A Conceptual Framework and Content Model for Next Generation Presentation Solutions

Roels, Reinout Franciscus; Signer, Beat

Published in:
PACMHCI

DOI:
10.1145/3331149

Publication date:
2019

Proceedings of the ACM on Human-Computer Interaction (PACMHCI), 3 (EICS, Article 7), June 2019
The final publication is available at https://doi.org/10.1145/3331149

A Conceptual Framework and Content Model for Next Generation Presentation Solutions

REINOUT ROELS and BEAT SIGNER, Vrije Universiteit Brussel, Belgium

Mainstream presentation tools such as Microsoft PowerPoint were originally built to mimic physical media like photographic slides and still exhibit the same characteristics. However, the state of the art in presentation tools shows that more recent solutions start to go beyond the classic presentation paradigms. For instance, presentations are becoming increasingly non-linear, content is quickly evolving beyond simple text and images and the way we author our presentations is becoming more collaborative. Nevertheless, existing presentation content models are often based on assumptions that do not apply to the current state of presentations any more, making them incompatible for some use cases and limiting the potential of end-user presentation solutions. In order to support state-of-the-art presentation functionality, we rethink the concept of a presentation and introduce a conceptual framework for presentation content. We then present a new content model for presentation solutions based on the Resource-Selector-Link (RSL) hypermedia metamodel. We further discuss an implementation of our model and show some example use cases. We conclude by outlining how design choices in the model address currently unmet needs with regards to extensibility, content reuse, collaboration, semantics, user access management, non-linearity, and context awareness, resulting in better support for the corresponding end-user functionality in presentation tools.

CCS Concepts: • **Information systems** → *Document structure*; *Multimedia information systems*; • **Applied computing** → Digital libraries and archives; • **Human-centered computing** → Hypertext / hypermedia.

Additional Key Words and Phrases: Conceptual Framework; Content Model; Presentations; PowerPoint; Slideware; MindXpres

ACM Reference Format:
Reinout Roels and Beat Signer. 2019. A Conceptual Framework and Content Model for Next Generation Presentation Solutions. *Proc. ACM Hum.-Comput. Interact.* 3, EICS, Article 7 (June 2019), 22 pages. https://doi.org/10.1145/3331149

1 INTRODUCTION

Presentation solutions such as Microsoft PowerPoint[1] were born as a tool for digitally designing photographic slides (e.g. for slide projectors) before making them permanent on physical media. Therefore, digital slides inherited many of the affordances of their physical counterparts, also including some limitations. For instance, digital slides are still restricted in size, they are usually traversed in a linear sequence and presentation content is often relatively static compared to other digital media. However, the state of the art in both commercial presentation solutions as well as academic prototypes shows that we are gradually letting go of the restrictions imposed by physical media and start to embrace the full potential of digital presentation devices. For example, Prezi[2] has

[1] https://products.office.com/en/powerpoint
[2] https://prezi.com

Authors' address: Reinout Roels; Beat Signer, {rroels,bsigner}@vub.be, Vrije Universiteit Brussel, Brussels, Belgium.

2573-0142/2019/6-ART7 $15.00
https://doi.org/10.1145/3331149

shown the viability of zoomable user interfaces for presentations, clickers and mobile devices allow audience members to actively participate in the presentation and various plug-ins and third-party tools enable more dynamic and interactive content. However, one could argue that the lowest layer, the raw content and its internal representation has evolved very little in the context of presentations. Existing frameworks and content models are based on the classic presentation paradigms and cannot represent recent advancements. As an example of such a paradigm, PowerPoint insists that a presentation is always a linear sequence of slides, all the way down to the data model. By supporting only specific paradigms certain functionality cannot be represented in the existing data models and as a result it is difficult to provide well-integrated runtime solutions.

We argue that in order to provide more effective solutions for well-documented shortcomings we need to rethink the concept of a presentation and make changes to how presentation content is modelled and stored. Typically, presentations are stored in monolithic XML-based files in the file system. In order to reuse or share content, content has to be duplicated which causes situations where multiple versions have to be kept manually in sync if changes need to be made. Knowledge workers often maintain collections of hundreds of presentations [2], which are not only difficult to manage but the massive amounts of duplicate content (e.g. images or videos) unnecessarily take up storage space. Existing work shows that the reusability of content should not be ignored as it can save time and money as well as improve the efficiency and effectiveness of knowledge transfer [39]. Because of how content is currently stored, it is also difficult for multiple people to work on the same presentation at the same time. Recent studies clearly show that reusing presentation content and collaboration is in high demand, but support in tools is lacking [21]. Furthermore, the content types supported by existing formats are rather fixed and they do not take into account the new types of content and content navigation we see in state-of-the-art research. Other work also motivates the need for non-linear presentations [26, 38], context-aware delivery methods [33] and content semantics [39] in presentations. However, existing content models and storage systems do not take these concepts into account and the resulting presentation solutions are isolated stand-alone applications; often with limitations because they have to work around existing shortcomings of the underlying content layer. As detailed later, we believe that features such as reuse, collaboration, semantic annotations, versioning, user access control, content-awareness and different kinds of linking in the content layer can lead to advancements in presentation solutions. Note that the focus of this paper lies on the underlying information layer for next generation presentations, but the graphical authoring of such presentations is ongoing research with different goals and challenges. Further, the presented work forms part of the MindXpres [31] project. The discussed information system for presentations will allow us to make further contributions related to presentation authoring, content visualisation and content interaction in future work. We base ourselves on the requirements of existing academic work and our contributions address shortcomings of existing solutions but the presented solution also serves as a reusable platform for the research community.

We start by discussing related work in Section 2 and show how existing models do not offer enough flexibility to represent the state of the art in presentation solutions, or lack core concepts unique to the domain of presentation solutions. In Section 3 we present a conceptual framework that covers existing as well as next generation presentation paradigms. We then introduce a new content model for presentation solutions in Section 4, which unifies and fulfils existing requirements. An implementation of the model as a functional information system for presentation content is discussed in Section 5. After a discussion and some example use cases in Section 6, we outline future work and provide some conclusions in Section 7.

2 RELATED WORK

In order to illustrate the need for new abstractions and content models, we have a look at novel presentation tools and highlight characteristics that break with classic presentation paradigms. We then discuss existing conceptual frameworks and content models to show that some recent concepts cannot be represented.

2.1 Alternative Presentation Solutions

First of all, we list some newer solutions that offer alternative ways of *visualising* and *navigating* content. The linearity of PowerPoint has, for instance, been criticised by Tufte [38]. Popularised by Prezi, pptPlex[3] and impress.js[4], Zoomable User Interfaces (ZUIs) offer an alternative to the classic sequence of slides and have been explored by academic work such as Fly [18], CounterPoint [9] and iMapping [11]. As a ZUI does not impose an explicit order on the content, we see that it is very natural to navigate content in a non-linear way. However, ZUIs are not the only approach for non-linear presentations. Solutions such as NextSlidePlease [36], HyperSlides [6], MultiPresenter [17], reveal.js[5] and [22] allow users to create complex non-linear navigational meshes over slides. Furthermore, Palette [23] and PaperPoint [35] offer physical paper-based interfaces to support non-linear content traversal. The content types that are supported by mainstream presentation solutions are often relatively basic and static. The majority of commercial PowerPoint plug-ins try to solve this issue, but some content types (e.g. source code, geographical information or raw numerical data) remain difficult to present in established presentation tools. This is addressed by tools such as Slithy [43], Hans Rosling's Gapminder[6], or MindXpres [31] which offers dynamic source code visualisation [29, 30] and interactive data-driven storytelling based on raw data [27, 28]. Established presentation solutions typically store content for a single presentation in a monolithic file. However, when users want to *reuse* their content, they are forced to copy and paste the content. As a result, a user now has two versions of the same content that need to be kept in sync in case of future updates. This also makes it more difficult to collaboratively create presentations since each user has their own copy and changes are difficult to merge. Microsoft Research confirms these difficulties [5] and proposes a technical solution to help users merging slides and presentations. Office SharePoint 2007, 2010 and 2013 used to offer the now discontinued concept of *Slide Libraries* allowing users to store slides in a shared repository. However, users were required to manually upload, re-download and merge slides. Research projects such as SliDL [3] and SidePoint [19] also focus on improving content reuse whereas solutions such as Google Slides[7], SlideWiki [13] and Webstrates [14] mainly deal with *collaborative editing*. Although not a feature seen in typical presentation solutions, it can be beneficial to provide content with *semantics* to make it machine understandable, for instance by semantic annotation as investigated in SidePoint [19], iMapping [11] or CPoint [15]. For example, a specific text fragment might be tagged as a definition and associated with the concept it defines. This semantic data could then be used to offer additional functionality during the presentation or while authoring in the form of rich queries or content recommendations. Some related work even explores the (semi-)automated assembly of presentations from existing semantically enhanced content [1, 20, 40]. Finally, we also point out a shift in *user roles*. In a classic presentation scenario the users are usually classified as a presenter or as part of the audience. However, if we take current advancements into account, this classification becomes a lot more complex. Content reuse and collaboration imply that content might be owned and authored my multiple users and concepts

[3] https://www.microsoft.com/en-us/download/details.aspx?id=28558
[4] https://impress.js.org
[5] https://revealjs.com
[6] https://www.gapminder.org/tools/
[7] https://www.google.com/slides/about/

such as ownership and access rights become more relevant. Also at the time when the presentation is given audience response systems allow audience members to participate in for instance quizzes via clickers or mobile devices [12]. In some cases these tools will keep track of individual results which breaks down "the audience" into a set of individual users. MindXpres further motivates the need for additional instructor roles and subgroups of audience members [32].

