IEEE SA
STANDARDS
ASSOCIATION

IEEE Standard for Augmented Reality Learning Experience Model

STANDARDS

IEEE Computer Society

Developed by the
Learning Technologies Standards Committee

IEEE Std 1589™-2020

IEEE

IEEE Standard for Augmented Reality Learning Experience Model

Developed by the

Learning Technologies Standards Committee
of the
IEEE Computer Society

Approved 30 January 2020

IEEE SA Standards Board

Abstract: Augmented reality (AR) promises to provide significant boosts in operational efficiency by making information available to employees needing task support in context in real time. To support implementations of AR training systems, in this document, an overarching integrated conceptual model is proposed to describe interactions between the physical world, the user, and digital information, the context for AR-assisted learning and other parameters of the environment. Two data models are defined, as well as their binding to XML and JSON for representing learning activities (also known as “employee tasks and procedures”) and the learning environment in which these tasks are performed (also known as the “workplace”). The interoperability specification and standard is presented in support of an open market where interchangeable component products provide alternatives to monolithic AR-assisted learning systems. Moreover, the creation of experience repositories and online marketplaces for AR-enabled learning content is facilitated. Specific attention is given to reuse and repurposing of existing learning content, as well as to catering to “mixed” experiences combining real-world learner guidance with the consumption (or production) of traditional contents such as instructional video material or learning apps and widgets.

Keywords: augmented reality, e-learning, IEEE 1589™, immersive learning environment, learning activity, learning experience, performance support, mixed reality, workplace training

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF: ISBN 978-1-5044-6448-2 STD24056
Print: ISBN 978-1-5044-6449-9 STDPD24056

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notices and Disclaimers Concerning IEEE Standards Documents." They can also be obtained on request from IEEE or viewed at http://standards.ieee.org/ipr/disclaimers.html.

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association ("IEEE SA") Standards Board. IEEE ("the Institute") develops its standards through a consensus development process, approved by the American National Standards Institute ("ANSI"), which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed through scientific, academic, and industry-based technical working groups. Volunteers in IEEE working groups are not necessarily members of the Institute and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied "AS IS" and "WITH ALL FAULTS."

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854 USA

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and

adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit IEEE Xplore at http://ieeexplore.ieee.org/ or contact IEEE at the address listed previously. For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website at http://standards.ieee.org.

Errata

Errata, if any, for IEEE standards can be accessed via https://standards.ieee.org/standard/index.html. Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in IEEE Xplore: https://ieeexplore.ieee.org/browse/standards/collection/ieee/. Users are encouraged to periodically check for errata.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at https://standards.ieee.org/about/sasb/patcom/patents.html. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this IEEE standard was completed, the P1589 Working Group had the following membership:

Fridolin Wild, *Chair*
Christine Perey, *Vice Chair*
Brandt Dargue, *Secretary*

Edilson Arenas
Avron Barr
Jim Belanich
Stefano Bianchi
Judy Bloxham
Ioannis Brilakis
Christopher Brown
Paolo Civardi
John Cook
Ingo Dahn
Jayfus Doswell
Matei Dusik
Pavel Dvořák
Ashley Fehr
Mikhail Fominykh
Mark Grant
Will Guest
Jason Haag
Kaj Helin
Alex Hill
Joseph Howse
Erik Isaksson
Geoff Jarvis
Keith Kamikawa
Jaakko Karjalainen
Gerry Kim
Ralf Klamma
Roland Klemke
Douglas Knisely
Anne Knowles
Alexander Komar
Paul Lefrere
Robert McCaffrey
Johanna Pirker
Barry Po
Mikel Salazar
Giuseppe Scavo
Patrick Selby
Carl H. Smith
Maximilian Speicher
Daniel Spikol
Joe Tojek
Alla Vovk
Pete Wassell
Craig Wiggins
Louise Yarnall

The following members of the individual Standards Association balloting group voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Robert Aiello
Bakul Banerjee
Avron Barr
Mitchell Bonnett
Juan Carreon
Keith Chow
S. Claassen
David Fuschi
Randall C. Groves
Marsali Hancock
Werner Hoelzl
Maria Blanca Ibanez
Moriyuki Ikeuchi
Atsushi Ito
Piotr Karocki
Thomas Kurihara
Mikhail Lagoda
Haobo Lai
Charles Ngethe
Paul Nikolich
Christine Perey
Annette Reilly
Tyde Richards
Robert Robinson
Robby Robson
Terence Rout
Hamadou Saliah-Hassane
James Schoening
Carl Singer
Thomas Starai
Walter Struppler
Gerald Stueve
Paul Work
Weizhi Yao
Oren Yuen
Daidi Zhong

When the IEEE SA Standards Board approved this standard on 30 January 2020, it had the following membership:

Gary Hoffman, *Chair*
Vacant Position, *Vice Chair*
Jean-Philippe Faure, *Past Chair*
Konstantinos Karachalios, *Secretary*

Ted Burse
J. Travis Griffith
Grace Gu
Guido R. Hiertz
Joseph L. Koepfinger*
John D. Kulick
David J. Law
Howard Li
Dong Liu
Kevin Lu
Paul Nikolich
Damir Novosel
Jon Walter Rosdahl
Dorothy Stanley
Mehmet Ulema
Lei Wang
Sha Wei
Philip B. Winston
Daidi Zhong
Jingyi Zhou

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1589-2020, IEEE Standard for Augmented Reality Learning Experience Model.

The next generation of performance support systems and tools for learning at the workplace is likely to be delivered in augmented reality (AR). While there are many impressive prototypes and bespoke applications, interoperability has been neglected so far and standards for creating and converting urgently needed AR training content at scale are missing. To address this gap, the IEEE Computer Society provides this standard for an AR Learning Experience Model (ARLEM) as a remedy, elaborated by Working Group P1589.

The standard provides a data format for the enrichment and exchange of AR learning content, consisting at its core of an activity description language and a workplace model.

It is built for describing AR learning experiences such as bringing a handbook to life or hands-free operator training with smart glasses. For example, astronauts can be trained using an AR learning experience while on the ground or when in space, practicing how to perform an assembly procedure of a temporary stowage rack. In this case, the AR training system executes the learning activity represented, using the activity description language set out in this standard. To provide procedural guidance live and in context, the attention of the astronauts in training can be directed to relevant parts of the rack and wall mount by overlaying visual instruction and 3D animations, explaining step by step what needs to be done.

Another example can be found in aeronautics, where service technicians in training build up expertise in how to rig an aircraft engine. For this, the AR training system provides activity guidance on how to handle and adjust the various components associated with the control system to optimize engine performance.

To make yet another example, doctors in training can practice the diagnostic procedure for detecting pulmonary embolism, being guided and receiving explanatory support for different conditions and the way these conditions manifest with different imaging technologies using an AR training system.

Contents

IEEE Standard for Augmented Reality Learning Experience Model

1. Overview

1.1 Scope

This standard defines two description languages for expressing augmented reality (AR) learning experiences. This document shows how to represent activities conducive to developing or upgrading knowledge, skills, abilities, and other characteristics in a standardized interchange format. The interchange format links the representation of learning activities with their environment and context in the actual (or simulated) workplace, classroom, or other environment in which an AR-enhanced training system may execute them. The specification aims to lower entry barriers for authoring of learning experiences that involve interaction with the real world, sensors, computer vision, and web applications.

1.2 Purpose

This standard for augmented reality learning experience models (ARLEMs) provides an overarching integrated conceptual model and the corresponding data model specifications for representing activities, learning context, and environment (aka "workplace"), while linking with other data model components needed for AR-enhanced learning activities.

The standard distinguishes slow-changing data for environment description from fast-changing data for step-by-step guidance. It defines the required data models and modeling languages and their bindings to Extensible Markup Language (XML; see XML 1.0) and JavaScript Object Notation (JSON; see ECMA-404).[1]

The purpose of this standard is to support the discovery, retrieval, transfer, and execution of AR-enabled learning content, thereby facilitating the creation of repositories and online marketplaces.

Finally, the standard supports reuse and repurposing of existing (learning) content in "mixed" experiences that combine real-world guidance with traditional media such as instructional video material or existing web applications and widgets.

[1] Information on references can be found in Clause 2.

1.3 Word usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (shall equals is required to).[2,3]

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (should equals is recommended that).

The word *may* is used to indicate a course of action permissible within the limits of the standard (may equals is permitted to).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (can equals is able to).

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

ECMA-404, The JSON Exchange Network.[4]

ISO 639-1, Codes for the representation of names of languages—Part 1: Alpha-2 code.[5]

ISO 7010, Graphical symbols—Safety colours and safety signs—Registered safety signs.

OASIS, Message Queuing Telemetry Transport (MQTT), 2014.[6]

Rustici Software, Experience API (xAPI).[7]

Rustici Software, xAPI Client Libraries.

W3C Recommendation, Extensible Markup Language (XML) 1.0, 5th ed., 2008.[8]

W3C Recommendation, Packaged Web Apps (Widgets)—Packaging and XML Configuration, 2nd ed., 2012.

3. Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.[9]

[2] The use of the word *must* is deprecated and shall not be used when stating mandatory requirements, *must* is used only to describe unavoidable situations.
[3] The use of *will* is deprecated and shall not be used when stating mandatory requirements, *will* is only used in statements of fact.
[4] ECMA publications are available from Ecma International (http://www.ecma-international.org/).
[5] ISO publications are available from the International Organization for Standardization (http://www.iso.org/) and the American National Standards Institute (http://www.ansi.org/).
[6] OASIS publications are available from OASIS Open (https://www.oasis-open.org/).
[7] Rustici Software publications are available from xAPI (http://www.xapi.com).
[8] W3C publications are available from the World Wide Web Consortium (https://www.w3.org/).

activity: The execution of a planned workflow following a specific process, leading typically from beginning to end, regardless of whether the anticipated learning outcome is achieved.

augmentation: 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.

augmentation primitive: The types of annotations available are defined in the augmented reality (AR) training system. They include, but are not limited to, audio, video, images, animations, labels, and vibrotactile patterns.

augmented reality (AR): Human perception is enhanced with additional computer-generated sensorial input to create a new user experience, including, but not restricted to, enhancing human vision by combining natural with digital offers.

augmented reality (AR) tracking subsystem: A component of the augmented reality (AR) training system that detects position markers and anchors, typically (but not limited to) using computer vision to determine the location of markers, image targets, or spatial anchors in a room scan.

augmented reality (AR) training system: An augmented reality (AR) training system consists of a single software application or a set of software applications, and connected delivery hardware (such as a head-mounted display, HMD), that allow trainees to practice and perform predefined learning activities in a given workplace. It may also include authoring functionality to facilitate the creation of new learning activities.

detectable: Entities that link to fiducial markers, target feature models, or other sensor state properties providing input to computer vision and other sensor processing systems. They have a unique identity and link to data enabling their tracking with the help of the sensor processing system referenced.

experience API (xAPI): A learning analytics application programming interface (API), standardizing communication with a learning record store for logging learner performance. Any augmented reality (AR) training system may optionally use the xAPI to keep and/or retrieve records of the users' learning activities. If xAPI logging is supported, xAPI statement queries may be used in the if-rules to control the activity.

learning activity: An activity that motivates the development of competence (i.e., knowledge, skills, abilities, and other characteristics).

learning experience: The cognitive and sensory-motoric effect on the user from performing a learning activity in a particular workplace.

predicate: Reusable instructional augmentations, configuring a specific augmentation primitive for its use in activities. The set of predicates defined in a workplace model is a domain-specific language for instruction in the workplace under consideration, typically including all required verbs of handling and movement and their visual overlay animations signifying them.

sensor: A device that detects or measures physical characteristics and communicates the data generated digitally, such as a heart rate sensor, a gyroscope, or a nondescript Internet-of-Things sensor that signals, for example, whether a specific button has been pressed during operation of a manufacturing machine.

trigger: A mechanism firing an event on completion of an action step that allows the statements in the exit section of the action step to be released.

---

[9] *IEEE Standards Dictionary Online* is available at: http://dictionary.ieee.org. An IEEE Account is required for access to the dictionary, and one can be created at no charge on the dictionary sign-in page.

validation constraint: Observed user behavior is checked, recorded in an experience application programming interface (xAPI) endpoint, to control the flow of actions. It includes a query and the corresponding if–then rules defining with which action step to react to the given results' constraints.

warning signs: Graphical symbols that indicate a potential hazard, obstacle, or requirement relating to the identification and management of risks in the workplace.

workplace: A physical environment in which users perform and learn to perform tasks with real-world and virtual objects.

4. Learning in activities and workplaces

Descriptions of learning experiences consist of two types of data. Learning activities contain task-dependent descriptions of *how to interact* with real and virtual elements in the sequence of actions. Workplace environment and context descriptions consist of data concerning *which* real and virtual elements exist and are featured within the experience.

The definition of an activity mixture for learning by experience includes orchestration of user interaction across multiple devices. Furthermore, it integrates the tracking of and responses to user interaction across devices and sensors. Optionally, if-rules may be defined and evaluated as part of the activity to test for user achievements or action outcomes using data from one or more software and/or hardware sensors.

The *activity modeling language (activityML)* in this standard is a domain-specific language used to describe the activity mixture. The *workplace modeling language (workplaceML)* in this standard complements the activity representation language to realize interoperability of applications interpreting activityML. It is a domain-specific language for describing the tangibles, configurables, and basic triggers of a workplace (see Figure 1).

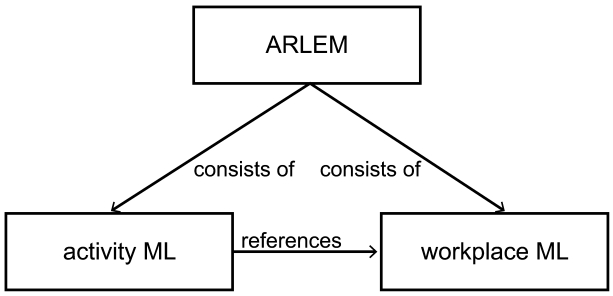


Figure 1—Two modeling languages of ARLEM

Activities are containers that include a sequence of action steps, progressing from the start to the end of the activity, and metadata that describe the overall activity in human- and machine-readable terms. Activity models reference workplace models that define where, when, and how an activity takes place. Each action step of an activity specifies the following:

- **Activation and/or deactivation of augmentations:** Specification and configuration data of media files to play, render, or invoke in other ways, when entering or exiting the current action step.
- **Triggers:** Reference and configuration of the release mechanism instigating the state change from enter to exit.
- *Optional:* **Messages:** Communication with other devices handled by the user or other users. This may include control commands such as launching an action step.

— *Optional:* **If-Rules:** Specification of queries and logic for analyzing and reacting to observed user behavior, recorded in an experience API endpoint.

Workplaces describe the tangible things, persons, and places as well as the virtual objects in a concrete workplace. Moreover, they describe software and hardware, with which interaction may take place, including specifying how specific apps may be launched or which particular delivery devices are supported by the workplace. Furthermore, how to communicate with the sensors supported in the workplace is described here. An example of a sensor is a device measuring physical characteristics such as energy levels using an Internet-of-Things protocol. Additionally, workplace models define the possible detectables, overlay primitives, and warnings signs supported. An example of a detectable is a marker that can be detected using a computer vision system. An overlay primitive, for example, specifies, how the predicate "rotate" should be visually depicted.

5. Activity modeling language for activity scripts

Activity scripts hold one activity element each, which contains at least one action step or a sequence of actions steps as the only direct subordinate elements. Since the activity container element defines an obligatory start action, a minimal activity shall contain at least one action referenced in activity metadata as the action to start with (see Figure 2).

Each action has the following elements specifying how to handle augmentations and determine the action workflow:

a) The mandatory *enter* and *exit* elements are responsible for controlling augmentations, messages between devices and to other users, and simple constraint validation (based on if-rules) of user behavior, as follows:

   1) Both enter and exit elements may contain *activate* and *deactivate* elements, specifying which augmentations and/or actions to launch or remove, respectively. They also define where these augmentations are anchored or what they are attached to in the workplace.

   2) Optional *messages* may specify communication with other devices and/or users.

   3) Optional *if-rules* check observed user behavior, recorded in an experience API endpoint, with the help of a query URL and guide, based on the retrieved results, the branching to other actions.

b) The mandatory *triggers* element contains at least one *trigger* entry, defining how the user initiates the action's state change that executes the statements of the exit block and that moves on to the next action step or activity.

c) The optional *instruction* element contains a human-readable title and description to be displayed or read out to the trainee. This element is optional, allowing the creation of action steps that display only augmentations instead of cluttering the display with instructional prompts. It may be replaced by an *app* element to launch specific web apps or widgets instead.

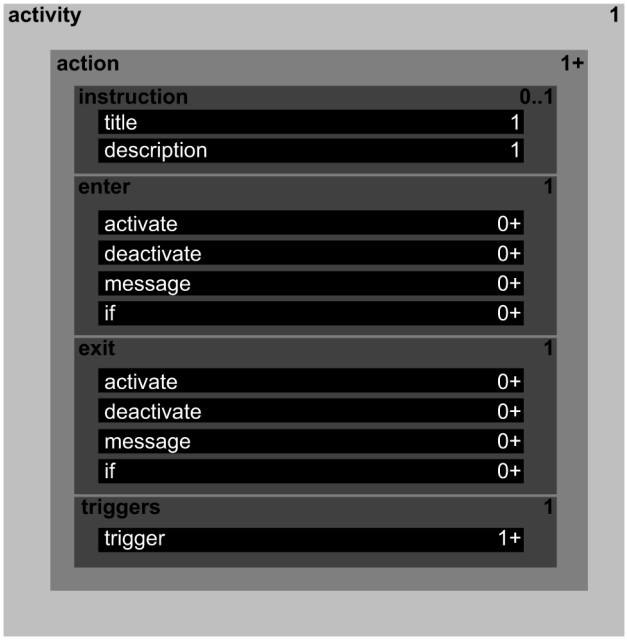


Figure 2—Overview showing the element structure of an activity

The sequence of actions is determined by the activate/deactivate statements and/or validation constraints in each action's enter and exit blocks.

