A Case Study in the Banking Sector: An Ontology for the Selection of Agile and Lean Software Development Methodologies

Itza Morales, Belén Bonilla-Morales, and Miguel Vargas-Lombardo(✉)

FISC, Research Group GISES, Technological University of Panama, Panama City, Panama
{itza.morales1,belen.bonilla,miguel.vargas}@utp.ac.pa

Abstract. Nowadays, the application of methodologies that allow to guide the process of development of Software in the companies has become a fundamental aspect to achieve the suitable management of the processes in the projects. In view of the diversity of existing methodologies, there is a growing interest in defining strategies that allow the selection and application of the correct methodology, which adjusts to the characteristics of the work teams and the software projects. The aim of this work is to develop an ontology for the selection of the methodology that, according to its principles, is most appropriate and beneficial for the development of software projects. The domain of ontology is limited to the Agile and Lean approaches, without defining for Agile the specific method that it applies, but it involves any method that is governed by the agile values and principles established in the Agile manifesto. Onto-logy is applied in two organizations in the banking sector, allowing recommendations to be inferred for the use of Agile methodology in both, which will make it possible to reduce the delivery time of software products, improve communication between project participants, and facilitate the engineering of requirements. On the other hand, the ontology suggests co-regulating characteristic aspects of the Lean practices in order to minimize costs, optimize processes in the software projects and improve the organizational culture.

Keywords: Ontology · Agile software development · Lean software development · Methodologies · Risks

1 Introduction

In the field of software engineering, a growing advance has been perceived in research on methodologies and good practices of software development, which allows the efficient management and optimization of processes and resources, reduces conflicts in the assignment of routines and knowledge of the complexity of the projects [1, 2]. In this context, software development methodologies such as Agile and Lean are implemented depending on the characteristics of an organization. In fact, the authors in [3–5] discuss that the Agile methodology emerged in 2001 as an "Agile Manifesto" focused on reducing the concern about the inefficiency of iterative cycles, customer feedback, workflow visualization, which led to the evaluation of new practices to establish principles for the correct manufacturing of software products.

However, Agile was aimed at small working groups, so a new “Lean” concept was introduced. It was firstly aimed at the industrial sector and then adopted, and parameters were established to enable organizations to scale up in the software product market focusing on three bases: design thinking, Lean production, and Agile development; This reduces the waste of unnecessary processes and increases the organizational culture within the team [19, 21].

Indeed, the Agile and Lean methods have made it possible to examine aspects and factors involved in the analysis prior to the selection of methodologies. To support this approach, ontologies are constructed. They include a common and unambiguous vocabulary for a given domain. In the context of software development, their use provides organizations with an adjustment to the extent of validating their implementation because they are generally selected and adopted incorrectly due to ignorance of their principles, implying delay in the management of processes in software projects. In this sense, studies [18, 36] they emphasize that there are different ontologies focused on evaluating the Software development cycle, the processes or the communication between those involved, for the development of Agile Software and for Software development in general. Likewise, recent studies include ontologies for Agile software development methodology, but not for Lean and/or the link between the two in the Software life cycle processes and within the work team.

Therefore, the need arises for an ontology aimed at the selection of Agile or Lean Software development methodologies in the banking sector, which responds to the analysis of aspects of these two approaches. This research considers the characteristics and requirements of the methodologies for the construction of an ontology with the information supplied from the two banking organizations in Panama. In this way, the study provides organizations with a recommendation on the type of methodology to be selected for their software department and indicates the risks or points to be improved.

The general structure of this study includes: Sect. 2 presents the background that covers the Agile and/or Lean Software development methodologies; Sect. 3 describes the proposed Ontology (development and construction); Sect. 4 presents results; Sect. 5 the discussion and Sect. 6 the conclusions and future work.

2 Background

In organizations, it is essential to consolidate a stable organizational environment, and effectiveness in the delivery of the Software product. In this context, the studies related to Agile/Lean software development methodologies in organizations include characteristic aspects, advantages and risks present during the software product development cycle. Likewise, the approach of Agile and Lean are based on continuous integration and design focused on user feedback [8, 11, 29]. Indeed, Agile and Lean are related to the variables of requirements, productivity, process, and communication between team members. In addition, the authors at [2, 5, 15, 24, 26, 29, 33, 39] mention that these parameters explain that there must be organizational culture to reduce inventory operating costs, and obtain a return on investment from the use of resources, contributing to the flow of quality, and maximizing customer value. Therefore, from the IEEE 1074–1995 standard to achieve the progress of a project, the monitoring, and control of the processes must be reviewed and measured; in addition, this standard assesses the background and risk analysis, contingency planning, project management, and the record-keeping [17]. In turn, Table 1 presents in detail the aspects of the Agile and Lean approaches that are involved in the software development process. It shows the main terms, and, according to both approaches, a description is provided.

