Automated Comparison of Process Improvement Reference Models based on Similarity Metrics

Simona Jeners, Horst Lichter, Elena Pyatkova
RWTH Aachen University, Research Group Software Construction
Aachen, Germany
{simona.jeners, horst.lichter, elena.pyatkova}@rwth-aachen.de

***Abstract*— A variety of reference models such as CMMI, COBIT or ITIL supports IT organizations to improve their processes. Although these process improvement reference models (IRM) cover different domains they also share some similarities. There are organizations that address multiple domains and want to take the guidance of different IRMs. As IRMs overlap in some processes, we present an approach to compare parts of IRMs (the IRMs' procedures) that is based on a common IRM integration model and on similarity metrics. Our approach enables organizations to efficiently adopt and assess multiple IRMs by automatically identifying similarities and specific details of the different IRMs.**

Keywords— reference models; software process improvement; comparison; meta-models; similarity metrics

I. Introduction

Nowadays, clients are requesting better and cheaper software products. However, the Standish Group regularly reports a high failure rate of IT-projects: 68% of IT-projects neither meet the deadlines nor achieve the requested quality or are cancelled [1]. One important factor to project success is the quality of the applied processes. Hence, more and more organizations want to establish and improve their processes systematically. Because the process improvement road is quite long and expensive it needs to be guided. To support process improvement different improvement reference models (IRM) such as CMMI (2010), ISO/IEC 15504 (2007) or COBIT (2007) can be considered and applied. IRMs are collections of best practices (often called procedures) based on experience and knowledge of many organizations.

The adoption and assessment of multiple IRMs bring additional benefits to organizations. The adoption allows organizations to exploit IRM synergy effects. On the one hand organizations can coordinately address different and common areas of IRMs. On the other hand the weaknesses of a single IRM can be overcome by the strengths of others. Furthermore, the assessment of the organizations' internal processes according to multiple IRMs increases the competition strength on the IT market.

One premise for organizations to be able to exploit the synergy effects of multiple IRMs and to efficiently assess them is an integrated view of IRMs allowing to compare procedures from different IRMs and to identify dependencies between them. Thus, organizations can effectively and efficiently adopt and assess multiple IRMs; the efficiency increases through an automated comparison approach.

II. Challenges and Goals

According to ISO/IEC 24744 [2] different IRMs "vary in format, content and level of prescription". Therefore, an automated comparison cannot be done without some preparatory steps. Our approach, MoSaIC (**Mo**del based **S**election of **I**mprovement **C**oncepts), enables a fine granular integration of multiple IRMs based on a common structure and terminology. Our comparison approach is based on this fine granular integration, i.e. we compare IRMs by comparing their concepts, such as activities, outputs, inputs, roles and purposes. Similar concepts from the different IRMs are stored in a different model (Integrated Concept Model-ICM) and connected by similarity relations, such as *composedOf* or *generalizationOf* [3]. The uniqueness of the ICM concepts, their consistent identification, the similarity relations between them and their traceability back to the original concepts of the IRMs allows to automatically identify similarities of different IRMs. A natural language processing tool for the extraction of such concepts according to predefined rules was developed to model these concepts consistently. An automated comparison would not be possible without a consistent normalization of the structure and of the terminology.

III. Related Work

Although considerable research has been devoted to the comparison of IRMs, the existing approaches do either not compare IRMs fine-grained or the comparison is done manually based on bilateral relations between parts of IRMs.

Ferreira, Machado and Paulk [4] define metrics to measure "size" and "complexity" of IRMs. To measure the "size", the shared scope of the IRMs' process areas (the number of common process areas) and their differences in the description detail are considered. The complexity is measured based on the internal coupling and the dependencies of the process areas. Therefore, this approach roughly compares several IRMs but does not consider their content. Content based comparisons are often provided by the publishers of IRMs which offer mappings between process areas or procedures (e.g. ISACA offers mappings between COBIT/CMMI and COBIT/ITIL). However, the similarities are subjective and only rough indicated, i.e. do not provide enough details about commonalities and differences necessary to support organizations in the adoption of multiple IRMs.

