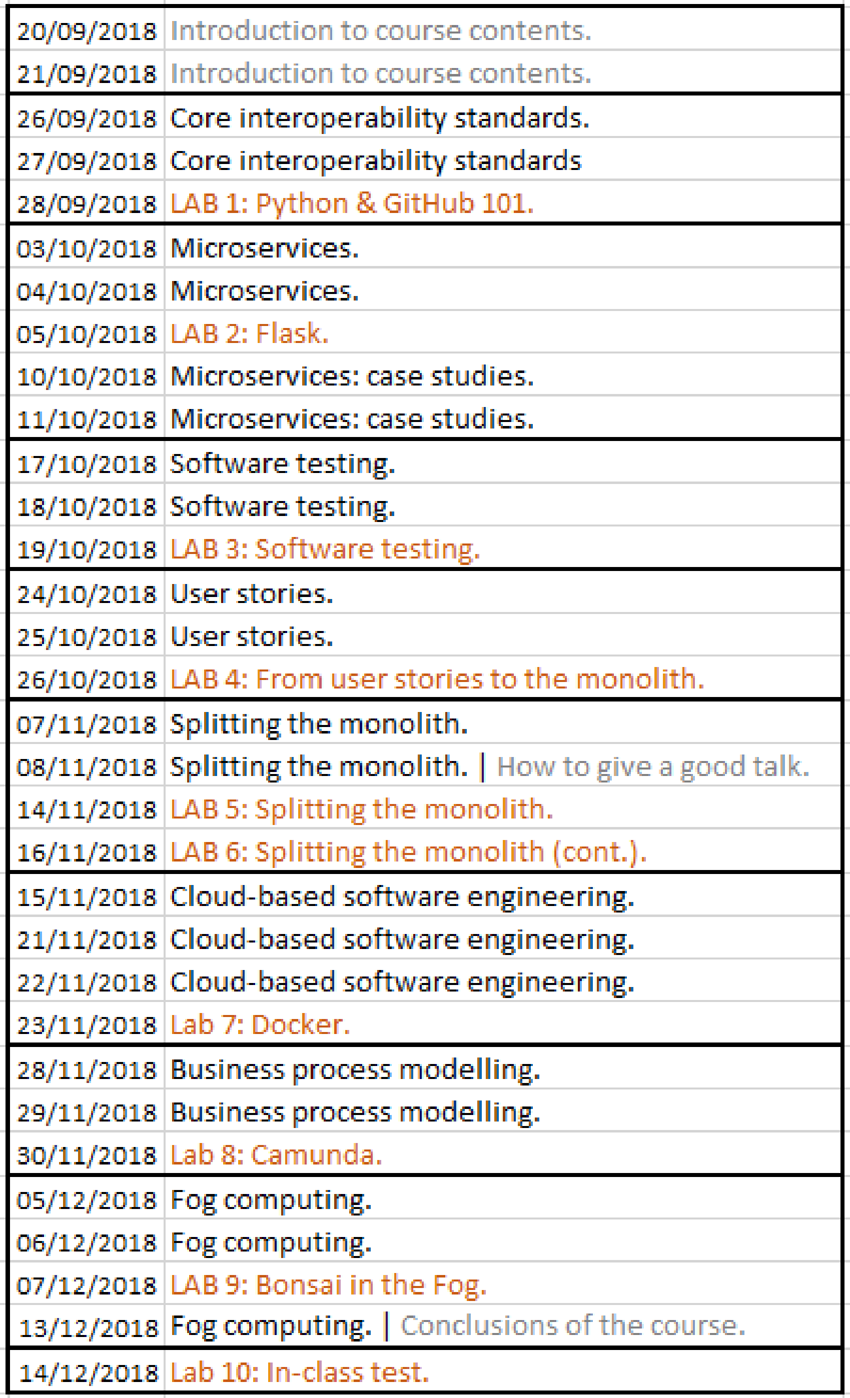
Concluding remarks

Antonio Brogi

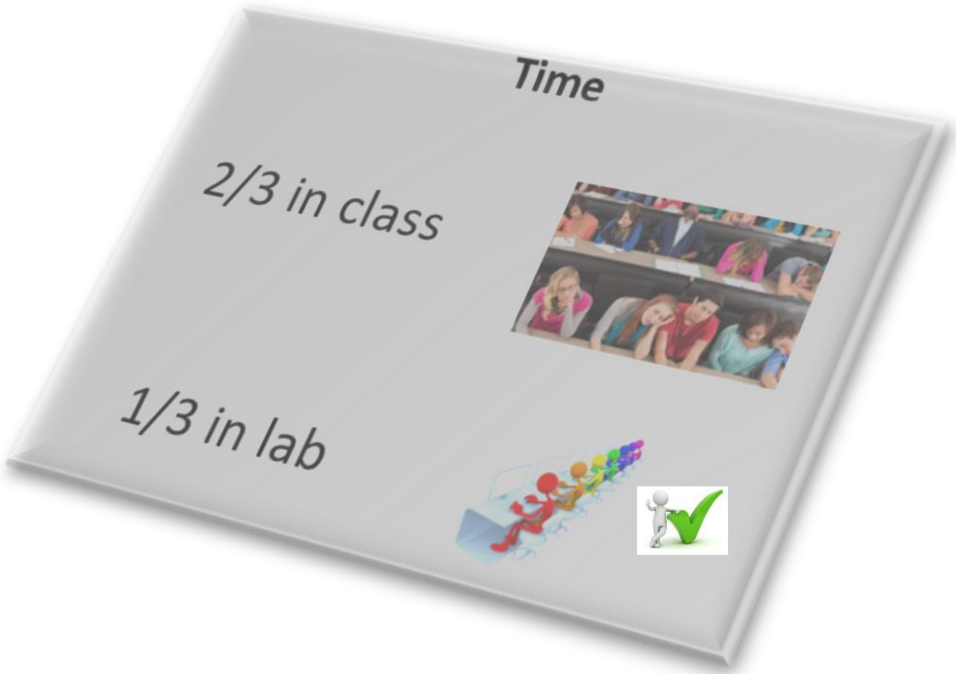
Department of Computer Science
University of Pisa

