Научная визуализация, 2018, том 10, номер 2, страницы 138 - 150, DOI: 10.26583/sv.10.2.11

Визуальное обеспечение творческой разработки программных систем

А.И. Разумовский

ИПУ РАН, Москва
ORCID: 0000-0002-1449-2873, razumowsky@yandex.ru

Аннотация

Описаны основания и принципы визуальной поддержки индивидуальной творческой активности при разработке программных систем. Показана важность визуального обеспечения процесса разработки, посредством которого организуется восполнение недостаточной информации, и ставится барьер утечке индивидуально значимых данных. Сделана попытка подчинить процесс создания программных систем визуальной избыточности информационного содержания посредством объединения управления как над алгоритмическими и структурными элементами, так и над любым графически возможным неформальным представлением данных, что позволит обеспечить целостность разработки программных систем, а также повысит удобство выработки проектных решений. Определены базовые условия доминанты творческого человеческого фактора: личный опыт, комфорт и эвристика доступности, которая играет связующую роль между элементами разработки при творческом контроле: она действует как качественная оценка частоты или вероятности выбора за счет легкости воспоминания или ассоциации. Приведена упрощенная схема действий разработчика с использованием избыточного множества проектных элементов. Для позиционирования в едином визуальном контексте данных предложена креативно-контекстная форма, которая служит позиционированию и фиксации всевозможных информационных элементов, что провоцирует творческий инсайт очередного решения. Приведен пример реализации такой программной системы, а также пример алгоритма, разработанного ее посредством. Описана функционально- информационная модель программного комплекса «Эврика», реализующего поддержку и обслуживание деятельности творческого процесса разработчика алгоритмов и программных систем.

Ключевые слова: программная система, творческий процесс, креативно- контекстная форма, избыточность описания, индивидуальность описания, индивидуальность задачи, инсайт решения, эвристика доступности.

1. Введение

Предмет настоящего исследования часто относят к области визуализации программного обеспечения, а также к информационной или когнитивной визуализации. Это справедливо лишь в плане самого объекта исследования, то есть визуализации необходимых при разработке программного обеспечения данных, но не со стороны целей, средств и значимости визуализации для человека в процессах разработки им программных систем и алгоритмов. И, хотя как самостоятельная научная дисциплина визуализация программного обеспечения и информационная визуализация оформилась еще в 80-90-ых годах XX века, на сегодняшний день остается белым пятном изучение процессов взаимозависимости и взаимозаменяемости элементов системы человеко- компьютерного взаимодействия с точки зрения качества и особенностей процесса выработки человеком решения о развитии или коррекции алгоритма. Среди наиболее известных работ по настоящему предмету обращают на себя внимание работы J. Raskin, M. Lanza, C. Ware, M. Petre, S. Ducasse, N. Fenton и S.L. Pfleeger, J. J. Thomas и K. A. Cook, K. Zhang, J. T. Stasko, B. Shneiderman, E. R. Tufte [1-14], а также результаты исследо-

ваний отечественных ученых по информационной и когнитивной визуализации [15-18]. Если когнитивную и информационную визуализацию часто соотносят с образовательным процессом, сообразуя «перенесение в процессе познавательной деятельности из внутреннего плана во внешний мыслеобразов, форма которых стихийно определяется механизмом ассоциативной проекции» [17], то целями визуализации программного обеспечения являются поддержка понимания структуры программных систем и алгоритмов, а также их анализ и исследование ошибок в плане коррекции внутреннего содержания [1-3]. В работе [9] утверждается, что визуализация предоставляет возможность понимать огромные объемы данных, при том что частые проблемы с пониманием возникают на уровне интерпретации слишком упрощенной информации [1]. В работе [3] для лучшего восприятия информации предлагается набор метрик, сообщающих разработчикам программных решений сопутствующую коду информацию, выраженную численно: структура иерархий, размер и связность классов и методов, использование атрибутов. Предложенный в работе [4] инструмент визуализации CodeCrawler использует двумерные пространства для визуализации объектно-ориентированного программного обеспечения. Узлы представляют собой программные объекты или абстракции из них, а ребра - отношения между этими объектами, при одновременном отображении до пяти метрических измерений на одном узле. В работе [7] информационная визуализация была сосредоточена на создании подходов к передаче абстрактной информации интуитивно понятными способами. За счет изменения плотности отображения полезной информации в работах [12, 13] утверждается возможность сведения к минимуму ошибок визуализации.

Так или иначе, исследования по визуализации программного обеспечения и когнитивной визуализации направлены на формирование и манипуляции с числовой и абстрактной информацией, что не позволяет решить проблему полноты усвоения человеком используемых данных, а главное - эффективного приложения присущих конкретному человеку опыта, знаний и ответственности. Настоящее исследование делает попытку восполнить этот пробел, определяя своей целью исключить множественные информационные потери на разных этапах и уровнях разработки программных решений. Ставится задача обеспечить фактическое проникновение информации внутрь сознания индивида, накопление разнородных данных с применением средств визуализации, сохранение индивидуально значимых данных, как соответствующих творческому инсайту решения, с последующим, по необходимости, воспроизведением их вновь для продолжения разработки.

