Towards a Software Defined Truck

Subhojeet Mukherjee, Jeremy Daily

Colorado State University

Introduction

- Heavy vehicles are the lifeblood of logistics operations
  - Equipped with electronic controls, embedded networking, remote monitoring and diagnostic
- Cyber-security concerns have been raised lately
- SAE-J3061 recommends concurrent testing and verification for cyber-security analysis [8]
  - Trucks are expensive with challenging logistics
  - Not all parts are readily available

BBC | Sign in | Home | News | Sport | Reel | Worklife | Travel

NEWS

Home | Coronavirus | Video | World | US & Canada | UK | Business | Tech | Science | Stories | Entertainment & Arts

Tech

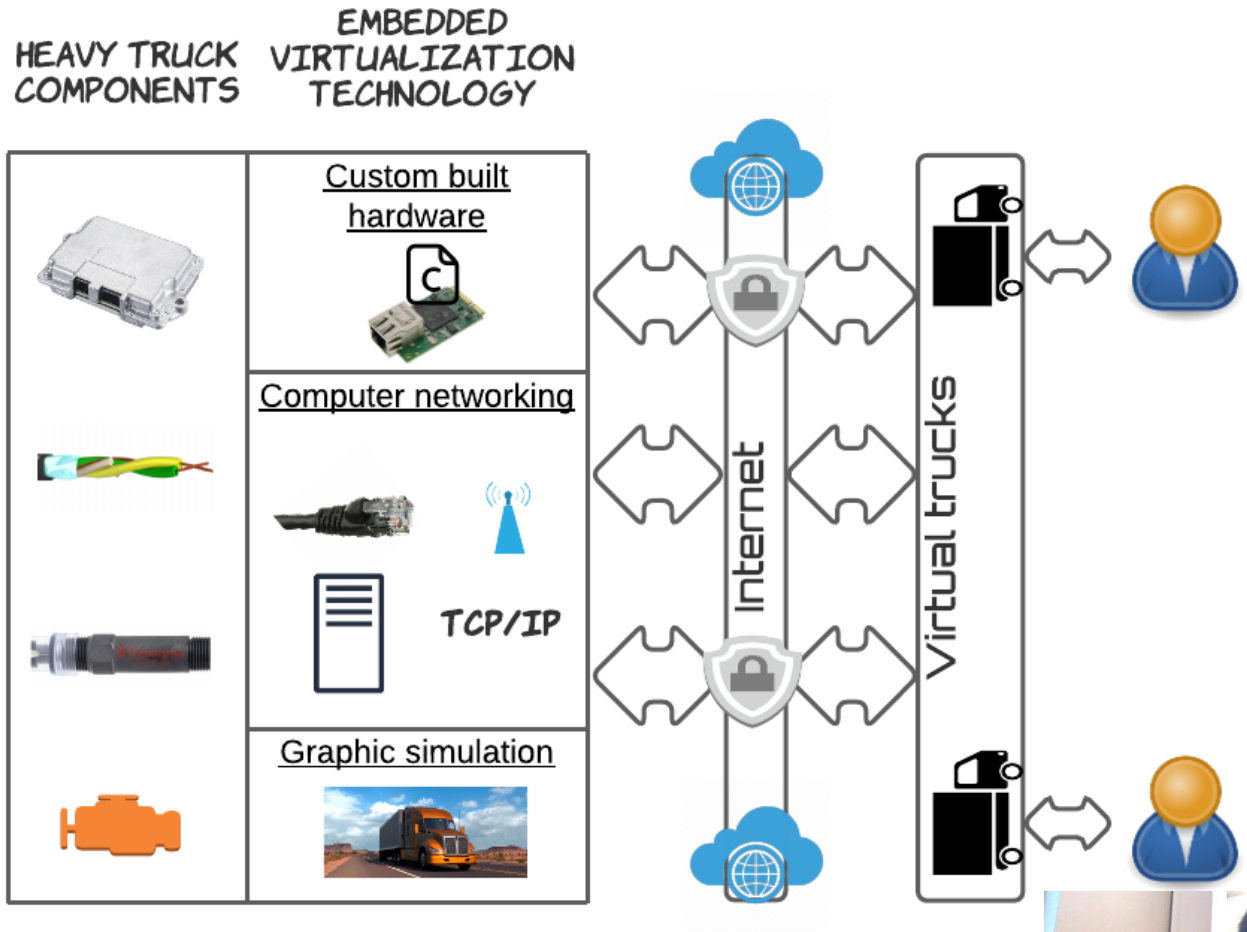
Fiat Chrysler recalls 1.4 million cars after Jeep hack

24 July 2015

Needs Analysis

- A scalable, cost-effective approach to testing cybersecurity posture of heavy vehicles.
- Use actual heavy vehicle electronic control units (ECUs)
  - Incorporate actual logic
  - Accommodate testing of proprietary software
- Remote access for a distributed workforce.
- Reconfigurable systems

Software Defined Truck



Programmable trucks in the cloud for the people, by the peop

Contribution

- A novel idea
  - The concept of Software defined network (SDN) exists
  - Previous proposals on car-specific testbeds [3, 4] do not address reconfigurability
  - A handful of proposals [1,2] to extend SDN for automotive systems
    - Lack of structured requirement engineering
    - Reprogrammable networks only
    - Require modification of the target ECUs
    - Do not consider security implications like information leakage and intentional destruction of testbed components
  - An amalgamation of technologies, some existing and others created in-house
- This preliminary report presents
  - A mix of the conceptual and practical underpinnings of designing, implementing, and maintaining a software defined truck through its lifecycle
- A generalization is the software-defined vehicle but the focus here is on heavy
  - The concepts of SDT can be used/extended by other embedded testbed providers
  - Can be used for other research purposes aside cyber-security

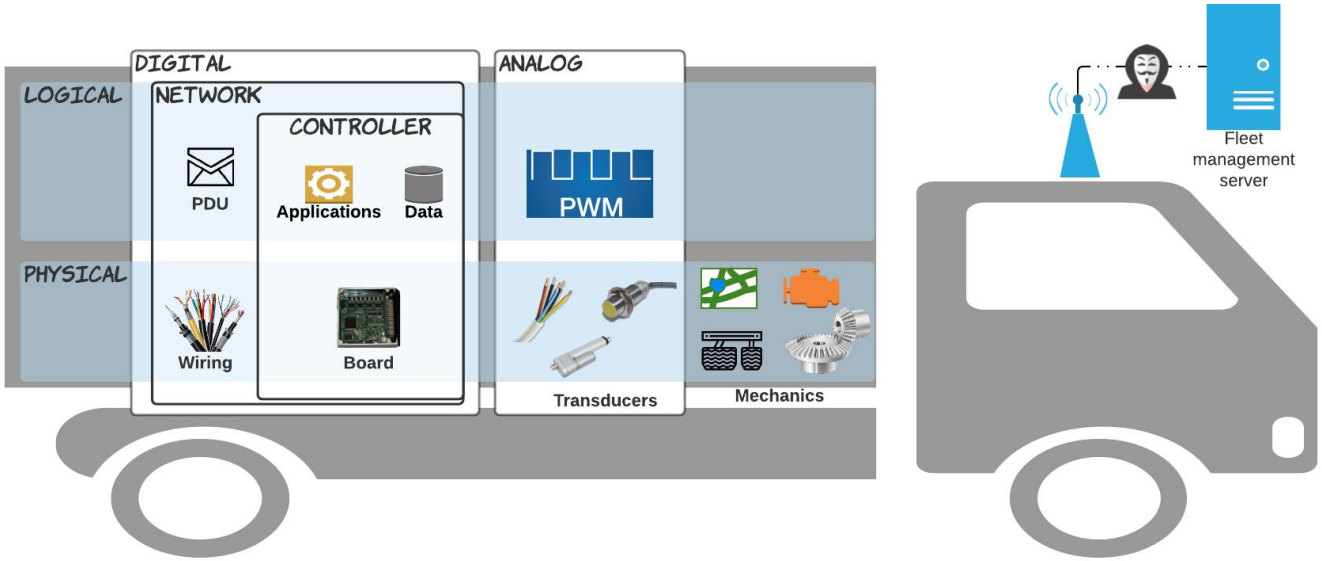
Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
5. Future work

Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
5. Future work

Embedded Electronics



- A congregation of
  - Mechanics
  - Analog transducers used to sense and manipulate mechanicals parameters
  - Electronic control units (ECU) that
    - Receive analog input from sensors
    - Process corresponding digital data
    - Send analog output through actuators

- Communication with external servers
  - Diagnostic and management
  - Predominantly chosen remot vector

Embedded Communication

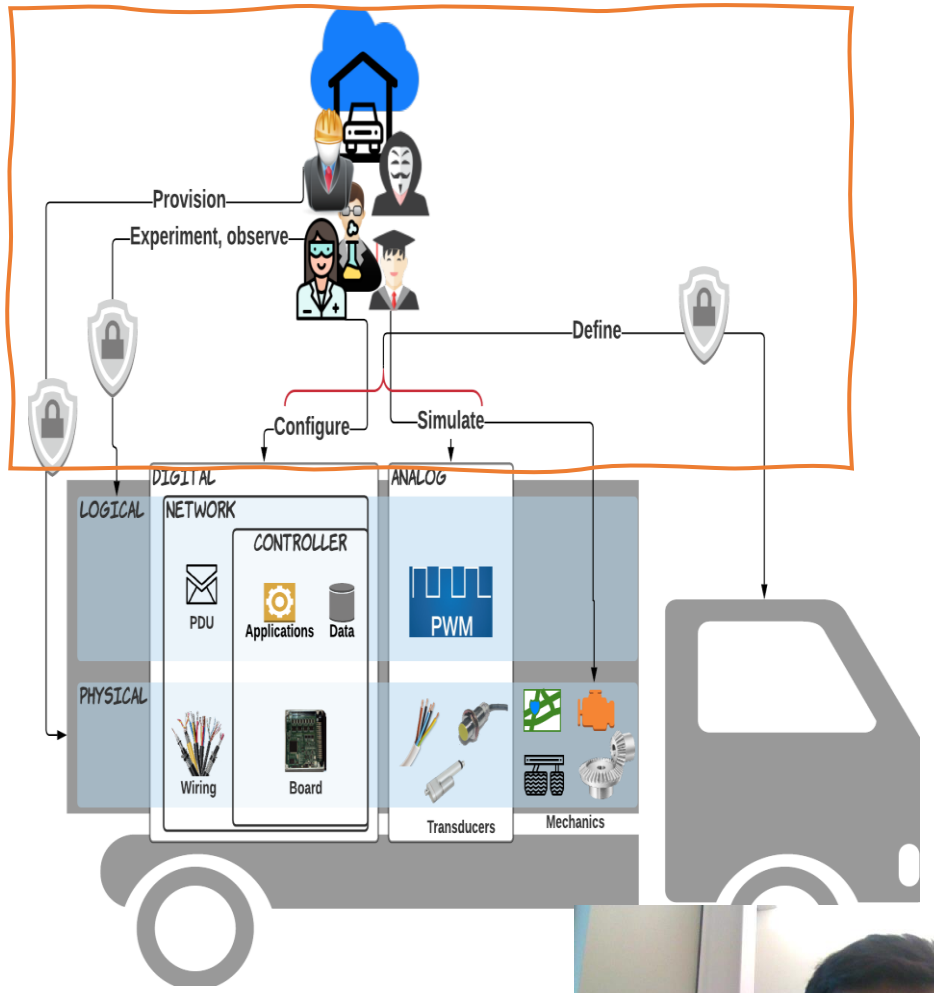
- ECUs communicate with each over physical wiring using sequence of bits
- A set of ECUs communicating with each other forms a network
  - Typically obeys communication specifications from the Controller Area Network (CAN) 2.0 standards
    - CAN is a **low-latency** serial communication protocol
  - SAE-J1939 specify high-level networking constructs including message formats
    - Messages are carried by CAN
    - Messages are made of the forms **<ID><Data>**
- Networks can be interconnected using bridges or gateways

Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
5. Future work

Use case 1,2 of 4

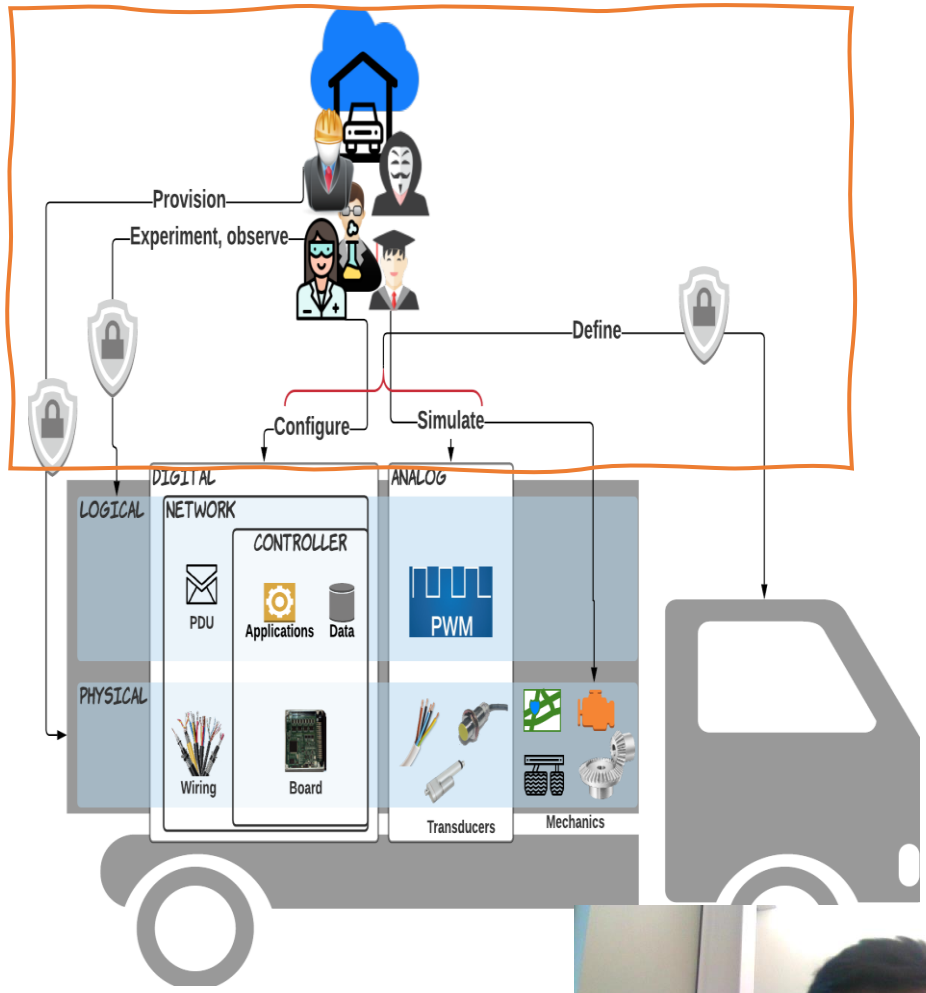
- **Use case**: Provision
  - **Actor**: Anybody with the possession of a fitting device or network or even a truck, typically hardware vendors, garage personnel, university research groups etc.
  - **Discussion**: May make some features unavailable if required
- **Use case**: Configure
  - **Actor**: Anybody, typically researchers, tutors, students, hackers etc.
  - **Discussion**: No physical manipulation, remote only



"*For the people, by the people*": Unlike r
proposals, does not rely on a single poir
delivery instead this is **crowdsourced**

Use case 3,4 of 4

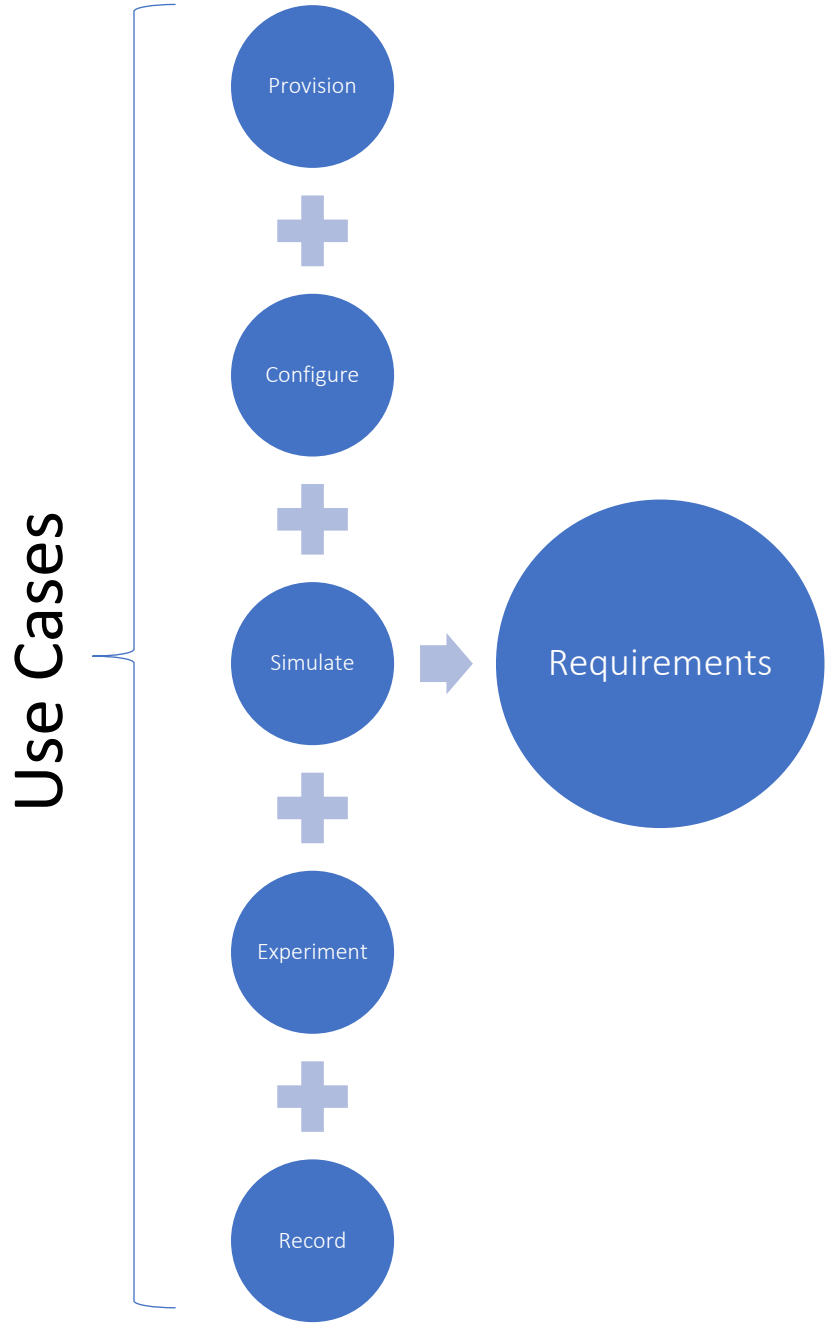
- **Use case**: Simulate
  - **Actor**: Anybody, typically researchers, tutors, students, hackers etc.
  - **Discussion**: Best-case replacement for physically unavailable components; e.g. transducers, driver, driving environment etc.
- **Use case**: Experiment, observe
  - **Actor**: Anybody, typically researchers, tutors, students, hackers etc.
  - **Discussion:** Targets can be (configured) real-world hardware or simulations



"*For the people, by the people*": Unlike r
proposals, does not rely on a single poir
delivery instead this is **crowdsourced**

Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
5. Future work

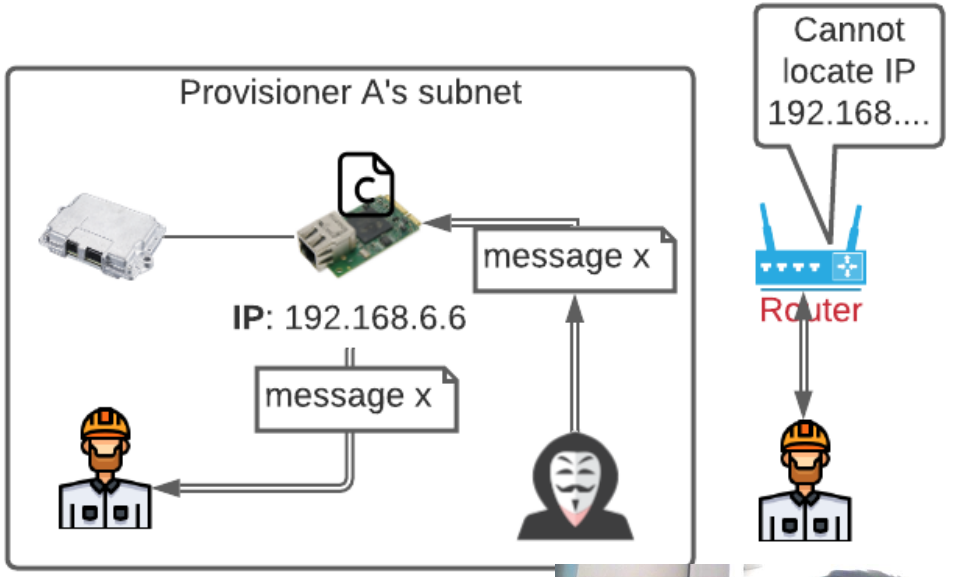


High-level Requirements for the Software Defined Truck

1. Facilitating real-world testing on a remotely accessible testbench
2. Being able to crowdsource
3. Being able to reconfigure and simulate truck system components
4. Facilitate application development

