SOFTWARE REQUIREMENT SPECIFICATION

FOR

PPTX TO HTML5 CONTENT CONVERTER

PREPARED BY SHAMIL FARAJULLAYEV

RUSTAM HASHIMOV

NAHID HAMIDLI

ÖMER BAYKAL

Table of Contents

1 INTRODUCTION

This is the software requirement specification for PPTX to HTML5 content converter project. We begin with an introduction section. The introduction section consists of a clear definition of the problem we are trying to solve by this project, purpose and scope of this software requirement specification report, literature survey on existing products similar to ours, user survey to find out potential audience that our project addresses, information about definitions and abbreviations used throughout this document, the list of references that we have referred to, and an overview for the rest of this software requirement specification report.

1.1 PROBLEM DEFINITION

Educational institutions and education departments of companies are faced with a problem during the process of instructing the employees which is being unable of keeping the track of employees' development. They arrange instructive meetings for the workers which costs the companies a lot time and effort. Therefore, some of them starts to look for alternative solutions like cooperating with some consulting companies. Most of the education programs that prepared for employee education are presented as Power-point slides; thus, it is impossible for employers to follow the process directly. To solve this problem, these consulting companies convert the Power-point files to e-learning packages with SCORM[1] (Sharable Content Object Reference Module) standard manually. We intend to solve this difficulty by developing a Power-point plug-in which will automatically convert PPTX documents into HTML5 formated documents, that is accessible from any platform and its only requirement is a web browser, without almost any human effort on SCORM standards while keeping the structure of slides as well as allowing the instructor to see which parts of the lesson are studied. Thank to this method, a lot of time and effort will be saved.

1.2 PURPOSE

The purpose of this SRS report is to give a complete description of the PPTX to HTML5 content converter project that is to be developed. This document provides the terms of agreement between the customer and the supplier company. The agreement includes what is agreed to be done by the software. The document also makes it easier for developers of the project to develop and draws a rough line of process. Furthermore, this document will be the basis for the design documents of the project to be written later. In addition, this document also helpful for users of the product that will be developed, by getting some pre-knowledge of the system.

1.3 SCOPE

The PPTX to HTML5 content converter is intended to be a plug-in for Microsoft Power-Point. What it will provide to its users is after making them create or edit the hierarchy of slides, by only a button click, do the conversion automatically. This system will enable its users to make categorization on slides. As clarified in the problem definition section, the system will help the educators to inspect the training process and lessen time and effort.

1.4 USER AND LITERATURE SURVEY

Theoretically, everyone will be able to use our product. That is because it will be very easy to use and requires no experience or specific knowledge. However, practically, it will focus on educational institutions, education departments of companies and the university alumni.

There are some examples of such type of applications at the market such as:

- Articulate Presenter[2]
- Adobe Presenter[3]

which convert Microsoft Power-point files to flash (.swf) format. Unfortunately, both programs cannot transform .pptx files to html5 which is said to be the future of the web and available for all platforms.

1.5 DEFINITIONS AND ABBREVIATIONS

Below there are some abbreviations that have used before and will be used throughout this document.

SRS: Software Requirement Specification

SCORM: Sharable Content Object Reference Model

MS: Microsoft

OS: Operating System

1.6 REFERENCES

Online Resources

[1] http://scorm.com/scorm-explained

[2] http://www.articulate.com/products/presenter.php

[3] http://www.adobe.com/products/presenter.php

[4] http://en.wikipedia.org/wiki/Software_development_process

1.7 OVERVIEW

This first chapter of the document was an introduction to the project. As a last section of first chapter, the organization and contents of the rest of the document will be provided in here.

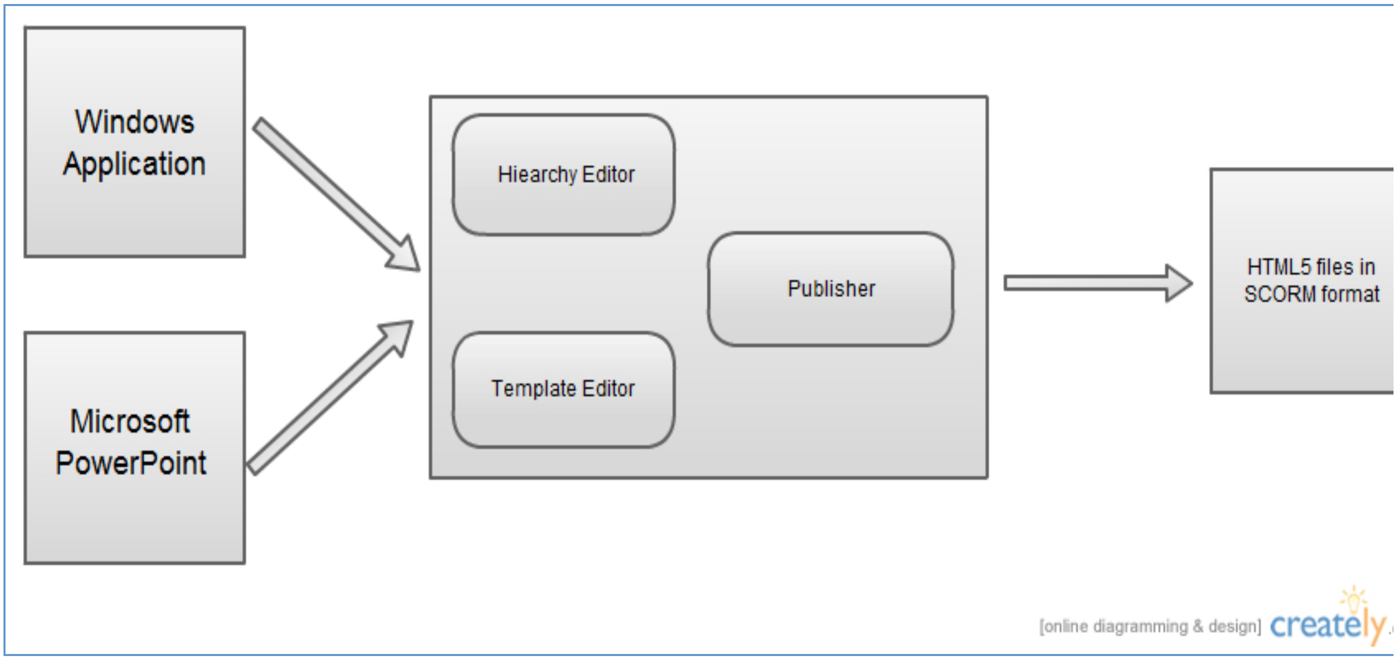
The following chapters are, respectively, the overall description, the specific requirements, data model and description, behavioral model and description of project. Then, in the planning chapter, information about development team and scheduling will be provided. And finally, we will finalize our SRS with a conclusion chapter.

2 OVERALL DESCRIPTION

The software aims to solve the problem defined above, which is, in brief, the lack of a tool for educationists to train the employee about a certain program taught by Microsoft Power-point slides and more importantly, to follow the education progress. Thus, the software to be developed will need an implementation to convert the PPTX files into a specific form that will be able to provide detailed observation capabilities for the employer/trainer. The software is supposed to be as simple and understandable as possible for any educationists with basic Power-Point knowledge such that the process can be done with an effort as little as clicking more or two buttons.

