Project Summary/Abstract

Project Title: **Coupling Sensor Networks and HPC Facilities with Advanced Wireless Networks for Near-Real-Time Simulation of Digital Agriculture**
Shantenu Jha, Brookhaven National Laboratory (Principal Investigator)
Rich Wolski, University of California, Santa Barbara (Co-PI)
Douglas Thain, University of Notre Dame (Co-PI)
Mehmet Can Vuran, University of Nebraska-Lincoln (Co-PI)

Combining large-scale computing capabilities with scientific instruments and sensors (of all scales) to function as a single system has become an essential requirement for new science. Until recently, the physical and logical distance between the sensor networks and high-performance computing (HPC) facilities prevented effective connections between them due to a lack of low-latency, high-throughput, predictable communication. **We hypothesize that 5G/6G networks offer novel capabilities that can be exploited and extended to provide the low latency, high throughput, and reliability needed to perform the near-real-time coupling of edge sensor networks with simulations running in HPC facilities.**

To evaluate the proposed hypothesis, we will research and deliver software systems that will: 1) enable seamless processing of data from sensors in an Internet of Things (IoT) context with DOE leadership-class computers, 2) leverage and extend emerging 5G/6G network technologies via novel sensor slicing technology to virtualize data from both physical sensor network(s) and existing data sets, and 3) effectively compose data from the edge with data on HPC platforms. Our driving applications are from digital agriculture, a DOE priority area, for example, coupling real-time simulation of pesticide application using distributed environmental sensors communicating over a 5G/6G network to seed and update a high-fidelity atmospheric simulations on HPC.

Currently, no software effort or programming framework considers the multiscale nature of computing and devices (small edge sensors to large computing platforms). No single scale of a computing element, storage device, or sensor is sufficient. In addition, the requirements for unifying multiscale resources mandate that the system software adapts to changing conditions, failures, etc. The forthcoming advanced wireless technologies comprising 5G/6G infrastructure must include sufficient computing and storage capabilities, e.g., for low-latency decision-making, actuation, and control at the edge, particularly in remote areas that cannot be monitored or served by near-field radios. Current 5G/6G designs include "network slicing" features that enable per-flow Service Level Agreements (SLAs). However, these network-slicing technologies do not allow virtualizing end users (i.e., IoT devices). Furthermore, no 5G/6G slicing technologies or approaches are specifically architected to enable HPC coupled with real-time or near-real-time data acquisition. Lastly, large-scale batch systems, highly tuned for throughput, are not engineered to remain responsive and available in the face of fluctuating remote device and sensing availability.

Advanced wireless networks will make near-real-time synchronization of sensors and simulations possible. Even so, unexpected network events may still occur, and the system design must accommodate asynchronous operation and resynchronization as conditions dictate. We will advance and enable "end-to-end" services to create a new coupled computing environment for the computing continuum.

Our overarching research objective is to develop the software and middleware that couple computing, storage, sensing, and actuation at multiple resource scales. Such system research will be effective only if it unifies across resources and throughout the software stack. Thus, the second research objective is to investigate the software system comprehensively and end-to-end rather than as an integration of separate technologies, each developed for a different purpose. A virtualized sensor-slicing solution will be developed so the same physical sensor network(s) can provide different sensing processes without needing separate deployments. The sensor slicing concept extends the emerging 5G network slicing solutions to the individual sensing elements at the network level. Accordingly, a physical sensor can be virtualized to provide multiple

roles in each feedback loop. This innovation will seamlessly integrate *in situ* collected and locally stored data and effectively virtualize data from physical sensor network(s) and existing data sets.

Developing the end-to-end software infrastructure necessary to couple large-scale computing capabilities at one end with field-deployed sensing and actuation components at the other (while leveraging intermediary resources in between) will enable new "on-demand" applications that put HPC in the loop for critical decision-making. This multiscale, coupled infrastructure is essential for developing new applications that meet nationally critical priorities, such as biosurveillance of the national biofuel crop.

Project Title: **Coupling Sensor Networks and HPC Facilities with Advanced Wireless Networks for Near-Real-Time Simulation of Digital Agriculture**

Applicant Institute: Brookhaven National Laboratory, Upton, NY

Street Address/City/State/ZIP: Building 725, Upton, NY 11973

Lead PI: Shantenu Jha, Chair, Computation and Data-Driven Discovery (C3D)
Brookhaven National Laboratory
(631) 344-4780, shatenu@bnl.gov

Administrative Contact: Nicole Medaglia, 631-344-6245 medaglia@bnl.gov

FOA Number: DOE-FOA-0003300

DOE/SC Program Office: Office of Advanced Scientific Computing Research

DOE/SC Program Office Technical Contact: Robinson Pino Robinson.Pino@science.doe.gov

PAMS Preproposal tracking number: PRE-0000038808

Research area: Topic D) Advanced Wireless

1 Introduction

Increasingly, the ability to couple large-scale computing capabilities with scientific instruments and sensors (of all scales) so that they can function together as a single system has emerged as a key requirement for new science and for new scientific impact. Mitigating the effects of climate change on agriculture and ensuring U.S. energy independence both require the modeling of, and responses to, dynamically changing physical phenomena (e.g. pathogen virulity, propagation and their dependence on external conditions, etc.) that are difficult to predict. While predictions of the relevant phenomena will improve, key to this improvement is the ability to study such phenomena at ever smaller time scales in as close to real time as possible.

To bring about the necessary technological advances that will enable this next generation of science requires new computer systems research that must advance computational capabilities along two dimensions. The future technologies must be

1. *Multiscale, Multimodal, and Resilient*: they must be able to couple and decouple computing and storage resources with instruments, sensors and actuators at all scales. No one scale of computing element, storage device, or sensor is sufficient. Further, the necessary computational and storage capabilities must be designed for long-lived, unattended operation in the face of changing conditions. This resilience must be a fundamental design requirement that unifies all levels of the system software stack at all resource scales.

2. *Adaptive*: The requirements for both unifying multiscale and multimodal resources mandate that the system software adapt to changing conditions, capabilities, usage modalities/QoS constraints, failures. This necessitates the importance of agile and adaptive resource management and workload execution properties.

In this work, *we hypothesize that 5G/6G networks offer novel capabilities that can be exploited and extended to provide the low latency, high throughput, and reliability needed to perform the near-real-time coupling of edge sensor networks with simulations running in HPC facilities.* More specifically, the coupling of large-scale systems with scientific instruments, sensors, and actuators at all scales requires a new approach to adaptive workflow management and new system software abstractions. In particular, this coupling requires that components describe and expose performance and resilience properties in a way that new scheduling and software infrastructure can and must exploit though novel scheduling algorithms and distributed system software.

We propose to investigate this hypothesis in the form of XGFABRIC – an end-to-end, multi-scale, and adaptive distributed system that couples leadership class resources with scientific instruments, sensors, and actuators through a tiered network of supporting computational and storage resources. Central to XGFABRIC are new workload partitioning and scheduling approaches that enable scale, adaptivity, and resilience for applications using coupled and multiscale infrastructure.

Our research approach will be empirical and artifacts-based. We plan to develop XGFABRIC as a evolving prototype that permits continuous evaluation of the research results this project will generate. We will also use "real-world" applications focused on digital agriculture as driving applications. W will deploy XGFABRIC in the field using sensing, edge, and high-performance computing resources as part of our efforts to evaluate its effectiveness.

Multiscale, Multimodal Systems

The state-of-the-art computational science focused on digital agriculture, its dependence on climate change and its relationship to energy independence (e.g. carbon neutrality, renewable energy sources, etc.) requires the most powerful leadership class machines that are available. However, because the phenomena that are critical to understand and predict are changing more dynamically, the resolution with which these phenomena must be measured and modeled is becoming ever finer.

While the need for high-resolution computational science is rightfully associated with the twin needs for data and computational scale, when it is driven by increasing dynamics, scale alone is insufficient. For example, in a disease propagation response context, the speed with which the data can be acquired, the resilience of the infrastructure (hardware and software) with respect to the pathogen itself, and the ability to

adapt computation and data management to change conditions emerge are essential properties that must be part of the overall system that is responsible for delivering data and computational scale.

At present, no software effort takes these requirements to be essential in this computational context. The cloud, for example, provides no resilience guarantees (or even service level objectives) for edge devices or sensors in an "Internet-of-Things (IoT)" context. The coming cellular technologies that comprise 5G and 6G infrastructure do not include sufficient computing and storage capabilities, particularly in remote areas that must be monitored but cannot be served by near-field radios. Large-scale batch systems, highly tuned for throughput, are not engineered to remain responsive and available "on-line" in the face of changing phenomenological dynamics and fluctuating remote device and sensing availability. Our work will advance and enable resilience "end-to-end" to create a new coupled computing environment for multi-scale, dynamic, and "streaming" science applications.

