of UML diagrams has already been shown empirically (see [BM02, BM06, BM08]).

2.3 Procedure for deriving requirements on modeling languages

Since the decomposition conditions are based on the BWW representational model [We97] and modeling languages differ regarding their ontological completeness [Re09], the question arises in how far heterogeneous modeling languages are able to support Wand and Weber's decomposition conditions. To answer this question we adhere to the following procedure.

Figure 1: Procedure for deriving requirements and evaluating modeling languages

In a first step, the decomposition conditions, which have their origin in information systems, are being specified for business process modeling. Based on this specification, metrics are derived (step 2) to judge whether a process model adheres to the decomposition conditions as defined in step 1. These metrics serve as an objective basis for the evaluation of process models. The metrics are obtained from our specification of the decomposition conditions (step 1). Thus they address those modeling constructs that are focused for evaluating process models regarding their fulfillment of the decomposition conditions. By looking at the metrics and the modeling constructs they address, requirements on modeling languages can be defined straightaway. The third step of our procedure (figure 1) contains the formulation of the requirements on modeling languages. Using these requirements, common modeling languages are evaluated regarding their support of the decomposition conditions (step 4). In doing so, it becomes obvious as to which degree a decomposed process model that was designed by using a specific modeling language can be judged regarding its coherence with the decomposition conditions. This investigation is part of a research project which aims to define conditions for a good decomposition. The research project is based on the design science research method (see [He04]). Thus decomposition conditions will be build and evaluated afterwards. The investigation at hand serves to prove the capabilities of existing knowledge (modeling languages and Wand and Weber's decomposition model) and builds upon design science principles as well. We evaluate existing artifacts (modeling language) using requirements derived by a proposed solution (Wand and Weber's decomposition model) for a given problem (decomposition).

3 Business Process Modeling Languages

Manifold notations exist for modeling business processes. Especially the Business Process Modeling Notation (BPMN), the enhanced Event-driven Process Chains (eEPCs), and UML activity diagrams (UML ADs) have gained considerable attention in the field of business process modeling [MR08, Me09]. BPMN and UML have been developed and promoted by the OMG as standards in the modeling domain [MR08]. However, not only the ratification by the OMG, but also the growing tool support have contributed largely to their popularity in today`s business process modeling projects [MR08].

eEPCs are characterized by a high user acceptance [Me09, STA05], especially in the German-speaking community. A lot of reference models for different areas of application (e.g. computer integrated manufacturing, logistics or retail) are designed using eEPCs, while the notation is supported by manifold modeling tools as well [Me09]. Whereas other modeling languages exist (such as Petri-nets) (see [Mi10]), most of them were developed for analysis purposes and not for communicating models to business people and employees [Mi10] which hampers their popularity. Thus, in the following, we focus on eEPCs, UML ADs and BPMN. In addition, all of these languages support modeling constructs such as "collapsed subprocesses" (BPMN), "sub-activities" (UML AD), or "hierarchical functions" (eEPC) enabling the process design on different model levels.

Enhanced Event-driven Process Chains (eEPCs): Event-driven Process Chains were developed in the early 1990s for visualizing an integrated information system from a business perspective (see [STA05]). The EPC is part of the ARIS framework (see [STA05]). The ARIS framework comprises several views (e.g. data view, function view or organization view) that can be used to specify an EPC-model through additional information, for example data elements or organizational units [STA05]. In that context, we speak of enhanced Event-driven Process Chains (eEPCs).

Business Process Modeling Notation (BPMN): BPMN was officially introduced in 2004. The idea was to create a graphical standard to complement executable business process languages such as BPEL or BPML, for example [MR08]. In the meantime, Version 2.0 of the standard was released by the OMG. BPMN offers a variety of graphical modeling elements which are separated into basic and extended elements [OMG10].

UML Activity Diagrams (UML ADs): UML can be seen as a standard in the field of object-oriented modeling [Ru06]. It plays a dominant role in software engineering, since the functionality as well as the static structure of software can be described by several diagram types [Ru06]. In that context, activity diagrams (UML ADs) are important for modeling business processes, software is supposed to support. In the meantime, Version 2.4.1 of UML was released by the OMG [OMG11]. An "action" is the central element of UML activity diagrams for describing the behavior within a business process [Ru06].

The terminology in the field of business process modeling techniques is not standardized. We therefore stick to the terminology of Vanderfeesten et al. [Va07] which can be used for nearly all common business process modeling languages. Therefore we consider activities, events, data elements, directed arcs, connectors, and resources as constructs of a process model. Contrary to Vanderfeesten et al. [Va07] we also list events as separate elements, since events are an important concept during process execution [Mi10] which is emphasized by modeling languages such as the EPC. We adhere to this terminology in the following. This allows us to specify the decomposition conditions regardless of the business process modeling languages used (e.g. eEPC, BPMN etc.).