2.2 Existing Presentation Content Models

Common presentation tools store content in monolithic documents that are, for example, based on PresentationML (part of the OOXML standard), Apache's OpenOffice Open Document Format (ODF), the Simple Standards-based Slide Show System (S5)[8], slideDeck.js [37] or the W3C's Slidy [25]. These document formats are tailored to the classic presentation paradigms that they enforce, the most obvious one being the linear sequence of slides. However, some recent presentation solutions use paradigms that are not based on slides which makes it difficult or impossible to represent their content in such a document format. Further, the commonly used OOXML and ODF formats are restricted in the supported content types and can only represent basic types such as text, images, charts, videos and audio. Other than the models of the document formats mentioned above, we did not encounter any formal models made specifically for presentations. However, there are relevant conceptual frameworks, markup languages and content models for learning material in general with concepts that could or have been applied to presentations. Projects such as CNXML, PCML [16] and eLML [8] are based on models that define a fixed set of media objects, semantic types (e.g. "definition" or "exercise") and layout specifiers that have corresponding tags in their XML-based languages. As XML-based markup languages they are difficult to extend and they do not focus on reusing or sharing content. More relevant are content models such as SCORM, NCOM, NETg, Learnativity, CISCO RLO/RIO, dLCMS, New Economy, PaKMaS/LMML or SLM. For a detailed description and comparison of these models we refer to [39, 42]. Although each of these models is unique and has different characteristics, the general idea is more or less the same. Each model defines a set of media objects (e.g. text, image or video) that can be aggregated into a multi-level hierarchy (e.g. some text objects are grouped into a chapter which is part of a lesson which is part of a course). Some content models also deal with pedagogical aspects such as learning objectives, prerequisites or roles and most of them provide a way to define semantics for media objects or their aggregations (e.g. "exercise", "definition" or "example"). It is interesting to point out that these models enable reuse at different granularity levels. For example, one can reuse an entire lesson, a specific exercise or just a single image. It is also important to mention the IEEE Learning Object Metadata standard (LOM) [41], which is used by other content models such as SCORM and defines syntax and semantics to describe learning objects via their metadata.

2.3 Conclusions Drawn from Related Work

It is clear that presentation tools are slowly evolving. We see improvements in terms of content visualisation, navigation, content management and collaboration. There are some contributions that focus on individual aspects and provide stand-alone technical solutions. However, these solutions are often stand-alone technical tools and their functionality cannot always be represented in existing presentation models. To the best of our knowledge, there is no formal presentation-specific model that is generic enough to support the state-of-the-art presentation functionality. Content models for educational content in general can be used to draw inspiration, but they cannot be directly applied to presentations since they include concepts that are not relevant for presentations and lack some core components to enable higher-level features. For this reason we argue that a

[8] http://meyerweb.com/eric/tools/s5/

new unifying content model is needed that covers the functionality of existing work but that is also generic enough to support features that have yet to be explored. In order to provide an overview over the currents trends in presentation functionality, we summarise the features of the discussed next generation presentation tools and highlight them in Table 1.

Table 1. A summary of high-level features of academic next generation presentation tools, used as minimum requirements for our unifying content model

	Complex Content Structures	Non-linear Navigation	Complex Content Types	Content Reuse (within presentations)	Content Reuse (across presentations)	Collaboration	Semantics	User Roles and Access Management	Automatic Context Adaptation
Fly [18]	✔	✔							
iMappping [11]	✔	✔					✔		
NextSlidePlease [36]		✔							✔*
HyperSlides [6]	✔	✔							
MultiPresenter [17]				✔					
Palette [23]		✔							
PaperPoint [35]		✔							
Slithy [43]			✔						
SliDL [3]					✔				
SidePoint [19]					✔		✔		
Drucker2006 [5]					✔				
Webstrates [14]						✔			
CPoint [15]							✔		
Moscovich2004 [22]	✔	✔							
SlideWiki [13]				✔	✔	✔		✔	

* NextSlidePlease only adapts content automatically based on time constraints

Based on the functionality presented in Table 1 and lessons learned from content models for general educational content, we derived the following requirements (**R1**-**R6**) for our new content model. In Section 6 we revisit these requirements and detail how our content model and implementation fulfil these requirements.

- **R1: Extensible and Flexible Core Components** Existing models make certain assumptions about what a presentation or document should be and predefine a lot of concepts and semantics in their core. For instance, they limit the supported content types (e.g. text, images and video) and force a certain structure on the document (e.g. based on slides). However, related work shows that the learning process requires that content is well integrated in the greater whole, both structurally and visually [10], which requires more complex structures than a two-dimensional sequence of slides. In educational content models we also see the importance of multi-level content aggregation but it would be a mistake to fix the supported content types, aggregation levels, associations or other components. In order to unify current demands and also to provide support for future functionality we require extensible and flexible core components that can be adapted for a wide range of presentation styles.
- **R2: Fundamental Support for Reuse and Collaboration** Existing work shows that the reusability of content should not be ignored as it can save time and money as well as improve the efficiency and effectiveness of knowledge transfer [39]. Proper support for reuse and collaboration introduces many smaller requirements such as storing content at a small granularity and separating content and style. Studies on reusing presentation content also show a clear need for versioning, management of ownership and user access rights, and access to external content repositories [21].
- **R3: Extensible Semantics and Semantic Interoperability** We also saw that the semantic annotation of content is important. Even though many content models use the LOM standard for the metadata, there is no real consensus on the vocabulary used to name specific content units or their aggregated forms. This is, for instance, illustrated in ALOCOM [39] where the need for a mapping between existing models is motivated. We set the requirement that users of the model should be able to introduce their own semantics, which can still be one of the existing ontologies. As such, our model remains interoperable with existing models instead of trying to enforce yet another ontology specifically for our use case.
- **R4: User Roles and Access Management** Also some concepts related to *users*, *groups of users* and *roles* should be incorporated in the content model. Considering our vision, we would need to store user-related metadata on a much finer granularity. As a presentation might consist of many reused content fragments from various sources, it is important to keep track of who created what, who owns what, who made changes to what, and so on. In order to enforce *privacy* and *security* in the implementation layer, there should also be a notion of access rights in order that owners can decide who can view, change and reuse their content. The notion of roles is also important for run-time functionality as it can determine which interactions are possible for certain users.
- **R5: Complex Navigational Structures** In some content models the order of navigation is implied by for instance the order of the content. Other models do incorporate the concept of navigational paths but usually they are limited to navigational links such as those found on the Web. In order to fully meet current demands for non-linear presentations we require support for complex navigational meshes at the content level.
- **R6: Context-aware and Adaptive Presentations** Last but not least, we require support for representing context and context-aware reasoning. The need for context-aware content delivery is often mentioned [33] but has barely been explored in presentation tools. Some aspects such as how context is determined and the logic for context-aware reasoning is a runtime matter. However, the fact that context might influence various parts of a presentation should be represented in the document model. For instance, depending on a given context, the document structure, content, navigation or semantics might be adapted.

3 A CONCEPTUAL FRAMEWORK FOR PRESENTATION SOLUTIONS

In order to conceptually represent the new features seen in state-of-the-art presentation solutions, we have abstracted and unified them in a single *generic conceptual framework for presentation solutions*. The proposed framework is interoperable with established as well as next generation presentation paradigms and includes abstractions for currently unsupported concepts. We have identified four distinct categories of concepts that describe presentation concepts such as those presented in Section 2. The five conceptual layers of our framework are the *Content*, *Structure*, *Semantics*, *Navigation* and *Users* layers as highlighted in Figure 1.

