# UTILIZING A COMPOSITIONAL SYSTEM KNOWLEDGE FRAMEWORK FOR ONTOLOGY EVALUATION

## *A Case Study on BioSTORM*

H. Hlomani, M. G. Gillespie, D. Kotowski and D. A. Stacey
*School of Computer Science, University of Guelph, 50 Stone Road East, Guelph, Canada*

Keywords: Ontologies, Ontology capture, Ontology evaluation, Knowledge engineering, Knowledge identification, BioSTORM, Context, Adaptability, Knowledge base.

Abstract: With the advent of such platforms as Service Oriented Architecture (SOA) and the open source community came the possibility of accessing free software/services. These may be in the form of web services, coded algorithms, legacy systems, etc. Users are able to define workflows through the combination of these software components with the aide of systems known as Ontology Driven Compositional Systems (ODCS). These systems have ontologies as their fundamental components that provide the knowledge bases that provide the rich descriptions of the ODCS components. Since these ontologies underlie ODCS, greater efforts must be spent in the engineering of these artifacts. We have thus proposed a knowledge identification framework that can be used as a guide within ontology engineering methodologies to perform such tasks as ontology capture and evaluation. In this paper we demonstrate the usage of this framework in a case study to evaluate the ontologies defined in the BioSTORM project. We do this by using a checklist (founded on the knowledge identification framework) through which we can evaluate the adaptability of the context of an ontology.

# 1 INTRODUCTION

With the advent of such platforms as Service Oriented Architecture (SOA) and the open source community came the possibility of accessing free software/services. These may be in the form of web services, coded algorithms, legacy systems, etc. While these may be self-contained and providing some useful function, a more complex (combination of one or more) form of these services may provide some added value. With the aide of computers, users would compose (either automatically or semi-automatically) a resultant system by discovering, ranking, selecting and orchestrating previously implemented software to achieve their goal. This technique is referred to as **Compositional Systems**.

Current research has focused on **Ontology Driven Compositional Systems**, a variant of compositional systems that employs a central knowledge base to provide rich descriptions of its components (Arpinar et al., 2005; Cardoso and Sheth, 2005; Crubezy et al., 2005; Gillespie et al., 2010; Hlomani and Stacey, 2009). This knowledge base is made mostly of semantic web technologies referred to as *ontologies*. These are formal representation of knowledge through the definition of concepts within a domain and the relationships between these concepts (Gruber, 1993). Since ontologies underlie ODCS and other semantic web based systems, there has been substantial research on the creation of unified ontologies. This is, however, proving to be a daunting task since each semantic web implementation often has its own modelling perspective (Gillespie et al., 2011; Burstein and Mcdermott, 2005). This difficulty then cascades to knowledge engineering processes such as knowledge identification, ontology capture and ontology evaluation. To handle this shortcoming we proposed (in our previous work (Gillespie et al., 2011)) a generalized knowledge identification framework that could be used within ontology engineering methodologies to capture possible knowledge that could be represented in ontological models for ODCSs.

In this paper, we conduct a case study on BIOSTORM ontologies. We do this by using the knowledge identification framework to evaluate the ontologies paying particular interest to the adaptability of the context of the categories of knowledge within the framework.

Hlomani H., G. Gillespie M., Kotowski D. and A. Stacey D..
UTILIZING A COMPOSITIONAL SYSTEM KNOWLEDGE FRAMEWORK FOR ONTOLOGY EVALUATION - A Case Study on BioSTORM.
DOI: 10.5220/0003631901670175
In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development* (KEOD-2011), pages 167-175
ISBN: 978-989-8425-80-5

# 2 BACKGROUND

## 2.1 Ontology Engineering

From the moment that ontologies have become a practical choice for representing knowledge within software, there has been great effort by researchers and the software community to formalize their creation and development process (Gillespie et al., 2011). From this rose the notion of knowledge engineering and the formalization of the knowledge meta-process.

