Vis Comput (2015) 31:1287–1306
DOI 10.1007/s00371-014-1011-9

ORIGINAL ARTICLE

Conceptual knowledge-based modeling of interactive 3D content

Jakub Flotyński · Krzysztof Walczak

Published online: 8 August 2014

Abstract Three-dimensional content offers a powerful medium enabling rich, interactive visualization in virtual and augmented reality systems, which are increasingly used in a variety of application domains, such as education, training, tourism and cultural heritage. The creation of interactive 3D presentations is typically a complex process covering diverse aspects of the content such as geometry, structure, space, appearance, animation and behavior. Recent trends in the development of the semantic web provide new opportunities for simplifying 3D content creation, which may be performed at different levels of abstraction and may encompass the inference of hidden knowledge, which may influence the created content. However, the available approaches to 3D content creation do not enable conceptual knowledge-based modeling of 3D content. The main contribution of this paper is an approach to semantic creation of 3D content. The proposed solution leverages the semantic web techniques to enable conceptual, knowledge-driven content creation. The proposed approach has been implemented and evaluated. It has been shown that the approach can significantly simplify modeling of advanced 3D content presentations in comparison with the available approaches.

Keywords 3D web · Semantic web · 3D content · Ontology · Semantic 3D · Virtual and augmented reality

This research work has been supported by the Polish National Science Centre Grant No. DEC-2012/07/B/ST6/01523.

J. Flotyński (✉) · K. Walczak
Department of Information Technology, Poznań University of Economics, Niepodległości 10,
61-875 Poznañ, Poland
e-mail: jakub.flotynski@ue.poznan.pl

K. Walczak
e-mail: krzysztof.walczak@ue.poznan.pl

1 Introduction

Widespread use of interactive 3D technologies has been recently enabled by the significant progress in hardware performance, the rapid growth in the available network bandwidth as well as the availability of versatile input-output devices. The 3D technologies have become increasingly popular in various application domains on the web, such as education, training, tourism, entertainment, social media and cultural heritage, significantly enhancing possibilities of presentation and interaction with complex data and objects. The primary element of any VR/AR system, apart from interface technologies, is interactive 3D content. Dependencies between components of interactive 3D content may include, in addition to its basic meaning and presentation form, also spatial, temporal, structural, logical and behavioral aspects. Hence, creating and composing interactive 3D content on the web are more complex and challenging tasks than in the case of typical web resources.

The potential of VR/AR applications accessible on the web can be fully exploited only if the interactive 3D content is created with efficient and flexible methods, which conform to the recent trends in the development of the web. In 2001, the W3C (World Wide Web Consortium) initiated the research on the semantic web, which aims at the evolutionary development of the current web towards a distributed semantic database linking structured content and documents. Semantic description of web content makes it understandable for both humans and computers, achieving a new quality in building web applications that can "understand" the meaning of particular components of content as well as their relationships, leading to much better methods of creating, searching, reasoning, combining and presenting web content.

The semantic web consists of content described with common schemes, ontologies and knowledge bases, which specify the meaning of particular content components at different levels of abstraction. In particular, in the domain of computer graphics, content may be described using concepts that are specific to 2D/3D modeling as well as concepts that are specific to an application or an application domain. Furthermore, the semantic description of content includes not only the knowledge (content properties, dependencies and constraints) that has been explicitly specified by the content designer, but also knowledge that has not been explicitly specified. Such hidden knowledge may be inferred from the available data in the knowledge discovery process and influence the final form of the content being modeled. The knowledge-based approach to modeling content liberates content developers from the specification of all content elements and the implementation of complex algorithms that specify content elements, e.g., using chains of properties of content elements or setting properties on the basis of multiple constraints on elements. However, although a few approaches have been proposed for semantic modeling of 3D content, they do not provide comprehensive solutions for conceptual knowledge-driven content creation.

The main contribution of this paper is a new approach to semantic creation of 3D content. The proposed solution leverages semantic web techniques to enable content creation by referring to the meaning of particular content components at different levels of abstraction, with regards to content properties, dependencies and constraints, which may be either explicitly specified or dynamically extracted on the basis of the available content representation.

The remainder of this paper is structured as follows: Section 2 provides an overview of the current state of the art in the domain of semantic modeling of 3D content. Sections 3, 4 and 5 present the proposed SEMIC approach, including the semantic 3D content representation and the method of semantic creation of 3D content with a comprehensive, illustrative example of conceptual knowledge-based content creation. In Sect. 6, the implementation of SEMIC is described. In Sect. 7, qualitative and quantitative evaluations of the approach are presented. Section 8 contains a discussion of the approach. Finally, Sect. 9 concludes the paper and indicates the possible directions of future research.

2 State of the art

Numerous works have been devoted to semantic description and semantic modeling of 3D content. The works can be categorized into three groups.

The works in the first group are mainly devoted to describing 3D content with semantic annotations to facilitate access to content properties. In [32], an approach to designing interoperable RDF-based semantic virtual environments, with system-independent and machine-readable abstract descriptions has been presented. In [5,6], a rule-based framework using MPEG-7 has been proposed for the adaptation of 3D content, e.g., geometry and texture degradation as well as filtering of objects. Content can be described with different encoding formats (in particular X3D), and it is annotated with an indexing model. In [36], integration of X3D and OWL using scene-independent ontologies and semantic zones has been proposed to enable querying 3D scenes at different levels of semantic detail. In [29], an approach to semantic description of architectural elements based on the analysis of architectural treaties has been proposed. In [25], searching for semantic correspondences between man-made 3D models and recognizing functional parts of the models has been addressed. In [10,12,14,15], an approach to building semantic descriptions embedded in 3D web content and a method of harvesting semantic metadata from 3D web content have been proposed.

The second group encompasses works devoted to modeling of different aspects of 3D content, including geometry, appearance and behavior. In [23], an ontology providing elements and properties that are equivalent to elements and properties specified in X3D has been proposed. Moreover, a set of semantic properties have been proposed to enable description of 3D scenes with domain knowledge. However, the semantic conformance to X3D limits the possibilities of efficient modification of entire content layers, including multiple components related to a common aspect of the designed content, e.g., appearance or behavior.

In [43–45], a method of creating VR content on the basis of reusable elements with specific roles and behavior has been proposed. The method has been developed to enable 3D content design by non-IT specialists. This solution does not rely, however, on the semantic representation of content. The use of semantic techniques could further facilitate content creation by users who use arbitrarily selected application-specific ontologies and knowledge bases.

In [7,40], an approach to generating virtual environments upon mappings of domain ontologies to particular 3D content representation languages (e.g., X3D) has been considered. The following three content generation stages are distinguished: specification of a domain ontology, mapping the domain ontology to a 3D content representation language, and generation of a final presentation. The solution stresses spatial relations (position and orientation) between objects in the scene. It enables mapping between application-specific objects and 3D content components, but it does not address complex logical relationships between application-specific concepts and 3D content components and properties. In particular, it is not possible to reflect compositions of low-level content properties and relations between content components by high-level (e.g., application-specific) elements (properties, individuals and classes) and combinations of such high-level elements. In addition, this approach does not enable separation of concerns between users involved in the process of modeling content.

In [20], a semantic model of virtual environments based on the MPEG-7 and MPEG-21 standards has been proposed to enable dynamic scaling and adapting the geometry and functions of virtual objects. In [37], an approach to semantic modeling of indoor scenes with an RGBD camera has been presented.

Several approaches have been proposed to enable modeling of 3D content by example. The approach proposed in [19] includes segmentation of 3D models, searching for models with parts that match queries by semantic properties and composition of parts of the models into new models. The approach proposed in [3] enables parameterized exploration and synthesis of 3D models based on semantic constraints such as size, length, contact and symmetry. In [48], an approach to generating 3D models on the basis of symmetric arrangements of other models has been proposed. The approach and the system presented in [9] leverage semantic attributes, which are selected by the user in the content creation process and describe created models with different strengths determining their final form.

Several works have been conducted on modeling behavior of VR objects. The approach proposed in [34,35] facilitates modeling of complex content behavior by providing temporal operators, which may be used for combining primitive behaviors. A rule-based ontology framework for feature modeling and consistency checking has been presented in [47]. In [21], an ontology-based approach to creating virtual humans as active semantic entities with features, functions and interaction skills has been proposed.

Finally, the third group encompasses works that have been devoted to the use of semantic descriptions of 3D content in artificial intelligence systems. The idea of semantic description of 3D worlds has been summarized in [27]. In [39], a review of the main aspects related to the use of 3D content in connection with the semantic web techniques has been provided. In [4], diverse issues arising from combining AI and virtual environments have been reviewed. In [8,30], abstract semantic representations of events and actions in AI simulators have been presented. In [26,28,46], a technique of integration of knowledge into VR applications, a framework for decoupling components in real-time intelligent, interactive systems with ontologies and a concept of semantic entities in VR applications have been discussed. In [38], a camera controlling approach to exploration of virtual worlds in real time by using topological and semantic knowledge has been proposed.

3 The SEMIC approach

Although several approaches have been proposed for semantic modeling of 3D content, they lack general and comprehensive solutions for modeling of interactive 3D content on the web. Recent trends in the development of the web provide new requirements for efficient and flexible content creation, which go beyond the current state of the art in modeling of 3D content.

1. The approach to content creation should enable declarative modeling of 3D content stressing the specification of the results to be presented, but not the way in which the results are to be achieved.
2. Content creation should be supported by discovery of hidden knowledge covering content properties, dependencies and constraints, which are not explicitly specified, but which may be extracted from the explicit data, and which have impact on the modeled content.
3. The approach should enable conceptual modeling of content components and properties at arbitrarily chosen levels of abstraction, including both the aspects that are directly related to 3D content and the aspects that are specific to a particular application or domain.
4. The approach should enable decoupling of modeling activities related to different parts of content representation, enabling separation of concerns between different modeling users with different expertise, who are equipped with different modeling tools, e.g., to facilitate content creation by domain experts who are not IT specialists.
5. Content created with the approach should be independent of particular hardware and software platforms to enable creation of multi-platform 3D content presentations.

In this paper, an approach to *semantic modeling of interactive 3D content* (SEMIC) is proposed. SEMIC covers various aspects of 3D content such as geometry, structure and space, appearance, scene, animation and behavior. In the approach, semantic web techniques are applied to satisfy the aforementioned requirements. SEMIC combines two elements, which have been partially described in previous works. The first element is the *semantic content model* (SCM) proposed in [11,13,17], which provides concepts that enable 3D content representation at different (arbitrarily chosen) levels of abstraction. SCM allows for incorporation of application-specific knowledge in content representations in order to simplify the content creation process. Such application-specific knowledge (at different levels of abstraction/semantics) is modeled using ontologies. The second element of SEMIC is the *semantic content creation method* (SCCM) proposed in [16], which consists of a sequence of steps, in which the concepts of SCM are used to create desirable content. Some of the steps in SCCM need to be performed manually by a human, whereas the other steps are accomplished automatically—by algorithms that transform different content representations, thus enabling transitions between the subsequent steps of SCCM. The particular elements of SEMIC are described in the following sections.

4 The semantic content model

In [13,17], the SCM has been proposed. It is a collection of ontologies that enable semantic representation of 3D content at different levels of abstraction—low-level concrete content representation (CrR), which reflects elements that are directly related to 3D content, and arbitrarily high-level conceptual content representation (CpR), which reflects elements that are abstract in the sense of their final representation—they are not directly related to 3D content (Fig. 1). Both representations are knowledge bases based on semantic concepts (classes and properties), which are used for specifying semantic individuals, their properties and relations between them.

4.1 Concrete representation of 3D content

A concrete representation of 3D content (CrR) is a knowledge base built according to the *multi-layered semantic content model* (ML-SCM), proposed in [17]. The model enables separation of concerns between several layers corresponding to distinct aspects that are specific to 3D content—*geometry layer*, *structure and space layer*, *appearance layer*, *scene layer*, *animation layer* and *behavior layer*.

