Carnegie Mellon University
Software Engineering Institute

Computing at the Edge

Challenges and Priorities for Software Engineering and AI

Kevin Pitstick, Marc Novakouski, Grace Lewis, and Ipek Ozkaya

WHAT IS EDGE COMPUTING?

Unlike traditional computing, where processing is mainly performed on local servers and in the cloud, edge computing pushes applications, data, and computing power to the edge of the Internet—to mobile devices, sensors, and end users. The number of devices connected to the Internet, and the volume of data being produced by those devices and used by governments and businesses, is growing far too quickly for traditional computing approaches and data center infrastructures to accommodate.

WHY EDGE COMPUTING?

Moving compute to the edge has many benefits:

- **Latency:** When data is processed close to its origin, it avoids round-trip time to remote servers or the cloud. This leads to faster response times, which enhances the human-machine experience critical to the success of edge missions.
- **Bandwidth:** When bandwidth is limited, edge computing provides an avenue to filter and optimize which data gets sent back to the cloud.
- **Privacy & Security:** Processing sensitive data locally avoids the risk of interception in network transmissions. Data can be anonymized or cleaned before sharing, if needed.
- **Resiliency:** If connectivity to the cloud is unavailable, edge computing allows processing to continue locally until connectivity is restored.

WHY ARE EDGE SYSTEMS IMPORTANT FOR THE DOD?

The Department of Defense (DoD) 2018 National Defense Strategy highlights the need to shift focus to operating in contested environments, which are significantly harder to operate in. These locations are hard to reach, actively denied, and resource constrained. Advanced edge systems are vital for the DoD to insert capabilities into these regions. In addition, artificial intelligence and machine learning (AI/ML) have the potential to add autonomy to these systems, allowing them to operate in areas where operators are unable to go.

Edge Computing Characteristics and Challenges

Many challenges for systems operating in edge environments stem from traditional challenges of distributed computing environments. However, some characteristics of edge environments exacerbate existing concerns and create new software and AI engineering challenges.

1. **Limited Computing Resources.** Due to size, weight, and power (SWAP) limitations in edge environments, devices deployed to the edge are limited in their computing resources, such as CPU, RAM, and GPU.

   Accessing cloud resources may not be possible at the edge. To instead bring these capabilities to the edge, software engineering efforts must design for *distributability,* which is the ability of system components to be split across multiple compute nodes. While a capability may be too compute-intensive for a single edge node, separating the capability into smaller components—such as microservices, each in its own container—across multiple nodes can enable its deployment on distributed, heterogenous devices.

   Designing ML models for the edge often requires making trade-offs between accuracy, speed, and hardware available at the edge. Industry offers a number of model acceleration techniques to maximize speed gains and minimize accuracy losses—such as network pruning, knowledge distillation, and quantization—and is releasing ML inference engines that are optimized for different hardware. However, rigorous experimentation is required to ensure that capabilities work properly with hardware and that observed trade-offs are acceptable for targeted missions.

2. **Intermittent/Denied Network Connectivity.** Many computing capabilities are designed for "always connected" networks and do not handle intermittent connections gracefully. Software written for edge nodes must be robust to handle missions that operate in areas of limited network infrastructure or where communications are actively denied.

   Reliability, which is the ability of a system to continue operation under fault conditions, is a key quality attribute to consider for edge connectivity. Edge software should be designed to reconnect automatically without locking up when the network disconnects. Temporary recovery strategies could be used, such as spinning up local services to replace unreachable remote services.

   When data is being collected and transmitted for AI tasks, keeping all of the data is often not an option due to the combination of limited edge storage and network connectivity. Summarization and filtering techniques, depending on the type of data and storage requirements, can also help reduce the amount of data that is sent over the network.

3. **Privacy.** Data collected at the edge could have personally identifiable information (PII), requiring privacy-preserving components that clean data before sending it to the cloud. Systems may need to do all of the processing on the edge node if PII data cannot be sent to the cloud, either due to policy restrictions or lack of user consent.

   ML algorithms are frequently trained or retrained on data collected at the edge. When cleansing data to remove PII, special care must be taken to ensure that the data is still useful for training purposes. If PII is removed entirely, there may not be a way to associate between data samples. If raw data cannot be stored, there is no opportunity to reprocess it if the cleaned data has issues.