(Sure et al., 2009) presented a generalized ”Knowledge Meta Process” method for creation, refinement and maintenance of ontologies (i.e. knowledge engineering). This process involves several steps: Feasibility Study, Kick-off, Refinement, Evaluation, and Application/Evolution. The first two steps focus mostly on understanding a set knowledge requirements (i.e, Kick-off) required to be represented in a ontology-driven application, and the last three steps focus on iteratively refining, evaluating, and evolving the knowledge representations for an application. All steps/phases of an ontology development process present their own unique characteristics, however the main focus always addresses the represented knowledge. For the context of our case-study, we have decided to focus on considering ontology evaluation techniques.

## 2.2 Ontology Evaluation

To define ontology evaluation two important concepts should be considered: the role ontologies play within applications (*e.g.* ODCS) and perception. (Brank et al., 2005) view ontologies as the “piece” that shifted the focus of information systems from “data processing” towards “concept processing”. Hence system components are given context through the definition of their semantics. Ontologies are built as conceptualizations of a domain and hence are based on one’s view of the domain. (Brank et al., 2005) argue that it is therefore possible for several ontologies to conceptualize the same domain. Given this, focus must then be given to evaluating ontologies not only for their correctness but their *suitability* as well.

There is no consensus on the “best” or preferred ontology evaluation approach, however, there are several variables that can influence the decision to use a specific methodology. These include: the purpose of the evaluation, where the ontology is to be used and the aspects of the ontology to be evaluated (Brank et al., 2005). Each of these evaluation approaches will be classified as: golden standard comparison, evaluation of the application using the ontology, comparison to source data about a modelled domain, or as human performed assessment of predefined criteria, standard or requirement (Vrandečić, 2009; Brank et al., 2005).

In addition to the above mentioned categorizations of approaches to ontology evaluation, (Brank et al., 2005) proposes a grouping of these approaches based on aspects of the ontology (also known as the level of evaluation (Vrandečić, 2009)). Their argument is that an ontology is a fairly complex structure and hence it would be more practical to evaluate each level separately rather than a holistic approach. These include: a.) Lexical, vocabulary, concept, data. b.) Hierarchy, taxonomy. c.) Other semantic relations. d.) Context, application. e.) Syntactic f.) Structure, architecture, design. With that said, utilizing our knowledge framework presented in Section 2.3, an ontology evaluation process can focus on an aspect that best fits the goal of the developer/evaluator.

## 2.3 Knowledge Identification Framework

We recently presented a knowledge identification framework (Gillespie et al., 2011) in hopes of improving the engineering of ontologies for ODCSs. The framework could act as a complimentary guide during an engineering methodology. Figure 1 illustrates our proposal.

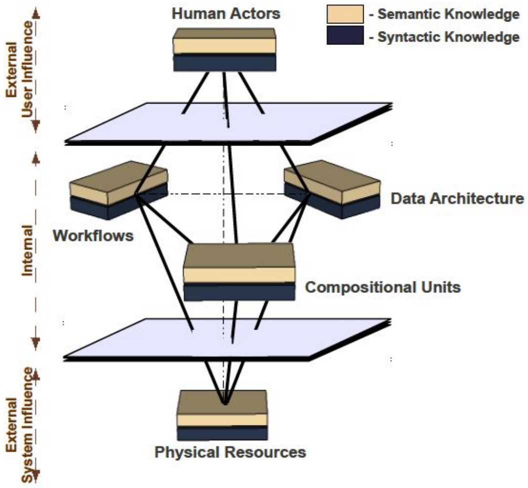

Figure 1: A proposed framework to guide engineers in the identification of ontological knowledge driving ODCS processes (Gillespie et al., 2011).