NOTE—The AR training system decides how title and/or description of the instruction element are rendered. For example, the system may read out title and/or description of the instruction, using text-to-speech, instead of displaying a dialogue message, or the display area where the instructions are shown may vary.[10]

6. Workplace modeling language

The workplace model describes the environment in which the user re-enacts activities, including properties and configuration data of real and virtual resources with which the user interacts. Each workplace model contains a single *workplace*, which holds all other data (see Figure 3).

[10] Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

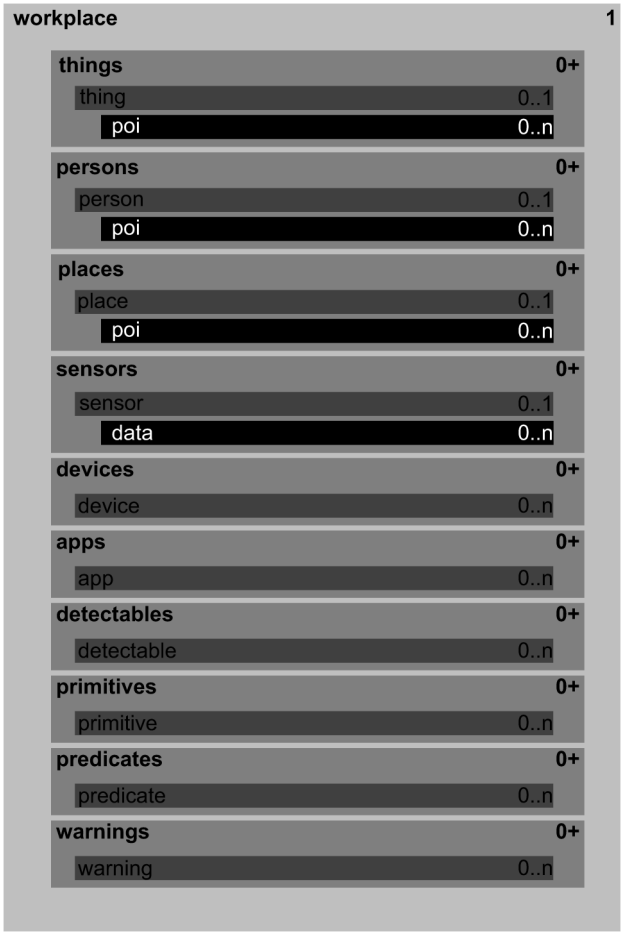


Figure 3—Overview showing the element structure of a workplace

Four types of *resources* are defined within each workplace element: *tangibles*, *configurables*, *detectables*, and *augmentations*. Thereby, *tangibles* distinguish the following three distinct types:

— *Things* refer to physical objects such as materials, tools, and/or machines.

— *Persons* define the people in the workplace with whom the user may interact. Note that the mbox attribute holds the identifier of that person that should also be used for xAPI logging statements, if an xAPI is supported by the AR training system.

— *Places* define locations within the workplace.

For each thing, person, and place, specific points of interest (POIs) may be defined that allow referencing particular parts. Positional offsets along *x*-, *y*-, and *z*-axes as well as *x*-, *y*-, and *z*-rotation shall be relative to a default location on the object. If no default point of interest is defined, the default shall be the center of the object.

NOTE 1—Computer vision (or other sensor-processing units) determine the physical location of an object. For example, the center of an object detected with the help of a fiducial marker will be the center of the marker in practice.

There are three types of configurables, as follows:

— The entries listed under *Sensors* define how to communicate with devices or hardware components of the AR training system that detect or measure physical characteristics. There are various Internet protocols available for communicating with sensors [the mandatory MQTT; further also

Constrained Application Protocol (CoAP), Advanced Message Queuing Protocol (AMQP), Extensible Messaging and Presence Protocol (XMPP), OPC Unified Architecture (OPC-UA), User Datagram Protocol (UDP), or Transmission Control Protocol (TCP)], which all have different levels of adoption in different industries. The sensor entries in the workplace model provide the configuration data required to connect to these sensors, define what data keys to listen for, and how to parse them. This allows triggers in the activity script to react to sensor data.

— *Devices* provide information on what hardware and software resources are available to re-enact a learning experience.

— *Apps* statements hold information on how to launch web apps, that is, Open Social gadgets and widgets (see the W3C recommendations provided in Clause 2). Apps may be included in action statements of activities and then typically replace any instruction element. Apps are web apps, gadgets, and widgets.

Detectables are used to provide configuration information for computer vision and other sensory processing systems, instructing them how to detect any of the tangibles introduced earlier.

Finally, *augmentations* collect configuration data on how to render overlays. They refer to the following sets of elements: augmentation *primitives*, *predicates*, and *warnings*:

— The augmentation *primitives* enumerate the types of AR annotations possible. They include animation, image, video, audio, and label. Thereby, the image primitive instructs the system to create a generic primitive that may be featured as predicate in activate statements in the activity script to overlay a two-dimensional (2D) image with the file provided via URL. In analogy, audio and video provide primitives for loading audio and video files. The label “primitive” instructs the system to provide a generic for overlaying text annotations.

— *Predicates* are reusable instructional augmentations, configuring a specific augmentation primitive for its use in activities.

— *Warnings* define the graphical symbols used to provide health and safety instruction.

NOTE 2—The set of predicates in the workplace model is a domain-specific language for instruction and training in that particular workplace. The set typically includes all necessary verbs of handling and movement.

NOTE 3—Support for multiple devices is required when an activity has user interaction with, for example, both smart glasses and a tablet computer.

7. ActivityML base schema

Table 1 defines the base schema for the activity modeling language (activityML).

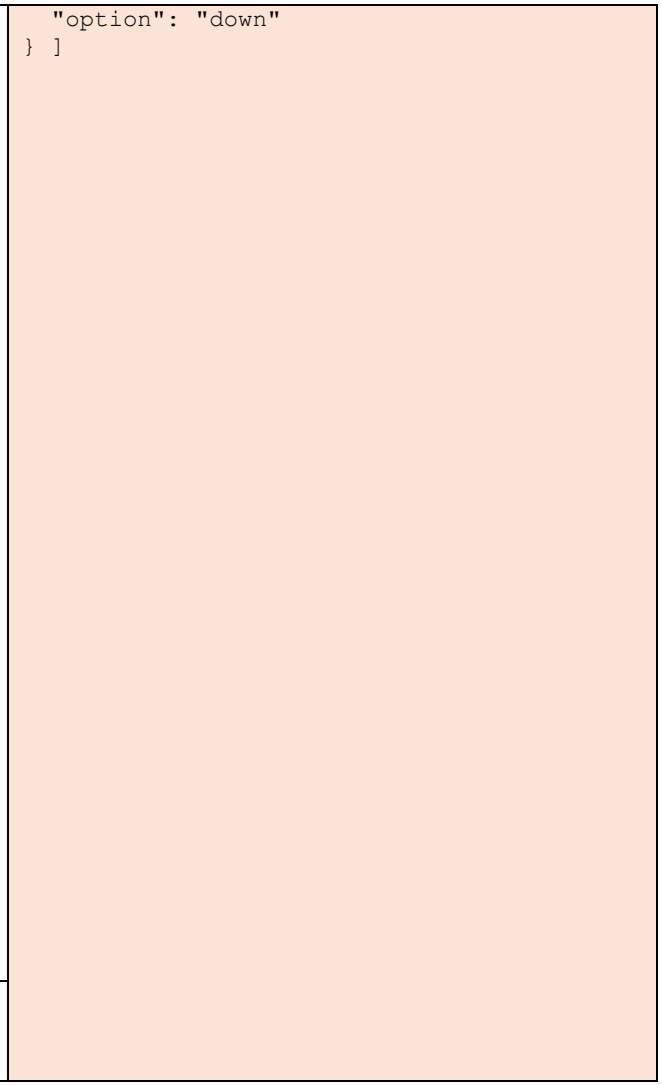
Table 1—ActivityML 1.0 base schema

Number	Name	Explanation	Value Space	Datatype (Maximum Length)	Examples
1	activity	The activity contains a sequence of action steps, leading the user through the workflow needed to develop and demonstrate competence in the intended learning outcome NOTE 1—There is only one activity element per activity file. NOTE 2—In the JSON representation, the activity root node is not named.	Holds action elements only (see 1.1) One activity per script		XML: <activity id="assembly120" name="Assembly of a stowage rack" description="An extensive description" language="en" workplace="http://wekit.eu/workplace.xml" start="step1" > </activity> JSON: { "id": "assembly120", "name": "Assembly of a stowage rack", "description": "An extensive description", "language": "en", "workplace": "http://wekit.eu/workplace.json", "start": "step1" }
1a	*id*	Unique identifier	Alphanumeric	Character String (100)	
1b	*name*	Human readable name of the activity	Alphanumeric	Character String (1000)	
1c	*description*	Human readable description of the activity	Alphanumeric	Character String (10000)	
1d	*language*	Specifies the language with which the activity expresses human-readable instructional prompts	ISO 639-1 language code	Character String (2)	
1e	*workplace*	Reference to the workplace model	workplace.id	URL (2000)	
1f	*start*	Reference to the action to start with	action.id (see 1.1a)	Character String (100)	
1.1	action	Action elements describe the steps of the learning activity NOTE—In the JSON representation, the action nodes are represented as non-named objects.	One or more action elements		XML: <actions><action></action><actions> JSON: "actions": []
1.1a	*id*	Unique identifier (unique within the activity), used for referencing	Alphanumeric	Character String (100)	XML: id="step1" JSON: "id":"step1"

