Integrating Culture Awareness and Formalisation in Software Process Assessment and Improvement for Very Small Entities (VSEs)

Tatsuya Nonoyama[2(✉)], Edward Kabaale[2], Lian Wen[1,2], David Tuffley[1,2], and Zhe Wang[1,2]

[1] Institute for Integrated and Intelligent Systems, Griffith University, Brisbane, Australia
[2] School of Information Communication Technology, Griffith University, 170 Kessels Road, Brisbane, Qld 4111, Australia
{t.nonoyama, e.kabaale, l.wen, d.tuffley, z.wang}@griffith.edu.au

Abstract. Software process assessment and process improvement are fundamentally essential if software companies are to improve their development processes and perform at best practice level. However, established software process assessment models (PAMs) like SPICE or CMMI are scaled to be applicable to Small and Medium Enterprises (SMEs) all the way up to very large companies. So far, there is no mature PAM applicable to Very Small Entities (VSEs), which are usually defined as companies with 25 or less employees. As the majority of software companies are classified as VSEs, the lack of a mature PAM is a glaring omission in the Software Engineering domain in need of rectification. A major challenge with producing a VSE-appropriate PAM is the cultural diversity of VSEs. It is not amenable to a one size fits all solution suitable for all VSEs. Another challenge is the high human resource cost of a SPICE or CMMI process assessment. It is often cost-prohibitive for VSEs. This paper therefore proposes a model to meet the need; the *Culture Impact on Software Assessment* (CISA). CISA facilitates the development of PAMs that are both lightweight, making them easy and cheap to apply, while also being highly relevant to individual VSE, thereby significantly increasing the efficiency of PAMs. Additionally, this paper introduces a formal specification (ontology) of CISA to enable future development of software process semi-automatic assessment tools that will greatly reduce the cost for software process assessment for VSEs.

Keywords: Process assessment model · Process improvement
Ontology · Culture evaluation · Very small enterprise · VSE

X. Larrucea et al. (Eds.): EuroSPI 2018, CCIS 896, pp. 123–135, 2018.
https://doi.org/10.1007/978-3-319-97925-0_10

1 Introduction

Software Engineering (SE) standards are by necessity being adopted in software company of all sizes. ISO/IEC 29110 [1, 2] has been introduced particularly for Very Small Entities (VSEs) with less than 25 employees [3]. It has reduced the number of software processes from more than 50 as defined in ISO/IEC 12207: Systems and Software Engineering-Software Life Cycle Processes down to two most important software processes [4]). However, what is still lacking is a predefined software assessment model for ISO/IEC 29110. Where VSEs are interested in certification and recognition, the requirement for assessors still makes assessment a prohibitively expensive exercise [6]. The development of deployment packages (DP) helps VSEs to understand the details of ISO/IEC 29110 and facilitate the implementation thereof. Boucher [6] argues that process reference models still need to be tailored to the situational background of adopting VSEs.

Among the factors that influenced the design of ISO/IEC 29110 standard for VSEs was that it was supposed to be compliant with ISO/IEC 15504 standard (aka SPICE) that is transiting to ISO/IEC3300xx series to accommodate more domains and other process characteristics other than process capability [7]. This factor led to the development of ISO/IEC 29110 part 3 that provides guidelines for using and tailoring ISO/IEC 15504 to assess software development following the ISO/IEC 29110 standard. However, these guidelines and ISO/IEC 15504, do not address the divergent cultural issues of VSEs while carrying out assessment in VSEs, except the emphasis of the guidelines is put on the number of processes and capability levels selected from the parent standard.

Several studies have proposed process assessment models (PAMs) for a VSE based on ISO/IEC 29110 [8] using the part 3 guidelines to tailor ISO 15504 usage in VSEs; The study [9] proposed a PAM and method for VSEs based on ISO/IEC 29110 using the guidelines in part 3 to tailor ISO/IEC 15504 for use in VSEs. As a result, an exemplar PAM for VSE is suggested. This PAM limits its scope on the process dimension (capability level 1) of ISO/IEC 15504. Their justification is that the current profiles of ISO/IEC 29110 are currently limited to only process dimension of ISO/IEC 15504. In the same line [10] proposed another PAM still tailoring ISO/IEC 15504 using ISO/IEC 29110 part 3 guidelines. However, none of these PAMs consider the cultural situations of a VSE during assessment. According to Jaakkola [11], there are several culture sensitive processes that must be fine-tuned for VSEs to adopt. This is especially true in VSEs where resources are limited.