Within this framework, five different categories of knowledge can be represented within the ontologies that drive an ODCS: Human Actors, Compositional Units, Workflow, Data Architecture, and Physical Resources. With a stronger definitions provided in (Gillespie et al., 2011), the five categories are described in this paper as the following:

**Human Actors.** The representation of knowledge that identifies various types of human users who interact with an ODCS in some fashion (*e.g.* end-users, software developers, domain-experts, etc.)
**Compositional Units.** The representation of knowledge that identifies previously implemented pieces of software that could be composed into a resultant system (*e.g.* algorithms, web services, distributed agents, etc.)
**Workflow.** The representation of knowledge that identifies the process flow of different compositional units to complete a given objective/task/goal (*e.g.* the composition of a data aggregation script, statistical model, and data plot module to complete a modelling workflow).
**Data Architecture.** The representation of knowledge that identifies the various forms of data sources and specifications that could be input, output, or flow through the resultant system and the individual compositional units within it (*e.g.* a CSV file containing emergency department visit time-series data).
**Physical Resources.** The representation of knowledge that identifies physical executional environments that could systematically execute a constructed resultant system by an ODCS (*e.g.* a personal computer with a specific operating system or a supercomputer with a large number of processors).

To complement the five categories of knowledge depicted in Figure 1, three more conceptual considerations are illustrated: human and system influences, syntactic and semantic knowledge representation and the relationships between the different categories of knowledge.

A differentiation between syntactic and semantic knowledge representation is illustrated in Figure 1. Essentially, entities of knowledge that are described as ”syntactic” would represent physical objects considered within an ODCS (e.g., algorithm, web service, data source, data set, person, a computer server, etc.), where ”semantic” knowledge entities would represent the ’realization’ of the syntactic entities (e.g., programming language, functional purpose, dimensions/structure of data, human actor role, operating system environment, etc.) In terms of semantic representation, five sub-types can be considered: function, data, execution, quality, and trust. Gillespie et al (2011) and Cardoso (2005) describe these further.

Finally, the framework identifies the relationships between the categories of knowledge. These relationships can also be described as either syntactic or semantic.

# 3 UTILIZATION OF THE FRAMEWORK FOR ONTOLOGY EVALUATION

As stated in Section 2.3, the framework performs as a tool to facilitate effective ontology engineering methodologies for ODCS ontological knowledge. In this section we suggest how the framework can be utilized in the context of ontology evaluation by presenting a knowledge framework checklist. This checklist can be applied by any ontology engineer who is investigating the ontological knowledge for an ODCS. Following the work of (Brank et al., 2005), (Vrandečić, 2009) provided a description of different aspects of ontology evaluation. As discussed in section 2.2, one of these aspect is **context**. Our focus for this paper is to evaluate the adaptability of context. Context is defined in terms of considering the aspects of the ontology in relation to other variables in its environment (Vrandečić, 2009). ODCS-specific examples may include human influence, an application using the ontology, a data source the ontology describes, etc. Due to the high-level categorical representation that the knowledge identification framework provides, context is the aspect of ontology evaluation that best fits our assessment.

An ontology evaluation is assessed by how well a given aspect satisfies certain criteria/metrics (Vrandečić, 2009). In terms of the knowledge framework and the nature of ODCS applications, **adaptability** is considered. In (Gillespie et al., 2011), we argued that the framework can assist with questions such as *“How can ontological knowledge represented in ODCS ‘A’ be utilized or integrated into the ontologies for ODCS ‘B’?”*. Adaptability deals with the extent to which the ontology can be extended and/or specialized without breaking or removal of existing axioms (Vrandečić, 2009). Therefore, within this ontology evaluation example we plan to assess the adaptability of the context in a specific ODCS’s ontologies.

## 3.1 Evaluation Checklist: A Concept from Software Quality Assurance

Within the software engineering industry, long standing initiatives have been put in place for software quality assurance (SQA) (International Standards Organization, 2001; McCall et al., 1977). One of the main SQA standards calls for the development of software that is strongly portable. In the investigation of portability, standards usually isolate that a software developer or quality assurance professional must consider dynamics such as *adaptability* and *flexibility* (McCall et al., 1977). This concept is strongly related to the ontology evaluation focus we wish to pursue.