2.1 PRODUCT PERSPECTIVE

There will be two different implementations; therefore two different solutions, namely a plug-in and a windows application, depending on if the user has a valid copy of the official Microsoft Office Power-point 2007 program which is able to create/edit PPTX files. If the user has it, then it is best to install and use the Power-point plug-in, which natively sits on the original Power-point 2007 user interface, and provides more user friendly and interactive way of doing the whole operation. Plug-in lets the users to adjust and see the final product of the conversion process in real-time and side-by-side with the original slides. On the other hand, if the user does not have the Microsoft Office 2007, then it is recommended to use the windows application since it will ask user only to choose the PPTX file and the desired options. Application will have several windows/tabs such as main window, hierarchy window and the templates selection window.



2.2PRODUCT FUNCTIONS

Our software product which is converter PPTX to HTML5 has three major functions: Publish function, Hierarchy function and Template selector function.

Publish function:

This function is main function. The main purpose of this function is to convert PPTX to HTML5.After converting, by means of publish function, we will save HTML5 pages in zip file which obey to SCORM standards.SCORM is s a set of technical standards for e-learning software products.SCORM tells programmers how to write their code so that it can “play well” with other e-learning software

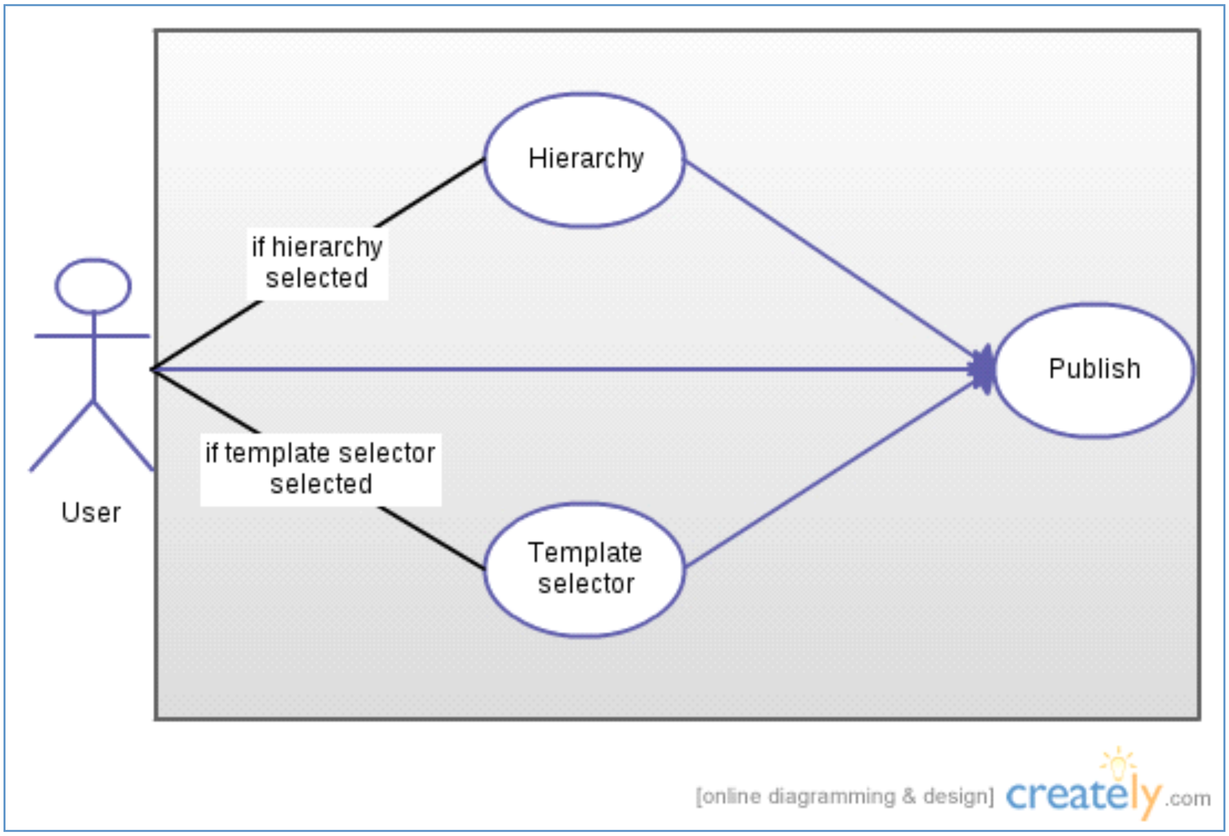
Hierarchy function:

This function is optional function.Using this function ,there will display window where hierarchy of the slides is listed there.By using this function, customers easily list power point slides how they want.The main difference with the publish function is being optional.

Template selector function:

This function is optional function as hierarchy function. There will be some slide templates that are suggested to customer. The aim is to help customer in design of slides.

Hierarchy and Template functions are pre-publishing function.


Hierarchy
if hierarchy selected
User
Publish
if template selector selected
Template selector
[online diagramming & design] creately.com

2.3 CONSTRAINTS, ASSUMPTIONS AND DEPENDENCIES

a) Hardware limitations

Hardware has to be able to run Microsoft PowerPoint.

b) Interfaces to other applications

Microsoft PowerPoint is required in order to run Plug-in Application.

c) Higher-order language requirements

Application will be implemented in C# language. Therefore, Microsoft Visual Studio will be used.

d) Reliability requirements

Both form of Application (Plug-in and Windows app) will be satisfy the claims below:

- Conversion of PowerPoint file to HTML5 file format
- Suitabilityof theexported files to SCORM e-learning standards.

e) Criticality of the application

In generally, the software will be used for improvement in the education of employees. So, failure of application can cause delay in educational tasks of companies.

3 SPECIFIC REQUIREMENTS

3.1 INTERFACE REQUIREMENTS

In general, the product will have 3 different user interfaces. One is hierarchy interface, used to assign hierarchical structure to the slides. The second one is template interface, which allows user to select the best matching template for the slide. As mentioned above, those preferences are optional and user may decide not to use any of them. Finally, there will be a publishing interface which will ask user to select the PPTX file at the start, and to click publish button at the end to get the final result. Those three interfaces will be triggered by three buttons by the same name as “publish”, “hierarchy” and “templates”.

3.1.1 Hierarchy Interface

This UI of the plug-in will sit on the left side of the main Power Point interface as an extra tab once it’s triggered by the “hierarchy” button and will show all the slides of the presentation with extra buttons namely, “move right”, “move left”, “move up”, “move down” calling the functions move_right, move_left, move_up, move_down respectively.

Buttons will situate at the bottom side of the slide thumbnails and will affect those thumbnails only independent from the original slides.

3.1.2 Templates Interface

Template interface will be triggered by the “templates” button described above. It will open a new window, which contains pre-created theme templates for the currently active/selected slide. Once the desired template is selected (and previewed) a button named “select” must be clicked to finish the template selection process. Having selected the theme, the respective HTML file of the slide will be created based on this chosen template.