The well-accepted PAM as defined by ISO/IEC 15504 is too heavy and expensive for VSEs to adopt. VSEs need the PAMs to be cost-efficient and to address their common issues. Furthermore, the lack of cultural acknowledgement may lead to a failure to detect desirable inputs, outputs and work products of the VSE [4]. To acknowledge cultural diversity of VSEs, we need to design cost-efficient PAM that is applicable to evaluate the culture of a VSE.

Therefore, in this paper, we proposed CISA framework that takes into consideration cultural diversity of VSEs while carrying out process assessment. We have critically compared our framework with existing frameworks such as the culture sensitivity

assessment model (CSAM) developed in the STEP project [11]. We have additionally formalised the framework in a form of ontology to lay a first stone towards developing semi-automated process assessment tools. Section 2 introduces the necessary literature background for our study. In Sect. 3, we introduce the proposed CISA framework and its formalisation using ontologies. In Sects. 4 and 5, we provide a brief discussion, conclusion and sketch future directions for our work.

2 Background

2.1 ISO/IEC29110 and ISO/IEC15504

In 2011, ISO/IEC29110: Software Engineering Lifecycle Profiles for Very Small Entities (VSEs) was published to help small software companies (VSEs) with 25 employees or less [9, 12]. It provides suitable software processes, process improvement and assessment guidelines for VSEs. In ISO/IEC 29110, project management process (PM process) and software implementation process (SI process) holds the key to a successful software company. In comparison to ISO/IEC12207, with 50 software processes, a lightweight ISO/IEC29110 has been compressed and selected only two software processes for VSEs.

The current PAM guideline of ISO/IEC 29110 reflects upon the six-point measurement scale system which derived from ISO/IEC 15504. Both standards use the concept of six-point measurement scale to determine process capability for future process improvement. Although, there is no mature PAMs for a VSE, the assessment guideline was published to help them assess software processes. ISO/IEC 29110-3 is reflected on specific assessment indicators from selected process outcomes in ISO/IEC 15504. The objectives of ISO/IEC 29110 software processes are associated with ISO/IEC 12207 process outcomes.

ISO/IEC 15504: Information Technology Process Assessment is the reference model for assessing software processes. The six-point scale consists of 0-incomplete, 1-performed, 2-managed, 3-established, 4-predictable and 5-optimized (ISO/IEC 15504, 2008). Each level indicates the maturity of process improvements with the scale being determined by a set of objectively assessed process attributes. For most VSEs, it is only expected to reach capability level 2 with basic work products and outcomes [7]. Some researchers argue that measurement scale of ISO/IEC 15504 takes a significant amount of financial resources to make an adjustment for VSEs.

For large software companies, formalizing a defined software assessment scheme is financially feasible. However, VSEs do not have the time or financial resources to establish formalised assessment methods. Apart from financial gaps between VSEs and large software companies, their process assessment practices are more influenced by cultural factors than large software companies [4, 13–15] For VSEs, it is vital to avoid major modification on PAMs. The strategic response to this limitation is to design the PAM that fit the software environments of a VSE.

2.2 Culture and Process Assessment Models (PAMs)

In general, culture is a collection of ideas, customs and social behaviours of a particular group of people [16, 17]. Culture is a generic form of how people behave in certain circumstances. The concept of culture can be divided into two broad components such as national and organisational culture. National culture can be defined as a set of norms, behaviours, beliefs and customs that inform the character of a country [18]. The component of national culture is large and there is a sub-component called organisational culture. The organisational culture can be related to values and behaviours that reflects upon the working environment of an organisation [16]. Furthermore, organisational culture can be decomposed into smaller sub-cultures such as working culture which associates with behaviors and attitudes.

The culture impacts on software process is difficult to evaluate especially for VSEs. Due to size differences, many researchers have argued that most culture evaluations are only applicable for a large organization. A PAM that supports the idea of Competing Values Theory (CVT) is a good example of culture evaluation for larger organisations. CVT allows comparing values using multiple assessment questions [21]. Furthermore, the Culture Sensitivity Assessment Model (CSAM) was developed for measuring the sensitivity level of a software company. This model would be the closest PAM with cultural consideration in process assessment. However, it is also intended for large enterprises. The ideal PAM should cover a diverse cultural and technical issues of a VSE [10, 15, 19]. We could scale down CSAM and implement simple but, it may be less effective culture evaluation for VSEs.

