Software Requirements Specification (SRS)

Electrical Power Assisted Steering

Authors: Richard Brush, Maxime Goovaerts, Tyler Olsen, Jacob Ray, Jourdan West

Customer: Tony Torre (Chrysler)

Instructor: Betty Cheng

1 Introduction

This Software Requirements Specification (SRS) provides an overview of the functionality of the Electrical Power Assisted Steering (EPAS) system specified by Tony Torre. This document covers the scope, organization, description, constraints, assumptions, requirements, specification models, and the prototype of the EPAS system.

1.1 Purpose

This document describes in a detailed fashion the EPAS system and its requirements specified by Tony Torre. It clearly explains the system’s intended functionality, constraints, and requirements. The content within is directed at all individuals involved in the project including the customer, the developers, and the users of the system.

1.2 Scope

The EPAS system to be provided is a system that takes multiple steering inputs and supplies a calculated amount of assistance to the driver. The EPAS will accept input from the driver and multiple Autonomous Driver Assist System (ADAS) modules. This system will run on embedded automotive computing hardware and assist the driver in steering the vehicle. At any time, the driver will be able to override the system, should it be necessary. If a failure within the system is detected, it will immediately shut down. Outside the scope of the system lies the actual physical entities that turn the car including the steering rack and the steering wheel. Data is only collected from and sent to those parts of the car.
1.3 Definitions, acronyms, and abbreviations

- **ABS** - Anti-lock Breaking System. A system designed to prevent the continuous locking of wheel rotation during braking. This system helps the driver maintain control of the vehicle.
- **ADAS** - Autonomous Driver Assist System. An example of an ADAS would be park assist or lane departure prevention. The system can assume control of the vehicle for a period of time to assist the driver with the specified task of parking or keep them from drifting out of their lane.
- **Autonomous Control** - EPAS system currently being controlled by an ADAS module.
- **Autonomous Functions** - EPAS system behavior under control of an authorized ADAS. The behavior is determined by the specific ADAS module.
- **AP** - Autonomous Parking. One of the ADAS modules, automatically parks the vehicle. This functionality is demonstrated in our prototype.
- **CAN** - Controller Area Network. The CAN bus of a vehicle allows ECU’s to communicate.
- **ECU** - Electronic Control Unit. A term for the controller of embedded systems contained within the vehicle. For example, an ADAS or EPAS would each have one or more ECU’s to process incoming sensor data.
- **EPAS** - Electrical Power Assisted Steering. This system uses an electric motor to assist the driver in steering the vehicle. In the past, this was been done via hydraulics.
- **EPS** - Electric Power Steering. Same as EPAS.
- **ESC** - Electric Stability Control. A system that applies braking power individually to wheels to help compensate for and maintain control under skidding conditions. Engine power may also be reduced.
- **LPC** - Lane Position Correction. One of the ADAS modules, keeps the vehicle in its current lane while driving. This functionality is demonstrated in our prototype.
- **Normal Control** - System currently being controlled by Driver input.
- **PAS** - Power Assisted Steering. Any method of steering control in which the driver’s input is assisted via an actuator used to decrease the amount of torque required by the driver to assert control of the vehicle.
- **PWM** - Pulse Width Modulation. A technique where the length (width) of an electrical pulse is varied, rather than its amplitude or voltage to encode information for transmission.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
● SRS - Software Requirements Specification, this document. Its purpose is to define and elaborate upon all requirements for the software system.
● UML - Unified Modeling Language. A language that specifies a set of models for describing the design and behavior of computer software.

1.4 Organization

The SRS is organized as such:

1. **Introduction**: A brief look at what is to be expected within this document
2. **Overall Description**: A description of the system along with background information to provide the reader with context. System functionality and restraints are also included.
3. **Specific Requirements**: Contains the list of requirements needed to ensure the system meets specifications.
4. **Modeling Requirements**: Provides detailed diagrams and descriptions to generate understanding of the system’s functionality.
5. **Prototype**: Gives an overview of the system prototype including sample scenarios and hardware requirements.
6. **References**: A bibliography of information used to write this document.
7. **Point of Contact**: Direct any questions to the person found in this section.

2 Overall Description

This section provides further information on the scope of the project. The various functions that the EPAS system will be expected to perform are covered in detail by section 2.2. The characteristics of the expected user of the system are given in section 2.3. The various constraints that have shaped the requirements are documented in section 2.4. Any assumptions that have been made during the design of the system are shown in section 2.5. Finally, due to time constraints, several requirements have been deemed outside of the scope of this project. These requirements are documented in section 2.6.

