Software Measures for Business Processes

Alessio Antonini[1], Alexandre Mello Ferreira[1], Sandro Morasca[2], and Giuseppe Pozzi[1]

[1] Politecnico di Milano, P.za L. da Vinci 32 I-20133 Milano -Italy-
[2] Università degli Studi dell'Insubria, via Carloni 78 I-22100 Como -Italy-
antonini@elet.polimi.it, ferreira@elet.polimi.it
sandro.morasca@uninsubria.it, giuseppe.pozzi@polimi.it

Abstract. Designing a business process, which is executed by a Workflow Management System, recalls the activity of writing software source code, which is executed by a computer. Different business processes may have different qualities, e.g., size, structural complexity, some of which can be measured based on the formal descriptions of the business processes. This paper defines measures for quantifying business process qualities by drawing on concepts that have been used for defining measures for software code.
Specifically, the measures we propose and apply to business processes are related to attributes of *activities*, *control-flow*, *data-flow*, and *resources*. This allows the business process designer to have a comprehensive evaluation of business processes according to several different attributes.
Keywords: Business process, Metrics, Size, Complexity, Coupling

1 Introduction

A *workflow schema*, also known as *graph* or *process model*, is the formal description of a business process (BP), where single atomic work units (*task*) are coordinated, scheduled and assigned to processing entities (*agent*) to achieve a common goal. BPs are car rentals, insurance claims, bank loans etc. Workflow schemata can be graphically described by several different formalisms: we use here the business process modeling notation BPMN.

The activity of defining a BP is similar to the activity of writing source code in a programming language: requirements are collected, analyzed, and implemented. Some Software Engineering measures apply to the activity of defining a BP and typically aim at quantifying the complexity, size, volume, of a software product for predicting its number of bugs, robustness, safety, and development and maintenance costs. For instance, there may be a significant correlation between the cyclomatic number of a software code, related to the control flow complexity of the code itself, and its fault-proneness [1].

We propose here a novel approach to define several measures for quantifying BP qualities, analyzing the formal description of a BP, and helping designers obtain a broader view of the BP being modeled. The remainder of this paper is organized as follows. Section 2 describes related work. Section 3 describes the proposed approach. Section 4 illustrates an application example and shows the results we obtained. Section 5 presents concluding remarks and future directions.

2 Related Work

In Empirical Software Engineering, a number of measures are defined for quantifying several software attributes for the coding phase and the other software life cycle activities. We mainly consider here the measures related to the software code. This section considers the related work, mainly focusing on the software measures from a pure Software Engineering approach (Section 2.1) and on process definition measures explicitly referred to BP (Section 2.2).

2.1 Software Measures

Many hundreds of software measures exist. It is typical in Empirical Software Engineering to divide attributes (i.e., qualities) into internal and external ones. Internal software attributes can be measured based only on the knowledge of a software artifact, such as *size*, *structural complexity*, *cohesion*, *coupling*, and *length* [2–4]. External software attributes (e.g. *maintainability*) can only be measured if additional knowledge is available about the environment of the software artifact and of the interactions between the software artifact and the environment. Internal software attributes may be easy to measure, while external software attributes may be much more difficult to measure [5].

The most relevant qualities for our approach are internal ones, namely, *size*, *structural complexity*, and *coupling*. When defining a new measure for an attribute, it is necessary to check that it complies with a few properties that may be expected for the measures of that attribute. Morasca [3, 4] defines properties, called axioms, which are introduced via a graph-based representation of a software artifact, called *system*. We now concisely summarize this approach, by referring to the simple example in Figure 1, which shows a system S composed of two *modules* M_1 and M_2. E_1 and E_2, E_3 and E_4 are the *elements* of M_1 and M_2, respectively, and are connected by a *relationship* R. In general, an element is a node of the graph, a relationship an arc of the graph, and a module a subgraph of a given graph. The axioms introduced for an internal software attribute are necessary properties. So, if a measure does not satisfy them, we exclude that measure as an adequate one for that attribute. If, instead, the measure satisfies them, then we have supporting evidence (though not necessarily sufficient) for the measure to be an adequate measure for the attribute, so we say that the measure is a candidate measure for that attribute, to be on the safe side.

We now provide an informal explanation [4] of the axioms of interest: *size*, *complexity*, and *coupling*. A measure is a candidate measure if it complies with all of the respective axioms.

