Task #212

Ретроспектива

Added by Evgeny Sorokin almost 3 years ago. Updated 8 months ago.

Status:	In Progress	**Start:**	2010-10-30
Priority:	Normal	**Due date:**	
Assigned to:	Kirill Kornyakov	**% Done:**	100%
Category:	-		
Target version:	-		

Description

- Обсудить, нужно ли Clean Code читать раньше
- Ввести лекцию по Unit-тестированию и лабу №1 может быть проводить именно по этой теме. TDD сразу не ложится студентам :-(

SOLID_new.pptx - новая презентация по солиду (с "демотиваторами") (7.1 MB) Evgeny Sorokin, 2010-11-10 23:03

feedback_agile.doc - Feedback Андрея Морозова (17.5 KB) Evgeny Sorokin, 2011-01-17 00:04

Related issues

Watchers

History

Updated by Kirill Kornyakov almost 3 years ago #1

Мои предложения:

1. Доработать лекцию Антона по Рефакторингу и включить в курс. Есть серьезные опасения что не все владеют этой техникой. И вне зависимости от этого можно такую лекцию забабахать, которую и всем взрослым полезно послушать - advanced материалы. Материалов полно.
2. Нужно обязательно собрать фидбек от студентов. Можно сделать это в виде последней контрольной ;-)
    1. Что вам понравилось/пригодилось/показалось полезным? Какие темы стоит углубить?
    2. Что вам не понравилось, что стоит убрать из курса, какие недостатки курса в целом вы видите?
    3. ??? Практика? Редмайн?

Updated by Kirill Kornyakov almost 3 years ago #2

Еще нашел у себя какой-то безумный TODO:

I don't like the term 'plan-driven' when it is used as an antonym for Agile, because Agile is as 'plan-driven' as any other approach; I would use the term 'forecast-driven' instead. Agile is feedback-driven, while non-agile approaches tend to be driven by forecasts of the future. In domains where these forecasts are likely to be accurate, you can assume that the forecast is fact and devise a plan based on that assumption. Just remember when variances occur that they are as likely to be caused by faulty assumptions as faulty execution.

есть хороший иллюстративный пример про видеопрокат

рисунок полного графа как проблема при коммуникациях

незаменимость бумаги при перечислении задач

перевести лекцию по scrum на русский

Weinberg (Weinberg, 1971) cites an anecdotal example of how an organisation wanted to stop programmers wasting time talking to each other around a coffee machine.
They removed the machine, then immediately had a dramatic increase in requests for formal programming assistance. As well as gossiping around the machine, people were solving each other s problems. This illustrates that companies need informal meeting places as well as formal conference rooms

- причина рождения такой науки как программная инженерия
- Об ученых-физиках из НАСА

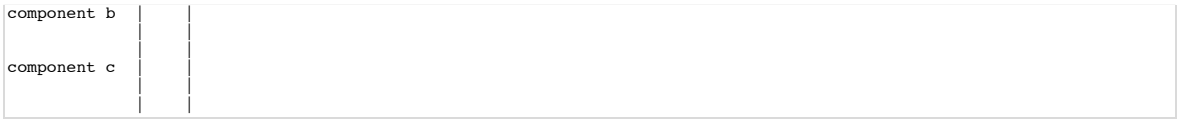
What refactoring is not:
- code cleaning
- re-engineering
- optimization

links
http://industriallogic.com/rtpdata/index.html

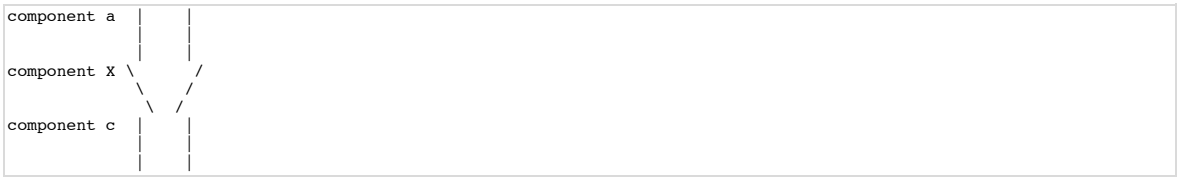
Updated by Evgeny Sorokin almost 3 years ago #3

На лекцию по SOLID нельзя отводить меньше одной лекции. Даже и пытаться не стоит :-)
Иллюстрация для LSP (а то студенты на прошлой паре не поняли) - это pipe, труба:

```
component a  |    |
             |    |
             |    |
```



пускаем по этой трубе воду. если мы хотим заменить компонент b на что-то другое (на X - его наследника), то наследник может ослаблять входные ограничения (сделать воронку шире сверху) и ужесточать выходные (воронку сделать уже к низу):



тогда вода не прольется. если воронку заузить сверху это фейл. если сделать шире снизу - тоже фейл. вроде наглядно.