The model encompasses concepts (classes and properties) widely used in well-established 3D content representation languages and programming libraries, such as X3D, VRML, Java3D and Away3D. The concepts are assigned to different layers depending on their role. The *geometry layer* introduces basic uniform individual geometrical components and their properties, e.g., planes, meshes and coordinates. The *structure and space layer* introduces complex structural components, which assemble geometrical components, allowing for definition of spatial dependencies between them, e.g., position, orientation and size. The *appearance layer* adds appearance to geometrical and structural components, e.g., color, transparency and texture. The *scene layer* extends structural components to navigable scenes with viewpoints. The *animation* and *behavior layers* enrich components, which have been defined in the previous layers, with animation and behavior. The layers are partly dependent—every layer uses its concepts and concepts specified in the lower layers.

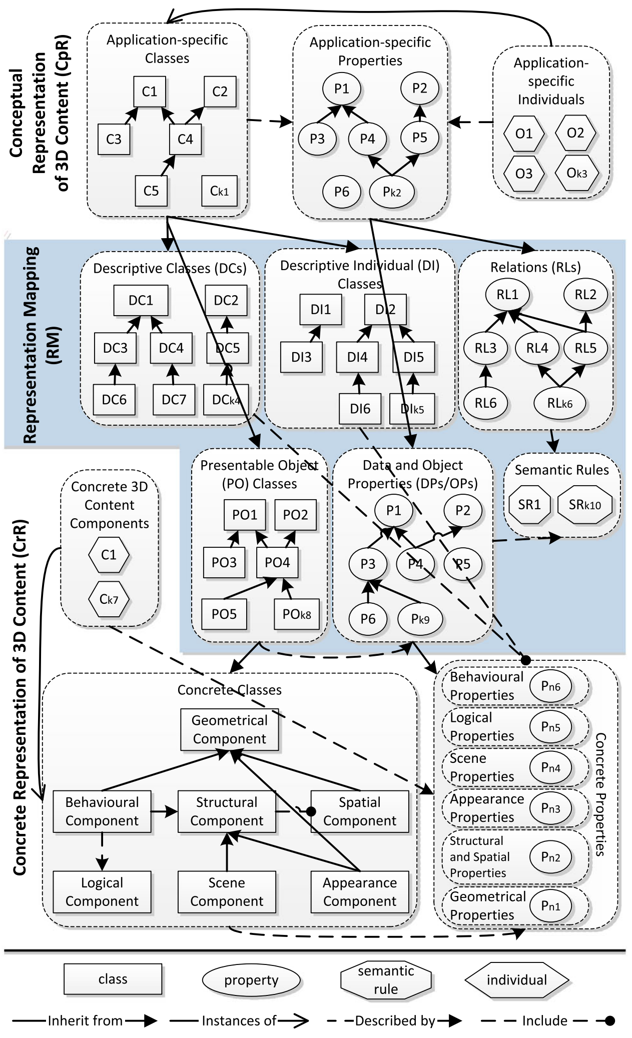

Fig. 1 The semantic content model (SCM)

4.2 Conceptual representation of 3D content

A conceptual representation of 3D content (CpR) is a knowledge base compliant with an application-specific ontology. A CrR consists of application-specific individuals, which are described by application-specific properties. Application-specific concepts may represent the created 3D content at an arbitrarily high (arbitrarily chosen) level of semantic abstraction. The concepts are abstract in the sense of their final presentation, as—in general—they can be presented in various manners (e.g., 2D graphics, 3D models and text). The concepts do not need to cover any aspects that are specific to 3D content, or such aspects do not need to be indicated directly. For instance, an abstract (conceptual) car does not have to be specified as a particular 3D shape, though it may be implicitly considered as such in terms of its final presentation, e.g., by belonging to a particular sub-class of cars (delivery van, limousine, etc.). Various dependencies may be specified for individual concepts using semantic web standards

(RDF, RDFS and OWL [42]), in particular multiple inheritance, restrictions on members of classes as well as domains and ranges of properties. Since various application-specific ontologies may be used in the proposed approach, neither creation nor selection of them is addressed in this paper. CrRs and CpRs are linked by semantic representation mappings (RMs).

4.3 Representation mapping

The goal of mapping is to make application-specific concepts (included in an application-specific ontology), which are abstract in terms of presentation, presentable by the use of the concrete concepts (included in ML-SCM), which are specific to 3D content.

SEMIC does not restrict the acceptable kinds or domains of application-specific ontologies; thus it may be used for creating contents of different types, for different domains and applications. In general, covering a particular application-specific ontology may be difficult, as it may require the specification of a large number of rules corresponding to various cases and contexts of the use of ontological concepts. Therefore, in the SEMIC approach, the general classes of cases and contexts of the use of particular concepts (e.g., domains and ranges of properties) are expected to be precisely specified by the domain experts who will use the ontology for 3D modeling. The purpose is to cover only well-specified cases and contexts of the use of the selected application-specific concepts, but not to cover all (potentially) possible use cases and contexts of all concepts available in the ontology. As the use cases and contexts are already well defined, the application-specific concepts may be mapped to 3D-specific concepts by representation mappings (RMs)—similarly to encapsulating low-level functions behind high-level objects' interfaces in object-oriented programming.

An RM is a knowledge base that links CrRs to CpRs. An RM complies with the *semantic mapping model* (SMM), which has been proposed in [13]. Each mapping assigns concrete representation concepts to application-specific concepts. Linking application-specific classes and properties used in a CpR to particular concepts of a CrR improves efficient modeling and reusability of the application-specific concepts, in contrast to defining individual concrete representations for particular application-specific objects and scenes. Mapping is performed using *mapping concepts*.

The following *mapping concepts* are distinguished in SMM: *presentable objects* (POs), *data properties* (DPs) with *literals*, *object properties* (OPs) with *descriptive individuals* (DIs), *descriptive classes* (DCs) and *relations* (RLs).

Every class from an application-specific ontology whose individuals are primary entities to be presented in the created content, is specified as a *presentable object* (PO) class, e.g., artifacts in a virtual museum exhibition, avatars in an RPG game or UI controls. For each PO class, various concrete representation properties related to geometry, structure and space, appearance, scene, animation and behavior can be specified.

POs may be described by application-specific properties represented by *data properties* (DPs), which indicate application-specific features of the POs (shape, material, behavior, etc.) that may be expressed by literal values (e.g., 'big cube', 'wood', 'flying object').

Descriptors are a functional extension of DPs, as they gather multiple properties of POs. The properties assigned to a *descriptor* do not describe this *descriptor*, but they describe the PO, the particular *descriptor* is linked to—descriptors only carry properties. Unlike POs, *descriptors* do not have individual 3D representations. There are two types of descriptors. *Descriptive classes* (DCs) are application-specific classes that may be assigned to POs to specify some concrete properties of them, e.g., a class of interactive rotating objects includes POs that rotate after being touched—the POs have common concrete properties related to interaction and animation. *Descriptive individuals* (DIs) are instances of classes that are linked to the described POs by *object properties* (OPs). For example, a piece of furniture (a PO) can be made of (an OP) different types of wood (DIs), each of which is described by a few DPs such as color, shininess, texture, etc.

A *relation* (RL) is an application-specific property or an application-specific individual that links different POs occurring in the created content. Every RL has at least two parts (participants), which are connected one to another by mutual dependencies related to some concrete properties of 3D content, e.g., a relation that specifies the relative position of some POs links these POs and determines their relative orientations and distances between them.

5 The semantic content creation method

In [16], the *semantic content creation method* (SCCM) has been proposed. The method enables flexible creation of 3D content at an arbitrarily chosen level of abstraction by leveraging the particular parts of SCM, which has been explained in the previous section. Creation of 3D content with SCCM consists of a sequence of steps, which correspond to different levels of semantic abstraction of the created content—design of a CrR, mapping the CrR to application-specific concepts, design of a CpR, expanding the semantic representation and building the final content representation (Fig. 2). In SCCM, succeeding steps depend on the results of their preceding steps.

The first three steps are performed manually—by a developer or a domain expert. These steps produce knowledge bases that conform to different parts of SCM. These steps

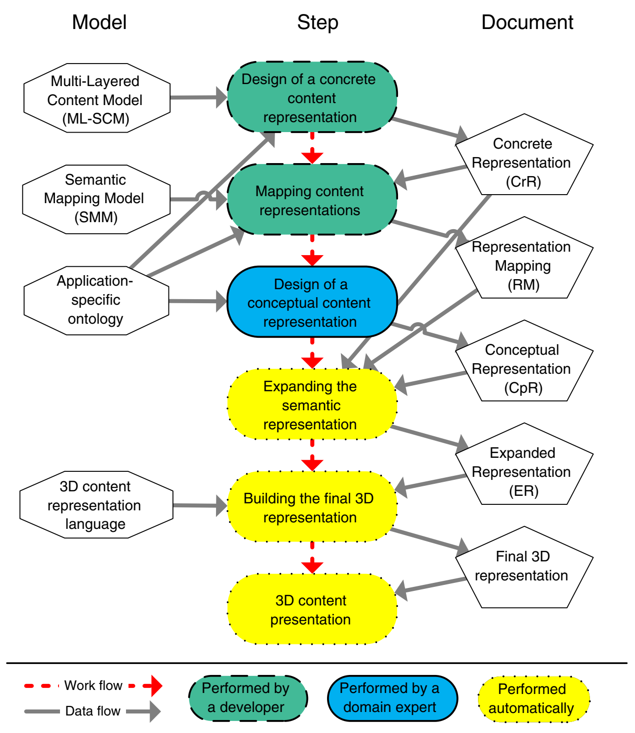
Model
Step
Document
Multi-Layered Content Model (ML-SCM)
Semantic Mapping Model (SMM)
Application-specific ontology
3D content representation language
Design of a concrete content representation
Mapping content representations
Design of a conceptual content representation
Expanding the semantic representation
Building the final 3D representation
3D content presentation
Concrete Representation (CrR)
Representation Mapping (RM)
Conceptual Representation (CpR)
Expanded Representation (ER)
Final 3D representation
Work flow
Data flow
Performed by a developer
Performed by a domain expert
Performed automatically

Fig. 2 Creation of 3D content based on the SEMIC approach

may be performed using a typical semantic editor (e.g., Protégé), however, the development of a specific visual semantic modeling tool is also possible. The other two steps are accomplished automatically. They precede the final 3D content presentation, which may be performed using different 3D content browsers and presentation tools. The following sections describe the subsequent steps of the modeling process, along with an example, in which different 3D content components are created and assembled into a 3D scene of a virtual museum of agriculture.

Step 1: design of a concrete content representation

The design of a CrR provides basic components of 3D content that are a foundation for presentation of application-specific concepts, which will be further used in Step 3. A CrR is a knowledge base compliant with the ML-SCM (cf. Sect. 4.1). The elements of a CrR are concrete components—concrete classes and concrete properties that are directly related to 3D content and whose formation is, in general, complex—it may require the use of additional specific hardware or software tools. For instance, the creation of a 3D mesh requires the use of a 3D scanner or a 3D modeling tool, while drawing a texture requires a 2D graphical editor. The design of a CrR may cover different layers of the ML-SCM, e.g., the design of a mesh is related to the *geometry layer*, while the design of a motion trajectory is related to the *animation layer*. CrRs represent neither particular coherent scenes nor particular compositions of objects, but they include (possibly independent) templates of reusable content components that may be flexibly composed into complex 3D objects and 3D scenes.

In most cases, concrete components need to be designed with regards to the application-specific concepts that are to be presented, e.g., a particular 3D mesh represents specific car models, a particular texture represents wood surface, etc. Hence, they need to be created in collaboration with domain experts, who will further use the components in Step 3.

Every component created in this step is represented by a class or a structure linking classes, which is described by properties. Every class is a subclass of an appropriate ML-SCM class, which is specific to 3D modeling, and it will be used to represent low-level objects in the created content (meshes, materials, viewpoints, events, complex objects, etc.) that have common values of properties. Data and object properties may be specified for classes linking the classes with literal values and other classes. Literal values may be directly used (interpreted) in the content representation (e.g., when reflecting coordinates or color maps) or they may indicate external data sets (e.g., paths to documents including meshes or images). The specification of complex values of data properties is done using external tools, and it is beyond the scope of this paper.

