Measuring Method Complexity of the Case Management Modeling and Notation (CMMN)

Mike A. Marin
University of South Africa
IBM Software Group
25131 Mackenzie Street
Laguna Hills, CA 92653, USA
mmarin@acm.org

Hugo Lotriet
College of Science
Engineering and Technology
University of South Africa
Florida Park, Johannesburg,
South Africa
lotrihh@unisa.ac.za

John A. Van Der Poll
Graduate School of Business
Leadership (SBL)
University of South Africa
Midrand, 1686, Gauteng,
South Africa
vdpolja@unisa.ac.za

ABSTRACT

In 2014, the Object Management Group (OMG) published the Case Management Modeling and Notation (CMMN) version 1.0 specification, which is a new process modeling specification to complement its Business Process Modeling and Notation (BPMN) specification. The declarative nature of CMMN is intended to supplement the procedural perspective of BPMN. CMMN takes a data-centric view to process modeling based on business artifacts to provide flexibility for knowledge workers, while retaining the advantages of business process management (BPM) systems.

As far as the researchers are aware, this research is among the first contributions to the understanding of CMMN's method complexity in the context of other process modeling notations. We used the meta-model-based method complexity approach introduced by Rossi and Brinkkemper to evaluate the method complexity of CMMN. The results were compared with other popular process methods, including BPMN, Unified Modeling Language (UML) Activity diagrams, and Event-driven Process Charts (EPC), all of which have undergone similar evaluations by other researchers. The initial results indicated that CMMN 1.0 compares favorably with BPMN 1.2.

Categories and Subject Descriptors

D.2.2 [**Design Tools and Techniques**]; D.2.3 [**Coding Tools and Techniques**]: Standards; D.10 [**Design**]: Representation

General Terms

Measurement, Design, Human Factors, Standardization, Theory

SAICSIT '14, September 29-October 1 2014, Centurion, South Africa
Copyright 2014 ACM 978-1-4503-3246-0/14/09 ... $15.00
http://dx.doi.org/10.1145/2664591.2664608.

Keywords

Adaptive case management, Case management, Case handling, CMMN, BPMN, Modeling complexity, Complexity metrics, Process modeling complexity

1. INTRODUCTION

This research evaluates the method complexity of the CMMN 1.0 specification [19] and compares it with other popular process-modeling notations. Method complexity allows us to compare modeling notations, and it is important, because it is expected to affect the learnability, ease of use, and overall use of a method [21, 23]. CMMN promotes a data-centric and declarative perspective to process modeling [11], hence, it is important to understand how it compares with other methods for modeling business processes, such as BPMN, UML Activity diagrams, EPC, and others. BPMN, UML Activity diagrams, and EPC are well-known process methods with modeling notations, and in this study we compared specific versions of their specifications to CMMN 1.0.

Our research is based on the CMMN 1.0 formal meta-model described in the OMG specification [19]. Meta-models are important, because they describe the expressive power of a method by representing its vocabulary (i.e., concepts and properties) and valid constructs (i.e., relationships and roles) [23]. Most current process modeling notations are described in their formal specifications using UML meta-models; that is the case for CMMN 1.0 [19] and BPMN 2.0 [18]. A small subset of the CMMN formal meta-model is presented in Figure 1, showing a portion of the case class diagram in UML. Other modeling notations like UML 2.4 [17] are also described using UML meta-models. As described in [17] a model is an instance from a meta-model, and there may be multiple levels of meta-modeling. For example, standards organizations like the OMG rely on multiple levels of UML models to describe their formal specification. The meta-models for CMMN, BPMN, and UML Activity diagrams are described using UML models. In turn, models described in CMMN, BPMN, or UML Activity diagrams conform to the corresponding specification's UML meta-model. In this research, we focused on the CMMN 1.0 formal meta-model as described in the specification [19] using UML.

The method complexity comparison used in this research was based on the meta-model method complexity metrics introduced by Rossi and Brinkkemper in 1996 [23]; these metrics have been used to compare several business process

methods. They were used by Siau and Cao [24] in 2002 to evaluate UML 1.4 [3, 14] and its techniques including UML Activity diagrams. The same method complexity comparison was used to compare subsets of BPMN 1.2 [15] by Indulska *et al.* in 2009 [8]. Then, the work of Siau and Cao on UML 1.4 was used by Recker *et al.* [21] in 2009 to compare UML 1.4 Activity diagrams to BPMN 1.2. This produced a corpus of evaluations that we used to compare with the the CMMN notation.

Using Rossi and Brinkkemper [23] terminology, a process modeling notation like CMMN is considered a *method*. Methods can have multiple *techniques*, which correspond to the multiple diagrams that can be created. For example, UML is a method with multiple techniques including UML Activity diagrams, UML Class diagrams, UML State-Chart diagrams, etc. Both CMMN 1.0 and BPMN 1.2 have a single technique; the case plan model for CMMN, and the business process diagram for BPMN. However, the latest BPMN 2.0 has multiple techniques, including process diagrams, collaboration diagrams, conversation diagrams, and choreography diagrams. In this study, we use Rossi and Brinkkemper's method complexity to compare specific versions of the CMMN, BPMN, EPC methods, and the UML Activity diagram technique.

