УДК 004.94

МЕТОД АНАЛИЗА СЛОЖНЫХ СИСТЕМ ПО КРИТЕРИЮ ФУНКЦИОНАЛЬНОЙ ПОЛНОТЫ: РАСШИРЕНИЕ И АДАПТАЦИЯ

С. М. Щербаков, канд. экон. наук, доцент кафедры экономической информатики и автоматизации управления ГОУ ВПО «Ростовский государственный экономический университет (РИНХ)»

В статье рассматривается метод сравнительного анализа сложных систем по критерию функциональной полноты, позволяющий проводить сравнение и выбор программных систем. Предлагаются расширения метода, ориентированные на его применение в области построения и использования интернет-приложений.

Ключевые слова: сравнение сложных систем, формализация, функциональная полнота, интернет-приложения

Метод сравнительного анализа сложных систем по критерию функциональной полноты проф. Г. Н. Хубаева [3] является простым и универсальным средством, позволяющим систематизировать знания о предметной области, сравнить изучаемые системы и обеспечить выбор в соответствии с требованиями потребителя. Метод может применяться как для выбора существующих систем, так и для сравнения проектов.

Метод сравнительного анализа сложных систем по критерию функциональной полноты широко используется для решения следующих задач: сравнительный анализ и выбор программной системы; анализ любых сложных технических систем; анализ документооборота; анализ программ подготовки специалистов; анализ защищенности информационных систем и т.д.

Задача выбора программного продукта предполагает сравнение десятков систем, включающих иногда сотни функций. Анализ по критерию функциональной полноты позволяет на количественном уровне сопоставить между собой сравниваемые программные продукты и оценить соответствие их требованиям потребителя. Использование метода позволяет, в частности, ответить на вопрос, чем предлагаемая система превосходит другие коммерческие и свободно распространяемые программные средства.

Еще одним преимуществом метода анализа сложных систем по критерию функциональной полноты является возможность классификации изучаемых программных средств. Метод позволяет построить граф подобия систем, который дает возможность выделить группы схожих программных средств. Эти группы отличаются своими особенностями и/или масштабом [1, 4].

Метод поддается автоматизации, одна из его реализаций описана в [1].

Краткий обзор метода.

Метод анализа сложных систем по критерию функциональной полноты основан на анализе тройки:

< S, F, X>,

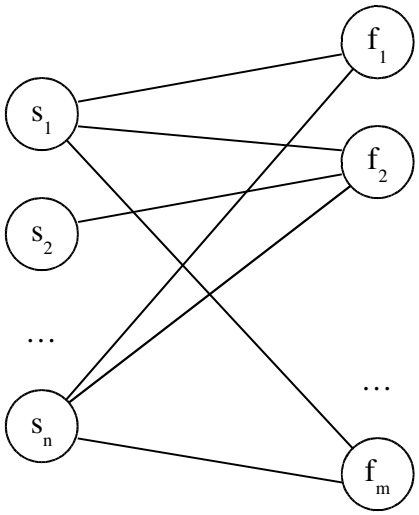
где $S = \{s_i\}, i = \overline{1,n}$ - множество анализируемых систем (n – число анализируемых систем);

$F = \{f_i\}, i = \overline{1,m}$ - множество функций программных систем (m – величина перечня функций);

$X = \{x_{ij}\}, i = \overline{1,n}, j = \overline{1,m}$ - матрица, определяющая реализацию системой s_i функции f_j.

$$x_j^i = \begin{cases} 1, \text{если функция j входит в систему i} \\ 0, \text{если функция j не входит в систему i} \end{cases}.$$

Исходные данные можно в целом представить в виде двудольного графа (рис. 1).



Р и с у н о к 1. **Отношения реализации между системами и функциями**

На основе исходных данных рассчитываются характеристики:

$P_{ik}^{11} = |S_i \cap S_k|$ – мощность пересечения систем S_i и S_k по функциям;

$P_{ik}^{01} = |S_k / S_i|$ и $P_{ik}^{10} = |S_i / S_k|$ - мощности разности систем.

В качестве меры рассогласования между системами S_i и S_k используется величина $R_{ik} = P_{ik}^{01} / \left(P_{ik}^{11} + P_{ik}^{10}\right)$; для оценки степени поглощения системой S_k системы S_i – величина $H_{ik} = P_{ik}^{11} / (P_{ik}^{11} + P_{ik}^{10})$; для оценки степени подобия систем используется мера подобия Жаккарда $G_{ik} = P_{ik}^{11} / (P_{ik}^{11} + P_{ik}^{10} + P_{ik}^{01})$.

После выполнения соответствующих расчетов формируются матрицы $\|P_{ik}^{01}\|, \|R_{ik}\|, \|H_{ik}\|, \|G_{ik}\|$. Рассчитанные матрицы преобразуются в логические матрицы P_0, R_0, H_0, G_0, в соответствии с различными пороговыми значениями.

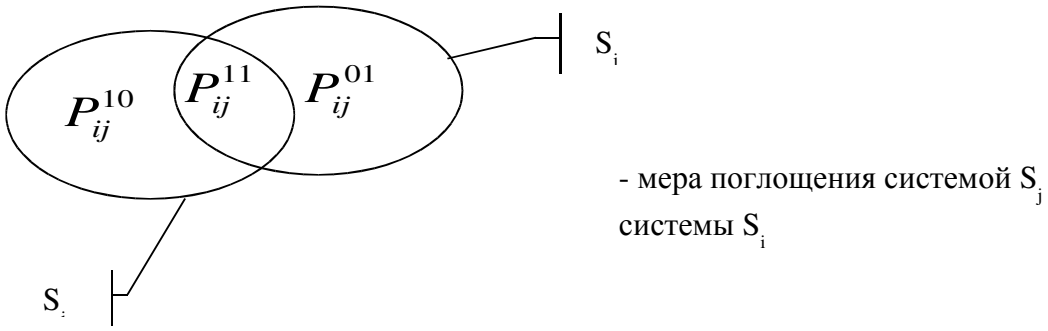
Например, для матрицы подобия систем на основе порогового значения ε_g рассчитается матрица $G_0=\{G^0_{ij}\}$, где:

$$G^0_{ij} = \begin{cases} 1, \text{ если } G_{ij} \geq \varepsilon_g \text{ , } i \neq j \\ 0, \text{ если } G_{ij} < \varepsilon_g \text{ , или } i = j \end{cases}$$

На основании матриц H_0 и G_0 могут быть построены графы поглощения и взаимосвязи.

Анализ матриц и графов дает возможность выделить группы подобных систем, определить системы превосходящие другие и выбрать систему или системы, которые в наибольшей степени соответствуют требованиям заказчика.

Для визуализации метода можно использовать диаграммы Эйлера-Венна, как это показано на рис. 2. Овалы символизируют множества функций, реализуемых каждой из двух систем. В расчетах используются мощность пересечения и мощность разности этих множеств.



