10. **Клеванский Н.Н.** Моделирование стратегии формирования расписания занятий ВУЗ'а средствами реляционной алгебры / **Н.Н. Клеванский, Е. А. Макарцова, С.А. Костин** // Прикладные проблемы образовательной деятельности: Межвуз. сб. научн. тр. – Воронеж: Центр. – Черноземн. книжн. изд-во, 2003. – Вып. 10. – С.71 – 74.

11. **Burke E**. Interactive Timetabling: Concepts, Techniques, and Practical Results in E. Burke, P. / T. Muller, R. Bartak // the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT2002), Gent, 2002, pp. 58-72.

12. **Калашников А. В.** Алгоритмы локальной оптимизации расписаний / **А. В. Калашников, В. А. Костенко** // Методы и средства обработки информации: Первая всероссийская научная конференция, Москва, 1 – 3 октября 2003 г. – М. МАКС Пресс, 2003.– С. 3 – 10.

13. **Моркун В.С.** Розробка системи управління ресурсами вишу при складанні розкладу занять / **В.С. Моркун, П.В. Бурнасов** // Гірничий вісник : науково-технічний збірник. - Вип. 99.-Кривий Ріг: ДВНЗ "КНУ"., 2015.-с.159-164.

14. **Моркун В.С.** Методи визначення якості розкладу занять ВНЗ/ **В.С. Моркун, П.В. Бурнасов** // Вісник східноукраїнського національного університету імені Володимира Даля №1 (225), 2016. с.129-138.

15. **Варшавский П.Р.** Методы правдоподобных рассуждений на основе аналогий и прецедентов для интеллектуальных систем поддержки принятия решений / **П.Р. Варшавский, А.П. Еремеев** // Новости искусственного интеллекта. ─ 2006. ─ № 3. ─ С. 39 - 62.

16. **Карпов Л.Е.** Методы добычи данных при построении локальной метрики в системах вывода по прецедентам / **Л.Е. Карпов, В.Н. Юдин** // ИСП РАН, препринт. – 2006. – №18.

Рукопис подано до редакції 28.03.17

УДК 681.03

И.Н. ВДОВИЧЕНКО*, канд. техн. наук, доц., Криворожский национальный университет

МОДЕЛИРОВАНИЕ ПРОГРАММНЫХ СИСТЕМ И ПРОЦЕССА ИХ РАЗРАБОТКИ

Цель. Целью работы является посроение моделей программных продуктов и процесса разработки программ. Рассматривается проблема моделирования программных систем. Предложены значимые характеристики программных систем, которые необходимо отобразить в модели. Выделены зависимости критериев и этапов разработки. Анализируется такая единица измерения временных показателей программирования, как человеко-месяц. Ясно, что стандартные методы математического программирования, дифференциального исчисления и теории множеств ограничены в использовании при построении моделей программных систем . Необходим подход на основании комбинированного метода.

Методы. Для решения поставленных задач используются методы аналитического и статистического имитационного моделирования процесса разработки программных систем. Применены элементы агрегирования и комбинирования.

Научная новизна. Предложены варианты элементов моделей программных систем. Рассмотренны модели этапов разработки программного обеспечения.

Практическая значимость. Предложенные модели можно использовать для общей оценки качества программных систем, рассчетов прогноза трудозатрат разработок, сложности программ, стоимости и времени программирования и др.

Результаты. Построены варианты элементов моделей программных систем. Отмечены показатели, оказывающие влияние на производительность программистов. Систематизированы количественные оценки процесса программирования. Выделены взаимосвязи показателей.

Ключевые слова*:* моделирование, программные системы, оценивание, производительность, программные ошибки, время разработки, человеко-месяц.

Проблема и ее связь с научными и практическими задачами. Структуры современных технических систем отличаются большим разнообразием и сложностью. В связи с этим перед разработчиками систем возникает ряд серьезных проблем, связанных с проведением качественного и количественного анализа эффективности функционирования систем.