Size. Considering Figure 1, the two axioms from Morasca [3, 4] for *size* measures assert that: a) the overall size of S can never be greater than the sum of the sizes of M_1 and of M_2, if every element E_i belong either to M_1, or to M_2, or to both of them; b) the overall size of S equals the sum of the sizes of M_1 and of M_2, if every element E_i belongs either to M_1, or to M_2, but *not* to both of them.

Complexity. Considering Figure 1, the two axioms for *complexity* measures assert that: a) the overall complexity of S can never be smaller than the sum of

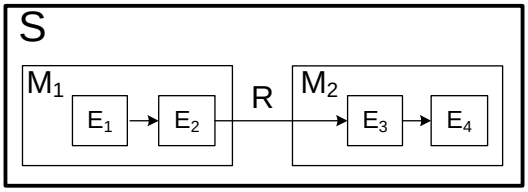

Fig. 1. An example system

the complexity of M_1 and of M_2, where S is composed by M_1 and M_2; b) the overall complexity of S equals the sum of the complexity of M_1 and of M_2, if M_1 and M_2 are two disjoint modules, with no connections from elements of one to elements of the other.

Coupling. Considering Figure 1, the four axioms for coupling measures relate to the relationships between the elements from one module and the elements from another module (*coupling*) in S: a) the coupling of a module with no external relationship is null; b) if we add a new relationship R_2 to an existing module M_1, the coupling of M_1 does not decrease; c) if we join the two modules M_1 and M_2, the coupling of the resulting module is never greater than the sum of the coupling of M_1 and of M_2; d) if we join two disjoint modules M_1 and M_2, the coupling of the resulting module is the sum of the coupling of M_1 and of M_2.

According to the above axioms, historical approaches of software measures can be reviewed. One of the most well known source code measures is LOC - lines of code, which simply counts how many lines there are within a source code. This measure is a *size* measure, according to the above axioms of *size*. The *information flow metric* from Kafura and Henry [6] was introduced as a complexity measure of the flow of information among the modules. This is not a measure for any of the above attributes.

Halstead's Software Science [7] moves from the number of distinct operators (n_1) and the number of distinct variables (n_2), as well as their total occurrences (N_1 and N_2, respectively). Software Science defines a measure for the *length* (N_1+N_2), *length estimator* ($n_1 \times log_2(n_1)+n_2 \times log_2(n_2)$), *volume* ($(N_1+N_2) \times log_2(n_1+n_2)$), and *difficulty* ($n_1/2 \times N_2/n_2$) of the code. The first measure is a *size* measure, as it fulfills all of the above axioms for *size*, but the others do not satisfy any of the above axiom sets for internal software attributes.

McCabe's cyclomatic complexity [8] counts how many linearly independent execution paths can be identified within a source code. Despite its name, this measure is not a *complexity* measure, according to the above axioms, though it may be transformed into one by simply subtracting 1 from it.

2.2 Quality Measures Applied to Process Definitions

A comprehensive review of the existing qualities and their extensions to the analysis of BPs comes from Cardoso et al. in [9, 10]. At first, LOC for a BP is defined as the number of activities (NOA), or the number of activities and

of control flow elements (NOAC), or the number of activities, joins, and splits (NOAJS). These are *size* measures, according to the above axioms.

The extension of the *information flow metric* of [6] is defined as $length(M) \times (fan-in(M) \times fan-out(M))^2$, where fan-in sums the local flows of information terminating in M plus the data structures read by M; fan-out sums the local flows of information originating from M plus the data structures written by M, consequently. The resulting measure does not fulfill any of the above axioms sets.

The extension of Halstead's *length estimator* (Halstead-based Process Complexity - HPC) takes n_1 as the number of activities, splits, joins, and control structures, n_2 as the number of variables or constants, N_1 as the overall number of occurrences of activities or of control structures, N_2 as the overall number of occurrences of variables or constants. Other measures, like *volume* and *difficulty* are computed as defined by [7]. The resulting measures *length estimator*, *volume*, and *difficulty* of HPC do not fulfill any of the above axiom sets.

