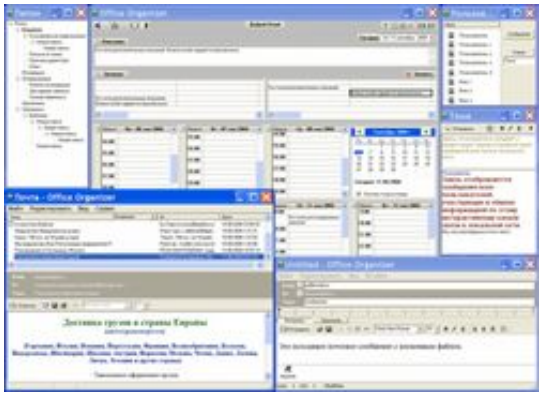
Проектирование программных систем с применением дополненных знаний



Автор статьи:
А.И. Разумовский,
к.т.н., старший научный сотрудник
Института проблем управления РАН
г. Москва, Российская Федерация

A.I. Razumowsky
Ph.D., Senior Research Fellow
Institute of Control Sciences RAS
Moscow, Russian Federation
razumowsky@yandex.ru

Аннотация: Дополненные знания (ДЗ) – это зафиксированные факты и фрагменты процесса проектирования. Они включают в себя сопроводительные данные – примечания, фотографии, спецификации, требования, а также комплексные данные, – объединенные в общем графическом контексте информационные элементы проекта и сопроводительные данные. Основное предназначение ДЗ – выработка и принятие проектных решений посредством опоры на эвристику доступности, которая действует как оценка частоты или вероятности выбора за счет легкости воспоминания или ассоциации, где воспоминания и ассоциации – зафиксированные ДЗ.

DESIGN SOFTWARE SYSTEMS USING ADDITION KNOWLEDGE

Abstract: The addition knowledge (AK) - is recorded facts and fragments of the design process. Its include supporting data - notes , photos, specifications, requirements , and complex data - combined in the overall context of the graphical elements of the project information and supporting data . The main purpose of AK - development and adoption of design solutions through reliance on the availability heuristic , which acts as an estimate of the frequency or probability of selection at the expense of ease of memories or associations , where memories and associations - fixed AK.

Keywords: addition knowledge, design context, software system, availability heuristic.

Проблемы, возникающие при проектировании программных систем (ПС), так или иначе связаны с недостаточной поддержкой творческих способностей человека и в срезе имеющихся технологий проектирования ПС могут быть выражены следующим образом:

- неполнота информации в процессе проектирования и алгоритмизации;

- узость туннеля видимости результатов алгоритма;

- унификация и деперсонификация данных проекта.

Эти проблемы имеют две основные причины:

- рационализация процесса разработки ПС;

- унификация средств разработки ПС.

В работе [1] авторы сетуют: "Мы никогда не отыщем процесс, который дал бы нам возможность проектировать программы строго рациональным образом". Однако далее они замечают: «Если мы держим в голове идеальный процесс, становится легче измерять успехи проекта", иначе говоря творческие способности проектировщика, считают авторы, должны быть подчинены определенным правилам процесса разработки ПС, что позволит обеспечить его управляемость. Таким образом, главным вопросом проектирования ПС остается проблема соотношения творческого и рационального в поиске, выработке и принятии решений. Сужение области творчества разработчиков приводит к слабому профессиональному росту, безынициативности, небрежности [2]. Более того, в работе [3] имеется следующее замечание: "Ограничения, накладываемые общей логикой и абстрактной математикой на процедуры принятия решений, практически исключают возможность изучения истинно творческих решений и сводят науку о решениях к совокупности механических, а потому скучных и однообразных примеров принятия решений". Поскольку творческий процесс является спонтанным, необходимо учесть, что рожденная идея также может и не быть напрямую связана с решаемой проблемой [4]. Последнее означает, что при проектировании ПС необходимо заручиться возможностью зафиксировать все приходящие и

имеющие информационно-техническое представление идеи. Фиксация рожденных идей, ассоциаций и оценок необходима также еще и из-за такой специфической особенности когнитивно-творческого аппарата человека, как ограниченный объем его кратковременной памяти – число Миллера: 7 +-2 [5]. В книге акад. Ларичева [6] дается предположение о двух основных путях решения человеком задач: «а) объединение отдельных единиц информации в блоки с целью одновременного восприятия этих блоков, б) использование специальных эвристик, приспосабливающих задачу к возможностям системы обработки информации человеком».