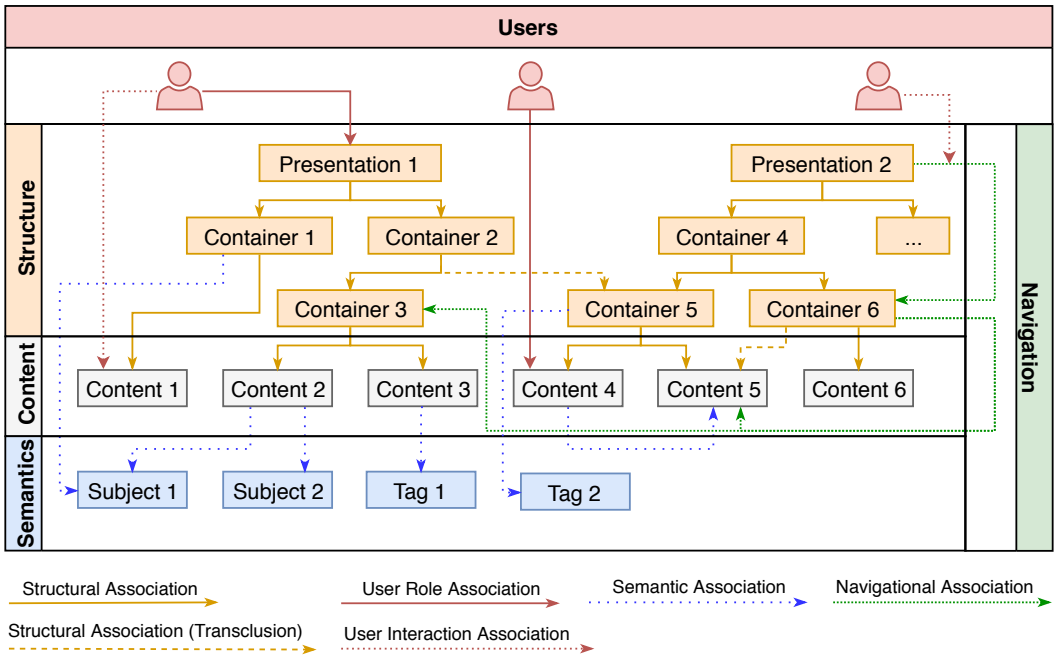


Fig. 1. Five conceptual layers: *Content*, *Structure*, *Navigation*, *Semantics* and *Users*

The *Content* represents individual pieces of content, such as images, videos, pieces of text, source code or collections of raw numeric data, that form the basic building blocks of any presentation. From related work in educational content we learned about the importance of multi-level content aggregation. This concept is represented in the *Structure* layer of our conceptual framework. Existing solutions already support some form of content aggregation, the most well-known form being the slide, which is in essence a container that conceptually and visually aggregates some content. However, in order to support other paradigms, we use the concept of nestable containers. For example, containers could be used to express that some image content is grouped into an image gallery container forming part of a slide container, which is part of a presentation. The same concept can also be used to express nested spatial regions on a 2D canvas for a ZUI, or to collect slides into "chapters" for a book-like presentation structure.

In order to compose any of these presentation structures from a set of content and containers, we define a specific type of association, the *Structural Association* represented by solid orange arrows in Figure 1. Structural associations are used to connect content and containers into a graph-like presentation structure. Nodes in a presentation structure can be a mix of content and containers, and

in the rest of this section we will use the term *node* if the difference is not relevant to our narrative. Any node (potentially a substructure) might be included in more than one presentation via these structural associations. Under the assumption that the content instances are easily available across computers, content can be integrated in other presentations by simply pointing to it, rather than creating a duplicate. This form of "inclusion by reference" was coined as *transclusion* by Nelson [24]. Radically different presentation structures and substructures can be built on top of a set of potentially distributed content. Also someone else's substructures (e.g. a slide or a chapter) can be transcluded into another presentation by using structural associations to make the corresponding container a part of that presentation's structure (represented by dashed orange arrows). We identify two main issues with this type of flexible reuse and sharing. In order to avoid unwanted external changes to transcluded nodes, it might be desirable to include a specific version of the node rather than always the latest version, which requires *versioning*. The sharing implies issues such as *ownership*, *access rights* and *privacy*. These are not represented in the conceptual framework as we classify them as part of the *Structure* and *Navigation* layers and discuss them in detail in Section 4.

The *Navigation* layer classifies concepts such as navigational paths or pre-defined state changes via the *Navigational Association* (represented by dotted green arrows). Traditional linear sequences of slides can be represented by chaining slide containers with navigational associations, but also complex graph-like structures as in NextSlidePlease [36] or HyperSlides [6] are possible. The sources and targets of navigational links can be both containers or content, making them applicable for a wide range of scenarios. They can mimic simple hyperlinks as seen on the Web, as well as represent a predefined spatial path on a 2D canvas in a ZUI. Another application worth mentioning is their use for representing state changes. For instance, different states of a data visualisation can be chained by navigational associations. Regardless of how they are used, any node in the navigational graph can have more than one outgoing link and the path to be followed can be decided by the presenter or by external factors such as time constraints.

Semantics are usually not an important aspect of the commonly used presentation tools but we see interesting use cases in related work as well as in the domain of the educational content. The *Semantics* layer aggregates concepts related to annotating nodes and their relationships with semantic metadata. We define the *Semantic Association* (represented by blue arrows) which allow semantic associations between nodes, or semantic associations between nodes and semantic values. They can, for instance, be used to relate nodes to a specific subject or nodes might be tagged with a semantic value such as "proof" or "example". Nodes can also be directly associated to represent relationships such as "this other slide contains a more detailed explanation of this text".

The *Users* layer abstracts the relationships between users and the components of the other conceptual layers. We define two kinds of associations concerning users, the *Role Association* and the *Interaction Association*. The role association (solid red arrows) is used to specify situations where a user has a certain status in relation to specific content (or other association). Role associations can, for example, be used to represent the fact that a specific user is the author of certain content or containers, or that a specific user is a presenter for a given presentation. The second type of user association, the interaction association, is used to represent possible interactions a user can have with components of the other layers. For instance, certain users might have the ability to trigger navigational links while others might not. Similarly, an interaction association might specify that a user is able to change the visualisation style of some content during the presentation (e.g. switching from a table representation to a chart visualisation). Note that these concepts are optional and offered for these cases where this level of expressiveness is required in a content model. Applications might, for instance, provide default interactions without reflecting this in the underlying model, or only special cases that override the defaults might be modelled.

Finally, we would like to clarify the absence of concepts related to *context, layout, styling and visualisation*. Given how easy it becomes to reuse and seamlessly integrate content, it is important to keep content and visualisation-specific details separated. We argue that visualisation-specific concepts are a part of the *Structure* layer. Layout and style-related metadata (e.g. fonts, colours or absolute positions and offsets) can be different every time a node is included in a presentation structure, so it should be attached to the structural association that makes a node part of a presentation. This way, style-related metadata is not associated with a node directly, and different metadata can be applied every time the same node is reused. It can also be argued that style and layout only become relevant at the application level and not at the content level. The same can be said about context-resolving. Context is determined at runtime and it can affect every aspect of every layer (semantics, content, structure, users and all the related associations). Nevertheless, in order to provide a complete solution we included concepts related to layout and context in the resulting content model discussed in the next section.

4 A GENERIC CONTENT MODEL FOR PRESENTATION SOLUTIONS

The presented content model is based on the Resource-Selector-Link (RSL) hypermedia metamodel [34] and we start with a brief introduction to the RSL metamodel. We then describe some extensions that were made to the core and introduce our content model as a domain-specific application of RSL.

4.1 The Resource-Selector-Link (RSL) Hypermedia Metamodel

The RSL hypermedia metamodel offers concepts to represent content objects, complex structures of content as well as relationships between them and has served as a foundation for various domain-specific models and applications. Compared to other hypermedia models, RSL treats links as extensible first-class citizens and offers advanced concepts such a media-specific selectors, context-aware links or user access management, which is also the reason why we decided to base our content model on RSL. For details about the RSL hypermedia model and a comparison with other hypermedia models we refer to [34]. Note that while the original RSL metamodel has been presented in OM notation [34], we present the relevant RSL components in the UML class diagram shown in Figure 2.