By using McCabe's cyclomatic complexity, Cardoso et al. compute the cyclomatic number for a graph G by Mills' theorem as: $V(G) = D+1$, where D is the number of control structures. If a control structure has n outgoing arcs, its D value is $n-1$. It can be easily observed that this measure does not fulfill the above axiom set for *complexity*. The authors then propose to compute the cyclomatic complexity as D, and identify the contribution from every control structure a to the overall measure (control flow complexity - CFC) as: $CFC_{AND-split}(a) = 1$, $CFC_{OR-split}(a) = 2^{fan-out(a)} - 1$, $CFC_{XOR-split}(a) = fan-out(a)$. The overall CFC of a graph is the sum of $CFC_{AND-split}$, $CFC_{OR-split}$, $CFC_{XOR-split}$ for any existing control structure of that graph. This measure fulfills all the above axioms for *complexity*.

Rolon et al. [11] analyze and validate the CFC metric on processes designed by the business process modelling notation (BPMN). They mainly focus on the pure control flow dimension and just scratch the process data dimension, without considering in detail the other dimensions of the process model.

Vanderfeesten et al. [12] focus on the control flow model, introducing cohesion, coupling, and cross-connectivity metrics to estimate the understandability and the error-proneness of the control flow. Mendling [13] proposes a milestone formalism for Event-driven Process Chains (EPC) in designing the control flow of a BP: the formalism is tailored to verify the soundness of the control flow and to predict errors. While [12, 13] consider several new interesting metrics on the control flow, the other dimensions (data, organization) are not considered.

Generally speaking, the above approaches do not move from the axioms on size, complexity, and coupling, and mainly focus on the control flow of a BP.

3 Our Approach

The above measurements may not suffice to consider **all** the facets of a BP. We here introduce measures for *activity*, *control-flow*, *data-flow*, and *resource*.

A BP is a formally defined oriented graph, including a finite set of nodes (N) and the relationships which define the flow (F) of the process, where $F \subseteq N \times N$.

Thus the process p is defined as: $p =< N, F >$. A node can be a task (T) or a split/join (routing task - RT): obviously, $T \cap RT = \emptyset$.

3.1 Activity Attributes

The *activity attributes* consider the activities in a BP. We define a size measure. As a reference, we assume that the size of a task equals one, i.e. $Size_A(task) = 1$. The graph of a BP may include some supertasks (aka subflows or subprocesses): the $Size_A(supertask)$ is the sum of the sizes of the n tasks in the supertask, i.e. the number (n) of tasks in the supertask. Thus, if a process p has n_T tasks - not included into any supertask - and n_{ST} supertasks, the overall size is:

$$Size_A(p) = \sum_{i=1}^{n_T} Size_A(task_i) + \sum_{i=1}^{n_{ST}} Size_A(supertask_i) \quad (1)$$

Such a measure is a *size*, as all the axioms concerning size are fulfilled.

Proof. We consider the two axioms of Section 2.1 holding for a *size* measure. For the first one, we consider the process $p =< E_p, R_p >$ made by E_p elements and R_p relationships (arcs). We split p in two subprocesses p_1 and p_2. We have that $p' = p_1 \cup p_2$, where $p_1 =< E_{p_1}, R_{p_1} >$ and $p_2 =< E_{p_2}, R_{p_2} >$.

If $p_1 \cap p_2 \neq \emptyset$, then p_1 is made of R_{p_1} elements, and p_2 is made of $R_p - R_{p_1} + 1$; consequently, p' is made of $R_p + 1$ elements, implying $p \subset p'$. The axiom holds.

With respect to the second axiom, we still consider the process $p =< E_p, R_p >$ made by E_p elements and R_p relationships (arcs). We split p in two subprocesses p_1 and p_2. We have that $p' = p_1 \cup p_2$, where $p_1 =< E_{p_1}, R_{p_1} >$ and $p_2 =< E_{p_2}, R_{p_2} >$. If $p_1 \cap p_2 = \emptyset$, then p_1 is made of R_{p_1} elements, and p_2 is made of $R_p - R_{p_1}$; consequently, p' is made of R_p elements, implying that $p = p'$. The second axiom holds, too. □

3.2 Control-Flow Attributes

The *control-flow attributes* are defined based on the pure static structure of the graph of a BP. We introduce a complexity measure and a size measure.

Our approach adheres to the previous work from Cardoso et al [9, 10]. As in Section 2.2, we define a *control flow complexity* for any of the rt routing tasks (AND, OR, XOR split): $CFC_{AND-split}(rt) = 1$; $CFC_{OR-split}(rt) = 2^{fan-out(rt)} - 1$; $CFC_{XOR-split}(rt) = fan-out(rt)$. The overall *control flow complexity* (CFC) for a process p is the sum of the complexities originated by the splits as:

$$Complexity_{CF}(p) = \sum_{rt \in AND-split} CFC_{AND-split}(rt) + \sum_{rt \in OR-split} CFC_{OR-split}(rt) + \sum_{rt \in XOR-split} CFC_{XOR-split}(rt) \quad (2)$$

This measure fulfills all of the axioms for *complexity*.

