An Ontology-based Framework for Automated Code Generation of web AR Applications

Lazar Djordjevic, Nenad Petrovic, and Milorad Tosic

***Abstract* — In this paper, we propose an ontology-based framework for automated web AR application code generation leveraging semantic descriptions of application structure and behavior in synergy with domain knowledge and annotations of builidng information model (BIM). Three case studies are presented: smart home energy consumption monitoring application, robotics testbed companion and virtual music mixing interface. The presented approach significantly reduces the time needed for web AR application development.**

***Keywords* — augmented reality, building information modeling, ontology, semantics.**

I. Introduction

With the purpose of enhancing the process of software development, the use of ontologies and semantic representations is becoming advocated by more and more authors in recent years. From requirements specification to design, implementation and maintenance, a significant number of works in this area has shown leveraging ontologies in various phases of software development has many benefits, making it both faster and more effective at the same time [1]-[3].

In most cases, the semantic representations are used to automatize certain steps relying on code generators. For example, in [3], a semantic knowledge base about various phases is used for automated construction of software project model. In [4], software documentation is automatically generated relying on ontologies. On the other side, ontologies were used for automated infrastructure code generation and system maintenance of container-based applications and virtual network functions in [5].

In this paper, an ontology-based approach to design and implementation of augmented reality (AR) applications is adopted. Semantic annotation of building information model (BIM) is used as a domain-knowledge and integrated in location-based AR applications [6], [7]. A framework for automated AR application code generation is proposed making use of ontologies and semantic representations. Three case studies are presented: a smart home application for energy consumption monitoring, robotics testbed companion and virtual music mixing interface. This paper is an extended and revised version of the paper presented at TELFOR 2019 conference [8]. It introduces new case studies and provides more detailed implementation descriptions, more aspects of evaluation together with further discussion of the achieved results. Moreover, in this version, the generated output code targets web AR, while offering wider compatibility and a better performance than the code produced in [8].

Paper received April 15, 2020; revise July 4, 2020; accepted July 17, 2020. Date of publication July 31, 2020. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Miroslav Lutovac.

This paper is an extended version of the paper presented at the 27th Telecommunications Forum TELFOR 2019 [8].

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

Lazar Djordjevic is with the University of Nis, Faculty of Electronic Engineering (e-mail: lazadjole38@gmail.com).

Nenad Petrovic is with the University of Nis, Faculty of Electronic Engineering (e-mail: nenad.petrovic@elfak.ni.ac.rs).

Milorad Tosic is with the University of Nis, Faculty of Electronic Engineering (e-mail: milorad.tosic@elfak.ni.ac.rs).

II. Background

A. Ontologies and semantic technology

The term ontology has been used in different contexts across various scientific fields and research areas. Originally, it referred to a philosophical branch studying the ways, concepts of being and their relations. In computer science, it refers to conceptualization of a certain knowledge domain often in a form of a vocabulary of relevant terms. In this context, ontology is defined as a representational artifact, which designates some combination of universals, defined classes, their properties and relations between them [9]. The formalization of this knowledge representation is made in a form understandable and processable by humans as well as computers.

Ontologies typically consist of classes, individuals, attributes and relations. Classes represent collections, abstract groups or kinds of objects. On the other side, attributes denote properties, characteristics and parameters of the classes. Individuals represent concrete instances or objects assigned a given set of properties defined by the class. Relations define ways in which individuals from different classes can be related to each other. Collection of facts specified according to the conceptualization defined by ontology is often stored separately and is called a semantic knowledge base. Both ontologies and semantic knowledge bases are represented using the RDF (https://www.w3.org/RDF/) standard language in the form of *(subject, predicate, object)* triplets and persisted inside the triple stores. On the other side, SPARQL (https://www.w3.org/TR/rdf-sparql-query/) is a language used for querying RDF semantic triple stores. By executing queries, it is possible to retrieve the results that may support