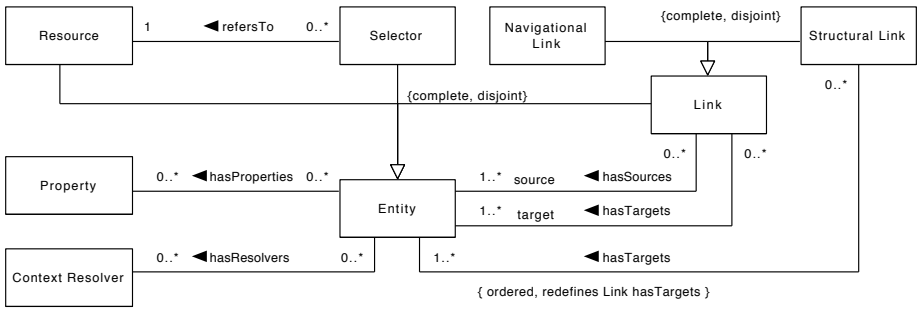


Fig. 2. Core RSL components

The core of the metamodel is formed by the `Resource`, `Selector` and `Link` classes which are all subclasses of the `Entity` class. As `Entity` subtypes, every instance of these classes has an associated set of key/value properties which can be used to flexibly store application-specific (meta)data. The `Resource` class can further be subtyped to model various types of basic content objects such as

text, images or videos. A `Selector` is a powerful concept allowing users to create subtypes for addressing smaller parts of a resources. For instance, a movie selector might allow a user to point to a specific time interval within a movie. The `Link` class is used for any kind of relationship between entities and offers some unique features. Different from most other hypermedia models, links are bidirectional and can have multiple sources (entities) and targets (entities). As an `Entity` subtype, links are considered first-class citizens and can for instance point to other links or benefit from the user management functionality described later. The core RSL model defines two types of links, the `Navigational Link` and the `Structural Link`. While navigational links are used to represent navigational paths through content (e.g. hyperlinks on web pages), structural links are used to aggregate content (e.g. chapters of a book). Therefore, the targets of structural links are always ordered (e.g. to maintain the order of chapters in a book). Note that from now on we use the terminology "linking" or "pointing" in the context of RSL links. In order to support user access and ownership, RSL offers the possibility to define permissions for any entity (links, resources or selectors) as shown in Figure 3. By combining whitelisting as well as blacklisting, permissions can be set for individual users, groups or any combination thereof. Last but not least, `Context Resolvers` can also be used for allowing or disallowing access to an `Entity` based on other contextual factors.

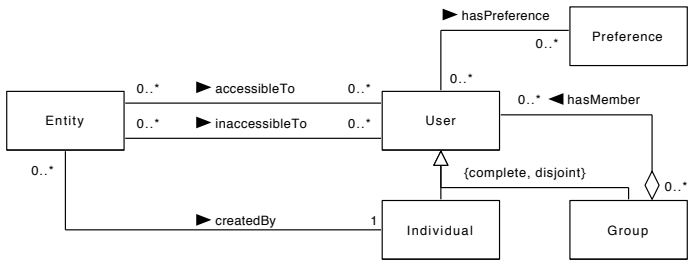


Fig. 3. User-related RSL components

4.2 RSL Extensions

We introduced some minor extensions for the core RSL model which are reflected in the final content model shown in Figure 4. First, if the flexible sharing and reuse of content is supported, we need to take into account that transcluded content might be modified by a third party, resulting in some unexpected changes. In order to address this issue, a simple form of versioning has been added to the RSL metamodel. When an entity is updated, a copy is created and the previous version is connected to the new version via the `hasPreviousVersion` association. Over time, a doubly linked list of versions is built for each entity. The order and numbering of the versions is implicitly encoded as part of the list but for convenience, any entity in the chain might store a pointer to its latest version (the last node in the chain) via the `hasLatestVersion` association. The idea is simple, yet powerful; depending on the use case, users can point to a specific version of an entity by directly linking to the desired version of the entity, or they can choose to always let the link resolve to the latest version. Also note that versioning is implemented at the entity level, allowing versioning of structural, navigational and semantic components as well. Second, RSL allows us to define properties for a `Link` instance, but there is no mechanism to attach different properties to individual targets of a multi-targeted link. We address this issue by allowing the `Link`'s `hasTarget` association to have its own set of associated properties, allowing different properties for each individual target.

4.3 A Domain-specific RSL Application for Presentation Solutions

In Figure 4 we present our content model based on the previously presented conceptual framework. In order to keep the model readable, only modified and added components are shown, but all the concepts and associations shown earlier in Figure 2 and Figure 3 still apply. While the core components of the presentation model are coloured in green, examples of application-specific extensions are shown in red.

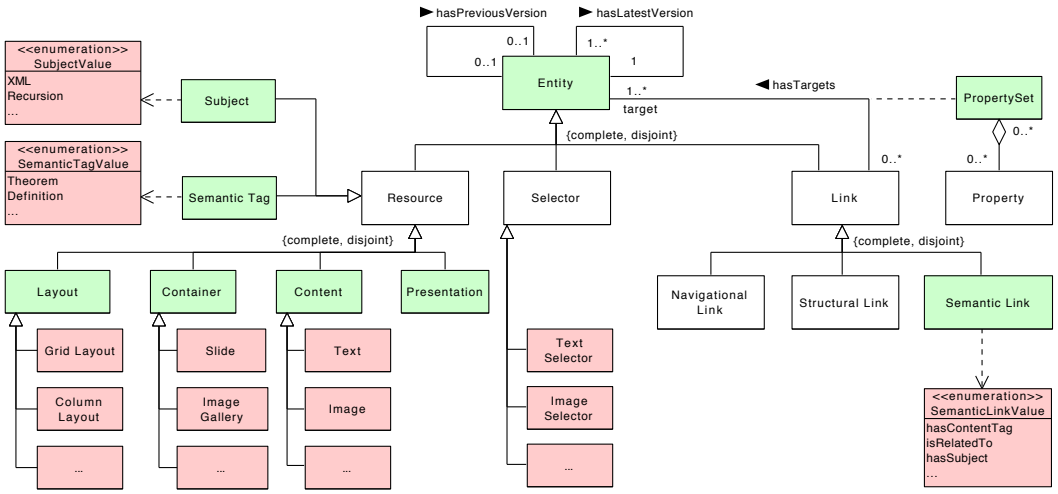


Fig. 4. A domain-specific RSL application for presentation solutions

The new `Content` resource acts as a superclass for any content instances (e.g. text or image) forming the basic building blocks of a presentation. The `Content` resource has to be extended for any content type that should be supported, allowing users to model the specifics for each content type separately. `Content` instances can be aggregated into larger structures by using containers. The new `Container` resource can be subtyped to model typed aggregations of content (or other containers). For instance, it could be used to model the concept of slide containers used in slideware or to represent clusters of content in a ZUI. A bullet list can also be modelled as a container as it is an ordered collection of text fragments or sublists.

Objects forming part of a container are associated with the container via a single `Structural Link` with one source (the container instance) and multiple ordered targets (e.g. content instances). Containers can also contain other containers, which is useful to, for example, group slide containers into "sections", or to allow slides to contain subcontainers such as an image gallery (a container for image resources). This allows multiple levels of aggregation to support all kinds of presentation structures and matches the concept of layerable typed aggregations often found in educational content models. No matter how the content model is extended, the result is always a presentation and therefore we predefine the concept of a `Presentation` as a container-like aggregation of (structured) content. On the other hand, `Container` has no predefined subtypes in the core model and should be extended by users to match their application-specific paradigms (a presentation might not be based on slides).

A `Container` can only contain content, selectors or other containers, but this constraint could not easily be modelled in UML as children are not directly associated to the container, but via an instance of a `Structural Link`. Since structural links are also used to create structures of other types, the constraints cannot be modelled on the structural link. Instead of a structural link subtype

for every required combination of sources and targets, we decided to keep the model readable and present these constraints in Table 2.

Table 2. Constraints between nodes of a presentation structure

Entity Type	Allowed Targets of the Structural Link	Min	Max
Presentation	{Container, Selector, Content}	0	*
Container	{Container, Selector, Content}	0	*
Selector	∅	-	-
Content	∅	-	-

The `Layout` resource allows users to create subtypes for reusable layouts. For instance, one could define a `Column Layout` or a `Grid Layout`. When content or substructures are incorporated in a presentation structure, a layout instance can optionally be attached to the corresponding structural link. This way, different layouts can be applied every time the content or substructure is (re)used. Note that the targets of structural links are always ordered and layouts should use this implicit information to position a link's targets accordingly (e.g. the second target of the structural link goes in the second row of the row layout). Layout implementations should not use absolute dimensions, positions or distances as they should be able to adapt to the available space. This way, a layout can also be reused and seamlessly integrated in other presentations even if different container types are used.

Style data, including visualisation-specific data such as fonts, colours and themes should also be kept separate from the content so that content can seamlessly blend into other presentations when it is reused. For this reason, styling data should also be attached to the structural link every time the content is included in a presentation structure. This way, different style data can be applied depending on the context. Concretely, the data itself should be stored in the property set associated with each structural link (inherited from `Entity`). This allows many different styling approaches to be implemented, including CSS-like systems where styling rules can be inherited from containers higher up in the presentation structure.

