

FROM CLOUD

UANTWERPEN

SERGE DEMEYER

PROF DR



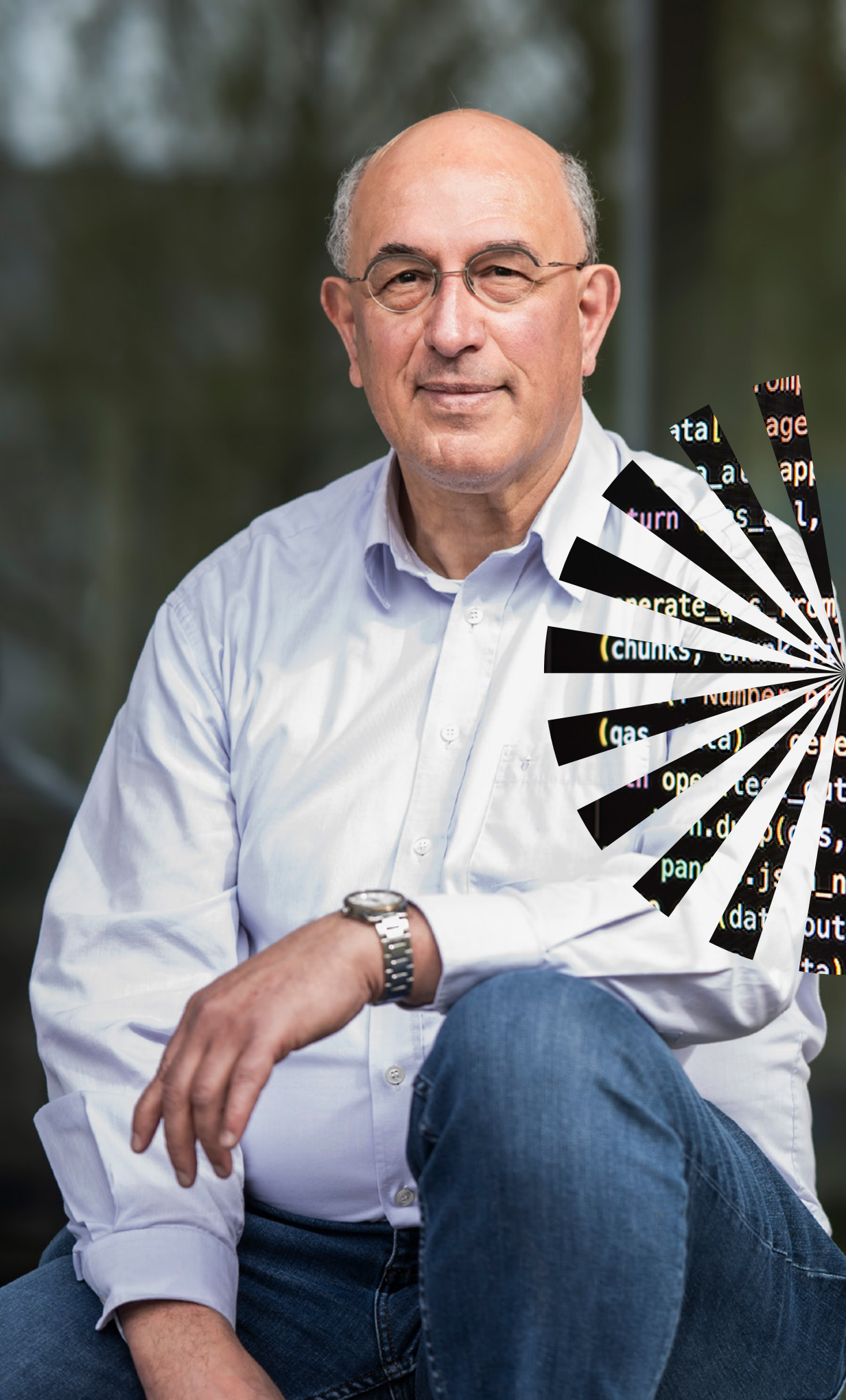
```

    entry in enumerate(output_chunks):
        qa = create_qa_entry(entry, i)
        qas_all.append(qa)
        data['qa_pairs'] = len(output_chunks)
    except ValueError as e:
        print(f"Output text is not a valid question")
        print(e)
        data['qa_pairs'] = 0
    data['model'] = model
    data['temperature'] = temperature
    data['prompt_timing'] = prompt_timing
    data['usage'] = usage
    data_all.append(data)
    return (qas_all, data_all)

def generate_qas_from_chunks_to_file(chunk_data,
                                     chunk_filenames) = get_chunk_data
    print(f"Number of chunks in {chunk_filenames} is {len(chunk_data)}")
    (qas, data) = generate_qas_from_chunks(chunk_data)
    with open(test_outfile_json, "w") as f:
        json.dump(qas, f, indent=4)
    df = pandas.json_normalize(data)
    df.to_csv(outfile_csv, sep=',', index=False)