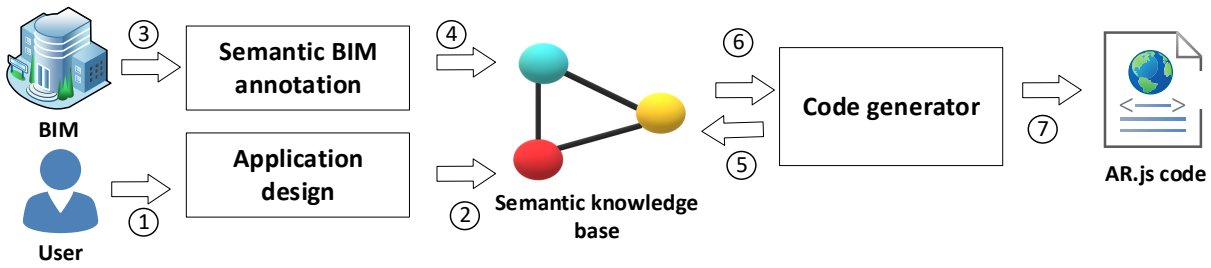

Fig. 1. Architecture and working principle overview: 1-Semantic description of application 2-RDF triplets 3-Building model 4-Semantic annotations of building model 5-SPARQL queries 6-Query results 7-Code generator output.

different reasoning mechanisms in order to infer new knowledge from the set of existing facts.

Regardless of the stage in sotware development process as a knowledge-intensive work, generic as well as domain-specific knowledge can be identified and represented by a corresponding ontology. The following types of ontologies are commonly used in the sotware development process [10]: 1) domain ontologies 2) method ontologies 3) status ontologies 4) process ontologies. Domain ontologies provide a definition of controlled, hierarchical and structured representation of the entities within the considered domain of interest. They are used to annotate data pertaining to entities in order to make the data both more acessable and processable by computers [11]. Method ontologies cover knowledge about possible ways to do some task within the software development process. The role of status ontology is to describe both static and dynamic aspects of a software. The static part describes the structure of a software, while the dynamic part describes its behaviour. The role of process ontology is to describe the components and their relationships that make up a process [10]. This way, they relate context, content and structure of three types of knowledge (enterprise, domain and information knowledge) .

B. Building Information Modeling (BIM)

BIM represents a new methodology for planning, modeling and creation of a comprehensive digital building model. The model itself unifies geometry, spatial relationships, geographical parameters, and technical descriptions of the element. When adopted well, BIM facilitates a more integrated design and construction process that results in better quality buildings, lower cost and reduced project duration. BIM can present the complete lifecycle of a building, from the construction process to various use and maintenance phases [12].

Building information model can be used for various purposes, such as 3D visualization of building, shop drawings, code reviews, cost estimation, construction sequencing, conflict, interference, collision detection, forensic analysis and facilities management [13]. As one of the most used software tools for creation and manipulation with building information model particularly stands out *Autodesk Revit* (https://www.autodesk.com/products/revit/overview).

Semantic BIM annotations or semantic mark-up implies tagging and classification of building's elements and resources with respect to a particular ontology, making these resources comprehensible to both humans and machines [14]. For semantic annotation of building information models for home and faculties buildings, Building Topology Ontology (BOT) was used. The BOT is a minimal ontology for describing core topological concepts of a building and relationships between the sub-components of a building [15]. Building information models were created in Revit. For data access and automation of annotation, Revit API add-in was used.

C. AR.js

Web is a promising platform for lightweight and cross-platform augmented reality services and applications [16]. The main drawbacks of vendor-specific mobile augmented reality platforms are the necessity of additional hardware accessories for smartphones, their high cost and a small set of compatible devices [16].

