



ARETE – DELIVERABLE 3.8

“WP 3- Deliverable 3.8 Interactive Authoring Toolkit integration with MirageXR ”

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under Grant Agreement No 856533

Deliverable number:	D3.8
Due date:	February 2022
Nature ¹ :	OTHER
Dissemination Level:	PU
Work Package:	3
Lead Beneficiary:	TOU
Beneficiaries:	TOU, UCD, ULE, VIC

Document History

Version	Date	Description
0.1	July 21	Outline
0.2	August 12, 21	Collecting inputs

1

Nature:

R = Report, P = Prototype, D = Demonstrator, O = Other

Dissemination level

PU = Public

PP = Restricted to other programme participants (including the Commission Services)

RE = Restricted to a group specified by the consortium (including the Commission Services)

CO = Confidential, only for members of the consortium (including the Commission Services)

Restraint UE = Classified with the classification level "Restraint UE" according to Commission Decision 2001/844 and amendments

Confidential UE = Classified with the mention of the classification level "Confidential UE" according to Commission Decision 2001/844 and amendments

Secret UE = Classified with the mention of the classification level "Secret UE" according to Commission Decision 2001/844 and amendments



0.8	October 13, 21	Merging SLR done by Will Guest, Tjeerd Olde Scheper, Fridolin Wild
0.9	February 23, 22	Several iterations lead to the final deliverable: augmentation documentation by Abbas Jafari, Alex Shubin, Fridolin Wild; augmentation data model and class document by Rob Hillman;

Disclaimer

The contents of this document are the copyright of the ARETE consortium and shall not be copied in whole, in part, or otherwise reproduced (whether by photographic, reprographic or any other method), and the contents thereof shall not be divulged to any other person or organisation without prior written permission. Only members of the ARETE Consortium, entered the ARETE Consortium Agreement, dated 24.04.2019, and to the European Commission can use and disseminate this information.

Content provided and information within this report is the sole responsibility of the ARETE Consortium and does not necessarily represent the views expressed by the European Commission or its services. Whilst this information contained in the documents and webpages of the project is believed to be accurate, the authors and/or any other participant of the ARETE consortium makes no warranty of any kind with regard to this material.



Table of Contents

Executive Summary	4
Repositories	4
1. Introduction.....	5
2. State of the Art: Systematic Literature Review.....	7
2.1 Key Concepts	7
2.2 Related Work.....	12
2.3 Research Methodology	13
2.3.1 Identification	14
2.3.2 Screening	14
2.3.3 Eligibility	14
2.3.4 Categorisation	15
2.4 Results.....	16
2.4.1 Selection Results.....	16
2.4.2 Categorisation Summary	17
2.5 Discussion.....	18
2.5.1 Early work (2001 - 2005)	18
2.5.2 Mobile AR materialises (2009 - 2012)	20
2.5.3 The appearance of head-mounted displays (2013 - 2021).....	22
2.5.4 Summary of Findings	26
2.6 Conclusion	27
3. System Architecture.....	28
3.1 Component architecture of the MirageXR apps.....	29
3.2 Augmentations: How to create a new augmentation primitive	31
3.2.1 Existing Augmentation Types	31
3.2.2 Implementing a new Augmentation Type.....	31
3.2.3 Programming your new augmentation type	36
4. Implemented augmentations.....	40
4.1 Image	40
4.2 Video	40
4.3 Audio.....	41
4.4 Label.....	42
4.5 Action Glyphs	43
4.6 GhostTrack	43
4.7 Visual Effect.....	44
4.8 3D model	44
4.9 Character Model.....	45
4.10 Pick & Place	47



4.11 3D Whiteboard	49
4.12 Image Marker	50
4.13 Plugin	50
5. Configuring MirageXR	51
5.1 API Keys.....	51
5.2 Configure Corporate Design and Identity with the BrandManager	52
5.3 Assets Mapping Table	53
5.4 Default servers	53
5.5 Terms and conditions.....	53
6. Repository service: Moodle mod_arete.....	53
7. Logging user behaviour with the IEEE Experience API (xAPI)	54
8. Discussion.....	56
9. Conclusion	59
References.....	60
Annex A: Orkestra Library.....	69
Code refactoring.....	69
Open-sourcing Orkestra Code	70
Proofs of concepts.....	71
Geography Demo	72
Music Demo	72
VR Demo.....	72



Executive Summary

This deliverable reports on the interactive authoring toolkit integration with MirageXR, reporting on key technical work of the ARETE project, delivered in WP3, “interactive AR toolkit”.

Work package 3 has four overarching objectives to design and develop a) interactive 3D learning content, b) AR apps, c) digital repository services (and related data models), and d) standards (esp. working towards IEEE P1589 ARLEM 2.0).

Within this deliverable, we are reporting on our achievements regarding the content authoring tool and player app, MirageXR. MirageXR is released as Open Source under an MIT licence.

Repositories

Name	Description	Link	Licence	Release
<i>MirageXR</i>	Authoring ToolKit and Player for AR learning experiences	https://platform.xr4all.eu/wekit-ecs/mirage-xr/	MIT	Public
<i>Moodle mod_arete</i>	Moodle module and repository interface for P1589-2020 compliant AR learning experience models	https://github.com/ARETEedu/moodle-mod_arete	GPLv3	Public
<i>ARLEM.js</i>	Validator component for testing IEEE P1589-2020 compliance	https://github.com/openARLEM/arlem.js	MIT	Public
<i>Moodle block_repository</i>	Widget to display latest AR learning experiences available in Moodle	https://github.com/ARETEedu/moodle-ARLEM_block_repository		
<i>ARETE market</i>	WordPress based market place to interface with Moodle repositories	https://github.com/ARETEedu/arete-market	GPLv2	Public
<i>orquestra</i>	A library that provides the mechanisms for communication, synchronisation and adaptation in multi-user and multi-device applications.	https://github.com/tv-vicomtech/orkestralib-unity (Access upon request)	GPLv3	




1. Introduction

MirageXR was first released as Open Source (MIT License) in October 2020². Since then, we have managed to gather 33 members into the developer community, which is now well organised and also links with project-external individuals and other collaborating organisations. The key stats of the developer community put MirageXR on the map in the top 10k projects, comparing well against the average unique monthly contributors, which can be assessed as between 8 and 47 unique monthly contributors (middle: 20)³.

The coder community is well managed with monthly sprints, sometimes in parallel, with the community members taking various roles from code developer, scrum master, code reviewer, designer, tester, user experience tester, designer, documenter, devops engineer, etc. Continuous integration and testing uses automated pipelines for each merge request to run a battery of unit tests, report the results automatically, check build status, and compile for the four platforms targeted (iOS, Android, Hololens-1 and Hololens-2).

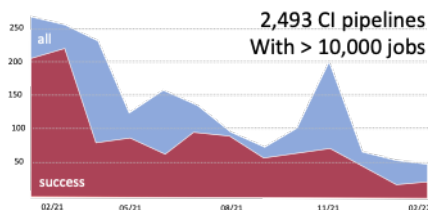
In the period working up to this deliverable, we resolved 612 issues in 20 sprints, creating 453 merge requests in 1,548 commits (an average of 2.3/day). The four apps for the four platforms were built from 8 releases, with TestFlight for beta testing set up for iOS, and sideloading available for Android and Hololens. The mobile apps were taken to the app stores in a first public release at the time of Online Educa Berlin in December 2021.

612 issues closed
in 20 sprints with
453 merge requests

 **4 apps**
8 releases
Open Source since 10/2020 **4 repository projects**
6 Moodle releases

Developer Community

35 PlayMode
63 EditMode
= 98 unit tests




33 members



² MirageXR continues in ARETE from the legacy of the WEKIT and XR4ALL projects, working along-side a The Open University (TOU) internally-funded project ‘OpenReal’.

³ <https://www.bvp.com/atlas/measuring-the-engagement-of-an-open-source-software-community>



Furthermore, the server web applications and APIs now include four repository projects on GitHub, of which the Moodle mod_arete has been released in six versions, currently in review (following revision) for the official Moodle plugin directory. The repository plugin for Moodle is at the time of writing installed on four servers worldwide (to our knowledge).

The 98 unit tests help to automatically verify code health with every merge request. The tests have been executed as part of 2,493 pipelines running over 10,000 jobs

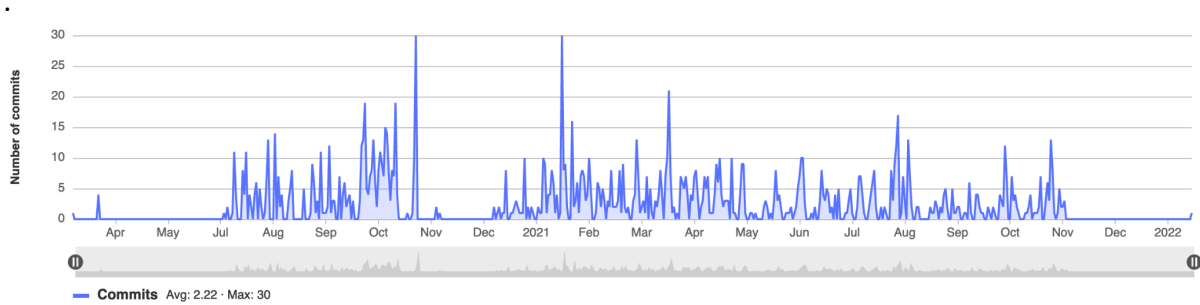


Figure 1.1: Commits over time.

The overall Unity code project has around 1.2M lines of code⁴. Of this, there are more than 220k lines of code in the C# classes. The core of MirageXR consists of 773,760 lines of code, of which 141,638 are in C# classes⁵. Since the dawn of the project, 14,354 files have been changed, adding up 2,501,057 insertions(+).

Language	files	blank	comment	code	
Unity-Prefab		856	22	0	584123
HTML	1129	3250	6504	180547	
C#	1557	28084	53751	146027	
XML	33	355	109	141637	
HLSL	236	3973	4303	18052	
JSON	15	59	0	15140	
JavaScript	374	207	274	8964	
Markdown		33	1006	0	3326
CSS	6	653	167	2903	
SVG	46	1	32	580	
Objective-C++		1	100	48	460
Bourne Shell		15	73	6	437
YAML	4	22	45	218	
DOS Batch		2	12	3	21
SUM:	4307	37817	65242	1102435	

Table 1.1: Counted Lines of Code (clov v1.92).

⁴ Code in languages recognised by cloc v1.92. The project has a total line count of 8M lines (wc -l).

⁵ Again measured with cloc v 1.92.



2. State of the Art: Systematic Literature Review

When blending digital content with our physical environment, we are not merely aligning visual phenomena, we are adding additional meaning to a meaningful place. Our everyday environment abounds with cues and clues, messages that prompt us to act or warn us of hazards. For each situation, we refer to others, infer unseen information and interact to continuously deepen our understanding of what is possible and what *might* be possible. Augmented Reality (AR) devices, especially those capable of seamless, high-fidelity integration of the digital and the physical, can not only provide access to digital content, but also alter the way we experience it. We are changing the content, the context and, to some degree, the mode of communication. From a constructivist point of view, we now have access to the fundamental fabric of the learning experience.

AR applications, with the possibility to attach informative or stylistic virtual content to any detectable physical surface, have readily found a role in education and training. Hundreds of papers have been published showing how this can be done effectively and many show that it can enhance the quality of the learning experience.

With such powerful and instructive AR tools at our disposal, it is the task of this paper to investigate how these systems have been put to use to leverage their learning potential. To facilitate this, we employ Bloom's Revised Taxonomy (referred to hereafter as 'the RT'), to guide our assessment of the various implementations of this technology.

The rest of this Section 1 is structured as follows: Subsection 1 provides background and definitions for key terms and introduces Bloom's Taxonomy. Subsection 2 looks at related work, namely other reviews conducted around the theme of AR. Subsection 3 details the methodology used, describing the steps taken in aggregating, screening and analysing the collected works. Subsection 4 presents the findings from the categorisation process and is followed, in Subsection 5, with a discussion of these findings. Subsection 7 concludes the review.

2.1 Key Concepts

Before going further, it may be helpful to outline certain concepts, as the idea of extracting learning affordances from AR environments is non-trivial relying, as it does, on a understanding of how the terms 'learning affordance', 'AR' and 'authoring' have developed and how they relate to one-another.

Learning Affordances

The first key concept in need of explanation is that of an affordance, more specifically a learning affordance since the word means an opportunity for action, which is a rather vague



notion. A sign, either to prompt action or prevent it, contains a clear affordance (assuming it is understood), just as objects present physical affordances that give us cues on how to interact with them (handles, switches etc.). These opportunities for action, first coined by Gibson (1979) to describe ecological developments, have since been adapted more closely to learning (Kirschner, 2002) and, even more specifically, to those that relate to information and communication technologies (Bower, 2008).

Drawing on similar work is a paper from Dalgarno and Lee (2009), which assessed the learning affordances of Virtual Learning Environments (VLEs), which they consider to require the features of three-dimensionality, smooth temporal changes and interactivity. Their work, although more directed to Virtual Reality, is helpful in conceptualising and studies the presentation of affordances.

Augmented Reality

Augmented Reality (AR), also known as Mixed Reality (MR), describes an experience that blends real and virtual objects in a persuasive way (Milgram et al., 1995). Since we are combining digital, software-driven, behaviour with real, physical environments, the results can be surprising, unintuitive or unrealistic, making an exact definition hard to achieve. We may, however, identify key features of this technology which definitively point to the presence of “augmented” or “mixed” reality. We here identify three features that are used to distinguish AR or MR and, without explicitly defining the terms, provide criteria for assessing whether existing research can be said to use the relevant technology. These are based on Azuma’s notion of AR (from 1997) and also follow Dalgarno and Lee’s work (2009).

First, virtual objects have a geophysical, though not necessarily fixed, position or relevance. This “world-locked” characteristic is a key element in generating a sense of realism and, consequently, immersion. Devices or systems that use only internal frames of reference can, at best, provide only fully digitised environments, or Virtual Reality (VR).

The second feature is interactivity; virtual objects present a designed affordance, meaning that they should benefit or hinder us in the completion of some activity. Through their affordances, the objects are interacting with us, enriching our experience. Implementations that do not possess affordances are likely to do little to change our sense of our reality. Consider an experience in which almost featureless objects (such as cubes or spheres) are placed randomly in space. The result may change what we see, but little opportunity for action can be derived from it. In this case, our practical “reality” is little changed and hardly “augmented” in any meaningful way.

Third and finally, the digital content must have persistence. In practical terms this means that the digital content will maintain its affordance even if it does not have our attention, such as a virtual signpost or visual trace of past action. Without this ability to encode knowledge, our “reality” is unaffected by our interaction with the digital aspect, rather than augmented by it. In computer science, persistence is the characteristic of a state that means



it outlives the process that created it. If we consider a world-locked, interactive object whose state is changed through an interaction, we can say that the new state must persist after the interaction has finished. This feature is unique in that it places a requirement on the digital environment – the framework within which the digital objects are made – namely that it must have the capability to maintain and update a record of the states of all digital objects. This puts a minimum level of complexity on any implementation that will scale with the amount and complexity of the content it contains.

The features above do not specify which kind of device or technology allows them to be realised. There are currently two types of technology that allow all three features: transparent waveguide-based head-mounted displays (HMDs) and video pass-through systems on hand-held devices (HHDs). The research included in this work will necessarily use one of these technologies.

AR Authoring

Given the above context, an “authoring system” or “authoring tool” is a piece of software that is used to craft AR experiences; adding, arranging, and configuring digital content so that it offers a coherent experience to a consumer of the designed experience. This multimodal content must be delivered in such a way that it is supporting an educational process, rather than provide a particular piece or set of information. To be included in this data set, the system described should:

- 1) include some form of content selection by the user. An experience in which existing objects are only modified (in colour, size, position etc.), or where all connections to virtual content are predetermined, are not authoring tools more than AR applications that feature some user interaction.
- 2) expose lower-level functionality. There should be some consideration of this reduction in complexity, so that non-specialists can create learning sequences. Specifically, the systems should be usable by non-programmers (i.e. the people that constructed them).
- 3) provide some distinction between the mode in which the objects are assembled or structured and the one in which they are then used (i.e. an expert/trainer mode and a viewer/trainee/learner mode).

Bloom’s Revised Taxonomy

Bloom’s Revised Taxonomy [Anderson et al., 2001] is a pedagogical tool that describes learning outcomes in a pragmatic, “hands-on” style, emphasising the importance of semantic knowledge, practice and reflection. The revised taxonomy (RT) addresses aspects of learning, teaching and assessment, categorising learning activities across two dimensions, the Knowledge Dimension and the Cognitive Process Dimension.



With regards to the learning process, there is a hierarchical structure where actions are spread across 6 levels. These levels are generally sequential in terms of expertise-building, but do not necessarily require preceding actions to be recognised in their own right. This will become evident later, when the taxonomy is applied.

Knowledge Dimension (excluding meta-cognitive knowledge)

Here we are concerned with the type of knowledge that the AR authoring tool provides, or those elements in the system that have the user’s attention. Four knowledge types are identified: factual, conceptual, procedural and metacognitive. Each of these is described in Table 2.1, which is a summary of that found in the RT.

Knowledge Dimension Summary		
Major Type	Description	Subtypes ⁶
Factual Knowledge	The basic elements students must know to be acquainted with a discipline or solve problems in it.	<ul style="list-style-type: none"> - Ko. terminology - Ko. specific details and elements
Conceptual Knowledge	The interrelationships among the basic elements within a larger structure that enable them to function together.	<ul style="list-style-type: none"> - Ko. classifications and categories - Ko. principles and generalisations - Ko. theories, models, and structures
Procedural Knowledge	How to do something, methods of inquiry, and criteria for using skills, algorithms, techniques, and methods.	<ul style="list-style-type: none"> - Ko. subject-specific skills algorithms - Ko. subject-specific techniques and methods - Ko. criteria for determining when to use appropriate procedures
Metacognitive Knowledge	Knowledge of cognition in general as well as awareness and knowledge of one’s own cognition.	<ul style="list-style-type: none"> - Strategic knowledge - Knowledge about cognitive tasks, including appropriate contextual and conditional knowledge - Self-knowledge

Table 2.1: The Knowledge Dimension

The exclusion of metacognitive knowledge

In this work, metacognitive knowledge will not be studied, for two reasons. First, because the field of Augmented Reality is still young, even with respect to Information Technology. As was the case in the earlier years of desktop computing, the most optimal arrangement of sensors, user interfaces and input methods is still being worked out and many of the studies here describe previously untouched areas of work. If we are not yet familiar with the tools we are using, there is unlikely to be much that can be said about the metacognitive impact on us. Second, and in a similar vein, the same nascency means that the complexity of development pipelines is high, limiting the functionality of the software. Often a variety of

⁶ Ko. = “Knowledge of”



development environments, toolkits and custom-written applications are needed to produce results. The consequence of this is that AR Authoring applications that can provide enough fidelity and complexity that we might use them for metacognitive insight are sparse and require a readiness level within the technology that is not yet present.

Cognitive Process Dimension

In this dimension we are looking at which learning activities are supported by the authoring implementation. Those that are specifically designed to be used or those that are shown in the course of using the system (even if not intended) should be captured.

Cognitive Process Dimension Summary		
Major Type	Description	Subtypes and Synonyms
Remember	Retrieve relevant knowledge from long-term memory.	Recognising, identifying Recalling, retrieving
Understand	Construct meaning from instructional knowledge, including oral, written, and graphic communication.	Interpreting, clarifying, representing Exemplifying, illustrating Classifying, categorising, subsuming Summarising, abstracting, generalising Inferring, concluding, predicting Comparing, contrasting, matching Explaining, construction models
Apply	Carry out or use a procedure in a given situation.	Executing, carrying out Implementing, using
Analyse	Break material into constituent parts and determine how the parts relate to one another and to an overall structure or purpose.	Differentiating, discriminating... Organising, integrating, structuring... Attributing, deconstructing
Evaluate	Make judgement based on criteria or standards.	Checking, detecting, monitoring Critiquing, judging, testing
Create	Put elements together to form a coherent or functional whole, reorganise elements into a new pattern or structure.	Generating, hypothesising Planning, designing Producing, constructing

Table 2.2: The Cognitive Process Dimension

Using the taxonomy to assess AR Learning affordances

Based on revised taxonomy and a deeper appreciation of how the dimensions relate to AR Authoring, we can draw up a table of learning affordances that we might expect to see. The following table provides a number of descriptions and examples that illustrate how certain features of an AR authoring system or toolkit can provide learning affordances across the various knowledge types and cognitive processes.



