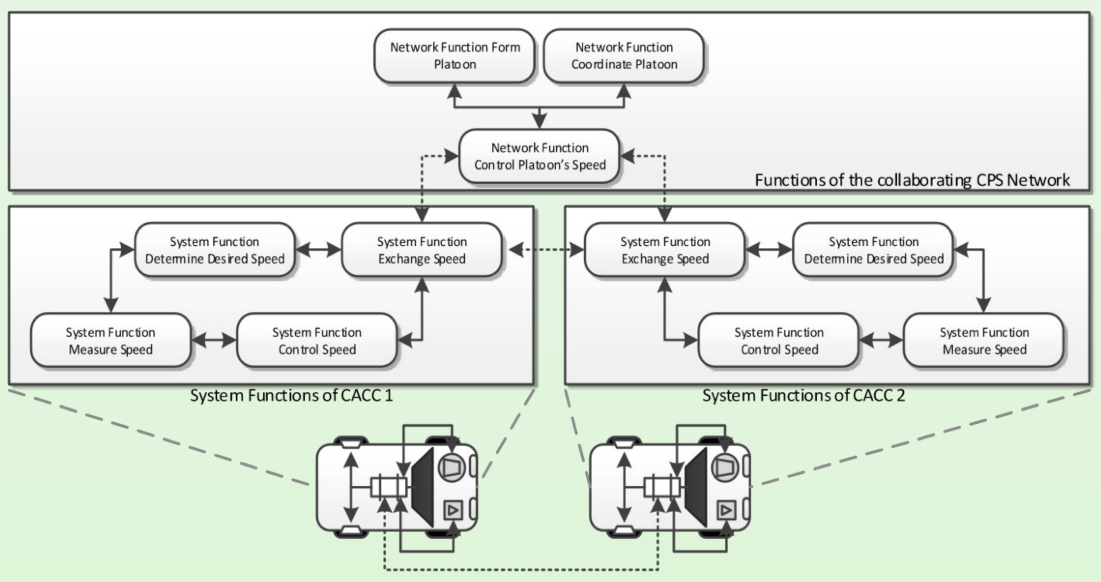
Implementation of a Cyber-Physical Autonomous Vehicle Testbed

Gregory Maldonado, Dr. Bastian Tenbergen

OSWEGO
STATE UNIVERSITY OF NEW YORK

Introduction

Autonomous driving is a grand challenge for technology development, not only in popular science [1], but also the software engineering research community [2]. An emerging technology called "Cyber-Physical Systems" (CPS) will enable future developments in this area [3, 4]. CPS are systems, which collaborate on tasks, no individual CPS can achieve alone ([3, 5]). On the one hand, CPS observe the environment through sensors and act upon the environment with actuators, much like Embedded Systems [6]. On the other hand, they communicate with other systems and interact with human users like Information Systems [3, 6]. CPS are special (i.e. what puts the "cyber-physicality" into the term) in that they collaboratively achieve goals, the individual systems cannot be designed to achieve and hence display emergent autonomous behavior [3, 6].

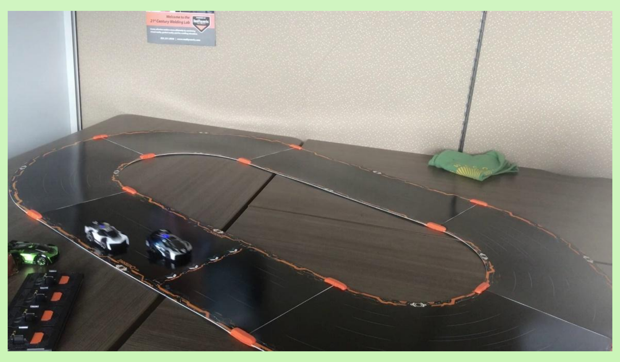


The figure above [6] shows the interaction of two cyber-physical automotive adaptive cruise controls (CACC). These systems collaboratively exchange desired vehicle speeds across the network. A vehicle can exchange an initial speed for the network and then can negotiate a speed change.

ANKI Overdrive Autonomous Vehicle Platform

To simulate CPS within autonomous vehicles, our research makes use of ANKI Overdrive smart "slot cars" as a surrogate platform for real cars. Below are pictures of two ANKI Overdrive vehicles on a track piece (left), the internal structure of one vehicle (middle), and two cars driving on a simplistic track (right). The ANKI vehicles drive straight and true on a predesigned track and communicate with other devices through Bluetooth. All vehicles differ in terms of acceleration, top speed, handling, brake force, etc., much like real cars do. This platform is particularly well suited for such a simulator due to its context-restricted nature. This means that the "driving" aspect is already implemented in the ANKI vehicles.