Updated by Evgeny Sorokin almost 3 years ago #4

Поиск совершенных чисел как пример для лабораторной по TDD?

Updated by Evgeny Sorokin almost 3 years ago #5

- **File** SOLID_new.pptx added

для иллюстрации СОЛИД принципов надо воспользоваться диминой презентацией

Updated by Evgeny Sorokin almost 3 years ago #6

1) Нужно, чтобы студенты маркировали контрольные, чтобы можно было идентифицировать номер работы.
2) Кирилл предложил, чтобы было 2 варианта контрольных - чтобы было труднее списать :-)
Но тут надо как-то контролировать, что каждый делает свой вариант.

Updated by Kirill Kornyakov almost 3 years ago #7

Есть предложение ввести стандарт на именование тикетов. Нужно чтобы там содержалась фамилия автора и номер лабы, так будет проще понимать ситуацию, глядя на список задач.

Updated by Evgeny Sorokin almost 3 years ago #8

Meeting minutes с тренинга:

- Важно, чтобы происходила конвертация знания в навык.
- открыть к обучению: нужно поставить задачу, чтобы человек провалился, или сделал неоптимальным образом. А потом не нужно рассказывать сразу, как правильно на блюдечке - нужно, чтобы он сам дошел, делать его руками

Вспомнилось про то, как мы объясняли LSP.

Updated by Kirill Kornyakov almost 3 years ago #9

Идеи очень здравые. Действительно стоит придумать как сделать так, чтобы неправильно кодить было очень неудобно. Вообще ведь большинство идей - это грабли, так как бы сделать так, чтобы шишки понабились. Идеи пока такие:

- Проанализировать что именно будет неудобно, если не соблюдать рекомендации. И в заданиях просить сделать именно это. Несколько представлений, менять требования, ???
- Все-таки на каком-то этапе менять студентов. Беда в том, что с нашими замечаниями коды уже более-менее чистые, глаз сильно резать не будет. Но можно устраивать кросс-проверки.
- Можно выделить минут 10-15 на чтение "хитпарад говнокода недели". Естественно все анонимно, предельно тактично, но пусть замечания идут из зала.

Updated by Evgeny Sorokin almost 3 years ago #10

Летучка XP-1: Хороший вопрос про парное программирование, а вот первые два проверять неудобно

Updated by Evgeny Sorokin almost 3 years ago #11

Нужна лаба на LSP - на основе фидбека студента Волкова

Updated by Evgeny Sorokin almost 3 years ago #12

Отзывы ребят на тему как улучшить курс:

- рассказать про mock/ninject/IoC-контейнеры
- предложить командный проект
- может быть, имеет смысл интеграция с другими курсами?
- терминалы - чтобы можно было совместно поработать (опять про командную разработку)

Updated by Kirill Kornyakov almost 3 years ago #13

Выгружу из памяти, пока не забыл:

1. Лаба по смешиванию цветов слишком простая.
2. Лаба по пересечению линий в 3d слишком сложная.
3. Data Access не покрыт лабами. Может можно NHibernate + SQLLite? Если репозиторий не убъем, можно внедрять.
4. На сдачу лаб можно повесить сроки и к этому привязать получение зачета автоматом. Но мне это не сильно нравится.
5. Использовать текущие коды (перлы из первых версий) для инструктажа прямо при постановке задачи, потому что никто wiki боюсь не читает.
6. Есть предложение раздавать презентации. Но вопросы летучек делать на понимание, плюс побольше и публиковать заранее. У студентов будет резон слушать внимательнее.

Updated by Evgeny Sorokin almost 3 years ago #14

- **Status** changed from *New* to *In Progress*

Нужно формализовать последнюю лабу - чтобы не приходилось гадать, как запустить скрипт и почему он не работает.
1) не должен быть зависимым от текущего пути
2) в описании тикета студентом должны прилагаться пре-реквизиты если надо (например, что именно нужно добавить в PATH, причем не словами, а инструкцией, которую можно скопировать и выполнить в CMD
3) интерпретаторы - может, их включить в репозиторий в след. раз, чтобы студенты писали сразу их используя?
4) На удивление, они не все тырят друг у друга эту лабу, как я предполагал. Это радует. Но все-таки, может быть, назначать им скриптовые языки как-то по вариантам? Надо обсудить персонализацию этой таски

Updated by Evgeny Sorokin almost 3 years ago #15

Я увидел на приемке, как они колупаются с мышкой...
В лекцию по модульному тестированию надо обязательно включить hot keys по работе с тестами - как их запускать с клавиатуры.

