Electronic version of an article published as:
Orłowski C., Kapłański P., Nguyen N., Pietranik M., The Use of an Ontotrigger for Designing the Ontology of a Model Maturity Capsule, INTERNATIONAL JOURNAL OF SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, Vol. 26, Iss. 05 (2016), pp. 715-731, DOI: 10.1142/S0218194016500236

https://www.worldscientific.com/doi/abs/10.1142/S0218194016500236

Cezary Orłowski, Ngoc-Thanh Nguyen, Paweł Kapłański, Marcin Pietranik

# The use of an ontotrigger for designing the ontology of a model maturity capsule

## Summary

The aim of this work is to give the definition and present the possibility of applying (introduced and defined here) ontotriggers (own original concept) to design the ontology of a maturity capsule used in the assessment of IT projects. The complexity of designing ontology processes raises the question of whether there is a need for designing ontologies in a situation where it is possible to map them. The work is divided into four main parts. The first part presents and defines the concept of an ontotrigger. The second part presents a model maturity capsule. Similarities to the maturity capsule of a project managed in accordance with the SCRUM methodology have also been indicated. The third part discusses the method of building ontologies for both capsules and indicates the possibility of mapping them. The fourth part presents the application of an ontotrigger which uses the ability to map both ontologies.

## Introduction

BigData is one of the main topics of modern information technology. The processing of BigData may be based on classic data processing (databases, multimedia databases, etc.), but may also involve semantic transformation processes [5]. Semantic search engines provides alternative mechanisms of searching for resources, and the transformation processes are supported by solutions compatible with the Semantic Web concept [10, 11]. Ontologies play an important role in these processes and designing them is a separate domain of knowledge supporting the processing of resources [8]. The concept of processing can take into account a number of different processes, but in terms of searching and inference, the process of mapping resources and their subsequent integration becomes an essential one. Ontologies, which are increasingly common, are becoming the mechanism which supports the mapping of resources [6]. In other words, the mapping of resources makes sense in a situation where the ontology of these resources is mapped. However, constructing an ontology requires a semantic analysis of concepts and their subsequent formalization with the use of ontology languages. Therefore, the authors put forward the idea of building ontotriggers - triggers whose main task is to map an ontology for enhancing the re-usability of reference ontologies. This concept is based on using an ontological stack with its main components, such as the ontology structure and its constituent classes and attributes [12].

## 1. The concept of ontotriggers

The concept of triggers has been used for years in the theory and application of databases [1]. Their role is to elicit specific procedures for handling databases and to allow the control of the processes of database processing. The authors borrowed this concept to automate the process of ontology mapping, to monitor the condition of ontologies and to indicate the need to modify them. The basis for testing the possibility of integration (ontology mapping effect), as well as

being the core of this work, is the concept of an ontological stack developed by the authors [12]; it is a mixed hierarchical-linear structure, on the one hand defining the structure of an ontology, and on the other the possibility of mapping such structures. The concept of a stack and its significance in analyzing the distance between logical expressions (for defining the semantics of attributes), as well as in assessing the potential conflicts, is considered to be a very important element of the proposed concept.

As the process of designing an ontology requires knowledge of the field (classes, attributes, and relationships), the proposed solution therefore requires the need for the ontology design to be examined. The already existing ontologies are analyzed and it is evaluated whether they can be the subject of mapping. Only if there are significant differences, mainly in the structure of the ontology, is the need to design taken into consideration. The process of the analysis and assessment of the needs for designing is carried out with the use of ontotriggers built on the basis of ontology structures. Figure 1 shows a diagram of conduct which forms the basis for the construction of an ontotrigger to assess the need for designing a SCRUM maturity capsule. The classes and attributes of the SCRUM maturity capsule are analyzed and then the possibility of using a standard maturity capsule for the design of the SCRUM capsule is examined. Then, based on the indications of the trigger, the decision whether to build or not is made. Such use of triggers enables the analysis and the application of already existing entities and their ontologies, while at the same time suggesting areas in which designing entities and their ontologies is essential.