We also introduce the concept of *size* of the graph of a BP, defined as the number of activities and control elements ($NOAC$) of a process. Thus, size is:

$$Size_{CF}(p) = NOAC \quad (3)$$

All the axioms concerning size are fulfilled: thus, this is a *size* measure.

3.3 Data-Flow Attributes

The *data-flow attributes* address the flow of information among the several activities involved in the graph of a BP. Intuitively, the more information flow among the activities, the higher the resulting complexity.

We define the set ($DataFlow$) of data managed by p as the set $v_1, v_2 \dots v_k$ such that: a) $\forall n \in N$, $Input(n)$ is the set of data received by the node n and it is defined as $V_i^n \subseteq DataFlow$; b) $\forall n \in N$, $Output(n)$ is the set of data produced by the node n and it is defined as $V_o^n \subseteq DataFlow$. Since a routing task (RT) can only read data, *Output* can be associated to normal tasks (T), only.

We extend the taxonomy of workflow relevant data from the *Workflow Management Coalition*, and consider four kinds of data managed by a process p:

i. Reference: these data (DR) univocally identify a process instance (e.g., customer_Id, student_Id, reservation_Id). In general, the path followed by these data is the control flow of the graph of a BP;
ii. Operational: these data (DO) are needed by an activity for its processing. Operational data (DO) include the data internally managed by a task, comparable to local variables, and - generally - not visible outside the task itself;
iii. Decision: these data (DD) are a subset of the operational data ($DD \subseteq DO$) and are used by routing tasks (RT) to selectively activate the outgoing/incoming arcs of the graph. Typically, conditions on arcs are defined by decision data DD (e.g., car_type="Sedan", student_level="M.Sc", meal_type="vegetarian");
iv. Contextual: these data (DC) belong to a wider category of data, are relevant for the BP, and can be used both as input and as output. Typically, DC include all the data managed by all the tasks of the process (e.g., customer_credit_card, examination_mark, reserved_flight_number).

Thus, we define $DataFlow$ as: $V_{i,o}^n = \{DR_{i,o}^n \cup DO_{i,o}^n \cup DC_{i,o}^n\}$. We also define an attribute considering the amount of data (number of data items) managed by the process p, where n_T is the number of the activities of the process.

$$Size_{DF}(p) = \sum_{j=1}^{n_T} V_{i,o}^j = \sum_{j=1}^{n_T} DR_{i,o}^j + \sum_{j=1}^{n_T} DO_{i,o}^j + \sum_{j=1}^{n_T} DC_{i,o}^j \qquad (4)$$

where $Size_{DF}$ for a process p is a *size*, since it fulfills all the axioms for size.

We can also define another measure, which relates to *complexity*: such a measure, takes into consideration both a component deriving from the routing tasks and a component deriving from the tasks which set up a BP. Since routing tasks (RT) have a lower complexity if compared with normal tasks (T), we can assume that the complexity of $DataFlow$ for a RT is: $Complexity_{DF}(RT) = 1$. On the other hand, the complexity of $DataFlow$ for a task T is: $Complexity_{DF}(T) = V_{i,o}^t$. Thus, the resulting overall complexity for data flows is the sum of the complexities of the two components:

$$Complexity_{DF}(p) = \sum_{j=1}^{n_{RT}} Complexity_{DF}(RT_j) + \sum_{j=1}^{n_T} Complexity_{DF}(T_j) \qquad (5)$$

The measure (5) is a *complexity*, since it fulfills all the axioms for complexity.

3.4 Resource Attributes

The *resource attributes* consider the resources required and used by the graph of a BP during process execution. If we assume to have R resources available for the execution of the n_T activities of a process p, we define a size measure as:

$$Size_R(p) = \sum_{i=1}^{n_T} r_i = R \tag{6}$$

This measure is very similar to the parameter $NOAC$ of Formula 3: again, all the axioms concerning size are fulfilled and the measure is a *size*.

