Технология решения задач в нормализованной среде радикалов

М. В. ПИРОГОВ
Научно-производственное объединение им. С. А. Лавочкина
e-mail: polvad@laspace.ru

А. В. ЧЕЧКИН
Московский государственный университет им. М. В. Ломоносова
e-mail: a.chechkin@mail.ru

УДК 519.95

Ключевые слова: радикал, сложная система, безопасность, конфликт, среда.

Аннотация

В статье рассматривается методический пример решения задачи при помощи среды радикалов и обсуждаются технологические проблемы решения задач жизненного цикла сложной системы.

Abstract

M. V. Pirogov, A. V. Chechkin, Technology of task-solving in a normalized radical media, Fundamentalnaya i prikladnaya matematika, vol. 15 (2009), no. 3, pp. 205—223.

In this article, a methodological example of task solving with the help of a radical media is considered and technological problems of solving complex system operational lifetime tasks are discussed.

1. Визуализация среды радикалов

Среда радикалов сложной системы предназначена обеспечить информационно-системную безопасность функционирования такой системы [1]. Человеку должна быть предоставлена возможность визуального контроля и интерактивного взаимодействия со средой радикалов. Таким образом, необходима разработка визуализации — основы интерфейса среды радикалов. Для визуализации (с использованием компьютерной графики) процессов навигации и выделения звеньев схем будем использовать отображение среды радикалов на плоскость [1]. При этом на экране будет отображаться не вся среда радикалов, а только её некоторая часть, которая определяет так называемую *точку обозрения* процесса решения задачи. Рассмотрим особенности безопасного решения задач с помощью среды радикалов на методическом примере.

Фундаментальная и прикладная математика, 2009, том 15, № 3, с. 205—223.

Пример. Определим точку обозрения среды радикалов с помощью контейнера cViewA0, в котором фиксируются уникумы, контейнеры, переменные и запросы, выделенные нами для наблюдения [1]. Введём на плоскости декартову систему координат, оси которой направим вправо и вниз. Звену FL (FirstLink) соответствует начало координат. Уникумы, контейнеры, переменные и запросы будем фиксировать на вертикальной оси. Связи (контейнеры) будем определять с помощью вертикально направленных векторов с указанием соответствующих направлений $(0, 1, \ldots)$ в контейнерах. Рассмотрение среды радикалов будем осуществлять поэтапно.

Этап 0. Пусть в состав сложной системы входят составляющие-уникумы u1, u2, u3, u4. Для этих уникумов имеется координатная система контейнеров (рис. 1). Контейнер c[i][0] R (c[i][0] CanPutOnRegulary) говорит о том,