Classes and labs



20/09/2018	Introduction to course contents.
21/09/2018	Introduction to course contents.
26/09/2018	Core interoperability standards.
27/09/2018	Core interoperability standards
28/09/2018	LAB 1: Python & GitHub 101.
03/10/2018	Microservices.
04/10/2018	Microservices.
05/10/2018	LAB 2: Flask.
10/10/2018	Microservices: case studies.
11/10/2018	Microservices: case studies.
17/10/2018	Software testing.
18/10/2018	Software testing.
19/10/2018	LAB 3: Software testing.
24/10/2018	User stories.
25/10/2018	User stories.
26/10/2018	LAB 4: From user stories to the monolith.
07/11/2018	Splitting the monolith.
08/11/2018	Splitting the monolith. | How to give a good talk.
14/11/2018	LAB 5: Splitting the monolith.
16/11/2018	LAB 6: Splitting the monolith (cont.).
15/11/2018	Cloud-based software engineering.
21/11/2018	Cloud-based software engineering.
22/11/2018	Cloud-based software engineering.
23/11/2018	Lab 7: Docker.
28/11/2018	Business process modelling.
29/11/2018	Business process modelling.
30/11/2018	Lab 8: Camunda.
05/12/2018	Fog computing.
06/12/2018	Fog computing.
07/12/2018	LAB 9: Bonsai in the Fog.
13/12/2018	Fog computing. | Conclusions of the course.
14/12/2018	Lab 10: In-class test.

64 hours: 44 in class + 20 in lab

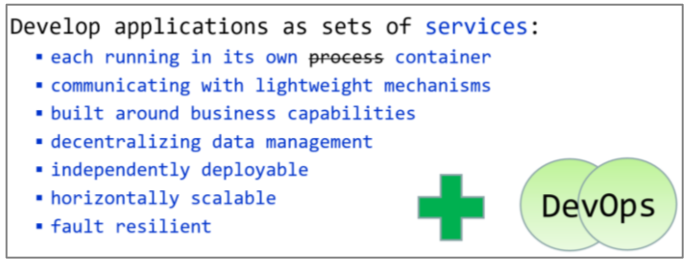


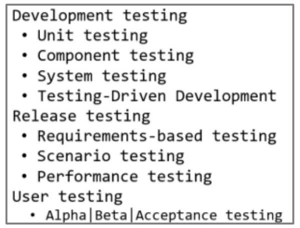
Classes and labs
Syllabus

20/09/2018	Introduction to course contents.
21/09/2018	Introduction to course contents.
26/09/2018	Core interoperability standards.
27/09/2018	Core interoperability standards
28/09/2018	LAB 1: Python & GitHub 101.
03/10/2018	Microservices.
04/10/2018	Microservices.
05/10/2018	LAB 2: Flask.
10/10/2018	Microservices: case studies.
11/10/2018	Microservices: case studies.
17/10/2018	Software testing.
18/10/2018	Software testing.
19/10/2018	LAB 3: Software testing.
24/10/2018	User stories.
25/10/2018	User stories.
26/10/2018	LAB 4: From user stories to the monolith.
07/11/2018	Splitting the monolith.
08/11/2018	Splitting the monolith. | How to give a good talk.
14/11/2018	LAB 5: Splitting the monolith.
16/11/2018	LAB 6: Splitting the monolith (cont.).
15/11/2018	Cloud-based software engineering.
21/11/2018	Cloud-based software engineering.
22/11/2018	Cloud-based software engineering.
23/11/2018	Lab 7: Docker.
28/11/2018	Business process modelling.
29/11/2018	Business process modelling.
30/11/2018	Lab 8: Camunda.
05/12/2018	Fog computing.
06/12/2018	Fog computing.
07/12/2018	LAB 9: Bonsai in the Fog.
12/12/2018	Fog computing. | Conclusions of the course.
14/12/2018	Lab 10: In-class test.

REST

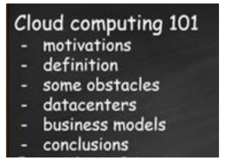
(XML) SOAP WSDL

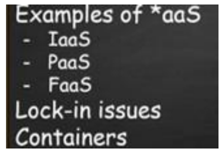

Develop applications as sets of services:
- each running in its own ~~process~~ container
- communicating with lightweight mechanisms
- built around business capabilities
- decentralizing data management
- independently deployable
- horizontally scalable
- fault resilient


Development testing
- Unit testing
- Component testing
- System testing
- Testing-Driven Development

Release testing
- Requirements-based testing
- Scenario testing
- Performance testing

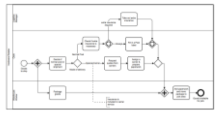
User testing
- Alpha|Beta|Acceptance testing

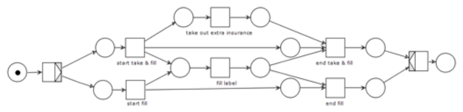

Cloud computing 101
- motivations
- definition
- some obstacles
- datacenters
- business models
- conclusions