2.1 Product Perspective

This product is intended to aid the driver in the steering of a vehicle. Given inputs from the various sensors throughout Figure 1, the system will calculate an amount of torque to be applied to the steering gear. This will facilitate use of the steering wheel. The EPAS is within a larger steering system consisting of direct hardware linkages from the steering wheel to the wheels. Shown below in Figure 2 is a data flow diagram indicating the flow of information through the EPAS system. The electric motor is
where the EPAS system comes into contact with the rest of the steering column as mentioned above.

The product interfaces with the above mentioned sensors that measure the vehicle’s speed and the torque that the driver exerts on the steering wheel. In addition, the EPAS can receive input from external ADAS modules such as Traffic Jam Assist. These ADAS modules provide a desired steering angle. The controller then calculates the appropriate torque to provide to the rack. The torque is generated by the electric motor that takes PWM as an input.

*Figure 1 - Inputs affecting controller calculations (Torre, Tony. 2014, October 29. Private Communications)*
Figure 2 - Steering Control Diagram (Torre, Tony. 2014, October 29. Private Communications)

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb@chengb.cse.msu.edu)
2.2 Product Functions

The EPAS controller will provide autonomous power steering functions and assist the driver by using current vehicle speed and an interpolation table to control the amount of torque assist provided (Torre). During implementation of the system, the test driver will have access to an emergency stop button that sends a signal to the EPAS controller that terminates all functionality of the system. This feature will be removed before production begins. A single failure in the EPAS shall not result in an “out of control” vehicle, instead the EPAS system turns off and the driver is given full manual control of the vehicle.

2.3 User Characteristics

There are two users of the EPAS system: driver and engineer. It is assumed that the vehicle operator is a licensed driver. The engineer should have general training in embedded software so that he knows how to properly calibrate the system.

2.4 Constraints

This section provides an enumerated list of system constraints.

1. Limited bandwidth on CAN bus will limit the number of ADAS modules the EPAS is able to accept input from.
2. Availability of non-volatile memory limits the amount of log data that may be kept.
3. The system must perform as specified while remaining within budget.
4. Due to safety and legal considerations, the driver must be able to override the system at any time during any state of operation.

2.5 Assumptions and Dependencies

This section lists assumptions we have made in designing this system.

1. The driver is properly educated in local laws and regulations, and strives to operate the vehicle in a safe manner.
2. The driver ultimately knows what is best in regards to steering control.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
3. The CAN bus has sufficient additional bandwidth available to support transmission of our messages.
4. No malicious or end user modifications will be attempted.
5. Multiple failures within the system will not occur simultaneously.
6. Input will be used from at most 1 ADAS module during autonomous functions.
7. Disabling the system in the event of a single failure will not adversely affect the driver’s ability to safely control the vehicle (they have the sufficient strength, awareness, and ability to control the vehicle without PAS).
8. We can adequately monitor all components for failure, including the ECU (EPAS controller).

2.6 Approporportioning of Requirements

Due to limited time, the customer has removed all requirements pertaining to communication over Ethernet from the project scope. In the future, due to limited CAN bandwidth, the system will be required to use Ethernet as well.

The following requirements were removed from the current scope of the project as a result of interactions with the customer:

- Password protection will be included in the packet with steering commands.
- Implement network communications using UDP and IPv4
- Require unused communications ports be disabled
- Provide two ports for using 100 BaseTx (i.e. ethernet)
- Allow acceptance of appropriate message packet data from either port.

3 Specific Requirements

This section contains an enumerated list of the project requirements broken down into four categories: safety, communications, controls, and security.

Safety
1. Create a system that safely aids vehicle steering.
2. No single failure will result in an “out of control” vehicle.
3. Given a failed torque sensor, give full manual control to the driver.
4. The test driver will have a discrete override switch which will disable the autonomous steering and return to Normal Control.
5. Autonomous Control will ramp down and be terminated in the event that the driver torques (applies rotational force on) the steering wheel.
6. All circuits that could result in a single point failure will be monitored for faults.
7. An identified fault will be mitigated by termination of the function or invoking other alternate circuits.
8. Interaction with the ADAS modules will not result in damage to vehicle, person, and or property.
9. In the event of a system failure, mechanical steering of the vehicle should remain operational.

Communications
10. Normal Control must read the torque sensor and calculate the power assist every 500 microseconds.
11. Restrict inputs to only authorized ECUs.
12. The last 15 seconds of data will be stored for diagnostic purposes.