During the review of software quality assurance evaluation methods, we discovered the persistent utilization of checklists to quickly illustrate the aspects of SQA that have and have not been satisfied by a piece of software (Ince, 1995). Large companies and organizations (such as (NASA, 2011)) utilize checklists to hold themselves accountable in the production of high quality products and services. Acknowledging the usefulness of this tool, we constructed a checklist that assists with the specific focus of our ontology evaluation (*i.e.* **adaptability of context**), using our framework as the structure for the checklist document.

### 3.2 From the Framework: Ontology Evaluation Checklist for ODCS

In section 2.2 we observed that an ontology evaluation strategy considers and is affected by variables such as the purpose of the evaluation, and the aspect of the ontology to be evaluated.

We also arrived at the conclusion that an ontology evaluation process should focus on an aspect that best fits the goal of the evaluator and the structure of the framework. Based on these notions, the structure of the checklist defined for the purpose of evaluating the ontologies in this case study addresses the content described in the knowledge identification framework (section 2.3). The evaluation of the adaptability of the context of the ontologies is the focus for the checklist and is structured as follows:Part A: ODCS & Ontology Overview; Part B1-B5: Categories of Knowledge(Syntax and Semantics); Part C: Internal Relationships; Part D: Human Actor Relationships; Part E: Physical Resource Relationships; Part F: Overall Assessment; Part G: Extra Space for Comments.

While the focus of this paper is on ontology evaluation of existing ontologies it is possible to apply this framework during the iterative process of developing an ontology.

## 4 CASE STUDY: UTILIZATION OF FRAMEWORK WITHIN AN ONTOLOGY ENGINEERING SCENARIO

This section presents the BioSTORM ontology evaluation case study. We start off by presenting an overview of the steps. To conduct the case study, an understanding of the BioSTORM prototype system was needed. Therefore, we provide this overview in section 4.2. This is followed by an assessment of the adaptability of the ontology context . Finally a presentation of the notable deductions from the case study is presented. The checklist was utilized explicitly to address items 2-4 of section 4.1 while Part A and B1 are illustrated in the Appendix.

### 4.1 Methodology: Running an Evaluation with the Checklist

To run an effective ontology evaluation session we aimed to utilize the proposed checklist as a pseudo-research method. Based on its document structure the following method was followed:

1. Visit the BioSTORM website (BioSTORM, 2009) and download all of the ontologies and supporting documentation and publications.
2. Before starting the checklist, read the related publications to understand the system-specific domain (*i.e.* composition software agents) and the domain-specific application (*i.e.* syndromic surveillance).
3. Run a preliminary overview evaluation by documenting Part A of the checklist: ODCS & Ontology Overview.
4. For each category of knowledge that exists within the BioSTORM ontologies, document the respective Part B.
5. Next, consider the all possible relationships that could exist between the categories of knowledge in Parts C, D, and E.
6. Provide an overall assessment (in Part F) utilizing the evaluation within the checklist document.

### 4.2 Biological Spatio-temporal Outbreak Reasoning Module (BioSTORM)

*Syndromic surveillance* is defined as a type of surveillance activity that uses health-related data (*e.g.* emergency room visits, sales of over-the-counter medications, etc.) to establish the probability of a disease outbreak that warrants a public health response. The problem domain of *syndromic surveillance* is characterized by the discovery of links between current data and previously unrelated data (Nyulas et al., 2008; Crubezy et al., 2005; Pincus and Musen, 2003). What this implies is the requirement to integrate many diverse, heterogeneous and disparate data sources. It also has the requirement of employing a varied number of computational methods/algorithms that can reason about the data from these different sources. The integration of the data source presents a major challenge in terms of *context*. Each data source and data concept has to be correctly understood less it be misinterpreted with dire consequences. These are the motivations for BIOSTORM, an experimental prototype system implemented at Stanford University that supports the configuration, deployment and evaluation of analytic methods for the detection of outbreaks. In this implementation, the quest to provide *context* of data sources is done through the use of ontologies. The ontologies serves as a model through which the semantics of data sources and their data can be described thereby giving their context.