Р и с у н о к 2 **Иллюстрация расчета характеристик подобия и поглощения систем**

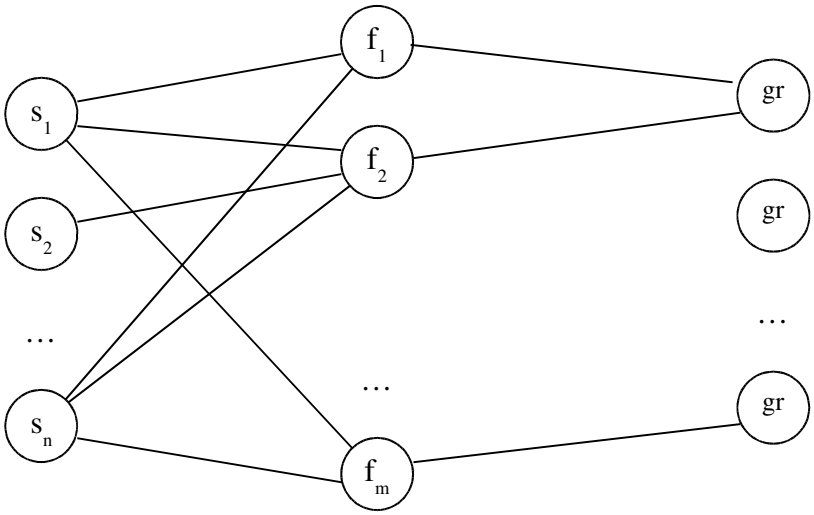
Рассмотрим расширения метода анализа сложных программных систем по критерию функциональной полноты, необходимые для решения задач формирования структуры информационных систем, в том числе таких систем как интернет-приложения.

Будем вводить в модель дополнительные сущности, при этом сохраняя общие принципы метода. Исходная матрица X, будет модифицироваться с целью интеграции дополнительных сущностей (будут меняться строки и столбцы). Таким образом, будет полностью сохранен алгоритм расчетов исходного метода.

Расширение 1. Использование групп функций.

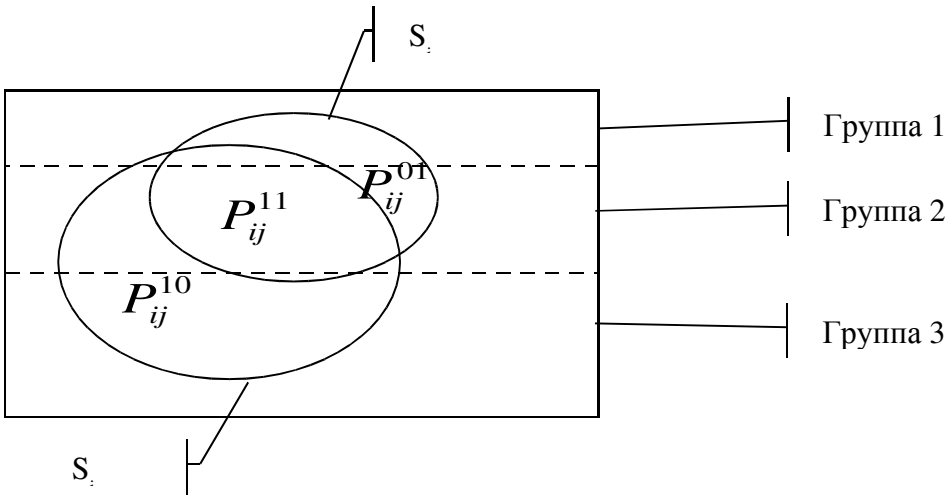
Введение групп функций (см. граф на рис. 3) с одной стороны упрощает пользователю системы работу с перечнем функций, с другой – дает возможность не просто сравнивать возможности систем, но и оценивать насколько одна система превосходит другую в определенной области.

Например, при сравнении различных систем автоматизации бухгалтерского учета можно увидеть, что первая система превосходит вторую. Однако с точки зрения группы учетных функций это превосходство может отсутствовать (рис 4).



Р и с у н о к 3 **Модифицированный граф исходных данных метода**

Несмотря на то, что система S_i реализует значительно большее число функции, по сравнению с системой S_j в контексте первой группы функций картина практически противоположная.



Р и с у н о к 4 **Разность множеств функций с учетом групп**

Использование групп требует внесения в алгоритм следующих изменений:

1) исходные данные дополняются множеством групп функций $Gr = \{Gr_k\}, k = \overline{1, Ngr}$.

2) Перед выполнением основных шагов алгоритма выполняется этап фильтрации. Фильтр формируется, как объединение групп функций, интересующих пользователя $Flt = Gr_{k1} \cup Gr_{k2} \cup Gr_{k3} \ldots$

Вместо исходных множеств S_i используются множества $S_i' = S_i \cap Flt$. Соответствующие изменения вносятся в матрицу X.

3) Проводятся стандартные шаги методики.

Дополнительным шагом может служить построение графа поглощения, связывающего системы с группами функций (рис. 5).

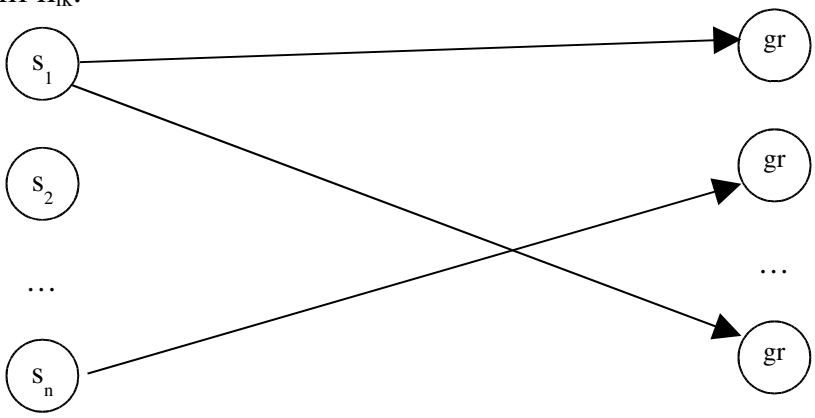
Для построения графа необходимо:

1) дополнить матрицу X строками, соответствующими группам функций. При этом:

$$x_j^{i+k} = \begin{cases} 1, \text{если функция j принадлежит группе k} \\ 0, \text{если функция j не принадлежит группе k} \end{cases}$$

2) Далее в порядке стандартных шагов выполнения методики рассчитываются промежуточные матрицы и матрица поглощения H.

3) По матрице H (либо по матрице H_0) строится подграф, включающий системы {S}, группы {Gr} и связи между элементами этих множеств, помеченные значениями h_{ik}.



Р и с у н о к 5 **Граф поглощения групп функций**

Анализ графа показывает, какая система в наибольшей степени реализует функции из той или иной группы.

Расширение 2. Использование количественных и качественных параметров систем.