Updated by Evgeny Sorokin over 2 years ago #16

Цитата из книги В.Папанека "Дизайн для реального мира".
Если закрыть повязкой один глаз, нам придется вести машину осторожнее: мы лишились восприятия дальности, так как видим пейзаж только с одной точки. Чтобы увидеть дорого (или проблему) полностью, нам надо смотреть на нее одновременно с двух наблюдательных постов. Оптически оба глаза будут выполнять эту задачу - по такому принципу действует и дальномер в фотокамере.

Можно приводить эту цитату, когда объясняем
1) триангуляцию в модульных тестах
2) парное программирование

Updated by Evgeny Sorokin over 2 years ago #17

Возможно, некоторые записки здесь следует отнести к материалам для будущей методички\книги.(как, например, цитата из предыдущего поста)
--
"Просто увеличьт предмет, не меняя его форму, и, не желая того, вы измените все его свойства" - Джулиан Хаксли.
Есть о чем подумать, как это применимо кархитектуре и дизайну ПО.
--
Обучение должно стать экстатическим переживанием, как утверждает Джордж Б. Леонард в своей книге "Обучение и экстаз". (Далее приводится пример, как много людей водят автомобили): "мчатся с большой скоростью, а расстояние между машинами - дюймы". Это, говорит, поразительное достижение, это усвоенный навык. "Возможно, это самая высокоструктурированная неинстинктивная деятельность, которую водители машин выполняют в жизни. (...) Ключ успеха - в оригинальном обучении вождению.(...) И начинающий водитель, и машина, а также дорожная система, другие машины и учитель составляют саморегенерирующуюся систему, которая положительно реагирует на всякий успех обучеющегося"

К чему это все? Возможно, это для объяснения почему парное программирование для новичка в команде так эффективно. Возможно, стоит подумать, как сделать процесс обучения более интерактивным.

"Специализация в живой природе обычно приводит к вымиранию" (оттуда же, из Папанека "Дизайн для реального мира") - это, как ты уже догадался, к плюсам кросс-функциональности agile-команд

Updated by Evgeny Sorokin over 2 years ago #18

Фидбек от студентов:

Хорошо:
Практические задачи. Можно попробовать что-то, а не просто слушать.
Еженедельный опрос - это хорошо
4+ из 5
Изучение TDD, техник рефакторинга, шаблона PV, MVC, SOLID
В целом курс понравился, много перенял
Хорошо - технические темы (TDD, DDD)
Полная проверка требуемых знаний: качественные знания,
Актуальность тем
Хорошая идея с SVN
"P.S. Плюсы и так понятны - полезный интересный курс и т.п., поэтому подробно расписывать не стал, написал то, что возможно стоит изменить"

Плохо:
При довольно строгом контроле в течение семестра еще и зачет
Количество лаб и время ответа - ниже среднего
Лекции - неустоявшаяся система "густо-пусто"
Слишком много лабораторных работ
Материалы не выдавались

нужны конкретные сроки сдачи для каждоый лабы
Побольше практических примеров, иллюстрирующих IdentityMap, UoW, ORM
Надо чуть больше времени на летучки
Scrum, XP - их читают не первый раз, пока не будет практики, знания не пригодятся для использования
Полная проверка требуемых знаний: куча времени уходит на контроль знаний
Слабая мотивация: 50% деятельности на этот курс, и всего-то зачет
Затея с SVN вылилась в "хуже, чем в терминал-классах" из-за времени откликов
Схожие вопросы на экзамене и в текущих контрльных - зачем спрашивать одно и то же два раза?
Много полезной инфы, но часто просто недостаточно опыта, чтобы что-то проанализировать или ответить на вопрос, возможно не стоит так подробно углубляться в суть некоторых вещей
Презентации на англ. не есть гуд. Иногда важен русский перевод, да и ктому же это просто понятне

Что улучшить:
Возможно, добавить сроки сдачи лаб, чтобы не накапливались
Лекции должны быть в общем доступе
После каждой летучки для желающих пара с разъяснениями и досдачей долгов
Feedback по контрольным сразу
Нужны сроки сдачи лаб
Добавить лекции про mock'и (например moql) IoC-контейнеры (ninject)
Провести реальную практику Agile-разработки. Как - сложно сказать. Например, с отдельной группой желающих людей
Можно убавить кол-во контрольных и добавить лабы (например, на реализацию UoW)
Можно ввести дни проверок лаю. Тогда народ будет знать, в какой день ждать фидбек
Ввести еще несколько пар, когда вы кодите прямо при нас - очень полезно IMHO
Надо давать студентам лекции в электронном виде, может быть не сразу, а через 2-3 пары, но давать.