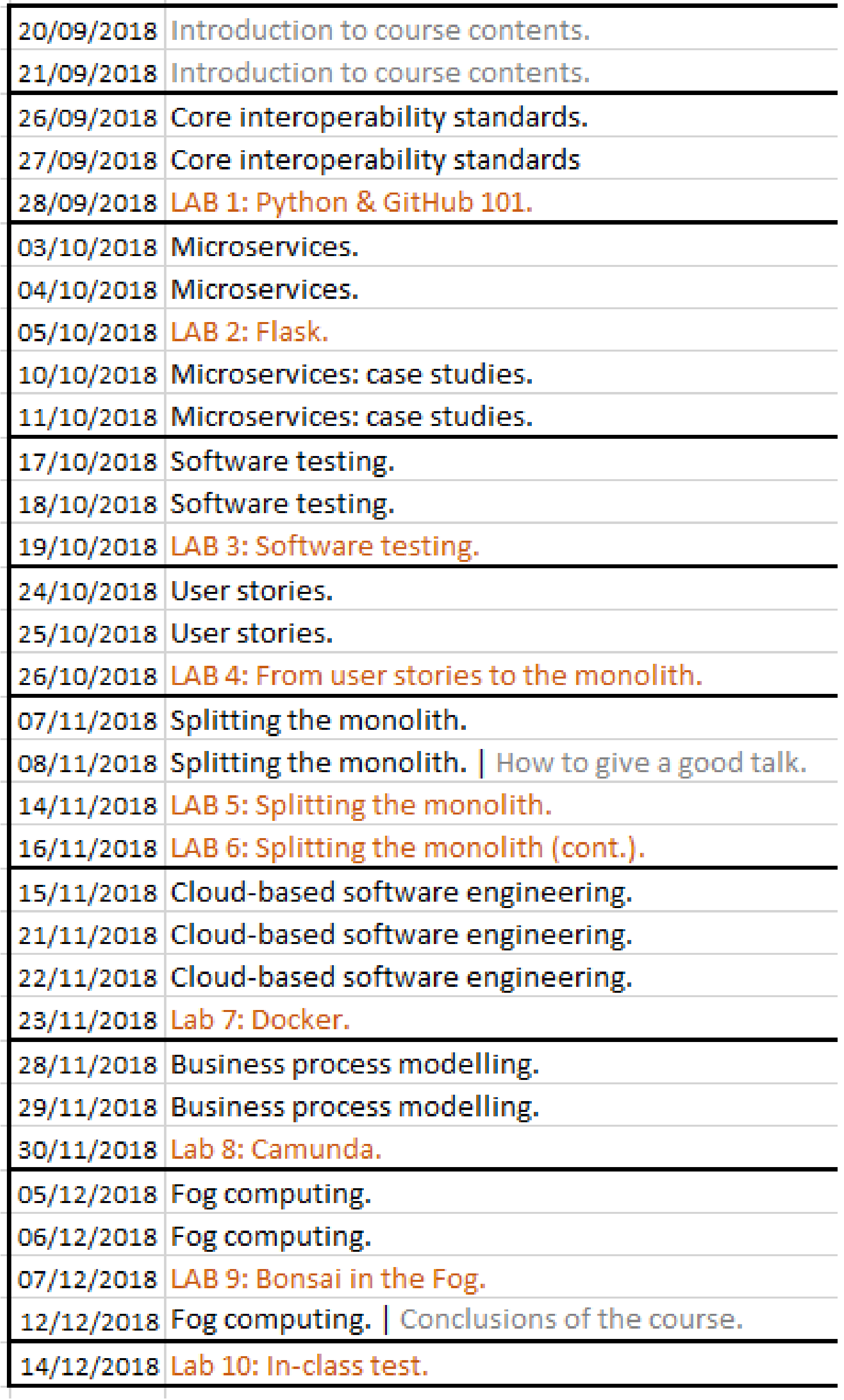

Examples of *aaS
- IaaS
- PaaS
- FaaS

Lock-in issues

Containers







20/09/2018	Introduction to course contents.
21/09/2018	Introduction to course contents.
26/09/2018	Core interoperability standards.
27/09/2018	Core interoperability standards
28/09/2018	LAB 1: Python & GitHub 101.
03/10/2018	Microservices.
04/10/2018	Microservices.
05/10/2018	LAB 2: Flask.
10/10/2018	Microservices: case studies.
11/10/2018	Microservices: case studies.
17/10/2018	Software testing.
18/10/2018	Software testing.
19/10/2018	LAB 3: Software testing.
24/10/2018	User stories.
25/10/2018	User stories.
26/10/2018	LAB 4: From user stories to the monolith.
07/11/2018	Splitting the monolith.
08/11/2018	Splitting the monolith. | How to give a good talk.
14/11/2018	LAB 5: Splitting the monolith.
16/11/2018	LAB 6: Splitting the monolith (cont.).
15/11/2018	Cloud-based software engineering.
21/11/2018	Cloud-based software engineering.
22/11/2018	Cloud-based software engineering.
23/11/2018	Lab 7: Docker.
28/11/2018	Business process modelling.
29/11/2018	Business process modelling.
30/11/2018	Lab 8: Camunda.
05/12/2018	Fog computing.
06/12/2018	Fog computing.
07/12/2018	LAB 9: Bonsai in the Fog.
12/12/2018	Fog computing. | Conclusions of the course.
14/12/2018	Lab 10: In-class test.

Syllabus

Core interoperability standards

Microservices

Software testing

User stories

Splitting the monolith

Cloud-based software engineering

Business process modelling

Fog computing

Core interoperability standards [1]

XML, REST, SOAP, WSDL

Microservices [2]

Motivations, definition, properties, case studies

Software testing [3]

Development testing, release testing, user testing

User stories [4]

Agile principles and user stories, examples

Splitting the monolith [5]

Code splitting, data splitting and transactions

Cloud-based software engineering [6,7]

Cloud computing 101 (definition, service models, deployment models, datacenters, business models), examples of IaaS, PaaS, FaaS, lock-in issues, containers

Business process modelling [8,1]

Business process models, BPMN, workflow nets

Fog computing [9,10,11]

Fog computing 101 (definition, characteristics, research challenges), QoS-aware app deployment in the fog

Reading list

Besides the slides used in class:

[1] A. Brogi, S. Forti. *Advanced Software Engineering – Lecture Notes*. 2018.

[2] J. Lewis, M. Fowler. *Microservices*. ThoughtWorks. 2014.

[3] I. Sommerville. *Software engineering*. Pearson. 2016. [Chapter 8]

[4] S.W. Ambler. *User stories: an agile introduction*.