Controls
13. Normal Control will operate using torque sensors on the steering column.
14. Normal Control use an interpolation algorithm based on vehicle speed and driver's input torque.
15. Changes from Autonomous Controls to Normal Control will ramp down from Autonomous Control torque to the torque required for Normal Control within 500 milliseconds.
16. Two control schemes will be utilized.
   a. Normal EPAS Open Control (Normal Control)
   b. Autonomous Function Wheel Angle Closed Loop Control (Autonomous Functions)
17. Torque sensors, used for Normal Control will be used, will be discrete wired.

Security
18. Data transfers of program and calibration data require encryption in the transfer.
19. When calibrations or program code is written, the identification number for the person making the change will be recorded.
4 Modeling Requirements

Below, in Figure 3, is the class diagram describing the Electronic Power Assisted Steering (EPAS) system. The EPAS system will receive input from the sensors and Advanced Driver Assistance Systems (ADAS’s), calculate a response using the interpolation table, and direct the steering angle via the assist motor. In addition, errors, diagnostic information, and information events are logged for debugging and traceability requirements. The states of external ADAS modules and sensors are represented internally using the shown ADAS and sensor classes.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
The following tables are the data dictionary that describes the purpose and relationships between the classes in the class diagram.

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPAS</td>
<td>Controller used in autonomous steering.</td>
</tr>
</tbody>
</table>

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state: State</td>
<td>Represents the state of the EPAS including whether it is on or off.</td>
</tr>
<tr>
<td>verified: bool</td>
<td>True if sensors of the same type reflect the same data. Else, returns false.</td>
</tr>
</tbody>
</table>

**Operations**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>powerOn(): bool</td>
<td>Turn on the EPAS. Returns true if EPAS was successfully powered on, false otherwise.</td>
</tr>
<tr>
<td>powerOff(): bool</td>
<td>Turn off the EPAS. Returns true if EPAS was successfully powered off, false otherwise.</td>
</tr>
<tr>
<td>VerifyData(): bool</td>
<td>Returns true if data read from sensors is consistent with what would be expected from the Interpolation Table. Else, returns false.</td>
</tr>
<tr>
<td>sendPWM(double): bool</td>
<td>Sends PWM to Motor Drive Power Amplifier. Anything beyond sending the PWM from the EPAS is beyond the scope of the EPAS. Returns true if PWM is successfully sent, false otherwise.</td>
</tr>
<tr>
<td>InitInputSources(): void</td>
<td>Initialize all input sources (NormalInput and ADASInput’s).</td>
</tr>
<tr>
<td>calibrate(): void</td>
<td>Calibrate the EPAS to modify how power-assisted steering behaves.</td>
</tr>
<tr>
<td>VerifySensors(): bool</td>
<td>Returns true if sensors of the same type reflect the same data. Else, returns false.</td>
</tr>
</tbody>
</table>

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at cse.msu.edu)
### Relationships

The EPAS uses an InterpolationTable, Log, at least three Sensors, and zero to many InputSource’s.

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NormalInput</td>
<td>Represents driver’s application of rotational force onto the steering wheel.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>torque: double</td>
<td>Torque read from the torque sensors.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Relationships</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NormalInput</td>
<td>NormalInput is associated with two torque sensors (class Torque) in that it reads the torque sensors to determine the torque applied by the driver. NormalInput derives from InputSource.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADASInput</td>
<td>Represents input to EPAS from an ADAS that will drive the vehicle in specific and limited operation. As far the EPAS is concerned, ADASInput comes in the form of a requested steering angle.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle: double</td>
<td>Steering angle requested by an ADAS.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
### Relationships

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADASInput derives from InputSource.</td>
<td></td>
</tr>
</tbody>
</table>

### InputSource

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>InputSource</strong></td>
<td>Represents a source of driver-based input or ADAS-based input to the EPAS.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attributes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>name: string</td>
<td>Name of the Input source</td>
</tr>
<tr>
<td>priority: int</td>
<td>Predefined priority of the Input Source. There are two types of input sources – NormalInput and ADASInput. NormalInput is defined to have the highest priority and can therefore override ADASInput.</td>
</tr>
<tr>
<td>sysID: int</td>
<td>Unique integer identifier of the InputSource. This not only assists in distinguishing NormalInput from ADASInput but also between inputs of the numerous ADAS's in the vehicle.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GetInput(): InputSource</td>
<td>Get the input source.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Relationships</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>InputSource has two derived classes – NormalInput and ADASInput. The EPAS uses zero to many input sources.</td>
<td></td>
</tr>
</tbody>
</table>

### InterpolationTable

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InterpolationTable</td>
<td>Contains data used to determine the torque the assist motor must apply to the steering rack.</td>
</tr>
</tbody>
</table>

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
### Attributes

| Description | N/A | N/A |

### Operations

Calculate(double torque, double speed): double

Given an applied torque and vehicle speed, calculate the amount of torque to be provided by the assist motor in order to autonomously assist the driver's steering.

