DATA MATERIALISATION: A NEW UNDERGRADUATE COURSE FOR A DATA DRIVEN SOCIETY

Beghelli, Alejandra; Huerta-Cánepa, Gonzalo; Segal, Ricardo

Universidad Adolfo Ibañez

ABSTRACT

Traditionally, data has been presented in textual format and the interaction with the user confined to the keyboard or touch screen to input data and the screen to deliver information. However, with the advent of a data-driven society, an opportunity for more natural and efficient ways of presenting data and interacting with it has emerged.
Although the area of Human-Computer Interaction has existed for a long time, its focus has always been on the interaction with an artefact (the computer). Today instead, we face the challenge of interacting with an intangible object: data. As a result, a key requirement emerges: How do we make legible the enormous amounts of data produced per day to ordinary people?
Designers, able to devise natural and smooth interaction experiences, should play a relevant role in this new scenario. However, they might lack the basic technical knowledge required to understand the possibilities of these new systems.
In this paper we present a brief how-to manual for an undergraduate course on data materialisation: the process of transforming an intangible object (data) in an artefact that can be interacted with in a physical way.

Keywords: Data Materialisation, Design education, Information management, Multisensory product experience, Technology

Contact:
Beghelli, Alejandra
Universidad Adolfo Ibañez
Faculty of Engineering and Sciences
Chile
alejandra.beghelli@uai.cl

Cite this article: Beghelli, A., Huerta-Cánepa, G., Segal, R. (2019) 'Data Materialisation: A New Undergraduate Course for a Data Driven Society', in *Proceedings of the 22nd International Conference on Engineering Design (ICED19),* Delft, The Netherlands, 5-8 August 2019. DOI:10.1017/dsi.2019.212

1 INTRODUCTION

With the advent of the Internet of Things (IoT) and the ubiquity of smartphones (that have expanded the role of users from data consumers to prosumers: producers and consumers of information), we live in a society that provides us with massive amounts of data. This situation has given birth to a new area: Human-Data Interaction (HDI) (Mortimer 2015). As a branch of the established Human-Computer Interaction area, this new discipline is concerned with providing mechanisms for people to interact explicitly with data and the systems that collect it and process it. In this paper, we focus on one aspect of HDI: How do ordinary people interact with data? Can we leverage on the potential advantages of data materialisation to facilitate the understanding of and interaction with the massive amount of information produced every day?

We should note that the term data materialisation is still very recent and thus, different people use different names for the same process and a final definition is yet to be reached. For example, according to (Starrett *et al.*, 2018), data materialisation differs from data visualisation in that the former "prioritizes the clear and accurate communication of data while data materialization prioritizes the design." That is, the appearance of the artefact should be more important than the accuracy with which data is communicated, and nothing is specified regarding data interaction. In (Huron *et al.*, 2017, Jansen *et al.*, 2015) the term data physicalization is used, defined as "the representation of data through the geometrical or physical properties of an artefact." In this paper, we understand data materialisation as "the process of transforming an intangible object (data) in an artefact that can be interacted with the aim of facilitating the understanding of data." Note that for us, data interaction is critical.

Data materialisation can be carried out by non-digital or digital means. The former is more accessible to implement by a wider audience - as reported in (Huron *et al.*, 2017) - but it is not suitable for live data, where information changes in short timescales. The latter has much higher entry barriers – knowledge of networks, computation and electronics is necessary, as reported in (Bordegoni 2017) – but it is suitable to adapt to rapid data changes and provide multimodal embodied interaction, as the project presented in (Hogan *et al.*, 2013). In this paper, we focus on digital data materialisation.
With the variety of low-cost sensors and actuators available today and the democratization of electronic circuit building triggered by Arduino, designers can contribute significantly to give meaning to data by exploring a wide range of ways of digital data materialisation without the need of becoming experts in the technical aspects of it. However, minimum technical knowledge is necessary to understand and manage the key technologies and the possibilities that they offer. Such knowledge is usually dispersed in different courses: Computer Networks, Electronics, and Computer Programming, that usually focus in much more detail than the minimum necessary to build a data materialisation project.