[5] S. Newman. *Building microservices*. O'Reilly. 2015. [Chapter 5]

[6] R. Buyya, C. Vecchiola, T. Selvi. *Mastering Cloud Computing. Morgan Kaufmann*. 2013. [Section 1.1]

[7] I. Miell, A.H. Sayers. *Docker in practice*. Manning. 2016. [Chapter 1]

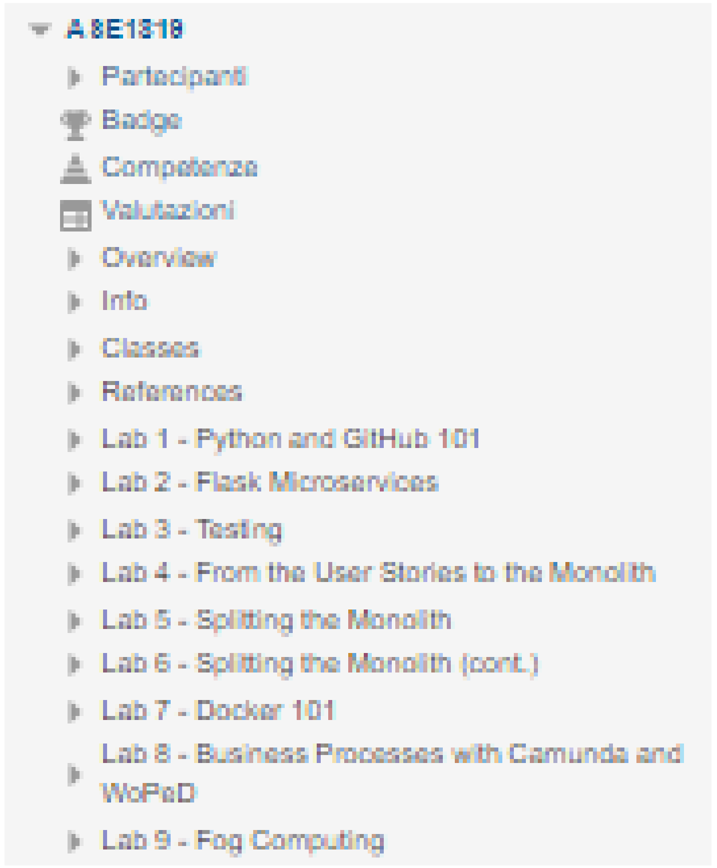
[8] OMG. *BPMN 2.0 by example*. 2010. [Section 5]

[9] A. Dastjerdi, R. Buyya. Fog Computing: *Helping the Internet of Things Realize Its Potential*. IEEE Computer 49(8): 112-116, 2016.

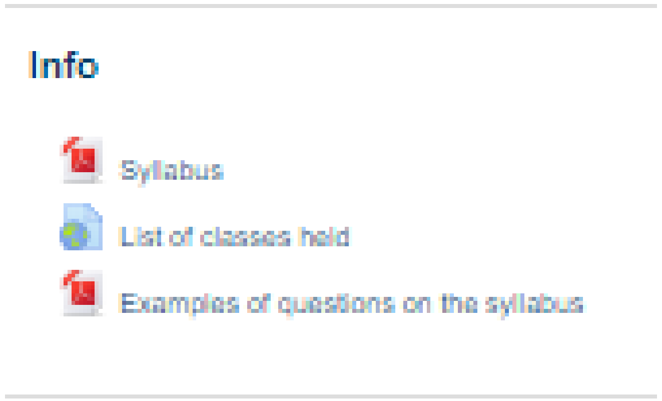
[10] A. Brogi, S. Forti, A. Ibrahim. *Deploying Fog Applications: How Much Does It Cost, By the Way?* Proceedings of CLOSER 2018, pages 68-77. 2018.

[11] A. Brogi, G. Ferrrari, S. Forti. *Secure Apps in the Fog: Anything to Declare?* Proceedings of CLOUDWAYS 2018.. (In press.)

Classes and labs
Syllabus
Everything in the Moodle!

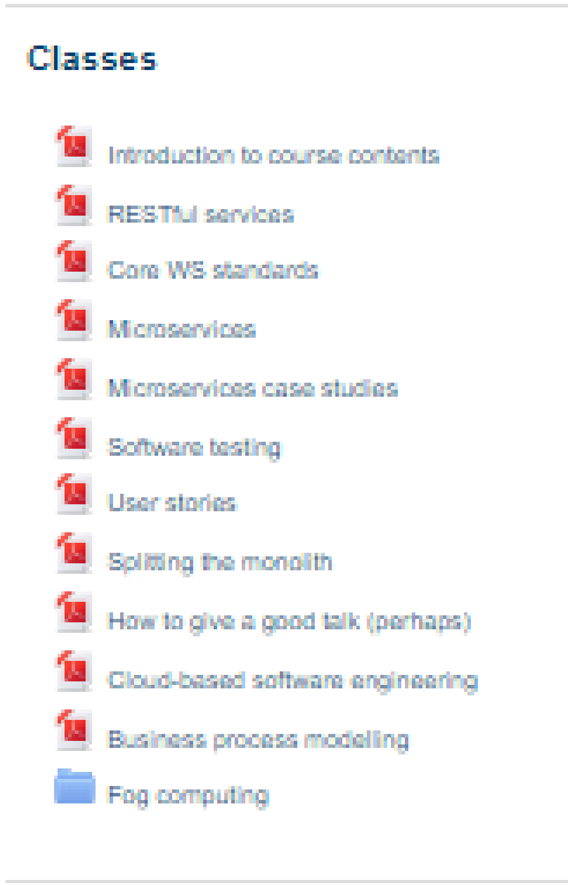


- A 8E1819
  - Partecipanti
  - Badge
  - Competenze
  - Valutazioni
  - Overview
  - Info
  - Classes
  - References
  - Lab 1 - Python and GitHub 101
  - Lab 2 - Flask Microservices
  - Lab 3 - Testing
  - Lab 4 - From the User Stories to the Monolith
  - Lab 5 - Splitting the Monolith
  - Lab 6 - Splitting the Monolith (cont.)
  - Lab 7 - Docker 101
  - Lab 8 - Business Processes with Camunda and WoPeD
  - Lab 9 - Fog Computing