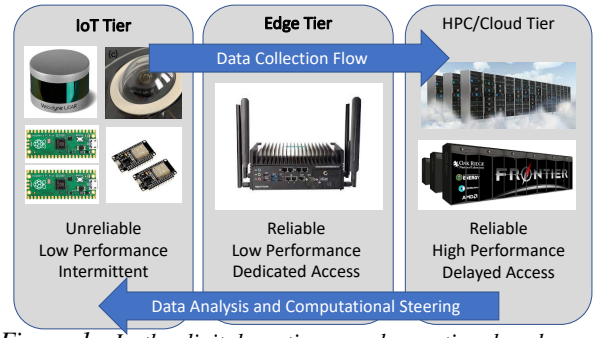

Figure 1: In the digital continuum, observational and sensing data flows from the edge to the center. Processed data and updates to AI/ML models flow from the leadership class platform to the edge and even to the field. (Adapted from DOE 5G Enabled Energy Innovation Workshop Report [5])

The coming advances in 5G and 6G connectivity will make it possible to build and maintain multi-scale, networked computational infrastructure that is capable of interfacing the smallest of devices at the edge with a spectrum of larger resources up through, and including, the cloud and leadership-class HPC systems. These computational conglomerates are *multi-scale* because their constituent components span several orders of magnitude in computational and storage capability. Devices at the extreme edge of the network that interface with low-power, highly durable sensors and actuatores may only support 10s of kilobyes of memory with single-core CPUs clocked at 10s of megahertz. These devices must interfaces to successvely larger small board computers (e.g. low-power Linux machines), edge-clouds [8], regional and private clouds, and (at the largest end of the scale), public clouds and HPC resources. These systems are also *multi-modal* in that the function of all of the resources (including device and device controllers) must be able to change in response to the needs of the application components they host. For example, a sensor controller interfacing with a physical sensor may need to also host (temporarily or permanently) telemetry "filtering" software to reduce the network message payload size and thereby conserve battery power.

The need to provision computational resources at multiple scales and to manage the modes in which these resources operate (particularly at the edge) gives rise to a new model for distributed systems that couples phsyical sensing and actuation with amalgamations of 5G and 6G interconnected resources. To leverage 5G and 6G technologies for multi-scale, multi-model systems requires a *digital-physical fabric* – a computational infrastructure in which the resources comprised by the system are *interwoven* by a common and unifying software infrastructure. Digital-physical fabrics are distinct from "digital twins" [32] and cyberphyical systems [4] in that they do not separate and categorize the resources as either digital or physical. These earlier paradigms focus on managing the interface between physical "real-world" interactions and computational capabilities – the digital and the physical are separate but equal. Alternatively, a digital-physical fabric blurs these distinctions through its software infrastructure so that the system functions as architected, end-to-end, from the physical interactions through final data analysis. In particular, it manages both the spectrum of scales that comprise the end-to-end system and the panopli or modes that the resources may assume. Indeed, digital twins and cyberinfrastructure can be important components of a digital-physical fabric – we describe the incorporation of a digital twin for biosurveillance and biosecrity in Section 2.1.

Our work described in this proposal will explore the development of digital-physical fabrics using 5G and 6G technologies as the digital fabric that makes a unified and durable end-to-end system possible.

2 Science Drivers for Advanced Wireless for Digital Agriculture

2.1 SD1: Citrus Under Protective Screens

The citrus production industry is currently developing remediation strategies for the Asian citrus psyllid which carries the huanglongbing (HLB) "citrus greening" disease. HLB has devastated the commercial citrus industry in Florida and Texas with an annual cost of more than $1B US [24]. From a biosafety perspective, HLB is a significant vector. Pesticides and disease-resistant cultivars have, so far, proved ineffective. Its effect on citrus production in the south has been rapid and irreversible.

In California, where the disease is present but not yet epidemic, growers are experimenting with siting orchards inside large, protective screen houses. The Citrus Under Protective Screening (CUPS) project is an at-scale pilot for screen-house citrus production located at the Lindcove Research Extension Center in Exeter, California. While CUPS is specifically testing HLB control, it represents an approach that is effective against any insect-born pathogen for which typical husbandry practices are ineffective.

The goal of CUPS is to understand the growing environment and commercial agricultural viability of screen-house citrus production. Citrus trees have useful production lifetimes that exceed 20 years. CUPS is effective as long as the trees that are introduced into the screen house are disease free and the screen remains in tact. For commercial viability, the screen houses must be large (covering several acres each) and they must accommodate tree canopy and harvesting equipment that require 25 to 30 feet of vertical space.

Detecting and rapidly repairing screen breeches in commercial scale CUPS is a critical open problem. While industrial accidents that cause screen damage can be detected and rapidly reported by workers, unobserved events (e.g. bird strike, foraging fauna, damage concomitant with theft, etc.) can cause screen breeches that must be detected.

Our team has been working to instrument and analyze the growing environment within the at-scale CUPS structure in Exeter. As part of that on-going work, we have developed a Computational Fluid Dynamics (CFD) model that can model to predict airflow within a CUPS screen house in near real time based on instantaneous wind, temperature, and humidity measurements taken and the screen boundaries (both inside and outside). Analytically, the goal of the model is to provide growers with decision support for input events such as pesticide or fertilizer spraying, frost prevention, etc. where the grower must make a decision regarding timing, location, and quantity of input to apply.

However, we are also exploring whether the model can detect screen breech. Specifically, once the model is calibrated, a deviation between predicted and measured airflow can portend a possible screen breech and, perhaps, an area of the structure where the breech may have occurred.

Note that we plan to structure the coupling of real-time senor data with CFD as a "digital twin" in which the true atmospheric conditions within the structure are "twinned" by the results of the CFD model for the interior of the structure. The model results will inform both modality changes in the sensing infrastructure and data calibrations (back tested against historical data) that are necessary to maintain model accuracy.

Our team will also be deploying a Farm-NG [13] wheeled robot with autonomous-driving capability within the CUPS structure. As a driver for XGFABRIC research, our plan is to investigate whether it will be possible to detect a potential breech (using a large-scale HPC machine to run the CFD model which is parameterized by real-time *in situ* boundary conditions), compare the modeled airflow to measurements taken for the same time period within the structure, and if they do not match, dispatch the robot to surveil the region of the screen where a breech may have occurred using an on-board camera. The XGFABRIC digital-physical fabric will incorporate robot-based sensing and robot route planning, thereby linking it to, and augmenting the CFD-based digital twin for the screen structure.

This ambitious application illustrates how a digital-physical fabric can enable new biosecurity capabilities. However, to bring it to fruition requires the ability to amalgamate computational resources at all scales, and to "close the loop" between sensing, computing and storage, and actuation.

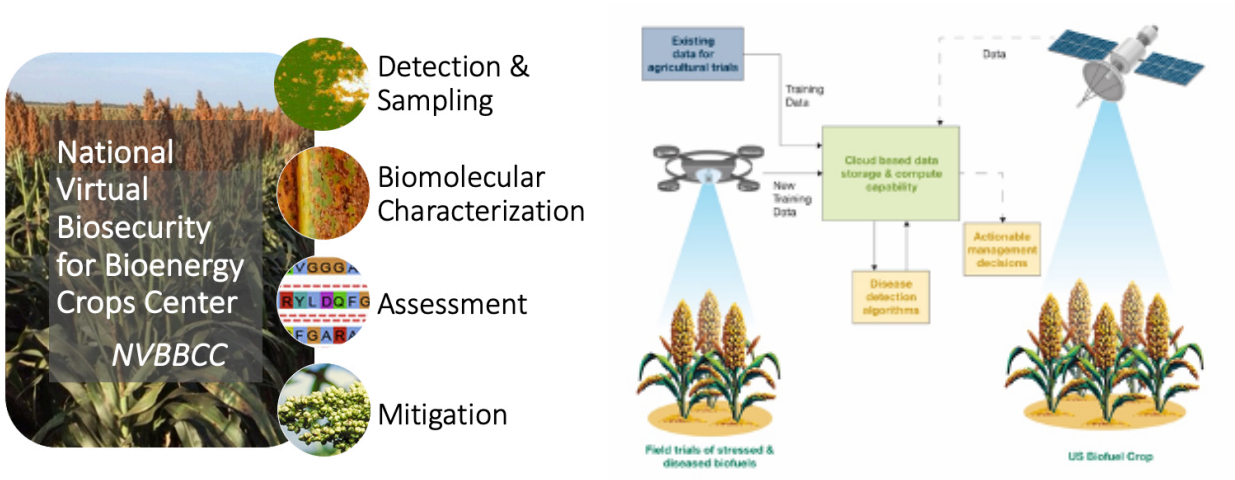

Figure 2: The National Virtual Biosecurity for Bioenergy Crops Center (led by Brookhaven National Laboratory) is tasked with designing the integrated infrastructure to support the collection of field data, detection of pathogens, and sampling of bioenergy crops with traditional simulation-modeling capabilities.