Проблемы, связанные с полнотой усвоения человеком используемой информации при разработке программных систем (ПС), определяются недостаточной поддержкой индивидуальных творческих способностей человека, и в срезе имеющихся технологий создания ПС могут быть выражены следующим образом:

- неполнота информации в процессе проектирования и алгоритмизации;
- унификация и деперсонификация данных проекта;
- утечка индивидуально значимой информации.

Эти проблемы имеют две основные причины:

- рационализация процесса разработки ПС;
- унификация средств разработки ПС.

Стремление к рациональному подходу при разработке и использовании ПС оправдано и естественно, однако существует и иная точка зрения: "мы никогда не отыщем процесс, который дал бы нам возможность проектировать программы строго рациональным образом"[19]. Однако далее отмечено: «если мы держим в голове идеальный процесс, становится легче измерять успехи проекта", иначе говоря, творческие возможности проек-

тировщика, считают авторы, должны быть подчинены определенным правилам процесса разработки ПС, что позволит обеспечить его управляемость. Таким образом, главным вопросом разработки ПС остается проблема гармоничного взаимодействия творческого и рационального в поиске, выработке и принятии решений. Следует также знать, что сужение свободы творчества разработчиков приводит к слабому профессиональному росту, безынициативности, небрежности [20]. Более того, в работе [21] выражается следующее мнение: "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 [22]. Последнее означает, что при разработке 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 [23]. В книге акад. Ларичева [24] 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, адекватно конструктивно описывающих творческий человеческий фактор (ТЧФ), не существует. Конструктивно это означает, как если бы математической формулой можно было бы заменить определенную интеллектуальную функцию ТЧФ, или точнее – его творческий импульс и стимул. Нет необходимости создавать, искать, отлаживать математические выражения, описывающие нечто не являющееся ТЧФ или его результатом. Все подобные модели в ту или иную меру используют логико-вероятностный подход, основанный на субъективной экспертной оценке, и, следовательно, далеко не вполне соответствуют действительности. В противоположность формальному подходу, предлагается обустраивать среду ТЧФ индивидуально – тогда субъективность представления не только не будет помехой, но наоборот послужит качественной опорой деятельности ТЧФ в разработке ПС.

2. Построение информационной модели среды визуализации

Общий порядок отдельного действия ТЧФ при разработке таков: ТЧФ субъективно оценивает локальную задачу и творчески вырабатывает решение. Имеется два класса информации, обуславливающих выработку решений, - действительная информация и воображаемая информация или перцептивное воображение [25]. 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 инсайтов прокладывает дальнейший путь развития проекта ПС. С творческой точки зрения воображаемая информация имеет перед действительной 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 частично зафикси-

ровать посредством ее графической визуализации с целью последующего многократного использования. Создание визуальных картин решений будет актуализировать комплексность и согласованность разнородных данных, выраженных визуально графически, и позволит не только находить новые решения, но и корректировать имеющиеся. А фиксация и воспроизведение в любой момент времени таких картин обеспечит восстановление всей цепочки разработки, с возможностью повторного и пристального рассмотрения уже выполненных шагов. Важно обратить внимание, что сама возможность возникновения решения, инсайт, существует лишь в случае, если ключ к нему уже содержится в неосознаваемом опыте [26]. Поэтому визуальное воплощение перцептивной информации является своего рода связующим элементом между творческой активностью и личным опытом решения конкретной задачи. Фиксируется и воспроизводится именно лично значимая информация, а не те данные, которые могут быть обобщены и затем выражены абстрактно. Такая лично значимая информация сообразуется с подготовкой и фиксацией данных инсайта и выражает собой элементы нового опыта.

Таким образом, накапливаемый опыт является существенным определяющим элементом выработки решения, и, следовательно, предпосылкой усиления доминанты ТЧФ станет условие накопления индивидуального опыта в общем процессе решения задачи. Кроме того, следует обратить внимание также на создание условий вызревания решения, что соответствует инкубации в описании этапов творчества, сделанном английским педагогом Г. Уоллесом в 1926 году [27]. Указанный факт может быть принят во внимание посредством создания подобия метода «информационной доски», предложенного в 1962 А. Ньюэллом и описанного, например, в книге Г.Буча [28].

