УДК 044.03; 681.518:061

МОДЕЛЬ ПЕРЕХОДА ОТ МАКРО- К МИКРОПРОЕКТИРОВАНИЮ СПЕЦИАЛИЗИРОВАННЫХ ИС

В.М. Левыкин, Н.В. Черненко.
Харьковский национальный университет радиоэлектроники.

В статье представлена модель перехода от этапа макропроектирования к этапу микропроектирования организационных информационных систем. Данная модель является имитационной. В статье предложены описание такой модели в виде граф-схемы алгоритма, модели оценки стоимости и сроков выполнения работ.

Ключевые слова: *проектирование, организационная информационная система, имитационная модель, граф-схема алгоритма*

Введение

При практической разработке средних и крупных организационных информационных систем (ИС) Разработчик и Заказчик на всех стадиях проектирования сталкиваются с различными проблемами. Среди них наиболее значимыми являются следующие: определение вида разрабатываемой системы, сроки и объем средств, необходимых на разработку ИС. На этапе макропроектирования процесса разработки системы эти проблемы особо актуальны, так как Разработчик не знает в совершенстве предметной области, а Заказчик не имеет информации об особенностях процессов проектирования. В то же время разработка ИС в целом во многом зависит от качества проведенных работ на этапе макропроектирования.

1. Постановка задачи

Нахождение компромисса между желаниями Заказчика (требования к разрабатываемой системе, сравнительно сжатые сроки разработки и минимизация затрат на разработку и внедрение ИС) и возможностями Разработчика является сложной задачей. При разработке ИС основная сложность заключается в том, что процесс автоматизации направлен не только на автоматизацию основного бизнес-процесса, но и на бизнес-процессы, обеспечивающие организацию деятельности таких подразделений, как отделы, службы и т.д. Исходя из этого, существует сложность обоснования необходимости разработки и внедрения ИС, в виду проблемы оценки экономического эффекта от внедрения ИС [1].

Существующие методологии (SSADM, методологии, описанные в отечественных и зарубежных стандартах – ГОСТ 34.602, ISO 12.207) не предоставляют эффективного инструментария взаимодействия Заказчика и Разработчика на этапе макропроектирования процесса разработки ИС [2].

Разработка ИС может проходить по нескольким альтернативным вариантам: по аналогу и с «чистого листа». Оба подхода имеют свои преимущества и недостатки. Так, при реализации проекта ИС по аналогу бизнес-процессы, реализуемые на объекте управления (ОУ), должны максимально совпадать с логикой работы, заложенной в готовых модулях системы-аналога, а у Заказчика должна быть возможность выделения крупных объемов финансовых ресурсов на стадиях разработки, подготовки ОУ к внедрению ИС и т. д. К преимуществам такого варианта относится короткий срок разработки (компоновки и наладки) и внедрения поставляемой системы. Реализация проекта ИС «с чистого листа» является более сложной задачей. До момента сдачи системы в эксплуатацию невозможно точно определить сроки и стоимость реализации проекта ИС, и поэтому предполагается более активное вовлечение конечных пользователей в процесс разработки. При этом, если такая система будет корректно реализована, она будет наиболее адекватна требованиям Заказчика, ему не придется платить за «лишнюю» функциональность в проектах реализации ИС (которая может не будет востребована) [1].

Для снижения ошибочных проектных решений в процессе разработки необходимо, чтобы и Заказчик, и Разработчик имели полное представление о проекте ИС, ее структуре в целом, а не об отдельных ее фрагментах.

2. Изложение основного материала исследования

В качестве модели жизненного цикла (ЖЦ) проекта ИС предлагается выбрать его эволюционный прототип. Эволюционный прототип – функциональная система, которая на начальных этапах разработки может быть описана приближенной моделью, а в процессе разработки эволюционировать в «совершенную» систему, удовлетворяющую требования конечных пользователей. Выполнение работ при реализации эволюционного прототипа подразумевает на самых ранних стадиях вовлечение пользователей в процесс разработки (позволяет снизить риски проектирования: разработки качественных решений, срыва сроков работ, невыполнения бюджета). Для реализации процесса разработки проекта ИС, согласно выбранному ЖЦ, может быть использована методология DSDM (разработана Консорциумом DSDM – Великобритания). Однако данный стандарт также не формализует механизм взаимодействия Заказчика и Разработчика при проектировании ИС. Для повышения эффективности процесса разработки ИС предлагается использовать на этапе макропроектирования имитационную модель, которая позволит сформировать различные варианты разрабатываемой ИС при изменении накладываемых ограничений, а также выбрать рациональный вариант реализации ИС, если такие есть, а в случае отсутствия вариантов реализации – указать тип ограничения (функциональность, ресурсы, время), из-за которого невозможно реализовать проект ИС [3].

