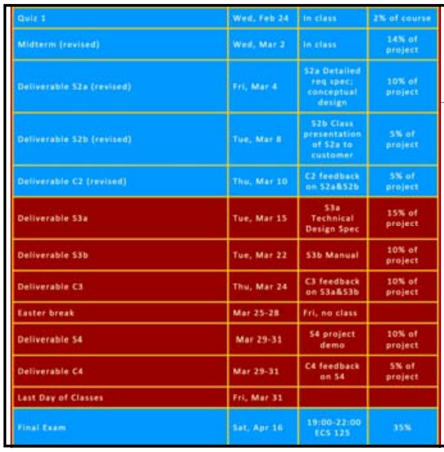
Professor Hausi A. Müller PhD PEng FCAE
Department of Computer Science
Faculty of Engineering
University of Victoria

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

SENG 321 Calendar



Quiz 1	Wed, Feb 24	In class	2% of course
Midterm (revised)	Wed, Mar 2	In class	14% of project
Deliverable S2a (revised)	Fri, Mar 4	S2a Detailed req spec; conceptual design	10% of project
Deliverable S2b (revised)	Tue, Mar 8	S2b Class presentation of S2a to customer	5% of project
Deliverable C2 (revised)	Thu, Mar 10	C2 feedback on S2a&S2b	5% of project
Deliverable S3a	Tue, Mar 15	S3a Technical Design Spec	15% of project
Deliverable S3b	Tue, Mar 22	S3b Manual	10% of project
Deliverable C3	Thu, Mar 24	C3 feedback on S3a&S3b	10% of project
Easter break	Mar 25-28	Fri, no class	
Deliverable S4	Mar 29-31	S4 project demo	10% of project
Deliverable C4	Mar 29-31	C4 feedback on S4	5% of project
Last Day of Classes	Fri, Mar 31		
Final Exam	Sat, Apr 16	19:00-22:00 ECS 125	35%

Announcements

- S2 & C2
  - Posted
  - S2 number of pages
  - Prototype sophistication
- Fri, March 4
  - S2a due
- Tue, March 8
  - S2b due
  - Presentations in labs
  - Attendance required
- Thu, March 10
  - C2 due
  - Feedback on S2a & S2b

- Final Exam
  - Sat, April 16
  - 19:00-22:00
  - ECS 125

The S2b Show

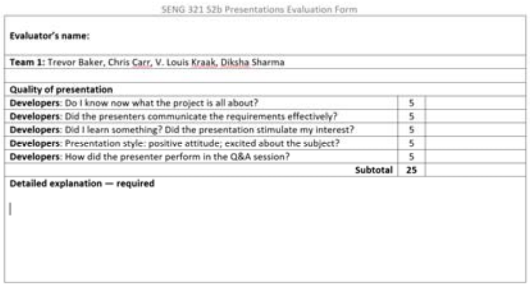
Prep
- 5 - 7 polished slides (at most) in pptx, ppt, or pdf form
- **Send slides to submit@rigiresearch.com by Monday — 11:55 pm**
- **Team number (e.g., Team 7) on every slide**
- Order of presentation arranged by TAs

Developers presentation
- Entire group must be on stage
- 7 min ➔ Presentation
- 2 min ➔ Questions
- Presenters: 1-4 people

Customers questions
- Entire group must be on stage
- Customers must ask two “good” questions

Audience
- Must evaluate every developer presentation using evaluation form

Evaluation Form



SENG 321 S2b Presentations Evaluation Form

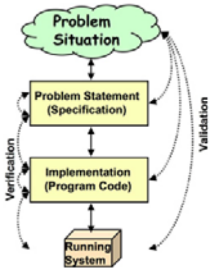
Evaluator’s name:		
Team 1: Trevor Baker, Chris Carr, V. Louis Kraak, Diksha Sharma		
Quality of presentation		
Developers: Do I know now what the project is all about?	5	
Developers: Did the presenters communicate the requirements effectively?	5	
Developers: Did I learn something? Did the presentation stimulate my interest?	5	
Developers: Presentation style: positive attitude; excited about the subject?	5	
Developers: How did the presenter perform in the Q&A session?	5	
Subtotal	**25**	
Detailed explanation — required		

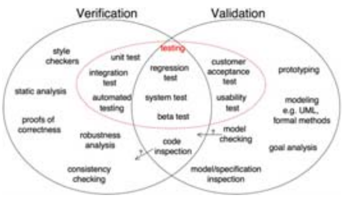
Validation vs. Verification

- Validation — Evaluate software requirements specification wrt. customer requirements:
  - Are we building the right system?
  - Is the specification what the customer wants?
- Verification — Evaluate software artifact wrt. existing artifacts:
  - Are we building the system right?
  - For example, does the design implement the spec?