To overcome these problems, some authors try to integrate IRMs using formal models on a fine granular level. Ferchichi and Bigand [5] and Liao, Qu and Leung [6] define a common structure to link IRMs and reveal their similarities. For this purpose similar procedures of IRMs are connected manually. However, no information about common and different elements of the similar procedures is provided. The need of a process architecture in a multi-model context on a more fine-grained level is mentioned in a series of articles from SEI [7]. Fine granular elements mentioned in [8], [9], [10] or [11] such as inputs, outputs, roles are connected to identify similar procedures. However, they consider only the bilateral semantic equivalence between these elements. Our approach uses different similarity relations to get all the similar procedures and not only the ones that share exactly the same elements. Furthermore, we do not connect elements by bilateral similarity relations but use a new model that contains all the IRMs' elements and the relations between them to allow a multiple comparison between the procedures [3].

As our approach is based on an ontology of IRMs' elements (activities, artifacts, roles and purposes), we analyzed also ontology- or schema-based matching approaches that might be applied to support the identification of similar elements. Many diverse solutions have been proposed so far (mentioned in [12] or [13]). To match different schemas their terminology and structure can be compared by using e.g. linguistic resources, such as lexicons, thesauri resp. graph matching algorithms or structure meta-data. We evaluated some online dictionaries tools (e.g. WordNet [14], Rensselaer MSR Server [15], Wikipedia Miner [16]). Unfortunately, their ability to identify similar elements was not satisfactory. Reasons could be, that the dictionaries mostly contain general terms and not the specific IRMs terminology or that the similarity relations between the specific terms are not documented. Furthermore, we used OntoGen [17] to generate an ontology from the IRMs' context that can be also used to verify if two elements are similar or not. As this automation also did not delivered good results, we have created our own ontology based on the information extracted from IRMs and on predefined guidelines. As already mentioned, we added to the ontology all IRMs' fine grained elements and their similarity relations to allow an automated comparison between the IRMs' procedures (that contain these fine grained elements).

The remaining of this paper is organized as follows. In the fourth section, relevant aspects of similarity theory are presented. Based on these aspects, we describe our approach to identify similarities between IRMs. In the fifth section we discuss the results of an evaluation done by professional CMMI, COBIT and SPICE experts. Furthermore, we give an overview of our future work. Conclusions and a summary conclude this paper in the last chapter.

IV. Determining Similarity between IRMs

In the following we present the MoSaIC similarity algorithm. First, we give a brief introduction to similarity theory. Then, we present our algorithm that is based on similarity metrics. Finally, we illustrate our comparison approach by some examples.

A. Similarity Theory

In general, similarity is an important property because it is fundamental for our cognition. According to Goldstone and Son [18] similarity plays a key role in problem solving, remembering, prediction, and categorization. In fact, if there were no similar objects and events, an individual would perceive each situation as a new one and would have to learn how to use each particular object. The notion of similarity is applied in different domains. For instance, in geometry two objects are similar if they have the same shape; in psychology they are similar if they can be put into the same category. As there is no common definition of "similarity" we refer to the definition of Goodman [19]: *Objects are similar if they have a set of common features*.

There are several methods to determine similarity between objects. Based on measurement theory we distinguish the following four categories: **a) Spatial methods** consider objects as points or vectors in the n-dimensional space [20]. Well-known examples of spatial methods are the Cosine Similarity Measure or the Euclidean Distance**; b) Feature-based methods** consider objects as a finite unsorted set of features; they calculate the similarity with respect to their features. For example, Tversky [21] combines the numbers of similar and different features of different objects to calculate their similarity; **c) Transformational methods**, e.g. the Levenshtein Distance [22], consider the features of two objects and their order. They count the transformations needed to convert one object into the other; i.e., the smaller the number of transformations, the higher their similarity; **d) Alignment methods** like Structure Mapping Engine [23] use features of objects and their relations to determine similarity.

As in our case the order of concepts (objects' features) should not be considered, the transformational methods could not be applied. Furthermore, the alignment methods compare two objects that are represented as hierarchies of features related by a certain relation. As we have different relations between the concepts, these methods could not be applied too. The feature-based methods consider only common and different concepts but not concepts that have something in common (that are not equal but also not different). This issue is considered by the spatial methods because they regard the distance between the features of compared objects. We used two similarity metrics in our approach:

TABLE I. Similiarity Methods

Cosine-Distance Variant	Weighted EuclideanDistance
$\mathrm{Sim}\,(n,m) = \dfrac{2\,\mathrm{depth}\,(\mathrm{LCA}(n,m))}{\mathrm{depth}(n) + \mathrm{depth}(m)}$	$d_{p,q} = \sqrt{\sum_{i=1}^{n} w_i\,(q_i - p_i)^2}$