We can also define a coupling measure for resources. This measure is strictly based on the BPMN notation used to graphically depict a BP. We consider the number of arcs which cross two (or more) swim lanes: every crossing means that the work item requires a new (different from the previous one) resource for its execution. We thus define a *coupling* measure as:

$$Coupling_R(p) = H \tag{7}$$

where H is the number of arcs which cross at least two swim lanes. The measure $Coupling_R$ for a process p is a *coupling*, since it fulfills all the axioms for coupling.

Proof. With respect to Figure 1, we consider the process S made by two modules M_1 and M_2, one module corresponding to one lane, only. The modules are connected via $r \neq 0$ relationships (i.e., r arcs which cross the two swim lanes), where $r \in \mathcal{N}$. Let H be the sum of the relationships between M_1 and M_2: hence, $Coupling_R(S) = H$. The coupling of the two separate modules is: $Coupling_R(M_1) = H_1$; $Coupling_R(M_2) = H_2$.

We have to consider the four axioms of Section 2.1 which hold for a *coupling* measure. With respect to the first axiom, if M_1 and M_2 have no external relationship, then $Coupling_R(p) = 0$. The coupling of a module with no external relationship is zero; the axiom holds.

With respect to the second axiom, if we assume to add a new relationship to M_1, the new value for the coupling will become $Coupling_R(M_1) = H_1 + 1$.

We define as $OuterR(m)$ the set of the relationships (arcs) outgoing from a module m. Thus, if M_2 is a subset of M_1 such that $< E_3, E_4 >=< E_1, E_2 >$, where E are the elements of the two modules, and $OuterR(m_2) \supseteq OuterR(M_1) \wedge$ $R_2 \supseteq R_1$, then: $Coupling_R(M_2) = H_2 + H_1 + 1 \geq H_1 + 1 = Coupling_R(M_1)$ and the axiom holds.

With respect to the third axiom, if M_1 and M_2 share r relationships, the resulting coupling is $Coupling_R(M_1 \cup M_2) = H_1 + H_2 - r$. Thus:

$Coupling_R(M_1 \cup M_2) = H_1 + H_2 - r \leq H_1 + H_2 = Couplig_R(M_1) + Coupling_R(M_2)$

and the axiom holds.

With respect to the fourth axiom, if M_1 and M_2 do not share r relationships, then $M_1 \cap M_2 = \emptyset \wedge OuterR(M_1) \cap OuterR(M_2) = \emptyset$. Thus, $r = 0$, and

$Coupling_R(M_1 \cup M_2) = H_1 + H_2 = Couplig_R(M_1) + Coupling_R(M_2)$

and the fourth axiom holds, too. □

4 Application Example

We introduce a reference process, and evaluate for it all the measures of Section 3.

4.1 Business Process of Reference

As reference process we use the "Manage Order" BP of Figure 2 according to the BPMN. The agent Sale Manager (topmost lane) receives the purchase order (Receive Order) from the customer and checks with the Finance Department (subprocess Check Finance) if the payment has been received. The order can be declined (Decline Order) or processed (Quantity Check, Quality Check). If the good is not in stock, the agent Production Planner (mid lane) plans the suitable production and waits for it (subprocess Produce). As soon as the supertask Produce is completed, the Sale Manager is informed and he/she can inform the customer (subprocess Notify Full Shipment). Finally, the agent Shipping Operator completes the process (subprocess Ship and Report).

The four subprocesses are quite simple: Check Finance is made by one task (Check Credit) and it is executed by the agent Finance Department; Notify Full Shipment is made by one task (Notify Customer) and it is executed by the agent Customer Support; Produce is made by one task (Assemble Good) and it is executed by the agent Manufacturing Department; Ship and Report includes two tasks (Ship and Delivery Report) and it is executed by the agent Shipping Operator.

Next, if we consider the data managed by the process (workflow relevant data), we find the following workflow variables: CustomerName, ProductName, OrderedQuantity, NumberOfItemsInStock, StockQuantityStatus, StockQualityStatus, NumberOfItemsToShip, NumberOfItemsToProduce, NumberOfProducedItems.

All in all, the process has $7+4+6 = 17$ tasks: 7 simple tasks, 4 complex tasks (aka subprocess: we do not count here how many tasks fit into one subprocess), and 6 routing tasks. The process has 3 main swim lanes and 4 resources in the subprocesses.