1.1b	*viewport*	The visual area in which any instruction contained in the action displayed NOTE—This may be a fixed area of the display or a named surface of the room.	"actions", "reactions", "warnings"	CharacterString (100)	XML: viewport="actions" JSON: "viewport": "actions"
1.1c	*type*	Specifies the category of the action and is used for styling visual appearance	"actions", "reactions", "warnings"	Character String (100)	XML: type="actions" JSON: "type": "actions"
1.1d	*[device]*	Specify on which device to execute the action	device.id in the workplace model	Character String (100)	XML: device="glasses1" JSON "device": "glasses1"
1.1e	*[location]*	*Optional:* Refer to a particular place where the action happens	place.id in the workplace model	Character String (1000)	XML: location="hallway" JSON: "location": "hallway"
1.1f	*[predicate]*	*Optional:* The action may specify what verb to log user interaction with via the xAPI. When no predicate is provided, logging shall use the standard xAPI verb "launched" for each action step. If xAPI is supported by the AR training system, then any predicate id used shall correspond to the id of the resultant xAPI verb statement definition. NOTE—xAPI support is optional.	predicate.id in the workplace model	Character String (1000)	XML: predicate="rotate" JSON: "predicate": "rotate"
1.1.1	instruction	*Optional:* Human-readable directives and prompts for the learner for this particular action step NOTE—In future versions of the standard, the vocabulary to describe instruction may be extended to cover the possibility to model multiple-choice items.			XML: <instruction> <title>Assembly of a stowage rack</title> <description>Point to the cabinet to start</description> </instruction> JSON:

1.1.1.1	title	The headline of the instruction	Alphanumeric	Character String (100)	"instruction":{ "title": "Intro", "description": "The Mars rover is starting an exploration mission. YOUR TASK: Evaluate the status of the solar panels and charge the battery. You will be guided while performing the task." }
1.1.1.2	description	More detailed instruction	Alphanumeric	Character String (5000)	
1.1.2	enter	Container for augmentations that will be activated or deactivated, and which messages and validation rules will be executed, once the action step is launched	Holds activate, deactivate, message, and if containers (see 1.1.2.1 to 1.1.2.4b)		XML: <enter removeSelf="false"></enter> JSON: "enter": { "removeSelf": false, … }
1.1.2a	*[removeself]*	*Optional:* If set to true, the action triggers the immediate execution of the exit loop once activate, deactivate, message, and if statements contained in this enter loop have been rendered. If not provided, default is assumed to be false.	"true", "false"	Boolean	
1.1.3	*exit*	Container for augmentations which will be activated or deactivated, and which messages and validation rules will be executed, once the action step is exited			XML: <exit removeSelf="true"></exit> JSON: "exit": { "removeSelf": false, … }

1.1.3a	[*removeself]*	*Optional:* If set to true, the action itself is removed (most notably: removal of the instruction; see 1.1.1) before the statements of the exit block are evaluated. If not provided, default is true. If set to false, the action's instruction and augmentations from its activate statements will persist to the next action steps until explicitly removed in any of the deactivate statements.	"true", "false"	Boolean	
1.1.2.1	activate	Statement to execute effector output: load and display an augmentation to the user NOTE—In the XML representation, the activate statements are encapsulated in an "activates" container element to provide similar access as in the JSON representation.			XML: <activates><activate target="board1" type="predicate" augmentation="point" poi="top" option="down" viewport="actions" /></activates> JSON: "activates": [{ "target": "board1", "type": "predicate", "augmentation": "point", "poi": "top",
1.1.2.1a	*target*	Reference to unique identifier of a tangible in the workplace model	thing.id or place.id or person.id	CharacterString (100)	
1.1.2.1b	*type*	Specify which augmentation type the id refers to.	"primitive", "predicate", "warning", "action"	CharacterString (100)	

1.1.2.1c	*augmentation*	Instruct the AR training system to play or display the augmentation with the target id provided (a predicate.id, primitive.id, or warning.id from the workplace model or an action.id from the activity model). NOTE 1—To instruct the AR training system to display an ISO 7010 hazard or warning sign (for example: "W012 electricity hazard" or "M004 wear eye protection"), there are two possibilities: If the activate statement is of type "warning", the augmentation="w012", and a target="board1", then the warning sign will be overlaid on board1. If there is no tangible target referenced, type="warning", and augmentation="m004", then the warning sign will be displayed at a fixed position in the viewport "warnings". NOTE 2—To display the predicate augmentation for "rotate" on top of "board1", the statement shall include type="predicate", augmentation="rotate", and target="board1". NOTE 3—To attach an audio recording, the statement features type="primitive", augmentation="audio", with or without tangible target. To attach an image, type="image", url="http://…".	predicate.id, primitive.id, warning.id, or action.id	CharacterString (100)	 "option": "down" }]
1.1.2.1e	*[poi]*	*Optional:* Specify the point of interest (poi) of the tangible where the augmentation will be displayed.	poi.id of tangible's poi in workplace model	CharacterString (100)	

1.1.2.1f	*[option]*	*Optional:* Configuration option for the augmentation NOTE—A predicate “point”, for example, may support different directions (“up”, “down”, “left”, “right”).		CharacterString (100)	
1.1.2.1g	*[viewport]*	*Optional:* Specify the area of the display where the augmentation is to be displayed.	“actions”, “reactions”, “warnings”	CharacterString (100)	
1.1.2.1h	image/video: *[url]*	*Special case: type=“primitive” and augmentation is “image” or “video”:* The image and video primitives require a URL from which to load a JPEG or PNG image or MP4 video file NOTE—URL is optional. If no URL is given, the system shall look for a resource packaged with the app with the name of the tangible.		URL (2000)	XML: <activate target="board1" type="primitive" augmentation="video" poi="leftside" url="http://myurl.org/myvideo.mp4" /> JSON: { "target": "rover", "type": "primitive", "augmentation": "image", "url": "http://myurl.org/myimage.png", "poi": "anchor15" }

1.1.2.1i	animation: *[url]* *[state]*	*Special case: type="primitive" and augmentation="animation":* The animation primitive allows to reference a URL from which to load an animation FBX file and move it to the key frame specified in "state" By default, if no state is specified, the animation shall be loaded, but not displayed and may then be moved to a different animation state in a subsequent action statement. NOTE—In all cases, the URL attribute is optional. If no URL is given, the system shall look for a resource packaged with the app with the name of the tangible.		url: URL (2000) state: Character String (100)	XML: <activate target="board1" type="primitive" augmentation="animation" poi="leftside" url="http://myurl.org/my3d.fbx" state="1" /> JSON: { "target": "board1", "type": "primitive", "augmentation": "animation", "url": "http://myurl.org/my3d.fbx", "poi": "anchor15", "state": "1" }
1.1.2.1j	label: *text*	*Special case: type="primitive", augmentation="label", and "text" attribute is provided:* Specify which character string to display at a specific tangible's default location or any of its specific points of interest (POI)		Character String (100)	XML: <activate target="board1" type="primitive" augmentation="label" poi="left" text="This is the left side" /> JSON: { "target": "board1", "type": "primitive", "augmentation": "label", "text": "This is the left side", "poi": "left" }

1.1.2.1k	label: *sensor* *key* *[option]*	*Special case: type="primitive" and augmentation="label", and sensor+key attributes present:* The label primitive may connect to a sensor with sensor.id to display data to display the value of variable key. *Optional:* When option is set to "stream", the data shall be read and displayed continuously		sensor.id: Character String (100) sensor.key: Character String (100) option: Character String (10)	XML: <activate target="board1" type="primitive" augmentation="sensor" poi="top" sensor="arduino" key="voltage" option="stream" /> JSON: { "target": "board1", "type": "primitive", "augmentation": "sensor", "poi": "top", "sensor": "arduino", "key": "voltage", "option": "stream" }
1.1.2.1l	audio: *url*	*Special case: type="primitive", id="audio", and url attribute is present:* The audio primitive requires a URL from which to load a WAV or MP3 file NOTE—In all cases, the URL attribute is optional. If no URL is given, the system shall look for a resource packaged with the app with the name of the tangible.		URL (2000)	XML: <activate target="board1" type="tangible" augmentation="audio" url="http://myurl.org/myaudio.wav" /> JSON: { "target": "board1", "type": "primitive", "augmentation": "audio", "url": "http://myurl.org/myaudio.wav" }