Ganesan et al. [24] proposed a variant of the **Cosine Distance method** to consider hierarchy information of the features and thus the similarity distance between these in the hierarchy. As ICM concepts may be related by the *generalizationOf*-relation they may form hierarchies as well. The *LCA(n, m)* is the Lowest Common Ancestor. The *depth(n)* is the number of edges from LCA(n, m) to n. The **Weighted Euclidean Distance** considers two features vectors with their corresponding feature pair weights. In our approach the procedures are composed of different element types that also have different weights (see next sections).

B. The MoSaIC Comparison Approach

The MoSaIC similarity algorithm considers IRMs' procedures. As we found reasonable similarities between COBIT control objectives, COBIT control practices, CMMI specific-goals, generic-goals, -practices, sub-practices, SPICE practices and Functional Safety objectives and requirements, we consider these procedures.

Procedures (PROCs) are similar, if their parts, called AUs (AUs) are similar. An AU contains concept elements (CEs): one activity and all its associated inputs, outputs, purposes and roles. Analogously, AUs are similar, if their concept elements (CEs) are similar. Hence, the basic idea of our similarity algorithm is to determine similarity on different levels (on the concept, AU and procedure level). A first similarity metric calculates the similarity of the concepts. Another similarity metric t aggregates these results to compute the similarity of the considered AUs. Finally, on the procedure level, some hints about the similarity of procedures are given. All metrics calculate a value between 0 and 1. The metric specifications are based on assumptions (A1-4) and have to meet some requirements (R1-3).

TABLE II. ASSUMPTIONS AND REQUIREMENTS

A1	*Activity* is the most important concept type.
A2	*Role*, *Input* and *Purpose* are the less important concepts types.
A3	*Output* is more important than *Role*, *Input* and *Purpose*, but less important than *Activity*.
A4	All part-concepts of a common whole-concept are not similar.
R1	Each metric should be differentiable (different inputs cause different results), comparable, reproducible (the same input always leads to the same value) and plausible (the values meet the representative condition) [25] [26].
R2	The calculated similarity values should reflect the importance of the conceptual elements.
R3	The number of conceptual elements of an AU should not influence its similarity value.

As the activities and outputs reflect the procedure's actual work, which is expected to be performed, we consider them as the most important elements of a procedure. Role, input and purpose are also important but only give additional information about how to perform an activity to produce an output. We consider the process of achieving an output (the activity) more important than its result (the output).

The last assumption (A4) refers to the composition of concepts. As in real world, part-concepts building a whole-concept are semantically different (e.g. "wheel", "door", "engine", "seat" are elements of a "car" and are not similar). If concepts share some properties, they are modeled as specializations of their parent ("*generalizationOf*"-relation).

On the **CE-level**, we compute similarity values for all pairs of concept elements (ce_1, ce_2) of the same type of two AUs. If a certain type (e.g. Role) is present only in one AU, a pair with the existing concept element and a null-element is created. Obviously, the similarity of such a pair is 0. The *SimCE* similarity metric takes into account possible semantic relations between the CEs in the ICM. We define *SimCE* as follows:

$SimCE(ce_1, ce_2) = 1$, iff ce_1 and ce_2 refer to the same concept.

$\boldsymbol{SimCE(ce_1, ce_2) = \frac{2\, depth\,(LCA(ce_1, ce_2))}{depth(ce_1) + depth(ce_2)}}$, iff ce_1 and ce_2 refer to concepts connected by *generalizationOf*-relations. *SimCE* is computed acc. to a variant of the Cosine Distance (see section 2). I.e., *SimCE* is high if CEs are located deeply in the *generalizationOf*-hierarchy (the hierarchy root does not specializes any other element) and LCA is close to both CEs.

$\boldsymbol{SimCE(ce_1, ce_2) = \prod_n^{i=1} partPercentage(l_i)}$, iff ce_1 and ce_2 refer to concepts that are connected by *composedOf*-relations. Based on assumption (A4), *SimCE* is the percentage a part (ce_2) represents of its whole (ce_1). Thus, each part has its own weight defined by experts that model the concepts. As there may be n part-of-levels l_i between ce_1 and ce_2 ($l_1 = ce_1$), *SimCE* is calculated by multiplying the similarity values of all part-of-levels between ce_1 and ce_2.