Математических моделей, конструктивно описывающих творческий человеческий фактор (ТЧФ), не существует. Конструктивно – значит, как если бы математической формулой можно было бы заменить определенную интеллектуальную функцию ТЧФ, или точнее – его творческий импульс. Нет никакой надобности создавать – искать, отлаживать – математические выражения, описывающие нечто не являющееся ТЧФ или его деятельностью. Все такие модели в ту или иную меру используют вероятностный подход, основанный на субъективной экспертной оценке, и, следовательно, являются неадекватными. Мы предлагаем обустраивать ТЧФ индивидуально – тогда субъективность представления не только не будет помехой, но, наоборот, послужит важнейшей качественной характеристикой процесса участия ТЧФ. Как это организовать?

Когда во главу угла ставится индивидуальная значимость, тогда речь может идти о личной ответственности за выработку и принятие решения. В этом видится ресурсоемкость процесса проектирования ПС. Сошлемся на теоретика менеджмента П. Друкера, который одно из наиболее важных свойств эффективного управления определяет как собственный вклад: «Человек, который заботится о своем вкладе и принимает на себя ответственность за результат, является звеном в высшей цепи управления» [7]. Ориентация на вклад – это по существу ориентация на эффективность.

ТЧФ субъективно оценивает локальную задачу и творчески вырабатывает решение. Имеется два класса информации, обуславливающие выработку решений, – реальная информация и воображаемая информация или перцептивное воображение [8]. Первая характеризуется всеми видимыми элементами процесса проектирования ПС: языковое и структурное представление алгоритма, комментарии, отладочные данные, интерфейсные элементы среды разработки. Воображаемая информация – это то, что скрыто от глаз и рук проектировщика, но ассоциативно и интуитивно через множество озарений прокладывает дальнейший путь развития проекта ПС. С ценностной точки зрения воображаемая информация имеет перед реальной безусловное преимущество, однако именно воображаемая информация и испытывает наибольшие потери, поскольку ее невозможно впоследствии в полноте воспроизвести, удастся лишь частично припомнить антураж творческого акта выработки решения.

Мы предполагаем, что воображаемую информацию можно зафиксировать с целью ее многократного повторного использования. Для наименования подобной информации предлагается выбрать термин: дополненные знания (ДЗ). ДЗ – это, главным образом, зафиксированные факты и фрагменты процесса проектирования. Они включают в себя сопроводительные данные – примечания, фотографии, спецификации, требования, а также комплексные данные – произвольно (индивидуально значимо) объединенные в общем графическом контексте (на экране) информационные элементы проекта плюс сопроводительные данные.

По аналогии с дополненной реальностью [9] ДЗ обеспечивают:

- совмещение проектной, аналитической и результатной информации;

- представляются в реальном времени;

- работают в 2D или 3D графических средах.

Основное предназначение ДЗ – выработка и принятие проектных решений посредством опоры на эвристику доступности, которая действует как оценка частоты или вероятности выбора за счет легкости воспоминания или ассоциации, где воспоминания и ассоциации зафиксированные ДЗ.

Аккумулированные вместе ДЗ позиционируются в едином плоскостном или пространственном контексте, который мы назовем креативно-контекстной формой, поскольку определяющим фактором ее значимости будет концентрация и фиксация всех возможных информационных элементов в момент творческого озарения. Это своего рода фиксированные следы памяти – временные связи в коре мозга, служащие физиологической основой запоминания и воспроизведения [10]. Кроме того, аккумуляция и фиксация ДЗ также сообразуются с теорией двойного кодирования А. Пайвио [11], согласно которой познание включает в себя деятельность двух отдельных подсистем: вербальной – обрабатывающей языковую информацию и невербальной (образной), предназначенной для неязыковых объектов и событий.

Для выработки и принятия решения важно иметь в поле восприятия – на экране – входящие данные, аналитические данные (сравнения, уточнения, ошибки), динамику изменений определенных параметров, а также результаты – численные и наглядные.

Рассмотрим пример построения алгоритма определения свойств треугольника при задании длин трех его сторон. То есть требуется найти, к какому виду треугольников – равностороннему, равнобедренному или произвольному будет относиться исследуемый треугольник, заданный тремя целыми числами. Для простоты положим: дано пять отладочных троек: (1,1,2), (2,1,3), (3,5,3), (2,2,2), (2,5,1).

Разобьем процесс проектирования на этапы, каждому из которых будет соответствовать добавление к программному коду хотя бы одного нового оператора или поля данных. Изначально нам нужно создать минимальный по содержанию алгоритм, который позволит получить итоговый результат. Такой результат, скорее всего, будет ошибочным, как и сам алгоритм, однако позволит поместить этот результат в контекст разработки и осуществлять его наблюдение в кругу иных значимых для поиска и принятия решения информационных элементов с целью коррекции и развития алгоритма.