Listing 1 A concrete content representation (CrR)

```
Prefixes:
  Multi−Layered Semantic Content Model (mlscm), CrR (crr)

crr:WomanMesh rdf:type owl:Class ;
  rdfs:subClassOf mlscm:Mesh3D ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:meshData ;
    owl:hasValue "woman.obj" ] .

crr:WoodMaterial rdf:type owl:Class ;
  rdfs:subClassOf mlscm:TextureMaterial ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:texture ;
    owl:hasValue "wood.png" ] ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:transparency ;
    owl:hasValue 0 ] .

crr:GlassMaterial rdf:type owl:Class ;
  rdfs:subClassOf mlscm:ColorMaterial ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:color ;
    owl:hasValue "green" ] ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:transparency ;
    owl:hasValue 0.7 ] .

crr:PaintedWomanMash rdf:type owl:Class ;
  rdfs:subClassOf crr:WomanMesh ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:texture ;
    owl:hasValue "statueTexture.png" ] ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:textureCoordinates ;
    owl:hasValue "..." ] .

{crr:GranaryMesh, ..., crr:BadgeMesh}
  rdf:type owl:Class ;
  rdfs:subClassOf mlscm:Mesh3D ,
  [ rdf:type owl:Restriction ;
    owl:onProperty mlscm:meshData ;
    owl:hasValue "..." ] ,
```

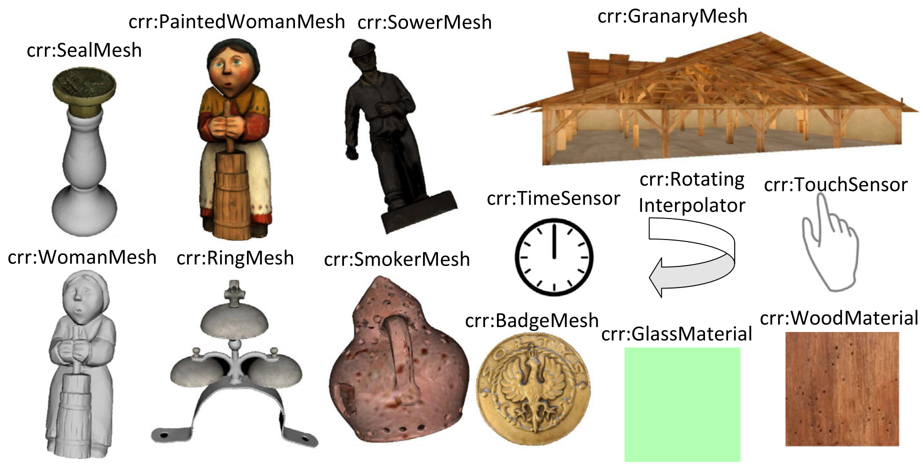

Fig. 3 An example of concrete content components

```
43     [ rdf:type owl:Restriction ;
44       owl:onProperty mlscm:texture ;
45       owl:hasValue "..." ] , ... .
46
47   crr:TouchSensor rdf:type owl:Class ;
48     rdfs:subClassOf mlscm:TouchSensor ,
49     [ rdf:type owl:Restriction ;
50       owl:onProperty mlscm:activates ;
51       owl:someValuesFrom crr:RotatingInterpolator].
52
53   crr:RotatingInterpolator rdf:type owl:Class ;
54     rdfs:subClassOf mlscm:OrientationInterpolator ,
55     [ rdf:type owl:Restriction ;
56       owl:onProperty mlscm:key ;
57       owl:hasValue "0 1.5 3" ] ,
58     [ rdf:type owl:Restriction ;
59       owl:onProperty mlscm:keyValue ;
60       owl:hasValue "0 0 0 0 3.14 0 0 0 0" ] ,
61     [ rdf:type owl:Restriction ;
62       owl:onProperty mlscm:controller ;
63       owl:someValuesFrom crr:TimeSensor ] .
64
65   crr:TimeSensor rdf:type owl:Class ;
66     rdfs:subClassOf mlscm:TimeSensor,
67     [ rdf:type owl:Restriction ;
68       owl:onProperty mlscm:interval ;
69       owl:hasValue 3 ] .
```

This step of modeling is typically performed by a developer with technical skills in 3D modeling, who is equipped with a specific additional software (e.g., a 3D modeling tool) or hardware (e.g., a 3D scanner) for creating visual (2D or 3D), haptic or aural elements.

Listings 1–5 and Figs. 3–5 show a simplified example of using SEMIC for conceptual knowledge-based 3D content creation. In Step 1 in the example, a developer creates several virtual objects, which represent real museum artifacts. First, the developer uses specific modeling tools to create graphical elements required for low-level content representation—a 3D scanner to capture the geometry of: a granary, a woman statuette, a sower, a smoker, a ring, a seal and a badge, and a 2D graphical tool—to prepare textures for selected models (Fig. 3). Second, the developer uses a semantic modeling tool (which may be a plug-in to a modeling package, e.g., Blender or 3ds Max), to create a CrR, which semantically reflects the created graphical elements. Listing 1 presents an example CrR encoded in the RDF Turtle format. Some content components and properties that are not crucial for the presented example are skipped. The prefixes used in the listing correspond to different semantic models and content representations. For every model and every texture, the tool generates OWL restrictions, which are classes with specific values of properties. The classes will be mapped in Step 2 to 3D objects and materials used by domain experts in Step 3. In the example, the woman statuette is to be used by domain experts in three different forms: as wooden and glassy virtual artifacts (lines 4–26) and as a painted virtual artifact with the inherent texture mapping (28–35). The other models are to be used in single forms (lines 37–45). In addition, the developer creates the `crr:TouchSensor` (47–51), which activates the `crr:RotatingInterpolator` (53–63) controlled by the `crr:TimeSensor` (65–69), which will enable the artifacts in virtual museum scenes to be rotated after being touched.

Step 2: mapping content representations

Mapping a CrR (created in Step 1) to application-specific concepts enables 3D presentation of application-specific knowledge bases (created in Step 3) by linking them to concrete components of 3D content included in the CrR. The result of this step is an RM, which is comprised of *mapping concepts* that inherit from concepts defined in SMM (cf. Sect. 4.3). Mapping is performed once for a particular application-specific ontology and a CrR, and it enables the reuse of concrete components for forming 3D representations of various application-specific knowledge bases, which conform to the application-specific ontology selected. An RM needs to cover all concepts (classes and properties) of the application-specific ontology that need to have representations in conceptually modeled 3D content.

This step of modeling may be performed using a typical semantic editor and it does not require the use of additional (complex) specific 3D modeling hardware or software, e.g.,

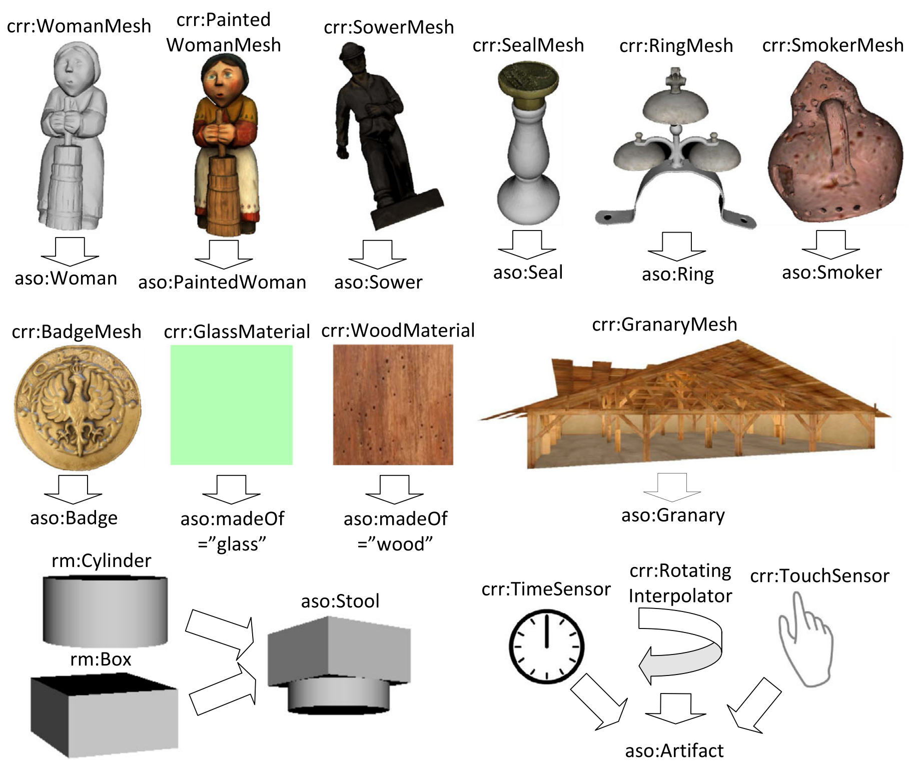

Fig. 4 Example mapping of concrete content components to application-specific concepts

linking a previously created animation to an object, inclusion of sub-objects within a complex structural object, etc. However, in terms of the semantic structures that need to be created, mapping is more complex in comparison to the design of a CrR, and it requires more semantic expressiveness. A specific visual mapping tool could be developed to simplify this step.

This step of modeling is typically performed by a developer or a technician, who is equipped with a semantic editor and has basic skills in semantic modeling.

In Step 2 in the example, a developer or a technician creates an RM (listing 2—the RDF Turtle and Prolog-like syntax) including semantic statements linking application-specific concepts (used in Step 3) to components of the CrR (created in Step 1). The `aso:Woman` (5–6) artifact is a PO class and a subclass of the `crr:WomanMesh`, so it inherits its properties related to the geometry, which were specified in the previous step. Every instance of this class may be made of wood or glass (as indicated by the `aso:madeOf` DP), thus having an appropriate material assigned using proper DCs (7–16). In contrast to wooden and glassy artifacts, every `aso:PaintedWoman` PO has a texture assigned, as indicated by its super-class (17–18). Mapping the other classes of the application-specific ontology to PO classes has been performed in a similar fashion (19–21). Moreover, two basic shapes (the `rm:Box` and the `rm:Cylinder`) are designed (23–32) and assembled into the `aso:Stool` PO class (33–40). Every `rm:Box` and `rm:Cylinder` included in a `aso:Stool` have dimensions and relative positions specified (41–52). To enable rotation of virtual museum artifacts, after being touched,

