CGS Architecture

Software Design Document

|  |  |
| --- | --- |
| Document Owner: | Ronen Mintz |
| Document File: | CGS\_architecture\_SDD.docx |
| Document Location: | <http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_architecture_SDD.docx> |

# Revisions

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Author | Description | Date |
| 0 | Ronen Mintz | Create | 18/10/2012 |
| 1 | Ronen Mintz | Renamed book to course & added course data structure & references | 6/11/2012 |
| 2 |  |  |  |

# Table of Contents

[Revisions 1](#_Toc339952927)

[Table of Contents 2](#_Toc339952928)

[1 Solution, Product Component or Feature Overview 4](#_Toc339952929)

[1.1 Definitions & Abbreviations 4](#_Toc339952930)

[1.2 Feature Description 4](#_Toc339952931)

[1.3 QA Requirements for the Feature 4](#_Toc339952932)

[1.4 Functionality Overview 4](#_Toc339952933)

[1.4.1 Behavior/Appearance 4](#_Toc339952934)

[1.4.2 User Experience and Compliance Requirements 4](#_Toc339952935)

[1.4.3 Configurability 4](#_Toc339952936)

[1.4.4 Usage and Procedures 5](#_Toc339952937)

[1.4.5 Error Handling 5](#_Toc339952938)

[1.4.6 Database Changes 5](#_Toc339952939)

[1.4.7 Internalization/Localization 5](#_Toc339952940)

[1.4.8 Performance 5](#_Toc339952941)

[1.4.9 Compatibility & Migration 5](#_Toc339952942)

[1.4.10 Supportability 5](#_Toc339952943)

[1.4.11 Portability and Platform-Specific Requirements 5](#_Toc339952944)

[1.4.12 Security 5](#_Toc339952945)

[1.4.13 Installer Changes 5](#_Toc339952946)

[1.5 Testing & Exit Criteria 5](#_Toc339952947)

[1.6 External Dependencies 6](#_Toc339952948)

[1.7 Assumptions 6](#_Toc339952949)

[2 Architecture 6](#_Toc339952950)

[2.1 Deployment Diagram 6](#_Toc339952951)

[2.2 Architectural Use Case Diagram 7](#_Toc339952952)

[3 Solution, Product Component or Feature Design 8](#_Toc339952953)

[3.1 Design Overview 8](#_Toc339952954)

[3.1.1 CGS Data Model 8](#_Toc339952955)

[3.2 CGS 10](#_Toc339952956)

[3.2.1 CGS Login 10](#_Toc339952957)

[3.2.2 CGS Login Description 10](#_Toc339952958)

[3.2.3 CGS Login Sequence 10](#_Toc339952959)

[3.2.4 Open Course 11](#_Toc339952960)

[3.2.5 Create Course 12](#_Toc339952961)

[3.2.6 Lock/Unlock Course Element 13](#_Toc339952962)

[3.2.7 View Course 14](#_Toc339952963)

[3.2.8 Edit Course 15](#_Toc339952964)

[3.2.9 Publish Course 16](#_Toc339952965)

[3.2.10 Manage Course Assets 17](#_Toc339952966)

[4 Traceability to Requirements 18](#_Toc339952967)

[5 Open Issues and Future Enhancements 18](#_Toc339952968)

[6 Risks 18](#_Toc339952969)

[6.1 Impact 18](#_Toc339952970)

[6.2 External Risks 18](#_Toc339952971)

[7 References 18](#_Toc339952972)

# Solution, Product Component or Feature Overview

## Definitions & Abbreviations

|  |  |
| --- | --- |
| DTP | Digital Teaching Platform |
| CGS | Content Generation Studio |
| TOC | Table of Content |
| LA | Learning Activity |
| LO | Learning Object |
| DL | Dynamic Layouts |
| MHE | McGraw Hill Education |

## Feature Description

This document describes the high level architecture of the CGS platform. The CGS enables content editor to create edit & publish Timetoknow Courses.

