АРХИТЕКТУРА СИСТЕМЫ ОБУЧЕНИЯ КОЛЛЕКТИВНЫМ МЕТОДАМ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Бежанишвили Л., Гогишвили З.
Грузинский Технический Университет

Резюме

Совершенствование проектирования программного обеспечения и связанных с ним процессов способствует созданию надежного программного обеспечения, оптимизации стоимости разработки и появлению программ, требующих низких затрат на техническую поддержку. При создании высоконадежных программных комплексов важное значение приобретает правильная организация управления проектами. Кроме того становится ясно, что коллектив разработчиков должен быть готовым к использованию предлагаемых ему методов и средств коллективной разработки. Следовательно, большое значение имеет построение системы, обучающей коллективным методам разработки программного обеспечения, архитектура которой предлагается в данной статье.

Ключевые слова. Архитектура системы обучения. Коллективная разработка. CMM. Microsoft Team Foundation Server.

1. Введение

В процессе разработки программного обеспечения, которое должно отвечать требованиям к конфиденциальности, целостности и готовности, выявились серьезные проблемы. Для их решения необходим персонал с соответствующим уровнем образования, подготовкой и опытом. Именно по этой причине на передний план исследований стала выдвигаться концепция гарантии качества разрабатываемой системы. Качество разрабатываемой программной системы в соответствии с этой концепцией объявлялось свойством системы, которое должно быть планируемым и управляемым.

На практике проблемы управления гораздо чаще становятся причинами крушения проектов, чем неудачные технические решения. Понимание этого факта стало основополагающим при создании модели CMM (**Capability Maturity Model** – Модель Зрелости Возможностей**)**, которая определяет основные группы процессов разработки, формулирует характеристики разных уровней зрелости этих процессов и дает практические рекомендации по совершенствованию процессов для достижения ими определенного уровня.

Возникла необходимость обучения современным методам коллективной разработки, включающим накопление знаний на основе решения предыдущих задач, их обобщение и применение на новых, ранее не рассмотренных задачах. Таким образом, создание обучающей системы, очень актуальная задача для предварительной оценки возможностей организации, а также обучения принципам организационной культуры.

2. Основная часть

С целью облегчения процесса обучения коллективным методам разработки программного обеспечения предлагается использование интеллектуальной обучающей/когнитивной системы на базе модели CMM (**Capability Maturity Model).** Назначением системы является *обучение* методам коллективной работы, а также *оценка возможностей* коллектива разработчиков – исполнителей, определяя наиболее приоритетные шаги для улучшения возможностей "программирующей организации". Назначение системы состоит в содействии целям и согласованным оценкам возможностей потенциальных исполнителей разработки ПО в соответствии с методами современной *технологии программирования* и *обучение* их этим методам. Такие оценки должны проводится либо в процессе квалификации предложений о работах, либо в процессе выбора формальных исполнителей, либо в обоих этих случаях.

Для реализации инженерных аспектов системы и проектировании базы данных в системе использованы программные продукты Microsoft Team Foundation Server и Microsoft SQL Server. На рис.1 представлена архитектура системы, обучающей коллективным методам разработки прграммного обеспечения на базе данной модели. Правильная работа системы в значительной степени определяется одним из основных ее компонентов – информационной базой системы. Это сложный компонент, содержащий знания о проектных прецедентах (лучших практиках), процессах, квалификационных тестах.