RSL's `Navigational Link` is used to represent navigational paths over the presentation structure. In the simplest case, navigational links could be used to lead a user from slide to slide to recreate the classic sequence of slides. However, they can also be applied to create complex non-linear presentations. First of all, navigational links can point to any container or content so one could define a path from a slide to an individual image to a "section" container. This is needed to represent navigational paths used in ZUIs such as Prezi. Furthermore, navigational links can have multiple targets and a path may offer different routes that can be followed by the presenter during the presentation or chosen by external factors such as the audience type or time constraints [6, 36]. Finally, sources of navigational links can also be any container or content, so they could be used to implement classic hyperlinks (e.g. the users clicks on a particular word or image to navigate to related material). Many presentation solutions offer transition animations when navigating content. Navigating from one slide to another, or from one location in a ZUI to another location, or going from one state of your data visualisation to another state is all modelled with navigational links. Therefore, if transition animations are desired, they should be specified in the properties of the navigational links. This might be the name of the effect that should be applied or it could be more complex data such as velocity or arc height for spatial navigation in a ZUI.

In terms of an ontology for semantic metadata, we have previously stated that there is no real consensus and it is therefore better to be interoperable with existing as well as future ontologies.

First of all, the new `Semantic Link` is used to represent semantic associations. One should be able to type these links to distinguish between associations such as *"is related to"* or *"has subject"*. However, in contrast to previously-described extensibility, users should not create new types of semantic links by subtyping. The goal is to simply *tag* a semantic link with a specific meaning. For that we deem it unnecessary to create subtypes as no additional functionality or information would be added to any subtypes. Instead, we use the semantic link's property set (inherited from `Entity`) to store the semantic meaning of each instance. The `Semantic Tag` resource is a similar concept that allows users to tag content or structures with semantics such as "theorem" or "definition". These semantic tags are entities themselves and a `Semantic Link` should be used to connect for example a content instance with the instance of the semantic tag that representing a "theorem". As semantic links can have multiple targets, content or structures can be associated with multiple semantic tags. RSL links are also bidirectional, so by making a new entity instance for each tag (instead of abusing properties as mentioned earlier) we can easily provide answers to queries such as *"find all theorems"* by following the associations from the "theorem" tag to the content or structure. The `Subject` resource works in the same way and allows content or structures to be associated with one or more specific subjects. Semantic tags and subjects are also not extended by subtyping but by setting their meaning in the properties of each instance. The use of entity properties for storing the semantic meaning of `Semantic Link`, `Semantic Tag` and `Subject` instances makes them very easy to extend. As the model does not need to be extended to add new semantic types, there is also no need to make changes to the database schema when new semantics is added and semantic types can be treated in a uniform way in the application layer. More importantly, the chosen design makes our model interoperable with existing content models. An existing ontology can easily be adopted by using the same vocabulary for the semantic link and tag values. Ontologies might also be combined and used next to each other, or a custom ontology might be used and mapped to other ontologies in the application layer.

As a last feature, we want to discuss the ability for end users to "invisibly" annotate content with their own text. The most common use for this in existing solutions is the ability to write presenter notes for every slide. In our model, we again use `Entity` properties to store this data. This way, annotations can not only be stored for containers such as slides but also for higher-level structures as well as individual pieces of content if desired.

5 IMPLEMENTATION

In order to demonstrate the viability of the model in practice, we have implemented it as a functional information system for presentation content. Our model is highly extensible and offers a lot of freedom on the conceptual level, but it would be inconvenient if large parts of the implementation would have to be refactored for every application-specific presentation model. To address this issue, we designed the implementation as a reusable model-driven implementation of the RSL hypermedia metamodel. Our implementation takes any RSL-based model and offers the related functionality automatically without the need for adding any logic. This link service was developed in Java 8 as an embeddable software library together with an optional server wrapper that turns it into a stand-alone link server. The general architecture of the link service is shown in Figure 5. The library defines core classes for the core RSL components (e.g. resources, selectors, links and users, presented earlier in Figure 2 and Figure 3). These classes implement all described functionality, for instance for property management, linking, user access and context resolvers. In order to use the library, the user should provide a model that builds on these RSL components. A model is specified in a custom JSON-based format and specifies the application-specific `Resource`, `Selector` and `Link` subtypes and their properties. Optionally, users can also provide certain linking restrictions for link subtypes, specifying which model components are allowed as sources or targets for this

type of link. After a model has been provided, the library uses JavaPoet[9] to generate Java classes for the components defined in the provided model. On the technical level, these new classes are subtypes of the predefined core classes and inherit a major part of their core functionality. The new classes are then dynamically compiled and (re)loaded at runtime. This allows models to be added, removed or modified at runtime, and the server will automatically generate and load optimised code for representing the currently loaded models. The library uses generics and reflection to provide high-level methods to interact with instances of new classes at runtime. This includes the creation and deletion of instances, linking them, checking access rights or resolving context. In order to persistently story objects, the library defines an abstract persistence interface that allows different storage backends to be plugged in. The reason for this is that there are many database types that can be used (e.g. graph databases or object databases) and depending on how the system is used a different back-end may be more performant. This initial implementation is backed by ObjectDB[10], allowing us to directly map in-memory instances of the models to the persistence layer. The RSL implementation further provides some utility functions for querying content, for instance based on property values or object type. A basic crawler is provided which allows users to crawl the graph of linked data and aggregate content of interest (e.g. by type). This is, for example, used to efficiently answer queries such as *"get all content related to this presentation"*. Google Guava's EventBus[11] is used in a publish-subscribe engine that coordinates the various components of the framework, allowing them to work asynchronously and potentially in parallel.

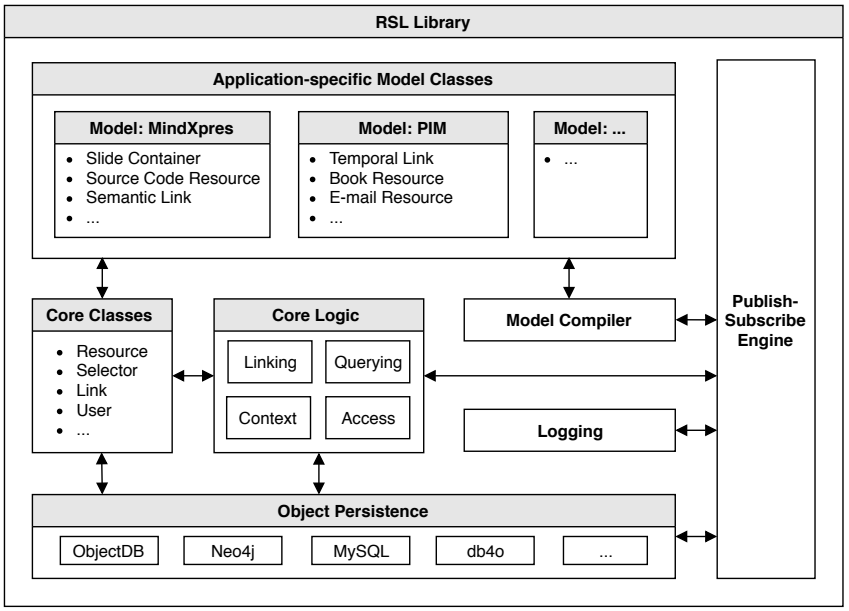


Fig. 5. Architecture of the model-driven RSL link service implementation

The library is provided as a set of JAR files that can directly be embedded in another application. Although this approach gives the best performance, it requires the application to either use the same technology (Java VM) or complex language bindings, and it does not allow multiple applications

[9] https://github.com/square/javapoet
[10] https://www.objectdb.com
[11] https://github.com/google/guava/wiki/EventBusExplained

to access the same content repository. For this reason we also provide a server wrapper that adds additional server functionality on top of the library and allows for client-server architectures. The architecture of the server wrapper is shown in Figure 6 and further builds upon the RSL Library. The communication with clients is realised via an abstract interface that allows different communication interfaces to be plugged in. Implementations for TCP/IP, WebSockets and RESTful interfaces have been created and can operate in parallel. These communication interfaces receive requests from clients, translate them to a shared internal representation, forward them to a request handler, and then translate responses back to the medium-appropriate format before sending them back to the client. Also here the publish-subscribe engine is used for asynchronous and parallel request handling. Higher-level application-specific functionality can be added to the server via plug-in microservices. For instance, in the case of presentations it would be useful to have a method that sets some metadata on all components of a specific presentation, instead of having to make hundreds of individual requests. Finally, the server wrapper also has a file storage component allowing referenced content such as images or video to be stored and shared via the server. The request handler uses reflection to map requests on specific microservices, objects and methods. It is possible to copy specific content between servers, but also real transclusion is supported. When a reference is made to external content, a "proxy" entity is inserted locally that points to the external location where the entity, and attached entities, can be retrieved via the public API when needed.

