Опыт сочетания теории и практики в обучении программной инженерии[1]

Д.В Кознов, Я.А Кириленко, Санкт-Петербургский государственный университет

Введение

В настоящее время в России ведутся активные работы по созданию и утверждению программ бакалавров и магистров по программной инженерии [1,2] в соответствии с международными образовательными стандартами IEEE/ACM Software Engineering Curriculum 2004 [3]. Министерство образования и науки утвердило программу бакалавров информационных технологий (номер 010400), идет разработка аналогичной программы для магистров. Однако очень остро стоит задача эффективной реализации этих программ в рамках отдельных вузов. Любые стандарты (как зарубежные, так и российские) фиксируют лишь необходимые знания и компетенции, которые студенты должны получить в рамках данной специальности. Но вопрос о том, как организовать учебный процесс, остается открытым. Здесь и не может быть единого решения: наладка учебного процесса в отдельно взятом вузе является уникальной, творческой задачей, зависящей от его кадрового потенциала, традиций, связей и прочего. При этом возможно применение различных методов и подходов.

Необходимо отметить, что преподавание программной инженерии имеет одну объективную трудность – студентов необходимо обучать решению задач и проблем, с которыми они не знакомы на практике [4]. Выходом здесь может быть имитация подходящей практической ситуации путем специальных упражнений и заданий, а также проведение тренинговых занятий – см. [5-7]. Однако возникает вопрос, как вписать различные нестандартные подходы в традиционное (в смысле принятых в России образовательных форм - лекции/семинары) учебное расписание.

---

[1] Предложенные здесь подходы опробованы в рамках ряда педагогических проектов лаборатории СПРИНТ, созданной в СПбГУ при поддержке корпорации Intel. Эксперименты с картами памяти выполнены в рамках проекта «Collaborative Learning with Comapping», поддержанного компанией Hewllett-Packard.

.

Данная статья посвящена изложению опыта преподавания программной инженерии на математико-механическом факультете СПбГУ. Основная задача, на которой делается акцент в работе – поиск путей максимально полного практического освоения лекционных знаний. В статье обсуждается идея единых теоретико-практических учебных модулей, необходимость постоянной обратной связи от студентов во время учебного процесса, необходимость уменьшения объема теоретических знаний в сторону лучшего практического освоения оставшейся части, полезность «живых» примеров. Предлагается идея создания учебных тренингов на основе традиционных экзаменов. Рассказывается об использовании карт памяти [8] для организации обратной связи от студентов и экзаменов-тренингов, обсуждаются дополнительные практические формы

Теоретические занятия

Теоретические занятия в виде лекций различного формата – потоковых, специальных курсов и пр. – являются традиционными для университетского образования. Они позволяют одномоментно давать большому количеству людей одну и ту же информацию. Вопрос в том, могут ли слушатели эффективно воспринять эту информацию.

При чтении лекций, например, по тестированию, конфигурационному управлению, процессам разработки, архитектуре и образцам проектирования и пр. очень часто наблюдается следующая картина: студенты слушают, конспектируют (тщательно или не очень), мало задают вопросов – им вроде бы все понятно. Однако самые простые проверки знаний показывают, что они в лучшем случае запомнили материал, но очень далеки от его понимания. Ценность таких знаний невелика – они плохо складываются в общую картину, быстро забываются и оказываются совершенно бесполезными на практике.

К сожалению, не все лекции по программной инженерии имеют ярко выраженный акцент на практической применимости излагаемой информации. Существует глубоко укоренившееся среди педагогов мнение о необходимости общих знаний, общего образовательного уровня, общей ориентации в предмете и т.д. Не отвергая полезность этой точки зрения, отметим, что часто такой подход маскирует непонимание или просто нежелание разбираться, как излагаемый материал может и должен применять на практике.

Стоит также отметить, что лекции часто строятся на научных достижениях в области программной инженерии и изобилуют понятиями, концепциями и методами, которые без соответствующего контекста (основой которого является практический опыт) порождают

недоумение и непонимание. Необходимо перенастраивать лекции на возможность практического освоения студентами учебного материала, оставляя глубину раскрытия теоретических основ.

Отсутствию практической направленности лекций соответствует следующая позиция студентов. Они механически слушают, конспектируют и пытаются запомнить (а потом сдать и забыть), не пробуя «примерить знания на себя», собрать все в единую картину, осознать, осмыслить, чем все это полезно именно им в их практической деятельности. Воспитывается ужасный стереотип: обучение – это скучная, ненужная деятельность. Следствием этой позиции является потеря студентами интереса к занятиям в университете до их полного игнорирования. Тем более, что имеется множество возможностей уже на первых курсах университета устроиться на работу в промышленную компанию и получать опыт и знания (вместе с зарплатой!) уже там. Однако это не очень хороший выход – таким новичкам часто предлагают неквалифицированную работу, которая мало чему учит (например, простое тестирование, программирование Web-сайтов), либо же происходит ранняя узкая специализация, что приводит к ущербности базовых знаний и навыков, отсутствию профессионального кругозора.

Мы преодолеваем эти проблемы следующим образом.