Updated by Evgeny Sorokin over 2 years ago #19

- **File** feedback_agile.doc added

Updated by Kirill Kornyakov about 2 years ago #20

- **Status** changed from *In Progress* to *Done*

Updated by Kirill Kornyakov about 2 years ago #21

- **Status** changed from *Done* to *In Progress*
- **Assigned to** changed from *Evgeny Sorokin* to *Kirill Kornyakov*

Updated by Redmine Admin over 1 year ago #22

Фидбек от Ильи Лебедева

> *Привет.*
>
> *Курс был интересный и понравился.*
>
> *Материала было дано очень много, хотя до некоторых моментов пришлось докапываться самостоятельно(некоторые детали, например что такое метафоры, некоторые паттерны и т.п.). Контрольные были очень забавной тренировкой кратковременной памяти и мелкой моторики :), а некоторые вопросы были неожиданными и приходилось реально вспоминать. На первой паре понравилась фраза(не помню точной формулировки, но смысл я понял именно так) "можете списывать, но буду ловить". Иногда было легкое желание увидеть в контрольной вопросы типа: найдите какие принципы нарушены в приведенном коде, или отрефакторите этот код. Лабы немного огорчали, код который в худшем случае писать два дня, приходилось писать в несколько раз дольше, да еще и с жутко сложной структурой которую еще продумать в начале нужно. Хотелось бы увидеть что нибудь более занимательное, где громада DDD была бы хоть немного к месту, например несколько человек разрабатывают отдельно разные проги, а потом все это объединяется в одну структуру под DDD (например по тематике задания, они у тебя и так разделены по несколько человек с отдельными задачами на тему, например КАЛЬКУЛЯТОРЫ и подразделяется на 5 подзадач и т.д.) вот тут и независимая разработка разных слоев и активная работа с svn и работа в команде(одна из главных мыслей, проходящая через весь курс, но ни как не отразившаяся ни где).*
>
> *Извиняюсь что так долго не отвечал. С уважением, Илья Лебедев.*

Updated by Kirill Kornyakov over 1 year ago #23

Сейчас закрою форум за ненадобностью, копирую материалы сюда.

1. Можно сделать "фольклерную лекцию" про паттерны и антипаттерны. Под этим я понимаю довольно неформальную лекцию, целью которой является раздать студентам полезные ссылки. Можно включить разные байки из прагматика и продуктивного. Короче все то, о чем хотелось бы сказать, но непонятно куда засовывать.
2. Вообще говоря вполне нормальная лекция может быть про рефакторинг. Технический долг, экономика, какие-то advanced советы, например та же инструментальная поддержка.

Есть несколько идей на обсуждение:

1. Открыть проект (не этот, а спецкурса) в свободный доступ. Идея в том, чтобы тем самым показать пример. У нас довольно зрелый процесс и неплохо наполненный проект, думаю другим было бы полезно посмотреть на всю красоту.
2. Отдавать ребятам их листочки с летучек с нашими замечаниями. Им нужна обратная связь, в противном случае они пребывают в неведении.
3. Возможно стоит запостить наши презентации на http://www.slideshare.net. Вроде скачать при этом презентации нельзя, но просмотреть можно будет. Это я для коллабораторов. Впрочем вариант открыть лекции студента тоже рассматриваю.
4. Список вопросов предлагаю расширить, грубо говоря на несколько вариантов. Тогда мы можем не стесняясь публиковать его заранее. Кроме того нам самим полезно составить список этих вопросов из keypoints лекции, тогда хоть будет понятнее на что акцент делать. Плюс возможно студенты начнут задавать вопросы, прослушав лекцию и не найдя ответы на наши вопросы.

Updated by Kirill Kornyakov 8 months ago #24

Артем, спасибо!

Вообще у нас планы грандиозные, и они отчасти покрывают некоторые ваши замечания. Как бы то ни было, мы постараемся учесть все. Так, к следующему году я планирую видеокасты сделать, в которых будет показано как делать лабы. Начиная от скачивания IDE, заканчивая последними лабами. Думаю это сэкономит время на лекциях, и будет отличным подспорьем для отстающих.

Также курс планируется перевести на Eclipse и Java. На самом деле это даже усложнит кое-где лабы, но позволит работать на Linux и Mac, снимет вопросы лицензионности софта, и может быть мы попросим клиенты для Android. Но останется одна платформа на всех - иначе мы замучаемся проверять. Постоянно хотят писать на C++ и Python, но это сложно будет проверять.