AR Authoring Learning Affordance Matrix		The Cognitive Process Dimension					
		1. Remember	2. Understand	3. Apply	4. Analyse	5. Evaluate	6. Create
The Knowledge Dimension	A. Factual Knowledge	System supports the recall or naming of details or visual elements. (E.g. matching markers to specific virtual objects.)	The system supports the aggregation, comparison of factual elements.	The authoring tool supports the use of terminology, or encourages the user to make use of an augmentation.	The system aids the user in analysing real or virtual information, perhaps exposing the underlying structure of it.	The authoring tool supports or facilitates the evaluation or critiquing of facts.	The tool supports the generation of hypotheses or the production of new factual data.
	B. Conceptual Knowledge	System aids recognition or grouping of principles. Identify a known model, such as a set of map directions.	The software supports the abstraction, re-imagining or grouping of principles or ideas.	The software supports the user in applying a principle or model in a new situation.	The software supports the integration of disparate concepts or helps to structure	The tool provides sufficient insight on a concept or model so that its validity or consistency can be verified.	The authoring tool helps in planning or devising a model or theory that explains observed phenomena.
	C. Procedural Knowledge	The AR system prompts the use of a technique, skill or method (competence).	The system contextualises a skill or technique or compares practice.	Knowledge of the sequence or structure of the procedure is enhanced using augmentations.	The system supports the user in (re)organising the procedure according to their knowledge.	The software allows the user to compare procedure knowledge and supports decision making.	The software supports the design of complete, coherent models or their reorganisation into n
	D. Meta-cognitive Knowledge	Not included due to technology nascency and limited scope.					

Table 2.3: AR Authoring Learning Affordance Matrix

2.2 Related Work

Over the past few years, there has been enormous growth in the number of example industrial implementations for AR technology. There have been systematic reviews conducted of AR in industry (de Souza et al., 2019) and education (Garzon et al. 2019) and others that are industry-specific (e.g. automotive, aeronautic) or function-specific (e.g. maintenance, manufacturing).

Egger and Masood (2019) published an extensive review of Industrial AR, based around the technology, organisation, environment framework. Palmarini et al. (2019) focused on maintenance applications, extracting 30 studies published over two decades, to 2017, noting a preponderance of marker-based studies. Bottani and Vignali’s review, also in 2019, showed a marked increase in application and technical papers, beginning in 2013.

In education, Ibañez and Kloos (2018) performed a systematic review of AR in STEM settings, providing insight for instructional design processes. Parmaxi and Demetriou (2020) focused on AR language learning, highlighting trends in both activity classification - nearly



two thirds of papers supported skills training - and the specific skills - around a quarter tackled vocabulary.

Industry-specific reviews have also been conducted by Boboc et al. (2020) for the automotive industry, Fraga-Lamas et al. (2018), looking at the shipyard of the future, Safi et al. (2019) for aerospace and Cheng et al. (2020), who looked at the areas of architecture, engineering, construction, and operation (AECO).

The authors did not find any other reviews addressing authoring functionality or those that extract learning affordances from technical implementations of AR systems.

2.3 Research Methodology

In conducting this literature review, we followed the PRISMA⁷ structure for the construction of the dataset. This consists of a sequence of phases, shown in Figure 1.

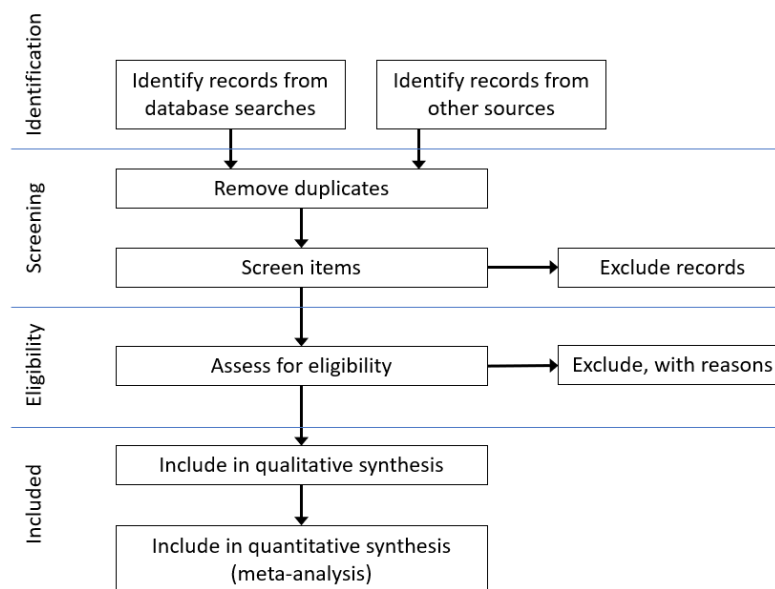


Figure 2.1: PRISMA systematic review phases, redrawn from Moher et al., 2009.

We first selected a set of search terms – those commonly used or precise descriptions of the process of creating AR experiences that are dynamically structured with selected content by the user, and for the purpose of training or education. These terms informed the collection of papers, journals, and articles from a wide range of sources. The resulting database was then cleaned and reduced to those relevant to the topic. The final set was then assessed and categorised along the lines of Bloom’s revised taxonomy, providing a picture of how the authoring toolkits support learning through the provision of affordances. This was done to provide answers to research questions shown in Table 2.4.

⁷ Preferred Reporting Items for Systematic reviews and Meta-Analyses (PRISMA)



RQ1: What is the current state of the research on AR Authoring tools?
RQ2: To what degree are the Bloom’s learning layers represented in existing AR Authoring tools?
RQ3: How do these tools convert technological affordances into learning affordances?

Table 2.4: Research questions.

2.3.1 Identification

The terms “augmented reality” and “mixed reality” are used interchangeably in the literature and, for the purposes of the search, were used interchangeably. In addition to this, the terms “authoring”, “composing” and “creating” were selected to capture research focused on the activity described above. Two other terms - “editor” and “content management” – were also included to capture those papers that were grammatically focused on a noun describing the system, rather than the function it performs. The final list of search terms is shown in table 1, with rows representing alternative (“OR”) terms.

“Augmented Reality”	AND	“Authoring”
OR		“Composing”
		“Creating”
“Mixed Reality”		“Editor”
		“Content Management”

Table 2.5: Search terms

2.3.2 Screening

The screening process removed results that were incomplete or incorrectly formed, those that were not accessible in English and those that could not be found or accessed through online portals. Duplicate entries were also identified and excluded, and patents were separated from the list.

Publications were also excluded if they could not be shown to be peer-reviewed, if they were themselves a review paper and not contribution additional implementations to the field, or if the publication was found to be a poster or workshop as these were thought to present to sparse a level of detail to sufficiently categorise them according to learning outcomes.

2.3.3 Eligibility

During this phase, a deeper level of assessment was made into the subject matter of each paper. First the abstract was read and it was determined if the paper met the criteria for



demonstrating both AR and, subsequently, AR Authoring. If there was ambiguity or these topics were not mentioned, the full text was read to determine its relevance.

Determining whether the research item was on topic involved an assessment of the relevance of the work – they should demonstrate the implementation and/or design of an AR authoring toolkit. Once the general theme of the paper was found to be relevant, the definitions (given above) of both AR and AR Authoring were taken into consideration. If the paper was found not to meet either one, the entry was marked as “not true AR” or “not authoring”, respectively. These categories can also show the amount of research that is excluded simply as a result of drawing up these definitions.

2.3.4 Categorisation

The assessment of the research and the assignment of categories to it is based on an understanding of the principles of both AR Authoring and the RT. Despite the clarity of the RT, the categorisation process relied on some assumptions being made about the learning affordances expressed, for the following reasons:

- The papers did not explicitly use the RT to design their learning outcomes, so there is no explicit intent to reveal or otherwise support a particular knowledge type, cognitive process.
- More broadly, there were rarely cases in which the learning outcomes were explicitly stated.
- Often the details given around the experience of the user are limited so the reader, or categoriser, must infer which learning outcome is being met.
- Though an authoring tool may have designed affordances, in implementation several reported functional limitations or errors in the system, restricting what the user could do and, as a consequence, impact their potential for learning.
- The functions that provide evidence for specific results will not overlap perfectly with those that support learning, so while a paper may focus on a feature that provides academic novelty, for instance, the learning outcomes might be unrelated, leading to little space being given for their description or explanation.

To account for these discrepancies, for each paper that was categorised, there was also recorded one or more quotes that provide evidence of the choice or highlight a key aspect of the authoring tool. These can be found in the appendix, alongside the categorisation values attributed to each paper.

It is hoped that these examples, together with the discussion below, the method of interpretation of the RT will become clear. Rather than provide an objective standard for what constitutes a (reported) learning affordance, we aim to present a lens to view AR authoring systems that others can use and which can contribute to the discussion on how to make better experiences with this technology.



2.4 Results

The searches conducted in the identification phase yielded 556 papers and 71 patents. The initial screening phase saw this number reduced to 524 papers by removing duplicates and incomplete or malformed entries. A second round of screening was then carried out to identify those papers that were clearly off-topic, not locatable online or not available in English.

2.4.1 Selection Results

After the eligibility phase, in which the research was assessed in detail for its relevance to the topic, 128 papers remained, which were downloaded and collated, in preparation for categorisation.

During this categorisation process the specific details of the papers were scrutinised in terms of learning affordances. This led to additional exclusions, mostly due to the work being marked as “not AR Authoring”, bringing the dataset to its final size.

Reason - Source	# excl.	Reason - Type	# excl.	Reason - Content	# excl.
Not found / no access	22	Poster or workshop	4	Not “AR”	28
Not available in English	30	Review paper	17	Not “Authoring”	43
		Not peer reviewed	10	Off-topic	242
Total excluded	396				

Table 2.6: Breakdown of reasons for exclusion from final dataset.

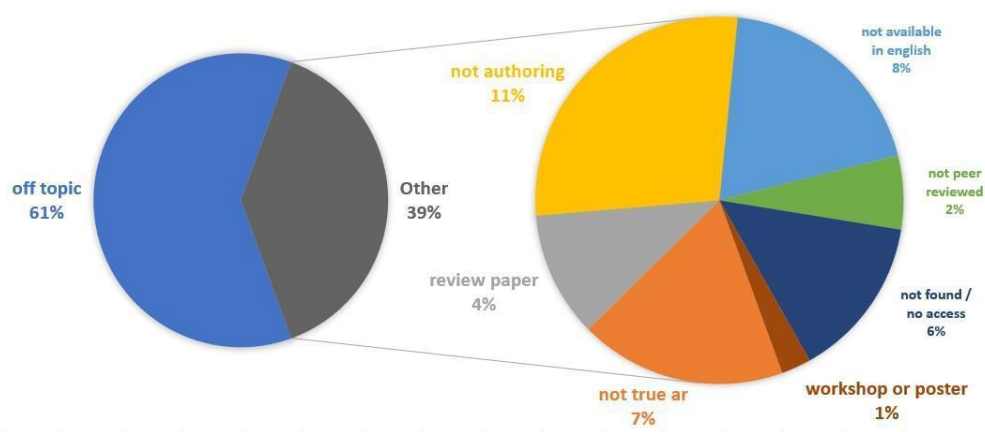


Figure 2.2: Reasons for exclusion from final dataset

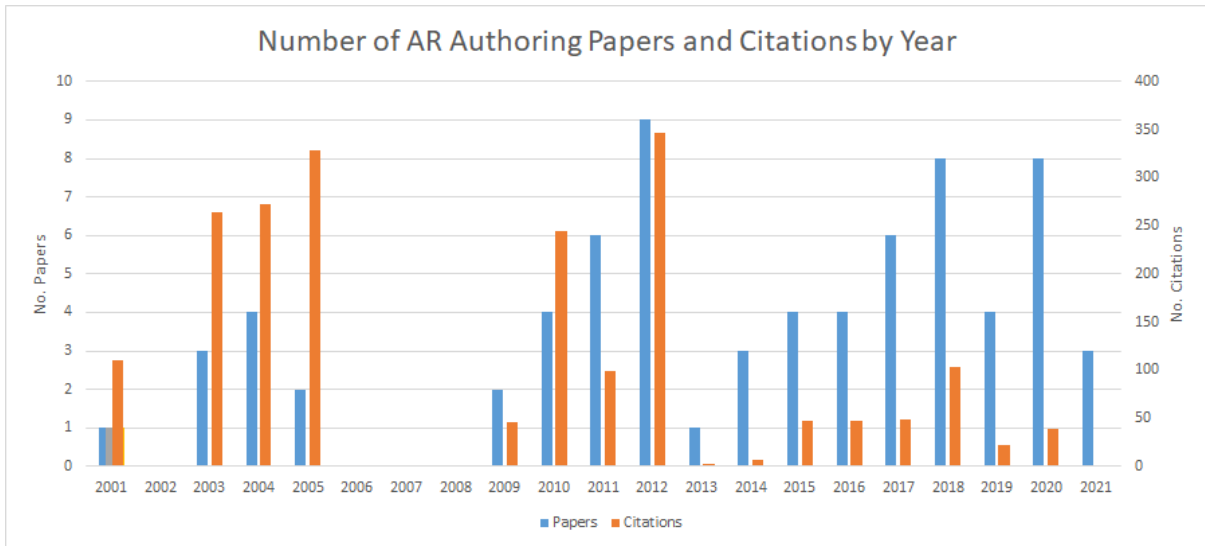


Figure 2.3: Final dataset - publication and citation chronology

The final list of papers consists of 72 peer-reviewed papers, all with demonstrations of AR authoring tools. Each meeting the criteria mentioned above and each demonstrating some learning affordances.

2.4.2 Categorisation Summary

A summary of how the categories identified in the dataset are distributed over time can be in Figures 2.4 and 2.5, showing the Knowledge Dimension and the Cognitive Process Dimension, respectively.

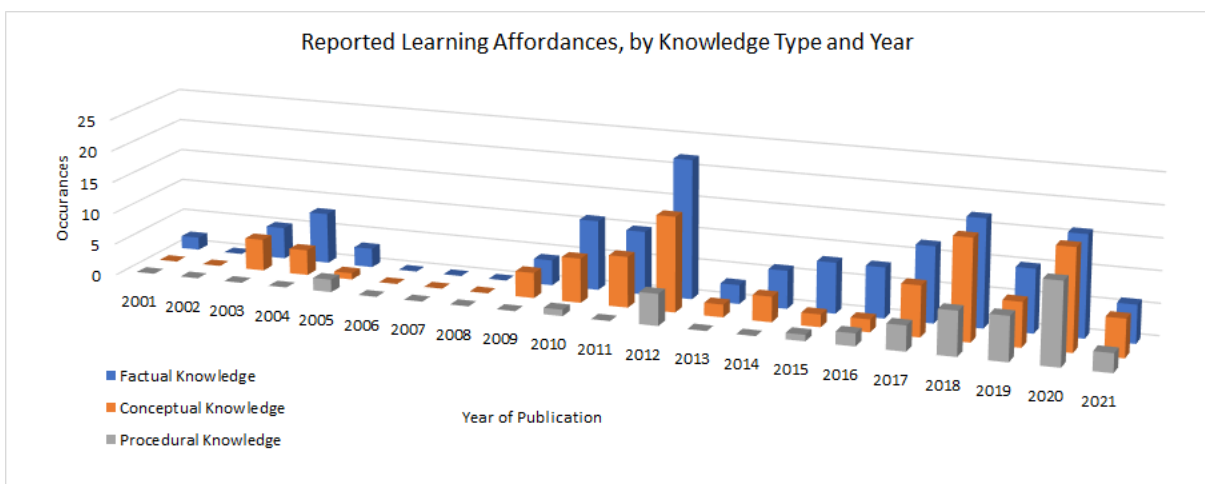


Figure 2.4: Prevalence of reported learning affordances, grouped by Knowledge Type.

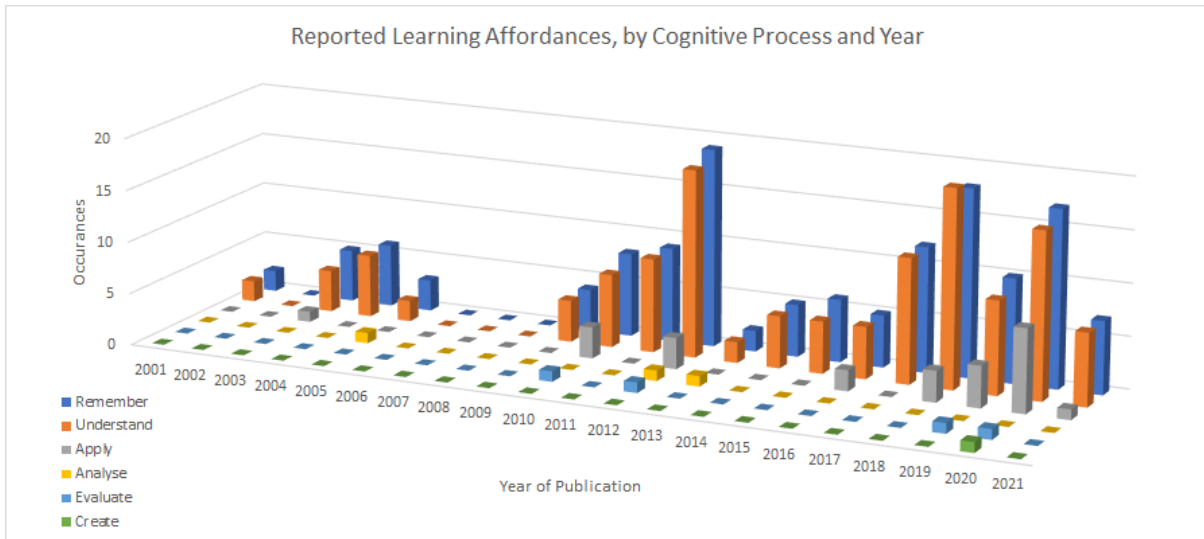


Figure 2.5: Prevalence of reported learning affordances, grouped by Cognitive Process.

2.5 Discussion

The chronology of the papers can be seen as having three phases, corresponding to improving levels of technology readiness and computational performance.

Early work, followed by a 3-year hiatus, was then picked up by a larger community of researchers, better equipped with smartphones and tablets. A third wave was brought on by new depth mapping equipment and their ultimate integration into head-mounted devices, adding significant benefit to the now hands-free user, but also introducing complexities to interface and interaction design.

The discussion that follows will trace the development of key systems that allowed these authoring tools to exist, focusing on the learning affordances that were designed by the developers and researchers involved. To clarify, we are interested in those functions that provide affordances to the learner, rather than the teacher or designer of the experience.

2.5.1 Early work (2001 - 2005)

Many of the authoring solutions use similar underlying technologies, such as ARToolkit, which was developed by Hirokazu Kato, Mark Billinghurst and Ivan Poupyrev, and first released in 1999. This was one of the first tracking libraries that would allow developers to connect markers to augmentations and is frequently cited in the research found in this dataset. The first open-source version was released through HIT labs in Washington in 2001.

The first example of an AR authoring system, Tiles, comes from the same group of researchers (Poupyrev et al. 2001). "The Tiles system is a prototype tangible augmented reality authoring interface that allows a user to quickly layout virtual objects in a shared



workspace and easily manipulate them without need of special purpose input devices” (p.7). The virtual content was arranged using square printed markers (tiles), which were each linked to a 3D visualisation.

The assignment of content to object can support recall (A1) and the use of functional tiles (such as the ‘delete’ option) requires understanding of the object (A2). Tiles could also be used in conjunction with others, producing results such as ‘copy-and-paste’, or provide application-level functionality. These types support learning about the interrelations of the objects (B1) and provide a deeper understanding of the content at hand (B2).

Most other early examples come from the Authoring Mixed Reality (AMIRE) framework. Abawi et al. (2004) highlight four tasks that the author can perform: “qualification of MR components, adaptation of the selected MR components, combination of the MR components to allow interaction between them and calibration of real objects and virtual objects in the application” (p.115). This illustrates the support of remembering factual knowledge (A1), its adaptation (A2) and the combination of components using a given model (B1) in order “to connect two parts of a story” (p.116), showing a degree of understanding of the theme (B2).

Zauner and Haller (2004), working on a related project, illustrate four modes related to MR placement: “The first one is the observation mode. This mode allows the author to take a look at the actual placement. Further, the placement-marker is visualized by a white... In the second mode, the author is able to modify the size of the object... The third mode is used to move the object...The rotation mode, which is the fourth and the last mode, enables the author to rotate the object.” (p.6).

