Definition of Key Terms and Components in Realizeit



Contents

[1. Introduction 4](#_Toc498415864)

[2. Target Knowledge and Content 4](#_Toc498415865)

[3. Target knowledge 4](#_Toc498415866)

[3.1. Curriculum 4](#_Toc498415867)

[3.2. Knowledge items and Nodes 5](#_Toc498415868)

[3.3. Prerequisite links 5](#_Toc498415869)

[4. Content 6](#_Toc498415870)

[4.1. Content titles 6](#_Toc498415871)

[4.2. Package 6](#_Toc498415872)

[4.3. File 7](#_Toc498415873)

[4.4. Content item 8](#_Toc498415874)

[5. Objectives and products 8](#_Toc498415875)

[5.1. Learning objectives 8](#_Toc498415876)

[5.2. Products 9](#_Toc498415877)

[5.3. Product versioning 9](#_Toc498415878)

[5.4. Changes to versioned products 10](#_Toc498415879)

[6. Publishing content 10](#_Toc498415880)

[6.1. Install and activate 10](#_Toc498415881)

[7. Ownership and sharing 11](#_Toc498415882)

[8. Evidence 11](#_Toc498415883)

[8.1. Direct Evidence 11](#_Toc498415884)

[8.2. Indirect evidence 12](#_Toc498415885)

[9. Metrics and Data Capture 12](#_Toc498415886)

[9.1. Ability 12](#_Toc498415887)

[9.2. Knowledge State 12](#_Toc498415888)

[9.3. Knowledge Covered 12](#_Toc498415889)

[9.4. Question logging 12](#_Toc498415890)

[9.5. Estimated times 12](#_Toc498415891)

[9.6. Time 13](#_Toc498415892)

[10. Triggers and Events 13](#_Toc498415893)

[10.1. Completing a node 13](#_Toc498415894)

[10.2. Unlocking a node 13](#_Toc498415895)

[10.3. Coloring on a student graph 13](#_Toc498415896)

[10.4. Completing an objective 14](#_Toc498415897)

[10.5. Movement between objectives 14](#_Toc498415898)

# Introduction

This document provides a description of the components of courses in Realizeit. Product versioning and publishing content are also overviewed.

# Target Knowledge and Content

At the core of the Realizeit Unified Learning Model is a Learning and Analytic Engine composed of complex artificial intelligence methods coupled with cognitive methods. This works in harmony with an evidence harvesting methodology, the engine providing sophisticated decision making to determine optimum pathways for each learner. The Engine integrates the Target Knowledge with the Content components of the system.

Target Knowledge

Artificial Intelligence Engine

Content

# Target knowledge

## Curriculum

Essentially a curriculum is a set of learning objectives or skills that a learner needs to attain. It can be designed to represent intended outcomes of learning, but may also cover information relating to the means by which these outcomes will be achieved.

A curriculum, or knowledge space, could encompass humanities, social sciences, STEM subjects, language learning, leadership training, advanced continuing education for high performing staff, skills training of any kind such as negotiation skills, communication, treasury management, an employee orientation course, occupational health and safety information - in fact any area of knowledge or skill set requiring learning.

Traditionally a curriculum is defined using a hierarchical representation i.e. the curriculum definition breaks down into a number of nodes, each of which can be further subdivided.



Figure 1: Curriculum hierarchy

Figure 1 is an example of a section of a curriculum hierarchy.

## Nodes

The curriculum consists of hierarchical and base nodes. Hierarchal nodes can be viewed as part of the organizational structure of a curriculum whereas base nodes, usually just referred to as nodes, represent knowledge items, concepts, skills or competencies of any kind. These could be delivered and assessed within to Realizeit or in some external system or setting.

To be delivered within Realizeit, learning content is set against nodes in this space.

## Prerequisite links

Prerequisite links between nodes can be created in Realizeit. This results in a connected graph of nodes. This graph will be used to plan different paths through the required learning objectives. The path chosen by the system will depend on the individual learner. In addition, this visual representation of the path allows for immediate visibility of the learner's achievement and progress at all points in the path.