With the objective of facilitating the entry of designers to the area of digital data materialisation, in this paper, we propose a 16-week undergraduate course aimed at students with basic knowledge of programming and digital manufacturing. This proposal was built after leading two groups of students in a data materialisation project during the spring term of 2018. One group was made of 40 1st year students of engineering, who worked during the whole semester in the project. A second group was made of 16 students in the 3rd year of engineering, who worked on the project for 4 weeks, due to their previous knowledge of Arduino and Python. The final goal of the proposed course is for the students to produce a physical device that allows natural interaction with live data. To provide the technical background needed for the project, the following modules are proposed: Arduino-based electronics, Python programming, and Networks and APIs, Hands-on module. The main contribution of this paper is presenting a course structure that merges all technical contents in a single course.

2 PREVIOUS WORK

An exhaustive search for academic papers reporting on the design of an undergraduate course focused on digital data materialisation did not yield any relevant results.

https://doi.org/10.1017/dsi.2019.212 Published online by Cambridge University Press

In the area of Design, several programmes offer technological training to students of design-related careers. However, as shown in Table 4 in (Blanco *et al.*, 2017), these courses do not cover the essential aspects of electronics, programming, and networks in a single course. Usually, they either focus on electronics (as the Basics Electronics for Product Design from Leeds University, UK or Basic Electronics and Engineering for Designers from the Royal College of Arts, UK), programming/informatics (as the course Informatics from University of Zaragoza, Spain) or networks and communications (as the course Digital and Connected World for the Master in Design and Communications from the Politecnico di Torino, Italy) separately.

In the Engineering field, several courses integrate the use of prototyping boards. We present here the three of them closest to our work. (Jamieson and Herdtner 2015) discuss the benefits of including microcontrollers or single-board computers (SBC) in engineering education and present some examples related to using prototyping boards in undergraduate classes. Nevertheless, their focus is on the engineering part, leaving aside both design and data considerations. (Ariza 2018) presents a curriculum that includes an introduction to Python programming, Arduino, Sensors, Raspberry and communication. However, the user interaction is based on computer applications, not on physical interfaces. (Zhong and Liang 2016) present an outline of a course on Internet of Things concepts using the Raspberry Pi platform. They present examples of some projects that students can achieve, but, as in previous cases, the focus is on the electronics and not in the physicalization of data.

On the specific topic of data materialisation, (Huron *et al.*, 2017) report a workshop on data materialisation, but the representation is not digital, and the workshop lasts for only one day. In (Bordegoni and Carulli 2017) a 5-day postgraduate workshop on digital data materialisation is reported, but no proposal for an undergraduate course is given.

To the best of the author's knowledge, this is the first paper to propose an undergraduate course that allows students from design-related programmes to learn the fundamental technologies required to develop a functional prototype for a data materialisation project.

3 LEARNING OUTCOMES

The learning outcomes (LO) of the course are the following:

- LO1: Operate electronic sensors and actuators
- LO2: Transfer real-time data between Internet and a local digital platform
- LO3: Build a digital prototype that materialises the data generated by a real-time phenomenon

4 THE TEACHING MODULES

The teaching modules were organised around the learning outcomes and the specific architecture for the data materialisation system, shown in Figure 1. Looking at Figure 1 from right to left, the data to be materialised is collected from a web server API by the Python Client Application and sent to the Arduino Platform by means of serial communication. Then, the physical artefact interacts with the user by means of sensors and actuators controlled by the Arduino Platform.

This architecture defines the teaching modules the students must take before having the necessary technical skills required to build a data materialisation project. Table 1 shows each teaching module as well as the learning outcome associated to it.

