Software Methods and Tools (Spring 2018)
Assignment 1

By Sneha Mishra
Student ID - 16242476

1. No Silver Bullet: Essence and Accidents of Software Engineering :
(a) Essence v/s Accident - Essence is the difficulties inherent in the nature of the software. Accidents are those difficulties that are not inherent but attend its production. The four essential difficulties of software systems discussed in Fred Brooks's paper are :-

- *Complexity* - Its not possible to know the whole domain, process, or system, because due to interaction of components, the number of possible states grows much faster. Complexity grows nonlinearly with size. Introduces technical and managerial problems leading to unreliability. Consequences of Complexity are (1) Technical Problems: communication among team members, enumerating (much less understanding) of all possible states of the program. (2) Management problems: conceptual integrity is hard to achieve, personnel turnover is a disaster (learning curve). Consequences of Complexity includes Communication overhead (cost overruns, schedule delays), unreliability, overview is difficult, hard to find and control all the loose ends, creates a tremendous learning and understanding burden, poor usability, poor maintainability.
- *Conformity* - Every Software must interface with existing systems. Conformity means that all new software must conform to the way things where done in the past, because it is hard to change everyone. Software is required to conform to its Operating environment and Hardware. Most of the time software systems have to interface with an existing system, but even for a new system, its easy to make software interfaces conform to other parts of the system than to change the whole existing system.
- *Changeability* - Change originates with new application/user/ machine/standard, hardware problems etc. so any software must model changing functionality, run on new hardware. Software is easy to change, unlike hardware for e.g. once an Intel processor goes to the production line, the cost of replacing it is enormous but if a Microsoft product has a bug, the cost of replacing it is negligible (just ask users to update their softwares). Successful software survives beyond the normal life because of its ability to change. Expectations from changeability are: (i) Software is easy to change than hardware (ii) Change can be done during development (iii) Changes after deployment (iv) New features (v) Software lives longer than hardware. Since software is easy to change software gets changed frequently and deviates from the initial design, adding new features, supporting new hardware. Conformity and Changeability are two of the reasons why reusability is not very successful in software systems (they make it difficult to develop component based software, since components keep changing).

- *Invisibility* - Software does not have a 3-D way in which it can be viewed also cannot visualize all of its aspect at once. But software has data flow graphs, time sequence diagrams, etc which are difficult to understand. Communication is very difficult since everyone sees it in a different way (no obvious representation). Software has no single map/graph, it has many. The Visualization tools for computer aided design are very helpful to computer engineers but there is nothing physical to visualize, so its hard to see an abstract concept (e.g. silicon chips have diagram, buildings have plans, UML has class & objects).