The BioSTORM implementation follows a JADE-based system architecture that deploys a number of agents that collaborate in analyzing data for outbreak detection (Nyulas et al., 2008). This is a three layered framework consisting of the knowledge layer, agent platform, and the data source layer.

1. A **Knowledge Layer** consisting of a surveillance method library, and their descriptive ontologies. These ontologies and the methods API describe the functionality of the system. Tasks, Methods and Connectors are defined in the ontology classes to model communication paths. Algorithms are also defined to model related tasks.
2. An **Agent Platform** that generates system agents based on the information retrieved from the knowledge layer. They assume a data-driven environment where each agent may not be aware of the producer or consumer of its information. Each agent publishes its results on the blackboard and consumes or uses information on the blackboard as per its needs and is not aware of the existence of other agents in the platform.
3. **Data Source Layer.** Based on the *Data Source* ontology, this layer describes the environment within which the agents interact.

A varied number of publications have resulted from the research related to BioSTORM. These publications depicted different information about the ontologies which has lead to some confusion and difficulty in evaluating the ontologies since the ontologies listed in the publication do not resemble those available in the BioSTORM repository.

## 4.3 Assessment of Adaptability of Context

The assessment set out in this section focuses on Part A and B1 of the checklist provided in the Appendix. Note that aspects of the evaluation relating to Part A.1 and A.2 have been addressed in section 4.2.

### 4.3.1 A.5.ii: Difficult Workflow Syntax Knowledge

With the *sm:Algorithm* entity, a workflow is explicitly defined by the utilization of an object property titled *sm#steps*. An instance of an algorithm *sm#steps* through *sm#Tasks*, *sm#BranchPoints*, and *sm#Tag*. In this evaluation, we could not locate any depiction of chronological ordering therefore we believe that the workflow knowledge is heavily dependant on hard-coded knowledge within the JADE multi-agent system. This can also be attributed to the choice of architecture (*i.e.* blackboard) which follows a parallel and distributed pattern with agents publishing to and using data on the blackboard.

### 4.3.2 B1.2.ii: JADE-CLASS Adaptability

The *smj:JADE-CLASS* is difficult to adapt to the context of other ODCS because it directly relates to a JADE Software Agent instantiation. Most ODCS would not utilize this multi-agent system, thus contextually this CU syntax knowledge entity is *Difficult* to adapt.

### 4.3.3 B1.4.iii: Adapt CU Function Semantics to other ODCS Ontologies

The *sm#Algorithm*, *sm#Task*, and *sm#Method* classes could be utilized in other CU representations for other ODCS. Within their axiom relationships, an *sm#Algorithm* primarily *sm#steps* through Tasks (we are ignoring *sm#BranchPoint* and *sm#Tag* for this checklist item). A *sm#Task* is composed of *sm#Methods* and sub-*sm#Algorithms*, where the methods could have more sub-*sm#Tasks*. If an ontology engineer wished to adapt this structural composition, s/he must accept the detailed object property relationship. In some cases, this may be too specific depending on other ontological specifications.

If this entity composition is favoured however, the syndromic surveillance algorithms, tasks, and methods are defined simply as sub-classes of the three abstracted *sm#Algorithm*, *sm#Task*, and *sm#Method* entities. Thus, other application domains could easily utilize these entities by engineering their own domain-specific classes.

### 4.3.4 Other Notable Deductions

As depicted in Part A.1 and A.5.iii of the check-list, the BioSTORM data source ontology represents the semantic definitions of data (*e.g.* datatype, data structure, temporal and spatial dimensions, etc.). These description are not domain specific and thus any data used for any type of software will have these characteristics. This renders these descriptions adaptable to a different context.

Throughout the case study a recurring point was observed: if a user wishes to use a JADE multi-agent system, these ontologies would allow for quick implementation, however outside of that specification utilizing some aspects of these ontologies could prove difficult. Having said that, some aspects of the ontologies can be seamlessly adapted to a different application. This conclusion was drawn based on the observations detailed in B1.4.i in the checklist that identifies the modelling of top classes as “meta-classes” that can be further sub-classed to represent knowledge in the relevant domain.