Section 2 briefly provides background information about CMMN and its relationship with BPMN. Section 3 describes the methodology used in this study to calculate CMMN's model complexity. Section 4 describes our findings in the context of previous studies. Section 5 provides suggestions for future research, and our conclusions are presented in Section 6. Appendix A uses an example by Korherr [10] to illustrate the four process modeling notations (i.e., UML Activity diagrams, EPC, BPMN, and CMMN).

2. BACKGROUND

An organization uses process modeling to describe the business processes to be automated by organizing the activities that need to be performed to achieve a business goal in the correct sequence. A business process model is described in a visual manner and represents the way that business representatives conduct the operation of a business [1]. Currently, BPMN is one of the most popular and widely used process modeling notation [20]. BPMN, as most process modeling standards focus on the tasks or activities and the control flow among them [7]. An example of a BPMN model is shown in Figure 5.c. UML Activity diagrams and EPCs are also common process modeling notations, and an example of a process modeled on each is shown in Figures 5.a and 5.b.

Case management [5, 27], or case handling, was first introduced by Van Der Aalst and Berens in 2001 [29] and by Reijers *et al.* [22] in 2003 to support the flexibility required by knowledge workers during a process and to help them better process exceptions that may occur during that process. Unlike workflow, BPM and most other process methods that focus on what *should* be done in a process, case management focuses on what *can* be done to achieve the business goal of the process [30].

In 2009, the Object Management Group (OMG) issued a request for proposal (RFP) for the creation of a standard modeling notation for use with case management [16] and to serve as a complement to its BPMN specification. The result of the OMG's effort was CMMN 1.0 [19], which was published in 2014. The CMMN specification addresses the case described in [29, 30, 22, 5, 27] with a data-centric approach based on business artifacts [11]. Case management as defined by CMMN provides flexibility to knowledge workers with regards to what tasks or activities should be performed and when they should be performed [7]. The main difference of CMMN from BPMN is the shift from procedural to declarative models [11]. An example of a CMMN model is shown in Figure 5.d.

The evaluation of method complexity for CMMN and its comparison to BPMN are important topics, because CMMN is designed to complement BPMN [16]. BPMN is widely used; nevertheless, it has its fair share of critics. Some researchers have criticized the BPMN 1.2 standard as being too complex for users. Zur Muehlen *et al.*, 2007 [33] concluded that BPMN has a complex modeling vocabulary. Recker *et al.* [21] found BPMN 1.2 to be more complex than UML 2.2. In order to deal with the complexity of BPMN, some researchers have identified common subsets of the BPMN notation that are in use today. In particular, Zur Muehlen and Recker [32] have identified three subsets of the BPMN notation in common use. These are a subset used by the U.S. department of defense [28, 8], a subset of commonly used BPMN constructs based on the analysis of 120 models [8, 32], and a use case analysis of a truck dealership in the U.S. [8, 31]. Those subsets were analyzed by [8], so we used those subsets to compare BPMN with CMMN in this study.

Siau and Rossi [25] asserted that there are a large number of modeling methods, and new methods continue to be created by practitioners and researchers. Siau and Rossi described the problem as not the large number of modeling methods, but the lack of techniques for evaluating and comparing those methods. They developed a comprehensive review of the approaches for evaluating modeling methods. Siau and Rossi provided four reasons to compare methods. First, researchers need to understand the nature of the methods to classify, study, and improve them. Second, practitioners can use comparisons as a way to select between methods. Third, method developers need to know the strengths and weaknesses of the various methods to design better methods. Four, as no method is suitable for all situations, comparison can help to select the right method for a particular situation. Siau and Rossi's assessment of modeling methods is applicable to process modeling methods.

There are at least three widely used process modeling methods, EPC, UML Activity diagrams, and BPMN. These modeling methods have overlapping functionality, and so, most processes can be modeled using any of the three methods. In 2014, CMMN was added to the list of processing modeling methods by the OMG. Historically, according to Mendling [12], EPC was introduced in 1992 by Keller *et al.* [9] and became popular in the 1990s as a conceptual business process modeling language. UML Activity diagrams appeared in 2001 in UML 1.4 intended to model organizational processes (i.e. *workflows*) according to Dumas and Hofstede [6]. In 2004, BPMN 1.0 was published as a graphical notation to model business processes [4]. The authors of BPMN evaluated several process modeling methods, including EPC and UML Activity diagrams, and decided to consolidate some of the ideas into BPMN 1.0 [4]. Because of the overlapping functionality and although CMMN addresses the specific use case described in [29, 22, 5, 27]; it is useful to compare these modeling methods.

Siau and Rossi [25] categorized the evaluation techniques into empirical and non-empirical techniques. Empirical evaluation techniques include surveys, laboratory experiments, field experiments, case studies, and action research. Non-empirical evaluation techniques include feature comparison, meta-model analysis, metrics analysis, paradigmatic analysis, contingency identification, ontological evaluation, and approaches based on cognitive psychology. Using the Siau and Rossi categorization, Rossi and Brinkkemper's [23] meta-model-based method complexity is categorized as metrics analysis, because it uses the method meta-model to compute metrics that are used for comparison purposes.

