RH RQ

Bringing great research ideas into open source communities

Ayse Coskun

On machine learning for operations and how AI can push analytics to the speed of software deployment

+

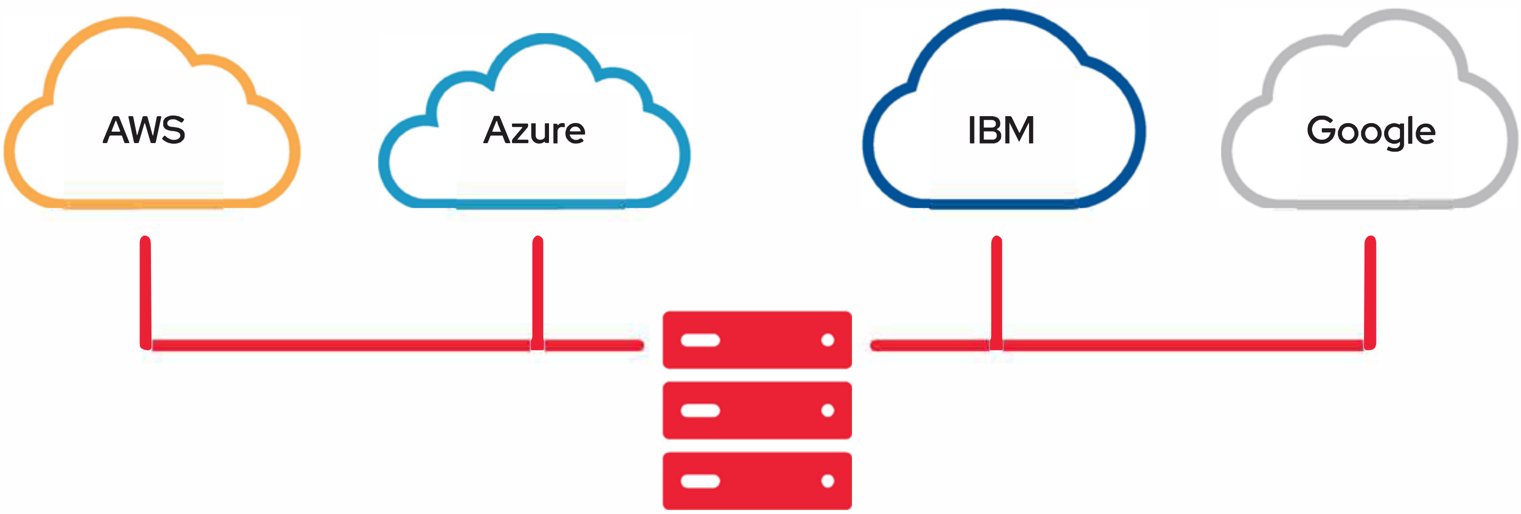
Adaptive streaming

RISC-V for FPGA

Horizon EU: funding open source research

Red Hat
Research Quarterly
Volume 4:1 | May 2022 | ISSN 2691-5278

Clouds that compete



should still connect.

Our code is open_

Red Hat

redhat.com/ourcode

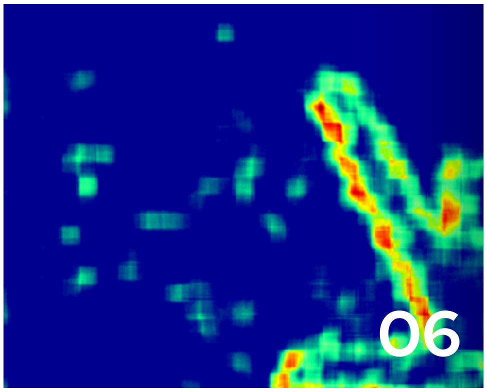
Copyright © 2022 Red Hat, Inc. Red Hat and the Red Hat logo are trademarks or registered trademarks of Red Hat, Inc., in the U.S. and other countries.

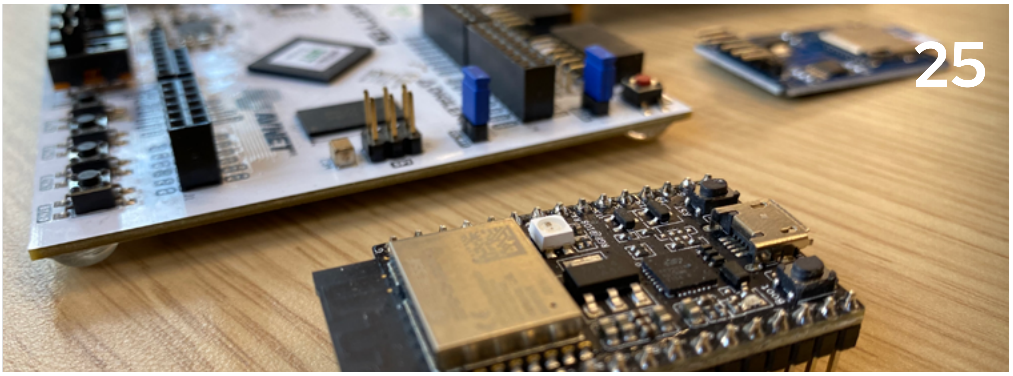
RESEARCH QUARTERLY

VOLUME 4:1

Red Hat

Table of Contents





Departments

Features

ABOUT RED HAT Red Hat is the world's leading provider of open source software solutions, using a community-powered approach to provide reliable and high-performing cloud, Linux®, middleware, storage, and virtualization technologies. Red Hat also offers award-winning support, training, and consulting services. As a connective hub in a global network of enterprises, partners, and open source communities, Red Hat helps create relevant, innovative technologies that liberate resources for growth and prepare customers for the future of IT.

NORTH AMERICA
1 888 REDHAT1

EUROPE, MIDDLE EAST, AND AFRICA
00800 7334 2835
europe@redhat.com

ASIA PACIFIC
+65 6490 4200
apac@redhat.com

LATIN AMERICA
+54 11 4329 7300
info-latam@redhat.com

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

From the director

Three years of making new mistakes—and some great solutions

by Hugh Brock

About the Author
Hugh Brock is the Research Director for Red Hat, coordinating Red Hat research and collaboration with universities, governments, and industry worldwide. A Red Hatter since 2002, Hugh brings intimate knowledge of the complex relationship between upstream projects and shippable products to the task of finding research to bring into the open source world.

Three years ago, I opened my first column in the first issue of this magazine by expressing my sense of good fortune at being able to start something completely new: not just a magazine, but an entire organization devoted to research on computer infrastructure done entirely in open source. Looking back on it today through a post-pandemic lens, I'm surprised to find that my optimism was not misplaced. Despite the uncertainty of building an organization and a research model from scratch, and the Big Surprise of the world more or less shutting down for two years, we have had some remarkable successes at Red Hat Research since 2019. We've learned a lot about how to connect with researchers remotely, brought some great research work into productive use, and expanded our reach in the United States, Europe, and Israel to include many more universities than I ever imagined we would find time to work with.

Of course we have also made some interesting mistakes. I wildly underestimated the effort required to create collaborations among universities in different cities using Red Hat as a conduit. It turns out such efforts are rarely successful, mostly because of the management and communication effort required. To correct this, we have focused on a more decentralized model, which is producing results across the hybrid cloud space with surprisingly little conflict and overlap. We've also found that the gap between

The first issue of Red Hat Research Quarterly

research proof of concept and actual working code in an open source project is even larger than we thought. This limits our ability to take on new work because there are only so many volunteer engineering days available to turn the work we've already committed to into something worthwhile. I suppose this is true of all software, in the end: it takes longer than you think.

Happily, returning to some of our earliest stories reminds me that there is a lot of value in persistence. For example, in our inaugural issue of RHRQ we introduced the wide variety of projects happening at the Red Hat Collaboratory at Boston University. This month's cover interview features BU Prof. Ayse Coskun, whose proposal AI for Cloud Ops was among the first recipients of the Red Hat Collaboratory Research Incubation Awards. Red Hat Research's $1 million grant to her group is the largest award we have ever given to a single project, and it's no surprise. We (and our review panel) believe that the challenge of operations at scale is the greatest challenge confronting computing in general and open source computing in particular. We must learn how to operate large-scale systems and take that learning into the open, and the only way we will be successful is if we can use AI and learning systems to help us.

A crucial part of using AI for this work is covered in another of our featured articles, on the challenges of making stream processing efficient. Stream processing is a key part of AI Ops and many other AI systems: a system often needs to apply AI to draw inferences from a stream of data or events, rather than grinding through a large pool of data looking for answers. Newcastle University researcher and Red Hatter Adam Cattermole describes his work on a library designed to make optimizing stream processing more efficient and more repeatable. This kind of work will be critical as the volume of nodes at the far edge sending data grows by leaps and bounds in the coming years.

Since its inception, we've had a number of pieces on open hardware in RHRQ, which should give you a sense of what an important area this is. This issue's update, from Red Hatter Ahmed Sanaullah, is particularly compelling: he describes using an open source ISA (RISC-V) to make the open source custom circuits a programmer might design into their application easier to access from normal code. One of the signature features of RISC-V is its support for easily and securely extending the ISA, which means that using a softcore—a RISC-V processor mapped onto an FPGA—to provide a consistent interface to custom logic on the same FPGA board can be as simple as calling a function from code. The point of the work is all about making it easier for developers to write for custom hardware. If Ahmed's team is successful, in time developers will wonder how they managed without being able to write custom logic directly to a board and access it from their main application.

Finally, I want to highlight a short review from US Research Director Heidi Picher Dempsey on our signature program to connect university researchers and Red Hat engineers, our Research Interest Groups. Our two RIGs, one for Greater New England and one for Europe and Israel, meet regularly to review ideas for new research we should consider funding or highlighting. The meetings are short and dedicated to technical discussion and debate. If you are interested in what's happening on the cutting edge, reach out to Heidi to learn more.

We must learn how to operate large-scale systems and take that learning into the open, and the only way we will be successful is if we can use AI and learning systems to help us.

News

About the Author
Jason Schlessman is Principal Software Engineer at Red Hat Research, focused on novel artificial intelligence and machine learning innovations that lead to pragmatic and feasible solutions. He especially targets projects that serve the well-being of humanity, fostering ethical uses of technology. He can be found online @ EldritchJS.

UNIVERSITY OF NOTRE DAME

An open source tool to fight visual disinformation

by Jason Schlessman

Red Hat Research is participating in an initiative in the space of image disinformation detection, that is, determination of false information within images that are intended to mislead. The project began January 2021 in response to the need for more mature tooling in the fairly nascent space of image forensics and analysis using statistical and classical machine learning methods.

The project is a collaboration between Red Hat Research and the University of Notre Dame, led by Jason Schlessman on the Red Hat side and Professor Walter Scheirer, a member of UND's Computer Vision Research Lab. The project has produced an open source Python toolkit and is in the process of assessing performance results using a ground-truth dataset.