## 5 DISCUSSION AND CONCLUSIONS

In this paper we described a knowledge identification framework developed in our previous work. This framework emerged from the realization of a gap between existing ODCS and their ability to share knowledge. Hence, the knowledge identification framework would guide ontology engineering methodologies with such tasks as requirements gathering, ontology capture, and evaluation of ontologies for ODCS specifically. In this paper we have demonstrated the usage of the framework to evaluate the context of BioSTORM ontologies by presenting a knowledge framework checklist. The evaluation centred around whether the context of the ontology is adaptable.

Challenges were encountered during the evaluation process. These include the lack of documentation and confusion where documentation was present due to gaps that may have been created by revisions of the ontologies. Through the usage of the developed checklist, we were able to identify the categories of knowledge that were and were not represented in the ontologies. In this case study, the *Human Actors* and *Physical Resources* categories were not represented. We attribute this absence to the context and nature of the application. BioSTORM is an agent-based implementation and thus the agents will only operate in their environment and nowhere else (in this case, the JADE platform).

Throughout the case study we observed that the descriptions provided were strongly tied to the JADE multi-agent domain. Therefore if a user wishes to use a JADE multi-agent system, these ontologies would allow for quick implementation, however outside of that specification utilizing some aspects of this ontology could prove difficult. Having said that, we also observed that some aspects of the ontologies can be adapted to different contexts or used in a different ODCS. An example of this is the CU functional semantics. This is modelled through the definition of the *Algorithm*, *Task* and *Method* as top level classes. These classes can be further sub-classed to specialize them so as to model the domain of interest.

It is important to recognize that although the checklist can explicitly imitate an evaluation methodology it should not be used as such. We observed through our experience that during an ontology evaluation session, an analyst will fluidly move through the object properties between entities. Thus, an analyst is consistently assessing different categories of knowledge and their respective relationships during the session. The checklist proved to be a tool that facilitated this dynamic. While we have used the framework only for evaluation, it is our postulation that an ontology engineer may use this framework to guide him/her during the capturing of valid ontologies as well. Due to space constraints not all material is covered in this paper. To obtain a more comprehensive technical report visit: http://www.ontology.socs.uoguelph.ca

## REFERENCES

Arpinar, I. B., Zhang, R., Aleman-Meza, B., and Maduko, A. (2005). Ontology-driven Web services composition platform. *Information Systems and e-Business Management*, 3(2):175–199.

BioSTORM (2009). Welcome to BioSTORM-Project Description. Available Online http://biostorm.stanford.edu/.

Brank, J., Grobelnik, M., and Mladenic, D. (2005). A survey of ontology evaluation techniques. In *Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005)*, pages 166–170. Citeseer.

Burstein, M. H. and Mcdermott, D. V. (2005). Ontology

Translation for Interoperability Among Semantic Web Services. *AI Magazine*, 26(1):71–82.

Cardoso, J. and Sheth, A. (2005). *Introduction to Semantic Web Services and Web Process Composition*, volume 3387 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Crubezy, M., Connor, M., Pincus, Z., Musen, M., and Buckeridge, D. (2005). Ontology-Centered Syndromic Surveillance for BioTerrorism. *IEEE Intelligent Systems*, pages 26–35.

Gillespie, M. G., Hlomani, H., and Stacey, D. A. (2011). A Knowledge Identification Framework for Engineering Ontologies in System Composition Processes. In *12th IEEE Conference on Information Reuse and Integration (IRI 2011) to appear*, Las Vegas.

Gillespie, M. G., Stacey, D. A., and Crawford, S. S. (2010). Satisfying User Expectations in Ontology-Driven Compositional Systems: A Case Study in Fish Population Modelling. In *International Conference of Knowledge Engineering and Ontology Development*, Valencia.

Gruber, T. R. (1993). A translational approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220.

