Professor Hausi A. Müller PhD PEng FCAE
Lorena Castañeda
Department of Computer Science
Faculty of Engineering
University of Victoria

http://www.engr.uvic.ca/~seng321/
https://courses1.csc.uvic.ca/courses/201/spring/seng/321

Announcements

- Website up and running
- Slides from the first lecture posted
- Next topics
  - Quality attributes
  - Software life cycles
- Tuesday
  - RFP presentations
  - Project selection
- Thursday
  - Team selection
- Reading Assignment
  - Chapters 1-3 Elicitation
  - Chapters 8-10 Elicitation and Modelling

Evaluation

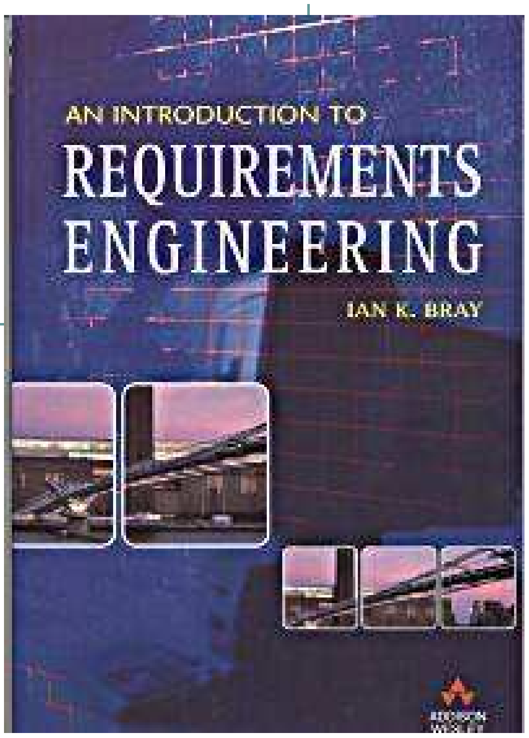
3 Quizzes	**6%**
Participation	**5%**
Midterm exam	**14%**
Group Project	**40%**
Final exam	**35%**

- Remarks
  - Midterm is in-class; final is scheduled by UVic
  - You ***have*** to pass the project and the final exams to pass the course

Text Books

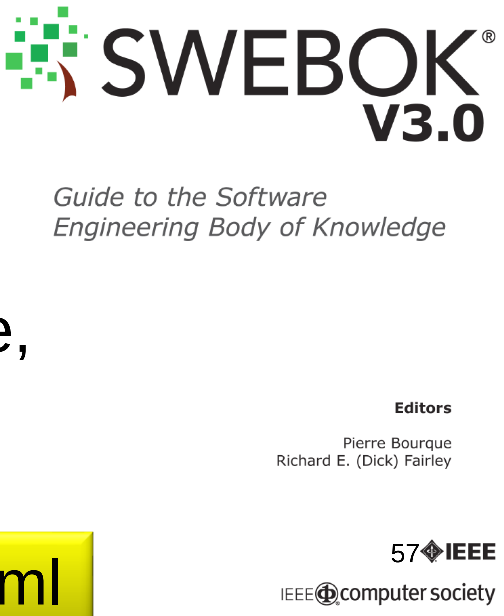
Required Textbook

- Ian K. Bray: An Introduction to Requirements Engineering, Pearson (2002)



Recommended Textbook

- P. Bourque, R. Fairley: SWEBOK V3.0: Guide to the Software Engineering Body of Knowledge, IEEE Computer Society (2014)


SWEBOK®
V3.0

Guide to the Software Engineering Body of Knowledge

Editors

Pierre Bourque
Richard E. (Dick) Fairley

IEEE computer society

http://www.engr.uvic.ca/~seng321/resources.html

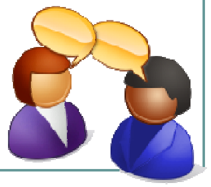
SENG 321 Calendar

First day of classes	Tue, Jan 5
Labs begin	Tue, Jan 12
Reading break	Feb 8-12
Midterm	Fri, Feb 23
Easter break	Mar 25-28
Project presentations	Mar 29-31
Last day of classes	Fri, Mar 31

Detailed course calendar: deliverables deadlines
http://www.engr.uvic.ca/~seng321/calendar.html

Students must participate in all project presentations in class & labs
No show results in a 25% reduction in the mark for that presentation