Listing 2 A representation mapping (RM)

```
Prefixes: application−specific ontology (aso),                          1
  Multi−Layered Semantic Content Model (mlscm),                         2
  Semantic Mapping Model (smm), CrR (crr), RM (rm)                      3
                                                                        4
aso:Woman rdfs:subClassOf                                               5
  smm:PresentableObject , crr:WomanMesh .                               6
{rm:WoodenObject,rm:GlassyObject}                                       7
  rdf:type owl:Class ;                                                  8
  rdfs:subClassOf smm:DescriptiveClass ;                                9
  owl:equivalentClass                                                   10
    [ rdf:type owl:Restriction ;                                        11
      owl:onProperty aso:madeOf ;                                       12
      owl:hasValue "{wood,glass}" ] ,                                   13
    [ rdf:type owl:Restriction ;                                        14
      owl:onProperty mlscm:material ;                                   15
      owl:someValuesFrom {crr:WoodMaterial,crr:GlassMaterial} ] .       16
aso:PaintedWoman rdfs:subClassOf                                        17
  smm:PresentableObject , crr:PaintedWomanMesh .                        18
{aso:Granary,...,aso:Badge} rdfs:subClassOf                             19
  smm:PresentableObject ,                                               20
  {crr:GranaryMesh, ..., crr:BadgeMesh} .                               21
                                                                        22
{rm:Box,rm:Cylinder} rdfs:subClassOf                                    23
  {mlscm:Box,mlscm:Cylinder} ,                                          24
  [ rdf:type owl:Restriction ;                                          25
    owl:onProperty mlscm:size ;                                         26
    owl:someValuesFrom                                                  27
      {rm:BoxSize,rm:CylinderSize} ] ,                                  28
  [ rdf:type owl:Restriction ;                                          29
    owl:onProperty mlscm:position ;                                     30
    owl:someValuesFrom                                                  31
      {rm:BoxPos,rm:CylinderPos} ] .                                    32
aso:Stool rdfs:subClassOf                                               33
 smm:PresentableObject , mlscm:StructuralComponent,                     34
  [ rdf:type owl:Restriction ;                                          35
    owl:onProperty mlscm:includes ;                                     36
    owl:someValuesFrom rm:Cylinder ] ,                                  37
  [ rdf:type owl:Restriction ;                                          38
    owl:onProperty mlscm:includes ;                                     39
    owl:someValuesFrom rm:Box ] .                                40
{rm:BoxSize,rm:CylinderSize} rdf:type owl:Class ;                41
  rdfs:subClassOf mlscm:Vector ,                                 42
  [ rdf:type owl:Restriction ;                                   43
    owl:onProperty x ;                                           44
    owl:hasValue "..." ] ,                                       45
  [ rdf:type owl:Restriction ;                                   46
    owl:onProperty y ;                                           47
    owl:hasValue "..." ] ,                                       48
  [ rdf:type owl:Restriction ;                                   49
    owl:onProperty z ;                                           50
    owl:hasValue "..." ] .                                       51
{rm:BoxPos,rm:CylinderPos} rdf:type owl:Class ; ...              52
                                                                 53
aso:Artifact rdfs:subClassOf                                     54
  smm:DescriptiveClass                                           55
  [ rdf:type owl:Restriction ;                                   56
    owl:onProperty mlscm:sensor ;                                57
    owl:someValuesFrom crr:TouchSensor ] .                       58
                                                                 59
aso:incorporates                                                 60
  rdfs:subPropertyOf smm:BinaryRelation ;                        61
  owl:equivalentProperty mlscm:includes .                        62
                                                                 63
aso:standsOn                                                     64
  rdfs:subPropertyOf smm:BinaryRelation .                        65
{mlscm:x(APos, AX),mlscm:z(APos, AZ)} :−                         66
  aso:standsOn(A, B) ,                                           67
  mlscm:position(A, APos) ,                                      68
  mlscm:position(B, BPos) ,                                      69
  {mlscm:x(BPos, BX),mlscm:z(BPos, BZ)} ,                        70
  {AX=BX,AZ=BZ} .                                                71
mlscm:y(APos, AY) :−                                             72
  aso:standsOn(A, B) ,                                           73
  mlscm:position(A, APos) ,                                      74
  mlscm:position(B, BPos) ,                                      75
  mlscm:y(BPos, BY) ,                                            76
  mlscm:size(B, BSize) ,                                         77
  mlscm:sy(BSize, BSY) ,                                         78
  mlscm:size(A, ASize) ,                                         79
  mlscm:sy(ASize, ASY) ,                                         80
  AY = BY + (ASY + BSY)/2 .                                      81
```

every artifact is linked to a `crr:TouchSensor` using a DC (54–58). Furthermore, two RLs have been specified. The `aso:incorporates` RL is an equivalent to the `mlscm:includes` (60–62), while the `aso:standsOn` RL determines the `x`, `y` and `z` coordinates of the object by semantic rules (64–81). The created mapping is depicted in Fig. 4.

Step 3: design of a conceptual content representation

The design of a CpR enables declarative creation of 3D content at an arbitrary level of abstraction that is permitted by the application-specific ontology selected. This step can be performed multiple times for a particular application-specific ontology, a CrR and an RM when new 3D content is required for a particular, specific 3D/VR/AR application. This step of modeling focuses on application-specific semantic concepts and does not cover concrete components of 3D content, which are hidden behind the RM.

A CpR, which is a knowledge base compliant with the application-specific ontology, consists of semantic statements (facts) and semantic rules (implications), which declaratively represent content at a conceptual level of abstraction. Both the statements and the rules are built upon application-specific concepts and objects. A CpR explicitly specifies properties and relations between content objects as well as constraints on object properties and object relations that will be further used in knowledge discovery in the next step of SCCM. In contrast to CrRs, which include possibly independent components, CpRs reflect coherent 3D scenes or complex 3D objects.

This step is typically performed by a domain expert, who is not required to have advanced technical skills. A domain expert uses an application-specific ontology to focus only on application-specific semantic concepts and does not need to work with concrete components of 3D content. A domain expert may be equipped with a semantic editor. However, a visual semantic modeling tool could be also developed.

In general, this step of modeling is independent of the steps described previously, and a CpR may be created before a CrR and an RM are created, e.g., when a domain expert designs an accurate digital equivalent to a known real object. However, when designing non-existing objects or scenes (e.g., a planned virtual museum exhibition) the availability of the CrR and the RM may be desirable to enable the preview of the results during the modeling.

Listing 3 A conceptual content representation (CpR)

```
Prefixes:                                                        1
  application−specific ontology (aso), CpR (cpr)                 2
                                                                 3
{cpr:granary,...,cpr:badge}                                      4
  rdf:type {aso:Granary,...,aso:Badge} .                         5
{cpr:woodenWoman,cpr:glassyWoman}                                6
  rdf:type aso:Woman ;                                           7
  aso:madeOf "{wood,glass}" .                                    8
cpr:paintedWoman rdf:type aso:PaintedWoman .                     9
                                                                 10
{cpr:stool1,...,cpr:stool8} rdf:type aso:Stool .                 11
                                                                 12
cpr:stoolPositions(Index, N) :−                                  13
  Index<N, aso:Stool(S), cpr:noPosition(S),                      14
  X is Index div 4, prolog:assert(aso:x(S, X)),                  15
  Z is Index mod 4, prolog:assert(aso:z(S, Z)),                  16
  prolog:assert(aso:y(S, 0)),                                    17
  NewIndex is Index+1,                                           18
  cpr:stoolPositions(NewIndex, N).                               19
cpr:noPosition(S) :− aso:x(S, X), !, false.                      20
cpr:noPosition(S).                                               21
                                                                 22
cpr:deployment(A, S) :−                                          23
  aso:Artifact(A), aso:Stand(S),                                 24
  cpr:notStandsOnOthers(A),                                      25
  cpr:nothingStandsOnIt(S),                                      26
  prolog:assert(standsOn(A, S)).                                 27
cpr:notStandsOnOthers(A) :−                                      28
  cpr:standsOn(A, S), !, fail().                                 29
cpr:notStandsOnOthers(A).                                        30
cpr:nothingStandsOnIt(S) :−                                      31
  cpr:standsOn(A, S), !, fail().                                 32
cpr:nothingStandsOnIt(S).                                        33
                                                                 34
aso:incorporates(X, Y) :−                                        35
  aso:Granary(X),                                                36
  (aso:Artifact(Y) ; aso:Stool(Y)),                              37
  X!=Y.                                                          38
```

In Step 3 in the example, a domain expert creates a CpR (listing 3) including instances of application-specific concepts (classes and properties) that have been mapped to concrete content components and properties included in the CrR. The domain expert creates several artifacts (4–5) and three woman statuettes (6–9). The first two statuettes are made of different materials (`wood` and `glass`), while the third statuette is inherently covered by a texture (as specified in the RM). Furthermore, eight stools are created (11), and `x`, `y` and `z` coordinates are declaratively specified for them by assertions (13–21). In line 20, a cut-off and a negation-as-failure are used to determine a stool, for which no `x` coordinate has been specified. Next, in one declaration all artifacts are assigned to stools (23–33). Finally, all artifacts and stools are incorporated in the `cpr:granary` (35–38).

Step 4: expanding the semantic representation

The first three steps of the modeling process provide a comprehensive semantic representation of 3D content at different levels of semantic abstraction. This representation covers all modeling elements that must be resolved by a human—the specification of how application-specific concepts should be reflected by concrete components and the specification of what application-specific individuals should be included in the created content. The remaining steps of the content creation process may be completed automatically.

So far, the concrete semantic components of 3D content (designed in Step 1) are assigned to application-specific concepts (classes and properties) by *mapping concepts* (designed in Step 2), but they are not directly linked to application-specific individuals (instances of classes and instances of properties—designed in Step 3). To enable presentation of the application-specific individuals, the overall semantic content representation (encompassing the CpR and the CrR) is expanded according to the RM, and the concrete components are linked to the application-specific individuals in the following three transformation stages. In the first stage, reasoning is performed to discover the hidden OPs, which determine the structure of the created content. In the second stage, a structure linking semantic individuals is created for every PO on the basis of the OPs discovered. Finally, in the third stage of the expanding process, DPs are discovered for the created semantic individuals to determine the presentational effects of the POs. The exact description of the expanding algorithm is out of the scope of this paper.

In the result of the above transformation stages, the CpR is transformed to an expanded content representation (ER). The created ER is equivalent to the original CpR in terms of the represented content, but both representations use different levels of abstraction. While a CpR reflects the created content using only application-specific concepts (abstract in terms of presentation), an ER is a counterpart to a CpR that reflects the content using only the ML-SCM concepts (directly specific to the 3D domain). An ER is a structure of linked concrete individuals, which are described by concrete properties.

Listing 4 An expanded content representation (ER)

```
Prefixes:
  Multi−Layered Semantic Content Model (mlscm), CpR (cpr)

{cpr:granary,...,cpr:badge} rdf:type mlscm:Mesh3D.

{cpr:woodenWoman,cpr:glassyWoman}
  rdf:type mlscm:Mesh3D ;
  mlscm:material
    {er:woodMaterial,er:glassMaterial} .

{er:woodMaterial,er:glassMaterial}
  rdf:type
    {mlscm:TextureMaterial,mlscm:ColorMaterial} ;
  {mlscm:texture,mlscm:color}
    {"wood.png","green"} ;
  mlscm:transparency {0,0.7} .

cpr:paintedWoman rdf:type mlscm:Mesh3D ;
  mlscm:material er:paintedWomanMaterial .

er:paintedWomanMaterial rdf:type
    mlscm:TextureMaterial ... .

{cpr:woodenWoman,...,cpr:badge} mlscm:sensor
   {er:touchSensor1,...,er:touchSensor8} .

{er:touchSensor1,...,er:touchSensor8}
  rdf:type mlscm:TouchSensor ;
  mlscm:activates
    {er:rotatingInterp1,..,er:rotatingInterp8} .

{er:rotatingInterp1,..,er:rotatingInterp8}
  rdf:type mlscm:RotatingInterpolator ;
  mlscm:key "0 1.5 3" ;
  mlscm:keyValue "0 0 0 0 3.14 0 0 0 0" ;
  mlscm:controller
    {er:timeSensor1,...,er:timeSensor8} .

{er:timeSensor1,...,er:timeSensor8}
  rdf:type mlscm:TimeSensor ;
  mlscm:interval 3 .

{cpr:stool1,...,cpr:stool8}
  rdf:type mlscm:StructuralComponent ;
  mlscm:includes
    {er:cylinder1,...,er:cylinder8} ,
    {er:box1,...,er:box8} .

{er:cylinder1,...,er:cylinder8,er:box1,...,er:box8}
  rdf:type {mlscm:Cylinder,mlscm:Box} ;
  mlscm:size {er:size1,...,er:size16} ;
  mlscm:position {er:pos1,...,er:pos16} .

{er:size1,...,er:size16}
  rdf:type mlscm:Vector ;
  mlscm:x "..." ;
  mlscm:y "..." ;
  mlscm:z "..." .

{er:pos1,...,er:pos16}
  rdf:type mlscm:Vector ; ... .

{cpr:stool1,...,cpr:stool8}
  mlscm:position
    {er:stool1Pos,...,er:stool8Pos} .

{er:stool1Pos,...,er:stool8Pos}
  rdf:type mlscm:Vector ; ... .

{cpr:woodenWoman,...,cpr:badge}
  mlscm:position
    {er:woodenWomanPos,...,er:badgePos} .

{er:woodenWomanPos,...,er:badgePos}
  rdf:type mlscm:Vector ; ... .

cpr:granary
  mlscm:includes
    {cpr:woodenWoman,...,cpr:badge} ,
    {cpr:stool1,...,cpr:stool8} .
```

In Step 4 in the example, the overall semantic content representation (including the CrR and the CpR) is automatically expanded according to the RM. Consequently, new semantic individuals are generated and linked to the individuals of the CpR by OPs, and their DPs are set properly (listing 4). All artifacts are specified as `mlscm:Mesh3D` individuals (4). For wooden and glassy woman statuettes (6–9), appropriate individuals reflecting materials are generated (11–16). The `cpr:paintedWoman` and its material are created in a similar way (18–22). Furthermore, for every artifact (24), an `mlscm:TouchSensor` is generated (27–30). Every `mlscm:TouchSensor` activates an `mlscm:RotatingInterpolator` (32–37), which is controlled by an `mlscm:TimeSensor` (39–41). Next, every `aso:Stool` is expanded to an `mlscm:Structural Component` that includes an `mlscm:Cylinder` and an `mlscm:Box` with appropriate dimensions and relative positions (43–61). For every `aso:Stool` a position is determined (63–68), and artifacts are assigned to stools by getting appropriate positions (70–75) according to the constraints (declarative rules) specified in the previous step. Finally, all objects (artifacts and stools) are included in the `cpr:granary` (77).