Анализ технологического опыта лидеров производства программных продуктов показывает, насколько дорого обходится несовершенство прогноза трудозатрат, сложности программ, негибкость контроля и управления, приводящие к последующей трудоемкой его переделке. Эти обстоятельства, требуют тщательного отбора методик, моделей, методов оценки качества программных систем.

Постановка задачи. Повышение требований к функционированию влечет за собой совершенствование средств и методов изучения и разработки программных систем. Фредерик Брукс отмечает: "Слабо развиты наши методы оценок". В методах оценок достигнутый результат подменяется затраченными усилиями. Кроме этого, типовые методы и средства, используемые

* © Вдовиченко И.Н., 2017

в других инженерных дисциплинах, при разработке программного обеспечения почти не применяются. Это могло бы обеспечить ускорение создания и повышение качества программ.

Для анализа, при обнаружении дефектов в системах, целесообразнее использовать модели систем. Поэтому совершенствование методов и средств моделирования программного обеспечения является актуальной задачей.

Сложности при моделировании программных систем обусловлены: разнородностью элементов, изменчивостью структуры, разнообразием режимов работы, сложностью программного обеспечения и др. Для моделирования сложных программных систем непригодны или ограничены в применении методы математического программирования, дифференциального исчисления и теории множеств.

Изложение материала и результатов. В моделях необходимо учитывать: производительность программистов, частоту программных ошибок, методы оценок. Кроме этого желательно учитывать: ясность цели (в виде коэффициента), человеческий ресурс, время, интенсивность обмена информацией, адекватность технологий, сложность программной системы.

Аналитический вариант модели программных систем

При моделировании процесса создания программных систем, особенно их временных показателей, оценивания и планирования, используют такую единицу измерения как человеко-месяц. Мы считаем, что эта единица измерения не всегда соответствует действительности. Моделировать стоимость программного продукта, как произведение числа занятых на число месяцев работы над программой можно в том случае, когда работники последовательно не согласовывают результаты. Если модель взаимодействия при разработке программной системы имеет вид представленный на рис.1, то затраты возрастают как $n(n-2)/2$ [1].

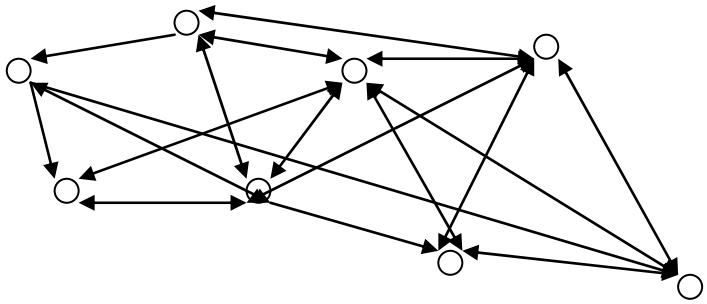

Рис.1 Модель согласования информации между программистами

При построении модели процесса разработки программного обеспечения используют следующее эмпирическое правило [1]:

на планирование – 1/3 времени разработки;

на написание программ – 1/6 времени разработки;

на тестирование компонентов и предварительное системное тестирование – 1/4 времени разработки;

на системное тестирование при наличии всех компонентов – 1/4 времени разработки.

При построении модели временных затрат на разработку необходимо учитывать: затраты времени на планирование, написание программы, время на обмен информацией, время на составление документации, тестирование, время на отладку, системную интеграцию, обучение.

При построении модели временных затрат на разработку программного продукта, которая бы максимально точно описывала процесс разработки, необходимо добавить время на сбои программного и аппаратного обеспечения, срочные мелкие задания, составление бюрократических бумаг, болезни, личное время. Все это может занимать до 50 % времени на программирование и отладку.

Время решения задач ограничено уравнением продуктивности

$$\text{Время решения задачи} = \sum(\text{частота})_i \cdot (\text{длительность})_i.$$

При построении модели стоимостных показателей учитывают, что стоимость сопровождения составляет 40% стоимости разработки [1] (·).