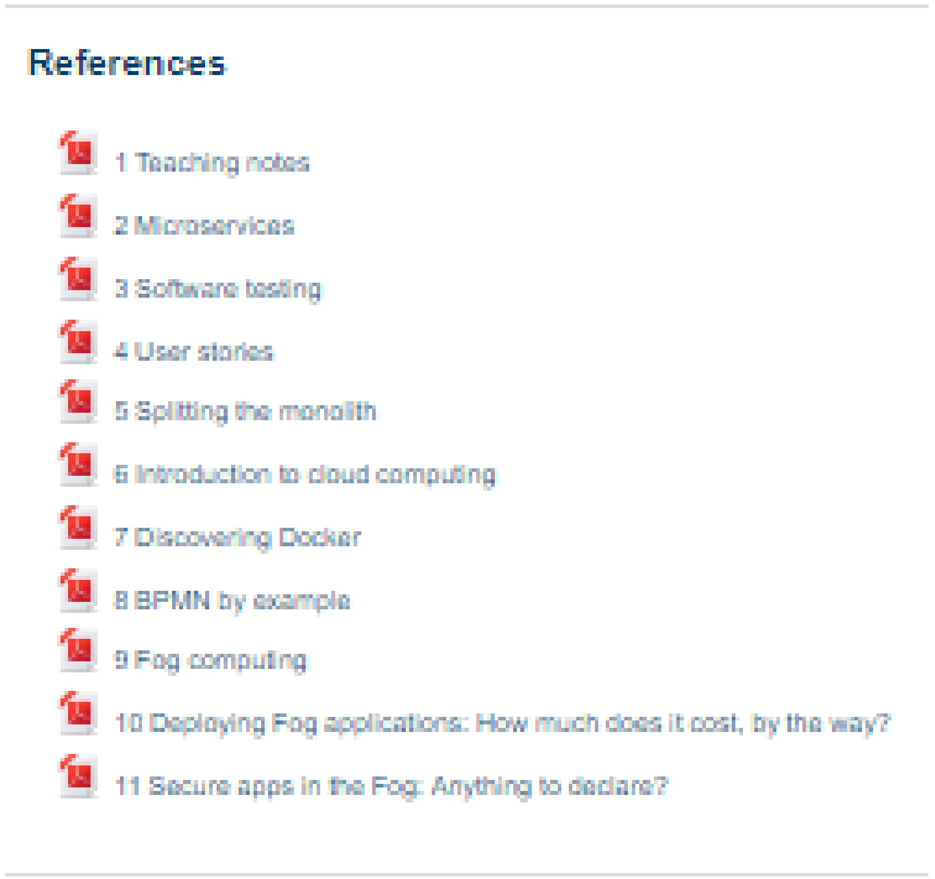
Info

- Syllabus
- List of classes held
- Examples of questions on the syllabus



Classes

- Introduction to course contents
- RESTful services
- Core WS standards
- Microservices
- Microservices case studies
- Software testing
- User stories
- Splitting the monolith
- How to give a good talk (perhaps)
- Cloud-based software engineering
- Business process modelling
- Fog computing

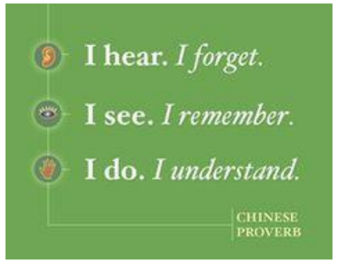


References

- 1 Teaching notes
- 2 Microservices
- 3 Software testing
- 4 User stories
- 5 Splitting the monolith
- 6 Introduction to cloud computing
- 7 Discovering Docker
- 8 BPMN by example
- 9 Fog computing
- 10 Deploying Fog applications: How much does it cost, by the way?
- 11 Secure apps in the Fog: Anything to declare?

Classes and labs
Syllabus
Everything in the Moodle!
Course self-assessment

The "Colorado river"



I hear. *I forget.*

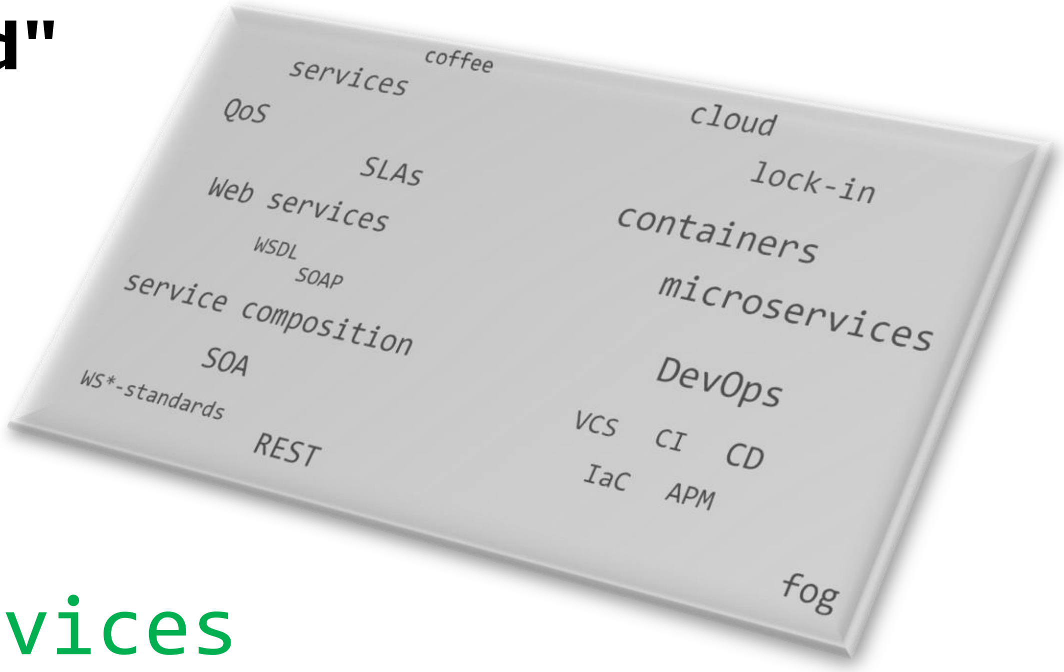
I see. *I remember.*

I do. *I understand.*

CHINESE PROVERB

- User stories
- Testing
- Monolith
- Microservices
- Improving service interactions
- Monitoring
- Securing
- Packaging
- Deploying

The "word cloud"



coffee

services

QoS

SLAs

Web services

WSDL

SOAP

service composition

SOA

business process modelling

WS*-standards

REST

cloud

lock-in

containers

microservices

testing

DevOps

user stories

VCS CI CD

IaC APM

fog

+ & -

+ "contemporary" contents
+ thematic weeks & focussed labs
+ homeworks, group homeworks