Метод «информационная доска» включает в себя три элемента: собственно информационную доску, набор источников знаний и контроллер - управляющий этими источниками, причем активный элемент, контроллер, выполняет следующую роль: каждый источник знаний связан с контроллером и посылает ему свои соображения, кроме того контроллер может активизировать источники знаний. Однако роль контроллера мы можем расширить, позволив ему совершать любые произвольные действия с источниками знаний: добавлять, удалять, перемещать, актуализировать, группировать, сохранять и воспроизводить. В таком воплощении контроллер превращается в управляющий активный элемент, действующий вне произвольных заранее определенных правил и ограничен лишь объемами информационной доски и множества источников знаний. Программная реализация такого контроллера очевидно сопряжена не только с трудностями создания обширной базы описаний правил и условий автоматизированного синтеза решений, но и со сложностями пользовательского характера, когда удобство взаимодействия с компьютером пользователя станет обратно пропорционально размеру базы знаний. Кроме того, программная реализация потребует тщательной проработки формальной составляющей системы, включая классификацию и структуризацию информационных элементов и правил. Поэтому резонным разрешением этой проблемы предлагается замена программного контроллера на ТЧФ или индивидуальный творческий процесс. А саму информационную доску следует теперь именовать иначе - креативно- контекстной формой (ККФ), подчеркнув фундаментальную роль творческого процесса в выработке решений, (рис. 1).

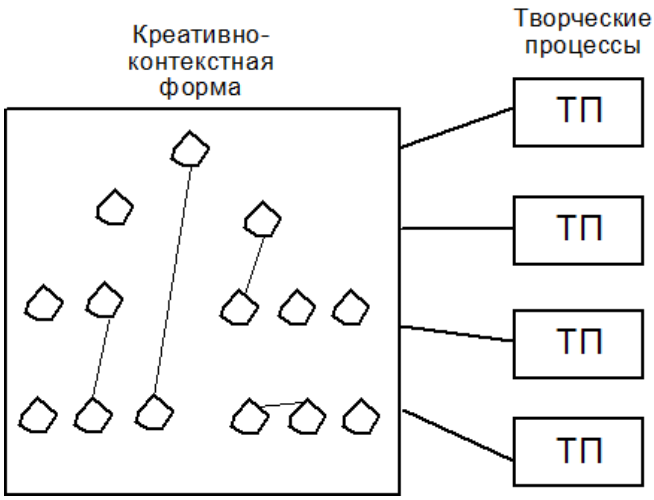

Рис 1. Информационная модель соответствия ККФ и ТЧФ.

Контекстная сторона ККФ выражается в том, что выработка и принятие решения совершается человеком на основе личного опыта, удобства и информационной доступности. Поэтому контекстное взаиморасположение элементов на ККФ основано на так называемой эвристике доступности, которая определяется как возможность или оценка выбора за счет легкости воспоминания или ассоциации [29] и является определяющей компонентой в поиске и принятии человеком решения, а также в приобретения им знаний.

Таким образом, визуально- творческая поддержка разработки ПС лежит в области свободы проявления отдельной творческой активности индивида, при этом, регистрации всей возможной полноты информации, участвующей в процессе разработки. После регистрации, необходимо эти последовательно собранные данные зафиксировать и затем с течением всего процесса разработки дополнять и видоизменять их взаимосвязи по мере расширения накапливаемой информации.

3. Моделирование программной визуализации

При разработке ПС существует несколько планов или уровней взаимодействия человека и вычислительной мощности, основными среди которых являются проектный, алгоритмический и пользовательский. Однако информация, связанная с непосредственным творчеством при выработке, апробации и реализации идей, в значительной степени теряется без возможности непосредственного восстановления. Эту проблему невозможно решить без привлечения средств визуализации способных сыграть связующую роль между элементами разработки ПС и творческой активностью индивида. ККФ берет на себя роль аккумулятора данных и знаний, выражаемых графически и соотносимых с опытом, удобством и эвристикой доступности (рис. 2).

Процесс разработки ПС проходит итерационно, последовательно наращивая мощность и качество программного продукта. Информационное содержание разработки технически опирается на средства визуализации через ККФ, а практически служит основой для выработки инсайтов решений. Вся необходимая для этого информация позиционируются на ККФ определенным, индивидуально значимым образом, чтобы иметь возможность зафиксировать ее в базе данных с последующим по запросу воспроизведением. Таким образом, предполагается значительно сократить утечку значимых для выработки решений данных.

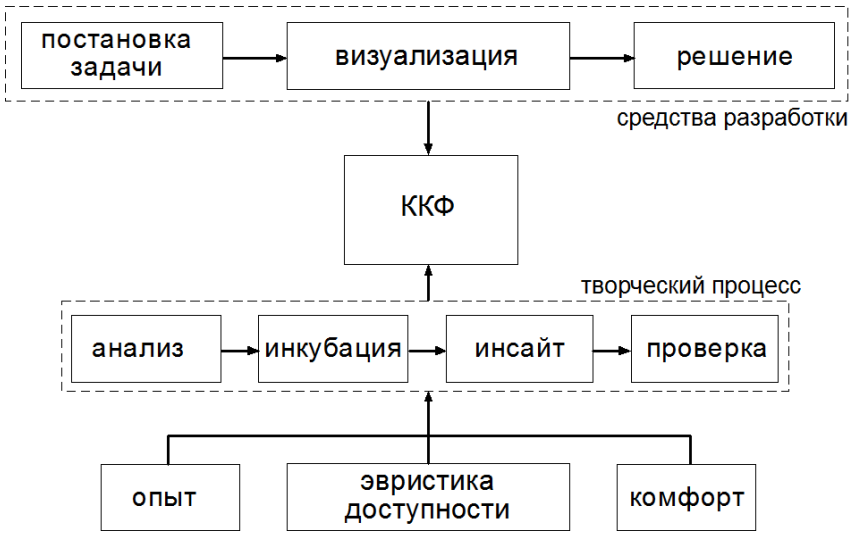

Рис. 2. Информационная схема согласования средств разработки ПС с ТЧФ.