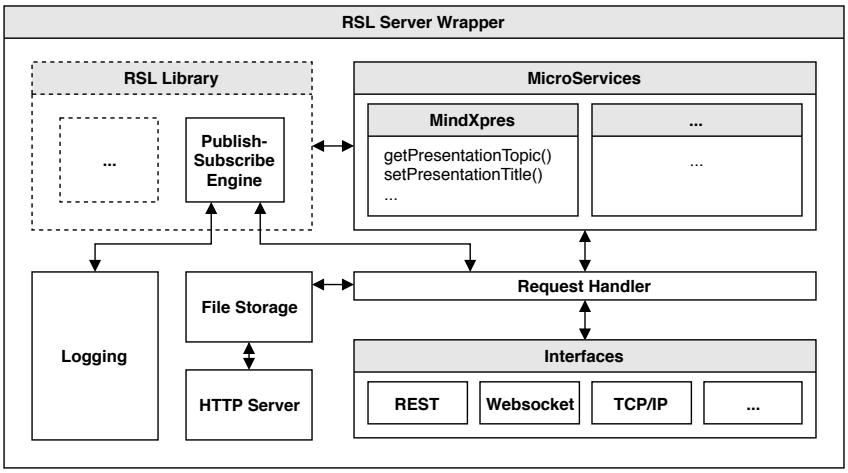


Fig. 6. Architecture of the server wrapper for the RSL software library

In order to be generic enough for any RSL-based model, the library is a rather literal translation of the metamodel, also on the technical level. For instance, versioning is achieved by maintaining a doubly-linked list of entity instances, as described conceptually. In the future this might be further optimised by, for instance, using delta compression for resources such as text, storing only the differences between versions. Context resolving is also implemented in a basic way, as it is described in the core model. In the RSL library a context resolver is essentially a callback to a user-provided method, delegating the context resolution to user-provided application logic which should determined if the object is accessible or not in the current context. In the server wrapper, context resolvers can be mapped to methods provided in the microservices which is the recommended approach. In the future a more complex system will be investigated. For example, the AHAM hypermedia model [4] defines a custom syntax for defining rules. This would allow some context resolution rules to be provided directly in our JSON models.

6 DISCUSSION AND EXAMPLE USE CASES

In Section 2.3 we identified the lack of a content model suitable for our goals and defined a set of requirements for a unifying and future-proof content model for presentation content. We revisit these requirements and explain how they are fulfilled in our content model and implementation. For some requirements we also provide some examples to further illustrate the mentioned concepts.

R1: Extensible and Flexible Core Components The content model defines generic reusable components that can be extended to match the requirements of many different presentation styles. The model can represent classic presentations by using slide containers, but other types of containers can be created for a zoomable canvas presentation, a presentation where content is aggregated into book-like chapters, or any other imaginable structure. In the same way, layouts, content types and semantics can be filled in as required. These customised and extended presentation models are easily translated into a working application through our model-driven link service for application-specific RSL models. Figure 7 shows an example instance of a model that represents one of the many MindXpres [31] presentation styles. The parts of the model annotated with numbers are similarly highlighted in the screenshot of the resulting presentation on the right. In this case the presentation uses a hybrid visualisation approach, where a zoomable canvas is combined with classic slides. Multiple levels of content aggregation and layouts are used: "sections" are laid out in rows and each section contains a grid of content which can optionally be grouped in slides. Note the use of a selector which is used to include only a part of the image in the presentation. For reasons of clarity, concepts such as transclusion, users, context and navigation have been left out in this example.

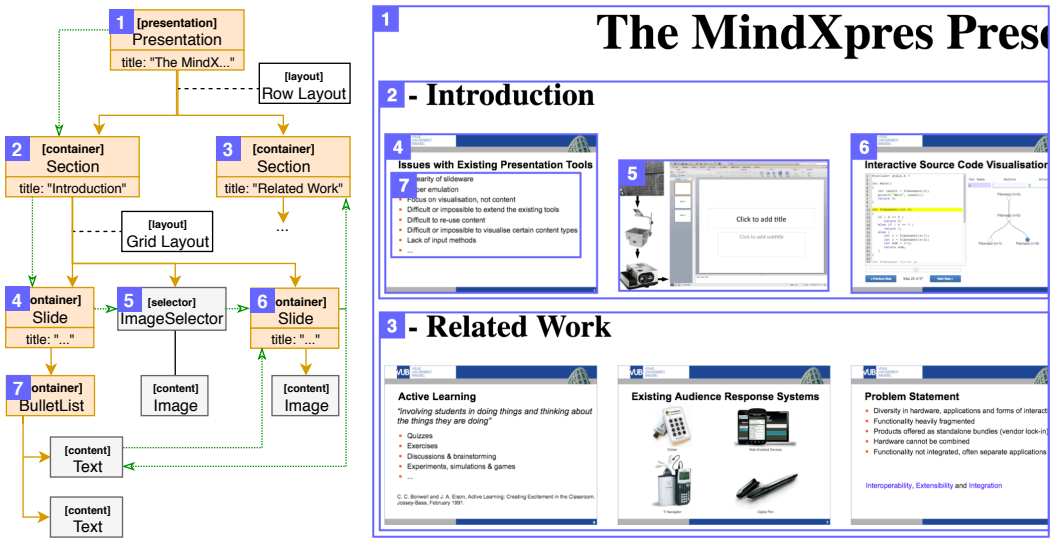


Fig. 7. Model instance and annotated screenshot of the resulting hybrid canvas and slide-based zoomable MindXpres presentation

R2: Fundamental Support for Reuse and Collaboration In contrast to some related work, reuse and collaboration is not added as an afterthought and all related concepts are present in the core of the content model. Content is stored at a fine granularity and can be transcluded across presentations, even across content repositories. We extended RSL to support versioning and RSL's user-related components provide the concepts needed to deal with other issues such as user access management and ownership. These features apply to all entities (including links), which is not always the case in other systems. Support for the envisioned reuse and collaboration was also achieved in the implementation. Figure 8 illustrates how transclusion via structural links (shown with dashed orange lines) enables flexible reuse of content and structures. Presentation A reuses a slide and its content from another presentation in the same content repository. Presentation A also reuses an image from an external repository, but an image selector is used to extract only a part of the image for inclusion. A user might even apply a colour filter to the extracted image region, without requiring duplication and without affecting the original image. Presentation A further includes another slide from an external repository. Multiple users can operate on the same content repository—which is not necessarily located on their own computer—allowing for collaboration.

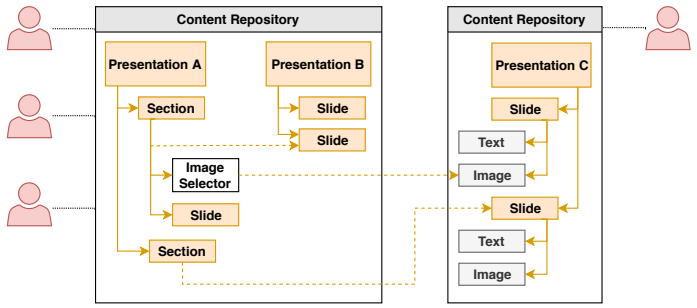


Fig. 8. Collaboration and reuse of content, within and across content repositories

R3: Extensible Semantics and Semantic Interoperability We provide semantic associations as well as semantic tagging to allow entities and their relations to be semantically annotated. To avoid shortcomings of other document models, we do not enforce a specific ontology. Instead, semantics can be extended to either match existing ontologies or to implement a custom vocabulary. Also note that semantics can be applied on a small granularity (e.g. individual content fragments or associations), avoiding shortcomings of some existing models [7]. Figure 9 shows an example where various parts of a presentation structure are semantically annotated. For the sake of the example, a mixed vocabulary from different existing ontologies is used. Parts of the presentation can also be directly associated for semantic purposes, for instance via a `isRelated` association, a custom typed extension based on the `Semantic Link`. The flexible semantic annotations enable functionality related to semantic queries or content recommendations for presentations, and the flexible vocabulary makes the presentation content interoperable with existing systems.

R4: User Roles and Access Management RSL's user components (shown in Figure 3) allowed us to easily fulfil this requirement and they are also provided in the implementation to further support other requirements such as R2. This functionality is for instance used to keep track of creators, authors and owners of content (and their versions) and also allows users to decide who can view, edit or copy their content. This satisfies currently unfulfilled user requirements identified by studies such as [21]. Note that the context adaptation described in R6 can make use of this information to adapt presentations to specific people or groups.

