

PM WORLD TODAY – FEATURED PAPER – OCTOBER 2009

Cross-Referencing Task Dependencies for Large Projects
using MS Project Macros

By Larissa Gourevitch

1. Introduction: Problem Statement

Handling a large project may significantly complicate your life as a Project Manager. You are the one taking responsibility for the outcome, and hence the one who has to keep a lot of things handy in your memory. One of these things is the need to follow up project progress and risks (obviously!) and hence to understand the interrelations of tasks and activities in the project schedule. For larger projects with complex and long schedules, answering question “which tasks will be affected if I change duration of activity # NNN?” is tricky, especially if you rely on your memory only.

Running mental what-if tests is definitely helpful in answering this type of question. However, for a project with schedule consisting of more than 1000 lines, most likely you would keep only an abbreviated “logical model” of WBS or critical path in your memory, so probability to miss an important task dependency in course of a mental test is pretty high. The situation may be aggravated when you don’t have intimate knowledge of certain areas. So, mental tests will be not as beneficial and precise. Of course, if you keep the schedule in a tool of choice, such as Microsoft Project, you can just scroll down to see which tasks get highlighted when you make a change to the properties of the specific one. This works just fine for smaller projects; for large projects most likely you will miss something since the test is based on observation only. Sometimes, you may consider a work-around, such as dividing the project into smaller – more manageable – sub-projects. This approach may complicate task of figuring out the cross-dependencies even more, because in addition you would need to handle cross-project dependencies.

In this article, the relatively simple and intuitive way of project tasks cross-referencing will be presented. This method provides 100% correct and consistent answer to the question mentioned above in the first paragraph – under the condition that you keep your project schedule in MS Projects as a Gantt chart. Another important thing about this method is that it can be used as-is, following instructions from this paper, since it was designed to be utilized by Project Management practitioners having limited to none knowledge of information technology.

2. Method

2.1. Requirements

The initial assumption was that task cross-reference report would be easily available within the tool, because MS Project is praised for providing consistent scheduling and planning services while managing a project. However, help and Internet searches returned no results, except several postings in which the question regarding tasks cross-referencing was asked rather desperately. In addition, there was no indication that other Project Management/ Project Scheduling tools can provide for tasks cross-referencing.

So Requirements for a possible solution were identified. They can be summarized as the following:

ID	Requirement/Constraint	Description	Implement by using
R.1	Requirement	Find answer to the question “Which tasks will be affected if duration of activity # NNN changes?”	Report/Macro
R.2	Requirement	Information is to be conveniently available on top, while working with schedule.	Macro
R.3	Constraint	Solution should be developed, tested, and implemented really promptly.	Report/Macro
R.4	Constraint	Tool of choice: Microsoft Project	Report/Macro

Table 1 Solution Requirements

Two options were suggested as a possible solution path:

1. to create a custom report for Microsoft Project
2. to create a Macro for Microsoft Project

As a result, to meet requirement R.2 of making information handy on top of the screen all the time, MS Project Macro was selected as a solution to pursue.

2.2. Change to MS Project User Interface

To make the information conveniently available according to Requirement R.2, minor changes to User Interface in MS Project were needed. It was decided that insertion of a column, which will contain the list of fields referencing given ones, will do the trick.

Steps to add custom column to the Gantt Chart (MS Project):

- a. Bring your project to Gantt Chart view.
- b. Put the cursor at the desired place for new column and press button “Insert”. “New column” dialogue will appear. Select “Text1” field (this is crucial!) and click “OK”. New column will appear in the project grid (highlighted in Fig. 1).
- c. After adding the column “Text1”, double-click on its header. Select “Customize Fields”, then “Rename” and choose new name, say “Cross-reference” - similar to the dialogue presented in the Fig.1