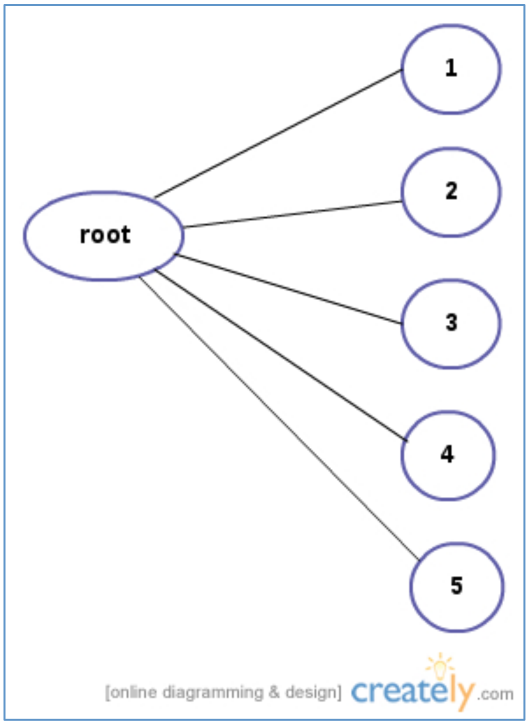
3.1.3 Publish Interface

This interface will be triggered by the “publish” button described above.There will be an address bar, a browse button, and a “publish” button which is different from the previously defined one. “Browse” button will open default Windows directory selection window and will specify the exact directory where the output will be placed. Directory will be displayed in the address bar, which can be edited even without using the “browse” button. “Publish” button will be the final action that the user will do. It will call the appropriate function which will be defined at the next sections and display the result.

3.2 FUNCTIONAL REQUIREMENTS

3.2.1. Hierarchy Editor

Hierarchy Editor, described in Hierarchy Interface (3.1.1.), aims to create a hierarchy tree for the slides. An example of hierarchy tree is shown below:



In hierarchy tree, slides are ordered as a node. All hierarchy design is performed in this tree. Slides are not affected from this design.

For hierarchy design, editor has move buttons. Besides, open button is used to open hierarchy editor tab.The output of hierarchy editor is a data that contains tree structure. The buttons of editor are explained below:

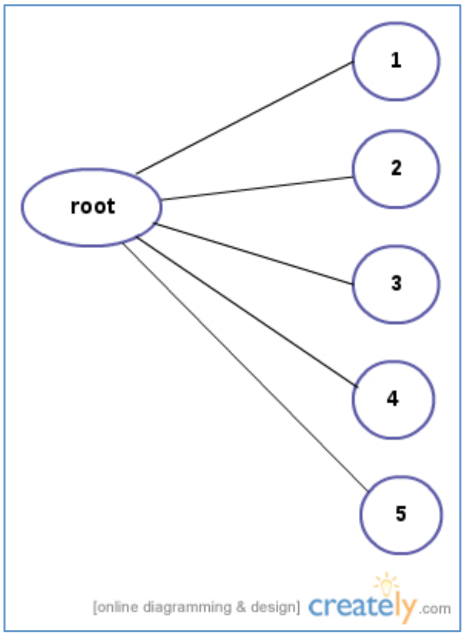
1. **Move Right/ Left Buttons:**

These buttons is used to create hierarchical relation between slide nodes in the tree.

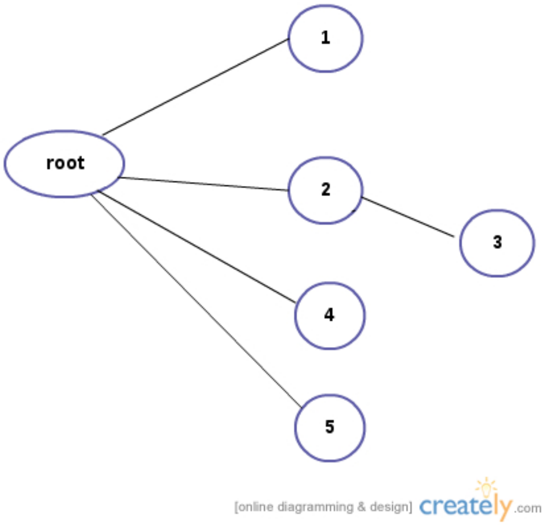
- Move Right:

This button moves the selected slide node to its previoussibling’s child.

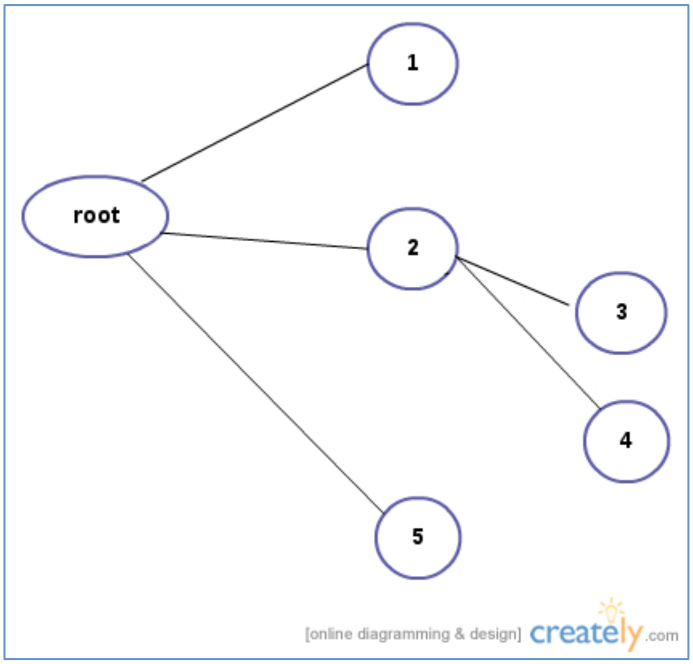
Example:Initial Tree:



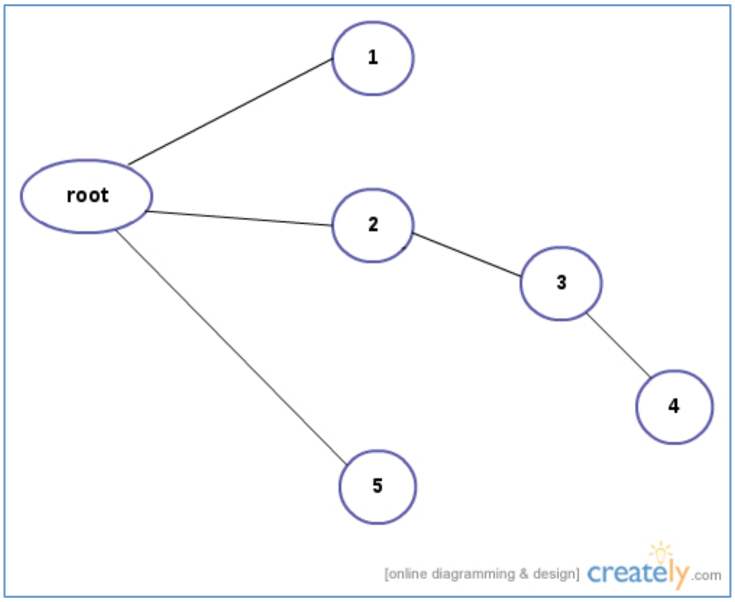
If we choose 3 and press move right, slide 3 moves to the child position of slide 2.



If we choose slide 4 and press move right, the result is:



After move right is pressed for 4 again, the tree is:

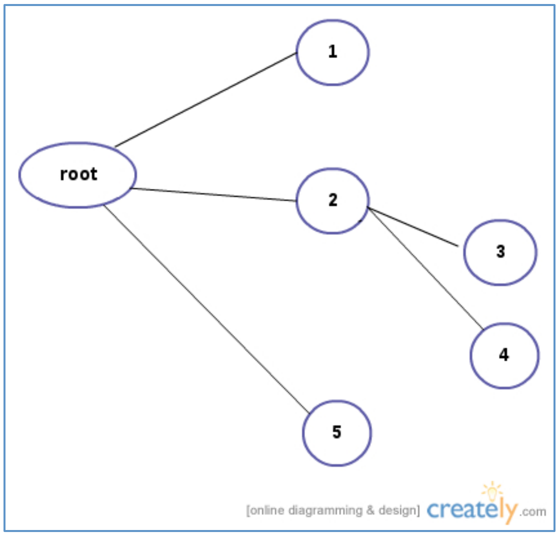


- Move Left:

Reverse of move right button. Move slide node to an upper level of tree.