The CGS is designed as thick client application. This means for example, that the CGS will download the entire book to the local file system before it could be viewed or edited. Most of the operations will be performed locally and will be saved to a repository server only once the user chooses to save his work (as opposed to a standard web application which only updated the current working screen).

## QA Requirements for the Feature

<Describe any requirements needed by QA to test this feature (e.g. backdoors, special log entries etc.).>

## Functionality Overview

### Behavior/Appearance

The CGS is a RIA which is designed as a 2-Tier architecture. The client side is conceptually designed as a standalone application (thick client) which implements most of the content editing logic. The server side has a thin logic which is used for collaboration functionality between different users or systems.

For more information see CGS [frontend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx) and [backend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_backend_SDD.docx) SDDs.

### User Experience and Compliance Requirements

See CGS [frontend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx) SDD.

### Configurability

See CGS [frontend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx) and [backend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_backend_SDD.docx) SDDs.

### Usage and Procedures

N/A

### Error Handling

See CGS [frontend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx) and [backend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_backend_SDD.docx) SDDs.

### Database Changes

See [CGS backend SDD](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_backend_SDD.docx)

### Internalization/Localization

See CGS [frontend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx) SDD.

### Performance

Because the CGS client is designed as a standalone application (and not a thin web client) it will download the entire eBook when a user opens an existing book. This process might take a few minutes but once the entire book has downloaded the user experience should be very fast, while editing or viewing the book.   
This is due to the fact that all the data resides on the local drive of the browser and is not fetched every time via network calls from the server.

### Compatibility & Migration

#### Backward Compatibility

N/A

#### Platform Compatibility

* OS: Windows/Mac
* Browsers: Chrome only

#### Migration/Upgrade Requirements

N/A

### Supportability

See CGS [frontend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx) and [backend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_backend_SDD.docx) SDDs.

### Portability and Platform-Specific Requirements

N/A

### Security

* Login Process: will be perform by integrating with local content provider LDAP server
* All CGS repository server exposed services are secured and will not be exposed to non-authenticated users.

For more information see CGS [frontend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx) and [backend](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_backend_SDD.docx) SDDs.

### Installer Changes

An automatic installation will be created for the CGS.

## Testing & Exit Criteria

* Automation tests: 80% code coverage
* Stress Testing: 100 concurrent users

## External Dependencies

The CGS has integration point with three main systems:

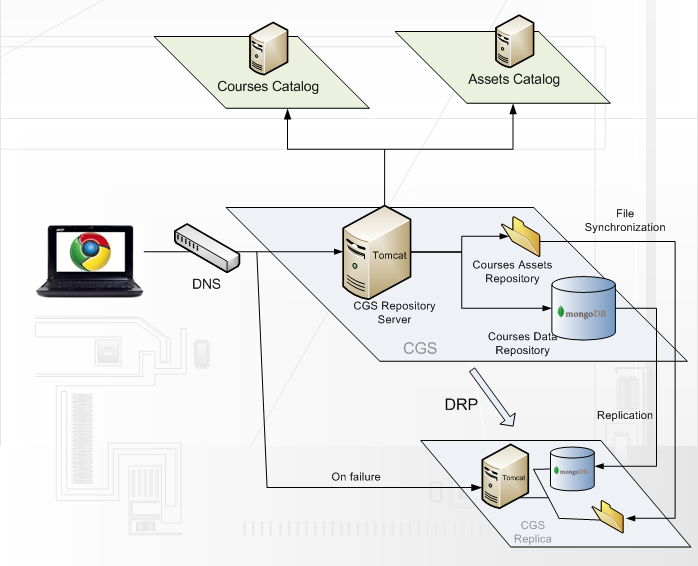
* LDAP server: used to authenticate users
* Assets Catalog: used to import assets, such as images and movies, while editing Courses.
* eBook Catalog: used to publish Timetoknow completed Courses.

