CHALMERS
UNIVERSITY OF TECHNOLOGY

Distributed Computing and Systems
Chalmers university of technology

Distributed Systems I

Lab introduction

How to recognize us in the labs:

Yiannis Nikolakopoulos

ioaniko

Iosif Salem

iosif

Jakob Kallin

jakob.kallin

@chalmers.se

Agenda

- **Seattle platform**
- Lab overview
- Lab 1

Seattle

- A platform for getting access to computers around the world.

- We will use Seattle to study a distributed system in practice!

Seattle intro

- Seattle
  - Write code in a subset of Python (Repy = Restricted python) and run on machines all over the world
  - These machines are provided by the community
  - https://seattle.poly.edu/html/

- A perfect platform for teaching distributed systems
  - Experience distributed nature: machines all over the globe
    - Machines may disconnect, break, limited bandwidth, …
  - Use a modern language: Python
    - Fast prototyping

Seattle Resources

- 5 min video
  - https://seattle.poly.edu/wiki/UnderstandingSeattle/DemoVideo
- Installation, tutorials, etc.
  - https://seattle.poly.edu/wiki/ProgrammersPage
- Repy tutorial
  - https://seattle.poly.edu/wiki/RepyTutorial
- Repy API
  - https://seattle.poly.edu/wiki/RepyApi

Seattle Demo

- Register on Seattle Clearinghouse
- Install demokit.zip from your profile page
- Generate your private/public key pair
- Download your keys and put them to the demokit directory

Seattle demo: Repy examples

- Seash Shell command
  - https://seattle.poly.edu/wiki/SeattleShell
  - https://seattle.poly.edu/wiki/RepyTutorial

- Hello World example:
  - example.1.1.repy
- Hello World to web users:
  - example.1.2.repy
  - example.1.3.repy: no global in Repy

Agenda

- Seattle platform
- **Lab overview**
- Lab 1 introduction

Lab logistics

- 4 Labs, 10 points each
- PASS = at least 31/40 points

- Late submissions:
  - within 1 week after the deadline
    → -1 point from your score on that lab
  - within 2 weeks after the deadline
    → -3 points from your score on that lab
  - No submissions accepted 2 weeks after the deadline

Lab deadlines

- **Preassignment**: November 9, 23:59
- **Lab 1**: November 16 , 23:59
  - Demo: Nov. 13
- **Lab 2**: November 27 , 23:59
  - Demo: Nov. 24 & 27
- **Lab 3**: December 11 , 23:59
  - Demo: Dec. 8 & 11
- **Lab 4**: January 9 , 23:59
  - Demo: Dec. 18

Use the Lab slots to demo your solutions to us before submission!

Deadlines are hard, but you can submit afterwards with a penalty.

Hand in

- Code
  - Well structured
  - Well documented

- Results
  - In a video
    - 1-2 minutes screencast to demonstrate your results
    - Some screencast software: http://en.wikipedia.org/wiki/Screencast
    - Screencast software in Lab rooms: RecordMyDesktop
  - **Or** in a report (as a pdf file)

If you choose report instead of video

- Include the same information as in the video

- That is
  - Document that your solution works with screenshots and explanations
  - All stages of the execution should be included in your documentation

Grading

- Solution: 4 points
- Code structure: 2 points
- Code documentation: 2 points
- Video or report: 2 points

Code Structure and Documentation

- “Code is written *primarily* to be read by humans. It has to be acceptable to the compiler too, but the compiler doesn’t care about how it looks or how well it is written.”

- In your professional life, you will mostly use, fix and improve code that already exists...

Code Structure and Documentation: Guidelines

- Descriptive variable names
  (no ‘a’, ‘b’, ‘apa’ are not descriptive…)

- Comment on any blocks of code that are doing something subtle or that is not immediately obvious.

