University of Dublin
Trinity College

CS7CS3: Advanced Software Engineering
Introduction

Prof. Siobhán Clarke *(she/her)*
Ext. 2224 – L2.15
www.scss.tcd.ie/Siobhan.Clarke/

Prof. Siobhán Clarke

>240	publications
>9400	citations
40	h-index
>19M	competitive funding

Career Highlights

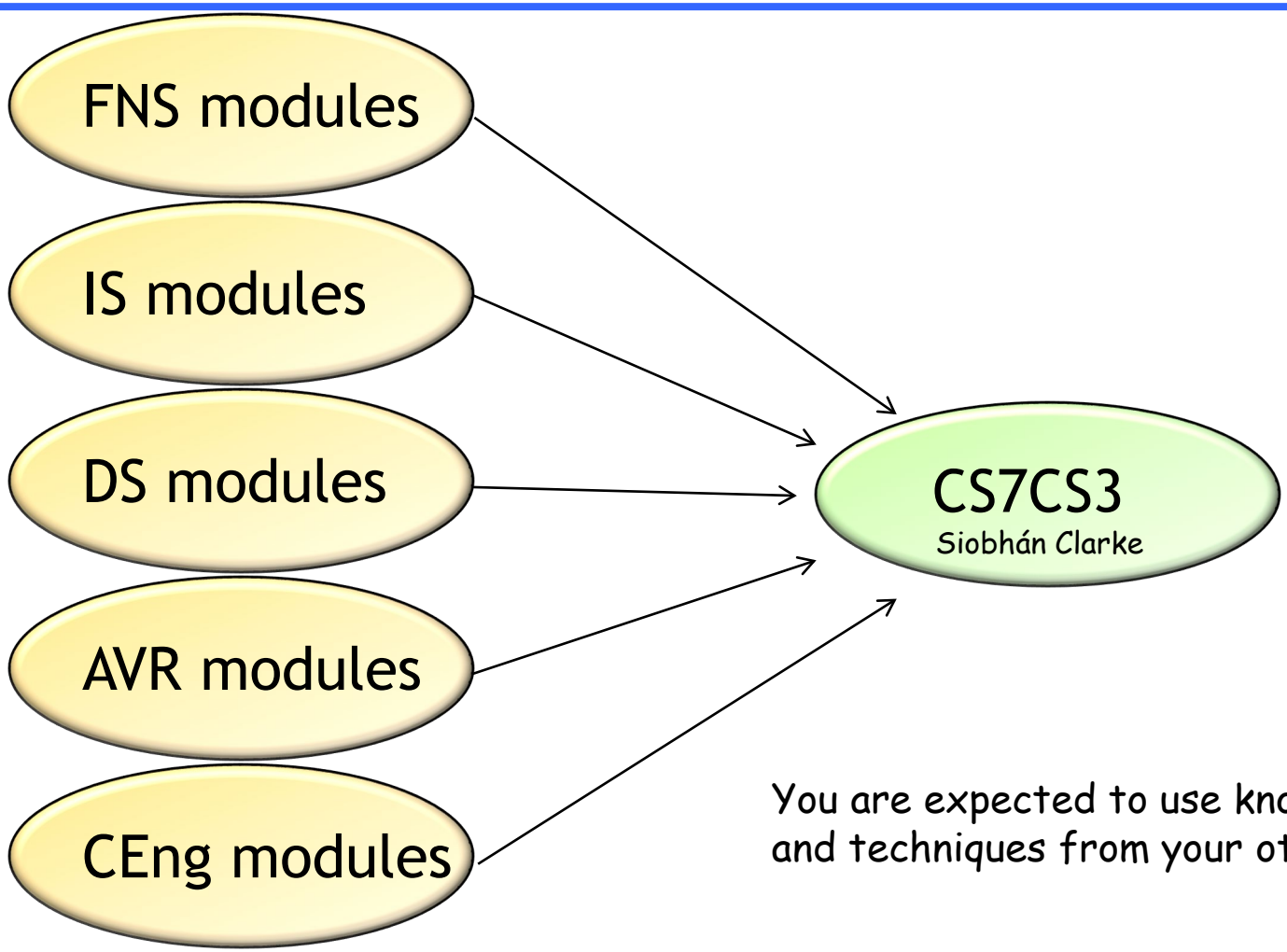
2019 – present	Professor of Software Systems, Trinity College Dublin
2018 – present	Director, SFI Enable Research Programme (Smart Cities/Communities and the IoT)
2018 – present	Head of the Discipline of Networks and Distributed Systems, School of Computer Science and Statistics.
2006 – present	Head of Distributed Systems Group, School of Computer Science and Statistics
2009 – 2011	Director of Teaching and Learning (Postgraduate), School of Computer Science & Statistics.
2006 – present	Fellow of Trinity College Dublin.
2000 – present	Faculty, Trinity College Dublin
1997 – 2000	PhD Candidate
1986 – 1997	Senior Software Engineer, IBM