## Assumptions

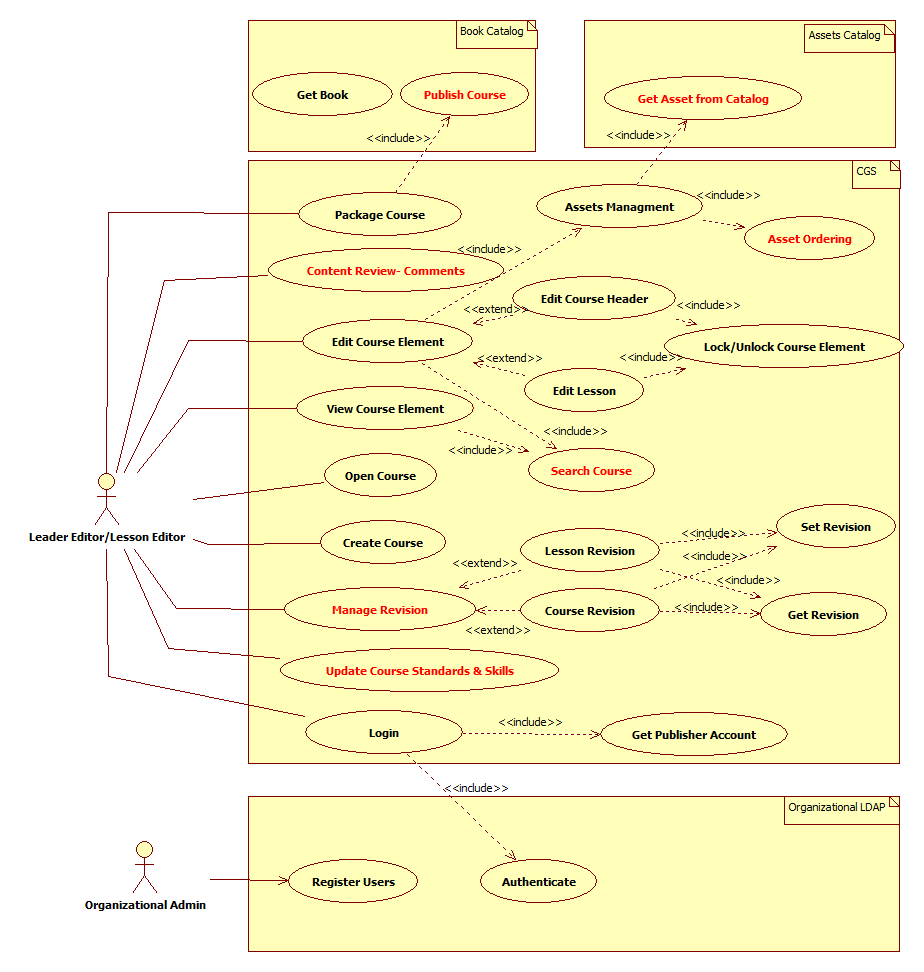
The main assumption behind the current design is that users will use Chrome in order to access the application. For better performance it is suggested that users will work within a high speed LAN network.

# Architecture

## Deployment Diagram



## Architectural Use Case Diagram



# Solution, Product Component or Feature Design

## Design Overview

The CGS is a RIA which is designed as a 2-Tier architecture. The client side is conceptually designed as a standalone application (thick client) which implements most of the content editing logic. The server side has a thin logic which is used for collaboration functionality between different users or systems.

The client side is HTML 5 based and depends on Chrome APIs to access the local file system.

The server side is composed of a Tomcat server which distributed the eBook data between two main repositories:

* Book Data Repository: is responsible to store the eBook JSON documents. This repository uses MongoDB as its persistence database.   
    