Thus, validation is concerned with checking that the system will meet the customer’s actual needs, while verification is concerned with whether the system is well-engineered, error-free, and so on. Verification will help to determine whether the software is of high quality, but it will not ensure that the system is useful.

Validation vs. Verification


Problem Situation
Problem Statement (Specification)
Implementation (Program Code)
Running System
Verification
Validation



Steve Easterbrook
University of Toronto

7

www.easterbrook.ca/steve/2010/11/the-difference-between-verification-and-validation/

Validation Criteria

- Validation criteria include:
  - Correctness
  - (Un)ambiguity
  - Completeness
  - Consistency
- We are checking:
  - Whether the software requirements specification captures stakeholders' requirements
  - User satisfaction that the system as specified will meet their needs, is usable and useful

8

Classic Quality Criteria for a Requirements Specification

Requirements Spec Properties	Interpretation
Correct	Each requirement reflects a need
Complete	All necessary requirements included
Unambiguous	All parties agree on meaning
Consistent	All parts match, e.g., E/R and event list
Ranked for importance and stability	Priority and expected changes per requirement
Modifiable	Easy to change, maintaining consistency
Verifiable	Possible to see whether requirement is met
Traceable	To goals/purposes, to design/code
Understandable	By customers and developers
Necessary AND Feasible	

From: Soren Lauesen:
Software Requirements
© Pearson / Addison-Wesley 2002

9

Desirable Characteristics for a Requirements Specification

Requirements Spec Properties	Interpretation
Clear, concise and understandable	Easy to read and acts as a good communication tool for stakeholders
Unambiguous	Single interpretation which cannot be misunderstood
Checkable (complete, consistent)	Can be checked for errors
Consistent	All parts match, e.g., E/R and event list
Testable / verifiable / measurable	Can easily verify if we met the requirements
Traceable	Contains rationale and requirements are linked back to business rules and priorities

From: Soren Lauesen:
Software Requirements
© Pearson / Addison-Wesley 2002

10

Characteristics High Quality Requirements Specifications

Requirements Spec Properties	Interpretation
Correct	Should involve customers to ensure you get the correct requirements, instead of developers guessing; should not contradict other requirements
Feasible	Should be feasible using known limitations and capabilities; need to have a developer involved to provide a reality check
Necessary	Each requirement should originate from an authoritative source
Prioritized	
Unambiguous	
Verifiable	

From: Soren Lauesen:
Software Requirements
© Pearson / Addison-Wesley 2002

11

Validation Challenges

- In a typical project, there exist few documents that can be used as the basis for validation ☹
- When validating a specification, we are validating it against the stakeholders' requirements.
  - Some of these may not be documented!
  - If they are documented, they are probably expressed in natural language
    - ➔ open to multiple interpretations
- In short, validating a document is a time intensive and error-prone process.

12

Validation Techniques

- Reviews
  - Walkthroughs
  - Formal inspections
  - Focused inspections
  - Active inspections
  - Checklists
- Testing
- Prototyping
- Formal validation

Reviews

- Actively used in industry
- One of the most successful techniques
- Basic idea
  - Humans (often semi-outsiders) read and analyze artifacts, look for problems, meet to discuss these problems, and agree on a set of actions to address the identified problems.
  - Often, they will have a good idea of likely problem areas both inside and outside problem domain.
  - Need both domain experts and domain-ignorant developers.

Reviews

- Broad industrial consensus: Reviews work!
  - They find more errors than testing does.
  - They find errors faster than testing does.
  - Everyone believes in them, even Microsoft.
- Requirements reviews are the most widely used technique of requirements validation.

Reviews work: One of the great industrial success stories !!!!!!!!!

Advantages and Disadvantages of Reviews

- Advantages
  - Can review all kinds of software artifacts, not just code, e.g., specs, test suites, design docs
  - Helps catch errors sooner when they are much cheaper to fix!
  - Good for educating newcomers—brings the entire development team together into the big picture
- Disadvantages
  - It is though work that is time-consuming and expensive which requires preparation, paperwork, follow-ups
  - But it is usually cheaper than the alternatives!

Social Problems with Reviews

- Reviewers are usually software developers who have their own work they need to do as opposed to professional reviewers
  - Reviewers have their own deadlines and will give their own work higher priority.
  - Assigning concrete responsibilities to reviewers and / or taking an “egoless” (product centered, group buy-in) approach often works, but is difficult to realize
- Why not have the author act as reviewer?