Step 5: building the final content representation

The last step of the content creation process is a transformation of an ER (including the concrete semantic components), to a final content representation (including final 3D counterparts of the concrete components), which is encoded in a particular 3D content representation language. This part of the content creation process can be performed automatically with a transformation knowledge base that links concrete components to their corresponding final counterparts. The transformation can cover a wide range of target presentation platforms based on either declarative (e.g., VRML, X3D and XML3D) or imperative (e.g., Java, ActionScript and JavaScript) content representation languages. Building final 3D content presentations on the basis of semantic knowledge bases has been explained in detail in [18].

In Step 5 in the example, a final 3D scene is generated on the basis of the ER created in the previous step (listing 5—the X3D/XML syntax). For every artifact, a `Transform` node with a position indicated by the `translation` attribute is generated (e.g., 8–24). It includes a `Shape` node with a material and an optional texture. Moreover, every artifact is equipped with

Listing 5 A final 3D content representation

```
1  <?xml version="1.0" encoding="UTF−8"?>
2  <!DOCTYPE ...">
3  <X3D ... >
4  <head>...</head>
5  <Scene>
6    <Transform>
7      <Group>
8        <Transform DEF="woodenWoman" translation="...">
9          <Shape>
10           <Appearance>
11             <Material transparency="0" />
12             <ImageTexture url="wood.png" />
13           </Appearance>
14           <IndexedFaceSet coordIndex="...">
15             <Coordinate point="..."/>
16           </IndexedFaceSet>
17         </Shape>
18         <TouchSensor DEF="touchSensor1" enabled="false"/>
19         <OrientationInterpolator DEF="rotatingInterp1" key="0 1.5 3"
                   keyValue="0 0 0 0 3.14 0 0 0 0" />
20         <TimeSensor DEF="timeSensor1" cycleInterval="3"/>
21         <ROUTE fromNode='touchSensor1' fromField='touchTime' toNode=
                   'timeSensor1' toField='startTime'/>
22         <ROUTE fromNode='timeSensor1' fromField='fraction_changed'
                   toNode='rotatingInterp1' toField='set_fraction'/>
23         <ROUTE fromNode='rotatingInterp1' fromField='value_changed'
                   toNode='woodenWoman' toField='rotation'/>
24       </Transform>
25
26       <Transform DEF="glassyWoman" translation="...">
27         <Shape>
28           <Appearance>
29             <Material transparency="0.7" diffuseColor="0 1 0" />
30           </Appearance>
31           <IndexedFaceSet ...>...</IndexedFaceSet>
32         </Shape>
33         <TouchSensor .../> <OrientationInterpolator ... /> <TimeSensor ... />
                   <!−− Routes −−>
34       </Transform>
35
36       <Transform DEF="paintedWoman" translation="...">
37         <Shape>paintedWoman model</Shape>
38         <TouchSensor .../> <OrientationInterpolator ... /> <TimeSensor ... />
                   <!−− Routes −−>
39       </Transform>
40       <Transform DEF="granary" translation="...">
41         <Shape>granary model</Shape>
42         <TouchSensor .../> <OrientationInterpolator ... /> <TimeSensor ... />
                   <!−− Routes −−>
43       </Transform>...
44       <Transform DEF="badge" translation="...">
45         <Shape>badge model</Shape>
46         <TouchSensor .../> <OrientationInterpolator ... /> <TimeSensor ... />
                   <!−− Routes −−>
47       </Transform>
48
49       <Transform DEF="stool1" translation="...">
50         <Transform DEF="box1" translation="..." scale="...">
51           <Shape>
52             <Appearance><Material /></Appearance>
53             <Box />
54           </Shape>
55         </Transform>
56         <Transform DEF="cylinder1" translation="..." scale="...">
57           <Shape>
58             <Appearance><Material /></Appearance>
59             <Cylinder />
60           </Shape>
61         </Transform>
62       </Transform>
63       <!−− stool2, ..., stool8 −−>
64     </Group>
65   </Transform>
66 </Scene>
67 </X3D>
```

a `TouchSensor`, an `OrientationInterpolator` and a `TimeSensor`, which are connected by `ROUTE` nodes and enable the rotation of the artifact after it is touched. Stools are generated as `Transform` nodes including two shapes—a `Cylinder` and a `Box` with positions and scales (e.g., 49–62). All objects are enclosed in a common `Transform` node (6–65). The final 3D scene generated is presented in Fig. 5.

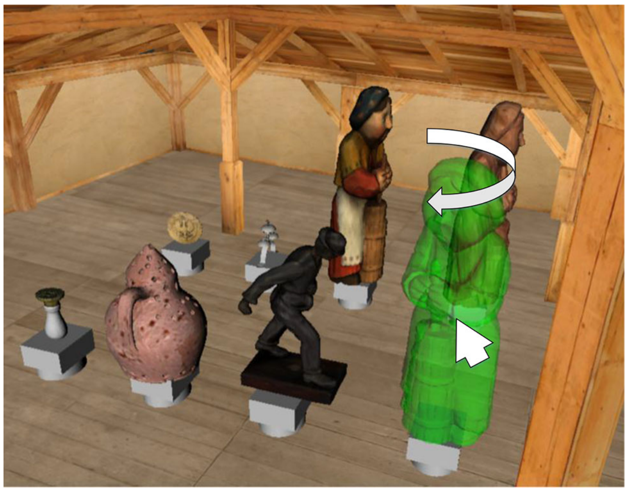
Fig. 5 The example final 3D content representation

6 Implementation

SCM, SMM as well as semantic content representations have been implemented using the semantic web standards (RDF, RDFS and OWL), which express facts as semantic statements, as well as the Prova declarative language [24], which expresses semantic rules as horn clauses. The restrictive use of the formally specified semantic web standards—the Resource Description Framework (RDF), the Resource Description Framework Schema (RDFS) and the Web Ontology Language (OWL)—in SEMIC is preferred over the use of other concepts (in particular rules, which have high semantic expressiveness) because of the following two reasons: First, the semantic web standards provide concepts, which are widely accepted on the web and can be processed using well-established tools, such as editors and reasoners. Second, complexity measures have been investigated and specified for these standards, including a number of typical reasoning problems (such as ontology consistency, instance checking and query answering) [41], which allows for building applications with more predictable computational time.

The steps of SCCM that may be performed automatically—expanding semantic content representations and building final content representations—have been implemented as Java-based applications—an *expander* and a *compiler*. The Pellet reasoner [33] and the Apache Jena SPARQL engine [2] are used in both applications to process semantic statements, while the Prova rule engine [24] is used in the *expander* to process semantic rules. The selected target languages are: VRML, X3D and ActionScript with the Away3D library. The 3D content representations encoded in ActionScript are presented using the Adobe Flash Player, while the representations encoded in VRML and X3D are presented using VRML and X3D browsers, e.g., Cortona3D and Bitmanagement BS Contact.

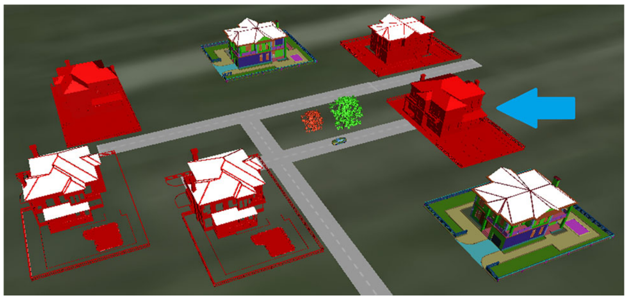
Fig. 6 A generated 3D scene: mark in *red* all buildings, to which a road from the building (indicated by the *arrow*) leads

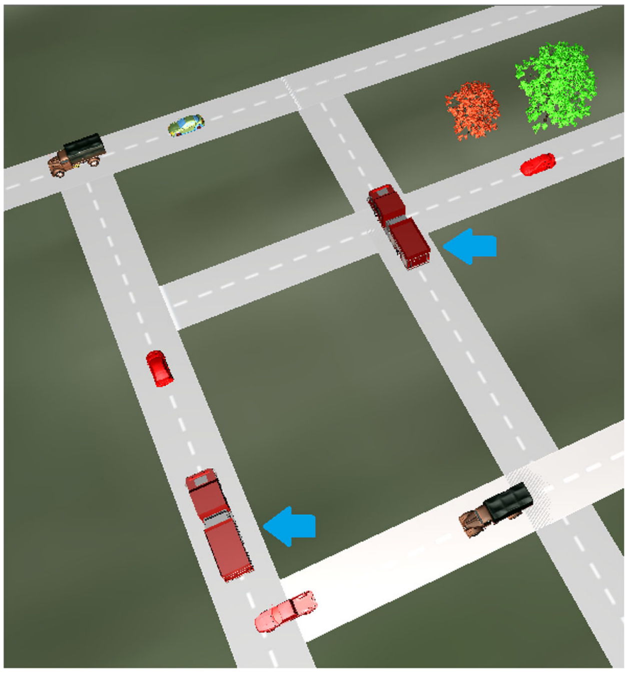
Fig. 7 A generated 3D scene: mark in *red* all cars that are close to priority vehicles (indicated by the *arrows*) going on the same road

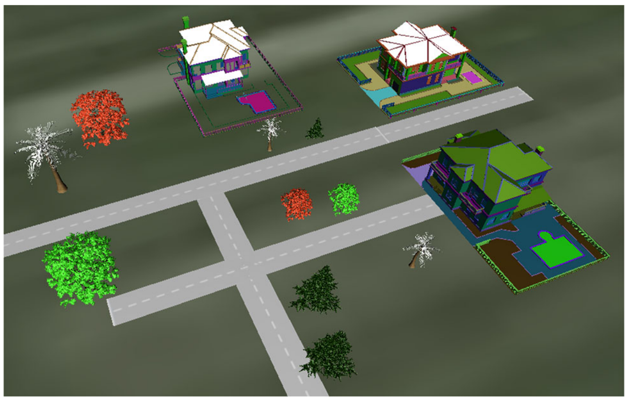
Fig. 8 A generated 3D scene: the trees that are closer to buildings are lower than the other trees

Several 3D scenes generated with the SEMIC implementation are presented in Figs. 6–8. The scenes have been conceptually modeled using an ontology with concepts reflecting selected elements of cities and they have been expanded by knowledge discovery.

7 Evaluation

Qualitative and quantitative evaluations have been performed to compare the SEMIC approach with selected approaches to 3D content creation.

7.1 Qualitative evaluation

The qualitative evaluation performed includes a comparison of SEMIC with selected approaches to 3D content creation in terms of functionality. The selected approaches are leading in terms of functionality, available documentation and the community of users. The evaluation covers approaches to semantic content creation (proposed by Latoschik et al., Troyer et al. and Kalogerakis et al.), imperative programming languages and programming libraries (ActionScript with Away3D and Java with Java3D) as well as environments for visual content creation (advanced environments—Blender and 3ds Max, and user-friendly environments—SketchUp and 3DVIA). The qualitative evaluation (presented in Table 1) aims to indicate the major gaps in the available approaches, which are to be covered by the SEMIC approach (cf. Sect. 3).