$SimCE(ce_1, ce_2) = SimCE(ce_1, ce_{int}) \cdot SimCE(ce_{int}, ce_2)$, iff ce_1 and ce_2 refer to concepts that are connected by both *composedOf*- and *generalizationOf*-relations. *SimCE* is calculated according to the corresponding formulas until their intersection (the element ce_{int}) in the hierarchy tree and then the results are multiplied.

$SimCE(ce_1, ce_2) = 0$, otherwise.

To better understand the *SimCE* metric, we explain its application based on the following example (see Figure 1).

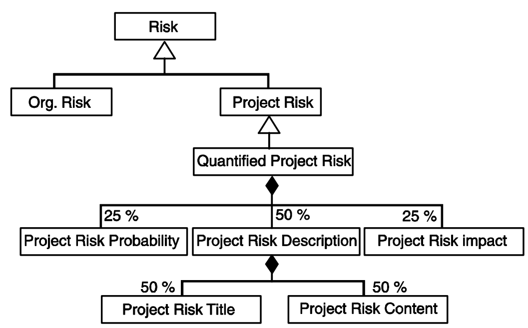

Figure 1. Example of CE structures

$$SimCE(OrgRisks, QuantPrjRisks) = \frac{2\, depth\,(Risk)}{depth(OrgRisk) + depth(QuantPrjRisk)} = 0.4$$

$$SimCE(QuantPrjRisks, PrjRiskCont) = 0.5 \cdot 0.5 = 0.25$$

$$SimCE(OrgRisk, PrjRiskProb) = SimCE(OrgRisk, PrjRisk) \cdot SimCE(PrjRisk, PrjRiskProb)\ \frac{2}{4} \cdot 0.25 = 0.125$$

On the **AU-level**, the similarity of two AUs au_1 and au_2 is calculated based on the *SimCE*-values for all type-equal pairs of contained concepts (calculated on concept level). This is done by applying a variant of the Weighted Euclidian Distance. We consider the difference between two elements in the Euclidean Distance as the average between the *SimCE*-values of all CEs of the same type.

Let *Type = {act, out, inp, role purpose}* be the set of all CE types. The algorithm performs the following steps:

For each ce_i of au_1 and au_2 the pair (ce_i, x) with the highest *SimCE*-value (best pair) is determined.

For each t in *Type* the *SimCE* average value of the best pairs is calculated (AVG_t). There is one exception: if an au_1 contains a concept *ce* and au_2 contains all part-concepts of *ce* the similarity should 1. Therefore, we consider all pairs (whole-concept, part-concept) as a united pair and take the *SimCE* sum value instead of the *SimCE* average value of all these pairs.

The number of different CE types occurring in au_1 and au_2 are determined. For each t in *Type,* a type weight is calculated. According to assumptions (A1, A2, A3) we define type importance constants as follows: $IMP_{act} = 4$; $IMP_{out} = 3$; $IMP_{inp} = IMP_{role} = IMP_{purpose} = 1$. If one type is not present, its IMP-value is 0. Furthermore, as the number of occurring CE types should not influence the *SimAU*-value (R3), we calculate the weight for each t in *Type* dynamically as follows:

$$WEIGHT_t = \frac{IMP_t}{IMP_{act}+IMP_{out}+IMP_{inp}+IMP_{role}+IMP_{purpose}}$$

The similarity value of the AUs au_1 and au_2 is calculated:

$$SimAU(au_1, au_2) = \sum_{t \in Type} WEIGHT_t \cdot AVG_t$$

As procedures of IRMs are described differently (one can contain only one AU, the other several AUs), the aggregation of all *SimAU*-values could lead to low similarity values although the compared procedures contain very similar AUs.