В случае реализации проекта в виде эволюционного прототипа конечные пользователи постепенно вовлекаются в процесс разработки. При этом, если они увидят положительный эффект в своей работе от внедрения прототипа на ОУ, то будут способствовать его развитию, и наоборот: если пользователям предоставить окончательный вариант системы и заставить изучить принципы работы с ней в краткие сроки, то возникает вероятность их самоустранения от процесса проектирования.

В связи с описанными проблемами и возможными путями их решения, модель 1-го уровня прототипа ИС представим следующим выражением:

$$P_1=<Ц, Тр, F, D, R, T>, \quad (1)$$

где

Ц – множество целей, реализуемых системой;

Тр – множество требований к разрабатываемой системе;

F – множество функциональных задач;

D – множество выходных документов, реализуемых системой;

R – ресурсы, выделенные на разработку и внедрение ИС;

T – время разработки системы.

При этом Заказчиком накладываются следующие ограничения:

- $F \to \Delta\varepsilon$, $\Delta\varepsilon$ – предельно допустимое расширение функциональности (определяет порог перехода ИС на качественно другой уровень конфигурации);

- на сроки выполняемых работ: $T^* \le T_{крит}$, где $T_{крит}$ – время, отведенное на разработку заказчиком;

- на стоимость проводимых работ: $R^* \le R_{крит}$, где $R_{крит}$ – ресурсы, выделяемые заказчиком на разработку ИС.

С учетом этих требований скорректированная модель 2-го уровня описания прототипа принимает следующий вид:

$$P_2=<Ц^*, Тр^*, F^*, D^*, R^*, T^*>, \quad (2)$$

где

Ц* – множество целей, реализуемых скорректированным вариантом конфигурации ИС;

Тр* – множество функциональных и нефункциональных требований, выдвигаемых к конкретной конфигурации ИС;

F* – множество функциональных задач, реализуемых конкретным окончательным вариантом конфигурации ИС;

D* – множество выходных документов, реализуемых конкретным окончательным вариантом конфигурации ИС;

R* – ресурсы, выделенные на разработку и внедрение определенного варианта конфигурации ИС;

T* – время разработки определенного варианта конфигурации ИС.

Переход от модели 1-го уровня описания прототипа системы к модели 2-го уровня с использованием классических математических методов и моделей невозможен из-за проблем описания множества факторов, влияющих на данный процесс. Поэтому наиболее удобным средством решения этой проблемы является разработка соответствующей имитационной модели [4]. На рис. 1 представлены этапы реализации такой имитационной модели разработки эволюционного прототипа ИС.

На рис. 2 описан алгоритм реализации имитационной модели.