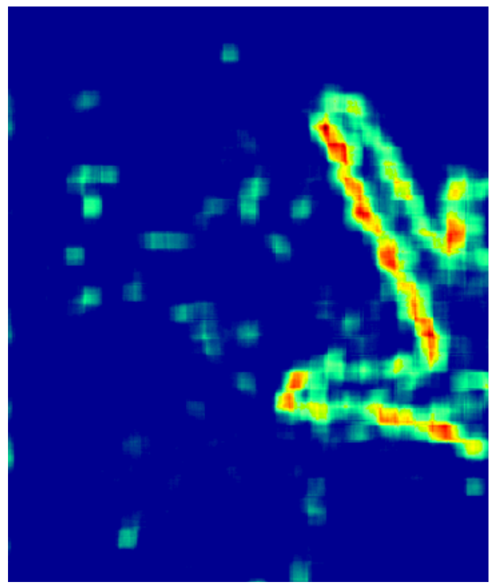
THE UBIQUITY OF DISINFORMATION

Methods for detecting altered images are of particular interest, for many reasons. We live in an image driven world: the apps we use, the sites we browse, and, perhaps most importantly, the social media we engage are fundamentally image centric. This is not limited to static individual images; videos are sequences of images that have their own impact. Each of these sources serves myriad new images we then ingest and process. They have the power to leave a lasting impression on the viewer, which can then propagate rapidly among other individuals due to our society's vast connectedness.

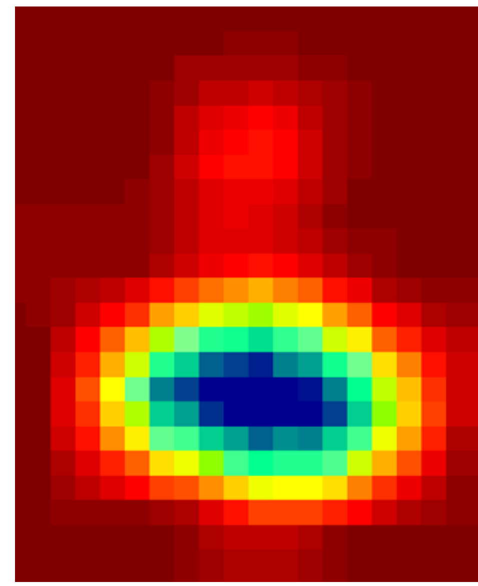
While this level of connectedness and data richness is of huge value to society, we cannot deny the potential for adversarial actors to take advantage of these influential information sources. Given the pervasiveness of image and video data, detecting the operations of these actors is a problem space whose pursuit offers benefits beyond technological progress, having impact legally, socially, and politically. A search for disinformation in a search engine or on a news site will find instances of convincingly altered images spreading as memes that have the potential to disrupt sociopolitical sentiment, from local elections to the Russian invasion of Ukraine.

We enjoyed a brief period when convincing alterations to images were possible only for those with digital art expertise. However, the same technological advances that provide the ability to manipulate images convincingly

Manipulated image with hand symbol added to signal support for a political position



Heat map detecting added hand



Heat map detecting added text

also make it easier for those without domain-specific prowess. We live in a deep fake world that brings these methods to the masses.

FINDING A SCALABLE SOLUTION

In the absence of reliable methods and tooling for detecting fakes, recognizing these images has required explicit one-off identification methods. As technologists, we believe it is imperative to help find efficient means of determining the provenance of images and thus the information they provide to the citizen internet user, as well as the technology executive, the journalist, or the data scientist.

When the Red Hat/UND collaborative effort began, the primary tool available for research beyond deep-learning approaches was a software library that was academic in nature. Meanwhile, Professor Scheirer and other researchers were in need of software development and repeatable workflows approaching the enterprise level Red Hat provides. The collaboration's goals include providing image data security for all and acquiring input from both industry and academia; therefore, the efforts coming from this collaboration are open source and freely available.

A Python package providing an image manipulation toolkit, pyIFD was released publicly in August 2021. Since then, efforts have been made to assess the algorithmic performance of the methods provided in pyIFD with respect to deep-learning approaches, using a ground-truth image dataset. Results from these efforts will be published once complete. Ongoing work is underway regarding the real-time performance potential of these methods. We wish to determine if these methods could be deployed in a live system for immediate detection, aimed at all internet users. For example, could my phone give a warning for a suspect image, or must the detection occur on the server? If we do server-side analysis, given the number of images posted online daily, are the methods scalable?

Beyond this work, a member of Professor Scheirer's lab, William Theisen, will conduct research at Red Hat as a summer 2022 intern. He will explore using multimodal detection methods for combating online disinformation at scale. This would specifically target images having text information (e.g., memes). Does this additional information lead to stronger detectors? Can their performance be achieved using off-the-shelf models on real data? Can this work keep up with the speed of information online? Stay tuned to find out!

About the Author

Gordon Haff is a Technology Advocate at Red Hat, where he works on emerging technology product strategy, writes about tech trends and their business impact, and is a frequent speaker at customer and industry events. His books include How Open Source Ate Software, and his podcast, in which he interviews industry experts, is Innovate @ Open.

Learn more about the history of the MOC in "Massachusetts Open Cloud advances research and innovation," RHRQ 1:1 (May 2019).

What's new in Massachusetts computing infrastructure research?

As the universe of open research clouds keeps expanding, so does the visibility they provide.

by Gordon Haff

In 2014, Orran Krieger, Professor of Electrical and Computer Engineering at Boston University (BU), and Peter Desnoyers, Associate Professor at the Khoury College of Computer Sciences at Northeastern University, launched an initiative called the Mass Open Cloud (MOC). Since then, the MOC has focused on providing researchers with a cloud that enables visibility into the cloud's underlying infrastructure. Such transparency is essential to better understanding performance and other metrics associated with cloud operations.

But transparency—often for good reasons—isn't provided by public cloud providers, which are essentially defined around providing an abstraction at a higher level of the hardware/software stack. As a result, customers have no visibility into the underlying infrastructure's design, behavior, or operation. Public clouds also use a variety of proprietary technology and vendor-specific APIs. Therefore, in addition to providing visibility and access to the underlying infrastructure—thereby enabling researchers to investigate infrastructure design choices that would optimally match with new applications—the MOC set out to provide a cloud that would emphasize open standards and interoperability.

The latest news from the MOC is the formation of an expanded partnership. It will maintain an on-premises Kubernetes-based Red Hat® OpenShift® deployment that includes everything from event reporting and log management to user onboarding and quota reporting. It's essentially a minimum equipment list for operating something like OpenShift in the real world. In addition, there will be other services that run on top of that core infrastructure—for example, Open Data Hub services.

The new alliance of MOC supporters plans to establish a staff that will take responsibility for decisions such as vetting the services that will go into this template and determining that, when a new version of a particular service comes out, it is OK to use. Red Hat intends to fund a project manager, an engineering manager, and four or five developers whose primary job will be effectively curating services to run on top of the Kubernetes infrastructure. Initially, the infrastructure will run supported OpenShift products based on prior donations to BU and Harvard. As participation increases, the infrastructure will expand to include upstream community-supported code as well.

Ongoing activity isn't limited to the MOC, however. A wide variety of overlapping initiatives are taking place in the Commonwealth of Massachusetts, a state with many academic institutions and a long history of research and commercial activity in computer science and engineering.

Most of the hardware associated with these initiatives is housed at the Massachusetts Green High Performance Computing Center (MGHPCC), an intercollegiate high-performance computing facility in Holyoke, MA, completed in 2012. It is a joint venture of BU, Harvard, MIT, Northeastern, and the University of Massachusetts system. In addition to the current MOC-operated OpenShift on bare metal cluster, the original MOC OpenShift on OpenStack cluster in the MGHPCC and the New England Research Cloud (NERC) provide resources for researchers.

> The expanded partnership will maintain a deployment that includes everything from event reporting and log management to user onboarding and quota reporting.

NERC is a newer joint project of BU and Harvard. It will provide access to both OpenStack and OpenShift clusters for researchers at those schools. With Red Hat's assistance, NERC has set up a production OpenStack deployment, which is nearly ready for production. NERC's next project will be to set up a single large OpenShift cluster running on bare metal. The long-term ambition for BU and Harvard is for NERC to become a shared cloud resource for all Massachusetts universities (of which there are many) and eventually for the broader Northeast US region. Because of the way funding works in a research university context, this will be an incremental process that will roll out access to new institutions and research projects over time.

A final related initiative is Operate First, also physically housed at the MGHPCC as a segment of the MOC. Operate First is a project to extend the open source development model to include operating, testing, and validating code in a production cloud environment. In addition to the MOC, Operate First is closely associated with a variety of overlapping initiatives, including OpenInfra Labs (under the Open Infrastructure Foundation) and the Red Hat Collaboratory at Boston University. OpenInfra Labs also hosts the Telemetry Working Group, one of the working groups included under the Operate First umbrella. The working group's goal is to improve the observability of distributed systems through the automatic collection and transmission of data about a system.

As we've seen, a great deal of varied activity is taking place in Massachusetts to provide the computing foundations for ongoing research. The expanded MOC partnership is the latest noteworthy news item, but new developments continue to happen on many different fronts.

Learn more about Operate First in "Ops is the new code: Operate First brings open source to operations," RHRQ 3:4 (February 2022).

News

About the Author
Toke Høiland-Jørgensen is a Principal Kernel Engineer working on networking and BPF. He holds a PhD on the topic of network performance and bufferbloat.

New research on eBPF and security begins at Karlstad University

by Toke Høiland-Jørgensen

January 1, 2022, marked the official start of a new three-year research collaboration between Red Hat Research and Karlstad University around eBPF and security in the Linux kernel. eBPF is a technology that supports running sandboxed code in the running Linux kernel without having to change the source code of the kernel itself. PhD student Bolaji Gbadamosi will be working on this project with joint supervision from Red Hat and from Karlstad University's Dr. Per Hurtig and Dr. Tobias Pulls.

With the introduction of eBPF into the Linux kernel, we are seeing a sea change in the traditional application model. With eBPF, it is now possible to execute parts of the application logic in kernel space, leading to a novel hybrid userspace/kernel model. This is an exciting development that brings with it many opportunities but also some challenges, especially in the area of security.

The goal of the new project is to assess the various security issues that arise with eBPF technology, as well as possible mitigations for any issues identified. Possible topics for exploration include resource allocation and constraints, memory safety, eBPF in safety-critical applications, and using cryptographic signatures for eBPF programs. These topics are likely to evolve as the work continues.

The project builds on the existing research collaboration between Red Hat and Karlstad University around the topic of programmable networking using eBPF. The research team seeks to draw on both Red Hat's expertise in Linux kernel and eBPF development and Karlstad University's expertise in academic security research. Project results will be available as open access, open source software, and open data, so stay tuned for more news to come.

Red Hat will offer collected teaching materials online

by Matej Hrušovský

Red Hatters have been teaching at universities for almost fifteen years. Now, all that amassed expertise will be available on the Red Hat Research website to be used more widely.

The relationship between Red Hatters and these universities has a long history. This is especially true in Brno, where a substantial portion of today's senior engineers were once students of local institutions. Many of these graduates continued to have good relations with their alma maters even after they started their careers as engineers and eventually chose to teach on the side. Soon, Red Hat began to support these activities, which were the beginnings of today's incredibly successful, still-growing research program.