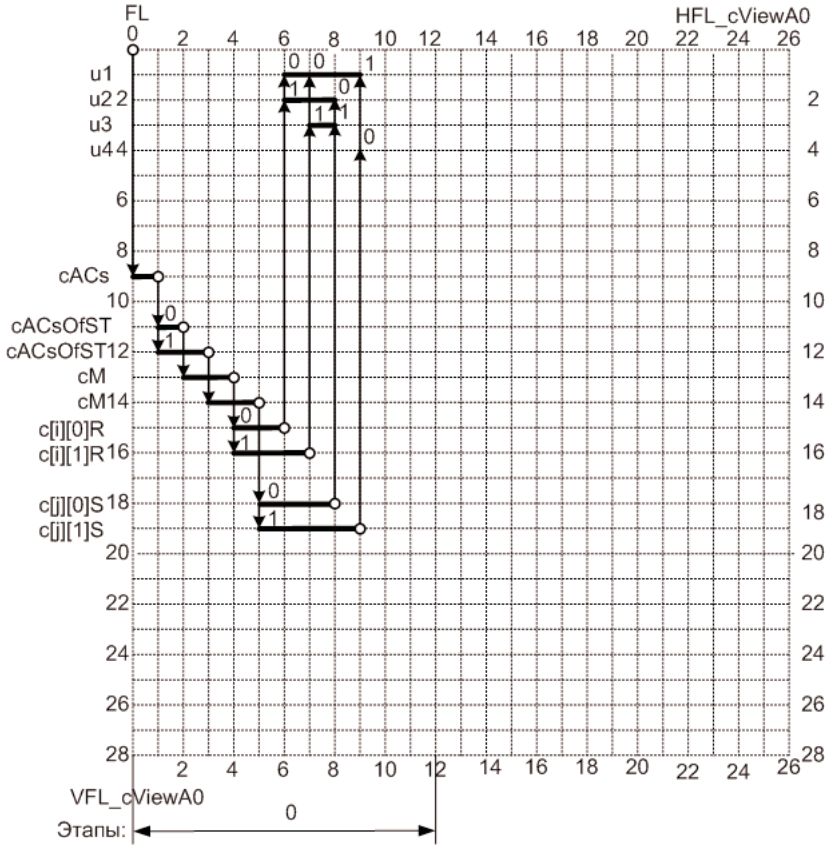


Рис. 1. Пример координатизации предметной области и визуализации среды радикалов

что u1 может включить u2 штатно, контейнер c[i][1] R — о том, что u1 может включить штатно u3. Некоторые составляющие являются дублёрами других составляющих: u2 дублирует u3, а u4 дублирует u1. Такие связи будем представлять контейнерами вида c[j][0] S и c[j][1] S (cSubstituteOfUnit). Для доступа к контейнерам cR и cS используется контейнер всех контейнеров cACs (cAllContainers), а также контейнеры однотипных контейнеров cACsOfST (cAllContainersOfSmthType), в каждый из которых вложен контейнер вида cM (cMany) — множество.

Пусть имеется следующее ультраоснащение среды радикалов (рис. 2). В среду включён ультраконтейнер U2CE (Ultra2C_CanPutOnExtremely), объединяющий переменные Var1, Var2 и Var3, которые могут быть связаны с уникумами-составляющими. Согласно ультраконтейнеру U2CE составляющая Var1

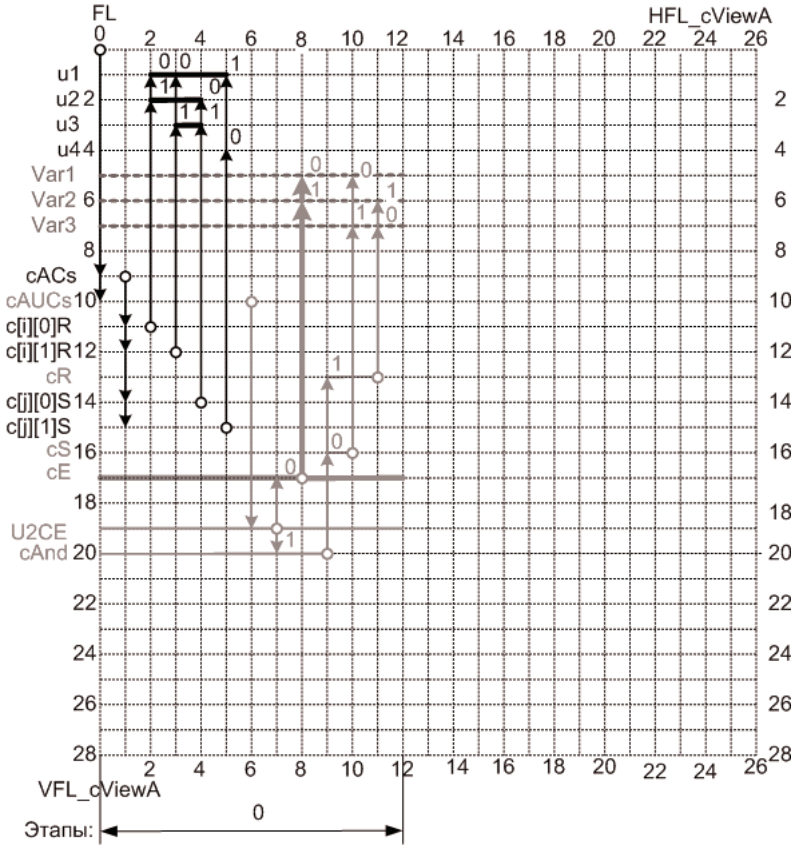


Рис. 2. Пример ультраоснащения среды радикалов

может включить аварийно составляющую Var2 (об этом говорит контейнер заключения cE (cCanPutOnExtremely)), если Var1 — дублёр Var3 (что описывается контейнером cS (cSubstituteOfUnit)) и Var3 может включить штатно Var2 (контейнер cR (cCanPutOnRegulary)).