На рисунке 3 приведена упрощенная алгоритмическая схема операций при творческом подходе к разработке ПС.

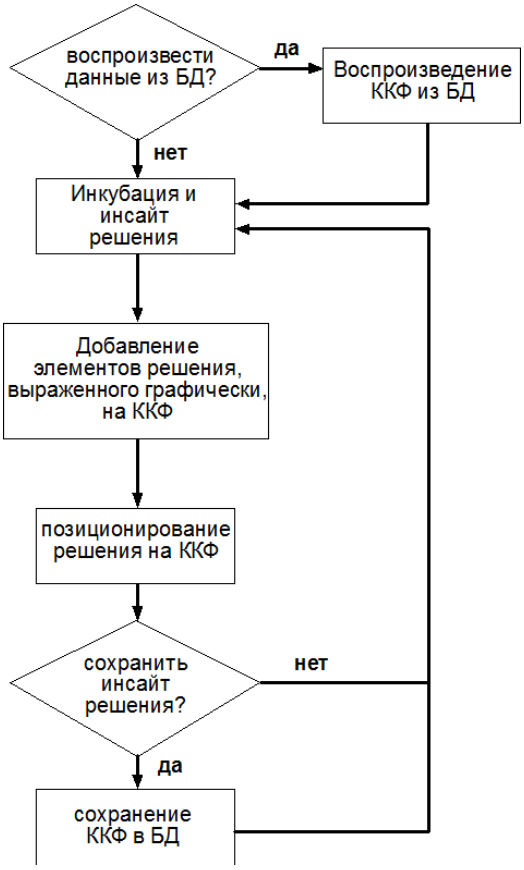

Рис. 3. Схема алгоритма действий творческого «контроллера»

Каждый элемент итерации выработки очередного решения начинается с его инкубации на основании представленных на ККФ информационных элементов. После того, как новое решение выработано и принято, оно оформляется в алгоритме на проектном или программном уровне и затем заносится и позиционируется на ККФ. В качестве данных, заполняющих ККФ, выступают наиболее значимые для разработчика элементы или их фрагменты. Это могут быть детали программного кода, комментарии, отладочные фрагменты, сопроводительные схемы и рисунки результата выполнения программы. Их ценность

состоит не только в сохранении в базе данных вообще, но и в соотнесении друг с другом определенным, важным для разработчика образом. Правильная совмещенность в избыточном контексте обеспечивает комфорт использования разработчиком эвристики доступности и, соответственно, инкубацию очередного решения.

Любое решение может быть немедленно сохранено в базе данных ККФ или, при не полной заполненности ККФ, такое сохранение может быть отложено. Позиционирование решений на ККФ и их сохранение в базе данных позволит не только осуществить лучшую инкубацию последующего решения через эвристику доступности, но и воспроизвести воспоминание решения с сопутствующим информационным антуражем. Это особенно важно при необходимости повторных возвратов к предыдущим итерациям разработки и выполняет отчасти роль расширенной кратковременной памяти, что непосредственно способствует лучшему обзору и использованию данных ККФ для последующей инкубации и очередного инсайта.

4. Программный комплекс «Эврика»

С целью осуществлять качественное и гибкое взаимодействие человека и компьютера в процессе проектирования и программирования алгоритмов и ПС разработан специальный программный комплекс, реализующий поддержку и обслуживание деятельности творческого процесса разработчика. Была поставлена задача: найти и обеспечить такой режим взаимодействия человека и компьютера, при котором творческий процесс разработчика в выработке и принятии решений получил бы перед компьютером доминирующий приоритет.

В своей основе комплекс, получивший название «Эврика», содержит блок визуализации, оперирующий ККФ.

Задача творческого поиска алгоритма определяется следующими одновременно достигаемыми условиями:

- сохранение избыточности описания представления проектируемого решения;
- сохранение индивидуальной выработки алгоритмической идеи, основывающейся на личной значимости информационного элемента в конкретных условиях его применения;
- сохранение индивидуальности решаемой проблемы, без необходимости обобщения и преждевременной классификации;
- поддержка исходного неформального представления информационных элементов;
- обеспечение видимости локальных результатов разработки, отладки и функционирования проекта или алгоритма, а также возможности их соотнесения с прочими информационными элементами проекта или алгоритма.