AR.js (https://github.com/AR-js-org/AR.js) is an open-source JavaScript library for development of augmented reality applications run in a web browser. It provides wide compatibility and a solid performance on mobile devices without demanding powerful hardware. Even on older smartphones, it performs at around 60 frames per second [17]. Moreover, its ideology is quite simple allowing the development of full augmented reality applications in just tens lines of HTML and JavaScript code by using it [17]. On the other side, it includes a huge set of built-in features and components relevant to the development of AR applications, such as 3D object loader, position/location handling and different camera marker presets (QR and barcodes). Therefore, we decide to use JavaScript/HTML relying on AR.js as a target code generated by the code generator leveraging the semantic representations defined according to the corresponding ontologies.

III. Related Work

There are several existing works that leverage ontologies and semantics in AR applications. In [18], formal semantics was used to enable context-awareness within AR systems, illustrated in cases of tourist guide and cultural heritage mobile applications. Moreover, in [19], an ontology was used to support the development of dynamic AR presentations based on semantic descriptions, in a contextual manner, which was demonstrated on the example of faculty staff information application. However, its usability was limited to Unity 3D plugin.

In [20], a tool for generation of e-learning web AR applications based on reusable pre-defined components was presented. On the other side, [7] provides an overview of how BIM can be used for generation of historical reconstruction AR applications.

In our solution, ontological representation of various AR application aspects together with semantic BIM annotations are used for automated code generation. This way, it provides platform-independent knowledge representation that enables high reuse and better extendability. However, compared to other solutions, our approach to automated AR application generation is much more versatile and general, which is confirmed by the heterogeneity of the presented case studies. In [21], we have adopted a similar approach to generation of interactive 3D applications and approved it as quite effective.

IV. Implementation

A. Framework overview

In Fig. 1, an illustration of the proposed framework's architecture and working principle is given.

The first step of AR application generation is application design. User creates semantic descriptions of the desired application manually with respect to ontologies from the semantic framework. During that process, the corresponding triplets are inserted to the semantic triple store, relying on an auxiliary tool [22].

After that, if it is required by the application design itself, the corresponding building information model can be imported. Moreover, the imported model is parsed and semantically annotated. Once the triplets about both the application design and corresponding building information model become available, a code generation process can be executed. During this step, SPARQL queries are executed in order to retrieve the data necessary for the code generation algorithm from the semantic triple store. The retrieved data is used to parametrize the corresponding AR.js code templates (such as event handler or 3D model loader) which are appended to the output file. The final product of code generation process is HTML page with JavaScript code and specific elements relying on the capabilities of AR.js library.

B. Ontologies

The ontologies used within the proposed framework are classified as follows (see Fig. 2): 1) AR application ontology 2) domain-specific ontologies 3) process ontologies 4) building topology ontologies.

AR application ontology (shown in Fig. 3) is crucial for the description of application design aspects. The triplets from application design phase are stored according to this ontology and they are obligatory for code generation. It is inspired by the general AR application pattern presented in [23]. Its main role is to describe both the structure and behavior of AR application which is leveraged within semantic-driven code generation process. As it can be seen, there are several different types of possible inputs in AR applications which are considered: markers (QR code, barcode or custom), object and face recognition, optical character recognition, spoken phrases and specific position coordinates. Moreover, to each input type, a desired output is mapped, such as appearance of text, 3D models or sound playback. The output appears on a certain part of the screen, such as corner of display, near some target object, etc. However, the generated output can perform some specific actions (like sound looping or 3D model rotation) that stops when a certain event occurs (such as arrival to the target destination, for example). Additionally, the generated output might trigger some external service calls outside AR application that are necessary for the generation of output results, such as energy consumption prediction (in one of the presented case studies), retrieval of specific information and similar. Each external service call has specific parameters that are used as input of some algorithm or procedure executed remotely. However, the other ontologies are optional. Their role is to provide additional infromation about the domain-specific characteristics, such as key concepts, their relations, interactions and underlying processes. Moreover, BOT ontology and its extensions (like FBOT presented in [8]) can be leveraged for location-based AR applications.