Table 1. Teaching modules and corresponding learning outcomes

Teaching module	Learning Outcome
Arduino module	LO1, LO2
Python module	LO2
Networks & APIs module	LO2
Hands-on module	LO3

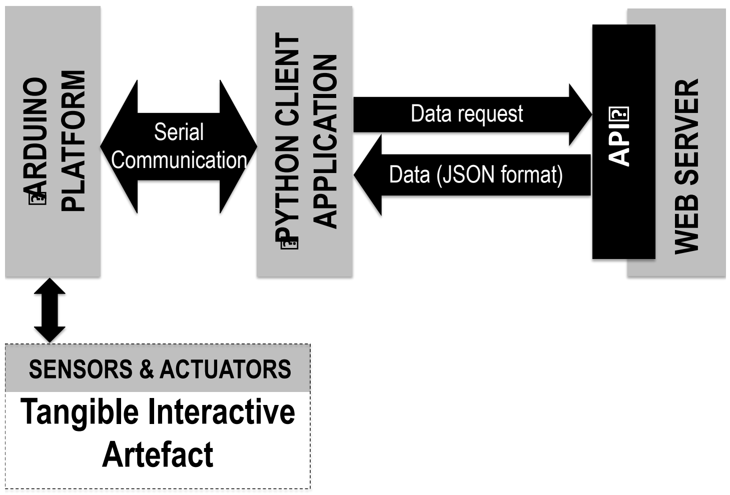

Figure 1. Data materialisation architecture system

For this course to fit in the 16-week timeframe, students should be familiar with the basics of programming: information input (from keyboard)/output, variables and operators, if-else statements, loops, and functions. Thus, the Arduino and Python module focus on how to transfer that previous knowledge to the corresponding platforms. Basic knowledge of digital manufacturing (how to use a laser cutting machine and a 3D printer) is also required to build the artefact during the Hands-on module.

All modules have a hands-on style, with the students receiving the basic explanation of the different skills to learn to then develop their own circuits (Arduino), programmes (Python), and data collection system (Networks and APIs).

4.1 Arduino module (4 weeks)

The objective of this module is for the students to be able to develop the circuits that control the sensors and actuators of the physical artefact. In our experience, the fastest and easiest way of learning the basics of Arduino-based electronics in a few weeks is working along the projects proposed in the book Arduino Projects Book, accompanying the Arduino Starter Kit.

Week 1
The first week is devoted to learning the basics of electronics. The topics of this week are:

- Basic concepts: The concept of electric charges, flow of charges and the role of conductors. The concept of battery and electrical potential. The importance of electrical resistance
- Circutis: Basic resistive circuits. Ohm and Kirchhoff laws. Implementation of circuits on breadboards; measurement of voltage and electrical current. Simple resistive circuit with one LED. Serial and parallel resistive circuits. Voltage divisors.

A practical individual assignment is given to students. A recommended assignment is the control of a RGB LED with three potentiometers (one potentiometer for each colour).

Week 2
After students have learnt the basics of electronics, they start working with the Arduino board. After a brief presentation of the board, students begin building the projects described in the Arduino Starter Kit. We recommend students to work in pairs.
During this week the students should progress through the projects 2-5 from the Arduino Starter kit. Project 2 should be built in class. Remaining projects are built as assignments. The estimated time for each project is about 45-60 min.
By working on projects 2 to 5, students learn how to use switches, LEDs, temperature sensors, photoresistors, and servo motors. In terms of the Arduino board, they learn to work with input digital pins (to read the status of a switch), output digital pins (to activate LEDs), input analog pins (to read

https://doi.org/10.1017/dsi.2019.212 Published online by Cambridge University Press

data from temperature sensors and photoresistors) and output analog pins (PWM) (to control the intensity of LEDs).

Week 3
During this week students work on projects 6-9 from the Arduino Starter Kit. By working on these projects, they learn how to produce sound with a piezo (projects 6 and 7), how to detect movement with a tilt sensor (project 8), how to work with a direct current motor (project 9) and reinforce the utility of voltage divisors (project 7). All projects have an estimated building time of fewer than 50 minutes. We recommend building project 9 in class, to take the opportunity to briefly explain the induction phenomenon of motors.