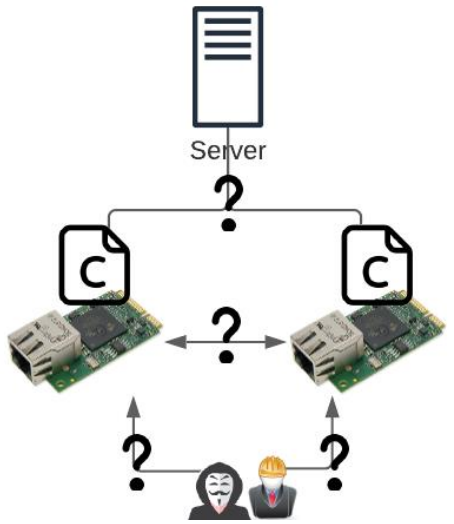
1. Facilitating real-world testing on a remotely accessible testbench

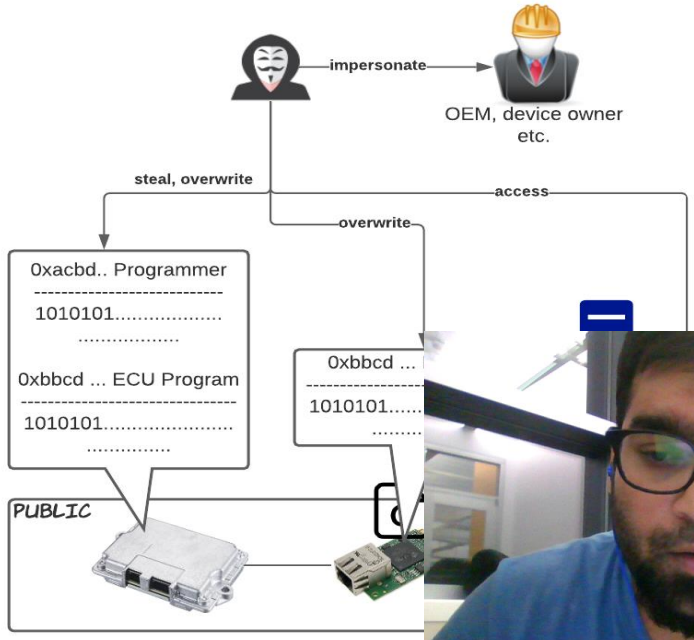
- **Functional challenge**:
  Private local area networks (LANs) are not reachable from the outside.
  - Embedded systems are configured to access the Internet through private IP addresses and Network Address Translation (NAT)
  - Amazon web-services (AWS) provides a framework for embedded device connectivity and management, but...
    - Not automotive specific
    - Is not open-sourced
- **Resource Access Control challenge**:
  Exclusive access to devices being actively used in experiments to ensure experiments do not interfere with each other [5]



2. Make crowdsourced systems publicly accessible

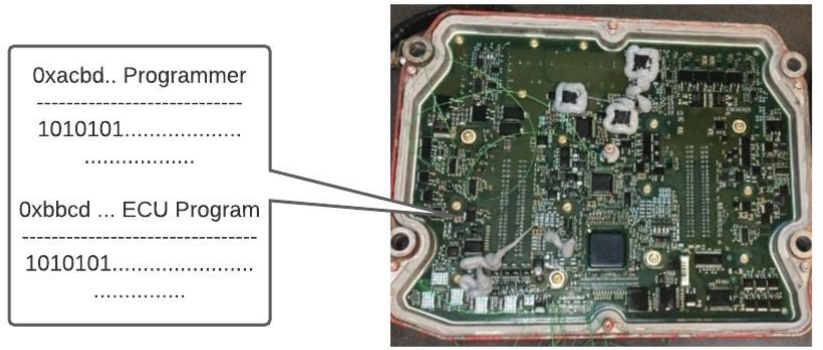
- **Functional challenges**:
  - Crowdsourcing requires a homogenous set of steps to set up and maintain participation
  - Testbench components need to interoperate and communicate with the SDT framework
- **Security challenges:**
  - Unintended use of system reserved for private usage
  - Information leakage; e.g. intellectual property, identifying information [5]
  - Intentional damage to system [3,5]
  - Pivoting external attacks [5]
  - Identity spoofing
    - Can be used to bypass all other security controls

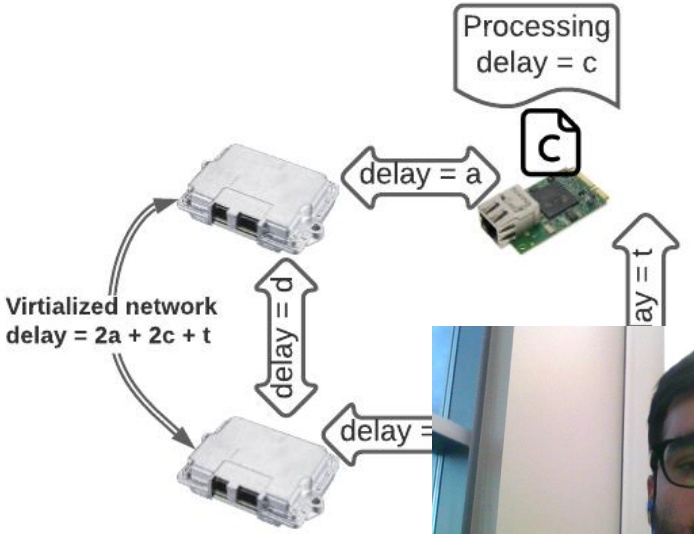




3. Being able to reconfigure and simulate truck system components

- **Functional challenge:**
  Digital components on the truck can be reconfigured by altering contained data but data items are not readily accessible
    - Requires complicated procedures such as finding debug ports, manual soldering and reflashing [6]
- **Availability challenge:**
  May become dysfunctional or “bricked” in the process thus increasing cost of maintenance
    - ECU firmware is intellectual property of the device manufacturer
- **Quality assurance challenge:**
  Achieving a level of acceptable fidelity of the testbed [5]
    - The testbench must resemble a real-world MHD vehicle system.
    - Latency between the user’s application and the physical testbench must be minimal
    - Simulating some components may will degrade fidelity
        - Virtualized networks will bear non-zero latency overhead



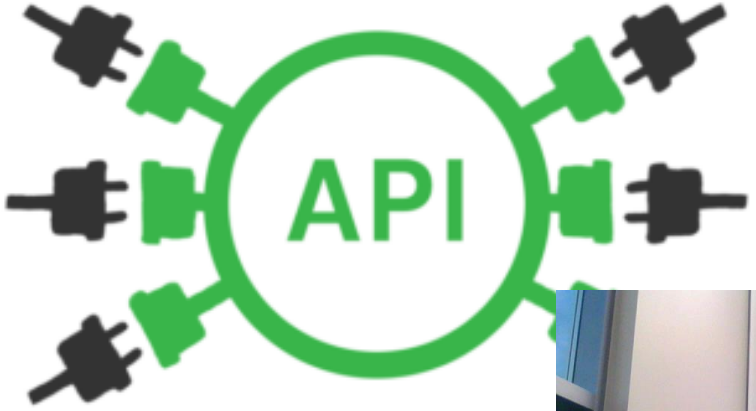


4. Facilitate application development

- **Functional challenges**

  Related to software engineering and lifecycle management

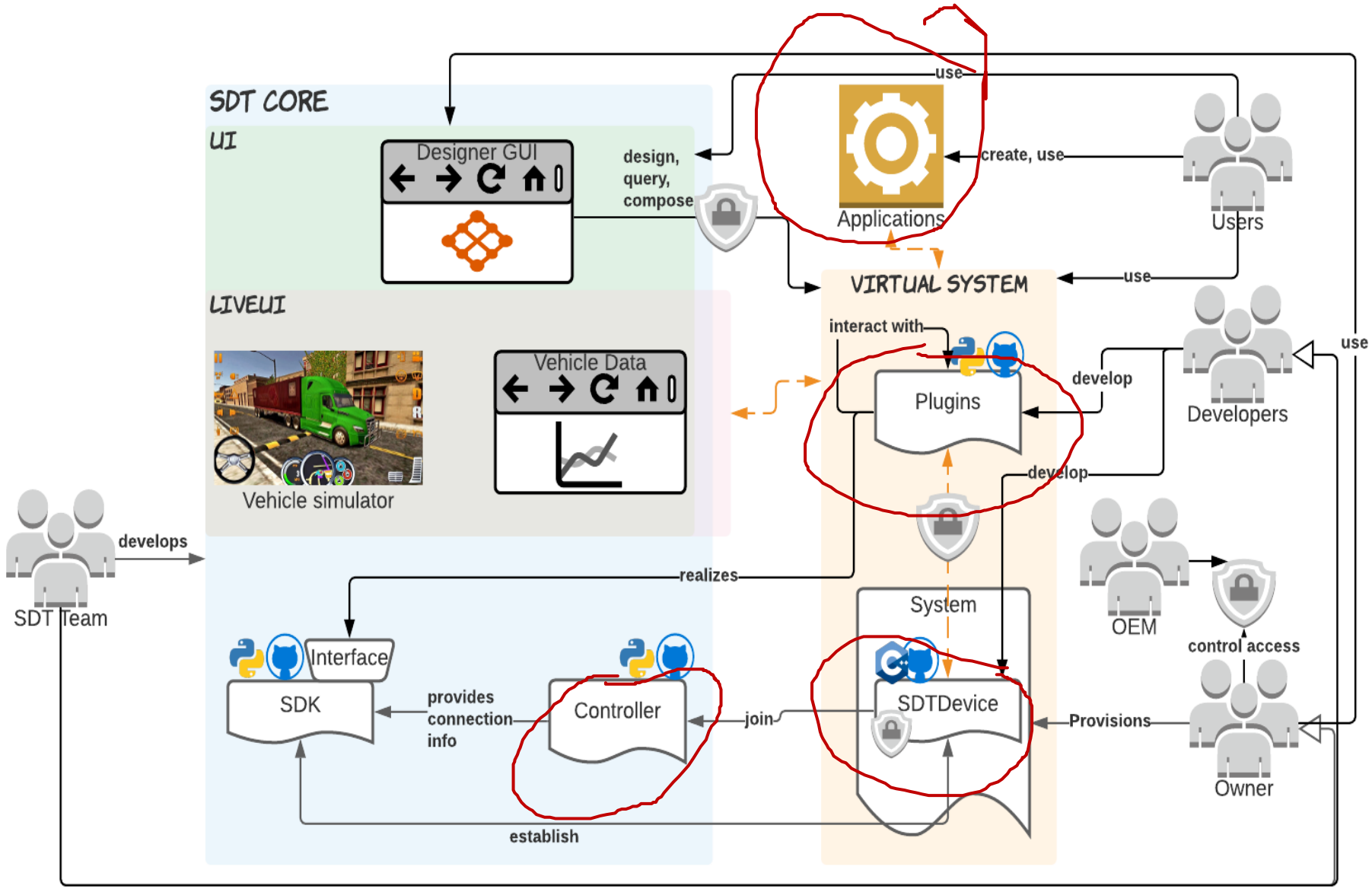
  - Provision of application programming interfaces
  - Provision of pre-defined classes
  - Provision of extendable templates