Prerequisite relationships indicate elements of the curriculum that must be learned before another node can be attempted. They can be used in determining the best course of action to take when a learner is encountering difficulty at a particular point. The system can determine prerequisite areas where the student has a weakness and automatically suggest or deliver remedial action.

The system allows a prerequisite view to be designed alongside (or instead of) the hierarchical. The relationships described in this way can then be used to guide or direct teachers and learners through a program.

This example shows a small section of a prerequisite view of a Mathematics curriculum. The linking arrows in the diagram indicate the prerequisite relationship. The item from which the link begins is a prerequisite of the item at the head of the arrow.



Figure 2: A prerequisite network

# Content

## Content titles

Content titles provide a mechanism for grouping related pieces of learning content. In a book-publishing context a content title would typically represent a book, but the system allows content to be structured in the most appropriate manner for different scenarios.

In the case of a book ingested to the system, a content title representing the book would be created automatically.

In Figure 3, content titles have been created for organizing learning content. Typically, each curriculum will have its own content title.



Figure 3: Content titles

## Package

A Realizeit package is a self-contained piece of learning content that can be installed into the system’s learning delivery system. It is made up of HTML files (containing the learning material and the questions) and resources of various kinds. A SCORM package can be uploaded to the system, and used as the basis of a Realizeit package.

In the case of a book ingested to the system, packages representing each chapter of the book would be created automatically.

## File

A file represents a single lesson within a package. A file can be associated with any number of nodes, but typically will just be associated with one.

The authoring file contains both the content (split into pedagogical sections) and the questions for that lesson.

The file is attached to a node (typically an end/child node) in the curriculum.

Figure 4 shows a package containing files relating to various topics on Integers. The package details are provided on the *Details* tab. The *Files* tab includes a list of the files in the package. A list of resources contained in the package is also accessible.



Figure 4: Packages page, details tab



Figure 5: Packages page, files tab

## Content item

A package needs to be installed into the system, and activated, before it becomes available to the learning delivery system. This means that an author can work on a package, previewing the content created, without making it available for use by a learner.

When a package is installed and activated a ‘content item’ is created. This represents the actual content that a learner will use.

# Objectives and products

## Learning objectives

A learning objective is a subset of one or more knowledge spaces, or curricula. An objective could be an entire knowledge space. However, since the purpose of creating an objective is so that it can be assigned to one or more learners, a full knowledge space, which could consist of several hundred knowledge elements, would tend to be overwhelming for them, so appropriate subsets will tend to be chosen instead. This means that one can easily see from the visual representation of the graph the progress that is being made, both at a group and an individual level. The advantage, from a motivational point of view, is that an attainable target can be seen by the student.

Note that an objective can span different curricula.

An objective can be constructed by choosing the elements and/or attributes that are needed. The system will determine all of the prerequisite material that is required for the objective. There are a number of options available that can be used to refine the nature or behavior of the objective.

The curriculum tool can be used to visualize and plan objectives, so it is possible to select the elements for inclusion in the objective from the curriculum tool. There are two modes for viewing a knowledge space in the curriculum tool, *Hierarchy view* and *Prerequisite view*.

In some cases, the knowledge elements for an objective may have a great number of prerequisite elements preceding them. Prerequisites can be excluded, or assumed, in an objective definition to make this manageable.

## Products

Each course in the system is represented as a product. A typical product consists of a content title (such as English Composition) and a number of objectives (these objectives represent the units or phases, each of which typically covers a week).

As an example, we could have a product *Math 112*.



This might consist of the content title *College Algebra*, and the following objectives:

1. Algebra Essentials
2. Linear Equations and Inequalities
3. Functions and Graphs
4. Quadratic Equations and Quadratic Functions
5. Systems of Equations

Products can share content titles but will typically have their own objectives.