1.1.2.2	deactivate	Statement to remove augmentations NOTE—In the XML representation, the deactivate statements are encapsulated in a “deactivates” container element to provide similar access as in the JSON representation.			XML: <deactivates><deactivate target="board1" type="primitive" augmentation="label" poi="default" /></deactivates> JSON: "deactivates": [{ "target": "board1", "type": "primitive", "augmentation": "label", "poi": "default" }]
1.1.2.2a	*target*	Specify target id of a tangible in the workplace model to remove all or specific augmentations attached to it, or an action.id from the activity Use wildcard “*” to remove from all tangibles	thing.id, place.id, person.id, or action.id	Character String (100)	
1.1.2.2b	*type*	Specify which type of workplace element the id references Use wildcard “*” for all types	“primitive”, “predicate”, “warning”, or “action”	Character String (100)	
1.1.2.2c	*augmentation*	Specify id of the augmentation or action to remove Use Wildcard “*” for all augmentations	predicate.id, primitive.id, warning.id, action.id, or “*”	Character String (100)	
1.1.2.2d	*poi*	*Optional:* Specify the point of interest (poi) from where to remove the augmentation	poi.id of the tangible referenced in 1.1.2.2a	Character String (100)	
1.1.2.2e	*viewport*	*Optional:* Specify from which viewport to remove the augmentation	“actions”, “reactions”, “warnings”	Character String (100)	

1.1.2.3	message	Specify message statements to allow sending control commands and communication from device to device and user to user NOTE—In the XML representation, the message statements are encapsulated in a “messages” container element to provide similar access as in the JSON representation.		Character String (10000)	XML: <messages><message target="jake" type="person" viewport="alerts" > Job is done! </message></message> XML: <message target="22ffdab321" type="sensor" key="messages" launch="action15" > Hi Fridolin, please start with action 15! </message> JSON: "messages": [{ "target": "jake", "type": "person", "text": "First find the control panel. Then activate rover brakes by removing the red key from the control panel." }]
1.1.2.3a	*target*	Reference to person.id, device.id, or sensor.id	person.id or device.id or sensor.id	Character String (100)	
1.1.2.3b	*type*	Specify whether to send communication message to a person or control message to a device or sensor	“person”, “device”, “sensor”	Character String (100)	
1.1.2.3c	*[viewport]*	*Optional:* Specify a particular viewport in which to display the message	“alerts”, “actions”, “reactions”	Character String (100)	
1.1.2.3d	*[key]*	*Optional:* Specify which variable key (MQTT “topic”) to use for broadcasting the message contents	Alphanumeric	Character String (100)	
1.1.2.3e	*[launch]*	Specify which action shall be launched on the recipient device The AR training system shall publish launch ids to other devices or sensors on the MQTT topic target + “/” + id.	action.id	Character String (100)	
1.1.2.3f	*[text]*	The message content as human readable text is included as the text node of message element	Alphanumeric	Character String (1000)	

1.1.2.4	if	Specify a rule for validating user behavior and other characteristics with defined queries to determine branching of the flow of action NOTE 1—As xAPI get queries return exactly none, one, or several statements, this query can only be evaluated for whether it returns statements (or not). If min and/or max are provided, it shall evaluate whether the number of statements returned falls into the interval min <= number of results <= max. NOTE 2—In the XML representation, the if statements are encapsulated in an "ifs" container element to provide similar access as in the JSON representation.			XML: <ifs><if url="http://example.com/xAPI/ statements?verb=passed&activity=id" min=2 max=10 then="startAction" else="prohibitAction" /></ifs> JSON: "if": [{ "url": "http://example.com/xAPI/ statements?verb=passed&activity=id ", "min": 2, "max": 10, "then": "startAction", "else": "prohibitAction" }]
1.1.2.4a	*url*	The query URL (for statement retrieval via get), including all parameters	URL	URL (2000)	
1.1.2.4b	*then*	Specify which action to trigger if the query yields one or more (or the specified number) of results	action.id	Character String (100)	
1.1.2.4c	*else*	Specify which action to trigger if the query yields no results	action.id	Character String (100)	
1.1.2.4d	*[min]*	Check whether there are at least *min* number of results	Integer	0 … 65535	
1.1.2.4e	*[max]*	Check whether there are no more than *max* number of results	Integer	0 … 65535	
1.1.4	*triggers*	Specifies which different triggers instigate the state change from enter to exit of an action	Holds trigger elements		XML: <triggers> <trigger

1.1.4.1	trigger	The trigger is enabled after the statements on the enter stack have been evaluated, starting the execution of the stack of exit statements when released NOTE—In the JSON representation, each trigger is modelled as an unnamed object.			
1.1.4.1a	*id*	Reference of the action, sensor, or tangible the trigger is listening to	*action.id* or *tangible*.id or *sensor.id*	Character String (100)	
1.1.4.1	*mode*	There are three different types of triggers, including the ones driven by the user (“click” and “voice” triggers), the ones driven by visibility of tangibles (“detect” triggers), and the ones driven by sensors (“sensor” triggers). Voice triggers shall react to the key word “next”. Click triggers shall offer a confirmation button (“checkbox”) where the instruction”s title and description are displayed to the user.	“sensor”, “detect”, “click”, “voice”	Character String (100)	
1.1.4.1c	*[duration]*	For “detect” triggers, a duration for the gaze lock may to be specified	Milliseconds	Integer	
1.1.4.1d	*[type]*	*Optional:* Specify type of the entity it is sensitive to	“tangible”, “action”	Character String (100)	
1.1.4.1e	*[viewport]*	*Optional:* Specify area of the display	“actions”, “reactions”, “warnings”	Character String (100)	

```
    id="start"
    mode="click"
    type="action"
    viewport="actions"
  />
  <trigger
    mode="detect"
    id="board1"
    type="tangible"
    duration="3"
  />
</triggers>


JSON:
"triggers": [
 {
   "mode": "click",
   "type": "action",
   "viewport": "actions",
   "id": "start"
 }, {
   "mode": "voice",
   "type": "action",
   "viewport": "actions",
   "id": "start"
 }
]
```

	[key] *[value]* *[operator]*	*Special case: mode="sensor":* Specify for which variable "key" to look whether it reaches "value", optionally comparing with a different "operator". The unique key shall be defined in the sensor section of the workplace model. *Optional:* Specify an operator for the comparing value read from value specified. NOTE 1—If no operator is specified, the equal operator shall be assumed. NOTE 2—For between, the interval shall be indicated by a pair of numbers, separated by a semi-colon ";".	operator: "exceed", "below", "equal", "between"	key: Character String (100) value: Character String (100) operator: Character String (100)	XML: <trigger mode="sensor" id="arduino" key="connectionA2B" value="1" /> XML: <trigger mode="sensor" id="heartrate" key="rate" value="80" operator="exceed" /> JSON: "triggers": [{ "mode": "sensor", "id": "black_samurai", "key": "module1_state", "value": "error", "operator": "equal" }]

7.1 Instruction

Action elements in the activity may include an instruction element to provide human-readable directives and prompts on what to do in this particular action step. This element is optional to allow creating action steps that, for example, only display image or 3D overlays instead of cluttering the primary display with instructional prompts. Instructions' title elements contain a short summary of the to do, and the description elements contain narrative text describing in more detail what the user is supposed to do.

If provided, the title shall be listed in the list of action steps as well as directly when presenting the description to the user once the action step is activated. The list of action steps shall provide a confirmation button, typically a checkbox, to allow the user to release "click" triggers (Figure 4; see also 7.5).

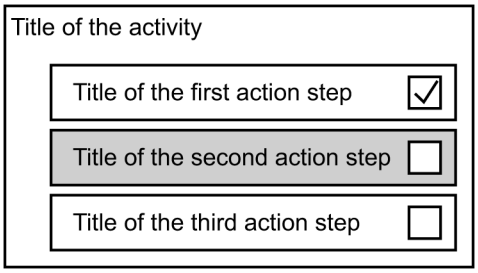


Figure 4—Example list of actions including checkbox trigger and highlighted current step

The AR training system shall support both providing the list of actions (on demand) to the user, to provide an overview of the activity, as well as delivering the instruction title and description for each action step to the user, once the action step is activated. This may be done, for example, in the form of an "action card" (see Figure 5) or as audio using text-to-speech. If an action step does not provide an instruction element, it shall be listed as an empty entry in the list.

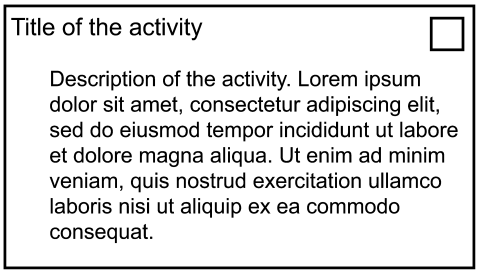


Figure 5—Example action card with checkbox trigger

7.2 Entry, exit, triggers

To define the flow of action within the activity, each action step specifies what triggers the state change from entry to exit. Moreover, each step defines what shall happen when the action is launched (executing the specified statements in the "enter" element) and what shall happen when the trigger moves to the next action (or whatever the "exit" statements define).