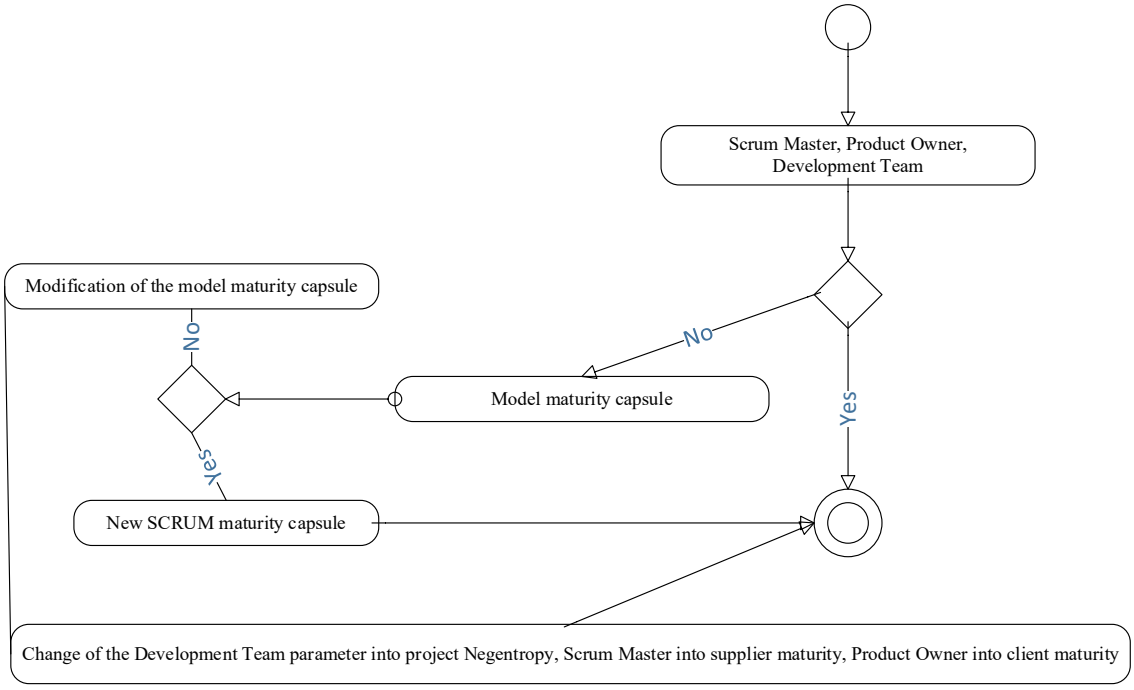


Fig. 1 The mechanism of mapping a model maturity capsule onto a SCRUM maturity capsule (the structure of an ontotrigger)

### 1.1. Ontotrigger definition

We take into consideration a knowledge management system [5], [16], which allows the processing of many independent ontologies. The ontologies should be consistent with the OWL2 [15] + SWRL [16] standard, although this is not a necessary condition. It is important for the analyzed knowledge management system to allow the performance of the following tasks on any of the selected ontologies O:

1) *Task IsTrue(O, <condition>) – giving the answer true/false depending on <condition> in the context of a given Ontology O (carried out by Reasoner).*
2) *Task AddAxiom(O, <axiom>) – adding <axiom> to a given Ontology O*

We assume that the following rule is called here an Ontological Reactive Rule:

$$\text{IF IsTrue(O,<P>) THEN AddAxiom(O,<Q>)} \quad (1)$$

where <P> is an ontological premise and <Q> an ontological consequence. Both the ontological premise <P> and the consequence take the form of logical tasks. While the premise expresses a test (the condition - which may be true or not in the given ontology - can take a value of true/false), the consequence specifies the knowledge with which the ontology is enriched[1] if the premise occurs. It should be noted that both the premise <P> and the consequence <Q> have the form of correct statements in the context of the knowledge management system (in the case of OWL + SWRL they are statements in description logic).

We also assume that if there is an algorithm A which for any statement written in the context of ontology X is able to build an analogous sentence in the context of ontology Y, then such mapping is called: Mapping Ontology X onto Ontology Y. In other words, for two ontologies X and Y, the mapping of ontology X onto ontology Y is the function $A_{X->Y}$: E→E, where E is a valid expression from the point of view of the knowledge management system. If the mapping used in $A_{X->Y}$ is not 1:1 but requires a comparison of similarities between the structures of the ontologies, then the cutoff point (parameter v, at which it is acknowledged that ontologies are the same) is specified as one of the parameters $A^{v}_{X->Y}$ and the mapping of rule A takes the form of $A^{[]}_{[]->[]}$ becoming a function dependent on one variable and three parameters.

Hence, a Reference Ontology is a selected ontology managed by the knowledge management system. Let us establish a Ontological Reactive Rule R for this ontology. With a Reference Ontology O, a reactive rule R, and any other ontology X, which can be used as arguments in ontology A mappings, the ontotrigger is defined as three elements <O, R, $A^{[]}_{[]->[]}$>. The ontotrigger, applied to any ontology X, modifies it according to the

[1] If we allowed the removal of knowledge from the ontology, it (the knowledge) would no longer be monotonic [5], and (depending on the implementation) could no longer be deterministic. If we allow only the addition of new (AddAxiom) knowledge, we do not lose monotonicity. Ontological Reactive Rules lead to the loss of decidability of the knowledge base which they refer to, unless they are "safe" (DL-Safe in the case of knowledge bases relying on Descriptive Logic).