Контейнеры-посылки cS и cR объединены контейнером cAnd, доступным в ультраконтейнере U2CE, который, в свою очередь, вложен в контейнер всех ультраконтейнеров cAUCs (cAllUltraContainers).

Этап 1. В среду радикалов поступает исходный запрос ?Q[0]: требуется найти составляющие Var4, которые могут быть включены аварийно (контейнер cE — целевой контейнер запроса) составляющей u4. Исходный запрос погружается в радикал-активатор (cActivator), который и будет искать ответы. Контейнер cActivator объединяет четыре направления. Направление 0 активатора ведёт к начальному звену схемы, на которой ищется ответ на запрос:

$$\text{cActivator} \to \text{d}[0] \to \text{FL}.$$

Направление 1 активатора ведёт к контейнеру исходного запроса cQ, объединяющему исходный запрос и ответы на него (пока ответ — пустое звено «;»). По направлению 2 активатора приходим к контейнеру активированных запросов cAQs (cActiveQuestions). Это исходный запрос и, возможно, промежуточные запросы, генерируемые активатором при обработке начального запроса. Каждый активированный запрос помещается в контейнер активированного запроса cAQ, объединяющий три направления.

По направлению 0 контейнера cAQ доступен активированный запрос. По направлению 1 контейнера cAQ доступен активированный контейнер среды радикалов, который используется при поиске ответа на запрос (пока это пустое звено). По направлению 2 контейнера cAQ доступен ответ на активированный запрос (пока это также пустое звено).

Вернемся к направлениям, непосредственно доступным в контейнере cActivator. Направление 3 активатора ведёт к контейнеру cNowAQ (cNowActiveQuestion) — это контейнер запроса, обрабатываемого активатором в текущий момент времени. В контейнере cNowAQ имеются следующие направления: направление 0, ведущее к запросу; направление 1, ведущее к активированному контейнеру среды радикалов, который используется при поиске ответа на запрос; направление 2, ведущее к ответу на запрос.

Этап 2. Активатор ищет в среде радикалов контейнер того же вида, что и целевой контейнер cE исходного запроса ?Q[0]. Поиск облегчён координатной системой контейнеров. В результате поиска найден контейнер U2CE, в котором по направлению d[0] доступен контейнер заключения cE. Контейнер cNowAQ преобразуется (преобразование a1): теперь имеем

$$\text{cNowAQ} \to \text{d}[1] \to \text{U2CE};$$

(рис. 3).

