A Racket-Based Robot to Teach First-Year Computer Science

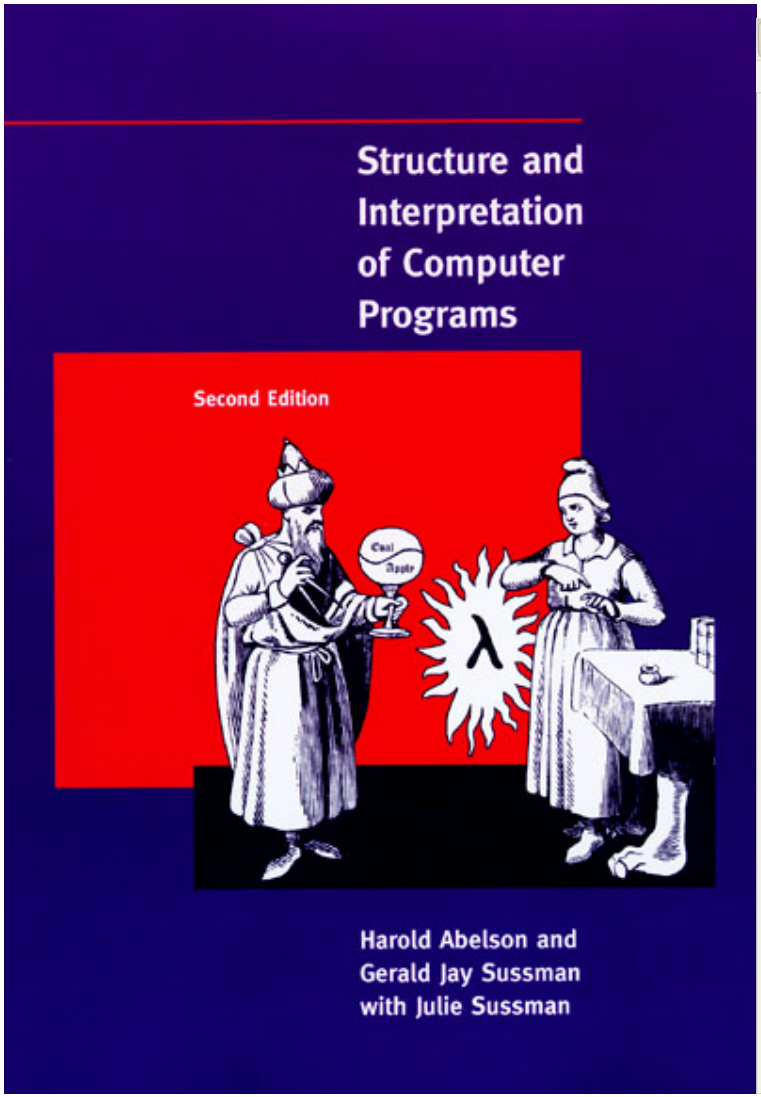
K.Androutsopoulos, N. Gorogiannis, M. Loomes, M. Margolis, G. Primiero, F. Raimondi, P. Varsani, N. Weldin, A.Zivanovic