Ontology Mapping in the context of the Reference Ontology O using the Reactive Rule R along the following scheme.

$$\text{IF IsTrue}(O, A^{v}_{X->O}(<P_X>)) \text{ THEN AddAxiom}(X, A^{v}_{O->X}(<Q_O>)) \quad (2)$$

where $<P_X>$ is the premise in the context of ontology X, which is mapped onto ontology O with the use of A. <Qo> is the consequence of the reactive rule which is transformed and added to ontology X.

The ontotrigger, defined in this way, has a global logical consequences and is applied to all ontologies in the knowledge base for which the Ontology Mapping is defined. When modifying ontology X, the condition mapped to a reference ontology O is checked, the reactive rule is applied and, in consequence, ontology X will be modified. From the point of view of any ontology X, the ontotrigger can be seen as a structure hidden in the mechanisms of the knowledge management system, which is closed in the Reference Ontology O, but which has real consequences that are revealed during the launch of the ontotrigger.

## 2. Maturity capsules

The description of maturity capsules will begin with the description of a model maturity capsule. The concept of a model maturity capsule C-S-P should be understood as a set of evaluations of the maturity of the client organization, the supplier organization and the project (estimated by scalar project negentropy) [9]. Using the integrated approach to the evaluation of maturity in the form of the maturity capsule C-S-P has changed the philosophy - from the independent treatment of the concepts of maturity of the client organization and the supplier organization, as well as the difficult-to-define concept of project complexity - into an integrated concept of a maturity capsule: referring to the client, supplier and project.

$$\mathbf{z}_t = \begin{bmatrix} z_t^c \\ z_t^s \\ z_t^p \end{bmatrix} \quad (3)$$

$\boldsymbol{z}_t$ – multi-dimensional evaluation of the IT project management
$z_t$ – scalar evaluation of the level of IT project negentropy, <0, 5>
$z_t^c$ – sub-level of management resulting from client maturity, <0, 5>
$z_t^s$ – sub-level of management described by provider maturity, <0, 5>
$z_t^p$ – sub-level of management related to negentropy, <0, 5>

This integration expresses the need for a simultaneous evaluation of the maturity of these three entities in assessing the level of project management. Moreover, while focusing the effort of modeling on technology management, it is easy to distinguish its management/key factors, such as the levels of the client and the supplier, and the status of the project, which determine the predestination (success or failure) of the entire project. The concept of the maturity capsule, introduced here, integrates these key factors of the project and creates favorable conditions (in the form of the environmental provision of input data) for a practical implementation of the information technology management model. In this way, the

management of technology can become a dynamic selection of methods and tools which are adequate for the current level of maturity.

The measurements of scalar negentropy for the development of enterprise architecture were introduced on the basis of two cases: the Continuum and the ADM process, and the so-called main development cycle [2,3]. Taking the two cases (with different negentropies) into consideration enriched the environment and conditions for the selection of measurements of negentropy. The case of building the Continuum formed the basis for classifying project negentropy in rough categories (1 to 5), or in a pentavalent linguistic scale (from very small to very large negentropies). The analysis of both cases allowed for a specification of the measurements of negentropy which is based on the schema for determining the variables of the width of the design architectures repository, the length of the documentation catalogue and the height of the ITM section ($a_t$ , $d_t$ , $pr_t$). These variables are further defined with the use of appropriate levels of processes. Monitoring them relies on the outcome of competence questions, which allows a linguistic evaluation of the growth of negentropy.

To specify the 'supplier organization maturity' variable, the relevance of using the CMMI model for the assessment of this variable was analyzed. It was noted that in terms of service organizations, it is a better solution to use the ITIL standard [10]. The concept is related to the changes which are taking place on the IT market, where the majority of organizations function as support organizations, evaluated and developed on the basis of the specifications of this standard. While maintaining the (three-level) structure of the description of organization maturity, it is natural, therefore, to use the areas of the organization which were treated as organization evaluation variables, instead of using the key areas. These areas had processes subordinated to them which are key to their functioning, whereas the analysis of the extent to which these processes were realised (as in the case of project negentropy) was evaluated on the basis of sociological studies involving the analysis of answers to test questions on the quality (level) of the processes of the organization.

The specifications of the key processes of the organization were presented in detail. Their method of assessment was also changed (relative to the traditional approach), proposing a linguistic interpretation on a four-level scale, as the five-point one was too difficult for the members of the client organization. However, the evaluations obtained on the basis of the test questions were transposed to a more convenient pentavalent rating scale of organization maturity. The dichotomy of evaluations (the four-level scale in response to the posed questions and the five-level scale of the evaluation of organization maturity) reflects a natural divergence between the precision of expert opinions and the required accuracy of the evaluation of the organization.

The third of the specified variables was 'client maturity'. It turned out that - as in the case of supplier maturity - the adopted variables, suitability and matching describe small client organizations or their representatives [13]. For larger organizations, the use of these variables was not sufficient to determine how the organizational culture of large client organizations is reflected in the quality of cooperation between the supplier and the client. The maturity of the

client organization, therefore, is expressed with a variable subjected to the evaluation process where the COBIT standard is applied, and a linguistic description is introduced [4,5].