Organization

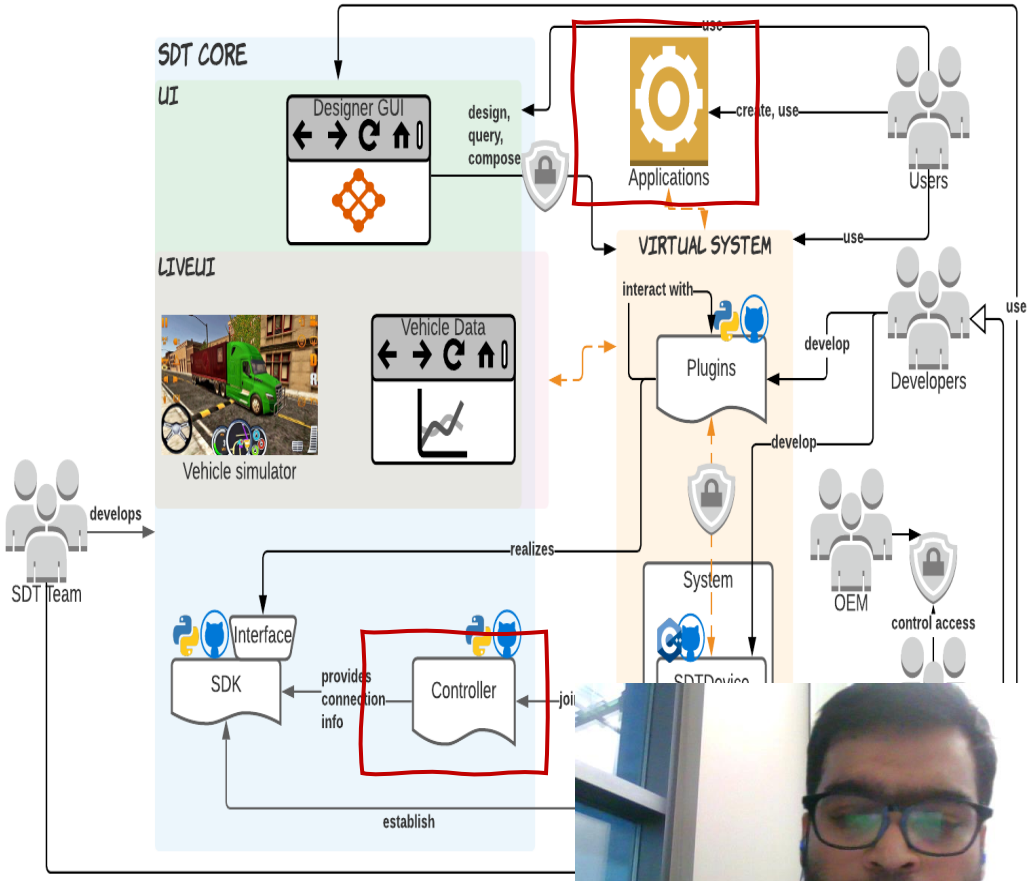
1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
5. Future work

The Software Defined Truck Ecosystem



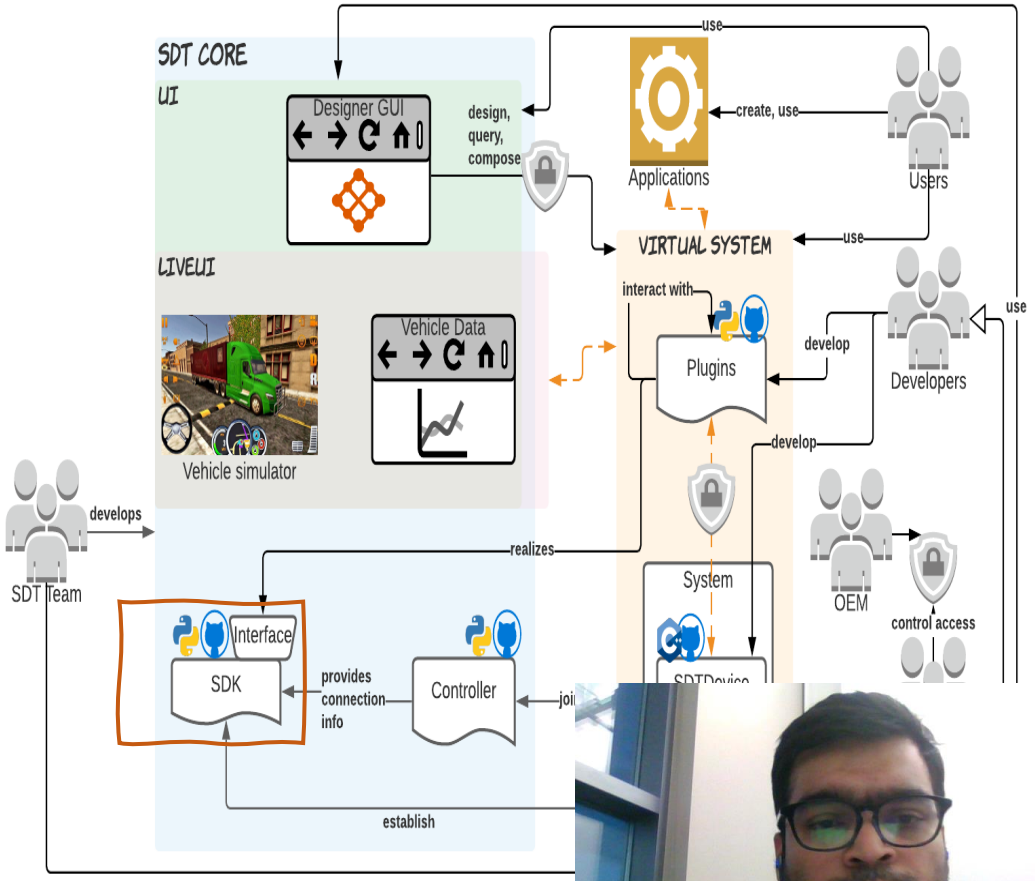
Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
    1. Component Description
        1. SDT core
            1. Software development kit (SDK)
            2. User interface (UI)
        2. SDT Virtualization support
            1. SDT Hardware
            2. SDK Plugins
    2. SDT Workflow
5. Future work

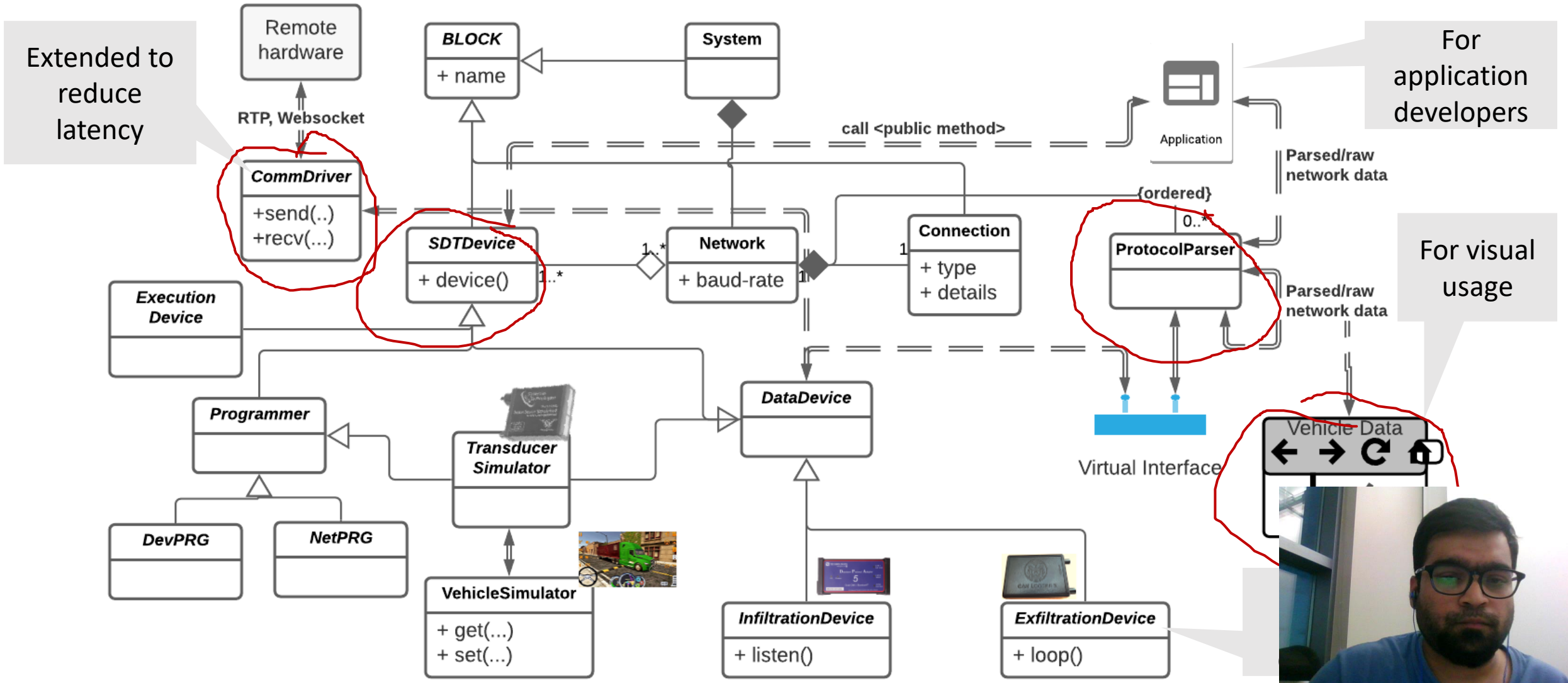


Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
    1. Component Description
        1. SDT core
            1. Software development kit (SDK)
            2. User interface (UI)
        2. SDT Virtualization support
            1. SDT Hardware
            2. SDK Plugins
    2. SDT Workflow
5. Future work