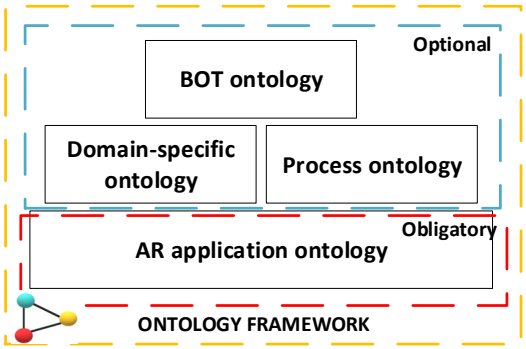

Fig. 2. Ontology framework hierarchy.

C. Code generation algorithm

First, all the possible inputs for AR application with a given id are retrieved. For each specific input i_i, an event handler template is generated. Moreover, a set of outputs $(o_1 \ldots o_n)$ for input i_i is retrieved. For each output o_j that

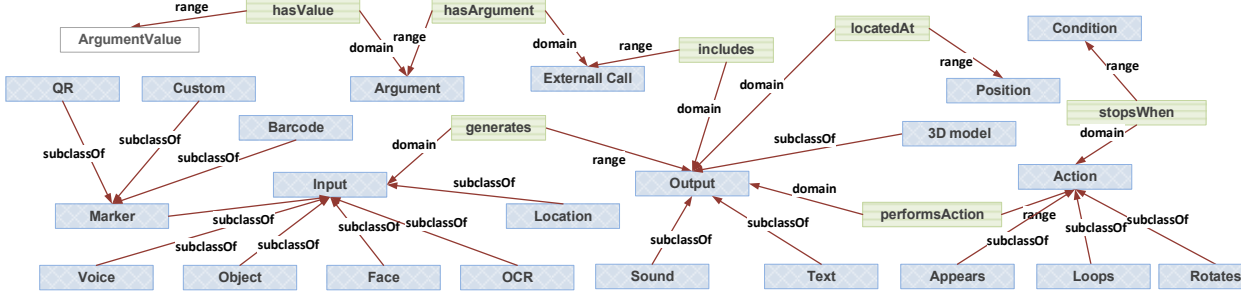

Fig. 3. AR application ontology.

appears at position p_j on the screen, a set of actions $(a_1 \ldots a_m)$ is retrieved, where each action a_k stops when condition c_k is satisfied. The generated event handler template is populated by the code that performs the specified actions that are over once the stopping condition occurs. During the action code generation, if necessary, the action-specific parameters are populated leveraging the external service calls. These remote procedures are used for SPARQL queries against domain and semantically annotated BIMs or invocation of complex calculations (value predictions, classifications) or processes (robot coordination). A pseudocode of AR application code generation procedure is provided in Listing 1. The implementation was done in Java programming language.

```
Input: semantic knowledge base, AR application id
Output: AR.js application HTML/JavaScript code
Steps:
1. Retrieve all inputs for given AR application id;
2. For each input i in inputs
3.     Create event handler template according to type of i;
4.     Retrieve all outputs for i;
5.     For each output o in outputs
6.         For each action a in output.actions
7.             Generate "if(!o.a.stopsWhen.condition)" ;
8.              If(o includes ExternalCall ec)
9.                Generate call for ec.hasArgument.arguments
10.             End if
11.             If(o instance of Text and a instance of Appears)
12.               Generate code for showing text located at position p;
13.             End if
14.             …
15.             If(o instance of Sound and a instance of Loops)
16.               Generate code for looping sound;
17.             End if
18.             If(o instance of 3D model and o instance of Rotates)
19.               Generate code for loading and rotating 3D model;
20.             End if
21.         End for each
22.     End for each
23.     Append event handler template to output;
24. End for each
25. End
```

Listing 1 AR application code generation algorithm

D. *Semantic annotation of BIMs*

In order to enable semantic annotation of building information models, an auxiliary program was written in C#. First, a model is read using Revit API. After that, the user specifies a list of target parameters which will be extracted and annotated. The model is parsed and queried for values of desired parameters. For each of the matched parameters, the corresponding triplets are inserted according to the BOT ontology. The previously described algorithm is given as a pseudocode in Listing 2.

```
Input: BIM model, list of parameters
Output: semantic annotation of BIM
Steps:
1. Load model using Revit API;
2. Parse model;
3. For each element in model.elements
4.     if (element is in list of parameters)
5.     InsertTriplet(parameter.Name, hasValue, element.value);
6. End for each
7. End
```

