DAAD
Deutscher Akademischer Austausch Dienst
German Academic Exchange Service

"Ss. Cyril and Methodius" University in Skopje
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

ADP COURSE EVOLUTION – SIX YEARS LONG JOURNEY

N. Ackovska and M. Kostoska
Faculty of Computer Science and Engineering
University St. Cyril and Methodius, Skopje, Macedonia
nevena.ackovska@finki.ukim.mk
magdalena.kostoska@finki.ukim.mk

16th Workshop on "Software Engineering and Reverse Engineering"
Jahorina, Bosnia and Hercegovina, 21-27 August 2016

Overview

- Basics
- The team
- Course specifics
  - Topics
  - The groups
- Results
- Conclusions

Architecture, design and patterns (ADP) Basics

MSc level course

Course syllabus created as part of the TEMPUS Software Engineering studies

Our first experience with the course – 2010

Very different groups of students per year

- Groups vary from 12 – 2 students
- Diverse backgrounds

The team of ADP in Skopje

Nevena Ackovska
- Lectures

Magdalena Kostoska
- Exercises
- Valuable team member

Experts from the industry

Very colorful group of students

33 students

11 working in Software Industry – manager level

- Know it all!

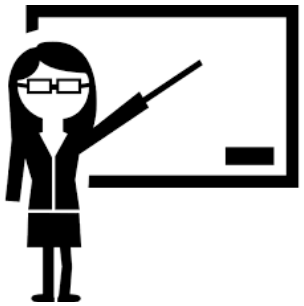
18 working in Software Industry – developers

- Do it all!

1 in education

2 unemployed (non IT background, during the studies)

1 abroad (during studies)



Students' specifics

About 1/2 studied Design and software architecture and other Software engineering courses during Undergraduate studies

Most of the students have good knowledge of specific programming language
- Some students were more familiar with C# and .NET platform, rather then Java

Small number of students didn't know programming (!!!)
- Dropped the course during the lecture time

Most good in delivering group software projects
- As leaders or team members
- How good are they on their own???

The structure of the classes

Weekend type, Fridays afternoons, other afternoons…

- Most of the students work

Lectures were pretty standard

- But changed with the evolution of the knowledge of students an IT industry in general

Java and C# based exercises, homework and projects

- Magdalena Kostoska takes care

Topics

- Introduction to Software Architecture
- Analogy with Classical Architecture
- The Deliverables of SA
- Elements of SA
- Analysis and Evaluation of SA
- Architecture, processes, and organization
- Model Driven Architecture
- Design patterns
- Frameworks and tools
- Refactoring
- Design characteristics and metrics

SDA - Topics

Introduction to software architecture, design and patterns

Design patterns

- Factory, Prototype, Composite, Adapter, Decorator, Observer, Template Method, Strategy and finally MVC

Refactoring

- Introduction to refactoring and usage

More about SA modularity, cohesion and examples of specific software architecture

Additional upgrade

Students ask to have greater intro in Patterns and Design in general

- Although most of the students get a glimpse on this during Undergrad studies, they lack real experience → project

Added 1 expert lecture

- Seminar on Software Processes and Structure
- Whole day event
- Listen to ex students who did well

Projects

3 possible types

Java and C# based coding of bigger software problems

Design and Patterns usage: given specific task and choice to use Java or C#

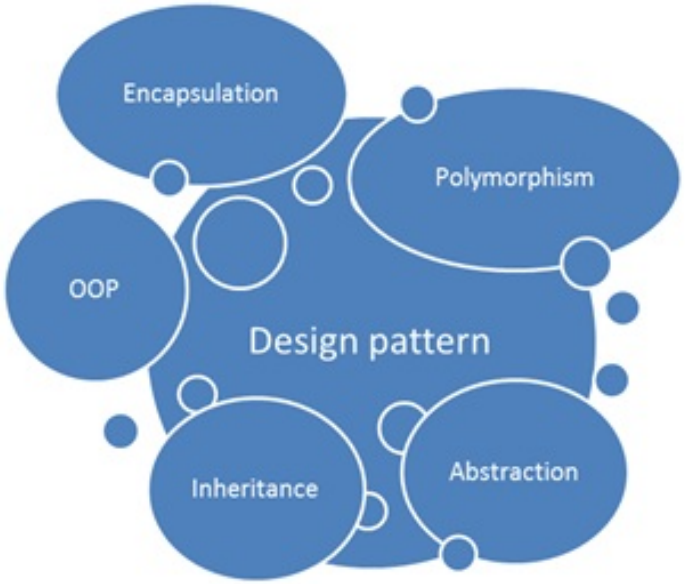
Upgrade of the living projects in the companies they work in