Выделим некоторые моменты, которые надо учитывать при построении модели программного продукта:

нельзя строить модель разработки по части относящейся к написанию программы, ведь эта часть только 1/6 всего процесса;

нельзя модели и оценки малых программ использовать для сложных и больших программных систем;

надо учитывать, что при сопровождении, исправление ошибок влечет появление новых с вероятностью 0,35.

Практика показала, что объем работ растет как степенная функция размера программного продукта [1]

$$\text{Объем работ} = (\text{константа}) \cdot (\text{количество операторов в программе})^{1,5}.$$

Учитывая указанные соотношения можно предложить следующие модели.

Стоимость программного продукта=(число занятых умноженное на число месяцев работы)$\cdot$ $n(n\text{-}2)/2$;

Пусть:

стоимость программного продукта – S;

число занятых – n;

число месяцев работы – m;

стоимость сопровождения – S_{sp}.

С учетом (·) получаем, полная стоимость равна

$$S = n \cdot m \cdot n(n\text{-}2)/2 + S_{sp} = n \cdot m \cdot n(n\text{-}2)/2 + 0{,}4n \cdot m \cdot n(n\text{-}2)/2 = 0{,}7n^2m(n\text{-}2).$$

Модель процесса разработки программного продукта можно представить в следующем виде

Время разработки = Количество дней планирования+
+Количество дней на написание программ+
+Количество дней на тестирование компонентов и предварительное системное тестирование +
+Количество дней на системное тестирование при наличии всех компонентов.

Пусть временя разработки – T . С учетом эмпирического правила можно записать:

количество дней на планирование – p = 1/3 времени разработки;

количество дней на написание программ – r = 1/6 времени разработки;

количество дней на тестирование компонентов и предварительное системное тестирование – x=1/4 времени разработки;

количество дней на системное тестирование при наличии всех компонентов – y=1/4 времени разработки.

Тогда $T=p+r+x+y$; а с учетом эмпирического правила можно записать так:

$$T=1/3T+1/6T+1/4T+1/4T.$$

Но в модели надо так же учитывать:

время на обмен информацией – t_o;

время на составление документации – t_s;

время на отладку – t_{ot};

время на обучение персонала – $t_{ob.}$

Время на сбои программного и аппаратного обеспечения, время на срочные мелкие задания, время на составление бюрократических бумаг, болезни, личное время и т.п. – $t_{tp}=1/2(r+t_{ot})$.

Тогда можно записать

$$T=p+r+x+y+t_o+t_s+t_{ot}+t_{ob}+t_{tp} = p+r+x+y+t_o+t_s+t_{ot}+t_{ob}+1/2(r+t_{ot}) =$$
$$= p+1{,}5r+x+y+t_o+t_s+1{,}5t_{ot}+t_{ob}.$$

Для обеспечения моделирования без ощибок можно использовать процедуру разработки, которую годами используют лучшие прграммисты – модель нисходящего проектирования.

В этом случае модель делится на модель архитектуры, модель разработки, модель реализации.

При нисходящем проектировании можно уменьшить количество ошибок т.к.:

прозрачность структуры облегчает формулировку требований к модулям и функциям;

помогает избежать системных ошибок;

скрытие деталей помогает увидеть ошибки в структуре;

тестирование можно выполнять после каждого шага подмодели.

При моделировании сложности программного продукта надо учитывать, что программы принципиально отличаются от других объектов созданных человеком, которые имеют повторяющиеся части в большом количестве. В программе эти части объединяются в подпрограммы. Поэтому сложность программных систем очень зависит от их размеров. У программных систем число возможных состояний на несколько порядков выше, чем у самых сложных устройств. А эти состояния надо понять, описать, тестировать. При увеличении размеров про-

граммы, ее элементы взаимодействуют нелинейно, следовательно, сложность растет быстро. Сложность программ это важное их свойство.