After the presentation of a standard maturity capsule, the SCRUM maturity capsule will now be discussed. The developed model maturity capsule, designed to evaluate the state of realization of the enterprise architecture project, is relatively complex. Hence, it was further modified for the analysis of the Scrum process, by the introduction of the concept of the Scrum Maturity Capsule. The creators of Scrum did not provide methods for measuring the maturity of management processes. Nonetheless, there are attempts to answer the team's question - What does Scrum really do? One of the most popular methods is to use Henrik Kniberg's "Unofficial Scrum Checklist". The aim of the "Unofficial Scrum Checklist" used by the ScrumMaster is to see "how mature" the process is.

For this reason, it is advisable to apply the CSP capsule to evaluate the status of SCRUM. The suitability of the CSP capsule can be determined for the following recipients:

- the ScrumMaster for the analysis of data from the evaluations of the project in a short time. By using the capsule, the ScrumMaster can focus on the identification of obstacles which should be removed;
- the project manager who focuses on long-term analysis and team collaboration within the framework of Scrum. On the basis of the maturity of the management process, the project manager is able to determine the impact on the Product Owner, which in turn helps in taking appropriate action.

The CSP capsule has been mapped to the three important elements of SCRUM, represented by roles, such as the Scrum Development Team, the Product Owner and the ScrumMaster. To measure these elements, a research questionnaire was applied to evaluate the Product Owner and the ScrumMaster. For the evaluation of the supplier organization (team) the measurement of normalized Velocity was applied. The mapping of roles in Scrum is shown in Figure 2.

Five levels of values were assumed. It was recognized that many evaluations of the implementation of Scrum may go below the range "very small" and be above "very high". One of the consequences of this fact will be to adopt the same limit value for the implementation of Scrum, no matter how mature these elements are. It is assumed, however, that most organizations will lower the level of maturity to the proposed range of values (as adequate to their state).