Putting together a semester-long course (in Brno, this means thirteen to fourteen lectures per semester) is a significant time investment. It took years to build these courses. Red Hat instructors have taught Python, Linux administration, Ruby, and Java. Some of the courses became obsolete over time, but others are still current, and new courses have also emerged.

To foster the growth of industry involvement in this area, Red Hat has decided to share many of its teaching materials online. In most cases, the materials have always been public, but now they will be collected in one place and easy to find. This will allow other enthusiasts to teach the same topics in their own corner of the world and, hopefully, encourage more materials to be shared by others.

Visit research.redhat.com/courses to see the teaching materials currently available.

About the Author

Matej Hrušovský has been with Red Hat for more than eight years, six of which have been spent managing the university program in EMEA. Aside from attracting new talent mainly from universities and schools, the core of Matej's job is to find and put the right people from Red Hat and academia in the same room together.

Machine learning for operations

Can AI push analytics to the speed of software deployment?

An interview with ***Ayse Coskun***

RESEARCH QUARTERLY

VOLUME 4:1

Red Hat

Interview

RHRQ asked Professor Ayse Coskun of the Electrical and Computer Engineering Department at Boston University to sit down for an interview with Red Hatter Marcel Hild. Professor Coskun is one of the Principal Investigators on the project AI for Cloud Ops, which recently won a $1 million Red Hat Collaboratory Research Incubation Award. Their conversation delves into the need for operations-focused research on real-world systems and the capacity of more mature AI technology to solve problems on a large scale.

About the Author

Marcel Hild

has 25+ years of experience in open source business and development. Now he researches the topic of AIOps in the Office of the CTO at Red Hat, proving how AI will help operating machines and applications.

Marcel Hild: Let's start with learning a bit about you and how you got where you are now. You grew up in Turkey and then moved over to the United States, yes?

Ayse Coskun: Yes. I went to college in Turkey as well, to Sabanci University in Istanbul, and I completed a degree in microelectronics engineering. Originally I wanted to pursue a PhD in circuit design, because that was familiar, but one thing led to another and I landed in the University of California San Diego's Computer Science and Engineering department. I started working at the intersection of systems, electronic design automation (EDA), and computer architecture, which is where most of my work still lies.

My PhD focused on finding better ways to manage the temperature and energy efficiency of processors, particularly multicore processors. For three years of my PhD I was also working part-time at Sun Microsystems (later acquired by Oracle). It was a really sweet spot: I could do my research but still learn about industry, get involved in patents, and get involved in product-related things. Sun was a unique company that designed hardware, operating systems, and applications, so it gave me a lot of visibility into different layers of the system. After graduation, I became a professor at BU, and I've been here since 2009. Some of my research is still looking at energy efficiency, but at different levels of computer systems.

Over the last decade, I started to work on cloud and large-scale systems broadly. I created this new research thread in my group where we started looking into datacenters and energy issues, but eventually my research evolved into "analytics" on large-scale computing systems, like those serving the cloud. Improving energy efficiency is a significant goal, but there are other interesting problems too, like performance issues or vulnerabilities.

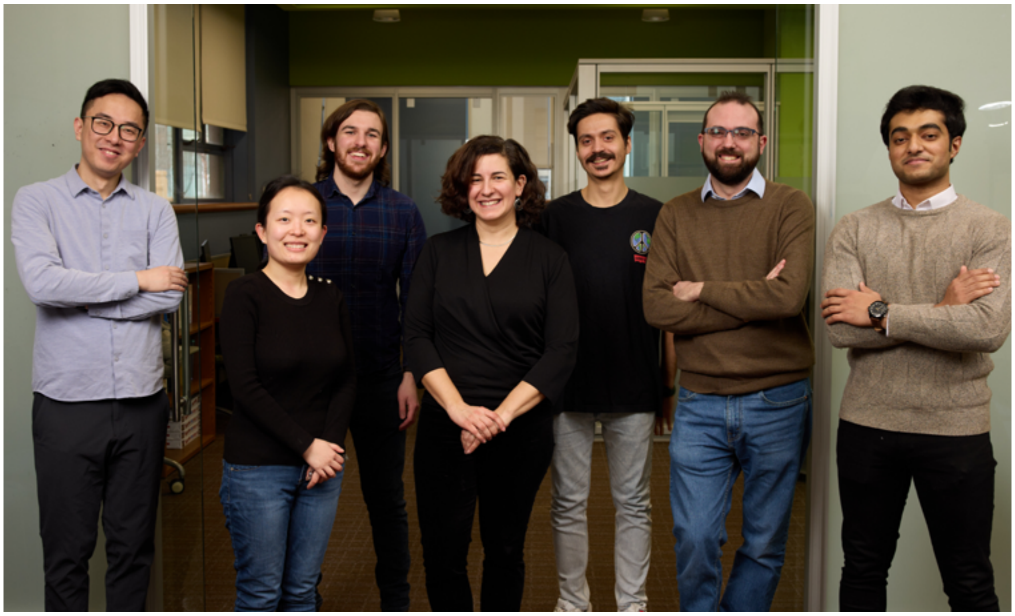
Marcel Hild: I like this transition from studying integrated circuits to focusing on energy efficiency, eventually ending up in artificial intelligence and algorithms to improve something. In the end, it all boils down to making things more efficient.

Red Hat's Collaboratory, which is supporting the AI for Cloud Ops project, is a partnership between Red Hat and BU. Red Hat benefits by having a positive impact on the projects we're involved in and providing some exposure to the research community. Is there something you hope to gain by working with Red Hat or with industry in general?

Ayse Coskun: Definitely. The funding component is an enabler, so we can have a team dedicated to working on these cool problems. But in the end, what we want is innovative research and impact. Having an industry partner really gives us an edge, because we get feedback on the problems and constraints that are most relevant. Getting this kind of feedback helps us solve actual problems for society, and we can see immediate results on actual products.

Normally it takes a long cycle for a research idea to get on to a system where it meets real people. This is by design: as researchers, we work on problems that are not immediately the next step in a product cycle. But the cloud is a very specific area. Things change so fast—people design and deploy new tools, applications, and software stacks all the time. Having industry support enables you to see how these research outcomes will work on real systems. Otherwise, we might find a cool technique, but by the time somebody in industry looks at it, it could be obsolete.

That's an exciting factor, and it excites my team and my students too. PhD students want to solve problems—they want to make a dent in this world. When they see the industry interested in what they're doing, when they see their method working on an actual system solving a problem, they get super enthusiastic about doing more.



The AI for Cloud Ops team, left to right: Alan Liu, Lesley Zhou, Anthony Byrne, Ayse Coskun, Mert Toslali, Gianluca Stringhini, and Saad Ullah

Marcel Hild: That's also what excites me when working out of the Office of the CTO and with academia—we have that freedom of researching and doing something without product constraints, but it always also goes back to something that can be applied. We don't just reinvent an algorithm for the tenth time—we actually solve problems. That's how I came to the AIOps definition that I now hold, that AIOps is not just a product, but a cultural change where we apply AI and machine learning tools to the realm of operations.

Some of the recurring themes in your proposal sound similar to AIOps. Can you talk about why we need AI these days in the domain of operations and in the domain of clouds? Was there something that changed fundamentally?

Ayse Coskun: That's a great question. We can run a cloud without AI. But several things are now calling for AI and better, more automated approaches. And there are some enablers: we are at a point where AI, as well as the tools and the techniques that come with AI, have reached a certain level of maturity. There's more expertise all around, and it's possible to do more complex applied things more easily.

The other side is the cloud, and the multiple layers of software running on top of each cloud instance. Things change so fast. We have these continuous integration, continuous deployment (CI/CD) cycles, where people are constantly changing something related to their code. With a traditional approach, you might have an expert who understands the system well, and they write really good code to manage some resource or fix a problem.

But things become so fragile in an environment where software is updated multiple times a day. You can write some script to look for a file that exists, but then the version of the software that kicked in today doesn't have that file anymore, or the file size changed. Now your script will need to be updated too. Relying on human-involved practices can be very costly. If you have an expert, that means they have valuable information. You are paying those people a lot of money to maintain systems and debug problems.

It's inefficient. This kind of expertise doesn't scale. I can't suddenly train up a hundred people to manage a new system. That's why it's the right time to bring more automation into the operations of the cloud.

One clarification: when people talk about machine learning and the cloud,

Whiteboarding with the AI for Cloud Ops team

they are often talking about running machine learning applications on the cloud. Here, I'm talking about applying machine learning to cloud operations: how resources and software are managed, how vulnerabilities are determined, how compliance checks are processed, and things like that. Other fields have seen a lot more applied machine learning already. We are recognizing images with AI; there's AI in robotics and in our shopping assistants. But when we look at how computers themselves are managed, it's still heavily reliant on human expertise.

Marcel Hild: Some of the recurring themes in your proposal were performance, resilience, and security. You also broke it down into two main areas: software discovery and runtime analysis. Should we start with software discovery?

Ayse Coskun: Yes, but first let me take a step back. What we want to achieve is bringing AI to cloud operations. That's a broad problem. There are many things one can do, and we want to do this because we want to find performance problems quickly. We want to find vulnerabilities quickly. We want to make systems more resilient: if something is crashing, we want to determine what's causing it and eliminate it quickly. Software discovery looks into what's happening in the system in terms of what specific software is running.

Here's an example. We talked about CI/CD, where people keep changing their code, deploying it on the cloud, and running it. Sometimes this happens through Jupyter Notebooks: I'm doing this data science application, I change a little bit in my code, and I send it and run it again. It used to take five minutes, but for some reason now it takes an hour. Or, I used to have everything working, now it's crashing. Or, apparently I included a vulnerable application that has a known problem, and eventually it's going to make me open to privacy leaks.

These are the kinds of things we want to determine right away, even before running the software when possible. Some of the work we've done looks into which files were created and modified during software installation. We have efficient techniques to create fingerprints of what's going on in the system and use machine learning to determine, say, that a certain version of this application shouldn't run on a certain cloud. Some techniques are also good at looking into the code. Maybe you just used a library or implemented something new in a function. We can get some signatures of that code and see if it is known to cause trouble.

Runtime analytics looks into what's going on while your application is running on the cloud. Can we collect the right amount of information to understand why something slowed down? Can we get these cross-layer analytics—meaning the information collected from different layers of the cloud? And then can we mesh this data, structure it, and use it to understand if there's a security breach, or malicious activity, or some problem in the network?

Then we can tie this information back to a specific problem. For instance, I made a change in my software, and now it's slow. Runtime analytics can tell me what's driving the latency problem. Then I can tell the developer, "Since you installed this library, or since you included this function in your code, it's been causing this particular problem."

Red Hat

One thing that enables this line of research is the community embrace of a strong open source culture over the last few years

This would have tremendous value and save a lot of cycles in debugging and system management effort. These examples can be expanded for performance problems, security problems, and resilience problems.

Marcel Hild: That sounds very complex. You're not doing it just on your own; you have a team. How many people are on your team for this?

Ayse Coskun: We have three principal investigators (PIs). I'm leading the team. Gianluca Stringhini is an assistant professor here at BU and a system security expert. He brings a lot of value to the project because he has already been doing work that brings machine learning and analytics into security problems and privacy issues.