4.2 Quality Evaluation

We now apply the measures of Section 3 to the reference process of Section 4.1.
Activity Attributes. According to the Formula 1 and to the definitions from Section 3.1, for the process of Figure 2 we obtain $Size_A(ManageOrder) = 7 \times 1 + 3 \times 1 + 1 \times 2 = 12$. In fact, we have seven tasks, three supertasks made of one task each, one supertask made of two tasks.
Control Flow Attributes. By the Formulae 2, 3 and the definitions of Section 3.2, for the process of Figure 2 we obtain $Complexity_{CF}(ManageOrder) = 2 + 0 + 4 = 6$. In fact, we have two AND splits, no OR split, and four XOR splits. Next, we have $NOAC = 17$ (7 tasks, 4 complex tasks, 6 routing tasks). Thus, $Size_{CF}(ManageOrder) = 17$.
Data Flow Attributes. By the Formulae 5, 4, the definitions of Section 3.3 and the complete process definition (omitted here), we obtain that the number of workflow variables used as input or/and output of a task is 48, while 5 workflow

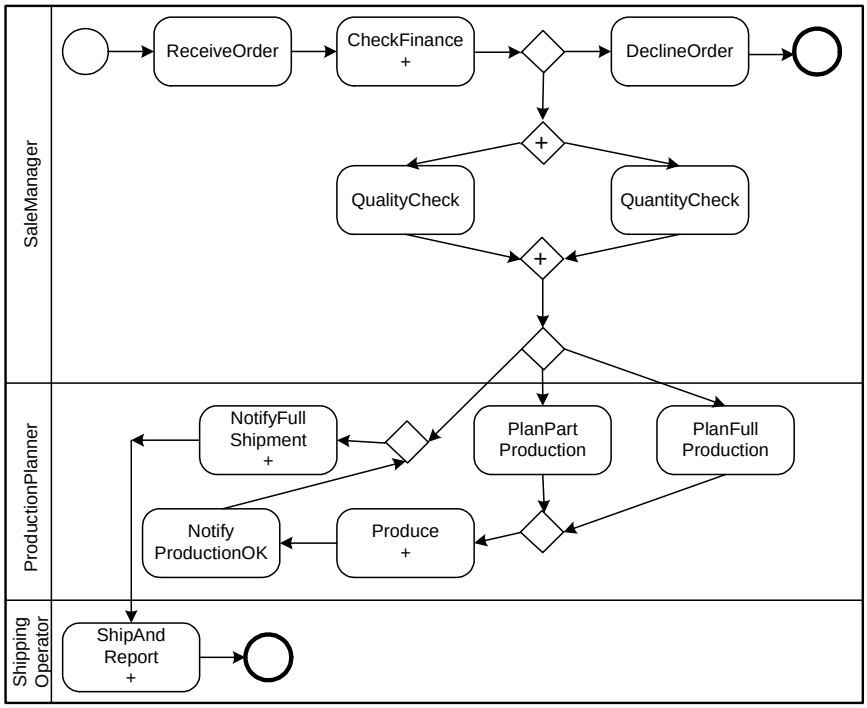

Fig. 2. The Manage Order process, which manages the activities inside an organization when selling goods. The process is graphically described according to the formalism from BPMN, and its code is saved in X-PDL. Supertasks are denoted by the "+" sign.

vars are used as an input to *split* tasks. Thus, $Size_{DF}(ManageOrder) = 48+5 = 53$. On the other hand, for the complexity of the data flow we observe one component coming from the normal tasks and one component coming from the *split* tasks. In our example, we have $Complexity_{DF}(ManageOrder) = 20 + 5 = 25$.

Resource Attributes. According to the Formulae 6, 7 and to the definitions from Section 3.4, for the process of Figure 2 we obtain $Size_R(ManageOrder) = 3$ since we have 3 resources involved in the execution of the process. In fact, we have 3 swim lanes, and we do not consider the resources involved in the execution of subprocesses. If we count how many lines (connecting arcs) cross the swim lanes, we obtain $Coupling_R(ManageOrder) = 4$.

4.3 Results

In order to test the approach, we consider a simplified version of the process of Figure 2, where in the topmost lane the activity Quality Check and the routing tasks labeled AND1 and AND2/XOR2 were removed - see Figure 3. We can now evaluate the measures of Section 4.2 for the new process of Figure 3.