4 The Decomposition Model

Wand and Weber`s decomposition model [We97] focuses on the decomposition of information systems and specifies five conditions: (1) minimality, (2) determinism, (3) losslessness, (4) minimum coupling and (5) strong cohesion.

Some of the conditions can also be found in neighboring disciplines such as data modeling or business process modeling (see e.g. [BCN92, Be95, Va07]). These findings are referred to in order to specify the conditions for the purpose under study appropriately. In addition, Green and Rosemann [GR00] as well as Recker et al. [Re09] reflect modeling languages regarding the BWW model. Their results, too, help to specify the conditions.

Minimality condition: Following Weber [We97] a decomposition *„is good only if for every subsystem at every level in the level structure of the system there are no redundant state variables describing the subsystem"*. In the information systems domain this means that every subsystem of an information system should be characterized by the minimum number of attributes necessary for describing the subsystem [We97]. Minimality is an aspect that has also been addressed both in data modeling (see e.g. [BCN92]) and business process modeling (see e.g. [Be95]). According to Batini et al. [BCN92] a model is minimal if no object can be removed without causing a loss of information. If there is a loss of information or not, is to be judged by the end-user, and is therefore highly subjective. In addition, it is also the end-user who decides whether a specific modeling element is necessary or not. As already stated, a software engineer may expect more details in a process model than, for instance, a normal employee (see [BRB07]). Therefore a modeling construct can be seen as needless, if it is not required by the end-user. Another important aspect of minimality is seen in avoiding redundancies. But, while designing redundant-free models is a realistic goal in data modeling, this does not apply to business process models [Be95]. Therefore redundancies in business process models are quite common and may be necessary to design semantically correct models. Becker [Be95] gives some hints as to when activities in a business process model can be merged to avoid redundancies. Nevertheless the user's perception plays a central role in deciding whether a construct within a model should be modeled more than once (see [Be95]). Sometimes redundant-free process models may be difficult to understand because complex structures of different connectors (e.g. OR, XOR, AND) are needed. Therefore we distinguish between wanted and unwanted redundancies. Only unwanted redundant elements, however, should be avoided. The final decision whether an object in a process model is to be considered as unwanted redundant should be up to the end-user. Therefore, to evaluate different designs of a process model as regards minimality we propose the following (see table 1):

Verification of minimality	Metric	No.
Number (#) of activities, events, data elements, resources that are not required by the end-user or unwanted redundant in relation to all activities, events, data elements, resources.	# not required or unwanted redundant activities, events, data elements, resources/ # all activities, events, data elements, resources	1

Table 1: Verification of minimality

Regarding the metric, the size of the business process model is reflected upon when evaluating minimality. As mentioned, the end-user`s perspective is crucial at that point.

Determinism condition: According to Weber [We97] determinism can be defined the following way: *"For a given set of external (input) events at the system level, a*

decomposition is good only if for every subsystem at every level in the level structure of the system an event is either (a) an external event, or (b) a well-defined internal event".
The decomposition model mentions internal and external events [We97, Re09, GR00]. According to Burton-Jones and Meso [BM02], internal events are those events that occur during the execution of a process. Whether a specific internal event occurs, depends on decisions made or activities performed. The completeness check of a "purchase order" indicates that an order is either "complete" or "incomplete", depending on prior steps in the process, for example. The decomposition model requires internal events to be "well-defined" [We97, Re09, GR00]. This means that knowledge concerning the prior state enables a user to predict the subsequent event that will occur [We97]. In literature, there has been discussion about the relation between OR-splits and their effect on the instantiation of a process [ADK02]. It becomes obvious that the use of OR-splits often leads to designs in which events or subsequent states are hard to predict and may lead to complications during the actual execution of the process [ADK02]. Therefore the determinism of a process model suffers from the use of OR-splits. In addition, Cardoso showed the negative effect of OR-splits on the understandability of process models (see [Ca05]). Therefore he introduced the Control-Flow-Complexity-Metric [Ca05] that relates the complexity of a process model to the use of specific connectors. Negative impacts on the understandability of business process models are also caused by XOR-splits that are not based on conditional expressions. BPMN models using event-based XOR-splits, for example, are hard to interpret, since the branch to be chosen after the XOR-split depends on an event to occur, mainly the receipt of a message [OMG10]. In that case, internal events are only modeled in an implicit way, while the process flow actually comes to an abrupt stop at that point. External events, on the other hand, are triggered by factors that are beyond a company`s influence, for instance, a server crash at a supplier which prevents the regular stockpiling of the company`s warehouse [We97]. While the existence of such events should be recognized, it is hard to predict their effects on the actual process execution. When external events are known, activities to react to these external influences can be specified within a process model. Nevertheless it is often hard to identify all external events that may have an impact on a process. Therefore a modeler can only be expected to model external events, insofar as she/he is able to identify them. If a process model has few external events this can either be an indicator that the process is only little affected by external influences or that the modeler has not identified all external events properly. Despite these problems the relation between the number of external events and all events of the model can be used to judge to which degree a process model is stamped with external events. To evaluate different designs of a process model we therefore propose the following:

Verification of determinism	**Metrics**	**No.**
Number (#) of OR-splits in relation to all Split-operations of the model.	# OR-splits/ # all Split-operations	2
Control-Flow-Complexity-Metric according to [Ca05]. OR-splits have the most negative impact on the complexity of the model.	see [Ca05]	3
Number (#) of XOR-splits that are not based on conditional expressions in relation to all Split-operations of the model.	# XOR-splits not based on conditional expressions/ # all Split-operations	4
Number (#) of external events in relation to all events of the model.	# external events/ # events of the model	5

Table 2: Verification of determinism

Losslessness condition: Weber [We97] believes that *"a decomposition is good only if every hereditary state variable and every emergent state variable in a system is preserved in the decomposition"*. Simply speaking, the decomposition model demands "not to lose properties" of a thing that is being decomposed [WW89, We97]. No information must get lost during the decomposition. The ideas of Moody [Mo98] concerning the completeness of data models can be used to specify this aspect for business process models. A model therefore suffers from "losses", if certain constructs (e.g. activities, events) are required by the target group but cannot be found in the process model itself. The perspective of the target group once again becomes decisive in that context. In addition, Weber [We97] exemplifies that decomposition can lead to a false reproduction of the real world. This means that the semantics of a business process model may be distorted during decomposition and losses of the required semantics will occur. Considering resources can be of great help during decomposition. While two activities may look equal at first sight (e.g. "checking account"), they can be different regarding both the person performing the activity and the resources needed (see [Be95]). The underlying semantics can be completely different for these activities (e.g. "checking private customers` account" vs. "checking business customers` account"). What is more, syntactical errors occurring during decomposition will lead to misinterpretations and losses of the required semantics, too. As a consequence, the syntactical correctness of a model must be guaranteed for all model levels. Therefore "losslessness" of a model can be checked by means of the following metrics; the relation once again considers the size of the model:

Verification of losslessness	Metrics	No.
Number (#) of missing activities, events, data elements, resources on all model levels considering an original model (or the requirements of a user).	# missing activities, events, data elements, resources/# all activities, events, data elements, resources of an original model (or the requirements of an user)	6
Number (#) of wrongly designed constructs (syntactically and semantically) in relation to all required constructs.	# wrongly designed constructs/ # all required constructs	7

Table 3: Verification of losslessness

Minimum coupling condition: Weber [We97] states that *"a decomposition has minimum coupling iff the cardinality of the totality of input for each subsystem of the decomposition is less than or equal to the cardinality of the totality of input for each equivalent subsystem in the equivalent decomposition"*. Another aspect of the decomposition model addresses the coupling of the subsystems [We97]. The condition demands a minimum coupling which requires a minimum cardinality of the totality of the input [We97]. In process models, inputs are seen as data elements and the minimum cardinality refers to the number of relations between incoming data elements and activities. In the context of business process modeling, this idea is also supported by Vanderfeesten et al. [Va07]. According to Vanderfeesten et al. [Va07, VCR07] "coupling" measures the number of interconnections between the activities of a business process model. Thus it becomes obvious in how far various activities are dependent on each other [VRA08]. *"Two activities are coupled, if they contain one or more common data element(s)"* [Va07]. Accordingly, the degree of coupling of a business process model can be calculated by counting the number of coupled pairs (see [Va07, VRA08]).

The activities have to be selected pairwise beforehand. The mean is then determined on the basis of the total number of activities [Va07]. This approach has a strong focus on the data elements. With "minimal coupling" the activities in a business process model are neither too small nor too big (see [Va08]). Nevertheless, Wand and Weber admit that the meaning of "minimum coupling" is unclear and different interpretations can be found in literature (see [We97]). A further interpretation of Wand and Weber's definition of "minimum coupling" in business process models could be seen in the possibility to measure the strength of the connection between the activities (see [Va08]). In that case, mainly the control-flow would be focused. The degree of coupling depends on the complexity and the type of connections (e.g. XOR, AND, OR) between the activities [VCR07]. In Vanderfeesten et al. [Va08] the so called "Cross-Connectivity-Metric (CC)" is introduced for that purpose. The coupling of a business process model is thus determined by the complexity of the connections between its activities. To compare different designs as regards their degree of "coupling", the following metrics can be used (the relation once again considers the size of the model):

Verification of minimum coupling	Metrics	No.
Number (#) of "coupled pairs" (activities sharing the same data element) in relation to all activities (see [Va07]).	# coupled pairs/ # all activities*(# all activities-1)	8
Cross-Connectivity-Metric according to [Va08]. The strength of the connections between activities is considered by assigning weightings to the paths of the model.	see [Va08]	9