3. METHODOLOGY

For this study, we used Rossi and Brinkkemper's [23] meta-model-based method complexity that counts meta-model objects and properties. The cumulative method complexity $C'(\mathcal{M})$ in this approach is a vector in three-dimensional space, where the axes represent the count of objects $n(O_{\mathcal{M}})$, the count of relationships $n(R_{\mathcal{M}})$, and the count of properties $n(P_{\mathcal{M}})$. This allows for the comparison of the different vectors that correspond to the complexity $C'(\mathcal{M})$ of the methods being analyzed.

In this research, we were careful to indicate the versions of the different specifications being analyzed, because we expect the method complexity to change from one version of a specification to the next. We used BPMN version 1.2, and UML Activity diagrams version 1.4, because they had been studied by other researchers [8, 21, 24]. For EPC, we used the evaluation done by Indulska *et al.* [8], which used the version of EPC defined by Nüttgens and Rump [13] and based their calculations on the meta-model created by Becker *et al.* [2].

3.1 Meta-model-based method complexity

Rossi and Brinkkemper [23] recognized that most modeling methods are composed of techniques corresponding to the type of diagrams that can be created in the method. They formally defined the model of a technique in the object, property, relationship, role (OPRR) modeling language as a six-tuple $M_T = \langle O, P, R, X, r, p \rangle$, where,

- O is a finite set of object types.
- P is a finite set of property types.
- R is a finite set of relationship types.
- X is a finite set of role types.
- r is a mapping $r : R \rightarrow \{x \mid x \in \mathfrak{P}(X \times (\mathfrak{P}(O) - \{O\})) \wedge n(x) \leq 2\}$, where $n(x)$ is the cardinality of x, and $\mathfrak{P}(O)$ is the power set of set O.
- p is a partial mapping $p : NP \rightarrow \mathfrak{P}(O)$, where $NP = \{O \cup R \cup X\}$ is the set of non-property types

A method is a set of techniques. Therefore, for Rossi and Brinkkemper [23] the model of a method $\mathcal{M}$ is

$$M_{\mathcal{M}} = \bigcup_{T \in \mathcal{M}} M_T$$

. They defined 17 metrics, 12 for techniques M_T and the rest for the method itself $M_{\mathcal{M}}$. Indulska *et al.* [8], and Recker *et al.* [21] used a small subset of Rossi and Brinkkemper's metrics. We used the same subset that Indulska *et al.*, and Recker *et al.* used. That subset includes,

- $n(O_{\mathcal{M}})$ is the count of objects in the method, which corresponds to the count of objects in all the techniques
$$n(O_{\mathcal{M}}) = \sum_{T \in \mathcal{M}} n(O_T)$$
- $n(R_{\mathcal{M}})$ is the count of relationships in the method, which corresponds to the count of objects in all the techniques
$$n(R_{\mathcal{M}}) = \sum_{T \in \mathcal{M}} n(R_T)$$
- $n(P_{\mathcal{M}})$ is the count of properties in the method, which corresponds to the count of properties in all the techniques
$$n(P_{\mathcal{M}}) = \sum_{T \in \mathcal{M}} n(P_T)$$
- $C'(M_T) = \sqrt{n(O_T)^2 + n(R_T)^2 + n(P_T)^2}$ is the complexity of the technique
- $C'(\mathcal{M}) = \sqrt{n(O_{\mathcal{M}})^2 + n(R_{\mathcal{M}})^2 + n(P_{\mathcal{M}})^2}$ is the cumulative complexity of the method.

Note that by definition the complexity of a technique $C'(M_T)$ can be compared to the cumulative complexity of a method $C'(\mathcal{M})$. In our research, we compared the UML technique for Activity diagrams, with the methods for EPC, BPMN, and CMMN. Hence we treated UML Activity diagrams as a method. CMMN contains a single technique (i.e. case plan model), therefore $M_{\mathcal{M}} = M_T$, $n(O_{\mathcal{M}}) = n(O_T)$, $n(R_{\mathcal{M}}) = n(R_T)$, and $n(P_{\mathcal{M}}) = n(P_T)$.

In their work, Rossi and Brinkkemper [23] used the OPRR modeling language as implemented in MetaEdit [26] to model the methods to be analyzed. However, later studies conducted by Indulska *et al.* [8] and Recker *et al.* [21] were based on UML meta-models instead of OPRR meta-models. Therefore, in order to produce results comparable with previous evaluations [8, 21, 24], we used the meta-model method complexity metrics of Rossi and Brinkkemper, but we explicitly identified and followed the approach used by Indulska *et al.* This allowed us to use the normative CMMN meta-model described using UML class diagrams in the CMMN specification [19]. In addition, to avoid confusion and to be consistent with Rossi and Brinkkemper, UML meta-model classes are referred to as objects, and attributes are referred to as properties for the purposes of this study.