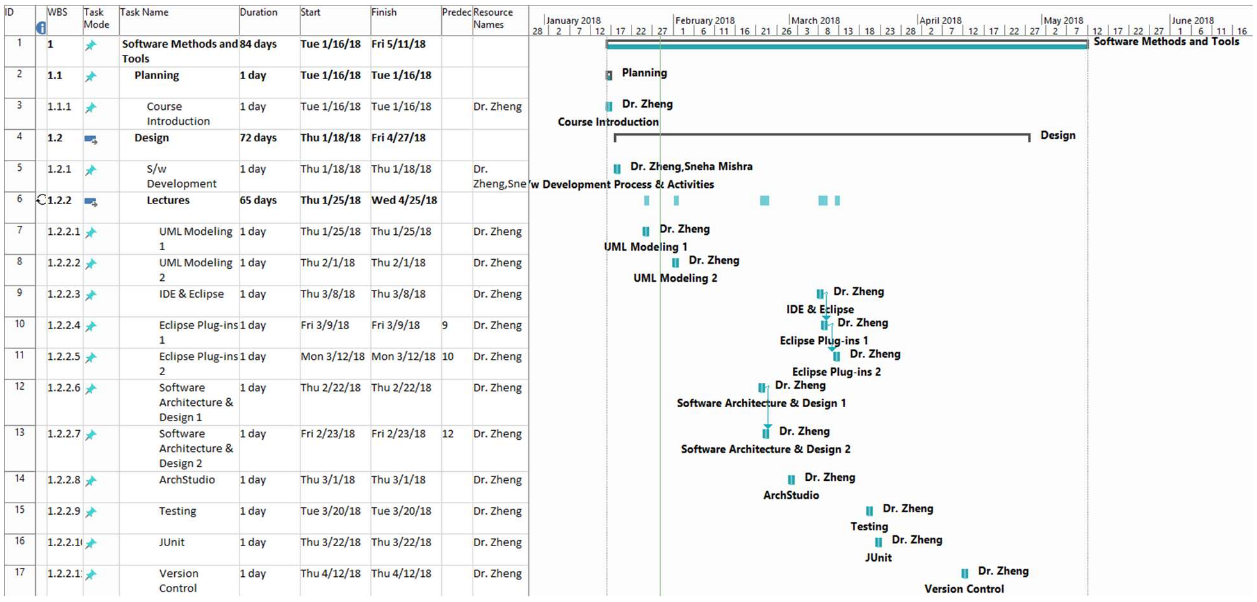
(b) The tool that I am familiar with is - **IBM Rational Rhapsody**. According to me its a "promising attack" on the essential difficulties mentioned above. The tool is very easy to use and does help us a lot when it comes to complexity, changeability and a little bit of invisibility. I will try to explain why I feel so when it comes to this tool, hopefully I will do justice to the topic. IBM Rational Rhapsody (with Design Manager) is a proven solution for modeling and design activities. The family provides a collaborative design, development and test environment for systems engineers and software engineers.
The software supports UML, SysML and AUTOSAR, and complies with standards such as ISO 26262 and more.
IBM Rational Rhapsody will allow you to manage the complexity many organizations face in today's product and systems development:

- Continuous validation via rapid simulation, prototyping and execution to address errors earlier when they are least costly to fix.
- Automatic consistency checking to enhance agility and improve reuse with collaboration to reduce both recurring and non-recurring costs.
- Share, collaborate and review your engineering lifecycle artifacts created with Rational Rhapsody or other design tools, such as Mathworks Simulink, with the extended engineering team.