Purpose of CS7CS3

- To be exposed to the theory of software engineering and team-based software development;
- To assess the emerging practices and technologies in software development;
- To assess the current state of the art in software engineering practice and research.

Course details on Blackboard

Module Context



You are expected to use knowledge and techniques from your other modules

Module Overview

Specific topics addressed in this module include:

- Software architecture;
- Agile process, in particular eXtreme Programming (XP);
- Test-driven development;
- Object-oriented design principles;
- Refactoring.

I will give lectures on these in the first few weeks. From then, you will be following these practices in your projects and we will discuss the practices, in your groups.

Module Assessment: 100% Coursework

Coursework evaluation is based on the end-of-year project documentation, a demonstration to the course lecturer and, where relevant, other stakeholders, an oral examination within teams, and peer assessment.

65% of the marks are group-wide, and

35% is individual.

Module Assessment: 100% Coursework Semester 1

Assessment Component	Brief Description	% of total	Week set (*)	Week due (*)
Send me group suggestions	Form your own groups where that is your preference. For students for whom that is not their preference, I will assign you to groups		3	7
Group Project requirements specification	Analyse and document the requirements for the group project	10% for group	10	14 (on 2nd December)
Group project architecture	Application of an appropriate architectural model in the team-based application assigned.	10% for group	10	14 (on 2nd December)
"Thin slice" implementation	Implementation of a thin slice of project functionality across ALL architectural components	10% for group	14	23 (on 3rd February)

Note (*) I am using the week numbers from College's academic year structure – see

https://www.tcd.ie/calendar/academic-year-structure/academic-year-structure.pdf

Note: details subject to change (e.g., deadlines, deliverable content) throughout the year. Any change will be communicated to the class.

Module Assessment: 100% Coursework Semester 2

Assessment Component	Brief Description	% of total	Week set (*)	Week due (*)
"Thin slice" implementation (started in semester 1)	Implementation of a thin slice of project functionality across ALL architectural components	10% for group	14	23 (on 3rd February)
Group development project	Evaluation is based on the end-of-year project documentation, a demonstration to the course lecturer and, where relevant, other stakeholders (or video), an oral examination within teams (if possible), and peer assessment. 40% of the marks are group-wide, and 40% is individual. Criteria for evaluation are: 1. Application of agile process to group project 2. Application of appropriate systems' algorithms in group project; 3. Code quality within group project code-base; Note, the default is that the individual mark will equal the group mark. This may change based on an individualised assessment, against the three evaluation criteria, which will be based on team-members' peer reviews, combined with lecturer/TA observation throughout the semester, and Q&A on software engineering theory.	35% for group; 35% for individual (**)	10	33 (on FRIDAY 14th April)

Note (*) I am using the week numbers from College's academic year structure – see

https://www.tcd.ie/calendar/academic-year-structure/academic-year-structure.pdf

Note (**) Any student who fails the individual component (<17 out of 35), will have been deemed to have failed the module.

Note: details subject to change (e.g., deadlines, deliverable content) throughout the year. Any change will be communicated to the class.

So, Software Engineering. Why bother?

Software engineering is hard!

- academic discussions/lectures seem useless!

  easy to code small systems/apps
  easy to figure out what code to write with small problems
  easy to manage communication with 1 person team
  easy to integrate with code you write yourself
  easy to forget about when maintenance not required

- school of hard knocks gives better appreciation

Quick poll:

How many years PROFESSIONAL software engineering experience to you have?