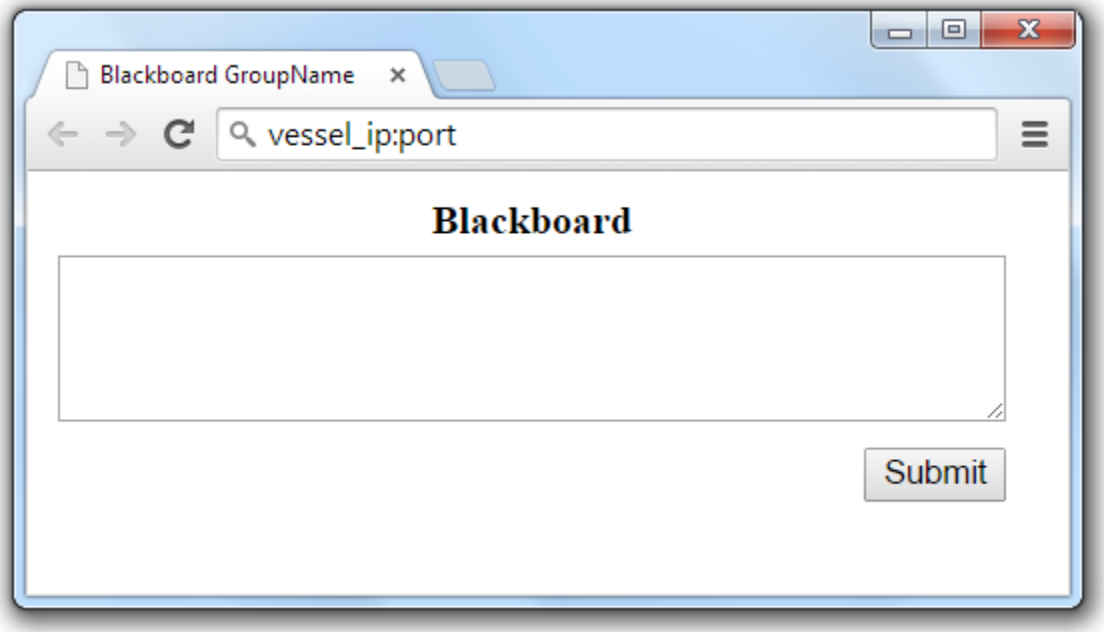
Code Structure and Documentation: Guidelines

- Document **what** each function does (not how), arguments, returned values, side effects etc (see Repy API).

- If you can't do this in a few sentences, that suggests that you may need to rethink your abstraction.

Lab scheme

A (basic) blackboard should look like this:



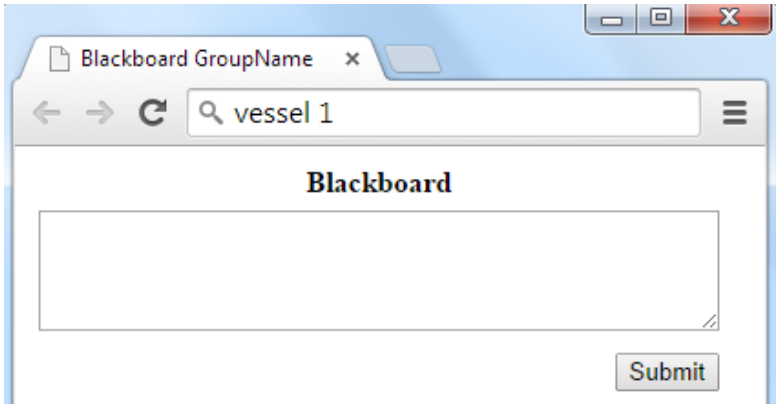
Lab scheme

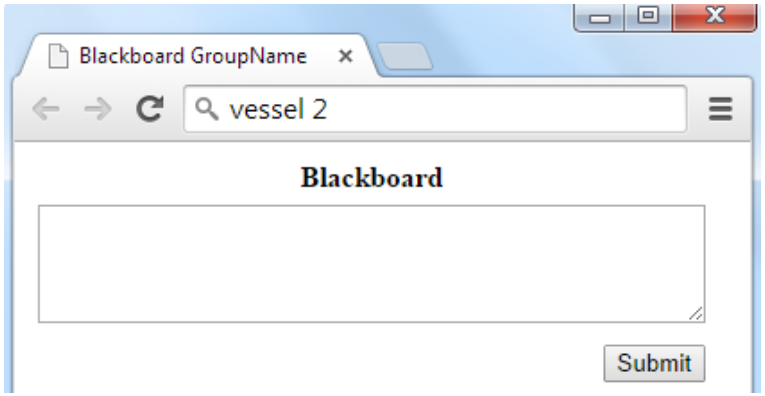
This lab: **distributed blackboard!**

- A number of machines around the world, each one having each own blackboard
- When a message is written in one board, it should be also propagated to all other boards