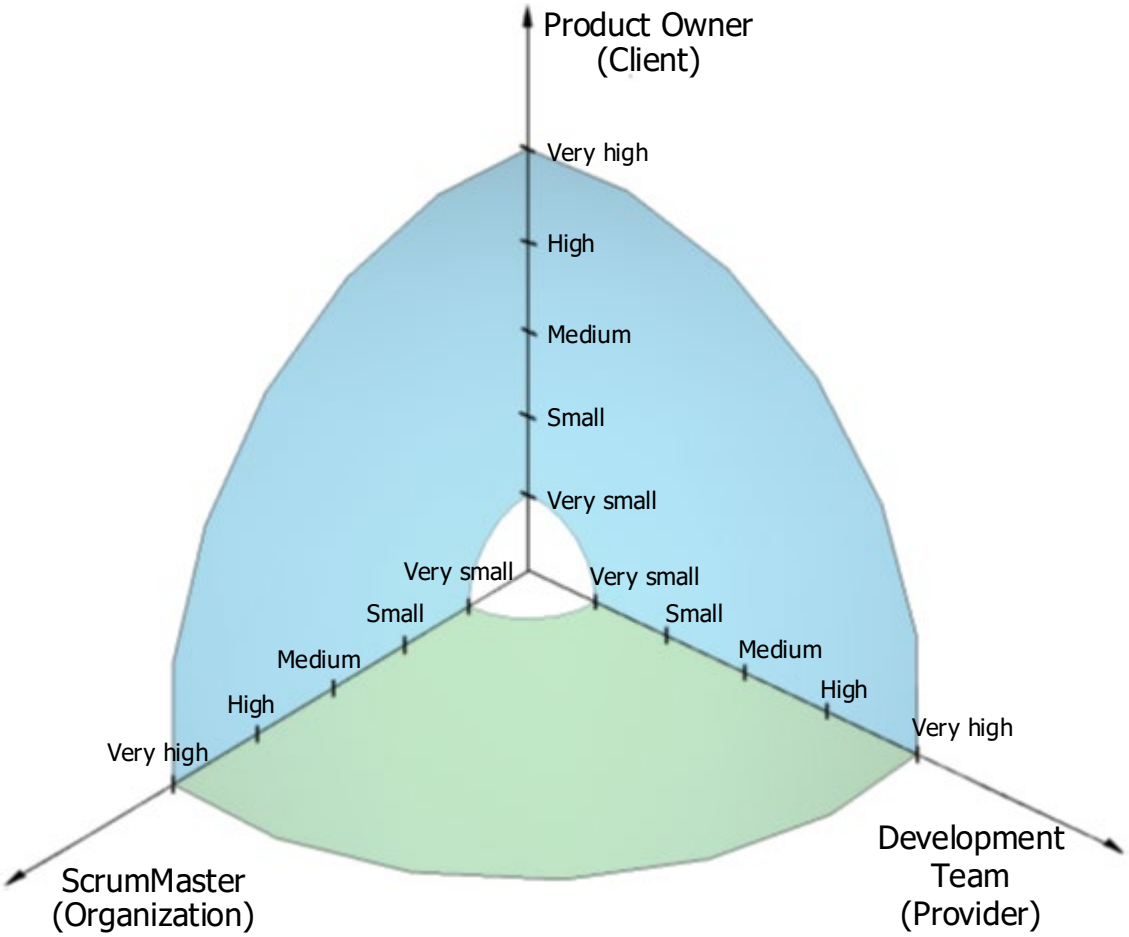


Figure 2 Scrum Maturity Capsule

## 3. Ontologies of Maturity Capsules

After discussing the concepts of the ontotrigger and both maturity capsules, the next issue arise the method of constructing both maturity capsules. The ontologies of the maturity capsules were presented with the use of a (controlled) natural language [14], which allows for the OWL+SWRL formalisms to be transferred to the form of a pseudo-natural language via which the content of the ontologies can be communicated to experts. It is worth noting that the statements quoted below have a direct representation in the OWL+SWRL formalisms.

It was assumed that the maturity capsule can be defined formally as an aggregator of parameters. Hence the maturity capsule aggregates the maturities of the client, the supplier and the project:

```
Every maturity-capsule describes a supplier.
Every maturity-capsule describes a client.
Every maturity-capsule describes a project.
```

Each of the described entities is described with the use of maturity measurements expressed in integers from a set {1..5}, where 0 means none, 1 – very small, 2 – small, 3 – medium-sized, 4 – big, 5 – very big.

```
Every value-of level is (either 0, 1, 2, 3, 4 or 5).

Every supplier has-maturity-evaluation-level (some level value).
Every client has-maturity-evaluation-level (some level value).
Every project has-scalar-project-negentropy-level (some level value).
```

On the other hand, the described entities are characterised by certain measurable values. The measurable values are directly connected with the methodology of running projects. They are values which can be obtained (unlike the maturity values, which are difficult to measure). The measurable values are also expressed in integers from a set {1..5}, having lexical interpretations due to the fact that the measuring process is usually carried out via questionnaires.

```
Every supplier has-service-delivery-level (some level value).
Every supplier has-support-services-level (some level value).
Every supplier has-planning-services-level (some level value).
Every supplier has-application-management-level (some level value).
Every supplier has-infrastructure-management-level (some level value).
Every supplier has-business-perspective-level (some level value).
Every supplier has-safety-management-level (some level value).

Every client has-planning-and-organizing-level (some level value).
Every client has-acquisition-and-implementation-level (some level value).
Every client has-provision-and-support-level (some level value).
Every client has-monitoring-and-evaluation-level (some level value).

Every project has-width-of-repository-design-architectures-level (some level
value).
Every project has-length-of-documents-directory-level (some level value).
Every project has-height-of-the-vertical-it-management-level (some level
value).
```

The connection between the maturity values and the measurable values happens through binding rules. Binding rules transform measurable values to the maturity capsule:

Rule-A) `If a provider has-service-delivery-level greater-or-equal-to 1 and the provider has-support-services-level greater-or-equal-to 2 then the provider has-maturity-evaluation-level equal-to 3.`

Rule-B) `If a provider has-service-delivery-level greater-or-equal-to 2 and the provider has-support-services-level greater-or-equal-to 1 then the provider has-maturity-evaluation-level equal-to 4.`

**Example:**
In order to illustrate the rules presented above, the cooperation of the client and the supplier in the realization of the project was considered. Acme is a client of Alfa-Beta-Software-House, for whom the latter carries out a project involving a Flight-Simulation-Implementation.

```
Acme is a client.
Alpha-Beta-Software-House is a provider.
Flight-Simulation-Implementation is a project.
```

The definition of a maturity capsule in this case is as follows:

```
Capsule-01 is a maturity-capsule.
Capsule-01 describes Acme.
Capsule-01 describes Alpha-Beta-Software-House.
Capsule-01 describes Flight-Simulation-Implementation.
```

Let us assume that the supplier (Alpha-Beta-Software-House) is characterized by the following measurable values:

```
Alpha-Beta-Software-House has-service-delivery-level equal-to 2.
Alpha-Beta-Software-House has-support-services-level equal-to 1.
Alpha-Beta-Software-House has-planning-services-level equal-to 3.
Alpha-Beta-Software-House has-application-management-level equal-to 3.
Alpha-Beta-Software-House has-infrastructure-management-level equal-to 2.
Alpha-Beta-Software-House has-business-perspective-level equal-to 1.
Alpha-Beta-Software-House has-safety-management-level equal-to 3.
```