Example:

When move left is pressed for slide 4 in the tree above, the result:



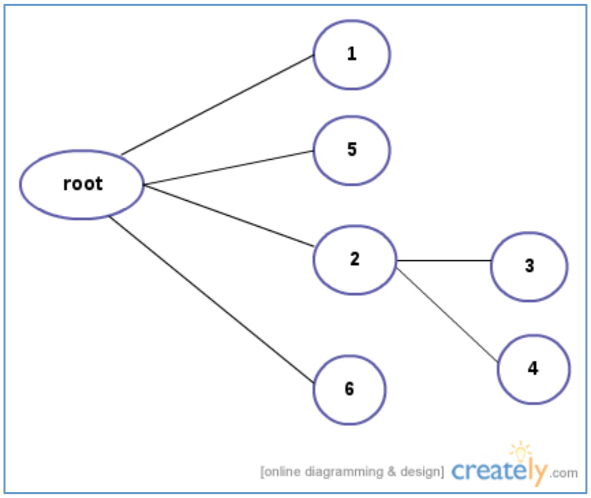
2. **Move Up /Down**

These buttons is used for changing the order of the sibling nodes.

- **Move Up:**

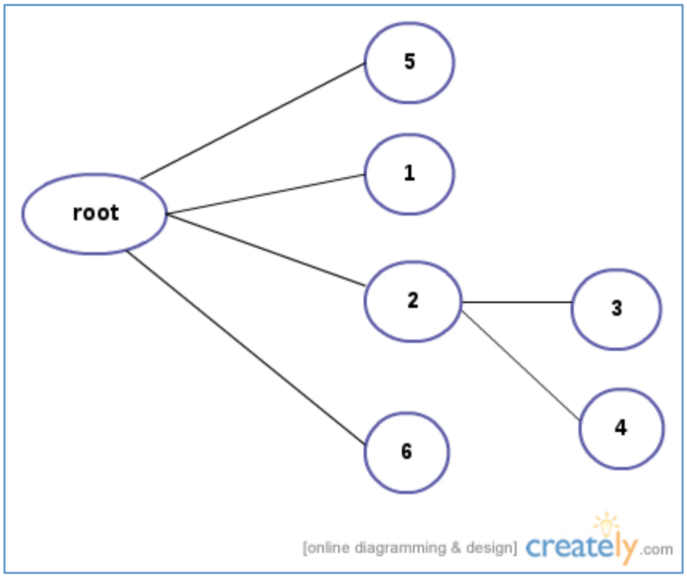
  Example:

  If we choose 5 and press move up within the slides above:



- Move down

  Press move down on 1:



3.2.2. Template Editor

Template editor contains some buttons:

1. Open:

   Opens template editor tab.
2. Convert:

   Opens a preview window which contains the result of conversion of current slide to chosen template.
3. Select:

   Selects the result slide for the HTML5 output file. There will be no change on the presentation.
4. Cancel:

   Cancels conversion and closes the preview window.

3.2.3 Publish Function

This Function exports HTML5 file within SCORM format taking hierarchy and template editor into consideration:

1. Hierarchy editor output:
   HTML5 file will have hierarchy structure. If no change made in hierarchy editor tab, output will contain tree with slide order of presentation.

2. Template output:
   If a slide has template editor output, this output will be used instead of original slide.

3.3. NONFUNCTIONAL REQUIREMENTS

3.3.1 Performance Requirements

There is no strict performance requirements for the product to work properly. However, obviously, Microsoft Office 2007 or later and Windows OS is needed.

3.3.2 Design Constraints

We will use C# programming language with .NET Framework in Microsoft Visual Studio environment. Our product will be used for the commercial purposes.

4. DATA DESCRIPTION

This section describes what are the product’s data objects, such that which kind of elements developers are dealing with. In addition, the way these elements are related to one another will be displayed by two ERD schemas.

4.1. DATA OBJECTS

In this subsection, data elements (PPTX, PresentationML and HTML) are described in detail under the aspect of how they will be used through the development process of the product.

4.1.1. PPTX

PPTX is an MS Office 2007 extension of the Power Point. Our product will be able to convert PPTX files to HTML format on Windows XP, Vista and 7 OS only. PPTX files are open to public as an XML combination of files. This collection has a special name, named by Microsoft as PresentationML.

4.1.2. PresentationML

The document structure of a PresentationML document consists of the <presentation> (Presentation) element that contains <sldMaster> (Slide Master), <sldLayout> (Slide Layout), <sld> (Slide), and <theme> (Theme) elements that reference the slides in the presentation. (The Theme element is the root element of the DrawingMLTheme part.) These elements are the minimum elements required for a valid presentation document.

In addition, a presentation document might contain <notes> (Notes Slide), <handoutMaster> (Handout Master), <sp> (Shape), <pic> (Picture), <tbl> (Table), and other slide-related elements. (Table elements are defined in the DrawingML schema.)

Other features that a PresentationML document can contain include the following: animation, audio, video, and transitions between slides.

A PresentationML document is not stored as one large body in a single part. Instead, the elements that implement certain groupings of functionality are stored in separate parts. For example, all comments in a document are stored in one comment part, while each slide has its own part. A separate XML file is created for each slide.

Important Presentation Parts

Presentation Part

A PresentationML package's main part starts with a <presentation> root element. That element contains a presentation, which, in turn, refers to a slide list, a slide master list, a notes master list, and a handout master list. The slide list refers to all of the slides in the presentation. The slide master list refers to the entire set of slide masters used in the presentation. The notes master contains information about the formatting of notes pages. The handout master describes how a handout looks. (A handout is a printed set of slides that can be handed out to an audience for future reference.)

Presentation Properties Part

The root element of the Presentation Properties part is the <presentationPr> element.

An instance of this part type contains all the presentation's properties.

A package shall contain exactly one Presentation Properties part, and that part shall be the target of an implicit relationship from the Presentation part.

Slide Master Part

The root element of the Slide Master part is the <sldMaster> element.

An instance of this part type contains the master definition of formatting, text, and objects that appear on each slide in the presentation that is derived from this slide master.

A package shall contain one or more Slide Master parts, each of which shall be the target of an explicit relationship from the Presentation part, as well as an implicit relationship from any Slide Layout part where that slide layout is defined based on this slide master. Each can optionally be the target of a relationship in a Slide Layout part as well.

Slide Layout Part

The root element of the Slide Layout part is the <sldLayout> element.

An instance of this part type contains the definition for a slide layout template for this presentation. This template defines the default appearance and positioning of drawing objects on this slide type when it is created.

A package shall contain one or more Slide Layout parts, and each of those parts shall be the target of an explicit relationship in the Slide Master part, as well as an implicit relationship from each of the Slide parts associated with this slide layout.

