How to Create Requirements Spec. in Minutes?

Written Date : June 02, 2015

Writing software requirements spec. is typically cumbersome and time consuming because much effort has to be spent on making diagrams and model specifications reflected in the document. Maintaining a specification is difficult as well because any small change in model entails changes over the document and, very often, changes made in model are hardly traceable.

Fortunately, Doc. Composer is there to help you get the job done efficiently and reliably. Doc. Composer processes special pieces of text called Doc Fields in your document, and substitute them with actual model details. You don't need to perform any diagram exporting or copy-and-paste. All you need to do is to write a short query statement in your document. The rest will be taken care by Doc. Composer.

Overview of this Tutorial

In this tutorial you will learn how to produce a simple requirements spec. with the help of Doc. Composer. Here is an outline of what will be covered in this tutorial:

1. Write the Doc Base in Microsoft Word
2. Open the Doc Base in Doc. Composer
3. Generate Word file from Doc. Composer

In order to complete this tutorial, you must have the following software installed:

- Visual Paradigm 13.0
- Microsoft Word 2007 or higher

Writing Doc Base

A Doc Base is a semi-completed version of your project documentation or report. It contains only background information like introduction, project vision and objective. The design details are leave empty and be filled by Doc. Composer. Doc Base provides a base for documentation production. You provide such a base to Doc. Composer and then Doc. Composer fill the empty parts for you by embedding model data extracted from your Visual Paradigm project into your document.

To write a Doc Base:

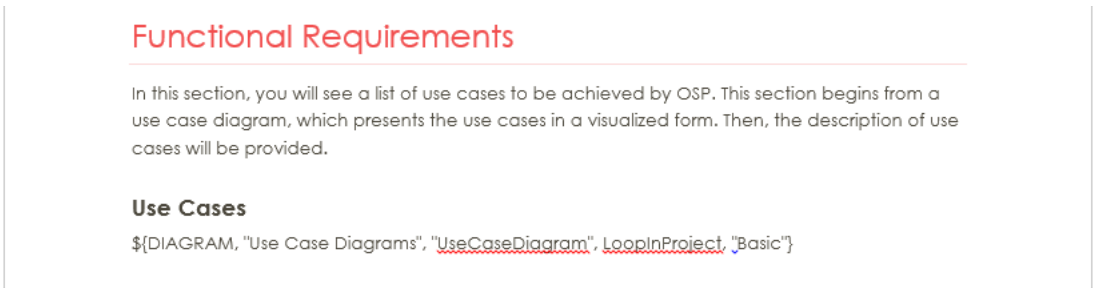
1. Download OSP-Software-Requirements-Specification.docx. You can also find this file at the bottom of this tutorial.

2. Open the downloaded file in Microsoft Word and take a look. This document is a simple, incomplete Doc Base of software requirements specification. It contains a cover page, a table of contents, a table of revision history, and a page of background information like the introduction and project description.

What you need to do now is to complete the specification by writing the necessary Doc Fields to retrieve data from a Visual Paradigm project.

3. Open Page 5 of the document. Under the **Functional Requirements** section there are three sub-sections - **Use Cases**, **Actors** and **User Stories**. Let's fill them in one by one. Now, enter the following text under the header **Use Cases**:

```
&#36;{DIAGRAM, "Use Case Diagrams", "UseCaseDiagram",
LoopInProject, "Basic"}
```



Functional Requirements

In this section, you will see a list of use cases to be achieved by OSP. This section begins from a use case diagram, which presents the use cases in a visualized form. Then, the description of use cases will be provided.