Table 1. Descriptive aspects of the Agile and Lean approaches to software development

Terms	Description	References
Risks (Weak points or failures)		
- *Agile*	-Ignorance of principles, difference in communication (planning, development, and collaboration), and there is no operational difference in the performance of the team -Requirement requests, assignment and distribution of responsibilities	[6, 10, 11, 39]
- *Lean*	-Risk management involves the organizational disposition of the client and the team, by implementing a new way of working	[19, 25, 35]
Characteristics		
- *Agile*	- Evaluate the iterations between individuals, less time in documentation, collaboration with clients, and adaptability to changes - Set requirements and resource allocation	[4, 6, 12, 20, 28]
	-Number of people less than 7 per team -Flexibility, task distribution, functionality, execution quality, process evaluation and remote communication	[3, 5, 7, 9, 15, 16, 24, 26, 27, 29, 32, 40]

(*continued*)

Table 1. (*continued*)

Terms	Description	References
- *Lean*	-Making the assumptions before the requirements -Analytical and usability reports, information on previous attempts, stakeholder analysis, and competitive analysis -Use communication strategies (roles): use of a distributed design framework maintaining multiple prototypes in parallel, adaptability to media fidelity faced with the need for communication and employ agile use cases to line up the team -Value with a robust, flexible, and iterative approach in collaboration with Agile end users -Uses a feedback loop known as "build-measure-lean" that seeks to minimize the risk of the project and uses the client's comments -Tools for the automation of process flows and more people (8–15) per teams	[2, 11, 19, 21, 25, 29, 35]
Advantage		
- *Agile*	- Reduces collaboration and/or culture at a distance and temporary distance **-**Offers clients project progress as soon as possible and quality software	[5, 35]

(*continued*)

Table 1. (*continued*)

Terms	Description	References
- *Lean*	-Allows the identification and elimination of waste of those unnecessary processes, facilitates an enhancement in decision -Making in the work team to simplify development time and increase learning	[31, 39]

The information presented in Table 1 describes the key elements for both approaches (Agile and Lean) and offers an overview for the construction and development of ontology.

2.1 Ontological Language (OWL)

The ontological language [22] processed for years has been transferred to the evolution of applications in the areas of software engineering. Therefore, the ontologies developed in [7, 16, 32, 40] are implemented in the sub-parts of the DS in general, which include: collaboration, workflow, process evaluation, cooperative design, and remediation. In agreement with [22, 23], the ontologies developed to ensure the traceability of the requirements in the DSA; Whereas, in [38] consider the types of requirements and their options for evaluating the time and cost of entering the data on traceability, the difference between traceability, the existence of points of view of practical benefits, problems in the organization and support of trade trusts. On the other hand, the ontology for requirements allows giving an intelligence support guide of the techniques to be used and an evaluation of the quality metrics of the traceability requirements. In essence, for [14, 36] the ontologies developed in the DSA, are used as a medium to identify the changes between external clients and the SD equipment; allow permission to improve fundamental communication in the organization and the capacity to meet the requirements Meanwhile, the [1, 21, 34, 37] risk management is focused and these were classified by levels. In this context, according to [17] the IEEE 1074 standard provides that the types of risks involved in organizations can be processed or activities, where each process derives an activity in the development of the Software.

3 Material and Methods: Proposed Ontology

This section presents the construction of an ontology focused on the selection of software development methodologies (SDM) through language-semantic web axis (OWL). This facilitates the decision-making from the analysis of the domain knowledge and gives an answer to the problem [13, 30]. The synthesis of previous studies provides support for the construction of the ontology.

This allows you to use a specific vocabulary and domain to analyze the information from two organizations in the banking sector in Panama. It also provides you with a recommendation on the most appropriate one and draws conclusions on those aspects that need to be improved.

In this manner, the seven steps that the ontology to be built must include are presented. Figure 1 shows the classification of these in two stages: the first includes the description and definition of the domain and the second the construction and results of the ontology; shown in the following diagram:

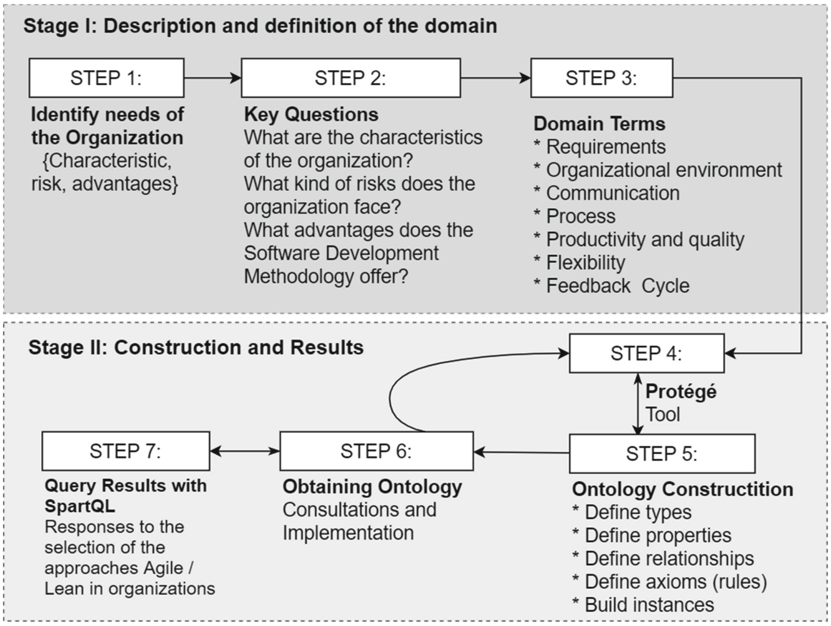

Stage I: Description and definition of the domain
STEP 1:
Identify needs of the Organization {Characteristic, risk, advantages}
STEP 2:
Key Questions
What are the characteristics of the organization?
What kind of risks does the organization face?
What advantages does the Software Development Methodology offer?
STEP 3:
Domain Terms
* Requirements
* Organizational environment
* Communication
* Process
* Productivity and quality
* Flexibility
* Feedback Cycle
Stage II: Construction and Results
STEP 4:
Protégé Tool
STEP 5:
Ontology Constructition
* Define types
* Define properties
* Define relationships
* Define axioms (rules)
* Build instances
STEP 6:
Obtaining Ontology
Consultations and Implementation
STEP 7:
Query Results with SpartQL
Responses to the selection of the approaches Agile / Lean in organizations

Fig. 1. Model with the steps to build the ontology

3.1 Description and Definition of the Domain

The first stage includes the description and definition of the domain, in which the identification of the organization's needs is presented with respect to the survey applied to the two Panamanian organizations in the banking sector. Likewise, Table 2 shows these 18 questions distributed as follow: questions 4 to 11 evaluate the Agile Software development methodology and from 11 to 18, the Lean approach.