In this description of system use we find evidence of the recall and understanding of factual knowledge (A1, A2) as the user can implement affine transformations (translation, rotation and scaling) but, without the application to a conceptual area or the integration with procedural knowledge, no further categorisations are assigned. This principle is adhered to throughout the categorisation of the sources, as there is often a demonstration of the functions supported by the user interface, without an application to a particular model, principle or concept (conceptual knowledge).

In 2005, Haller et al. also described the functionality within the AMIRE project, adapting the previous tools to support a simple construction task, where “[a]ugmented, highlighted boards help users to find the right boards and to mount them in the right order” (p.5). Given that the toolkit was deployed on a hand-held device (which must be put down to handle the parts), this provides clear evidence in support of procedural knowledge retention (C1). In addition, “users get information where to place the different components and how to connect them” (p.5), exemplifying their procedural knowledge (C2).

Another notable example of pioneering work in the AR Authoring field comes from Kirkley and Kirkley (2005). The most cited of all the papers in this dataset, this research presents an



authoring tool called Information In Place Inc. (IIPI) CREATE™. The system, originally prototyped for military use, discusses many of the constructivist learning principles reflected in the RT, but provides little evidence of the way in which this tool demonstrates learning affordances in AR. The authors state that the tool “does not intend to replace the existing tools that various designers/developers use but to provide an organizing, shared framework for various types of individuals as they create these next generation learning environments” (Kirkley & Kirkley, 2005; p.50). Based on this, the system is shown to support the recall and summarisation of other factual information (A1, A2) and can help to organise the principles and models found in other design environments (B4).

2.5.2 Mobile AR materialises (2009 - 2012)

While early studies showed promise, a lack of technology readiness limited their utility in the wider market. This began to change with the release of the first iPhone in early 2007 and the first Android phone in November 2008. January 2010 saw the first iPad and, later that same year, both Samsung and Research in Motion (RIM) launched devices with the same form factor.⁸

These devices, although the first in their class, paved the way for more interactive, powerful applications. Given the computational requirements of mobile “pass-through” AR, it was at this point that the tools found a wider audience and application. ARToolkit also evolved, with a Flash version being released in 2008, entitled FLARToolkit.

This toolkit was used extensively by creators of AR authoring systems during this period. The lighter implementations, such as those by Sano (2011, 2012) and Jee et al. (2011) used only the aforementioned affine transformations (A1, A2), with others, such as Wang et al. (2010) extending this so that “a user can type in the question and choose the correct answer to the question in the question input section” (p.286), demonstrating the evaluation of the facts presented to them (A5). Such lightweight demonstrations were, by this point, generally being overtaken by more feature-rich systems, such as that demonstrated by Mader and Urban (2010), which included hand tools and visual instructions for disassembly or repair operations. Instructions such as “remove the screws that hold the hard drive using a Philips screw driver size PH2” (p.12) illustrate knowledge of terminology (A1), factual understanding (A2) and knowledge of subject-specific techniques (C1). Furthermore, they incorporated the use of schematics (B1) to guide the user through actions (B2).

This use of AR authoring to more closely follow a procedure is also reflected in work by Fei et al. (2012), who used an action graph - “an augmented version of the basic finite-state machine” (p.12) to allow the design of complex procedures. “[A]ctions [...] in the action graph, that represent interactive objects or passive animations of the scenario and can be bound to the virtual objects, transitions (namely edges) represent changes of the storyline

⁸ (<https://www.theguardian.com/technology/2012/jan/24/smartphones-timeline>, accessed 09-09-21)



and can be bound to certain events” (p.22). The use of familiar objects (A1) and storyline transitions (B1) and their application in a procedural flow (C3) has the ability to approximate real-world complexity and captures the idea of contingency based on both object selection (A2) and the prediction of consequences (B2).

In this time period, three papers stand out for their relevance to later work (evidenced by the number of citations received). These are the AR Gamebuilder implementation (Klopfer & Sheldon, 2010), Dunser et al.’s Augmented Books (2012) and Langlotz et al.’s system for in situ content creation (2012). To date these have been cited almost 500 times, nearly a third of all citations represented by the dataset.

Klopfer and Sheldon (2010) present authoring software that “includes the ability to bring in new maps, position characters and items, make logical connections between these items, incorporate data, and insert media” (p.89). Encouraging the user to recall locations (A1), their relations to the virtual content assigned there (B1) promote recall. In later versions of the software, branching logic was also implemented, which could “offer a richer opportunity for students to understand issues in their communities and express their understanding of the relationships that surround them” (p.92). The user is thus encouraged to develop an understanding of both factual (A2) and conceptual knowledge (B2).

Dünser et al. (2012) produced an authoring system which was used to create interactive physics education, augmenting printed books. In a similar fashion to other marker-based tools “[b]asic interaction is provided by manipulating the book. Rotating or tilting the book pages shows virtual content from different positions and angles, and flipping book pages changes the virtual content displayed” (p.107) (A1, A2). However, the designers also added interactive features: “a user could select the desired image and start an animation by placing a finger over the black border ... In addition to this, some AR markers had interactions that allowed the participants to change the direction of particular components of the model, for example reversing the direction of the current in a wire” (p.111). The selection of markers, either for clarification or exemplification, shows understanding (B2) and the possibility to apply physical principles to an unfamiliar task is categorical evidence of the application of the user’s knowledge (A3, B3).

The interaction methods in work by Langlotz et al. (2012) were instead performed in the traditional way, using a touchscreen interface, introducing challenges around how to position the device and work on the content simultaneously, a hurdle that many subsequent studies would face. They solved this with a freeze mode, during which content would be added. Two types of features are mentioned - First that an “object can be translated, scaled and rotated by selecting the axis of transformation in the GUI” (p.628) (A1, A2). Second, in order to “enable the creation of more realistic models, we provide the functionality to assign different colors to the objects as well as to texture the object” (p.628), which demonstrates the recognition of the principles of colouring and texturing (B1).



The final grouping of research worthy of note in this time period is that arising from the workshop sessions run during the International Symposium on Mixed and Augmented Reality (ISMAR) during 2011 and 2012, which together added five research items to the final dataset.

Grubert et al. (2011) augmented posters and, while no virtual object transformation or interaction was mentioned, “the user selects regions for triggering the appearance of content as well as regions where to actually display the content” (p.2), mapping the content on the page to their design principles (B1, B2). Janer et al. (2011) devised a “tool that allows the creation of an augmented soundscape from a user-contributed sound repository” (p.1), requiring memory and interpretation (A1, A2) and an understanding of location-based concepts, such as map overviews (B1, B2). Woznievs et al., in the same conference, showcased their Object Creator, which was capable of cutting elements from the visual feed to use as an augmentation: “A template is chosen from a menu, and depending on the selection, a semi-transparent overlay will identify the part of the image to be mapped onto the geometry” (p.3). This encourages differentiation and discrimination of the local environment (A4) and represents new ground in the way that augmentations could be crafted.

The following year, McClean et al. (2012) put forward an authoring system that “consists of dragging and dropping 3D widgets into the front parallel view of the façade. The user can then set the position, orientation and scale of these widgets” (p.2), covering the transformational affordances (A1, A2) and an appreciation of parallelism and perspective for proper placement (B1, B2). Finally, Jens de Smit (2012) also published a system, entitled Layar Creator, which allowed organising sets of related augmentations into ‘campaigns’. “Once at least one page has been added to the campaign, users can drag “buttons” from the right hand side of the screen onto a page. These buttons ... have an associated action to trigger when an end user taps the augment on the mobile client. Possible actions include, but are not limited to, opening a website, playing a video and initiating a phone call” (p.2). The use of content selection tools indicated support for recall and understanding of the facts present in the campaign (A1, A2) and their embedding in a context that utilising web-based connectivity promotes the use of conceptual knowledge (B1), so that the campaign can have a broader impact (B2).

2.5.3 The appearance of head-mounted displays (2013 - 2021)

The third phase, the period in which AR also becomes a head-mounted phenomenon, is supported by development in two areas. One is a new generation of depth-aware sensors, led by Microsoft in their release of the Kinect for Xbox One in 2013, which introduced time-of-flight technology into the devices. The Kinect, already a commercial success in the world of interactive gaming, could achieve better resolution, cheaper manufacturing costs (due to electronic calibration) and had better resistance to ambient light distortions. This improvement would also feature in the HoloLens, released in 2016, which combined the



depth camera with an optical waveguide - a transparent near-eye display - and advanced processing hardware. The second success was that of the Oculus Rift Kickstarter campaign, run in 2012, that would rekindle interest in the HMD. Examples combining time-of-flight cameras with VR headset visualisation could create an immersive equivalent of the mobile video passthrough.

In terms of software development kits, the Vuforia computer vision library would be the next widely successful entry, released in 2011 and improved over a decade. Two dedicated AR development toolkits, ARKit for iOS and ARCore for Android, would follow in 2017 and 2018, respectively. Additional platforms gave rise to a more consistent and varied type of authoring toolkit and, in this period, many new features and approaches have been demonstrated.

The singular authoring toolkit identified in 2013, from Ruminski et al., was one of few to use a quiz format to support learning (A4): “the designer can select the onPresentationBegin event. Then he or she can choose the showQuiz action and can select from a menu an id of the quiz. The selected quiz will be shown when the virtual object appears on the screen.” (p.11).

Yoo and Lee (2014) explored the use of gestures to control authoring functions, demonstrating the apprehension of a conceptual framework (B1) as well as eliciting indicative actions from the user (B2, B3). While Ty et al. (2014) demonstrated, in detail, the pipeline for building augmentations (including the use of affine transformations), Santos and Luebke (2014) went on to show how these could be used in a learning setting, to help recognise and understand the images in context (B1, B2): “teachers can modify the appearance of the image. In this example, it is desirable to scale and position the lungs correctly on the body” (p.555).

Noletto et al. (2015) gamified their approach to authoring, illustrating how the construction of conceptual models (B1, B2) can be made more intuitive: “For each mission. the game designer may include game mechanics” (p.106). Furthermore, they used the same framework to encourage the players to remember procedure sequences (C1): “The mission to destroy the barricades will be composed of five mechanics ordered sequentially” (p.106). Their outdoor, location-based mobile game also requires an internet connection, a feature emerging across many of the modern systems.

Kim and Park (2015) developed a content management system that “stores components such as 3D model, marker information, audio and video in the database through CMS Web Interface” (p.64), but otherwise only demonstrated visualisation functionality in the software (A1, A2). Similarly, the Argon framework, described by Speigner et al. (2015) used web-sourced content and focused on the anchoring and visualisation of virtual content (A1, A2). This iteratively developed system did provide different modes for content viewing, for instance, “[i]f the user is at the locations featured, she can turn off the provided panorama



image and have a true augmented reality experience overlaid on the live video feed from their device” (p.7). This practice, of considering different modes that consider the benefits and challenges of the target device, would also become more frequent.

The first authoring systems to exclusively target a head-mounted device is described by Shim et al. and Yang et al. who, with their colleagues in the Media System Lab, published two papers in 2016 with this ambition. While a hand-held device was used, it was as a controller rather than a central display. Their work focused on gesture training, investigating tangible (i.e. physical) interaction, gesture based interaction and a blend of the two, using a marker on a smartphone to provide additional configurability to the marker layout and use. For hardware they used an Oculus Rift connected to a SoftKinetic DS325, which is a short-range time-of-flight camera capable of hand tracking. Considering the learning affordances, each of the interaction types presents a somewhat different set of possibilities. The tangible interaction provides the user with a chance to apply procedural knowledge (C3) but without a broader conceptual basis outside normal movement. The touch screen interface described by Shim et al. (2016) manages typical affine transformations (A1, A2). Finally, the blended element encompasses both of the above: “The mobile device interactions are tangible to the user and support discrete control; they also provide continuous interaction for manipulating 3D objects in combination with interactions that use hand gestures” (p.1430). This furnishes learners with clearer examples and use of the interaction model (B1, B2).

From 2017 onward there is a greater proliferation of authoring systems, certainly impacted by the excitement surrounding the release of the HoloLens the previous year. The term ‘Augmented Reality’ was becoming pervasive in technology circles and more use cases were being trialled. Procedural knowledge - that which is related to *how* to go about meeting a learning objective - was more frequently addressed and the overall complexity of the software architectures was increasing. In terms of research, this makes greater specification and, given a larger body of contemporary knowledge to draw on, has allowed the authors to make more specific claims about the benefits of the technology.

One area where this trend is evident is around the augmentation of text. In 2017 Raso et al. combined data mining operation with AR visualisation, showing how the content could be situated in a physical (A2), as well as semantic, context (B2). Lytridis and Tsinakos (2018a, 2018b) present ARTutor, which “enables students to enhance their understanding of the teacher’s material by displaying explanatory interactive digital content on top of the traditional book” (p.10), again allowing a user to extend their understanding of the material (A2), representing the information in other forms (B2).

Shekhar et al., writing in 2019, describe the connection of AR and natural language processing in such a way that “allows the user to include various commonplace objects and ... can describe their relations (e.g., “A chair next to the bed”) and can include humans or animals (e.g., “A man is sitting on a sofa with a dog”)” (p.64). This presents learners with



the opportunity to reimagine spoken or printed ideas using visual aids (A2) and also in recalling facts (A1) and ideas (B1) related to the content.

With more connections being made between aspects of the various components of the authoring tools, it is no surprise that web-based implementations were also becoming more prevalent. In 2017 alone, four of the six authoring tools identified relied on web-based communication (Maia et al., Rodrigues et al., Reynolds, and Raso et al.).

The maturity of the systems being produced is also reflected by their implementation into workplace environments. Many early systems were used in a lab setting or controlled environment, avoiding many of the subtle and unpredictable elements of an active workplace. More robust localisation, scanning and mapping algorithms meant that quick movements, hard-to-scan surfaces and light levels were less disruptive, reaching a usability threshold that led to commercial interest. The first instance of this was reported by Schleuter in 2018, who used the Unity game engine to provide a remote-assistance service that, in one instance, “guided a user through all disassembly steps, including the removal of the front bolts, plate, gasket, spring, and pistons” of an engine (p.47).

A study by Blattgerste et al. (2019) is unique in that it is the only authoring toolkit that directly aims to support those with cognitive impairments. A specific case was presented, “where the workers have to perform the cleaning of a modern coffee maker ... that incorporates 20 distinctive steps (e.g. pressing specific buttons and disassembling, emptying and cleaning specific parts of the machine” (p.7). There is a clear engagement for the application of knowledge, of both factual (A3) and procedural types (C3), supported by the range of typical functions seen in previous authoring systems, including the retention and understanding of both facts and principles (A1, A2, B1, B2).

The use of sensors has also recently been demonstrated as an extension to previous authoring toolkits, where they are either distributed in space (Jin et al., 2018) or around the body (Limbu et al., 2018). In the first, the authors constructed a system called “TanCreator, which allows children to create AR maze games and operate AR characters with paper tokens and sensors in the real world” (p.85). The use of definable inputs allow learners to apply previously understood knowledge (A3, B3) and, given the system also permitted the sequencing of actions to achieve set goals, also the application of procedural knowledge (C3).

The use of body-worn sensors in an authoring toolkit was reported by Limbu et al. (2019). In that framework, which was called Wearable Experience for Knowledge Intensive Training (WEKIT), the use of the HoloLens permitted the tracking of both the head and, when visible, the hands of the user, known as a ‘Ghost track’, which “allows visualization of the whole-body movement of the expert or the earlier recording of the trainees themselves for imitation and reflection” (p.160). In this sense, the system offers the learner not only a



method to apply procedural knowledge (C3), but also a method to evaluate the action, with reference to that of the expert (C5).

T Wang et al. (2020) produced a similar recording track of the body and hand motion, building it into a context-aware application (CAP): “An instructor wants to demonstrate his/her routine task of repairing a bike, so he/she creates a CAP using CAPturAR with three sequentially connected events, shaking the lubricant, spreading it on the front wheel and then on the back wheel. A novice comes and follows the tutorial [...] CAPturAR detects once the novice completes a step and starts to play the demonstration of the next step.” Again, the ability to mimic the procedural demonstration of an expert facilitates evaluative learning (C5) while also providing information to support the understanding of the procedure (C2), together with the factual knowledge about the parts of the bicycle (A2) and their interconnections (B2).

Perhaps one of the most visually complex, but nonetheless graphical, is that described by Nguyen et al., BlocklyAR (2020). This tool is a graphical coding framework, where elements can be fit together to produce a definition of an AR scene, which is then converted into executable code. “The definition describes the block’s appearance and how it behaves, including the text, color, shape, and the connections to other blocks while the generator translates the block to executable code” (p.5). The ability of this framework to let the user manage affine transformations (A1, A2), non-linear sequencing of events and triggers (B1, B2) and apply this knowledge (A3, B3) gives us the first set of general affordances. Since “the visual AR component enables enthusiasts to experience their coding schemes in the mixed 3D space” (p.10), we can also attribute a creative affordance to this system, as the user is constructing an entire model, based on the factual information provided by the tool (A6).

Chidambaram et al. (2021), in a system targeting head-worn devices, also incorporate object recognition algorithms into their system, though use registration for initial location of the virtual content, as performance limits remain a factor when performing complex tasks (feature recognition). In practice, this means that “novices can view videos previously overlaid by the expert users to complete the current procedural task in progress” (p.243). This assists the user in understanding the video, containing either factual (A2) or conceptual knowledge (B2) but will also support recall (A1, A2). The focus on retaining and understanding procedural knowledge (C1, C2) and also supports its application (C3): “ProcessAR also enables the user to perform voice recordings for the purpose of clarifying tasks or to explain possible error cases” (p.242).

2.5.4 Summary of Findings

Despite early implementations focusing almost exclusively on factual and conceptual knowledge, new technologies supporting body tracking (with either machine vision or a



HMD) have provided an excellent basis for adapting AR in support of procedural knowledge learning.

The top three levels in the Cognitive Process Dimension -- analyse, evaluate and create -- were rarely demonstrated in the literature. It is clear that more complex implementations of AR authoring systems will tend to have more classifications, as each learning affordance requires additional programmed functionality. The complexity of code needed for, say, a geometric comparison of two 3D models (as part of evaluation: A5) will be greater than that needed to only visualise the models (in support of memorisation: A1). As such, the degree of support for increasingly complex learning objectives has steadily increased, though this has been punctuated by the development of improved technology or entirely new devices.

Despite such common foundations among the authoring toolkit documented here, there is little standardisation around how to test the systems. While many opt for the “quick and dirty” System Usability Scale put forward by Brooke, the manner in which the questions are framed is usually not reported and there is little in the way of statistical analysis, in part because the sample sizes are often small (10 - 30 people). The risk is that a subjective notion of usability can be interpreted as an objective measure of value. More than an omission or fault, the issue is that SUS is simply not designed to provide this kind of measure.

2.6 Conclusion

This study has investigated the development and implementation of systems that support users in constructing their own versions of AR. The applications all feature world-locked visualisations presented in a way that allows the user to design, configure or otherwise manage their presence and functions in this new reality. They vary considerably in their complexity and ambition, but all offer an opportunity for a user to learn something new. These opportunities, framed and defined by Bloom’s Revised Taxonomy, have been used to compare them.

In summary, what is apparent is the maturing of a new type of application that is constrained by technology and the environment and encourages both participation and creativity from the user. From Tiles in 2001 to BlocklyAR in 2020, this paper has mapped the first learning affordances to the creation of an entire, testable framework. It spans the entire range of the Cognitive Process Dimension (for Factual Knowledge), showing how software developers, just as any other learners, build upon the work of others and make their way through the levels of constructivist learning.



3. System Architecture

The ARETE system landscape consists of several interoperable systems. The most important platforms are:

- The <https://github.com/openARLEM/>: a Moodle, currently in version 3.10, hosted by UCD under <https://arete.ucd.ie>. The Moodle hosts the ARLEM repository plugin, developed by OU (see upcoming deliverable D3.4, <https://github.com/ARETEedu>, version 2.0). The plugin utilises the ARLEM.js javascript-based validator developed for UCD (<https://github.com/openARLEM/>, version 1.0-beta).
- The **MirageXR cross-platform applications**: MirageXR is currently available for iOS, Android, Hololens-1 (X86), and Hololens-2 (ARM). The applications combine an editor (the MirageXR ‘Producer’) and an execution engine (the MirageXR ‘viewer’).