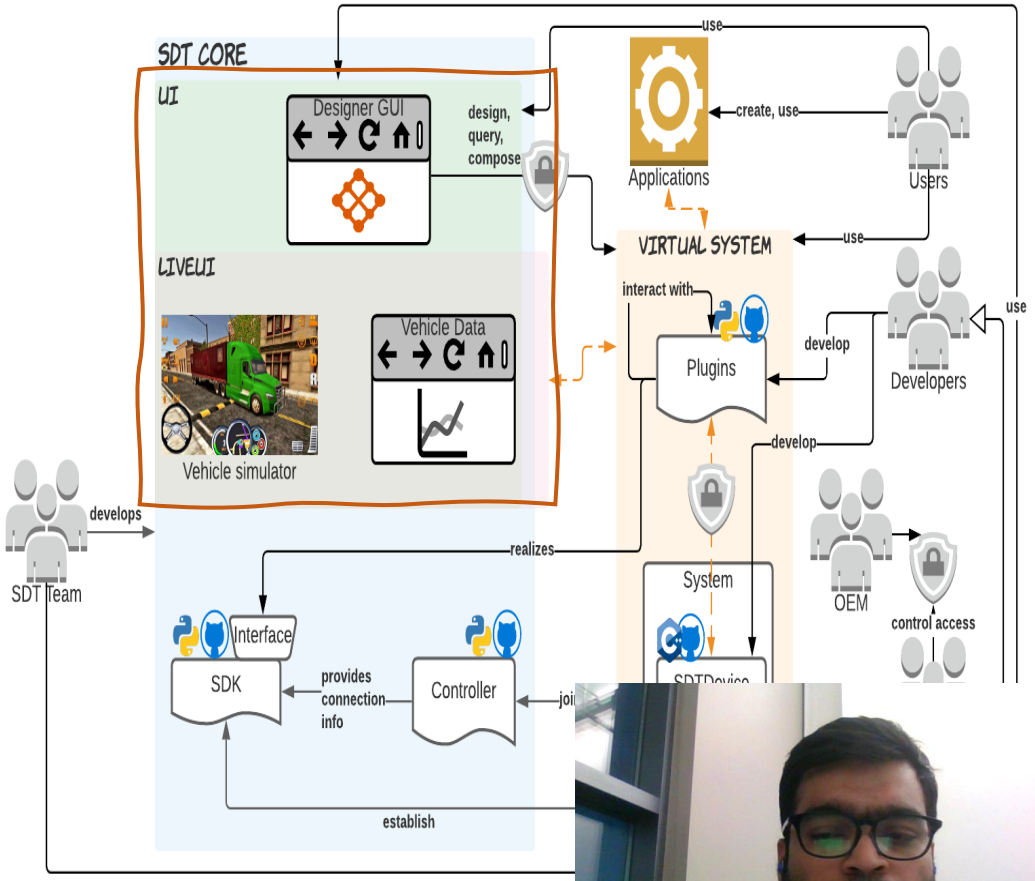


Overview


Extended to reduce latency
Remote hardware
RTP, Websocket
BLOCK
+ name
System
call <public method>
Application
For application developers
CommDriver
+send(..)
+recv(...)
SDTDevice
+ device()
1..*
Network
+ baud-rate
1
1..*
Connection
+ type
+ details
{ordered}
0..*
ProtocolParser
Parsed/raw network data
Parsed/raw network data
For visual usage
Execution Device
Programmer
DevPRG
NetPRG
Transducer Simulator
DataDevice
Virtual Interface
Vehicle Data
VehicleSimulator
+ get(...)
+ set(...)
InfiltrationDevice
+ listen()
ExfiltrationDevice
+ loop()

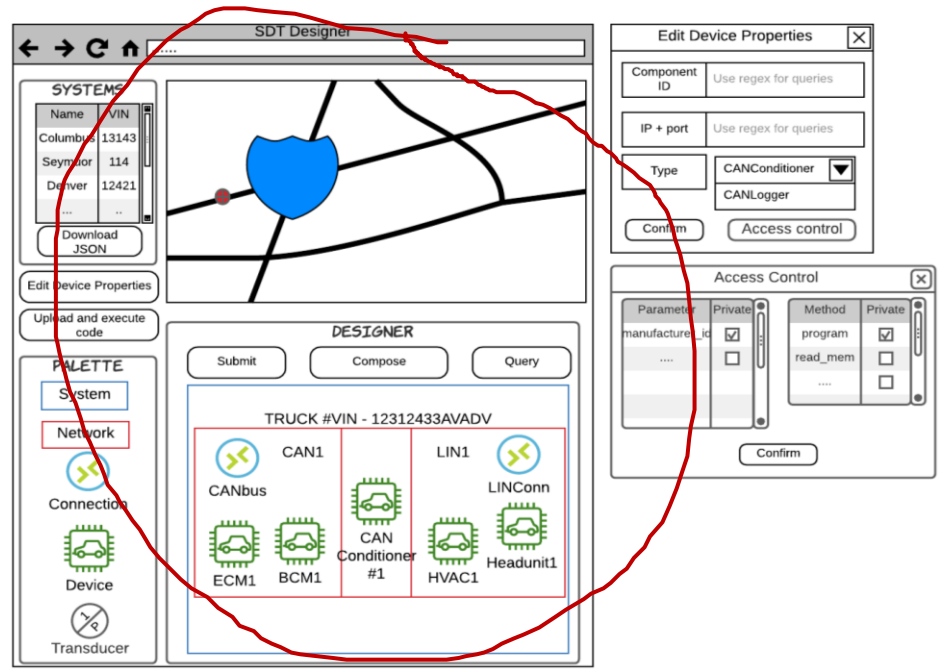
Organization

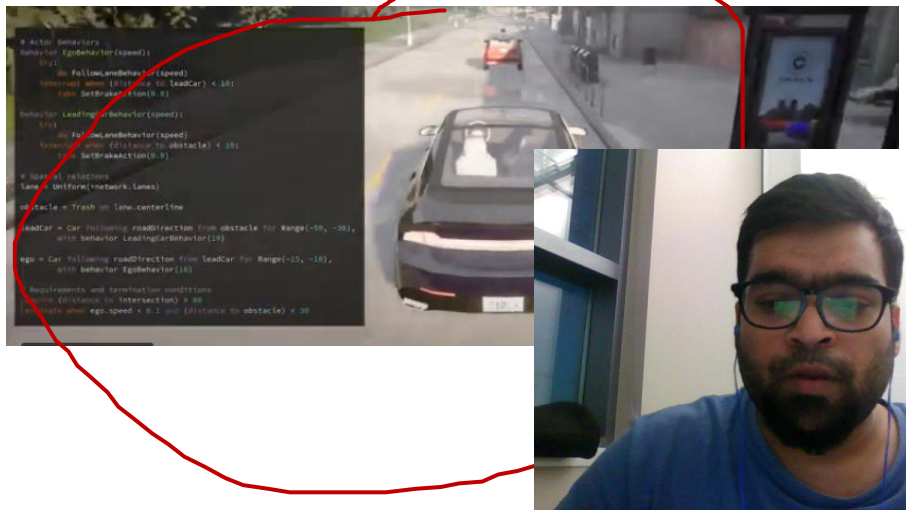
1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
    1. Component Description
        1. SDT core
            1. Software development kit (SDK)
            2. User interface (UI)
        2. SDT Virtualization support
            1. SDT Hardware
            2. SDK Plugins
    2. SDT Workflow
5. Future work



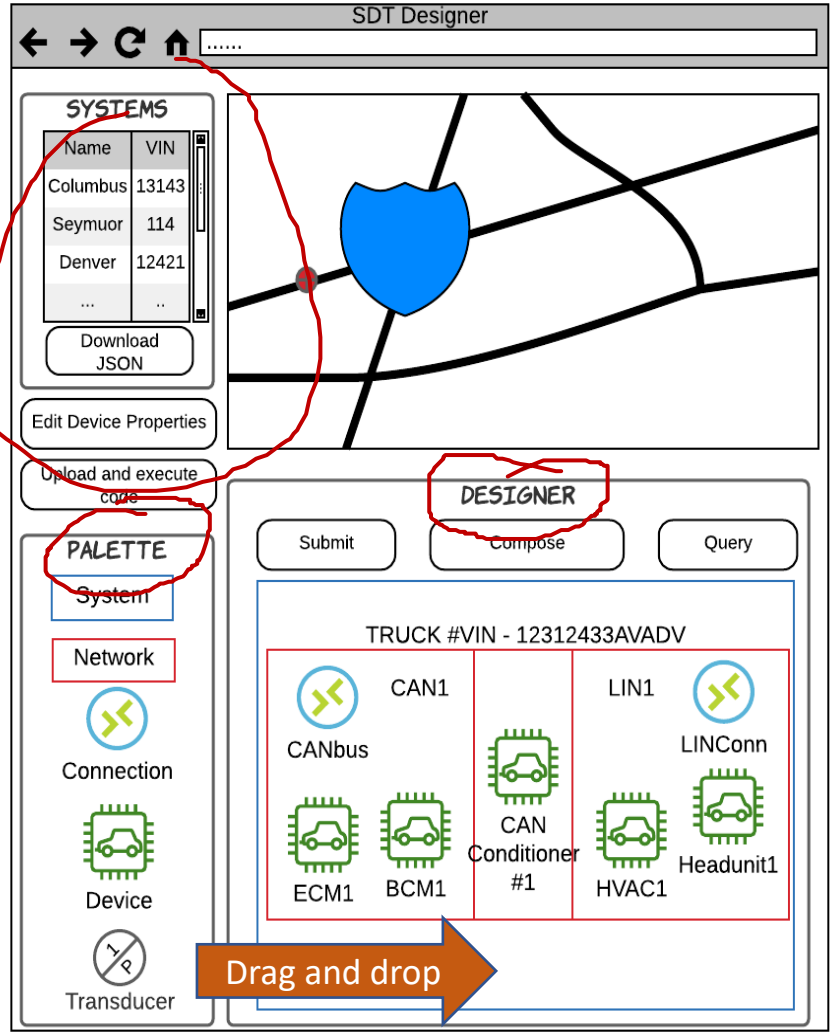
Overview

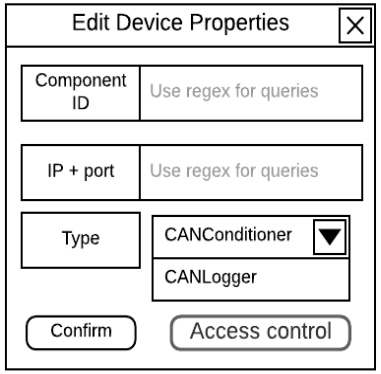
- Designer GUI
  - User's gateway to SDT's backend controller
  - Query and provision systems
- Rendering GUI
  - Vehicle simulator
    - The CARLA (Dosovitskiy et al., 2017) simulation engine is used.
    - CARLA provides a Python application programming interface (API) and enables simulating the driving environment aside the actual vehicle. This can be especially critical for automated driving related research on an SDT.
  - Data visualization interface is provided to observer high-level abstractions from network data