  The rational for choosing MongoDB as the persistency store is the following: due to the thick client architecture the server side has a role of a JSON persistency store. MongoDB fits perfectly for this type of requirements due to its ability to fetch and store JSON/partial JSON objects efficiently and in a reliable way.
* Assets Repository: is responsible to store all BLOB data such as images, sound files, movies and other content players internal data structures.   
  This repository is persisted to the file system and is managed by the CGS repository server.

### Course Packaged Data Structure

Note:

* See manifest example [here](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/course.json).
* See lesson example [here](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/lesson.json).

## CGS

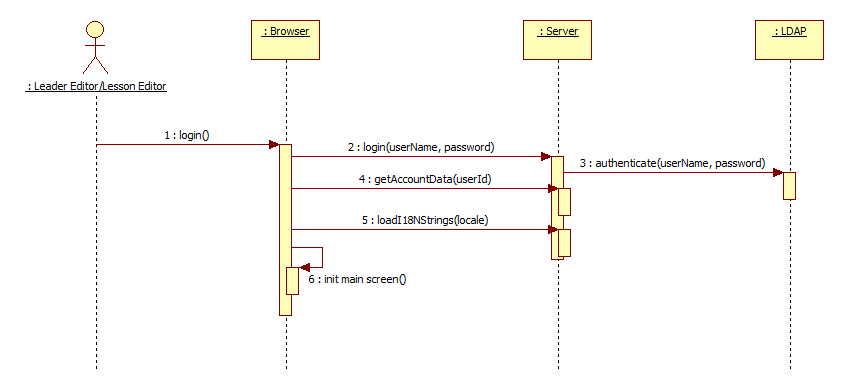
### CGS Login

### CGS Login Description

The following use case describes the login process.

Because the CGS is a multitenant system a user is associated to a specific account after the login process is complete.