2.2 SD2: Biosurveillance and Biosecurity of Bioenergy Crops

The US and other nations have set net-zero carbon goals, which require the development of comprehensive and scalable technologies to replace fossil-carbon energy sources and remove atmospheric carbon through net-negative emission technologies. Expanding the US Bioeconomy is part of the US Government's "all of the above" approach to meet its net-zero goals (US Long-term Climate Strategy, 2021). The US aims to build the US bioeconomy, which includes enhancing biosafety and biosecurity to reduce risks to the US bioeconomy. In an expanded bioeconomy, a significant fraction of the agricultural sector will be devoted to growing bioenergy crops to support the production of biofuels and bioproducts. The emergence and spread of diseases among bioenergy crops could significantly impact yields and destabilize the US bioeconomy. However, with the projected expansion of bioenergy crops, the risks to crop health associated with the emergence of new diseases, the climate-driven spread of known diseases into new regions, and the accidental or deliberate release of plant diseases need to be considered. In this context, based on experimental and computational facilities, new capabilities are needed to detect, characterize, model, and mitigate biothreats to bioenergy crops.

There are unique challenges in handling extensive remote agricultural field data in real-time streaming analysis with a performance guarantee. Notably, it requires: 1) reliable network connectivity to transmit large volumes of field phenotype data (i.e., drone flying with hyperspectral imaging), 2) intelligence surveillance planning (i.e., optimal scanning strategies), and 3) efficient real-time streaming processing at the edge. (2) and (3) often must be coupled to traditional modeling and simulation capabilities.

These workflow challenges pose an opportunity to incorporate and improve 5G or upcoming 6G wireless communication to enable (1) high-throughput transmission with low latency and large coverage area, (2) autonomous data curation/collection using AI or OED (Optimal Experimental Design), and (3) real-time edge processing capabilities.

Advanced software services and analytics must address additional challenges to play an essential role in the biosecurity of bioenergy crops. These include model validation, sparse datasets that cause ML training difficulties, and integrating data from different modalities to create interoperable datasets. Data processing capabilities at the edge must be aligned with traditional simulation and modeling capabilities to create a unified, integrated modeling approach for bioenergy crop security.

2.3 SD3: Real-time Soil Monitoring and Irrigation Automation

Eighty percent of the 24 million Americans without high-speed Internet live in rural areas [3] and farm to feed the world's population. The world population is expected to reach approximately 9.1 billion people by the year 2050 [34], which requires a 70% increase in food production with minimal adverse environmental impacts. Rural broadband connectivity, both on-farm and off-farm, is key to a digital agricultural infrastructure to improve U.S. food security [17].

Despite connectivity challenges, soil moisture measurements have informed irrigation decisions for decades. However, real-time measurement and automated irrigation solutions still face challenges. The challenges include a lack of robust rural wireless coverage and difficulty installing and removing sensors before and after a growing season. The need for real-time in-situ information from large and diverse agricultural fields has given rise to ***A*gricultural Internet of Things (Ag-IoT)**, which could provide in-situ monitoring capabilities (e.g., soil moisture, temperature, electrical conductivity) and, when interconnected with existing field machinery (seeders, irrigation systems, sprayers, combines), enable field autonomy. Field autonomy through autonomous irrigation scheduling, variable rate irrigation to water different parts of the field with different amounts of water, and autonomous fertigation, will pave the way for better food production solutions [39]. One of the biggest challenges facing precision agriculture is getting information to growers and enabling them to take action. To this end, there is a growing need for in-field sensors, aerial or satellite imagery, communication devices, cloud-based analytics, remote monitoring, and real-time control of in-field equipment. A high-bandwidth, wide-area network with good penetration of foliage, buildings, and terrain is needed to address the challenges of limited or nonexistent coverage in rural areas.

Timely and high-resolution insights into soil characteristics are challenging due to the difficulty of seasonal installation and removal of soil sensor systems, along with the lack of cost-effective and high-performing data acquisition. Coupled with the recent, reliable, high bandwidth wireless connectivity in rural communities, these systems will contribute significantly to the adoption of technology in production.

The interplay between digital agriculture and rural broadband connectivity is best exemplified in existing scientific and commercial practices. For example, at academic labs (e.g., UNL's East Nebraska Research Education and Extension Center (ENREEC) in Mead, Nebraska), scientists and researchers utilize advanced sensing equipment (e.g., hyperspectral sensors mounted on a spidercam structure or unmanned aerial vehicle (UAV)-mounted high-resolution cameras) to collect large volumes of data (Fig. 3). However, this data is only stored in an SD card and manually transferred to a data center for post-processing. The lack of rural broadband connectivity prevents real-time access to these data and stifles scientific understanding and progress. There is a need to transform emerging 5G technology verticals (e.g., network slicing) for agricultural requirements such that unique characteristics of agricultural fields (e.g., limited wireless infrastructure and coverage, unique built infrastructure, unique wireless propagation characteristics due to dynamic changes in crop canopy height and field operations) are leveraged.

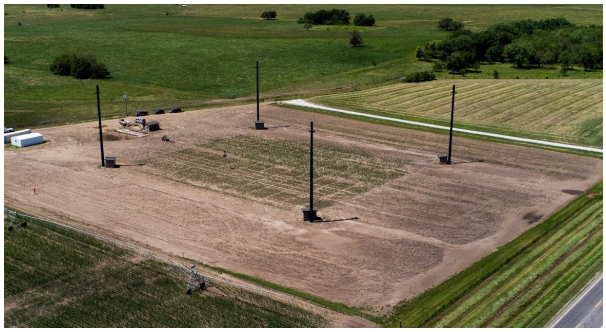

Figure 3: UNL Spidercam field phenotyping facility.

3 Work Track 1: Digital-Physical Systems

The innovations in terms of connectivity that 5G and 6G bring militate for a new approach to closing the loop between sensing, computation and storage, and actuation. We term the unified hardware and software system that allows computational and storage resources at all scales to act in a coordinated way a *digital-physical* system. Ubiquitous, high-quality, and provisionable connectivity makes it possible to consider digital and physical interactions within the same system architecture.

With Track 1, we will develop a multi-scale, distributed digital-physical fabric to unify components in an end-to-end deployment. We will investigate a "full-stack" approach, which consists of language-level, runtime-system middleware, operating systems-level, and networking innovations, to unify a heterogeneous, distributed, and multi-scale set of resources as a tiered end-to-end system. Note that our approach will combine fully native "bare metal" deployments, with middleware that can function as part of an existing software ecosystem. SensorSlicer (Section 3.1) will provide QoS-enabled provisioning capabilities for end-devices that the "FabriStack" (Section 3.2) will unify to create an end-to-end application platform.

3.1 SensorSlicer: End-to-end Network/IoT Slicing for Virtual Data Collection – Lead: UNL

We will leverage and extend emerging 5G/6G network technologies via novel sensor slicing technology to virtualize data from both physical sensor network(s) and existing data sets.

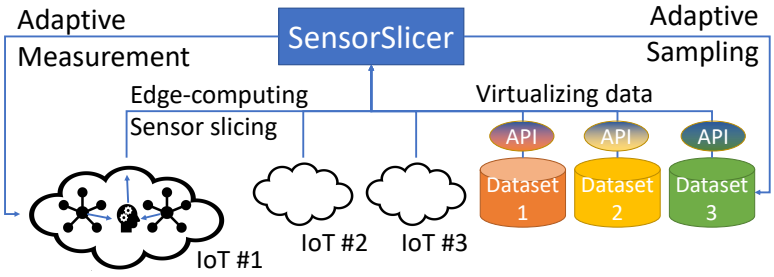

Figure 4: SensorSlicer: Virtualizing sensors in IoT networks and datasets for cost-effective data collection and adaptive measurement.