Первый этап проектирования алгоритма завершается реализацией всего одного оператора сравнения двух из трех входных величин (рис. 1). По результатам этого сравнения делается заключительный вывод об искомом свойстве треугольника.

```
if(x==y)
  return "треугольник - равнобедренный"
```

Рис. 1. Программный код первого этапа проектирования алгоритма

Установим соответствующую 1-му этапу креативно-контекстную экранную форму, на которой разместим, кроме программного кода (рис. 1), также рисунок, отражающий в нашем сознании понятие «равнобедренный треугольник». Кроме того, осуществим запуск скомпилированного кода и поместим на форме исходные данные рядом с соответствующими результатами выполнения программы. Для каждой отладочной тройки получаем конечный результат, соответственно: «равнобедренный», «», «», «равнобедренный», «». Как мы видим, правильный результат получен лишь для 1-й тройки – (1,1,2). Таким образом, конечное содержание экранной формы, выражающей состояние поиска и выбора решения следующего 2-го этапа проектирования, будет подобным изображенному на рис. 2. С учетом этого переходим к следующему этапу проектирования.

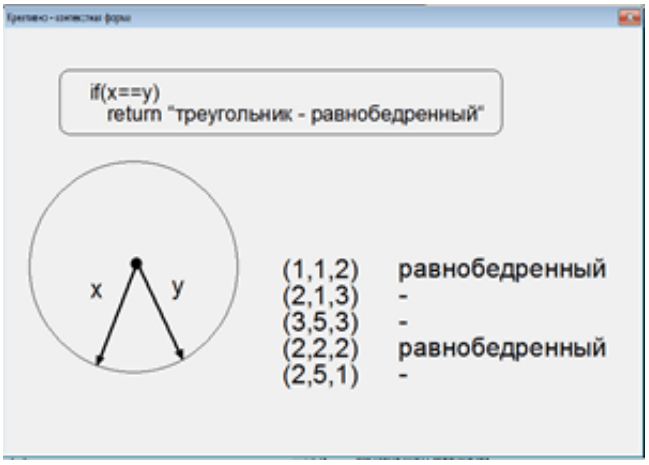


Рис. 2. Креативно-контекстная экранная форма выработки решения 2-го этапа

2-й этап проектирования диктуется необходимостью получения правильных результатов для остальных 4 отладочных троек. Это означает необходимость коррекции оператора, записанного на 1-м этапе. Итак, добавим новый вывод при отрицательном ответе в операторе сравнения (рис. 3).

```
if(x==y)
  return "треугольник - равнобедренный"
else
  return "треугольник - произвольный
```

Рис. 3. Изменение программного кода проекта на 2-м этапе.

Осуществим запуск измененного скомпилированного кода. Аналогично 1-му этапу результаты программы поместим на креативно-контекстную форму (рис. 4). На рис. 4 отражено ее содержание, выражающее состояние требуемого поиска и выбора решения для очередного, 3-го этапа проектирования. Как видно из полученных результатов в сопоставлении с исходными данными, правильных результатов стало три из пяти.

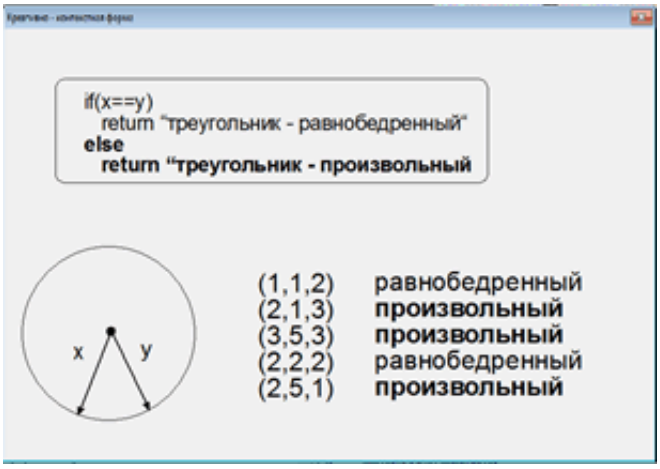


Рис. 4. Креативно-контекстная экранная форма выработки решения 3-го этапа

Действуя подобным же образом, последовательно преодолеваем 3-й и 4-й этапы разработки. На 3-м этапе добавляем условие сравнения для параметра Z, при этом не удаляем, а комментируем неправильный код предыдущего этапа. Следует обратить внимание на то, что в подобной технологии разработки нет необходимости удалять лишние элементы – их всегда можно поместить на креативно-контекстную форму соответствующего этапа (рис. 5).

Рис. 5 иллюстрирует коррекцию алгоритма на 3-м этапе. Запускаем измененный скомпилированный код. Результаты помещаем на ККЭФ. Хорошо видно, что, поскольку результаты полностью верны, можно считать процесс проектирования завершенным.