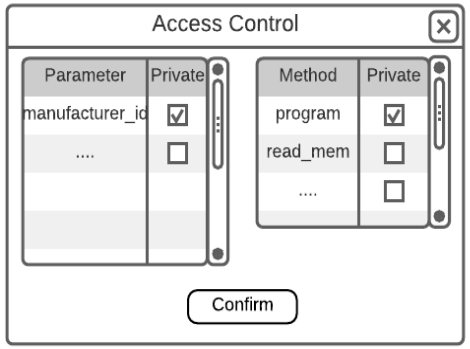


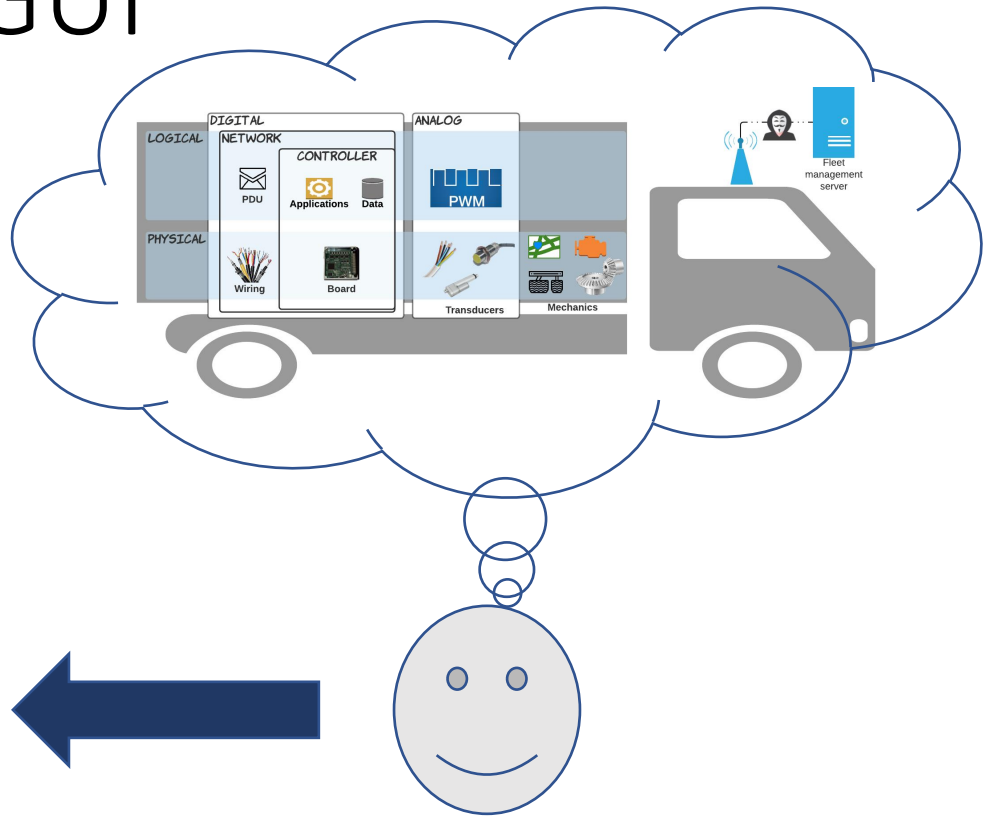


Description of the Designer GUI



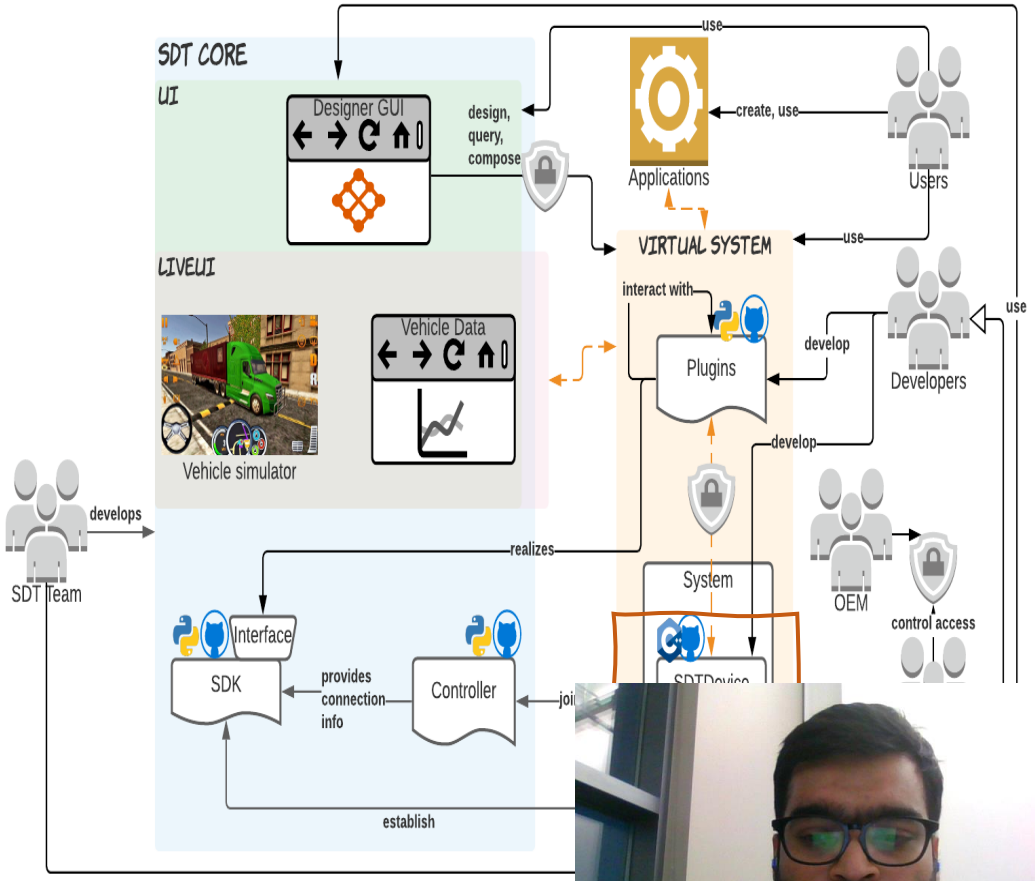






Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
    1. Component Description
        1. SDT core
            1. Software development kit (SDK)
            2. User interface (UI)
        2. SDT Virtualization support
            1. SDT Hardware
            2. SDK Plugins
    2. SDT Workflow
5. Future work