Research Challenge: The forthcoming advanced wireless technologies comprising 5G/6G infrastructure must include sufficient computing and storage capabilities, e.g., for low-latency decision-making, actuation, and control at the edge, particularly in remote areas that cannot be monitored or served by cloud-based core sites. The envisioned suite of science drivers (Section 2) emphasizes the crucial need for robust, reliable connectivity solutions in farms. More specifically, swarm robotics for efficient citrus farm management necessitates seamless, reliable, low-latency connectivity for real-time coordination and control. Similarly, hyperspectral images from drones and real-time surveillance data flow require high-capacity data transfer and low-latency connectivity for effective coupling with modeling and simulation tools. Moreover, emerging Ag-IoT networks with numerous end-points require field-level coverage with high-density and energy-efficient connectivity. Integrating such diverse use cases with specific per-flow connectivity requirements, highlights ***the central role that advanced network infrastructure plays*** in realizing the potential of HPC with scientific instruments and sensors.

Furthermore, predominantly in rural areas, farm fields suffer from limited infrastructure. Integrating diverse use cases in a single farm is cost-prohibitive with existing approaches. More specifically, current solutions require either multiple sensor networks to be deployed for multiple use cases, increasing the cost per use case; or sharing the data collected by a sensor network at the cloud end, which risks privacy. To this end, virtualizing data from both physical sensor network(s) and existing datasets is a major challenge.

Network Research: Current 5G/6G designs include ***network slicing*** features that enable per-flow resource allocation such that networking, computation, and storage elements can be assigned to each type of traffic flow in a more granular fashion. Thus, we virtualize a single physical network infrastructure using multiple network entities tailored to each type of traffic flow (e.g., high throughput, low latency, high density). Network slices are structured based on Service Level Agreements (SLAs), according to which resources are allocated to an end-point to meet SLA requirements. However, emerging 5G network-slicing technologies do not allow virtualizing end users (i.e., IoT devices) because they are primarily geared towards general-purpose networks, where the network infrastructure provider do not have access to end-point devices (e.g., cellphones). However, further virtualization of the IoT end devices is possible in more specific, purpose-driven networks, such as agricultural IoT. Accordingly, ***the same physical sensor device could be virtualized to act as different sensors for different use cases***, significantly decreasing deployment costs.

SensorSlicer abstraction along with 5G network slicing technology (Fig. 4) provide ***several advantages for seamless integration of in-situ measurements, established datasets, and scientific computing***: (1) The

same physical sensor network deployment could be utilized for different workflows, (2) Workflow tasks could be virtualized deployed anywhere throughout the field-edge-network-cloud path, and (3) Adaptive data collection methods, which fuse in-situ sensor sampling and database sampling, could be developed. To this end, we will focus on two major research tasks.

SensorSlicer: At early stages of the project, we will deploy a 5G standard-compliant private cell using existing well-maintained open-source radio access network (RAN) and core software packages (e.g., OAI [1], srsRAN [2]). These packages allow connected common-off-the-shelf (COTS) endpoints (e.g., 5G dongles) and fully open-source software-defined radio-based endpoints while allowing innovations in the 5G stack. Accordingly, the network slicing functionalities will be tailored to the field-to-cloud workflow management using COTS endpoints. Then, a virtualized sensor-slicing solution will be developed using an open-source SDR-based endpoint implementation such that the same physical sensor network can provide different sensing processes without the need for separate deployments. The sensor slicing concept extends the emerging network slicing solutions at the network level to the individual sensing elements. Accordingly, a physical sensor can be virtualized to provide multiple roles in each workflow. In the second part of the project, we will work on providing an adaptive sampling functionality on a per-flow basis such that workflow-adaptive sampling could be supported, leading to an end-to-end feedback loop for field-to-cloud workflows as we describe next.

Field-to-HPC Adaptive Sampling: In sensor-aided workflows, current practices involve in-situ measurements that are taken with prior assumptions to build an associated model. However, validating these assumptions or improving resulting models is challengeing because our our understanding of the datasets evolves. Existing practices of deploying sensor networks for each workflow implies a considerable footprint in terms of energy consumed for sensing, communication, processing, and storage. Therefore, we will develop workflow-aware adaptive data measurement solutions as a key component of SensorSlicer. Specifically, we will develop novel sensing systems for adaptive sampling in agricultural fields. Relying on our decade-long experience in deploying Ag-IoT systems [26, 27], we will develop new sensors that are capable of real-time adaptation and edge-based adaptive sensing. Our edge platform (FabriStack) will deploy lightweight AI-based sampling processes in-situ such that data transfer and cloud storage can be minimized without affecting information fidelity. Since computing processes are virtualized, processes can be deployed across the field-to-HPC continuum to take advantage of these new sensing and network slicing capabilities.

Evidence that this approach will succeed: SensorSlicer relies on 5G network slicing concepts that are a part of the recent 5G standards. More importantly, recent open radio access network (Open-RAN) efforts, which aim to softwarize RAN deployments through open source software and commodity hardware, have resulted in standard-compliant implementations of 5G architectures (e.g., OAI [1], srsRAN [2]). Our group has extensive experience deploying these packages in large-scale wireless experimental testbeds [20].

Nebraska Experimental Testbed of Things (NEXTT) is a state-wide remotely accessible advanced wireless infrastructure developed in collaboration with the City of Lincoln and UNL [41]. The testbed includes 5G sites deployed on a Lincoln traffic intersection (AVC site) and four rooftop sites on the UNL campus as well as an indoor site. A sixth site was deployed and operationalized at ENREEC agricultural fields, replicating this site design. High-end sub-6GHz MIMO software-defined radio (SDR) transceivers are connected to UNL's Holland Computing Center (HCC) through 20Gbps dedicated fiber links. NEXTT has been designated an FCC program experimental license for sub-6GHz and mmWave bands.

NEXTT employs a fully containerized architecture with a continuous integration/continuous deployment (CI/CD) process to automatically build, test, and release code changes as they are committed. CI/CD and containerization simplify integration of existing open-source RAN packages in a modular fashion. To date, we have tested integration of different combinations of srsRAN, OAI's EPC, and RAN packages.

In this project, we will build on the established 5G sites [41], fully containerized software architecture for 5G RAN [20], and our decade-long experience in deploying remotely accessible agricultural sensor net-

works with autonomous irrigation systems [7, 27, 28, 33, 39]. The proposed SensorSlicer architecture relies on established network slicing solutions and requires extensions of this approach to resource-constrained IoT devices. We will utilize open-source implementations of 5G endpoint architectures to implement SensorSlicer and validate the proposed solutions in real-life agricultural field deployments. By doing so, SensorSlicer will coherently integrate with the full-stack system software (FabriStack) that we describe next.

3.2 The FabriStack: System Software for Digital-Physical Deployments – Lead: UCSB

The XGFABRIC system software must support a common set of software abstractions across all resource scales required by an application that couples devices, instruments, moderate computing and storage resources, and large-scale computing facilities. Further, these abstractions must be defined to support automated deployment, performance, and security at all scales. Finally, the system must be usable by scientific programmers and developers (i.e., the abstractions cannot be so complex that only a handful of experts can use them).

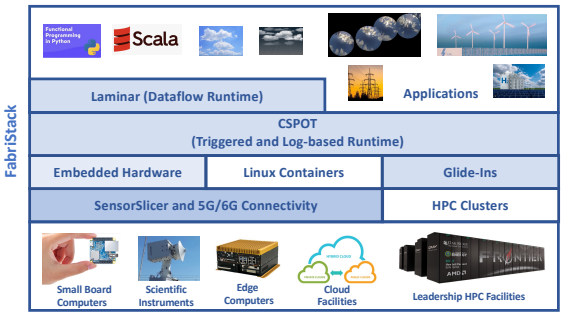

Figure 5: The Proposed FabriStack Layered Software Architecture. Shaded components comprise the FabriStack. CSPOT provides triggered computation and log-based storage universally – natively and as a guest in containerized or virtualized environments, as middleware on native operating systems or as a "glide in" on batch systems. Applications (at the top of the figure) use either Laminar dataflow programming or the CSPOT API.

To achieve this unification, we propose a layered software architecture. Figure 5 shows the relationship between the systems components we propose (shaded in the figure), multi-scale resources (along the bottom), existing systems, technologies, and applications (along the top). This *FabriStack* consists of 3 interoperating systems that enable applications to transparently span multi-scale, heterogeneous resources: CSPOT, Laminar, and Glide In.

CSPOT investigates multi-scale (sensors-edge-cloud) systems software for IoT deployments in remote locations where there is no access to the electrical power grid or networking [8, 14]. Laminar is a dataflow program representation that uses CSPOT as a runtime system. It is an innovation we propose to ease the programming burden associated with developing CSPOT applications natively. Finally, Glide In (a term borrowed from HTCondor [12]) is a facility for dynamically deploying and then decommissioning CSPOT to and from a large-scale, batch-controlled HPC systems. Together, these advances form a set of abstractions and runtime services that span the digital continuum that can be treated by developers and operations personnel as a single unified platform. Metaphorically, the FabriStack "weaves" the continuum of digital "threads" into a single, consistent "fabric" using 5G/6G connectivity.