Каждое из указанных условий имеет непосредственное отношение к реализации выражения индивидуальных предпочтений и указывает на отсутствие каких-либо преград в выработке индивидуально комфортного решения, что означает беспрепятственную деятельность ТЧФ. Основной целью на пути выработки решения провозглашается достижение такого состояния взаимно соотнесенных и расположенных на ККФ имеющихся данных, при котором возникнет инсайт, соответствующий факту рождения искомой идеи [30].

Предполагается три основных направления обработки ТЧФ данных, находящихся в контексте ККФ (рис. 4):

- формирование картины решения, включая сюда любые действия с данными, направленное на совершенствование и лучшее понимание локальной проблемы и обеспечивающее инкубацию конечной идеи;
- сохранение картины, соответствующей моменту озарения;
- воспроизведение сохраненных картин инсайта.

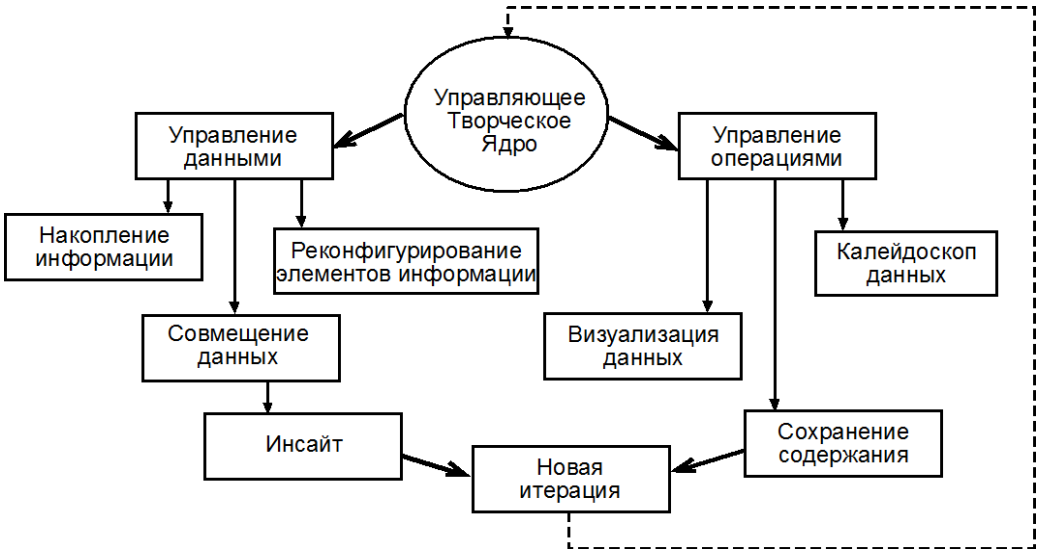

Рис. 4. Функционально- информационная модель программного комплекса «Эврика»

Основной движущей силой комплекса «Эврика» является информационно-визуальная поддержка достижения очередного инсайта, который инкубируется посредством наблюдения над соотнесенными между собой данными ККФ. В настоящей модели на первом плане оказывается естественное итерирование разработки.

В качестве примера использования программного комплекса «Эврика» рассмотрим эвристическую выработку алгоритма построения эквидистанты плоского контура (ЭПК).

Особенность данного алгоритма заключается в необходимости хранения и использования всей возможной полноты информации о структуре исходного контура и деталях последовательного изменения алгоритма, относя сюда также такие данные, которые являются результатами локальных итераций проекта, а также разнообразные промежуточные расчеты. Необходимая целостность знаний об алгоритме может быть обеспечена только за счет создания такой информационно-функциональной среды, которая бы определяющим образом имела возможность накапливать, представлять и сохранять весь спектр информации.

Содержательная суть алгоритма расчета ЭПК заключается в последовательном нахождении значимых биссектрис углов между содержащими отрезки контура прямыми, а затем пересечении полученной сетки биссектрис эквидистантными к каждому из отрезков контура линиями. Полученные при последнем пересечении точки будут последовательно образовывать целевые эквидистантные контуры (рис. 5).