Помимо функций при выборе следует рассматривать также параметры сравниваемых систем. В ряде случаев значения параметров системы могут определять возможность или невозможность ее использования для решения поставленных задач. В отличие от функций параметры не могут быть выражены в бинарном виде.

В таблице 1 приведен пример нескольких систем, которые необходимо подвергнуть сравнительному анализу и последующему выбору.

Т а б л и ц а 1

Пример сравниваемых систем

	Ф1	Ф2	Ф3	Ф4	Интерфейс	Производительность
Система 1	1	1	0	1	отличный	10000
Система 2	1	0	1	1	хороший	5000
Система 3	0	0	1	1	плохой	1000

Вместо исходной тройки теперь необходимо анализировать конструкцию:

< S, F, X, <Pkol, ZPkol>,<Pkach,Varkach,ZPV>>,

где $Pkol = \{Pkol_k\}, k = \overline{1, Npkol}$ – множество количественных параметров;

$ZPkol = \{ZPkol_{ikk}\}, k = \overline{1, Npkol}, i = \overline{1, Nl}$ – матрица значений параметров;

$Pkach = \{Pkach_k\}, k = \overline{1, Npkach}$ – множество качественных параметров;

$Varkach = \{Varkach_{kj}\}, j = \overline{1, Nvkach_k}, k = \overline{1, Npkach}$ - множество возможных значений качественных параметров.

$ZPV = \{ZPV_{ki}\}, k = \overline{1, Npkach}, i = \overline{1, N}, ZPV_{k,i} \in \{Varkach\}_j$ - матрица, показывающая значения всех качественных параметров для всех систем.

Для преобразования к бинарной форме используем следующие правила. Для качественных параметров дополняем матрицу X столбцами. Каждому параметру будет соответствовать h-1 столбцов, где h – число вариантов этого параметра. Наихудший из вариантов исключается – он служит базой.

$$x_i^l = \begin{cases} 1, \text{если } \mathrm{ZPV_{ki}} \succ Varach_l \\ 0, \text{иначе} \end{cases}$$

Для количественных параметров необходимо ввести несколько пороговых значений (преобразовав их, таким образом, в качественные) и повторить предыдущую процедуру.

Таблица рассмотренного примера теперь примет вид, показанный в таблице 2.

Таблица 2

Пример сравниваемых систем

	Ф1	Ф2	Ф3	Ф4	Интерфейс		Производительность
					Ф5 (хороший)	Ф6 (отличный)	Ф7 (>3000)
Система 1	1	1	0	1	1	1	1
Система 2	1	0	1	1	1	0	1
Система 3	0	0	1	1	0	0	0

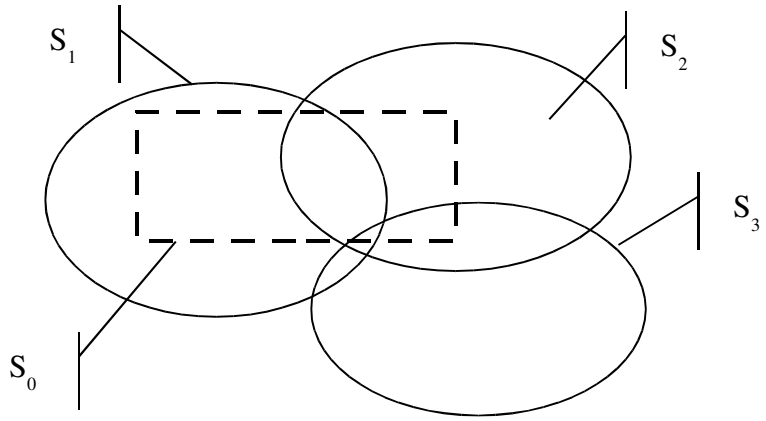
Дальнейший анализ проводится по стандартной методике.

Расширение 3. Использование нескольких систем одновременно.

Часто для решения задачи можно использовать не одну систему, а некоторую комбинацию систем. Например, возможно ведение складского учета в системе «1С: Предприятие» и начисление заработной платы в системе, реализованной отделом АСУ организации. Если речь идет не о прикладных программных системах, а об инструментах разработки, использование нескольких систем встречается довольно часто.

Как показано на рис. 6, требования пользователя к функциональной полноте S_0 (пунктирный прямоугольник) могут быть удовлетворены с помощью комбинации систем S_1 и S_2.

Разумеется, невозможно рассматривать все возможные сочетаний сравниваемых систем. Нужен способ отбора комбинаций, которые будут представлены в виде строк матрицы X и рассмотрены наряду с одиночными системами. Не все сравниваемые системы совместимы между собой, поэтому в анализ нужно внести матрицу совместимости. Далее, очевидно есть смысл комбинировать те системы, которые ориентированы на реализацию различных множеств функций.



Р и с у н о к 6 **Использование комбинаций систем**

Таким образом, будет рассмотрена четверка:

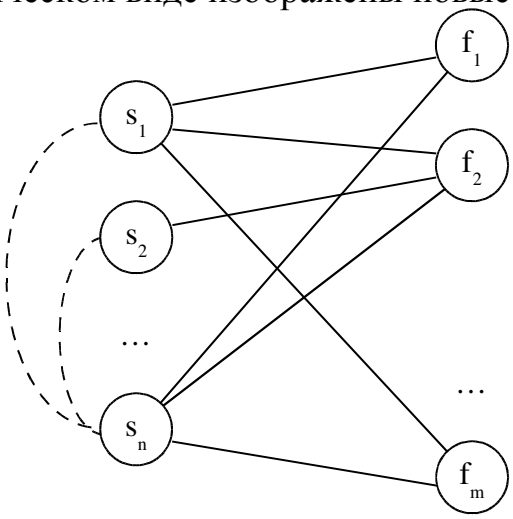
$$< S, F, X, C>,$$

где

$C = \{c_{ij}\}, i = \overline{1,n}, j = \overline{1,n}$ - матрица, определяющая совместимость систем s_i и s_j.

$$c_j^i = \begin{cases} 1, \text{если системы } s_i \text{ и } s_j \text{ совместимы} \\ 0, \text{иначе} \end{cases}.$$

На рис. 7 в графическом виде изображены новые исходные данные.



Р и с у н о к 7 **Исходный граф, дополненный отношениями совместимости между системами** (пунктир)

При формировании комбинаций будем придерживаться следующего правила:

$$S_k = S_i \cup S_j \left| c_{ij} = 1, G_{ij} \le \varepsilon, i, j = \overline{1,n}, i \ne j \right.$$

В матрицу X добавляется строка, соответствующая комбинации систем S_i и S_j, если:

1) эти системы совместимы;

2) степень подобия систем не превышает определенный порог.

Этот шаг может быть повторен для комбинаций из трех или более систем. Матрица совместимости для этого модифицируется следующим образом: $c_{hk} = c_{ih} \cdot c_{jh}$, то есть любая система совместима с комбинацией систем, если она совместима с каждой из составляющих этой комбинации.

В таблице 3 представлен пример ситуации, когда используются несколько функций.