Project Deadlines and Marks



1.	Call for Project Proposals		**6 Jan (Class)**
2.	Request for Proposal (RFP)		**8 Jan**
3.	Project selection		**12 Jan (Lab)**
4.	Team selection		**14 Jan (Lab)**
5.	Informal Requirements Definition (S0)	5%	**21 Jan (Lab)**
6.	Project website up and running (S0)	5%	**21 Jan (Lab)**
7.	Customer Feedback on S0 (C0)	5%	**26 Jan (Lab)**
8.	Formal Requirements Spec (S1)	10%	**16 Feb (Lab)**
9.	Customer Feedback on S1 (C1)	5%	**18 Feb (Lab)**
10.	Detailed Requirements Spec (S2a)	10%	**1 Mar (Lab)**
11.	Prototype demo (S2b)	5%	**3 Mar (Lab)**
12.	Customer Feedback on S2a-b (C2)	5%	**8 Mar (Lab)**
13.	Final Requirements Spec (S3a)	15%	**15 Mar (Lab)**
14.	User Manual (S3b)	10%	**22 Mar (Lab)**
15.	Customer Feedback on S3a-b (C3)	5%	**24 Mar (Lab)**
16.	Demo Final Project (S4)	10%	**29,31 Mar (Lab)**
17.	Customer Feedback on S4 (C4)	5%	**29,31 Mar (Lab)**
18.	Instructor and TA Evaluations (S5)	5%	**1 Apr**

Next Week

Tuesday

- Project selection
- 80 proposals
- 2 mins elevator speech per student
- Select 19-20 projects
- Lecture and lab
- ELL 167

Thursday

- Team selection
- 80 students
- Select 19-20 well balanced teams of 4 students
- Lab ELL 167

SENG 321 RFP

Request for proposals
to develop the ultimate
problem solving program.

X SENG 321 Student

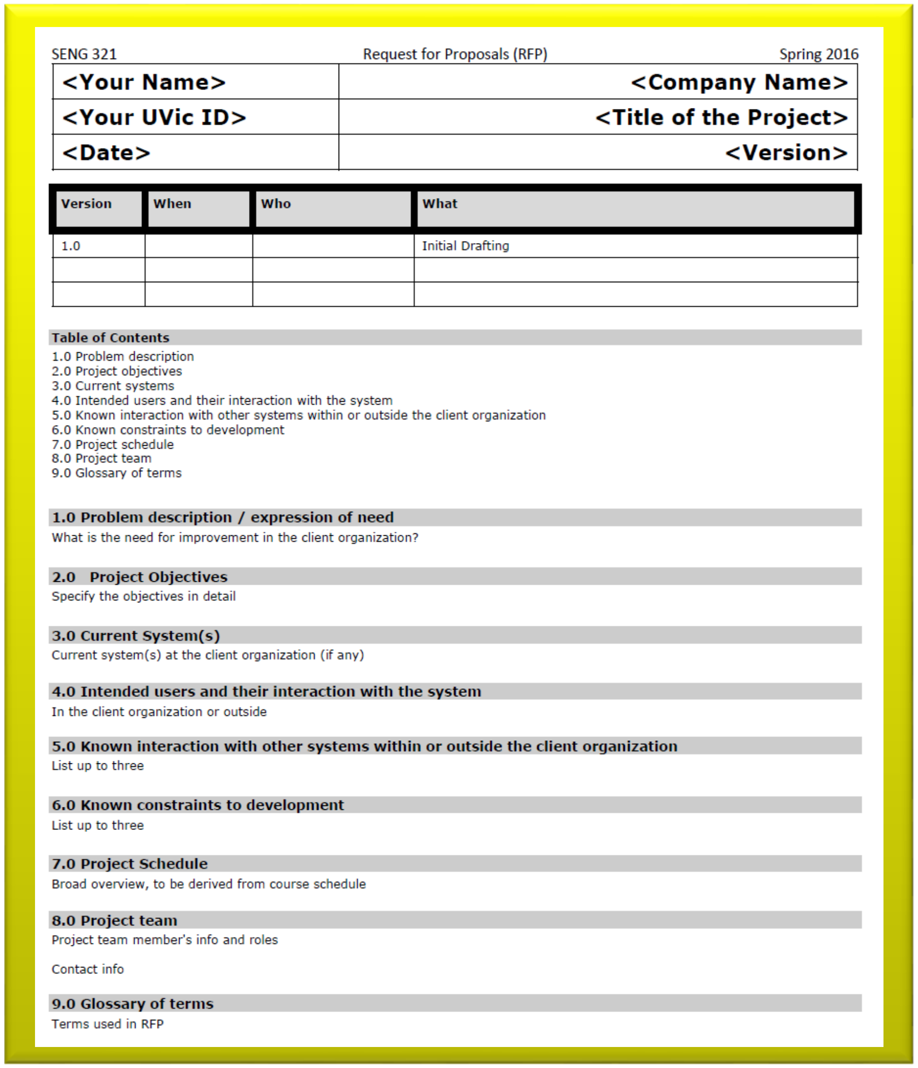
Request for Proposal (RFP)