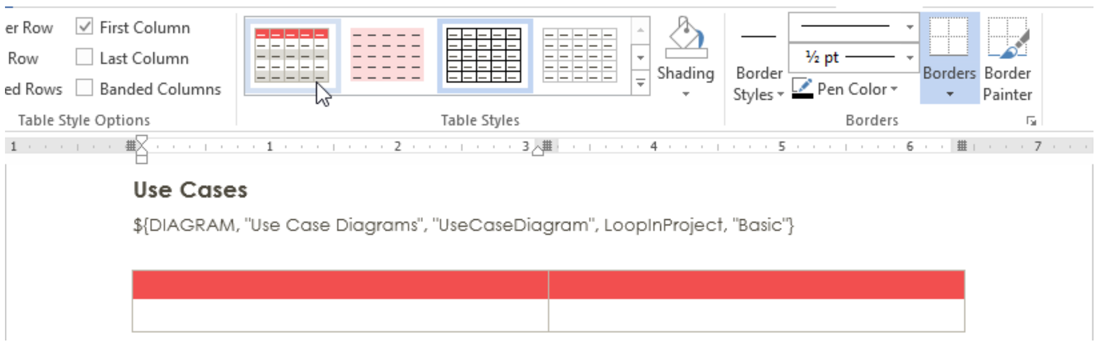
Use Cases

${DIAGRAM, "Use Case Diagrams", "UseCaseDiagram", LoopInProject, "Basic"}

What you just entered is what we called a Doc Field. A Doc Field is a special piece of text within a Doc Base. Doc Fields will be replaced by your actual project content when being processed by Doc. Composer during document generation.
A typical Doc Field is written in this format: **&#36;{FIELD_NAME, list_of_parameters}**. Below is a description of the different parts of this Doc Field:
&#36;{DIAGRAM,...} - This is a &#36;{DIAGRAM} field for querying diagram details.
"Use Case Diagram" - Name of this Doc Field.
"UseCaseDiagram" - The type of diagram to query.
LoopInProject - To query diagrams, in this case use case diagrams, from project.
"Basic" - What to show there when a diagram is found. "Basic" means to output content base on the "Basic" element template. If you want to show the name of diagram you will replace "Basic" with "PROPERTY=name".

4. Create a new line by pressing **Enter**.

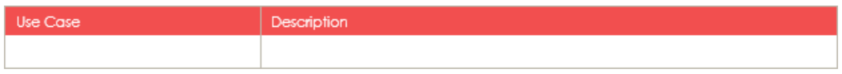
5. Let's create a table that lists out the use cases in the project. Create a table with two rows and apply a style to it to make it looks more stylish.



Use Cases

${DIAGRAM, "Use Case Diagrams", "UseCaseDiagram", LoopInProject, "Basic"}

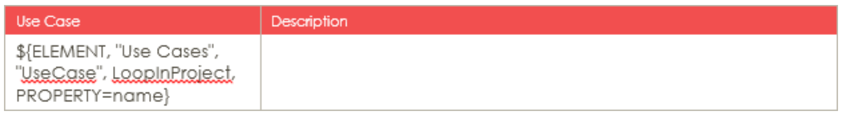
6. Enter *Use Case* and *Description* in two table cells of the first row.



Use Case	Description

7. Click on the first cell of the second row and enter the following Doc Field.

```
&#36;{ELEMENT, "Use Cases", "UseCase", LoopInProject,
PROPERTY=name}
```

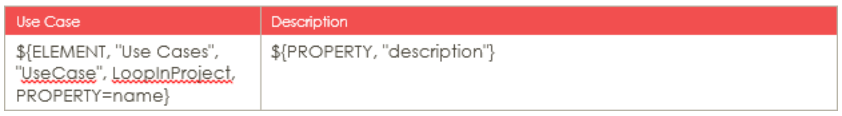


Use Case	Description
${ELEMENT, "Use Cases", "UseCase", LoopInProject, PROPERTY=name}	

The row that contains Doc Field will replicate itself to list out all the elements queried. So you can imagine, there will be a number of rows appended to this table, with each one showing the name of a specific use case.

8. In the next table cell, enter the following Doc Field:

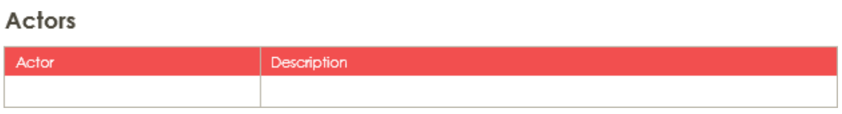
```
&#36;{PROPERTY,
"description"}
```



Use Case	Description
${ELEMENT, "Use Cases", "UseCase", LoopInProject, PROPERTY=name}	${PROPERTY, "description"}

This means that for each row of use case, show the description of that use case on the second cell.