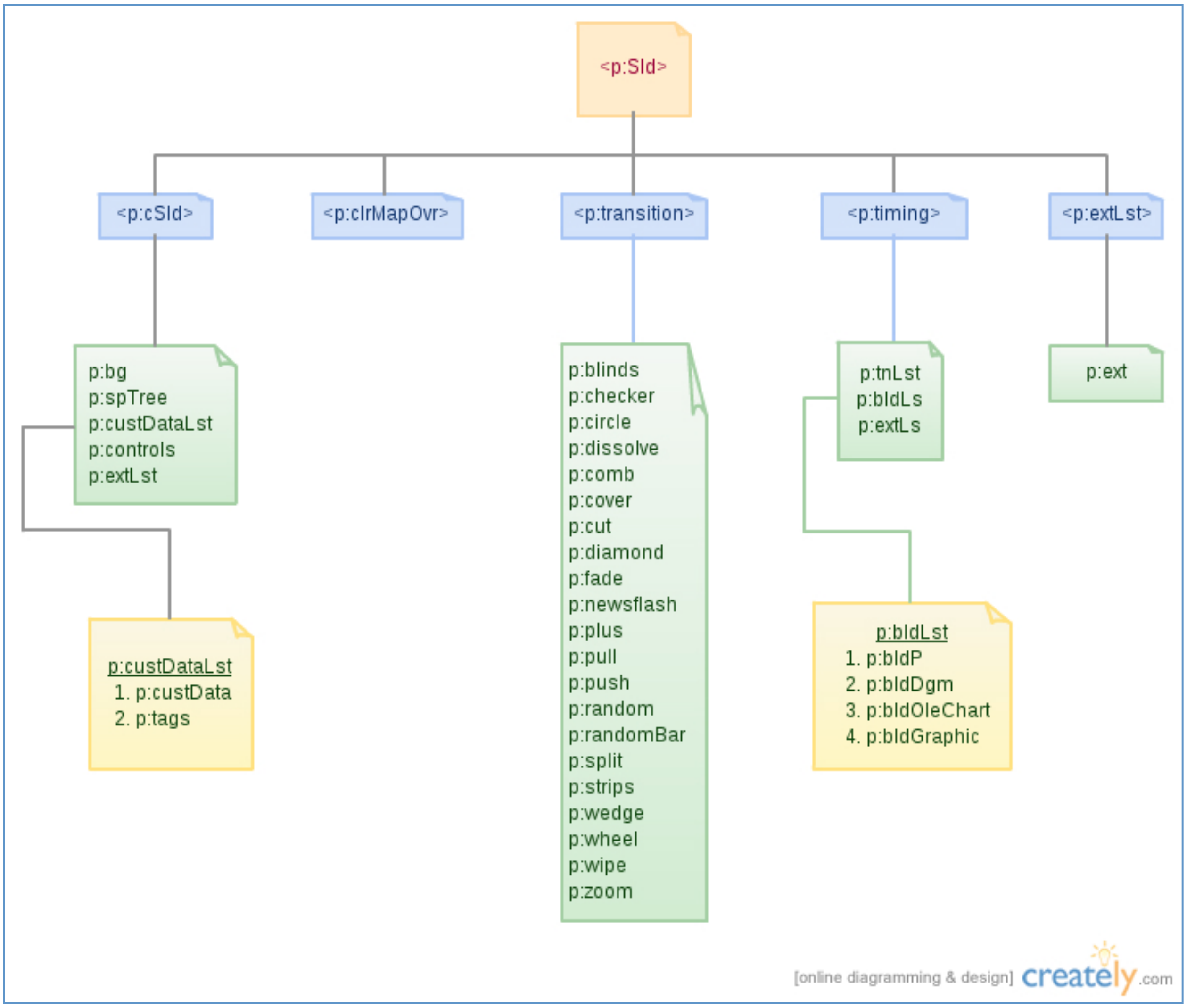
Slide Part

The root element of the Slide part is the <sld> element.

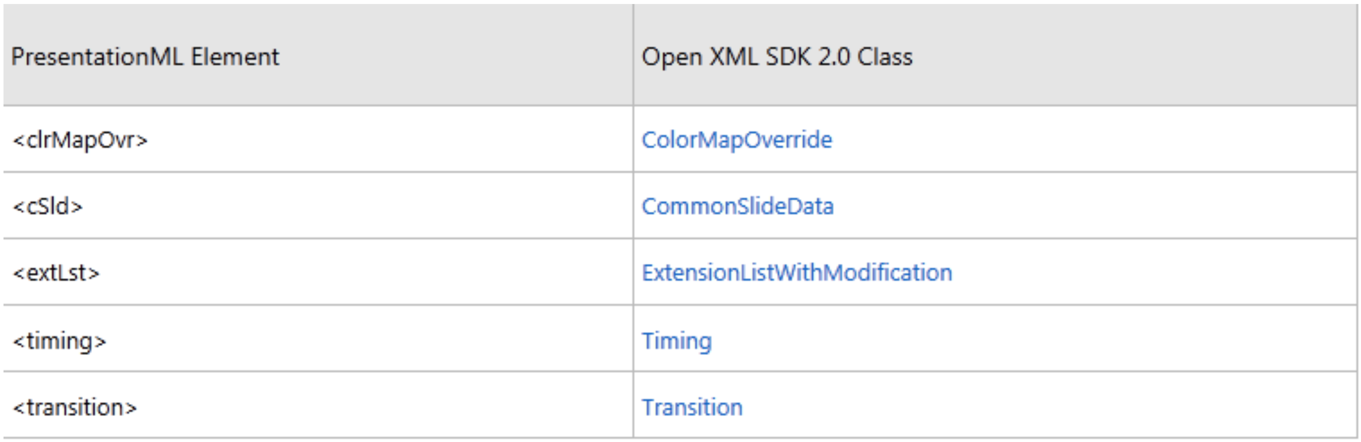
As well as text and graphics, each slide can contain comments and notes, can have a layout, and can be part of one or more custom presentations. A comment is an annotation intended for the person maintaining the presentation slide deck. A note is a reminder or piece of text intended for the presenter or the audience.

A Slide part contains the contents of a single slide.A package shall contain one Slide part per slide, and each of those parts shall be the target of an explicit relationship from the Presentation part.

The following tree lists the child elements of the <sld> element used when working with presentation slides.



The following table lists the child elements of the <sld> element used when working with presentation slides and the Open XML SDK 2.0 classes that correspond to them.



PresentationML Element	Open XML SDK 2.0 Class
<clrMapOvr>	ColorMapOverride
<cSld>	CommonSlideData
<extLst>	ExtensionListWithModification
<timing>	Timing
<transition>	Transition

CommonSlideDataClass :

This element specifies a container for slide information that is relevant to all of the slide types. All slides share a common set of properties that is independent of the slide type; the description of these properties for any particular slide is stored within the slide's cSld container. Slide data specific to the slide type indicated by the parent element is stored elsewhere.

Timing :

This element specifies the timing information for handling all animations and timed events within the corresponding slide. This information is tracked via time nodes within the timing element. More information on the specifics of these time nodes and how they are to be defined can be found within the Animation section of the PresentationML framework.

Transition:

This element specifies the kind of slide transition that should be used to transition to the current slide from the previous slide. That is, the transition information is stored on the slide that appears after the transition is complete.

Theme Part

The root element of the Theme part is the <officeStyleSheet> element.

An instance of this part type contains information about a document's theme, which is a combination of color scheme, font scheme, and format scheme (the latter also being referred to as effects). For a PresentationML document, the choice of theme affects the formatting of slides, handouts, and notes via the associated master, among other things.

A PresentationML package shall contain zero or one Theme part per Handout Master, Notes Master, Slide Master or Presentation part via an implicit relationship.