- Submit three documents
  - Your RFP — use 2016 SENG 321 RFP template
  - Your 1-page PDF slide for 2-mins elevator speech
  - Your 1-page résumé documenting your experience

- This assignment counts as class participation

- Due date Fri, Jan 8 — 1:00 pm
- Submit to submit@rigiresearch.com

Deliverable 1a
RFP Template



SENG 321 Request for Proposals (RFP) Spring 2016

<Your Name>	<Company Name>
<Your UVic ID>	<Title of the Project>
<Date>	<Version>

Version	When	Who	What
1.0			Initial Drafting

Table of Contents

1.0 Problem description / expression of need
What is the need for improvement in the client organization?

2.0 Project Objectives
Specify the objectives in detail

3.0 Current System(s)
Current system(s) at the client organization (if any)

4.0 Intended users and their interaction with the system
In the client organization or outside

5.0 Known interaction with other systems within or outside the client organization
List up to three

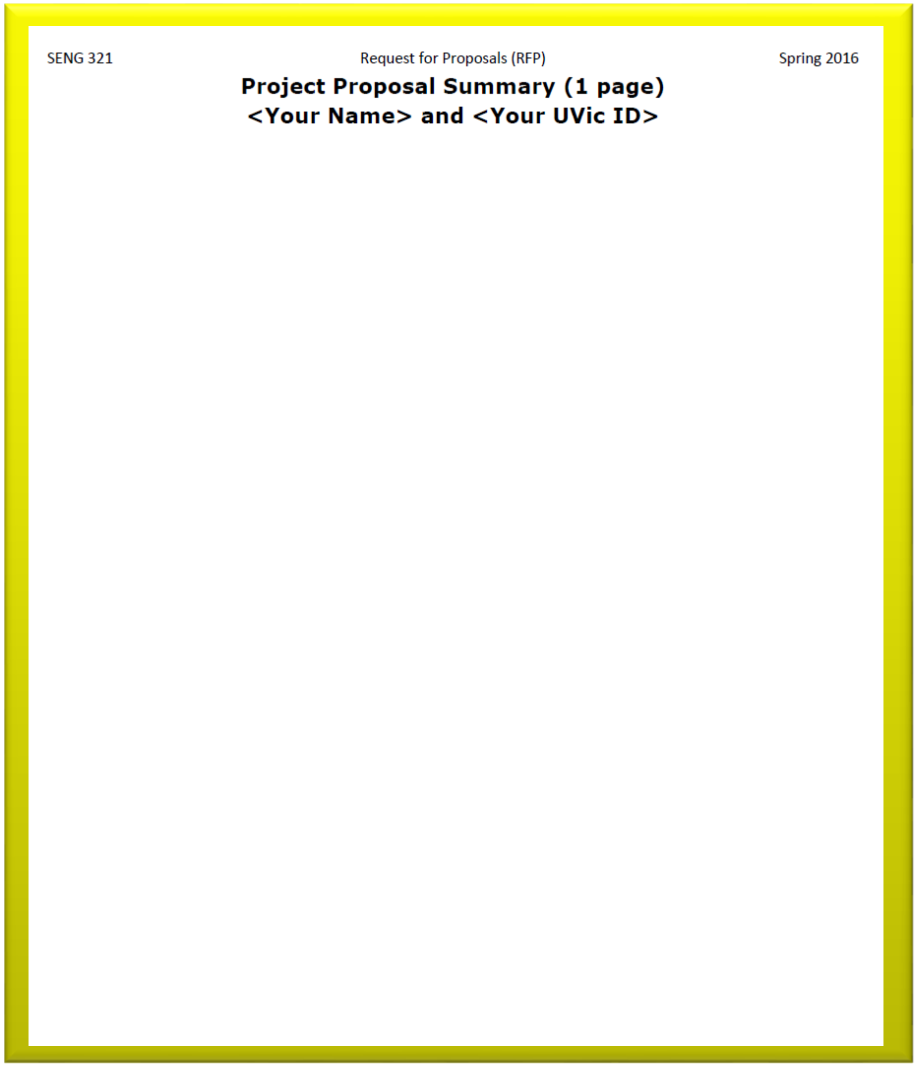
6.0 Known constraints to development
List up to three

7.0 Project Schedule
Broad overview, to be derived from course schedule

8.0 Project team
Project team member's info and roles

Contact info

9.0 Glossary of terms
Terms used in RFP



SENG 321 Request for Proposals (RFP) Spring 2016

Project Proposal Summary (1 page)
<Your Name> and <Your UVic ID>

Deliverable 1a
Statement of Work Template

I. **Scope of Work:** Describe the work to be done to detail. Specify the hardware and software involved and the exact nature of the work.

