1. **Задача:** Существуют ли какие-то проблемы, общие для всех разработчиков, мешающие программистам работать с большей эффективностью? Какое решение этих проблем Вы видите? Существуют ли какие-то следующие шаги, которые возможно потребуется предпринять, если Ваше решение указанных проблем будет реализовано на практике?

Думаю можно разделить проблемы, снижающие эффективность любого разработчика на ключевые группы

Организационные факторы:

- Неясность в постановке задачи или расплывчатые требования
- Неверные приоритеты
- Неконтролируемое изменение границ проекта
- Определение возможностей продукта без оценки интересов пользователей или заказчиков
- Безучастность менеджеров в рабочем процессе и их подключение в отдельные моменты только ради критики
- Придирки к несущественным мелочам
- Бесконечные совещания и различные отвлекающие моменты
- Некорректные или сильно сжатые сроки

Личные

- Плохие софт-скиллы
- Поиск легких решений и отсутствие любознательности
- Нежелание изучать основы, формирующие крепкий фундамент
- Фундаментальные навыки превращают плохого программиста в хорошего
- Осознаное допущения в выборе решений и написании кода, чтобы сделать продукт побыстрее
- Неструктурированное самообучение

Экологические факторы

- Организация рабочего места: шум, движение, дизайн рабочего пространства
- Режим работы и разумное сочетания периодов нагрузки и восстановления

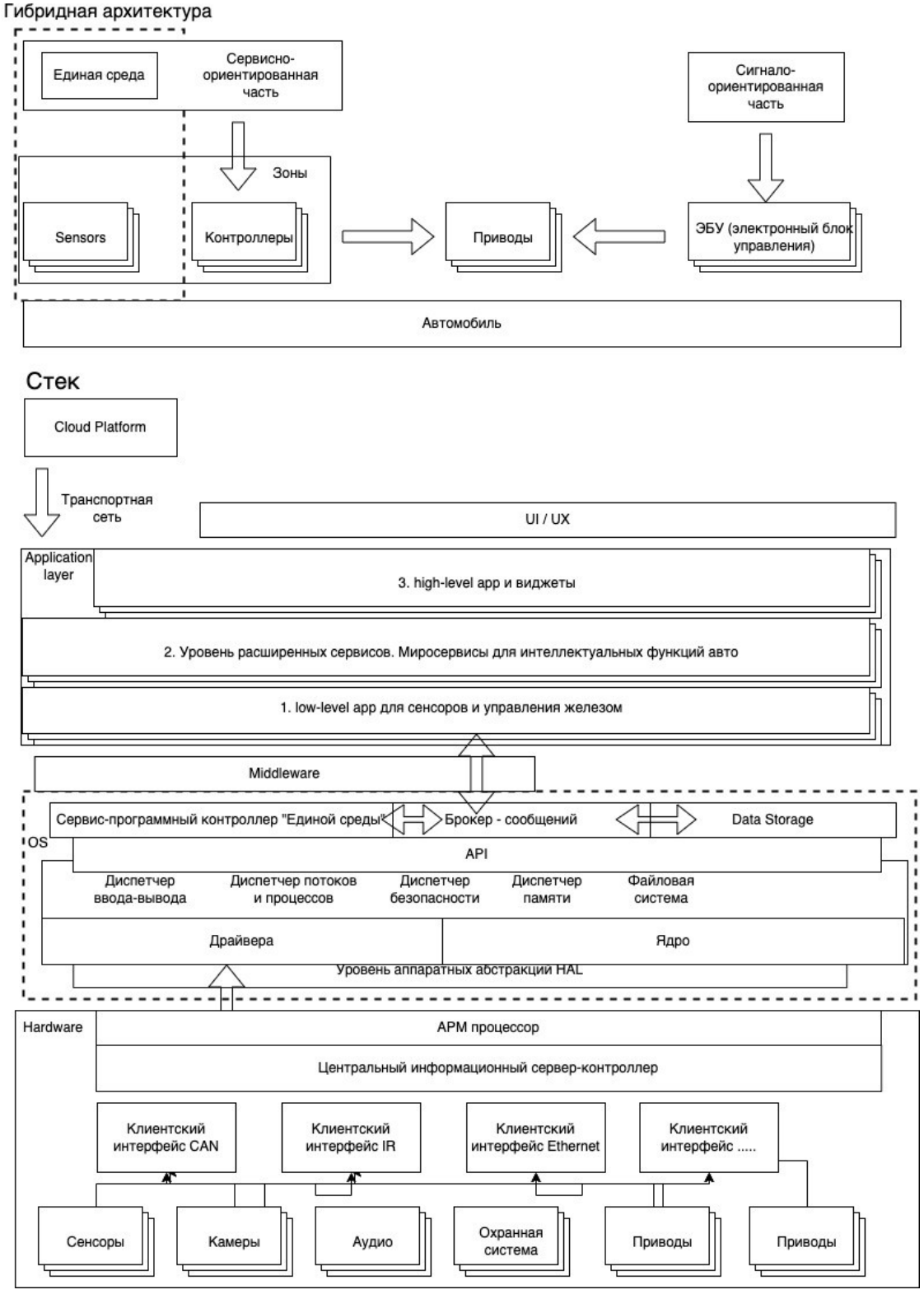
Технические

- Постоянные прыжки от технологии к технологии
- Отсутствие документирования проекта и кода
- Заведомо устарелый стек и нежелание модернизации