Indeed, the information obtained from the two organizations was classified according to the key questions in Fig. 1; while, from these, the purpose of this study is answered through 5 research questions to classify the information. Thus, Table 3 presents an ID for each research question, discussed later in Sect. 5.

On the other hand, the results obtained from the mapping of terms, set out in Sect. 2, have been classified by the characteristics, risks and advantages of both approaches

Table 2. Survey to find out the existing level or situation of the organization

1. Is an MDS implemented in your department?
2. What kind of MDS is implemented in the Software department?
3. Do you know the benefits of applying SMD Agile or Lean?
4. Do you consider that there is a relationship between customers and the software development team?
5. Is there an organizational culture in the team?
6. You consider that there are differences and lack of communication in the team
7. Do you think that the distribution of roles in the team is adequate?
8. The requirements are established and carried out: Risk analysis and planning
9. Requirements are established by carrying out: Risk analysis and planning: Evaluating the history of project failure
10. Requirements are established by: Evaluating background of project failures: Project records are kept, and requirements are traceable
11. During the development process, monitoring and control of functions is presented
12. Have there been any failures of requests after the requirements have been established?
13. How is the organization of people per team?
14. Do you think it is necessary to automate the process flow?
15. Iterations between the team and the design thinking are fundamental
16. Do you think that there is a lack of organizational disposition and that the organizational culture is traditional?
17. Are unnecessary processes occurring during project development?
18. There is a feedback loop for completed projects

Table 3. Research questions (RQ) of ontology

ID	Research questions
RQ1	What is the scope of the ontology in the selection of Agile and Lean Software development methodologies?
RQ2	What are the most outstanding characteristics of the Agile and Lean methodologies to be inferred from the results of the ontology?
RQ3	What are the risks faced by the two banks?
RQ4	What can be inferred from the recommendation generated by the ontology?
RQ5	How can the risks and failures identified through the ontology be improved in the two banking organizations?

(Agile and Lean). Thus, the main terms to use are the following: requirements, organizational culture, communication, process, productivity and quality, flexibility, and feedback loop.

4 Results: Construction of the Ontology

This section presents the formal definition of the aspects with which the "Protégé" software tool works. Initially, a representative class diagram is defined for the ontology and the relations between them. Therefore, Fig. 2 shows the terms and their relations using the UML (Unified Modeling Language) representation of the ontology domain.