There are some differences between the meta-model-based method complexity as defined by Rossi and Brinkkemper [23] and how it was applied by Indulska *et al.* [8]. First, as noted above, Rossi and Brinkkemper used OPRR, and Indulska *et al.* used UML. In particular, Indulska *et al.* developed a UML meta-model of BPMN 1.2 for their research, because the BPMN 1.2 specification [15] did not describes a normative UML meta-model. Second, Rossi and Brinkkemper described a set of 17 complexity metrics for both techniques in a method and for the method itself. Because BPMN 1.2 contains a single technique i.e., business process diagrams, Indulska *et al.* used a smaller subset

that focused on the total cumulative method complexity of a method $C'(\mathcal{M})$. Third, Indulska *et al.* introduced the concept of full and concrete notation for BPMN 1.2. The full notation consists of the objects, relationships, and properties from the notation meta-model, and the concrete notation consists of the objects, relationships, and properties derived from the graphical notation. We focused on the full notation in this study, in accordance to Rossi and Brinkkemper, who used a simple, conceptual complexity to compare methods based on the meta-model.

3.2 Counting principles

After careful analysis of the meta-model and the approach used by Indulska *et al.* [8], we identified the following counting principles to be used for specifications described in UML meta-models:

1. The count of objects includes all of the abstract classes.
2. The count of properties excludes references to other classes.
3. The count of properties includes all objects and relationship properties.
4. The count of properties excludes tool-generated properties. The meta-model used in Indulska *et al.* was developed for their research, hence it did not include tool-generated properties.
5. Enumerations are not counted.

3.3 CMMN analysis

Using the counting principles above, we created Table 1 with the resulting objects and their properties for CMMN 1.0. The data were extracted from the CMMN 1.0 specification [19]. As an example Figure 1 shows part of the CMMN class diagram and using the counting principles; in Table 1 we included *CMMNElement*, *Case*, *Role*, *Stage*, and *CaseParameter* as objects. We also included properties *name* for *Case*, *name* for *Role*, and *description* for *CMMNElement*. The *Id* in *CMMNElement* is a tool-specific property, because it should be generated by the implementing tool, and therefore it does not appears in Table 1. In CMMN, we identified 11 tool-generated properties and removed them from the count. Examples of tool-generated properties in CMMN are *Id*, *exporter*, *exporterVersion*, and *expressionLanguage*, which are expected to be populated by the tool implementing the specification.

Table 2 provides the resulting relationships and their properties for CMMN 1.0. The data were extracted from the CMMN 1.0 specification [19]. As described by the counting principles, all of the CMMN meta-model classes are included, but only the properties that are expected to be provided by a user, and not provided by the tool.

Relationships in CMMN are challenging for this analysis, because there is a single relationship connector (dashed line) in the CMMN notation, and it is optional. In CMMN, the connector is used in two situations. First, it is used optionally to indicate event propagation represented in the meta-model by the *OnPart*, of which there are three classes, one abstract class and two concrete classes. Figure 2 shows an example of the event propagation notation between a case file item and a task. Second, the same connector dashed line is used for an expanded planning table in a human task. In

Table 1: CMMN 1.0 Objects and properties

Objects O_M	Properties P_M
CMMNElement	description
Definitions	name
Import	location
CaseFileItemDefinition	definitionType
CaseFileItemDefinition	name
CaseFileItemDefinition	structureRef
Property	name
Property	type
Case	name
Role	name
CaseFile	
CaseFileItem	multiplicity
CaseFileItem	name
EventListener	
Milestone	
PlanItemDefinition	name
TimerEventListener	timerExpression
StartTrigger	
CaseFileItemStartTrigger	standardEvent
PlanItemStartTrigger	standardEvent
UserEventListener	
PlanFragment	
PlanItem	name
Sentry	name
IfPart	
Expression	body
Stage	autoComplete
TableItem	
DiscretionaryItem	
ApplicabilityRule	name
Task	isBlocking
ProcessParameter	
Parameter	name
ParameterMapping	
CaseParameter	
HumanTask	
ProcessTask	
Process	
CaseTask	
PlanItemControl	
ManualActivationRule	name
RequiredRule	name
RepetitionRule	name

Table 2: CMMN 1.0 Relationships and properties

Relationships R_M	Properties P_M
OnPart	
CaseFileItemOnPart	standardEvent
PlanItemOnPart	standardEvent
PlanningTable	

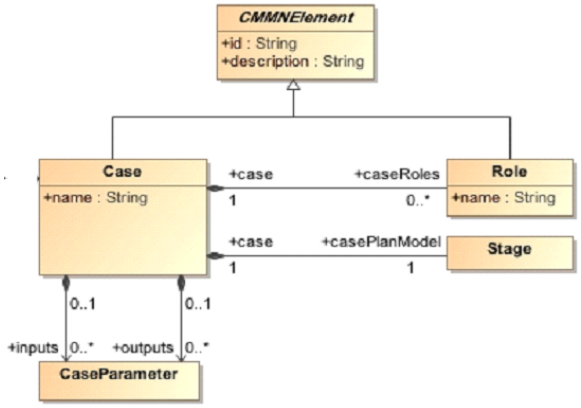

Figure 1: Portion of the CMMN Case class diagram

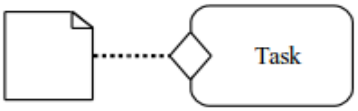

Figure 2: Event propagation notation between a case file item and a task

that situation, the connector is used to connect the human task to the discretionary items contained in the planning table. Figure 3 is an example of an expanded planning table in a human task containing two discretionary tasks. However, planning tables can also be used in stages, in which case the connector is never used. CMMN does not have an object in the meta-model to indicate the second situation. There is no object that represents the connection of an expanded table in a human task to its discretionary items. In Table 2, we decided to count the planning table as a relationship to account for that situation.