Table 4: Verification of coupling

Strong cohesion condition: According to Weber [We97] *"a set of outputs is maximally cohesive if all output variables affected by input variables are contained in the same set, and the addition of any other output to the set does not extend the set of inputs on which the existing outputs depend and there is no other output which depends on any of the input set defined by the existing output set"*. Whereas coupling tends to enlarge the size of an activity, cohesion downsizes activities [We97]. The *"strong cohesion condition"* requires for each activity of the process model that all output of an activity depends upon its input (see [We97, VRA08]). In literature, only few publications can be found on the "cohesion" of a business process model. Exceptions are Vanderfeesten et al. [VRA08] and Reijers and Vanderfeesten [RV04] who introduce metrics for measuring cohesion. A strong focus is placed on the "data elements" within an activity. These data elements are processed by operations. Operations can be understood as small parts of work within an activity [Re03]. Strong cohesion is given, if operations within an activity overlap by sharing "data elements", either as input or as output (*activity relation cohesion* according to [VRA08]). In addition, strong cohesion is also dominant when several of the data elements within an activity are used more than once (*activity information cohesion* according to [VRA08]). This definition comes very close to the definition in Wand and Weber's decomposition model, because they both define cohesion as mainly data-driven and focus the processing of data elements within the activities.

In summary, the cohesion of an activity is determined by the extent to which the operations of an activity "belong" to each other [VRA08, RV04]. Vanderfeesten et al. [VRA08] propose three metrics to determine the cohesion of an activity. The final process cohesion is then calculated on the basis of the cohesion values of the activities.

Verification of strong cohesion	Metrics	No.
The *activity relation cohesion* determines in how far the operations within one activity are related with one another [VRA08].	see [VRA08]	10
The *activity information cohesion* determines how many data elements are used more than once in relation to all the data elements [VRA08].	see [VRA08]	11
The *activity cohesion* is the product of the *activity relation cohesion* and the *activity information cohesion* [VRA08].	see [VRA08]	12

Table 5: Verification of cohesion

Although the conditions introduced are named decomposition conditions, they do not facilitate the procedure of decomposition. They are applied on the basis of the results of the decomposition and enable the evaluation of a decomposed process model by means of metrics (introduced above). The metrics` value helps to compare different alternative models, although the interpretation of differences between the metrics` values remains an open issue. Is it worth to reduce the value of coupled pairs in relation to all activities from 0.3 to 0.1, for example? Furthermore it has to be considered that the use of these metrics means an additional effort, since all metrics introduced can be calculated for all model levels of the designed alternatives. Since the decomposition of a process model into several, more detailed model levels always means adding semantics, user specifications have to be regarded as well. Therefore some metrics cannot be directly derived from the process models but have to imply users` knowledge or specification documents. These metrics are part of the conditions “losslessness” and “minimality”.

5 Evaluation of the Business Process Modeling Languages

5.1 Requirements based on the decomposition conditions

In the following, we derive requirements on modeling languages by looking at our specification of the decomposition conditions (see section 4) and the corresponding metrics that reflect our interpretation. In doing so, each requirement (see table 6) can be directly associated to the related decomposition condition as well as certain facets of our interpretation.

To fulfill the *minimality condition* according to our interpretation from section 4, a process model should not include unwanted redundant and not required elements (see also metric 1). Not only the decision whether an element is unwanted redundant, but also whether it is required or not, is up to the user. In this regard, the context of the process is decisive. Whereas the modeling language offers modeling constructs to represent the process, the user specifies them taking into account the context of the process. The modeling language cannot prevent the user from misinterpreting the requirements resulting from the context of the process (see [Mi10]). Therefore requirements for this condition cannot be defined.

The *determinism condition*, as it has been specified in section 4, requires a predictable control-flow of the process which implies that all internal events are well-defined. The aforementioned OR-splits often lead to designs in which events or subsequent states are hard to predict (see [ADK02]). This also becomes evident by metrics 2 and 3 we have introduced. In addition, if the conditions of the outgoing branches of an XOR-connector are not explicitly defined, the subsequent state is not determinable either (see [OMG10]).

This aspect is dealt with in metric 4, while the number of XOR-splits that are not based on conditional expressions should be minimal. A process modeling language should therefore enable the definition of conditions to specify the outgoing branches of an XOR-connector *(requirement 1 – see table 6)* and should not support an OR-connector *(requirement 3 – see table 6)* (see also metrics 2, 3 and 4). Contrary to poorly-defined internal events in a good decomposition, poorly-defined external events are permitted (see [We97, GR00]). This is due to the fact that it is often not possible to predict a subsequent state a priori that occurs as a result of an external event. The "determinism condition" only demands to represent external events in a model. External events in a process model are counted by metric 5 resulting from our specification of the condition. Accordingly, the process modeling language should be able to display external events *(requirement 2 – see table 6)*.