### Relationships

InterpolationTable is composed of zero to many instances of TableEntry. The EPAS uses a single InterpolationTable.

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TableEntry</td>
<td>Represents an entry in the Interpolation Table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Torque applied to the steering wheel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>speed: double</td>
<td>Vehicle Speed</td>
</tr>
</tbody>
</table>

### Operations

N/A

### Relationships

InterpolationTable is composed of zero to many TableEntry’s.

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log</td>
<td>A collection of recorded events.</td>
</tr>
</tbody>
</table>

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event</td>
<td>An occurrence that warrants being recorded. Events are recorded in the Log.</td>
</tr>
<tr>
<td>Attributes</td>
<td></td>
</tr>
<tr>
<td>date: Date</td>
<td>Date of recorded event.</td>
</tr>
<tr>
<td>id: int</td>
<td>Unique integer identifier of the event.</td>
</tr>
<tr>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Relationships</td>
<td>The Log is composed of zero to many Event's.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>An error in the EPAS that is to be recorded in the log. An example of an error is the failure of one or both torque sensors.</td>
</tr>
<tr>
<td>Attributes</td>
<td></td>
</tr>
<tr>
<td>message: string</td>
<td>Text description of the error event.</td>
</tr>
<tr>
<td>code: int</td>
<td>Integer error code.</td>
</tr>
<tr>
<td>Operations</td>
<td></td>
</tr>
</tbody>
</table>

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>Extends Event.</td>
</tr>
</tbody>
</table>

### Information Event

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information</td>
<td>An event associated with vehicle state data being written to the log. The vehicle state data encompasses vehicle speed and applied torque.</td>
</tr>
</tbody>
</table>

**Attributes**

- **data**: Data containing details of this Information Event.

**Relationships**

- Information extends Event.

### Diagnostic Event

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diagnostic</td>
<td>An event to be written to the log that is associated with an EPAS calibration performed by an engineer or a mechanic performing a repair.</td>
</tr>
</tbody>
</table>

**Attributes**

- **personId**: int Identification number of person performing the diagnostic.

**Relationships**

- Diagnostic extends Event.

### Sensor

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor</td>
<td>Contains data that can be read by the EPAS.</td>
</tr>
</tbody>
</table>

---

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
### Attributes

<table>
<thead>
<tr>
<th>name: string</th>
<th>Name of the sensor.</th>
</tr>
</thead>
<tbody>
<tr>
<td>unit: string</td>
<td>Unit of measurement of the sensor.</td>
</tr>
</tbody>
</table>

### Operations

| verifySensor(): bool | Has this sensor failed? |

### Relationships

This is one of many sensors in the vehicle containing data representing the state of the vehicle. There are at least three sensors in the vehicle: 1 Speed Sensor and 2 Torque Sensors.

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>Contains data about the vehicle's speed that can be read by the EPAS.</td>
</tr>
</tbody>
</table>

#### Attributes

| data: Data | Details about the vehicle’s speed. |

#### Operations

| N/A | N/A |

#### Relationships

Extends class Sensor. There is at least one Speed Sensor in the EPAS.

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Torque</td>
<td>Contains data about the torque currently being applied to the vehicle's steering wheel.</td>
</tr>
</tbody>
</table>

#### Attributes

| unit: string | Units of measurement for torque. |

---

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
<table>
<thead>
<tr>
<th>data: Data</th>
<th>Information about the applied torque.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Relationships</td>
<td>Extends class Sensor. There are two torque sensors in the vehicle located on the steering column.</td>
</tr>
</tbody>
</table>

The use case diagram is based on the requirements provided by Tony Torre and those contributed by the group. It is comprised of four actors, outside entities that interact with the system, labeled “Driver”, “ADAS’s”, “EmergencySwitch”, “Engineer”, and “Sensor Failure Indicator”. While they are represented in the diagram as stick figures, they are abstract representations of an interaction with the system. The large blue box labeled EPAS is the system boundary and represents the scope of the system. The ovals inside the boundary are the use cases that the diagram draws its name from. Each actor interacts with at least a single use case. There are three types of connections; the first, a black line, links an actor with a use case. The second type is a dashed line with an arrow with “Extend” above it. This line adds its origin use case to the case it is pointing at. The final type is similar to the extend line but has the label “Include.” Includes are used when there is a common element between two or more cases.