2.3 Culture Evaluations by Neeley and Trompenaars

The five work attitudes and behaviors proposed by Tsedal Neeley [20] to enhance cross-cultural collaboration. Based on Neeley's findings, a large Japanese enterprise Rakuten has successfully implemented these work patterns and attitudes. As the result Rakutan's approach to managing cross-cultural issues has proven efficacious. The communication is a vital part of process assessment and can lead into a major issue, if neglected [4]. Neeley's theory is useful to establish the basic procedure of process assessment in different cultural backgrounds.

(1) Embrace positive indifference.
(2) Seek commonality between cultures.
(3) Identify with the global organisation rather than your local office.
(4) Seek interaction with other, geographically distant subsidiaries.
(5) Aspire to a global career.

During an assessment VSEs are likely to encounter the 3 layers of culture. Dutch organisational theorist Fons Trompenaars and his research team acknowledged and improved on Edger Schein's ideas of multi-layered culture. In Trompenaar's perspective, culture is like an onion with different layers [22]. The deeper the layer is, the more difficult it is to be measured. The layer (1) The outside of the onion is how people perceive the surface of the onion. In other words, the artifacts of culture which includes the way we dress and how we conduct ourselves. The layer (2) The second layer

consists of norms and values. Layer (3) The third and deepest layer contains the shared assumptions among the members of that culture.

To apply the 3 layers of culture into SE perspective, a VSE can be divided into two layers of culture and this allows the VSE to evaluate. For example: most VSEs should consider assessing the impacts on national and organisational cultures of software development. This is because, different VSEs are more culturally diverse than large organisations which influenced by national and organisational policies [22–24]. Evaluating different layers of culture can determine the best approach to improve their practice.

2.4 Ontology Support for Culture Aware Software Process Assessment

Due to their powerful knowledge representation formalism and associated inference mechanisms, ontology-based approaches have been increasingly adopted to formally represent software engineering domain knowledge. Ontologies aim at capturing domain knowledge in a formal generalised way and provide a commonly shared understanding of a domain. This domain understanding can be reused, shared and standardised across applications [26]. A formal ontology is a tool independent and explicit representation of a domain in a human and machine understandable way that is amenable to computer inferences.

The Web Ontology Language (OWL) is a W3C standardised language for modeling ontologies on the semantic web. OWL is underpinned by Description Logics (Baada, 2003), a knowledge representation language for its formal semantics and interpretation. OWL comes in different flavours and in this study we consider OWL DL, a flavour that optimises the trade-off between expressivity and decidability. Moreover, OWL is well supported by optimised off the shelf inference engines such as *Pellet* and open source modeling environments like *Protege* from Stanford University. The need to support standards compliant software development with ontologies especially for small software companies has been recently evaluated in (Colomo-Palacios, 2016). In this evaluation, it was established that there is an evident gap in ontology support for the standardisation of software development processes especially for VSEs. While ontologies have been used to support software process standardisation before [25, 26], few studies such as [27] have utilised ISO/IEC 29110 in standards tailoring of software processes to VSEs. Moreover, we note from the preceding discussion that there is currently no publicly available formalised framework that supports culture aware software process assessment for VSEs. This motivates us to provide one that will enable the development of semi-automated process assessment tools for VSEs.

3 Introduction of Culture Impact on Software Assessment (CISA) Framework

In this section, we present the Culture Impact on Software Assessment (CISA) framework visualised in the Fig. 1. The purpose of CISA is to highlight the impact of culture difference on the software process implementation. CISA is developed in accordance with the guidelines prescribed for process descriptions by ISO/IEC TR

24774 where a process is defined through its purpose and outcomes. Process definition through its purpose and outcomes that has been used successfully in process assessment and improvement [28] has its roots in ISO/IEC 15504 which also provides a list of possible practices and work products that are used to prove that the process is able to achieve its purpose through the demonstration of process outcomes. The cultural dimension is derived from [20, 22].