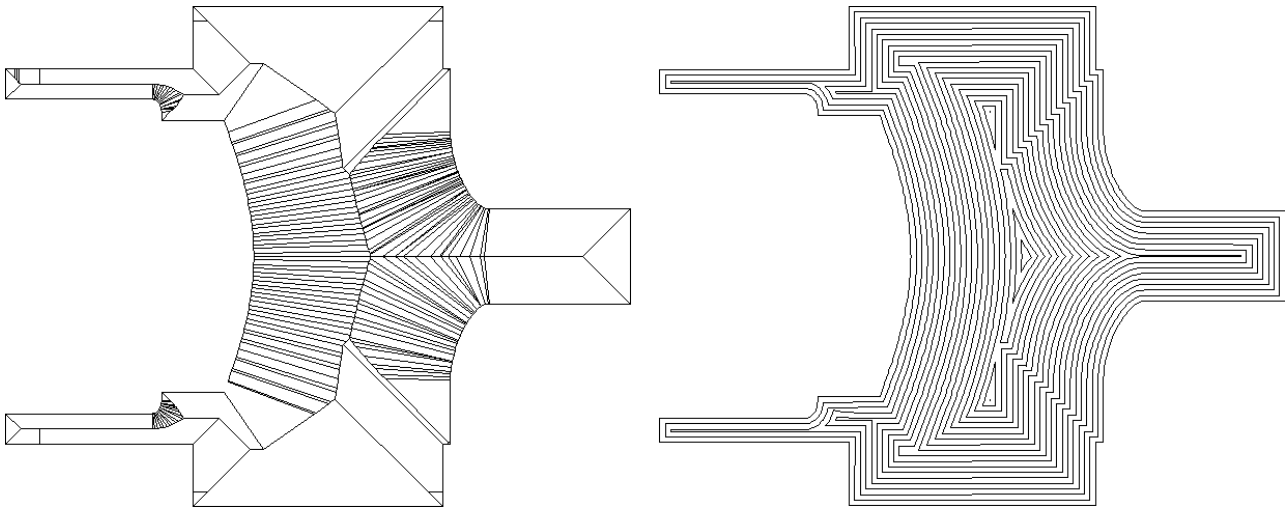

Рис. 5. Результаты алгоритма построения ЭПК.

Проведем качественные оценки множеств информационных элементов, потребных для выработки очередного локального решения. Для хранения данных о каждом ребре исходного контура используется структура, состоящая из 9 полей, причем каждое из перечисленных полей применяется не только непосредственно для очередных расчетов, но и для контроля промежуточных результатов в текстовом и графическом виде.

Сравним два случая сложности: в начале работы алгоритма и в середине. Ситуация начала работы алгоритма характеризуется в значительной мере лишь исходными данными, каждый из массивов биссектрис содержит всего одну точку, исходную, поэтому наблюдение над данными представляет незначительные трудности. Второй случай отличается от первого насыщенными массивами биссектрис, а также более сложным пониманием и интерпретацией текущих результатов, без которой нельзя выработать и принять решение о возможностях дальнейшего развития алгоритма.

Во втором случае количество необходимых для наблюдения информационных элементов возрастает примерно от 1.5 до 7 раз для контуров с числом точек не более 500. Оценим, какова приблизительная величина числа таких элементов для формирования условий инкубации очередного решения. Для первого случая число необходимых информационных элементов составляет примерно от 10 до 20. Для второго случая, таких элементов требуется, соответственно, от 15 до 140. Увеличение в 1.5 – 7 раз требуемых для наблюдения элементов происходит из-за того, что число точек биссектрис, координаты которых должны быть учтены, значительно возрастает от нуля в начале работы алгоритма. В простейшем случае имеем минимум 7 элементов - вначале работы алгоритма, и минимум 11 – в середине его работы, так как добавляются также еще две координаты и индекс отрезка пересечения. Но в среднем, важных для наблюдения элементов увеличивается сразу на 4-5 точек, по паре координат, с каждой стороны сегмента контура, что означает резкое возрастание элементов наблюдения и выработки решений, примерно в 4.5 раза.

Естественно, такие оценки делаются исходя из индивидуальных представлений о важности информационного элемента в контексте прочих. Осуществляя наблюдение, поиск и принятие решения в пространстве визуализации ККФ на базе эвристики доступности, разработчик выступает как эксперт, и сделанный им выбор опирается на личную эвристическую достоверность. В этом и состоит ценность творческого подхода, в противоположность компьютерному синтезу решений, где существует жесткая система правил выбора и мер. Напротив, в условиях свободы творчества, выработку решений и их оценку производит сам разработчик на основе

собственного опыта, знаний и способностей, применяемых вне технологических или аксиоматических препятствий.

Необходимость совмещения в едином контексте визуализации наиболее важную с точки зрения разработчика информацию о некоторой проблемной ситуации, решение которой необходимо осуществить, иллюстрирует рисунок 6. На этом рисунке изображена одна из полученных в системе «Эврика» ККФ при проектировании алгоритма ЭПК в ситуации возникновения неожиданной ошибки на 25 шаге итерации функционирования алгоритма.

ККФ включает 8 полей слайдов для выбранных информационных элементов, каждый из которых содержит важную информацию для творческой выработки решений по исправлению возникшей ошибки функционирования алгоритма. Данные 4-х слайдов отображают ситуацию с графической точки зрения в разном масштабе. Еще 3 слайда содержат текстовую отладочную и лог- информацию, относящуюся к текущим данным, связанную с пересечением биссектрис. Последнее поле слайда отображает сочтенные разработчиком важными отладочные данные, вырезанные, как часть экрана, в момент возникшей ошибки, из среды MS Visual Studio.