Notes Master Part

The root element of the Notes Master part is the <notesMaster> element.

An instance of this part type contains information about the content and formatting of all notes pages.

A package shall contain at most one Notes Master part, and that part shall be the target of an implicit relationship from the Notes Slide part, as well as an explicit relationship from the Presentation part.

Notes Slide Part

The root element of the Notes Slide part is the <notes> element.

An instance of this part type contains the notes for a single slide.

A package shall contain one Notes Slide part for each slide that contains notes. If they exist, those parts shall each be the target of an implicit relationship from the Slide part.

Handout Master Part

The root element of the Handout Master part is the <handoutMaster> element.

An instance of this part type contains the look, position, and size of the slides, notes, header and footer text, date, or page number on the presentation's handout.

A package shall contain at most one Handout Master part, and it shall be the target of an explicit relationship from the Presentation part.

Comments Part

The root element of the Comments part is the <cmLst> element.

An instance of this part type contains the comments for a single slide. Each comment is tied to its author via an author-ID. Each comment's index number and author-ID combination are unique.

A package shall contain one Comments part for each slide containing one or more comments, and each of those parts shall be the target of an implicit relationship from its corresponding Slide part.

Comments Author Part

The root element of the Comments Author part is the <cmAuthorLst> element.

An instance of this part type contains information about each author who has added a comment to the document. That information includes the author's name, initials, a unique author-ID, a last-comment-index-used count, and a display color. (The color can be used when displaying comments to distinguish comments from different authors.)

A package shall contain at most one Comment Authors part. If it exists, that part shall be the target of an implicit relationship from the Presentation part.

The Structure of a Minimum Presentation File

A minimum presentation file consists of a presentation part, represented by the file presentation.xml, as well as a presentation properties part (presProps.xml), a slide master part (slideMaster.xml), a slide layout part (slideLayout.xml), and a theme part (theme.xml). One or more slide parts (slide.xml) are optional.

The packaging structure of a presentation document contains several references between the parts, including some circular references. For example, slide layouts reference slide masters, and slide masters reference slide layouts.

4.1.3. HTML5

HTML5 is the 5th version of the web language HTML. It will be used as the output format of our product, especially due to the fact that it has a high potential to be used in the future. Detailed information can be found at: http://en.wikipedia.org/wiki/HTML54.2.RELATIONSHIPS

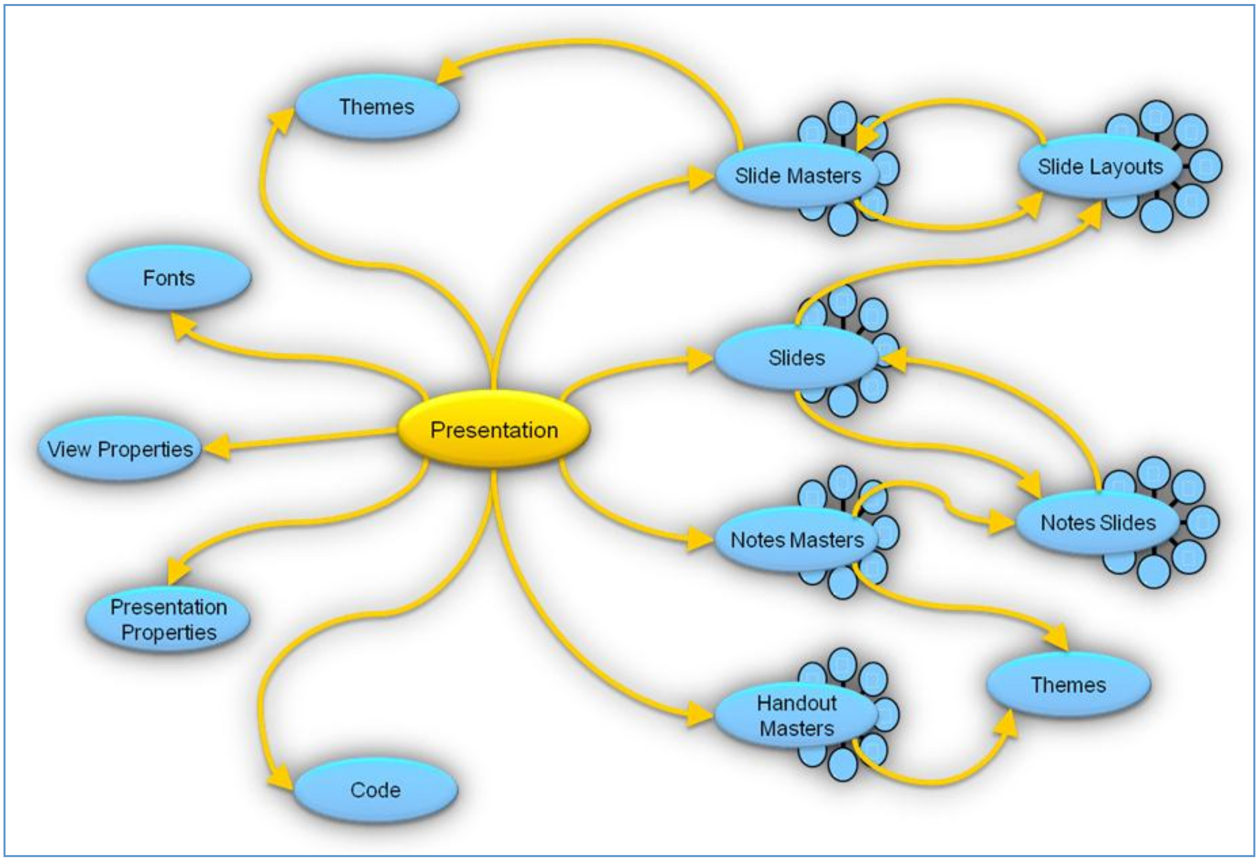
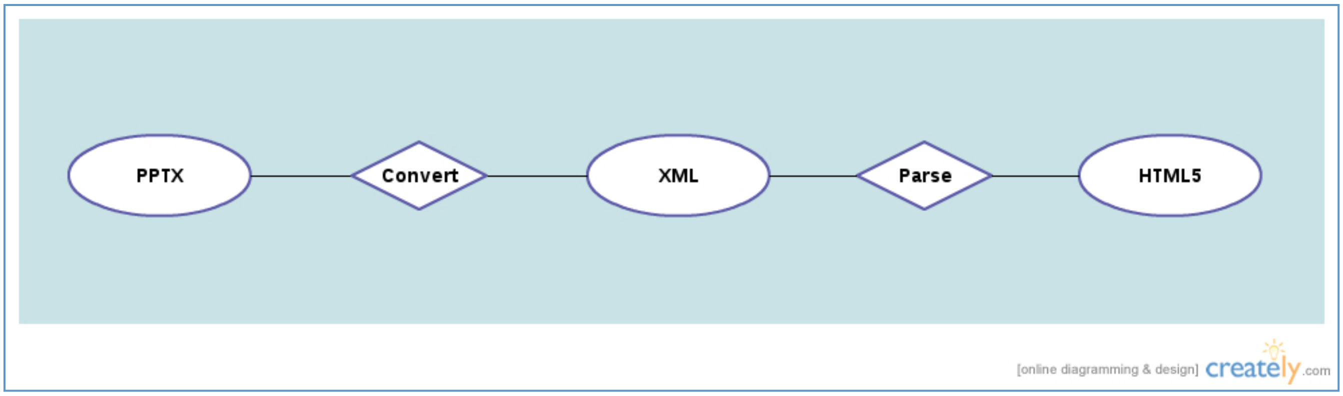
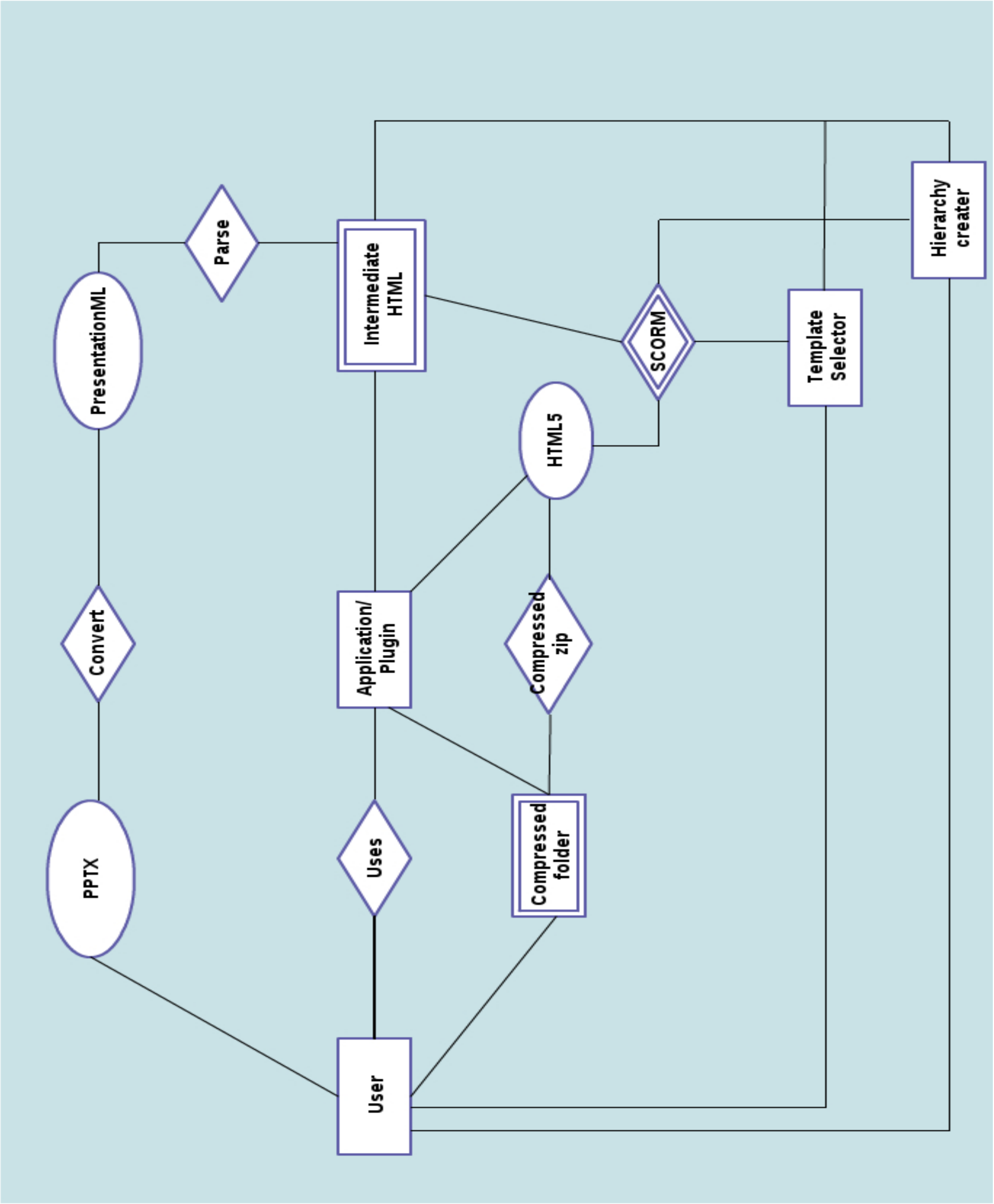