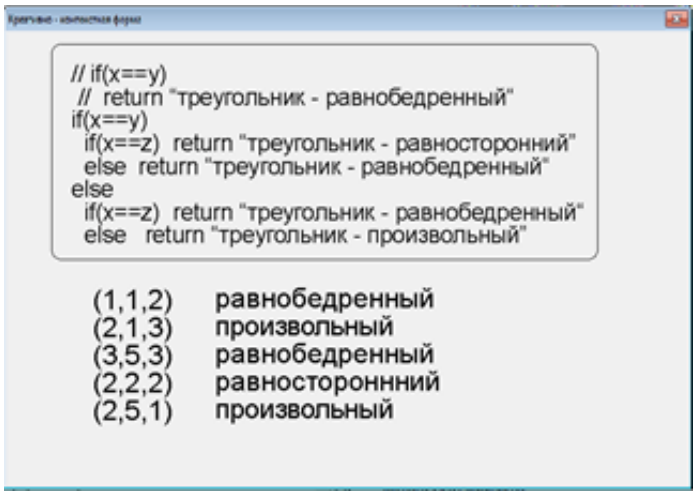


Рис. 5. Креативно-контекстная экранная форма конечной точки проектирования

Итак, подводя итог применения помещаемых на ККЭФ дополненных знаний, следует отметить, что предлагаемая технология проектирования посредством фиксации ДЗ влияет не непосредственно на структуру и качество алгоритма, поскольку вся необходимая информация в основном имеется в специализированных средствах отладки программ. ДЗ определяют возможность сохранения тех фактов, того знания об объекте разработки, которые являются свидетельством и фундаментом творческой выработки локальных проектных решений. Подобная фактология принятия решений всегда присутствует в процессе проектирования ПС, однако, оставаясь незафиксированной, теряется. При обращении к структуре или коду алгоритма эти данные часто весьма трудно восстановить, а значит, для коррекции алгоритма будет необходимо вновь повторить некоторые шаги разработки,

сличая результаты выполнения программы с ожидаемыми.

Организация сохранения и воспроизведения ДЗ позволит соединить и зафиксировать в едином контексте любые графически выразимые информационные элементы проекта. Располагаемые на ККЭФ такие данные содержат в своей совокупности тот строй мыслей, который соответствует найденному решению в момент творческого импульса, свидетельствующего об уверенности в выбранном решении.

Проблемы, решению которых способствует использование в процессе проектирования ПС креативно-контекстных форм, хранящих ДЗ, следующие:

- фиксация полной картины выработки локальных проектных решений;

- восстановление любой из картин в произвольный момент времени, что обеспечивает быстрое воспроизведение знаний о локальном состоянии проекта в индивидуально значимой форме;

- сохранение и воспроизведение локально-точечных картин коррекции и изменения элементов проекта.

Если ДЗ не использовать, то главное – теряются данные творчески выработанных решений, и, значит, коррекция алгоритма в некоторой точке будет требовать больших интеллектуально- творческих усилий, большего времени для нового «погружения» в содержательную суть алгоритма. Более того, такое «погружение» может оказаться неадекватным и привести к падению, а не к повышению качества алгоритма. Важно также отметить, что если не применять ДЗ с ККЭФ, то вовсе нельзя будет найти решения в сложных случаях, когда такое решение сопряжено с одновременным осознанием значительного множества информационных элементов, превышающих возможности кратковременной памяти. Например, при разработке алгоритма построения эквидистанты плоского контура используется оценка корреляции одновременно более 2 десятков информационных элементов, которые как раз и можно совместить единственным образом, на общей ККЭФ.

Список литературы:.

1. Parnas D. and Clements P. 1986. A Rational Design Process: How and Why to Fake It. - IEEE Transactions on Software Engineering, vol. SE-12(2).

2. Страуструп Б. Язык программирования C++. 3-е изд., СПб.;М.; “Невский диалект” - “БИНОМ”, 1999.

3. Акофф Р., Эмери Ф.. О целеустремленных системах. – М.: ЛКИ, 2008. – 272 с.

4. Богоявленская Д.Б. Психология творческих способностей. М.: ИЦ «Академия», 2002. – 320 с.

5. Miller, G. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. The Psychological Review vol.63(2), p.86., March.1956. (Миллер Дж. Магическое число семь, плюс или минус два. – В кн.: Инженерная психология. М. 1964)

6. Ларичев О. И. Объективные модели и субъективные решения. М.: Наука, 1987. – 144 с.

7. Друкер Питер Ф. Эффективный руководитель. Киев: Вильямс, 2007 – 224 с.

8. Андерсон, Дж. Р. Когнитивная психология / Р. Джон Андерсон – СПб.: Питер, 2006. – 589 с.

9. Azuma R. A Survey of Augmented Reality Presence: Teleoperators and Virtual Environments, pp. 355–385, August 1997.

10. Зинченко Т.П. Когнитивная и прикладная психология.- М., Воронеж, 2000.– 608 с.

11. Paivio A. Imagery and verbal processes. New York: Holt, Rinehart, and Winston. – 1971.