3.2.1 CSPOT – Serverless Platform of Things written in C (UCSB)

The Research Challenge: To allow an application to amalgamate devices, computers, and storage at all resource scales (from embedded systems to supercomputers) requires a common set of unifying runtime system and programming abstractions. The alternative of integrating separate technologies developed for disparate purposes (e.g., embedded systems and cloud-web service for e-commerce) yields poor security, poor performance, a lack of maintainability, and low programmer productivity. Our team used the commercial cloud IoT technologies extensively in agricultural settings. These applications require between 7 and 15 separate code stacks and services to work together to enable basic computation and storage. Further, porting these technologies to new devices can be laborious. For example, an AWS-compatible implementation of TLS for a new and particularly low-power microcontroller took more than 7 weeks by an expert embedded systems programmer working in our group.

Addressing this challenge is difficult because it is difficult to "miniaturize" abstractions developed for full systems (and collections of full systems, like clouds) so that they work efficiently at *all* resource scales, including the small ones. Our previous work, and work proposed in this effort addresses this challenge.

CSPOT research to support XGFABRIC: For XGFABRIC, our research will focus on adapting CSPOT to use SensorSlicer to implement QoS and networking optimization in a network transparent way. Currently CSPOT implements network transparency using URNs to name its abstractions and all network performance optimizations and resilience functions are implemented by the CSPOT protocols. SensorSlicer will make it possible to create QoS-aware CSPOT logs and to tune internal resilience parameters (e.g. message timeouts). This integration not only extends the reach of CSPOT through its ability to leverage 5G and 6G connectivity, it improves its ability to scale out, through better timeout management and latency guarantees, and down, by using QoS guarantees to minimize log sizes.

This latter optimization is especially needed at the device level where memory is often severely restricted and persistent storage carries a heavy power cost. For embedded sensor controllers, the operation of the network via radio is often the largest part of the power budget allocated to each duty cycle. SensorSlicer will make it possible to offload the responsibility for implementing QoS to the network infrastructure, thereby reducing the network processing load on the device. We will integrate this capability transparently in the FabriStack using CSPOT as the integration vehicle.

Evidence that this approach will succeed: CSPOT imposes two unusual constraints on its programmers. The first is that all persistent storage must be log based. Append-only storage semantics have long been the basis for scalable, replicated storage (typically using eventual consistency replica protocols [22]) which has utility in an IoT context. However, log-based storage adds to this scalability strong crash-consistency semantics and good network partition recovery properties, both of which are essential for IoT. The second property is that computation can *only* be triggered when data is appended to some CSPOT log (called a *WooF* in CSPOT parlance). This property ensures that CSPOT (which uses a log to dispatch all computations internally) captures causal ordering of events automatically. That is, the CSPOT dispatch logs are used to record all append operations and scheduled computations making it possible to determine, as ground truth, causal execution dependencies.

These two peculiar requirements, originally introduced to support IoT in remote "infrastructureless" geographic locations, have important ramifications for XGFABRIC. First, CSPOT is designed specifically to address the tension between resilience and performance. All CSPOT functions execute concurrently and the only synchronization is the assignment of a sequence number to a log append. As a result, regardless of the degree of computational concurrency, a power failure or a crash leaves the logs in a consistent state (much like in a log-structured file system [25]). Similarly, a log append either fails to occur (in which case it will be retried) or it succeeds and the assigned sequence number in the response is lost (in which case a retry may increment the sequence number for a duplicate append). CSPOT client-side generated sequence numbers can then be used to implement duplicate suppression.

A second attractive property is that logs can be implemented very efficiently (more efficiently than file systems) at all resource scales. For example, CSPOT runs on an ESP8266 [11] microcontroller with 112k bytes of memory and an 80Mhz clock designed for embedded systems. A CSPOT `Put()` operation (the API command to append to a WooF) requires approximately 26 microseconds on this device [40]. This same simple approach (and API) scales up to larger computers and, ultimately, to clouds (where CSPOT is 1 to 3 orders of magnitude faster than FaaS platforms) creating a unified way to implement efficient resilient computation at *all* resource scales.

3.2.2 Laminar – Dataflow Programming for CSPOT (UCSB)

Research Challenge: CSPOT programs are notoriously difficult to write and to debug, particularly for programmers unfamiliar with development for distributed execution. While the CSPOT `Put` and `Get` operations are simple and act as synchronous functions, handler invocation is asynchronous and concurrent. Further, the only synchronization mechanism available is the transaction that assigns a sequence number (which is exposed through the `Put/Get` API) to a log entry when it is appended. Adding to this source of difficulty, CSPOT's append-only semantics essentially creates a new version (indexed by sequence number)

of each state update in a program each time the variable is updated. When program variables are structured or aggregate, the tendency is to serialize them in the log. Thus, the programmer familiar with typical concurrency management (e.g. locks, semaphores, condition variables) and mutable data structures often write "naive" CSPOT programs that "spin" polling for a sequence number increment as a synchronization mechanism, and log scans (to find the latest update to an aggregate data structure).

Addressing this challenge has proved difficult for several reasons. First, much of CSPOT's performance and all of its multi-scale portability (to the low end) accrue to its simplicity. It is analogous to an "assembly language" for building multi-scale, distributed applications end-to-end.

Programmability Research to Support xGFabric: We believe that a dataflow programming model can significantly enhance programmer productivity for applications that span multi-scale systems. To enable this, we have designed Laminar as a dataflow intermediate representation that both encodes dataflow computations (expressible as Directed Acyclic Graphs or DAGs) in CSPOT WooFs and also implements a dataflow runtime execution of these computations. In particular, the Laminar runtime implements efficient synchronization between data availability and handler invocation on behalf of the programmer.

We represent complex programs in Laminar as a hierarchy of DAGs where each DAG takes a set of inputs and produces a set of outputs. This hierarchical structure (inspired by IF1/IF2 [29, 30]) has two attractive properties. First, it supports program composition and modularity. A subgraph can be treated as a node within the graph in which it is embedded. Secondly, the graph hierarchy can encode functionality that is difficult to represent as a DAG. Iteration expressed a "for loop," for example, can be represented as a compound node consisting of three subgraphs. While powerful, dataflow necessarily elides architectural details at the application level. A developer cannot reason about "where" each dataflow node will fire nor make assertions about "when" it will execute once it is enabled by the presence of its inputs.

We plan to address these restrictions in two ways. First, Laminar programs consist of a "preamble" that specifies the application's deployment and the "body" which implements the application's logic. The preamble is generated using the same code generator as the body, but the code that is created is sequential. It initializes CSPOT WooF's to encode the deployment information (e.g. URN's associated with each CSPOT log) and the connectivity information. Once distributed, the preamble need only be executed once and the resulting deployment state is available to multiple executions of the application body.

Our plan is to enhance the preamble with language pragmas that allow a developer to attach QoS and capacity requirements to each node and edge in a Laminar program. These requirements will allow the Laminar runtime to select appropriate execution sites and provision the necessary QoS from SensorSlicer for the intervening network connectivity. For example, in a case where functionality can be implemented either on resource restricted devices (say one Laminar node per device) or (with a higher latency) on an edge computer, in the event of device failure(s), it will be possible to combine the nodes from the failed devices into a "supernode" that can be executed on the edge computer without changing the semantics of the Laminar program, and to provision the necessary network capacity for the supernode using SensorSlicer.

We also plan to use this functionality to incorporate large-scale computing facilities into Laminar programs. Rather than expressing all computations at the finest level, we envision special "macro nodes" that embody whole programs (e.g. written in C, C++, or Fortran using MPI) that will execute on systems where xGFabric must act strictly as a guest (i.e. is operating as middleware) as on the LCF. These macro nodes act as nodes within the Laminar program that cannot be decomposed (at least, automatically) into finer-grained nodes (although other nodes may be combined with them should they too require execution on a large system). Our Laminar benchmark suite includes both Map-Reduce and Logistic Regression implemented as macro nodes with embedded parallelism. The high-level workflow is triggered and managed by Laminar and while the embedded computations within each node can either be Laminar primitives or programs written in high-level languages that gather their inputs and return their outputs through Laminar.

In this way, Laminar becomes a high-level control program that unifies execution across a multi-scale

resource deployment, where it can execute either on "native" CSPOT or as a middleware interfacing to "macro nodes" that do not use CSPOT as a runtime. Because dataflow implements applicative program semantics, we also plan integrations with popular high-level languages such as Functional Python and Scala. It should also be possible to using graphical tools (such as Graphviz [9]) with Laminar to aid in program development and debugging.