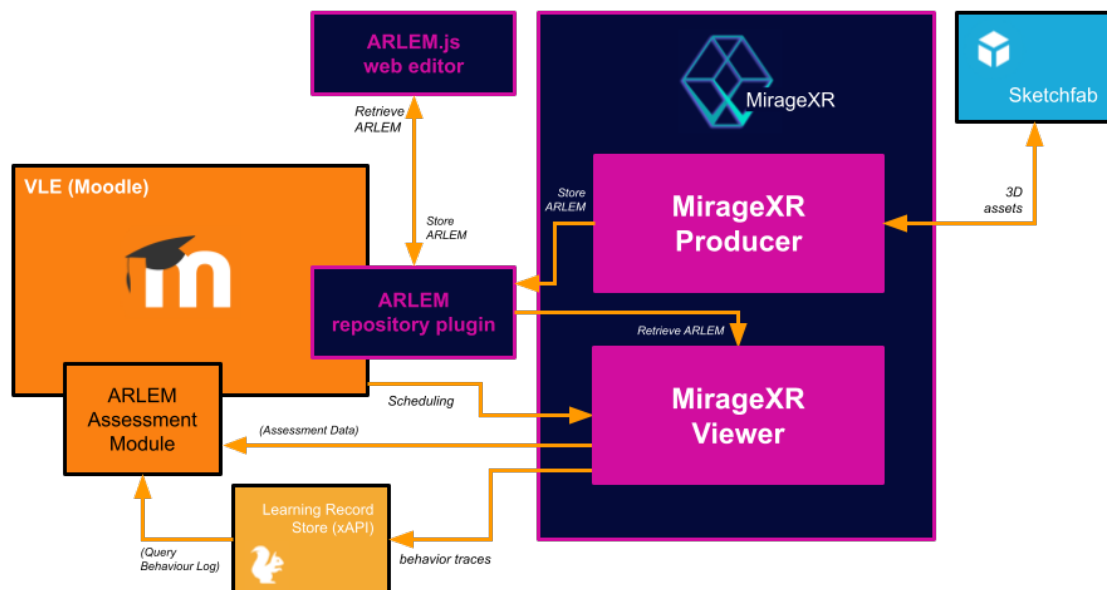


Figure 3.1. System landscape: VLE, LRS, plugins, MirageXR, Sketchfab.

Communication between the two core systems happens via the web-services provided by the ARLEM repository plugin. This includes authentication, search, listing available learning experiences, and scheduled learning experiences. Moreover, storage and update routines allow to move locally created learning experiences to the server for distribution to students. There are, however, several additional systems involved. An interface to sketchfab allows retrieving more than four million 3D objects and animations, so that teachers designing learning experiences can integrate them at ease. Moreover, a learning analytics interface (xAPI) logs behavioural traces to a learning record store (LearningLocker, v5.1.0).

In preparation is currently additional assessment functionality - in Figure 3.1 depicted as ‘ARLEM Assessment Module’, which is supposed to provide a teacher interface to monitor progress and check assignments.



3.1 Component architecture of the MirageXR apps

When looking at the basic component structure of MirageXR, the following can be summarised. On a macro-level, the system is layered into data, service and app layer. The app layer implements model-view-controller patterns. On a meso-level, the system distinguishes the following key components: login (yellow), communication (orange), activity management (green), spatial mapping (light purple), tracking (dark purple), and interaction/event management (red).

The login management uses the underlying Moodle, allowing users to log in to see non-public resources (scheduled from courses, private). Only authenticated users can upload and store XR learning experiences on the Moodle server, otherwise they remain local on the device they are created on.

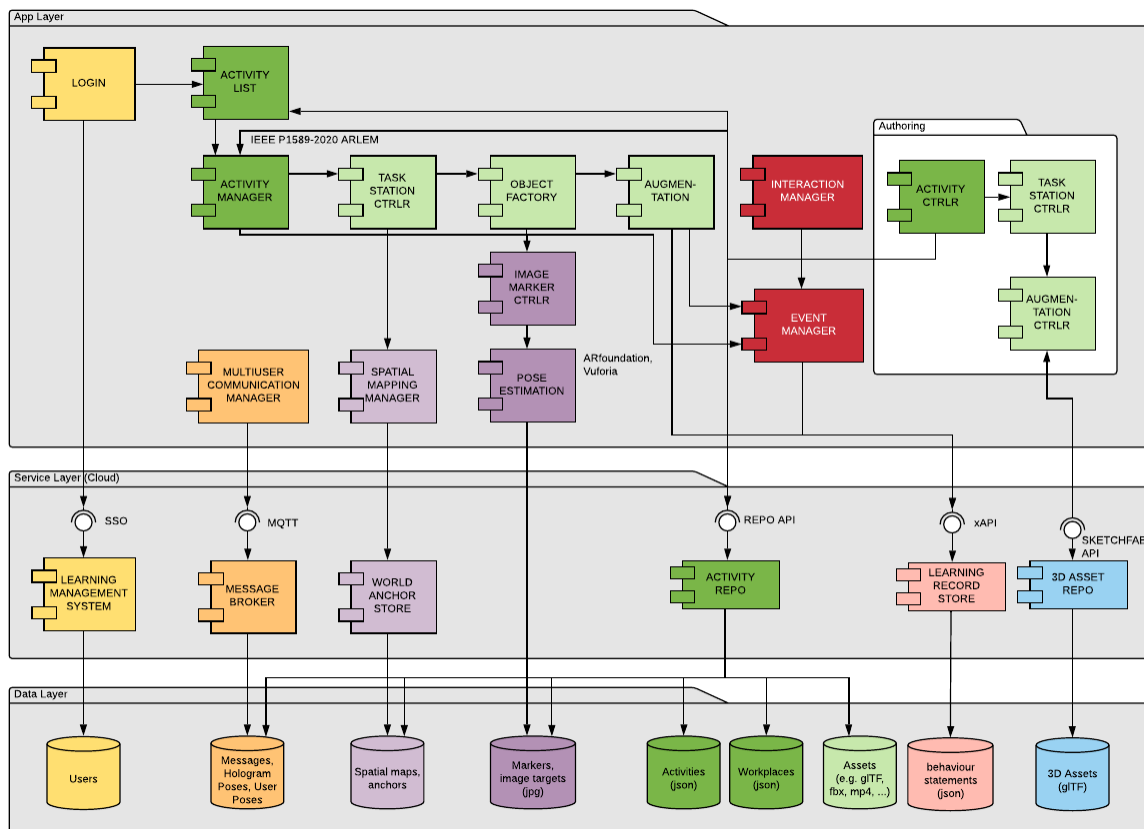


Figure 3.2. MirageXR basic component architecture.

The heart piece of the applications in the green area, starting with the activity list. Selected units will be downloaded from the repository back-end, stored locally, and then executed using the activity manager. The activity manager manages the task stations where the actions take place, using an object factory to instantiate a rich set of augmentations (see



Section 4, augmentations). For the producer, this is complemented by an activity controller, task station controller, and augmentation editor controller.

An event centre is used to separate concerns, using callbacks to distribute events to listening components. For example, the event centre has a start activity event, which then is used by the experience service to log an according behaviour statement to the learning record store ('user started activity abc').

An interaction manager encapsulates device specific user interaction methods so that those participating UI components that are working similar across platforms do not have to bother implementing device-specific behaviour.

At the time of writing, the communication component (orange) is still being overhauled, and at the moment only the basic mqtt support remains, which can be used to communicate with Internet-of-things sensors and send messages. However, as this has a certain degree of complexity, at the moment there is no visual editor in the MirageXR producer for this and activity scripts have to be modified in the repository back-end in the ARLEM.js script editor.

There is a spatial mapping helper which links with the device specific behaviour for the spatial map and an image marker controller implements cross-device image target / marker tracking and pose estimation (mobile: ARfoundation, smart glasses: Vuforia).

There is the possibility to further extend this component architecture to transmit spatial mapping data to the server, in order to support discovery services, server-sided spatial understanding, and more sophisticated spatial adaptation. We experimented with Spatial Understanding services (e.g. ground plane identification, basic object recognition), but have not yet merged any of this experimental code back into the development or master branch. This would also offer possibilities for world knowledge extraction (e.g. in-app 3D scanning).

Moreover, a boundary box for the room and the augmentation-relative indoor user position might be useful to track in the xAPI, so as to gather some usage context information about the adaptivity required for learning experiences that are re-enacted in different rooms with differing geometry.

Important deliberations about this component architecture were already introduced in deliverable D3.1, Section 3 "Identified Functional Requirements", in particular the deliberations regarding reference points as specified by ETSI ARF.



3.2 Augmentations: How to create a new augmentation primitive

What are augmentations? The IEEE standard P1589-2020 defines an augmentation as the "digital representation of effector outputs that serve to stimulate the sensory experience of the user, including output to visual, audio, haptic, and other modalities". Typically, this means an augmentation is a hologram or spatial audio output, or both. Few systems currently support other modalities.

The IEEE standard defines further specific augmentation types (in the wording of the standard: so-called "augmentation primitives"). These are types of annotations, the most basic ones are: an image, a video, audio, a 3D model, or a label.

MirageXR implements all basic mandatory types of the IEEE standard, but provides a range of additional augmentation types. Most notably, this includes a GhostTrack, where a trainer hologram can be recorded from body movement and voice of the trainer to then be replayed independently by the user.

3.2.1 Existing Augmentation Types

This is the full list of augmentation types supported by MirageXR: Audio, Image, Video, GhostTrack, Label, Visual Effects, Action Glyphs, 3D Model, Character Model, Pick & Place, Image Marker, Plugin, Drawing.

3.2.2 Implementing a new Augmentation Type

When you want to extend the augmentation types available, you will have to take care of registering the new type in various places, which has to be implemented using specific classes to ensure editor, player, data storage, etc. know what to do.

3.2.2.1 Register with Model, View, and Controller

The new augmentation type needs to be added in several places: to the Data Model, the viewer classes for EditMode and PlayMode

3.2.2.2 Register the new augmentation with the Data Model



First, you need to add a new content type to the [ContentType](#) class:

```
public enum ContentType
{
    UNKNOWN,
    IMAGE,
    VIDEO,
    AUDIO,
    GHOST,
    LABEL,
    ACT,
    VFX,
    MODEL,
    CHARACTER,
    PICKANDPLACE,
    IMAGEMARKER,
    PLUGIN,
    DRAWING
}
```

Also add a name of type string here:

```
private const string UNKNOWN = "Unknown";
private const string IMAGE = "Image";
private const string VIDEO = "Video";
private const string AUDIO = "Audio";
private const string GHOST = "Ghost";
private const string LABEL = "Label";
private const string ACT = "Action";
private const string VFX = "Vfx";
private const string MODEL = "Model";
private const string CHARACTER = "Character";
private const string PICKANDPLACE = "Pick and place";
private const string IMAGEMARKER = "Image marker";
private const string PLUGIN = "Plugin";
private const string DRAWING = "Drawing";
```



And add the according type description here:

```
private const string UNKNOWN_HINT = "Unknown";
private const string IMAGE_HINT = "Take a photo and add it as an augmentation to this action step.";
private const string VIDEO_HINT = "Record a video and add it as an augmentation to this action step.";
private const string AUDIO_HINT = "Record an audio and add it to this action step.";
private const string GHOST_HINT = "Ghost track lets you record your movement in the real world and adds it to the
this action as a virtual avatar.";
private const string LABEL_HINT = "Label augmentation lets you add a text.";
private const string ACT_HINT = "Pre-define models which represents verbs.";
private const string VFX_HINT = "Visual Effects lets you add effects like fire, explosion, etc.";
private const string MODEL_HINT = "You can import 3d models from Sketchfab.com to this action step. Sketchfab is a
library with more than 3 million 3d models which half millions of them are free.";
private const string CHARACTER_HINT = "Add an AI character to this action. You can choose between different
characters that each of them can do different tasks. ";
private const string PICKANDPLACE_HINT = "Add flags on an objects.";
private const string IMAGEMARKER_HINT = "Image marker allows to take a photo of an object (or select a pretrained
image target) and thus allow to move task stations with the marker around.";
private const string PLUGIN_HINT = "Augmentations that are created for specific activities";
private const string DRAWING_HINT = "Draw in 3d space";
```

Add the path to the mobile (screen space UI) icon:

```
private const string IMAGE_IMAGE_PATH = "Materials/Textures/imageeditor";
private const string VIDEO_IMAGE_PATH = "Materials/Textures/videoeditor";
private const string AUDIO_IMAGE_PATH = "Materials/Textures/audioeditor";
private const string GHOST_IMAGE_PATH = "Materials/Textures/ghosteditor";
private const string LABEL_IMAGE_PATH = "Materials/Textures/labeleditor";
private const string ACT_IMAGE_PATH = "Materials/Textures/glypheditor";
private const string VFX_IMAGE_PATH = "Materials/Textures/vfxeditor";
private const string MODEL_IMAGE_PATH = "Materials/Textures/modeleditor";
private const string CHARACTER_IMAGE_PATH = "Materials/Textures/charactereditor";
private const string PICKANDPLACE_IMAGE_PATH = "Materials/Textures/pickandplaceeditor";
private const string IMAGEMARKER_IMAGE_PATH = "Materials/Textures/imagemarkereditor";
private const string PLUGIN_IMAGE_PATH = "Materials/Textures/plugineditor";
private const string DRAWING_IMAGE_PATH = "Materials/Textures/drawingeditor";
```



and the IEEE P1589-2020 predicate identifier for the data model:

```
private const string PREDICATE_UNKNOWN = "unknown";
private const string PREDICATE_LABEL = "label";
private const string PREDICATE_IMAGE = "image";
private const string PREDICATE_AUDIO = "audio";
private const string PREDICATE_AUDIO_V2 = "sound";
private const string PREDICATE_VIDEO = "video";
private const string PREDICATE_GHOST = "ghosttracks";
private const string PREDICATE_GHOST_V2 = "ghost";
private const string PREDICATE_ACT = "act";
private const string PREDICATE_VFX = "vfx";
private const string PREDICATE_MODEL = "model";
private const string PREDICATE_MODEL_V2 = "3d";
private const string PREDICATE_CHARACTER = "char";
private const string PREDICATE_PICKANDPLACE = "pickandplace";
private const string PREDICATE_PICKANDPLACE_V2 = "pick&place";
private const string PREDICATE_IMAGEMARKER = "imagemarker";
private const string PREDICATE_PLUGIN = "plugin";
private const string PREDICATE_DRAWING = "drawing";
```

3.2.2.3 Register new augmentation type with the editor

We also need to declare the new augmentation primitive also to the editor subsystem in the [ActionEditor.cs](#) class.

First, we create a new SerializeField to which we assign the according editor prefab in the inspector:

```
[SerializeField] private GameObject characterAugmentationPrefab;
```

Then add the new augmentation in listOfPois.

```
#if UNITY_ANDROID || UNITY_IOS
    private readonly string[] listOfPois = { "image", "video", "audio", "ghost", "label", "act", "vfx", "model", "character",
    "pick&place", "image marker", "plugins" };
#else
    private readonly string[] listOfPois = { "image", "video", "audio", "ghost", "label", "act", "vfx", "model", "character",
    "pick&place", "image marker", "plugins", "drawing" };
#endif
```



Then we add according code segments for instantiating the augmentation, editing the augmentation, and disabling all editors:

```
public void OnAnnotationAddItemSelected(ContentType type)
{
    ...
    case ContentType.LABEL:
        labelEditor = LoadEditorPanel<LabelEditor>(TextEditorPrefab);
        labelEditor.SetAnnotationStartingPoint(DefaultAugmentationStartingPoint);
        labelEditor.Open(detailView.DisplayedAction, null);
        break;
    ...
}
public void EditAnnotation(ToggleObject annotation)
{
    ...
    case "label":
        labelEditor = LoadEditorPanel<LabelEditor>(TextEditorPrefab);
        labelEditor.Open(detailView.DisplayedAction, annotation);
        break;
    ...
}
public void DisableAllPoiEditors()
{
    ...
    if (labelEditor)
        labelEditor.Close();
    ...
}
```

3.2.2.4 Register the augmentation type with the Player

The instantiation of the augmentations happens in the object factory [ObjectFactory.cs](#), where the info from the activity data model is processed, and the according augmentation game objects are instantiated and destroyed.

WARNING: since the IEEE P1589-2020 data model allows several ways of instantiating augmentations (directly, or attached to a 'tangible'), the way your new augmentation is expressed in the data model may vary - and, consequently, its instantiation.



You need to register the new augmentation to the Toggle method. Here is an example (REMEMBER: can vary, depending on use of IEEE P1589-2020):

```
private static void Toggle(ToggleObject obj, bool isActivating)
{
    ...
    case "label":
        if (isActivating)
            ActivatePrefab("LabelPrefab", obj);
        else
            DestroyPrefab(obj);
        break;
    ...
}
```

And with DestroyPrefab. Here is an example (REMEMBER: can vary, depending on use of IEEE P1589-2020):

```
private static void DestroyPrefab(ToggleObject obj)
{
    ...
    case "label":
        temp = GameObject.Find(path + "label_" + obj.text.Split(' ')[0]);
        break;
    ...
}
```

3.2.2.5 Register for saving / deleting data events

If your augmentation need to save some extra data as a for example JSON file you can create a method and hook it into the SaveData() method in the [ActivityManager](#). Soon to be replaced with a new event based system.

Consequently, deleting extra data needs to be implemented for when a user deletes the augmentation. At the moment, you need to add your data deletion in DeleteAnnotation().

This will soon to be replaced with the new class system, and the ActivityController will accept registration of callbacks for the new event-based system.

3.2.3 Programming your new augmentation type

3.2.3.1 Editor Panel Prefab

First, create the user interface for the editor and store this as a prefab:

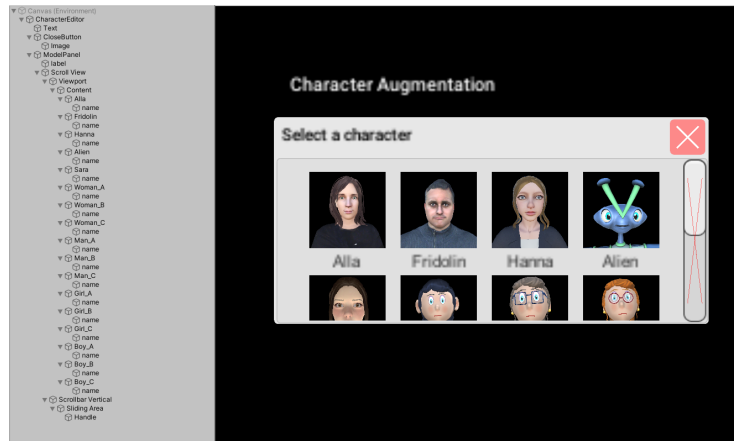


Figure 3.1.3.1. Example of an editing panel prefab.

This user interface is for the author and producer of a new learning experience. It will list all options that the user can set.

A sensible name for this prefab is 'augmentationNameEditor', where augmentationName is replaced by the new augmentation name (e.g. 'characterEditor').

3.2.3.2 Editor View Class (Spatial UI, e.g. HoloLens)

Next, we need to create the corresponding editor view class. In our example here, this class is called 'CharacterEditorView' and assign it to the CharacterEditor prefab in the previous step. This class contains the common methods that all augmentations should have including `close()`, `open(Open(Action action, ToggleObject annotation))`, and `Create()`.

`close()`: Close the editor panel

`open()`: Open the editor panel

`create()`: Create the augmentation

In the `create()` method you will define the type and the name of the augmentation. This string is stored in `ToggleObject.predicate` and will be displayed on the annotation augmentation button (e.g. `char:Alla`). This is also needed to identify the instance, when, for example, initiating or destroying the augmentation's game object.

3.2.3.3 Editor View Class (Mobile UI, e.g. Android and iOS)

The new editor class must inherit from the `PopupEditorBase` class and contain an implementation of the `OnAccept()` method and the `editorForType` property.

`OnAccept()`: this method that is called when the content being created is saved.

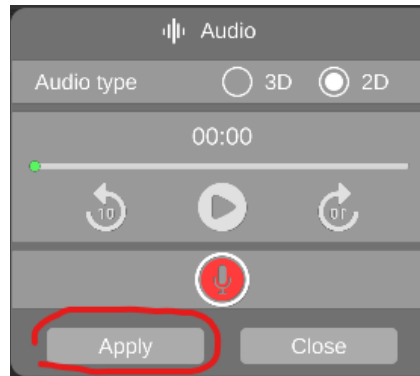


Figure 3.1.3.3. Example editor panel with 'accept' button.