Having identified the appropriate set of objects, relationships, and properties using an approach similar to that of Indulska *et al.* [8] by using the derived counting principles, we proceeded to use Rossi and Brinkkemper's [23] method complexity calculations by counting cells in the tables that were filled. Based on Table 1, there are 39 non-duplicated object types $n(O_{\mathcal{M}})$ in the CMMN method. Based on Table 2, there are four non-duplicated relationship types $n(R_{\mathcal{M}})$. Based on Tables 1 and 2, there are 28 properties $n(P_{\mathcal{M}})$.

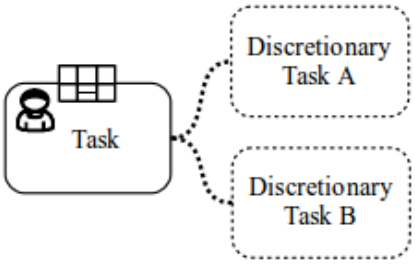

Figure 3: Planning table in a human task containing two discretionary tasks

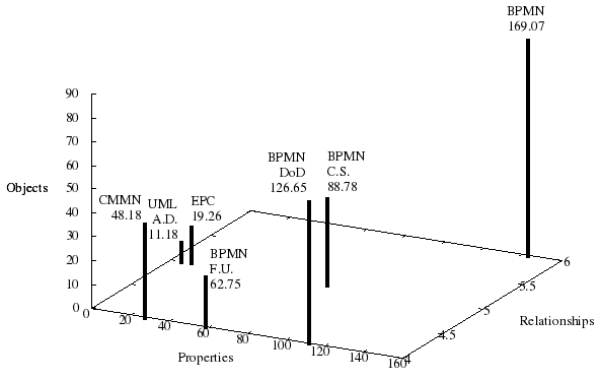

Figure 4: Object-Relationship-Property cube for process modeling notations

Therefore, the calculated cumulative method complexity $C'(\mathcal{M})$ for CMMN 1.0 was $C'(\mathcal{M}) = \sqrt{n(O_{\mathcal{M}})^2 + n(R_{\mathcal{M}})^2 + n(P_{\mathcal{M}})^2} = \sqrt{39^2 + 4^2 + 28^2} = 48.18$.

4. FINDINGS

Table 3 shows the CMMN 1.0 method complexity in the context of other popular process notations. The table is organized based on the cumulative method complexity $C'(\mathcal{M})$. For benchmark purposes, the methods included were reported by Siau and Cao [24], and Indulska *et al.* [8] using the BPMN 1.2 subsets identified by Zur Muehlen *et al.* [31, 32, 33].

Figure 4 shows the data from Table 3 in a three dimensional cube, similar to the cubes used by Rossi and Brinkkemper [23]. The lines are plotted starting in the (x, y) plane (properties, relationships) and indicates the number of objects i.e., starting from the point $\big(n(P_{\mathcal{M}}), n(R_{\mathcal{M}}), 0\big)$ with a length of $n(O_{\mathcal{M}})$. We labeled the lines with the corresponding method name from Table 3. However, for space reasons we abbreviated BPMN Case Study to BPMN C.S., BPMN Frequent Use to BPMN F.U., and UML Activity diagrams to UML A.D. We also added the cumulative complexity $C'(\mathcal{M})$ value, which by definition corresponds to the magnitude of the vector $\big(n(O_{\mathcal{M}}), n(R_{\mathcal{M}}), n(P_{\mathcal{M}})\big)$.

4.1 Implications

A calculated, cumulative complexity of 48.18 for CMMN 1.0 indicates that it is more complex than EPC, which has a cumulative complexity of 19.26, and UML 1.4 Activity diagrams, which has a cumulative complexity of 11.18, but less complex than BPMN 1.2, which has a cumulative complexity of 169.07. Figure 4 clearly shows how BPMN 1.2 makes extensive use of properties, relationships, and objects; much more than all the other methods. As indicated by Rossi and Brinkkemper [23], this may also indicate that BPMN 1.2 is more expressive than CMMN 1.0, which in turn may be more expressive than EPC and UML 1.4 Activity diagrams.