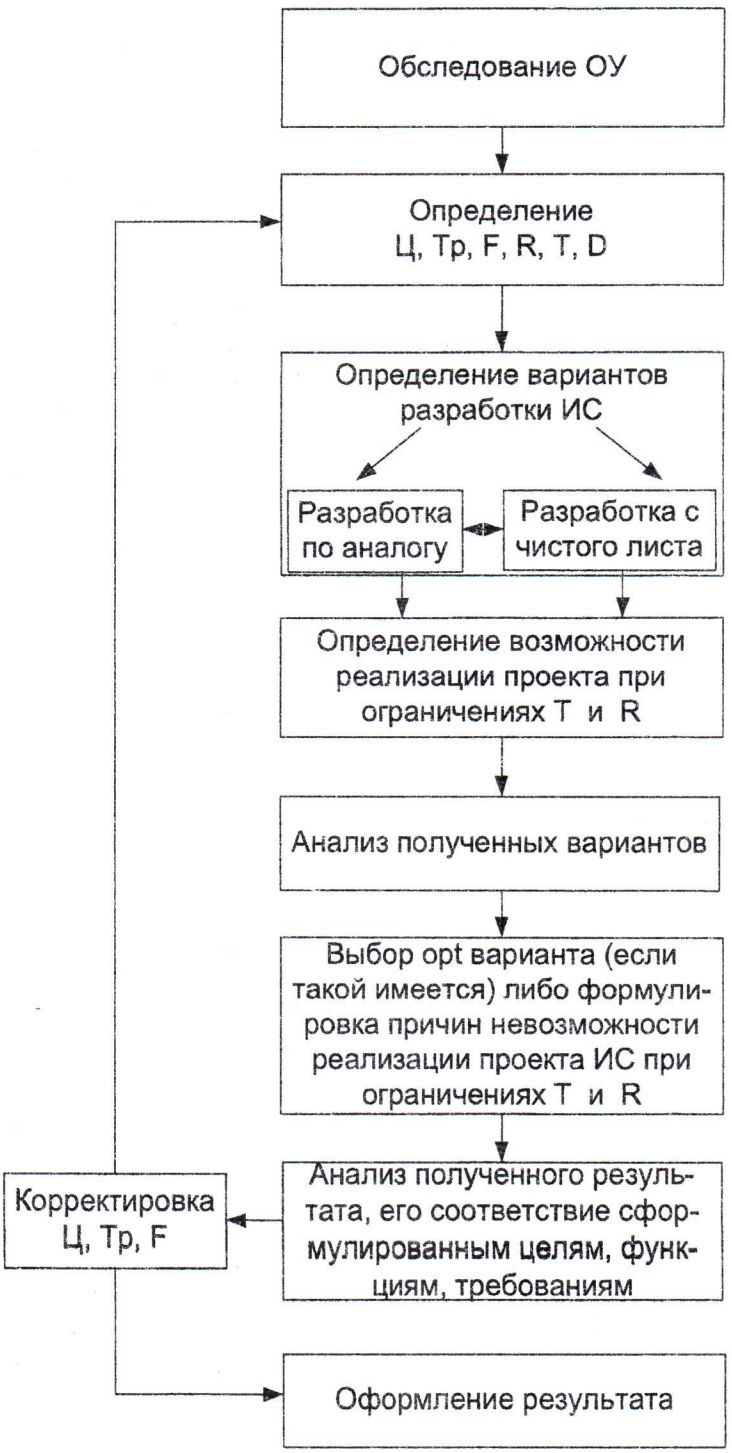

Рис. 1. Этапы имитационной модели разработки прототипа ИС