Lab scheme

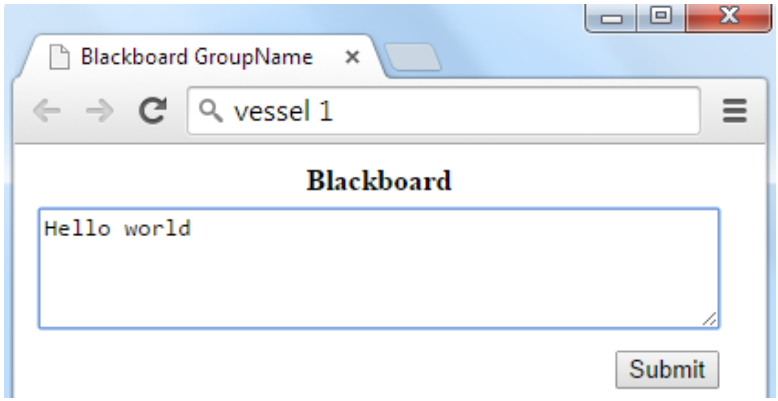
Initially: empty blackboard on all vessels



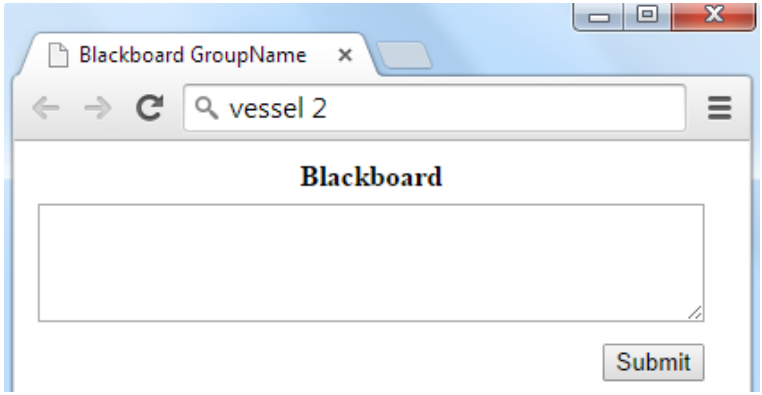


Lab scheme

On vessel 1:
Write text and submit

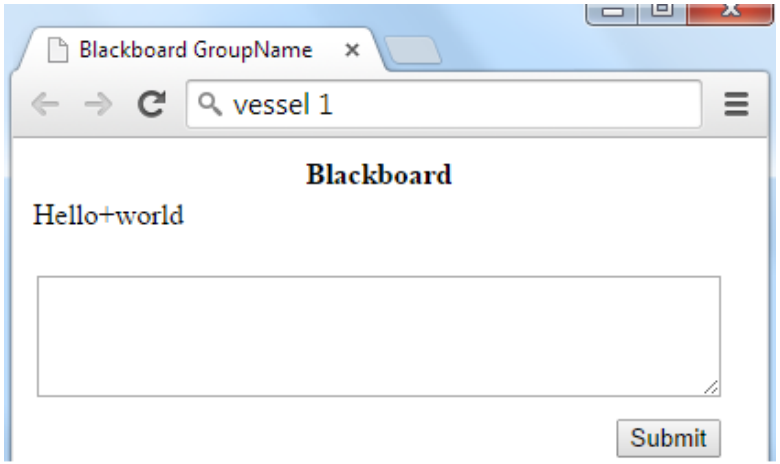


On vessel 2:
No action on the browser

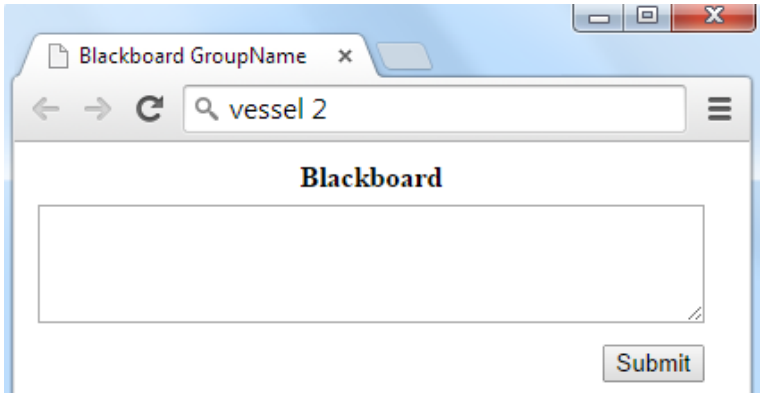


Lab scheme

On vessel 1:
Write text and submit



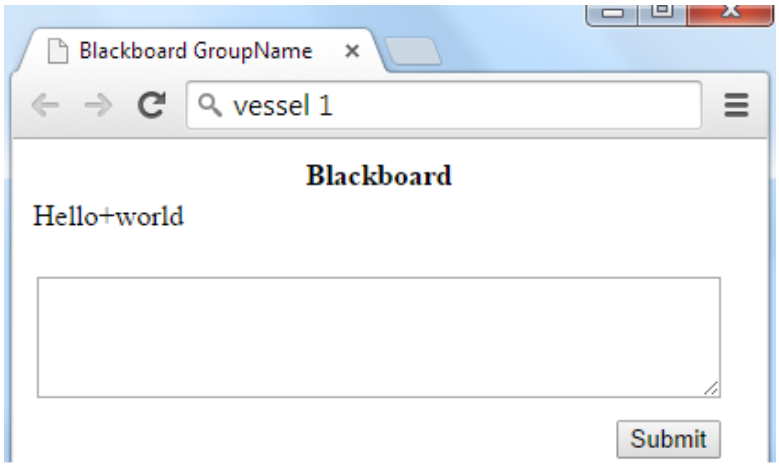
On vessel 2:
No action on the browser

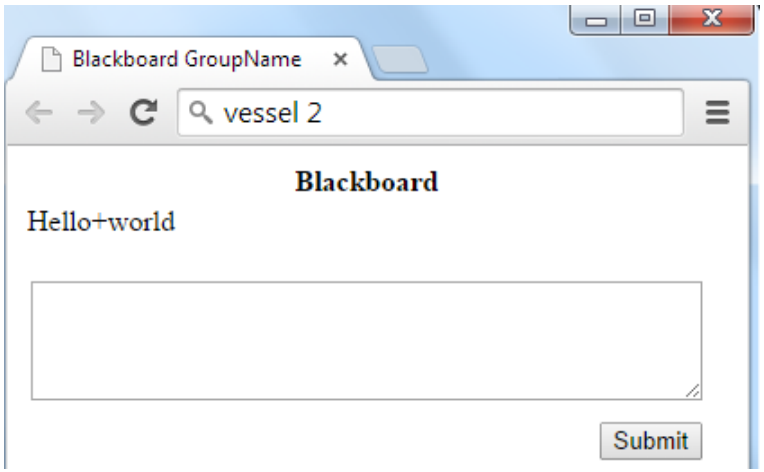


Lab scheme

On vessel 1:
No action on the browser

On vessel 2:
Hit refresh and see post!





Lab scheme

Over the course we will make this blackboard more

- Reliable
- Consistent
- Efficient
- …

Agenda

- Seattle platform
- Lab overview
- **Lab 1 introduction**

Solution submitted last year

(see video)

Make it work

- Keep a list of all vessels in each vessel
  - (We know, this is not a scalable design, you will work on this aspect in the following labs)
- Upon a post
  - Send the update to all other vessels
  - To be shown (along with the previous board content) when a user visits again the page from the browser
- Note
  - You can use TCP connections between the vessels to send the updates
- Roughly 80 lines of code (depends on HTML)

Pitfalls & Hints

- Repy & Python
  - No global variables across functions!
    - There is a solution, cf Repy API.
  - Python has excellent built-in string manipulation functions.
  - Python is Dynamically & Strongly typed.
    Objects have types, not the variables.

    x = 10

    x = “Hello”