The results are encouraging, as they may indicate that CMMN should be simpler to learn than BPMN. As intended by Rossi and Brinkkemper [23], these results should be validated by empirical studies. Although empirical validation of the results is needed practitioners that find BPMN difficult to use may want to explore CMMN as an alternative for knowledge workers intensive processes that follows the use

Table 3: Method complexity comparison

Method	Objects $n(O_{\mathcal{M}})$	Relationships $n(R_{\mathcal{M}})$	Properties $n(P_{\mathcal{M}})$	Cumulative Complexity $C'(\mathcal{M})$
BPMN 1.2$^{\text{FULL}}$ [8]	90	6	143	169.07
BPMN 1.2 DoD$^{\text{FULL}}$ [28, 8]	59	4	112	126.65
BPMN 1.2 Case Study$^{\text{FULL}}$ [8, 31]	36	5	81	88.78
BPMN 1.2 Frequent Use$^{\text{FULL}}$ [8, 32]	21	4	59	62.75
CMMN 1.0	39	4	28	48.18
EPC$^{\text{FULL}}$ [8]	15	5	11	19.26
UML 1.4 Activity diagrams [24]	8	5	6	11.18

cases identified by Clair *et al.* [5], Swenson [27], Van Der Aalst and Berens [29], and by Reijers *et al.* [22].

The reliability and validity of the comparisons may be compromised by the mix of the meta-models and counting principles involved. We were careful to follow the original approach described by Rossi and Brinkkemper [23], and we adjusted it to compare the results with the work done by Siau and Cao [24], Indulska *et al.* [8], and Recker *et al.* [21]. In the process, it was noted that Rossi and Brinkkemper [23] and Siau and Cao [17] used OPRR meta-models to compare their results with Rossi and Brinkkemper results; Indulska *et al.* [8] used an UML meta-model; and Recker *et al.* [21] used the two meta-models. For CMMN 1.0 [19], we used the normative UML meta-model from the specification.

5. FUTURE RESEARCH

Future work in this area should start by calibrating the meta-model method complexity proposed by Rossi and Brinkkemper [23] to the UML meta-model. This is important, because organizations, e.g., the OMG, are currently using the UML meta-model to describe process modeling methods. The work of Rossi and Brinkkemper used the OPRR method's modeling language as implemented in MetaEdit [26] to describe meta-models. However, most of the modern methods, including most of those described in this paper, use UML to describe their meta-models. The use of UML introduces new nuances to the meta-models that were not present in OPRR. Therefore, we suggest that research be conducted to calibrate Rossi and Brinkkemper's approach to UML. In addition, it will be important to be explicit on how to consistently count the objects, relationships, and properties for UML models. In accordance with Rossi and Brinkkemper, enhancing UML's tools to automatically generate the metrics could be beneficial.

Standard specifications evolve and seem to become more complex over time; therefore, we carefully identified the version of the specification. The calculations for UML Activity diagrams and BPMN, done by Siau and Cao [24] and Indulska *et al.* [8], were based on early versions of those specifications. For example, the latest version of BPMN is 2.0 [18], which includes multiple techniques (i.e., process diagrams, collaboration diagrams, conversation diagrams, and choreography diagrams), while BPMN 1.2 described a single technique. Therefore, after Rossi and Brinkkemper's approach is recalibrated for UML, and solid guidelines are created based on it, future research to calculate method complexity for the latest versions of the specifications should be conducted.

Another venue for future research is to identify subsets of the CMMN notation. As process modelers begin to use CMMN, it will be useful to identify the subsets of the specification that start to emerge, similar to the work of Zur Muehlen *et al.* [31, 32, 33] for BPMN 1.2.

Following Rossi and Brinkkemper [23] suggestion, an empirical validation of this research should be conducted. The meta-model-based method complexity approach proposed by Rossi and Brinkkemper and used in this research provides an analysis of the conceptual part of the techniques and methods; but it should be empirical validated.

6. CONCLUSIONS

This exploratory research is among the first contributions to the understanding of CMMN's complexity in the context of other process modeling methods. We analyzed the method complexity of CMMN 1.0 using the approach proposed by Rossi and Brinkkemper [23]. In order to compare the results to other popular process modeling methods, we adjusted the approach to match the work done by Indulska *et al.* [8]. Then, we compared the CMMN 1.0 results with the results obtained by Siau and Cao [24], Indulska *et al.* [8], and Recker *et al.* [21]. Based on our findings, CMMN holds much promise, as the data indicates it compares favorably to BPMN. We suggest that future research include the calibration of the approach to UML-based meta-models, and empirical validation of this research.

7. REFERENCES

[1] W. Bandara and M. Rosemann. What Are the Secrets of Successful Process Modelling? Insights From an Australian Case Study. *Journal of Strategic Information Systems*, 10(3):47–68, 2005.

[2] J. Becker, P. Delfmann, T. Falk, and R. Knackstedt. Multiperspektivische ereignisgesteuerte Prozessketten. In *Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, Proceedings des 2. GI-Workshop EPK*, pages 45–60, Bamberg, Germany, 2003.

[3] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1999.

[4] BPMI.org. Business Process Modeling Notation (BPMN) v 1.0, 2004.

[5] L. C. Clair, C. Moore, and R. Vitti. Dynamic Case Management - An Old Idea Catches New Fire. Technical report, Forrester, Cambridge, MA, December 2009.

[6] M. Dumas and A. H. M. t. Hofstede. UML Activity Diagrams As a Workflow Specification Language. In *Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools*, pages 76–90, London, UK, UK, 2001. Springer-Verlag.