```

After completing his PhD at the VUB, he moved to the University of Bern. He has now been active as a researcher and professor at Antwerp University for over a quarter of a century. Whether he is working with Flanders Make or the industry, Serge Demeyer is always on the lookout for the perfect software. Even though he knows this is a pipe dream.



```
...pairs'] = len(output_pairs)
...error as e:
...output text is not a valid json string'

pairs'] = 0
...= model
...ture'] = temperature
...timing'] = prompt_timing

...= image
...nd(data)
...data)

...chunks to write(chunk_dir, num_gpus_per_chunk
...frames) = write_chunks(chunk_dir, num_gpus_per
...chunks = {chunk_dir: {len(chunks)}}
...rate_gpus_from_chunks(chunks, chunk_filenames
...ie_json, ... as f:
...f, index=4)
...make(data)
...e_csv, sep=CSV_SEP, decimal=DEC_SEP)
```

CONSOLE

TERMINAL

PORTS

SOURCE CONTROLS

estgeneration % █

SOFTWARE does not fail, testing strategies DO

```
describe(name: 'Booking API - Invalid Cases', () :void => {
  it(name: 'should check if the API allows creating a booking with nonsensical data LI
    const nonsensicalBooking: any = {
      age: 250 // Humans don't live that long
    };

    const response :{ status: number; data: BookingResponse } = await bookingDriver.createBookin

    expect(response.status).toBe( expected: 400);
  });

  it(name: 'should handle creating a booking with a checkout date before the checkin da
    // Checkout date is before checkin date
    const invalidBooking: Booking = {
      firstname: 'Jim',
      lastname: 'Brown',
      totalprice: 111,
      depositaid: true,
      bookingdates: {
        checkin: '2026-05-10',
        checkout: '2026-04-01'
      },
      additionalneeds: 'Breakfast'
    };

    const response :{ status: number; data: BookingResponse } = await bookingDriver.createBookin

    // Expect a client error due to invalid date chronologies.
    expect(response.status).toBe( expected: 400);
  });

  it(name: 'should reject creation of booking without mandatory fields', async () : Prom
    // Checkout date is before checkin date
    const invalidBooking = {
      // ...
    };
  });
```

ASSERTIONS	9	Y	5	5
	10	Y	5	5
	11	Y	7	4
	12	Y	8	4
	13	Y	10	9
	14	Y	10	10
	15	Y	5	5
CODE COVERAGE	16	N	Late -5 to 0	0



From healthcare to mobility, nowadays, software drives just about every sector. At the same time, software systems are becoming increasingly complex. They must be scalable, continuously available and rapidly adaptable. How do you monitor quality in that context? With BaseCamp Zero, a partnership between the University of Antwerp and the VUB, **Serge Demeyer** and his colleagues show how fundamental research can result in concrete tools and methods that have an industrial impact.

10 seconds for Zalando

A structural engineer designs safe bridges that do not collapse. Similarly, a software engineer develops software that must remain reliable, Serge Demeyer explains. "That's why research should not be viewed separately from in practice. In my case, that is the software industry, but the Software Languages Lab at the VUB has also developed tools for the Tax-on-Web application of the FPS Finance, for example."

"Bear in mind: the fact that we work closely with the industry, does not make our research any less fundamental. But just as a biologist has to go into the field to study birds, we have to go out into the industrial world to investigate software problems in their 'habitat'. This conviction is the common theme that runs through my career. It is also evident from my twisting trail of projects and collaborations, and in particular my work at the Antwerp Systems and Software Modelling Research Group, or AnSyMo for short."

"AnSyMo explores how to design, analyse and maintain software-intensive systems. The group brings together four research labs; I myself lead the Lab on REengineering, or LORE. I am involved in almost everything we do, from software testing to the architecture. Because I maintain an overview, we avoid thinking in rigid categories. Those who see the whole picture can shift gears faster, recognise opportunities and build bridges."