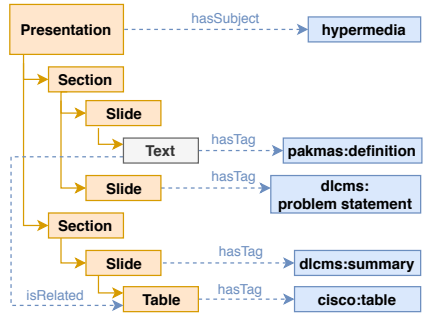
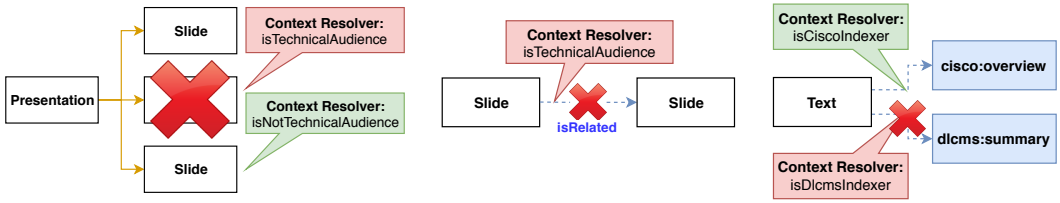


Fig. 9. Semantically annotated presentation content

R5: Complex Navigational Structures The model's `Navigational Link`, based on the generic but powerful RSL `Link` makes it possible to represent the complex navigational meshes required by some tools. The concept of selectors also allow arbitrary small subparts of any resource (not just text) to act as sources or targets of links ("anchor points"). Figure 7 illustrates some example navigational links in green. How they are visualised and triggered at runtime is application specific and up to the developers.

R6: Context-aware and Adaptive Presentations Last but not least, RSL's context resolvers allow us to control access to any entity based on the current context. Basic support for content resolvers is also provided in the implementation allowing a presentation to adapt content, structure, navigation or semantics. This allows presentations to go beyond related work such as NextSlidePlease [36] which only adapts to time restrictions. Figure 10 shows three examples of how context resolvers can be used to adapt a presentation based on contextual information. In Figure 10a the document structure is adapted based on the background knowledge of the audience. Assuming a non-specialist audience for this scenario, the second slide is not part of the presentation while the third one is. Note that context resolvers can also be placed directly on links, and a context resolver on a structural link might also affect the document structure. Similarly, navigational or semantic links can be disabled in certain contexts. This is demonstrated in Figure 10b where two slides are only considered related (a semantic association) in the case of a specialist audience. Figure 10c shows a scenario where the semantics of a piece of context can change. In this scenario the resource is tagged with similar semantics, but from different ontologies. Depending on the capabilities of the content indexer (e.g. the ontology that it supports) different semantic tags are given.



(a) Adaptive document structure (b) Adaptive associations (c) Adaptive semantics

Fig. 10. Adapting presentation content, structure, navigation or semantics based on context

7 FUTURE WORK AND CONCLUSIONS

We have seen that existing conceptual frameworks and content models for presentations are often based on outdated presentation paradigms and cannot represent next generation presentation concepts seen in the state of the art. In order to address this issue, we have introduced a conceptual framework for presentations which unifies classic as well as next generation presentation concepts. Based on our conceptual framework we defined a new flexible and extensible content model for presentations. Our model is interoperable with classic as well as new paradigms and its generic concepts can be extended to support emerging presentation paradigms. We further ensured that sharing, reuse and collaboration is not an afterthought by offering the necessary concepts in the core of our content model. Recent trends in educational content have inspired us to integrate semantics in the core of the model, which offers a lot of unexplored potential in the context of presentations. Special care has been taken to also make the semantic aspect interoperable with existing ontologies which not only makes content exchange possible with other RSL-based systems, but also with platforms for educational content such as e-learning repositories. The presented content model can represent existing as well as future presentation paradigms, and further serves as a platform for new functionality on the application level. The powerful content collaboration and reuse mechanisms help knowledge workers to manage presentation content more effectively and reduce storage space. Support for semantics can change the way presentations are navigated, queried and authored. For instance, if a presenter selects a complex word during the presentation, the tool might pop up its definition or suggest a navigational path to slides with more details. At authoring time, the tool might also automatically suggest internal or external related content, as briefly explored by SidePoint [19]. Adaptation to context can further improve knowledge transfer [33], for instance by adapting content to the background knowledge of the audience.

We have implemented a new model-driven RSL content server, allowing us to use the presented content model as back-end for MindXpres presentations and to develop novel end-user solutions. In terms of future work, we already mentioned how the implementation can be further optimised. For instance, adding delta compression for content versioning might increase the storage space savings introduced by our content reuse mechanisms. We also provided the option to easily replace the storage back-end for the object persistence component so that we can experiment with different database types and database-specific optimisations to improve the performance for our use case. As detailed before, also the way context resolution is performed can be improved, for instance by providing a custom syntax for defining context rules [4] in our JSON-based model definitions. One of the biggest future challenges is to bring the complex functionality of the content model to end users via authoring tools. Currently, MindXpres presentations are authored in a declarative XML-based language and compiled into portable stand-alone presentation bundles, similar to how LaTeX works for text documents. As a next step, we plan to investigate the abstractions and interactions needed to bring the complex hypermedia concepts of the model to a user-friendly graphical presentation editor.

Our contributions, including the implementation, serve as a research platform for further MindXpres functionality but also aid other researchers and developers. Currently, too many prototypes have to be implemented from scratch since existing presentation tools cannot easily be extended for radically different approaches. In combination with the extensible MindXpres visualisation runtime [31], we provide a flexible and extensible solution which can be used to rapidly prototype and evaluate new presentation ideas, allowing the community to focus on the creation of interesting new use cases.

REFERENCES

[1] Payam M. Barnaghi and Sameem Abdul Kareem. 2006. A Flexible Architecture for Semantic Annotation and Automated Multimedia Presentation Generation. In *Proceedings of the 1st International Conference on Semantic-Enhanced Multimedia Presentation Systems (SEMPS 2006)*, Vol. 228. Athens, Greece, 72–87.

[2] John Brand. 2004. Presentation (Mis) management: Content and Collaboration Strategies. *Delta* 3057 (2004).

[3] José H. Canós, María Isabel Marante, and Manuel Llavador. 2010. SliDL: A Slide Digital Library Supporting Content Reuse in Presentations. In *Proceedings of the International Conference on Theory and Practice of Digital Libraries (TPDL 2010)*. Glasgow, UK, 453–456. https://doi.org/10.1007/978-3-642-15464-5_55

[4] Paul De Bra, Geert-Jan Houben, and Hongjing Wu. 1999. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In *Proceedings of the Tenth ACM Conference on Hypertext and Hypermedia: Returning to Our Diverse Roots: Returning to Our Diverse Roots (Hypertext 1999)*. Darmstadt, Germany, 147–156. https://doi.org/10.1145/294469.294508

[5] Steven M. Drucker, Georg Petschnigg, and Maneesh Agrawala. 2006. Comparing and Managing Multiple Versions of Slide Presentations. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology (UIST 2006)*. Montreux, Switzerland, 47–56. https://doi.org/10.1145/1166253.1166263

[6] Darren Edge, Joan Savage, and Koji Yatani. 2013. HyperSlides: Dynamic Presentation Prototyping. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2013)*. ACM, Paris, France, 671–680. https://doi.org/10.1145/2470654.2470749

[7] Emmanuel Fernandes, Hend Madhour, Sami Miniaoui, and Maia Wentland Forte. 2005. Phoenix Tool: A Support to Semantic Learning Model. In *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies (ICALT 2005)*. Kaohsiung, Taiwan, 948–949. https://doi.org/10.1109/ICALT.2005.220

[8] Joöl Fisler and Susanne Bleisch. 2006. eLML, The eLesson Markup Language: Developing Sustainable e-Learning Content Using an Open Source XML Framework. In *Proceedings of the 2nd International Conference on Web Information Systems and Technologies (WEBIST 2006)*. WEBIST, Setúbal, Portugal. https://doi.org/10.5167/uzh-77757

[9] Lance Good and Benjamin B. Bederson. 2002. Zoomable User Interfaces as a Medium for Slide Show Presentations. *Information Visualization* 1 (2002), 35–49. https://doi.org/10.1057/palgrave.ivs.9500004

[10] Alan G. Gross and Joseph E. Harmon. 2009. The Structure of PowerPoint Presentations: The Art of Grasping Things Whole. *IEEE Transactions on Professional Communication* 52, 2 (2009), 121–137. https://doi.org/10.1109/TPC.2009.2020889

[11] Heiko Haller and Andreas Abecker. 2010. iMapping: A Zooming User Interface Approach for Personal and Semantic Knowledge Management. *SIGWEB Newsletter* Autumn, Article 4 (2010), 4:1–4:10 pages. https://doi.org/10.1145/1850770.1836295

[12] Robin H. Kay and Ann LeSage. 2009. Examining the Benefits and Challenges of Using Audience Response Systems: A Review of the Literature. *Computers & Education* 53, 3 (2009), 819–827. http://doi.org/10.1016/j.compedu.2009.05.001

[13] Ali Khalili, Sören Auer, Darya Tarasowa, and Ivan Ermilov. 2012. SlideWiki: Elicitation and Sharing of Corporate Knowledge Using Presentations. In *Proceedings of the International Conference on Knowledge Engineering and Knowledge Management (EKAW 2012)*. Springer, Galway, Ireland, 302–316. http://doi.org/10.1007/978-3-642-33876-2_27

[14] Clemens N. Klokmose, James R. Eagan, Siemen Baader, Wendy Mackay, and Michel Beaudouin-Lafon. 2015. Webstrates: Shareable Dynamic Media. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST 2015)*. Charlotte, USA, 280–290. https://doi.org/10.1145/2807442.2807446

[15] Andrea Kohlhase. 2007. Semantic PowerPoint: Content and Semantic Technology for Educational Added-Value Services in MS PowerPoint. In *Proceedings of the World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED-MEDIA 2007)*. AACE, Vancouver, Canada, 3576–3583.