[7] R. Hull, J. Su, and R. Vaculin. Data Management Perspectives on Business Process Management: Tutorial Overview. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 943–948, New York, NY, USA, 2013. ACM.

[8] M. Indulska, M. Z. Muehlen, and J. C. Recker. Measuring Method Complexity: The Case of the Business Process Modeling Notation. BPM Center Report BPM-09-03, Business Process Management Center, 2009.

[9] G. Keller, M. Nüttgens, and A. W. Scheer. Semantische Prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK). Technical Report 89, Universität des Saarlandes, Germany, Saarbrücken, Germany, Jan. 1992.

[10] B. Korherr. *Business Process Modelling - Languages, Goals, and Variabilities*. Phd, Vienna University of Technology, 2008.

[11] M. Marin, R. Hull, and R. Vaculin. Data Centric BPM and the Emerging Case Management Standard: A Short Survey. In M. Rosa and P. Soffer, editors, *Business Process Management Workshops*, volume 132 of *Lecture Notes in Business Information Processing*, pages 24–30. Springer Berlin Heidelberg, 2013.

[12] J. Mendling. *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*. Springer Publishing Company, Incorporated, 2008.

[13] M. Nüttgens and F. Rump. Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In *Promise 2002 - Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen, Proceedings des GI-Workshops und Fachgruppentreffens*, pages 64–77, Potsdam, 2002.

[14] OMG. OMG Unified Modeling Language Specification, version 1.4, November 2001. Document formal/2001-09-67.

[15] OMG. Business Process Model and Notation (BPMN), version 1.2, January 2009. Document formal/2009-01-03.

[16] OMG. Case Management Process Modeling (CMPM) Request for Proposal, 2009. Document bmi/09-09-23.

[17] OMG. OMG Unified Modeling Language (OMG UML), Infrastructure, version 2.4.1, August 2011. Document formal/2011-08-05.

[18] OMG. Business Process Model and Notation (BPMN), version 2.0.2, December 2013. Document formal/2013-12-09.

[19] OMG. Case Management Model and Notation (CMMN), version 1.0, May 2014. Document formal/2014-05-05.

[20] J. C. Recker. Opportunities and constraints: the current struggle with BPMN. *Business Process Management Journal*, 16(1):181 – 201, 2010.

[21] J. C. Recker, M. zur Muehlen, K. Siau, J. Erickson, and M. Indulska. Measuring method complexity: UML versus BPMN. In *15th Americas Conference on Information Systems*, pages 6–9, San Francisco, 2009.

[22] H. A. Reijers, J. Rigter, and W. M. P. Van Der Aalst. The Case Handling Case. *International Journal of Cooperative Information Systems*, 12(3):365–391, 2003.

[23] M. Rossi and S. Brinkkemper. Complexity metrics for systems development methods and techniques. *Information Systems*, 21(2):209–227, 1996.

[24] K. Siau and Q. Cao. How Complex is the Unified Modeling Language? In K. Siau, editor, *Advanced Topics in Database Research Vol. 1*, pages 294–306. IGI Global, Hershey, PA, USA, 2002.

[25] K. Siau and M. Rossi. Evaluation of information modeling methods-a review. In *System Sciences, 1998., Proceedings of the Thirty-First Hawaii International Conference on*, volume 5, pages 314–322 vol.5. IEEE Comput. Soc, Jan 1998.

[26] K. Smolander, K. Lyytinen, V.-P. Tahvanainen, and P. Marttiin. MetaEdit – A flexible graphical environment for methodology modelling. In R. Andersen, J. Bubenko, JanisA., and A. Solvberg, editors, *Advanced Information Systems Engineering*, volume 498 of *Lecture Notes in Computer Science*, pages 168–193. Springer Berlin Heidelberg, 1991.

[27] K. Swenson. *Mastering the Unpredictable: How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done*. Landmark books. Meghan-Kiffer Press, 2010.

[28] U.S. Department of Defense. Enterprise Architecture based on Design Primitives and Patterns Guidelines for the Design and Development of Event-Trace Descriptions (DoDAF OV-6c) using BPMN. Technical report, Business Transformation Agency, 2009.

[29] W. M. P. Van Der Aalst and P. J. S. Berens. Beyond Workflow Management: Product-Driven Case Handling. In *Proceedings of the 2001 International ACM SIGGROUP*, pages 42–51, New York, 2001. ACM Press.

[30] W. M. P. Van Der Aalst, M. Weske, and D. Grunbauer. Case Handling: A New Paradigm for Business Process Support. *Data and Knowledge Engineering*, 53(2):129–162, 2005.

[31] M. Zur Muehlen and D. T. Ho. Service Process Innovation: A Case Study of BPMN in Practice. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, pages 372–372, Waikoloa, Big Island, Hawaii, Jan. 2008. IEEE.

[32] M. Zur Muehlen and J. C. Recker. How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. In *20th International Conference on Advanced Information Systems Engineering*, pages 465–479, Montpellier, France, 2008. Springer-Verlag.

[33] M. Zur Muehlen, J. C. Recker, and M. Indulska. Sometimes Less is More: Are Process Modeling

Languages Overly Complex? In *2007 Eleventh International IEEE EDOC Conference Workshop*, pages 197–204. IEEE, Oct 2007.