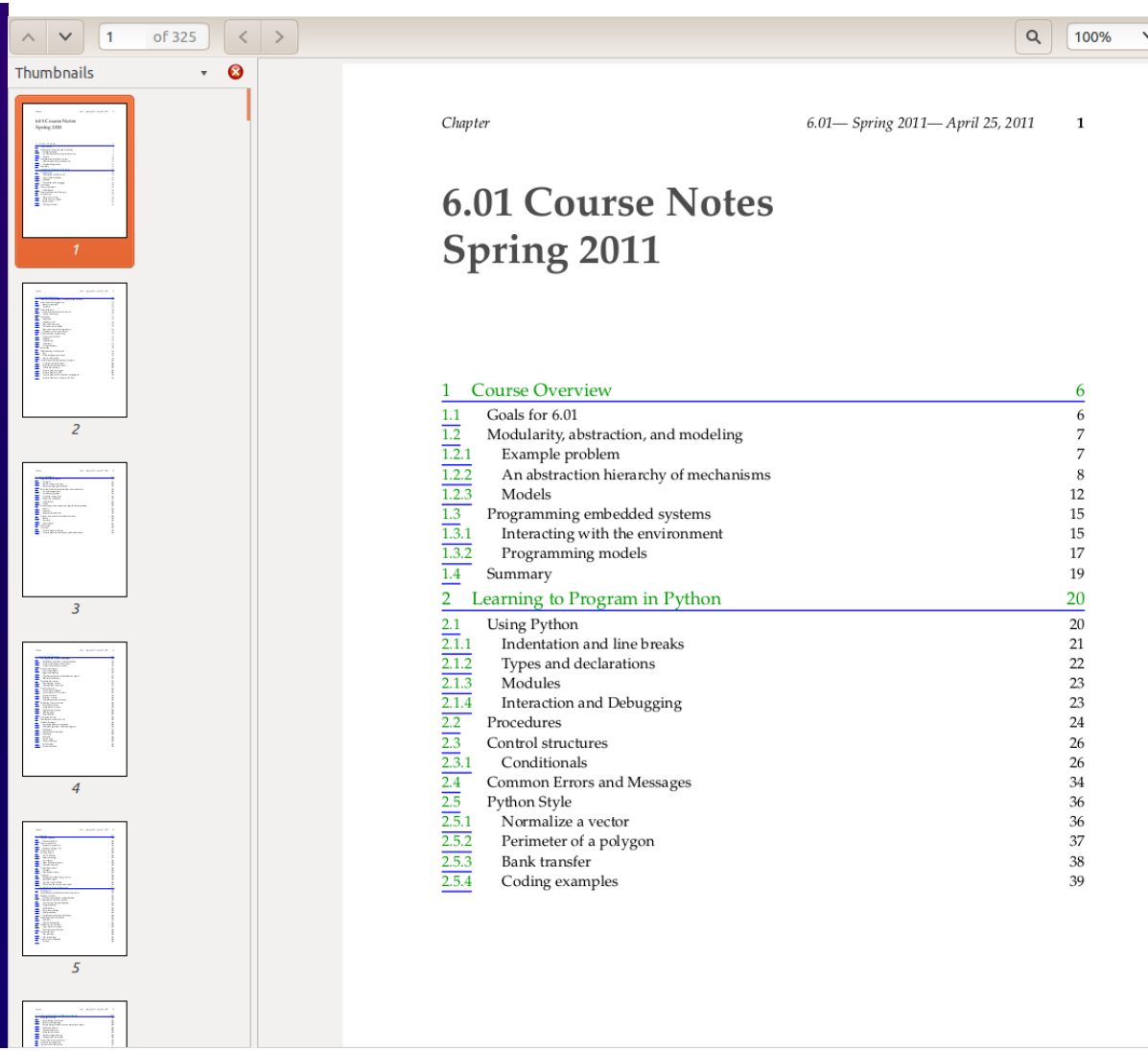
Department of Computer Science
School of Science and Technology
Middlesex University, London
http://www.cs.mdx.ac.uk

European Lisp Symposium 2014

1. The really (in)famous precedent
2. The context
3. Racket & Mirto
4. Applications
5. Assessment & Evaluation

MIT CS 6.001


Structure and
Interpretation
of Computer
Programs
Second Edition
Harold Abelson and
Gerald Jay Sussman
with Julie Sussman


1 of 325
100%
Thumbnails
1
2
3
4
5
Chapter
6.01— Spring 2011— April 25, 2011
1
6.01 Course Notes
Spring 2011
1 Course Overview 6
1.1 Goals for 6.01 6
1.2 Modularity, abstraction, and modeling 7
1.2.1 Example problem 7
1.2.2 An abstraction hierarchy of mechanisms 8
1.2.3 Models 12
1.3 Programming embedded systems 15
1.3.1 Interacting with the environment 15
1.3.2 Programming models 17
1.4 Summary 19
2 Learning to Program in Python 20
2.1 Using Python 20
2.1.1 Indentation and line breaks 21
2.1.2 Types and declarations 22
2.1.3 Modules 23
2.1.4 Interaction and Debugging 23
2.2 Procedures 24
2.3 Control structures 26
2.3.1 Conditionals 26
2.4 Common Errors and Messages 34
2.5 Python Style 36
2.5.1 Normalize a vector 36
2.5.2 Perimeter of a polygon 37
2.5.3 Bank transfer 38
2.5.4 Coding examples 39

MDX CSD 1000

Computer Science at Middlesex University

- New Computer Science programme for the academic year 2013/2014
- Teach students how to become autonomous learners
- *Racket*: solid mathematical background and language-independent programming skills
- *Real hardware*: Arduino, Raspberry Pi, and the Robotic Platoform *Mirto*
- Completely revised delivery and assessment methods:
  - no modules or courses
  - activities run seamlessly across the projects
  - Assessment through *Student Observable Behaviours* (SOBs).