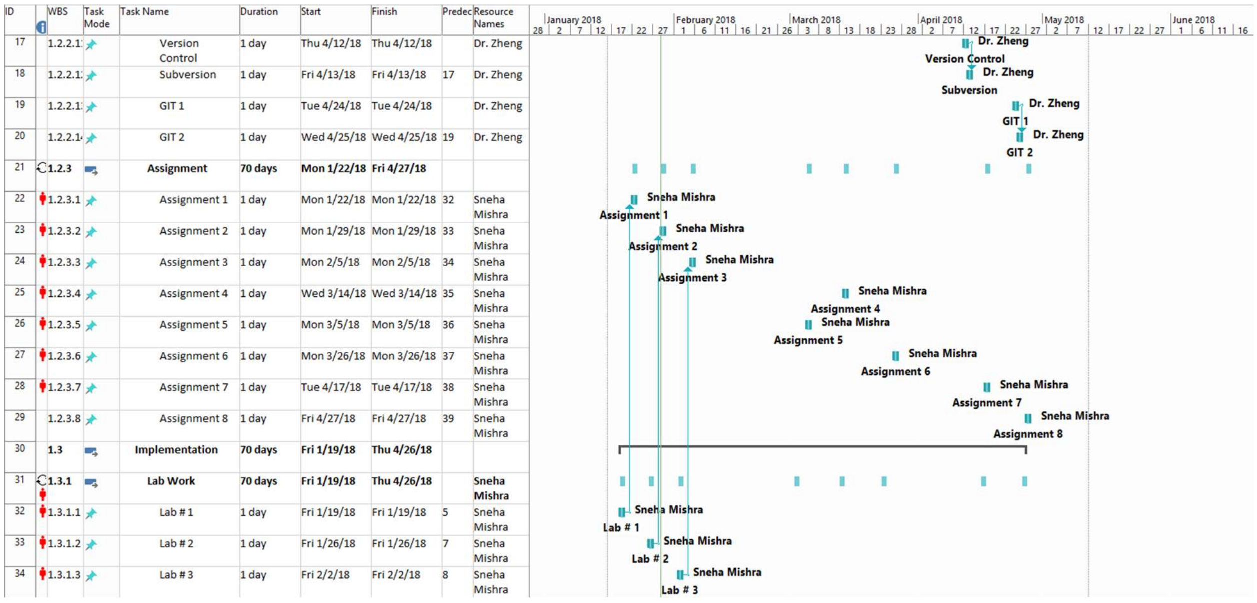
Rational Rhapsody, is a modeling environment based on UML is a visual development environment for system engineers and software developers, real-time or embedded systems and software. Rational Rhapsody uses graphical models to generate software applications in various languages including C, C++, Java, C#.
Developers use Rational Rhapsody to understand and elaborate requirements, create model designs using industry standard languages, validate functionality early in development, and automate delivery of high structured products.
Rational Rhapsody Design Manager is a web based application that stakeholders, developers, and other team members use to collaborate on the design of products, software, and systems. The product contains a server that hosts model designs which have been developed in Rational Rhapsody. A client extension component included with Rational Rhapsody allows users to connect to a Design Manager server. After connecting to the server, models can be moved into project areas with specific modeling domains based on the industry standard languages supported by Rational Rhapsody. Rhapsody Design Manager also integrates with the Rational solution for Collaborative Lifecycle

Management (CLM). In this environment, artifacts can be associated with other lifecycle resources such as requirements, and change requests.

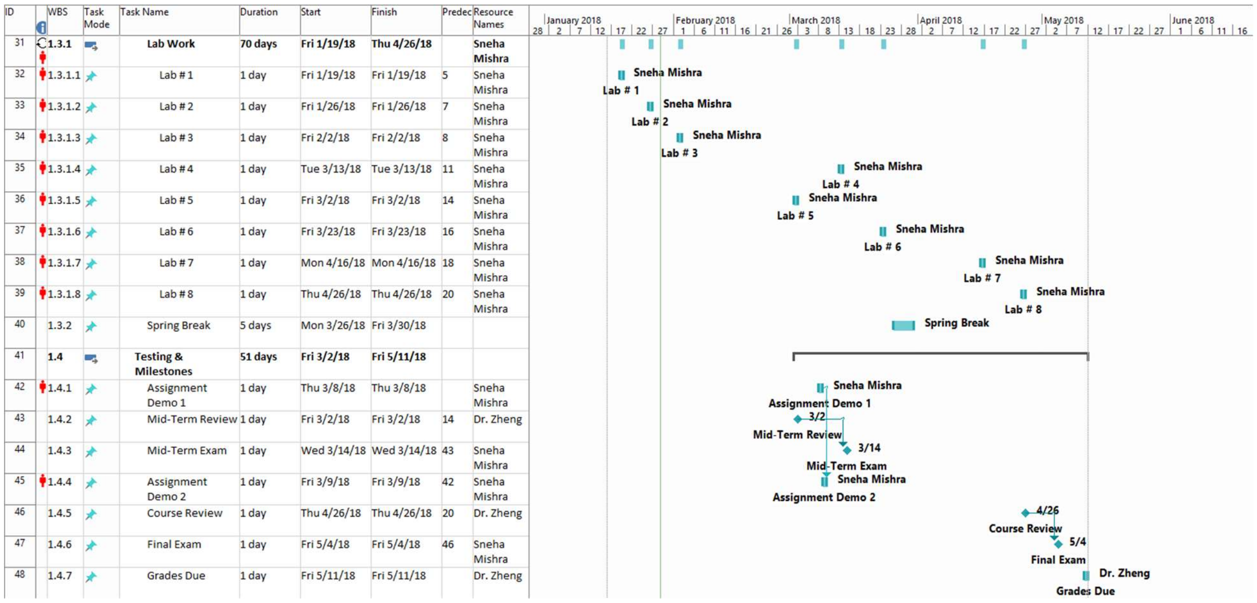
2. My class schedule for "Software Methods and Tools" :