Alan Liu is also an assistant professor at BU. He's a networks and systems researcher, so he's going to bring his expertise in making efficient telemetry work for understanding problems on the network layer. Typically there's no shortage of data: you can collect data from the application and from the system, you can also get data from the firmware/hardware. The problem is doing this efficiently. If you collect everything, who's going to look at it? And what kind of overhead does that create? Alan's work on network telemetry and "sketches" is a game changer, because he captures the relevant information while being very efficient about it.

We'll also have four PhD students on the project: Anthony Byrne, Mert Toslali, Saad Ullah, and Lesley Zhou. Anthony is already interning at Red Hat at the moment. Mert has done some really nice work with IBM Research on automating software experimentation on the cloud. We'll have some undergraduates working as well. Brian Jung, an undergraduate student, is already working with us on building the first version of a software discovery analytics agent for Jupyter Notebooks.

Marcel Hild: Before this, you've been working on different puzzle pieces. Now you're applying it to the complete stack and a living environment. What is one of the first outcomes you expect there? Will there be a unified user interface, will there be a unified proof of concept where all the things come together?

Ayse Coskun: True, and great questions. We played around quite a bit on these topics prior to this project. We learned some things, we had some ideas that didn't work, and we saw some success stories. For instance, some of our initial work was focused mainly on virtual machines. Then it focused on containers, then microservices. Now we'll look into the Operate First environment. Each of these changes brings some specific challenge. We have a lot of know-how, but we'll need to make it work for these Operate First clouds.

One of the first things we want to demonstrate is a system visualization tool utilizing different kinds of data—cross-layer, telemetry, tracing, runtime—for the developer or system manager. We want to create an interface where this data is captured and then presented back through relevant APIs, so it's easy for the developer, user, or administrator to query this data.

Another goal is related to software discovery. We have two tools that have already been released to the open

source community over the last few years. One is Praxi, which determines whether a software installation has any vulnerable code or prohibited application in it. The other is ACE, or Approximate Concrete Execution. ACE executes functions (approximately) and creates signatures of them very quickly, then searches across libraries of functions known to be problematic, identifies them, and gives feedback to the user. Now, we want to implement and apply similar techniques for efficient searches of software. The efficiency comes from machine learning. Then we want to apply similar techniques for the bring-your-own-notebook concept. People come in with a Jupyter Notebook; they are writing some code. Through these APIs, we want to tell them, "Hey, you just installed something vulnerable, so remove it," or "You are about to run something that's known to be buggy."

Our second year will focus more on getting everything to work together, including the analytics for security, fault diagnosis, and efficient cross-layer telemetry. Eventually we'd like to have a full deployment of our unified API and analytics engines on the cloud systems.

Marcel Hild: I like that you focus on the people that give you feedback. If you provide a visual tool for humans, you get immediate feedback on whether you are solving problems for them or just creating your personal echo chamber where the model is predicting something mathematically but doesn't bring any value to the user. And you are bringing it to this Operate First community, where you have people using your tools and providing some traffic and usage to prototype and develop what you're doing. Otherwise you would be working with just synthetic data on hypothetical setup.

How can people reach out to you about this? How do you envision working with the community and engineers?

Ayse Coskun: Another great question. I mentioned earlier that there are now enablers for this research to happen at this scale. Similarly, one thing that enables this line of research is the community embrace of a strong open source culture over the last few years. If everything were proprietary, it would be difficult to make an impact in this space. We need real systems and real users to demonstrate things in a rapidly changing environment.

And how will we make that happen? Obviously we'll make our artifacts available in places like GitHub, but maybe that is also a question for you! We'd like to work with Red Hat to identify the right communities to push some of these things upstream. Open sourcing is not the end of the story. I can make our work open, but if nobody looks at it, it's hard to call that success. But if the right tools reach the right communities, that's where we can make an impact.

Marcel Hild: If this project is a success, it can act as a role model for how other large projects work. You need the whole platform stack to create something that works; therefore, you need a community where the whole stack is practiced and where it's available. The Operate First community is growing and can provide that.

Ayse Coskun: One thing I want to add is that Red Hat has already been doing some work in the AI for cloud operations area. One of our strategies is to avoid reinventing the wheel. It's better to deploy our methods in existing

infrastructures, tools, and projects. For instance, we identified Project Thoth, which is managed as part of Red Hat's AI Center of Excellence, as a place where we can deploy some of our methods.

It's already there. It has a community around it. So perhaps we can leverage Thoth and similar projects. This is a good way to reach an existing community, but offer them more, as opposed to putting standalone independent code repositories out there that people may or may not use. Instead, the plugin becomes available through something users are already engaged in.

Marcel Hild: Yes, absolutely. It's AI for operations, and how can you engage with people in operations without operating something?

I also want to discuss a different side of your work. You're the founder and organizer of the Advancing Diversity in Electronic Design Automation (DivEDA) forum. Tell us about that.

Ayse Coskun: One of the topics I've been passionate about in conjunction with my technical work is diversity.

There are far fewer women and underrepresented minority researchers in computer science and engineering in general. I'd like to change this so women and underrepresented minorities have a stronger role in shaping our future in computing.

There are many efforts towards the same goal at different layers: educational efforts, efforts at different companies, and so on. What we did was establish a forum as part of two conferences (one in the US and the other in Europe). The forum included panelists who talked about their experiences and how they overcame difficulties in their careers. There were panelists from different stages of their career, people from industry and academia, people who are more senior or more junior. It was a really lively, interactive environment. We also had some speed mentoring sessions where more senior, more accomplished people met with PhD students or people in the early stage of their careers. The idea was, okay, you meet a couple of people during the session, and maybe next time you see them, you can have more of a conversation and build a mentoring relationship.

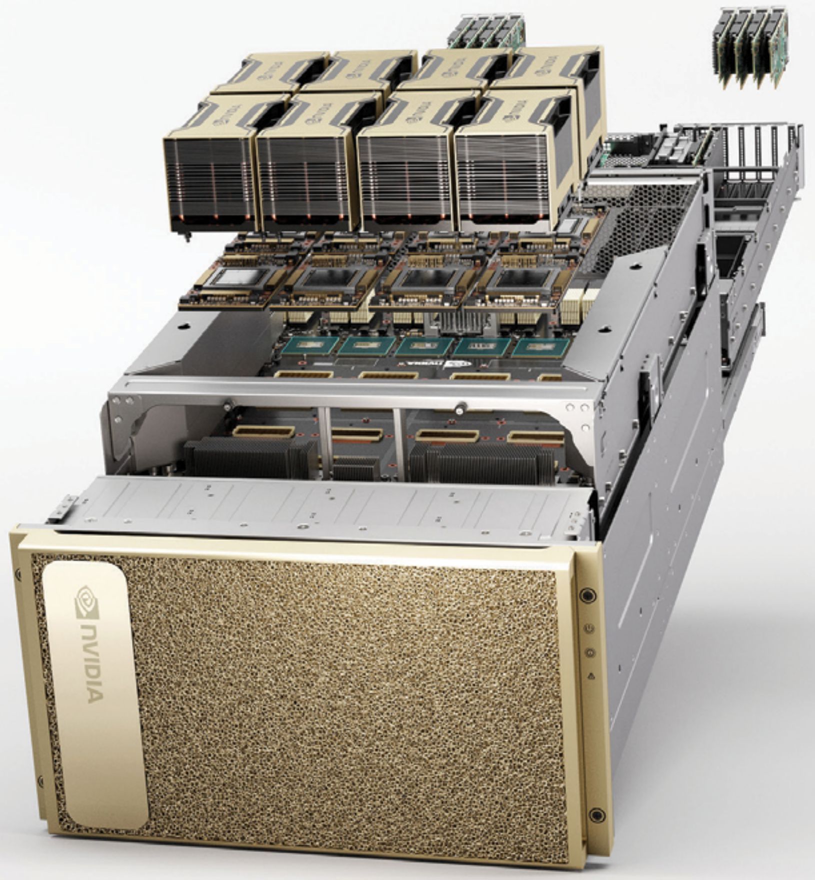
Studies have shown that having a mentor helps a lot in shaping up your career: making sure that you get nominated for the right things, that you attempt to get promoted and go for certain accomplishments. We wanted to create an enabler to facilitate that. We ran the forum a few times, and then the pandemic happened. This year DivEDA is finally back, but it's going to be virtual.

Marcel Hild: That's awesome. I would be super excited if you also bring some of these ideas to the AI for Cloud Ops project, or to the Operate First community. The more people we involve, the better. I don't know if we get there in our lifetime, but I like the idea of making the lack of diversity an obsolete problem.

Ayse Coskun: Yes, I love that idea too. I would love it to become obsolete, but I think it's going to be with us for a while. Unlike the continuous integration and deployment environment, societal change happens more slowly. <laugh> We can't deploy it like new software a few times a day, but change does happen. And sometimes in our lifetime!

Marcel Hild: I'm looking forward to working with you in this Operate First context, and in this AIOps context. We've finally come to a point where we can go beyond just a proof of concept or a paper. Now we can bring all the stakeholders together and really do something.

Ayse Coskun: I agree. We are very excited and hoping for some cool demonstrations soon.



NVIDIA

THE UNIVERSAL AI SYSTEM FOR HIGHER EDUCATION AND RESEARCH

NVIDIA DGX A100

Higher education and research institutions are the pioneers of innovation, entrusted to train future academics, faculty, and researchers on emerging technologies like AI, data analytics, scientific simulation, and visualization. These technologies require powerful compute infrastructure, enabling the fastest time to scientific exploration and insights. NVIDIA® DGX™ A100 unifies all workloads with top performance, simplifies infrastructure deployment, delivers cost savings, and equips the next generation with a powerful, state-of-the art GPU infrastructure.

Learn More About **DGX** @ nvda.ws/dgx-pod
Learn More About **DGX on OpenShift** @ nvda.ws/dgx-openshift

© 2020 NVIDIA Corporation. All rights reserved. NVIDIA, the NVIDIA logo, and DGX are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and/or other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

RESEARCH QUARTERLY

VOLUME 4:1

Feature

Adaptive streaming using Strimzi and Apache Kafka

The competing demands of cost and performance make it challenging to optimize stream-processing applications. Current research is exploring new options.

by Adam Cattermole

About the Author
Adam Cattermole is a PhD candidate in the Cloud Computing for Big Data Centre for Doctoral Training (CDT) at Newcastle University, UK. His research interests are primarily in distributed stream processing, containerization, and automated deployment to both the cloud and the edge. A former Red Hat intern, Adam is in the final stages of his PhD program and recently joined Red Hat as a Software Engineer working as part of the Red Hat® OpenShift® API management team.

Extracting value from streams of events generated by sensors and software has become key to the success of many important classes of applications. However, writing streaming data applications is not easy. Developers are confronted with major challenges, including processing events arriving at high rates, distributing processing over a set of heterogeneous platforms ranging from sensors to cloud servers, and meeting nonfunctional requirements such as energy, networking, security, and performance. The data within these applications can be largely dynamic, which requires the streaming system to adapt to ever-changing demands.