Week structure

- **General Lecture:** introduction to topic and related project;
- **Design Workshop:** design skills in software or hardware, systems engineering (UML), HCI, security;
- **Programming Workshop:** exercises, master-classes, coaching sessions, restricted to Racket;
- **Physical Computing Workshop:** from simple logic gates to microcontrollers (Arduino) and other specialist devices controlled through Racket;
- **Synoptic Workshop:** 4 hours to investigate foundations, design, build, test and discuss projects.

Three Projects

1. traffic light system
2. dungeon game
3. Middlesex Robotic Platform

The Platform

- Base platform:
  - two HUB-ee wheels with motors and encoders (to measure actual rotation) built in
  - front and rear castors
  - two bump sensors
  - an array of six infra-red sensors
  - a rechargeable battery pack
  - an Arduino microcontroller board
- Top layer:
  - a Raspberry Pi connected to the Arduino
  - Linux with Racket (current version 5.93)
  - USB-WiFi adapter for SSH and network
  - Additional: cameras, microphones and text to speech with speakers

MIRTOlib

- Library developed by the teaching team
- Takes care of low-level serial communications

```
(send-sysex-int-msg #x7D 5 power)
```

- Students deal only with high-level Racket programs

```
(define (setMotors speed1 speed2)
  (setMotor 0 speed1)
  (setMotor 1 speed2))
```

- Students can read IR values with

```
(getIR 2)
```

Line-following with PID

```
(define proportional (- error 2000))

;; Integral component: we reset to 0 when error is 0
(cond ( (= 0 proportional) (set! intError 0))
        (else (set! intError (+ intError proportional)))
)
;; we assume dt constant, so this is just the difference
;; If derivative < 0, we moved to the left of the line
(define derivative (- proportional (- prevError 2000)))
(set! prevError error)
;; The correction is the sum of a proportional component,
;; integral component and a derivative component.
(define correction (+ (* Kp proportional)
                      (* Ki intError)
                      (* Kd derivative)) )
(cond

    ((> correction 0) ;;we are to the right
     (setMotors PWR (- PWR correction)))

    (else ;; we are to the left
    (setMotors (+ PWR correction) PWR))
)
```