Week 4
The last week is devoted to developing a minor Arduino project where students can demonstrate that they can build a system with sensors and actuators independently. The project to be built is defined by the students (projects available on the Internet are forbidden) and adjusted by the teachers to have an appropriate level of difficulty.

4.2 Python module (4 weeks)

This module focuses on transferring the previous knowledge about programming to the Python programming language and then dive deep into this programming language. Python is recommended due to several aspects: it is the most used language nowadays, its community is enormous, meaning a high availability of modules for different tasks at hand, and it is multiplatform. The proposed timetable for this module is as follows:

Week 1
This week starts with a brief overview of the Python programming language and the facilities provided by Python with respect to other available programming languages. Next, we present the three alternatives available for a Python development environment: using an online editor, using an offline editor, or using an interactive notebook. In our experience, students like online development environments, such as repl.it, very much. Nevertheless, it has two main drawbacks: first, there are situations where Internet access might be temporarily restricted; and second, an online editor does not allow Python to access local resources. Therefore, we ask students to install an offline distribution provided at Python’s official website [https://www.python.org/downloads/]. This distribution contains both an offline editor called IDLE and an interactive notebook called Jupyter. While IDLE is simple, it also may confuse some students, since it does not integrate the Python environment on it. Therefore, we suggest Jupyter as the default programming environment.
This week focuses on the basic Python instructions belonging to the following categories:

- Input/Output: print(), input(), int() and str().
- Storing and manipulating data: variables, assignment (=), arithmetic operations (+,-,*,/,%,**)
- Decision-making: if-else statements.

Short programmes, such as the typical “Hello World!” or a very basic calculator, are used to make students familiar with the syntax of Python.

Week 2
In week one, students had a first approach to the existence of data types in Python through the use of the int() and string() methods. In this week we discuss data types in detail, with focus on strings, numbers (integer and floating point) to finally reach three of the most relevant data types for a data collection system: lists, tuples, and dictionaries.

The most critical aspect to clarify about these data types is that they are a collection of different types of data with different purposes. A list is a simple collection of values of any type, with no particular order, and that can be modified to add, remove or update values from it. In contraposition, a tuple also is a simple list of values, but elements cannot be added to or removed from it after its creation. Finally, we discuss dictionaries, a type of associative array, which allows us to find elements in a collection

https://doi.org/10.1017/dsi.2019.212 Published online by Cambridge University Press

based on a key. Understanding dictionaries will be key for the students to learn (in the next module) how data is transferred from web servers.

The Python instructions to be learned during this week belong to the following categories:
- Data Types and conversion among them: type(), float(), boolean(), '+' and '*' operators
- Creating and manipulating lists: list(), [], append(), [] notation for indexing
- Creating tuples: list(), ()
- Creating and manipulating dictionaries: dict(), {}

Week 3
After students are familiar with the Python programming language, we focus on loops and functions. We notice that most of the students with no background in Python find it hard to understand the way that Python groups a set of instructions, using indentation instead of curly braces or similar elements. Therefore, we emphasize this aspect of the Python programming language during this week.
Loops, as the name implies, consist of a set of instructions repeated zero or more times, based on an end condition. Python provides two types of loops: while and for loops. The former consists of a set of instructions repeated when a condition is met, while the later repeats a set of instructions a specified number of times. In this workshop, the primary objective of loops is to traverse collections, such as lists and tuples.
Next, we teach how to use functions included in the Python programming language, importing the modules that contain them. The example we use is the random module and the functions to create random numbers from it to create a lottery game. The Python instructions to be learned during this week belong to the following categories:
- Loops: while, for, range
- Modules and functions: import, from/import, the random module, random(), randint()