- Option 1: None or University internship only
- Option 2: <2 years
- Option 3: 2-4 years
- Option 4: 4-8 years
- Option 5: > 8 years

Quick poll:

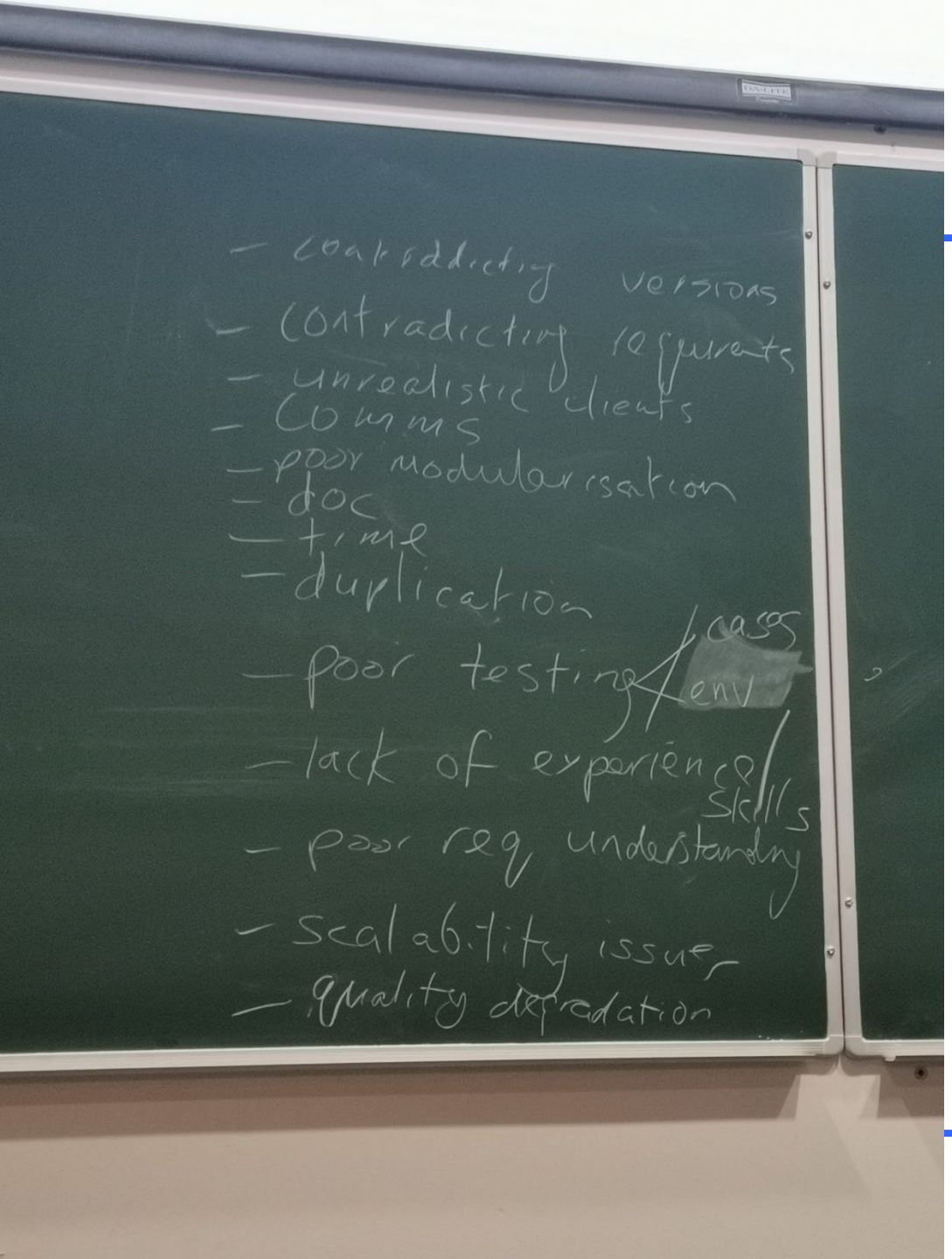
What is the biggest software development team you have worked on?

- Option 1: <= 4 people
- Option 2: 5-8 people
- Option 3: 9-12 people
- Option 4: 13-20 people
- Option 5: >20 people

Quick poll:

Which of the following describes the largest project with the biggest team you have worked on?