To fulfill the *losslessness condition* according to our interpretation (see section 4), hereditary and emergent elements of a process are to be preserved in the decomposition. Since only being based on the knowledge of users or specification documents with which a missing (see metric 6) or wrongly designed element (see metric 7) can be identified, the process modeling language is not able to support this condition. To detect syntactically wrongly designed elements, the process modeling language has to be specified by means of its metamodel *(requirement 4 – see table 6)*.

In order to be able to define the minimum cardinality of the *minimum coupling condition* (according to section 4) and evaluate a process regarding metric 8, the process modeling language has to display inputs and the flow between data elements and activities *(requirement 5 – see table 6)*. Earlier on (section 4), we made a suggestion to fulfill the *minimum coupling condition* which is not based on inputs, namely to investigate the strength of the connections between the activities by applying the Cross-Connectivity-Metric (see [Va08] and metric 9). The strength of the connection between the activities is measured considering all nodes (activities and connectors) and arcs. Therefore the process modeling language has to display activities, connectors as well as the arcs between them *(requirement 6 – see table 6)*.

Decomposition condition	Requirements	Corresponding metrics
Minimality	No requirements can be defined	metric 1
Determinism	The process modeling language has to provide modeling constructs for: • conditions to specify outgoing arcs of an XOR-connector **(requirement 1)** • external events **(requirement 2)** The process modeling language should not support an OR-connector **(requirement 3)**	metric 4 metric 5 metrics 2,3
Losslessness	The process modeling language is defined by its metamodel **(requirement 4)**	metrics 6,7
Minimal coupling	The process modeling language has to provide modeling constructs for: • input data elements and the flow between data elements and activities **(requirement 5)** • activities, connectors and arcs between them **(requirement 6)**	metric 8 metric 9
Strong cohesion	The process modeling language has to provide modeling constructs for: • input data elements **(requirement 7)** • output data elements **(requirement 8)** • intermediate data elements **(requirement 9)** • the flow between the data elements **(requirement 10)**	metrics 10,11,12

Table 6: Requirements on business process modeling languages

The *strong cohesion condition* (we have introduced in section 4) is related to the functionality a subsystem performs [WW89, We97] and requires for each activity of the process model that all output of an activity depends upon its input [We97]. To be able to measure cohesion with the suggested metrics (see metrics 10, 11 and 12) the process modeling language has to display all inputs and outputs for every activity *(requirements 7 and 8 – see table 6)*. The flow between input and output as well as possibly existing data elements between them by means of intermediate results are to be regarded as well *(requirements 9 and 10 – see table 6)*. A short overview of the identified requirements is given in table 6. It becomes obvious that ten requirements can be derived from our interpretation of the decomposition conditions. These requirements cover the range of modeling constructs needed to evaluate a process model regarding the decomposition conditions. The process of modeling a real-world situation is, however, subjective and thus not considered at this point.

5.2 Capabilities of business process modeling languages

Support of determinism*:* The *determinism condition* focuses on modeling constructs representing external events, OR-connectors, and conditional expressions related to XOR-operations (see section 4 and 5.1). In eEPCs, the outgoing branches of an XOR-split are specified by the events to follow. While it is possible in modeling tools such as ARIS to add attributes to the arcs which specify conditional expressions, these are usually not modeled on a graphical level. In recent years, the eEPC notation was enhanced by modeling constructs for visualizing inter-organizational business processes (see [KKS04]). As a consequence, external events of cooperation partners, too, become evident. It is also possible to use start events for expressing external events (see [GR00]). The eEPC-notation provides an OR-connector. BPMN supports the exclusive gateway. The decision which one of the outgoing arcs of the exclusive gateway is chosen depends on a condition that is visualized by labeling the outgoing arcs [OMG10]. BPMN offers a variety of event-types and different triggers [OMG10] that can be used to visualize the occurrence of an external event in the process model [Re09]. BPMN supports OR-connectors as well. In UML ADs, the decision node (and a corresponding conditional expression) is used for XOR-operations [Ru06]. UML 2.0 introduces the “accept event” which can be used to express external events [Ru06]. Contrary to BPMN and eEPCs, OR-operations are not considered by UML ADs.

Support of losslessness*:* For all notations considered, official metamodels are available (see [Sch98, OMG10, OMG05]). But these metamodels are either too focused on specific aspects of the modeling language (e.g. for BPMN: metamodel for choreography activity, artifacts metamodel, external relationship metamodel) or mainly address technical aspects. Nevertheless, literature provides metamodels which were derived from the available specifications providing a more manageable means for a practitioner to design syntactical correct business process models. In that context, Rosemann [Ro96] presents a comprising metamodel for eEPCs which also takes into account connectors and views, while Korherr and List [KL07] design a metamodel for BPMN. Bordbar and Staikopoulos [BS04] develop a metamodel for UML ADs in particular.