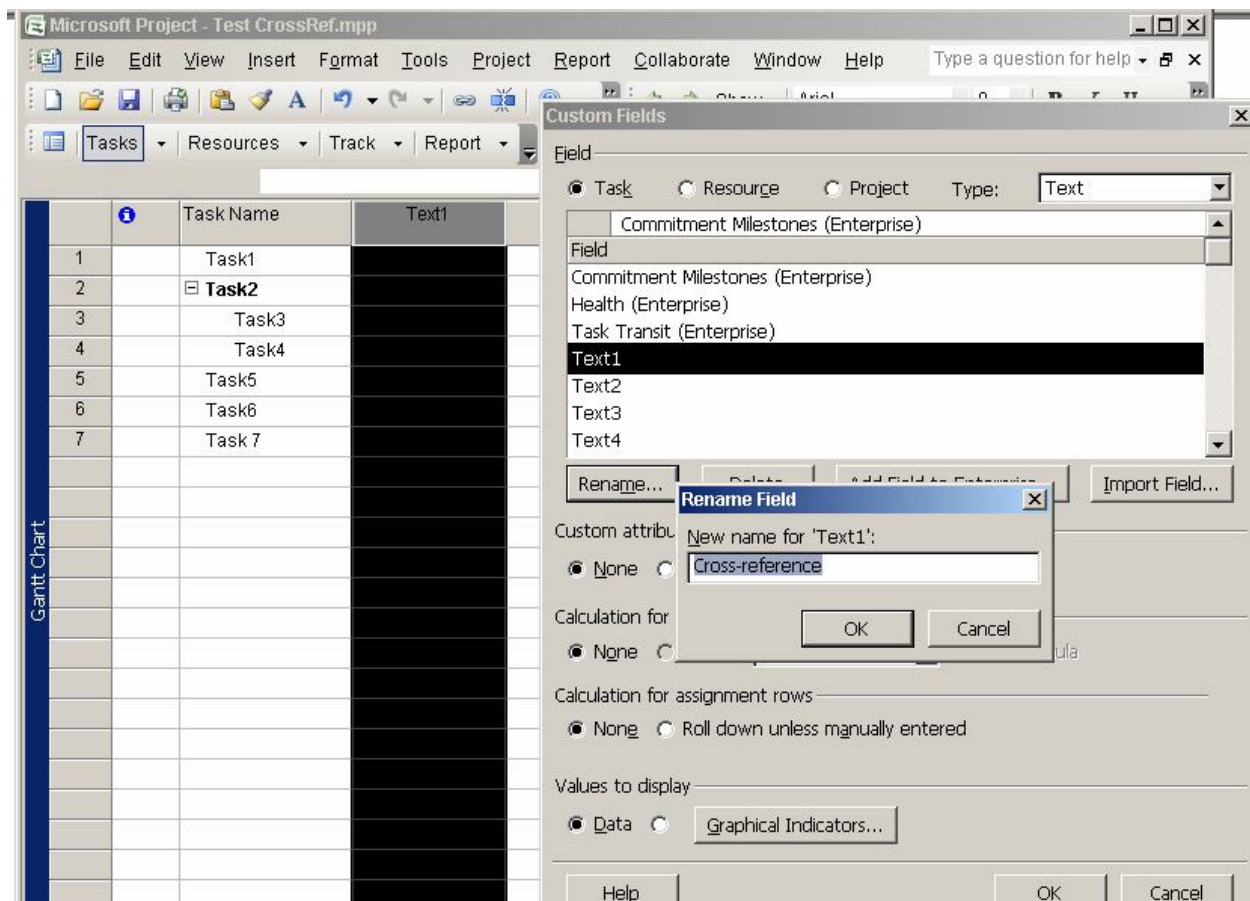


Figure 1 Adding and renaming custom field

Voila! You have the custom field. The only thing which is left is to fill it in. This will be accomplished by creating and running the Custom Macro.

2.3. Adding Custom Macro

2.3.1. Macro Design

The purpose of Macro which we are creating is to check other Tasks in your current Project if they have this current Task as a Predecessor. So outline for the Macro may look as the following:

- a. Iterate though all tasks in the project
 - i. For each of the project tasks:
 - ii. Iterate through all project tasks again
 - iii. Check if any of these tasks has our tasks as a Predecessor
 - iv. If yes, add this task ID to the list
- b. Add cumulative list of IDs from point iv above to our custom field (Text1)

Below, implementation of this pseudocode is explained in details. Please note that section 2.3.2 provides comments regarding the script code; thus if you don't feel that you can understand the coding, you can skip next section and proceed directly to section 2.3.3.

2.3.2. VBA Script Uncovered

For Microsoft project, Macros are created using VBA (Visual basic for Applications). So we will be using VBA scripting language to create our Macro according to the Design. Components of the script are explained below; code is explained using comments shown here in green.