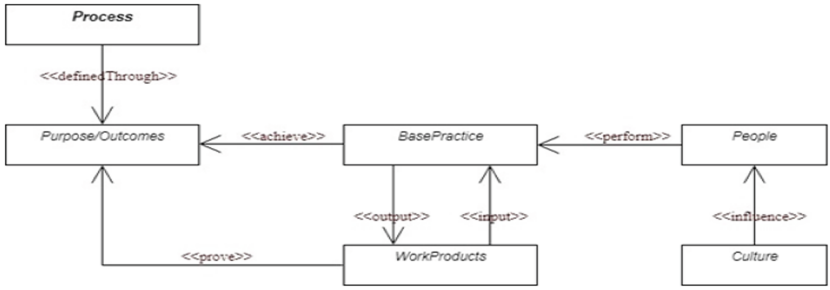

Fig. 1. CISA framework

In CISA framework, a process is defined through its intended purpose and evidenced by the objective achievement of the process outcomes. Process outcomes are sufficient and necessary conditions to achieve the software purpose. In ISO/IEC 29110, process objectives are mapped to process outcomes in ISO/IEC 12207 [29]. To achieve the process outcomes, base and generic practices are performed by roles (people) to produce work products. Work products are also inputs to the practices that are used to achieve the process outcomes. Work products are used to prove that the process outcomes are being achieved.

For example, in software requirements analysis process, the software requirements specification (SRS) can be used to prove that the analysis process was carried out. People perform practices to achieve the process outcomes, however, people are different, given that they can come from different backgrounds and cultures. Different cultures will influence the way people carry out their activities to meet the process outcomes. While the process outcomes may be the same, different people can use different practices to achieve the same outcomes. The outcomes simply say what the successfully performed process will look like, not how to go about performing it, thus allowing individuals to devise their own optimum methods.

3.1 Examples of How CISA Can Be Applied to Different Culture Types

Figure 2 shows the concept of CISA in a simple way. Suppose One Software Process is implemented by two different teams (Team A & Team B) with two different culture types (A & B as well). For example, let Team A as individualist VSE based in U.S and B as Collectivist VSE based in Japan. These two teams (A and B) have different practices in software requirement analysis process. Alternatively, we can assume the final goal is to secure the software contract but, they are more likely to have different

approaches to reach the same goal. The outcomes of the process may be fixed, but the two teams are likely to achieve the outcomes through different practices and work products. The culture type A and B influencing the teams that carry out practices based on their cultures to produce the work products that prove the process outcomes. While the two different teams can use different practices, they will nonetheless achieve the same process outcomes.

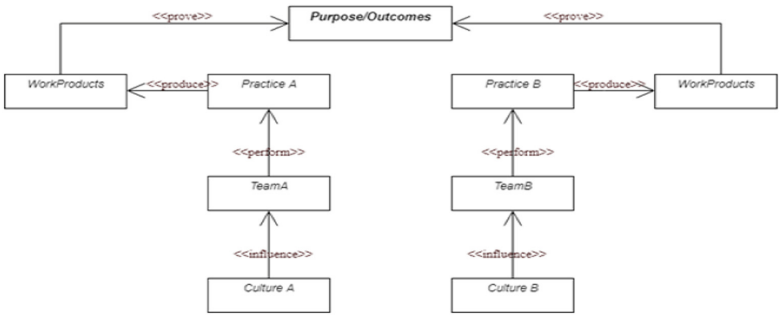

Fig. 2. Practices based on two different culture types

Table 1 illustrates the characteristics of two culture types. We use the characteristics of culture type A and B to produce different sets of practices. The allocation of culture types can help VSEs to examine their characteristics of software process without abandoning traditional ways of producing software. Once the characteristics are defined, a VSE can follow the assessment tips to help achieve the same process outcome. CISA is an appropriate culture framework to assess Software Requirement Analysis in SI process. Many software practitioners and software developers can, on occasion underestimate the culture impacts on PAMs. Therefore; we want to highlight that the process of achieving the outcome (the goal) is different based on the culture of a VSE.