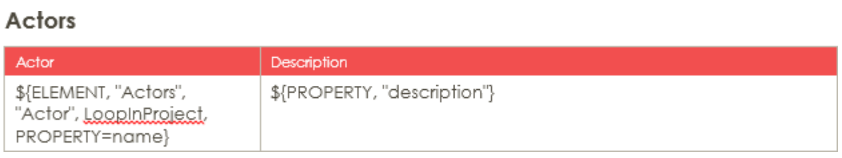
9. That's all for the **Use Cases** section. Let's move on to the **Actors** section by creating a table of actors. Similar to what you did above, create a table with two columns and name the headers *Actor* and *Description*.



Actors

Actor	Description

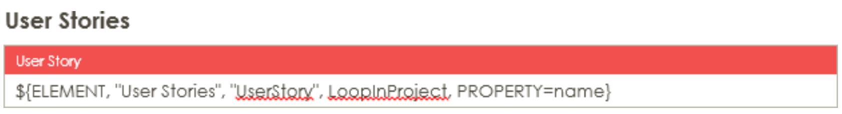
10. Add Doc Fields to the second row. Because we want to lists out actors in this table, name the Doc Field *Actors* and make it query *Actor* instead of *UseCase*.



Actors

Actor	Description
${ELEMENT, "Actors", "Actor", LoopInProject, PROPERTY=name}	${PROPERTY, "description"}

11. Let's continue to the **User Stories** section. Create a table like this:



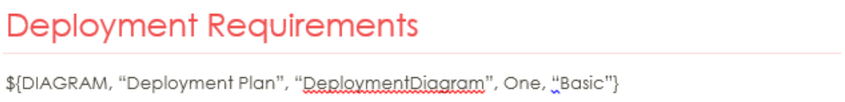
User Stories

User Story
${ELEMENT, "User Stories", "UserStory", LoopInProject, PROPERTY=name}

This table will list out all the user stories in the project.

12. Finally, let's write a Doc Field under the **Deployment Requirements** section to show a Deployment Diagram. Of course you can write a Doc Field similar to the one you wrote in step 3, but as a tutorial let's try something different. Now, write this:

```
&#36;{DIAGRAM, "Deployment Plan",
"DeploymentDiagram", One, "Basic"}
```



Deployment Requirements

${DIAGRAM, "Deployment Plan", "DeploymentDiagram", One, "Basic"}

\

Instead of querying all Deployment Diagrams in project, this Doc Field will just query one specific Deployment Diagram in project. The diagram to query will be selected by the person who use this Doc Base to produce document and we will go into detail in the next section. Note that instead of **One** you can enter **Any**, which allows the querying of multiple diagrams from a project.

13. Save the changes.

Opening Doc Base in Doc. Composer

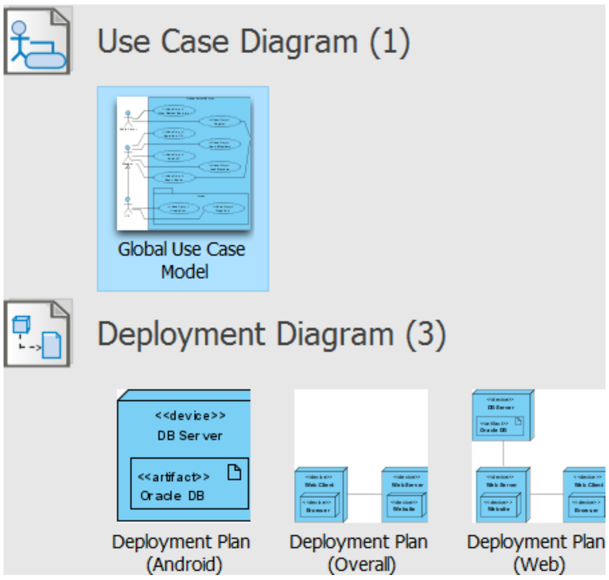
You have finished writing a Doc Base. In this section, you will open the Doc Base in Doc. Composer and touch-up the document to be generated.