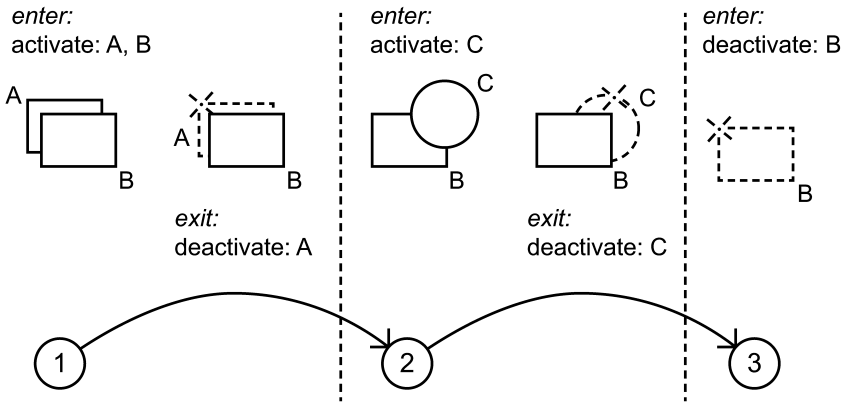


Figure 6—Conceptual model of associated processing behavior (example 1)

Augmentations shall remain active until explicitly deactivated. For example, if an action step 1 activates an augmentation A and then moves on to the next action 2, the augmentation A shall remain visible unless it was explicitly removed in the block of exit statements of step 1 or in the enter section of step 2. For an illustration of this processing behavior, see Figure 6.

NOTE—This ensures that animations with states activated one by one over multiple action steps do not flicker because of deactivation between steps.

7.3 Interacting with tangibles

The workplace model (see Figure 7) defines things, places, and persons, and how they can be detected using the computer vision engine provided by an underlying AR system. The activity script describes merely what to activate and deactivate in the entry and exit routines: this may, for example, activate or deactivate other actions or overlays once a tangible becomes visible and is detected.

7.4 Interacting with actions

Each action step defines which other action steps are activated (or deactivated) in the statements provided in its “enter” and “exit” sections. This provides sequencing information for the flow of the activity.

The AR system shall allow multiple action steps to be launched in parallel from the exit section of an action step to support branching within the flow of the activity. Any such action steps launched in parallel effectively offer alternative choices. When the user makes a choice, the selected action is triggered, which launches the target action defined. This target action then may remove the alternative steps offered previously (see Figure 7). Such processing behavior implements multiple-choice branching.

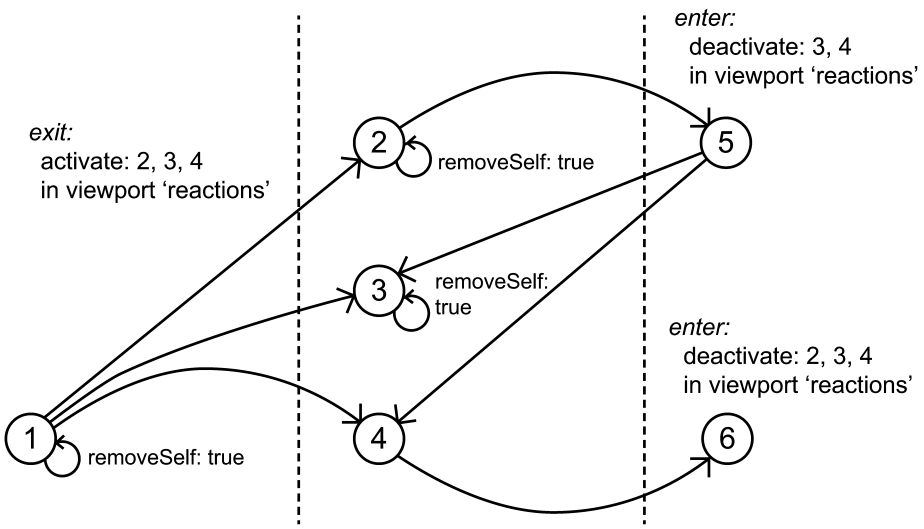


Figure 7—Conceptual model of processing behavior for multiple choice branching

For further notes on branching and other forms of flow control, see 1.1.2.4 of Table 1.

7.5 Triggers

Triggers define what induces the state change from after the action was launched (and all enter statements have been executed) to executing the exit statement stack. There are three different types of triggers, including the ones driven by the user ("click" triggers), the ones driven by the world of things ("detect" triggers), and the ones driven by sensors ("sensor" triggers).

The first type is a click trigger. They register an event hook with the necessary launch statements for the follow-up action to be summoned—and react to the user clicking (or air tapping) on the action step checkbox button near where the instruction's title and description are displayed to the user.

The second type, the detect triggers, react to the AR tracking subsystem signaling to the AR training system that a specific tangible has been detected. When the user gazes at the tangible for the specified duration, the trigger is released.

NOTE—It is recommended to indicate to the user the duration required for such gaze lock. For example, circle segments may be filled successively until the target duration is reached.

The third type, the sensor trigger, picks up on the data sent from connected sensors (see 8.1.2 sensors). Any sensor id used in such a trigger shall be defined in the specified workplace model (see 1.4 of Table 2). The AR system shall connect at the start of the activity to the sensor system. It shall then in each action step that references a sensor trigger monitor the sensor data stream for the appearance of the specified variable keys and check whether the data received meets the trigger conditions.

7.6 Logging user activity using the experience API (xAPI)

Launching of activities, their actions, and specific overlays (like warnings) can provide valuable insights into analytics, ranging from performance information about training success to ensuring accountability through providing evidence of competence.

Any AR training system may optionally communicate with a so-called “learning record store” implementing the experience API (xAPI) to perform logging of user activity. The activity’s action steps provide the semantics required to log an activity trace (a triplet statement of user, the verb listed as predicate, and tangible objects handled or acted on). If a predicate is provided, it shall be used to generate the resultant triplet statements for the learning record store. When no predicate is provided in an action step of an activity, logging to the xAPI shall use the standard xAPI verb “launched.”

NOTE—xAPI client libraries are available from Rustici Software; see Clause 2.

8. Workplace modelling language base schema

A workplace model contains information about the real and virtual environment in which the user interacts (see Table 2).

Table 2—WorkplaceML 1.0 base schema

Number	Name	Explanation	Value Space	Datatype (Maximum Length)	Example
1	workplace	The workplace element contains all information about a physical environment in which users learn by performing tasks with real and virtual objects			XML: <workplace id="ox" name="Oxford Brookes University" origin="001" /> JSON: { "id": "ox", "name": "Oxford Brookes University", "origin": "001" }
1a	*id*	Unique identifier of the workplace	Alphanumeric	Character String(100)	
1b	*name*	Human-readable descriptive name of the workplace	Alphanumeric	Character String(1000)	
1c	*origin*	The id of a detectable that marks the world origin for the work place All positions of all other detectables of type "anchor" shall be expressed relative to this world origin, so that this single detectable may be used to sync *x*-, *y*-, and *z*-coordinate offsets across devices and platforms. NOTE—Using an image target marker in a fixed position works well, such as, for example, a logo on the nose of a plane or a poster beside the entrance.	Alphanumeric	Character String (100)	
1.1	things	The container object holds an array of definitions of the physical workplace objects such as tools, machines, furniture, or materials			XML: <things> <thing id="helicopter" name="AW 109" detectable="333" /> </things> JSON: "things": [{ "id": "rover",
1.1.1	thing	The container element holds the definition of a single physical workplace object			
1.1.1a	*id*	Unique identifier of the thing	Alphanumeric	Character String(100)	
1.1.1b	*name*	Human readable short description of the thing	Alphanumeric	Character String(1000)	

1.1.1d	*detectable*	Specify by id which detectable (see 1.7) applies to this thing NOTE—The most common types of detectables are image markers and world anchors.	Alphanumeric	Character String(100)	"name": "Mars Rover", "detectable": "roverAnchor"]}
1.2	places	Places are tangibles. They refer to locations within the workplace.			XML: <places><place id="platform" name="Platform" detectable="006" /></places> JSON: "places": [{ "id": "hallway", "name": "Hallway", "detectable": "floormarkerhallway"]}
1.2.1	place	A specific location in the workplace			
1.2.1a	*id*	Unique identifier	Alphanumeric	Character String (100)	
1.2.1b	*name*	Human-readable short description of the place	Alphanumeric	Character String (1000)	
1.2.1c	*detectable*	Specify by id which detectable (see 1.7) shall be used to recognize this place	Alphanumeric	Character String (100)	
1.3	persons	Persons refer to the people in the workplace with whom the user may interact			XML: <persons><person id="fridolin" name="Fridolin Wild" twitter="fwild" mbox="wild@brookes.ac.uk" detectable="002" persona="learner" /></persons> JSON: "persons": [{ "id": "fridolin", "name": "Fridolin Wild", "twitter": "fwild", "mbox": "wild@brookes.ac.uk", "detectable": "002",
1.3.1	person	A particular individual			
1.3.1a	*id*	Unique identifier	Alphanumeric	Character String (100)	
1.3.1b	*name*	The individual's name	Alphanumeric	Character String (100)	
1.3.1c	*twitter*	The individual's twitter handle	Alphanumeric	Character String (15)	
1.3.1d	*mbox*	The identifier to be used for the xAPI	Email address	Character String (254)	
1.3.1e	*detectable*	The id of the detectable with which this person can be recognized	Alphanumeric	Character String (100)	