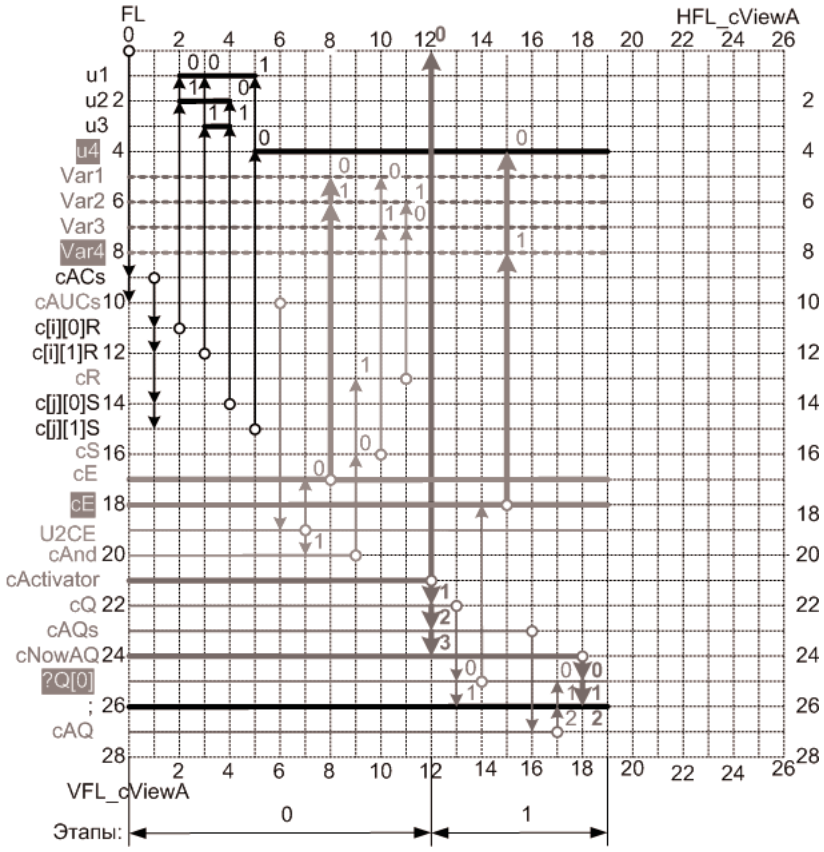


Рис. 3. Пример настройки активатора по запросу

Этап 3. Контейнер активированного запроса cAQ преобразуется соответственно (преобразование b1): теперь имеем

$$\mathrm{cAQ} \to \mathrm{d}[1] \to \mathrm{U2CE};.$$

Активатор работает с контейнерами cE из запроса и из ультраконтейнера. В результате переменная Var1 из ультраконтейнера связывается с уникумом u4, а переменная Var4 из запроса — с переменной Var2 из ультраконтейнера.

Контейнер cNowAQ преобразуется (преобразование a2): теперь

$$\mathrm{cNowAQ} \to \mathrm{d}[2] \to \ldots \to \mathrm{cVars};.$$

Контейнер cVars объединяет имеющиеся переменные в их состоянии на текущий момент времени:

$$\mathrm{cVars} \to \{\mathrm{d}[0] \to \mathrm{Var1} = \mathrm{u4};\ \mathrm{d}[1] \to \mathrm{Var2} = \mathrm{Var4} =;$$
$$\mathrm{d}[2] \to \mathrm{Var3} =;\ \mathrm{d}[3] \to \mathrm{Var4} = \mathrm{Var2} =; \}.$$

Контейнер активированного запроса вновь преобразуется соответственно (преобразование b2):

$$\mathrm{cAQ} \to \mathrm{d}[2] \to \mathrm{cVars};$$

(рис. 4).