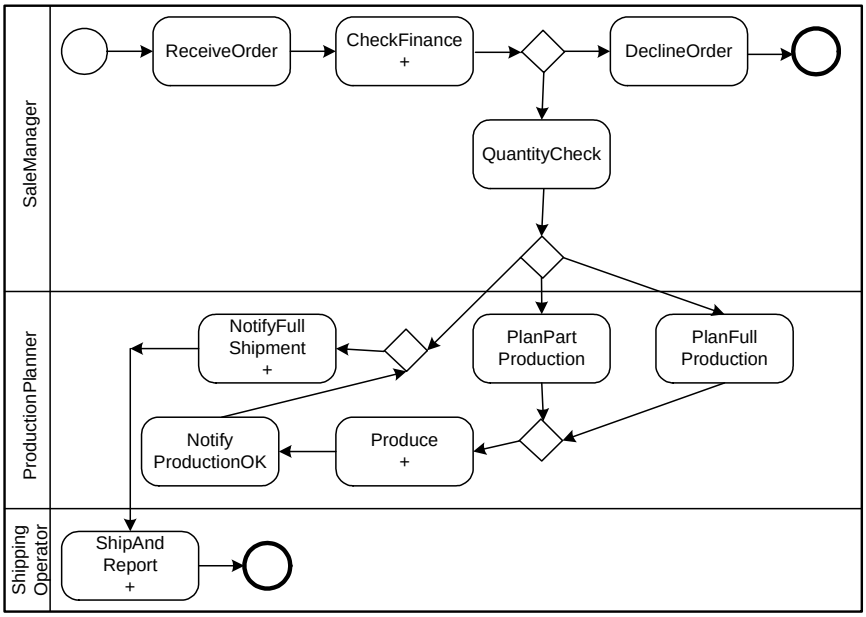

Fig. 3. The business process Manage Order$_2$ is a simplified version of the business process Manage Order. Supertasks are denoted by the "+" sign.

The new process has $6 + 4 + 4 = 14$ tasks: 6 simple tasks, 4 complex tasks (aka subprocess: we do not count here how many tasks fit into one subprocess) and 4 routing tasks. The process has 3 main swim lanes and 4 resources in the subprocesses.

Activity Attributes. As in Section 4.2, we compute $Size_A(ManageOrder_2) = 6 \times 1 + 3 \times 1 + 1 \times 2 = 11$. In fact, we have six tasks, three supertasks made of one task each, one supertask made of two tasks.

Control Flow Attributes. As in Section 4.2, $Complexity_{CF}(ManageOrder_2) = 0 + 0 + 4 = 4$. In fact, we have no AND split, no OR split, and four XOR splits. Next, we have $NOAC = 14$ (6 tasks, 4 complex tasks, 4 routing tasks): thus, $Size_{CF}(ManageOrder_2) = 14$.

Data Flow Attributes. As in Section 4.2, we obtain that the number of workflow vars used as input or/and output of a task is 44, while 4 workflow vars are used as an input to *split* tasks. Thus, $Size_{DF}(ManageOrder_2) = 44 + 4 = 48$. For the complexity of the data flow, we observe one component coming from the normal tasks and one component coming from the *split* tasks. In our example, we have $Complexity_{DF}(ManageOrder_2) = 18 + 4 = 22$.

Resource Attributes. As in Section 4.2, as the changes from Figure 2 to Figure 3 do not affect the swim lanes, we have $Size_R(ManageOrder_2) = 3$ and $Coupling_R(ManageOrder_2) = 4$. As one could easily expect, there is no difference in the resource qualities we evaluate on the two processes.

Observation. We now compare the measures of complexity and of size for the two processes of Figure 2 (Manage Order) and Figure 3 (Manage Order$_2$). Since Manage Order is heavier than Manage Order$_2$, which we derived from the previous as a simplified version, we expect that the measures on the two BP confirm that Manage Order has higher values than the corresponding values of Manage Order$_2$.

We consider the tuples for complexity ($Complexity_{CF}$, and $Complexity_{DF}$). Next, we consider the tuples for size($Size_A$, $Size_{CF}$, $Size_{DF}$, and $Size_R$):

$$\begin{aligned} Complexity(ManageOrder) &=< 6, 25 > \\ Complexity(ManageOrder_2) &=< 4, 22 > \\ Size(ManageOrder) &=< 12, 17, 53, 3 > \\ Size(ManageOrder_2) &=< 11, 14, 48, 3 > \end{aligned} \tag{8}$$

As we already outlined in Section 4.2, we do not consider here the *coupling* measure, since both processes present an identical value.

As expected, the results confirm that the simplified version of the process (Manage Order$_2$) shows smaller values for any measure both of *complexity* (second row of Formula 8) and of *size* (fourth row of Formula 8), if compared with the respective values for Manage Order (first and third rows of Formula 8). This enables us to assert that Manage Order is heavier than Manage Order$_2$.