Решение проблем частично заложены в их формулировки. Для этого достаточно не допускать указанные ошибки в процессе формирования команд, использования регулярного менеджмента для управления. Формирование команд разработок гибридной структуры - комбинация универсалов и специалистов. Заполнение всех необходимых ролей: бизнес-аналитик, менеджер проекта, менеджер продукта, дизайнер UX/UI, разработчики фронт-енд и

бэк-енд, инженеры по качеству, ревьюверы и тестировщики. Использования методологий Agile, Scrum и Kanban.
В качестве следующих, но не последних по значимости шагов, необходимо выстраивать и совершенствовать процесс адаптации сотрудников и наставничества, формализовать процесс личностного роста сотрудников через план развития, внутреннюю, а не внешнюю мотивацию.

2. **Задача:** Предложите оптимальную, на Ваш взгляд, программную архитектуру для бортового компьютера электромобиля. Обоснуйте свое решение.



Начну с начала, так как программная часть бортового компьютера не имеет смысла без аппаратной. Предлагается гибридное решение, когда часть задач реализовывается без использования ПО более высокого уровня. Т.е используется получение сигналов сенсоров на ЭБУ, который запускает работу необходимого привода. Таким образом можно для упрощения и ускорения переложить часть простых задач Сенсор>Реакция на железо.
Вторая часть — это создание сервисной архитектуры использующей так называемую единую внешнюю среду — совокупность информации со всех сенсоров со всех зон.
Стек организован так, что на железе в виде сенсоров, клиентских интерфейсов их подключения к центральному контролеру и процессору АРМ строиться программная архитектура.
Операционная система. Если нужно несколько операционных систем для разного класса задач необходимо еще иметь гипервизор между аппаратной и программной частями.
Далее в ОС обеспечены основные функции ядра на уровне системы. А на уровне пользователя в операционной системе реализованы:
- сервис — контролер единой среды, обрабатывающий аппаратные сигналы подключенного контроллера или устройств через драйвера и превращающего его в поток информации для журналирования в сервисе хранилище данных
- хранилище данных — для журналирования и создания отказоустойчивых очередей брокера сообщений
- брокер сообщений для обмена информацией между микросервисами уровня приложений, а также сообщениями между сервисом — контролером единой среды
Брокер сообщений реализован как сервис ОС, а не на уровне приложений.
Возможно необходим уровень middleware, если необходимо для части приложений.
Далее идут три уровня слоя приложений:
- приложения обеспечивающие базовый функционал работы с оборудованием. По типу хардварного сенсор → контролер → реакция
Выше идет слой микросервисов интелектуальной обработки данных. Когда требуется ML обработка или композиция информации.
Третий уровень это пользовательские приложения и виджеты имеющие взаимодействие с пользовательским интерфейсом

3. **Задача:** Есть последовательность идентификаторов, строящаяся по следующим правилам:
Первый идентификатор последовательности имеет вид «A1», второй — «A2», третий - «A3» и так далее. За «A9» следует «B1». Следующий после «Z9» имеет вид «A1-A1», потом «A1-A2» и так далее. После «A1-Z9» следует «A2-A1».
Максимальная длина идентификатора - десять групп по два символа.
В идентификаторах никогда не должны присутствовать буквы «D», «F», «G», «J», «M», «Q», «V» и цифра «0».
Необходимо реализовать класс, обеспечивающий работу с идентификатором по заданным правилам.
Класс должен обладать следующим функционалом:

- Метод, устанавливающий текущее значение идентификатора
- Метод, инкрементирующий значение идентификатора и возвращающий новое значение

Технические требования к решению:
1. Код должен быть потокобезопасным.

2. Код должен компилироваться.
3. Код должен быть кроссплатформенным (успешно собираться компиляторами msvc/gcc/clang).
4. Для решения задачи разрешается использовать только стандартную библиотеку C++ (стандарт до C++17 включительно).

Решение в приложенном архиве:

Возможно альтернативное решение. Когда группа в виде БуквaЦифра — отдельный класс. Который в результате композиции используется в классе Identifier — полный идентификатор из 10 групп БуквaЦифра. Хранимое поле это list<БуквaЦифра> позволяющий вставлять новую группу БуквaЦифра в начало списка за O(1). Также возможно итерирование от последнего к первому для икремента и от первого до последнего для вывода идентификатора. Немного усложняется процесс переноса carry между младшим и старшим разрядом(группой) идентификатора.

Не стал усложнять код, так как не ясна необходимость композиции классов, нужно ли будет переиспользовать класс БуквaЦифра вне этой задачи.

Реализовано решение на хранимом поле типа строка. Позволяющее иметь доступ к символу за O(1). Упрощается учет carry при инкременте. Все сделано в одном классе.

Identifier.zip

- CmakeLists.txt
- Identifier.cpp — класс согласно ТЗ
- Identifier.h — класс согласно ТЗ
- main.cpp — входная точка и тестовая процедура