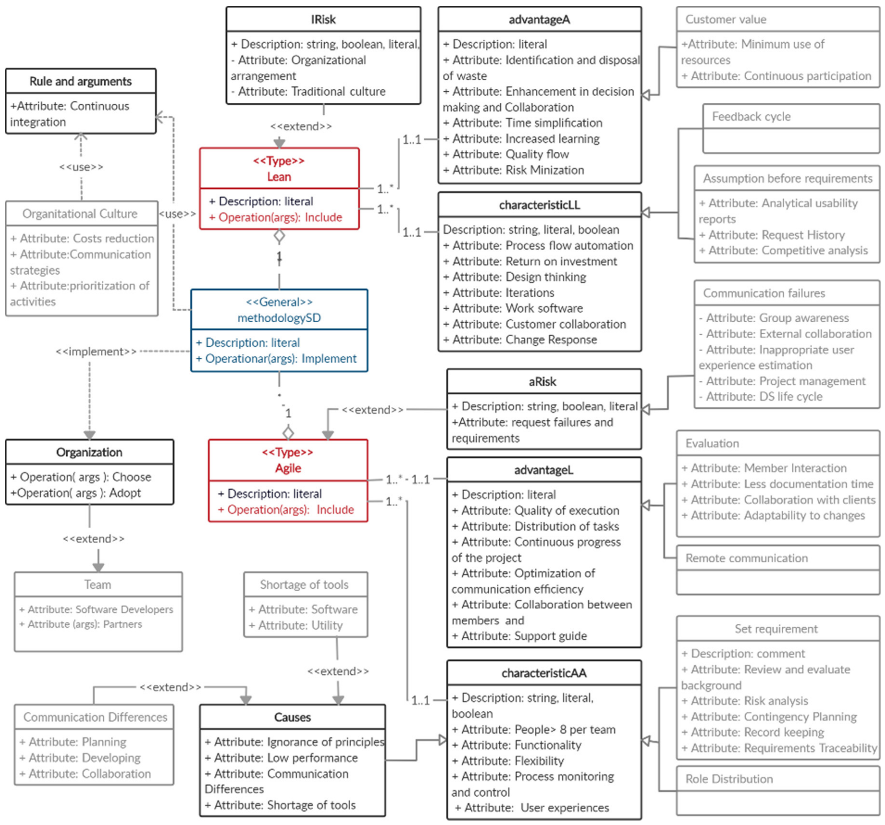
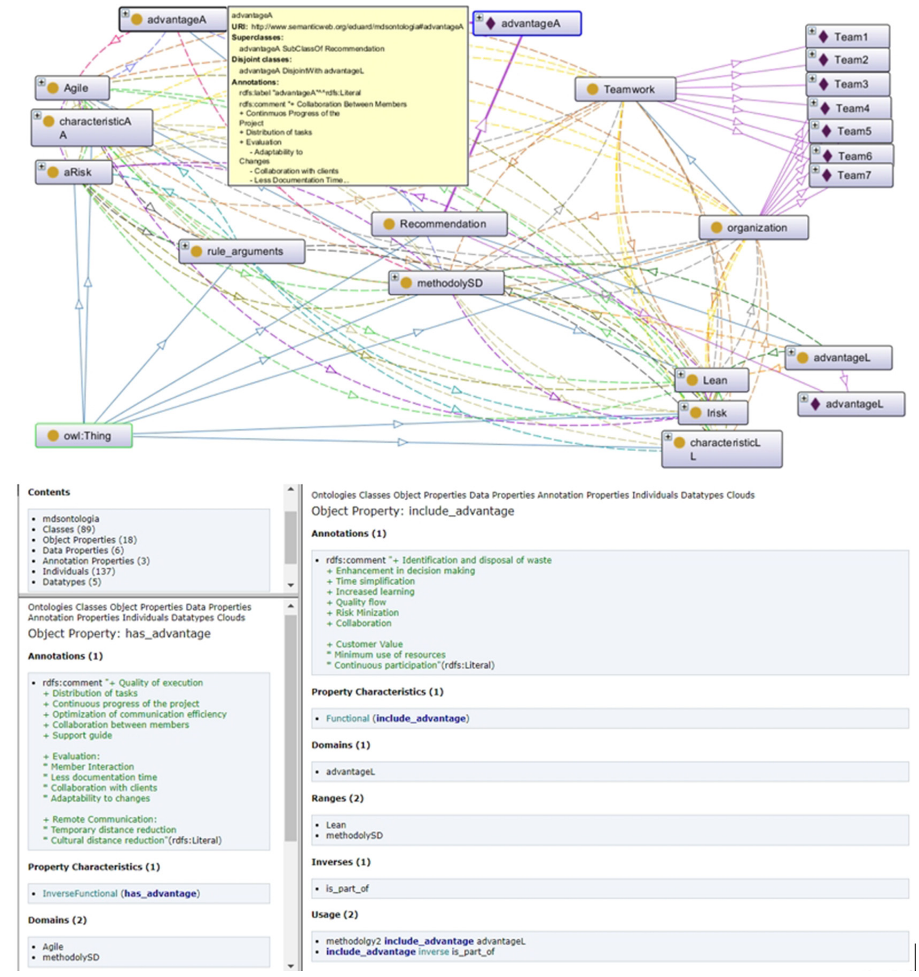

IRisk
+ Description: string, boolean, literal,
- Attribute: Organizational arrangement
- Attribute: Traditional culture
advantageA
+ Description: literal
+ Attribute: Identification and disposal of waste
+ Attribute: Enhancement in decision making and Collaboration
+ Attribute: Time simplification
+ Attribute: Increased learning
+ Attribute: Quality flow
+ Attribute: Risk Minization
Customer value
+Attribute: Minimum use of resources
+ Attribute: Continuous participation
Rule and arguments
+Attribute: Continuous integration
<<extend>>
<<use>>
<<Type>>
Lean
+ Description: literal
+ Operation(args): Include
1..1
1..*
1..*
1..1
Feedback cycle
Assumption before requirements
+ Attribute: Analytical usability reports
+ Attribute: Request History
+ Attribute: Competitive analysis
characteristicLL
Description: string, literal, boolean
+ Attribute: Process flow automation
+ Attribute: Return on investment
+ Attribute: Design thinking
+ Attribute: Iterations
+ Attribute: Work software
+ Attribute: Customer collaboration
+ Attribute: Change Response
Organitational Culture
+ Attribute: Costs reduction
+ Attribute:Communication strategies
+ Attribute:prioritization of activities
<use>>
<<General>>
methodologySD
+ Description: literal
+ Operationar(args): Implement
<<implement>>
Communication failures
- Attribute: Group awareness
- Attribute: External collaboration
- Attribute: Inappropriate user experience estimation
- Attribute: Project management
- Attribute: DS life cycle
aRisk
+ Description: string, boolean, literal
+Attribute: request failures and requirements
<<extend>>
1
Organization
+ Operation(args): Choose
+Operation(args): Adopt
<<Type>>
Agile
+ Description: literal
+ Operation(args): Include
1..* - 1..1
1..*
advantageL
+ Description: literal
+ Attribute: Quality of execution
+ Attribute: Distribution of tasks
+ Attribute: Continuous progress of the project
+ Attribute: Optimization of communication efficiency
+ Attribute: Collaboration between members and
+ Attribute: Support guide
Evaluation
+ Attribute: Member Interaction
+ Attribute: Less documentation time
+ Attribute: Collaboration with clients
+ Attribute: Adaptability to changes
Remote communication
<<extend>>
Team
+ Attribute: Software Developers
+ Attribute (args): Partners
Shortage of tools
+ Attribute: Software
+ Attribute: Utility
<<extend>>
<<extend>>
Set requirement
+ Description: comment
+ Attribute: Review and evaluate background
+ Attribute: Risk analysis
+ Attribute: Contingency Planning
+ Attribute: Record keeping
+ Attribute: Requirements Traceability
Communication Differences
+ Attribute: Planning
+ Attribute: Developing
+ Attribute: Collaboration
Causes
+ Attribute: Ignorance of principles
+ Attribute: Low performance
+ Attribute: Communication Differences
+ Attribute: Shortage of tools
1..1
characteristicAA
+ Description: string, literal, boolean
+ Attribute: People> 8 per team
+ Attribute: Functionality
+ Attribute: Flexibility
+ Attribute: Process monitoring and control
+ Attribute: User experiences
Role Distribution

Fig. 2. UML representation of elements for the ontology

Therefore, for the ontology construction process, there are 8 named classes: methodology, risks and characteristics of Agile and Lean; recommendations, rule arguments and the organization with the subclass (teamwork).