Therefore, on the **PROC-level**, we define a simple categorization metric which maps two compared procedures to one of the following three similarity categories: *equal*, *similar*, *different*. They are defined as follows:

SimPROC(proc₁, proc₂) → equal, **iff** for each AU au_{proc1} of $proc_1$ there is an AU au_{proc2} of $proc_2$ with $SimAU(au_{proc1}, au_{proc1}) = 1$ and vice versa (for each AU au_{proc2} of $proc_2$ there is an AU au_{proc1} of $proc_1$ with $SimAU(au_{proc1}, au_{proc2}) = 1$). This means that the procedures are equal if all best AU pairs are also equal; **iff** each AU au_{proc1} of $proc_1$ is composed of several AUs of $proc_2$ and no other AU is a part of au_{proc1} and vice versa. An AU au_{proc1} is composed of another AU au_{proc2} if for all type-equal pairs of contained concepts (ce_1, ce_2) $SimCE(ce_1, ce_2) = 1$ or ce_1 *is composedOf* ce_2. To summarize, the procedures are equal if one contains the whole-AU and the other all its part-AU.

SimPROC(proc₁, proc₂)=similar, iff there is at least one AU pair (au_{proc1}, au_{proc2}), with $SimCE(au_{proc1}, au_{proc2}) >= 0$.

SimPROC(proc₁, proc₂) → different, otherwise.

These categories give information about the similarity of two procedures. Although these categories are not very precise, it still gives relevant information to organization that need to adopt multiple IRMs. Procedures that are *equal* or *similar* need to be further analyzed to discover their commonalities and differences. Procedures that are *different* do not need to be considered.

C. Examples

In the following we explain how the algorithm and the proposed similarity metrics are applied. As an example we consider the following procedures both having only one AU: $Proc_1$ (**COBIT 4.1, PO10.8.2**): *Staff the roles based on available skills information* and $Proc_2$ (**CMMI-Dev, PP, SP 2.6**): *Plan the involvement of identified stakeholders.*

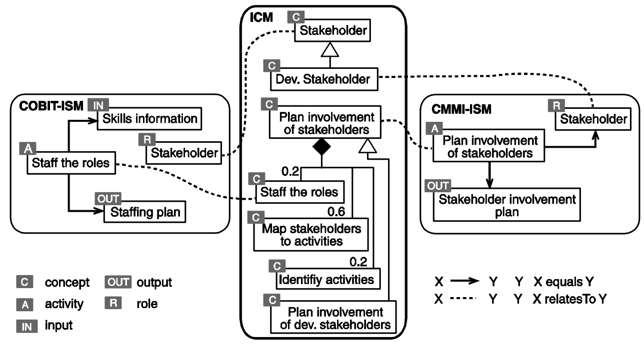

Figure 2. MoSaIC ISM and ICM models representing two CMMI and COBIT procedures

The activity "*Plan involvement of stakeholders*" is *composedOf* the activity "*Staff the roles*" with weight 0.2 in the ICM. As COBIT refers to all IT stakeholders while CMMI only refers to development stakeholder, the concepts of $proc_1$ and $proc_2$ are connected by the g*eneralizationOf*-relation. Furthermore, the activity "*Plan involvement of stakeholders"* is *composedOf* the activities "*Map stakeholders to activities*" and "*Identify activities*" with weights 0.6 resp. 0.2. The relations between the outputs are defined analogously. Obviously, both procedures contain only one AU.

CE-level: All possible CE pairs of the same type are generated and their similarity values are calculated.

TABLE III. COMPARISON CONCEPT LEVEL

Type	Content	*SimCE*
act	(staff prj. roles, plan involv. of prj. dev. stakeholders)	0.13
out	(staffing plan, prj. dev. stakeholder involv. plan)	0.13
inp	(prj. skills information, *null*)	0
role	(prj. stakeholder, prj. dev. stakeholder)	0.67

As there is exactly one pair for each CE type, it is also the best pair and averages are not needed.

AU-level: First, the weight value for each CE type is computed: $WEIGHT_{out} = 1/3$; $WEIGHT_{inp} = 1/9$; $WEIGHT_{role} = 1/9$; $WEIGHT_{oct} = 4/9$. The similarity value *SimAU* of the only AU pair is the following:

$$SimAU(au_{proc1}, au_{proc2})\ \frac{4}{9} \cdot 0.13 + \frac{1}{3} \cdot 0.13 + \frac{1}{9} \cdot 0.67 + \frac{1}{9} \cdot 0 = 0.17$$

PROC-level: As the two procedures contain only one AU, the value of *SimPROC* depends on the value of *SimAU*(au_{proc1}, au_{proc2} then the *SimPROC*($proc_1$, $proc_2$) =*similar.*

The similarity between the considered AUs is low due to the different contexts (IT and development) and to the low weighted whole-part relation between the concepts. The commonalities and the differences can be easily identified. Both AUs consider "*stakeholder involvement*" but in different contexts. One AU defines only one aspect of the other AU.