Hlomani, H. and Stacey, D. A. (2009). An ontology driven approach to software systems composition. In *International Conference of Knowledge Engineering and Ontology Development*, pages 254–260. INSTICC.

Ince, D. C. (1995). *Introduction to Software Quality Assurance and Its Implementation*. McGraw-Hill, New York, NY, USA.

International Standards Organization (2001). ISO/IEC 9126:2001.

McCall, J., Richards, P., and Walters, G. (1977). *Factors in Software Quality, RADC TR-77-369*, volume I, II, III. US Rome Air Development Center Reports.

NASA (2011). Software Quality. Available online http://sw-assurance.gsfc.nasa.gov/disciplines/quality/index.php.

Nyulas, C. I., O'Connor, M. J., Tu, S. W., Buckeridge, D. L., Okhmatovskaia, A., and Musen, M. a. (2008). An Ontology-Driven Framework for Deploying JADE Agent Systems. *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 573–577.

Pincus, Z. and Musen, M. a. (2003). Contextualizing heterogeneous data for integration and inference. In *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, pages 514–8.

Vrandečić, D. (2009). *Ontology Evaluation*, pages 293–313. Springer Berlin Heidelberg, Berlin, Heidelberg.

## APPENDIX

In completing the checklist, the user first gives an overview of the ODCS and the ontologies that underlie it (*i.e.* Part A of the checklist). This is done by answering the questions defined by indicating "Yes" if the aspect is represented, "No" if it is not represented, "Diff" if there is difficulty in telling, "NA" or "TBD" in the cases where further research needs to be done. When the overview has been done, Part B of the checklist can be completed. Note that Part B1 of the checklist can be replicated to equally assess other categories of knowledge as may be relevant.

Table 1: Part A of our proposed checklist is an overview of the ODCS being investigated and its ontologies. The answers within this table relate to our case study investigation of BioSTORM.

| | | **Yes/Diff/No NA/TBD** | **Comments** |
|---|---|---|---|
| **Part A: ODCS & Ontology Overview** | | | |
| A.1 | Does explicit documentation and publications exist to explain the application context of this ODCS? | *Yes* | *See Section 4.2* |
| A.2 | Does explicit documentation exist to describe the system architecture of this ODCS? | *Diff* | *See Section 4.2* |
| A.3 | List and describe the different ontologies in the ODCS. Provide a name-space acronym for each ontology. | | |
| | – *DataSource.owl (ds)*: provides descriptions for available data sources on the ‘blackboard’ to unify agents (with specific input/output) together in a process flow<br>– *SurveillanceMethods.owl (sm)*: describes the surveillance algorithms, tasks and methods that are related to evaluation, outbreak detection, and simulation.<br>– *SurveillanceEvaluations.owl (se)*: provides a description of an “evaluation analysis” (*i.e.* configurations of outbreak detection and simulation)<br>– *SurveillanceMethodJADEOntology.owl (smj)*: provides association between JADE agents and surviellance algorithms/tasks/methods for evaluation, detection, and simulation | | |
| A.4 | List the ontologies that are imported into ODCS | | |
| | – *temporal.owl*: a World Wide Web Consortium (W3C) ontology to specify temporal components and proportions.<br>– *beangenerator.owl*: an ontology utilized by the JADE multi-agent system and required for the implementation of BioSTORM integrated with JADE.<br>– *more TBD* | | |
| A.5.i | Is Compositional Unit Knowledge represented within the ontologies? Comment whether it is syntax, semantic, or both. | *Yes* | *Both* |
| A.5.ii | Is Workflow Knowledge represented within the ontologies? Comment whether it is syntax, semantic, or both. | *Diff* | *Syntax, see Section 4.3* |
| A.5.iii | Is Data Architecture Knowledge represented within the ontologies? Comment whether it is syntax, semantic, or both. | *Yes* | *Semantic* |
| A.5.ix | Is Human Actor Knowledge represented within the ontologies? Comment whether it is syntax, semantic, or both. | *No* | |
| A.5.x | Is Physical Resources Knowledge represented within the ontologies? Comment whether it is syntax, semantic, or both. | *No* | |
| A.6 | Do relationships between the categories of knowledge exist? | *Yes* | |
| A.6.i | If yes, indicate which relationships (ten possible permutations) | | |
| | – *CU–DA* (Compositional Units - Data Architecture)<br>– *CU–WF* (Compositional Units - Workflows)<br>– *WF–DA* (Workflows - Data Architecture) | | |