Таблица 3

Исходные данные примера

	Ф1	Ф2	Ф3	Ф4	Ф5	Ф6
Система 1	1	1	0	1	0	0
Система 2	0	0	1	1	1	1
Система 3	0	0	1	1	0	0
Система 4	1	0	1	0	1	1
Система 5	1	1	0	0	0	0
Система 0 (требования пользователя)	1	1	0	0	1	1

В таблице 4 представлена матрица совместимости рассматриваемых систем.

Таблица 4

Матрица совместимости

	Система 1	Система 2	Система 3	Система 4	Система 5
Система 1	1	1	1	0	0
Система 2	1	1	0	0	0
Система 3	1	0	1	1	0
Система 4	0	0	1	1	1
Система 5	0	0	0	1	1

Первый шаг состоит в расширении числа систем за счет введения комбинаций. При этом используются матрица совместимости и результат расчета степени подобия систем (таблица 5).

Таблица 5

Мера подобия Жаккарда

	Система 1	Система 2	Система 3	Система 4	Система 5
Система 1	1	0,17	0,25	0,25	0,67
Система 2	0,17	1	0,5	0,6	0
Система 3	0,25	0,5	1	0,2	0
Система 4	0,25	0,6	0,2	1	0,2
Система 5	0,67	0	0	0,2	1

При пороговом значении ε=0,2 множество систем S будет дополнено системами $S_6 = S_1 \cap S_2$, $S_7 = S_3 \cap S_4$, $S_8 = S_4 \cap S_5$. Матрица X, таким образом, примет вид, показанный в таблице 6.

Таблица 6

Результат формирования комбинаций

	Ф1	Ф2	Ф3	Ф4	Ф5	Ф6
Система 1	1	1	0	1	0	0
Система 2	0	0	1	1	1	1
Система 3	0	0	1	1	0	0
Система 4	1	0	1	0	1	1
Система 5	1	1	0	0	0	0
Система 1 + Система 2	1	1	0	1	1	1
Система 3 + Система 4	1	0	1	1	1	1
Система 4 + Система 5	1	1	1	0	1	1
Система 0 (требования пользователя)	1	1	0	0	1	1

Дальнейшие расчеты проводятся в соответствии со стандартной методикой.

Чтобы ограничить рост числа комбинаций, можно ввести дополнительное требование для включение пары систем исходя напрямую из требований пользователя, выраженных системой S_0. Можно отбирать только те комбинации, которые перекрывают потребности пользователя в достаточной степени.

$$S_k = S_i \cup S_j \left| c_{ij} = 1, G_{ij} \le \varepsilon, i, j = \overline{1,n}, i \ne j, \frac{\left|\left(S_i \cup S_j\right) \cap S_0\right|}{\left|S_0\right|} > \delta \right.$$

Дополнительное условие говорит, что степень поглощения комбинацией двух систем системы S_0, должна превышать некоторый порог δ.

Расширение 4. Зависимости между функциями.

На результат анализа и выбора программного средства большое влияние оказывают зависимости между отдельными функциями систем. Рассмотрим три наиболее часто встречающихся случая (рис. 8).

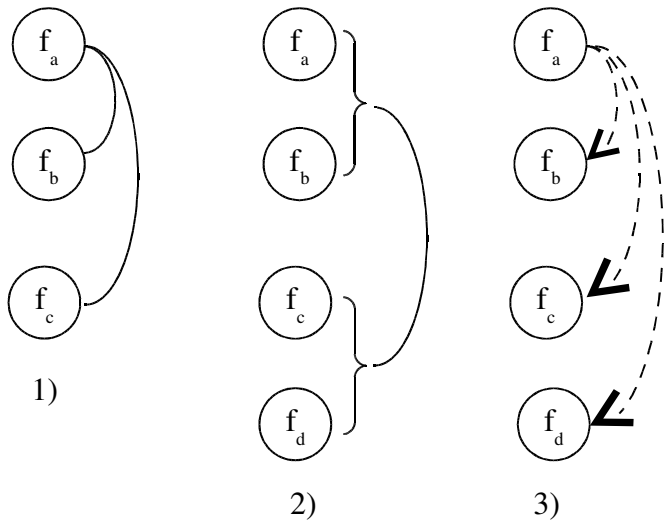


Рисунок 8 **Зависимости между функциями**

1) *Функция f_c соответствует паре функций f_a и f_b.* Решением является исключение функции f_c из множества функций с модификацией матрицы X следующим образом:

$$X_{ia}' = \begin{cases} 1, X_{ic} = 1 \\ X_{ia}, \text{иначе} \end{cases} \qquad X_{ib}' = \begin{cases} 1, X_{ic} = 1 \\ X_{ib}, \text{иначе} \end{cases}$$

Иными словами, если какая-то система реализует функцию f_c, то для этой системы устанавливается и реализация функций f_a *и* f_b.

2) *Комбинация функций f_a и f_b эквивалентна комбинации f_c и f_d.*

Матрица X, приведенная в таблице 7 иллюстрирует эту ситуацию. Считаем, что комбинация Φ_1 и Φ_2, эквивалентна Φ_3 и Φ_4. Следует выбрать систему «Система 1», поскольку она реализует функции Φ_1 и Φ_2 (которые эквивалентны функциям Φ_3 и Φ_4 , которые необходимы пользователю), а также функцию Φ_5. Ни одна другая система не удовлетворяет требования пользователя в полной мере. Однако необходимость такого выбора в условиях эквивалентности комбинаций функций из таблицы не видна.

Таблица 7

Выбор в условиях эквивалентных функций

	Ф1	Ф2	Ф3	Ф4	Ф5
Система 1	1	1	1	0	1
Система 2	0	1	1	0	1
Система 3	1	1	1	1	0
Система 0 (требования пользователя)	0	0	1	1	1

Возможное решение заключается во введении дополнительной функции f_e, которая эквивалентна комбинации функций f_a и f_b и комбинации f_c и f_d (функция Φ_6 в таблице 8).

Таблица 8

Добавление фиктивной функции

	Ф1	Ф2	Ф3	Ф4	Ф5	Ф6
Система 1	1	1	1	0	1	1
Система 2	0	1	1	0	1	0
Система 3	1	1	1	1	0	0
Система 0 (требования пользователя)	0	0	1	1	1	1

Далее необходимо исключить дополнительную функцию по правилу, описанному для предыдущего случая:

$$X_{ia}' = \begin{cases} 1, X_{ie} = 1 \\ X_{ia}, \text{иначе} \end{cases} \qquad X_{ib}' = \begin{cases} 1, X_{ie} = 1 \\ X_{ib}, \text{иначе} \end{cases}$$

$$X_{c}' = \begin{cases} 1, X_{ie} = 1 \\ X_{ic}, \text{иначе} \end{cases} \qquad X_{id}' = \begin{cases} 1, X_{ie} = 1 \\ X_{id}, \text{иначе} \end{cases}$$

В таблице 9 представлен пример после преобразования. Теперь следование стандартной методики приведет к корректному выбору.