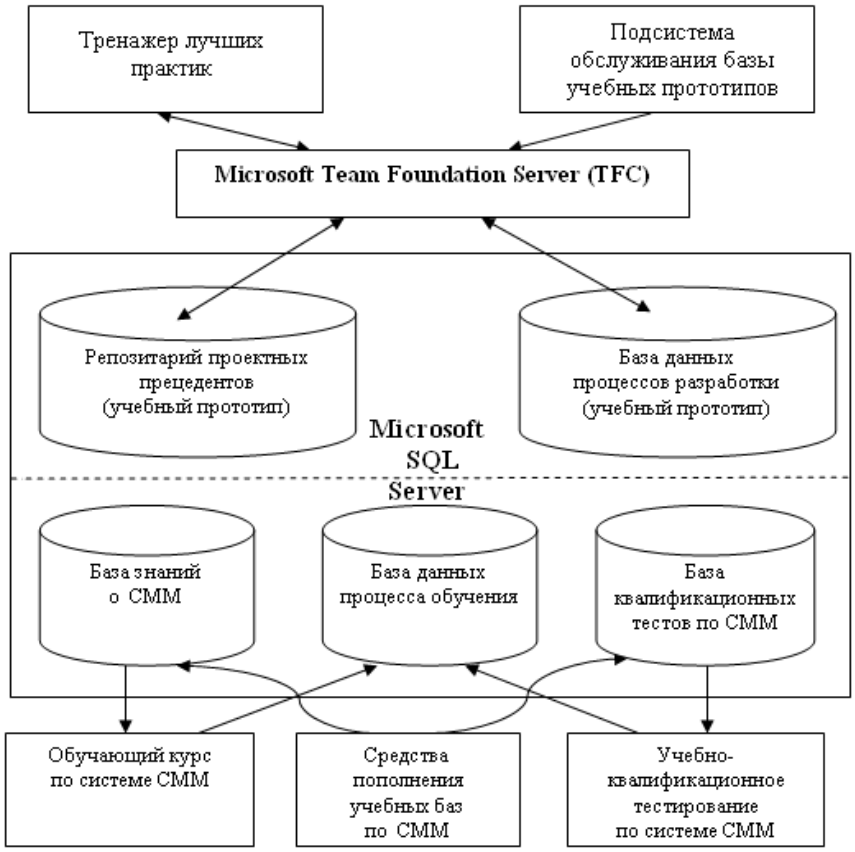

Рис.1. Архитектура системы обучения коллективным методам разработки программного обеспечения

База данных процессов содержит данные по производительности завершенных проектов. Она включает данные по рискам, трудоемкости, ошибкам и их распределению, размеру ПО и другим характеристикам проектов. Менеджер проекта может лучше спланировать свой проект, если имеет доступ к опыту, накопленному в завершенных проектах. Для этого существует репозитарий проектных прецедентов. В качестве языка представления базисных знаний выбран предусмотренный для анализа и проектирования объектно-ориентированный универсальный язык моделирования UML. Соответственно разработаны механизмы объектно–ориентированного представления знаний с помощью диаграмм UML и методы и алгоритмы перевода их в семантические сети. Представление в виде семантических сетей осуществлено с использованием языка XML. Компонента обучения на базе модели CMM предназначена для обучения основным принципам правильной организации процесса разработки программного обеспечения; компонента тестирования определяет уровень зрелости разработчиков.

С одной стороны, она может быть использована как помощь для организаций, разрабатывающих ПО, при их внутренней оценке их собственных возможностей в этой области. С другой стороны, система предназначена для оценивании возможностей субподрядчиков/исполнителей в области *технологии программирования* (ТП), а также для их обучения, она также может быть значима в оценке возможностей в области разработки программных систем для конкретных коллективов исполнителей.

Так как понимание подходящей практики ТП сейчас только образуется, стандартных, широко признанных методов измерений в этой области еще не существует. Методология оценивания основана на принципе, в соответствии с которым предварительный опыт есть лучшая основа для предсказания будущих характеристик. Процесс оценивания фокусируется на определении и прояснении положительных атрибутов хорошего опыта в ТП. Программа обучения предусматривает несколько занятий по обзору оценочных вопросов в деталях и обсуждению материалов и поддерживающих средств, которые доступны для демонстрации характеристик каждого вопроса.