II. **Location of Work:** Describe where the work must be performed. Specify the location of hardware and software and where the people must perform the work

III. **Period of Performance:** Specify when the work is expected to start and end, working hours, number of hours that can be billed per week, where the work must be performed, and related schedule information.

IV. **Deliverables Schedule:** List specific deliverables, describe them in detail, and specify when they are due.

V. **Applicable Standards:** Specify any company or industry-specific standards that are relevant to performing the work.

VI. **Acceptance Criteria:** Describe how the buyer organization will determine if the work is acceptable.

VII. **Special Requirements:** Specify any special requirements such as hardware or software certifications, minimum degree or experience level of personnel, travel requirements, and so on.

Deliverable 1a
Many Web Resources on RFPs

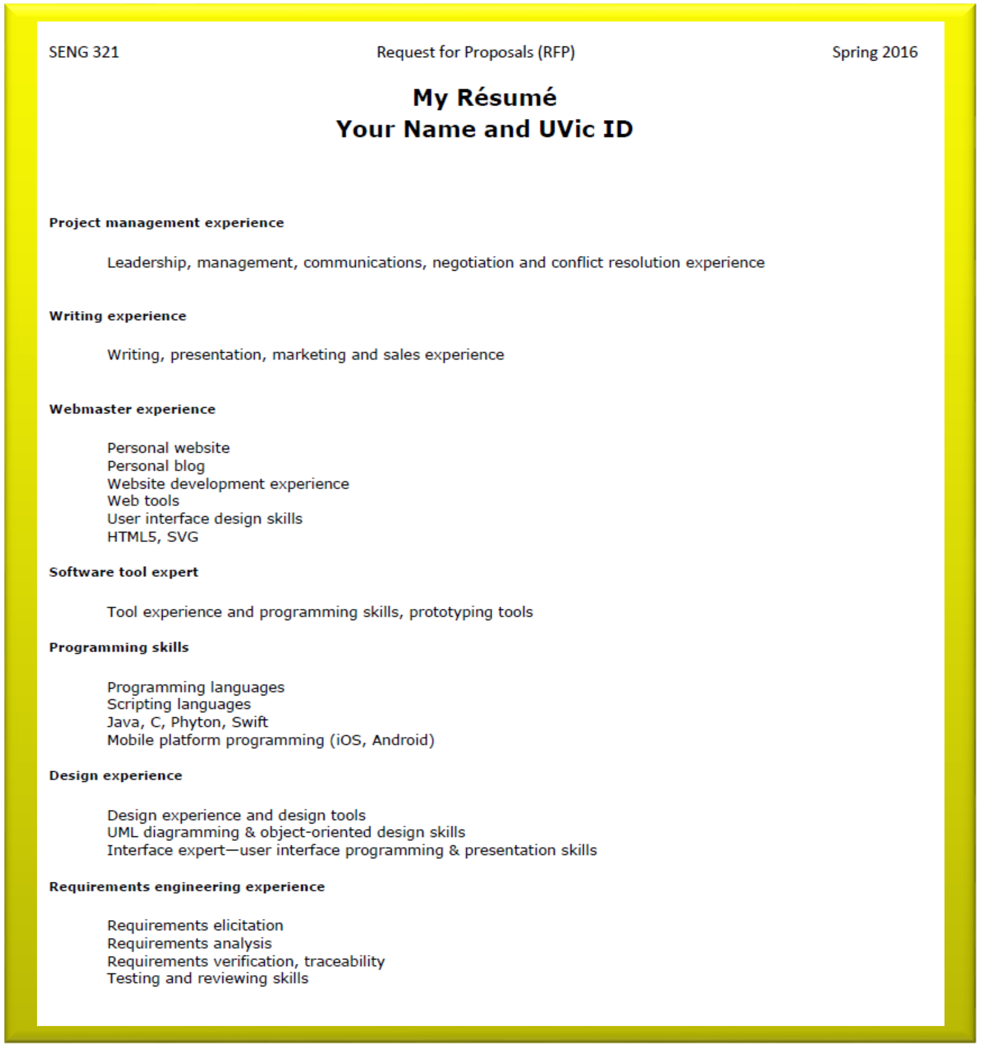
- How to respond to an RFP http://www.slideshare.net/MarianneKolodiy/how-torespondtorfp