### CGS Login Sequence

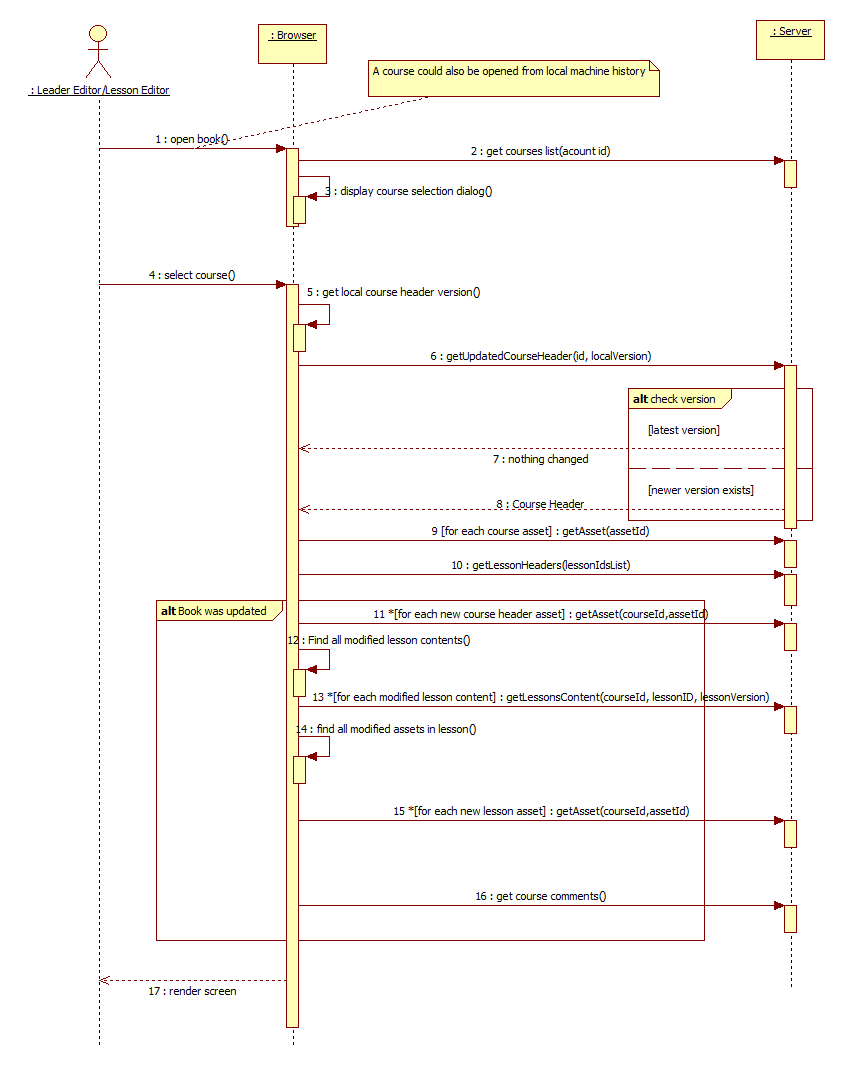


### Open Course

#### Open Course Description

This section describes the use case in which a user opens an existing Course which is currently under development.

#### Open Course Sequence Diagrams

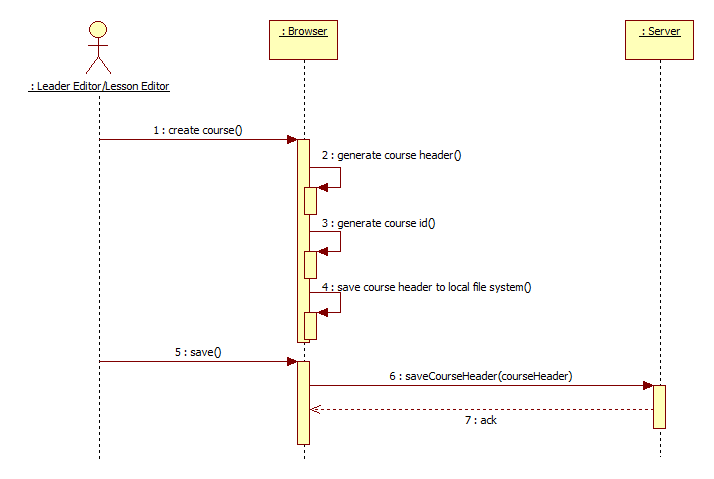


### Create Course

#### Create Course Description

This section describes the use case in which a user (usually a lead editor) creates a new course.