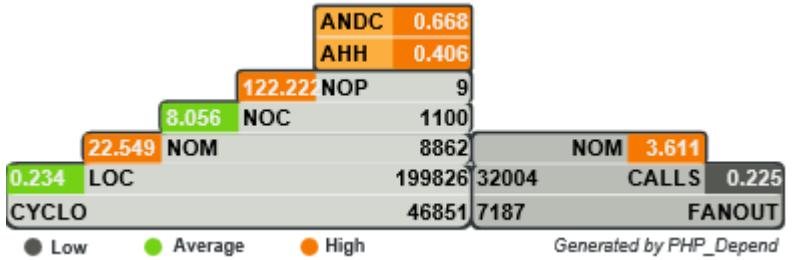
2 fold benefit – for the student and for the company

Refactoring: two bigger software project are given in two programming languages: Java and C#, student choose one of the two offered projects

Recently students ask to do that or their company projects

Metrics: run the metrics for an existing system





ADP – Grading

Two projects

- 2010-13: patterns & refactoring
- 2013 onward (depending of the background knowledge):
  - *Patterns & (metrics || refactoring)*

Results

19 students passed – Mostly grades 10 or 9

(+ 5 in progress not in the analytics)

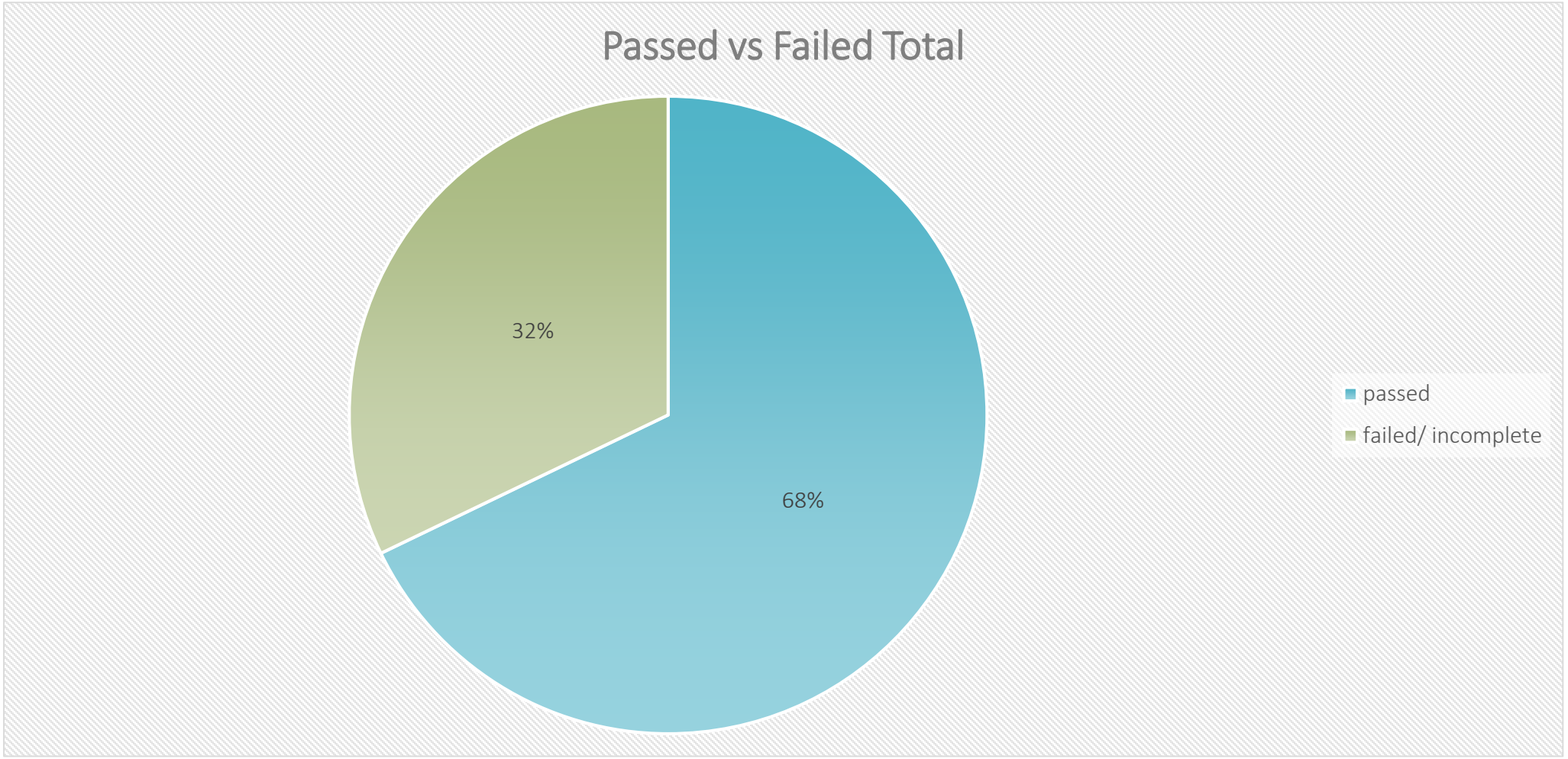
6 inactive

Got the tasks and never came back

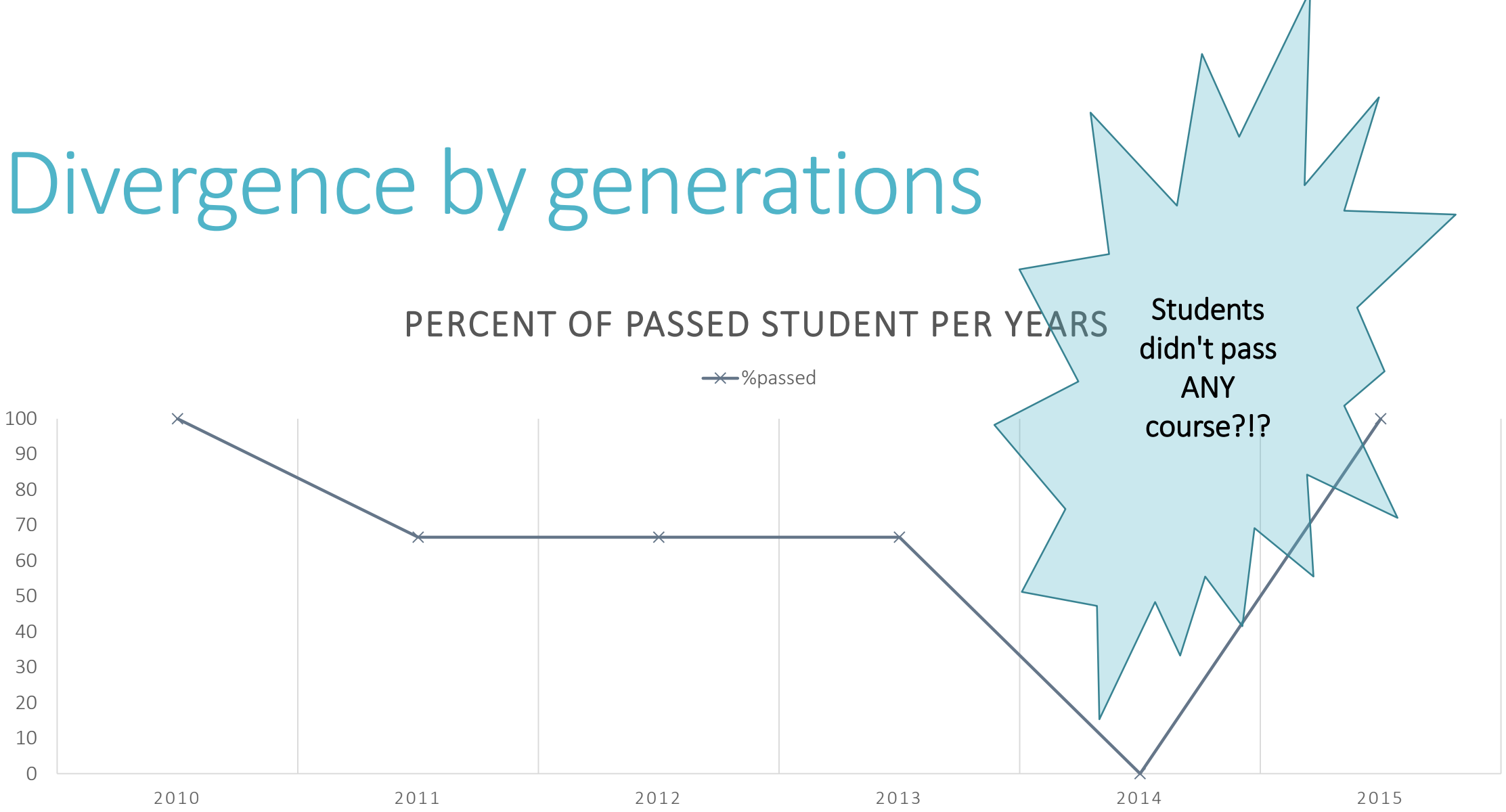
2 left the country during the studies

1 quit the studies

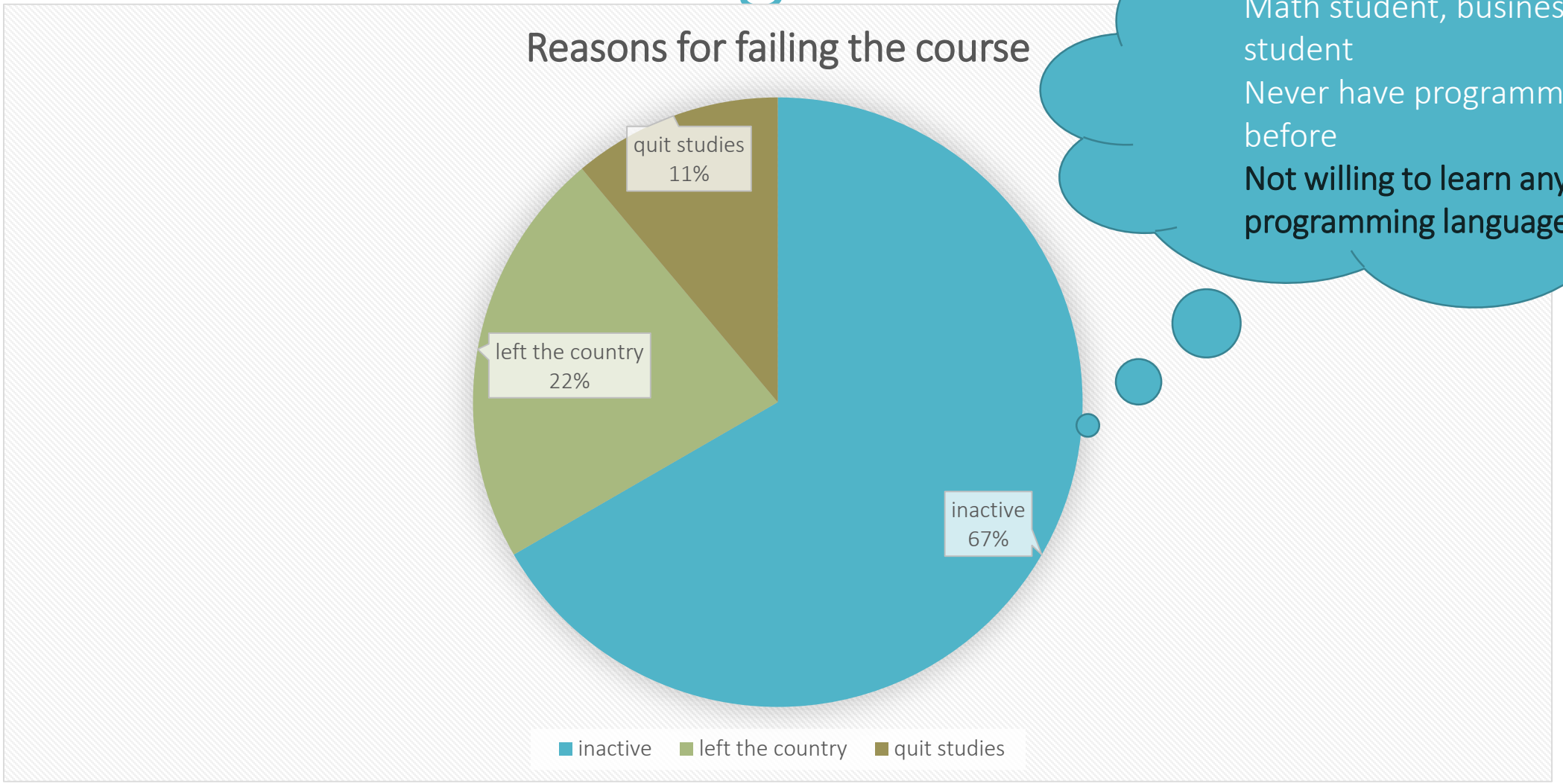