In the body of this method you need to deactivate the old content if there was one and fill it with new data and activate it.

```
protected override void OnAccept()
{
    if (_content != null)
    {
        EventManager.DeactivateObject(_content);
    }
    else
    {
        _content = ActivityManager.Instance.AddAnnotation(_step, GetOffset());
    }

    _content.predicate = $"char:{_prefabName}";
    EventManager.ActivateObject(_content);
    EventManager.NotifyActionModified(_step);

    SetupCharacter();
    Close();
}
```

Old content is inserted as an argument to call the popup window:

```
private void OnContentClick()
{
    var type = ContentTypeExtension.ParsePredicate(_content.predicate);
    var editor = _parentView.editors.FirstOrDefault(t => t.editorForType == type);
    if (editor == null)
    {
        Debug.LogError($"there is no editor for the type {type}");
        return;
    }
    PopupsViewer.Instance.Show(editor, _parentView.currentStep, _content);
}
```



You will have to set the `editorForType`, which is the property on the basis of which name and icon for the popup window will be set.

```
public override ContentType editorForType => ContentType.CHARACTER;
```

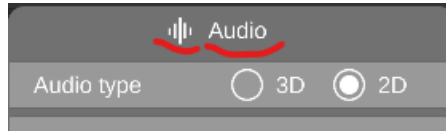


Figure 3.1.3.3 Example icon chosen based on `editorForType` property.

The created prefab must be nested in the list of editors of the `ContentListView` class in the View prefab.

```
[SerializeField] private PopupEditorBase[] _editors;
```

3.2.3.4 Controller Class

Next, we need to create the controller class, 'CharacterController'. This class will be assigned to the instantiated augmentation prefabs that represent the augmentation - which will be instantiated at runtime. For instance, it will be attached to the character model for character augmentation and or to the label prefab for the label augmentation. You can implement all needed methods in this class, but of course additional classes can be linked if complexity requires it.

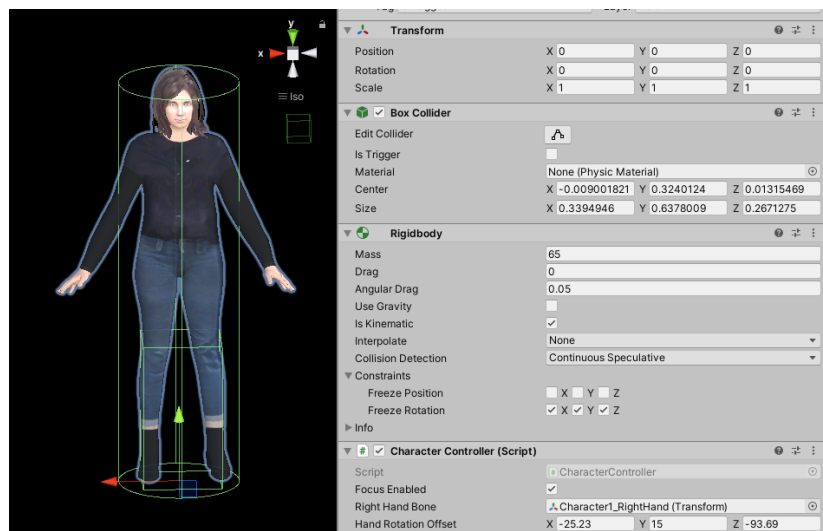


Figure 3.1.3.4 Controller Class attached component.

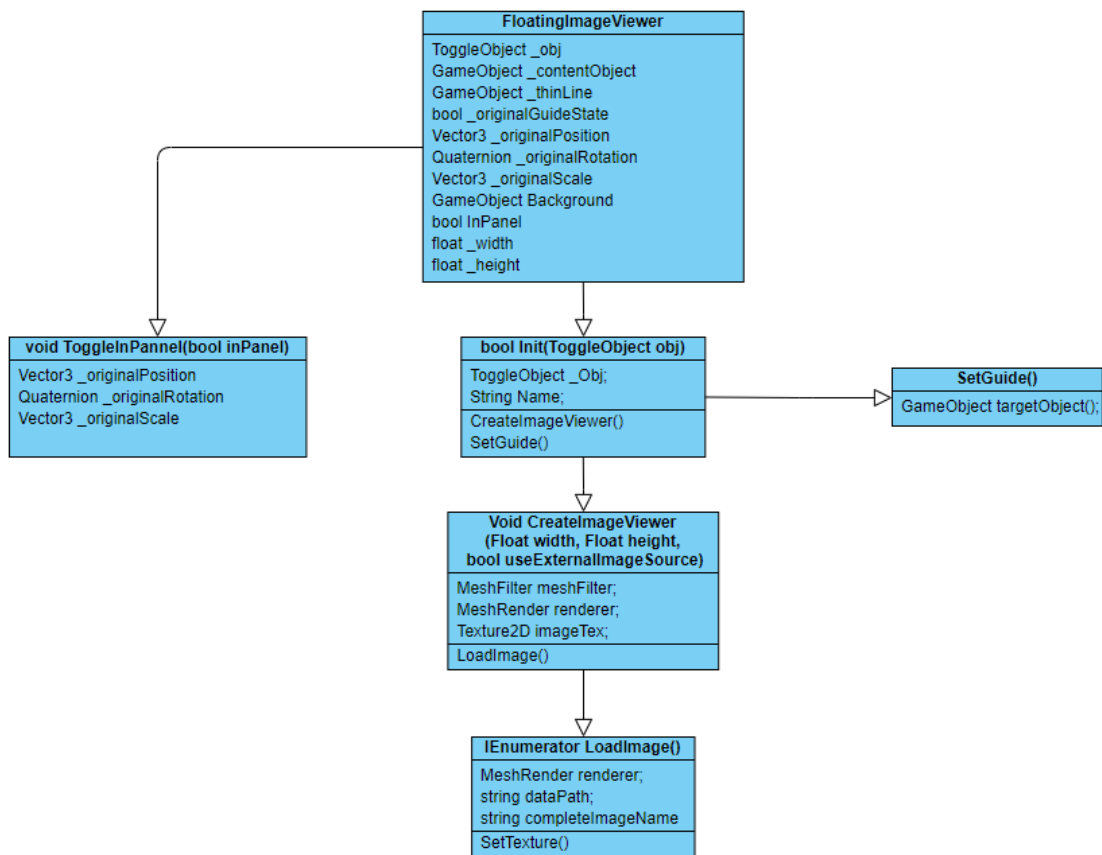


4. Implemented augmentations

4.1 Image

The image augmentation allows a user to take a photograph using their device camera. The augmentation is then displayed to the user as a floating billboard in 3D space.

Class Diagram:

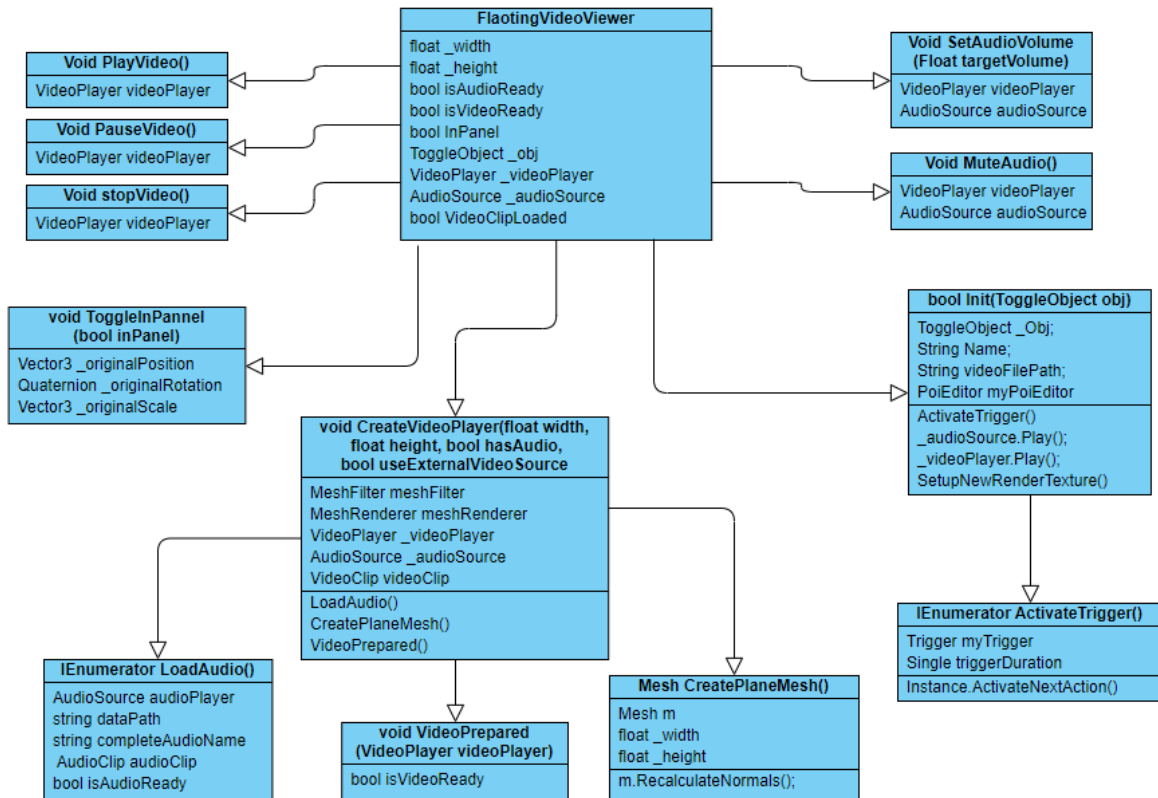


4.2 Video

Video augmentations are created and function almost exactly the same way as the image augmentation, except using a video as opposed to an image.



Class Diagram:



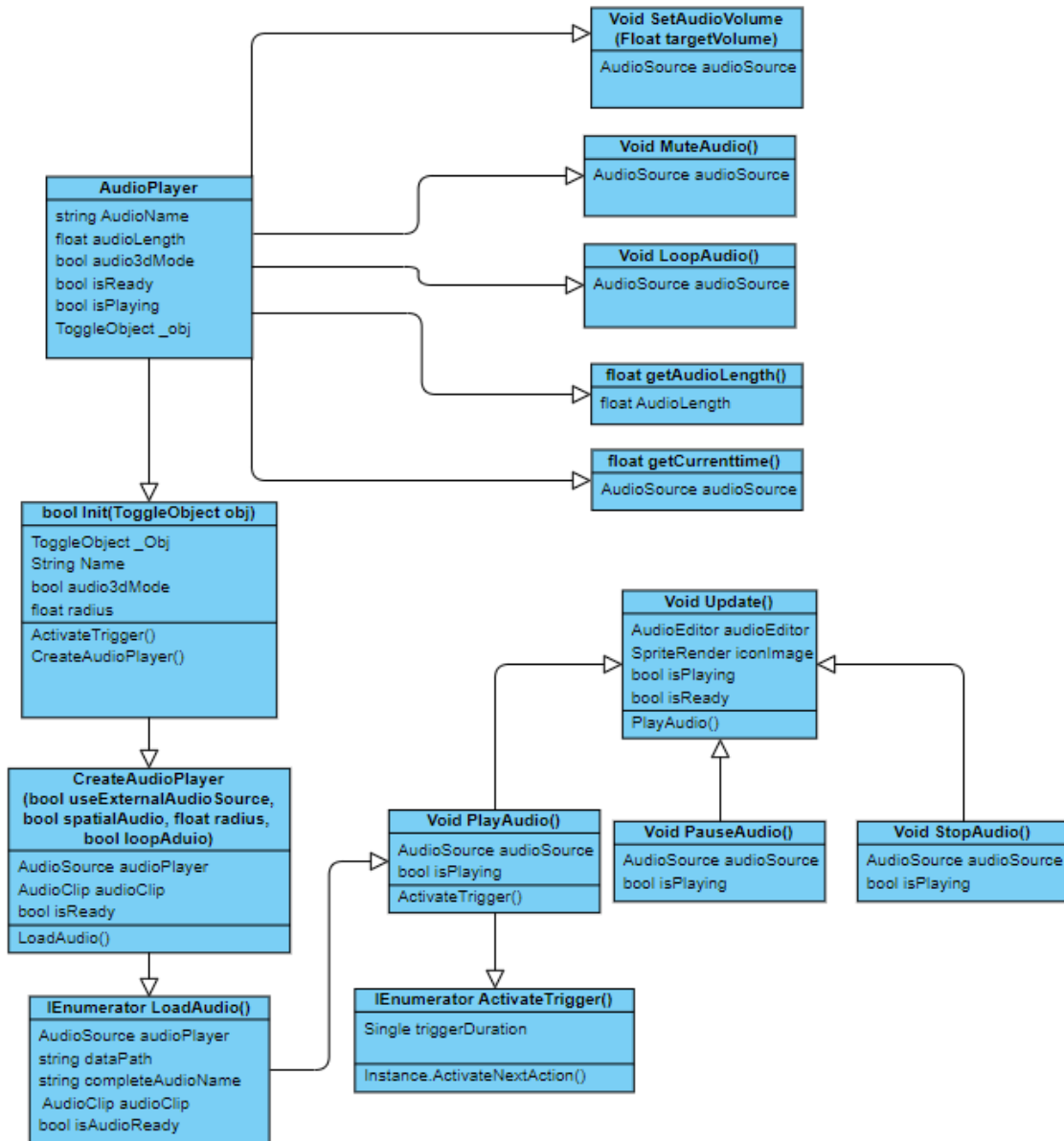
4.3 Audio

Users can record audio, typically a voice recording, using their device microphone. They can play the recording back.

Audio can be played in either 2D or 3D space, in case of the latter attaching the source of the audio to a specific location, with volume decreasing with the user's distance to the spot.

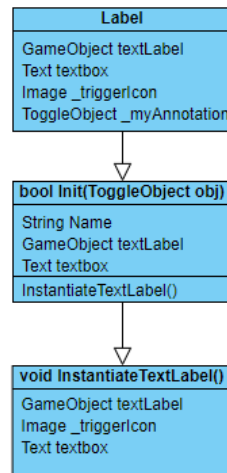


Class Diagram:



4.4 Label

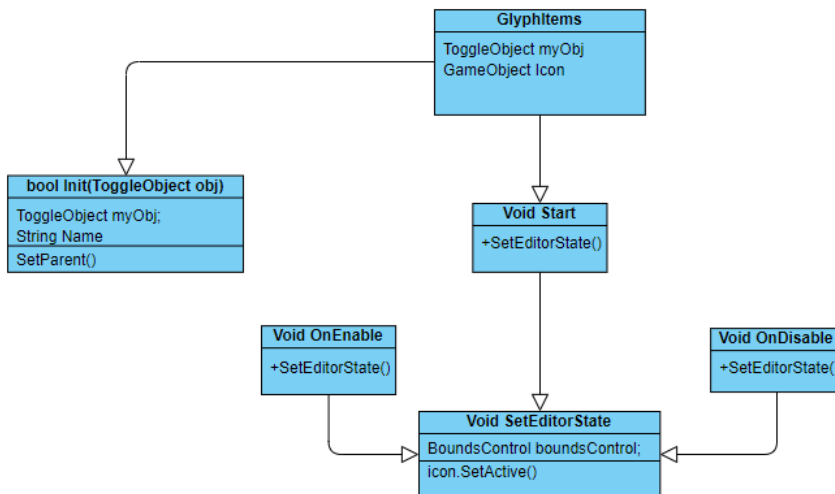
Labels are used to add text to a given action step. This augmentation allows a user to type text to be displayed on a label object which can then be placed anywhere in the 3D space



4.5 Action Glyphs

This augmentation contains a visual language - 3D glyphs that instruct the user on how they should be handling and moving objects in the real world. These symbolic instructions inform the user or trainee to, for example, *locate*, *move*, or *rotate* an object.

Class Diagram:



4.6 GhostTrack

The ghost augmentation tracks users’ movements in the environment, including position of head, body and hands. Furthermore, an audio recording is added. A visualisation of the captured actions can be previewed using the play function.

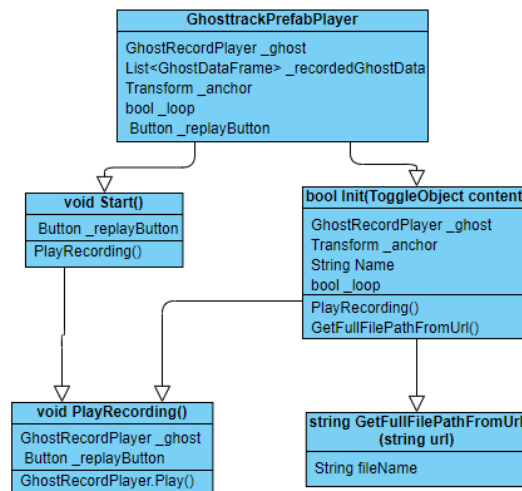


The ghost augmentation allows a user to record both movement and voice, which is then played back using a male or female abstract 3D avatar.

When using a HoloLens, the ghost augmentation will not only record the body location of the user, but will also track their hand movements.

Unfortunately, due to the lack of sensors, hand tracking is not currently available on mobile devices. Ghosts recorded on HoloLens, however, will track hand movements, and these recordings will play back correctly (with hand movement) also on the mobile apps.

Class Diagram:



4.7 Visual Effect

VFX augmentation lets user add different visual effects (vFx) to the activity, including fire, explosion, or steam. The VFX augmentation has the same structure as glyph augmentation.

4.8 3D model

The model augmentation provides an interface to sketchfab.com, a big repository of free and for pay 3D content. Using the Sketchfab API and the user interface provided for it, the user can search for suitable 3D models, download them, and place them in the activity.



4.9 Character Model

Character augmentation allows the user to create an intelligent avatar which can do different behaviors including following a path or the player, talk and play different types of animations. All behaviors are configurable in all action steps.

All setting for all steps will be saved in a JSON file in a "characterinfo" folder in activity folder. Each character will create its own JSON file in this folder with the name of the character POI identifier. At the start of each step, a method "ParseCharacters()" will be called, which loads the character and the settings for that action step.

The setting panel is only available in Edit Mode. Some behaviours become visible only in Play Mode. This includes, for example, looping for 'follow path' or 'trigger' for auto-progression to the next action step once animations or audio finished playing (whichever of the two takes longer).

Key classes:

- CharacterAugmentation: Create the character and the first node. Common methods with other augmentations are implemented here.
- CharacterController: Responsible for all behaviors and also loading and saving the data. This is the main class of the characters.
- Destination: Control the path nodes. Each node has a destination component attached.
- CharacterSettings: Holds the character settings

Character Augmentation base schema

#	Name	Explanation	Value Space	Datatype (Maximum Length)	Examples
1	scale	The size of the character in all steps	One scale per script	Float	JSON: { "scale": 1.0 }
2	steps	steps elements describe the steps of the learning activity and keeps a series of settings for the character in that action step.		List<Step>	JSON: { "steps": [{ "actionId": "TS-9dab6662-b5e6-4757-9f4e-92a1b5dbbc9c", "animationLoop": false, "animationType": "Hello", "destinations": { "points": [{ "index": 0, "name": "CharacterDestination(Clone)", "position": { "x": 1.5347528457641602, "y": -1.4765422344207764, "z": 0.72869873046875 } } } } }
2a	<i>actionId</i>	Action.id of the step	Alphanumeric	Character String (1000)	
2b	<i>animationLoop</i>	Specifies the selected animation loop	"true", "false"	Boolean	
2c	<i>animationType</i>	Specifies the type of the animation which the character will play.	"Idle", "Point", "Hello", "Bye", "Sitting", "Thumb Up",	Character String (1000)	



			“Thumb Down”, “Writing”, “Image Display”		<pre> }, "rotation": { "w": -4.371138828673793e-8, "x": 0.0, "y": 1.0, "z": 0.0 } } }, "returnPath": true }, "dialSaveName": "TS-9dab6662-b5e6-4757-9f4e-92a1b5dbbc9c_AN-93465f7b-67c7-4f08-8797-6fa5ab897b09.wav", "dialogLoop": false, "imagePoild": "", "movementType": "followpath" } } } </pre>
2d	<i>destinations</i>	The nodes which will create a path that the character will follow it		List<Destinat ion>	
2d .1	<i>points</i>	Holds the information of each node		List<Point>	
2d .1. 1	<i>index</i>	The index of the node in the nodes list	numeric	Integer	
2d .1. 2	<i>name</i>	Name of the node objects in the scene. It ends with (Clone) since the nodes will be created at runtime	Alphanumeric	Character String (1000)	
2d .1. 3	<i>position</i>	The local position of the node in the scene. It is a vector3 which contains three float number as x, y and z		Vector3	
2d .1. 4	<i>rotation</i>	The local rotation of the node in the scene. It is a Quaternion which contains four float number as w, x, y and z		Quaternion	
2. d. 2	<i>returnPath</i>	Specifies the loop of the path following. The character can repeat the following the path or just stop and play the animation clip at the end of the path.	“true”, “false”	Boolean	
2e	<i>dialSaveName</i>	The file name of the recorded audio which will be played by the character in the current step. The name contains (action.id + ‘_’ +	Alphanumeric	Character String (1000)	



		poi.id + '.wav') where poi.id is the id of the character poi			
2f	<i>dialogLoop</i>	Specifies the loop of the dialog player.	“true”, “false”	Boolean	
2g	<i>imagePoild</i>	Poi id of the image augmentation which will be displayed by the character when “Image Display” is selected as animation type	Alphanumeric	Character String (1000)	
2h	<i>movementType</i>	Specifies the type of the character movement in each step	“followpath”, “followplayer”	Character String (1000)	

4.10 Pick & Place

The Pick and Place augmentation allows users to add goal orientated holograms to their action steps. The augmentation consists of two objects, the pick and place object and the target sphere.