## Product versioning

There are a number of critical structures involved in the preparation of a course:

* Curriculum (including elements)
* Objectives
* Content titles (including packages and files)
* Products

These items combine to define the material that is required for a course. There will always be a working (current) version of each of these entities.

When preparing for a new session, a new version of each product needs to be created. This ensures that ongoing changes to the main (or base) version do not affect the courses that are in progress. For reporting, analytics and regulatory purposes, it means that a snapshot of the course, as it was when delivered during a particular session, is always available.

Course development should always be carried out on the primary (non-versioned, base, main) objects.

When a product is versioned the system creates a copy of the product, the content title(s) used (along with all packages, files, resources etc.), the objective(s) used and it creates a version of the curriculum.

For each of the non-curriculum entities the system creates actual copies and links them back to the source. Curricula are a little different because the system keeps track of the different versions of a curriculum that exist. These exist within the context of the single curriculum. When a product is versioned the system will create a new version within each curriculum that is used within the product; it will then store as part of the product record the curriculum versions that should be used.

The system will suggest a version label which can be amended if desired. The institution usually has a labeling convention which encapsulates the session name/code/date.

## Changes to versioned products

Changes to the main versions of the objects can be made; these changes will not affect any versions.

A versioned product can be updated to reflect the contents of the underlying base objects, by selecting the product and pressing the "Refresh version from base product" button. This will create, update or delete any underlying structures that are required to bring the product version into line with the current version.

When a product version is created its content are now separate from the main (live) version. It is possible to make changes directly in the versioned content, depending on the nature of the change. For example if one typographical error needed to be corrected the change might be made directly in the version (as well as in the base) to avoid having to do a full refresh of all of the content.

# Publishing content

It is important to note that, when changes are made to content, these changes are not visible to students until the content has been republished (installed and activated).

## Install and activate

Content can be installed and activated at the content title level or at the package level. Typically, before the start of a session the following will happen:

* A new version of a product will be created (this will create a new version of the associated content title).
* The content for the newly created content title version will be installed and activated.

The author can make further changes to the files within a package at any point. These changes are not automatically made available to students. A new version of the content can be made available to learners whenever it is convenient by installing and activating again.

# Ownership and sharing

Access to all of the principle data items in the system is controlled by ownership and sharing. The Ownership and sharing details for an item can be seen by viewing that item. Each item has an owner who has full access to the item. An organization and a group for sharing can also be defined for each item, with a level of sharing specified for each.

There is also an *Additional sharing details* link that allows the item to be shared to further organizations and groups, and to individuals.

When an item is shared, the access level is specified as one of the following:

* None
* View existence
* View code/name
* View most details
* View all details
* Change data
* Change owner
* Delete
* Full access

Each of these includes the preceding rights.

# Evidence

## Direct Evidence

Direct evidence is information that is observed and relates directly to a node. This currently can come from several sources but other external sources of evidence are also possible due to the generic nature of the underlying model.

* Questions asked during a lesson delivery (from the Questions section) - note that questions asked as part of examples do not contribute to evidence.
* Questions asked during Practice
* Questions asked during Determine Knowledge (if at least one question is correct e.g. 0/2 does not constitute direct evidence because we have only learned that the person does not know the topic, whereas 1/3 indicates that they have some knowledge of the topic)
* Slips; a question on node A can produce a slip for node B so direct evidence is applied to node B for the slip.

## Indirect evidence

The system applies direct evidence to nodes and then uses the EPN and EAN (the evidence networks) to apply indirect evidence to connected nodes. Indirect evidence is the propagation of direct evidence through the networks.

# Metrics and Data Capture

## Ability

The system maintains a probability function for each person for each node that they use. The ability for a student for a node is the ability value (in the range 0 to 1) that has the highest probability of being true. The probability function is adjusted using Direct and Indirect evidence. Note that Ability is not the same as the "score" achieved by answering questions. For example, a score of 8/10 will contribute as direct evidence to the probability function, but does not necessarily mean that the ability value will then be 80%. The 8/10 evidence will be combined with any previous evidence to give a new ability value.