Сложность - причина многих ошибок в программе;
сложность - причина повышения стоимости разработки;
сложность - причина не выполнения графика разработки;
сложность - причина трудности вызова функций;
сложность - причина трудности развития программ;
сложность - причина нарушения систем защиты;
сложность - причина технических проблем;
сложность - причина нагрузки при обучении персонала.

Поэтому создавая упрощенную модель программы нельзя игнорировать сложность, так как на ней замыкаются многие другие характеристики. Иначе модель потеряет важнейшие свои свойства – адекватность и информативность.

Необходимой харатеристикой, которую надо рассмотреть в модели, является время реакции системы. Оно не должно быть меньше порога скорости человеческого восприятия, что равно 100 миллисекунд.

При моделировании надо учитывать, что для создания системного программного продукта надо в 9 раз больше напряжения от разработчиков, чем для прикладного.

Метод статистического моделирования программных систем. Это один из вариантов построения модели программной системы. Структура программной системы является основой для построения ее математической модели. Основываясь на структуре системы, строим ее имитационную модель. Статистическим имитационным моделированием называется следующая последовательность действий:

многократный прогон модели;
накопление результатов прогонов;
статистическая обработка накопленных результатов;
расчет средних значений необходимых характеристик;
расчет математических ожиданий искомых характеристик.

Статистическое моделирование может быть организовано по принципу приращения времени, по принципу особых состояний, по принципу логико-вероятностного моделирования. В нашей работе использован принцип приращения времени, т.к. два следующих принципа, хоть и уменьшают машинное время обработки, но требуют построения сложных подмоделей, что в итоге сводит на нет выигрыш во времени. Данный принцип основан на следующем алгоритме: ввод исходных данных; формирование исходных условий; формирование случайных событий; приращение времени; прогон системы; анализ результатов; статистическая обработка; проверка достижения нужного числа прогонов и результатов; анализ окончания моделирования; документирование.

При изучении программных систем, определении их сложности, надежности и других характеристик, оптимально применение комбинированного моделирования. Для определения и исследования системных показателей применяем статистическое моделирование, для определения и изучения верификационных характеристик, цикломатического числа - строим графическую модель в виде управляющего графа. При построении комбинированной математической модели программных систем возможно так же, использование метрики сложности по Холстеду:
Словарь программы: $n=n1+n2$;

длина программы: $N=n1\ \log_1(n1)+n2\ \log_2(n2)$;

объем программы: $V=N\ \log_2(n)$;

прогноз числа ошибок: $B=N\log_2(n)\ /\ 3000$;

индекс сопровождаемости кода: $MI=MAX(0,\ (171–5{,}2\cdot\ln(V)–0{,}23CC–16{,}2\cdot\ln(LoC))\cdot 100/171)$;

время программирования программы:

$$T=(n1N2\ (n1\log_2(n1)+n2\log_2(n2))\log_2(n1)/2\ n2\ S,$$

где S – число Страуда; CC – Цикломатическая сложность; LoC – количество строк кода; $n1$– число операторов; $n2$ – число операндов. Объединив аналитический и статический вариант модели можно получить уточнения в моделировании.

Выводы и направленность дальнейших исследований. Полученные элементы модели облегчат разработчикам программных систем и аналитикам решение проблем, связанных с

проведением качественного и количественного анализа эффективности функционирования систем, прогноза трудозатрат, времени разработки и стоимости программмирования. При построении моделей объединены несколько методов моделирования и опыт ведущих отечественных и зарубежных программистов. Модели требуют доработки с учетом тех закономерностей, которые были отмечены в статье, но не учтены в моделях, а так же иных положений не учтенных в ней. В перспективе возможны построения элементов моделей сложности программных систем, моделей расчета количества возможных ошибок (отличной от модели Холстеда) и т.п.