Each of these classes constitutes the domain of the ontology, and its relations, restrictions, Object properties, Data properties and Data Types. In fact, all classes are bound to the base class < owl.Thing >, which contains the knowledge in which the information is analyzed. Thus, Fig. 3 shows the general scheme of the ontology. A short description is presented in a yellow box about the selected class and overview of the content in OWL.



advantageA
URI: http://www.semanticweb.org/eduard/mdsontologia#advantageA
Superclasses:
advantageA SubClassOf Recommendation
Disjoint classes:
advantageA DisjoinWith advantageL
Annotations:
rdfs:label "advantageA"^^rdfs:Literal
rdfs:comment "+ Collaboration Between Members
+ Continmuos Progress of the Project
+ Distribution of tasks
+ Evaluation
- Adaptability to Changes
- Collaboration with clients
- Less Documentation Time...

Contents

- mdsontologia
- Classes (89)
- Object Properties (18)
- Data Properties (6)
- Annotation Properties (3)
- Individuals (137)
- Datatypes (5)

Ontologies Classes Object Properties Data Properties Annotation Properties Individuals Datatypes Clouds

Object Property: has_advantage

Annotations (1)

- rdfs:comment "+ Quality of execution
+ Distribution of tasks
+ Continuous progress of the project
+ Optimization of communication efficiency
+ Collaboration between members
+ Support guide

+ Evaluation:
" Member Interaction
" Less documentation time
" Collaboration with clients
" Adaptability to changes

+ Remote Communication:
" Temporary distance reduction
" Cultural distance reduction"(rdfs:Literal)

Property Characteristics (1)

- InverseFunctional (**has_advantage**)

Domains (2)

- Agile
- methodolySD

Ontologies Classes Object Properties Data Properties Annotation Properties Individuals Datatypes Clouds

Object Property: include_advantage

Annotations (1)

- rdfs:comment "+ Identification and disposal of waste
+ Enhancement in decision making
+ Time simplification
+ Increased learning
+ Quality flow
+ Risk Minization
+ Collaboration

+ Customer Value
" Minimum use of resources
" Continuous participation"(rdfs:Literal)

Property Characteristics (1)

- Functional (**include_advantage**)

Domains (1)

- advantageL

Ranges (2)

- Lean
- methodolySD

Inverses (1)

- is_part_of

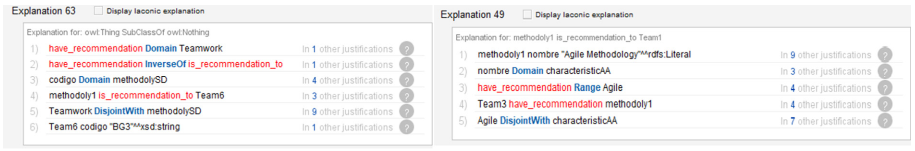
Usage (2)

- methodolgy2 **include_advantage** advantageL
- **include_advantage** inverse is_part_of

Fig. 3. Scheme overview of relationships of the ontology components.

5 Discussion

As an initial result of the survey, it states that 71% of teams use MDS while 29% it does not use them. Consequently, the 42.9% uses the Agile methodology, the 14.3% uses another type of alternative methodology, and the rest do not use any. This reveals that the Lean methodology is not used in the Software departments. On the other hand, the defined ontology analyzes theoretical knowledge through Individuals (organization's information) to generate the results through the verification queries; and using SPARQL Query check that each subject (subclass) would correspond to class (object), through the expression: SELECT? subject-object WHERE {?subject rdfs: subClassOf?object}. For example, the logical reasoner HermiT, which incorporated Protégé, was used to analyze the entered knowledge. The analysis is implemented for 7 banking facilities. This presents in a visual way the inference by each team. Figure 4 presents two of the generated results (Left: organization 1; right: organization 2):



```
Explanation 63    ☐ Display laconic explanation
Explanation for: owl:Thing SubClassOf owl:Nothing
1) have_recommendation Domain Teamwork                          In 1 other justifications
2) have_recommendation InverseOf is_recommendation_to           In 1 other justifications
3) codigo Domain methodolySD                                    In 4 other justifications
4) methodoly1 is_recommendation_to Team6                        In 3 other justifications
5) Teamwork DisjointWith methodolySD                            In 9 other justifications
6) Team6 codigo "BG3"^^xsd:string                               In 1 other justifications
```

```
Explanation 49    ☐ Display laconic explanation
Explanation for: methodoly1 is_recommendation_to Team1
1) methodoly1 nombre "Agile Methodology"^^rdfs:Literal          In 9 other justifications
2) nombre Domain characteristicAA                               In 3 other justifications
3) have_recommendation Range Agile                              In 4 other justifications
4) Team3 have_recommendation methodoly1                         In 4 other justifications
5) Agile DisjointWith characteristicAA                          In 7 other justifications
```