Week 4
The final week focuses on interacting with local resources, namely: local files and the serial port. The former is useful when it is necessary to back results obtained during the code execution. The latter, when data exchange between Arduino and Python is required, which is one of the relevant aspects when dealing with data materialisation.
When working with local files, the focus is on writing. The reason is that we want to create execution logs, which are useful for debugging. Consequently, we teach the creation of text files in writing mode, and how to write lines of text into the created files.
This week primary goal is to connect a Python programme with an Arduino board using the serial port. We start installing the pyserial library, explaining how this is done using the Python package manager (pip) in Jupyter. We then set a simple circuit and code in Arduino using LEDs to display information received from the serial port. The code is tested using the serial monitor available in the Arduino IDE. Once students test the code, we introduce the usage of the serial library in Python and ask students to send commands to the port the Arduino board is connected to. The result should be the same as when testing the code from the Arduino's serial console. If something is not working correctly on the Python side, we ask students to create an execution log for debugging.
The Python instructions to be learned during this week belong to the following categories:
- Creating local files: open(), with keyword, write()
- Using the serial library in Python: import pyserial, Serial, write(), close()

4.3 Networks & APIs module (3 weeks)

In this module students learn the basics of the client-server model used to collect data from web servers. To do so, they must develop programmes in Python and learn how to use the functions of the libraries *requests* and *json*.

https://doi.org/10.1017/dsi.2019.212 Published online by Cambridge University Press

Week 1
The week starts with an introduction to the client-server model and the *requests* library. The client-server model is explained through an analogy to the web browser and the content on the Internet, telling students that the browser is the client that retrieves information from computers that are available on the Internet, called servers. Then we introduce students to the *requests* library [http://docs.python-requests.org/en/master/] that allows creating a customized web client. Afterward, a simple project is given to the students: to create a simple web browser that receives as an input from a user a URL to get the content from, and then save the content as an HTML file that they will open using the web browser on their computers. The steps to create this browser are:

1. Install the *requests* library using pip
2. Ask the user for a URL using the input() method
3. Retrieve the content of the URL using the get() method and save the content in a local text file
4. Students then open their files, discuss what is missing from the content they got (mainly images) and think how the web browser works and how their naive web client should be modified.

Week 2
The second week focuses on APIs and the JSON format. Leveraging on the web browser case discussed in the previous week, we introduce HTTP, and the "verbs" used when interacting with servers. We explain what these verbs are and how this way of interacting with servers allows users not only to retrieve web pages with graphics and great visual appearance, but also plaintext information. We then describe what a Restful API is, and how it leverages the HTTP verbs and codes. Next, we ask the students to use their simple web browser to retrieve information from the Internet, but this time from an API server rather than a web server. As an example, we utilize the International Space Station API [http://api.open-notify.org/]. Once they download the content, we ask them to open it and to compare it with the content downloaded from a webpage during the previous week. The goal is for them to realise that the format is different to give us the opportunity to introduce the JSON file format. After explaining the format, using the previous knowledge students have about associative arrays, we then introduce the Python library *json*, which enables them to read and write data in that format. We teach them how to integrate both libraries (*requests* and *json*), and ask them to obtain and display the current coordinates of the ISS.

Week 3
Finally, students are prepared to create their first approach to data materialisation. The goal is to display a green led when the ISS is above the current location of the user. In order to achieve this, we need to explain first some further concepts including HTTP error codes, QueryStrings for API requests and DateTime conversions using the *datetime* library. We then ask students to integrate everything they have learned from Arduino and Python to perform the following actions:

1. Retrieve the information about the time when the ISS will be above the students' current location (using the Pass Times method on the ISS API)
2. Read the response and convert it from JSON into a Python dictionary
3. Convert the information stored in the dictionary to the local time
4. Check whether the current time is in an interval stored in the dictionary. If so, send a signal to the Arduino board using the serial library to turn a LED ON
5. When the current time is no longer in range, the LED should be turned OFF
6. The information should be retrieved every 30 seconds to refresh the status of the LED.