Список литературы

1. **Фредерик Брукс**. Как создаются программные системы. – Санкт-Петербург, Симбо, 2001. – 298с.
2. **Вдовиченко И.Н**. Построение математической модели программных систем. Збірник матеріалів Міжнародної науково-технічної конференції «Сталий розвиток промисловості та суспільства». Кривой Рог, КНУ, 2016.
3. **Соммервилл Иан**. Инженерия программного обеспечения. – М.: Изд. дом Вильямс, 2002. – 624 с.
4. **Роберт Т. Фатрелл**. Управление программными проектами. – М.: Издательский дом "Вильямс", 2003. – 1125 с.
5. **Лешек Мацяшек**. Анализ требований и проектирование систем. – М.: Издательский дом "Вильямс", 2003. – 651с.
6. **Орлов С.А**. Технологии разработки программного обеспечения: Разработка сложных программных систем Изд. 3-е, 2004.
7. **Липаев В.В.** Программная инженерия. Методологические основы : Учеб. / В. В. Липаев ; Гос. ун-т — Высшая школа экономики. — М. : ТЕИС, 2006. — 608 с.
8. **Бабенко Л.П**., **Лаврищева К.М.** Основи програмної інженерії. Навчальний посібник. – К.: Знання, 2001. – 415 с.
9. **Лингер Р**., **Миллс Х., Уитт Б.** Теория и практика структурного программирования. – М.: Мир, 1982.
10. **Салливан Э**. Время – деньги. – М.:Microsoft Press, Русская редакция, 2002.
11. **Вендров А.М.** Проектирование программного обеспечения экономических информационных систем.–М. : Финансы и статистика, 2007.
12. Материалы сайта http://www.uml.ru
13. Материалы сайта http://www.omg.org/technology/documents/formal/uml.htm
14. Материалы сайта http://www.citforum.ru
15. Материалы сайта http://sorlik.blogspot.com

Рукопись поступила в редакцию 02.03.17

УДК 622.14

П.И. ФЕДОРЕНКО, д-р техн. наук, проф., А.В. ПЕРЕМЕТЧИК, канд. техн. наук, доц.,
Т.А. ПОДОЙНИЦЫНА, старший преподаватель, Криворожский национальный университет

ПРОГНОЗИРОВАНИЕ И МНОГОФАКТОРНАЯ ГЕОМЕТРИЗАЦИЯ КАЧЕСТВЕННЫХ ПОКАЗАТЕЛЕЙ ЖЕЛЕЗОРУДНЫХ МЕСТОРОЖДЕНИЙ НА ОСНОВЕ ЭВРИСТИЧЕСКИХ МЕТОДОВ

Цель. Целью настоящей работы является совершенствование методики геометризациии качественных показателей железорудных месторождений для построения такой горно-геометрической модели месторождения, которая давала бы возможность описать закономерности размещения важнейших качественных показателей в пространстве с тем, чтобы спрогнозировать их изменение в процессе развития горных работ. Особенно важным аспектом применения геометризации месторождений железорудных полезных ископаемых является горно-геометрическое прогнозирование их качественных показателей для решения заданий перспективного и текущего планирования с тем, чтобы наладить с максимальной эффективностью работу горнодобывающего предприятия в режиме усреднения качества руды и повысить рационализацию освоения месторождения.

Методы исследования. Задача работы определила применение комплексного метода исследований, включающего проведение теоретических исследований, лабораторные и промышленные эксперименты. При проведении отдельных исследований были использованы геостатистические методы и методы программирования для ЭВМ.

Научная новизна. Описан многомерный эвристический алгоритм прогнозирования, эффективно реализующий уравнения математической модели многомерного случайного геохимического поля, путем использования предложенного полинома произвольной степени. Показано, что в качестве математического описания элементов прогнозируемого горного массива целесообразно принимать систему уравнений многомерного случайного геохимического поля. Установлено, что в качестве метода обработки маркшейдерско-геологической информации, полученной по

© Федоренко П.И. Переметник А.В., Подойницына Т.А., 2017