In summary, metamodels exist in literature which are less complex than those presented in the official specifications, helping a practitioner to design syntactically correct models.

Support of minimal coupling: On the one hand, minimum coupling can be determined by the interconnections between functions/activities/actions based on common data elements. On the other hand, the structure of the process model provides information to calculate the coupling degree [Va08, VCR07]. The first option takes a data-oriented view while the second option focuses the control-flow. All modeling languages considered offer modeling constructs to calculate the coupling degree according to our specification and design models with "minimum coupling". eEPCs support the data view (see [STA05]), while in BPMN data objects are used for presenting both information and data (see [OMG10]). UML ADs have object nodes representing data elements that are transferred from one action to another one [OMG05]. These can either be attached to an action symbol as a "pin" or to an object flow. In all modeling languages the connection between the functions/activities/actions is the control-flow.

Support of strong cohesion: The *strong cohesion condition* is based on a data-oriented view (see [VRA08, Re03]). As already stated, eEPCs support data elements, while the distinction between input and output data elements is possible. However, the flow between the data elements themselves is not visualized within an eEPC (see [STA05]). Additional diagrams would be necessary in that context (see [STA05]). In addition, possible intermediate data elements that are produced within a "basic" function while transforming an input data element to an output data element are not modeled. If the function was a "hierarchical function" with further model levels subjacent, additional data elements would be given. In BPMN, data objects can be differentiated as data input and data output on a graphical level, while the flow between the data objects is not explicitly modeled (see [OMG10]). Regarding basic activities no intermediate data elements are modeled that may be produced within the activity to create the final output data (see [OMG10]). In UML ADs, object nodes represent data elements, while the object flow respectively the "pin symbol" characterizes them as input or output data (see [Ru06]). While all modeling languages considered support input and output data, additional diagrams are necessary to highlight the flow between the data elements. Intermediate data elements within a function/activity/action in the sense of Vanderfeesten et al. [VRA08] are not explicitly modeled or supported. Therefore the degree of cohesion [VRA08] cannot be calculated by just looking at the process models.

Decomposition condition	Requirements	eEPC	BPMN	UML AD
Determinism	**Requirement 1** (conditions for arcs of a XOR-connector)	x	✓	✓
	Requirement 2 (constructs for external events)	✓	✓	✓
	Requirement 3 (no support of OR-connector)	x	x	✓
Losslessness	**Requirement 4** (definition of a metamodel)	0	0	0
Minimal coupling	**Requirement 5** (constructs for input data elements and flow between data elements and activities)	✓	✓	✓
	Requirement 6 (activities, connectors and arcs between them)	✓	✓	✓
Strong cohesion	**Requirement 7** (constructs for input data elements)	✓	✓	✓
	Requirement 8 (constructs for output data elements)	✓	✓	✓
	Requirement 9 (constructs for intermediate data elements)	x	x	x
	Requirement 10 (constructs for flow between data elements)	x	x	x

Key: ✓: fulfilled; 0: partly fulfilled; x: not fulfilled

Table 7: Results of the evaluation

Table 7 summarizes the findings. None of the modeling languages entirely fulfills the requirements derived in section 5.1. Major differences between the languages can be seen in the requirements regarding the *determinism condition*.

Some restrictions become obvious when evaluating the languages against the requirements derived from the *losslessness* and *strong cohesion condition*.

6 Summary and Outlook

The use of Wand and Weber's decomposition model for business process modeling is meant to facilitate the decomposition of the process model. This enables a better comprehensibility of the model for its users. Whereas this statement is the basis for our complete research project and will have to be empirically validated, we have only just started our investigation with this paper. The objective was to find out which of the three selected business process modeling languages (BPMN, eEPC, UML AD) is best able to support the decomposition conditions. A first result is that requirements on business process modeling languages could not be defined for all decomposition conditions. The capabilities of the modeling languages do not vary for most of the requirements which stresses the similarities of the languages. The main differences could be detected when fulfilling the requirements of the *determinism condition*. An important result to be incorporated into the research project is that the business process modeling languages can meet most of the requirements and that, for all deficiencies, supplementary models or an extension of the process modeling language can be provided. In that context, it is of special interest that none of the business process modeling languages is capable to model the data elements as it is required for the *strong cohesion condition*. Intermediate data elements as well as the flow between the data elements have to be documented in supplementary models which will be verified by means of conducting the empirical validation of the decomposition conditions. The results of the investigation underline the need for a better integration of data elements into business process modeling. As a restriction to the above, it has to be stated that the requirements on the modeling languages were derived from the authors` interpretation of the decomposition conditions by Wand and Weber [WW89, WW90, We97]. The conditions were specified by metrics allowing an objective evaluation of different design alternatives. Nevertheless there may be other interpretations of these conditions in the context of business process modeling. While process modeling itself is a subjective task, evaluation procedures in the field of process modeling, too, may underlie subjectivity. This refers to section 5.2 in particular. In addition, the investigation is limited, because only three process modeling languages were investigated. With the next steps of the research project we aim to validate the decomposition model and derive a decomposition method which comprises principles and practical guidelines for business analysts.