- Организация обучения на основе единых теоретико-практических модулей. Например, вводный курс по программной инженерии для бакалавров II курса читается по две пары в неделю в классе, где есть компьютеры. Таким образом, есть возможность чередовать различные упражнения с лекциями в произвольном порядке, продиктованном особенностями именно этих студентов, тем, как в этом случае складывается процесс усвоения материала. Студенты, прослушав теоретическую тему, сразу же выполняют соответствующие упражнения, например, по работе с требованиями (извлечение требований, создание технических заданий), пишут простые build-скрипты, осваивают базовые техники проектирования на основе UML и т.д. При этом теоретический материал не дается сразу весь, а перемежается с упражнениями. После прохождения нескольких таких модулей студенты могут уже принять участие в учебном групповом проекте (например, студенческом проекте – см. ниже). Им уже не надо объяснять элементарные вещи, некие базовые навыки они уже имеют – все это необходимо закреплять более реалистичными практическими заданиями. Ограничение обучения в рамках таких модулей – возможность работы с относительно небольшой группой студентов (до 20-ти человек), необходимость высокой и разносторонней квалификации

преподавателя и существенных эмоциональных затрат. Мы считаем, что возможны различные варианты организации преподавания таких целостных теоретико-практических модулей.

- «Лучше что-то не успеть, чем что-то не понять» – гласит плакат в одной из семинарских аудиторий в Петербургского отделения математического института имени Стеклова (ПОМИ). Мы абсолютно согласны с этим тезисом. Во многом, мифическая «полнота охвата» предмета часто происходит в ущерб пониманию и осознанию отдельных аспектов и всей картины в целом (на это просто не остается времени и сил!), а также приобретению действительно полезных и нужных практических навыков в этой области. Поэтому мы при необходимости смело уменьшаем объем теоретических знаний в курсах, которые не поддаются практическому освоению, и более углубленно прорабатываем оставшийся материал.
- Использование большого количества «живых» примеров в лекциях. Лучше всего из опыта преподавателя, но, разумеется, можно использовать и чужие примеры. В последнем случае они должны быть досконально осмыслены преподавателем. Образцы таких хорошо осмысленных и разобранных примеров можно найти, например, в курсе по программной инженерии [9].
- Постоянный сбор обратной связь от студентов во время чтения лекций и проведения занятий. Это позволяет возвращаться к отдельным темам, давать дополнительные задания и формулировать просьбы к смежным курсам и практическим семинарам.
- Особая организация экзамена. Известно, что активность студентов при подготовке к сдаче экзамена существенно повышается по сравнению с остальным временем обучения. К этому можно по-разному относиться, но это реальный жизненный факт. Мы предлагаем его использовать. Мы разбиваем экзаменующихся на группы подходящего размера, готовим специальные задания и организуем «погружение» на несколько часов для данной группы. Бывают очень удачные, продуктивные тренинги, значительно повышающие у студентов осознанность, целостность и связность изученного материала. Более подробно этот подход описан в работе [10].

Одним из интересных инструментов взаимодействия преподавателей и студентов являются карты памяти (Mind Maps) – техника, предложенная и развитая английским психологом Тони Бьюзеном [8] в конце 70-х годов прошлого века. Она очень простая и используется при работе с какой-либо информацией, для ее структурирования, осмысления, лучшего усвоения и запоминания. На листе бумаги, в

центре, рисуется объект, обозначающий ту тему или предмет, который мы рассматриваем. Далее рисуются вторичные объекты, которые поясняют и уточняют данный и соединяются с ним дугами. И так далее. Эта техника очень хорошо подходит для быстрой диагностики уровня знаний студентов (карты памяти быстро создаются при наличии навыка), для обзора общей картины (проверить 20-30 карт памяти можно очень быстро), а также для проверки усвоения знаний в таких областях, которые сложно охватить семинарами и упражнениями – например, различные методологии разработки ПО (CMMI, RUP, MSF). Карты памяти также помогают построить общую картину изученного знания, осознать дополнительные связи в материале. Они могут использоваться также совместно с техникой реструктуризации знаний, которая хорошо себя зарекомендовала, например, при выявлении требований к ПО [11]. Мы рисуем карты памяти «в ручную», а также используем программный продукт Comapping (www.comapping.com), в разработке которого сотрудники нашего университета принимают активное участие. Более подробно использование карт памяти в преподавании программной инженерии изложено нами в работах [10,12].

Дополнительные практические занятия

Упражнений в компьютерном классе не достаточно для формирования полноценного специалиста, хотя такие упражнения и являются полезными и необходимыми - в составе теоретико-практического модуля или в виде отдельных заданий (например, для обучения программированию списков, очередей, конечных автоматов, алгоритмов сортировки и т.д.) Для организации дополнительной практической работы студентов мы используем следующие подходы.

- С первого курса, при выполнении студентами любых упражнений, мы начинаем предъявлять требования к архитектуре создаваемых программных приложений (например, четкое разделение логики и пользовательского интерфейса программы), оформлению программного кода, созданию возможностей для повторного использования. Все эти навыки требуют именно воспитания, а не только лекционных знаний, многократного повторения, «заходов» с разных сторон.
- Персональное кураторство студента опытным преподавателем в рамках обязательных курсовых и дипломных работ – студенты должны научиться программировать. Темы курсовых и дипломных работ могут быть абсолютно из разных областей науки (в настоящем году – от лингвистики до биоинформатики), но особо отслеживается качество выполнения программной составляющей.