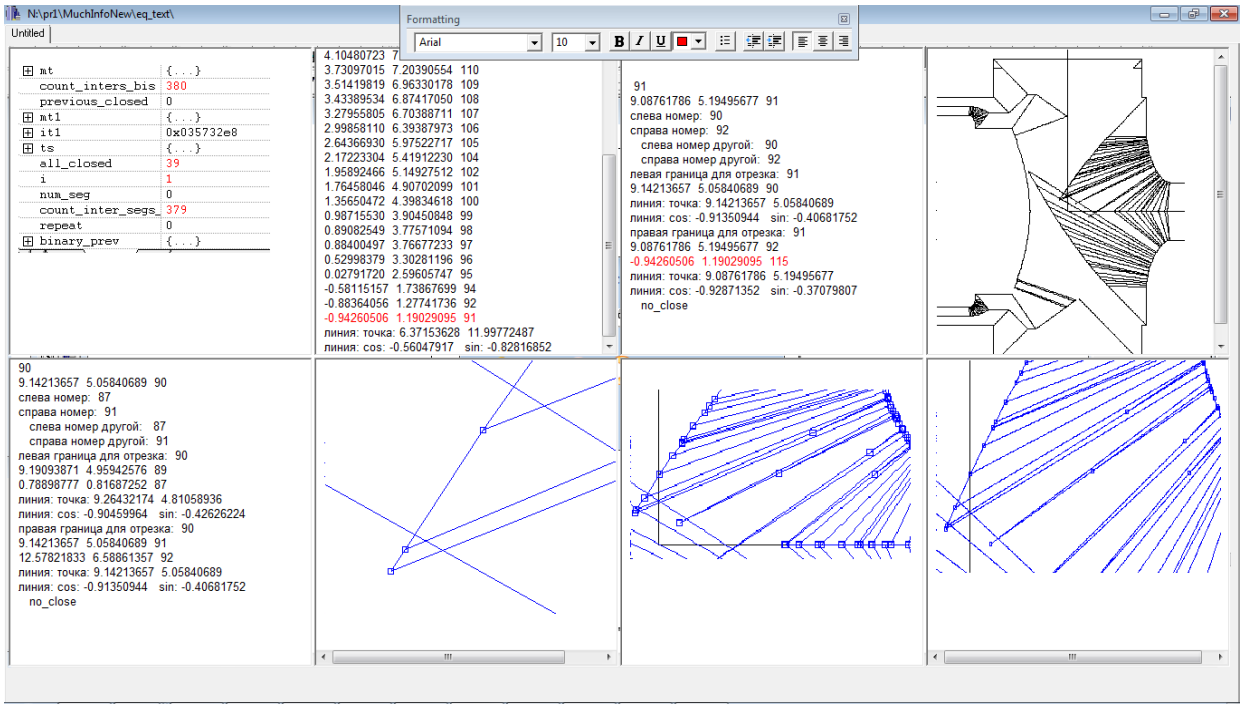

Рис. 6. ККФ 25-го шага итерации функционирования алгоритма ЭПК.

Порядок анализа пользователями комплекса «Эврика» визуально представленной на ККФ информации предполагает наличие полноты картины. Таким образом, возникающий творческий инсайт, исходя из своей природы сопровождающийся эмоциональным всплеском, сигнализирует о достижении такой полноты, то есть картина решения становится ясной. В момент наступления ясности, ККФ фиксируется, данные помещаются в хранилище, чтобы при необходимости быстро вернуть разработчика творчески к произошедшим прежде ситуациям. Отсутствие информационных потерь обеспечит эффективный путь коррекции или расширения проекта. Полученный при помощи комплекса «Эврика» массив данных ККФ придаст по-настоящему бесценный импульс разработке ПС на уровне проявления индивидуально творческих качеств человека, вооруженного уверенностью, что выбранное им алгоритмическое или проектное решение не приведет в фатальный тупик, а может быть, напротив, с относительной легкостью исправлено. Такое исправление воз-

можно за счет последовательной визуализации соответствующих ККФ, хранящих всю важную информацию о шагах и содержании пути разработки.

5. Заключение

Главные цели, достигаемые посредством визуального обеспечения творческой разработки ПС, следующие:

- непосредственное наблюдение при помощи данных на ККФ расширения или изменения алгоритма;
- восстановление в деталях информационного содержания любого прежде зафиксированного этапа разработки;
- одновременное наблюдение над несколькими вариантами алгоритма, и, как следствие, обеспечение принятия качественно лучших вариантов решений;
- наблюдение и поиск решений в индивидуально значимом, конкретном контексте избыточных, совместно позиционируемых данных.

Рассмотренный в статье порядок организации накопления и визуализации информационных элементов позволит объединить в едином контексте и зафиксировать любые графически выразимые информационные элементы разработки ПС. Располагаемая на ККФ индивидуально значимым образом информация в своей совокупности содержит тот опыт и строй мыслей, который соответствует найденному решению в момент творческой инкубации и инсайта, свидетельствующего о верности решения.