References

[ADK02] van der Aalst, W.M.P.; Desel, J.; Kindler, E.: On the semantics of EPCs: A vicious circle. In: EPK 2002: Business Process Management using EPCs, 2002; p. 71–80.

[BCN92] Batini, C.; Ceri, S.; Navathe, S.B.: Conceptual Database Design - An entitiy relationship approach. Benjamin/Cummings Publishing, Redwood City et al., 1992.

[Be95] Becker, J.: Strukturanalogien in Informationsmodellen: Ihre Definition, ihr Nutzen und ihr Einfluß auf die Bildung von Grundsätzen ordnungsmäßiger Modellierung (GoM). Wirtschaftsinformatik 95. Physica, Heidelberg, 1995, p. 133-150.

[BM02] Burton-Jones, A.; Meso, P.: How Good Are These UML Diagrams? An Empirical Test of the Wand and Weber Good Decomposition Model. In: International Conference on Information Systems (ICIS), 2002; p. 101-114.

[BM06] Burton-Jones, A.; Meso, P.: Conceptualizing Systems for Understanding: An Empirical Test of Decomposition Principles in Object-Oriented Analysis. Information Systems Research 2006; 17:38-60.

[BM08] Burton-Jones, A.; Meso, P.N.: The Effects of Decomposition Quality and Multiple Forms of Information on Novices' Understanding of a Domain from a Conceptual Model. Journal of the Association for Information Systems 2008; 9:748-802.

[BRB07] Bobrik, R.; Reichert, M.; Bauer, T.: View-Based Process Visualization. Lecture Notes in Computer Science 2007; Volume 4714/2007:88-95.

[Br06] vom Brocke, J.: Design Principles for Reference Modelling - Reusing Information Models by Means of Aggregation, Specialisation, Instantiation, and Analogy. In (Fettke, P., Loos, P. eds.): Reference Modelling for Business Systems Analysis. Idea Group Publishing, Hershey, USA, 2006.

[BS04] Bordbar, B.; Staikopoulos, A.: On Behavioural Model Transformation in Web Services. Lecture Notes in Computer Science 2004; 3289/2004:667-678.

[BWW09] Becker, J.; Weiß, B.; Winkelmann, A.: A Business Process Modeling Language for the Banking Sector - A Design Science Approach. In: Fifteenth Americas Conference on Information Systems (AMCIS), 2009; p. 1-11.

[Ca05] Cardoso, J.: How to Measure the Control-flow Complexity of Web Processes and Workflows. In (Fischer, L. ed.): Workflow Handbook. Lighthouse Point 2005.

[FS06] Ferstl, O.K.; Sinz, E.J.: Modeling of Business Systems Using SOM. In (Bernus, P., Mertins, K., Schmidt, G. eds.): Handbook on Architectures of Information Systems. Springer, Berlin etc., 2006, p. 347-367.

[GL07] Gruhn, V.; Laue, R.: Approaches for Business Process Model Complexity Metrics. In (Abramowicz, W., Mayr, H.C. eds.): Technologies for Business Information Systems. Springer, Berlin, 2007; p. 13-24.

[GR00] Green, P.; Rosemann, M.: Integrated process modeling: An ontological evaluation. Information Systems 2000; 25:73-87.

[He09] Heinrich, B. et al.: The process map as an instrument to standardize processes: design and application at a financial service provider. ISeB 2009; 7:81-102

[He04] Hevner et al.: Design Science in Information Systems Research. MISQ 2004; 28:75-105

[KKS04] Klein, R.; Kupsch, F.; Scheer, A.-W.: Modellierung inter-organisationaler Prozesse mit Ereignisgesteuerten Prozessketten, 2004.

[KL07] Korherr, B.; List, B.: Extending the EPC and the BPMN with Business Process Goals and Performance Measures. In: 9th ICEIS, 2007.

[KLS95] Krogstie, J.; Lindland, O.I.; Sindre, G.: Towards a Deeper Understanding of Quality in Requirements Engineering. In: Proceedings of the 7th CAISE, 1995; p. 82-95.

[Ma99] Malone, T.W. et al.: Tools for Inventing Organizations: Toward a Handbook of Organizational Processes. Management Science 1999; 45:425-443.

[Me09] Mendling, J.: Metrics for Process Models - Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness. Springer, Berlin et al., 2009.