First of all, any script is to be named. Let's name ours "CalculateDependencyReport". Besides, the script should start from variables initialization, and end up with destroying these variables:

```
Public Sub CalculateDependencyReport()  
  
Dim prj_Tasks As Tasks, pred_Tasks As Tasks  
'prj_Tasks are all Project Tasks, pred_Tasks are Predecessor tasks  
Dim one_Task As Task, dep_Task As Task, loop_Task As Task  
'single tasks we'll iterate through  
Dim one_Task_text As String  
'text buffer for the list  
  
'do staff ...  
  
'clean-up  
Set one_Task = Nothing  
Set prj_Tasks = Nothing  
Set loop_Task = Nothing  
Set pred_Tasks = Nothing  
  
End Sub
```

Code Snippet 1 Initialization of the script

Also, we have to take care of any possible errors. If an error was encountered accidentally, it should not freeze or damage your project schedule. So let's introduce this code to mitigate the risk to the schedule file:

```
'script name declaration
On Error GoTo ErrorHandler

'script body
'error handling
GoTo EndCalculation

ErrorHandler:
    MsgBox "Processing Error"

EndCalculation:

'complete the script
```

Code Snippet 2 Error Handling

Using variables declared above in Code Snippet 1, let's iterate through the project tasks:

```
Set prj_Tasks = ActiveProject.Tasks

For Each one_Task In prj_Tasks
    If Not one_Task Is Nothing Then 'consider one_Task now. We know its ID = one_Task.ID
        one_Task_text = "" 'text buffer

        'iterate via whole project again. dep_Task has potential dependency
        For Each dep_Task In prj_Tasks
            If Not dep_Task Is Nothing Then 'is not initialized so we do nothing
                Set pred_Tasks = dep_Task.PredecessorTasks
                For Each loop_Task In pred_Tasks
                    If loop_Task.ID = one_Task.ID Then 'get ID accounted for if one_Task is a Predecessor
                        If one_Task_text = "" Then 'add ID to comma-separated list
                            one_Task_text = one_Task_text & dep_Task.ID
                        Else
                            one_Task_text = one_Task_text & "," & dep_Task.ID
                        End If
                    End If
                Next loop_Task
            End If
        Next dep_Task
        'put everything into Text1 field
        one_Task.Text1 = one_Task_text
```

Code Snippet 3 Iterating through project tasks

That's all! We have written the code pieces covering ideas from Design section. Now, let's put it all together.

2.3.3. Putting All Together

Final script assembled from the pieces will look as the following:

```
Public Sub CalculateDependencyReport()
On Error GoTo ErrorHandler

Dim prj_Tasks As Tasks, pred_Tasks As Tasks
Dim one_Task As Task, dep_Task As Task, loop_Task As Task
Dim one_Task_text As String

Set prj_Tasks = ActiveProject.Tasks

For Each one_Task In prj_Tasks
  If Not one_Task Is Nothing Then
    one_Task_text = ""

    For Each dep_Task In prj_Tasks 'dep_Task has potential dependency
      If Not dep_Task Is Nothing Then
        Set pred_Tasks = dep_Task.PredecessorTasks
        For Each loop_Task In pred_Tasks
          If loop_Task.ID = one_Task.ID Then
            If one_Task_text = "" Then
              one_Task_text = one_Task_text & dep_Task.ID
            Else
              one_Task_text = one_Task_text & "," & dep_Task.ID
            End If
          End If
        Next loop_Task
      End If
    Next dep_Task

    one_Task.Text1 = one_Task_text

  End If
Next one_Task

Set one_Task = Nothing
Set prj_Tasks = Nothing
Set loop_Task = Nothing
```

```

Set pred_Tasks = Nothing
GoTo EndCalculation
ErrorHandler:
    MsgBox "Processing Error"
EndCalculation:
End Sub
    
```

Code Snippet 4 VBA Script – bring all parts together

This code has to be added to MS Project instance. To do so, in Gantt chart view, go to Tools/Macros/VB option, click on Microsoft Project Objects/This Project, as it is shown below, and paste contents of Code Snippet 4 into the window. To finalize addition of the Macro to MS Project instance, you just have to save it in VBA editor and subsequently compile, using VB buttons shown in the picture below.