Furthermore, the essence of these two AUs can be easily extracted by identifying the abstract concepts: the activity *"plan the involvement of project stakeholder*" receives as an input the "*project skills information*", involves the role "*project stakeholder*" and produces the output "*project stakeholder involvement plan*". Therefore, the organization can adopt this abstract AU to be conformant to both IRMs. Another usage scenario would be the assessment of this abstract AU only once to verify its conformance.

The identification of similar AUs allows an organization to benefit from the synergies between the IRMs. The IRMs do not define only procedures but also other additional information related to this procedure (e.g. detailed description of the procedure, sub-procedures). For example, COBIT adopters who want to consider the "*stakeholder involvement*" can learn from the additional information given in CMMI (e.g. that the "*stakeholder involvement plan*" should also contain the "*rationale for stakeholder involvement*").

Another representative example is the comparison of the procedures: $Proc_1$ (**COBIT 4.1, PO9.2.1**): *Evaluate risks qualitatively according to their impact (catastrophic, critical, marginal), probability (very likely, probable, improbable) and time frame (imminent, near term, far term)* and $Proc_2$ (**CMMI-Dev, RSKM, SP 2.2**): *Evaluate each identified risk using defined risk categories and parameters, and determine its relative priority*. As the contexts and the details are different ($proc_1$ considers "*organizational risks*", $proc_1$ "*project risks*" resp. $proc_1$ gives more details about risk evaluation than $proc_1$), the computed similarity is medium. However, both procedures consider "*risk evaluation*" and CMMI adopters can learn from COBIT and vice versa (e.g. risk impact or probability categories).

The final example compares two procedures that are equal (and hence shows the need of the similarity category): $Proc_1$ (**SPICE SPL2.BP13**): *The product is delivered to the intended customer with positive confirmation of receipt* and $Proc_2$ (**CMMI-Dev, PI SP3.4.5**): *Deliver the product (..) and confirm receipt.* These procedures are equal as the AU of $proc_1$ is composed of the two AUs of $proc_2$. This is important information for the organizations that adopt CMMI and SPICE. The comparison on the AU-level does not provide this kind of information (further examples in [27]).

The identification of similar concepts (*SimCE*) is not the sole basis for computing the similarity between procedures but also allows identifying dependent procedures in and over the borders of an IRM. For instance, consider these procedures: $Proc_1$ (**SPICE, SPL.1.BP8**): *Formally confirm the agreement to protect the interests of customer and supplier* and $Proc_2$ (**CMMI-Dev, SAM SP 1.3.3**): *Document supplier agreement.* They share the artifact "*supplier document*" (*SimCE = 1*). $Proc_1$ depends on proc2, as the artifact must be first created and then be shared with the customer and be confirmed.

V. EVALUATION

In the following we present the evaluation results of our comparison approach. First, we evaluate the proposed similarity metrics by applying the metrics to procedures defined by CMMI, COBIT and SPICE. Secondly, we present the evaluation results of the comparison on the procedure level.

To evaluate the proposed metrics we validated the defined requirements (R1-R3). R3 requires that the number of conceptual elements of an AU should not influence its similarity value. This is achieved, as in our metric the weight for each type is dynamically calculated (see formula (8)). Furthermore, the weight for each conceptual element type is defined according to its importance (R2).

As our metrics are based on the procedure elements, the results of the comparison are differentiable, comparable and reproducible. R1 also requests that the results are plausible. For this purpose, we performed the following experiment. First, we manually determined similar CMMI/COBIT and CMMI/SPICE procedures ([28] for CMMI/COBIT was used). Second, the ISMs and their corresponding common ICM were created. Third, we computed the similarity values for 76 pairs of AUs (36 in CMMI-COBIT and 40 in CMMI-SPICE) in 36 procedure pairs (18 CMMI-COBIT and 18 CMMI-SPICE). Then, professional experts evaluate the AUs similarity to five categories and we mapped our results to these categories: [1,1] as *identical;* [0.67, 1) as *high;* [0.3, 0.67) as *medium;* (0, 0.3) as *low;* [0,0] as *different.* We defined the threshold between *high* and *medium* as 0.67, because the similarity between two AUs that are equal but occur in different contexts has a similarity value of 0.67. According to the experts, their similarity is high. Finally, we asked professional experts to subjectively evaluate the results on the procedure level.