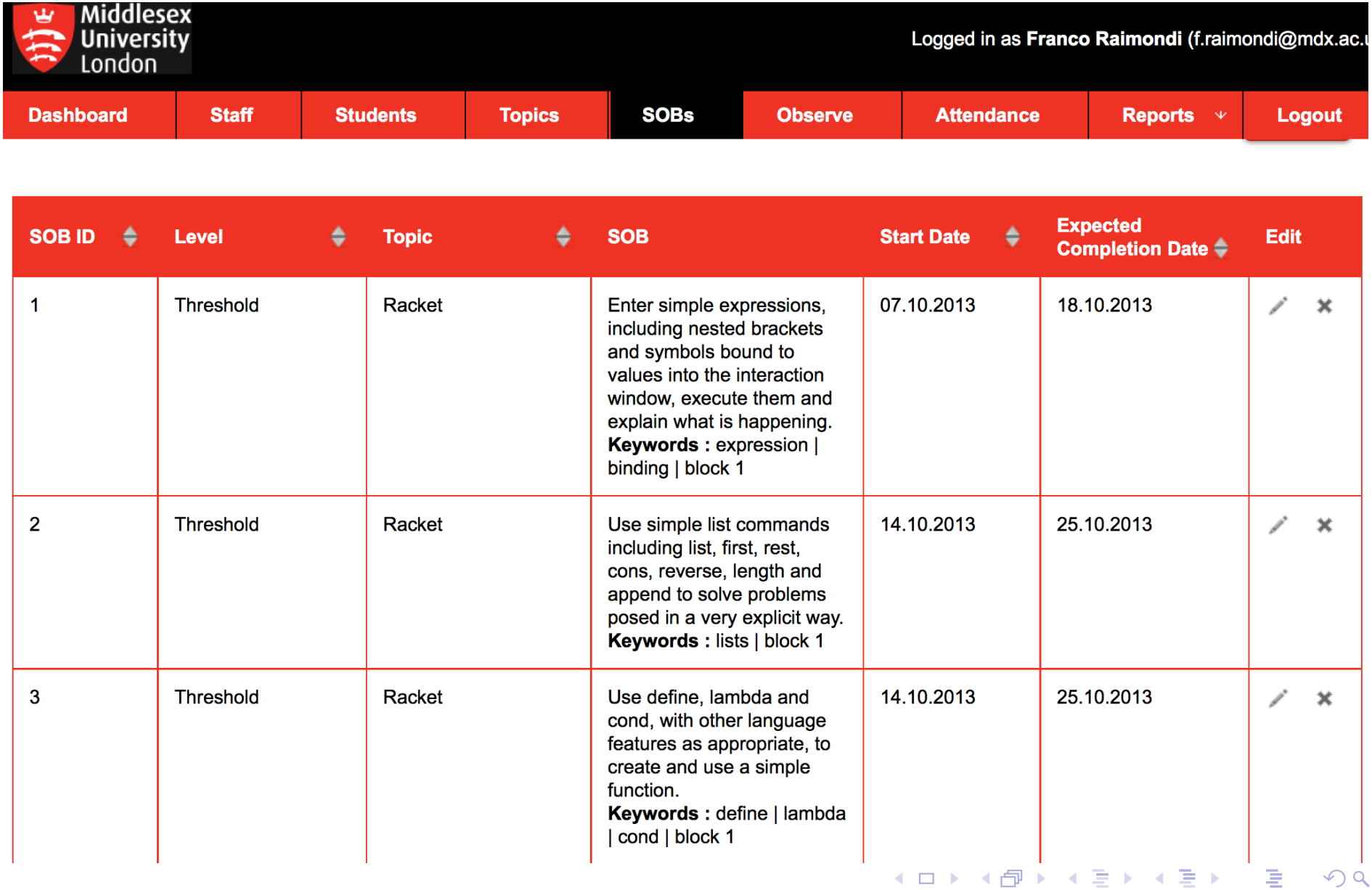
Others

- Speech-recognition: PocketSphinx connected to Racket
- Graphical Interface using X on Pi
- Web-server running on Pi
- Twitter controlled Robot

SOBs

1. **Threshold level**: essential to pass the year.
2. **Typical level**: expected for a good honours degree.
3. **Excellent level**: identifies outstanding achievements.

SOBs Tool


Middlesex University London
Logged in as Franco Raimondi (f.raimondi@mdx.ac.u
Dashboard
Staff
Students
Topics
SOBs
Observe
Attendance
Reports
Logout
SOB ID
Level
Topic
SOB
Start Date
Expected Completion Date
Edit
1
Threshold
Racket
Enter simple expressions, including nested brackets and symbols bound to values into the interaction window, execute them and explain what is happening. Keywords : expression | binding | block 1
07.10.2013
18.10.2013
2
Threshold
Racket
Use simple list commands including list, first, rest, cons, reverse, length and append to solve problems posed in a very explicit way. Keywords : lists | block 1
14.10.2013
25.10.2013
3
Threshold
Racket
Use define, lambda and cond, with other language features as appropriate, to create and use a simple function. Keywords : define | lambda | cond | block 1
14.10.2013
25.10.2013

SOBs Tool

List of Students

S.No	Student Number	First Name	Last Name	Email	Threshold
1	M00[redacted]	[redacted]	[redacted]	[redacted]@live.mdx.ac.uk	0 ✓
2	M00[redacted]	[redacted]	[redacted]	[redacted]@live.mdx.ac.uk	0 ✓
3	M00[redacted]	[redacted]	[redacted]	[redacted]@live.mdx.ac.uk	5 !
4	M00[redacted]	[redacted]	[redacted]	[redacted]@live.mdx.ac.uk	0 ✓
5	M0[redacted]	[redacted]	[redacted]	[redacted]@live.mdx.ac.uk	0 ✓
6	M00[redacted]	[redacted]	[redacted]	[redacted]@live.mdx.ac.uk	0 ✓
7	M00[redacted]	[redacted]	[redacted]	[redacted]@live.mdx.ac.uk	0 ✓