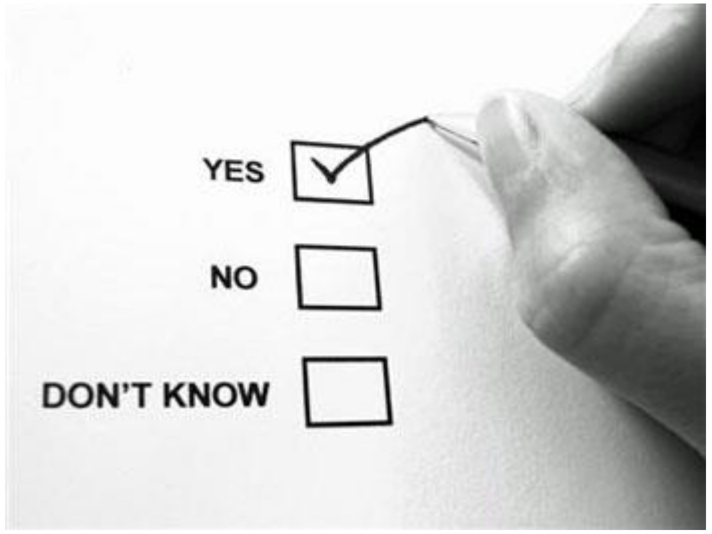
(- big effort to renew course)

auar inglisch uos anderstendabol, ui op

Of course, there is room for improving ...

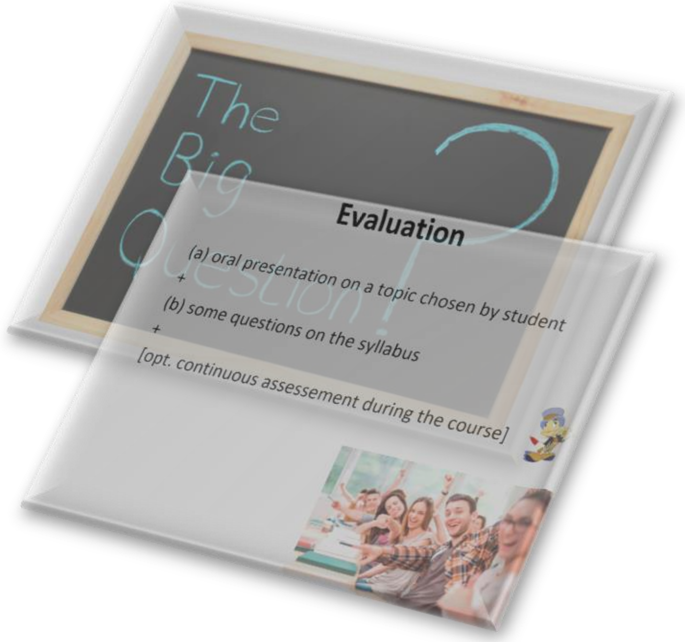
- more on QoS/SLAs?
- more on DevOps tools?
- more on design patterns??
- less homework assignments?
- …

Please do not forget to fill the course(s) questionnaire



Classes and labs
Syllabus
Everything in the Moodle!
Course self-assessment
The big question

How to get the credits


The Big Question?

Evaluation

(a) oral presentation on a topic chosen by student
+
(b) some questions on the syllabus
+
[opt. continuous assessement during the course]

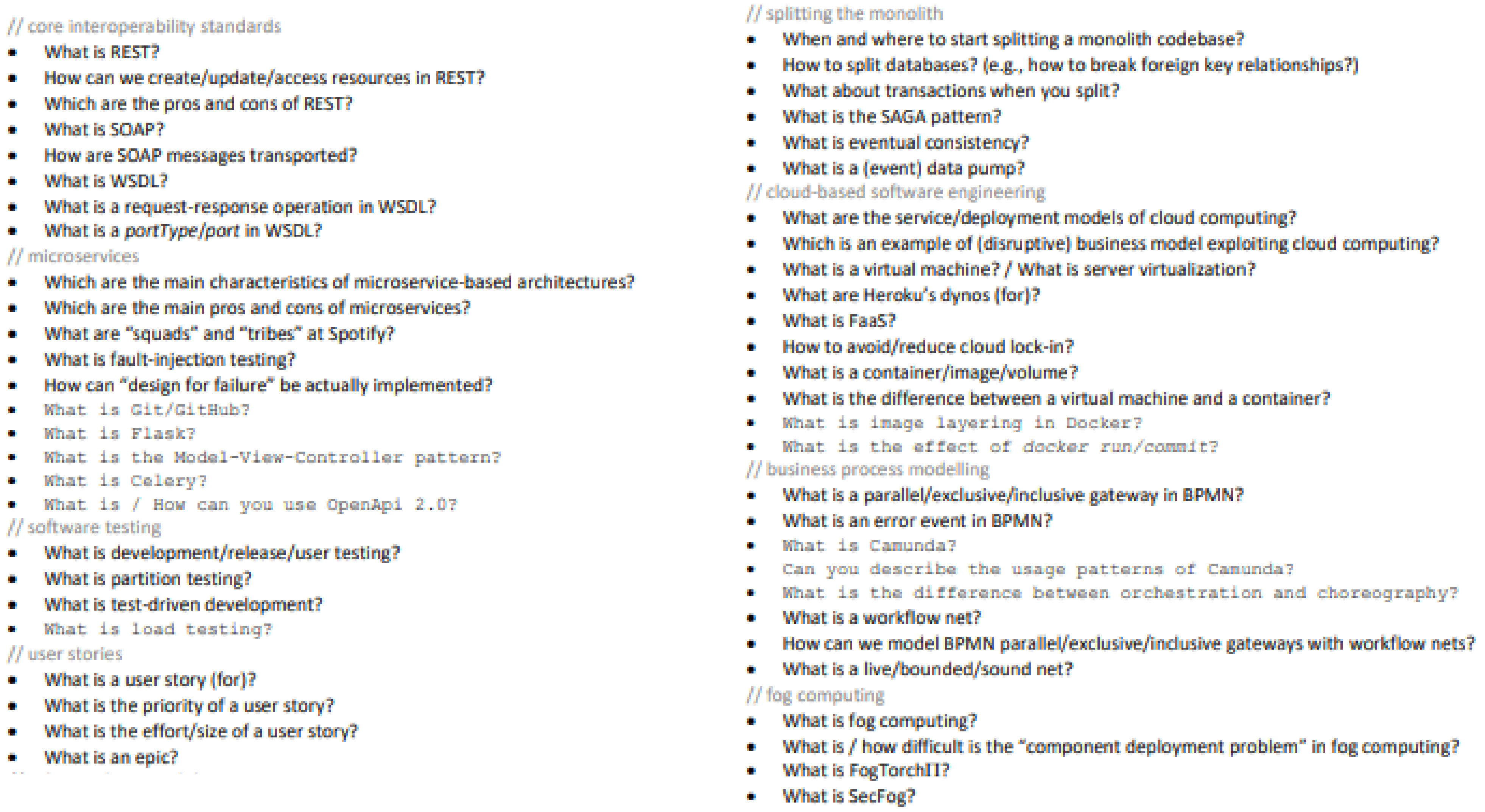
(a) oral **presentation** on a topic chosen/proposed by the student