Figure. Elements of a PresentationML file

4.2 RELATIONSHIPS



4.3. COMPLETE DATA MODEL



4.4. DATA DICTIONARY

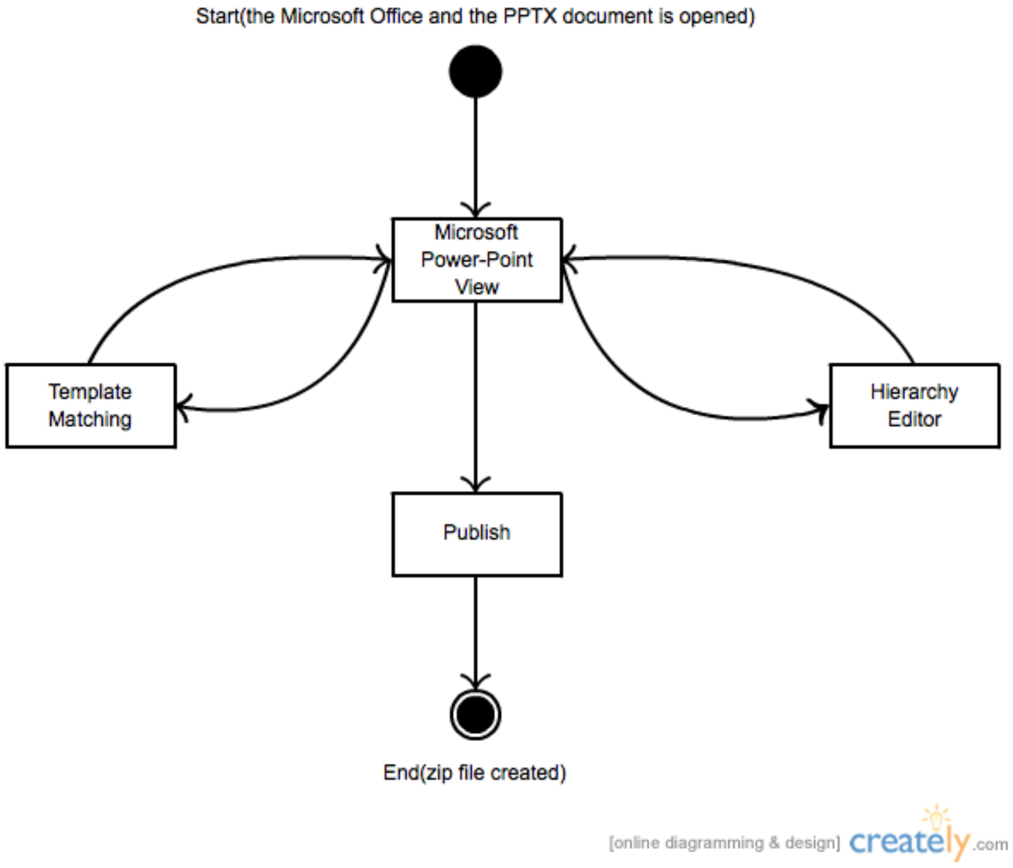
Data/ERD Element	Description
User	User is the costumer who wants to convert PPTX files into HTML5 format
PPTX	PPTX is the Microsoft Power Point 2007 (or later) files
PresentationML	PresentationML is an XML format specially designed for PPTX files by MS
HTML5	HTML5 is a web language which will be the output of the product
Application/Plugin	The product which will be in two forms: Windows app and Power Point plug-in
Hierarchy creator	A user interface which asks the user to manually create a hierarchy between slides
Template selector	A user interface which asks the user to manually create a hierarchy between slides
Use	The action between user and the product
PPTX to XML converter	An intermediate step to retrieve PresentationML data from PPTX file
XML parser	A function which will provide the detailed structure of the slide
SCORM	SCORM is an education standard which allows the educator to inspect the training process
Compress	A function which will compress a folder of HTML files into a single .zip file
Intermediate HTML step	This is the step where HTML files are partly created without SCORM format applied
Compressed folder	The output of the main program, with .zip extension