Declarative content creation is an aspect that significantly distinguishes semantic approaches from imperative languages and visual environments. Semantic approaches allow for content reflection by facts and rules, for which the order of appearance in the content representation is not important, in contrast to the instructions of imperative languages. Therefore, declarative content representation can be more intuitive for content authors, who may specify desirable content properties and constraints, instead of specifying sequences of instructions and analyzing the order, in which they are processed. Moreover, declarative content description significantly facilitates automatic content management (indexing, searching and analyzing) in repositories by referring to content attributes, which may conform to common schemes and ontologies. Since the knowledge bases used in SEMIC consist of semantic triples (expressing facts) and horn clauses (expressing rules), which declaratively represent the content, the approach satisfies the requirement 1 (Sect. 3).

Knowledge-based 3D content creation has been considered in terms of building content representations with regards to discovered properties and dependencies of content objects, which may be hidden (not explicitly specified), but they are the logical implications of facts and rules that have been explicitly specified in the knowledge base. On the one hand, this aspect of content creation is not available in imperative languages, including the languages used in the visual environments. On the other hand, although the available semantic approaches could be extended to enable knowledge-based modeling, currently, they do not support content creation based on extracted data. Since ERs in SEMIC include not only facts and rules explicitly specified during the first three steps of the modeling process, but also inferred logical implications of the facts and rules, the approach satisfies the requirement 2.

Table 1 Comparison of the selected approaches to 3D content creation

Criterion\approach	Modelling paradigm						
	Semantic content creation				Imperative content creation	Visual content creation	
	SEMIC	Latoschik et al.	Troyer et al.	Kalogerakis et al.	ActionScript (Away3D), Java (Java3D)	SketchUp, 3DVIA	Blender, 3ds Max
Declarative content creation	✓	✓	✓	✓	–	–	–
Knowledge-based content creation	✓	○	–	○	–	–	–
Conceptual content creation	✓	○	○	○	✓	–	✓
Separation of concerns	✓	–	–	–	–	–	–
Multi-platform content creation	✓	–	–	–	–	○	○

✓ meets, – does not meet, ○ meets partially

Conceptual content creation has been considered in terms of representation of 3D content at different levels of abstraction (detail) and the use of the well-established semantic web concepts (classes, individuals, properties, facts and rules) in 3D content creation process. Overall, the available semantic approaches enable the use of basic semantic expressions (combinations of semantic concepts), such as classes and properties, at different levels of abstraction in modeling of content. However, they do not permit a number of more sophisticated combinations of concepts, which are essential to visualization of complex knowledge bases and which are covered by SEMIC. The imperative languages and visual environments permit complex conceptual content representations at different levels of abstraction, however, expressed imperatively, which is not convenient for knowledge extraction, reasoning and content management in web repositories. Since the knowledge bases used in SEMIC represent the content at the concrete (specific to 3D modeling) and conceptual (specific to an application or domain) levels of abstraction, the approach satisfies the requirement 3.

The previous approaches do not support the separation of concerns that are related to substantially different aspects of content creation between different modeling users, who have different modeling skills and experience, and are equipped with different modeling tools. Although, the available approaches allow for assigning different tasks to different users (e.g., creating different content components or writing different parts of code), the tasks are difficult to be accomplished using different tools and require similar expertise from all involved users (e.g., in the use of a common visual environment or a common language). Separation of concerns has been achieved in SEMIC by decoupling particular aspects of the modeled content into the separate steps of the modeling process, which can be performed manually—by different modeling users, or automatically—by specific algorithms (the requirement 4).

Multi-platform content representation has been considered in terms of flexible and generic content transformation to different formats. Although, the analyzed visual environments support different content formats and the advanced environments (Blender and 3ds Max) enable the introduction of new formats (e.g., by implementing appropriate plug-ins), they do not enable generic description of content transformation that is independent of particular content representation formats. Such description could facilitate the introduction of new content formats and permit content presentation on new platforms. Overall, the semantic approaches do not permit generic, flexible and extensible content transformation for different platforms. Since ERs in SEMIC are platform- and standard-independent, and since they can be transformed to different content representation languages and presented using different presentation platforms, the approach satisfies the requirement 5.

7.2 Quantitative evaluation

The quantitative evaluation performed includes the complexity of 3D content representations and profit from conceptual modeling of 3D content. The evaluation covers CpRs, ERs and final content representations. The CpRs and ERs have been encoded with SCM using the RDF-Turtle format. The ontology used for building the CpRs provides different types of application-specific complex objects and complex properties assembling multiple concrete components of 3D content. Final content representations have been encoded with the VRML, X3D and ActionScript (with Away3D) languages.

The evaluation has been carried out starting from CpRs, which are scenes assembled from various numbers of objects. The number of objects has varied over the range 5–50 with the step equal to 5. For every number of objects, 20 scenes have been randomly generated and the average results have been calculated. The test environment used is equipped with the Intel Core i7-2600K 3.4GHz CPU, 8 GB RAM and the Windows 7 OS.

7.2.1 Complexity of 3D content representations

The complexity of 3D content representations has been evaluated with the following metrics: the structured document complexity metric [31], the number of bytes, the number of logical lines of code (LLOC) [1] and the Halstead metrics [22]. The first metric has been used to measure the complexity of representation schemes of conceptual, concrete and final content representations, whereas the other metrics have been used to measure the complexity of particular content representations.

Structured document complexity metric. The structured document complexity metric has been used to measure the complexity of representation schemes regarding unique elements and attributes, required elements and attributes as well as attributes that need to be specified at the first position within their parent elements. The metric may be calculated for XML- and grammar-based documents. The values of the structured document complexity metric calculated for the schemes of conceptual, concrete and final content representations (encoded in VRML, X3D and ActionScript) are presented in Table 2. The results obtained for VRML and X3D representations are equal, because both languages use schemes with equivalent basic elements and attributes. While in VRML and X3D representations, different hierarchical document elements have been classified as unique elements, in ActionScript representations, different classes and data types (potentially corresponding to different elements of VRML/X3D) have been classified as unique elements. In ERs, unique elements cover different RDF, RDFS and OWL elements as well as semantic properties of 3D content, which are encoded by document elements (according

Table 2 Structured document complexity metric of the content representation schemes and the mapping scheme

Structured document complexity metric					
Criteria	VRML/X3D	AS	ER	CpR	RM
Unique elements	15	24	24	7	36
Unique attributes	21	27	3	14	8
Required elements	1	1	12	4	4
Required attributes	8	5	3	2	2
Elements at position 1	0	0	0	0	0
Sum	45	57	42	27	50

to the RDF syntax). In comparison to ERs, CpRs include a lower number of unique elements, which are semantic combinations of multiple elements of ERs. Different properties occurring in hierarchical VRML/X3D elements or properties of objects in ActionScript representations have been classified as unique attributes in the metric calculation. Since, in ERs the content properties are encoded using document elements, only a few attributes, which are primary RDF, RDFS and OWL attributes, may be classified as unique attributes in ERs. In comparison to ERs, CpRs have a higher number of unique attributes, which is determined by the application-specific ontology used. RMs incorporate the highest number of unique elements of all analysed documents, because they indicate unique elements of both concrete and conceptual representations. Unique attributes of RMs are semantic properties specified in SMM. In VRML/X3D and ActionScript representations, the scene and the view are the only required elements, and they have a few required attributes. The elements and the attributes that are required in CpRs, ERs and RMs, are basic RDF, RDFS and OWL elements and properties. The calculated values of the structured document complexity metric show that the overall complexity of CpRs is much lower than the overall complexity of ERs, which is a little lower than the overall complexity of VRML/X3D and much lower than the overall complexity of ActionScript representations. The overall complexity of RMs is highest of all the complexities calculated.

Size metrics. The number of bytes (Fig. 9) and the number of logical lines of code—LLOC (Fig. 10)—without comments—have been used to measure the size of content representations. The graphs present the metrics in relation to the number of application-specific components included in CpRs.

The differences in size are relatively high between different types of representations. In both comparisons, CpRs are more than twice as concise as the corresponding final representations, as the CpRs assembly multiple concrete content elements. ERs are most verbose, which confirms that RDF-Turtle is the most verbose encoding format of all the encoding

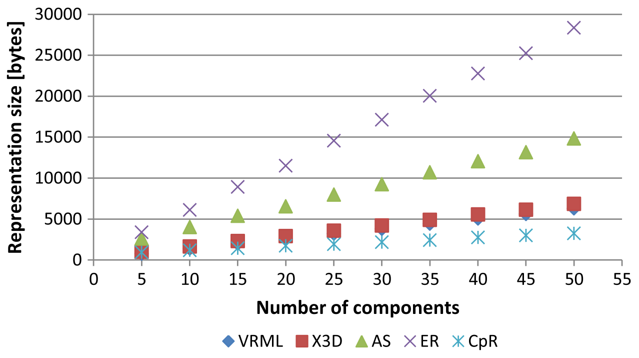

Fig. 9 The size of representation (bytes) depending on the number of components

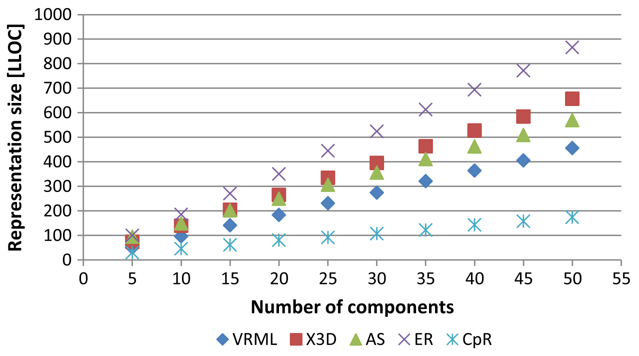

Fig. 10 The size of representation (LLOC) depending on the number of components

formats used, taking into account that the elements of ERs are semantic equivalents to the elements of the corresponding final representations.

Halstead metrics. The Halstead metrics have been used to measure the complexity of content representations in several respects: the size of vocabulary and length of the content representations, volume corresponding to the size of the representations, difficulty corresponding to the error proneness of the representations, effort in implementation and analysis of the representations as well as estimated time required for the development of the representations. The particular Halstead metrics in relation to the number of components of CpRs are presented in the graphs in Figs. 11–16. The VRML and X3D representations that have been considered in the presented evaluation are based on equivalent basic elements and attributes, therefore they have been presented together.