1.3.1f	*persona*	Identifier of the role or group membership of this person, so as to allow for identifying next best alternatives in case of unavailability as well as for the retrieval of additional info about (e.g., access rights)	Alphanumeric	Character String(1000)	<pre>"persona": "learner"]}</pre>
1.1.1e, 1.2.1d, 1.3.1g	*pois*	List of points of interest for this tangible (thing, place, person)			<pre>XML: <pois> <poi id="default" x-offset="0" y-offset="0" z-offset="0" x-rotation="40" y-rotation="-40" z-rotation="0" /> </pois> JSON: "pois": [{ "id": "default", "x-offset": "1.15", "y-offset": "-1.45", "z-offset": "0.85", "x-rotation": "359.15", "y-rotation": "289.62", "z-rotation": "1.23" }]</pre>
1.1.1.1	poi	Each tangible thing, place, and person (1.1, 1.2, 1.3) by default has at least one point of interest ("poi") at the origin, with additional, optional points specifiable. If no default "poi" is provided, the system shall assume the geometric center to be the default location.			
1.1.1.1a	*id*	Unique identifier	Alphanumeric	Character String (100)	
1.1.1.1b	*x-offset*	The offset on the x-axis (in cm)	cm.mm: 0.00 – 1000.00	Float	
1.1.1.1c	*y-offset*	The offset on the y-axis (in cm)	cm.mm: 0.00 – 1000.00	Float	
1.1.1.1d	*z-offset*	The offset on the z-axis (in cm)	cm.mm: 0.00 – 1000.00	Float	
1.1.1.1e	*x-rotation*	Pitch in Euler angles	degrees: 0.000 – 360.000	Float	
1.1.1.1f	*y-rotation*	Yaw in Euler angles	degrees: 0.000 – 360.000	Float	
1.1.1.1g	*z-rotation*	Roll in Euler angles	degrees: 0.000 – 360.000	Float	
1.4	sensors	Sensors are hardware devices that detect or measure physical characteristics and communicates the data generated digitally			<pre>XML: <sensors><sensor id="arduino"</pre>

1.4.1	sensor	For each sensor, the workplace model describes how to connect and what data streams to subscribe to			url="mqtt://wekit.eu:1883" username="wekit" password="****" > <data key="heartrate" type="integer" /> </sensor></sensors> JSON: "sensors": [{ "id": "black_samurai", "url": " mqtt://test.mosquitto.org:1883", "username": "wekit", "password": "****", "data": [{ "key": "adjust_o_matic", "type": "float" }] }]
1.4.1a	*id*	Unique identifier	Alphanumeric	Character String (100)	
1.4.1b	*url*	Uniform resource locator (including protocol and port) of the API endpoint for the Internet-of-Things communication The AR training system shall support MQTT, but it may also support additional protocols.	{protocol://host.domain[:port]}	Character String (1000)	
1.4.1c	*username*	Authentication information	Alphanumeric	Character String (100)	
1.4.1d	*password*		Alphanumeric	Character String (100)	
1.4.1e	*data*	One or more data elements describe how to parse the data stream			
1.4.1e.1	*key*	The variable identifier in the data stream	Alphanumeric	Character String (100)	
1.4.1e.2	*[type]*	The data type of the variable	"string", "float", "integer", "boolean"	Character String (50)	
1.5	*devices*	Devices are the AR hardware delivery systems that the AR training system may communicate with to render the learning experience for the user			XML: <devices><device id="22ffdab321" type="hololens" name="Fridolin's Smart Glasses" owner="fridolin" url="mqtt://test.mosquitto.org:1883" topic="hangarchat" username="" password="" /></devices> JSON:
1.5.1a	device	The hardware device that may be used to deliver activities or parts of activities created in this workplace			
1.5.1b	*id*	Unique identifier	Alphanumeric	Character String(100)	
1.5.1c	*type*	Type of device (e.g. "hololens", "ipad")	Alphanumeric	Character String(100)	

1.5.1d	*name*	Human-readable name of the device	Alphanumeric	Character String(1000)	"devices": [{ "id": "22ffdab321", "type": "hololens", "name": "Fridolin's smart glasses", "owner": "fridolin", "url": "mqtt://test.mosquitto.org:1883", "topic": "hangarchat", "username": "", "password": "" }]
1.5.1e	*[owner]*	*Optional:* The person id of the owner of a device	Alphanumeric	Character String(100)	
1.5.1f	*[url]*	Uniform resource locator (including protocol and port) of the API for the device communication (shall support MQTT; others may be supported)	{protocol://host.domain[:port]}	Character String (1000)	
1.5.1g	*[topic]*	The channel name on which the device listens for messages	Alphanumeric	Character String (1000)	
1.5.1h	*[username]*	Authentication information for the API	Alphanumeric	Character String (100)	
1.5.1j	*[password]*		Alphanumeric	Character String (100)	
1.6	*apps*	The apps container element holds definitions of how to interface the AR training system with other apps and widgets			XML: <apps><app type="widget" id="rating" name="User Feedback" url="http://wekit.eu/widgets/rating.wdgt" /></apps> JSON: "apps": [{ "id": "rating", "type": "widget", "name": "User Feedback", "url": "http://wekit.eu/rating.wdgt" }]
1.6.1	app	Widget or app			
1.6.1a	*type*	The type of application: HTML widget, launch command, or app prefab	"widget", "app", "prefab"	Character String(100)	
1.6.1b	*id*	Unique identifier	Alphanumeric	Character String(100)	
1.6.1c	*name*	Human-readable short description	Alphanumeric	Character String(100)	
1.6.1d	*url*	The uniform resource locator of the manifest file, launch command, or download link for an app prefab asset	Alphanumeric	Character String(1000)	

1.7	detectables	Detectables are entities that link to fiducial markers, target feature models, or other sensor state properties providing input to the computer vision system NOTE—Detectables have a unique identity and link to data enabling their tracking with the help of the AR tracking subsystem referenced.			XML: <detectables><detectable id="005" sensor="tracking" type="marker" url="http://wekit.eu/marker.asset" /></detectables> JSON: "detectables": [{ "id": "001", "sensor": "tracking", "type": "marker", "url": "http://wekit.eu/marker.asset" }]
1.7.1	detectable	A detectable instructs the system how to recognize tangibles (things, places, or people)			
1.7.1a	*id*	Unique identifier (referenced in 1.1.1d, 1.2.1d or 1.3.1e)	Alphanumeric	Character String (100)	
1.7.1b	*[sensor]*	*Optional:* The type of AR tracking subsystem to be used for its detection, routing the configuration data (upon parsing of a workplace) to the correct detection engine If no sensor is specified, the AR system shall use a default subsystem for markers and a default subsystem for world anchors.	"tracking", "mapping"	Character String(100)	
1.7.1c	*type*	May be either a marker or a workplace anchor relative to the workplace origin	"marker", "anchor"	Character String(100)	
1.7.1d	*[url]*	*Optional:* Uniform resource locator pointing to asset bundle with additional data (e.g., data about point clouds of feature targets / about image targets)	Alphanumeric	Character String (2000)	

1.8	primitives	Augmentation primitives define the fundamental types of annotations available in the AR training system NOTE—The workplace model informs the AR training system, which of the primitives are supported in this workplace. Moreover, for each augmentation primitive, the workplace model defines default size and/or volume level.			XML: <primitives><primitive id="animation" x-size="0.01" y-size="0.01" z-size="0.01" /></primitives> JSON: "primitives": [{ "id": "animation", "x-size": "0.01", "y-size": "0.01", "z-size": "0.01" }]
1.8.1	primitive	Definition of a particular type of augmentation			
1.8.1a	*id*	Unique identifier	"animation", "image", "video", "audio", "label"	Character String (100)	
1.8.1b	*[x-size]*	*Optional:* Initial sizing values for images and objects	cm.mm: 0.01 – 1000.00	Float	
1.8.1c	*[y-size]*		cm.mm: 0.01 – 1000.00	Float	
1.8.1d	*[z-size]*		cm.mm: 0.01 – 1000.00	Float	
1.8.1e	*[volume]*	*Optional:* A volume setting, used for auditory primitives	0 – 100	Integer	
1.9	predicates	Predicates are reusable instructional augmentations, configuring a specific augmentation primitive for its use in activities NOTE—The set of predicates defined in a workplace model should form a domain-specific language for instruction in the workplace under consideration, typically including all required verbs of handling and movement and their visual overlay animations signifying them..			XML: <predicates><predicate id="point" type="animation" scale="1" url="http://wekit.eu/point.fbx" /></predicates> JSON: "predicates": [{ "id": "point", "type": "animation",