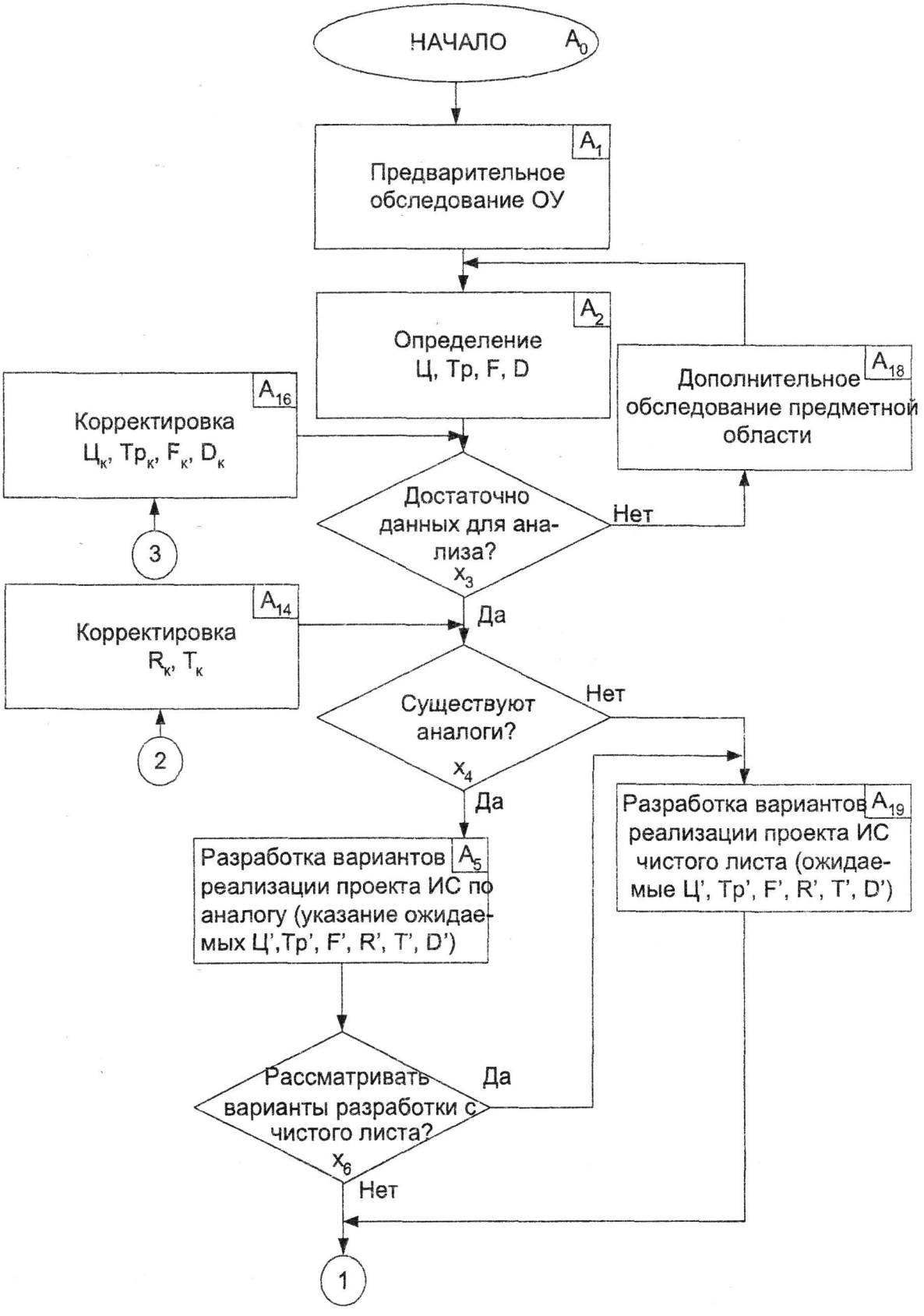

Рис. 2. Граф-схема алгоритма реализации имитационной модели разработки прототипа ИС