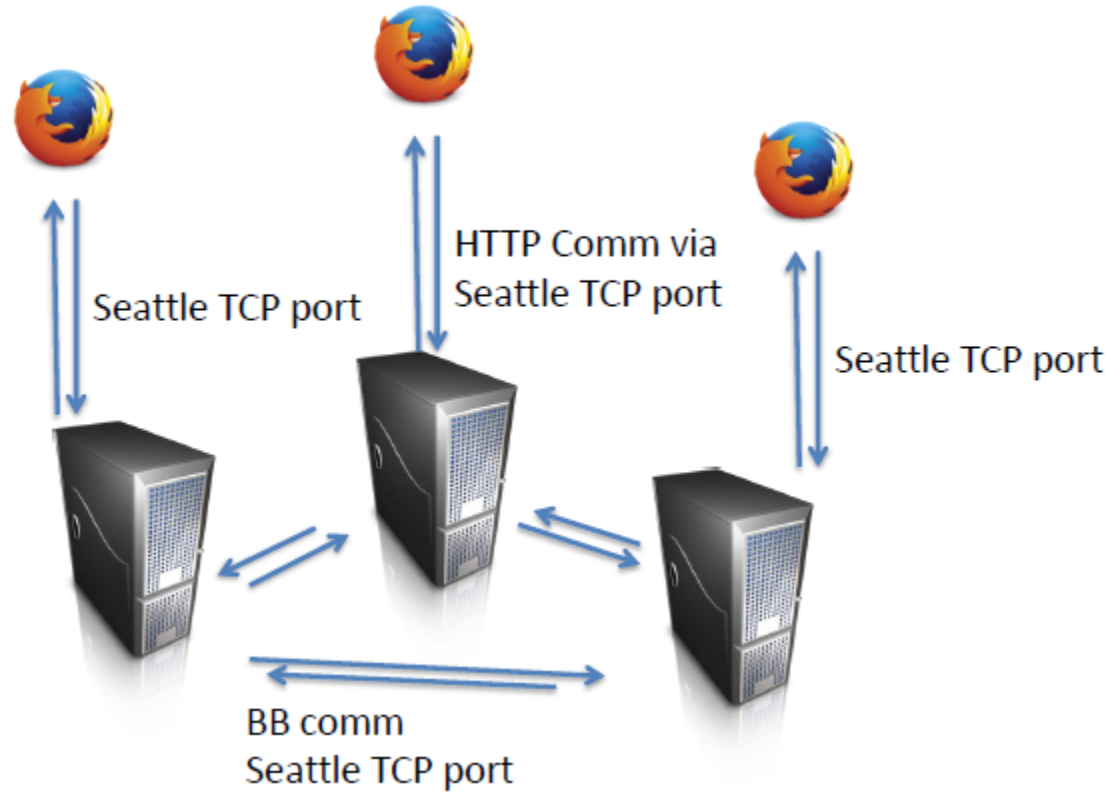
    The above sequence is valid.

Pitfalls & Hints

- HTTP
  - A browser awaits a response after a ‘POST’ request…
  - Use print on received GET and POST headers to understand the message structure (not in your submitted version)
  - We don’t care if the board content is encoded: “Hello+world” is considered a valid entry

Pitfalls & Hints

- “User interface” vs Communication



What you will get from us

- A sample HTML file
  - Change it if you want (totally optional)
- A skeleton Repy file
  - Based on Hello world from Repy tutorial
- Answers to your questions in the labs
- Demo slots

Task 1

- Demonstrate that your distributed blackboard works by submitting a 1-2 minutes video or a report
  - Document: Do 2 or 3 posts and show them appearing on the other blackboards
  - Use at least 8 vessels/blackboards
  - Hint: show the browser windows and (optionally consoles) next to each other on your screen
  - Record your screen and document what is happening by using your mouse and your voice
    - No video editing, cutting, etc. required

Task 2

- Can it happen that two vessels show different blackboards?
  - Even when all data was reliably send to all vessels, and then we hit refresh afterwards
  - Hint: what happens when two posts are submitted at (more or less) the same point in time?
  - Submit max 1-2 minutes video or a report
    - Explaining your thoughts
    - If you can make it happen: document it in your video or report

Hand in

- Code
  - Well structured
  - Well documented
- Task 1
  - Video or report
- Task 2
  - Video or report

Questions?