4. **Security.** In tactical edge environments, computing nodes are operating near adversaries, who are always looking for opportunities to sabotage the mission.

   If edge nodes are operating on local networks, the assumption must be that adversaries are monitoring the network and all data must be encrypted. If edge nodes are operating with sensitive or classified information, special zeroization functionality may need to be included so that data can be quickly destroyed if the node is compromised.

   A major security risk on edge notes in contested environments is adversaries' attempts to influence the decision-making process of the algorithms, including ML algorithms. These attempts could be through direct control of the software, manipulation of the input data, or data poisoning. Adversarial ML is a growing field that investigates these attacks as well as strategies for developing AI systems robust to them.

5. **Uncertainty.** Operations at the edge, whether they are disaster relief missions or active combat, often come with some level of uncertainty. Edge systems must be able to adapt to meet constantly shifting priorities.

   For edge systems to be robust to uncertainty, engineers must design them with *adaptability* and *flexibility* in mind. Edge software should be able to adapt during operations to both the data it is processing and the type of computation it is executing. Because an operator may not have the attention to adjust the software on the fly, setting up specific "modes" for different phases of the mission can provide flexibility with simplicity for the operator.

   Mission uncertainty poses challenges for ML models as well. Because ML models are trained on specific datasets, a drift in operational data can lead to large decreases in model accuracy. AI-enabled edge systems facing this challenge should be capable of domain adaptation, which is the ability to adapt a model to a new domain.

6. **Scalability.** Devices at the edge tend to be smaller and more specialized. Hundreds or thousands of nodes may need to discover each other, connect, and share data, leading to scalability challenges. These nodes are often heterogenous, including different hardware and software elements. In addition, tactical and humanitarian environments often have multiple entities—such as local populations or non-governmental organizations—with distinct devices and computing nodes. Resulting scalability challenges include how to get all of the nodes to interoperate, how to process all of the data, and how to minimize load on the network.

   To address these challenges at the edge, a key attribute to design for is *interoperability,* which is the degree to which two or more systems can usefully exchange meaningful information via interfaces. Taking advantage of messaging middleware can ease the process of systems integration. Using cross-platform network serialization libraries, such as protocol buffers and interfaces that clearly define formats, can help prevent data formatting errors.

   Scalability is a particularly important ML challenge at the edge. Not only is ML model inference compute-intensive, but also it generally requires specialized hardware, such as GPUs or TPUs. This can lead to an increase in device heterogeneity.

7. **Limited Attention.** Both in humanitarian and tactical edge systems, operators must perform their missions under extremely stressful conditions with full attention and rapid decision making, leaving little to no time to focus on additional devices and interfaces.

   To accommodate limited attention, edge systems must be architected with the attributes of *usability* and *simplicity* in mind. Visual interfaces must be simple and uncluttered and should display only the most relevant information. Input devices should be simple, configurable, intuitive, and tactile.

   When integrating AI and ML components into edge systems, human-machine teaming is an important area of consideration, and special care should be taken to design how the AI components interact with the operator.

8. **Data.** While data has been mentioned as part of many of the challenges, it deserves to be called out specifically because it is central to edge computing. One key motivation for bringing computing to the edge is to support sensors that can collect large amounts of high-fidelity data that is central to the mission. This desire for more data must be balanced with the limited capabilities of edge systems, and trade-offs must be made.

How the SEI Can Help

The SEI has a dedicated applied research and development team who creates and transitions innovative solutions, principles, and best practices for architecting and developing systems to support teams operating at the tactical edge and using AI/ML techniques for improved capabilities and mission support. Contact us to learn more about the benefits of edge computing and how we can collaborate to solve today's and tomorrow's edge computing challenges.

About the SEI

The Software Engineering Institute is a federally funded research and development center (FFRDC) that works with defense and government organizations, industry, and academia to advance the state of sthe art in software engineering and cybersecurity to benefit public interest.

Part of Carnegie Mellon University, the SEI is a national resource in pioneering emerging technologies, cybersecurity, software acquisition, and software lifecycle assurance.

Contact Us

CARNEGIE MELLON UNIVERSITY
SOFTWARE ENGINEERING INSTITUTE
4500 FIFTH AVENUE; PITTSBURGH, PA 15213-2612

sei.cmu.edu
412.268.5800 | 888.201.4479
info@sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM21-0904