Table 2: Part B1 of our proposed checklist is an investigation of Compositional Unit Knowledge. The answers within this table relate to our case study investigation of BioSTORM. Note that not all checklist questions for this "part" are included because this presentation is merely a proof-of-concept.

| | | **Yes/Diff/No NA/TBD** | **Comments** |
|---|---|---|---|
| **Part B1: Compositional Unit (CU) Knowledge** | | | |
| ***Syntax*** | | | |
| B1.1 | Is the CU syntax knowledge explicitly represented in an ontology? | *Yes* | |
| B1.1.i | If yes, list and describe the classes/entities | | |
| | – *smj#JADE-CLASS*: represents physical surveillance agents that can be executed on the JADE multi-agent system | | |
| B1.1.ii | If no, where is it represented? | | *TBD* |
| B1.2 | Do explicit mappings to ***imported*** ontologies for the CU syntax knowledge exist? | *Yes* | |
| B1.2.i | If yes, list the mappings | | |
| | – *smj#JADE-CLASS* entity is an explicit mapping from *beangenerator.owl* to automatically incorporate an ontology-defined agent into the JADE execution tool-kit. | | |
| B1.2.ii | Also if yes, are these imported classes/entities adaptable for other ODCS? | *No* | *See Section 4.3* |
| B1.3 | For CU syntax knowledge, do ***other*** ODCS explicitly map to ***this*** ODCS's ontologies? | *TBD* | |
| B1.3.i | If yes, list the mappings and describe their adaptability. | | *TBD* |
| B1.3.ii | Could the identified CU syntax knowledge be adapted into candidate mappings for other ODCS ontologies? | *NA* | *only uses imported entities* |
| ***Semantic*** | | | |
| B1.4 | Is the context for CU Function Semantics explicitly represented in the ontologies? | *Yes* | |
| B1.4.i | If yes, list and describe the main classes/entities. Note how the semantics describe the syntax classes/entities above. | | |
| | – *sm#AnalysisEntity*: top-level class describing analysis actions for evaluation, outbreak detection, and simulations.<br>– – *sm#Algorithm*: top-level representation of an evaluation, outbreak detection, or simulation process<br>– – *sm#Task*: a composition of a series of methods to perform a certain action<br>– *sm#Method*: a top-level collection of Primitive Methods and TaskDecompositionMethods<br>– – *sm#PrimitiveMethod*: single execution statement with no sub-tasks required<br>– – *sm#TaskDecompositonMethod*: is a task (or sub-task) that is another series of methods<br>***Note***: All algorithms, task, and methods entities directly correlate to the semantic context of a *smj:JADE-CLASS*. | | |
| B1.4.ii | For Function semantics, do explicit mappings to ***imported*** ontologies exist? | *No* | |
| B1.4.iii | Could the identified Function semantics be adapted into candidate mappings for ***other*** ODCS ontologies? | *Diff* | *see Section 4.3* |
| B1.5 | Is the context for CU Data Semantics explicitly represented in the ontologies? | *Yes* | |
| B1.5.i | If yes, list and describe the main classes/entities. Note how the semantics describe the syntax classes/entities above. | | |
| | – *se#InputSpecification & se#OutputSpecification*: representation of semantic context of input/output into algorithms, tasks, and methods | | |
| B1.5.ii | Do explicit mappings to ***imported*** ontologies exist? | *No* | |
| B1.6 | Is the context for CU Execution Semantics explicitly represented in the ontologies? | *No* | ***Note***: *same* answer for Quality and Trust semantics |