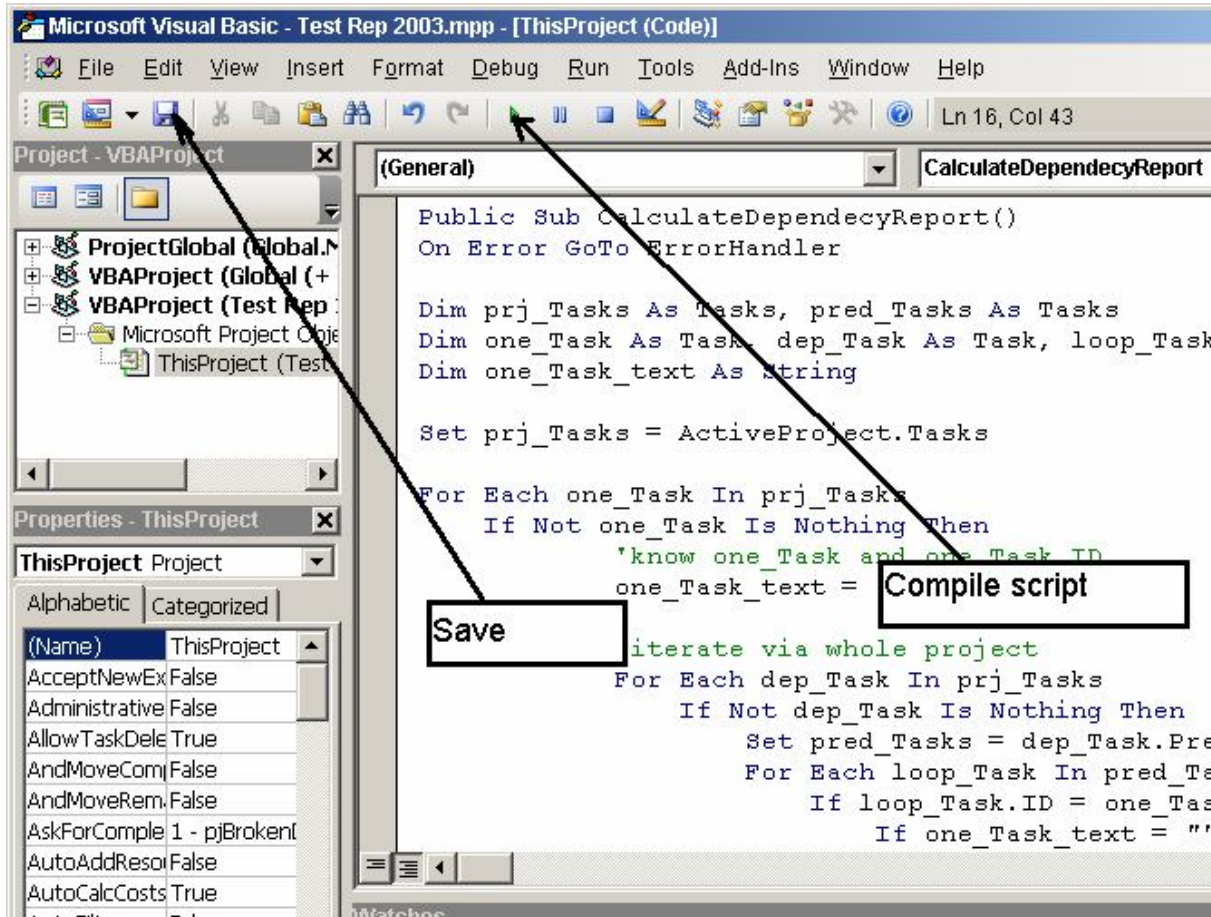


Figure 2 Saving VBA Script

2.4. Using Custom Macro

Now, your schedule in MS Project has its own little Macro which can be run on demand. To apply the Macro, you have to select Tools/Macro option in main screen of Gantt chart for your project schedule:

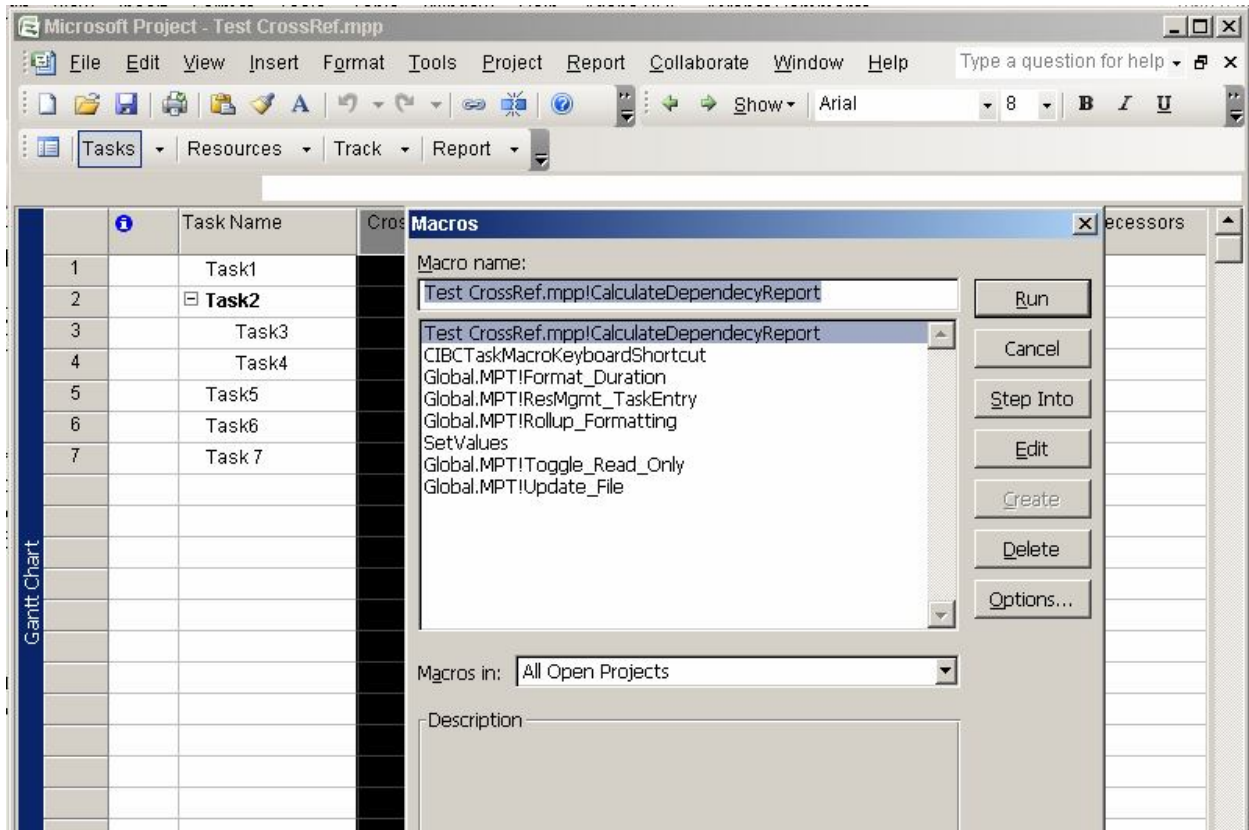


Figure 3 Running custom Macro

After highlighting CalculateDependencyReport and clicking “Run” button, cross-references will appear in new renamed custom column as it is shown in the Fig. 8 below. Wow! You will never miss a dependency again!