The throughput



Divergence by generations



Reasons for failing the cours



2 students can't pass – EVER!
Math student, business student
Never have programmed before
Not willing to learn any programming language

Something to think about

We want students from different backgrounds

- But they have to be prepared to learn

Should we allow students that don’t want to learn programming on software master studies?

- Should take care upon admission

Unexpected results

True/False grade distribution (10 or 9, or nothing!!!)

Master Theses chosen that evolve from this course

Students really liked the invited guest
- Real
  - *implementation*
  - *software processes*
  - *Structure evolves*

Students realize that in order to go forward, one must know at least the current technology
- Good to be great in a specific technology, but one has to be aware of the evolution

How we did so far?

The dynamics is …
- … Good!
- 5 master thesis
  - *3 finished, 2 in progress*

Even with such a diverse group

Keep it colorful, but up-to-date and useful
- Teacher – student routine gets better with
  - *Invited guests*
  - *Talk about work*
  - *Do for work*

Conclusions

The suggested lectures worked well, but we added extra lessons, and shortened some
- According to the newly accredited undergraduate studies
- According to the students' interests

We get very diverse group of students
- Some standardization upon submission might be needed

Some students really like the course
- Obtained MSc in following subject
- Implemented the knowledge in their companies
  - *Gain both for the company and for the student*

Questions?