When in edit mode, the target sphere can be moved and placed wherever the user requires, this could be inside a real world object or another 3D object. The goal is to place the pick and place object onto the target sphere. When in play mode, the target sphere is not visible or editable as to hide the target location from a student.

The p&p object has two buttons attached when in edit mode, a lock button and a change model button.

- The lock button is used to set the p&p object’s starting location in the action step. To do so a user can place the p&p object anywhere in their 3D space and click the lock button, once locked the p&p object will bounce back to the locked location when it has been misplaced (i.e. not correctly placed on the target sphere).
- The ‘change model’ button allows a user to change the 3D model used for the p&p object to any 3D model that can be found on sketchfab. By default, the p&p object is an orange arrow with user written text on one side. By changing the model a user can use more accurate objects for a given scenario, for example, a p&p object could



be changed to a 3D model of a defibrillator paddle for an activity demonstrating how to use a defibrillator.

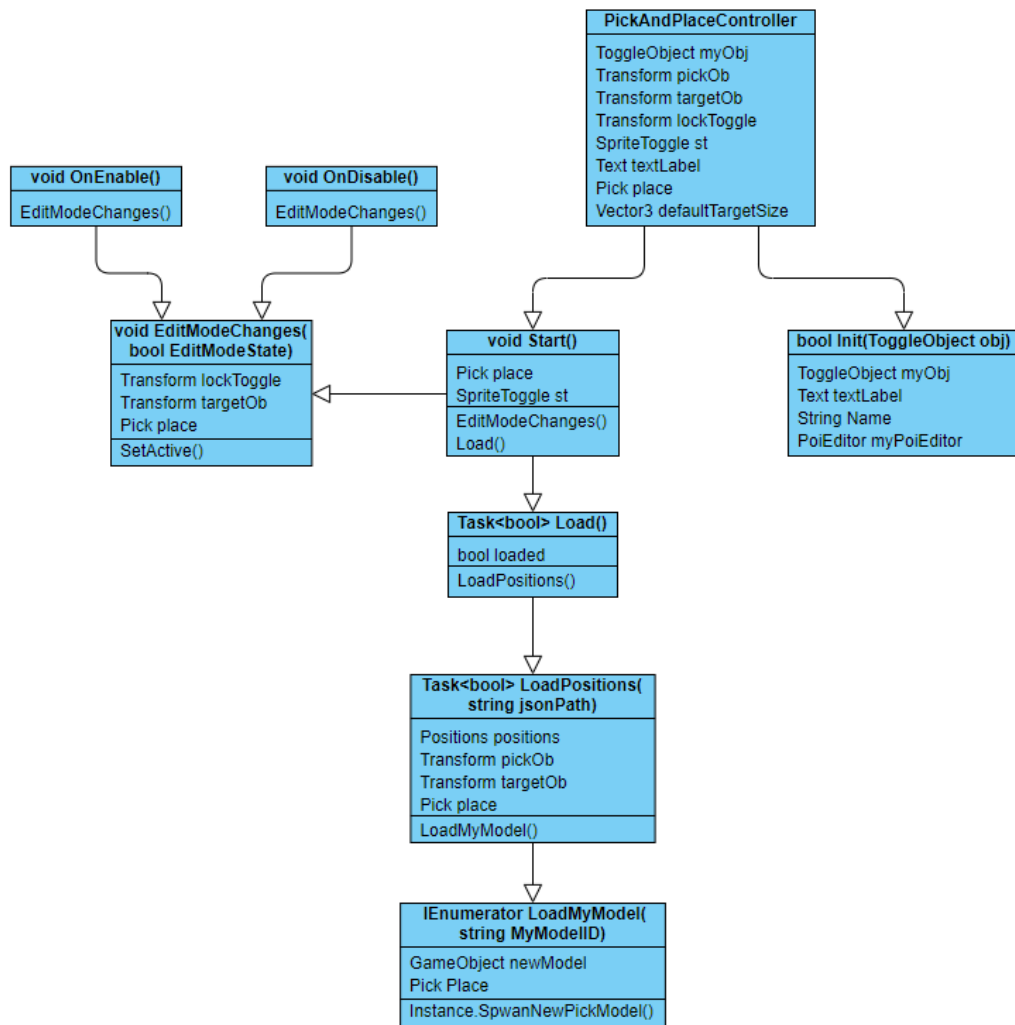
The pick and place augmentation requires a separate JSON file to store additional data that doesn't fit into the ARLEM standard. This data stores the locations and rotations of both the p&p object and target sphere, as well as the lock state and position. This data is represented in the table below;

#	Name	Explanation	Value Space	Datatype (Maximum Length)	Examples
1	Pick	This is the JSON file which store the additional pick and place data.			{ "pickObjectPosition":{ "X":-0.5158615708351135, "Y":0.0887712836265564, "Z":0.8843992352485657},
1a	<i>Pick Object Position</i>	A Vector3 used to store the local position of the pick object which in the current state of this feature is the arrow object	(x,y,z)	Vector 3	"pickObjectRotation":{ "X":-0.20961745083332063, "Y":0.22136011719703675, "Z":0.6743713021278381, "W":0.6725203394889832},
1b	<i>Pick Object Rotation</i>	A Quaternion used to store the local rotation of the pick object which in the current state of this feature is the arrow object	(x,y,z,w)	Quaternion	"targetObjectPosition":{ "X":-0.24698609113693238, "Y":0.028038620948791505, "Z":0.5703811645507813}, "targetObjectScale":{ "X":0.5,"y":0.5,"z":0.5},
1c	<i>Target Object Position</i>	A Vector3 used to store the local position of the target sphere	(x,y,z)	Vector 3	"resetPosition":{"x":- 0.5158615708351135 ,"y":0.0887712836265564 ,"z":0.8843992352485657},
1d	<i>Target Object Scale</i>	Sets the size of the target sphere	(x,y,z)	Vector 3	"moveMode":false, "Reset":false,
1e	<i>Reset Position</i>	stores the position of the pick object reset location	(x,y,z)	Vector 3	"modelID":""}
1f	<i>moveMode</i>	Determines whether or not the the lock toggle is on or off for a given pick objects	"true", "false"	Bool	
1g	<i>reset</i>	Determines whether or not the the pick location	"true", "false"	Bool	



		will be reset after a action step change			
1h	<i>modelID</i>	Stores a 3D model ID if the pick object is changed using sketchfab	Alphanumeric	Character String (1000)	

Class Diagram:



4.11 3D Whiteboard

An experimental 3D whiteboard feature has been added, but is available only on Hololens for authoring, as it requires hand tracking. This augmentation uses the Google TiltBrush Open Source project to implement facilities for users to paint in 3D using their fingers and a

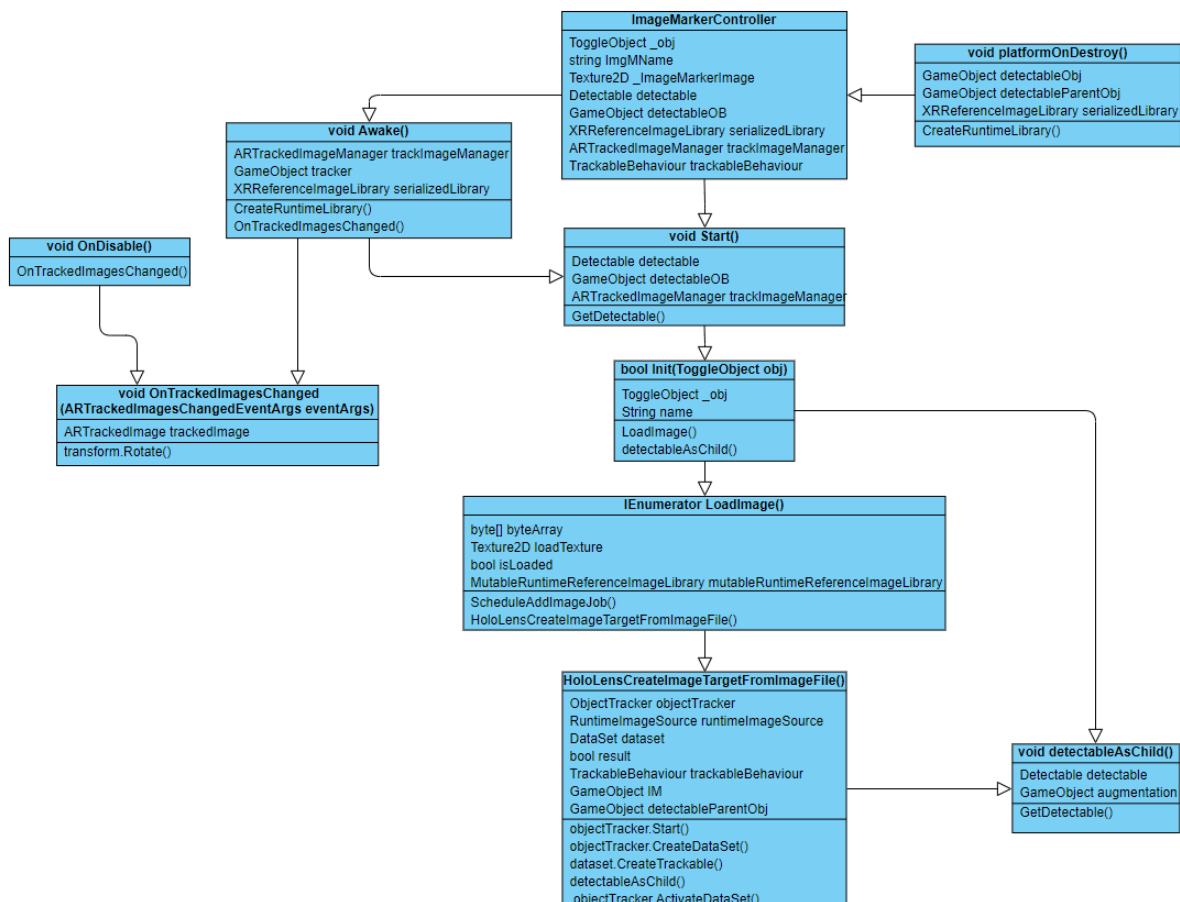


wide palette of brushes, styles, and colours. The data model used to record the animated 3D doodle over time is the TiltBrush data format⁹.

4.12 Image Marker

This augmentation allows a user to take a photograph of a real world object using their device's camera. On hololens builds the image can be cropped in order to better select the image marker target. Once set up the image taken is used to track the real world object. At this point in time the tracked object is used to orientate and position the task station for a given action step. As all other augmentation types that are used in an action step are connected to that step's task station, they will also be re-orientated and re-positioned based on the tracked object relative to the task station. Objects that make good image markers have strong defined features, e.g. bold lines, colour contrast, larger objects.

Class Diagram:



4.13 Plugin

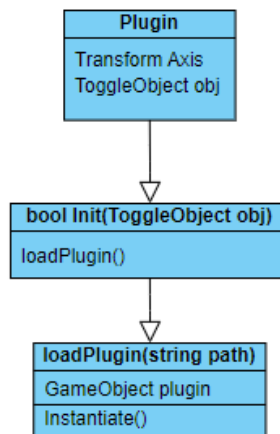
Plugins are different to all other augmentations, they allow for users to add new and project specific augmentations to an activity. At this point in time plugins can be added to a build of

⁹ <https://fileinfo.com/extension/tilt>



mirage via unity and are represented in ARLEM using app. In future the plugin augmentation will be expanded to allow a user to create a new plugin folder to an activities ARLEM zip using the plugin editor in app. Once the new plugin folder has been added, a user will be able to add the relevant game objects and scripts to this folder via either a devices file explorer or the online interface.

Class Diagram:



5. Configuring MirageXR

Developers have to configure a few things before being able to compile, and there are some additional optional settings that are good to know about.

- The system utilises several APIs which require API secrets to be configured. This includes IBM Watson and SketchFab.
- The user interface and app presentation can be skinned via config files and mapping tables.
- Default server URLs can be changed.
- Terms and conditions of use can be specified via a text file.

5.1 API Keys

The development team has API keys that we share in the development. If you have been invited to contribute, you should have received these keys or should have been granted access.

- *Sketchfab API secret*: now in the Unity credentials system, or alternatively to be placed into a file.
- *IBM Watson*: you need to put a non-tracked `ibm-credentials.env` file into the root folder to be able to build the natural language processing capabilities for the character models. You will need to add the API secret key credentials for the



following IBM Watson services: Tone Analyzer, Assistant, Speech-to-text, Text-to-speech, and the language translator (optional) services. The file is listed below, just replace the %KEY% values with the according API secrets for each service provided by the IBM Watson dashboard:

```
TONE_ANALYZER_APIKEY=%KEY%
TONE_ANALYZER_IAM_APIKEY=%KEY%
TONE_ANALYZER_URL=https://api.eu-gb.tone-analyzer.watson.cloud.ibm.com/instances/5f989edc-233e-42a0-98c5-9e1cefcf36c5
TONE_ANALYZER_AUTH_TYPE=iam
ASSISTANT_APIKEY=%KEY%
ASSISTANT_IAM_APIKEY=%KEY%
ASSISTANT_URL=https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/7ca75d0a-966f-4350-91b8-06fb0e882de5
ASSISTANT_AUTH_TYPE=iam
SPEECH_TO_TEXT_APIKEY=%KEY%
SPEECH_TO_TEXT_IAM_APIKEY=%KEY%
SPEECH_TO_TEXT_URL=https://api.eu-gb.speech-to-text.watson.cloud.ibm.com/instances/00f5f658-9d9d-4b76-9505-d07885513919
SPEECH_TO_TEXT_AUTH_TYPE=iam
TEXT_TO_SPEECH_APIKEY=%KEY%
TEXT_TO_SPEECH_IAM_APIKEY=%KEY%
TEXT_TO_SPEECH_URL=https://api.eu-gb.text-to-speech.watson.cloud.ibm.com/instances/a597ad6c-b7a1-479f-a592-4fde6d504b2a
TEXT_TO_SPEECH_AUTH_TYPE=iam
LANGUAGE_TRANSLATOR_APIKEY=%KEY%
LANGUAGE_TRANSLATOR_IAM_APIKEY=%KEY%
LANGUAGE_TRANSLATOR_URL=https://api.eu-gb.language-translator.watson.cloud.ibm.com/instances/a1d20d77-8dae-49db-9aa3-5604e9d07753
LANGUAGE_TRANSLATOR_AUTH_TYPE=iam
```

5.2 Configure Corporate Design and Identity with the BrandManager

MirageXR supports white labelling. This is implemented via the BrandManager class and the BrandConfiguration Unity editor window class, which determine key corporate design elements from settings in a configuration file or via a Unity editor panel. Most notably, this allows to configure: app name, version number, splash screen image, logo image, and a set of primary colours.

These classes allow configuring all key UI elements centrally. The values are stored in Assets/MirageXR/Resources/MirageXRConfig.txt and will be filed in upon start of the app.



```
companyName:MyOrganisation
productName:MyApp
version:$appVersion
splashScreen:Assets/logo/logo.png
logo:Assets/logo/logo_620x620.png
SplashBackgroundColor:#000000FF
primaryColor:#2EC4B6FF
secondaryColor:#FF9F1CFF
textColor:#FFFFFFF8
iconColor:#FFFFFFC8
taskStationColor:#FF9F1CFF
pathColor:#FFFFFFB5
nextPathColor:#00FFE7C8
```

5.3 Assets Mapping Table

Moreover, there are asset mapping tables to be set up which allow reducing resource load in the application. As a side effect, this allows rapid adaptation of the UI on a deep level. To set up this required mapping table for prefab assets (using the addressable methods), Unity will automatically prompt developers to configure ("build") a mapping table for where to load asset prefabs from.

5.4 Default servers

- xAPI: The learning record store URL (LRS URL) is set in MirageXRServiceBootstrapper.cs
- LMS: The URL to Moodle is set

5.5 Terms and conditions

An End User License Agreement is filed in from the file `TermsOfUseDefault.txt`, filed in by the BrandManager class.

6. Repository service: Moodle mod_arete

As the system landscape diagram in Section 3 shows, the apps communicate via web services with the repository system, implemented as a 'module' in Moodle (mod_arete). The module and according web services are documented in deliverable D3.4, Digital Repository. For convenience, the key UML sequence diagram is reproduced here to illuminate how communication between the apps and the repository work.

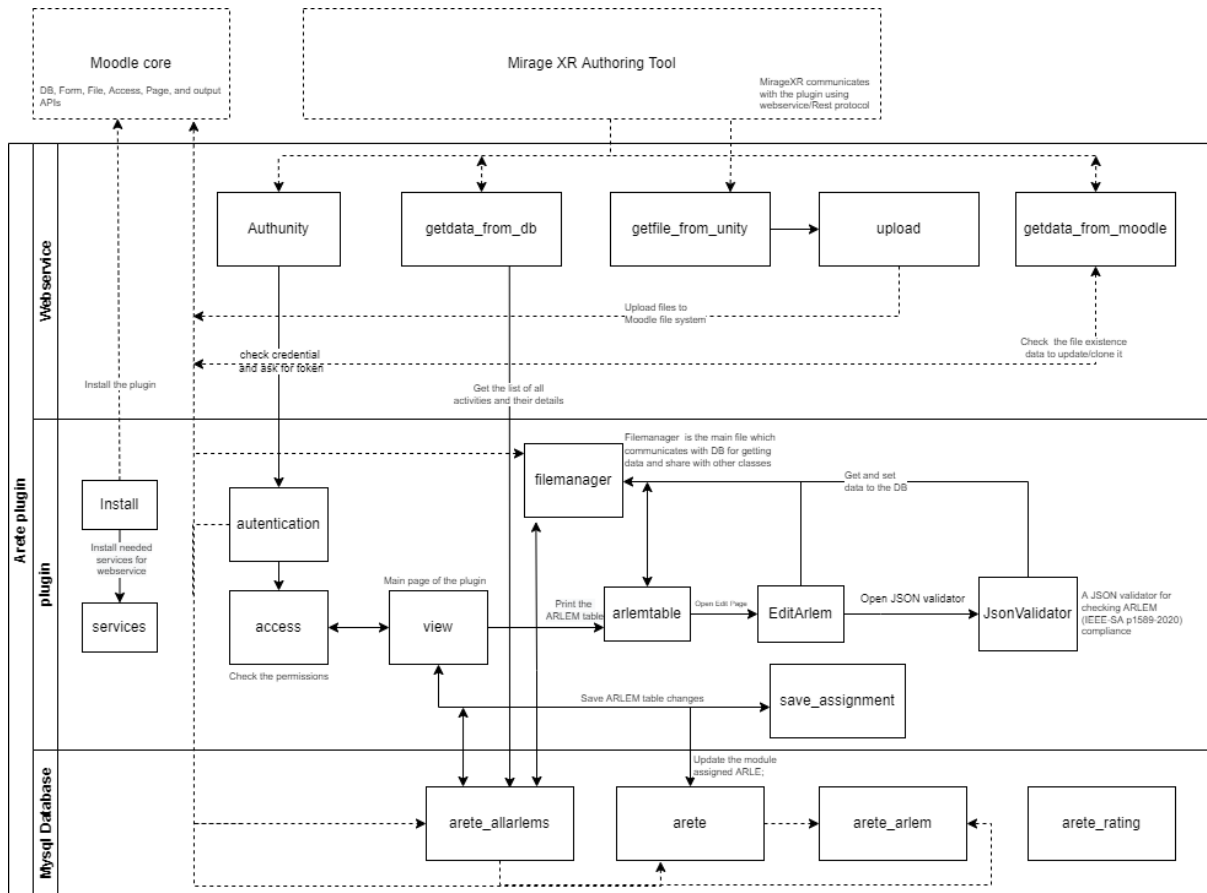


Figure 6.1. Sequence diagram of communication between app and repository.