#### Create Course Sequence Diagrams

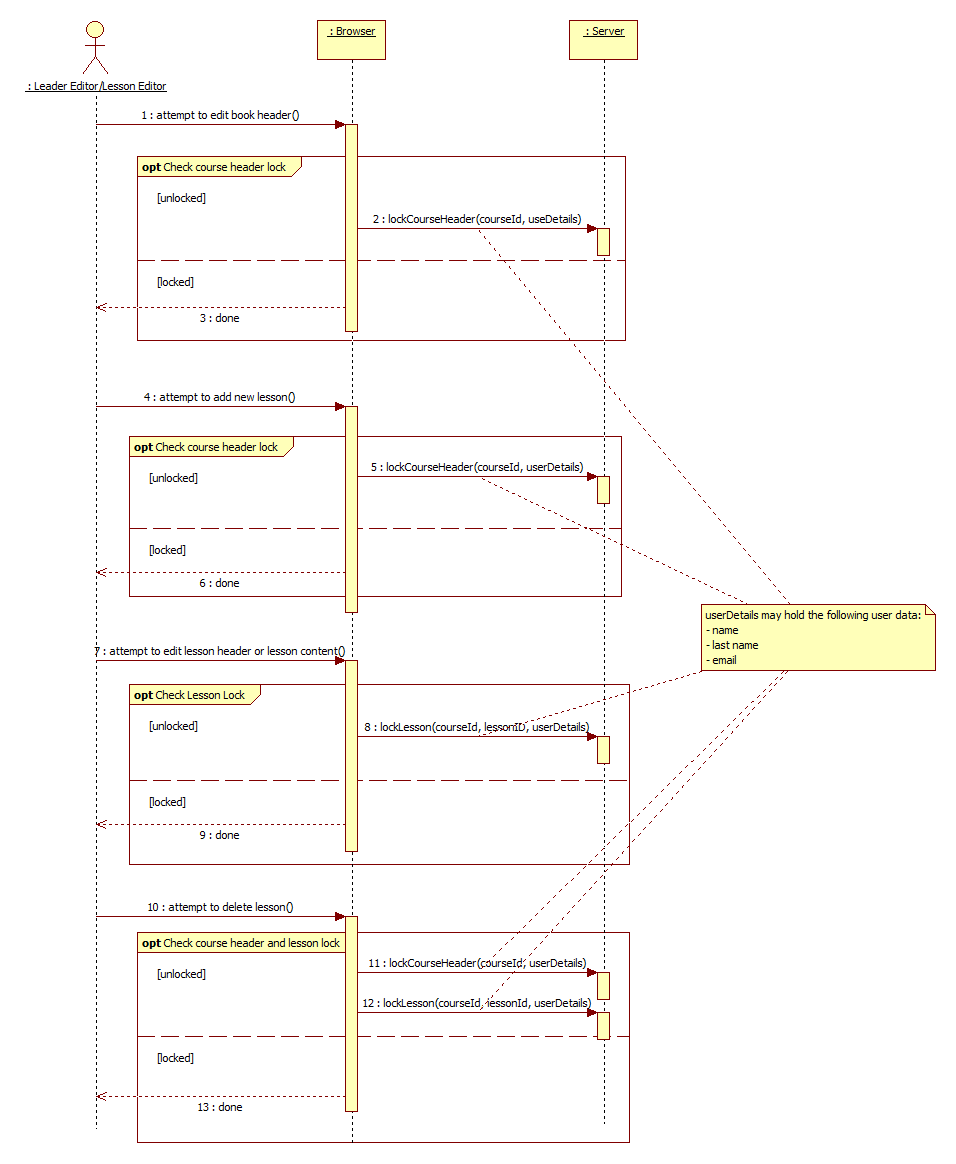


### Lock/Unlock Course Element

#### Lock/Unlock Course Element Description

This section describes the locking mechanism which is required in order to synchronize concurrent modifications in a Course.

#### Lock/Unlock Sequence Diagram

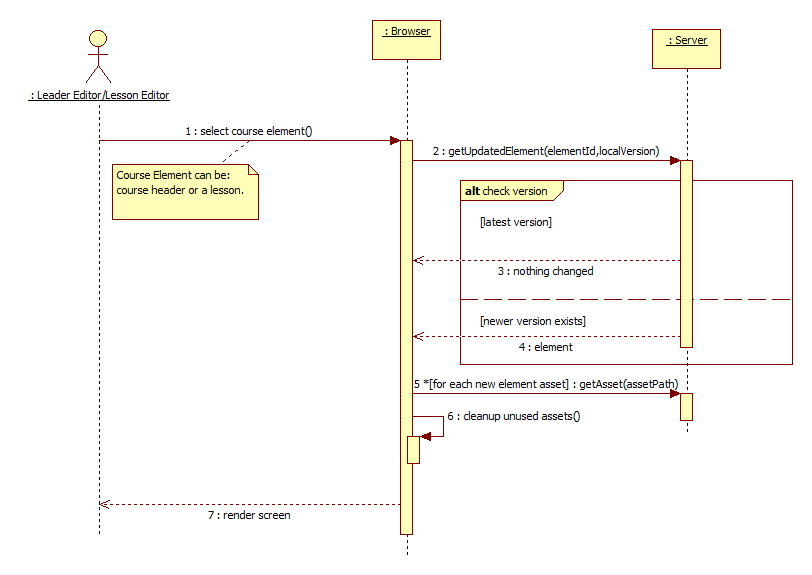


### View Course

#### View Course Description

This section describes the data synchronization process between the browser and the server while a user browses through different course lessons.  
The synchronization process is essential due to the nature of the offline design of the client application.