Three experts participated in the evaluation: one employer of an insurance IT company with over 5 years experience in CMMI; one consultant with over 20 years experience in COBIT, ITIL and CMMI; one consultant with over 15 years experience in CMMI and 5 years in SPICE. We obtained good results by comparing the similarity metric results (SM) on the AU level and the experts' judgments: **0.27 for CMMI-COBIT** (on average less than every third metric result deviates by more than one point from the given category) and **0.4 for CMMI-SPICE** (on average less than every second metric result deviates by more than one point from the given category). Some positive examples of compared AUs showing the SM- and EJ-values are:

TABLE IV. POSITIVE RESULTS OF PROCEDURES COMPARISON

Result	Procedures
SM=0.88 (High) **EJ**=High	***SPICE ENG.2.BP2*** *Analyze the identified system requirements in terms of technical feasibility, risks and testability,* ***CMMI RD SP3.3.3*** *Analyze requirements to ensure that they are complete, feasible, realizable, and verifiable*

***SM*=0.58 (Med) **EJ**=Med**	**SPICE SPL.2.BP8** *The packaging for different types of media is identified,* **CMMI PI SP3.4.2.** *Use effective methods to package the assembled product.*
***SM*=0.09 (Low) **EJ**=Low**	***COBIT PO1.3.3*** *Define the roles of the stakeholders involved in the strategic planning process,* ***CMMI PP SP2.6*** *Plan the involvement of identified stakeholders.*

On the **AU-Level**, there are some small deviations between the metric results and the expert judgments. One reason is that some of the experts weighted the output of an AU as being more important than the activity. Another reason is, that sometimes the activity did not count for the experts (e.g. ***SPICE SPL.2.BP13*** *The product is delivered to the intended customer with positive confirmation of receipt* and ***CMMI PI SP 3.4.5*** *Confirm receipt of the delivered product*.) Here the outputs are semantically equivalent and the experts' judgment value was 1. As the activities are different, our results were smaller. We will especially analyze the relation between AUs and the contained activities in or further evaluations. On the **PROC-level**, there was no deviation between the calculated categories of the compared procedures and the expert judgment. Moreover, the experts consider this information a good starting point in the comparison of procedures and thus, found it valuable and relevant.

VI. CONCLUSION AND FUTURE RESEARCH

In this paper, we presented an approach to compare procedures of different IRMs based on their similarity. To enable the comparison, a normalization of the structure and terminology of IRMs is needed. Based on the MoSaIC meta-models and on similarity methods we defined a notion of similarity and developed an algorithm that uses dedicated similarity metrics. The results obtained so far are promising. An analysis of the results of the compared IRMs' procedures allows organizations to identify the differences between the compared procedures and supports organizations to exploit the synergies between IRMs. By analyzing the similarity relations between the compared concepts, the organization discovers what exactly needs to be implemented to be conformant to both regarded IRMs.

In our future research we intend to develop a dedicated tool support for all steps of the comparison approach to provide a much larger integrated model for the most popular IRMs. Based on further evaluations we want to improve and calibrate the proposed metrics. This will offer organizations a better support to indentify similarities of IRMs in order to avoid redundancies in the adoption of multiple IRMs and their assessment.

REFERENCES

[1] The Standish Group International: CHAOS Summary 2009. Chaos (pp. 1-4). Retrieved from http://www.statelibrary.state.pa.us/portal/server.pt/document/690719/, 2009.

[2] ISO/IEC TR 24774: Systems and software engineering – Life Cycle Management – Guide-lines for process descriptions, 2010.

[3] Jeners S., H. Lichter: Towards an Integration of Multiple Process Improvement Reference Models based on Automated Concept Extraction, D. Winkler, R.V. O'Connor, and R. Messnarz (Eds.): EuroSPI 2012, CCIS 301, pp. 205--216. Springer, Heidelberg (2012).

[4] Ferreira, A.L., Machado, R.J., Paulk, M.C.: Quantitative Analysis of Best Practices Models in the Software Domain. Proc. of the 17th Asia Pacific Software Engineering Conference, 433-442, 2010.