Table 1. Characteristic comparison between individualist and collectivist

Characteristic	Individualist VSE (A)	Collectivist VSE (B)
Adopting new features in software process	Only accepts features with a detailed analysis & facts	Accepts features if the manager wants it
The interaction between an employee and a consumer	When it's necessary	Frequent interaction is acceptable
Software requirement approval	Detailed plan is needed	Less detailed plan but, flexibility is expected
The impact on software process	Work equality is important	Consumer needs
Country	America	Japan

To make assessment easier for VSEs to adopt, we recommend both culture types A and B to use culture evaluation such as Neeley (N) and Trompenaars (T) in their workplace. VSEs from both culture types can use the assessment tips to kick start the culture evaluation. **(1) Assessment Tip for Trompenaars**: provides 3 layers of culture. This theory is important when the VSE wants to examine the culture deeply. It is also useful to analyze and compare other software company's culture from the same country. **(2) Assessment Tip for Neeley:** can help standardize communication channels and networks. This concept is very useful once the VSE have established an ideal organisational culture of assessment process.

3.2 Onto-CISA: An OWL Formalisation of CISA

Even though the developed CISA framework enhances and tailors VSE's software development activities according to their operational culture, it still needs formal enhancement to enable process automation and verification [25, 30]. A formal software process specification is a specification expressed in a language whose vocabulary, syntax and semantics are formally defined and well understood. Software process formalisation supports formal software process definition and assessment, automated analysis, verification and validation, and aiding in choosing the appropriate process for a given project [30]. Formal methods such as OWL ontologies and composition trees (CT) have been used in modeling and formalising software processes before. For example, composition trees have been used to formally model and compare software processes in [19, 31]. Ontology based approaches (Tarhan 2017) have also been proposed to model, validate, constraint and query software process descriptions.

Due to the great diversity and complexity of software processes from different standards and process models coupled with varying situational contexts, in earlier work [30], we developed an axiom based metamodel for process formalisation where key process concepts were grounded in a metamodel. The rationale for a metamodel for process formalisation is to provide a uniform underpinning for the various process concepts during their formalisation. This is in line with the upper ontology developed for ISO software engineering standards [32] to provide a uniform and consistent terminology for all ISO present and future software engineering standards.

A formal defined process serves as a standard by enabling collaboration and cooperation between teams. They also enable process automation, dynamic assembly and tailoring of process elements, formal reasoning and queries about activity specifications and reusable workproducts. It also creates the opportunity to measure the process usage and ensures process compliance [25]. Ontologies are primarily used in standards compliant software development approaches to formally and unambiguously represent the concepts in the domain or as a basis to construct conceptual models of different elements in software organisations [27] such elements include processes, activities, tasks, roles, artefacts and roles. We provide a formalisation of the developed CISA framework based on the translation algorithm and metamodel for software process formalisation developed in our earlier studies [25, 30]. We also make use of concepts and properties in earlier developed ontologies for the SE domain such as the Software Lifecycle Ontology (SLO) and Software Implementation Process Ontology (SIP) developed in the ALIGNED project. Our ontology development follows a similar

approach like the one for CISA framework development. For example, we model CISA classes as ontology concepts, CISA relations as ontology object properties and CISA individuals as the instances in the ontology. For brevity we use DL [33] notation for our CISA framework formalisation in Fig. 1.

$$\text{Process} \sqsubseteq \exists \,\text{defined Through.ProcessPurpose} \tag{1}$$

$$\text{ProcessPurpose} \equiv \text{ProcessOutcomes} \tag{2}$$

$$\text{Practices} \sqsubseteq \exists \,\text{achieve.ProcessOutcomes} \tag{3}$$

$$\text{WorkProducts} \sqsubseteq (\exists \,\text{inputOf} \sqcap \text{outputOf}).\text{Practices} \tag{4}$$

$$\text{WorkProducts} \sqsubseteq \exists \,\text{prove.ProcessOutcomes} \tag{5}$$

$$\text{People} \sqsubseteq \exists \,\text{peform.Practices} \tag{6}$$

$$\text{Culture} \sqsubseteq \exists \,\text{influence.People} \tag{7}$$

We have implemented the CISA framework in Protege. Protege is a free open source ontology building tool from Stanford university and is widely used in ontology building for both academic and commercial purposes. Importantly its embedded with reasoning services provided by FaaCT++ and Hermit reasoners. In Fig. 3, we visualise the formalisation of CISA framework in Protege. OWL-DL facilitates different reasoning services both at the class and instance level. In our example, we have used consistency and instance checking to make sure that all the ontology classes and instances are consistent.