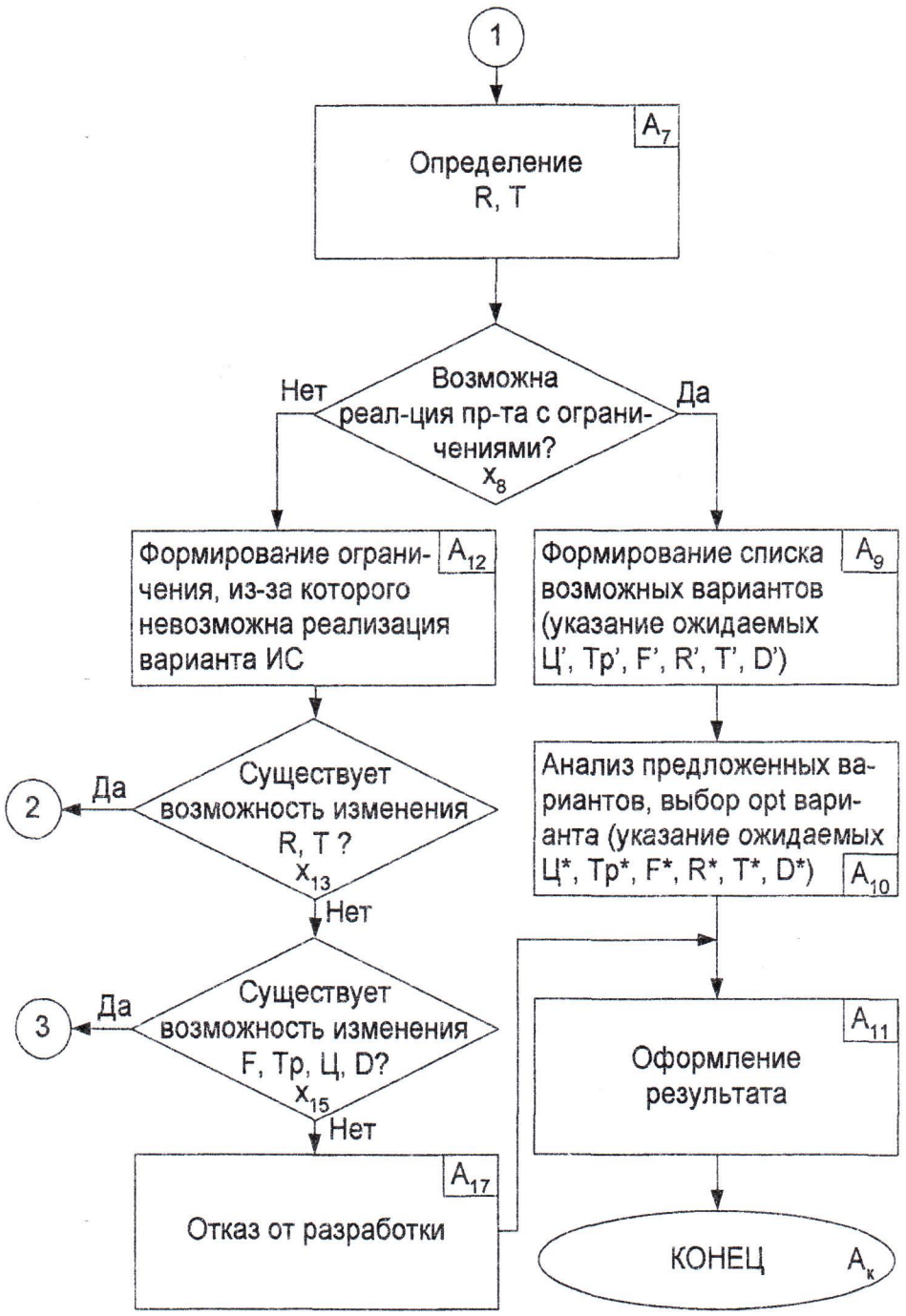

Рис. 2. (лист 2 – продолжение)

Перейдем к структуре моделирующего алгоритма. Необходимы следующие операторы:

A_0 – оператор, описывающий событие «принятие решения о необходимости разработки проекта ИС»;

A_1 – оператор предварительного обследования объекта управления (определение, согласование и выполнение работ заказчиком и разработчиком);

A_2 – оператор определения целей, требований, функций и управляющих воздействий (в виде документов) разрабатываемого проекта ИС экспертами (представители заказчика и разработчика);

x_3 – оператор определения достаточности данных для проведения моделирования;

x_4 – оператор определения наличия аналогов;

A_5 – оператор разработки вариантов реализации проекта ИС по аналогу с указанием ожидаемых Ц', Тр', F', R', T', D';

x_6 – оператор определения необходимости рассмотрения разработки дополнительных вариантов проектов ИС;

A_7 – оператор определения бюджета проекта (R – объем ресурсов, которые выделяются заказчиком на разработку и внедрение ИС) и сроков выполнения проекта (T) экспертами;

x_8 – оператор определения возможности реализации варианта проекта ИС с заданными ограничениями;

A_9 – оператор формирования списка возможных вариантов реализации проекта ИС с указанием Ц', Тр', F', R', T', D' по каждому отдельному варианту;

A_{10} – оператор анализа сформулированных вариантов проекта ИС, выбора оптимального варианта реализации проекта ИС по группе критериев с установленными степенями важности, формирования каталога требований для выбранного варианта;

A_{11} – оператор оформления результата, оформление всей документации, инициирующей разработку в случае утверждения полученного варианта либо разрыв договора;

A_{12} – оператор определения и указания вида ограничения, из-за которого невозможна реализация проекта ИС (в случае невозможности реализации проекта);

x_{13} – оператор определения возможности изменения бюджета и сроков разработки;

A_{14} – оператор корректировки R и T;

x_{15} – оператор определения возможности изменения F, Тр, Ц, D;

A_{16} – оператор корректировки F, Тр, Ц, D;

A_{17} – оператор отказа от разработки с указанием причин, привлекших к отказу;

A_{18} – оператор выполнения работ экспертами в случае необходимости дополнительного обследования предметной области;

A_{19} – оператор разработки вариантов реализации проекта ИС «с ноля» с указанием ожидаемых Ц', Тр', F', R', T', D';

$A_к$ – оператор перехода на уровень микропроектирования (либо отказа от разработки).

Модель реализации перехода от 1-го уровня описания прототипа ко 2-му для представленной граф-схемы алгоритма имеет вид:

$$M_п = (A_0\ A_1\ ^{1,18}A_2\ ^{2,16}x_{3\downarrow 18}\ ^{3,14}x_{4\downarrow 19}\ A_5\ x_{6\downarrow 19}\ A_7\ x_{8\downarrow 12}\ A_9\ A_{10}\ ^{10,17}A_{11}\ A_{12}\ x_{13\downarrow 15}\ A_{14}\ x_{15\downarrow 17}\ A_{16}\ A_{17}\ ^{\uparrow 3}A_{18}\ ^{\uparrow 4,6}A_{19}\ ^{11}A_к). \quad (2)$$

Одним из основных элементов модели (1) являются требования к ИС.