#### View Course Sequence Diagrams

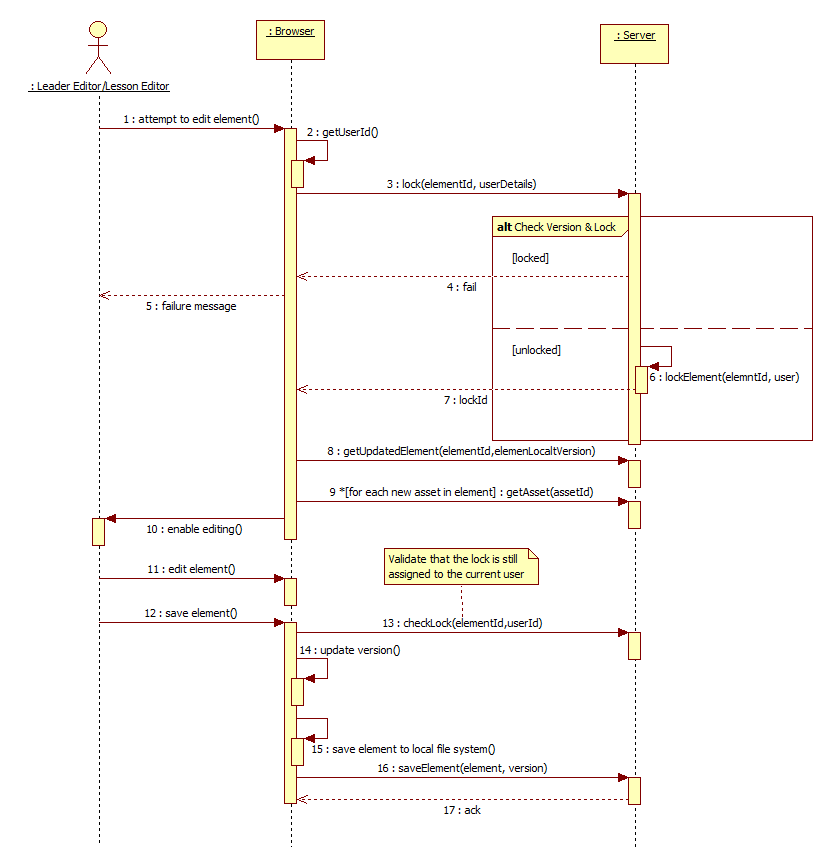


### Edit Course

#### Edit Course Description

This section describes the flow of editing a Course element (which may be the Course header or lesson).