+

(b) some questions on the topics of **syllabus**

To get a topic assigned for your oral presentation, you can pick one of the available topics available in the Moodle site or email the Instructor to propose a new topic

(Examples of) questions on the syllabus



// core interoperability standards

- What is REST?
- How can we create/update/access resources in REST?
- Which are the pros and cons of REST?
- What is SOAP?
- How are SOAP messages transported?
- What is WSDL?
- What is a request-response operation in WSDL?
- What is a *portType/port* in WSDL?

// microservices

- Which are the main characteristics of microservice-based architectures?
- Which are the main pros and cons of microservices?
- What are "squads" and "tribes" at Spotify?
- What is fault-injection testing?
- How can "design for failure" be actually implemented?
- What is Git/GitHub?
- What is Flask?
- What is the Model-View-Controller pattern?
- What is Celery?
- What is / How can you use OpenApi 2.0?

// software testing

- What is development/release/user testing?
- What is partition testing?
- What is test-driven development?
- What is load testing?

// user stories

- What is a user story (for)?
- What is the priority of a user story?
- What is the effort/size of a user story?
- What is an epic?

// splitting the monolith

- When and where to start splitting a monolith codebase?
- How to split databases? (e.g., how to break foreign key relationships?)
- What about transactions when you split?
- What is the SAGA pattern?
- What is eventual consistency?
- What is a (event) data pump?

// cloud-based software engineering

- What are the service/deployment models of cloud computing?
- Which is an example of (disruptive) business model exploiting cloud computing?
- What is a virtual machine? / What is server virtualization?
- What are Heroku's dynos (for)?
- What is FaaS?
- How to avoid/reduce cloud lock-in?
- What is a container/image/volume?
- What is the difference between a virtual machine and a container?
- What is image layering in Docker?
- What is the effect of *docker run/commit*?

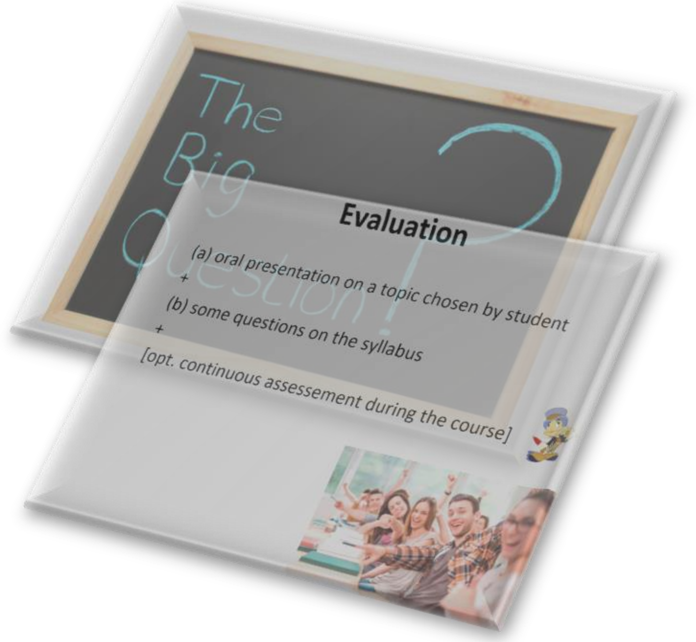
// business process modelling

- What is a parallel/exclusive/inclusive gateway in BPMN?
- What is an error event in BPMN?
- What is Camunda?
- Can you describe the usage patterns of Camunda?
- What is the difference between orchestration and choreography?
- What is a workflow net?
- How can we model BPMN parallel/exclusive/inclusive gateways with workflow nets?
- What is a live/bounded/sound net?

// fog computing

- What is fog computing?
- What is / how difficult is the "component deployment problem" in fog computing?
- What is FogTorchΠ?
- What is SecFog?

How to get the credits



Evaluation

(a) oral presentation on a topic chosen by student
+
(b) some questions on the syllabus
+
[opt. continuous assessement during the course]

Students that have (successfully) participated in the optional continuous assessment activities:

- Get a **bonus** in the final evaluation (determined by averaging the best 4 grades of each student)
- Are exempted from "(a) oral **presentation**"
- Can take the exam **also on the January 10th** date

Exam dates

- *January 10th, 2019 (restricted)*
- January 22th, 2019
- February 13th, 2019
- …

Recall that to participate in the exams
you **MUST** register on esami.unipi.it by the set deadlines

Classes and labs
Syllabus
Everything in the Moodle!
Course self-assessment
The big question
Thanks

Thanks, Stefano!

