

CMPT 373  
Software Development Methods

# Introduction

Nick Sumner  
wsumner@sfu.ca

# Introduction

---

- Who am I?
  - Nick Sumner ([wsumner@sfu.ca](mailto:wsumner@sfu.ca))
  - Research Faculty

# Introduction

---

- Who am I?
  - Nick Sumner (wsumner@sfu.ca)
  - Research Faculty
- Who is your TA?
  - Chandan Kumar Mishra

# Introduction

---

- Who am I?
  - Nick Sumner (wsumner@sfu.ca)
  - Research Faculty
- Who is your TA?
  - Chandan Kumar Mishra
- What is the course website?
  - <http://www.cs.sfu.ca/~wsumner/teaching/373/>
  - OR: just search for “CMPT 373 sumner”

# Introduction

---

- Who am I?
  - Nick Sumner (wsumner@sfu.ca)
  - Research Faculty
- Who is your TA?
  - Chandan Kumar Mishra
- What is the course website?
  - <http://www.cs.sfu.ca/~wsumner/teaching/373/>
  - OR: just search for “CMPT 373 sumner”
- **Where can you discuss course issues?**
  - CourSys  
(<https://courses.cs.sfu.ca/2016fa-cmpt-373-d1/discussion/>)

# What is this course?

---

- What have you heard?

# What is this course?

---

- What have you heard?
- My perspective... hands on experience
  - workflows
  - tools
  - project management
  - writing better code
  - dealing with a (possibly troublesome) customer
  - dealing with (and avoiding) problems

# What is this course?

---

- What have you heard?
- My perspective... hands on experience
  - workflows
  - tools
  - project management
  - writing better code
  - dealing with a (possibly troublesome) customer
  - dealing with (and avoiding) problems
- **Slightly different than many courses**
  - Less emphasis on “getting the right answer”
  - More emphasis on being aware & using the right skills



# Why take this course?

---

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”

# Why take this course?

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”

Start

Ideal

# Why take this course?

---

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”



# Why take this course?

---

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”



Ideal

# Why take this course?

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”

Start

Bad

# Why take this course?

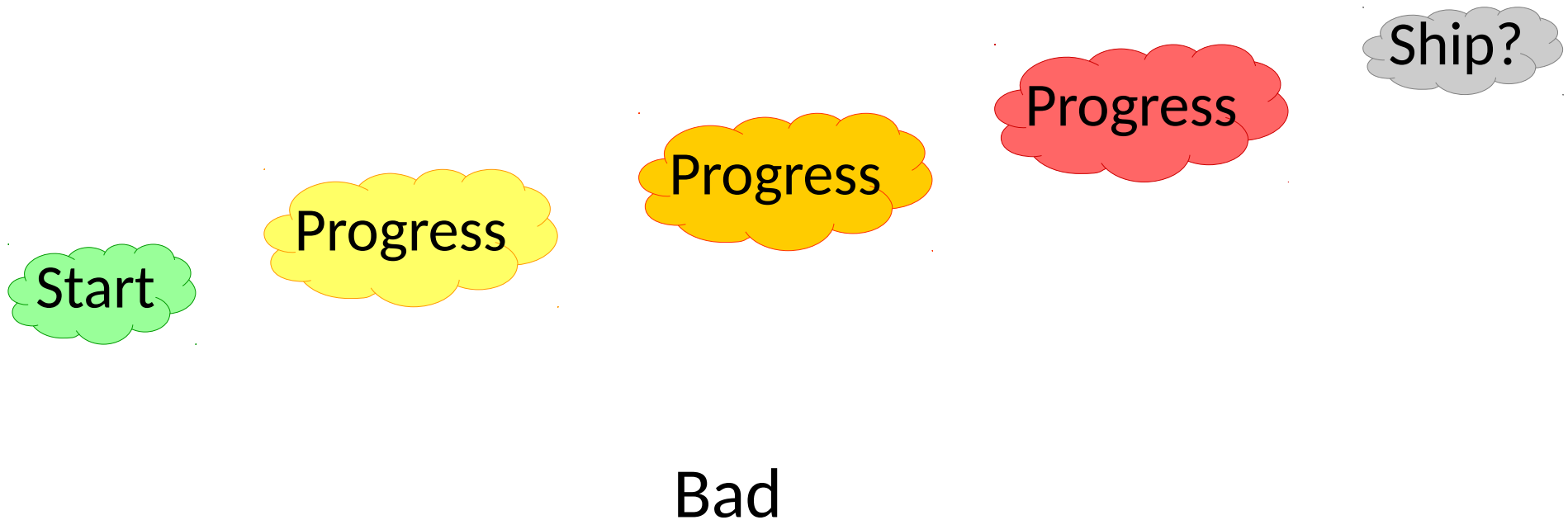
- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”



# Why take this course?

---

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”



# Why take this course?

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”

Start

Goal



# Why take this course?

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”



# Why take this course?

---

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”

Corrective Measures  
and Process



# Why take this course?

---

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”
- Most graduates with a CS degree are not ready
  - Software engineering is about *process* and *awareness*
  - Software development is a *craft* that requires practice

# Why take this course?

---

- Most software projects fail(!)
  - Up to 85% depending on definition of “*failure*”
- Most graduates with a CS degree are not ready
  - Software engineering is about *process* and *awareness*
  - Software development is a *craft* that requires practice
- **Hands on experience yields an advantage**
  - You can better understand how to create a product that has value both now and in the future.