Project Goals

In this project, we want to develop a cyber-physical autonomous vehicle simulator using the Overdrive platform. We implemented the following features:

a) An Automotive Cyber-Physical System Network Protocol (ACPSNP) specification. ACPSNP is a behavior-oriented messaging protocol that allows vehicles on the same network to communicate and execute agreed upon behaviors.

b) A network that allows a vehicle to multicast a behavior to all other vehicles on the network. Multicasting to the network is a safety-critical function that will allow all vehicles to execute the behavior as soon as they receive the request to avoid collision.

c) A collaborative network that allows vehicles to communicate with each other for behavior consensus. The ability to negotiate behaviors will allow the network to accomplish tasks together ([3, 5]).

d) Simulation of cooperative traffic jam resolution [6] using the ACPSNP. The scenario is a vehicle in the front of the platoon slows down which effects the speed of all of the vehicles behind. The resolution to the traffic jam is too cooperatively propagate a speed reduction behavior. Once all vehicles have received the message, negotiation of an increased speed can begin.

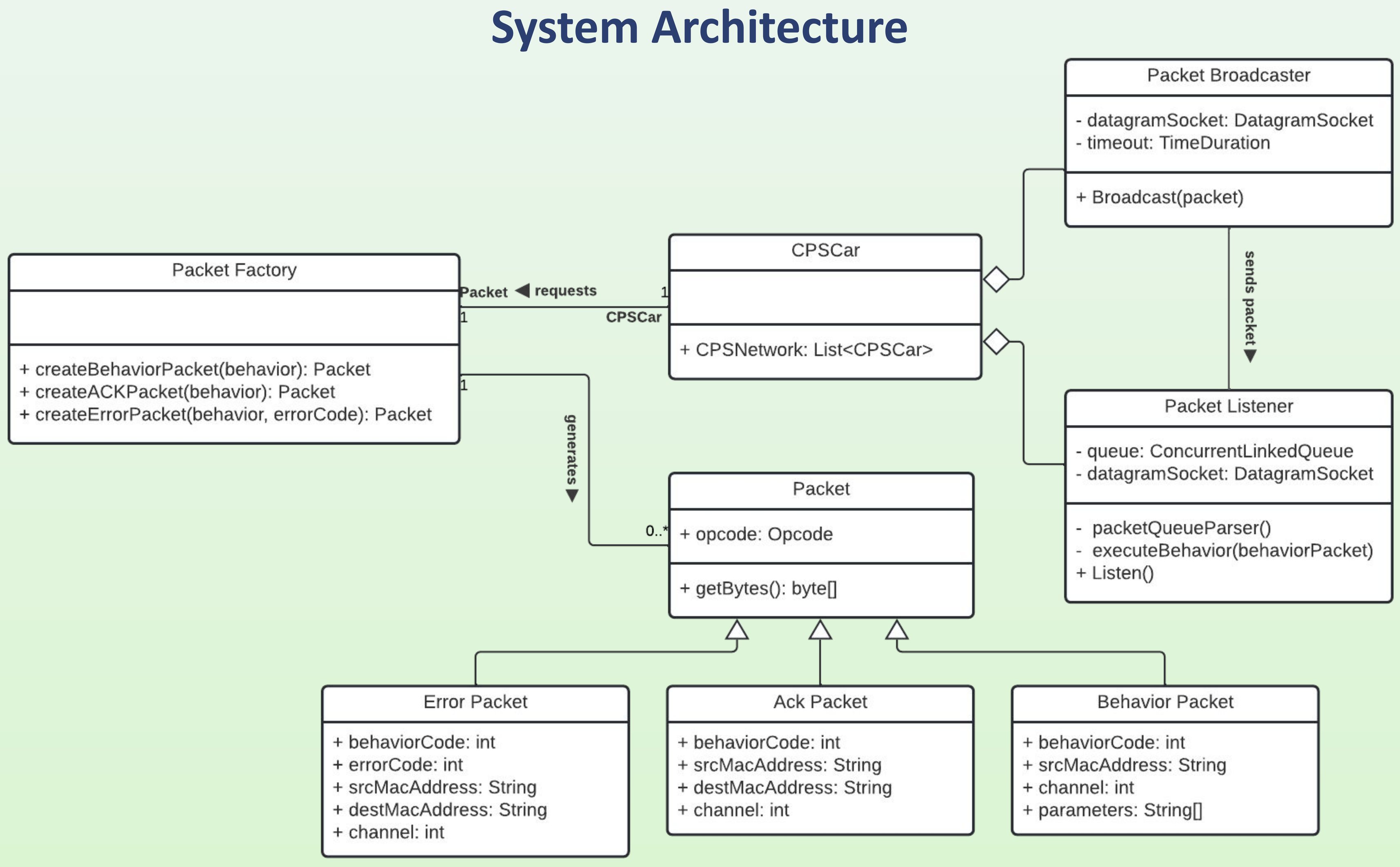
Implementation

A network specification was first proposed to outline the functionality of the ACPSNP. Within the specification, each network packet has been designed and stated use cases in which a packet may be used. The implementation of the protocol first started with developing a Packet.java class in which all packets outlined in the specification can be polymorphically extend as shown in the UML class diagram. Using a factory design pattern, a CPSCar.java instance can generate a packet that is needed for any given situation. Abstracting the process of packet generation is important for future additions of packets and/or behaviors to the system. Since the system has a behavior-oriented design, a behavior of sending and listening for packets was added to the list of behaviors. Once instantiated, a CPSCar can concurrently listen and send packets, perform self assigned behaviors and perform external requested behaviors.

Implementing a thread safe packet listener and external behavior executor was vital to the success of the system. Using a java ConcurrentLinkedQueue, CPSCar's are able to enqueue receiving packets while concurrently dequeuing, parsing, and executing the packets.

Automotive Cyber-Physical System Network Protocol (ACPSNP)

System Architecture