The application of binding rules (see Rule-B) to the supplier gives the conclusion that the supplier maturity level is 4. Similarly, the rules can be used to evaluate the client.

```
Acme has-planning-and-organizing-level equal-to 3.
Acme has-acquisition-and-implementation-level equal-to 2.
Acme has-provision-and-support-level equal-to 1.
Acme has-monitoring-and-evaluation-level equal-to 3.
```

Below, there are rules for the evaluation of the project:

```
Flight-Simulation-Implementation has-width-of-repository-design-architectures-
level equal-to 3.
Flight-Simulation-Implementation has-length-of-documents-directory-level
equal-to 2.
Flight-Simulation-Implementation has-height-of-the-vertical-it-management-
level equal-to 1.
```

**The ontology of the SCRUM Maturity Capsule**

The process of building an ontology for the SCRUM maturity capsule was presented in a similar way. In the case of projects carried out with the use of the SCRUM methodology, the capsule aggregates the maturity of the project team, the representative of the client and the ScrumMaster:

```
Every scrum-maturity-capsule describes a development-team.
Every scrum-maturity-capsule describes a product-owner.
Every scrum-maturity-capsule describes a scrum-master.
```

As was the case for the standard maturity capsule, the measurable values can be defined for the above-mentioned elements of the SCRUM maturity capsule.

```
Every development-team has-maturity-evaluation-level (some level value).
Every product-owner has-maturity-evaluation-level (some level value).
Every scrum-master has-maturity-evaluation-level (some level value).
```

```
Every development-team has-work-speed-level (some level value).
Every development-team has-enthusiasm-level (some level value).

Every product-owner has-sprint-planning-level (some level value).
Every product-owner has-sprint-evaluation-level (some level value).

Every scrum-master has-adherence-to-the-principles-of-team-work-and-design-
practices-level (some level value).
Every scrum-master has-streamlining-assessment-processes-sprints-level (some
level value).
Every scrum-master has-it-management-level (some level value).
```

On the basis of the above rules, the definition of the Scrum-Capsule-01 can appear as follows:

```
John is a product-owner.
Dream-Team is a development-team.
Gabi is a scrum-master.

Scrum-Capsule-01 is a scrum-maturity-capsule.
Scrum-Capsule-01 describes John.
Scrum-Capsule-01 describes Dream-Team.
Scrum-Capsule-01 describes Gabi.

Dream-Team has-work-speed-level equal-to 3.
Dream-Team has-enthusiasm-level equal-to 2.

John has-sprint-planning-level equal-to 3.
John has-sprint-evaluation-level equal-to 2.

Gabi has-adherence-to-the-principles-of-team-work-and-design-practices-level
equal-to 3.
Gabi has-streamlining-assessment-processes-sprints-level equal-to 2.
Gabi has-it-management-level equal-to 1.
```

## 4. The structure of a sample Ontotrigger

After the presentation of both capsules and the process of development of both ontologies, the application of an ontotrigger for the mapping of both ontologies has been presented.

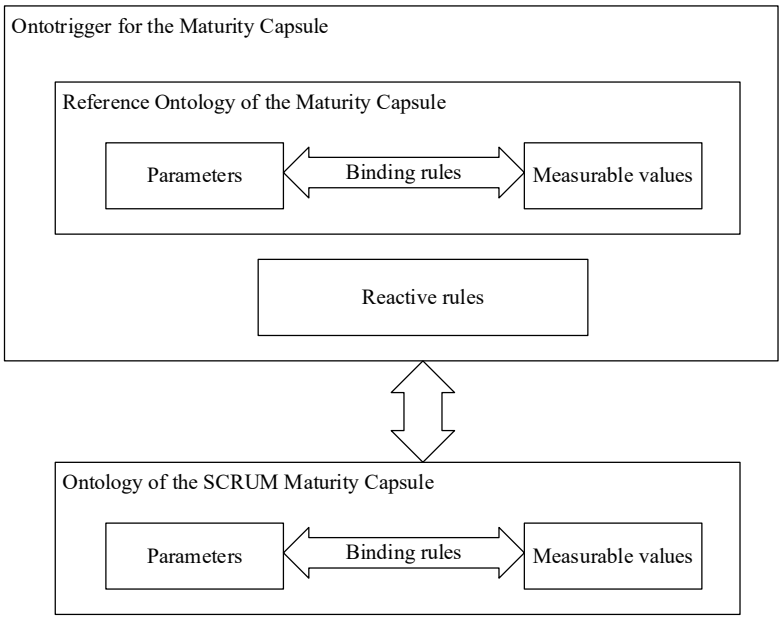


Fig. 3 The mechanism of mapping the model maturity capsule to a SCRUM maturity capsule (ontotrigger implementation)

For simplicity, let us assume a simple mapping between the SCRUM capsule and the general maturity capsule implemented with the use of is-mapped-to:

```
Rule C) Every product-owner is-mapped-to the client.
Rule D) Every development-team is-mapped-to the project.
Rule E) Every scrum-master is-mapped-to the provider.
```