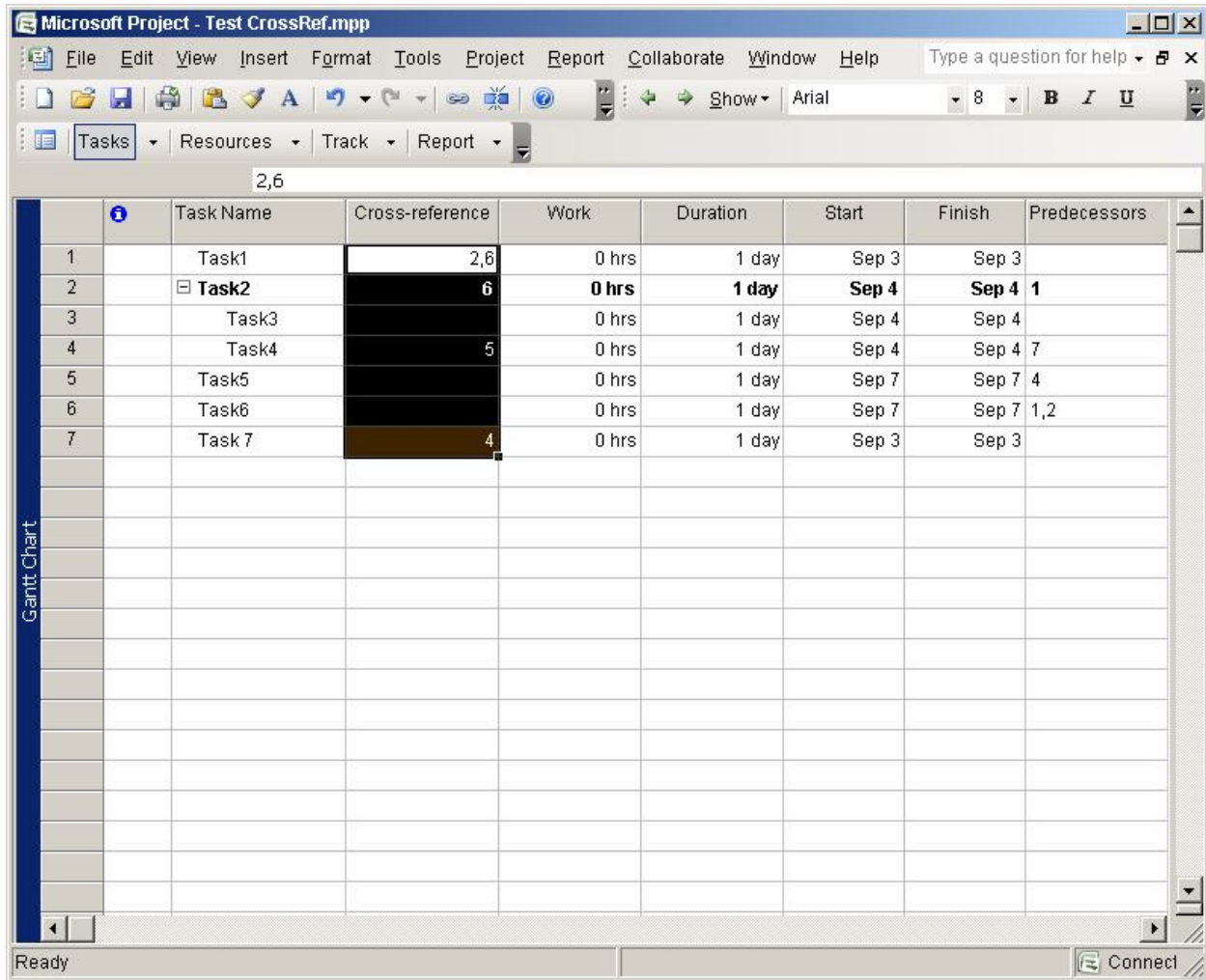


Figure 4 Reviewing Project cross-references

2.5. Limitations of the Method

So far, no limitations for this method were encountered; however some suggestions can definitely be made.

1. Running this Macro may take some time, since it iterates via whole project for every task. For the schedule consisting of 1000 lines, the Macro completes calculations in about 2 minutes.
2. If your schedule resides on the Project Server, most likely security settings are in effect. It means that you may lack access rights allowing you to permanently save this (or any other!) Macro with the schedule. However, while in MS Project Environment, you still will be able to run and use Macro results since it will be put into computer memory. The

only inconvenience is that when you leave the environment, the Macro will be lost. Hence, you will need to copy Macro code from Code Snippet 4 and repeat instructions from section 2.3.3 every time you re-start your project from Project Server and need to cross-reference project tasks.

3. Conclusion

In this paper, the problem of calculating cross-dependency of interrelated project tasks was described. This problem can arise in many vital situations, such as verifying critical path or estimating project risks. While lack of capability to cross-reference project tasks can be considered as a significant gap in a Project Management tool, easily applicable solution for Microsoft Project tool was identified and its implementation and application were described.

About the Author:**Larissa Gourevtich, PhD, PMP**

Author



Larissa Gourevich, PhD, PMP is an IT professional with 15+ years of extensive experience and knowledge in IT analysis, research, and project management. She is a PMP with PhD in Mathematical Modeling; and also she has Bachelor of Commerce degree from Ryerson University in Toronto. She has worked for Financial Industry and provincial Government, and was teaching at Ryerson University in Toronto. Currently, she is employed by CIBC, one of the “big five” Canadian banks, as IT Project Manager within PMO. Larissa closely follows current and emerging Project Management and IT trends and issues. As PMO member, she has the responsibility to set up precise standards and guidance for IT staff, providing greater success through quality and innovative approach to IT services. Based in Toronto, Dr. Gourevitch can be contacted at lgourevi@ryerson.ca.