With this final guided project, students are prepared to build their own data materialisation project. If students are already familiar with Arduino or Python, such modules can be shortened and an optional module, related to the development of a custom voice assistant, could be also provided at the end of the Networks & API module. Including such optional module has a two-fold objective: to give students the possibility to interact with the user using a voice-based interface, and to teach them how to authenticate in APIs. That module would take 2 weeks.

4.4 Hands-On module (5 weeks)

This module starts presenting the concept of data materialisation with some examples of previous projects. The module can be divided in two stages. In the first one, focused on presentation of data, students are given the following instruction: "Build a digital system that presents live data to the user in a easy to understand manner. Textual data is forbidden". In the second one, interaction must be added. The scheduling of this module is as follows:

Week 1: Presentation of the proposal (including data presentation and interaction)
Week 2: First prototype for data presentation only
Week 3: Second prototype for data presentation only
Week 4: First prototype for data materialisation (data presentation and interaction)
Week 5: Final prototype for data materialisation (data presentation and interaction)

5 EVALUATION OF LEARNING

The evaluation of 1st year and 3rd year students was different, due to the duration of the project and the maturity of the students. In this section we propose an evaluation based on the assessment activities we carried out for 1st year students, since it is more aligned to a 16-week course. Every week, 1st year the students had to upload one or several documents reporting the work of the week. During the feedback session, the students expressed that they felt overloaded with so many deliverables. Hence, we propose only one deliverable per week. Table 2 shows a summary of the weekly deliverables proposed for this course as well as the learning outcome being evaluated by each deliverable.

Table 2. Summary of proposed assessment instances

Week	Deliverable		Week	Deliverable
1	30 sec video showing a home-made battery used to light a LED (LO1)		**9**	Code .py of a simple web browser (LO2)
2	2min-video showing projects 2-5 from Arduino Starter Kit (LO1)		**10**	Code .py for data collection from the ISS (International Space Station) API (LO2)
3	2min-video showing projects 6-9 from Arduino Starter Kit (LO1)		**11**	In class presentation of group project materialising the position of the ISS (International Space Station) (LO2)
4	In class presentation of group project (groups of 4-5 students) (LO1)		**12**	In class proposal of final project (LO3)
5	Code .py of a basic calculator (LO2)		**13**	In class presentation of 1st prototype (LO3)
6	Code .py of a programme that creates and manipulates a dictionary (LO2)		**14**	In class presentation of 2nd prototype (LO3)
7	Code .py that creates a dictionary made of random elements (LO2)		**15**	In class presentation of 3rd prototype (LO3)
8	1-min video of Python-Arduino Serial Communication (LO1, LO2)		**16**	In class presentation of final prototype (LO3)

Each deliverable from weeks 1-15 is worth 5% of the final mark whilst the final project is worth 25%.

6 A SELECTION OF PROJECTS

In this section we present some of the projects developed by our students at the first and second stage of the hands-on module. Thus, some of them were able to include user interaction and others just focused on data presentation. For each project, we describe its functionality and main features.

Can we sail? This project, made by 1st year students, materialised wind data (Figure 2, left), to inform mariners whether it is safe to sail or not. Controlled by a servo motor, the boat of the prototype rotates to display the wind direction. The team also utilised the wind intensity, measured using the Beaufort scale, to display the roughness of the sea, pushing water with a piece of acrylic controlled with a stepper motor (inspired in BeWave, described in (Bordegoni 2017)). Finally, a line of LEDs allows visualising the Beaufort scale.
Let's get the party started This project, made by 1st year students, materialised some characteristics of the music they play on Spotify (Figure 2, right): the energy of the song using a LED strip, the danceability using a dancing character displayed on a LED matrix, and the tempo of the song using a water fountain created by them using water pumps controlled by a solenoid.
ISS, where are you? This project, made by 3rd year students, materialised the position of the ISS using a map equipped with LEDs (Figure 3, left). The red LED shows the current position of the ISS (in the figure, in the north of Russia) and the following position is signalised by a blinking LED.
Seismic time lapse This team of 3rd year students materialised the occurrence of seismic events in one of the most seismic places on Earth for the prototype: Chile (Figure 3, right). They used servomotors to materialise this data about the last 15 earthquakes as a sequence of different intensity movements of the different geographic regions where the earthquakes occurred, emulating what time lapses do.