[Mi10] Mili, H. et al.: Business process modeling languages: Sorting through the alphabet soup. ACM Computing Surveys 2010; 43:1-54.

[Mo98] Moody, D.L.: Metrics for Evaluating the Quality of Entity Relationship Models. Lecture Notes in Computer Science 1998; 507/1998:211-225.

[MR08] zur Muehlen, M.; Recker, J.: How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. Lecture Notes in Computer Science 2008; 5074/2008:465-479.

[MRA10] Mendling, J.; Reijers, H.; van der Aalst, W.: Seven process modeling guidelines. Information and Software Technology 2010; 52:127-136.

[MRC07] Mendling, J.; Reijers, H.A.; Cardoso, J.: What Makes Process Models Understandable? Lecture Notes in Computer Science 2007; 4714/2007:48-63.

[OMG05] OMG Unified Modeling Language (OMG UML) – Superstructure, 2005.

[OMG10] OMG: Business Process Model and Notation (BPMN) – Version 2.0, 2010.

[OMG11] OMG Unified Modeling Language, Infrastructure – Version 2.4.1, 2011.

[Ös95] Österle, H.: Business in the information age Springer, Berlin et al., 1995.

[Re03] Reijers, H.A.: A Cohesion Metric for the Definition of Activities in a Workflow Process. In: Eighth CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design, 2003, p. 116-125.

[Re09] Recker, J. et al.: Business process modeling: a comparative analysis. Journal of the Association for Information Systems 2009; 10:333-363.

[RI07] Recker, J.; Indulska, M.: An Ontology-Based Evaluation of Process Modeling with Petri Nets. Interoperability in Business Information Systems 2007; 2:45-64.

[RM08] Reijers, H.; Mendling, J.: Modularity in Process Models: Review and Effects. Lecture Notes in Computer Science 2008; 5240:20-35.

[RMD10] Reijers, H.A.; Mendling, J.; Dijkman, R.: On the Usefulness of Subprocesses in Business Process Models. BPM Report, 2010.

[Ro96] Rosemann, M.: Komplexitätsmanagement in Prozeßmodellen. Gabler-Verlag, Wiesbaden, 1996.

[Ro09] Rosemann, M. et al.: Using ontology for the representational analysis of process modelling techniques. International Journal of Business Process Integration and Management Decision 2009; 4:251-265.

[RRK07] Recker, J.; Rosemann, M.; Krogstie, J.: Ontology- Versus Pattern-Based Evaluation of Process Modeling Languages: A Comparison. Communications of the Association for Information Systems 2007; 20:774-799.

[Ru06] Russell, N. et al.: On the suitability of UML 2.0 activity diagrams for business process modelling. In: 3rd Asia-Pacific conference on Conceptual modelling, 2006.

[RV04] Reijers, H.A.; Vanderfeesten, I.T.P.: Cohesion and Coupling Metrics for Workflow Process Design Lecture Notes in Computer Science 2004; 3080:290-305.

[Sch98] Scheer, A.-W.: ARIS - Modellierungsmethoden - Metamodelle - Anwendungen. Springer, Berlin et al., 1998.

[SR98] Schütte, R.; Rotthowe, T.: The Guidelines of Modeling – An Approach to Enhance the Quality in Information Models. LNCS 1998; 1507/1998:240-254.

[STA05] Scheer, A.-W.; Thomas, O.; Adam, O.: Process Modeling Using Event-Driven Process Chains. In (Dumas, M., van der Aalst, W., Hofstede, A.T. eds.): Process-aware information systems. John Wiley and Sons 2005, p. 119-146.

[Va07] Vanderfeesten, I.T.P. et al: Quality Metrics for Business Process Models. In (Fischer, L. ed.): BPM and Workflow Handbook 2007. Future Strategies, p. 179-190.

[Va08] Vanderfeesten, I. et al.: On a Quest for Good Process Models: The Cross-Connectivity Metric. Lecture Notes in Computer Science 5074 2008; 5074:480-494

[VCR07] Vanderfeesten, I.; Cardoso, J.; Reijers, H.A.: A weighted coupling metric for business process models. In: Proceedings of the CAiSE 2007, p. 41-44.

[VRA08] Vanderfeesten, I.; Reijers, H.A.; van der Aalst, W.M.P.: Evaluating workflow process designs using cohesion and coupling metrics. Computers in Industry 2008; 59:420-437

[We97] Weber, R.: Ontological Foundations of Information Systems, Queensland, 1997.

[WW89] Wand, Y.; Weber, R.: A Model of Systems Decomposition. In: Tenth International Conference on Information Systems, 1989; p. 42-51.

[WW90] Wand, Y.; Weber, R.: Toward a theory of the deep structure of information systems. In: International Conference on Information Systems, 1990; p. 61-71.