Требования делятся на функциональные и нефункциональные.

$$Tp = FT \cup NFT, \text{ где} \quad (3)$$

Tp – множество требований, выдвигаемых к системе;

FT – множество функциональных требований, выдвигаемых к системе, $FT \in Tp$

NFT – множество нефункциональных требований, выдвигаемых к системе, $NFT \in Tp$;

В свою очередь, при оценке нефункциональных требований можно выделить следующие группы требований: нефункциональные требования, сформированные в виде прямых затрат – $\overline{ft}^1 \in NFT$ (множество нефункциональных требований, которые непосредственно влияют на сроки и стоимость разработки) и нефункциональные требования, сформированные в виде опосредованных (косвенных) затрат $\overline{ft}^2 \in NFT$ (множество нефункциональных требований, которые опосредованно влияют на сроки и стоимость разработки). Исходя из сказанного, множество нефункциональных требований к системе определяется как объединение множеств видов нефункциональных требований: $\overline{ft}^1 \cup \overline{ft}^2 \in NFT$.

В качестве примера $\overline{ft}^1$ можно привести следующие:

- количество наработок на отказ;
- класс точности датчиков, необходимых для снятия информации и т. д.

В качестве примера $\overline{ft}^2$ можно привести следующие:

- реализация информационной системы с применением конкретного подхода (web, архитектура «тонкий» или «толстый» клиент);
- реализация решений по видам обеспечений ИС с применением конкретных технологий и инструментария.

В случае, если разработчик имеет достаточный опыт разработки определенного класса систем для определенной предметной области и располагает готовыми типовыми решениями, тогда стоимость и сроки реализации функциональных требований можно определить по стоимостной матрице [5]. Предлагается рассмотреть несколько вариантов стоимостных матриц:
- детерминированная стоимостная матрица;
- вероятностная стоимостная матрица.

Детерминированная стоимостная матрица более проста в использовании и в реализации, однако получаемые решения являются недостаточно гибкими и могут содержать дополнительную функциональность, в которой не нуждается заказчик, однако, вынужден будет за нее платить. Элементами детерминированной стоимостной матрицы являются вероятность вхождения определенной функции в определенную конфигурацию (вероятность может принимать значения 0, если функция не входит в конфигурацию или 1, если функция в полном объеме входит в конфигурацию).

Вероятность вхождения i-й функции в j-ю конфигурацию ИС определяется экспертами:

$$P_j = \begin{cases} 0, \\ 1 \end{cases} \quad i \in (1,n), j \in (1,k), \text{ где} \tag{4}$$

n – множество функций,

k – множество конфигураций ИС.

Вероятностная стоимостная матрица является более гибкой в определении конфигурации системы, однако предусматривает дополнительные работы, которые реализовали бы вероятность вхождения какой-либо функции в конфигурацию. Подразумевается, что функция с вероятностью P = 1 полностью входит в конфигурацию, функция с вероятностью P = 0 не входит в конфигурацию, а функция с вероятностью P = 0,5 входит в конфигурацию системы с ограниченной функциональностью (часть возможностей недоступна).

Вероятность вхождения i-й функции в j-ю конфигурацию определяется экспертами:

$$P_{ij} \in [0,1] \quad i \in (1,n), j \in (1,k), \tag{5}$$

где n – множество функций,

k – множество конфигураций.

В общем виде стоимостную матрицу можно представить следующим образом:

$$R = \begin{bmatrix} P_{11} & P_{12} & \dots & P_{1k} \\ P_{21} & P_{22} & \dots & P_{2k} \\ & \dots & & \\ P_{n1} & P_{n2} & \dots & P_{nk} \end{bmatrix}. \tag{6}$$

По вертикали откладываются варианты конфигурации ИС, а по горизонтали – указываются возможные функции. Стоимость конфигурации определяется как сумма реализации всех функций, входящих в конфигурацию (имеют нормированное значение по предыдущему опыту). В случае использования вероятностной стоимостной матрицы, нормируемая величина стоимости реализации функции умножается на коэффициент, который определяется вероятностью вхождения функции в определенную конфигурацию. Следует отметить, что разработчик может задать минимальную вероятность вхождения функции в конфигурацию ИС.