"AnSyMo also has a direct link to practice: we are a key partner of Flanders Make, the strategic research centre for the manufacturing industry. Software plays an increasingly important role in that world, and above all: updates follow each other at an ever-increasing pace. You see this at Tesla, which releases a new version of its operating system every six weeks. You also see it with e-commerce

players like Bol and Zalando, which perform an update just about every ten seconds. But all those updates need to be tested, and no human tester can keep up with that pace. That's where BaseCamp Zero comes in."

Chasing a bug

"Given how rapidly the AI landscape is evolving, you could actually call BaseCamp Zero an experienced player. We started in 2019 - tools like ChatGPT and Copilot didn't exist yet. But even then, we saw that the need for efficient testing methods was rapidly increasing, and that artificial intelligence could play a key role in this need. We collaborated with researchers from the VUB led by my counterpart Coen De Roover."

"Our project rests on four pillars. The first is the so-called flaky test. New software is often tested using scripts that automatically run through all the functions. But what if such a test reports an error one time and not the next? Does the problem reside in the code or in the test itself? We have developed a visualisation tool that maps error messages and analyses whether they follow a predictable pattern. If they do, we may have to rework the script."

"Failure reproduction is the second challenge. When a user reports a bug, it often proves difficult to reproduce that error precisely. Without reproduction, diagnosis is not possible, and without a diagnosis, there is no solution."



The bug can be anywhere, especially with cloud applications. Therefore, the VUB created a tool that automatically generates targeted input for the system, allowing us to mimic errors more quickly and consistently.”

“The third pillar is resilience testing. Cloud systems run on networks of collaborating servers. Anything can go wrong there: a server fails, a connection slows down, or you suddenly face overloads. We inject bugs into systems to see how robustly they respond. This allows us to address weaknesses before users are inconvenienced.”

“Finally, on a larger scale, we work with test amplification: using AI to generate new tests or expand existing ones. We are exploring how to refine such prompts as well as where human expertise remains indispensable. AI can produce relatively simple tests, but in complex scenarios, our role remains crucial.”

Just as a biologist
has to go into the
field to study birds,
we have to go out
into the industrial
world to investigate
software problems in
their habitat

Beware of perfection

“That brings me to an important side note. Developers often strive for error-free software. Everything has to work perfectly. Our pilot projects show that such absolute perfection is rarely realistic. For many engineers, it is sufficient when a test automatically detects 97 per cent of bugs. They take care of the remaining 3 per cent – often the most complex cases – themselves. In fact, the total absence of any errors arouses suspicion: if everything always runs smoothly, something is usually wrong.”

“Such insights can only arise through close interaction with the industry. This is also where the added value of FWO research can be found: it is basic research that can be capitalised at the same time. Whether this takes the form of spin-offs, follow-up projects or something completely different. In this respect, FWO forms a continuum

with other research institutions, such as Flanders Make. On the scale of so-called technology readiness levels, or TLRs, fundamental FWO research is generally situated between levels 1 and 4: concept development and early prototypes. Flanders Make operates between levels 4 and 6, where applications are made ready for market. Without in-depth basic research, that progression is impossible.”

More AI = more people

“With BaseCamp Zero, we also want to take that next step ourselves. In time, we will translate our results into concrete, marketable tools. We also develop protocols: guidelines that organisations can use to apply our results independently.”

“Moreover, our approach opens up new perspectives, for example for scientific software. Researchers developing this



FWO research is basic research offering value that can be capitalised at the same time. That's also its added value, whether through spin-offs, follow-up projects or something completely different

type of software are often domain experts but not specialised computer scientists. How can we support them, for example, through automation and intelligent testing strategies?"

"You often hear claims that automation and AI will reduce the number of software engineers - an argument that you see reflected in the declining enrolment figures for computer science courses. I expect just the opposite. As AI plays a bigger role, so does the impact of errors. And when AI generates parts of the code, those errors become more complex and subtle. Conclusion: the more we automate, the greater the need for human quality control."

"Therefore, software development is a social activity par excellence. The cliché of the developer sitting alone behind a screen wearing headphones is outdated.



Quality is only achieved when you work together. FWO also plays a role in this: by investing in our knowledge economy, by encouraging collaboration and - very specifically - by financially supporting two of our PhD students."