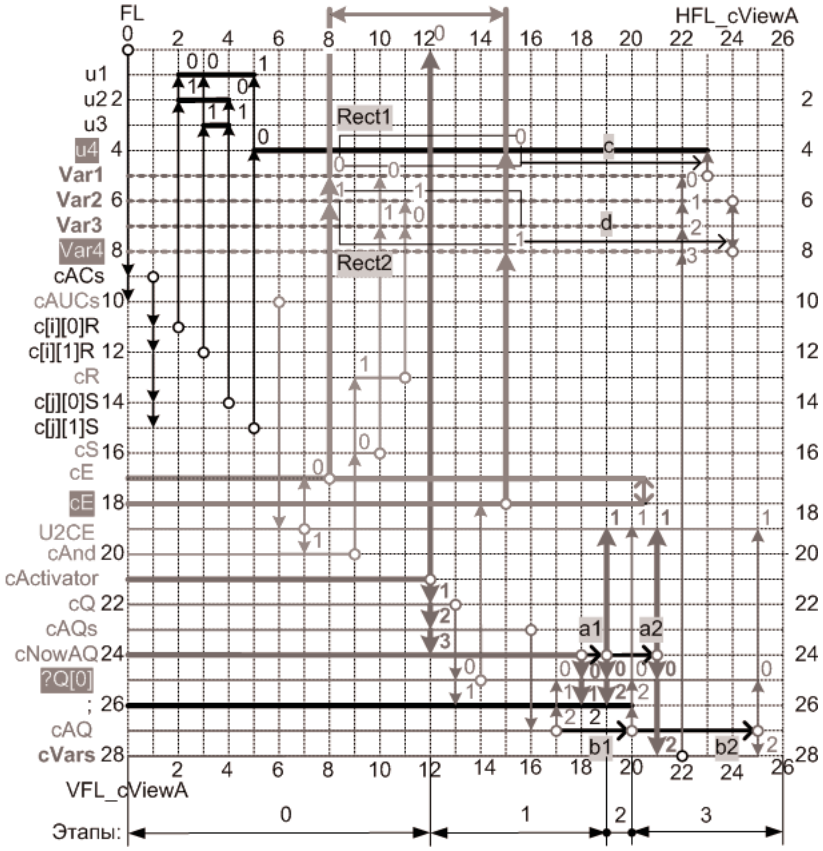


Рис. 4. Пример активации среды радикалов с целью поиска ответа

На этом этап 3 работы активатора завершается, но поиск ответа будет продолжен. Мы не будем подробно рассматривать дальнейшие этапы работы активатора, поскольку они во многом аналогичны этапам 1, 2 и 3, реализующим его базовый цикл. Приведём лишь краткие замечания.

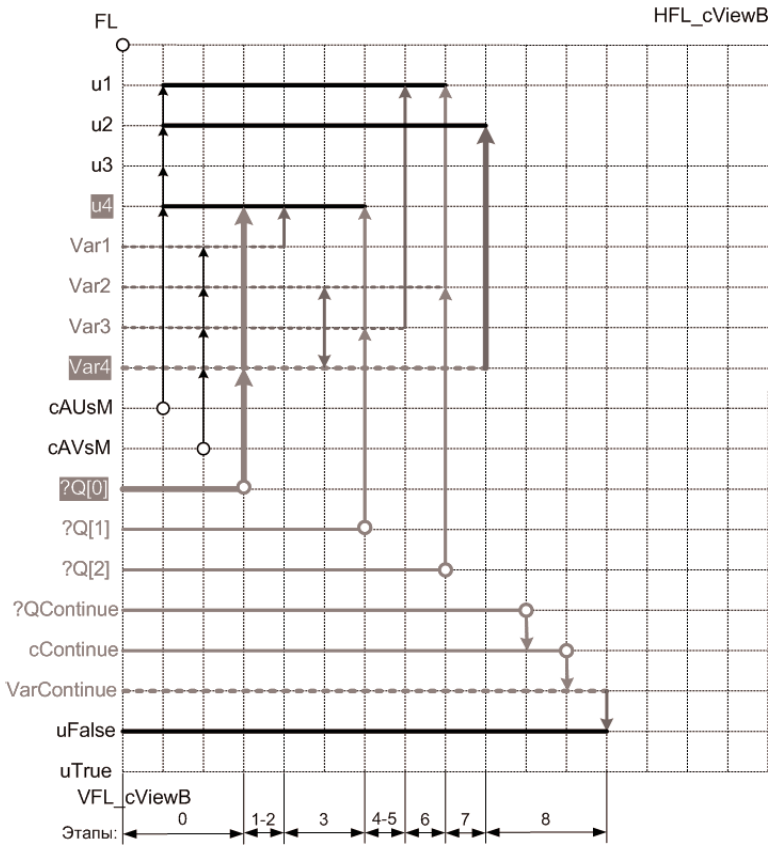


Рис. 5. Пример связывания переменных и уникумов при получении ответа на запрос. Получение первого ответа и отказ от поиска других ответов

Этапы 4—8 (рис. 5). Сменим точку обозрения cViewA среды радикалов и перейдём к представлению cViewB, которое содержит уникумы u1, u2, u3, u4 — составляющие сложной системы; переменные Var1, Var2, Var3, Var4; контейнеры cAUsM (cAllUnitsMany) и cAVsM (cAllVariablesMany), объединяющие соответственно используемые уникумы и переменные; исходный запрос ?Q[0], а также промежуточные запросы ?Q[1], ?Q[2], сгенерированные активатором при обработке исходного запроса. Кроме того, представление cViewB содержит схемы ?QContinue, cContinue, VarContinue, uFalse и uTrue, с помощью которых либо прерывается поиск ответов на запрос, либо инициируется его продолжение.