ID		WBS	Task Mode	Task Name	Duration	Start	Finish	Predec	Resource Names
1		**1**		**Software Methods and Tools**	**84 days**	**Tue 1/16/18**	**Fri 5/11/18**		
2		**1.1**		**Planning**	**1 day**	**Tue 1/16/18**	**Tue 1/16/18**		
3		1.1.1		Course Introduction	1 day	Tue 1/16/18	Tue 1/16/18		Dr. Zheng
4		**1.2**		**Design**	**72 days**	**Thu 1/18/18**	**Fri 4/27/18**		
5		1.2.1		S/w Development	1 day	Thu 1/18/18	Thu 1/18/18		Dr. Zheng,Sne
6		**1.2.2**		**Lectures**	**65 days**	**Thu 1/25/18**	**Wed 4/25/18**		
7		1.2.2.1		UML Modeling 1	1 day	Thu 1/25/18	Thu 1/25/18		Dr. Zheng
8		1.2.2.2		UML Modeling 2	1 day	Thu 2/1/18	Thu 2/1/18		Dr. Zheng
9		1.2.2.3		IDE & Eclipse	1 day	Thu 3/8/18	Thu 3/8/18		Dr. Zheng
10		1.2.2.4		Eclipse Plug-ins 1	1 day	Fri 3/9/18	Fri 3/9/18	9	Dr. Zheng
11		1.2.2.5		Eclipse Plug-ins 2	1 day	Mon 3/12/18	Mon 3/12/18	10	Dr. Zheng
12		1.2.2.6		Software Architecture & Design 1	1 day	Thu 2/22/18	Thu 2/22/18		Dr. Zheng
13		1.2.2.7		Software Architecture & Design 2	1 day	Fri 2/23/18	Fri 2/23/18	12	Dr. Zheng
14		1.2.2.8		ArchStudio	1 day	Thu 3/1/18	Thu 3/1/18		Dr. Zheng
15		1.2.2.9		Testing	1 day	Tue 3/20/18	Tue 3/20/18		Dr. Zheng
16		1.2.2.1(		JUnit	1 day	Thu 3/22/18	Thu 3/22/18		Dr. Zheng
17		1.2.2.1:		Version Control	1 day	Thu 4/12/18	Thu 4/12/18		Dr. Zheng