Fig. 4. Results for organization 1 and organization 2

From all the results inferred by the ontology, the information is synthesized in Table 4. This presents the organizations (teams), methodology to be used or maintained, as well as the suggestion of aspects to improve within the Software Development department. For its part, in the items (Name: Recommendation and fundamentals to improve) they are considered the most critical points for each organization; that is, the risks and characteristics whose level of acceptance are subject to immediate changes in the Software Development work process, and need to be improved and properly applied. Table 4 substantiates the information as follows:

The results provide conclusive support to the onto-logy proposal for the selection of the Agile or Lean methodology based on the questions. Thus, this synthesis of the information describes that organizations face challenges and risks where choosing the correct methodology becomes the greatest investment in terms of benefits for the work team and the use of resources. Therefore, the results described in this section allowed inferring in 5 questions to answer the ontology, which are discussed below:

RQ1: The proposal provided a solution window to these organizations through the knowledge inferred through the semantic web language (OWL) to consult and evaluate the profiles of the organizations before implementing a Software development methodology. In fact, the constructed ontology allows to know those weak points of the software development process and to suggest the type of methodology that should be selected. Thus, if it is implemented by the team, to indicate the aspects that represent a risk factor in its development process.

RQ2: In perspective to the scope of the ontology, it is important to highlight that the focus of this study was to initially identify the most outstanding characteristics of Agile and Lean for the two banking organizations, described in Sect. 2:

– The findings allowed them to be categorized into two groups: Agile characteristics: difference in communication, low performance, lack of knowledge of principles, scarcity of tools, functionality, monitoring and control of processes, flexibility and a team of less than 7 people.

Table 4. Recommendations to organizations in the selection of Agile and / or Lean

Name: Recommendation	Organization 1	Organization 2
Agile	**4 Team Work** (Team:1, 2, 6, 7) *Maintain: Team 6,* *Implement: Team 1, 2, 7*	**3 Team Work** (Team: 3, 4, 5) *Maintain: Team 5* *Implement: Team 3, 4*
Fundamentals to improve	**Teams**	
* All Agile principles	Team 2	Team 3, Team 4, Team 5
* Risk analysis and requirements planning	Team 1, Team 2	Team 3, Team 5
* Monitoring and control of functions	Team 1, Team 2, Team 6	Team 4, Team 5
* Background to failures and record keeping requirements	Team 2, Team 6, Team 7	Team 5
* Communication difference	Team 7	Team 4
* Distribution of roles	---	Team 3, Team 4, Team 5
Optional: Applicability of Lean		
* Automate workflows and processes	---	Team 4
* Organizational Culture and Organizational Disposition	Team 1, Team 2, Team 6, Team 7	Team 3
* Feedback cycle	Team 1, Team 6	Team 4, Team 5
* Unnecessary processes	Team 6, Team 7	Team 5

– Characteristics of Lean: Assumption before requirements, response to change, collaboration with the customer, design thinking and the team is more than 7 people.

The analysis of the results supports the importance of these characteristics in the ontology and infers a pattern for both organizations in the seven work teams: the groups are less than 7 people. They lack the applicability of the principles and the organizational culture presents difficulties. Even the establishment of the requirements is one of the key tasks. They also present low attention from the software developers.

RQ3: This raises the question of indicating the risks for the two banks, where the following should be considered for the organization 1: improving risk analysis and requirements planning, failure background and maintenance of the requirements register, communication differences, monitoring and control of functions. While, for organization 2, an additional risk was identified and indicates the distribution of roles.

RQ4: The preceding analysis from the conceptualization of characteristics to the recommendation generated by the ontology shows that both Panamanian banking organizations are suggested to select the Agile Software development methodology, because the characteristics and risks of greater incidence are adjusted both in the number of people per team, as well as, in its principles. This implies that when implementing Agile, it must offer both organizations some advantages, such as:

- Better collaboration and interaction between team members and customers, task distribution, remote communication, improved organizational culture and reduced time distance.
- Continuous project progress and optimization of communication efficiency and provides support guidance.
- Improved evaluation of adaptability to changes and a reduced documentation time.
- In practical terms, the ontology inference used a knowledge domain manager of the team's characteristic aspects in the software development process.

RQ5: The two banking organizations can improve the weak points previously addressed and increase the team's productivity, reducing the process execution time and the practices that can be adopted afterwards. This implies that it helps in the efficiency, better planning of the requirements and fulfilling not only what is requested by the client. Also, through the ontology inference, the organizations are provided with those aspects that are within the Lean context, but that can be improved to automate and minimize risks in the Software development process execution time.

6 Conclusions and Future Work

The ontologies provide a common and unambiguous vocabulary to refer to the terms in the applied area, being able to share or reuse these among different applications that make use of the Ontology. This is the case of software agents (in the field of Web technologies), which can adequately recognize the elements of an ontology as long as the previous conditions are met. As a consequence of this, it is worth mentioning that the ontology in the framework of software development allowed the identification of:

In addition to a common vocabulary, they specify a taxonomy or inheritance of concepts that establish a categorization or classification of the domain entities. A good taxonomy is simple and easy to remember. It separates its entities in a mutually exclusive way and defines groups and subgroups without ambiguity.

The vocabulary and taxonomy represent a Conceptual Framework for the analysis, discussion or consultation of information from a domain.

An ontology includes a complete generalization/specification of its classes and subclasses, which are formally specified (including their relationships and instances) ensuring consistency in the deductive processes.

Ontologies are implemented in specific Ontology representation languages so that the specification of their classes, relations between them and their restrictions will depend on the characteristics of that language.

