Инженерный подход к разработке SW/HW. Model-checking

Что такое инженерный подход?

- Формулировка требований к разрабатываемому объекту
- Точная формулировка свойств объекта на основе требований
- Проверка того, что выполенение этих свойств ведёт к удовлетворению требований (валидация)
- Разработка кода/RTL/etc
- Проверка того, что свойства кода именно те, что нами были сформулированы (верификация)

Зачем это? Быстрее же хлоп-хлоп и в продакшн? Time to market и всё такое...

- Есть области с очень высокой стоимостью ошибки (Ariane 5, Therac-25) Подробнее: https://en.wikipedia.org/wiki/Therac-25, http://is.ifmo.ru/present/2011/karpov-presentation.pdf
- Есть множество примеров, где применение инженерного подхода позволило существенно сократить сроки разработки и снизить стоимость как самой разработки, так и дальнейшей эксплуатации (OpenComRTOS)
- То есть, инженерный подход позволяет как сделать SW/HW более надёжным, так и (как бы парадоксально это ни звучало) быстрее его разработать и иногда даже существенно дешевле.

Применение в промышленности

- Разработка “железа”: протокол когерентности кешей, например, в Intel был смоделирован на TLA+/TLC (https://lamport.azurewebsites.net/tla/intel-excerpt.html)
- Разработка высоконагруженного сетевого распределённого ПО: Амазон архитектуру и алгоритмы своего облачного системного ПО моделирует в TLA+/TLC (https://lamport.azurewebsites.net/tla/formal-methods-amazon.pdf)
- TLA+/TLC используются для проверки разных алгоритмов консенсуса и пр.
- Alloy Analyzer использовался для проверки разных алгоритмов, например с его помощью были обнаружены баги в алгоритме DHT Chord (https://www.cs.cornell.edu/conferences/formalnetworks/pamela-slides-i.pdf)
- Это только очень маленькая часть того, где используется model-checking (Некоторые случаи использования Alloy можно глянуть тут: https://alloytools.org/citations/case-studies.html, TLA+: https://lamport.azurewebsites.net/tla/industrial-use.html)
- И TLA+, Alloy – это только небольшая часть большой и постоянно развивающейся области model-checking (SPIN/Promela, AtelierB, ProB, PRISM, SMV, nuSMV, muCRL, etc, etc, etc)

Примеры из личной практики

- С помощью SPIN/Promela на модели удалось доказать и отстоять инструкцию с атомарной семантикой доступа к системным регистрам
- С помощью TLA+/TLC удалось разобраться с ошибкой в модифицированном протоколе рандеву для системного кода
- TLA+/TLC позволили найти очень тонкую багу в коде, которую не ловил огромный пул тестов (даже статью написал https://habr.com/ru/company/yandex/blog/471012/)
- С помощью model-finder Alloy Analyzer посчитал метрики некоторых свойств новой архитектуры ПО в hi-load backend почты Яндекс
- Используя тот же Alloy Analyzer на моделях показал багу с фильтрами в почте
- Ещё много где применял в неотносящихся к работе проектах

Основные выводы из личного опыта

- При разработке архитектуры многопоточного/параллельного ПО позволяет сэкономить огромный объём времени на отладке (кто-нить дебажил многопоточное/распределённое ПО в проде на редко-стреляющих невоспроизводимых багах? Врагу такого не пожелаешь)
- Можно проверять разные идеи по архитектуре и алгоритмам, не тратя время на кодинг и тесты. И самое главное, чего не даёт прототипирование в коде, - это быть уверенным, что не будет архитектурных и алгоритмических багов требующих переписывания огромных объёмов кода.
- Моделирование (и в целом формальные методы) учит чётко и ясно мыслить. Это весьма важный навык для разработки не только SW/HW но и по жизни. Зачастую формулировка нужных свойств – это уже половина решения (если не больше, принимая во внимание отладку, багфиксинг, редизайн и тд).
- Прокачиваются такие важные навыки мышления как абстракция и уточнение. Что само по себе позволяет дизайнить софт и железо на новом уровне. Начинаешь лучше понимать где, как и какие интерфейсы сделать, где какие свойства важны, а какие не важны и тд.