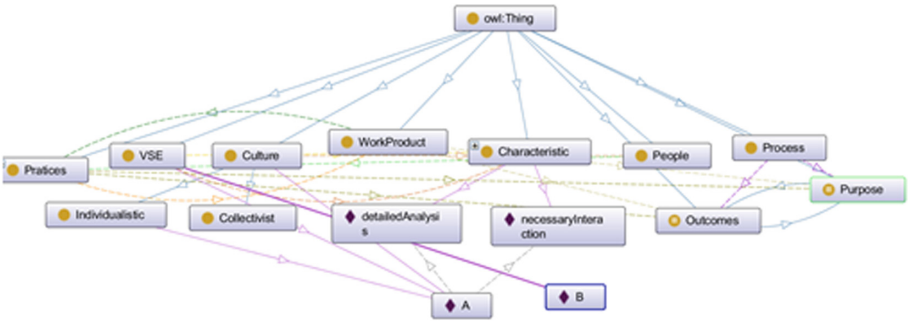

Fig. 3. Excerpt of Onto-CISA in Protégé

On the other hand, our proposed formalisation can help in recommending a set of practices for a given VSE based on a given set of cultural characteristics exhibited by a VSE. For example, if a VSE exhibits characteristic like only accepting new process features with a detailed analysis and facts, interactions between customers and developers can only happen when necessary, software requirements approval needs a

detailed plan, then such a VSE is likely to be an individualistic one and therefore individualistic practices would suit such a VSE. We use SWRL to formalize such rules that enhance the expressivity of the ontology.

$$\text{VSE}\ (?x) \wedge \text{hasCharaterisctics}(?x, ?y) \wedge \text{Characteristics}(?y)\ \rightarrow \wedge\, \text{Individualistic}(?x) \tag{8}$$

This rule enables VSE classification based on cultural characteristics exhibited. This enables the VSE management to align their culture sensitive processes to individualist practices as identified in Table 1 above. Based on this, we used instance checking via the DL query tab to check if culture type individualistic entails any instance. The Query retrieves A (see Fig. 1), an instance of VSE that has the characteristics highlighted above. We hope to expand on the range of reasoning services that the ontology can provide to various VSEs depending on their characteristics.

4 Comparison Between CISA and Other Culture Evaluations

In this paper, we have stressed the importance of culture impacts on the PAMs and formalizing of the developed CISA framework for VSEs. The differences were significant when comparing CISA to other culture based PAMs. Due to cultural diversity of VSEs, culture based PAMs were too complex for VSEs. Based on our evaluation, a single culture evaluation of Hofstede is not enough to cover software environments of a VSE. Biiro et al. [34] proposed a third dimension to CMMI. This model was based on Hofstede's cultural framework where CMMI was extended with a third cultural dimension. Related to the above model, CSAM was proposed in STEP project [11] where organizational processes drawn from ISO/IEC 15504 were integrated with Hofstede's cultural dimension. In this model, each process is assessed to take into account the cultural effects. However, these two models were meant for large organizations without considering the unique features of VSEs. As opposed to the above culture models for software process analysis, CISA has been developed following a lexical analysis [11] of the process descriptions in ISO/IEC 29110 a standard specifically developed for VSEs with appropriate number of processes and work products to fit the unique features of VSEs. To ease the uptake of CISA by VSEs, it has been formalized in an ontology creating a foundation for its automation. While, other PAM supports the idea of CVT to evaluate work values of an organization, using assessment questions [21]. Both CVT and CASM contains a rich amount of information on culture evaluations yet, there is only a limited support on modifications for VSEs. These two models consume time and money which made it difficult for VSEs to consider. We therefore propose a PAM called CISA to support simplified and effective culture assessment for VSEs. At this stage of research, CISA framework is still in prototype stage, our plan is to enhance the content of assessment support and two given cultural concepts (the 3 layers of culture and five attitudes and behaviors).