На этапе 8 мы будем иметь Var2 = Var4 = u2, т. е. получен первый ответ на исходный запрос: составляющей u4 может быть включена аварийно составляющая u2. Далее активатор генерирует специальный запрос ?QContinue

о необходимости продолжения поиска других ответов на исходный запрос. Запрос ?QContinue обращается к человеку, работающему в среде радикалов (либо к схеме более высокого уровня, управляющей работой данного активатора). В запрос ?QContinue вложен контейнер cContinue, в котором доступна переменная VarContinue. Эту переменную человек может связать либо с уникумом uFalse, либо с уникумом uTrue. В первом случае поиск ответов будет прерван, во втором продолжен.

Пусть действия человека привели к связыванию VarContinue = uFalse (см. рис. 5). В этом случае Var4 = u2 останется единственным ответом на исходный запрос. Дальнейшего поиска других ответов не будет.

Рассмотрим теперь случай, когда действия человека привели к связыванию VarContinue = uTrue. В этом случае поиск других ответов на исходный запрос будет продолжен. Рассмотрим соответствующие этапы работы активатора.

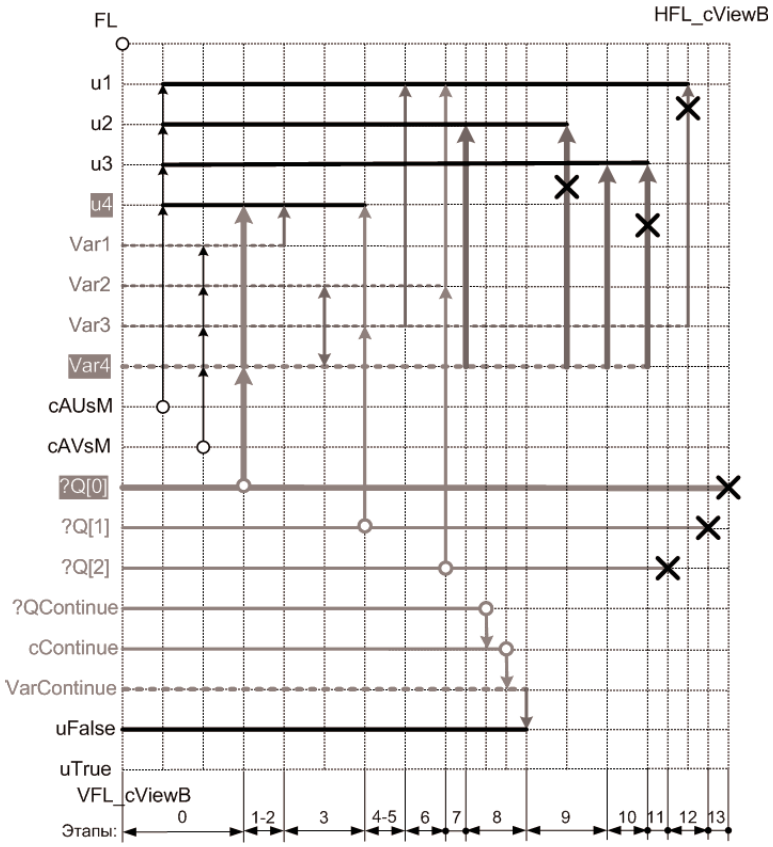


Рис. 6. Пример завершения поиска ответов на запрос

Этапы 9—13 (рис. 6). Активатор продолжит поиск, и на этапе 10 мы получим второй ответ на исходный запрос: Var2 = Var4 = u3, т. е. составляющей u4 может быть включена аварийно составляющая u3.

И вновь активатор генерирует запрос ?QContinue о необходимости продолжения поиска других ответов на исходный запрос. (На рисунке соответствующие схемы не показаны.) Пусть действия человека привели к связыванию VarContinue = uTrue. Вновь будет продолжен поиск ответов, и на этапах 11—13 будет выяснено, что ответов на исходный запрос больше нет.

На этом решение методического примера заканчивается.