Vocabulary (Fig. 11), which is the sum of unique operators ($n1$) and unique operands ($n2$) of the representation:

$$\text{Voc} = n1 + n2 \tag{1}$$

is lowest for CpRs and it is highest for VRML/X3D, because of a high number of unique operands, which are individual scene graph nodes. In contrast to the other languages,

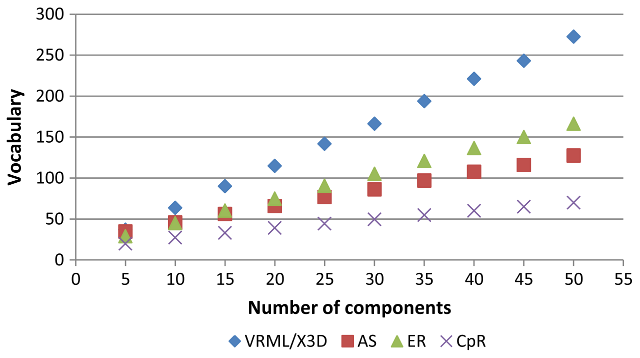

Fig. 11 The vocabulary of representation depending on the number of components

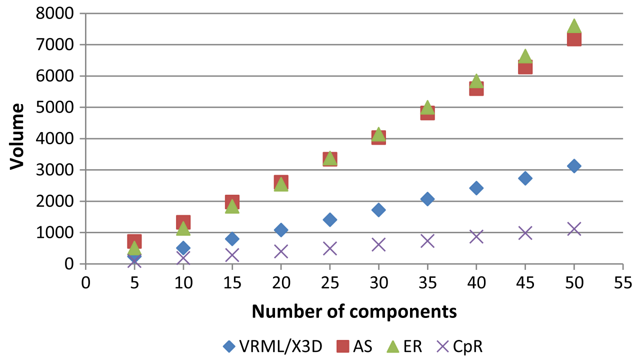

Fig. 13 The volume of representation depending on the number of components

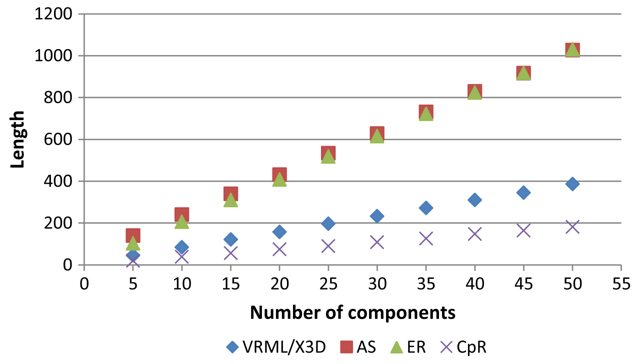

Fig. 12 The length of representation depending on the number of components

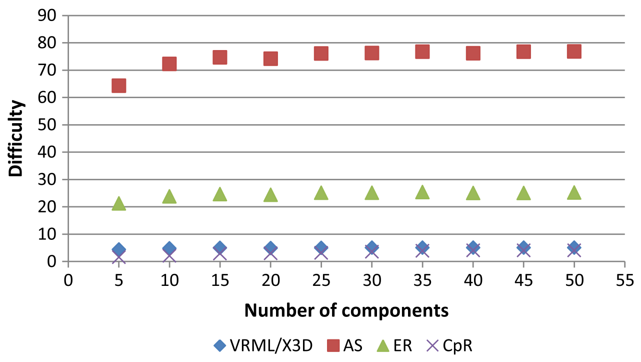

Fig. 14 The difficulty of representation depending on the number of components

in VRML/X3D, a relation between two components in the generated representation is reflected by nesting one component in another component with specifying all intermediate nodes, which are also classified as unique operands, e.g., applying a `Material` to a `Shape` requires an intermediate `Appearance` node to be nested in the `Shape` node. In the other languages, such associations are usually described directly—without any intermediate nodes.

Length (Fig. 12), which is the sum of the total number of operators (N1) and the total number of operands (N2) of the representation:

$$\text{Len} = N1 + N2 \tag{2}$$

is lowest for CpRs. VRML/X3D representations predominate concrete and ActionScript representations, as their operands typically occur once and all references to them are specified by nesting attributes and other nodes in them. Therefore, the operands do not require to be additionally explicitly indicated (e.g., by proper instructions or statements), and thus the length of VRML/X3D representations is lower than the length of the other representations, in which all references to operands must be explicitly declared by referring to their identifiers.

The graph of volume (Fig. 13), which depends on the length and vocabulary of content representations:

$$\text{Vol} = \text{Len} \times \log_2(\text{Voc}) \tag{3}$$

is similar to the graph of length for all types of representations.

In contrast to the other Halstead metrics discussed, difficulty (Fig. 14), which is given by the formula:

$$\text{Diff} = \frac{n1}{2} \times \frac{N2}{n2} \tag{4}$$

has similar values for different numbers of components in the scene. It is lowest for conceptual and VRML/X3D representations (low error proneness) because of the relatively low values of the number of distinct operators and the total number of operands, and relatively high values of the number of distinct operands. Relatively high difficulty of ActionScript representations (high error proneness) is caused by relatively high values of the first two factors and a relatively low value of the third factor.

Effort (Fig. 15) and time (Fig. 16) required for the implementation or analysis of the representation, which are the products of its difficulty and volume:

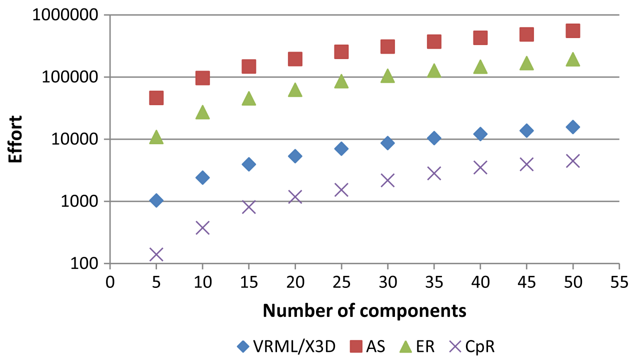

Fig. 15 The effort in analyzing the representation depending on the number of components

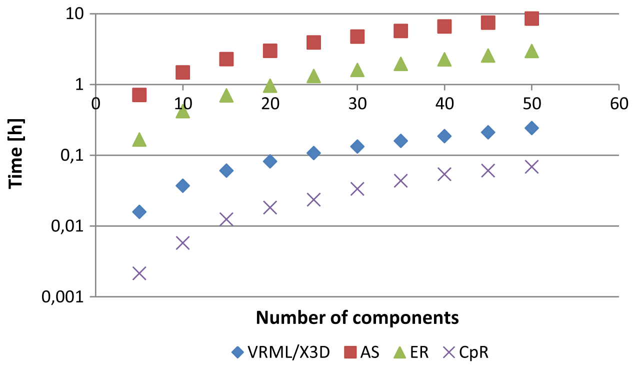

Fig. 16 The time of implementing the representation depending on the number of components

$$\text{Eff} = \text{Diff} \times \text{Vol} \tag{5}$$

$$\text{Time (h)} = \frac{\text{Eff}}{18 \times 3600} \tag{6}$$

are lowest for CpRs because of the relatively low values of their difficulties and volumes. Higher values of effort and time occur for the other representations.

7.2.2 *Profit from conceptual content creation*

The profit from conceptual content modeling is directly proportional to the number of scenes that have to be created and their size, and it is inversely proportional to the sum of the size of the RM (that must be implemented) and the size of the primary CpRs that have to be transformed to final representations. The metric is given by the formula:

$$\text{Profit} = \frac{N \times \text{AvgSize(FR)}}{\text{Size(RM)} + N \times \text{AvgSize(CpR)}} \tag{7}$$

where: N is the number of scenes (final content representations) that have to be created, Size and AvgSize are the size and the average size of a scene.

The values of profit in relation to the number of scenes are presented in the graphs in Figs. 17–19. The cost of the creation of the RM is not acceptable for low numbers of simple scenes (the values of profit lower than 1) and it is acceptable for high numbers of complex scenes (the values grater than 1). The profit is higher for the more verbose languages (ActionScript) and it is lower for the more concise languages (VRML).

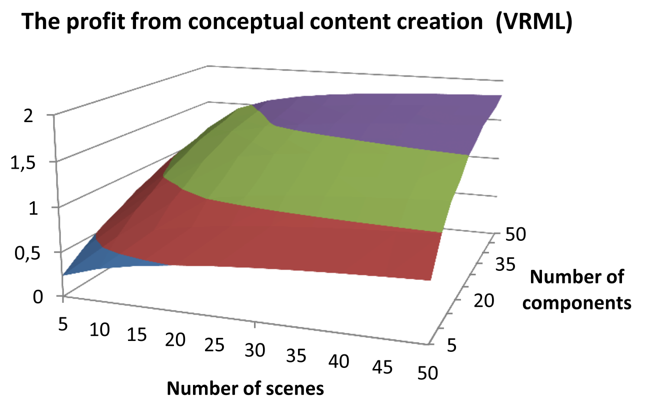

Fig. 17 The profit from conceptual content creation for VRML (for representation size in bytes)

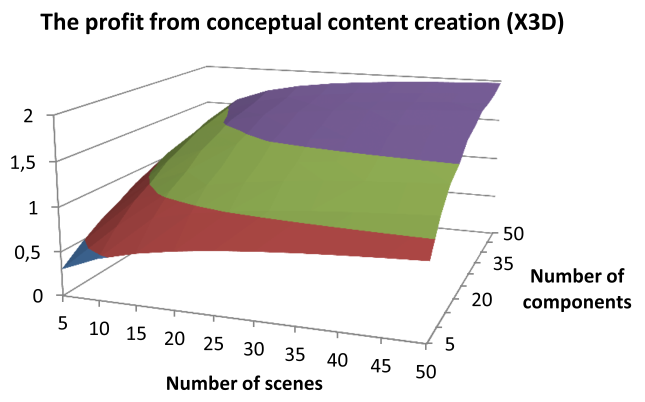

Fig. 18 The profit from conceptual content creation for X3D (for representation size in bytes)

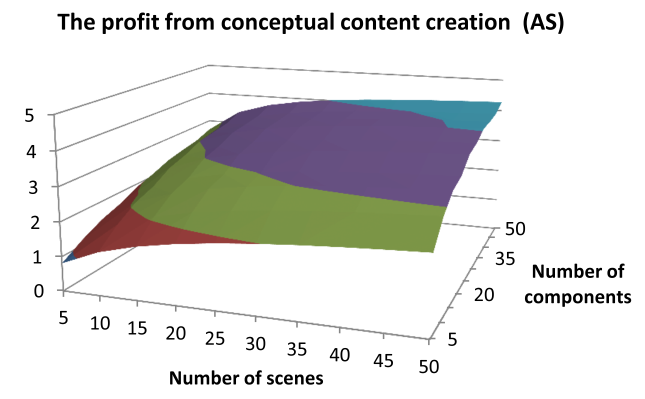

Fig. 19 The profit from conceptual content creation for ActionScript (for representation size in bytes)

8 Discussion

The tests carried out show that the SEMIC approach outperforms the previous approaches to 3D content creation in terms of functionality as well as the parameters of content representation schemes and generated content representations.

The advantage in terms of the size and complexity of content representations results from modeling 3D content at a higher level of abstraction and leads to creating objects that are assemblies of multiple low-level content components. Although the overall size and complexity of an RM and a CpR may be higher than the size and the complexity of the resulting VRML, X3D or ActionScript scene, RMs are created only once and are common for all CpRs compliant with the selected application-specific ontology, which are the only documents created by domain experts. The size of CpRs is typically much smaller than the size of the corresponding final content representations, which are encoded with the widely used 3D content representation languages and programming libraries. The smaller size can enhance storage and transmission of 3D content to target users, e.g., from distributed web repositories to content presentation platforms.

Furthermore, the proposed approach provides much better possibilities of content implementation (requires less effort and shorter time) by enabling the use of representation schemes whose complexity can be lower than the complexity of commonly used content representation languages. As a result, the vocabulary, length, volume and difficulty of content representations are lower and the creation as well as the understanding of the representations requires less effort and time. Hence, the proposed approach can be used to accelerate and improve effectiveness of 3D content creation using application-specific ontologies and tools that are less complicated than the tools currently used for 3D content creation with the available representation languages.

Moreover, the approach enables content creation by users without considerable expertise in 3D modeling, e.g., domain experts. The calculated values of the profit from conceptual creation of 3D content show that the use of the proposed approach is effective even for systems with relatively low numbers of scenes that have to be designed, in comparison to modeling all the scenes using the available languages.

In the quantitative evaluation, ERs and CpRs have been considered separately, because a CpR and its corresponding ER independently and comprehensively represent the content at different (conceptual and concrete) levels of abstraction—each of them includes all elements that have to be presented. Summing up the complexities of ERs and CpRs may be important if the created content needs to be managed (indexed, searched or analyzed) simultaneously at different levels of abstraction, e.g., storing and accessing content in repositories. The values of size, length, volume, difficulty, effort and time calculated for ERs are much higher than the values calculated for VRML/X3D representations. However, ERs are generated automatically on the basis of CpRs, so it is not a burden on content authors to create them.

9 Conclusions and future works

In this paper, a new approach to semantic creation of 3D content has been proposed. The SEMIC approach leverages the semantic web techniques to satisfy the requirements for modeling of 3D content, which have been specified in Sect. 3. It goes beyond the current state of the art in the field of conceptual knowledge-based modeling of 3D content, and it can improve the creation of complex 3D content for VR/AR web applications in various domains.