"First useful lab in my studies" [anonymous]

Thanks also to

Davide

Luca

Alessandro

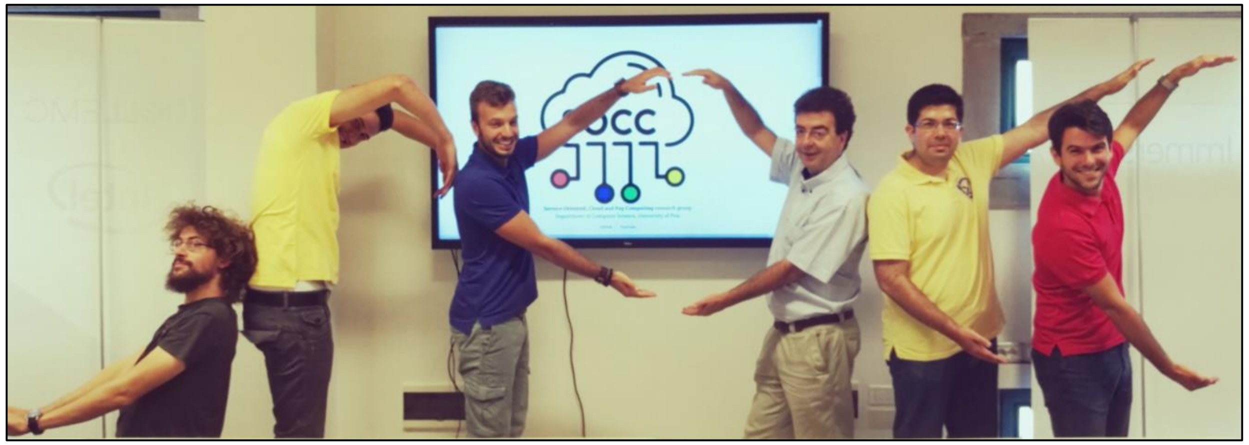
And … thank you!

The End

Classes and labs
Syllabus
Everything in the Moodle!
Course self-assessment
The big question
Thanks

PS: Topics for MS theses

Service-oriented, Cloud and Fog Computing Group



Recent projects

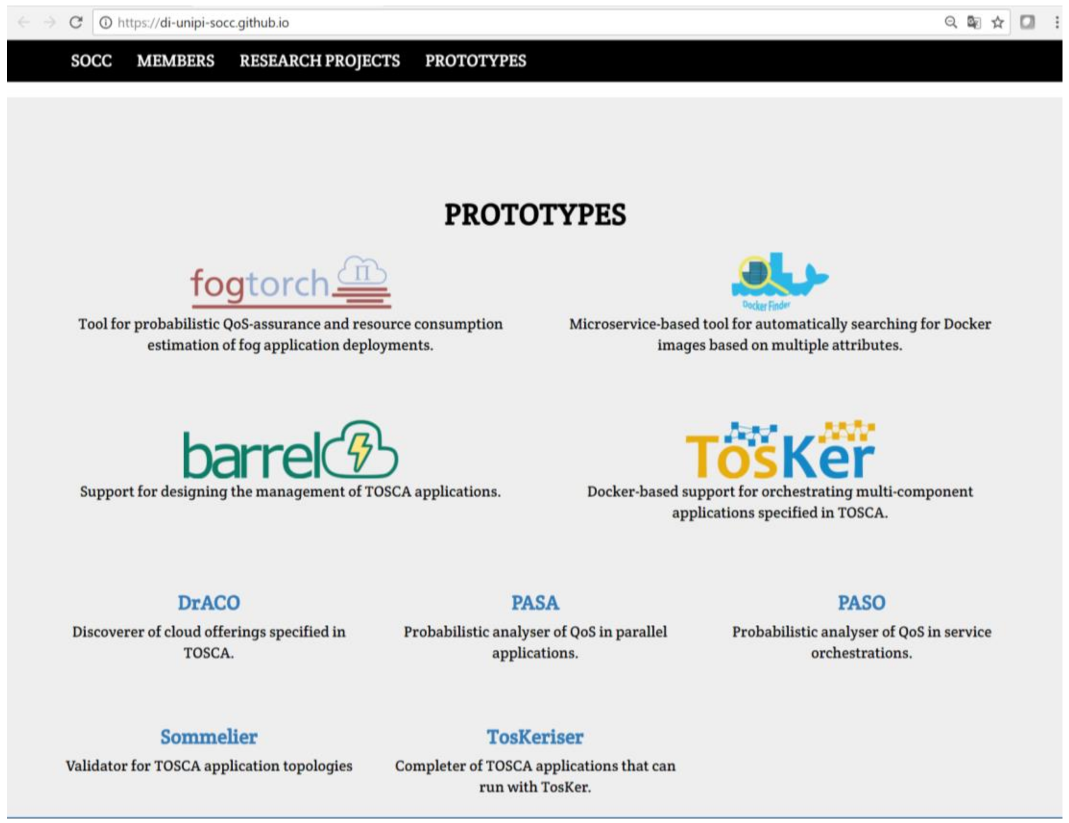
SeaClouds — EU cloud-centered project aiming at devising a novel standards-based platform supporting **agile deployment and adaptive management across multiple heterogeneous** (IaaS and PaaS) **clouds**. *[oct13,mar16]*

Through the fog — UNIPI project aiming at devising new models and techniques to deploy existing applications over the fog, to program new fog applications and to support **fog computing**. *[nov15,oct16]*

Services, cloud and fog computing — UNIPI/DI project to devise techniques to **model**, **analyse** and **adapt service-based** and **cloud** applications, to support the design and management of **microservices**-based applications, to predict the **QoS** of service-based, cloud and fog applications, to model, analyse and manage **fog** applications. *[nov16,oct18]*

AMaCA — POR-FSE project aimed at proposing novel solutions for **automating the management of microservice-based applications**, providing both design-time and run-time support in a DevOps style. *[jan18,dec19]*

DECLware — UNIPI project aiming at devising new declarative methodologies to design and deploy distributed applications taking into account QoS and security constraints. *[jul18,jun20]*


https://di-unipi-socc.github.io

SOCC MEMBERS RESEARCH PROJECTS PROTOTYPES

PROTOTYPES

fogtorch — Tool for probabilistic QoS-assurance and resource consumption estimation of fog application deployments.

Docker Finder — Microservice-based tool for automatically searching for Docker images based on multiple attributes.

barrel — Support for designing the management of TOSCA applications.

TosKer — Docker-based support for orchestrating multi-component applications specified in TOSCA.

DrACO — Discoverer of cloud offerings specified in TOSCA.

PASA — Probabilistic analyser of QoS in parallel applications.

PASO — Probabilistic analyser of QoS in service orchestrations.

Sommelier — Validator for TOSCA application topologies

TosKeriser — Completer of TOSCA applications that can run with TosKer.

https://di-unipi-socc.github.io

DevOps software engineering

- portability & automated management of cloud apps
- microservices architectures
- container orchestration

Main current topics

Fog computing

- predictive deployment of fog apps
- management of fog apps
- security in the Fog
- drones in the Fog?