Listing 2. Semantic annotation of BIM model

V. CASE STUDIES

A. *Smart home energy consumer monitoring*

In each room, it is possible to see the average daily electric energy consumption of household devices by turning a camera to the corresponding marker. Moreover, the temperature of room will be shown on top of camera preview screen. For large heating devices, an alert will be shown if it is overheating or causing some anomalies to the electric grid. In case that energy spending is becoming higher than predicted, a message also appears. The capabilities of energy consumption prediction and electric signal anomaly detection rely on a data-driven framework for energy efficiency presented in [24]. Moreover, if the temperature is too low, an alert telling that adjustment for a certain number of degrees is needed will pop up. Furthermore, over the device's marker sticker, a 3D cube appears allowing the users to select the desired action as a response, such as turning off the consumer device or increasing/decreasing the heat level. The desired action is triggered by touching the cube. The domain ontology behind the application presented in this case study is shown in Fig. 4.

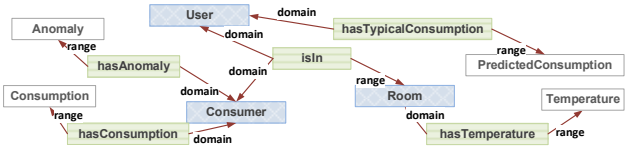

Fig. 4. Domain ontology for smart home energy consumption monitoring application.

A screenshot of the generated AR application for smart home energy consumer monitoring is displayed in Fig. 5.

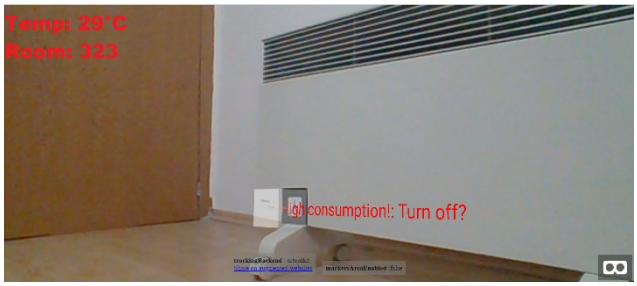

Fig. 5. Smart home energy consumer monitoring AR application screenshot.

B. *Robotics testbed companion*

AR is becoming an emerging direction in intuitive robot programming [25]. This case study builds upon our previous work on a semantic approach to the coordination of robotic IoT devices within experimentation testbeds [26]. This AR application serves as a companion to new users of robotics testbed, providing a simple interface for robot coordination in specific situations.

Each robot has two markers. The first one is used to check the list of available sensors and related capabilities. The second shows a set of possible coordination scenarios which appears near the 3D cube above the robot's marker. For example, if fire was detected in a laboratory room, a coordination scenario would be to send a camera-equipped

robot to observe the endangered area from distance, while another one without a camera would just avoid fire area to arrive at a safe place. The desired scenario is confirmed by touching the cube. An excerpt of semantic framework adopted from [26], extended and used as a process ontology in this case study is given in Fig. 6.

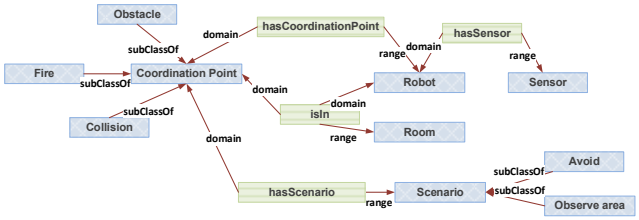

Fig. 6. An excerpt from robot coordination process ontology.