5 Conclusion

To conclude this paper, VSEs are still facing difficulties in adopting available PAMs like ISO/IEC 15504 that are met for large organizations. While ISO/IEC 29110 provides guidelines for adopting ISO/IEC 15504 in VSEs, these are not yet well utilised in VSEs due to the cultural diversity of VSEs. Based on our research evaluation, the culture impacts on PAMs and a formalized scheme are significantly important for process improvement. To address cultural issues of a VSE, we have proposed a simple culture framework called CISA and a formalized it into an ontology to enable future semi-automated software process assessment tools. We compared our framework against CSAM model an existing model for assessing the level of culture sensitivity in a large organisation. As earlier stated, CSAM is a model for large organisations and therefore not suitable for VSEs. In general, VSEs are much more diverse in their culture as compared to large software companies. The developed CISA is still in an early stage with promising results and provides suggestions for different culture types [35]. As part of our future work, we intend to extend CISA to include other suitable sets of practices for both culture types to accomplish the same software outcomes of ISO/IEC 29110. We also intend to fine tune the formalisation of the CISA and extend it with more rules and perform comprehensive evaluations in different VSE types.

References

1. O'Connor, R.V., Laporte, C.Y.: Deploying lifecycle profiles for very small entities: an early stage industry view. In: O'Connor, R.V., Rout, T., McCaffery, F., Dorling, A. (eds.) SPICE 2011. CCIS, vol. 155, pp. 227–230. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21233-8_27
2. O'Connor, R., Laporte, C.: The evolution of the ISO/IEC 29110 set of standards and guides. Int. J. Inf. Technol. Syst. Approach **10**(1), 1–21 (2017)
3. O'Connor, R.V., Laporte, C.Y.: An innovative approach to the development of an international software process lifecycle standard for very small entities. Int. J. Inf. Technol. Syst. Approach **7**, 1–22 (2014)
4. Nonoyama, T., Wen, L., Rout, T.: Current challenges and proposed software improvement process for vses in developing countries. In: Clarke, Paul M., O'Connor, R.V., Rout, T., Dorling, A. (eds.) SPICE 2016. CCIS, vol. 609, pp. 437–444. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38980-6_33
5. Nonoyama, T., Wen, L., Rout, T., Tuffley, D., O'Connor, R.V.: The Impact of cultural issues on the software process of very small entities. Softw. Qual. Prof. **20**(2), 59–68 (2018)
6. Boucher, Q., Perrouin, G., Deprez, J.-C., Heymans, P.: Towards configurable ISO/IEC 29110-compliant software development processes for very small entities. In: Winkler, D., O'Connor, Rory V., Messnarz, R. (eds.) EuroSPI 2012. CCIS, vol. 301, pp. 169–180. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31199-4_15
7. Rout, T.: The evolving picture of standardization and certification for process assessment. In: Proceedings of the 7th QUATIC, pp. 63–72. IEEE (2010)

8. O'Connor, R.V., Sanders, M.: Lessons from a pilot implementation of ISO/IEC 29110 in a group of very small Irish companies. In: Woronowicz, T., Rout, T., O'Connor, R.V., Dorling, A. (eds.) SPICE 2013. CCIS, vol. 349, pp. 243–246. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38833-0_23
9. Varkoi, T., Makinen, T.: Process assessment in very small entities - An ISO/IEC 29110 based method. In: QUATIC, pp. 436–440 (2010)
10. Ribaud, V., O'Connor, R.V.: Blending process assessment and employees competencies assessment in very small entities. Systems, Software and Services Process Improvement. CCIS, vol. 543, pp. 206–219. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24647-5_17
11. Jaakkola, H.: Culture sensitive aspects in software engineering. In: Düsterhöft, A., Klettke, M., Schewe, K.-D. (eds.) Conceptual Modelling and Its Theoretical Foundations. LNCS, vol. 7260, pp. 291–315. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28279-9_20
12. Laporte, C.Y., O'Connor, R.V.: Software process improvement in graduate software engineering programs. In: Proceedings 1st International Workshop Software Process Education, Training and Professionalism (SPEPT 2015), CEUR Workshop Proceedings, pp. 18–24 (2015)
13. Suryaningrum, D.H.: Knowledge management and performance of small and medium entities in Indonesia. IJIMT **3**(1), 35–41 (2012)
14. Sánchez-Gordón, M.-L., O'Connor, R.V., Colomo-Palacios, R., Sanchez-Gordon, S.: A learning tool for the ISO/IEC 29110 standard: understanding the project management of basic profile. In: Clarke, Paul M., O'Connor, R.V., Rout, T., Dorling, A. (eds.) SPICE 2016. CCIS, vol. 609, pp. 270–283. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38980-6_20
15. Roldan, M.D.: Sustaining "Lilliputs" in the global knowledge-based economy: prospects for micro, small, and medium scale enterprises in the developing world. Eur. J. Sustain. Dev. **4** (2), 269–274 (2015)
16. Abraham, L.R.: Cultural differences in software engineering (2009)
17. Yilmaz, M., O'Connor, R.V., Colomo-Palacios, R., Clarke, P.: An examination of personality traits and how they impact on software development teams. Inf. Softw. Technol. **86**, 101–122 (2017)
18. Norbury, P.: A Traveller's Guide to Custom and Culture, 1st edn. Graphic Arts Center, London (2003)
19. Wen, L., Rout, T.: Using composition trees to validate an entry profile of software engineering lifecycle profiles for very small entities (VSEs). In: Mas, A., Mesquida, A., Rout, T., O'Connor, R.V., Dorling, A. (eds.) SPICE 2012. CCIS, vol. 290, pp. 38–50. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30439-2_4
20. Neeley, T.: How to successfully work across countries, languages, and cultures. Harvard Business Magazine, pp. 1–3 (2017)
21. Muller, S.D., Kraemmeergaard, A., Mathiassen, L.: Managing cultural variation in software process improvement: a comparison of methods for subculture assessment. IEEE Trans. Eng. Manag. **56**(4), 584–599 (2009)
22. Trompenaars, F., Hampden-Turner, C.: Riding the Waves of Culture. McGraw-Hill, New York (1998)
23. Hofstede, G.: Motivation, leadership, and organization: do American theories apply abroad? Organ. Dyn. **9**(1), 42–63 (1980)