Таблица 9

Выбор в условиях эквивалентных функций (результат преобразования)

	Ф1	Ф2	Ф3	Ф4	Ф5
Система 1	1	1	1	1	1
Система 2	0	1	1	0	1
Система 3	1	1	1	1	0
Система 0 (требования пользователя)	1	1	1	1	1

Промежуточный шаг с добавлением фиктивной функции можно не выполнять, вместо этого, преобразование можно проводить напрямую по правилу:

$$X_{ia}' = \begin{cases} 1, X_{ic} = 1, X_{id} = 1 \\ X_{ia}, \text{иначе} \end{cases} \qquad X_{ib}' = \begin{cases} 1, X_{ic} = 1, X_{id} = 1 \\ X_{ib}, \text{иначе} \end{cases}$$

$$X_{ic}' = \begin{cases} 1, X_{ia} = 1, X_{ib} = 1 \\ X_{ic}, \text{иначе} \end{cases} \qquad X_{id}' = \begin{cases} 1, X_{ia} = 1, X_{ib} = 1 \\ X_{id}, \text{иначе} \end{cases}$$

3) Функция f_a может использоваться для реализации функций f_b, f_c.

Достаточно распространенной является ситуация, когда функция f_a представляет, например, конструктор запросов, а функции f_b и f_c – конкретные запросы,. Рис. 9 иллюстрирует подобный случай. Система S_i реализует лишь малую часть необходимых пользователю функций S_0 (пунктирный прямоугольник). Однако потенциально система S_i расширенная до S_i` (эллипс из точек) может практически полностью перекрывать требования пользователя S_0.

Возможным решением является введение фиктивных функций для всех тех функций, которые могут быть реализованы с помощью универсальной функции f_a (конструктора).

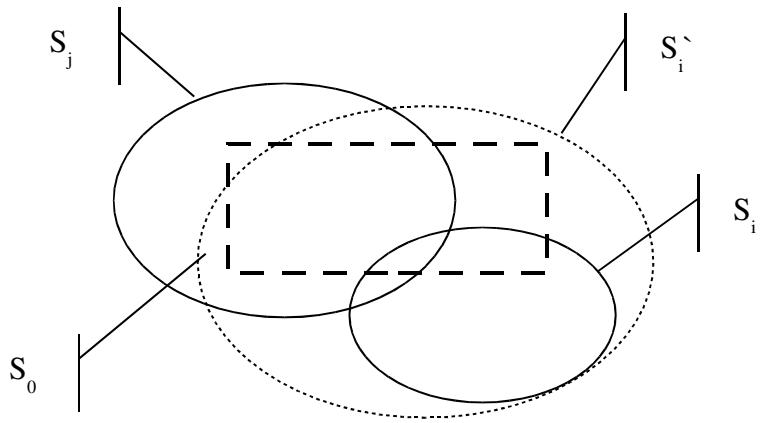
Правило введения следующее:

$$X_{ib'} = \begin{cases} 1, X_{ia} = 1 \\ 0, \text{иначе} \end{cases}, \forall i > 0 \qquad X_{0b'} = \begin{cases} 1, X_{0a} = 1 \\ X_{0b}, \text{иначе} \end{cases}.$$

То есть фиктивная функция добавляется при наличии в системе конструктора, способного ее реализовать. Таким образом, фиктивная функция представляет возможность реализации соответствующей исходной функции с помощью конструктора.

При этом для системы, представляющей требования пользователя, фиктивная функция f_b` добавляется, если соответствующая функция f_b присутствует в системе S_0.

Применение подобного расширения имеет смысл, если затраты на реализацию функции с помощью встроенного конструктора не велики, по сравнению со стоимостью системы.



Р и с у н о к 9 **Возможность расширения системы**

Пример рассматриваемой ситуации представлен в таблице 10.

Т а б л и ц а 10

Выбор в условиях возможного расширения системы

	Конструктор	Запросы		
	Ф1	Ф2	Ф3	Ф4
Система 1	1	0	0	0
Система 2	1	1	0	0
Система 3	0	1	0	1
Система 0 (требования пользователя)	0	1	1	1

С помощью конструктора $Ф_1$ могут быть реализованы запросы $Ф_2$, $Ф_3$, $Ф_4$. Здесь следует выбрать не «Систему 3», которая имеет самую высокую степень поглощения «Системы 0», а сначала «Систему 2», затем «Систему 1», поскольку все функции необходимые пользователю могут быть реализованы с помощью имеющихся у этих систем конструктора.

Эта возможность представлена в виде функций $Ф_2$`, $Ф_3$`, $Ф_4$` (таблица 11). Функции $Ф_2$, $Ф_3$, $Ф_4$ сохранены для того, чтобы избежать выбора в пользу «пустого» конструктора.

Т а б л и ц а 11

Введение фиктивных запросов

	Конструктор	Запросы			Фиктивные запросы		
	Ф1	Ф2	Ф3	Ф4	Ф2`	Ф3`	Ф4`
Система 1	1	0	0	0	1	1	1
Система 2	1	1	0	0	1	1	1
Система 3	0	1	0	1	0	0	0
Система 0 (требования пользователя)	0	1	1	1	1	1	1

Степень поглощения H_{0i} условной системы сравниваемыми системами: H_{01}=0.50; H_{02}=0.67; H_{03}=0.33. Таким образом, первой будет выбрана «Система 2», затем «Система 1».

Расширение 5. Использование технологий.

Достаточно частым является случай, когда при сравнительном анализе и выборе программных систем значительную роль играют технологии их реализации. Во-первых, требования пользователя могут относиться не только к функциям системы, но и к функциям технологии, например, необходима возможность расширения возможностей системы в будущем или требуется определенный уровень производительности. Иногда система может быть реализована в рамках одной из нескольких технологий. Выбор технологии и выбор системы связаны между собой и должны выполняться одновременно.

Особенно часто такая проблема возникает, когда сравниваются не программные системы, а проекты их разработки. В таких случаях требования пользователя часто затрагивают не только создаваемые компоненты, но и возможности их сопровождения, модификации и т.д.

В анализ необходимо дополнительно ввести множество технологий, множество функций технологий и матрицу поддержки, аналогично исходной тройке. А также вводится матрица, которая показывает, с помощью каких технологий могут быть реализованы функции сравниваемых систем.