ACPSNP is written on top of the User Datagram Protocol (UDP), a connectionless protocol for fast throughput and one-to-many communication, which is needed for a CPS network where there is an unknown amount of participants at runtime. A CPSCar in need of a behavior consensus may multicast a behavior packet to the discovered CPS network found at runtime. Any CPSCar listening can receive the packet and execute the behavior. Once the behavior is executed, an acknowledgement that the packet has been received, can be sent to the sender through the form an ACK Packet. Due to the connectionless nature of UDP, an ACK must be sent so the sender knows there is at least one listener on the network. The sender will continuously send packets to the known network until either all CPSCar's on the network sends an ACK packet or a timeout in which the CPSCar will try to rediscover a new CPS network and try to resend the same packet to the newly formed network.

A component of CPS networks is collaboration of goal achievement. When a CPSCar multicasts a behavior packet, recipients must try to act on the behavior to ensure safety to the network. Once the initial behavior has been executed, the network may begin negotiating a behavior consensus. A behavior consensus is important when exploring the solution space for the traffic jam resolution [6].

Future Work

We are constantly adding new behaviors for CPSCar to allow more functionality to the system. Outlined in the network specification are channel request packets in which a CPSCar can send a request to a subset of CPSCar's on the network to listen for packets on a specified channel much like a radio frequency. Although other CPSCar's not on the channel can see packets, they will not execute the packets. A scenario in which a channel packet could be used is a CPSCar needs all following CPSCar's to reduce their speed. A channel packet can allow for only CPSCar's in the same lane to reduce the speed without affecting other CPSCar's in different lanes.

Contact

Web https://github.com/tenbergen/Automotive-CPS

Email gmaldona@oswego.edu | bastian.tenbergen@oswego.edu

References

1. Davis, A. (2018) Automous Driving – The Complete Guide. Wired Magazine, https://www.wired.com/story/guide-self-driving-cars/ accessed 8/9/20.
2. Basarke, C.; Berger, C.; Rumpe, B. (2007) Software & Systems Engineering Process and Tools for the Development of Autonomous Driving Intelligence. Journal of Aerospace Computing, Information, and Communication 4(12), pp. 1158-1174.
3. Daun, M.; Salmon, A.; Tenbergen, B.; Weyer, T. (2015) Today's Challenges and Potential Solutions for the Engineering of Collaborative Embedded Systems. Proc. 2nd Intl. WS EITEC.
4. Berger, C.; Rumpe, B. (2012). Autonomous Driving - 5 Years after the Urban Challenge: The Anticipatory Vehicle as a Cyber-Physical System. Proc. 10th WS Automotive Soft Eng.
5. Shanaa, W.; Spier, S.; Tenbergen, B. (2017) A Case Study into the Development Process of Cyber Physical Systems. Proc. 3rd Intl. WS RESACS.
6. Tenbergen, B.; Daun, M.; Aluko Obe, P.; Brings, J.: View-Centric Context Modeling to Foster the Engineering of Cyber-Physical System Networks. In: Proc IEEE Intl. Conf. Soft Architecture (ICSA), 2018.