Stream-processing applications are commonly referred to as topologies. A topology consists of a set of stream-processing operations partitioned and deployed on available resources. The systems executing these partitions vary from low-power edge devices and sensors to virtual machines running in the cloud. This presents a challenge for the developer: how to partition the topology and how to make the best placement decisions.

An important aspect of this challenge is satisfying performance measures, particularly throughput and latency. When first deployed, a system might meet the desired performance criteria, but when the data it is processing changes, performance can suffer. Twitter is a good example: a major event occurring somewhere in the world can cause a large spike in traffic. Depending on the application, these data characteristics may be predictable, so the streaming topology could be modified in advance of peak load to satisfy the demand (e.g., during a US election). In other cases, the data rate may be more volatile, requiring a reactive approach.

Much of the existing research in this space aims to develop cost models to dynamically scale one or more of the partitions within a topology as the load increases, or to migrate partitions to increase resource utilization. However, we are interested in more complex adaptations to the streaming topology. If the network cost between two partitions is high but the processing required is low, is it possible to combine two partitions into one in real time? If the inverse is true, can we split one partition into multiple partitions? We describe these adaptations as operator fusion and operator fission, respectively.

A FUNCTIONAL APPROACH

Over the course of the research project, we developed StrIoT, a functional stream-processing system written in the Haskell programming language. The StrIoT library provides a set of stream-processing operators that can combine to create complex applications on streaming data. This includes definitions of operations commonly found among existing streaming systems, including Map, Filter, Window, and Combine style operations. Each of these functions was designed to be simple to use and understand, and they are easy to compose.

To provide a simple example, imagine a temperature sensor that generates a stream of type Int.

```
tempSensor :: Stream Int
```

Suppose that the only values of interest are those over 100. The application developer can define a function for this:

```
over100 :: EventFilter Int
over100 temp = temp > 100
```

The stream can then be filtered:

```
streamFilter over100 tempSensor
```

The application developer may want to use 100 as the baseline temperature and represent all temperatures as their value over 100. To do this, the developer can define a function:

```
amountOver100 :: EventMap Int Int
amountOver100 temp = temp - 100
```

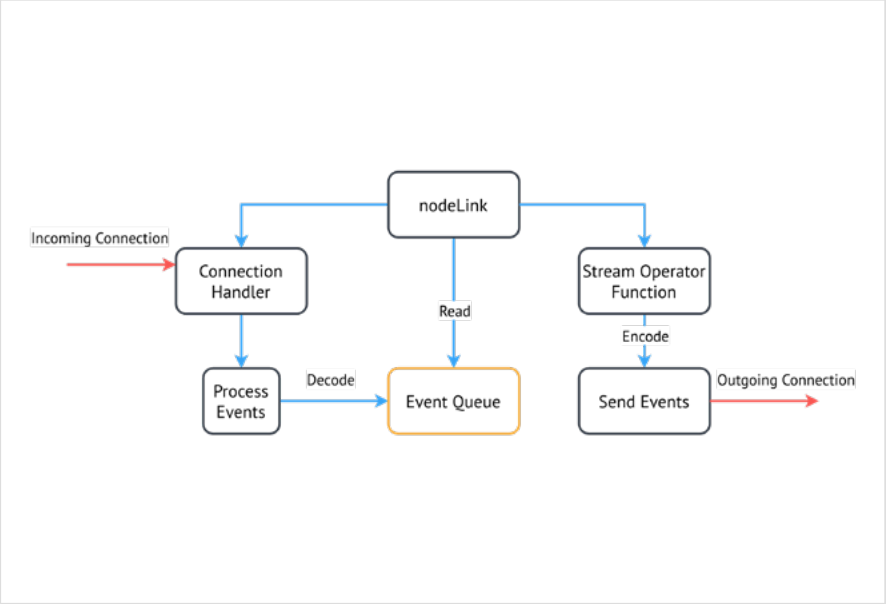


Figure 1. *Partition internal architecture*

Now, the developer can make a single application that combines both of these functions. The $ symbol is used in Haskell to chain functions so that the output of one becomes the input of another.

```
streamMap amountOver100
     $ streamFilter over100 tempSensor
```

Each partition contains a set of one or more of these composed stream-operator functions and follows the architecture of a typical networked service. **Figure 1** depicts a link-style partition, where events are arriving from an upstream partition and must be sent to a downstream partition post-processing. An input thread, running concurrently to the main execution thread performing the stream operators, processes these events. The events are decoded and input into a fixed-size queue ready for processing by the stream operator(s), then encoded before emitting downstream. The contents of each partition in this form are compiled like any other Haskell program.

Given that the template for each of the partitions is very similar, the deployment code is generalized to allow particulars of the operation to be determined at runtime. As each partition would likely be deployed in a container or onto a virtual machine, reading from environment variables is the most appropriate method of providing configuration to the application. The main benefit to this approach is that each partition of the streaming topology can be built ready for deployment without containing the networking configuration, which can instead be provided at runtime.

DEPLOYMENT

We considered several different partition deployment methods during the development phase of the StrIoT system. These included the possibility of using cloud virtual machines and the use of serverless technologies, as well as the option of using containerization. We decided on containerization because containers allow a developer to encapsulate their application in a predefined environment. This environment contains all the dependencies required to run the application, and the images can be deployed on a variety of platforms without concern for the hardware specifics. This ensures that environmental conflicts do not occur from one user to another, and that the application conditions are reproducible. In the case of StrIoT, all of the build dependencies and tools required to build and compile a Haskell program are provisioned within the container. All of the packages that a standard partition requires are included in a base image. Then we extend the base image to include source code and any custom resources that a specific partition of the streaming system needs.

We chose Kubernetes as the means of deployment, as it provides a simple interface to deploy, scale, and manage containerized applications. The smallest possible set of one or more running containers for any deployed application is a pod. At runtime, the number of pod replicas can easily be changed. Kubernetes handles the creation and scheduling of new containers. When using Kubernetes as a platform for StrIoT, each partition of a topology is deployed as a separate application, and the individual pods are networked together. In this case, each pod consists of a single container with the same base control-flow behavior as shown in Figure 1, but with differing stream operators to match the partition.

ADAPTIVITY

Previously, I introduced two complementary adaptations with differing benefits and costs: fusion and fission.

The main concern when performing either of these adaptations is ensuring that "effectively-once" processing is unaffected; i.e., that events are not duplicated, nor events lost, provided there are no failures. "Effectively-once" describes the effect of processing the stream and the expected output. If the adaptation results in sending the same event multiple times, duplicates must be filtered out before being processed so the application and result are the same as if the event were sent exactly once. Events cannot be lost; in the case of a missing event, it must be re-sent in the original sequence so it does not affect the expected result of the application.

Another issue to address is the stream-operator functions themselves: how will they behave and be affected by adaptations. Some of the stream-operator functions are classified as stateless functions, meaning the processing is performed on the current event and has no dependency on previous events. Other operators contain state that is used to compute the new values of the stream, which are classified as stateful operators. If a partition containing a stateful operator is the subject of an adaptivity operation, any state contained within the partition must be unaffected.

Fusion

In the fusion adaptation, two separate adjacent partitions are combined into a single partition, as shown in **Figure 2**. Fusion results in a single partition that contains all the stream-operator functions that were in the original two partitions. The fusion adaptation requires a performance trade-off, and the benefits depend on the application in question. By fusing two partitions into one, we remove the boundary between partitions, resulting in reduced network transfer costs. The events do not need to be serialized/deserialized and sent between partitions over the network. However, fusion also results in reduced pipeline parallelism. Prior to adaptation, both partitions were processing events at different points in the stream simultaneously. However, the combined partition is only processing the most recent event received at the partition.

Fission

The fission adaptation is the counterpart of fusion. In fission, the action of fusion is reversed: a single partition containing several stream-operator functions is split into two different partitions, each containing a subset of the original stream-operator functions (see **Figure 3**). The benefits and costs of fission are the inverse of fusion: potentially increased performance through pipeline parallelism, with additional network transfer costs due to an additional partition boundary.

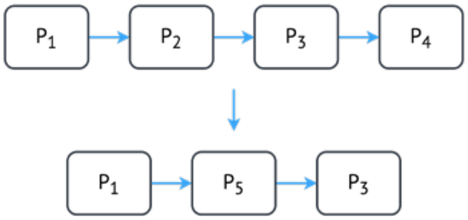


Figure 2. *Partition internal architecture*

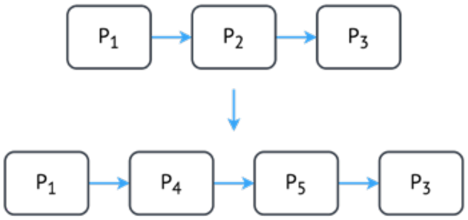


Figure 3. *Partition internal architecture*

Implementation

To ensure the ordering of events between partitions involved in adaptivity, we used Apache Kafka. Initially, Kafka was deployed standalone outside of a Kubernetes cluster. However, during an internship at Red Hat I discovered Strimzi, a Kubernetes operator that handles the deployment and management of Kafka resources. A Kubernetes operator (not to be confused with the StrIoT operator functions described above) is used to deploy and manage custom components within a Kubernetes cluster, typically automating the tasks that would be required of a cluster administrator. This allows the developer to submit custom resources to the cluster to be handled by the operator associated with the resource type. Strimzi simplifies the process of deploying Kafka brokers (alongside Apache Zookeeper) within the cluster. It can also be used to create, update, and delete KafkaTopic custom resources required for adaptation. The internship also prompted me to investigate the creation of a StrIoT Kubernetes operator that can handle the deployment of topologies and be used to trigger and manage adaptations.

As an example, the following code block represents a four-partition topology custom resource (this would look similar to the starting topology for fusion in Figure 2, with a Kafka topic as the connection channel between P_2 and P_3):

```
apiVersion: striot.org/v1alpha1
kind: Topology
metadata:
  name: example-topology
spec:
  partitions:
    - id: 0
      image: striot/striot-node-0:latest
      connectType:
        egress:  "TCP"
      edge: true
    - id: 1
      image: striot/striot-node-1:latest
      connectType:
        ingress: "TCP"
        egress:  "KAFKA"
      edge: true
    - id: 2
      image: striot/striot-node-2:latest
      connectType:
        ingress: "KAFKA"
        egress:  "TCP"
      edge: true
    - id: 3
      image: striot/striot-node-3:latest
      connectType:
        ingress: "TCP"
      edge: false
  order: [0,1,2,3]
```

The StrIoT Kubernetes operator handles the topology to create the deployments for each of the prebuilt partition images, providing environment variables for connection information. This allows the partition to read from the environment on startup and make connections to upstream and downstream partitions. In the case of Kafka connection types, the operator would interact with Strimzi to create a new KafkaTopic with a universally unique identifier (UUID) and pass the topic information to the relevant partitions. There is also an edge field, which the operator uses to apply NodeAffinity rules to ensure the container is placed on nodes within the cloud, or onto a small, low-powered edge device during experimentation.

The operator also triggers and manages the adaptivity itself. A management message is injected into the stream and passes through the partitions involved in adaptation. This triggers the partitions to acknowledge all messages up to this point in time and save any necessary state before safely shutting down. One or more new partitions are started and initialized with state in the case of stateful operator functions; they then resume from the same point in the stream. This results in a small pause in processing while transitioning to a more effective topology.