In Fig. 7., an excerpt from 2D view of semantically annotated faculty building model with respect to FBOT ontology [8] obtained as a result of BIM model processing used in case studies *A* and *B* is given.

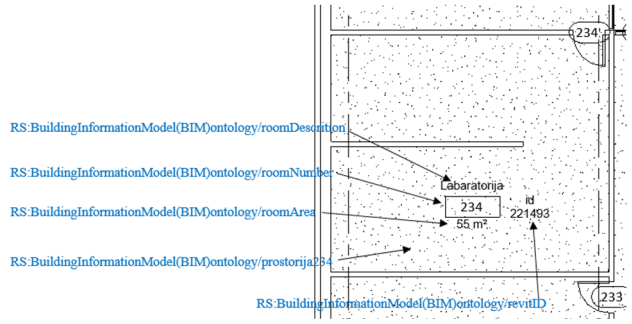

Fig. 7. Semantic annotation of faculty building model.

In Fig. 8, a screenshot of the descirbed application is given.

Fig. 8. Screenshot of robotics testbed companion app.

C. *Music mixing interface*

Each technological leap brought us new ways to both create and perform music [27]. It was also the case with augmented reality [27, 28]. In this paper, we generated an AR-based interface for loop sampler which enables live music mixing. The interface consists of barcode markers that are printed on paper. By pointing camera to the paper, 3D cubes in different colors appear in front of the markers. A sound loop is allocated to each marker. The corresponding sound is included or excluded from the generated music mix by placing hands or fingers over the corresponding barcode marker. When a loop is stopped, the cube disappears. A screenshot of the generated music interface is given in Fig. 9. In our application, there were three numeric barcode markers, as it can be seen. The first one triggers a drum loop, the second triggers a synth line, while the third is used for playing the arpeggio sequence.

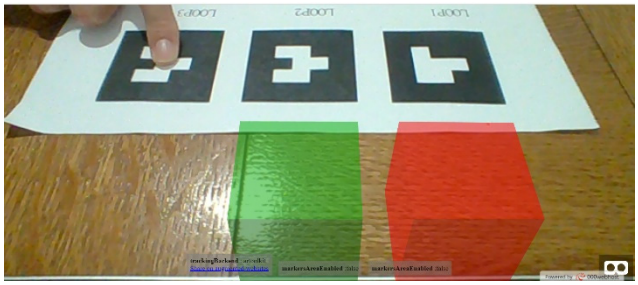

Fig. 9. AR-based music mixing interface screenshot.

VI. Evaluation

In Table 1, the evaluation results for previously described case studies are given. The evaluation was performed on a laptop equipped with Intel i7 7700HQ CPU, 1TB HDD and 16GB of DDR4 RAM. Quantitative evaluation was performed by comparing the time needed for automated code generation leveraging ontologies against the manual development procedure. The goal of evaluation is to show that the proposed approach speeds up AR application development.

The first column denotes the considered case study. The next column contains time in seconds required for triplet insertion describing the AR application design. The third column is the time spent for building model annotation (if applicable). The fourth column shows the average time needed to generate the target AR application using the previously inserted semantic descriptions in seconds (based on 100 runs). The fifth column represents overall code generation time which is a sum of the second, third and fourth column. The next column shows the approximate time spent for manual development of the considered AR application. Finally, the last column represents the speed-up achieved when the proposed approach for automated AR application generation was used compared to manual development time, calculated as Manual/Total.

Table 1: Evaluation results.

Case	Triplet insert [s]	BIM ann. [s]	Code gen. [s]	Total [s]	Manual [hours]	Speed-up [times]
A	946	1.46	8.43	955.89	6	22.60
B	872		6.97	880.43	5.5	22.49
C	487	-	2.31	489.31	4	29.43

According to the achieved results, it can be concluded that the proposed approach significantly reduces the time needed for development of AR web applications, despite the fact that triplet insertion is still done manually. In all the case studies, the overall application generation time relying on the proposed approach does not exceed 16 minutes, while it takes from 4 to 6 hours for manual development of the equivalent software. The code generation was the fastest in the case study *C*, as it only relies on AR application ontology and does not involve semantic BIM annotation

and external calls.