1.9.1	predicate	A particular verb NOTE—If xAPI is supported, each verb listed here shall have a corresponding xAPI verb statement.			<pre>"scale": "1", "url": "http://wekit.eu/point.fbx" }]</pre>
1.9.1a	*id*	Unique identifier	Alphanumeric	Character String(100)	
1.9.1b	*type*	Is an instance of primitive.id (1.9.1a)	"animation", "image", "video", "audio", "label"	Character String(100)	
1.9.1d	*scale*	A (normalized) scaling factor to be applied to the primitive, across all axes	cm.mm: 0.01 – 1000.00	Float	
1.9.1e	*url*	Uniform resource locator pointing to a downloadable file with the augmentation	Alphanumeric	Character String(1000)	
1.10	warnings	Definition of the supported warnings and instructions NOTE—The AR system shall support all warnings listed in ISO 7010.			<pre>XML: <warnings><warning id="M004" type="image" symbol="eyeprotection" scale="2" /></warnings> JSON: "warnings": [{ "id": "M004", "type": "image", "symbol": "eyeprotection", "scale": "2.00" }]</pre>
1.10.1	warning	A graphical symbol that indicates a potential hazard, obstacle, or requirement relating to the identification and management of risks in the workplace			
1.10.1a	*id*	Unique identifier	Alphanumeric	Character String(100)	
1.10.1b	*type*	Is an instance of primitive.id (1.9.1a)	"animation", "image", "video", "audio", "label"	Character String(100)	
1.10.1c	*symbol*	Name of the symbol prefab	Alphanumeric	Character String(100)	
1.10.1d	*scale*	A (normalized) scaling factor to be applied to the primitive, across all axes.	cm.mm: 0.01 – 1000.00	Float	

8.1 Tangibles, configurables, and augmentations

Four different types of resources are described in a workplace model: tangibles, configurables, detectables, and augmentations. Tangibles are things, places, and persons (anything that is "real"). Configurables group together all software and hardware information, such as sensors, devices, and their styling, as well as apps. Detectables collect information on how to detect tangibles using computer vision. This includes fiducial markers, image targets, point clouds, or world anchors. Finally, augmentations group together information on which augmentation primitives, predicate augmentations, and warning signs are defined in this workplace and how they are signified with visual, audio, or haptic signals.

8.1.1 Tangibles—things, places, persons

Activities consist of action steps, where users interact with the workplace. During each step, this may involve handling, movement, communication, or other forms of interaction with the things, places, or persons in that workplace that matter to the activity. Each tangible is defined with a unique identifier and a human-readable name. Moreover, it links to a detectable (see 8.1.3) to specify how it can be discovered and identified by the AR training system automatically. Each tangible thing, place, or person has a default point of interest. If none is provided in the workplace model, the AR training system will assume the geometric center of its detectable to be the default location. Additional points of interest may be added so that they can be referred to via the "poi" attributes in the activity models.

The default point of interest of a tangible is relative to the location of its detectable. This may be the geometric center of the position of a marker (possibly moving around in space) or an anchor defined relative to the workplace origin. The workplace origin is typically configured using a single workplace calibration marker. All additional, nondefault points of interest are defined relative to the default location. All anchors are specified relative to the workplace origin.

8.1.2 Configurables—sensors, devices, apps

There are three types of configurables: sensors, devices, and apps. Devices provide information on what hardware is available to reenact the learning experience. For example, there may be activities that require both smart glasses and a tablet computer to be used in interplay.

Apps define how to launch web apps such as Open Social gadgets and widgets (see the W3C recommendations provided in Clause 2). Apps may be included in action statements of activities, typically replacing any instruction element.

Sensors are hardware devices that detect or measure physical characteristics and that communicate the data generated digitally. For each sensor, the workplace model describes how to connect and to what data streams to subscribe.

8.1.3 Detectables—markers, image targets, point clouds, world anchors

The detectables are used to provide configuration information for computer vision and other sensory processing systems, instructing them on how to detect tangibles.

8.1.4 Augmentations—primitives, predicates, warnings

Augmentations in the workplace model refer to three different sets of elements: generic overlay primitives, predicates, and warnings. The overlay primitives provide configuration information for augmented instructional domain-specific languages, including sets of standard predicates tuned to the particular work place and sets of warnings and hazard signs.

8.1.4.1 Primitives—three-dimensional models/animations, images, videos, audio, labels

The AR training system shall implement the following generic augmentation primitives to allow the dynamic inclusion of external resources in the activity models. The minimum set of augmentation primitives that shall be supported include 3D overlays and animations ("animation"), "image," "video," "audio," and text "label."

Animations, the first generic predicate primitive, may either be embedded already within the app or they may be provided dynamically via a URL. For dynamic animations, the app may download either when launching the activity or on the fly, when the overlay is activated.

All 3D animations are animation sequences with states. Each animation sequence shall be addressed via the "state" attribute. By default, if no state is specified, the animation shall be loaded but not displayed and may then be moved to a different animation state in a subsequent action statement.

For videos dynamically included from external resources, the URL attribute shall convey the link to an MPEG-4, or equivalent, file.

Similarly, images dynamically included shall point to the resource URL with the JPEG or PNG file.

The final generic primitive is "label." Text labels shall be placed at or near a specific tangible's default location or any of its specific points of interest (POIs).

8.1.4.2 Predicates

Predicates contain predicate primitives to define the basic vocabulary for handling and motion in a particular workplace. Selecting predicates for handling and movement will depend on the application area.

NOTE—For example for the manufacturing of helicopters, textiles, and furniture, the following verbs are a good starting point: point, assemble, close, cut, disassemble, drill, inspect, lift, locate, lower, lubricate, measure, open, pack, pick, place paint, plug, rotate (clockwise), rotate (counter clockwise), screw, unfasten, unpack, unplug, unscrew. This list is certainly not complete, but it contains key predicates that can be expressed directly with visual instruction.

Where necessary, additional parameters shall be provided on instantiation in activate and/or deactivate statements (for example: option="up").

There are other, more generic primitives that every system shall implement: forbid and allow. Forbid thereby signifies what the user is not allowed to touch, whereas allow may be used to highlight explicitly what the user is expected to touch.

8.1.4.3 Warnings

The AR training system shall implement an interface for displaying the ISO 7010 warnings. The supported warning signs shall be specified in the resultant workplace model. The activity model then may specify, where needed, whether the warning sign is placed in space near the relevant location/object or into a fixed

viewport area. To display a warning sign near a tangible, the activity model's activate statement shall reference a type="tangible" and the resultant tangible id in the id attribute. To show a warning sign in a fixed position of the display area, the type shall be type="warning" and the id shall be the id of the warning sign.

NOTE—As a general recommendation, in most situations, elements relevant to the health and safety of the person (e.g., "wear protective gloves") should best be displayed in a fixed position of the viewport, whereas object- or location-related warnings (e.g., "do not touch") should better be placed near, but not occluding, the actual object.

Annex A

(informative)

Bibliography

Bibliographical references are resources that provide additional or helpful material but do not need to be understood or used to implement this standard. Reference to these resources is made for informational use only.

[B1] Guest, W., F. Wild, A. Vovk, M. Fominykh, B. Limbu, R. Klemke, P. Sharma, J. Karjalainen, C. Smith, and J. Rasool, "Affordances for capturing and re-enacting expert performance with wearables," *European Conference on Technology Enhanced Learning*, 2017, pp. 403-409.

[B2] Helin, K., T. Kuula, C. Vizzi, J. Karjalainen, and A. Vovk, *User Experience of Augmented Reality System for Astronaut's Manual Work Support,* R. Front. Ed., 2018. doi:10.3389/frobt.2018.00106.

[B3] Limbu, B., M. Fominykh, R. Klemke, M. Specht, and F. Wild, "Supporting training of expertise with wearable technologies: the WEKIT reference framework," in *Mobile and Ubiquitous Learning*. New York: Springer, 2018, pp. 157-175.

[B4] Limbu, B., H. Jarodzka, R. Klemke, F. Wild, and M. Specht, "From AR to expertise: A user study of an augmented reality training to support expertise development," *Journal of Universal Computer Science*, vol. 24, no. 2, pp. 108-128, 2018.

[B5] Sharma, P., R. Klemke, and F. Wild, "Experience capturing with wearable technology in the WEKIT project," in *Perspectives on Wearable Enhanced Learning,* I. Buchem, F. Wild, and R. Klamma, Eds. Berlin, Germany: Springer, 2019.

[B6] Wild, F., "The future of learning at the workplace is augmented reality," *IEEE Computer*, vol. 40, no. 1, pp. 96-98, 2016. http://doi.ieeecomputersociety.org/10.1109/MC.2016.301.

[B7] Xue, H., P. Sharma, and F. Wild, "User satisfaction in augmented reality-based training using Microsoft HoloLens," *Computers*, vol. 8, no. 1, p. 9, 2019, doi:10.3390/computers80010009.

IEEE SA
STANDARDS ASSOCIATION

RAISING THE WORLD'S STANDARDS

Connect with us on:

Twitter: twitter.com/ieeesa
Facebook: facebook.com/ieeesa
LinkedIn: linkedin.com/groups/1791118
Beyond Standards blog: beyondstandards.ieee.org
YouTube: youtube.com/ieeesa

standards.ieee.org
Phone: +1 732 981 0060

IEEE