NEXT STEPS

We have evaluated this system using a real-world data set, but there are two future research directions. First, although we have implemented a set of adaptations, others should be investigated. While not discussed in this article, operator re-ordering is another option for adaptation. This requires prior knowledge of the processing within the operators and dependencies to ensure that application correctness is unaffected. A simple example is a map operation followed by a filter. If the filter is moved before the map, less data would need to be processed by the map. However, if the map modifies part of the data that the filter is dependent on, moving the filter first would result in different semantics. Second, for the purposes of evaluation, the adaptations discussed in this article were triggered after a predetermined period of time. By using real-time metrics of the processing performance, the StrIoT operator should determine the most beneficial adaptations and perform them as necessary. The cost models must consider the performance of the operator functions themselves, the available resources on the devices present in the streaming system, and the metrics outlining topology performance.

ⓘ

Acknowledgments:
This work is undertaken within the Newcastle University Cloud Computing for Big Data, EPSRC, Centre for Doctoral Training (CDT) scheme, supported by Red Hat. Credit goes to Prof. Paul Watson (Newcastle University, UK) and Jonathan Dowland (Red Hat).

Want to **change the cloud world** and directly **impact the next generation of students?**

MOC

The Mass Open Cloud (MOC) & our partners have various open positions.

For more details, please visit massopen.cloud/rhrqjob

Since 2014, Red Hat has been heavily involved in the MOC, a collaboration between the Commonwealth of Massachusetts, universities (Boston University, Northeastern, MIT, Harvard, and UMass), and industry to create an open production cloud. The project is going through major growth, with job openings and opportunities for industry engagement at both university and industry partners.

RESEARCH
QUARTERLY

VOLUME 4:1

Feature

RISC-V for FPGAs: benefits and opportunities

Why open source hardware will play a key role in emerging technologies research

by Ahmed Sanaullah

RISC-V Instruction Set Architecture (ISA)-based microarchitectures are an important part of all Field Programmable Gate Array (FPGA)-based research projects in the Red Hat Collaboratory at Boston University. Having CPU cores in FPGA designs is important: partitioning workloads between special purpose FPGA circuits and these general purpose cores allows for better functionality, flexibility, power consumption, and development turnaround than what could be achieved if entire designs were built using only special purpose FPGA circuits. In addition, RISC-V is our choice of ISA for these CPU cores because the ISA is open source, the toolchain is stable, and there is support for custom extensions.

FOUNDATIONS OF ARCHITECTURE

Before we get into detail, however, let's review four fundamental aspects of the computer architecture ecosystem:

- Microarchitectures
- Toolchains
- Runtime environments
- ISAs

A microarchitecture is the organization of the actual hardware of a processor. It specifies the various hardware blocks used, their parameters/sizes, and how these blocks connect to form the processor. Microarchitectures are typically developed as code using hardware description languages (HDLs). This design can be instantiated directly on the FPGA as a so-called softcore using chip resources or fabricated and made into physical CPU chips.

A toolchain is the software needed to compile code so that it can be executed on a processor. It reduces development complexity by allowing code to be written in high-level languages while still having direct control over the hardware details. The code itself could be both the user code and the system runtime.

A runtime environment defines the execution model for a given code by constraining how software and hardware interact. This environment can be as simple as compiling code using an embedded C library such as Newlib so that the code can interface hardware directly—effectively implementing a microcontroller unit (MCU). It can also be as complicated as an operating system (OS) with process scheduling. Hardware is interfaced (indirectly) through the OS, and thus requires linking to OS libraries during code compilation.

Finally, an ISA is the glue that holds the microarchitecture and toolchain together. It is an abstraction that sits at the boundary of hardware and software and provides guarantees to both—provided, of course, that both the toolchain and microarchitecture are compliant with the ISA. To microarchitectures, the ISA guarantees that any

About the Author
Ahmed Sanaullah is a Senior Data Scientist at Red Hat. His current focus is building open source tooling for FPGAs that enables developers to create custom hardware solutions easily and efficiently, regardless of prior hardware development expertise. He received his PhD in Computer Engineering from Boston University in 2019, winning the Outstanding CE Dissertation Award.

binary built by any compliant toolchain can be correctly executed. In the other direction, the ISA guarantees that any compliant microarchitecture will have the necessary hardware required to run any binary that the toolchain generates.

WHY RISC-V?

So how did we end up choosing the RISC-V ISA and its ecosystem? The first requirement was that the ISA needed to be open source. Having an open source solution is essential for enabling support for customizations to the softcore microarchitecture based on target workloads and constraints. As a result, we could not use ISAs such as ARM and x86.

The second requirement was that the ISA needed a full toolchain and a modern runtime environment. Both the toolchain and runtime need to be in active development. This excludes architectures whose tooling and runtime are outdated, such as SuperH(SH)-2 and SH-3.

Finally, the third requirement was that the ISA should provide support for easily utilizing microarchitecture extensions from code running on the software, for example, through the use of custom instructions. Given these requirements, RISC-V was the only modern ISA available and thus our default choice.

The RISC-V toolchain has a large, active community, which results in a complete toolchain. That toolchain, available as part of the community Linux distribution Fedora or in the Fedora-derived Extra Packages for Enterprise Linux (EPEL), is feature rich and easy to modify based on requirements. Several open source RISC-V softcores are available on the microarchitecture side, with a wide range of supported functionality and resource usage. This allows RISC-V compliant softcores to be instantiated on almost any FPGA, from large datacenter-class versions with millions of look-up tables (LUTs) and thousands of digital signal processors (DSPs) to small edge-class FPGAs with a few thousand LUTs.

The RISC-V ISA has support for easily and securely extending the ISA, which effectively allows us to interface with additional IP blocks. These custom extensions enable us to streamline the division of tasks between softcores and custom hardware blocks, and they facilitate the complex and time-sensitive interactions that need to happen between them. Since the microarchitecture and custom hardware blocks are both reconfigurable, the exact division of tasks can be varied substantially and tuned for specific designs or chip/board specifications.

WHY SOFTCORES?

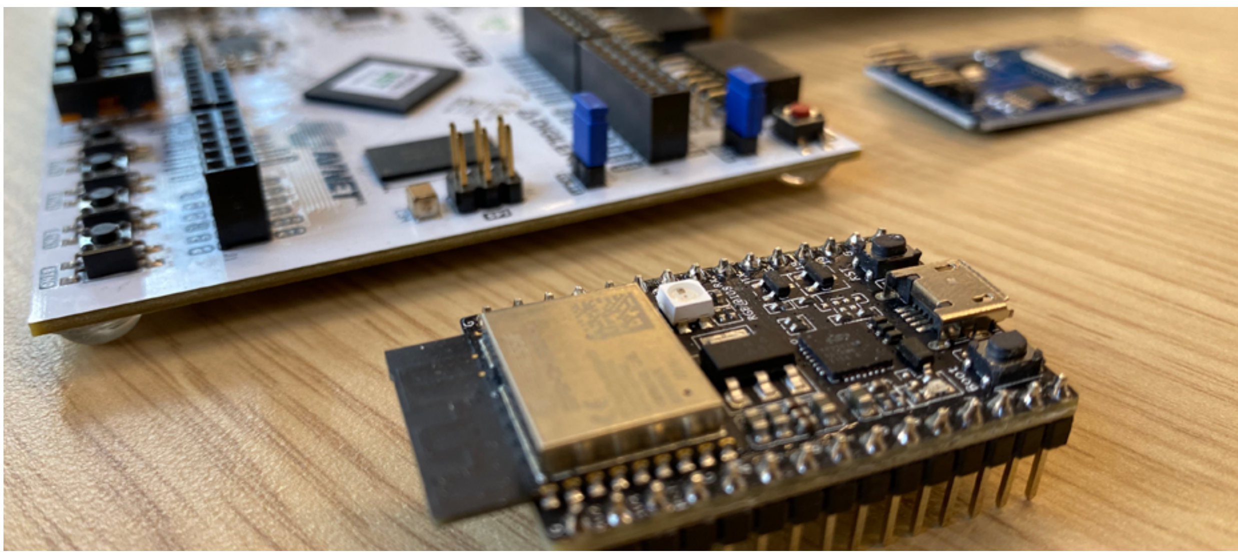
As noted previously, softcores are CPU cores that are coded using HDLs and instantiated on FPGAs using chip resources such as LUTs, memory blocks, DSPs, and interconnects. Because the softcore can vary in features and sizes, multiple softcores can be instantiated on a single FPGA fabric, limited by the total available FPGA resources.

FPGA-based softcores cannot achieve performance comparable with application-specific integrated circuits (ASICs) because of limitations placed by the FPGA fabric, such as a lower operating frequency, reduced complexity of the microarchitecture due to limited chip resources, and older processor technologies than the current state of the art (used in fabricated ASICs). However, these limitations are significant only for performance-critical tasks, such as application data plane operations. Our goal is to use softcores to implement tasks for operations that are not performance-critical, such as those on the application control plane.

Using softcores in this manner enables two key benefits. The first benefit is that developing, compiling, and maintaining high-level language code for softcores is typically easier than writing code for custom compute hardware using an HDL. If any functionality changes, we can easily modify the source code, generate the new binary, and load it onto the FPGA—a turnaround time in the order of seconds/minutes as opposed to hours/days for hardware generation.

The second benefit is that softcores take, on average, less of an FPGA's (limited) resources to do tasks. This is due to hardware reuse: instead of building custom hardware for each task, softcores leverage generic compute pipelines to execute multiple tasks sequentially. All softcore tasks reuse the same pipelines; thus, adding tasks incurs no additional resource costs beyond the memory needed to store new instructions. As a result, we free up resources that would otherwise have been used to implement custom hardware for non-performance-critical parts of the workload. These resources can then be used to scale up existing performance-critical hardware blocks or add more functionality.

Here's an example of how softcores can benefit FPGA design. Consider a host machine offloading match-action tables to an FPGA-based



The main ESP32 controller (front) with the FPGA (left), SD card reader shield (right) and a PMOD wire wrap board with a different ESP32 RISC-V microcontroller on it

Smart Network Interface Card (SmartNIC). In this use case, the FPGA would have to implement both data plane components—such as pipelines, buffers, and crossbars—and the control plane for setting up the different tables.

However, not everything put on the FPGA here requires highly specialized hardware to meet performance constraints. Specifically, we don't need to build custom state machines to maintain table entries. By replacing these state machines with a CPU core, we retain the same functionality, incur an additional cost in time, but end up likely reducing resource overhead. If additional resources are freed up, they can be used to scale up the data plane, potentially resulting in more performance for the parts of the offload that actually need it. Moreover, we could make minor changes to existing software-defined networking (SDN) code then compile it for the softcore, thus saving a substantial development overhead.

It is important to note, however, that while softcores are versatile, some functionality (such as connectivity, device control, etc.) cannot easily be done with the FPGA itself. An example of this is a full reconfiguration of the FPGA chip. Since the entire chip configuration is erased in this process, the reconfiguration has to be done by an external device. For such tasks, there are simple, cheap MCUs available in many forms and sizes that could be easily connected to an FPGA.