The repository API methods allow the app to store IEEE P1589-2020 compliant archives that hold the augmented reality learning experience with its data models and all associated media assets.

7. Logging user behaviour with the IEEE Experience API (xAPI)

The repository service for user logs, the so-called learning record store, was set up using the Open Source project LearningLocker. The service is documented in D3.4, Digital Repository, but for convenience, we reproduce below the vocabulary table of our xAPI vocabulary profile for MirageXR. Tracking of user activity was significantly improved over the run-time of the project. The Unity wrapper and ExperienceManager for the xAPI needed to be patched for that, in order to allow including context (activity!) and results (e.g., measures) in the statement payload.



Verb ID	Verb Name	Use in Objects
<i>Primitives</i>		
http://id.tincanapi.com/verb/viewed	viewed	label, image, model
http://activitystrea.ms/schema/1.0/find	found	detect
http://activitystrea.ms/schema/1.0/listen	listened_to	audio, sound
http://activitystrea.ms/schema/1.0/watch	watched	video
<i>Action glyphs</i>		
http://id.tincanapi.com/verb/focused	focused_on	act:Highlight, act:Point
http://activitystrea.ms/schema/1.0/open	opened	act:OpenBox
http://activitystrea.ms/schema/1.0/close	closed	act:CloseBox
https://wekit-community.org/verb/packed	packed	act:Pack
https://wekit-community.org/verb/unpacked	unpacked	act:Unpack
https://wekit-community.org/verb/picked	picked	act:Pick
https://wekit-community.org/verb/placed	placed	act:Place
https://wekit-community.org/verb/screwed	screwed	act:Screw
https://wekit-community.org/verb/unscrewed	unscrewed	act:Unscrew
https://wekit-community.org/verb/rotated	rotated	act:Rotate
https://wekit-community.org/verb/lowered	lowered	act:Lower
https://wekit-community.org/verb/located	located	act:Locate
https://wekit-community.org/verb/lubricated	lubricated	act:Lubricate
https://wekit-community.org/verb/painted	painted	act:Paint
https://wekit-community.org/verb/plugged	plugged	act:Plug
https://wekit-community.org/verb/unplugged	unplugged	act:Unplug
https://wekit-community.org/verb/unfastened	unfastened	act:Unfasten



https://w3id.org/xapi/dod-isd/verbs/measured	measured	act:Measure
<i>Further predicates</i>		
https://wekit-community.org/verb/noticed	noticed	vfx (vfx:*)
https://wekit-community.org/verb/met	met	character (char:*)
<i>Basic activity interaction</i>		
http://adlnet.gov/expapi/verbs/launched	launched	activity - started
http://adlnet.gov/expapi/verbs/initialized	initialized	activity - loaded
http://activitystrea.ms/schema/1.0/complete	completed	activity - completed
http://activitystrea.ms/schema/1.0/start	started	step - activated
http://activitystrea.ms/schema/1.0/experience	experienced	step - deactivated

8. Discussion

The systematic literature review has unveiled that research on authoring tools falls short of supporting higher level “learning objectives”, as Bloom calls them (Bloom et al., 1956). Our analysis shows that authoring tool support for these cognitive process dimensions falls particularly short with regards to ‘analyse’, ‘evaluate’, and ‘synthesise’, but has deficits already regarding ‘apply’.

The work presented above on MirageXR seeks to tackle this shortcoming, innovating ways of supporting not only remembering and understanding, but particularly the practical application of knowledge in guided procedure.

For ‘Apply’, the working principle of MirageXR embeds the more information presentation oriented functionalities typically used to support the dimensions ‘Remember’ and ‘Understand’ with facilities to execute instruction directly in workflows in the real world. In particular, this includes a system of action instruction glyphs, which provide a visual language for handling and movement that can guide the user to conduct work in the physical environment surrounding them. Procedural guidance is used to deliver information in the context where it is needed, providing memorable experiences that build neural pathways for cued recall.

The quiz mode of the Pick & Place augmentation works in a similar fashion and is geared towards applying knowledge in practice.



Furthermore, we have introduced both an analytics system geared towards reflection-on-action (Schön, 1983), as well as a set of action instruction glyphs aimed at supporting analytic processes.

The built in performance analytics (see Section 7) log meticulously the learner's behaviour in the learning or training activity, providing an interface to reflect on and 'Analyse' past action, using the Learning Locker dashboard and data pool. In the future, we intend to interface from within the learning experience with the learning record store, so as to provide enhanced adaptation facilities (reacting to past, logged learning experiences) and so as to provide direct feedback during the learning experience, e.g. in form of a generic performance dashboard at the end of the unit.

The probably best example of the action instruction glyphs for the analytic processes in 'Analyse' is the 'measure' glyph, which guides students to take measures of elements of the real world or the digital overlays, also storing the measured values to the xAPI endpoint. This can be used, for example, to measure the neck width of anthropologic skulls to witness the evolution to bipedalism (walking on two legs), when comparing ancient hominid skulls: when moving to upright, the hole for the spinal cord (the so-called 'foramen magnum') in the skull moves from the back to the centre. Other glyphs like "locate" or "inspect" instruct for specific analytical processes, too.



Bloom's Cognitive Processes	
Dimension	MirageXR support
Remember	Augmentation primitives like audio, image, video, label, 3d model help memorise
Understand	Instruction element and animations. Support for directed focus (e.g. visual effects, highlight glyph, marker augmentation with contextualised augmentations, annotation labels). Plugins for integrating interactive 3D objects.
Apply	System of action instruction glyphs provide a visual language for handling and movement
Analyse	Measure, Inspect, Locate xAP performance analytics for reflection-on-action
Evaluate	Possibility to use experience templates and fill them with inspection results, documenting outcomes, measuring, testing, judging. GhostTrack to record performance.
Create	Learners can create content as MirageXR has integrated editor and player. Produce new experiences, present in XR (instead of PowerPoint).

Table 8.1: MirageXR support for the Bloom's Cognitive Process Dimensions.

Analysis, evaluation, and synthesis require more than just reenactment of predefined learning experiences, but for that reason, we have designed MirageXR to have an integrated authoring mode. Every learner can always edit existing learning experiences¹⁰ or create new ones. These creative processes are supported in other authoring tools typically only with screen recording, storing the three dimensional interactive experience in a static two-dimensional video recording. MirageXR makes a difference here and opens up possibilities for content creation or modification.

For example, regarding 'evaluation', the GhostTrack can be utilised to record learner performance, enabling them to document in 4D (3D over time), what they did when and - using simple image, measure, or video augmentations - to what result. Since learners can fork existing resources and add their own content, this can be part of the instructional support. For example, it is possible to set up 'quality assurance' template learning

¹⁰ Storing a duplicate, of course, if they are not the original author of a resource!



experiences, providing a kind of blueprint learning design for users to fill with content: “now switch to edit mode and take a picture of the underside of the object you just created”; “record a ghost of how you approach this”; “record an audio with your reflection on this”; etc.

Moreover, MirageXR is purposely designed in a way that it encourages learner generated content - instead of creating a 2D powerpoint presentation, they now have the possibility to create a 3D (or even 4D) presentation of the assignment they work on.

9. Conclusion

This deliverable reports on the work done regarding the “Interactive Authoring Toolkit integration with MirageXR”. It provides an overview on the development community we successfully gathered around us, which also includes now numerous project external contributors from across Europe and the world. It reports on system landscape and provides a comprehensive introduction to the augmentation system, key to content authoring and delivery. Our work towards configuration (keyword ‘BrandManager’), content repository, and learning analytics is reported.

This development project will not stop there. Over the upcoming period, there is still one significant further extension planned - towards improved learning design support. At the time of writing, we already know this will include work towards improved built in tutorials and guidance, as well as methodological recommendations for design thinking inspired content production processes. This work will be reported in D3.5, “Learning Design”. The development community is by now rather established and current as well as future sprints are in planning, including refactoring of the p1589-2020 parsers and controllers, collaboration and sharing facilities for collaborative authoring, or a new user interface that is in preparation. Furthermore, we look forward to further feature requests and improvements emerging from the pilots.



References

- Abawi, D. F., Dörner, R., & Grimm, P. (n.d.). A Component-based Authoring Environment for Creating Multimedia-Rich Mixed Reality. 10.
- Abawi, D. F., Reinhold, S., & Dörner, R. (2004). A Toolkit for Authoring Non-linear Storytelling Environments Using Mixed Reality. In S. Göbel, U. Spierling, A. Hoffmann, I. Iurgel, O. Schneider, J. Dechau, & A. Feix (Eds.), *Technologies for Interactive Digital Storytelling and Entertainment* (Vol. 3105, pp. 113–118). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-27797-2_15
- Anderson, L. W., & Krathwohl, D. R. (2001). A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. Longman,.
- Azuma, R. T. (1997). A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, 6(4), 355–385.
- Barone Rodrigues, A., Dias, D. R. C., Martins, V. F., Bressan, P. A., & de Paiva Guimarães, M. (2017). WebAR: A Web-Augmented Reality-Based Authoring Tool with Experience API Support for Educational Applications. In M. Antona & C. Stephanidis (Eds.), *Universal Access in Human–Computer Interaction. Designing Novel Interactions* (Vol. 10278, pp. 118–128). Springer International Publishing. https://doi.org/10.1007/978-3-319-58703-5_9
- Billinghurst, M., Clark, A., & Lee, G. (2015). A survey of augmented reality.
- Blattgerste, J., Renner, P., & Pfeiffer, T. (2019). Authorable augmented reality instructions for assistance and training in work environments. *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia*, 1–11. <https://doi.org/10.1145/3365610.3365646>
- Bloom, B. S.; Engelhart, M. D.; Furst, E. J.; Hill, W. H.; Krathwohl, D. R. (1956). *Taxonomy of educational objectives: The classification of educational goals. Vol. Handbook I: Cognitive domain*. New York: David McKay Company.
- Boboc, R. G., Gîrbacia, F., & Butilă, E. V. (2020). The application of augmented reality in the automotive industry: A systematic literature review. *Applied Sciences*, 10(12), 4259.
- Borisov, A., Sieck, J., Ashikoto, L., Kamenye, G., Mwenyo, J., & Likando, N. (2018). Development of an efficient, cost-reducing content management system for augmented reality applications. *Proceedings of the Second African Conference for Human Computer Interaction: Thriving Communities*, 1–4. <https://doi.org/10.1145/3283458.3283471>
- Bottani, E., & Vignali, G. (2019). Augmented reality technology in the manufacturing industry: A review of the last decade. *IJSE Transactions*, 51(3), 284–310.
- Bower, M. (2008). Affordance analysis–matching learning tasks with learning technologies. *Educational Media International*, 45(1), 3–15.



- Camba, J. D., & Contero, M. (2015). From reality to augmented reality: Rapid strategies for developing marker-based AR content using image capturing and authoring tools. 2015 IEEE Frontiers in Education Conference (FIE), 1–6. <https://doi.org/10.1109/FIE.2015.7344162>
- Chen, Z. (2020). Towards futuristic visual storytelling: Authoring data-driven infographics in augmented reality (p. 991012818569603500) [Ph.D., The Hong Kong University of Science and Technology]. <https://doi.org/10.14711/thesis-991012818569603412>
- Chen, Z., Su, Y., Wang, Y., Wang, Q., Qu, H., & Wu, Y. (2020). MARVisT: Authoring Glyph-Based Visualization in Mobile Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics*, 26(8), 2645–2658. <https://doi.org/10.1109/TVCG.2019.2892415>
- Chidambaram, S., Huang, H., He, F., Qian, X., Villanueva, A. M., Redick, T. S., Stuerzlinger, W., & Ramani, K. (2021). ProcessAR: An augmented reality-based tool to create in-situ procedural 2D/3D AR Instructions. *Designing Interactive Systems Conference 2021*, 234–249. <https://doi.org/10.1145/3461778.3462126>
- Dalgarno, B., & Lee, M. J. (2010). What are the learning affordances of 3-D virtual environments? *British Journal of Educational Technology*, 41(1), 10–32.
- Dankov, S., Rzepka, R., & Araki, K. (2011). UIAR Common Sense: An Augmented Reality Framework for Creating Games to Collect Common Sense from Users. *Procedia - Social and Behavioral Sciences*, 27, 274–280. <https://doi.org/10.1016/j.sbspro.2011.10.608>
- de Smit, J. (n.d.). *Layar Creator: Self-service WYSIWYG AR authoring for print*. 3.
- de Sousa Moura, G. (2010). *An authoring tool of photo-realistic virtual objects for augmented reality* [Master's Thesis]. Universidade Federal de Pernambuco.
- de Souza Cardoso, L. F., Mariano, F. C. M. Q., & Zorzal, E. R. (2020). A survey of industrial augmented reality. *Computers & Industrial Engineering*, 139, 106159.
- Dekker, J. (2012). *Authoring 3D virtual objects with tracking based input in augmented reality on mobile Devices* [Master's Thesis]. Utrecht University.
- Drew, M. R., Falcone, B., & Baccus, W. L. (2018). What does the system usability scale (SUS) measure? *International Conference of Design, User Experience, and Usability*, 356–366.
- Dünser, A., Walker, L., Horner, H., & Bentall, D. (2012). Creating interactive physics education books with augmented reality. *Proceedings of the 24th Australian Computer-Human Interaction Conference on - OzCHI '12*, 107–114. <https://doi.org/10.1145/2414536.2414554>
- Egger, J., & Masood, T. (2020). Augmented reality in support of intelligent manufacturing—a systematic literature review. *Computers & Industrial Engineering*, 140, 106195.



- Fei, G., Li, X., & Fei, R. (2012). AVATAR: Autonomous visual authoring of tangible augmented reality. Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry - VRCAI '12, 21. <https://doi.org/10.1145/2407516.2407521>
- Fisher, S. S. (2003). An Authoring Toolkit for Mixed Reality Experiences. In R. Nakatsu & J. Hoshino (Eds.), *Entertainment Computing* (Vol. 112, pp. 487–494). Springer US. https://doi.org/10.1007/978-0-387-35660-0_59
- Fraga-Lamas, P., Fernandez-Carames, T. M., Blanco-Novoa, O., & Vilar-Montesinos, M. A. (2018). A review on industrial augmented reality systems for the industry 4.0 shipyard. *Ieee Access*, 6, 13358–13375.
- Garzón, J., Pavón, J., & Baldiris, S. (2019). Systematic review and meta-analysis of augmented reality in educational settings. *Virtual Reality*, 23(4), 447–459.
- Georgiou, Y., & Kyza, E. A. (2018). Relations between student motivation, immersion and learning outcomes in location-based augmented reality settings. *Computers in Human Behavior*, 89, 173–181.
- Gibson, J. J. (1979). *The theory of affordances. The ecological approach to visual perception*. Boston: Houghton Mifflin.
- Gimeno, J., Morillo, P., Orduna, J. M., & Fernandez, M. (n.d.). An Advanced Authoring Tool for Augmented Reality Applications in Industry. 6.
- Guyen, S., & Feiner, S. (2003). Authoring 3D hypermedia for wearable augmented and virtual reality. *Seventh IEEE International Symposium on Wearable Computers, 2003. Proceedings.*, 118–126. <https://doi.org/10.1109/ISWC.2003.1241401>
- Haller, M., Stauder, E., & Zauner, J. (n.d.). AMIRE-ES: Authoring Mixed Reality once, run it anywhere. 9.
- Han Kyu Yoo, & Lee, J. W. (2014). Mobile augmented reality system for in-situ 3D modeling and authoring. *2014 International Conference on Big Data and Smart Computing (BIGCOMP)*, 282–285. <https://doi.org/10.1109/BIGCOMP.2014.6741453>
- Ibáñez, M.-B., & Delgado-Kloos, C. (2018). Augmented reality for STEM learning: A systematic review. *Computers & Education*, 123, 109–123.
- IS-EUD 2017, K., Vassilis-Javed, Soute, I., De Angeli, A., Piccinno, A., Bellucci, A., & Technische Universiteit (Eindhoven) (Eds.). (2017). *IS-EUD 2017: 6th International Symposium on End-User Development : extended abstracts : work-in-progress, Workshop and doctoral consortium : Eindhoven University of Technology, The Netherlands, June 13-15, 2017*. Technische Universiteit Eindhoven.



- Janer, J., Roma, G., & Kersten, S. (n.d.). Authoring augmented soundscapes with user-contributed content. 4.
- Jee, H.-K., Lim, S., Youn, J., & Lee, J. (2014). An augmented reality-based authoring tool for E-learning applications. *Multimedia Tools and Applications*, 68(2), 225–235. <https://doi.org/10.1007/s11042-011-0880-4>
- Jeon, J., Hong, M., Yi, M., Chun, J., Kim, J. S., & Choi, Y.-J. (2016). Interactive Authoring Tool for Mobile Augmented Reality Content. *Journal of Information Processing Systems*, 12(4), 612–630. <https://doi.org/10.3745/JIPS.02.0048>
- Jin, Q., Wang, D., & Sun, F. (2018). TanCreator: A Tangible Tool for Children to Create Augmented Reality Games. *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, 82–85. <https://doi.org/10.1145/3267305.3267603>
- Kampa, A., & Spierling, U. (2017). Smart Authoring for Location-based Augmented Reality Storytelling Applications. https://doi.org/10.18420/IN2017_93
- Kegeleers, M., & Bidarra, R. (2021). Story Authoring in Augmented Reality: Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 55–66. <https://doi.org/10.5220/0010249800550066>
- Kim, B. (n.d.). Anchor-frame-based Efficient Authoring for Mobile Augmented Reality. 47.
- Kim, B. J., & Park, S. H. (2015). CARDA: Content Management Systems for Augmented Reality with Dynamic Annotation. 62–67. <https://doi.org/10.14257/astl.2015.90.14>
- Kirkley, S. E., & Kirkley, J. R. (2005). Creating next generation blended learning environments using mixed reality, Video Games and Simulations. *TechTrends*, 49(3), 42–53. <https://doi.org/10.1007/BF02763646>
- Kirner, C., & Santin, R. (n.d.). Interaction, Collaboration and Authoring in Augmented Reality Environments. 11.
- Kirschner, P. A. (2002). Can we support CCSL? Educational, social and technological affordances.
- Klopfer, E., & Sheldon, J. (2010). Augmenting your own reality: Student authoring of science-based augmented reality games. *New Directions for Youth Development*, 2010(128), 85–94. <https://doi.org/10.1002/yd.378>
- Langlotz, T., Mooslechner, S., Zollmann, S., Degendorfer, C., Reitmayr, G., & Schmalstieg, D. (2012). Sketching up the world: In situ authoring for mobile Augmented Reality. *Personal and Ubiquitous Computing*, 16(6), 623–630. <https://doi.org/10.1007/s00779-011-0430-0>



- Lee, G. A., & Kim, G. J. (2009). Immersive authoring of Tangible Augmented Reality content: A user study. *Journal of Visual Languages & Computing*, 20(2), 61–79. <https://doi.org/10.1016/j.jvlc.2008.07.001>
- Lee, G. A., Nelles, C., Billinghamurst, M., & Kim, G. J. (2004). Immersive Authoring of Tangible Augmented Reality Applications. *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, 172–181. <https://doi.org/10.1109/ISMAR.2004.34>
- Limbu, B. H., Jarodzka, H., Klemke, R., & Specht, M. (2018). Using sensors and augmented reality to train apprentices using recorded expert performance: A systematic literature review. *Educational Research Review*, 25, 1–22.
- Limbu, B., Vovk, A., Jarodzka, H., Klemke, R., Wild, F., & Specht, M. (2019). WEKIT. one: A sensor-based augmented reality system for experience capture and re-enactment. *European Conference on Technology Enhanced Learning*, 158–171.
- Lorenz, M., Knopp, S., Kim, J., & Klimant, P. (2020). Industrial Augmented Reality: 3D-Content Editor for Augmented Reality Maintenance Worker Support System. *2020 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, 203–205. <https://doi.org/10.1109/ISMAR-Adjunct51615.2020.00060>
- Mader, S., & Urban, B. (n.d.). Creating Instructional Content for Augmented Reality based on Controlled Natural Language Concepts. 5.
- Maia, L. F., Nolêto, C., Lima, M., Ferreira, C., Marinho, C., Viana, W., & Trinta, F. (2017). LAGARTO: A LocAtion based Games AuthoRing TOol enhanced with augmented reality features. *Entertainment Computing*, 22, 3–13. <https://doi.org/10.1016/j.entcom.2017.05.001>
- Mcclean, E., Gales, G., & Mcdonald, J. (n.d.). A fac, ade based approach to non-expert authoring in urban augmented reality. 4.
- Mesarosova, A., Hernandez, M. F., Mesaros, P., & Behun, M. (2017). Mixing augmented reality and EEG technology to create an unique learning tool for construction process. *2017 15th International Conference on Emerging ELearning Technologies and Applications (ICETA)*, 1–7. <https://doi.org/10.1109/ICETA.2017.8102505>
- Milgram, P., Takemura, H., Utsumi, A., & Kishino, F. (1995). Augmented reality: A class of displays on the reality-virtuality continuum. *Telemanipulator and Telepresence Technologies*, 2351, 282–292.
- Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., Altman, D., Antes, G., Atkins, D., Barbour, V., Barrowman, N., Berlin, J. A., & others. (2009). Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement (Chinese edition). *Journal of Chinese Integrative Medicine*, 7(9), 889–896.