#### Edit Course Sequence Diagrams

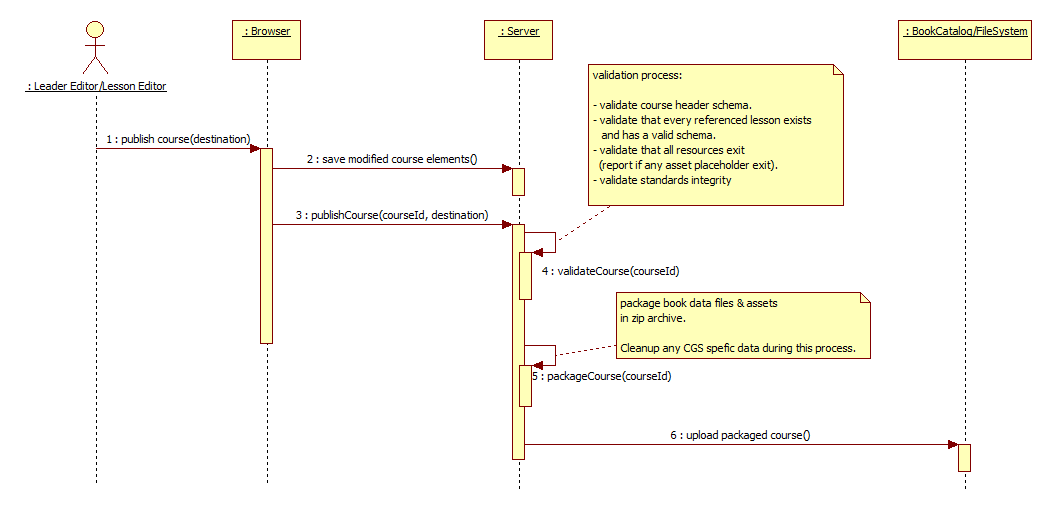


### Publish Course

#### Publish Course Description

This section describes the process of packaging a complete course to the Timetoknow Course catalog.

#### Publish Course Sequence

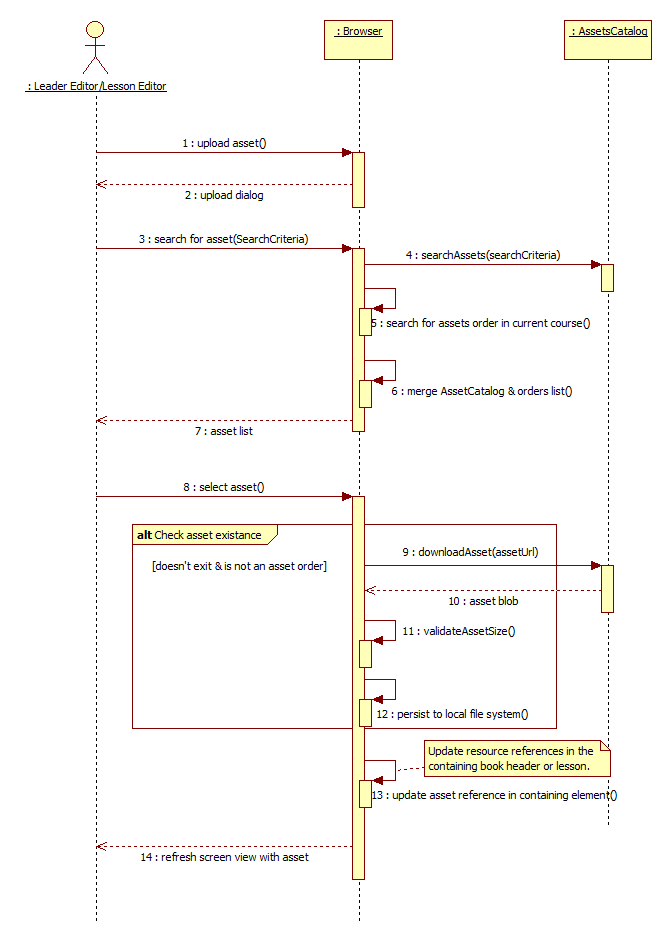


### Manage Course Assets

#### Manage Course Assets Description

This section describes the process of search and adding a new asset (image, sound or movie files) while editing a Course.

#### Manage Course Assets Sequence



# Traceability to Requirements

| # | Requirement Description | Reference to PRD | Release |
| --- | --- | --- | --- |
| 1 | CGS PRD | <http://portal/cge/Shared%20Documents/Products/CGE%207.0/PRD/CGS-PRD-1.docx> | 7.0 |
| 2 |  |  |  |
| 3 |  |  |  |

# Open Issues and Future Enhancements

<List any open issues and also list any possible future enhancements.>

# Risks

## Impact

N/A

## External Risks

Due to its reliability on chrome APIs, the CGS might be affected if Google will decide to change some of their APIs in future releases of their Chrome browser.

# References

* [CGS Frontend SDD](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_frontEnd_SDD.docx)
* [CGS Backend SDD](http://turtle.timetoknow.local:8080/svn/dev/t2kdev/trunk/cgs/doc/design/architecture/CGS_backend_SDD.docx)