Кроме того, предусмотрена компонента, осуществляющая тренаж на основе лучших практик завершенных проектов (проектных прецедентов). Прецедент - это описание проблемы или ситуации в совокупности с подробным указанием действий, предпринимаемых в данной ситуации

или для решения данной проблемы. Вывод на основе прецедентов - это метод принятия решений, в котором используются знания о предыдущих ситуациях или случаях (прецедентах). При рассмотрении новой проблемы (текущего случая) находится похожий прецедент в качестве аналога. Можно попытаться использовать его решение, возможно, адаптировав к текущему случаю, вместо того, чтобы искать решение каждый раз сначала. После того, как текущий случай будет обработан, он вносится в базу прецедентов вместе со своим решением для его возможного последующего использования. Прецедент включает: 1.*Описание проблемы; 2.Решение этой проблемы;3. Результат (обоснованность) применения решения.*

Описание проблемы должно содержать всю информацию, необходимую для достижения цели вывода. Описание результата может также включать ссылки на другие прецеденты, текстовую информацию.

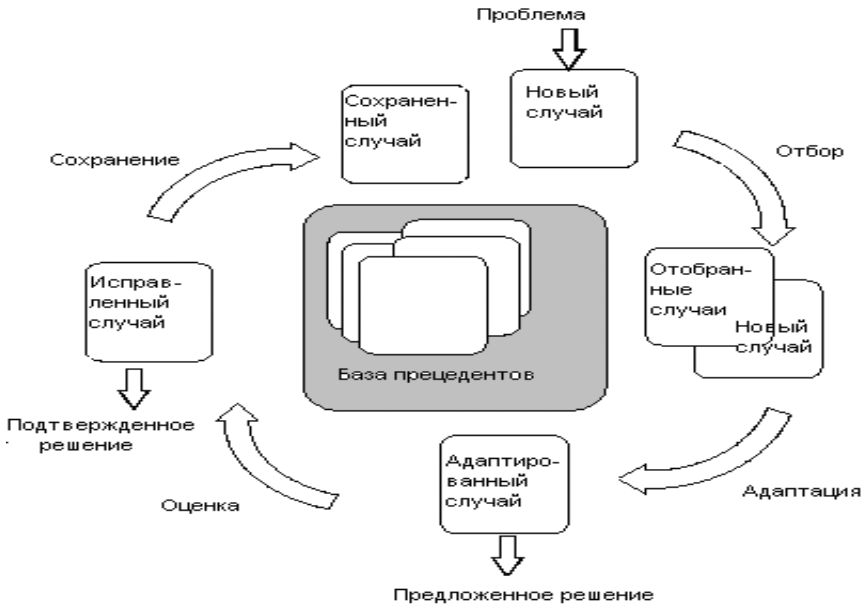

Рис. 2. Выбор решения на основе прецедентов

Наполнение базы прецедентов может происходить как до момента начала управления на основе априорной информации, с помощью реальных или смоделированных прецедентов, так и в процессе управления, после обработки итога управляющего воздействия.

После применения регулирующего воздействия и оценки итога этого воздействия текущая ситуация превращается в прецедент, который заносится в базу прецедентов. Отрицательный результат также является информативным и заносится в базу. После того, как прецеденты извлечены, нужно выбрать "наиболее подходящий" из них. Это определяется сравнением признаков в текущей ситуации и в выбранных прецедентах. Затем выбирается способ измерения степени близости прецедента и текущего случая по каждому признаку (будь это текстовый, числовой или булевский), который пользователь сочтет полезным для достижения цели.

Вводится метрика (расстояние) на пространстве всех признаков, в этом пространстве определяется точка, соответствующая текущему случаю, и в рамках этой метрики находится ближайшая к ней точка из точек, представляющих прецеденты. Каждому признаку назначают вес, учитывающий его относительную ценность. Полностью степень близости прецедента по всем признакам можно вычислить, используя обобщенную формулу типа:

$$\frac{\sum_j w_j * sim(x_{ij}, x_{kj})}{\sum_j w_j}$$

где wj - вес j-го признака, sim - функция подобия (метрика), xij и xkj - значения признака xj для текущего случая и прецедента, соответственно. После вычисления степеней подобия для всех прецедентов получаем их единый ранжированный список. В управлении важен еще и результат воздействия, то, насколько он приближает к цели. При вводе метрики можно учесть и этот критерий. Считая, что цель должна быть достигнута за конечное число шагов, можно считать более близким прецедент, позволяющий достичь цели за меньшее число шагов.