- How to write an RFP for web content management http://www.slideshare.net/Percussion/how-to-write-a-request-for-proposal-rfp-for-web-content-management

Deliverable 1b
1-page PDF Project Summary

- 1-page PowerPoint slide to sell your project to the entire class (PDF format)
- On Tuesday
  - 2 mins elevator speech per student
  - Select 20 projects from 80 submissions by voting for projects

Deliverable 1c
Your Résumé for Team Building



SENG 321 Request for Proposals (RFP) Spring 2016

My Résumé
Your Name and UVic ID

Project management experience

Leadership, management, communications, negotiation and conflict resolution experience

Writing experience

Writing, presentation, marketing and sales experience

Webmaster experience

Personal website
Personal blog
Website development experience
Web tools
User interface design skills
HTML5, SVG

Software tool expert

Tool experience and programming skills, prototyping tools

Programming skills

Programming languages
Scripting languages
Java, C, Phyton, Swift
Mobile platform programming (iOS, Android)

Design experience

Design experience and design tools
UML diagramming & object-oriented design skills
Interface expert—user interface programming & presentation skills

Requirements engineering experience

Requirements elicitation
Requirements analysis
Requirements verification, traceability
Testing and reviewing skills

The resume is required for balanced group assignments

Project Ideas

- Smart system
  - Context-aware
  - Self-adaptive
  - Mobile

- Example past projects
  - Web based sprinkler system
  - Food management system
  - Parking meter system
  - Travel management system
  - Inventory control system
  - Auction system
  - Smart home
  - Wardrobe advisor
  - Bus tracking system

Websites

- Each group will maintain a simple website with two components throughout the term:
  - Customer website
  - Developer website
- Website should be up by **21 Jan 2016**
  - Worth 5% of your project mark
  - If not kept up-to-date throughout the term, the entire team loses the 5%

Peer Reviews

- We expect all team members to receive the same marks for the project deliverables
- Customer reviews are required after each major deliverable
- Your mark and your team members’ marks depends on the reviews being sent in on time!

Lateness Policy for All Course Deliverables

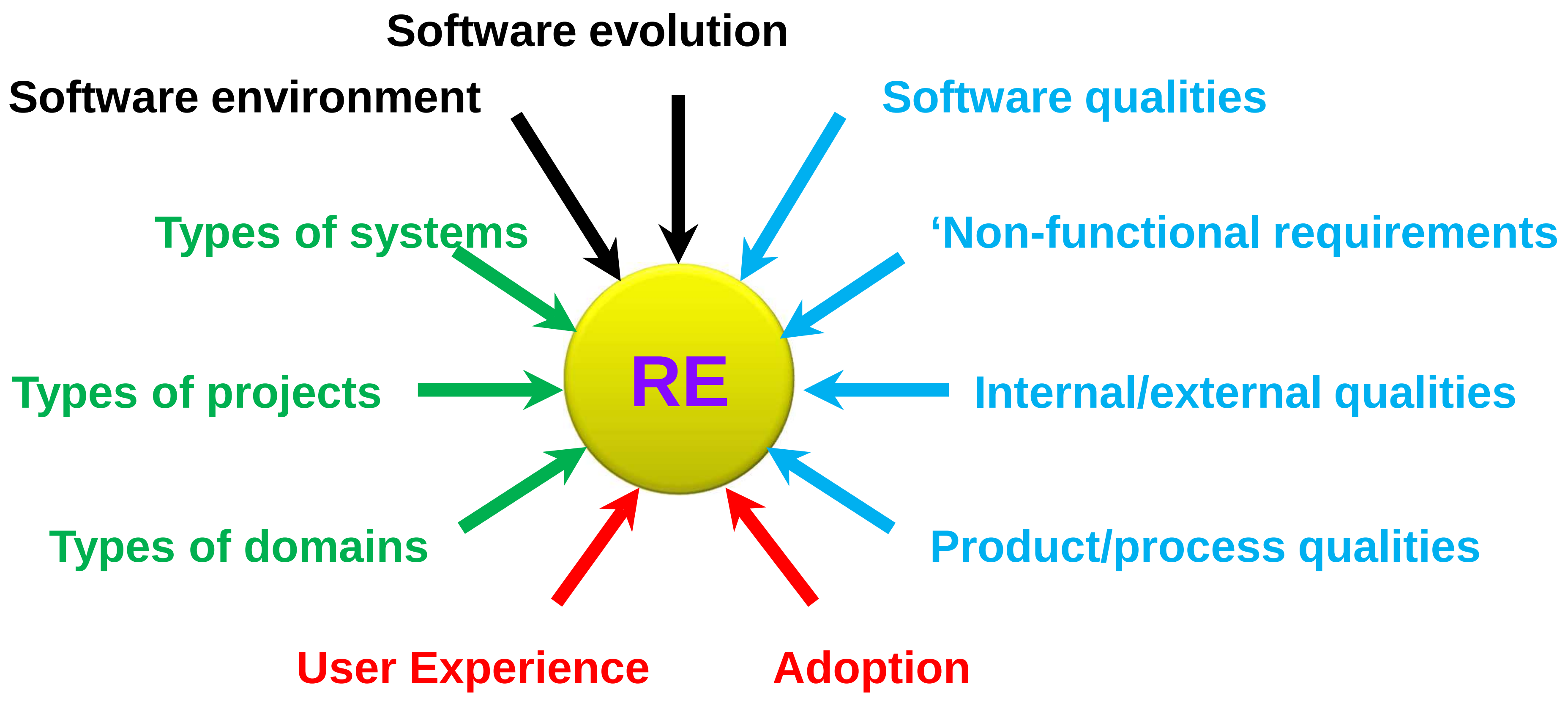
- All deliverables to be emailed to the course account to submit@rigiresearch.com