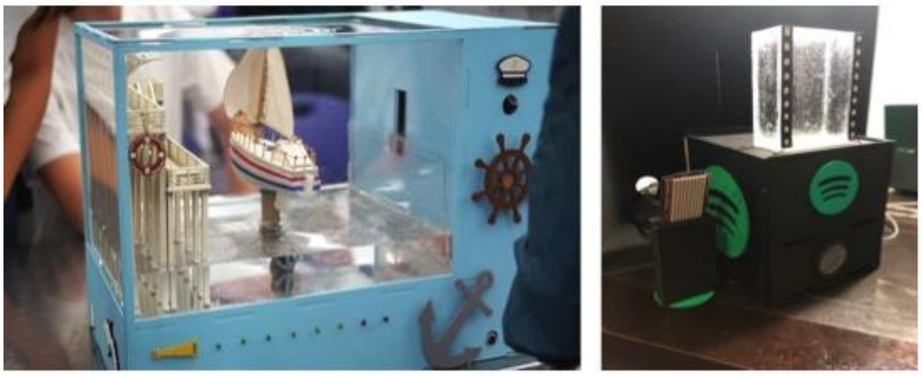

Figure 2. Prototype of data materialisation projects on live wind data on music beat, energy and danceability

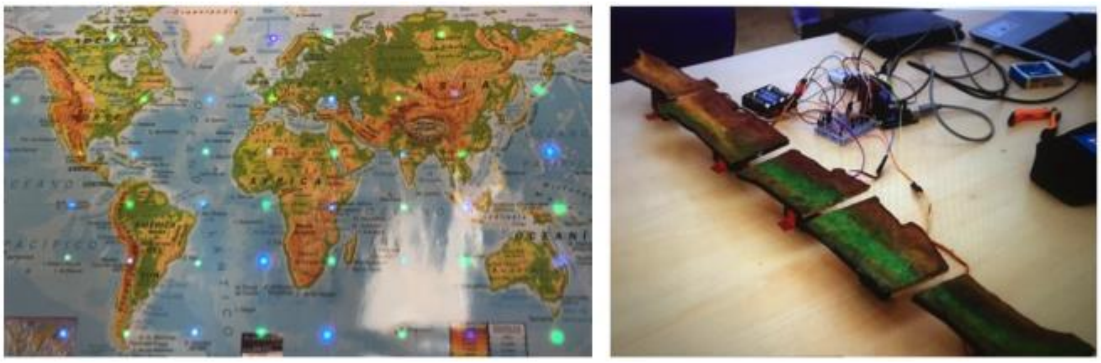

Figure 3. Prototype of a data materialisation project on live data from the ISS

7 STUDENT FEEDBACK

At the end of the course we asked the 1st year students to complete a survey about their learning experience. They used a 5-point Likert scale (1:"I totally disagree with this affirmation", 5: "I totally agree with this affirmation") to evaluate their level of agreement with some affirmations. 44 out of 51 students answered this survey. Table 3 summarises the results regarding 5 affirmations.

https://doi.org/10.1017/dsi.2019.212 Published online by Cambridge University Press

Table 3. Student feedback

Affirmation	1	2	3	4	5	Average
I had a high level of learning in this course	2	2	12	17	11	3.8
The course met my expectations	2	1	7	23	11	3.9
I have acquired new skills in this course	0	2	5	13	25	4.4
I would recommend this course to other students	1	2	10	18	12	3.8
I would take a continuation of this course	3	7	11	8	14	4.0

Regarding the difficulties encountered in the course, the 1st year students mentioned: the high amount of deliverables (more than one per week) and thus, the high amount of time they had to devote to the course, only one Arduino board per group (they had liked to have more to make tests individually) and the inability to change groups as the course progressed. They did not commented on technical difficulties.