Evidence that this approach will succeed: The recent rise and popularity of dataflow programming in cloud computing (e.g. Google Cloud Dataflow and Microsoft Analytic Dataflows, Node-RED, GNURadio) show that dataflow can simplify programming and management of complex applications while facilitating reuse and customization. Our work will bring these benefits to multi-scale, distributed, and failure prone settings by combining them with a foundation of portability and resiliency at the runtime layer.

4 Work Track 2: Edge-to-Exascale Resource and Workload Management

Research Challenge: The principal workflow challenges of task placement, resource shaping and selection, must be addressed in the context of dynamic and multiscale resources, and application and resource level heterogeneity.

Multiscale resources vary in terms of availability, capacity, and response over several orders of magnitude. Together with application and resource heterogeneity, an adaptive response based on a "one pass" translation of a task-graph into an execution plan is rendered ineffective. The workflow control system must provide translation via multiple intermediate representations. The specifics of translation are sensitive to the quality of state information and its uncertainty. The translation from task-graph to execution plan must also trade-off between timely response and near-optimal plans on the one hand, with resilience on the other.

Resilience challenges traditional notions of workflow performance: for example, execution plans must be optimized not for makespan, but for resilient execution. The restatement of the objective is simple, however the objective function of "maximal (optimal) resilience" is a complex, multidimensional function with often ill-defined trade-offs. Providing resilience, say using replication in a way that minimizes correlated failure probabilities, introduces performance costs, such as increased latency or decreased throughput. Workload management must also resolve this tension between resilience and performance.

Workflow Research The workflow middleware will provide a distributed workload management system that can select, acquire and dynamically scale resources across the edge-to-exascale continuum. Specifically, the middleware must enable: (i) resource selection and task placement, and management of these processes so as to allow collective decision making but independent control; (ii) a decision making framework for different modes of data and task placement, while being agnostic to specific resource, services and task properties, and (iii) tunable to diverse performance measures including reliability, response time, power or cost of data movement. These are in addition to associated performance measures such as makespan, latency tolerance, and scale.

Three specific research themes that will be explored: (i) In the presence of heterogeneity and dynamism, fine-grained control and adaptive coordination of the placement of data and tasks is needed. We will explore workload management and associated algorithms to provide the basis for an initial execution plan; (ii) We will investigate robust decision framework (based on models of system behavior) to respond to dynamism in the context of data-task execution and resource topology; (iii) We will perform the system software research to provide the collective and end-to-end integration of capabilities while preserving performance and resilience requirements.

An affinity-based workload management model predicated on the trifecta of data-resource-task, and the coupling of individual data units, resources and tasks (as measured by their relative "affinity" to other units) allows planning task placement, and the flow and processing of data on the edge. The data-resource-task affinity model provides the ability to influence the placement of data-resources-task without exposing either implementation details or "how". It will support diverse scenarios (e.g., data flows out to the edge, data-

joins from distributed edge locations etc.). A unique feature of Laminar will be the ability to decompose workloads into functions for fine-grained placement (e.g., device level) that can be aggregated to support more coarse-grained placement (e.g., on the edge, private cloud or exascale system). The granularity versus placement trade-off also present powerful adaptive response opportunities.

Because execution plans and predictions do not always agree with reality, we formulate a data-model "discrepancy function" about which we will reason about adaptive response. The discrepancy is an unknown function over which we can express a Bayesian uncertainty to be minimized as a middleware operational objective. However, many open research and implementations questions related to a resilient middleware that supports robust (decision making) adaptations must be addressed: How to formulate an adaptable execution plan and quantify uncertainty in adaptation? How to accurately assess the cost of a change in execution plan and implication for resilience? How to assess when to adapt and how to configure resources while preserving resilience and robustness?

The XGFABRIC system component of distributed resource management system will be the pilot abstraction. It will acquire resources and be responsible for the fine-grained resource partitioning and assignment on the diverse range of resource end-points. The immediate challenge here will be the design of a pilot system that is functionally capable and consistent with the established pilot paradigm but realized on multi-scale resources at very different points in the performance-persistence property space.

We will develop the interfaces to task-level services provided by Laminar and CSPOT needed by the workload management system, e.g., monitoring data, tools for getting data for analysis. To a CSPOT task the pilot system will be accessible as an external service, agnostic of the task semantics and internal CSPOT optimizations. The pilot system will expose well-defined interface that allow CSPOT programming system to probe at runtime to determine state, and declaration of where/when workloads are destined.

Information flow and interface considerations aside, fundamental challenges arising from integrating different system abstractions for distributed resource management (pilot) and runtime systems (CSPOT) need to be addressed. For example, how the pilot bootstraps CSPOT will likely differ across resource types. Two obvious modes warrant investigation: (M1) using the pilot to launch CSPOT tasks, which in turn manages tasks, and/or (M2) Using the pilot as an execution environment running within CSPOT.

In the case of (M1), the hierarchy of XGFABRIC abstraction will be that the glide-in acquires and configures resources, and delivers it to the pilot; the pilot starts its management components, which controls the execution of computing tasks (self-contained processes) on acquired resources; the CSPOT execution environment will in turn be described as a computing task, and will be processed by the pilot as a special task operating within the confines of the pilot.

In the case of M2, the execution chain that the glide-in acquires and configures resources, and launches CSPOT; then an application using CSPOT starts and interfaces with pilot, which controls the execution of tasks. Modes M1 and M2 differ in their resilience vs performance profiles, as well as how they qualitatively and quantitatively respond to dynamism. It is likely that M2 will be initial mode on edge and IoT devices. The challenge will be managing the cross-over behavior to M1 – which is the currently envisioned mode on HPC/data-center/cloud environments.

Evidence that this approach will succeed: Our workflow approach employs the "classic" three-level architecture. The three levels provide a trade-off between complexity and capability, extensibility and specificity, as well as generality and performance.

The pilot abstraction [6, 19, 21] is both powerful and general to employ across the multi-scale resources. To leverage this utility, XGFABRIC research will focus on "right-sizing" the selection of resources to be allocated to each pilot in a multi-scale deployment (i.e. from edge to HPC platform). To guide both qualitative and quantitative insights, we will also develop an emulator to investigate new algorithms and adaptive execution plans. The emulator will hide the complexity of the distributed environment, allow the project to validate new operational models, and derive insight without having to replicate the complex distributed and

dynamic environment. Finding optimal adaptive execution plans – already an NP-hard problem – in the presence of application performance constraints will require heuristics, and solutions with bounded deviations from optimality. As scale and dynamism increases, or resource (task) heterogeneity increase, improvements due to adaptivity will continue to increase. We will use the emulator for instant insight and judicious choice of algorithms, but not be constrained by it.

5 Work Track 3: Deployment, Testing and Validation

5.1 Deployment via Glide-In: The FabriStack as Middleware (UCSB, BNL, Notre Dame)

To demonstrate a usable end-to-end system spanning the entire range from IoT sensors to HPC facility, connected by 5G/6G networks, a global orchestration component must be created. The XGFABRIC orchestrator will accept an application definition, assess the availability of resources at the requested sites, then allocate appropriate resources from the edge, network and HPC facilities, and then deploy the XGFABRIC software. Once the workflow has run to completion, the outputs will be extracted, and the resources cleaned up.

The Research Challenge: The fundamental research challenge is to provide the system software facilities to "speed match" the parts of a streaming application that are immediately available when ever they are presented with data (and probably on some regular duty cycle) with the batch-controlled resources that must be provisioned in priority order so as to maintain good system resource utilization. Further, this alignment cannot mandate that the HPC resources install new software to meet this challenge. To leverage large-scale high-performance computing (HPC) resources as essential computing elements requires that the FabriStack be able to act as a "guest" when using these resources. Put another way, we do not expect that CSPOT, Linux containers, or "root" Linux access will be available to the FabriStack as native technologies from batch-controlled HPC resources. Thus, our approach is to "glide in" FabriStack functionality, dynamically, as middleware that operates as an unprivileged guest on these systems. Addressing this challenge is difficult because HPC resources appear, to applications using the FabriStack, as intermittently available resources. They are reliable (relative to small devices) in that they do not fail often once they become available, but batch queuing delays interpose availability.

Glide In for HPC resources Integrated by XGFABRIC: Our research approach for implementing FabriStack glide in will consist of three ideas for matching the cadence of the sensor or streaming applications outside of the an HPC facility with the "bursty" availability caused by batch queuing. These advances are predicated on research in adaptive resource management and response. The first is to provision staging areas, using the FabriStack, in private or public clouds where resources can be dynamically acquired and released "on demand." The purpose of these staging areas is to hold the inputs needed by an HPC computation once it has emerged from a batch queue and begins executing.