The presented solution has several important advantages in comparison to the available approaches to 3D content creation. First, the use of the semantic web techniques enables declarative content creation, which stresses the description of the results to be achieved, but not the manner in which the results are to be obtained. Second, it permits content creation at different levels of abstraction regarding hidden knowledge, which may be inferred and used in the modeling process. Third, the approach permits the reuse and assembly of common content elements with respect to their dependencies, which may be conceptually described with application-specific ontologies and knowledge bases—this enables content creation by different users equipped with different modeling tools. Furthermore, content creation with the SEMIC approach requires less time and effort in implementation, and it provides less complicated results, which can be stored and transmitted more effectively. Next, due to the conformance to the semantic web approach, the content created with SEMIC is suitable to be processed (indexed, searched and analyzed) in web repositories. Finally, the created content is platform- and standard-independent and it can be presented using diverse 3D content presentation tools, programming languages and libraries liberating users from the installation of additional software, which can improve the dissemination of 3D content.

Possible directions of future research incorporate several facets. First, a visual modeling tool supporting semantic 3D content creation according to SEMIC can be developed. Such a tool can allow for the extension of the current SEMIC evaluation with modeling of 3D content by domain experts. Such an evaluation can cover time required for creating CrRs, RMs and CpRs by users with low and high skills in 3D modeling, time required for creating content components of different types (geometrical, structural, behavioral, etc.) and scenes with different complexities, as well as profit from creating particular numbers of scenes using different tools. Second, the approach can be enriched with content creation based on semantic queries (expressed, e.g., in SPARQL) to content repositories. Third, the proposed approach could be extended with semantic transformation of declarative rule-based descriptions of complex content behavior. Finally, persistent link between semantic and final 3D content representations can be proposed to enable real-time synchronization of content representations.

References

1. Aivosto: Lines of code metrics (LOC). http://www.aivosto.com/project/help/pm-loc.html
2. Apache: Apache Jena. http://jena.apache.org/
3. Averkiou, M., Kim, V.G., Zheng, Y., Mitra, N.J.: ShapeSynth: parameterizing model collections for coupled shape exploration and synthesis. Comput Graphics Forum **33**(2), 125–134 (2014)
4. Aylett, R., Luck, M.: Applying artificial intelligence to virtual reality: intelligent virtual environments. Appl. Artif. Intell. **14**, 3–32 (2000)
5. Bilasco, I.M., Gensel, J., Villanova-Oliver, M., Martin, H.: 3dseam: a model for annotating 3d scenes using mpeg-7. In: ISM, pp. 310–319. IEEE Computer Society (2005)
6. Bilasco, I.M., Villanova-Oliver, M., Gensel, J., Martin, H.: Semantic-based rules for 3d scene adaptation. In: Proceedings of the Twelfth International Conference on 3D Web Technology, Web3D '07, pp. 97–100. ACM, NY, USA (2007)
7. Bille, W., Pellens, B., Kleinermann, F., Troyer, O.D.: Intelligent modelling of virtual worlds using domain ontologies. In: Proceedings of the Workshop of Intelligent Computing (WIC), held in conjunction with the MICAI 2004 conference, pp. 272–279. Mexico City, Mexico (2004)
8. Cavazza, M., Palmer, I.: High-level interpretation in virtual environments. Applied AI **14**, 125–144 (2000)
9. Chaudhuri, S., Kalogerakis, E., Giguere, S., Funkhouser, T.: Attribit: Content creation with semantic attributes. In: Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology. UIST '13, pp. 193–202. ACM, New York, NY, USA (2013)
10. Flotyński, J.: Harvesting of semantic metadata from distributed 3d web content. In: Proceedings of the 6th International Conference on Human System Interaction (HSI), June 06–08, 2013, Sopot (Poland). IEEE (2013)
11. Flotyński, J.: Semantic modelling of interactive 3d content with domain-specific ontologies. Procedia Computer Science (2014). In: 18th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, accepted for publication
12. Flotyński, J., Walczak, K.: Attribute-based semantic descriptions of interactive 3d web content. In: L. Kiełtyka (ed.) Information Technologies in Organizations-Management and Applications of Multimedia, pp. 111–138. Wydawnictwa Towarzystwa Naukowego Organizacji i Kierownictwa-Dom Organizatora (2013)
13. Flotyński, J., Walczak, K.: Conceptual semantic representation of 3d content. In: Lecture Notes in Business Information Processing: 16th International Conference on Business Information Systems, Poznań, Poland, 19–20 June, vol. 160, pp. 244–257 (2013)
14. Flotyński, J., Walczak, K.: Describing semantics of 3d web content with rdfa. In: The First International Conference on Building and Exploring Web Based Environments, Sevilla (Spain), January 27–February 1, pp. 63–68. ThinkMind (2013)
15. Flotyński, J., Walczak, K.: Microformat and microdata schemas for interactive 3d web content. In: Ganzha, M., Maciaszek, L., Paprzycki, M. (eds.) Proceedings of the 2013 Federated Conference on Computer Science and Information Systems Kraków, Poland, 8–11 September, 2013, vol. 1, pp. 549–556. Polskie Towarzystwo Informatyczne (2013)
16. Flotyński, J., Walczak, K.: Semantic modelling of interactive 3d content. In: Proceedings of the 5th Joint Virtual Reality Conference, pp. 41–48. Paris, France (2013)
17. Flotyński, J., Walczak, K.: Semantic multi-layered design of interactive 3d presentations. In: Proceedings of the Federated Conference on Computer Science and Information Systems, pp. 541–548. IEEE, Kraków, Poland (2013)
18. Flotyński, J., Walczak, K.: Multi-platform semantic representation of 3d content. In: Proceedings of the 5th Doctoral Conference on Computing, Electrical and Industrial Systems. Lisbon, Portugal (2014)
19. Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., Dobkin, D.: Modeling by example. ACM SIGGRAPH 2004 Papers. SIGGRAPH '04, pp. 652–663. ACM, New York, NY, USA (2004)
20. Gutiérrez,, M., Vexo, F., Thalmann, D.: Semantics-based representation of virtual environments. Int. J. Comput. Appl. Technol. **23**, 229–238 (2005)
21. Gutiérrez, A.M., García-Rojas, A., Thalmann, D., Vexo, F., Moccozet, L., Magnenat-Thalmann, N., Mortara, M., Spagnuolo, M.: An ontology of virtual humans: Incorporating semantics into human shapes. Vis. Comput. **23**(3), 207–218 (2007)
22. IBM: Halstead Metrics. http://pic.dhe.ibm.com/infocenter/rtrthelp/v8r0m0
23. Kalogerakis, E., Christodoulakis, S., Moumoutzis, N.: Coupling ontologies with graphics content for knowledge driven visualization. In: VR '06 Proceedings of the IEEE conference on Virtual Reality, pp. 43–50. Alexandria, Virginia, USA (2006)
24. Kozlenkov, A., Paschke, A.: Prova Rule Language. https://prova.ws/
25. Laga, H., Mortara, M., Spagnuolo, M.: Geometry and context for semantic correspondences and functionality recognition in man-made 3d shapes. ACM Trans. Graph. **32**(5), 150:1–150:16 (2013)
26. Latoschik, M.E., Biermann, P., Wachsmuth, I.: Knowledge in the loop: semantics representation for multimodal simulative environments. In: Lecture Notes in Computer Science, pp. 25–39. Springer, Berlin (2005)
27. Latoschik, M.E., Blach, R.: Semantic modelling for virtual worlds—a novel paradigm for realtime interactive systems? In: Proceddings of the 2008 ACM Symposium on VR Software and Technology, pp. 17–20. Bordeaux, France (2008)
28. Latoschik, M.E., Fröhlich, C.: Semantic reflection for intelligent virtual environments. In: IEEE Virtual Reality Conference 2007, pp. 305–306. Charlotte, USA (2007)
29. Luca, L.D., Véron, P., Florenzano, M.: A generic formalism for the semantic modeling and representation of architectural elements. Vis Comput **23**, 181–205 (2007)
30. Lugrin, J., Cavazza, M.: Making sense of virtual environments: action representation, grounding and common sense. In: Proceedings of the 12th International Conference on Intelligent User Interfaces, pp. 225–234. Honolulu, HI, USA (2007)
31. O'Reilly: Metrics for XML Projects. http://www.oreillynet.com/xml/blog/2006/05/
32. Otto, K.A.: Semantic virtual environments. In: Special Interest Tracks and Posters of the 14th International Conference on World Wide Web, pp. 1036–1037. Japan (2005)
33. Parsia, C.: Pellet: OWL 2 Reasoner for Java. http://clarkparsia.com/pellet/

34. Pellens, B., Kleinermann, F., Troyer, O.D.: A development environment using behavior patterns to facilitate building 3d/vr applications. In: Proceedings of the 6th Australasian Conference on Interactive Entertainment. ACM, Sydney, Australia (2009)
35. Pellens, B., Troyer, O.D., Bille, W., Kleinermann, F., Romero, R.: An ontology-driven approach for modeling behavior in virtual environments. In: Proceedings of Ontology Mining and Engineering and its Use for Virtual Reality (WOMEUVR 2005), pp. 1215–1224. Springer, Agia Napa, Cyprus (2005)
36. Pittarello, F., Faveri, A.: Semantic description of 3d environments: a proposal based on web standards. In: Proceedings of the 11th International Conference on 3D web Technology, pp. 85–95. Columbia, MD, USA (2006)
37. Shao, T., Xu, W., Zhou, K., Wang, J., Li, D., Guo, B.: An interactive approach to semantic modeling of indoor scenes with an rgbd camera. ACM Trans. Graphics **31**, 136:1–136:11 (2012)
38. Sokolov, D., Plemenos, D.: Virtual world explorations by using topological and semantic knowledge. Vis Comput. **24**, 173–185 (2008)
39. Spagnuolo, M., Falcidieno, B.: 3d media and the semantic web. IEEE Intell. Syst. **24**, 90–96 (2009)
40. Troyer, O.D., Kleinermann, F., Pellens, B., Bille, W.: Conceptual modeling for virtual reality. In: Tutorials, Posters, Panels and Industrial Contributions at the 26th International Conference on Conceptual modeling, pp. 3–18. Auckland, New Zealand (2007)
41. W3C: OWL. http://www.w3.org/2001/sw/wiki/OWL
42. W3C: World Wide Web Consortium. http://www.w3.org
43. Walczak, K.: Beh-vr: Modeling behavior of dynamic virtual reality contents. In: Proceedings of the 12th International Conference on Virtual Systems and Multimedia VSMM 2006, pp. 40–51. Lecture Notes in Computer Sciences, Springer, Heidelberg (2006)
44. Walczak, K.: Flex-vr: Configurable 3d web applications. In: Proceedings of the International Conference on Human System Interaction HSI 2008, pp. 135–140. Kraków, Poland (2008)
45. Walczak, K., Cellary, W., White, M.: Virtual museum exhibitions. Computer **39**(3), 93–95 (2006)
46. Wiebusch, D., Latoschik, M.E.: Enhanced decoupling of components in intelligent realtime interactive systems using ontologies. In: 2012 5th Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS), pp. 43–51. Orange County, CA, USA (2012)
47. Zaid, L.A., Kleinermann, F., Troyer, O.D.: Applying semantic web technology to feature modeling. In: Proceedings of the 2009 ACM symposium on Applied Computing, pp. 1252–1256. Honolulu, Hawaii, USA (2009)
48. Zheng, Y., Cohen-Or, D., Mitra, N.J.: Smart variations: Functional substructures for part compatibility. Comput Graphics Forum (Eurographics) **32**(2pt2), 195–204 (2013)

Jakub Flotyński received the M.Sc. degree in Electronics and Telecommunications (2011) and in Computer Science (2010) at the Poznań University of Technology in Poland. His research interests include multimedia systems, virtual reality and semantic web. He has worked at the Department of Information Technology at the Poznan University of Economics (Poland) since 2011.

Krzysztof Walczak holds a Ph.D. Hab. degree in Computer Science (Multimedia Systems, Gdańsk 2010) and is an Associate Professor in the Department of Information Technology at the Poznań University of Economics in Poland. His current research interests include virtual reality and augmented reality applications, multimedia systems, semantic web and distance learning.