Таким образом, будет рассмотрена конструкция:

<< S, F, X>, <T, TF, TX>, R>,

где $T = \{T_i\}, i = \overline{1, nt}$ – множество анализируемых технологий (nt – число анализируемых технологий);

$TF = \{tf_i\}, i = \overline{1, mt}$ - множество функций технологий (mt – величина перечня функций);

$TX = \{tx_{ij}\}, i = \overline{1, nt}, j = \overline{1, mt}$ - матрица, определяющая наличие у технологии t_i функции tf_j.

$$tx_j^i = \begin{cases} 1, \text{если функция j входит в ттехнологю i} \\ 0, \text{если функция j не входит в ттенологи i} \end{cases};$$

$R = \{r_{ij}\}, i = \overline{1, m}, j = \overline{1, nt}$ - матрица, определяющая возможность реализации функции f_i с помощью технологии t_j.

$$r_j^i = \begin{cases} 1, \text{если функция i может быть реализована с помощью ттехнологи j} \\ 0, \textit{иначе} \end{cases}.$$

Новые исходные данные изображены в графическом виде на рис. 10. Пунктирными линиями показаны отношения потенциальной реализации функций систем на основе технологий.

Модифицируем исходную матрицу X следующим образом:

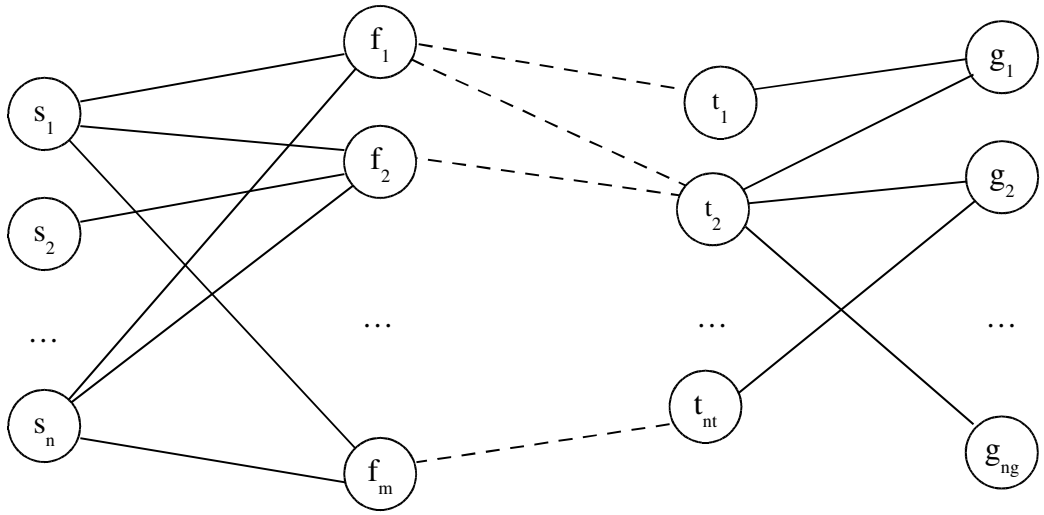
1) введем дополнительные столбцы, соответствующие функциям технологий:

$F' = F \cup TF$

2) добавим строки для всех возможных сочетаний функций и технологий:

$S' = \{(s_i, t_j) | x_{ik} = 1 \rightarrow r_{kj} = 1, \forall k\}$

То есть, включаем в результирующую матрицу строку для системы s_i на основе технологии t_j, если все функции системы s_i могут быть реализованы на основе технологии t_j.



Р и с у н о к 10 **Граф исходных данных с учетом технологий**

Рассмотрим пример применения метода в условиях использования технологий. В таблице 12 приведена матрица X, описывающая состав функций сравниваемых систем. В таблице 13 показана матрица ТХ, описывающая функции технологий. Таблица 14 показывает, какие функции систем могут быть реализованы в рамках той или иной технологии.

Т а б л и ц а 12

Сравниваемые системы

	Ф1	Ф2	Ф3	Ф4	Ф5
Система 1	0	0	1	1	0
Система 2	1	1	1	1	1
Система 3	1	1	1	0	1
Система 0 (требования пользователя)	0	1	1	0	1

Т а б л и ц а 13

Технологии построения систем

	ТФ1	ТФ2	ТФ3	ТФ4
Технология 1	1	0	1	1
Технология 2	0	1	1	1
Технология 3	1	1	1	0
Технология 0 (требования пользователя к технологии)	1	0	1	0

Т а б л и ц а 14

Реализация функций с помощью технологий

	Технология 1	Технология 2	Технология 3
Ф1	1	1	1
Ф2	0	1	1
Ф3	1	0	1
Ф4	1	0	1
Ф5	0	1	1

В таблице 15 представлен результат преобразования исходной матрицы.

Таблица 15

Вид матрицы X после преобразования

	Ф1	Ф2	Ф3	Ф4	Ф5	ТФ1	ТФ2	ТФ3	ТФ4
Система 1 на основе технологии 1	0	0	1	1	0	1	0	1	1
Система 1 на основе технологии 3	0	0	1	1	0	1	1	1	0
Система 2 на основе технологии 3	1	1	1	1	1	1	1	1	0
Система 3 на основе технологии 2	1	1	1	0	1	0	1	1	1
Система 3 на основе технологии 3	1	1	1	0	1	1	1	1	0
Система 0 (требования пользователя)	0	1	1	0	1	1	0	1	0

Дальнейший анализ проводится по стандартной методике.

Расширение 6. Иерархия систем.

Часто при построении информационной системы требуется сделать выбор не одной системы, а некоторой совокупности различных взаимосвязанных систем. Например, при создании интернет-приложения нужно выбрать средство разработки (php, asp, java и т.д.), web-сервер (Apache, MIIS и т.д.), СУБД (MySQL, MS SQL, Oracle, PostgreSQL и т.д.). Таким образом, необходимо принять решение сразу на нескольких уровнях:

$$\{< S^h, F^h, X^h >\}, h = \overline{1, nl},$$

где h-номер уровня, nl – число уровней.

Системы различных уровней могут быть несовместимы между собой, например, СУБД может работать только в рамках какой-то одной операционной системы. Для определения возможности совместного использования систем также используем матрицы совместимости:

$$\{c^{hg}\}, h = \overline{1, nl}, \mathrm{g} = \overline{1, \mathrm{nl}}, h \neq g,$$

$$c^{hg} = \{c_{ij}^{hg}\}, i = \overline{1, n_h}, j = \overline{1, n_g}$$

$$c_{ij}^{hg} = \begin{cases} 1, \text{если системы } \mathrm{s_i^h} \text{ и } \mathrm{s_j^g} \text{ совместимы} \\ 0, \textit{иначе} \end{cases}.$$