5 BEHAVIORAL MODEL AND DESCRIPTION

5.1 DESCRIPTION FOR SOFTWARE BEHAVIOR

When the user opens the PPTX document, (assuming that it Microsoft Power-Point has our plug-in installed), he/she will be able to choose hierarchical structure and best matching theme template for each slide. Hierarchical structure will provide categorization of slides in a tree-like method, and template matching will help the newly created HTML5 files to be more accurate. Hierarcy and template selection stages are continuous until the publishing process. Our product is going to produce one .html file for each Power-Point slide. Training documents will obey the rules of the SCORM e-learning standards. Finally, all HTML files will be compressed into a single .zip file, because it is a rule from SCORM standards.

5.2 STATE TRANSITION DIAGRAM


Start(the Microsoft Office and the PPTX document is opened)
Microsoft Power-Point View
Template Matching
Hierarchy Editor
Publish
End(zip file created)
[online diagramming & design] creately.com

6 PLANNING

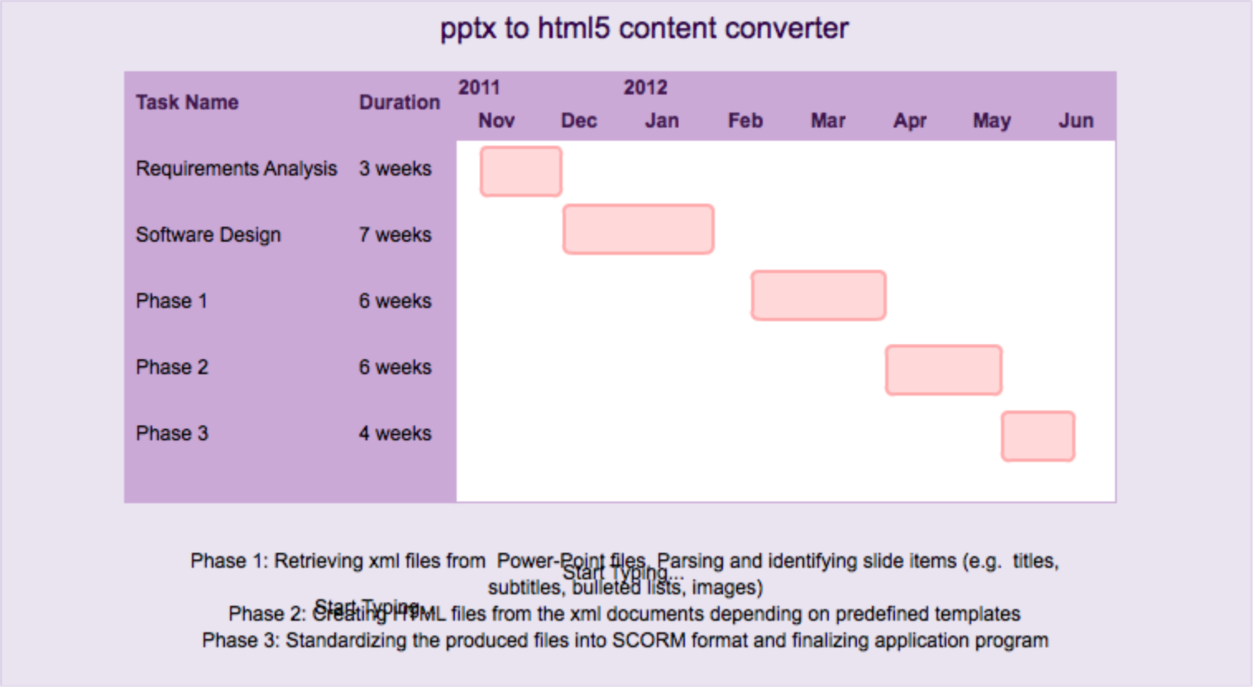
6.1 TEAM STRUCTURE

The team consists of four members:ShamilFarajullayev, RustamHashimov, Ömer Baykal, NahidHamidli.

Members of the team have no strictly specific contribution areas. Everyone is responsible with each phase of the project. The team has a collaborative decision mechanism. Every week there is at least one meeting to discuss what have been done and what is to be done.

6.2 ESTIMATION

Below there is a Gantt Chart of the overview of the project's development process. Until end of the first semester, we intend to finish the documentation of the project with a prototype to be demonstrated in our presentation. Throughout the second semester, we will implement the project. We have decided three milestones for us. (Details are below)



pptx to html5 content converter

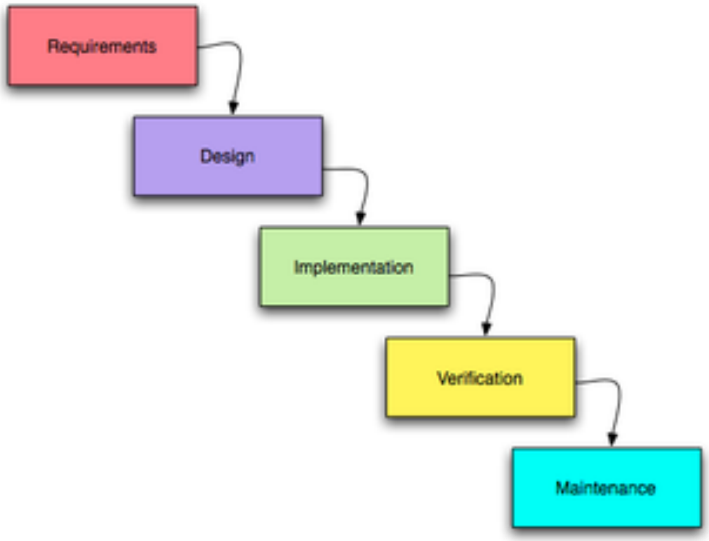
Task Name	Duration
Requirements Analysis	3 weeks
Software Design	7 weeks
Phase 1	6 weeks
Phase 2	6 weeks
Phase 3	4 weeks

Phase 1: Retrieving xml files from Power-Point files, Parsing and identifying slide items (e.g. titles, subtitles, bulleted lists, images)
Phase 2: Creating HTML files from the xml documents depending on predefined templates
Phase 3: Standardizing the produced files into SCORM format and finalizing application program

[online diagramming & design] creately.com

6.3 PROCESS MODEL

We will first do, actually we did right now, requirements analysis. Then design of the project and finally the implementation. We believe in that Waterfall Process Model best fits to our requisites.



Waterfall Process Model diagram[4]

7 CONCLUSION

Software Requirement Specification is an important part of the process of project development. Moreover, it is a prerequisite for creating the following design documentation. In this document, we have provided the information about general product description, data elements that the product deals with, specific requirements like product's interfaces and the functions will be implemented. In addition general behaviour of the product has been explained in order to make it easy for the customer to understand to product's usage clearly. Furthermore, structure and working plan of the developing team is given. This document has been created through the help of various researches and depending on the demands of the customer. However, some little specifications are prone to be changed in the future.