[5] Ferchichi, A., Bigand, M.: An Ontology for Quality Standards Integration in Software Collaborative Projects. Proc. of MDISIS 08 (Model Driven Interoperability for Sustainable Information Systems), Montpellier, FRANCE, 14, 2008.

[6] Liao, L., Qu, Y., Leung, H.K.N.: A Software Process Ontology and Its Application. Proceedings of IWFST-2005 (Int.Workshop on Future Software Technology), Shanghai., 1-10, 2005.

[7] Siviy, J., Kirwan, P., Marino, L., Morley, J.: Process Architecture in a Multimodel Envi-ronment. White Paper, Software Engineering Institute, CarnegieMellon, 2008.

[8] Wang, Y., King, G., Dorling, A., Wickberg, H. (1999): A unified framework for the software engineering process system standards and models, Proceedings 4th IEEE International Software Engineering Standards Symposium and Forum (ISESS'99), Curitiba , Brazil.

[9] Malzahn, D.: Assessing - Learning - Improving, an Integrated Approach for Self Assessment and Process Improvement Systems. Proc. of ICONS '09, the Fourth Inter-national Conference on Systems, Gosier, Guade-loupe, France, 126-130. IEEE. doi: 10.1109/ICONS.2009.31, 2009.

[10] Pardo, C., Pino, J. F., Garcia,.F, Piattini, M., Baldassarre, M.T: An Onotology for the harmonization of multiple standards and models, Computer Standards & Interfaces 34, 2012, 48-59.

[11] Soto, M., Münch, J. (2008):Using Model Comparison to Maintain Model-to-Standard Compliance, Proceedings of the ICSE Workshop "Comparison and Versioning of Software Models" (CVSM 2008), Leipzig, Germany, May 17, 2008.

[12] Shvaiko, P., Euzenat, J.: A survey of Schema-based Matching Approaches, Journal of Data semantics IV:146-171, 2005.

[13] Rahm, E. , Bernstein, P.A.: A survey of approaches to automatic schmea matching, VLDB Journal 10 (4), 334-350, 2001.

[14] WordNet, available at http://wordnet.princeton.edu/.

[15] Veksler, V.D. ; Gray, W.D.: Rensselaer MSR Server.available at http: //cwl-projects.cogsci.rpi.edu/msr/

[16] Wikipedia Miner, available at http://wdm.cs.waikato.ac.nz:8080/.

[17] OntoGen: Semi-automatic Ontology Editor. In Smith, M.J. ; Sal-vendy, G. (editor): Human Interface, Part II, HCII 2007, 309-318, 2007.

[18] Goldstone, R. L., Son, J.: Similarity. In Holyoak, K.J., Morrison R. (editor): Cambridge Handbook of Thinking and Reasoning, pp 13-36. Cambridge University Press, Cambridge, 2005.

[19] Goodman, N.: Problems and projects. Hackett Pub Co Inc, 1979.

[20] Groenen, P. J. F.: Past, Present, and Future of Multidimensional Scaling. http://carme2011. agrocampus-ouest.fr/slides/Groenen.pdf.

[21] Tversky, A., Gati, I.: Studies of similarity. Cognition and categorization, 1: pp 79-98, 1978.

[22] Levenshtein V.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10: 707–710, 1966.

[23] Goldstone, R. L.: Similarity, Interactive Activation, and Mapping. Journal of Experimental Psychology: Learning, Memory, and Cognition, 20(1): 3-28, 1994.

[24] Ganesan, P., Molina, H. G., Widom, J.: Exploiting hierarchical domain structure to compute similarity. ACM Transactions on Information Systems (TOIS), 21(1), 64-93, 2003.

[25] Fenton, N. E.: Software Metrics. A Rigorous Approach. Chapman & Hall, 1991.

[26] Ludewig, J., H. Lichter: Software Engineering – Grundlagen, Menschen, Prozesse, Techniken. 2. Aufl., dpunkt.verlag Heidelberg, 2010.

[27] Jeners, S, Pyatkova, E.: MoSaIC Reference Model Comparison Case Study, Software Construction Group, 2012, www.swc.rwth-aachen.de/docs/MOSAIC/APSEC/ComparisonExample.pdf.

[28] IT Governance Institute: COBIT Mapping: Mapping of CMMI for Development, V1.2, With COBIT 4.1, 2011