8 SUMMARY

With the aim of facilitating the entry of designers to the area of data materialisation, in this paper we described a proposal of a 16-week undergraduate course that equips the students with the basic skills required to device and implement a data materialisation project. The course is structured around 3 learning modules covering aspects of Arduino-based electronics, Python programming and data collection from API web servers and one final hands-on module where students must built their projects. Before starting this course, students must know how to program and the fundamentals of digital manufacturing. An initial pilot run with students from 1st year and 3rd year of engineering showed the feasibility of the course.

REFERENCES

Ariza, J. Á. (2018), "Towards education alternatives to teaching and learning of programming: A course experience using open hardware tools", *2018 IEEE Frontiers in Education Conference (FIE)*, San Jose, CA, USA, (2018), pp. 1–8. https://doi.org/10.1109/FIE.2018.8658657.

Blanco, T., Casas, R., Manchado-Pérezz, E., Asensio, A. and López-Pérez, J. (2017), "From the islands of knowledge to a shared understanding: interdisciplinary and technology literacy for innovation in smart electronic product design", *Int. Journal of Technology and Design Education* Vol. 27, pp:329–362. https://doi.org/10.1007/s10798-015-9347-7.

Carulli, M. and Bordegoni, M. (2017), "Rapid prototyping products mapping live-data streams into tangible user interfaces", *21st International Conference on Engineering Design, ICED 2017*, Vol.9: Design Education, Vancouver, Canada, 21–25 August.

Hogan, T. and Hornecker, E. (2013), "In touch with space: embodying live data for tangible interaction", *7th International Conference on Tangible, Embedded and Embodied Interaction*, Barcelona, Spain, 10-13 February, ACM, pp. 275–278. https://doi.org/10.1145/2460625.2460671

Huron, S., Gourlet, P., Hinrichs, U., Hogan, T. and Jansen, Y. (2017), "Let's Get Physical: Promoting Data Physicalization in Workshop Formats", 2017 *Conference on Designing Interactive Systems, Edinburgh, Scotland*, 10-14 June, ACM, pp. 1409–1422. https://doi.org/10.1145/3064663.3064798

Jansen, Y., Dragicevic, P., Isenberg, P., Alexander, J., Karnik, A., Kildal, J., Subramanian, S. and Hornbæk, K. (2015), "Opportunities and challenges for data physicalization", *33rd Annual ACM Conference on Human Factors in Computing Systems*, Seoul, South Korea, 18-23 April, ACM, pp. 3227–3236. https://doi.org/10.1145/2702123.2702180

Jamieson, P., and Herdtner, J. (2015), "More missing the Boat — Arduino, Raspberry Pi, and small prototyping boards and engineering education needs them," *2015 IEEE Frontiers in Education Conference (FIE), El Paso, TX*, 2015, pp. 1–6. https://doi.org/10.1109/FIE.2015.7344259

Mortimer, R. Haddadi, H., Henderson, T., McAuley, D., Crowcroft, J. and Crabtree, A. (2016), "Human-Data Interaction: The Human Face of the Data-Driven Society", *The Encyclopedia of Human Computer Interaction*, The Interaction Design Foundation http://arxiv.org/abs/1412.6159

Starrett, C., Reiser, S. and Pacio, T. (2018), "Data Materialization: A Hybrid Process of Crafting a Teapot", *Leonardo* Vol. 51, No. 4, pp. 381–385. https://doi.org/10.1145/3202918.3203087

Zhong, X., and Liang, Y. (2016), "Raspberry Pi: an effective vehicle in teaching the internet of things in computer science and engineering", *Electronics*, Vol. 5 No. 3, No. 56. https://doi.org/10.3390/electronics5030056