NO LATE DELIVERABLES!!

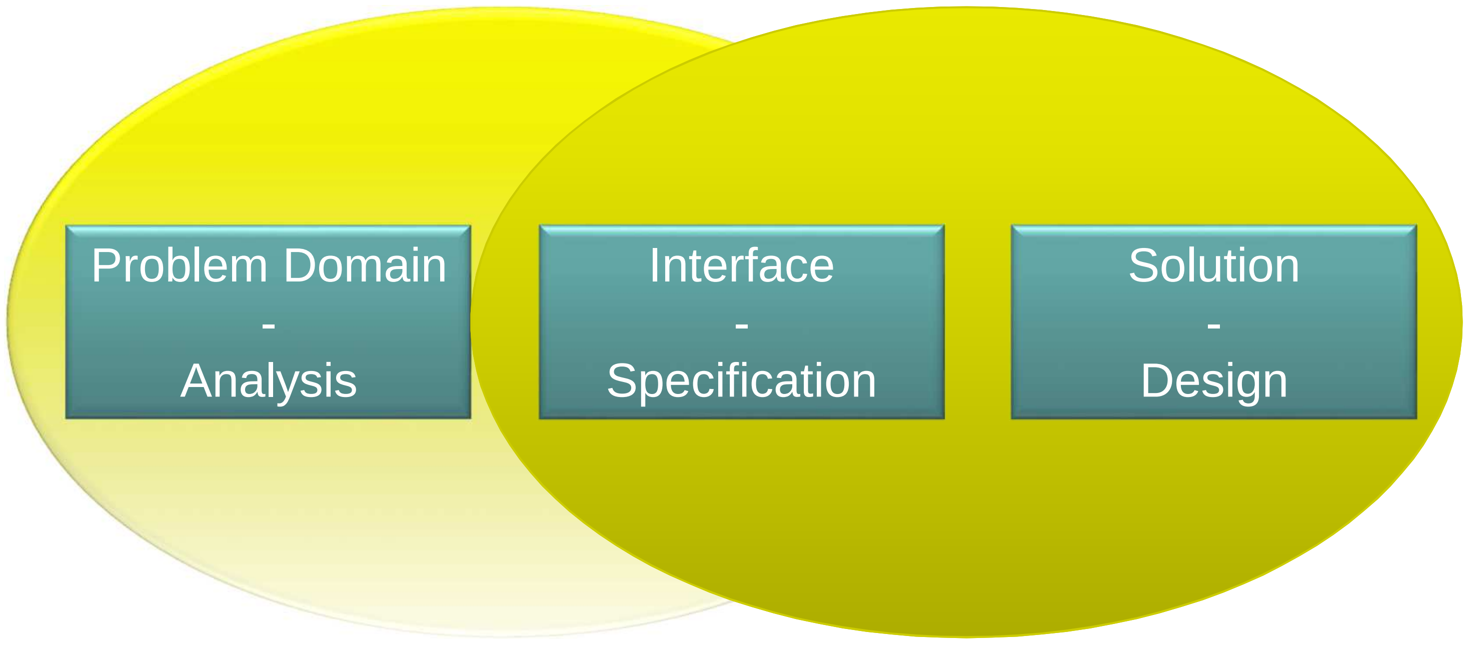
Academic Integrity and Cheating

- Cheating, plagiarism and other forms of academic fraud are taken very seriously by the University, the Faculty, and the teaching staff.
- Examples:
  - Submitting the work of another person as your original work
  - Incorporating others work in your work and not attributing it
  - It is permitted and encouraged to discuss projects with your peers on the whiteboard but **NOT** permitted to copy their solutions as they talk to you. Both parties would be penalized
  - All sources must be properly cited including websites
- Consult UVic's policy on academic integrity:
  - http://web.uvic.ca/calendar2009/FACS/UnIn/UARe/PoAcI.html