# What will we be doing?

---

- On your own
  - Reading (From 2 books)
  - Exercises with tools

# What will we be doing?

---

- On your own
  - Reading (From 2 books)
  - Exercises with tools
- In groups / tutorials
  - One development project with unclear requirements

# What will we be doing?

---

- On your own
  - Reading (From 2 books)
  - Exercises with tools
- In groups / tutorials
  - One development project with unclear requirements
- In class
  - Introduction to tools and techniques
  - Discussions about the reading
  - Discussions about the tools
  - Discussions about code

# Grading

---

- Subject to change as necessary
- Breakdown:
  - (15%) Responses to reading
  - (15%) Participation in class discussions
  - (45%) Useful contribution to semester project
  - (25%) Exercises



# Reading

---

- Assigned chunks of reading
  - Often ~200 pages per 1-2 weeks
  - Both books are available as e-books in library

# Reading

---

- Assigned chunks of reading
  - Often ~200 pages per 1-2 weeks
  - Both books are available as e-books in library
- Responses
  - A 2-3 page critical reaction to the reading
  - Single spaced
  - Must include 3 units of:
    - A quote, with citation
    - 1-2 paragraphs discussing the quote
  - Relate the material to your own experiences
  - Form an opinion about it, and *justify* it

# Reading

---

- Assigned chunks of reading
  - Often ~200 pages per 1-2 weeks
  - Both books are available as e-books in library
- Responses
  - A 2-3 page critical reaction to the reading
  - Single spaced
  - Must include 3 units of:
    - A quote, with citation
    - 1-2 paragraphs discussing the quote
  - Relate the material to your own experiences
  - Form an opinion about it, and *justify* it
- **First assignment posted after class**

# Semester project

---

- You will interact with me as a customer in tutorials

# Semester project

---

- You will interact with me as a customer in tutorials
- The requirements of the project ***will change***

# Semester project

---

- You will interact with me as a customer in tutorials
- The requirements of the project *will change*
- You will use (and be evaluated in part on) skills from the exercises in the project

# Semester project

---

- You will interact with me as a customer in tutorials
- The requirements of the project *will change*
- You will use (and be evaluated in part on) skills from the exercises in the project
- **All requirements gathering & customer interaction happens in the tutorials**

# Semester project

---

- You will interact with me as a customer in tutorials
- The requirements of the project *will change*
- You will use (and be evaluated in part on) skills from the exercises in the project
- All requirements gathering & customer interaction happens in the tutorials
- **Different teams (of 8) may receive different requirements**



# Semester project

---

- You will interact with me as a customer in tutorials
- The requirements of the project *will change*
- You will use (and be evaluated in part on) skills from the exercises in the project
- All requirements gathering & customer interaction happens in the tutorials
- Different teams (of 8) may receive different requirements
- You should expect to *personally* contribute  $\geq 1K$  **quality SLOC** in order to receive a good grade

# Project code policy

---

All code pushed to a project repository may be viewed, analyzed, and critiqued by all students *in class* (even in future semesters).

# Project teams

---

- Assigned teams of 8

# Project teams

---

- Assigned teams of 8
- Following an informal scrum-like process
  - Each tutorial meeting will involve:
    - Discussion of what you did since the last meeting
    - What the present obstacles are to meeting goals
    - A plan for the next meeting

# Project teams

---

- Assigned teams of 8
- Following an informal scrum-like process
  - Each tutorial meeting will involve:
    - Discussion of what you did since the last meeting
    - What the present obstacles are to meeting goals
    - A plan for the next meeting
- **I will act as both customer & coach during tutorials**

# Discussions

---

- Code Review Fridays:

# Discussions

---

- Code Review Fridays:
  - Each group will submit ~100 lines of code each week by Wednesday, 10pm

# Discussions

---

- **Code Review Fridays:**
  - Each group will submit ~100 lines of code each week by Wednesday, 10pm
  - I'll review & select 1 or 2 to send to the class (I may choose some other code entirely)



# Discussions

---

- **Code Review Fridays:**
  - Each group will submit ~100 lines of code each week by Wednesday, 10pm
  - I'll review & select 1 or 2 to send to the class (I may choose some other code entirely)
  - **We will review the code together in class on Friday.**

# Discussions

---

- Code Review Fridays:
  - Each group will submit ~100 lines of code each week by Wednesday, 10pm
  - I'll review & select 1 or 2 to send to the class (I may choose some other code entirely)
  - We will review the code together in class on Friday.
- In class discussions of both code & readings focus thematically on one core issue:

**Complexity**

# Goals

---

- Writing good code as a team
  - Some teammates will write well from the beginning.
  - Some will need help from teammates.

# Goals

---

- **Writing good code as a team**
  - Some teammates will write well from the beginning.
  - Some will need help from teammates.
  - **Working together is the only real way.**

# Goals

---

- Writing good code as a team
  - Some teammates will write well from the beginning.
  - Some will need help from teammates.
  - Working together is the only real way.
  - This is just as true in industry.

# Goals

---

- Writing good code as a team
  - Some teammates will write well from the beginning.
  - Some will need help from teammates.
  - Working together is the only real way.
  - This is just as true in industry.
- **Manage complexity & change**
  - Requirements will change in practice.
  - I will try to change requirements that force design changes.

# Goals

---

- Writing good code as a team
  - Some teammates will write well from the beginning.
  - Some will need help from teammates.
  - Working together is the only real way.
  - This is just as true in industry.
- **Manage complexity & change**
  - Requirements will change in practice.
  - I will try to change requirements that force design changes.
  - Better designs & process will make the transitions easier.

And we're off...

---