Moreover, in Table 2, a summarized overview of semantic descriptions used for code generation in the presented case studies is shown. For each case study, a set of possible inputs and corresponding outputs is given, together with actions performed by the output, position where output occurs and stop conditions. It serves to illustrate the complexity of case study applications.

TABLE 2: SEMANTIC DESCRIPTION OF CASE STUDY APPLICATIONS.

	In	Out	Action	Pos.	Stop
A	Marker: custom	Text: consumption	External: turn-off	Marker	Marker out of view
		Text: anomaly			
		3D model: cube			
A	Location: office	Text: temperature	External: get temp	Top-left	Location out of room
B	Marker: custom1	Text: sensors	Ext.: get sensors	Marker	Marker out of view
B	Marker: custom2	Text: scenarios	Ext.: get scenarios	Marker	Marker out of view
		3D model: cube	External: select scenario		
C	Marker: $barcode_i$ i=1,2,3	Sound: $loop_s$ s=drum, key, arpeggio	Loop plays	-	Marker hidden
		3D model: $cube_c$ c=blue, red, green			

VII. CONCLUSION AND FUTURE WORK

In this paper, an ontology framework is proposed to support automatic software code generation in AR web applications. It is based on seamless integration of domain-specific data represented by BIM into the software development process. Accoridng to the results, the proposed automated approach promises to significantly speed up the development of AR web applications. In our future work, we will expand the knowledge-intensive work to other types, such as assessment in e-learning. We would like to integrate the proposed framework with ontologies for assessment in biomedical education [29] and examine its potential within educational applications. That would enable novel usage scenarios, such as showing an automatically generated question of selected difficulty about human anatomy when a marker on a physical human model is detected by the AR application.

REFERENCES

[1] D. Gašević, N. Kaviani, M. Milanović, "Ontologies and Software Engineering", in *Handbook on Ontologies*, Springer, pp. 593-615, 2009. https://doi.org/10.1007/978-3-540-92673-3_27

[2] S. Pileggi, A. A. Lopez-Lorca, G. Beydoun, "Ontologies in Software Engineering", *29th Australasian Conference on Information Systems (ACIS2018)*, pp. 1-8, 2018.

[3] W. Hesse, "Ontologies in the Software Engineering Process", *Proceedings of the Workshop on Enterprise Application Integration (EAI 2005)*, pp. 1-13, 2005.

[4] M. P. S. Bhatia, A. Kumar, R. Beniwal, Ontology Driven Software Development for Automated Documentation, Webology, Volume 15, Number 2, December, 2018, pp. 86-112, 2018.

[5] N. Petrovic, M. Tosic, "SMADA-Fog: Semantic model driven approach to deployment and adaptivity in Fog Computing", *Simulation Modelling Practice and Theory*, 102033, pp. 1-25, 2019. https://doi.org/10.1016/j.simpat.2019.102033

[6] E. Patti et al., "Information Modeling for Virtual and Augmented Reality", *IT Professional* 19(3), pp. 52-60, 2017.

[7] L. Barazzetti, F. Banfi, "Historic BIM for Mobile VR/AR Applications", in *Mixed Reality and Gamification for Cultural Heritage*. Springer, pp. 271-290, 2017.

[8] L. Djordjevic, N. Petrovic, M. Tosic, "Ontology based approach to development of augmented reality applications", *2019 27th Telecommunications Forum (TELFOR)*, Belgrade, Serbia, pp. 1-4, 2019. https://doi.org/10.1109/TELFOR48224.2019.8971208

[9] T. Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing", *International Journal Human-Computer Studies* 43 (5-6), pp. 907-928, 1995.

[10] H.-J.Happel, S. Seedorf, "Applications of ontologies in software engineering" , ISWC 2006, pp. 1-14, 2006.