5 Conclusions and Future Research Directions

This paper evaluates some measures which are relevant when formally defining a business process (BP). Moving from the software engineering approach and from previous work, we define some axioms according to which we classify the measures as *complexity*, *size*, and *coupling*. We do not limit our analysis to the pure *control flow* (*control flow attribute*) of the BP, but we also consider some other measures related to the activities (*activity attribute*), to the data flow among the activities (*data flow attribute*), and to the resources involved (*resource attribute*). For all of these facets, we define the corresponding measure, being it a *complexity*, a *size*, or a *cohesion*.

We consider the BPMN notation and save the process models in the X-PDL format recommended by the *Workflow Management Coalition* - WfMC. As a first test of our approach, we consider a BP and a lighter version of the same BP. Obtained results confirm that the measures evaluated on the lighter version provide values which are smaller than the homologous values of the original BP. We also developed a software tool, which automatically returns the values of the qualities for that considered BP. Thus, we obtain a set of values for every BP.

Future Research Directions. Our approach has so far been applied to few BPs, and we report here about one. We plan to consider a much greater number of processes, to further test our approach. The obtained results will be used to check whether there are correlations with other information related to the process, such as costs in developing, running, maintaining the process itself.

Furthermore, the approach can also be used in estimating the additional load (*size*, *complexity*, *cohesion*) to a BP when enriching it to add new functionalities, such as exception management or transaction management.

Acknowledgements. This work has been partially supported and funded by the GAMES project (`http://www.green-datacenters.eu`) of the European Commissions IST activity, 7th Framework Program under contract number ICT-248514. This work expresses the opinions of the authors and not necessarily those of the European Commission. We also thank the B.Sc. students Mirco Caldera, Stefano Gorla, and the M. Sc. student Duc Xuan Quang VU, who contributed in implementing the running prototype of the system described by the paper.

References

1. Grady, R.B.: Successful applying software metrics. IEEE Computer **27** (1994) 18–25
2. Briand, L.C., Morasca, S., Basili, V.R.: Property-based software engineering measurement. IEEE Trans. Software Eng. **22** (1996) 68–86
3. Morasca, S.: Measuring attributes of concurrent software specifications in petri nets. In: IEEE METRICS, IEEE Computer Society (1999) 100–110
4. Morasca, S.: Refining the axiomatic definition of internal software attributes. In Rombach, H.D., Elbaum, S.G., Münch, J., eds.: ESEM, ACM (2008) 188–197
5. Morasca, S.: A probability-based approach for measuring external attributes of software artifacts. In: ESEM. (2009) 44–55
6. Henry, S.M., Kafura, D.G.: Software structure metrics based on information flow. IEEE Trans. Software Eng. **7** (1981) 510–518
7. Halstead, M.H.: Elements of Software Science (Operating and programming systems series). Elsevier Science Inc., New York, NY, USA (1977)
8. McCabe, T.J.: A complexity measure. IEEE Trans. Software Eng. **2** (1976) 308–320
9. Cardoso, J.: Evaluating the process control-flow complexity measure. In: ICWS, IEEE Computer Society (2005) 803–804
10. Cardoso, J., Mendling, J., Neumann, G., Reijers, H.A.: A discourse on complexity of process models. In Eder, J., Dustdar, S., eds.: Business Process Management Workshops. Volume 4103 of Lecture Notes in Computer Science., Springer (2006) 117–128
11. Aguilar, E.R., Cardoso, J., García, F., Ruiz, F., Piattini, M.: Analysis and validation of control-flow complexity measures with bpmn process models. In Halpin, T.A., Krogstie, J., Nurcan, S., Proper, E., Schmidt, R., Soffer, P., Ukor, R., eds.: BMMDS/EMMSAD. Volume 29 of Lecture Notes in Business Information Processing., Springer (2009) 58–70
12. Vanderfeesten, I.T.P., Reijers, H.A., Mendling, J., van der Aalst, W.M.P., Cardoso, J.: On a quest for good process models: The cross-connectivity metric. In Bellahsene, Z., Léonard, M., eds.: CAiSE. Volume 5074 of Lecture Notes in Computer Science., Springer (2008) 480–494
13. Mendling, J.: Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness. Volume 6 of Lecture Notes in Business Information Processing. Springer (2008)