При вычислении расстояния каждому прецеденту ставятся в соответствие дескрипторы описания, состоящие из кортежей ({…,Ci,…} типа Ci=<н,з,в,о>, где н – наименование свойства; з – его значение; в – важность или информационный вес свойства; о – ограничение на интервал допустимых значений). Ограничение определяет интервал значений, в рамках которого значение свойства может определять значение меры подобия. Сложность поиска решения и выявления различий между прецедентами в значительной степени зависит от используемых термов индексации. Каждому свойству (или размерности) присваивается определенный вес, соответствующий степени "важности" этого свойства. Из базы прецедентов выбирается тот, который "заслужил" самую высокую оценку. Вычисленное значение обычно называется агрегированной оценкой совпадения. Если же алгоритм работы системы предполагает и исследование альтернативных прецедентов, то оставшиеся должны быть ранжированы по полученным оценкам. В случае если найденный прецедент не является полным аналогом текущей ситуации, должна выполняться адаптация - модификация решения, которое имеется в выбранном прецеденте и направлено на решение целевой проблемы. Алгоритмы адаптации предполагают наличие зависимости между признаками прецедентов и признаками содержащихся в них решений. Такие зависимости могут задаваться человеком при построении базы прецедентов или обнаруживаться в базе автоматически методами добычи знаний. Процесс модификации решения включает ряд шагов, от простой замены некоторых компонентов в имеющемся решении, корректировки или интерполяции (числовых) признаков или изменения порядка операций до более существенных действий.

3. Заключение

Применение эффективных методов управления проектами увеличивает шансы на успех создания качественного программного продукта. Таким образом, создание обучающей системы, очень актуальная задача для предварительной оценки возможностей организации а также обучения принципам организационной культуры.

Литература:

1. Панкаж Дж.. Управление программным проектом на практике. "Лори", М., 2005
2. Руководство к Своду знаний по управлению проектами (Руководство PMBOK®). *Project Management Institute*, Inc., 388, Америк.нац. стандарт ANSI/PMI 99-001-2004
3. Herbsleb O. et al. Benefits of CMM-Based Software Process Improvement: Initial Results, tech.report CMU/SEI-94-TR-013, Software Engineering Institute, Carnegie Mellon University, 1994.

პროგრამული უზრუნველყოფის შექმნის კოლექტიური მეთოდების მასწავლი სისტემის არქიტექტურა

ლოლიტა ბეჟანიშვილი, ზურაბ გოგიშვილი
საქართველოს ტექნიკური უნივერსიტეტი

რეზიუმე

პროგრამული უზრუნველყოფის პროექტირების და მასთან დაკავშირებული პროცესების სრულყოფა იწვევს დამუშავების ფასის შემცირებას და პროგრამების გამოჩენას, რომელთა ტექნიკური მხარდაჭერის ხარჯები დაბალია. საიმედო პროგრამული კომპლექსების შექმნისას განსაკუთრებული მნიშვნელობა ენიჭება პროექტის მართვის სწორ ორგანიზებას. გარდა ამისა, ცხადია, რომ პროგრამული უზრუნველყოფის შემქმნელი კოლექტივი უნდა იყოს მზად დამუშავების შემოთავაზებული მეთოდების და ხერხების გამოყენებისთვის. აქედან გამომდინარე ძალიან მნიშვნელოვანია პროგრამული უზრუნველყოფის შექმნის კოლექტიური მეთოდების მასწავლი სისტემის აგება, რომლის არქიტექტურაც შემოთავაზებულია ნაშრომში.

ARCHITECTURE OF THE TRAINING SYSTEM FOR COLLECTIVE METHODS OF SOFTWARE DEVELOPMENT

Bejanishvili Lolita, Gogishvili Zurab
Georgian Technical University

Summary

Perfection of software design methods and associated design processes promotes optimization of expenses due to low technical support costs. Creation of software complexes with high reliability, the correct organization of project management has high importance. Besides, it becomes clear, that the team of software developers should be ready to use offered methods and instruments of collective software development. Therefore, very important is creation of the system able to train to collective methods of software development, which architecture is represented in this article.