In consideration, the findings of our research as mentioned in Sect. 4, through the ontology in the domain of software development methodologies, have shown that the Agile approach in the banking sector solves:

It allows each of the banking institutions to identify some aspects that are fundamental during the life cycle of the software product development.

The recognition of the inference of the most outstanding characteristic risks of both organizations, reduces the incidence to failures in the establishment of requirements, communication between team members, efficiency and time of execution of the processes.

It provides team managers with an overview of the current situation in contrast to the benefits offered by the correct implementation of the Agile methodology in their projects.

Likewise, this research differs from the studies presented in Sect. 2.1, in essence, because it has a scope at the level of the selection of the most appropriate methodology, using an OWL, where the knowledge analyzed covers the phase prior to the implementation of a methodology; that is, the ontology, through the knowledge of the case study infers as a main advantage to offer, in the context of software engineering, a previous analysis to improve the failures and provide suggestions for good practices to the organizations. On the other hand, as future work, it is proposed to carry out validity by investigating specific points where a risk assessment (requirements and organizational culture) was reported from the point of view of data analysis and working with other non-banking organizations, and their evolution process at the time of implementing the methodology suggested through this study. In parallel, currently in this line of the Agile Software development methodology, we are working on the analysis of requirements, and the initial phase of the construction of an ontology that includes a general domain language for any organization based on its characteristics, offering a set of suggestions of the type of practice within Agile to be implemented.

7 Authors Contribution.

Conceptualization I.M, B.B, M.V.; methodology I.M, M.V.; formal analysis I.M, B.B, M.V; research, I.M, M.V.; original-writing I.M, M.V.; writing—review and edition I.M, B.B, M.V.; Corresponding author, M.V.

Acknowledgments. We are grateful for the support provided by the Science, Technology and Innovation National Secretariat of Panama (SENACYT), Scientific Master program TIC-UTP-FISC-2019 and to the National Research System (SNI-SENACYT) which one author is member.

References

1. Abdelghany, A.S., et al.: An agile methodology for ontology development. Int. J. Intell. Eng. Syst. **12**(2), 170–181 (2019). https://doi.org/10.22266/IJIES2019.0430.17.
2. Alahyari, H., et al.: A study of value in agile software development organizations. J. Syst. Softw. **125**, 271–288 (2017). https://doi.org/10.1016/j.jss.2016.12.007
3. Alahyari, H., et al.: An exploratory study of waste in software development organizations using agile or lean approaches: a multiple case study at 14 organizations. Inf. Softw. Technol. **105**, 78–94 (2019). https://doi.org/10.1016/j.infsof.2018.08.006
4. Batova, T.: Extended abstract: lean UX and innovation in teaching. In: IEEE International Professional Communication Conference, pp. 1–3, November 2016. https://doi.org/10.1109/IPCC.2016.7740500

5. Cagliano, R., et al.: Lean, agile and traditional supply: how do they impact manufacturing performance? J. Purch. Supply Manage. **10**(4–5) SPEC., 151–164 (2004). https://doi.org/10.1016/j.pursup.2004.11.001.
6. Ching, P.M., Mutuc, J.E.: Evaluating agile and lean software development methods from a system dynamics perspective. In: 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology Communication and Control, Environment and Management HNICEM 2018, pp. 1–6 (2019). https://doi.org/10.1109/HNICEM.2018.8666338.
7. Clarke, P.M., O'Connor, R.V., Rout, T., Dorling, A.: Erratum to: software process improvement and capability determination. In: Clarke, P.M., O'Connor, R.V., Rout, T., Dorling, A. (eds.) SPICE 2016. CCIS, vol. 609, pp. E1–E1. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38980-6_34
8. Curcio, K., et al.: Usability in agile software development: a tertiary study. Comput. Stand. Interfaces **64**(January), 61–77 (2019). https://doi.org/10.1016/j.csi.2018.12.003
9. Dingsoeyr, T., et al.: Agile development at scale: the next frontier. IEEE Softw. **36**(2), 30–38 (2019). https://doi.org/10.1109/MS.2018.2884884
10. Dingsøyr, T., et al.: Coordination in multi-team programmes: an investigation of the group mode in large-scale agile software development. In: Procedia Computer Science, pp. 123–128 Elsevier B.V. (2017). https://doi.org/10.1016/j.procs.2017.11.017.
11. Fagerholm, F., et al.: Performance alignment work : how software developers experience the continuous adaptation of team performance in lean and agile environments. Inf. Softw. Technol. (2015). https://doi.org/10.1016/j.infsof.2015.01.010
12. Fink, L., Pinchovski, B.: It is about time: bias and its mitigation in time-saving decisions in software development projects. Int. J. Proj. Manag. **38**(2), 99–111 (2020). https://doi.org/10.1016/j.ijproman.2020.01.001
13. Gennari, J.H., et al.: The evolution of Protégé: an environment for knowledge-based systems development. Int. J. Hum. Comput. Stud. **58**(1), 89–123 (2003). https://doi.org/10.1016/S1071-5819(02)00127-1
14. Gobin, B.A.: An agile and modular approach for developing ontologies (2013). https://doi.org/10.4018/978-1-4666-4900-2.ch007
15. Hajrizi, E., Bytyci, F.: Agile software development process at financial institution in Kosovo. IFAC-PapersOnLine. **48**(24), 153–156 (2015). https://doi.org/10.1016/j.ifacol.2015.12.074
16. Hsieh, S.H., Lu, M.D.: Collaborative engineering software development: Ontology-based approach. Lecture Notes in Computer Science (including Subser. Lecture Notes in Artificial Intelligence, Lecture Notes in Bioinformatics). LNAI, vol. 4200, pp. 328–342 (2006). https://doi.org/10.1007/11888598_31
17. IEEE: STD 1074–1995: IEEE Standard for Developing Software Life Cycle Processes (1995)
18. Life, S.: IEEE Standard for Developing Software Life Cycle Processes (1995)
19. Liikkanen, L.A., et al.: Lean UX - The next generation of user-centered Agile development? In: Proceedings of the NordiCHI 2014: The 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational, pp. 1095–1100 (2014). https://doi.org/10.1145/2639189.2670285.
20. Liikkanen, L.A., et al.: Lean UX Applying Lean Principles to Improve User Experience (2014).https://doi.org/10.1145/2639189.2670285
21. McArthur, J.J., Bortoluzzi, B.: Lean-Agile FM-BIM: a demonstrated approach. Facilities **36**(13–14), 676–695 (2018). https://doi.org/10.1108/F-04-2017-0045
22. Murtazina, M.S., Avdeenko, T.V.: An ontology-based approach to support for requirements traceability in agile development. Procedia Comput. Sci. **150**, 628–635 (2019). https://doi.org/10.1016/j.procs.2019.02.044