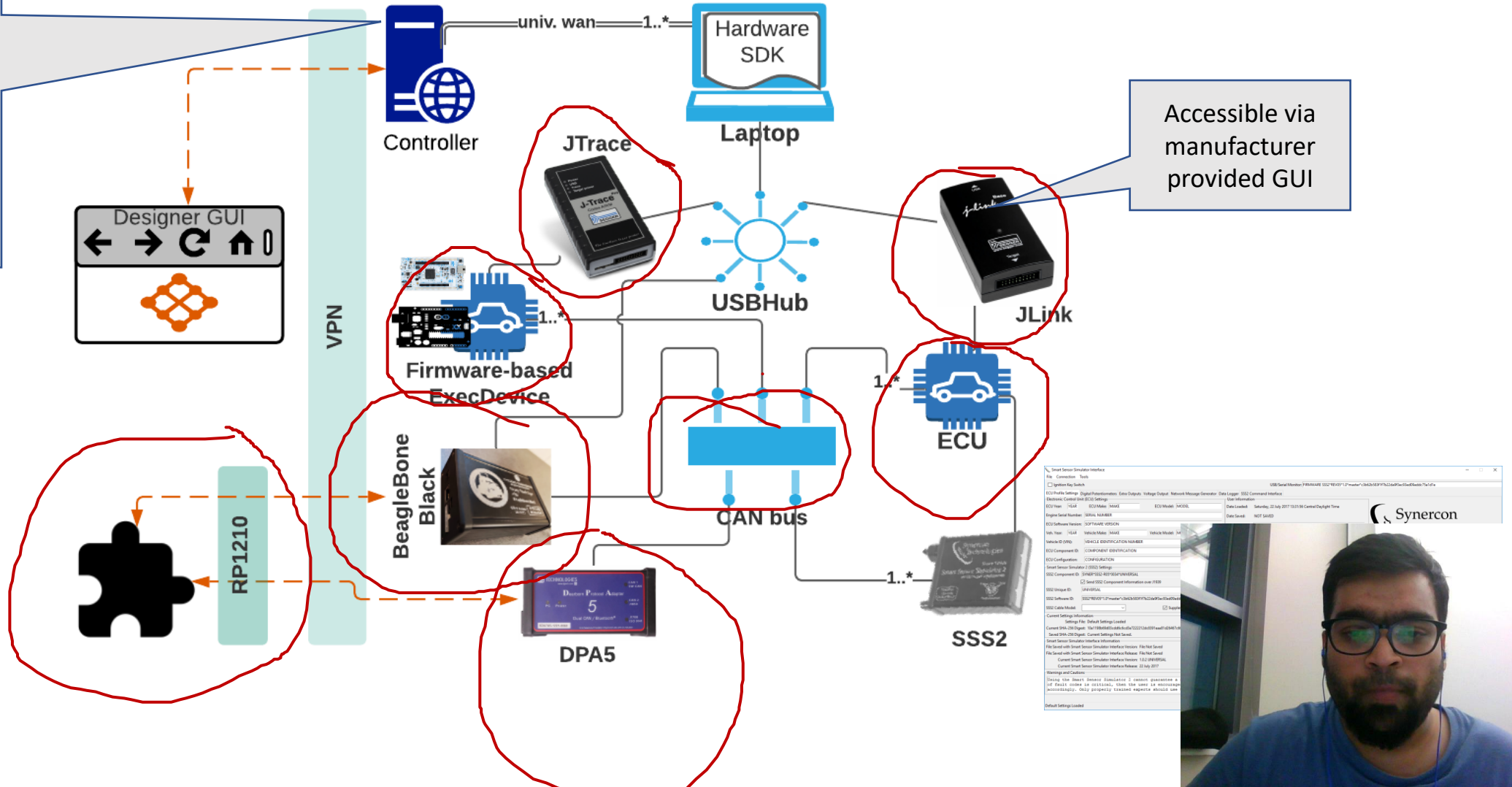


Overview

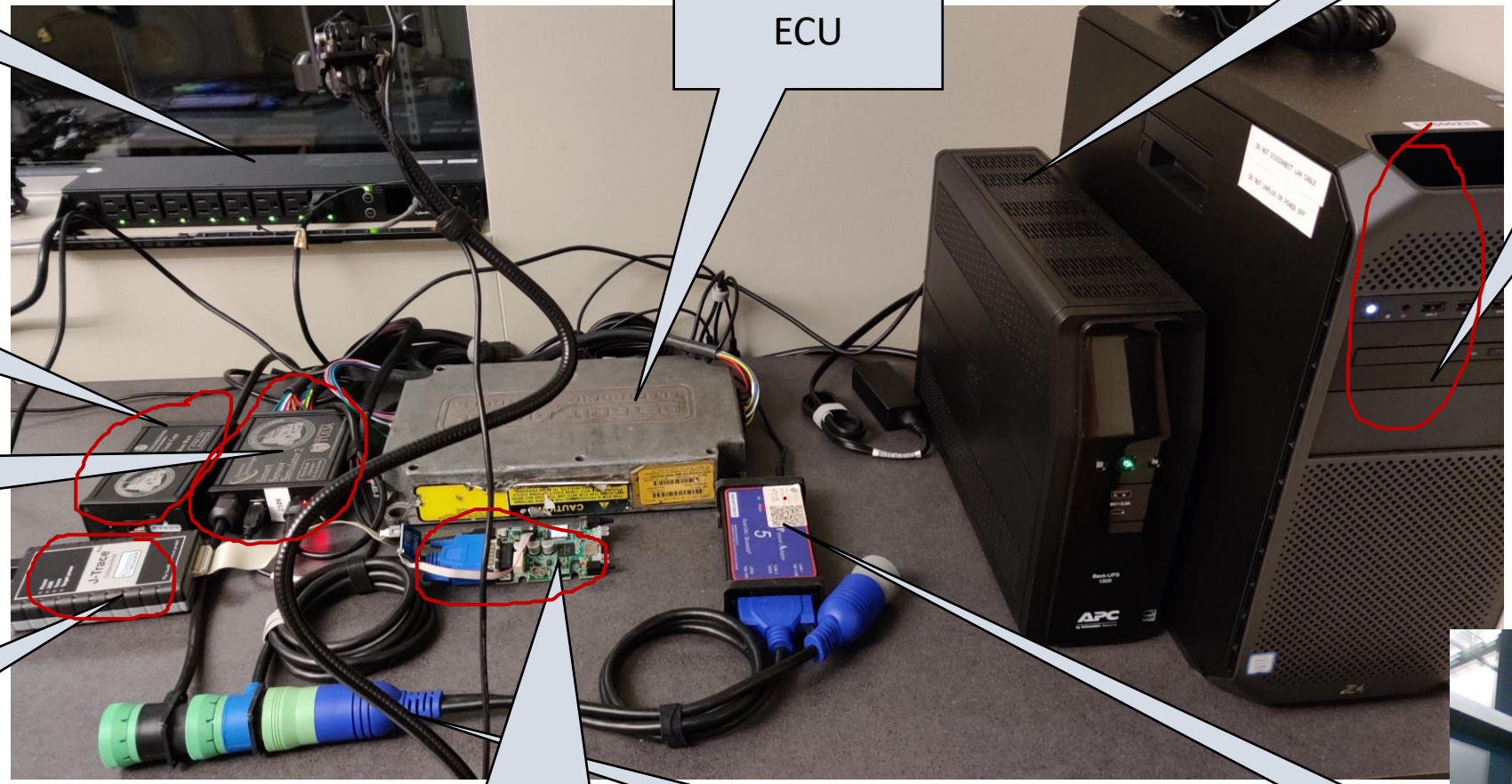
- Hardware counterparts of the software abstractions from the SDK.
- Because SDT follows a crowdsourced model the exact model of hardware cannot be ascertained beforehand.
  - Need the hardware SDK to be installed
  - We describe the preliminary in-house testbench model.
  - The model follows SYSML semantics i.e. blocks and connections with multiplicities.

Description

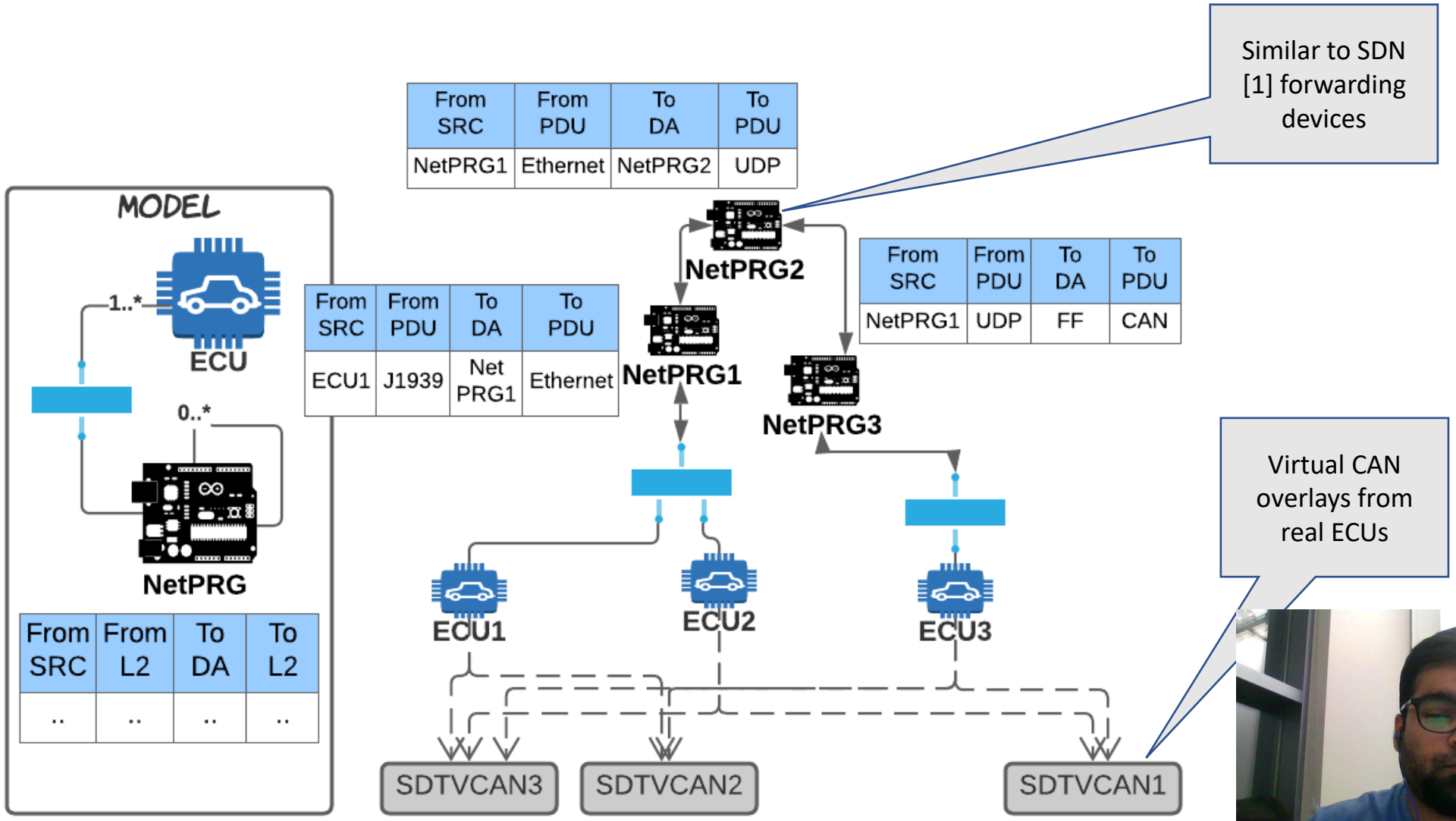
Ubuntu 20.04 controller that runs on 4 core non-SMT Intel(R) Xeon(R) CPU E5-2407 0 @ 2.20GHz processor.



An Instance At The Laboratory

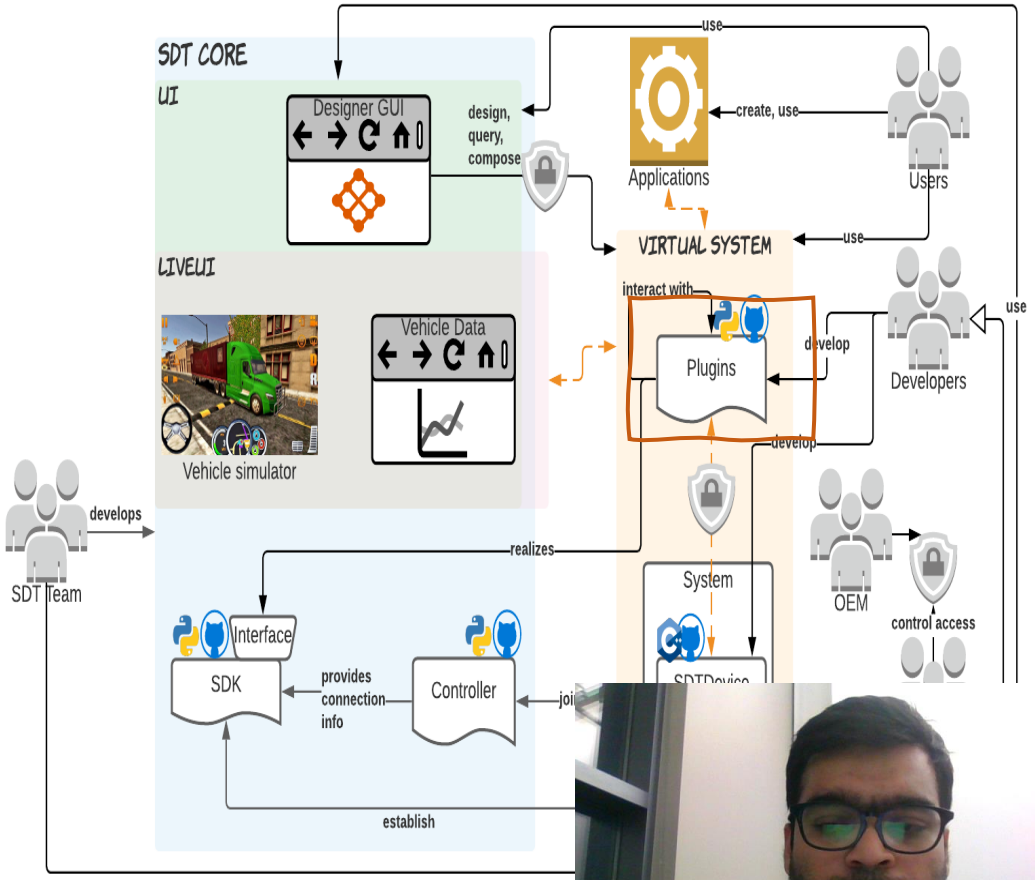


Description (Network programmers)



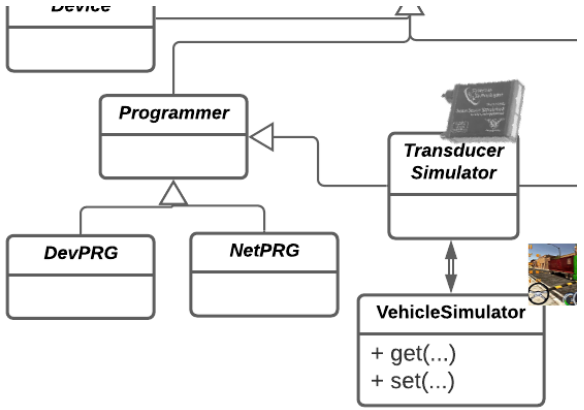
Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
    1. Component Description
        1. SDT core
            1. Software development kit (SDK)
            2. User interface (UI)
        2. SDT Virtualization support
            1. SDT Hardware
            2. SDK Plugins
    2. SDT Workflow
5. Future work