Figure : Student list with SOBs

SOBs Tool

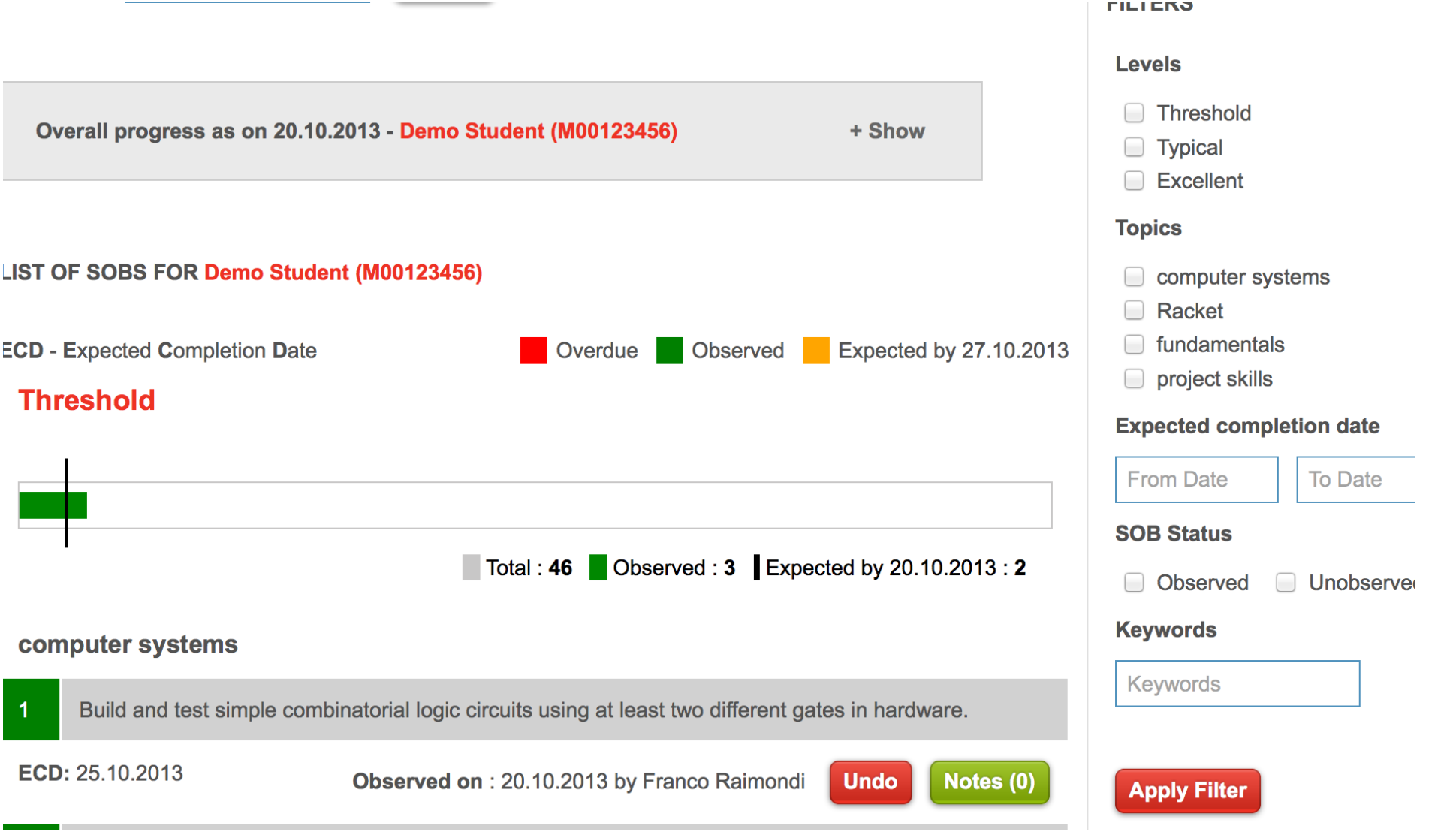


Figure : Observing a SOB for a student

SOBs Tool

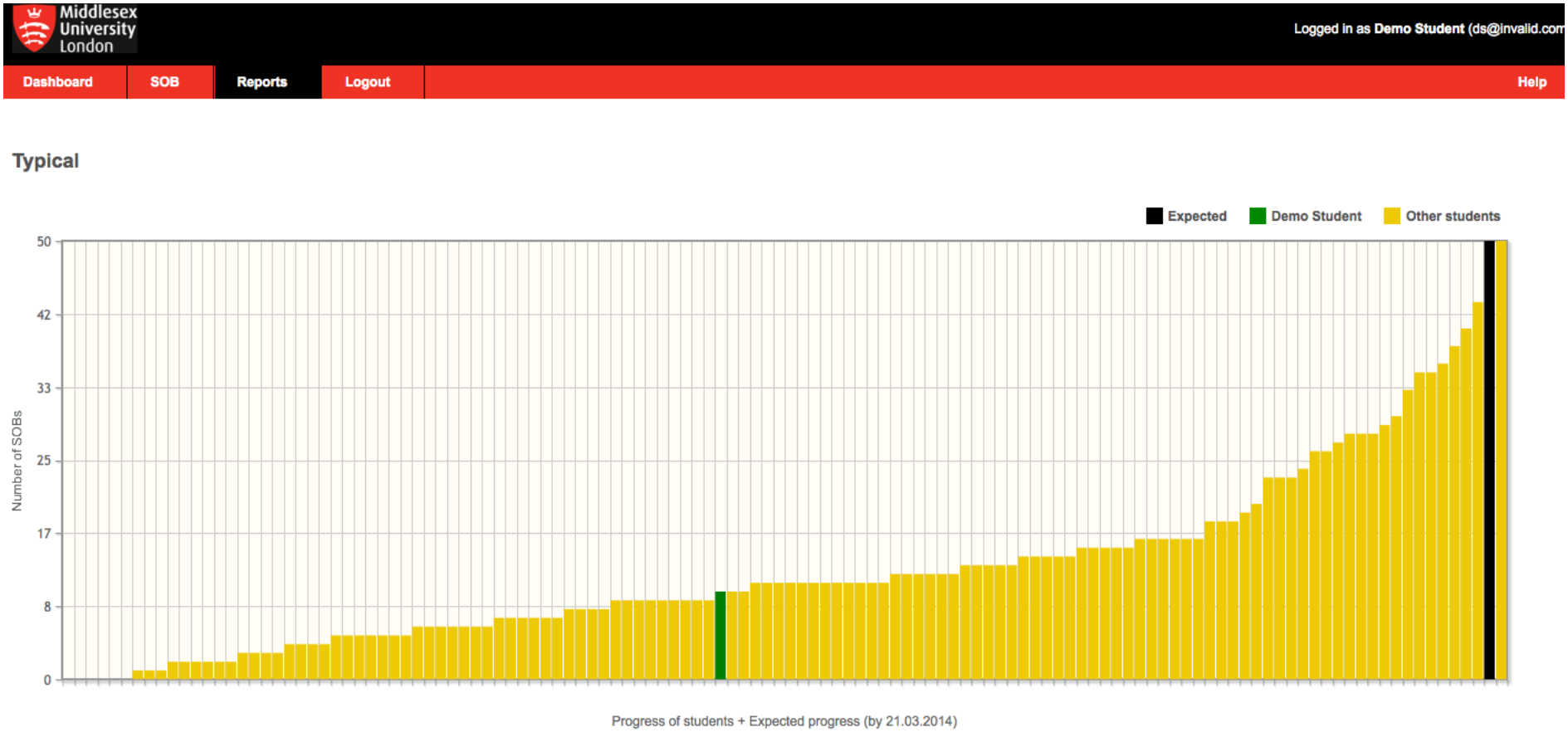


Figure : Student view: position with respect to class

Evaluation & Conclusion

- 85% success rate
- Average 90% attendance
- All students have progressed beyond threshold SOBs
- `https://github.com/fraimondi/myrtle/`
  (software and design files)