1. Download Social-Networking-Service.zip. You can also find this file at the bottom of this tutorial.

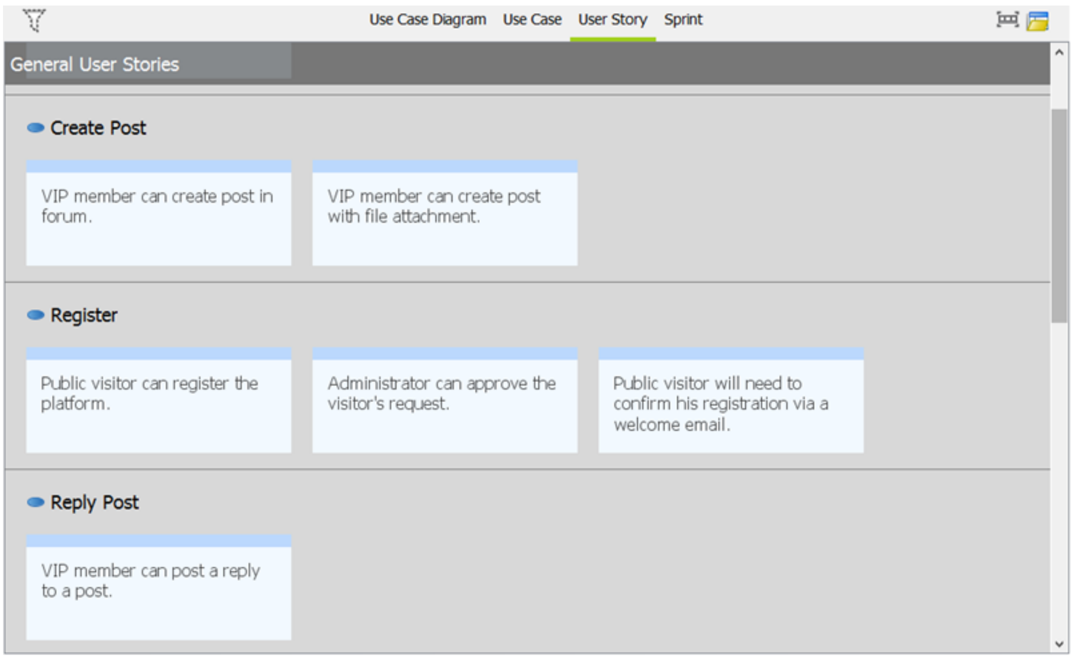
2. Unzip the downloaded file to obtain the .vpp project file.

3. Open the file in Visual Paradigm. Make sure the .vux file is placed alongside with the .vpp file

4. Let's take a quick look at what we have inside this project. Open the **Project Browser** by selecting **View > Project Browser** from the toolbar. In this project, there is one Use Case Diagram and three Deployment Diagrams.



5. This project also contains a number of user stories. Let's open UeXceler to take a look. Open UeXceler by selecting **UeXceler > UeXceler** from the application toolbar. The user stories are listed under the **User Story** tab.



General User Stories

Create Post

VIP member can create post in forum.

VIP member can create post with file attachment.

Register

Public visitor can register the platform.

Administrator can approve the visitor's request.

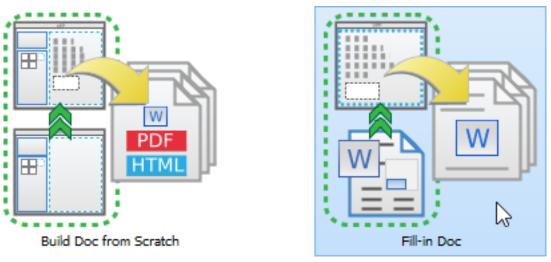
Public visitor will need to confirm his registration via a welcome email.

Reply Post

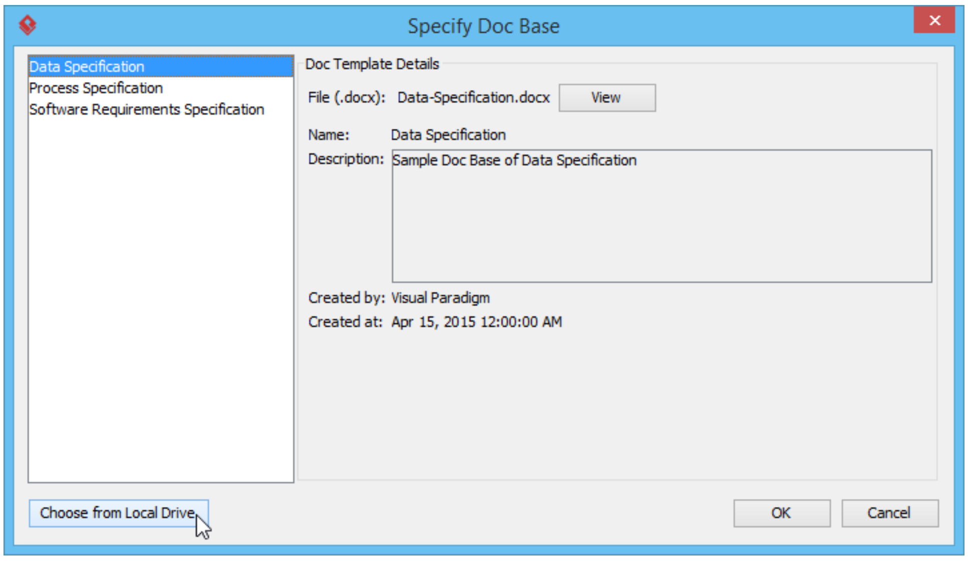
VIP member can post a reply to a post.