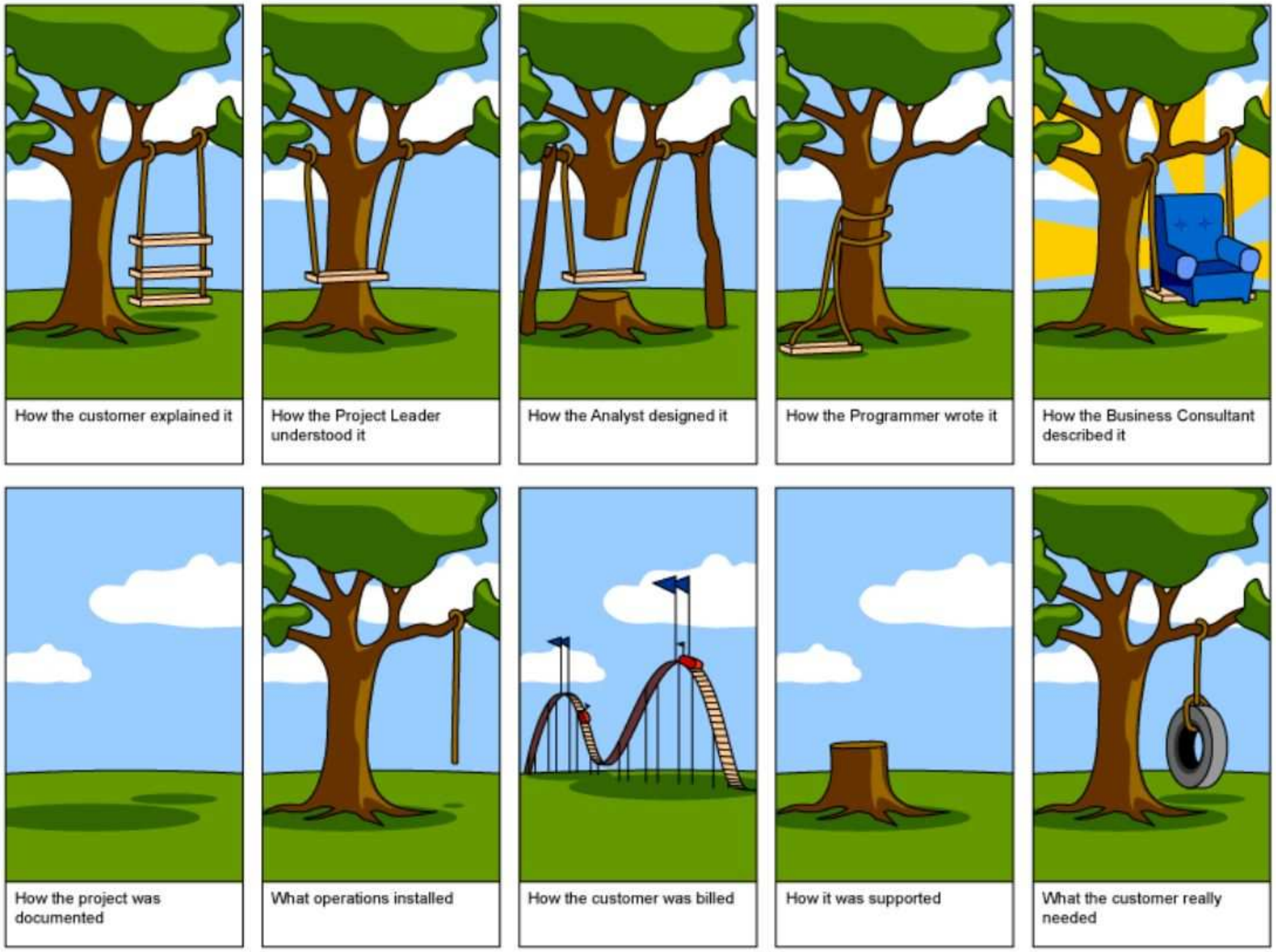
Requirements Engineering Many Forces at Work



Separation of Concerns



Why Requirements Engineering?


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Why Requirements Engineering?

- Good requirements leads to higher quality software
- Problem
  - Everybody is for it—under certain conditions.
  - Everybody feels they understand it—even though they wouldn't want to explain it.
  - Most people feel that problems in these areas are caused by other people—if only they would take the time to do things right.