При оценке сроков разработки каждой отдельной конфигурации ИС разработчик может использовать следующие варианты проведения работ:
- работы проводятся параллельно;
- работы проводятся последовательно;
- работы проводятся при параллельной и последовательной организации работ.

На рис. 3 представлены сетевые графы перечисленных вариантов компоновки работ.

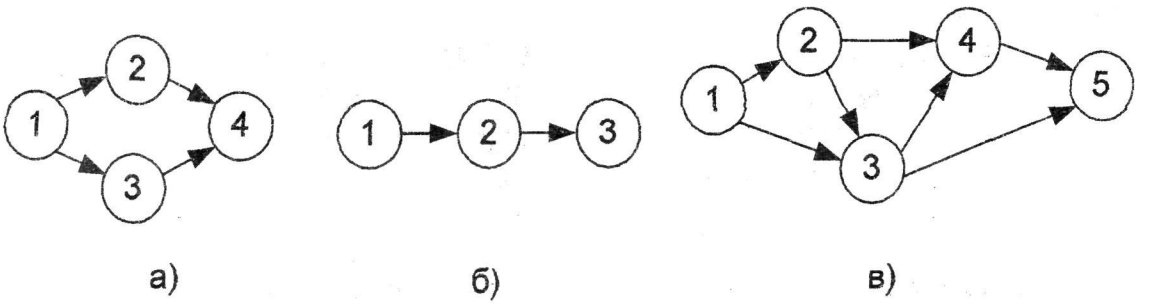

Рис. 3. Сетевые графы вариантов компоновки работ

В первом случае срок разработки конфигурации определяется временем реализации наиболее длительного срока выполнения работы, во втором – суммарное время выполнения работ, в третьем – срок разработки определяется критическим путем сетевого графа.

Выводы

При рассмотрении методологий (SSADM, DSDM и методологии разработки ИС согласно группе стандартов ГОСТ 34) был сделан вывод, что их использование практически не обеспечивает формализацию этапа макропроектирования, в частности не определяют механизм взаимодействия Заказчика и Разработчика на ранних этапах проектирования. В то же время, своевременный успех реализации проекта ИС в целом во многом зависит от корректности реализации работ на этапе макропроектирования процесса разработки ИС.

Для снижения рисков, с которыми могут столкнуться заказчик и разработчик (риски, связанные со срывом сроков и нарушением бюджета проекта, риски невыполнения проекта и т.д.), в данной работе предложена имитационная модель и алгоритм процесса перехода от этапа макропроектирования к этапу микропроектирования ИС. Предложены модели формализации оценки стоимости и сроков реализации функциональных требований, а также механизм оценки стоимости реализации нефункциональных требований.

Выполнение работ по реализации этапов имитационной модели согласно разработанному алгоритму позволит снизить риски разработки ИС, тем самым сохранить средства, время и репутацию организации-разработчика и организации заказчика.

ЛИТЕРАТУРА:

1. Левыкин, В. М. Концепция эволюционного прототипирования информационных систем [Текст] / В. М. Левыкин, И. В. Левыкин // АСУ та прилади автоматики. – 2008. – № 144. – С. 26-31 с.

2. Кириллов, В. П. SSADM – передовая технология разработки автоматизированных систем [Текст] / В. П. Кириллов // Компьютер + программы. – 1994. – № 2. – С. 8-16 с.

3. John Crinnion: Evolutionary Systems Development, a practical guide to the use of prototyping within a structured systems methodology. – Plenum Press, New York, 1991. – Page 18.

4. Советов, Б. Я. Моделирование систем [Текст] / Б. Я. Советов, С. А. Яковлев. – М. : «Высшая школа», 1985. – 270 с.

5. Левикін, В. М. Підхід до моделювання прйняття рішень при управлінні передпроектними стадіями створення інформаційної системи / В. М. Левикін, М. В. Євланов, В. О. Антонов // Вісник Харківського національного технічного університету сільського господарства ім. П. Василенка. – 2006. – Вип. 43. – Т. 2. – С. 177-181.

Поступила в редакцию 17.09.2009.

Левыкин Виктор Макарович, доктор технических наук, профессор. Харьковский национальный университет радиоэлектроники, заведующий кафедрой информационных управляющих систем. Тел.: (057) 702-14-51.

Черненко Николай Владимирович, апспирант кафедры ИУС. E-mail: ius@kture.kharkov.ua