24. Nonoyama, T., Wen, L., Rout, T., Tuffley, D.: Cultural issues and impacts of software process in very small entities (VSEs). In: Mas, A., Mesquida, A., O'Connor, R.V., Rout, T., Dorling, A. (eds.) SPICE 2017. CCIS, vol. 770, pp. 70–81. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67383-7_6
25. Kabaale, E., Wen, L., Wang, Z., Rout, T.: Representing software process in description logics: an ontology approach for software process reasoning and verification. In: Clarke, Paul M., O'Connor, R.V., Rout, T., Dorling, A. (eds.) SPICE 2016. CCIS, vol. 609, pp. 362–376. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38980-6_26
26. Tarhan, A., Giray, G.: On the use of ontologies in software process assessment: a systematic literature review. In: EASE (2017)
27. Eito-Brun, R.: Ontology-based tailoring of software process models. In: Terminology and Knowledge Engineering (2014)
28. Rout, T., et al.: SPICE in retrospect: developing a standard for process assessment. J. Syst. Softw. **80**(9), 1483–1493 (2007)
29. Marks, G., O'Connor, R.V., Yilmaz, M., Clarke, P.M.: An ISO/IEC 12207 perspective on software development process adaptation. Softw. Qual. Prof. **20**(2), 48–58 (2018)
30. Kabaale, E., Wen, L., Wang, Z., Rout, T.: An axiom based metamodel for software process formalisation: an ontology approach. In: Mas, A., Mesquida, A., O'Connor, R.V., Rout, T., Dorling, A. (eds.) SPICE 2017. CCIS, vol. 770, pp. 226–240. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67383-7_17
31. Wen, L., Tuffley, D., Rout, T.: Using composition trees to model and compare software process. In: O'Connor, R.V., Rout, T., McCaffery, F., Dorling, A. (eds.) SPICE 2011. CCIS, vol. 155, pp. 1–15. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21233-8_1
32. Gonzalez-Perez, C., Henderson-Sellers, B., McBride, T., Low, G.C., Larrucea, X.: An ontology for ISO software engineering standards: 2) Proof of concept and application. Comput. Stand. Interfaces (2016)
33. Baader, F., Horrocks, I., Sattler, U.: Description logics. In: Staab, S., Studer, R. (eds.) The Handbook on Ontologies in Information Systems. Springer, Berlin (2003). https://doi.org/10.1007/978-3-540-24750-0_1
34. Biro, M., Messnarz, R., Davison, A.G.: The impact of national cultural factors on the effectiveness of process improvement methods: the third dimension (2002)
35. Larrucea, X., O'Connor, R.V., Colomo-Palacios, R., Laporte, C.Y.: Software process improvement in very small organizations. IEEE Softw. **33**, 85–89 (2016)