Consequently, since our system would need both softcore and MCU-based CPU cores, we envision using an MCU with the same ISA to make the system programming easier, because the same toolchain can be used. An example of this is the ESP32-C3 from Espressif, which is based on the RISC-V ISA.

Overall, the combination of the RISC-V ISA and FPGAs presents exciting opportunities for research and innovation. A developer can not only improve the quality of custom hardware they generate, but also take advantage of abstractions and design methodologies that make hardware development faster and more accessible for software developers.

To learn more about the history of RISC-V, read "Fostering open innovation in hardware" in RHRQ issue 2:2 (August 2020).

Feature

Look to the Horizon: Europe's increased focus on funding open source research is creating new opportunities

Three principal software engineers and sought-after research collaborators share their insights on this critical EU innovation incubator.

by Luis Tomás Bolívar, Carlos Camacho, and Josh Salomon

In February 2021, the European Union launched Horizon Europe, the next phase in its flagship Framework Programme for Research and Innovation. Horizon Europe, which will fund research from 2021 to 2027, was created to drive innovation, research, and development initiatives, emphasizing EU social, economic, and environmental challenges.

The program has been a significant source of support for researchers in open source technology.

As participants in this and other funding initiatives, we want to share with you why this program is worth learning more about—no matter where you are in the world—and how funded projects grow and, ideally, bear fruit. Our goal is to motivate you to explore funding possibilities for your own open source research project. We are Red Hatters, but we are also active members of the broader open source community, so the more people who use this information, the better.

Read "BigDataStack delivers with contributions from industry and university partners" in RHRQ issue 3:2 (Aug 2021).

WHY HORIZON EUROPE?

Horizon Europe's emphasis on open science and open source technology makes it an attractive funding opportunity for open source software engineers. The program evolved from Horizon 2020, which provided financial support for research projects that promoted industrial competitiveness, advanced scientific excellence, or solved social challenges. The BigDataStack project, for example, to which both Luis and Josh contributed, was funded by Horizon 2020 to deliver a complete collection of open and interoperable technology building blocks for a variety of big data stakeholders.

Horizon Europe refines Horizon 2020 goals and increases the available funds to €95.5 billion—nearly a sixty percent increase from the previous research program, if all funds are used. The new funding pillars are Excellent Science, Global Challenges and European Industrial Competitiveness, and Innovative Europe. This change in focus reflects the European Commission's interest in strengthening international cooperation and emphasizing breakthrough innovations—which makes the program a good fit for the collaborative work done in open source research communities. A dedicated European Innovation Council will support and fund those initiatives of a more disruptive nature that might be too risky for private investors.

The European Commission has also established a robust open science policy. Open science practices are part of the award criteria during the application process, and mandatory open access to publications is enforced and applied throughout the whole program. Participants are also encouraged to use the European Open Science Cloud where applicable.

The European Commission has also committed to increasing the use of open source technology by making it a key element for its digital strategy, reinforcing its finding that open source software and hardware have shown significant economic benefits for the European Union. This explicit change in strategy is recognition of the power of open source, pushing its principles and practices into broader acceptance.

THE VALUE OF COLLABORATION

Our previous work on Horizon projects has demonstrated the value of collaborating for the businesses that partner together and the universities they partner with. When collaborating with consortium partners, we leverage our open source approach to working by promoting the sharing and reuse of software assets, knowledge, and technical skills. This way of working acts as a catalyst for innovation because it fosters cross-pollination between the ideas of the multiple parties involved. Bringing new people together allows us to:

- combine unlike ideas and methods
- find new use cases for products that are already developed
- build a network of researchers and company partners
- create actual innovative software products in an agile way
- improve diversity in the team

—among many other benefits.

> This makes the program a good fit for the collaborative work done in open source research communities

In our current work, for instance, we apply our expertise in integration, automation, and working in the open to extend the lower layers of infrastructure to match the needs of the high-level systems deployed on top. Our understanding of Kubernetes and extensive experience with Red Hat® OpenShift® are also valuable strengths we bring to a project, especially since modern cloud-based projects are almost always based on Kubernetes.

This collaborative approach has also allowed us to connect with an ecosystem of organizations with similar goals. This includes academic partners such as the National Technical University of Athens (Greece), Newcastle University (UK), the Polytechnic University of Madrid (Spain), and the University of Piraeus (Greece), and consulting and service companies like Atos, in addition to multiple technology companies.

It is true that working across institutions and geographic divides creates a challenge in terms of communicating the goals and desired outcome of the projects effectively, but overall there are many more benefits for all the academic institutions and companies involved.

WHAT MAKES FOR A GOOD RESEARCH PROJECT?

From an EU perspective, a good research project is one that promotes excellent science that measurably advances the state of the art, makes a measurable impact on challenges that affect people worldwide, or produces innovations that will help the European Union act as a leader in emerging technologies. (For more information on how projects are evaluated, see the inset **What does a good proposal look like?**) But how do we, as engineers, determine what makes a research project a good fit for us?

We look for projects that fit the directions we want to take in our own work. For example, projects with one or more of the following elements are a good fit for us:

- Work on managing workloads on the edge
- Work that creates new open source projects
- Work that improves existing open source projects

A good project is one that helps us understand potential new use cases for the ecosystem of open source technologies we contribute to. Gaining a better understanding of state-of-the art research allows us to identify new requirements

WHAT DOES A PROPOSAL REQUIRE?

Many good proposals don't get funded, simply because of the funding granter's fixed budget. To be seriously considered for funding, a proposal should contain all of the following:

- A demonstration of how the proposal goes beyond the state of the art, with a statement of potential for clear outcomes and impacts, including a way to measure outcome and impact
- A gender equity plan that provides a clear roadmap to identify and assess any gender bias and strategies to correct it
- A high-quality and efficient plan of work, including clearly defined roles and tasks for each participant, based on their expertise
- A commitment to openness and contributing to open source communities (working in upstream communities is not required, but it will hugely increase the impact of the project)
- A collaboration plan indicating consortium members (depending on the call for proposals, requirements may include small or large companies, academic institutions, or research institutes)
- A plan for exploitation, dissemination, and standardization, including blog posts, webinars, conference talks, and other means of transmission

ⓘ *An EC webinar is available to view with more detailed explanations of these criteria, along with tools to help develop and document them.*

to enhance and optimize the technologies we work on. For example, the BigDataStack project mentioned earlier featured major performance improvements to the kuryr project, and it contributed the xskipper project to the open source community.

In 2021, we and our colleagues at Red Hat contributed to the submission of twelve Horizon Europe proposals, most of them at the invitation and encouragement of other academic or industry participants. This represents a huge milestone in our contributions.

Many of these proposals targeted the funding opportunity "Future European platforms for the edge: Meta operating systems." The topics for that call perfectly align with our current work related to cloud computing technologies and their applications to edge, automation, and management platforms. That, together with the EU push for open source and Red Hatters' previous good work on innovative open source contributions, led to the record number of proposals in which Red Hat engineers play an essential role in building out infrastructure layers and automation. For example, our past work as engineers on Orbit, Superfluidity, and BigDataStack led to renewed invitations to participate in follow-up projects.

WHAT HAPPENS WHEN A PROPOSAL IS ACCEPTED?

The follow-up process has a number of milestones to make sure the project continues in accordance with the work promised in the proposal. We submit reports and create deliverables, such as reports or software assets. The European Commission, in turn, provides audits and revisions as needed to keep the project on track. At the end of the project, the team delivers a final report that includes a technical report with the description of the project's working packages, the results, conclusions, and the socioeconomic impact. The report must also include a financial report and a dissemination and exploitation plan.

The dissemination and exploitation plan is one of the most important results of a project, from the Commission's perspective. Without dissemination, the project will have no impact! Work on this element of the project should start at the very beginning, keeping in mind that all of the parties involved in the project's consortium are responsible for disseminating results.

A critical factor for increasing adoption—and therefore impact—is to make the software components reusable. Ideally, these components will be used not only by consortium partners but also by a broader community, such as other projects, universities, or other EU companies. An open source model is one of the best ways to encourage reuse. That is especially true when we are not only building a new software component and making the code available, but also developing new research projects and contributing to upstream communities (e.g., the Linux kernel, OpenStack components, Kubernetes). For instance, a modification done at the Linux kernel level is immediately available for millions of users worldwide, which hugely increases the project's impact.

It's important to us to choose projects that our company will likely support long-term. While it's possible for a company to take EU grant money, complete a project, and then stop contributing after a project finishes, we want to know that the work will continue. Pushing for upstream acceptance also ensures that the project is aligned with open source community efforts and can be maintained after the project ends. In addition, research that corresponds to corporate strategic goals gives product engineers an opportunity to work on projects with a larger scope than their daily work allows. What we're doing might not always be on the roadmap right now, but if it fits with the directions our employer is interested in, there's a better chance that the technologies we're working on will make a lasting impact on the problems we're trying to solve.

WHAT DO YOU GET OUT OF IT?

We each appreciate different things about participating in collaborative projects like these. The opportunity to work on less mature, more speculative technology is a benefit, and so is the opportunity to develop topics we are already working on in a wider scope. For example, in the PHYSICS project that Josh and Luis currently work on, there was an opportunity to introduce cost minimization on public clouds by scheduling workloads to cheaper instances, and in many of the proposals still in evaluation by the Horizon committee there is an opportunity to develop the concept of storage at the edge.

Luis adds, "For me, it is a great opportunity to learn about new technologies that may be mainstream in the next few years. Nowadays, hybrid cloud and containers are the default, but I started working on those in research projects ten years ago." The same may happen with current work on FaaS for the PHYSICS project; perhaps it will be a new hot topic in a couple of years. It is also a great way to learn about possible needs and use cases, which allows us to evolve an upstream project to account for them.

Josh observes that working on these projects with Red Hat backing has given us a great opportunity to influence the project. Since Red Hat is considered the infrastructure leader in almost all the projects we've participated in, we can determine how the infrastructure behaves. "Also," Josh says, "as someone who does not have a PhD, I enjoy the opportunity to work with academic researchers and help combine academic research with working in industry, something that I missed during my studential era."

"For me, It is a great way to network with other colleagues and different organizations," Carlos adds. "I get the opportunity to contribute to state-of-the-art projects and participate in building a community between universities and technology companies."

To learn more about research projects supported by Red Hat Research and various granting agencies, and to find other ways to get involved, visit research.redhat.com.

About the Author
Luis Tomás Bolívar
is a principal software engineer working on the networking team and involved in European research projects.

About the Author
Carlos Camacho
is a principal software engineer in the OpenStack Platform department focused on upgrades and migrations. His research interests include formal methods to model software product lines, distributed systems, and cloud computing.

About the Author
Josh Salomon
is an architect in the Ceph storage team, with more than 30 years of experience in software development and architecture roles.

Column

Why you—yes, you—should take another look at Red Hat's Research Interest Groups

About the Author
Heidi Picher Dempsey is the US Research Director for Red Hat. She seeks out and grows projects with academic and commercial partners in areas such as operating systems, hybrid clouds, performance optimization, networking, AI, and security. As part of the Red Hat Research program, she encourages diverse participation in computer science and engineering research and promotes collaborations with Red Hat researchers.