Roger Pressman quoting Philip Crosby

What is Quality (Pressman)?

- Conformance to explicitly stated requirements, standards, and implicit characteristics
- Functional and non-functional **requirements**
  - Foundation from which quality is measured
  - Lack of conformance 🡐 🡒 lack of quality
- Explicitly documented development **standards**
  - Development criteria guide manner software engineered
  - Criteria not followed 🡒 lack of quality
- Implicit characteristics expected of professionally developed software
  - Often go unmentioned (e.g., desire for good maintainability)
  - Even if explicit requirements met, failing to meet implicit requirements suggest suspect software quality

Quality Factors

- **Correctness:** fulfill specifications
- **Reliability:** perform function with required precision
- **Efficiency:** resources & code required to perform function
- **Integrity:** controlled access to software / data
- **Usability:** effort required to learn / operate / interpret
- **Maintainability:** effort to test program to ensure functionality
- **Flexibility:** effort required to modify operational program
- **Portability:** effort to transfer to other environments
- **Reusability:** extent to which components can be reused
- **Interoperability:** effort to couple system with another

Software qualities

- Software engineering is concerned with software qualities
- Qualities (a.k.a. “ilities”) are goals in the practice of software engineering
- The qualities are usually expressed as non-functional requirements during the early design stages
- External qualities
  - visible to the user
  - reliability, efficiency, usability
- Internal qualities
  - the concern of developers
  - they help developers achieve external qualities
  - verifiability, maintainability, extensibility, evolvability, adaptability

Software qualities …

- Product qualities
  - concern the developed artifacts
  - maintainability, understandability, performance
- Process qualities
  - deal with the development activity
  - products are developed through process
  - maintainability, productivity, timeliness

For each one of the following software solutions, rate **software qualities** as **not required** or **required**

- Facebook --- Social media
- Netflix --- Recommender system
- Google maps --- location and route
- Hangouts/Facetime --- Video call

Software qualities …

- Correctness
  - Ideal quality
  - Established with respect to requirements specification
  - Absolute
- Verifiabilty
  - Ease of establishing desired properties
  - Performed by formal analysis or testing
  - Internal quality

Software qualities …

- Reliability
  - Probability that the software will perform its logical operation in the specified environment without failure
  - Statistical quality
  - Probability that software will operate as expected over a give period of time
  - Relative
- Survivability
  - Probability that the software will continue to perform or support critical functions when a portion of the system is inoperable
- Robustness
  - Reasonable behaviour in unforseen circumstances
  - Subjective
  - A specified requirement is an issue of correctness

Software qualities …

- Availability
  - 24/7 availability
  - Minimum down time during upgrades (e.g., web services)
- Resiliancy
  - Ability to recover from a failure
  - Applies to hardware, software or data

Software qualities …

- Usability
  - Ability of end-users to easily use software
  - Extremely subjective
- Usefulness
  - How useful is a particular feature?
  - What are the most important bells and whistles?
  - Are there any superfluous bells and whistles?
  - Which operations are most relevant for my task?
- Understandability
  - Ability of developers to understand produced artifacts easily
  - Internal product quality
  - Subjective

Software qualities …

- Performance
  - Equated with efficiency
  - assessable by measurement, analysis, and simulation
- Reusability
  - ability to construct new software from existing pieces
  - must be planned for
  - occurs at all levels: from people to process, from requirements to code

Software qualities …

- Interoperability
  - ability of software (sub)systems to cooperate with others
  - easily integratable into larger systems
  - common techniques include APIs, plug-in protocols, etc.
  - data, control, and presentation integration
  - Examples: XML, SVG, W3C, .NET, Eclipse, scripting
- Scalability
  - ability of a software system to grow in size while maintaining its properties and qualities
  - assumes maintainability and evolvability
  - goal of component-based development

Software qualities …

- Heterogeneity
  - ability to compose a system from pieces developed in multiple programming languages, on multiple platforms, by multiple developers
  - necessitated by reuse
  - goal of component-based development
- Portability
  - ability to execute in new environments with minimal effort
  - may be planned for by isolating environment-dependent components
  - necessitated by the emergence of highly-distributed systems
  - an aspect of heterogeneity

Software qualities …

- Maintainability
  - the ease with which a software system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment [IEEE 90].
  - Addresses corrective, adaptive, perfective, and preventive maintenance
- Evolvability
  - addresses adaptive, perfective, and preventive maintenance
  - ability to add or modify functionality
  - addresses adaptive and perfective maintenance
  - problem: evolution of implementation is too easy
  - evolution should start at requirements or design