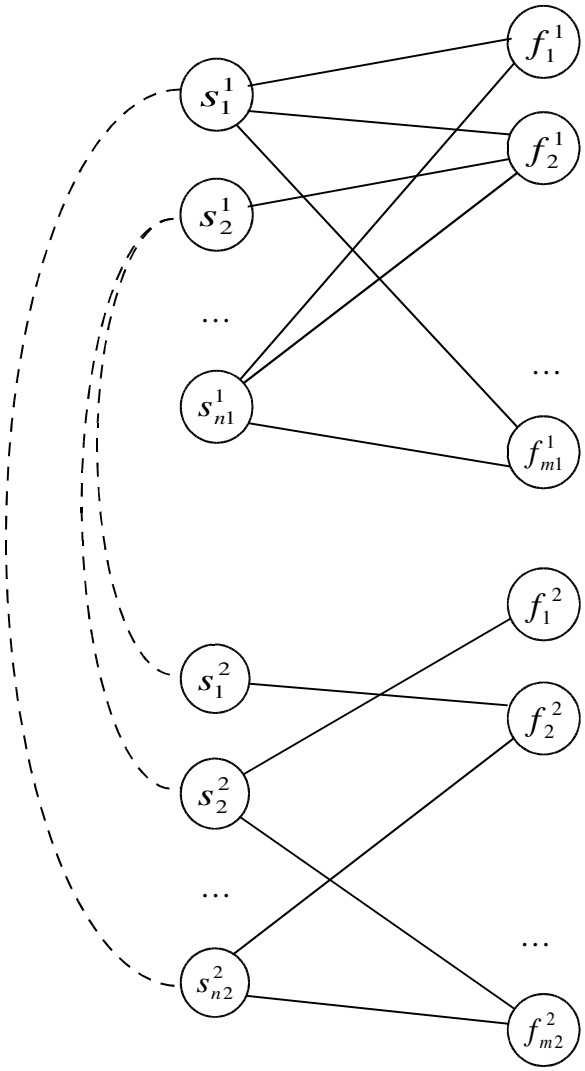
То есть матрицы характеризуют совместимость каждой системы определенного уровня со всеми системами других уровней. Рис. 11 иллюстрирует задачу выбора совокупности программных систем. Пунктирные линии показывают отношения совместимости между системами.

В таблице 16 представлен пример исходных данных. Условные системы «Система 10», «Система 20» и «Система 30» представляют требования пользователя к системам каждого уровня.

Таблица 16

Исходные данные для иерархии систем

Уровень 1		Функции				
		Ф11	Ф12	Ф13	Ф14	Ф15
	Система 11	1	0	1	0	0
	Система 12	1	1	1	0	1
	Система 13	0	0	1	1	1
	Система 10	1	0	1	0	1
Уровень 2		Функции				
		Ф21	Ф22	Ф23		
	Система 21	1	1	1		
	Система 22	0	1	1		
	Система 23	1	1	1		
	Система 24	1	0	0		
	Система 20	1	1	0		
Уровень 3		Функции				
		Ф31	Ф32	Ф33	Ф34	
	Система 31	1	1	1	0	
	Система 32	0	1	1	1	
	Система 33	1	0	0	1	
	Система 30	1	0	1	0	



Р и с у н о к 11 **Граф исходных данных для систем двух уровней**

Таблицы 17-19 содержат матрицы совместимости систем разных уровней.

Т а б л и ц а 17

Матрица совместимости систем первого-второго уровней

	Система 21	Система 22	Система 23	Система 24
Система 11	1	1	0	0
Система 12	1	1	1	1
Система 13	0	1	1	1

Т а б л и ц а 18

Матрица совместимости систем первого-третьего уровней

	Система 31	Система 32	Система 33
Система 11	1	0	0
Система 12	1	1	1
Система 13	1	1	1

Т а б л и ц а 19

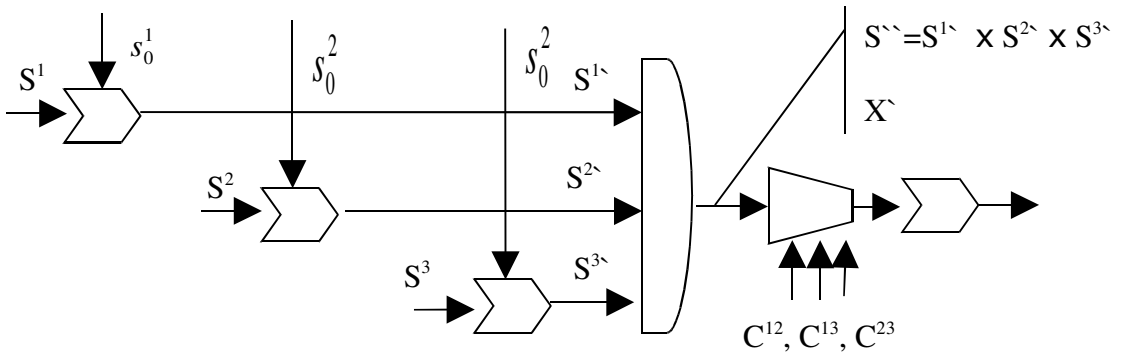
Матрица совместимости систем третьего-второго уровней

	Система 21	Система 22	Система 23	Система 24
Система 31	1	1	1	1
Система 32	1	1	0	1
Система 33	1	1	1	1

В некоторых случаях, при развитой открытой архитектуре рассматриваемой платформы может иметь значение только совместимость систем соседних уровней, тогда число матриц совместимости будет уменьшено.

Для построения единой матрицы необходимо: объединить столбцы матриц всех уровней, представляющие множество функций $F' = F^1 \cap F^2 \cap ... \cap F^{nl}$; добавить строки для всех возможных комбинаций систем всех уровней: $S' = \left\{ \left(S_{i1}^1, S_{i2}^2, ..., S_{inl}^{nl} \right) \middle| C_{ih,ig}^{hg} = 1, \forall h, g = \overline{1, nl} \right\}$. То есть отбираются такие комбинации систем, для которых все компоненты совместимы между собой.

Рассмотрим проведение анализа в несколько этапов, исключая системы, которые не соответствуют требованиям пользователя на каждом из уровней. Общая схема приведена на рис. 12.



Р и с у н о к 12 **Схема анализа сложных систем по критерию функциональной полноты**

в условиях иерархии систем (первый вариант)

Таким образом, на каждом уровне производится выбор систем, в наилучшей степени отражающих требования пользователя (в соответствии со стандартной методикой). Далее из отобранных систем каждого уровня формируется единая матрица X`, так, как это описано выше. Из декартового произведения отобранных систем отбрасываются системы, не удовлетворяющие условиям совместимости C^{12}, C^{13}, C^{23}.

Для предложенного примера будем проводить отбор систем на каждом уровне по рассчитанным матрицам поглощения H^1,H^2,H^3. Выберем пороговые значения $\varepsilon_H^1 = 0.67$, $\varepsilon_H^2 = 0.67$, $\varepsilon_H^3 = 0.67$ (в общем случае пороговые значения для разных уровней могут быть различны).

Для первого уровня: $H_{01}^1 = 0.67$, $H_{02}^1 = 1$, $H_{03}^1 = 0.67$, поскольку $H_{01}^1 \ge \varepsilon_H^1$, $H_{02}^1 \ge \varepsilon_H^1$, $H_{03}^1 \ge \varepsilon_H^1$, то отбираем системы S_1`={Система 11, Система 12, Система 13}.

Для второго уровня $H_{01}^2 = 1, H_{02}^2 = 0.5, H_{03}^2 = 1, H_{04}^2 = 0.5$. В соответствии с пороговым значением $\varepsilon_H^2 = 0.67$ отбираем системы S_2`={Система 21, Система 23}.