6. Let's create a document. Select **Tools > Doc. Composer** from the application toolbar.

7. Click **Fill-in Doc**.

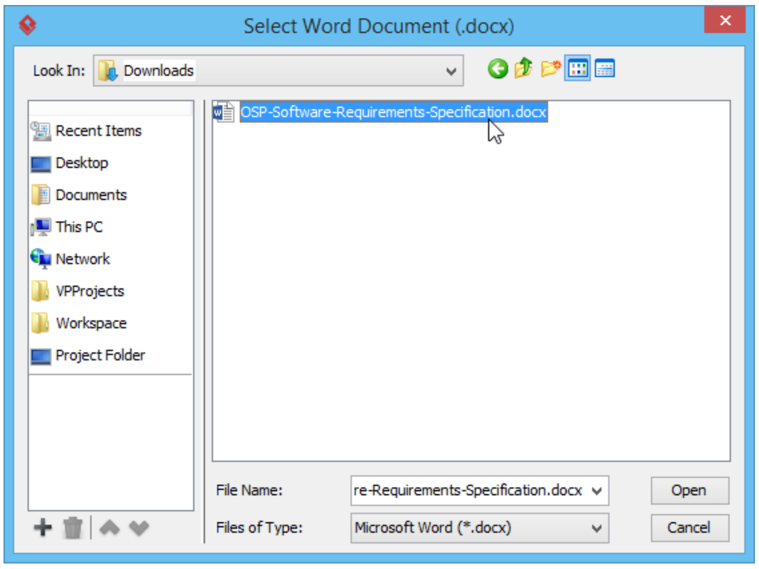


8. Specify the Doc Base to use in document production, which is a Word document file that contains both manually written content (e.g. Introduction, project scope, etc) and Doc Fields. There are two approaches from which Doc Base can be created from. One is to create from an external document file, another one is to duplicate from an existing Doc Base. For this tutorial, click **Choose from Local Drive** to select an external document file as Doc Base.