План курса на первый семестр

- ModelChecking обзор, немного истории, про применение, немного про теорию и тд.
- Пример OpenCom RTOS или как написать в 10 (десять!) раз меньше кода и не дебажить (практически), небольшой экскурс в устройство операционок, подготовка к инженерной разработке менеджера памяти
- Начальная версия спек на менеджер памяти на Alloy + TLA+, моделирование, кодирование.
  Как быть уверенным в том, что код соответвует модели? Основные способы.
- Сборщик мусора, формулировка свойств, моделирование на TLA+/TLC, кодирование.
- Уточнение спек на менеджер памяти, real-time (что такое, зачем нужно, как отразить новые свойства в спеках), моделирование
- Реализация нового менеджера памяти

1 лекция

- История
- Основные направления
- Основные теории в основе метода
- Популярные формализмы и инструменты
- Основные области применения

2 лекция

- Пример инженерного подхода при разработке OpenCom RTOS
- Основные результаты этого проекта
- Небольшой экскурс в операционные системы: основные компоненты, какие операционки бывают и тд
- Постановка проблематики управления памятью: менеджер памяти – что должен делать? Какими свойствами обладать? Как сформулировать свойства? Как проверить?
- Что такое спецификация? Формальная спецификация? Модель? Валидация?

3 лекция

- Структурные свойства памяти, спецификация, проверка на Alloy. Основные свойства операций с памятью, спецификация, формулировка свойств относительно структуры памяти, проверка.
- Проверка динамических свойств операций с памятью на TLA+/TLC
- Как отразить свойства в коде? Какие основные способы проверки свойств кода?
- Разработка кода.

4 лекция

- Сборка мусора. Что это? Какие основные алгоритмы сборщиков?
- Основные свойства сборщика мусора.
- Формулировка свойств. Проверка с использованием TLA+/TLC.
- Начальная версия кода сборщика мусора.

5 лекция

- Уточнение спецификаций: что это? Как отразить новые требования и свойства в модели?
- Пример нового свойства – свойство реального времени. Что это за свойство? Примеры программного обеспечения с этим свойством.
- Добавление требования реального времени к менеджеру памяти.
- Уточнение спецификаций. Моделирование.

6 лекция

- Обсуждение доработок кода менеджера памяти. Как новое свойство из модели перенести в код? Как проверить корректность?
- Доработка кода менеджера памяти.
- Тесты и проверка.

Организация учебного процесса

- Так как курс небольшой и сугубо практический, то основная работа по изучению терминов, азов теории и тд. будет вынесена на самостоятельную работу. На лекциях будет проводится разбор вопросов по самостоятельной работе и разбор и решение практических задач.
- Перед каждой лекцией будет небольшое задание на самостоятельную работу относительно предстоящей лекции, в основном это будет работа по поиску и ознакомлению с основными терминами и понятиями, чтобы на самой лекции уже не отвлекаться на терминологический аппарат.
- Так же к самостоятельной работе будут предложены некоторые задачи, например по формулированию свойств того, что планируется разработать. На лекции будем разбирать у кого и что получилось сделать/сформулировать.
- Возможно будут небольшие практические задания на самостоятельную разработку небольших моделей или кода. Это будет зависеть от уровня подготовки студентов. Для “продвинутых” студентов будут предложены и более интересные и “продвинутые” задачки.

Оценка

- Ещё раз повторюсь, что курс факультативный и в зачётку не идет.
- Тем не менее есть требование от кафедры по оценке студентов с целью дальнейшего распределения дополнительных стипендий и других “плюшек”
- Оценка будет основываться на активности участии в учебном процессе:
  - Активность в самостоятельной работе
  - Участие в дискуссиях
  - Решение задач
- Ошибки в задачах и тд. на оценку влиять не будут. Ибо ошибки – это тоже часть учебного процесса ☺ Не ошибается только тот, кто ничего не делает ☺
- Возможно даже наоборот, за самые интересные ошибки, которые послужат примером для разбора на лекциях, оценка будет повышена. Ибо, чтобы сделать интересную и “тонкую” ошибку, нужно хорошо поработать и разобраться до определённой степени в материале.

Библиография

- Юрий Карпов MODEL CHECKING. Верификация параллельных и распределённых программных систем
- Hillel Wayne Practical TLA+: Planning Driven Development
- Блог Hillel Wayne (автор книги "Practical TLA+") ссылка
- Блог Ron Pressler ссылка
- Лесли Лэмпорт TLA+, видеокурс ссылка
- Библиография в статьях на хабре: тут больше про Alloy - https://habr.com/ru/company/yandex/blog/457810/, а тут про TLA+/TLC: https://habr.com/ru/company/yandex/blog/471012/