Для третьего уровня $H_{01}^3 = 1$, $H_{02}^3 = 0.5$, $H_{03}^3 = 0.5$. В соответствии с пороговым значением $\varepsilon_H^3 = 0.67$ отбираем систему S_3`={Система 31}.

Далее из декартового произведения $S^{1`} \times S^{2`} \times S^{3`}$ исключаются системы, не удовлетворяющие условию совместимости: $S'' = \left\{ \left(S_{i1}^1, S_{i2}^2, ..., S_{inl}^{nl} \right) \middle| C_{ih,ig}^{hg} = 1, \forall h, g = \overline{1, nl} \right\}$ Например, комбинация (Система 11, Система 23, Система 31) не удовлетворяет условию $C_{13}^{12} = 1, C_{11}^{13} = 1, C_{31}^{23} = 1$, поскольку $C_{13}^{12} = 0$. В итоге получим следующее подмножество декартового произведения, удовлетворяющее условиям совместимости:

(Система 11, Система 21, Система 31)
(Система 12, Система 21, Система 31)
(Система 12, Система 23, Система 31)
(Система 13, Система 23, Система 31).

Множества функций трех уровней объединяется $F' = F^1 \cap F^2 \cap ... \cap F^{nl}$. Результат проведенных шагов приведен в таблице 20.

Т а б л и ц а 20

Единая матрица X в условиях иерархии систем

	Ф11	Ф12	Ф13	Ф14	Ф15	Ф21	Ф22	Ф23	Ф31	Ф12	Ф13	Ф14
Система 11, Система 21, Система 31	1	0	1	0	0	1	1	1	1	1	1	0
Система 12, Система 21, Система 31	1	1	1	0	1	1	1	1	1	1	1	0
Система 12, Система 23, Система 31	1	1	1	0	1	1	1	1	1	1	1	0
Система 13, Система 23, Система 31	0	0	1	1	1	1	1	1	1	1	1	0
Система 0 (требования	1	0	1	0	1	1	1	0	1	0	1	0

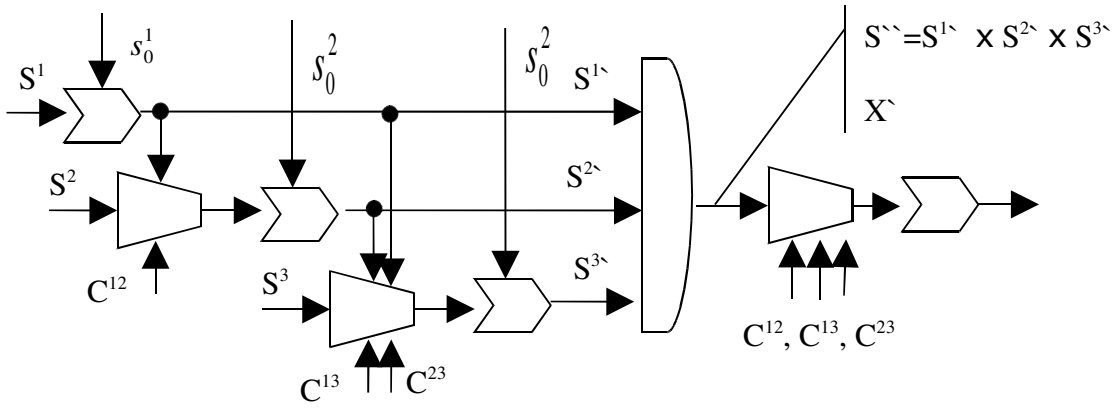
пользователя)												

Дальнейший анализ проводится по стандартной методике.

Возможна модификация предложенной схемы. Чтобы при проведении анализа на отдельных уровнях избежать независимой от других уровней оптимизации можно связать результаты анализа на каждом уровне.

На рис. 13 показана модифицированная схема. После отбора систем на первом уровне (в соответствии со стандартной методикой) множество систем второго уровня модифицируется, отбрасываются системы, не совместимые хотя бы с одной из отобранных систем первого уровня $S^{2^*} = \left\{ s_i^2 \middle| \exists s_j^1, c_{ji}^{12} = 1 \right\}$.

Далее по стандартной методике проводится анализ систем, относящихся ко второму уровню. А затем, результаты анализа первого и второго уровня, вместе с соответствующими матрицами совместимости используются для сокращения множества систем третьего уровня: $S^{3^*} = \left\{ s_i^3 \middle| \exists s_j^1, c_{ji}^{13} = 1, \middle| \exists s_k^2, c_{ki}^{23} = 1 \right\}$.



Р и с у н о к 13 **Схема анализа сложных систем по критерию функциональной полноты в условиях иерархии систем** (второй вариант)

Рассмотренные способы расширения и адаптации метода анализа сложных систем по критерию функциональной полноты могут использоваться для выбора и построения экономических информационных систем, например, для задач построения интернет-приложений [2, 6, 7].

Библиографические ссылки

1. Аручиди Н. А. Программный комплекс анализа сложных систем по критерию функциональной полноты / Н. А. Аручиди, С. М. Щербаков // Статистика в современном мире: методы, модели, инструменты. Материалы межвузовской научно-практической конференции. – Ростов н./Д.: Ростовский Государственный Экономический Университет «РИНХ», 2007. – С. 111-113.
2. Суворова А. Ю. Анализ систем управления контентом по критерию функциональной полноты / А. Ю. Суворова, Д. В. Суворов // Статистика в современном мире: методы, модели, инструменты. Материалы научно-практической конференции. – Ростов-на-Дону: РГЭУ «РИНХ», 2009. – С. 170-177.
3. Хубаев Г. Н. Сравнение сложных программных систем по критерию функциональной полноты /Г. Н. Хубаев // Программные продукты и системы (SOFTWARE&SYSTEMS). – 1998. – №2. – С.6-9.

4. Хубаев Г. Н. Экономическая оценка потребительского качества программных средств / Г. Н. Хубаев. – Ростов н/Д: РГЭА, 1997. – 94 с.
5. Хубаев Г. Н. ПС анализа сложных систем по критерию функциональной полноты «Ireland» / Г. Н. Хубаев, С. М. Щербаков, Н. А. Аручиди // Свидетельство об официальной регистрации программы для ЭВМ. - №2009615296. – М.: РОСПАТЕНТ, 2009.
6. Щербаков С. М. Основные направления исследования вопросов экономической эффективности интернет-приложений С. М. Щербаков // Научный поиск: по страницам докторских диссертаций. Выпуск 6. – Ростов н/Д.: РГЭУ «РИНХ», 2007. – С. 125-132.
7. Щербаков С. М. Экономические аспекты построения интернет-приложений: методы сравнительного анализа и выбора интернет-технологий / С. М. Щербаков С. М., Н. А. Аручиди // Экономические науки. – № 43. – 2008. – С. 381-387.