Note that Laminar's strict dataflow semantics will not trigger a macro node before all of its inputs are available. Our idea is to introduce a "pass through" node type into Laminar that will be assigned to a cloud (or other resource with sufficient capacity to stage the inputs needed by a large HPC computation). This pass-through node will be inserted into a Laminar program immediately ahead of any macro node assigned to an HPC resource. The pass through node will not "fire" (send its outputs to the HPC macro node) until all of the inputs are present. The Laminar and CSPOT retry mechanisms will then treat the HPC macro node that takes its inputs from the pass through node as if it is available across a potentially faulty network link. The computation corresponding to the macro node will be submitted to the batch queue and we will configure the pass through node to continue to retry the initiation of the macro node computation while the computation waits for execution in the queue.

We will need to implement the ability to instantiate a CSPOT deployment in a cloud dynamically to implement the pass-through node. If private clouds are available, we will dedicate some amount of private cloud resource to HPC staging of Laminar pass-through nodes. However, for public clouds (where idle resources incur unnecessary charges) we will dynamically instantiate CSPOT. We will explore using a pi-

lot [21] to effect the actual glide in. Once the computational elements of the HPC resource are provisioned to the macro node, the pilot will execute the tasks that are necessary to contact the pass-through node, acquire the staged inputs, start the computational tasks, and gather the results.

Evidence that this approach will succeed: Of the three components that comprise the FabriStack, we have the least preliminary experience with Glide In. CSPOT does have a replay facility [15] that is primarily for data repair when sensor data is deemed to be faulty, or when bugs are discovered and repaired in in-line computations within a streaming application. We believe we will be able to leverage this capability to dynamically provision CSPOT WooFs within a running Laminar application by creating an empty WooF and then "repairing" it as if it simply contained faulty data. If this approach fails, we have also developed a highly resilient implementation of a replicated "pub-sub" service based on Chord [31] and RAFT [23] (using CSPOT as the runtime) called Canal [16]. Canal is prepared to use RAFT's strongly consistent replication semantics and Chord's churn mechanisms to dynamically acquire and release CSPOT hosts. It is not yet clear whether log repair will be sufficient to implement Laminar pass-through nodes or whether a more resilient system such as Canal will be necessary.

5.2 Testbed and Deployment Plan

The XGFABRIC testbed will consist of a **software repository** containing the latest releases of each of system components, a **application repository** containing exemplar software applications that can be deployed, and a set of **experimental facilities** available to the team, noted in the Facilities appendix below.

We expect at the outset of the project that each software effort will quickly produce a "minimally compliant" artifact that can be used as the baseline to allow for integration testing and and evaluation of facility and application constraints. As the various software components (SensorSlicer, Laminar, CSPOT, Laminar, Glide-In) develop along their own tracks, the XGFABRIC software repository will identify milestone versions of these components and perform automated tests to evaluate basic internal compatibility. This will result in "known good" configurations that all project participants can rely upon for successful execution. Known good configurations will then be tested against benchmark applications on the available testbed facilities, which consist of the AgIoT facilities (e.g. CUPS at UCSB, Spidercam at UNL) combined with HPC facilities (e.g. HCC at UNL, CRC and ND). As integrating testing reveals inter-component problems, or component-facility problems, these will be fed back to the individual tracks for debugging and refinement. Once the integration produces working combinations, we will proceed to validation and testing.

5.3 Validation Objectives

We will consider the end-to-end execution of XGFABRIC to be **correct and successful** if a user at a "neutral" site is able to orchestrate the deployment, operation, and teardown of an analysis workflow that spans an agricultural facility and an HPC facility, connected by a 5G/6G network, without fixed allocations at either of those sites. For example, an operator physically located at Notre Dame should be able to connect the CUPS facility at UCSB with an HPC facility at BNL, execute the desired workflow, and then fully cleanup.

While the successful coordinated execution of such a complex system is the primary objective, it is also necessary to have sufficient performance to meet application needs. We will observe and address performance in the following dimensions: **Whole-system bandwidth and latency.** The system performance is ultimately constrained by the ability to move data all the way from the sensor, over the wireless network, to the relevant node on the HPC facility. We will measure to what extent the maximum wireless network performance is reduced by the end-to-end system components. **Resource allocation efficiency.** When allocating coordinated resources from multiple facilities (edge, network, HPC), it is unavoidable that some mismatch will result in under-use of one resource or another. (For example, the available network bandwidth may exceed what the edge facility is capable of producing.) We will measure the allocated and used capacity of each resource with an eye towards maximizing utilization withing performance constraints. **End-to-end iteration time.** The ability to take meaningful action on agricultural facilities will be limited by the end-to-

end iteration time needed to deploy, measure, simulate, and perform an actuation on the facility.

6 Timetable of Activities and Milestones

Activities are organized to ensure the rapid integration between research insight, middleware and system development, and science drivers, This is reflected in yearly integration and usage for science milestones (Track 3) based upon XGFABRIC releases 1.0 and 2.0 (Table 1).

The evaluation and impact of individual work packages are discussed in individual work package sections. Project-level evaluation metrics will include: (i) Uptake of prototypes beyond this project will be a consideration; (ii) ease of deployment and portability of applications across diverse test-beds and exascale systems, (iii) resiliency to faults and intermittent connectivity across the wide area, the efficiency of placement/migration decisions and the speed with which decisions can change in response to the system behavior and resource availability. Finally, we will also measure the speed and accuracy with which our prototypes can be controlled and adapted in response real-time data (fused from multiple sources).

To advance scientific goals, XGFABRIC will be deployed and integrated on platforms of practice, DOE leadership-class computing facilities, and other institutional testbeds and resources.

	Month 6	**Month 12**	**Month 18**	**Month 24**
Track 1	• Deploy srsRAN and OAI private cells at ENREEC • Extend CSPOT support to exascale systems • Define abstractions and services for geo-distributed monitoring	• Deploy 5G network slicing primitives • Define CSPOT APIs/services for WMS runtime capabilities • Specialize services to digital agriculture workloads and deployments	• Implement and deploy SensorSlicer • APIs/Services for data fusion from disparate sources • Tooling for validation and stress/-fault testing	• APIs for adaptive sensing • CSPOT task decomposition support for WMS • End-to-end validation and stress/-fault testing
Track 2	• Architecture, component and interface design for multiscale workload management system • Affinity model for data-resource-task	• Resource selection and task placement algorithms using affinity model • Implement pilot system for XGFABRIC	• Adaptation using dynamic information and optimization • Bidirectional system integration with CSPOT	• Evaluate and improve heuristics for optimal execution plans • Performance, Scale and Benchmarks
Track 3	• Collect minimally complete software components. • Collect science drivers.	• Develop orchestration and test on minimal components and science drivers. • Test integration of 1.0 components on testbeds.	• Develop resilience techniques for variable resource availability. • Test integration of 2.0 components on testbeds.	• Develop techniques for capacity matching between facilities. • Evaluate performance of 3.0 components on testbed.
Milestones	• Develop first minimal complete components • Deploy 5G private network	• XGFABRIC 1.0 and initial integration with science driver • Deployed on Exeter test-bed	• XGFABRIC full integration with additional science drivers • XGFABRIC 2.0	• XGFABRIC deployed on Sedgewick, Spidercam test-beds & DOE platforms • XGFABRIC 3.0

Table 1: Timetable of Project Activities and Milestones.

Appendix 1 – References

[1] OpenAirInterface. http://www.openairinterface.org/.

[2] srsRAN Project. https://www.srsran.com.

[3] Notice of proposed rulemaking. Technical report, FCC, Aug. 2019. https://docs.fcc.gov/public/attachments/FCC-19-77A1.pdf.

[4] Radhakisan Baheti and Helen Gill. Cyber-physical systems. *The impact of control technology*, 12(1):161–166, 2011.