23. Murtazina, M.S., Avdeenko, T.V.: Ontology-based approach to the requirements engineering in agile environment. In: 2018 14th International Scientific Conference on Actual Problems of Electronic Instrument Engineering Proceedings. APEIE 2018, pp. 496–501 (2018). https://doi.org/10.1109/APEIE.2018.8546144
24. Nidagundi, P., Novickis, L.: Introducing lean canvas model adaptation in the scrum software testing. Procedia - Procedia Comput. Sci. **104**, 97–103, December 2016. https://doi.org/10.1016/j.procs.2017.01.078
25. Nudelman, G.: Lean UX communication strategies for success in large organizations. Interactions **25**(5), 80–82 (2018). https://doi.org/10.1145/3236683
26. Nurdiani, I., et al.: The impacts of agile and lean practices on project constraints: A tertiary study. J. Syst. Softw. **119**, 162–183 (2016). https://doi.org/10.1016/j.jss.2016.06.043
27. Paasivaara, M., Lassenius, C.: Communities of practice in a large distributed agile software development organization – case Ericsson. Inf. Softw. Technol. **56**(12), 1556–1577 (2014). https://doi.org/10.1016/j.infsof.2014.06.008
28. Perkusich, M., et al.: Intelligent software engineering in the context of agile software development : a systematic literature review. J. Pre-proof, 1–76 (2019). https://doi.org/10.1016/j.infsof.2019.106241.
29. Rahman, N.A.A. et al.: Lean manufacturing case study with Kanban system implementation. In: Procedia Economics and Finance, ICEBR, vol. 7, pp. 174–180 (2013). https://doi.org/10.1016/s2212-5671(13)00232-3
30. Rao, L., et al.: Building ontology based knowledge maps to assist business process re-engineering. Decis. Support Syst. **52**(3), 577–589 (2012). https://doi.org/10.1016/j.dss.2011.10.014
31. Rodríguez, P., et al.: Combining lean thinking and agile methods for software development a case study of a finnish provider of wireless embedded systems. In: Proceedings of Annual Hawaii International Conference on System Sciences, pp. 4770–4779 (2014). https://doi.org/10.1109/HICSS.2014.586.
32. Roth, W.-M., Jornet, A.: From object-oriented to fluid ontology: a case study of the materiality of design work in agile software development. Comput. Support. Coop. Work (CSCW) **27**(1), 37–75 (2017). https://doi.org/10.1007/s10606-017-9297-6
33. Secchi, R., Camuffo, A.: Lean implementation failures : the role of organizational ambidexterity. Int. J. Prod. Econ. (2019). https://doi.org/10.1016/j.ijpe.2019.01.007
34. Shrivastava, S.V., Rathod, U.: A risk management framework for distributed agile projects. Inf. Softw. Technol. **85**, 1–5 (2017). https://doi.org/10.1016/j.infsof.2016.12.005
35. da Silva, T.S., Silveira, M.S., de O. Melo, C., Parzianello, L.C.: Understanding the UX designer's role within agile teams. In: Marcus, A. (ed.) DUXU 2013. LNCS, vol. 8012, pp. 599–609. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39229-0_64
36. Sitthithanasakul, S., Choosri, N.: Using ontology to enhance requirement engineering in agile software process. SKIM 2016 - 2016 10th Internationa Conference on Software, Knowledge, Information Management Application, pp. 181–186 (2017). https://doi.org/10.1109/SKIMA.2016.7916218.
37. Kruchten, P., Fraser, S., Coallier, F. (eds.): XP 2019. LNBIP, vol. 355. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-19034-7
38. de Souza, P.L., do Prado, A.F., de Souza, W.L., dos Santos Forghieri Pereira, S.M., Pires, L.F.: Improving agile software development with domain ontologies. In: Latifi, S. (ed.) Information Technology – New Generations. AISC, vol. 738, pp. 267–274. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77028-4_37

39. Wang, X., et al.: "Leagile" software development: an experience report analysis of the application of lean approaches in agile software development. J. Syst. Softw. **85**(6), 1287–1299 (2012). https://doi.org/10.1016/j.jss.2012.01.061
40. Wongthongtham, P., et al.: Ontology-based multi-site software development methodology and tools. J. Syst. Archit. **52**(11), 640–653 (2006). https://doi.org/10.1016/j.sysarc.2006.06.008