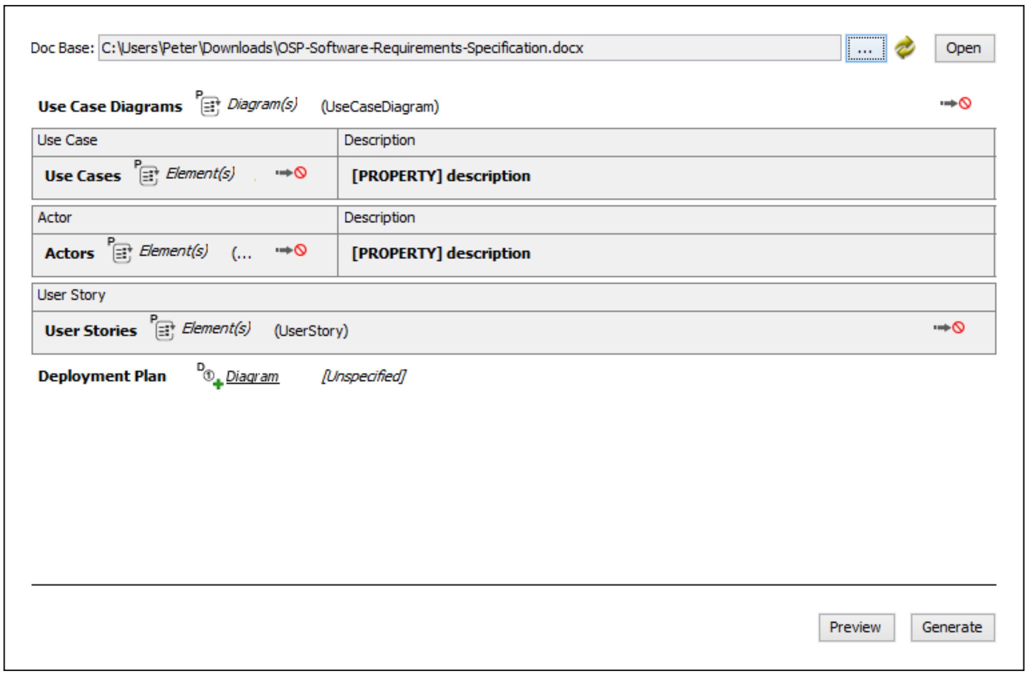


File (.docx): Data-Specification.docx
Name: Data Specification
Description: Sample Doc Base of Data Specification
Created by: Visual Paradigm
Created at: Apr 15, 2015 12:00:00 AM

9. Select the document file edited in the previous section.


Select Word Document (.docx)
Look In: Downloads
Recent Items
Desktop
Documents
This PC
Network
VPProjects
Workspace
Project Folder
OSP-Software-Requirements-Specification.docx
File Name: re-Requirements-Specification.docx
Files of Type: Microsoft Word (*.docx)
Open
Cancel

Doc. Composer analyzes your Doc Base and presents the Doc Fields that exist in your document. Your screen should look like this:



Doc Base: C:\Users\Peter\Downloads\OSP-Software-Requirements-Specification.docx

Use Case Diagrams Diagram(s) (UseCaseDiagram)

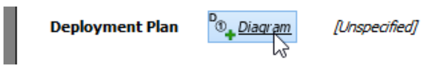
Use Case	Description
Use Cases Element(s)	[PROPERTY] description

Actor	Description
Actors Element(s) (...	[PROPERTY] description

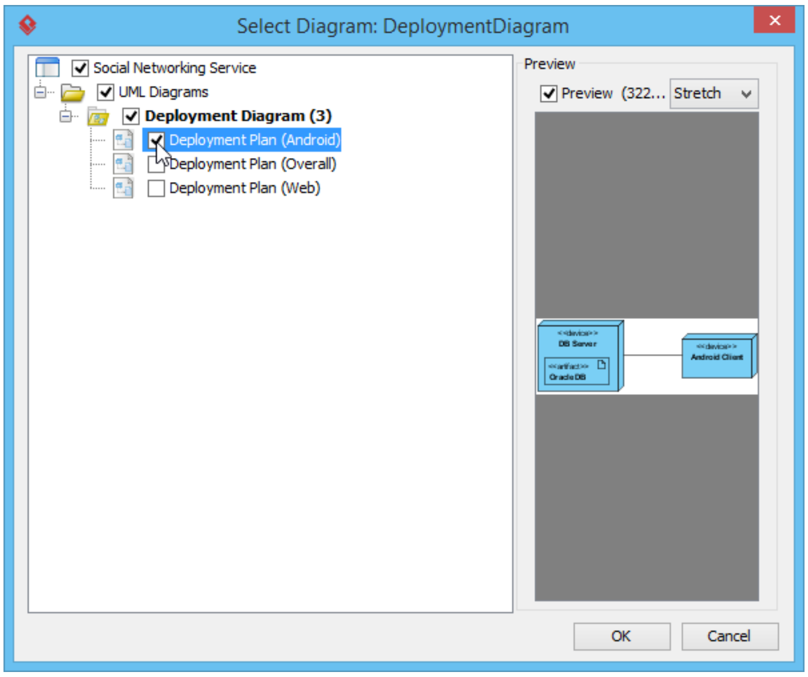
User Story
User Stories Element(s) (UserStory)

Deployment Plan Diagram [Unspecified]

10. Remember we have written a Doc Field that requires the selection of Deployment Diagram? It's time to select the diagram now. Click on the **Diagram** link next to **Deployment Plan**.