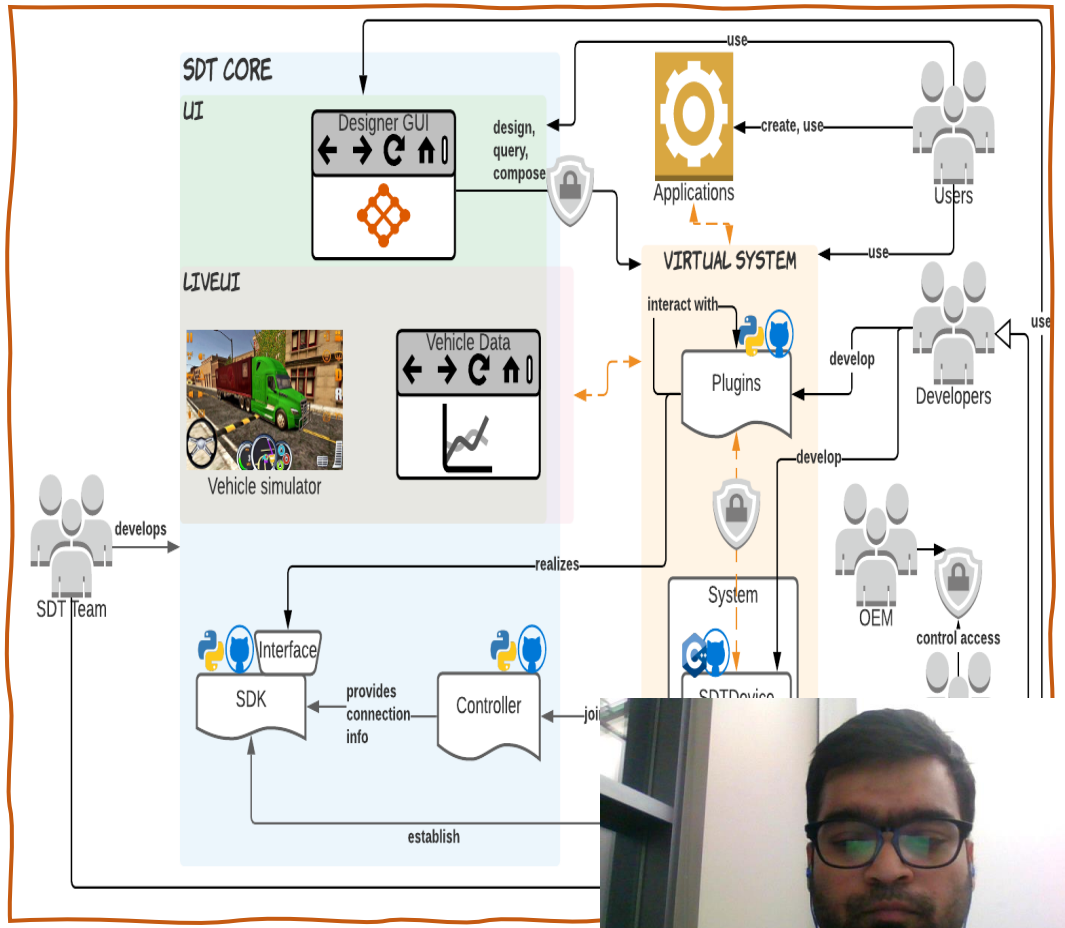
Description

- Extend abstract classes (named in italics) from the SDK
  - Abstracts away complexities of interacting with the hardware
  - Converts different hardware interfaces into SDT standard API
    - Ensures interoperability between hardware developed by different organizations
  - E.g., the smart sensor simulator GUI extends the TransducerSimulator class
- Plugins for
  1. SDT devices
     - Hardware developers can contribute by providing software level abstractions for their devices
     - SDT Team is creating some standardized APIs
  2. Protocol parsers
     - Abstract away the low-level details of network data processing
  3. Communication drivers
- Interact with physical hardware using standard networking technology

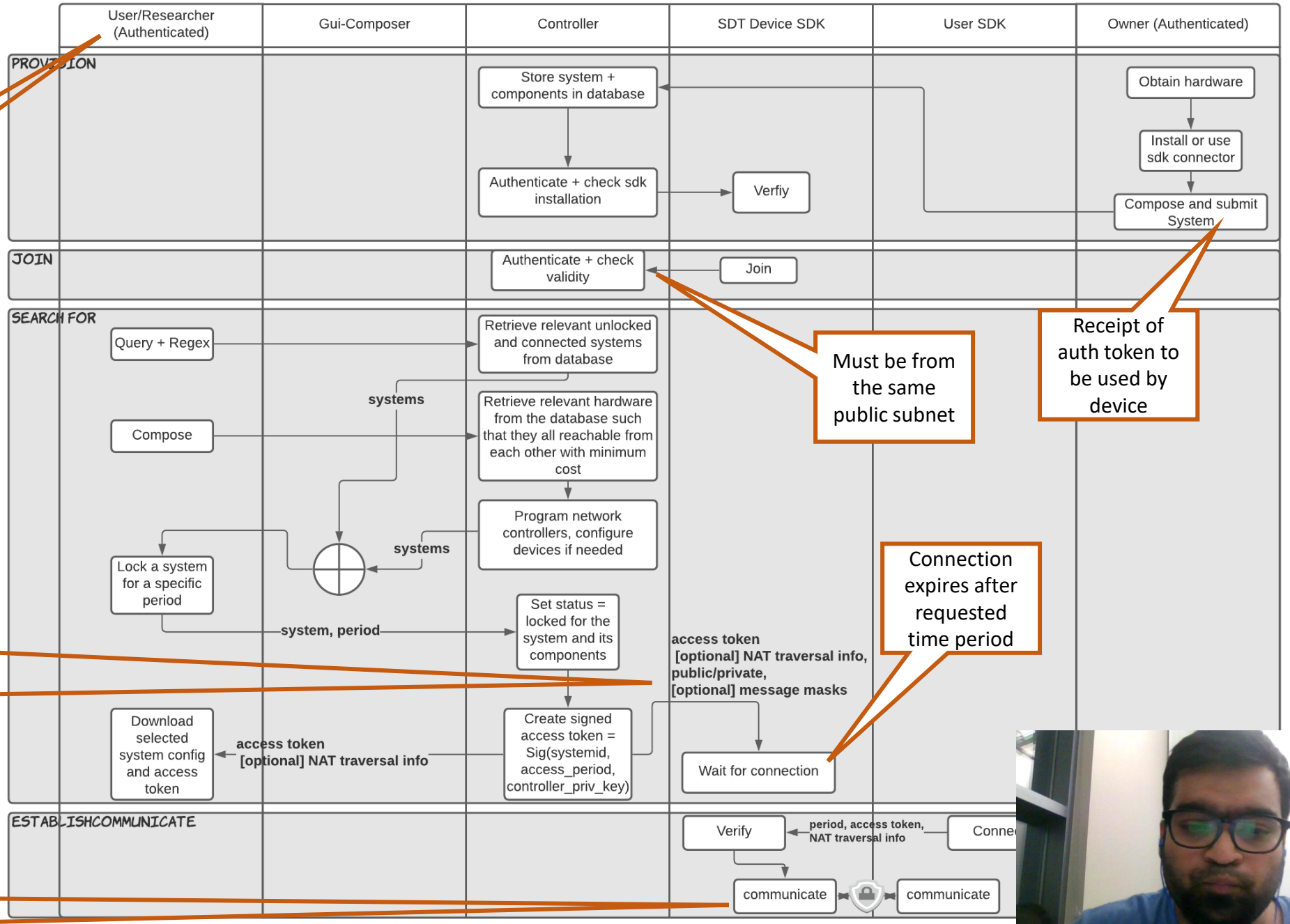


Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
    1. Component Description
        1. SDT core
            1. Software development kit (SDK)
            2. User interface (UI)
        2. SDT Virtualization support
            1. SDT Hardware
            2. SDK Plugins
    2. SDT Workflow
5. Future work



Activity Diagram



Public identity providers or affiliated emails

Must be from the same public subnet

Receipt of auth token to be used by device

Connection expires after requested time period

Example ID mask:
FF02<transmitter-address>
Corresponding data mask:
00000000FFFF

Techniques similar to online gaming

Organization

1. Background on Heavy Truck Electronics
2. Target Users and Use Cases
3. High-level Requirements and Challenges
4. (Preliminary) Design Details
5. Future work

- Further analysis and evaluation of the proposed models
  - Advancing our in-house testbench model
- Building remote API bridges between OEM provided programming tools and the hardware SDK.
- Investigating the usability of available approaches to address functional requirements like
  - Facilitating NAT traversal if required
  - Usability of the RT(S)P protocols to create overlays on CAN networks

References

1. Doering, Michael and Wagner, M. (2017) ‘Retrofitting SDN to classical in-vehicle networks: SDN4CAN’.
2. Halba, K. and Mahmoudi, C. (2018) ‘In-Vehicle Software Defined Networking’, in Proceedings of the 2nd International Conference on Information System and Data Mining. New York, NY, USA: ACM, pp. 93–97. doi: 10.1145/3206098.3206105
3. Everett, C. E., & McCoy, D. (2013). OCTANE: Open car testbed and network experiments bringing cyber-physical security research to researchers and students. *6th Workshop on Cyber Security Experimentation and Test, CSET 2013*
4. Daily, J., Gamble, R., Moffitt, S., Raines, C., Harris, P., Miran, J., ... & Johnson, J. (2016). Towards a cyber assurance testbed for heavy vehicle electronic controls. SAE International Journal of Commercial Vehicles, 9(2), 339-349.
5. Benzel, T., Braden, B., Kim, D., Neuman Anthony Joseph, C., Sklower Ron Ostrenga, K., Schwab, S., Braden, R., Neuman, C., Joseph, A., & Sklower, K. (2006). *EXPERIENCE WITH DETER: A TESTBED FOR SECURITY RESEARCH*
6. Carey, M., & Bathurst, R. (2013). Hacking Embedded Devices (Doing Bad Things to Good Hardware). *DEFCON 21*. https://www.defcon.org/images/defcon-21/dc-21-presentations/Phorkus-Evilrob/DEFCON-21-Phorkus-Evilrob-Hacking-Embedded-Devices-Bad-things-to-Good-hardware.pdf
7. Society of Automotive Engineers. (2016) Cybersecurity Guidebook for Cyber-Physical Vehicl SAE-J3061. https://www.sae.org/standards/content/j3061_201601/

Thank you for your time

QUESTIONS PLEASE?