Navab, N., Traub, J., Sielhorst, T., Feuerstein, M., & Bichlmeier, C. (2007). Action- and Workflow-Driven Augmented Reality for Computer-Aided Medical Procedures. *IEEE Computer Graphics and Applications*, 27(5), 10–14. <https://doi.org/10.1109/MCG.2007.117>

Nebeling, M., & Speicher, M. (2018). The Trouble with Augmented Reality/Virtual Reality Authoring Tools. 2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), 333–337. <https://doi.org/10.1109/ISMAR-Adjunct.2018.00098>

Nguyen, T. V., Mirza, B., Tan, D., & Sepulveda, J. (2018). ASMIM: Augmented Reality Authoring System for Mobile Interactive Manuals. *Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication*, 1–6. <https://doi.org/10.1145/3164541.3164592>

Nguyen, V. T., Jung, K., & Dang, T. (2020). BlocklyAR: A Visual Programming Interface for Creating Augmented Reality Experiences. *Electronics*, 9(8), 1205. <https://doi.org/10.3390/electronics9081205>

Noletto, C., Lima, M., Maia, L. F., Viana, W., & Trinta, F. (2015). An Authoring Tool for Location-Based Mobile Games with Augmented Reality Features. 2015 14th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), 99–108. <https://doi.org/10.1109/SBGames.2015.12>

Palmarini, R., Erkoyuncu, J. A., Roy, R., & Torabmostaedi, H. (2018). A systematic review of augmented reality applications in maintenance. *Robotics and Computer-Integrated Manufacturing*, 49, 215–228.

Parmaxi, A., & Demetriou, A. A. (2020). Augmented reality in language learning: A state-of-the-art review of 2014–2019. *Journal of Computer Assisted Learning*, 36(6), 861–875.

Pellas, N., Fotaris, P., Kazanidis, I., & Wells, D. (2019). Augmenting the learning experience in primary and secondary school education: A systematic review of recent trends in augmented reality game-based learning. *Virtual Reality*, 23(4), 329–346.

Poupyrev, I., Tan, D., Billinghamurst, M., Kato, H., Regenbrecht, H., & Tetsutani, N. (n.d.). Tiles: A Mixed Reality Authoring Interface. 8.

Raso, R., Werth, D., & Loos, P. (2017). Enriching Augmented Reality with Text Data Mining: An Automated Content Management System to Develop Hybrid Media Applications. *Hawaii International Conference on System Sciences*. <https://doi.org/10.24251/HICSS.2017.746>

Reynolds, B. F. (n.d.). An Augmented Reality Editor: Building data-focused tools to extend the capability, connectivity, and usability of a mobile Internet of Things browser. 111.

Rodrigo, M. M. T., Taketomi, T., Yamamoto, G., Sandora, C., & Kato, H. (2014). Authoring Augmented Reality as Situated Multimedia. *Proceedings of the 22nd International Conference on Computers in Education*. Japan: Asia-Pacific Society for Computers in



Education, 3. <http://penoy.admu.edu.ph/~alls/wp-content/uploads/2014/12/Santos-et-al-2014-poster.pdf>

Romano, M., Díaz, P., & Aedo, I. (2020). Empowering teachers to create augmented reality experiences: The effects on the educational experience. *Interactive Learning Environments*, 1–18. <https://doi.org/10.1080/10494820.2020.1851727>

Rumiński, D., & Walczak, K. (2013). Creation of Interactive AR Content on Mobile Devices. In W. Abramowicz (Ed.), *Business Information Systems Workshops* (Vol. 160, pp. 258–269). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-41687-3_24

Safi, M., Chung, J., & Pradhan, P. (2019). Review of augmented reality in aerospace industry. *Aircraft Engineering and Aerospace Technology*.

Sano, A. (2012). A Web Application for Creating Real-Size Augmented Reality Content without 3D Modeling Skills. *The Journal of Information and Systems in Education*, 10(1), 32–38. <https://doi.org/10.12937/ejsise.10.32>

Sano, A. (2011). An application for creating full-scale augmented reality content without 3d modeling skills. *Proceedings of the 2011 ACM Symposium on The Role of Design in UbiComp Research & Practice - RDURP '11*, 19. <https://doi.org/10.1145/2030031.2030039>

Santos, Marc Ericson & Santos, C & Ty, Jayzon & In, Arno & Luebke, Wolde & Mercedes, Ma & Rodrigo, T & Taketomi, Takafumi & Yamamoto, Goshiro & Sandor, Christian & Kato, Hirokazu. (2014). Authoring Augmented Reality as Situated Multimedia. *Proceedings of the 22nd International Conference on Computers in Education, ICCE 2014*.

Schön, D. A. (1983). *The reflective practitioner: how professionals think in action*. New York: Basic Books

Schlueter, J. A. (2018). Remote maintenance assistance using real-time augmented reality authoring (p. 12331520) [Master of Science, Iowa State University, Digital Repository]. <https://doi.org/10.31274/etd-180810-6087>

Shekhar, S., Maheshwari, P., J, M., Singhal, A., Singh, K. K., & Krishna, K. (2019). ARComposer: Authoring Augmented Reality Experiences through Text. *The Adjunct Publication of the 32nd Annual ACM Symposium on User Interface Software and Technology*, 63–65. <https://doi.org/10.1145/3332167.3357116>

Shim, J., Yang, Y., Kang, N., Seo, J., & Han, T.-D. (2016). Gesture-based interactive augmented reality content authoring system using HMD. *Virtual Reality*, 20(1), 57–69. <https://doi.org/10.1007/s10055-016-0282-z>

Silva, M. M. O. da, Teichrieb, V., & Smith, P. (2019). What are Teachers Needs Concerning Augmented Reality Digital Authoring Tools? *Anais Dos Workshops Do VIII Congresso*



Brasileiro de Informática Na Educação (CBIE 2019), 1452.
<https://doi.org/10.5753/cbie.wcbie.2019.1452>

Speiginer, G., MacIntyre, B., Bolter, J., Rouzati, H., Lambeth, A., Levy, L., Baird, L., Gandy, M., Sanders, M., Davidson, B., Engberg, M., Clark, R., & Mynatt, E. (2015). The Evolution of the Argon Web Framework Through Its Use Creating Cultural Heritage and Community-Based Augmented Reality Applications. In M. Kurosu (Ed.), *Human-Computer Interaction: Users and Contexts* (Vol. 9171, pp. 112–124). Springer International Publishing.
https://doi.org/10.1007/978-3-319-21006-3_12

Tang, K. S., Cheng, D. L., Mi, E., & Greenberg, P. B. (2020). Augmented reality in medical education: A systematic review. *Canadian Medical Education Journal*, 11(1), e81.

Ty, J. (n.d.). A MOBILE AUTHORING TOOL FOR AUGMENTED REALITY CONTENT GENERATION USING IMAGES AS ANNOTATIONS. 10.

Villanueva, A., Zhu, Z., Liu, Z., Peppler, K., Redick, T., & Ramani, K. (2020). Meta-AR-App: An Authoring Platform for Collaborative Augmented Reality in STEM Classrooms. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 1–14.
<https://doi.org/10.1145/3313831.3376146>

Wang, M.-J., Tseng, C.-H., & Shen, C.-Y. (2010). An Easy to Use Augmented Reality Authoring Tool for Use in Examination Purpose. In P. Forbrig, F. Paternó, & A. Mark Pejtersen (Eds.), *Human-Computer Interaction* (Vol. 332, pp. 285–288). Springer Berlin Heidelberg.
https://doi.org/10.1007/978-3-642-15231-3_31

Wang, T., Qian, X., He, F., Hu, X., Huo, K., Cao, Y., & Ramani, K. (2020). CAPturAR: An Augmented Reality Tool for Authoring Human-Involved Context-Aware Applications. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, 328–341. <https://doi.org/10.1145/3379337.3415815>

Wang, Z., Nguyen, C., Asente, P., & Dorsey, J. (2021). DistanciAR: Authoring Site-Specific Augmented Reality Experiences for Remote Environments. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 1–12.
<https://doi.org/10.1145/3411764.3445552>

Yang, Y., Shim, J., Chae, S., & Han, T.-D. (2016). Mobile Augmented Reality Authoring Tool. *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, 358–361.
<https://doi.org/10.1109/ICSC.2016.42>

Yang, Y., Shim, J., Seungho Chae, & Han, T.-D. (2016). Interactive Augmented Reality Authoring System using mobile device as input method. *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 001429–001432.
<https://doi.org/10.1109/SMC.2016.7844437>



Zauner, J., & Haller, M. (n.d.). Authoring of Mixed Reality Applications including Multi-Marker Calibration for Mobile Devices. 9.

Zauner, J., Haller, M., Brandl, A., & Hartman, W. (2003). Authoring of a mixed reality assembly instructor for hierarchical structures. The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings., 237–246.
<https://doi.org/10.1109/ISMAR.2003.1240707>

Zhu, J., Ong, S. K., & Nee, A. Y. C. (2012). An authorable context-aware augmented reality system to assist the maintenance technicians. The International Journal of Advanced Manufacturing Technology. <https://doi.org/10.1007/s00170-012-4451-2>



Annex A: Orkestra Library

The Orkestra Library, described in section 7.1 of Deliverable D3.7, is a library that allows the creation of multi-device and multi-user applications and enables communication and data-sharing between different users. Common applications of the library include sharing data between users, sending peer 2 peer or broadcast messages, provide timing and synchronization mechanisms, distribute content across multiple devices using several layouts, exchanging multimedia data using the WebRTC protocol.

As mentioned in D3.7, the objective of porting Orkestra from a web-based solution to C# was motivated by the fact that the AR applications developed for the three pilots are built using the Unity environment and thus a purely web solution could not be easily integrated in their codebase. In this appendix we will briefly present the updates related to the C# port of the Orkestra library as well as the proofs of concept developed to demonstrate its functionalities.

Code refactoring

The library code has been completely refactored and simplified. Thanks to the changes in the code, we were able to fix several bugs which affected the library performance, and we also reduced the amount of messages sent by more than 50%. This allows a significant increase in the maximum number of concurrent users without overwhelming the Orkestra server.

Other changes to the codebase regard how the code is structured: by using the Bridge and Façade design patterns the codebase is now decoupled from the external libraries used for Socket communication (WebSocketIOClient¹¹) and Json management (Newtonsoft¹²). This is especially relevant, since these libraries are not available for all platforms (for example, the Microsoft HoloLens SDK). By decoupling the core of the library from them, it will then be easy in the future to add new classes that work with other libraries, then increase the multiplatform functionalities of the library. This is also the reason why Orkestra has not been fully integrated into MirageXR since it requires full compatibility with HoloLens devices. Orkestra has been tested so far on desktop machines (with Ubuntu, macOS and Windows operating system), Android and iOS devices (on both mobile phones and tablets).

To simplify development, the library also includes a test suite for both Edit mode and Play mode (as shown in Figure 1) as well as a Data Viewer that allows checking at runtime the status of each client as well as the data shared between clients (as shown in Figure 2).

The library has been exported to a Unity package that can be directly included in any existing project, but now it can also run as a standalone application. This way, the developers can quickly check that the message passing and the data sharing functionalities work as expected.

¹¹ <https://github.com/doghappy/socket.io-client-csharp>

¹² <https://www.newtonsoft.com/json>



The standalone application was also used to check the compatibility between web and Unity implementation, and now web applications can exchange data to and from applications built in Unity. A typical use case for this is letting a teacher manage tests or quizzes from a webpage, sending questions to all or only some students using the AR applications from their device, and receiving the answers in forms that can later be automatically graded.

Finally, we also added the possibility to automatically manage the number of users who can share the data. This is especially relevant in classroom scenarios, where the students usually work in small groups (4-6 people) but where the whole classroom is using the same AR application at the same time. In this case, from within Orkestra the developer can specify a “main room” as well as the number of concurrent users for each “secondary room”. When students start the application, they will join the main room and will be assigned a secondary room. As soon as a secondary room reaches the maximum number of users, the server will create another one, and new students joining will be assigned to it. Standard users (the students) will only be able to receive and send data to other users in the same secondary room, while privileged users (typically the teachers) are able to send/receive data to all the clients in the main room. This approach is scalable, and the hierarchy of rooms could also be expanded to more levels, if necessary.

Open-sourcing Orkestra Code

Due to internal decisions taken in Vicomtech all the code for the Orkestra client (both the Javascript as well as the C# implementation) will be released as GPLv3. Additionally, the software was registered with the Basque Intellectual Property Agency at the end of September 2021. A final decision on whether the code for the Orkestra server will be made public has not been taken yet. Vicomtech will anyway provide access to the Orkestra server deployed on AWS to all the consortium partners for the duration of the project.

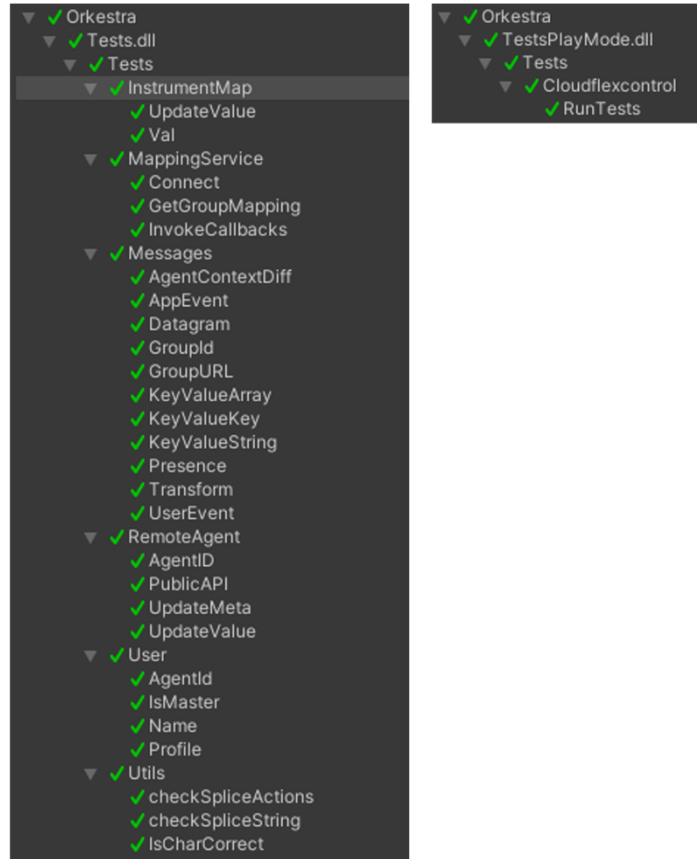


Figure Annex A.1: Orkestra Unity Test Suite

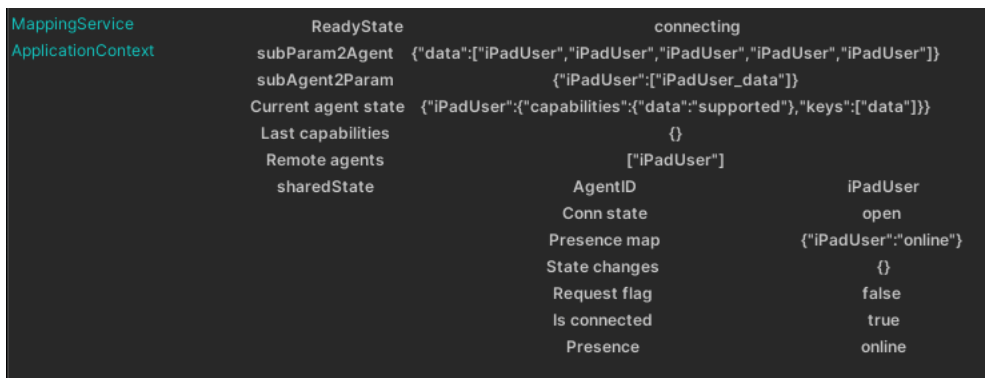


Figure Annex A.2: Orkestra Data Viewer

Proofs of concepts

Together with the library, we have developed three applications with the aim to demonstrate how to use Orkestra and what could be achieved with it.



Geography Demo

The first proof of concept is a geography AR app showing a 3D Earth that the users can see once the application detects the planes available to place the Earth. Through Orkestra, the users can see the position of the other users (thus letting other people know what part of the globe each user is watching) as well as the interactions of the users (where the user is clicking etc.). Additionally, through a web interface a user (ideally a teacher) can send text information to one or more users that will appear as augmented content in the AR app, as well as change the appearance of the globe.

Music Demo

The second proof of concept is another AR application which aims to test Orkestra's ability to exchange multimedia data such as audio tracks, as well as checking its synchronization capabilities. In this case, the app works with several markers, each one associated with a 3d model of a musical instrument and an associated audio track. When a marker is in the field of view of the mobile device, the corresponding 3D content is shown, and the associated audio track starts playing. A demo showing how 2 users can use the application is available at this [video](#).

VR Demo

The last proof of concept is a VR application where a user can open a web page and see the virtual world from the point of view of another user checking data on an Oculus Quest 2 (see Figure 3 or check the [video](#)). In this case, a user is browsing data in a VR environment where a 3D plot shows some images (the detections from a person and object detection neural network on some videos). The three axes represent the time in the video, the type of objects detected and the video, respectively. Through Orkestra, the users share the point of view of the Oculus user, as well as its interactions with the plot. Additionally, when the Oculus user starts playing one of the videos, the multimedia content is sent, synchronized, to the web user as well. Videos are sent through webRTC using a Janus gateway for publishing the multimedia content.

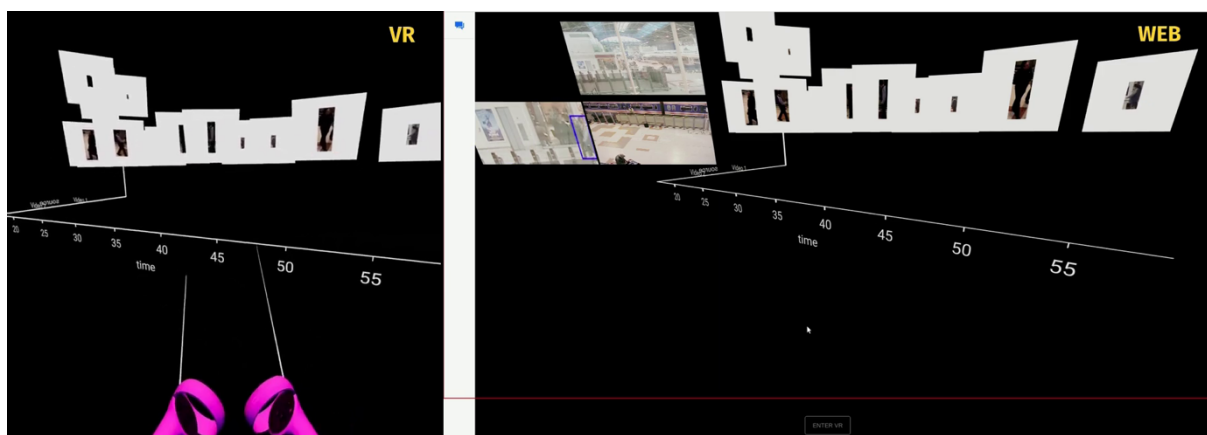


Figure Annex A.3: Usage of Orkestra in a VR environment. Through Orkestra, a client checking the webapp can watch how another user is browsing data in 3 dimensions, sharing its point of view as well as the images and videos he is highlighting