- Студенческие проекты совместно на базе бизнес-компаний. Они могут иметь формат летних учебных практик, производственных практик, дополнительных занятий в семестре. Здесь студентов обучают основам производственной культуры: работе со средствами контроля версий, этике электронной переписки, модульному тестированию, экспертным просмотрам кода (peer code review), составлению технических документов, практическому проектированию и др. Однако, при этом важно, чтобы такие проекты не были сугубо прагматичными, преследующими только цели компании (обучение студентов конкретным технологиям, использующимися ею в работе), а действительно расширяли бы профессиональный кругозор студентов. Важно также, чтобы был образовательный поток, а не нацеленность таких проектов на поиск подходящих кадров для компании. Кроме того, необходимо решать вопрос с материальным стимулированием студентов. Все это должно быть зафиксировано в соответствующих договоренностях между университетом и компанией. Одна из практик организации таких студенческих проектов, применяемая нами, изложена в [13].
- Участие студентов в научно-исследовательских проектах. Это позволяет привлечь студентов к интересным задачам и именно на их фоне проводить обучение различным неявным навыкам, о наличии и важности которых указывается, например, в [14]. К сожалению, в последние полтора десятилетия культура научных проектов в российских университетах и исследовательских институтах по разным причинам упала, и для ее возобновления требуются дополнительные усилия.

Заключение

В этой статье мы изложили те реальные проблемы, с которыми мы сталкиваемся при преподавании программной инженерии и те методы, которые мы используем для их преодоления. Наши усилия последних лет были направлены на наладку преподавания в рамках специальности по подготовке бакалавров 010400 по принципам, изложенным в [2]. Эту работу еще нельзя назвать в полной мере завершенной. Вот те проблемы, которые еще предстоит решить:

- подбор компетентных кадров по всем предметам учебной программы, сочетающих практический опыт в данной дисциплине и педагогические способности;
- создание скоординированных друг с другом программ для каждого курса;
- организация групповых студенческих проектов разных форматов.

Список литературы

[1] В.А. Сухомлин. ИТ-образование: концепция, образовательные стандарты, процесс стандартизации. М.: Горячая линия-Телеком, 2005.

[2] А. Н. Терехов, А.А.Терехов. Computing Curricula: Software Engineering и российское образование. // Открытые системы. 31/10/2006 № 08.

[3] Рекомендации по преподаванию программной инженерии и информатики в университетах. Перевод с английского. – М. ИНТУИТ 2007, 462 с.

[4] E. Stiller, C.LeBanc. Effective Software Engineering Pedagogy. Journal of Computing Sciences in Colleges, Volume 17, Issue 6, May 2002, pp. 124-134.

[5] V.L. Pavlov, N.Boyko, A. Babich, O.Kuchaiev, S.Busygin. Applying Pantomime and Reverse Engineering Techniques in Software Engineering Education. 37th ASEE/IEEE Frontiers in Education Conference, October 10 – 13, 2007, Milwaukee.

[6] B. Bracken. Progressing from student to professional: the importance and challenging of teaching software engineering. Journal of Computing in Colleges, Volume 19, Issue 2, December 2003, pp 358-368.

[7] R. Dawson Twenty Dirty Tricks to Train Software Engineers. Proceedings of the 22nd international conference on Software engineering, 2000, pp 209-218.

[8] Т. Бьюзен, Б.Бьюзен. Супермышление. / Пер. с англ. Мн.: ООО “Попури”, 2003, 304 с.

[9] В. В. Кулямин Технологии программирования. Компонентный подход. Учебное пособие. М: ИНТУИТ 2007. 463 с.

[10] Д.В.Кознов. Методика обучения программной инженерии на основе карт памяти. // В сб. «Системное программирование». / Вып. 3, под ред. А.Н.Терехова и Д.Ю.Булычева. СПб: Изд. СПбГУ, 2008. С. 121-140.

[11] В. В. Кулямин, Н. В. Пакулин, О. Л. Петренко, А. А. Сортов, А. В. Хорошилов. Формализация требований на практике. Препринт РАН. 2006 г. 50 с.

[12] D.Koznov, M.Pliskin. Computer-Supported Collaborative Learning with Mind-Maps. T. Margaria and B. Steffen (Eds.): ISoLA 2008, CCIS 17, pp. 478–489, 2008. Springer-Verlag, Berlin Heidelberg, 2008.

[13] Р.К.Гагарский Программа подготовки специалистов в IT-компании. // В сб. «Системное программирование». / Вып. 3, под ред. А.Н.Терехова и Д.Ю.Булычева. СПб: Изд. СПбГУ, 2008. С. 141-156.

[14] А.К.Петренко, О.Л.Петренко, В.В.Кулямин. Роль научных организаций в подготовке IT-специалистов. // Труды ИСП РАН, т. 15. М. 2008. С.41-50.