Про мотивацию тоже согласен. На самом деле это правило есть, что если ты все сделал вовремя, то у тебя есть автомат. Просто как-то сразу не задалось, да и самим нужно много времени тратить на своевременную проверку. Но я думаю если будут видеокасты и более четкие описания и требования, то мы сможем лучше мотивировать студентов.

Курс скорее всего останется спецкурсом, но когда мы его доделаем, я подниму вопрос о переводе его курс на 4. Все-таки рано его тоже нельзя давать, нужно чтобы люди хотя бы уже видели чужой код, а многие читали лишь свои лабы! НО, параллельно вынашивается план по разработке курса по инструментам разработки. И уж он нацелен на 2 курс, причем для всего потока ВМК. Там планируется рассказать про вещи, которые есть на любых платформах: текстовые редакторы, командная строка, системы контроля версий, билд системы, системы модульного тестирования, багтрекеры, современные хостинги и мир open source. К сожалению это перспектива не близкая, но было бы здорово года через два его опробовать. Будет интерес - присоединяйтесь =)

А так, большое спасибо за внимание! Надеюсь что и вы, когда опыта будет становиться все больше и больше, станете делиться им с подрастающими поколениями. Я слышал что-то планируется сделать в рамках семинара ИТЛаб - дело хорошее, мы кстати от Itseez в этом семестре тоже планируем ряд докладов сделать. Так что хвост пистолетом!

Спасибо,
Кирилл

2013/1/23 Artem Kalachev <artem.kalachev@me.com>

Кирилл,
добрый вечер.

Как и договаривались, присылаю feedback на курс:

1. Первое, хотел сказать искреннее спасибо за проведенный курс. Он был действительно интересный и полезный. Без лишней лести, этот курс можно назвать моим любимым за все 6 лет :-). Хотя на мой взгляд можно проводить этот курс и на более ранних курсах. На 2ом курсе это еще рано, но на 3тьем, а скорее 4ом курсе самое то. Данные знания будут просто незаменимы, и позволит ребятам применять эти знания в своих проектах (домашних, ITLab'вских, дипломах) и на работе.
2. Привязка на технологии .NET (в частности C#). Для меня наверно это было самым большим неудобством. Было бы удобно, если бы был выбор технологий, на которых можно было вести разработку. С другой стороны, за этот семестр, вдобавок ко всему, мне удалось освоить азы .NET. Возможно, просто стоит внедрить в курс некоторые особенности/ключевые моменты .NET, для облегчения процесса осваивания.
3. Мотивация. Возможно, студентов стоит больше мотивировать на выполнение лабораторных и сдачу летучек (помимо допуска к экзамену). Например, позволить получить освобождение от экзамена. Как вариант, можно поставить дополнительные условия: сдать все лабораторные и летучки в срок (заранее оговоренный) + дополнительная усложненная лабораторная (на разные уровень). У ребят может появиться цель, которая станет хорошим стимулом.
4. Разный уровень студентов. Думаю не секрет что у студентов совершенно разный уровень и разный опыт в программировании. Это очень хорошо заметно. Некоторые понимают, что это все очень тяжело и бросают эту затею. Другие, которым тоже тяжело борются, но результат получается средний или ниже. Другая противоположность, те студенты, которые все это знают (или думают что знают) и просто начинают халтурить, не понимая зачем им это нужно (для них очень будет полезен пункт 3). В нашей группе есть люди из каждой категории (к двум перечисленным еще добавляется группа середнячков). К сожалению не понятно как решить данную проблему, но часть студентов просто выпадает из учебного процесса.
5. Время занятий. Суббота это, скорее всего, идеальное время для проведения занятий. Почти все в группе работают, и посещаемость по будням была бы значительно ниже.
6. Дополнительные материалы. Лично мне было бы очень интересно, если бы в курс была добавлена лекция про паттерны. Многие их вообще не знают. Также было бы полезно рассказать про системы контроля версий и багтрекеры.
7. Пару комментариев по лекциям:
    1. MVP. Очень не хватает рассказа про MVC. В процессе подготовки была найдена интересная статья (http://www.rsdn.ru/article/patterns/modelviewpresenter.xml), мне кажется рассказывать про MVC и с чего все началось очень полезно.
    2. Лекция Clean Code и Refactoring - немного сложные для восприятия и для запоминания. Идея понятна, на запомнить, как и что нужно именовать сложно – все равно это приходит с практикой. Более того в разных технологиях правила именования различаются.
    Еще раз большое спасибо,

    Артем.