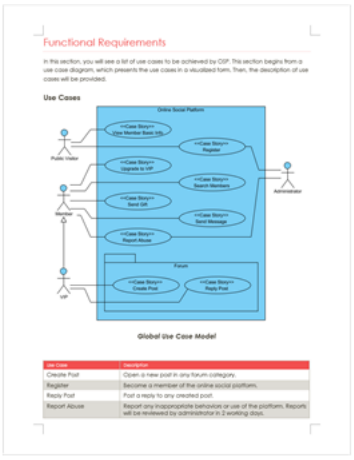


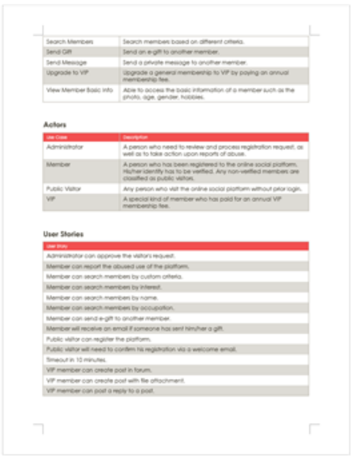
11. In the popup window, select the diagram *Deployment Plan (Android)*, and then click **OK**.

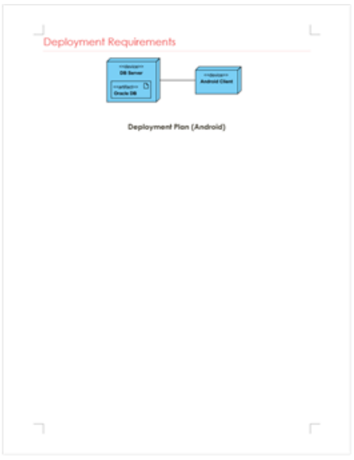


Generating Document

You are ready for producing the final document. Click **Generate** at the bottom right corner of Doc. Composer. Enter the filename of the document and confirm. A complete document file will then be generated, like this:

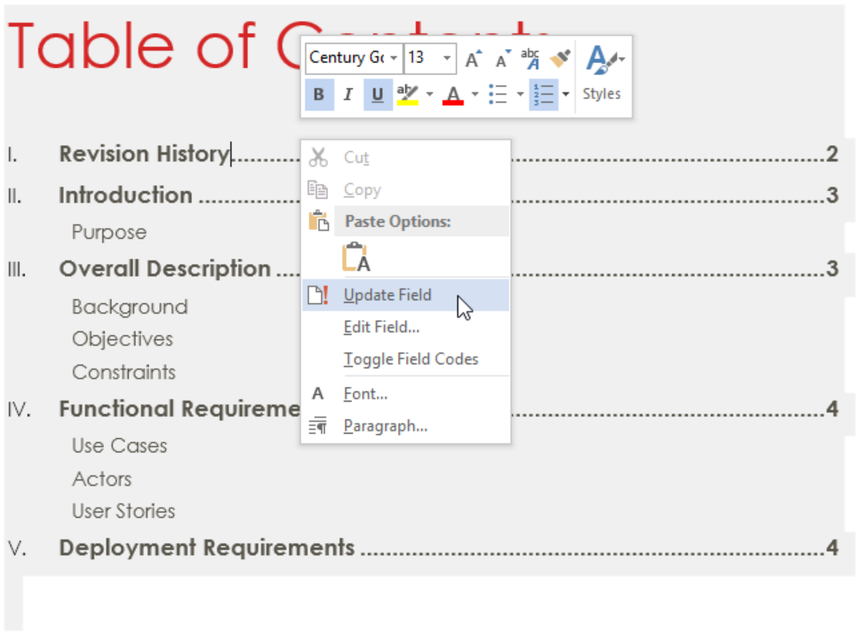






As you can see the Doc Fields are substituted by diagrams and model specification. The table rows have replicated themselves to lists out all the content required by the Doc Fields.

Finally, update the table of contents to make it reflect the real locations of generated contents.



Resources

1. Social-Networking-Service.zip
2. OSP-Software-Requirements-Specification.docx
3. OSP-Software-Requirements-Specification (Completed).docx

Related Links

- Doc. Composer Writer's Guide

Trademark Disclaimer

Microsoft Word is a registered trademark of Microsoft Corporation in the U.S.

Android is a registered trademark of Google, Inc.

Visual Paradigm

Visual Paradigm home page
(https://www.visual-paradigm.com/)

Visual Paradigm tutorials
(https://www.visual-paradigm.com/tutorials/)