Assuming reactive rules as a pair:

```
Rule F) If a scrum-master is-mapped-to a provider and the scrum-master has-
streamlining-assessment-processes-sprints-level equal-to 2 then the provider
has-service-delivery-level equal-to 1 and the provider has-support-services-
level equal-to 2.

Rule G) If a scrum-master is-mapped-to a provider and the provider has-
maturity-evaluation-level equal-to 3 then the scrum-master has-maturity-
evaluation-level equal-to 3.


To a question
Who-Or-What is described by Scrum-Capsule-01 and has-maturity-evaluation-level
equal-to 3?
```

The inference engine suggests Gabi. Let us note that Gabi - the Scrum Master, is mapped to the maturity capsule as an abstract supplier (the provider) using Rule C. Then, by applying the reactive rule - Rule F, the characteristics of the measurable values of this abstract supplier are created, and then Rule-A is applied. In turn, the reactive rule - Rule-G performs a backward

mapping of the ontology of the reference capsule to the Scrum capsule, and determines a resulting maturity equal to 3.

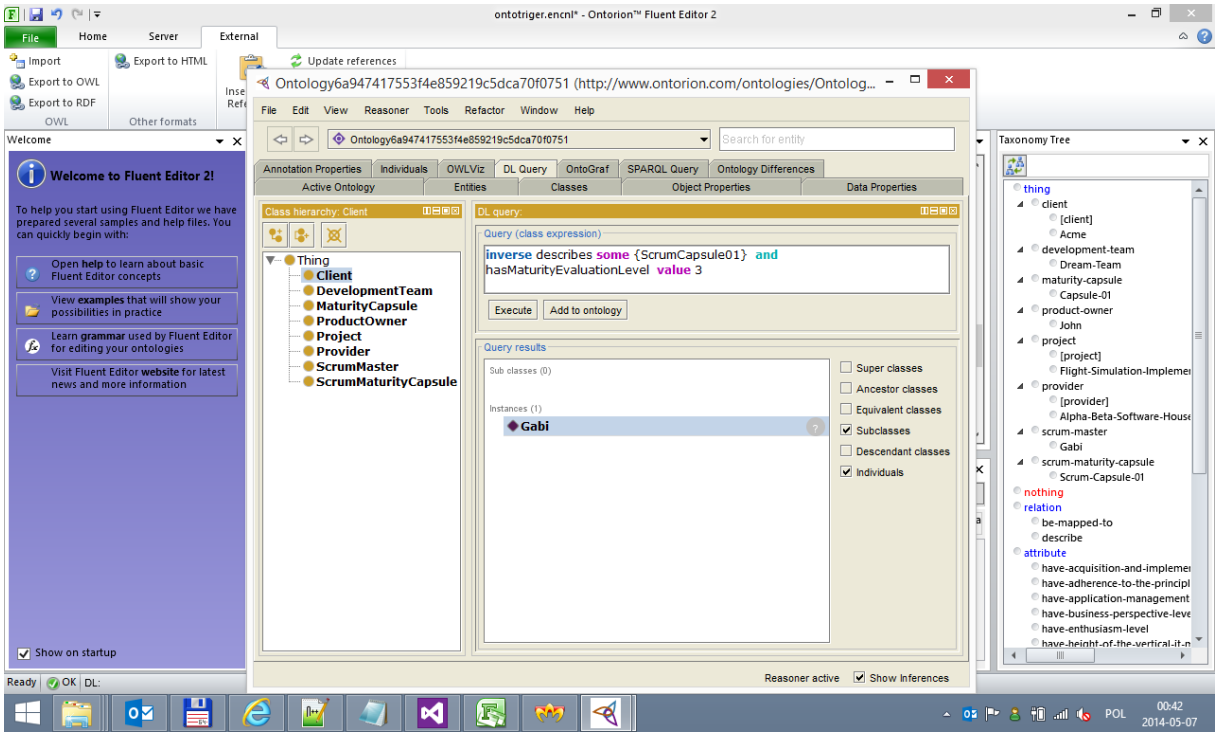


Fig. 4 A sample result of the mechanism mapping the model maturity capsule onto the SCRUM maturity capsule (an example of how the ontotrigger operates)

## 5. Conclusions

The aim of this work was to present the structure of an ontotrigger and its application in the process of mapping maturity capsules. For this purpose, the description of both capsules, as well as the construction processes of both ontologies were discussed. With both entities and their ontologies, the ontotrigger could be applied to the attempt of mapping them. Such a work layout resulted from the need to justify the use of ontotriggers as well as the possibility of their verification.