Show below in Figure 4 is the system’s use case diagram. The emergency switch is able to completely disable and power off the EPAS system. The driver and the ADAS’s interact with the system by sending their respective commands. The engineer is only capable of calibrating the system. The sensor failure indicator is responsible for monitoring for failures and reacting in an appropriate manner if any are detected. The appropriate response to a failure is to power down the EPAS system.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Use Cases:

**Use Case:** Disable EPAS  
**Actors:** EmergencySwitch  
**Description:** Driver presses a switch in the vehicle to disable the EPAS.  
**Type:** Primary  
**Includes:** Power Off  
**Extends:** N/A  
**Cross-refs:** 1, 4  
**Use Cases:** Power Off

**Use Case:** Power Off  
**Actors:**  
**Description:** Shuts down the EPAS system if the driver presses the discrete switch or in the event that a sensor failure is detected. The vehicle continues to operate in a safe fashion even with the EPAS disabled.  
**Type:** Secondary  
**Includes:** N/A  
**Extends:** N/A  
**Cross-refs:** 1, 2, 3, 4, 6, 8, 9

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
### Use Cases:
- Disable EPAS, System Failure,

### Use Case: System Failure

**Actors:** Sensor Failure Indicator

**Description:** Sensors of the same type are constantly comparing their data to each other in order to determine if they do not match. The EPAS performs a sanity check to determine if the sensors’ data are outside the acceptable range. If the sensor data does not match or the data is out of range, the system shuts down.

**Type:** Primary

**Includes:** Power Off

**Extends:** N/A

**Cross-refs:** 1, 3, 6, 7, 8, 9

**Use Cases:** Power Off, Driver-based Torque Command, Angle-based ADAS Command

### Use Case: Calibrate

**Actors:** Engineer

**Description:** The engineer is calibrates the system. The engineer is changing values in the interpolation table to modify how the system responds.

**Type:** Primary

**Includes:** N/A

**Extends:** N/A

**Cross-refs:** 18, 19

**Use Cases:**

### Use Case: Driver-based Torque Command

**Actors:** Driver

**Description:** The driver applies rotational force to the steering wheel providing input to the torque sensors. This input overrides any active ADAS modules.

**Type:** Primary

**Includes:** N/A

**Extends:** N/A

**Cross-refs:** 5, 10, 12, 13, 14, 15, 16a, 17

**Use Cases:** System Failure

---

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Use Case: Angle-based ADAS Command

Actors: ADAS’s

Description: ADAS provides angle-based command.

Type: Primary

Includes: N/A

Extends: N/A

Cross-refs: 8, 11, 12, 16b

Use Cases: System Failure

The following diagrams are sequence diagrams representative to the different scenarios the system is expecting to encounter.

The first sequence diagram, Figure 5, depicts the engineer calibrating the EPAS system. The engineer enters calibration mode and updates the EPAS core settings, after that is completed, data representing the calibration is recorded to the system’s log.

Figure 5 - Sequence Diagram 1: System Calibration

The second sequence diagram, Figure 6 represents a situation in which the emergency off switch is pressed while the EPAS system is on. The EPAS system will immediately power off and manual control is given to the driver.

Figure 6 - Sequence Diagram 2: Emergency Button Override

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
The third sequence diagram, Figure 7, represents the regular use case of input control. After the driver powers the system on and all safety checks are verified, the input sources are retrieved. The system evaluates the input source with the highest priority and provides an output PWM value based on internal calculations.

Figure 7: Sequence Diagram 3: Input Control

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
The state diagram, Figure 8, illustrates the various states that the EPAS controller component will be in at any given time. Yellow boxes indicate states of the system, while arrows indicate possible transitions between system states. The diagram is read in a fashion similar to a flow chart, with each possible state-to-state transition indicated by arrows.

![State Diagram](image)

*Figure 8 - State Diagram*

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
5 Prototype

This section will explain what the prototype does and display a few different scenarios. The prototype displays the EPAS system’s functionality, how various ADAS modules interact with each other and driver input. The driver can adjust the steering angle and vehicle speed at any time. Two ADAS modules are available for demonstration - LPC and AP.