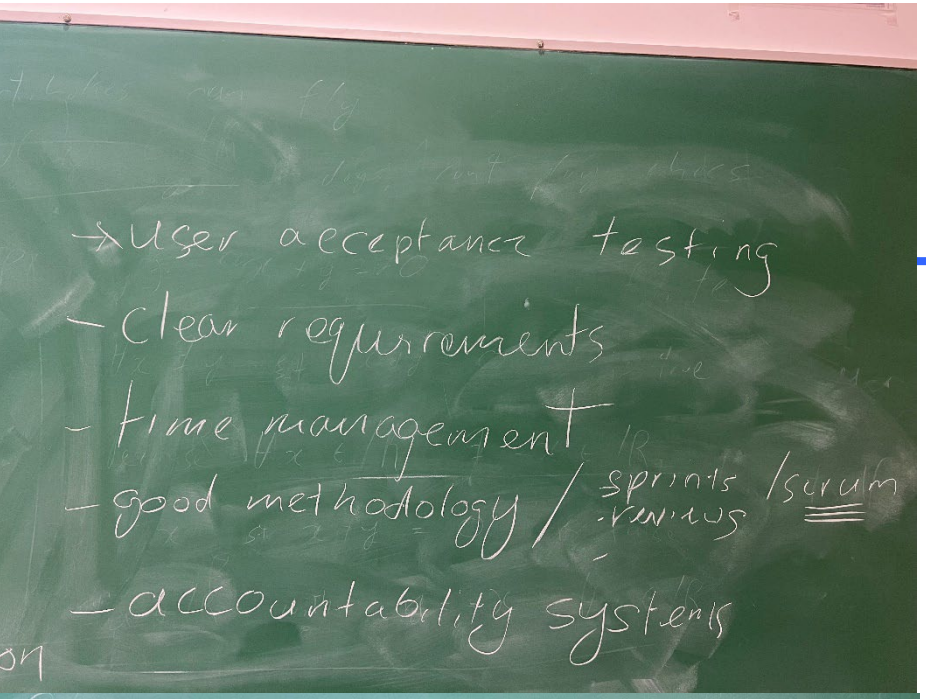
- Option 1: Very successful indeed!
- Option 2: Somewhat successful overall
- Option 3: Could have been better, could have been worse
- Option 4: Mostly unsuccessful
- Option 5: Complete disaster

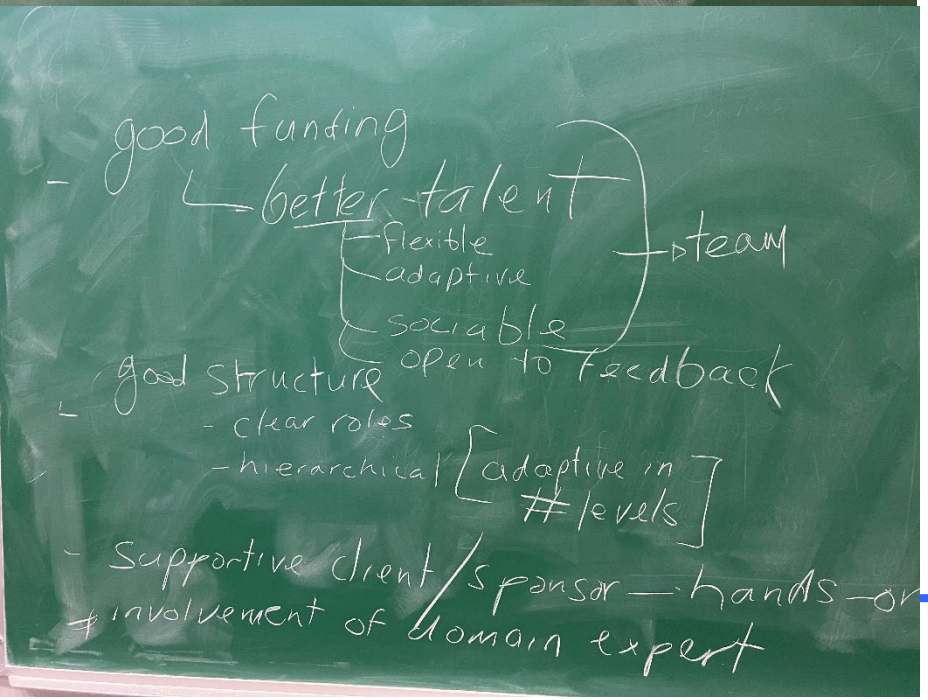


Why do projects fail?

What are the factors that contributed to any **failures** in the projects you worked on?

What makes projects succeed?





What are the factors that contributed to any **successes** in the projects you worked on?

Coming up:

Software Functional Architectures