ID		WBS	Task Mode	Task Name	Duration	Start	Finish	Predec	Resource Names
17		1.2.2.1:		Version Control	1 day	Thu 4/12/18	Thu 4/12/18		Dr. Zheng
18		1.2.2.1:		Subversion	1 day	Fri 4/13/18	Fri 4/13/18	17	Dr. Zheng
19		1.2.2.1:		GIT 1	1 day	Tue 4/24/18	Tue 4/24/18		Dr. Zheng
20		1.2.2.1		GIT 2	1 day	Wed 4/25/18	Wed 4/25/18	19	Dr. Zheng
21		**1.2.3**		**Assignment**	**70 days**	**Mon 1/22/18**	**Fri 4/27/18**		
22		1.2.3.1		Assignment 1	1 day	Mon 1/22/18	Mon 1/22/18	32	Sneha Mishra
23		1.2.3.2		Assignment 2	1 day	Mon 1/29/18	Mon 1/29/18	33	Sneha Mishra
24		1.2.3.3		Assignment 3	1 day	Mon 2/5/18	Mon 2/5/18	34	Sneha Mishra
25		1.2.3.4		Assignment 4	1 day	Wed 3/14/18	Wed 3/14/18	35	Sneha Mishra
26		1.2.3.5		Assignment 5	1 day	Mon 3/5/18	Mon 3/5/18	36	Sneha Mishra
27		1.2.3.6		Assignment 6	1 day	Mon 3/26/18	Mon 3/26/18	37	Sneha Mishra
28		1.2.3.7		Assignment 7	1 day	Tue 4/17/18	Tue 4/17/18	38	Sneha Mishra
29		1.2.3.8		Assignment 8	1 day	Fri 4/27/18	Fri 4/27/18	39	Sneha Mishra
30		**1.3**		**Implementation**	**70 days**	**Fri 1/19/18**	**Thu 4/26/18**		
31		**1.3.1**		**Lab Work**	**70 days**	**Fri 1/19/18**	**Thu 4/26/18**		**Sneha Mishra**
32		1.3.1.1		Lab # 1	1 day	Fri 1/19/18	Fri 1/19/18	5	Sneha Mishra
33		1.3.1.2		Lab # 2	1 day	Fri 1/26/18	Fri 1/26/18	7	Sneha Mishra
34		1.3.1.3		Lab # 3	1 day	Fri 2/2/18	Fri 2/2/18	8	Sneha Mishra



ID	WBS	Task Mode	Task Name	Duration	Start	Finish	Predec	Resource Names
31	**1.3.1**		**Lab Work**	**70 days**	**Fri 1/19/18**	**Thu 4/26/18**		**Sneha Mishra**
32	1.3.1.1		Lab # 1	1 day	Fri 1/19/18	Fri 1/19/18	5	Sneha Mishra
33	1.3.1.2		Lab # 2	1 day	Fri 1/26/18	Fri 1/26/18	7	Sneha Mishra
34	1.3.1.3		Lab # 3	1 day	Fri 2/2/18	Fri 2/2/18	8	Sneha Mishra
35	1.3.1.4		Lab # 4	1 day	Tue 3/13/18	Tue 3/13/18	11	Sneha Mishra
36	1.3.1.5		Lab # 5	1 day	Fri 3/2/18	Fri 3/2/18	14	Sneha Mishra
37	1.3.1.6		Lab # 6	1 day	Fri 3/23/18	Fri 3/23/18	16	Sneha Mishra
38	1.3.1.7		Lab # 7	1 day	Mon 4/16/18	Mon 4/16/18	18	Sneha Mishra
39	1.3.1.8		Lab # 8	1 day	Thu 4/26/18	Thu 4/26/18	20	Sneha Mishra
40	1.3.2		Spring Break	5 days	Mon 3/26/18	Fri 3/30/18		
41	**1.4**		**Testing & Milestones**	**51 days**	**Fri 3/2/18**	**Fri 5/11/18**		
42	1.4.1		Assignment Demo 1	1 day	Thu 3/8/18	Thu 3/8/18		Sneha Mishra
43	1.4.2		Mid-Term Review	1 day	Fri 3/2/18	Fri 3/2/18	14	Dr. Zheng
44	1.4.3		Mid-Term Exam	1 day	Wed 3/14/18	Wed 3/14/18	43	Sneha Mishra
45	1.4.4		Assignment Demo 2	1 day	Fri 3/9/18	Fri 3/9/18	42	Sneha Mishra
46	1.4.5		Course Review	1 day	Thu 4/26/18	Thu 4/26/18	20	Dr. Zheng
47	1.4.6		Final Exam	1 day	Fri 5/4/18	Fri 5/4/18	46	Sneha Mishra
48	1.4.7		Grades Due	1 day	Fri 5/11/18	Fri 5/11/18		Dr. Zheng

References :

1) http://worrydream.com/refs/Brooks-NoSilverBullet.pdf
2) https://www.slideshare.net/ArunBanotra/no-silver-bullet-essence-and-accidents-of-software-engineering-63310216
3) https://www-03.ibm.com/software/products/en/ratirhapfami