[5] Pete Beckman, Charlie Catlett, Moinuddin Ahmed, Mohammed Alawad, Linquan Bai, Prasanna Balaprakash, Kevin Barker, Pete Beckman, Randall Berry, Arup Bhuyan, Gordon Brebner, Klaehn Burkes, Anastasiia Butko, Franck Cappello, Ryan Chard, Scott Collis, Johnathan Cree, Dipankar Dasgupta, Anatoly Evdokimov, Jason M Fields, Peter Fuhr, Colby Harper, Yier Jin, Rajkumar Kettimuthu, Mariam Kiran, Robert Kozma, Praveen Ashok Kumar, Yatish Kumar, Linqing Luo, Lena Mashayekhy, Inder Monga, Bill Nickless, Thrasyvoulos Pappas, Elena Peterson, Trever Pfeffer, Shaloo Rakheja, Veroica Rodriguez Tribaldos, Sterling Rooke, Sumit Roy, Tarek Saadawi, Alec Sandy, Rajesh Sankaran, Nicholas Schwarz, Suhas Somnath, Marius Stan, Cory Stuart, Ryan Sullivan, Anirudha Sumant, Greg Tchilinguirian, Nhan Tran, Arun Veeramany, Angela Wang, Bin Wang, Andrew Wiedlea, Stijn Wielandt, Theresa Windus, Yuxin Wu, Xi Yang, Zhi Yao, Rose Yu, Yuping Zeng, and Yuepeng Zhang. 5g enabled energy innovation: Advanced wireless networks for science. *PNNL Workshop Report*, 3 2020.

[6] Georgios Chantzialexiou, Andre Luckow, and Shantenu Jha. Pilot-streaming: A stream processing framework for high-performance computing. In *2018 IEEE 14th International Conference on e-Science (e-Science)*, pages 177–188. IEEE, 2018.

[7] X. Dong, M.C. Vuran, and S. Irmak. Autonomous precision agriculture through integration of wireless underground sensor networks with center pivot irrigation systems. *Ad Hoc Networks*, 11(7):1975–1987, 2013.

[8] Andy Rosales Elias, Nevena Golubovic, Chandra Krintz, and Rich Wolski. Where's the bear?-automating wildlife image processing using iot and edge cloud systems. In *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 247–258. IEEE, 2017.

[9] John Ellson, Emden R Gansner, Eleftherios Koutsofios, Stephen C North, and Gordon Woodhull. Graphviz and dynagraph—static and dynamic graph drawing tools. *Graph drawing software*, pages 127–148, 2004.

[10] UCSB Early Research Scholars Program, 2023. https://ersp.cs.ucsb.edu [Online; accessed 3-Feb-2023].

[11] Espressif 8266 microcontroller. https://en.wikipedia.org/wiki/ESP8266, 2018. [Accessed electronically, September 2018].

[12] EM Fajardo, JM Dost, B Holzman, T Tannenbaum, J Letts, A Tiradani, B Bockelman, J Frey, and D Mason. How much higher can htcondor fly? *Journal of Physics: Conference Series*, 664(6), 2015.

[13] 2024. https://farm-ng.com [Online; accessed 5-Apr-2024].

[14] Nevena Golubovic, Rich Wolski, Chandra Krintz, and Markus Mock. Improving the accuracy of outdoor temperature prediction by iot devices. In *2019 IEEE International Congress on Internet of Things (ICIOT)*, pages 117–124. IEEE, 2019.

[15] W-T. Lin, F. Bakir, C. Krintz, R. Wolski, and M. Mock. Data repair for Distributed, Event-based IoT Applications. In *ACM International Conference On Distributed and Event-Based Systems*, 2019.

[16] Wei-Tsung Lin, Rich Wolski, and Chandra Krintz. A programmable and reliable publish/subscribe system for multi-tier iot. In *2021 8th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, pages 1–8. IEEE, 2021.

[17] K. LoPiccalo. Impact of broadband penetration on U.S. farm productivity. Technical report, Federal Communications Commission, Office of Economics and Analytics, Feb. 2021. https://docs.fcc.gov/public/attachments/DOC-368773A1.pdf.

[18] University of California Division of Agricultural and Natural Resources, Lindcove Research and Extension Center. http://lrec.ucanr.edu. [Online; accessed 13-mar-2023].

[19] Andre Luckow, Kartik Rattan, and Shantenu Jha. Pilot-edge: Distributed resource management along the edge-to-cloud continuum. *arXiv preprint arXiv:2104.03374*, 2021. Accepted for PAISE'21 (IPDPS 21).

[20] Mohammad M. R. Lunar, Jianxin Sun, John Wensowitch, Michael Fay, Halit Bugra Tulay, Sai Suman, Brian Qiu, Deepak Nadig, Garhan Attebury, Hongfeng Yu, Joseph Camp, Can Emre Koksal, Dario Pompili, Byrav Ramamurthy, Morteza Hashemi, Eylem Ekici, and Mehmet C. Vuran. OneLNK: one link to rule them all: Web-based wireless experimentation for multi-vendor remotely accessible indoor/outdoor testbeds. In *ACM Workshop on Wireless Network Testbeds, Experimental evaluation and CHaracterization (WiNTECH'21)*, Feb. 2021.

[21] Andre Merzky, Matteo Turilli, Mikhail Titov, Aymen Al-Saadi, and Shantenu Jha. Design and performance characterization of radical-pilot on leadership-class platforms. *IEEE Transactions on Parallel and Distributed Systems*, 33(4):818–829, 2021.

[22] James Murty. *Programming amazon web services: S3, EC2, SQS, FPS, and SimpleDB*. " O'Reilly Media, Inc.", 2008.

[23] Diego Ongaro and John Ousterhout. In search of an understandable consensus algorithm. In *2014 {USENIX} Annual Technical Conference ({USENIX}{ATC} 14)*, pages 305–319, 2014.

[24] Philippe Rolshausen. Prospects for farming citrus under protective screening. *Citrograph*, 14(4), 2023.

[25] Mendel Rosenblum and John K Ousterhout. The design and implementation of a log-structured file system. *ACM Transactions on Computer Systems (TOCS)*, 10(1):26–52, 1992.

[26] Abdul Salam, Mehmet C. Vuran, Xin Dong, Christos Argyropoulos, and Suat Irmak. A theoretical model of underground dipole antennas for communications in internet of underground things. *IEEE Transactions on Antennas and Propagation*, 67(6):3996–4009, jun 2019.

[27] A.R. Silva and M.C. Vuran. $(CPS)^2$: Integration of center pivot systems with wireless underground sensor networks for autonomous precision agriculture. In *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems, ICCPS '10*, pages 79–88, 2010.

[28] A.R. Silva and M.C. Vuran. Development of a testbed for wireless underground sensor networks. *Eurasip Journal on Wireless Communications and Networking*, 2010, 2010.

[29] Stephen K Skedzielewski and ML Welcome. Data flow graph optimization in if1. In *Functional Programming Languages and Computer Architecture: Nancy, France, September 16–19, 1985*, pages 17–34. Springer, 1985.

[30] Stephen K Skedzielewski and ML Welcome. Data flow graph optimization in if1. In *Functional Programming Languages and Computer Architecture: Nancy, France, September 16–19, 1985*, pages 17–34. Springer, 1985.

[31] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Netw.*, 11(1):17–32, 2003.

[32] Fei Tao, He Zhang, Ang Liu, and Andrew YC Nee. Digital twin in industry: State-of-the-art. *IEEE Transactions on industrial informatics*, 15(4):2405–2415, 2018.

[33] J. Tooker, X. Dong, M.C. Vuran, and S. Irmak. Connecting soil to the cloud: A wireless underground sensor network testbed. In *Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks workshops*, volume 1, pages 79–81, 2012.

[34] N. M. Trendov, S. Varas, and M. Zeng. Digital technologies in agriculture and rural areas – status report. Technical report, Food and Agriculture Organization of the United Nations, Rome, Italy, 2009. Licence: cc by-nc-sa 3.0 igo.

[35] UC Agriculture and Natural Resources, 2023. https://ucanr.edu[Online; accessed 3-Feb-2023].

[36] UC Natural Reserve System, 2023. https://ucnrs.org[Online; accessed 3-Feb-2023].

[37] UCSB RACELab, 2023. https://sites.cs.ucsb.edu/~ckrintz/racelab.html[Online; accessed 3-Feb-2023].

[38] 2019. http://sedgwick.nrs.ucsb.edu [Online; accessed 14-Feb-2019].

[39] Mehmet C. Vuran, Abdul Salam, Rigoberto Wong, and Suat Irmak. Internet of underground things in precision agriculture: Architecture and technology aspects. *Ad Hoc Networks*, 81:160–173, dec 2018.

[40] Rich Wolski, Chandra Krintz, Fatih Bakir, Gareth George, and Wei-Tsung Lin. Cspot: portable, multi-scale functions-as-a-service for iot. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, page 236–249. ACM, Nov 2019.

[41] Zhongyuan Zhao, Mehmet C. Vuran, Baofeng Zhou, Mohammad M.R. Lunar, Zahra Aref, David P. Young, Warren Humphrey, Steve Goddard, Garhan Attebury, and Blake France. A city-wide experimental testbed for the next generation wireless networks. *Ad Hoc Networks*, 111:102305, Feb. 2021.