## Knowledge State

The Knowledge State for an objective is calculated as the average ability for all nodes in the objective *that have direct evidence associated with them*.

The term "Knowledge State" is user definable via Terminology.

## Knowledge Covered

The Knowledge Covered for an objective is calculated based on the number of nodes in the objective that have been completed. If "Identify base knowledge" has been used for an objective then the system will only consider "new" nodes in the objective i.e. nodes that do not fall within the set of "Identify base knowledge."

The term "Knowledge Covered" is user definable via Terminology.

## Question logging

The system automatically logs details of all questions asked during Try, Practice, Determine Knowledge or a Question section during lesson delivery. Questions from other sections that are asked during question delivery are only logged if a student sends a message to an instructor that relates to the question.

## Estimated times

When the system has observed enough evidence relating to the usage of nodes it can create a formula to allow it to predict the time it will take somebody with a particular ability level to complete a node. In the absence of the data (referred to in other documents as Element metrics) the system uses a default value. A node specific default value can be defined. If this is not defined then the system uses an overall default of 20 minutes (for each node).

## Time

Time spent using the system is recorded and logged. This consists of:

* Time in lessons: this is calculated as the time difference between the start of the lesson delivery and the end of the lesson delivery. If the lesson is suspended and resumed then this will be calculated as the sum of the individual parts. If a lesson is left inactive for 20 minutes the system will automatically time out and save the lesson.
* Time in Practice / Determine Knowledge (assessments): this is calculated as the time difference between the start of the operation and the end of the operation.

Time details are stored for individual activities and are automatically aggregated for each objective.

# Triggers and Events

## Completing a node

A node is completed when a learner achieves a level of competency on the node. This can be achieved through:

* The Determine knowledge algorithm
* Trying a node successfully
* Completing a lesson successfully

A node is successfully tried or completed when the % of questions answered correctly on the node equals or exceeds the "Pass threshold."

The "Pass threshold" can be defined as part of a Skin or more specifically for a grouping/class. If it is not defined in either of these locations, the system uses a default value (40%).

If a node is opened via Determine Knowledge or the Try operation then the system will unlock and mark as completed any prerequisite nodes that are not currently accessible.

## Unlocking a node

A node is unlocked when all its prerequisite nodes have been completed.

If there are no prerequisites for a node then it is never locked.

If a node has multiple prerequisites then all of them must be completed.

The possible approaches to completing a node are defined above.

## Coloring on a student graph

Nodes are only colored on the graph when direct evidence is available i.e. the system will not color based solely on indirect evidence.

The color key used for the graph can be defined at the Organization level i.e. each school has its own ranges and colors that are applied.

## Completing an objective

An objective is completed when two conditions are met:

* All nodes in the objective have been completed
* The overall standard for the objective is greater than or equal to the "Minimum objective standard."

The "Minimum objective standard" can be defined locally for an objective, at the grouping/class level or as part of the Skin.

If no value is specified then the system will not apply a minimum objective standard i.e. completing all of the nodes in the objective is sufficient to complete the objective.

## Movement between objectives

In general, a learner will select an objective and work through the learning that it defines. However, there are cases when the system will prompt or direct a learner to another objective:

* If there are no "open" or "unlocked" nodes in the current objective then the system will indicate this to the learner and provide direction to objectives that do contain material that the learner can work through.
* When an objective is completed (as defined above) the system will direct a learner to the objectives that build upon the knowledge of the current objective; if there are none then the learner is directed to other objectives that contain "open" nodes.
* If a selected node is locked because of prerequisite nodes that are not included in the scope of the current objective then the system will indicate the objectives from which they are drawn and allow the learner to switch objective.
* If the objective is completed by completing a lesson then the end page of the lesson will offer possibilities for further objectives (as described above this will first aim for objectives that build on the current knowledge).