APPENDIX

A. PROCESS MODELING NOTATION EXAMPLES

For illustration purposes, we use a simple insurance claim process described by Korherr [10] to show an example of the four process modeling notations evaluated during this study. The very simple example is designed for illustration purposes only and does not reflect the complexity of a real insurance claim process. The example involves seven human activities (record the claim, calculate payment, contact the garage, check customer history, review results, pay for the damage, and do not pay for the damage). Korherr modeled the example in UML Activity diagram (Figure 5.a), EPC (Figure 5.b), and BPMN (Figure 5.c).

We developed a CMMN version of Korherr's example (Figure 5.d). Note that UML Activity diagram and BPMN are procedural, while EPC and CMMN are events driven notations. In this example, very little of the CMMN modeling notation was required. In CMMN, the seven human activities are modeled using human tasks (rounded rectangles with a human icon in the upper left corner) with entry criteria (diamond icon). In addition, a case file item representing the claim document was modeled, and connector (dashed line) representing event propagation between case file item and entry criteria was also used.

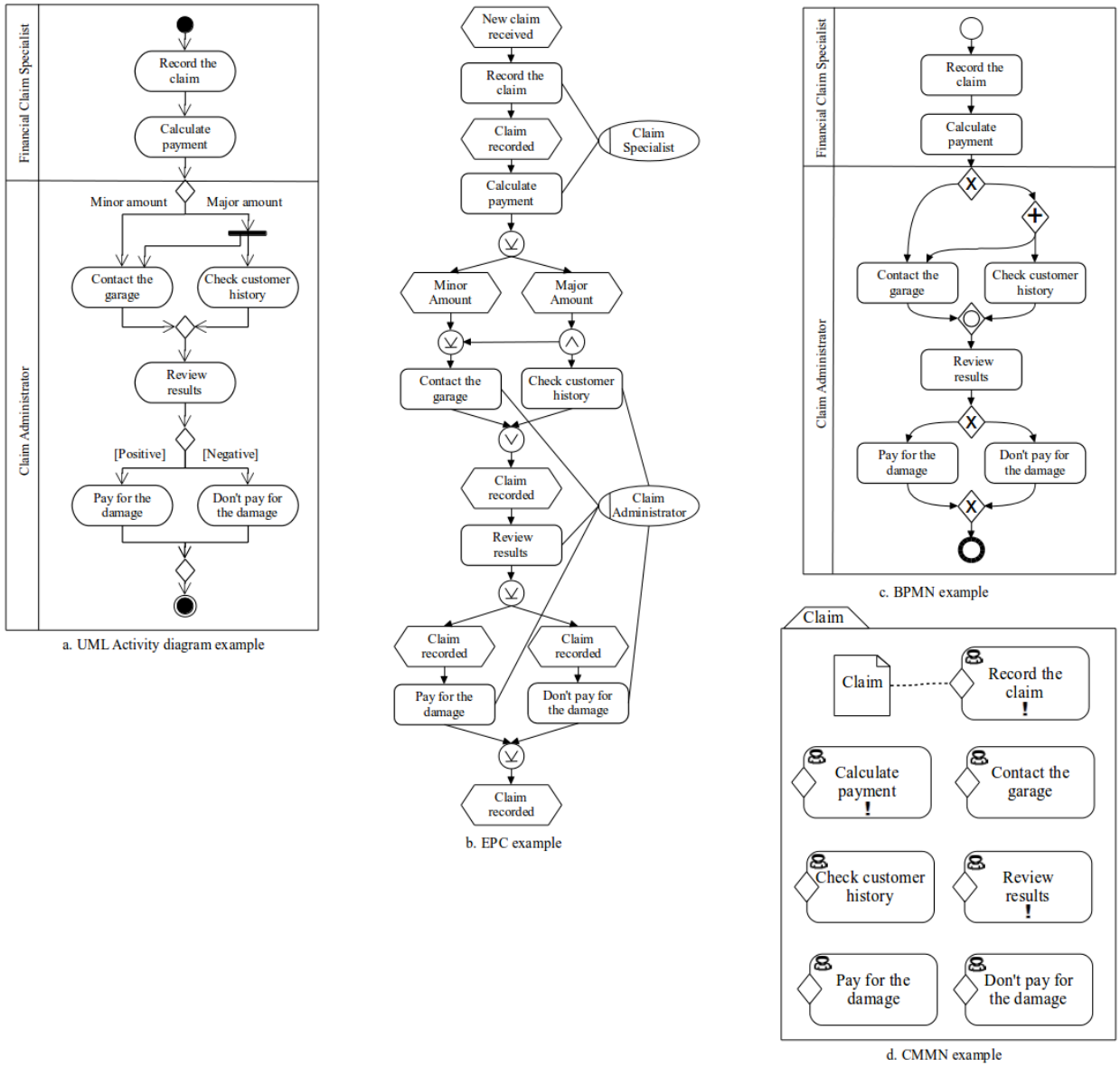

Figure 5: Example of a simple insurance claim process in four process notations