In the verification processes, a system was adopted which allows work with many independent ontologies. The ontology of a maturity capsule was used to justify that the ontotrigger always works in the context of a selected ontology. Other ontologies, such as the Scrum ontology, can then benefit from the ontology of the maturity capsule. The capsule is used indirectly. Whenever another ontology (e.g. Scrum) undergoes modification, it is also checked whether the fulfilled premise is mapped (through the Analogy Function). If it is the case, the Consequence is an inverse mapping to the Scrum Ontology, resulting in a modification in it.

It was also revealed how the presence of hidden knowledge (from the point of view of the Scrum Ontology), enclosed in the Reference Ontology, has real consequences for the Scrum Ontology via the ontotrigger.

## 6. Literature


1. Aufaure M.A., Legrand B., Soto M., Bennacer N., Metadata- and Ontology-Based Semantic Web Mining. W: Taniar D., Rahayu J.W. (Red.). Web Semantics Ontology. Idea Group Publishing, 2006
2. Czarnecki A., Orłowski C., Sitek T., Ziółkowski A.: Information Technology assessment Using a Functional Prototype of the Agent Based System// Foundations of Control and Management Sciences. - 2008, No 9, S. 7-28: 5 Fig., 5 Tab. - Bibliogr. 8 Pos. - ISSN 1731-2000
3. Czarnecki A., Orłowski C.: IT Business Standards as an Ontology Domain// Lecture Notes In Artificial Intelligence. - 2011 S. 582-591: 1 Fig. - Bibliogr. 10 Pos. - ISBN 978-3-642-23934-2 - ISSN 0302-9743
4. Czarnecki A., Orłowski C.: Ontology as a Tool for IT Management Standards Support// Lecture Notes in Artificial Intelligence. – 2010, No 6071, S. 330-339: 3 Fig. - Bibliogr. 14 Pos. - ISSN 0302-9743
5. Goczyła K., Ontologie W Systemach Informatycznych. Akademicka oficyna Wydawnicza EXIT (in Polish), Warszawa 2011
6. Gruber T.R., a Translation Approach to Portable Ontology Specifications. W: Knowledge Acquisition. 5(2):199–220, 1993
7. Hai Bang Truong, Trong Hai Duong, Ngoc Thanh Nguyen: A Hybrid Method For Fuzzy Ontology Integration. Cybernetics and Systems 44(2-3): 133-154 (2013)
8. Hepp M., Ontologies: State of the Art, Business Potential, and Grand Challenges. W: Ontology Management: Semantic Web, Semantic Web Services, and Business Applications. Hepp W., De Leenheer P., De Moor A., Sure Y. (Red.), Springer, New York, 2006, Ss. 3–22
9. Kowalczuk Z., Orlowski C.,: Advanced Modeling of Management Processes in Information Technology. Studies in Computational Intelligence 518, Springer 2014, ISBN 978-3-642-40876-2, pp. 1-203
10. Pastuszak J., Stolarek M., Orłowski C.: Service and Service Decomposition Model - Theoretical Foundation of IT Service Management// Foundations of Control and Management Sciences. - 2009, Nr 12, S. 59-73: 6 Fig., 3 Tab. - Bibliogr. 15 Pos. - ISSN 1731-2000
11. Pastuszak J., Stolarek M., Orłowski C.: Service and Service Decomposition Model - Theoretical Foundation of IT Service Management// Foundations of Control and Management Sciences. - 2009, Nr 12, S. 59-73: 6 Fig., 3 Tab. - Bibliogr. 15 Pos. - ISSN 1731-2000
12. Pietranik M, Nguyen Ngoc Thanh: Preliminary Experimental Results of the Multi-Attribute and Logic-Based Ontology Alignment Method. ICCCI 2013: 225-234
13. Sitek T., Orłowski C.: Model of Management of Knowledge Bases in the Information Technology Evaluation Environment // W: Information Systems Architecture and Technology: Models of the Organisation's Risk Management / Editors: Zofia Wilimowska [Et Al.]. - Wrocław 2008: Oficyna Wydawnicza Politechniki Wrocławskiej, 2008. - (Biblioteka Informatyki Szkół Wyższych). - S. 221-231: 5 Fig. - Bibliogr. 9 Pos. - ISBN 978-83-7493-413-8
14. Kaplanski P. “Controlled English Interface for Knowledge Bases”. Studia Informatica, Formerly: Zeszyty Naukowe Politechniki Śląskiej, seria INFORMATYKA, Volume 32, Number 2A (96), PL ISSN 0208-7286, QUARTERLY (s. 485-494), 2011.
15. P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph, “OWL 2 Web

Ontology Language Primer," W3C Recommendation, World Wide Web Consortium, October 2009.
16. "SWRL: A Semantic Web Rule Language Combining OWL and RuleML." Made available on 20 April 2014.
17. "Cognitum| Ontorion Semantic Knowledge Management Framework," 2013. Made available on 20 April 2014, http://www.cognitum.eu/semantics/ontorion/.