[16] Aksha Kumar and Mukundan Sasikumar. 2007. PCML: A Pedagogy-oriented Content Markup Language. *Innovations in E-learning, Instruction Technology, Assessment, and Engineering Education* (2007). https://doi.org/10.1007/978-1-4020-6262-9_3

[17] Joel Lanir, Kellogg S. Booth, and Anthony Tang. 2008. MultiPresenter: A Presentation System for (Very) Large Display Surfaces. In *Proceedings of the 16th ACM International Conference on Multimedia (MM 2008)*. Vancouver, Canada, 519–528. https://doi.org/10.1145/1459359.1459428

[18] Leonhard Lichtschlag, Thorsten Karrer, and Jan Borchers. 2009. Fly: A Tool to Author Planar Presentations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2009)*. Boston, USA, 547–556. https://doi.org/10.1145/1518701.1518786

[19] Yefeng Liu, Darren Edge, and Koji Yatani. 2013. SidePoint: A Peripheral Knowledge Panel for Presentation Slide Authoring. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2013)*. ACM, Paris, France, 681–684. https://doi.org/10.1145/2470654.2470750

[20] Utiyama Masao and Hasida Kôiti. 1999. Automatic Slide Presentation from Semantically Annotated Documents. In *Proceedings of the Workshop on Coreference and Its Applications (CorefApp 1999)*. College Park, USA, 25–30. https://doi.org/10.3115/1608810.1608816

[21] Yelena Mejova, Klaar De Schepper, Lawrence Bergman, and Jie Lu. 2011. Reuse in the Wild: An Empirical and Ethnographic Study of Organizational Content Reuse. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2011)*. Vancouver, Canada, 2877–2886. https://doi.org/10.1145/1978942.1979370

[22] Tomer Moscovich, Karin Scholz, John F. Hughes, and D. Salesin. 2004. *Customizable Presentations.* Technical Report. Technical Report CS-04-16, CS Department, Brown University. http://cs.brown.edu/research/pubs/techreports/reports/CS-04-16.html

[23] Les Nelson, Satoshi Ichimura, Elin Rønby Pedersen, and Lia Adams. 1999. Palette: A Paper Interface for Giving Presentations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 1999)*. Pittsburgh, USA, 354–361. https://doi.org/10.1145/302979.303109

[24] Theodor Holm Nelson. 1995. The Heart of Connection: Hypermedia Unified by Transclusion. *Commun. ACM* 38, 8 (1995), 31–33. https://doi.org/10.1145/208344.208353

[25] Dave Raggett. 2006. Slidy: A Web Based Alternative to Microsoft PowerPoint. In *Proceedings of XTech (XTech 2006)*. Amsterdam, Netherlands.

[26] Elke I. Reuss, Beat Signer, and Moira C. Norrie. 2008. PowerPoint Multimedia Presentations in Computer Science Education: What Do Users Need?. In *Proceedings of the 4th Symposium on Usability & HCI for Education and Work (USAB 2008)*. Graz, Austria. https://doi.org/10.1007/978-3-540-89350-9_20

[27] Reinout Roels, Yves Baeten, and Beat Signer. 2016. An Interactive Data Visualisation Approach for Next Generation Presentation Tools. In *Proceedings of the 8th International Conference on Computer Supported Education (CSEDU 2016)*. Rome, Italy, 123–133. https://doi.org/10.5220/0005806401230133

[28] Reinout Roels, Yves Baeten, and Beat Signer. 2017. Interactive and Narrative Data Visualisation for Presentation-Based Knowledge Transfer. *Communications in Computer and Information Science* 739 (2017). https://doi.org/10.1007/978-3-319-63184-4_13

[29] Reinout Roels, Paul Meştereagă, and Beat Signer. 2015. An Interactive Source Code Visualisation Plug-in for the MindXpres Presentation Platform. *Communications in Computer and Information Science* 583 (2015). https://doi.org/10.1007/978-3-319-29585-5_10

[30] Reinout Roels, Paul Meştereagă, and Beat Signer. 2015. Towards Enhanced Presentation-based Teaching of Programming: An Interactive Source Code Visualisation Approach. In *Proceedings of the 7th International Conference on Computer Supported Education (CSEDU 2015)*. Lisbon, Portugal, 98–107. https://doi.org/10.5220/0005445300980107

[31] Reinout Roels and Beat Signer. 2014. MindXpres: An Extensible Content-driven Cross-media Presentation Platform. In *Proceedings of the 15th International Conference on Web Information System Engineering (WISE 2014)*. Thessaloniki, Greece. http://doi.org/10.1007/978-3-319-11746-1_16

[32] Reinout Roels, Christophe Vermeylen, and Beat Signer. 2014. A Unified Communication Platform for Enriching and Enhancing Presentations with Active Learning Components. In *Proceedings of the IEEE 14th International Conference on Advanced Learning Technologies (ICALT 2014)*. Athens, Greece, 131–135. https://doi.org/10.1109/ICALT.2014.46

[33] Andreas Schmidt. 2005. Bridging the Gap Between Knowledge Management and E-Learning with Context-Aware Corporate Learning. In *Proceedings of the Biennial Conference on Professional Knowledge Management/Wissensmanagement (WM 2005)*. Kaiserslautern, Germany, 203–213. https://doi.org/10.1007/11590019_23

[34] Beat Signer and Moira C. Norrie. 2007. As We May Link: A General Metamodel for Hypermedia Systems. In *Proceedings of the 26th International Conference on Conceptual Modeling (ER 2007)*. Auckland, New Zealand, 359–374. https://doi.org/10.1007/978-3-540-75563-0_25

[35] Beat Signer and Moira C. Norrie. 2007. PaperPoint: A Paper-based Presentation and Interactive Paper Prototyping Tool. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction (TEI 2007)*. Bonn, Germany, 57–64. https://doi.org/10.1145/1226969.1226981

[36] Ryan Spicer, Yu-Ru Lin, Aisling Kelliher, and Hari Sundaram. 2012. NextSlidePlease: Authoring and Delivering Agile Multimedia Presentations. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 8, 4, Article 53 (2012), 53:1–53:20 pages. https://doi.org/10.1145/2379790.2379795

[37] Miles Thorogood. 2016. slideDeck.js: A Platform for Generating Accessible and Interactive Web-Based Course Content. In *Proceedings of the 21st Western Canadian Conference on Computing Education (WCCCE 2016)*. Kamloops, Canada, 13:1–13:5. https://doi.org/10.1145/2910925.2910941

[38] Edward R. Tufte. 2006. *The Cognitive Style of PowerPoint: Pitching Out Corrupts Within, Second Edition.* Graphics Press, Cheshire, USA.

[39] Katrien Verbert and Erik Duval. 2008. ALOCOM: A Generic Content Model for Learning Objects. *International Journal on Digital Libraries* 9, 1 (2008), 41–63. https://doi.org/10.1007/s00799-008-0039-8

[40] Katrien Verbert, Xavier Ochoa, and Erik Duval. 2008. The ALOCOM Framework: Towards Scalable Content Reuse. *International Journal of Digital Information* 9, 1 (2008), 1–24.

[41] Working Group for Learning Object Metadata. 2002. IEEE 1484.12.1-2002 - IEEE Standard for Learning Object Metadata. https://standards.ieee.org/standard/1484_12_1-2002.html. Accessed: 2019-01-09.

[42] Noraniah Mohd Yassin. 2005. Learning Object Content Models: A Comparative Analysis. In *Proceedings of the 1st Postgraduate Annual Research Seminar (PARS 2005)*. FSKSM, UTM, Malaysia.

[43] Douglas E. Zongker and David H. Salesin. 2003. On Creating Animated Presentations. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)*. San Diego, USA, 298–308. https://doi.org/10.2312/SCA03/298-308

Received October 2018; revised December 2018; accepted February 2019