Researchers, students, and software engineers all have something to gain and something to give when checking out research interest groups.

by Heidi Picher Dempsey

I was going through my coat pockets recently and found an old pair of Red Hat sunglasses. The plastic shade part of the sunglasses had popped out on one side, so if you put them on you got two very different views of the world at once.

The Research Interest Groups (RIGs) at Red Hat are like that. Within the framework of a regular discussion group, we are able to share many different views of technology and our place in that world, combining them into shared efforts to advance research and development around the globe. Sometimes in order to see more clearly, you need to shake up your usual way of looking at things, and the RIGs are a good place to do that.

The RIG idea started out a few years ago as a monthly meeting of Red Hat associates, students, and academic researchers in the Boston area, sharing updates on their existing projects and talking about ideas for new efforts they would like to start. Soon, we found there were many other researchers and Red Hatters who wanted to join the conversation, so we expanded to include researchers from all over the United States, changing the name (with characteristic Boston reserve) to the Greater New England RIG.

RIG meetings moved to open video conferencing, with topics that ranged from open FPGA programming demonstrations on how to detect fruit in live video, to using fuzzing to find OS software errors, to using AI in cloud operations. Students presented final research topics and theses. Researchers from US universities sought out collaborators, and Red Hat sponsored new academic projects encouraged by the RIG. Student projects to customize Fedora for daily student use and to experiment with building applications on Raspberry Pis were founded and became repeat campus offerings. Greater New England RIG meetings are open to all curious researchers, Red Hatters, and students, and are announced in the Events section of research.redhat.com. Presentations, notes, and videos from past meetings are also shared on the Greater New England RIG page of the Red Hat Research website.

Sharing ideas in a RIG went global in 2021 with the establishment of the Europe RIG. Born in the Covid era, the Europe RIG filled a need to bring together researchers from multiple countries around Europe and Israel to discuss research ideas and collaborations when there were very few opportunities to meet in person. The Europe

RIG investigates opportunities for joint proposals to EU research organizations and shares ongoing research projects in areas such as remote sensing, operating systems, security, and cloud computing. Although each RIG focuses on a particular region to keep the live discussion at reasonable times, all the RIG discussions are shared and incorporated into Red Hat Research's long-term planning. You can join the Europe RIG by contacting Martin Ukrop.

Each RIG fosters research that aligns with Red Hat's technical direction and open source mission while maintaining a distinct regional character. The RIGs also strengthen academic and research partnerships, such as the Red Hat Collaboratory at Boston University. They provide an opportunity for established researchers to introduce people with industry experience to their projects, and for students to share results from their own real-world research projects and contributions to upstream code development. Through internships and sponsored projects, students gain valuable experience that gives them a view of both academic and industry research communities and helps them find their place. Red Hat associates gain valuable new insights from researchers and students and a broader view of open source opportunities.

The Red Hat Research group welcomes your participation and new ideas for projects. We hope to see RIGs forming in new regions soon, and we look forward to a future so bright we gotta wear shades!

Research project updates

Each quarter, *Red Hat Research Quarterly* highlights new and ongoing research collaborations from around the world.

This quarter we highlight collaborative projects in Israel at The Technion, The Ben Gurion University of The Negev, Ariel University, Reichman University, and The Hebrew University.

PROJECT: SpotOS

ACADEMIC INVESTIGATOR:
Prof. Assaf Schuster (Technion)

RED HAT INVESTIGATOR:
Ilya Kolchinsky

The SpotOS project aims to devise a distributed cloud-based operating system that uses unreliable or temporarily available resources to provide a reliable and scalable execution experience with a high quality of service. The proposed system will achieve this vision by harnessing the power of spot instances, resources representing the currently unused cloud capacity. While spot instances are considerably cheaper than "regular" instances, they can be unexpectedly reclaimed by the cloud provider at any time. In this case, a very limited time window is given to the running application to back up its current state. SpotOS aims to overcome this limitation by providing a reliable, adaptive, self-healing, user-transparent layer with spot instances serving as the underlying unreliable building blocks.

The low-level design of SpotOS is complete. The next major step is implementing a prototype of the component representing the key innovation: the EDM (external distributed memory). With EDM, the application state is split among multiple storage units (that could be hosted on regular instances, spot instances, or a mix of both) in sufficiently small chunks to complete the evacuation on time. Designing and implementing the EDM will pave the way for the rest of the SpotOS framework.

PROJECT: Kubernetes optimized service discovery across clusters

ACADEMIC INVESTIGATOR:
Prof. Anat Bremler-Barr and Daniel Bachar (Reichman University)

RED HAT INVESTIGATOR:
Vishal Thapar

This research project aims to provide better and more balanced service discovery capabilities for Kubernetes multicluster deployments. Currently, the service discovery in this space is very basic. The project aims to investigate and assess different approaches to improve it by making it more balanced, reducing bottlenecks, and improving latency.

After several iterations on the design and implementation, code for simple round robin and weighted round robin is now available as part of the Lighthouse plugin in Submariner. Details on the design are available on GitHub. This project was also featured in the article "Optimizing Kubernetes service selection" in RHRQ 3:4 (February 2022).

PROJECT: CCO: cloud cost optimizer

ACADEMIC INVESTIGATOR:
Prof. Assaf Schuster (Technion)

RED HAT INVESTIGATOR:
Ilya Kolchinsky

Cost optimization is one of the core challenges for users of cloud computing platforms: given a workload, how can we minimize the monetary cost of its deployment and execution over the cloud? Accurately answering this fundamental question for arbitrarily complex real-life workloads turns out to be an exceedingly hard problem. The goal of this project is to develop a tool, which we call the cloud cost optimizer (CCO), capable of doing exactly that. CCO retrieves information regarding the prices and the capacities of all instance types, regions, availability zones, and so forth as advertised by the cloud provider. Based on this information, CCO calculates the optimal allocation of workload components to minimize the overall cost.

An early alpha version of the optimizer, limited to AWS and only supporting applications with a small-to-medium number of distinct components, is currently available for use. Our ongoing efforts focus on several areas, most importantly:

- Lifting the above limitations by introducing more sophisticated optimization algorithms and techniques
- Extending CCO to take into account additional operational aspects, such as performance and reliability, during the decision-making process
- Introducing a machine learning component for predicting future instance prices to improve the accuracy of the optimization process

PROJECT: OpenCEP: an advanced open source complex event processing engine

ACADEMIC INVESTIGATOR:
Prof. Assaf Schuster (Technion)

RED HAT INVESTIGATOR:
Ilya Kolchinsky

In RHRQ 2:1, investigators described their plan to build a scalable, real-time, cloud-based CEP engine capable of efficiently detecting arbitrarily complex patterns in high-volume data streams. The engine was designed to be implemented on top of the Red Hat® OpenShift® Container Platform and to be applicable to any domain where event-based streaming data is present. Researchers aim to advance the state of the art in complex event processing and combine academic research with the implementation and deployment of novel CEP mechanisms and techniques.

The beta version of OpenCEP is currently available for use. The project team is now mainly tasked with solving the open issues and bugs, addressing user feedback, introducing minor features and modifications, and improving the project documentation.

PROJECT: Advanced proactive caching for heterogeneous storage systems

ACADEMIC INVESTIGATORS:
Dr. Gabriel Scalosub and Dr. Gil Einziger (Ben Gurion University of The Negev)

RED HAT INVESTIGATORS:
Josh Salomon, Gabriel BenHanokh, Orit Wasserman, and Mark Kogan

Caching is a classic optimization technique ubiquitous both within single hosts and in large distributed systems. While the cache algorithms currently employed in real-life systems are highly efficient and yield significant performance improvements, their reactive behavior severely limits their ability to exploit various naturally arising patterns.

This project aims to overcome this limitation by developing a novel caching framework that performs proactive caching decisions, considers the various heterogeneous characteristics of the workload, and exploits recurring access patterns to speculatively prefetch items into the cache. In addition to the purely algorithmic contribution, we intend to pursue upstreaming our suggested solutions within the open source community, most notably within the Ceph architecture, for example, by extending the D4N framework.

The project's research phase is underway, with many promising approaches, insights, and results available. According to the current timeline estimations, the first prototype version of the proactive caching mechanism will be available mid-2023.

PROJECT: PHYSICS: oPtimized HYbrid Space-time servIce Continuum in faaS

RED HAT INVESTIGATORS:
Luis Tomas Bolivar and Josh Salomon

The PHYSICS project started in January 2021, with funding from the Horizon Europe research and innovation program and participation from fourteen international partners. It is scheduled to end in December 2023. The goal of PHYSICS is to unlock the potential of the Function-as-a-Service (FaaS) paradigm for cloud service providers and application developers. Specifically, it will enable application developers to design, implement, and deploy advanced FaaS applications in the scope of advanced cloud application design environments, leveraging proven design patterns and existing libraries of cloud/FaaS components. PHYSICS will offer a novel Global Continuum Layer that will deploy functions to optimize multiple application objectives simultaneously, such as performance, latency, and cost. PHYSICS will validate the benefits of its Global Continuum Layer and tools in the scope of user-driven application scenarios in three important sectors: healthcare, agriculture, and industry. Visit physics-faas.eu for more information.

The main architecture and components have already been defined, as well as the first prototype of them. The main architecture is based on several projects Red Hat is working on, such as OCM and Submariner for multicluster environments, MicroShift for the edge, and OperatorSDK for building the needed operators to glue different components together. Current work is focused on integrating the various components towards a first end-to-end demonstration.

PROJECT: Ceph: Wire-level compression: efficient object storage daemon communication for the cloud

ACADEMIC INVESTIGATORS:
Prof. Anat Bremler-Barr and Maya Gilad (Reichman University)

RED HAT INVESTIGATOR:
Josh Salomon

This project's purpose is to reduce storage network traffic (object, block, etc.) for the following cases: between the failure domains in cost-sensitive environments such as public clouds, and between nodes in cases where the network bandwidth is the bottleneck of the node performance.

The project was completed successfully. The code was merged to Ceph version 17 (Quincy) and will be part of RHCS v6 and ODF versions based on RHCS 6 (probably ODF 12 and on). More information can be found in the pull request.

PROJECT: Secured API in hybrid cloud

ACADEMIC INVESTIGATOR:
Dr. Amit Davir (Ariel University)

RED HAT INVESTIGATORS:
Avihu Goren and Noa Haim, mentored by Anna Sandler and Luiza Nacshon

The project was submitted successfully last year and has attracted much interest at the university and in the Red Hat products security group. That project led to creating a RapidPT tool for testing API security.

A new student has taken the lead in continuing last year's project, and we are now focusing on improving the PT tool to support a broader range of APIs and attack vectors. We plan to implement a user-friendly dashboard and a Firebase database to store results and user data in a more optimal way.

Contact academic@redhat.com for more information on any project described here, or explore more research projects at research.redhat.com.


intel AI
AI ON INTEL®
intel XEON® PLATINUM inside™
NOW BUILD THE AI YOU WANT ON THE CPU YOU KNOW.
Learn more at ai.intel.com
© Intel Corporation. All rights reserved. Intel, the Intel logo, Xeon and other Intel marks are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Copyright © Intel Corporation 2020.