Вообразим на мгновенье, что при разработке ПС визуализация и позиционирование на ККФ лично значимой, в первую очередь – перцептивной, информации не применяются. Тогда, результатом станут многочисленные утечки оттенков и атрибутов творческого инсайта, потери локальных данных проектирования и программирования, например, отладочных и сопроводительных данных, издержки отсутствия многомерности представления информации, и, наконец, потери от несогласования разноплановых данных между собой, что обеспечивается позиционированием, разнообразием и избыточностью элементов на ККФ. Иначе говоря, потребуются значительно больше интеллектуально-творческих усилий в выработке решений, большего времени для нового «погружения» в содержательную суть проекта и деталей процесса разработки. Это, в свою очередь, приведет к нахождению и принятию менее надежных в личном плане решений, а в некоторых сложных случаях, когда для их инкубации важно одновременно учесть значительно превышающих «магическое число» [23] элементов, найти правильный ответ станет и вовсе затруднено.

Список литературы

1. M. Petre, Why Looking Isn't Always Seeing: Readership Skills and Graphical Programming, Comm. ACM, vol. 38, no. 6, pp. 33-44, June 1995.
2. J. Bohnet et al.: Projecting Code Changes onto Execution Traces to Support Localization of Recently Introduced Bugs. 24th ACM Symposium on Applied Computing, ACM, pp. 438–442, 2009.
3. M. Lanza. Program Visualization Support for Highly Iterative Development Environments // Proceedings of VISSOFT 2003, Pp. 62 — 67.
4. S. Demeyer, S. Ducasse, and M. Lanza, A Hybrid Reverse Engineering Platform Combining Metrics and Program Visualization, Proc. Sixth Working Conf. Reverse Eng. (WCRE '99), Oct. 1999.
5. Zhang K. Software visualization: From Theory to Practice. Springer Science+Business Media, 2003.
6. Benjamin B. Bederson and Ben Shneiderman. The Craft of Information Visualization: Readings and Reflections, Morgan Kaufmann. 2003.
7. James J. Thomas and Kristin A. Cook (Ed.) Illuminating the Path: The R&D Agenda for Visual Analytics Archived - 2008-09-29 at the

Wayback Machine. National Visualization and Analytics Center. 2005. p.30.
8. N. Fenton and S.L. Pfleeger, Software Metrics: A Rigorous and Practical Approach, second ed. London: Int'l Thomson Computer Press, 1996.
9. C. Ware. Information Visualization. Morgan Kaufmann, 2000.
10. S. K. Card, J. D. Mackinlay, and B. Shneiderman, editors. Readings in Information Visualization - Using Vision to Think. Morgan Kaufmann, 1999.
11. J. T. Stasko, J. Domingue, M. H. Brown, and B. A. Price, editors. Software Visualization - Programming as a Multimedia Experience. The MIT Press, 1998.
12. E. R. Tufte. The Visual Display of Quantitative Information - Graphics Press, 2nd edition, 2001.
13. E. R. Tufte. Visual Explanations. Graphics Press, 1997.
14. Раскин Д. - Интерфейс: новые направления в проектировании компьютерных систем. Символ-Плюс, 2014. С. 200.
15. Мухина К.Д., Боченина К.О., Карсаков А.С., Бухановский А.В. Технологии когнитивной визуализации темпоральных комплексных сетей // Известия высших учебных заведений. Приборостроение -2017. - Т. 60. - № 3. - С. 195-203.
16. Мышев А.А. Компьютерная топография графических образов в когнитивных технологиях научной визуализации. Научная визуализация. 2015. Т. 7. № 5. С. 68-86.
17. Манько Н. Н. Когнитивная визуализация педагогических объектов в современных технологиях обучения. Образование и наука. Известия УрО РАО. 2009. № 8. С. 10-30.
18. Ломов П.А., Данилов Е.Ю. Визуализация с помощью когнитивных фреймов для передачи знаний. Информационные системы и технологии. 2015. № 3 (89). С. 10-18.
19. 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).
20. Страуструп Б, Язык программирования С++, 3-е изд., Спб.;М.; "Невский диалект" - Издательство "БИНОМ", 1999
21. Акофф Р., Эмери Ф.. О целеустремленных системах. – М.: ЛКИ, 2008. – 272 с.
22. Богоявленская Д.Б. Психология творческих способностей. М.: ИЦ «Академия», 2002. – 320 с.
23. 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)
24. Ларичев О. И. Объективные модели и субъективные решения. М.: Наука, 1987. – 144 с.
25. Андерсон, Дж. Р. Когнитивная психология / Р. Джон Андерсон - СПб.: Питер, 2006. - 589 с.
26. Пономарев Я.А. Психология творчества. М.: Наука, 1976.
27. Wallas G. The Art of Thought. (New York: Harcourt, Brace, and Company, 1926)
28. Буч. Г., Объектно-ориентированный анализ и проектирование с примерами приложений на С++, М.: "Бином", СПб: "Невский диалект",.1999
29. Канеман Д., Словик П., Тверски А. Принятие решений в неопределенности. Харьков: Гуманитарный фонд, 2005, 632 с.
30. Разумовский А.И. Проблема ответственного наблюдения при проектировании программных систем. - Труды международной конференции CAD/CAM/PDM-2012, М.: «Аналитик», 2012. с. 69-73.