LPC displays a vehicle on a road with two lanes. If LPC is on it will actively work to keep the vehicle in its current lane; by adjusting the vehicle's steering angle. If the driver adjusts the steering angle or speed while LPC is on, LPC input will be ignored for a couple seconds. This functionality represents the driver always being in control. This allows the driver to switch lanes even when LPC is active if they so choose to.

AP will display a series of parallel parking steps adjusting the steering angle and speed as it executes the steps. If the driver adjusts the steering angle or speed while AP is on, AP will be overridden and turn off. If the AP steps are fully executed without interruption and the vehicle has successfully parked AP will turn itself off.

5.1 How to Run Prototype

The prototype will run within our webpage – accessible at: http://www.cse.msu.edu/~cse435/Projects/F2014/Groups/STEERING2/web/prototype.html. It will not require any special plugins, will run on the most recent version of all major browsers: Chrome, Firefox, Internet Explorer, and Safari, and on both major Operating Systems: Mac and Windows. The prototype will be composed of HTML, CSS, and JavaScript.

The upper left section of this prototype will allow the user to change the vehicle's steering angles (adjustable with the slider) and speeds (adjustable with the dropdown menu). The steering torque value will adjust as the steering angle slider is changed. The steering wheel will rotate as the steering angle slider is adjusted.

The upper right section of this prototype will display the final torque out value; which will be calculated based on the steering angle and speed. The wheels and wheel angle will also adjust as the input values are changed.

The lower left section of this prototype displays the ADAS functionality. Two modes are available for execution: LPC and AP. The modes can be toggled by pressing their respective buttons. A red circle signifies that the system is off, green that the system is on, and yellow that the system is temporarily disabled. When the start button is pressed the prototype will execute a script. This script automatically changes the input values to recreate a scenario (such as parking a vehicle).

The lower right section of this prototype will display a graphic that represents the ADAS mode visually.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
5.2 Sample Scenarios

Start Screen:

Figure 9 displays our prototype’s start screen. Users will see this displayed when the prototype is first opened. The driver controls are at the top left, ADAS controls at the bottom left, output at the top right, and ADAS visualization at the bottom right. The ADAS modules are currently disabled, as marked by the red circles next to their buttons.

*Figure 9 - Prototype Start Screen*
Driver Control:

In sample scenario one, Figure 10, the driver is currently operating with both ADAS modules turned off. The wheel is turned to the right 75°, with a steering torque of 3. The car has a current speed of 15 mph. The power steering controller adjusts the output wheel angle to be 10°, based on the speed and steering angle of the vehicle.

Figure 10 - Prototype Sample Scenario One: Driver Control
Lane Position Correction - Off:

In sample scenario two, Figure 11, the vehicle is currently driving down a road with LPC turned off. The driver runs the risk of accidentally moving out of its current lane as displayed in Figure 12.

Figure 11 - Prototype Sample Scenario Two: LPC Off

Figure 12 - Prototype Sample Scenario Two: LPC Off fail

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Lane Position Correction - On:

In sample scenario three, Figure 13, the vehicle is currently driving down a road with LPC turned on. If the vehicle starts veering out of its current lane the APAS LPC system will kick in and adjust the steering angle to avoid this as displayed in Figure 14.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Lane Position Correction - Driver Override:

In sample scenario four, Figure 15, the driver is able to switch lanes even with LPC on. The LPC ADAS system is temporarily disabled when the driver provides input. If the driver were to relinquish control, the LPC ADAS module would become active once again.

![Image of Driver Controls, ADAS Controls, and ADAS Visualization]

**Figure 15 - Prototype Sample Scenario Four: LPC Override** [Steering 2, website]
Autonomous Parking - On:

In the sample scenario five, Figure 16, the vehicle is currently parking itself with AP turned on. Figure 17 displays the vehicle in its final stage after successfully completing the parking maneuver.

**Figure 16 - Prototype Sample Scenario Five: AP in progress**

**Figure 17 - Prototype Sample Scenario Five: AP Complete**

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Autonomous Parking - Interrupted:

In the sample scenario six, Figure 18, the parking maneuver was disrupted by the driver. At any time in the parking process, the driver may choose to interrupt it by moving the steering wheel or adjusting the speed; AP will disable itself.

*Figure 18 - Prototype Sample Scenario Six: AP interrupted*
6 References


7 Point of Contact
For further information regarding this document and project, please contact Prof. Betty H.C. Cheng at Michigan State University (chengb at cse.msu.edu). All materials in this document have been sanitized for proprietary data. The students and the instructor gratefully acknowledge the participation of our industrial collaborators.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)