[11] N. Mavetera, J. Kroeze, "An ontology-driven software development framework", *Business Transformation through Innovation and Knowledge Management: An Academic Perspective*, pp. 1713-1724, 2010.

[12] C. Eastman et al., *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. John Wiley & Sons, 2011.

[13] S. Azhar, "Building information modeling (BIM): Trends, benefits, risks, and challenges for the AEC industry.", *Leadership and management in engineering* 11.3 (2011), pp. 241-252, 2011.

[14] E. Oren et al., "What are semantic annotations", Relatório técnico. DERI Galway 9 (2006): 62, pp. 1-15, 2006.

[15] M. H. Rasmussen, et al. "BOT: the Building Topology Ontology of the W3C Linked Building Data Group", Semantic Web, 2019.

[16] X. Qiao, P. Ren, S. Dustdar, L. Liu, H. Ma, J. Chen, "Web AR: A Promising Future for Mobile Augmented Reality--State of the Art, Challenges, and Insights", *Proceedings of the IEEE*, pp. 1–16, 2019. https://doi.org/10.1109/jproc.2019.2895105

[17] J. Etienne, Creating Augmented Reality with AR.js and A-Frame [online]. Available: https://aframe.io/blog/arjs/

[18] T. Matuszka, A. Kiss, W. Woo, "A Formal Model for Context-Aware Semantic Augmented Reality Systems", DAPI 2016, pp. 91-102, 2016. https://doi.org/10.1007/978-3-319-39862-4_9

[19] D. Ruminski, K. Walczak, "Semantic model for distributed augmented reality services", Web3D '17, pp. 1-9, 2017.

[20] A. Rodrigues, D. Dias, V. Farinazzo, P. Bressan, M. De Paiva "WebAR: a web-augmented reality-based authoring tool with Experience API support for educational applications", UAHCI 2017, pp. 118-128, 2017.

[21] N. Petrovic, M. Tosic, "Ontology-Driven Generation of Interactive 3D Worlds", INFOTEH 2020, pp. 1-5, 2020. https://doi.org/10.1109/INFOTEH48170.2020.9066274

[22] M. Tosic, I. Seskar, F. Jelenkoivc, "TaaSOR – Testbed-as-a-Service Ontology Repository", Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 44., Springer, 2012, pp. 419-420.

[23] A. MacWilliams, T. Reicher, G. Klinker, B. Bruegge, "Design Patterns for Augmented Reality Systems", MIXER '04, pp. 1-8, 2004.

[24] N. Petrović, Đ. Kocić, "Data-driven Framework for Energy-Efficient Smart Cities", Serbian Journal of Electrical Engineering, Vol. 17, No. 1, Feb. 2020, pp. 41-63, 2020. https://doi.org/10.2298/SJEE2001041P

[25] S. Blankemeyer, R. Wiemann, L. Posniak, C. Pregizer, A. Raatz, Intuitive Robot Programming Using Augmented Reality, 7th CIRP Conference on Assembly Technologies and Systems, pp. 155-160, 2018.

[26] V. Nejkovic, N. Petrovic, M. Tosic, N. Milosevic, "Semantic approach to RIoT autonomous robots mission coordination", Robotics and Autonomous Systems 126:103438, pp. 1-19, 2020. https://doi.org/10.1016/j.robot.2020.103438

[27] I. Poupyrev, R. Berry, M. Billinghurst, H. Kato, K. Nakao, L. Baldwin, and J. Kurumisawa, "Augmented Reality Interface for Electronic Music Performance", HCI International 2001, pp. 805-808, 2001.

[28] K. Nishida, A. Yuguchi, K. Jo, P. Modler, M. Noisternig, "Border: A Live Performance Based on Web AR and a Gesture-Controlled Virtual Instrument", *Proceedings of the International Conference on New Interfaces for Musical Expression*, pp. 43-46, 2019.

[29] M. Radović, M. Tošić, D. Milošević, "An ontology based approach to assessment in medical education", eLearning-2017, pp. 46-51, 2017.