Замечание. Предполагается развивать интерфейс среды радикалов, используя преобразования векторов на плоскости: f1 — поворот; f2 — растяжение-сжатие; f3 — сдвиг и f4 — сложение для представления преобразований схем радикалов. Преобразования векторов базового цикла активатора, приведшие к связыванию на этапе 3 переменной Var1 с уникумом u4, представлены на рисунках 7, 8, 9.

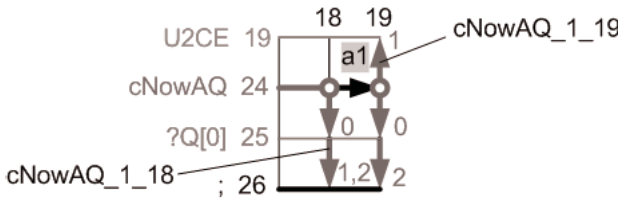


Рис. 7. Преобразование a1. Поворот: f1(cNowAQ_1_18). Растяжение-сжатие: f2(f1(cNowAQ_1_18)). Сдвиг: $\mathrm{f3}\Big(\mathrm{f2}\big(\mathrm{f1}(\mathrm{cNowAQ\_1\_18})\big)\Big) = \mathrm{cNowAQ\_1\_19}$

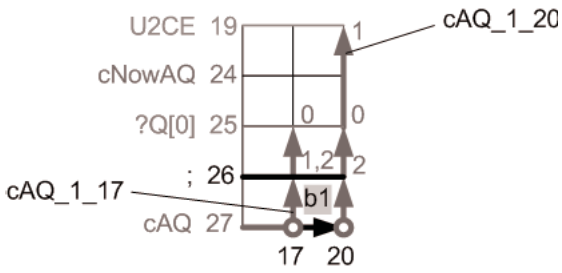


Рис. 8. Преобразование b1. Растяжение-сжатие: f2(cAQ_1_17). Сдвиг: $\mathrm{f3}\big(\mathrm{f2}(\mathrm{cAQ\_1\_17})\big) = \mathrm{cAQ\_1\_20}$

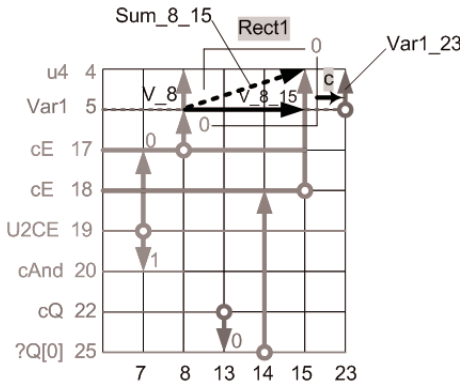


Рис. 9. Преобразование c. Сложение: $\mathrm{f4}(\mathrm{V\_8}, \mathrm{V\_8\_15}) = \mathrm{Sum\_8\_15}$. Поворот: $\mathrm{f1}(\mathrm{Sum\_8\_15})$. Растяжение-сжатие: $\mathrm{f2}\big(\mathrm{f1}(\mathrm{Sum\_8\_15})\big)$. Сдвиг: $\mathrm{f3}\Big(\mathrm{f2}\big(\mathrm{f1}(\mathrm{Sum\_8\_15})\big)\Big) = \mathrm{Var1\_23}$

2. Технология решения задач в активирующей подсистеме

Технология решения задач в среде радикалов требует создания специализированных методов, предназначенных для решения тех или иных классов задач. Это приводит к проблеме оценки схем среды радикалов и разработке их классификации. Необходимо иметь быстрый доступ к штатным задачам и, соответственно, к штатным методам решения задач. Кроме того, требуется создание «базового механизма» информационно-системно-безопасного решения задач жизненного цикла сложной системы, т. е. механизма учёта и устранения конфликтов в среде радикалов.

Были разработаны схемы радикалов (мы не будем рассматривать их подробно), с помощью которых представляется множество всех задач среды радикалов. При этом учитывается каждая задача (подзадача) среды радикалов, для которой доступны запросы, начальная схема, ресурсы, ответы, отчётные схемы, подзадачи (иерархия подзадач), методы решения.

Пусть uSTsk (uSmthTask) — уникум-задача. Для выделения всех контейнеров уникумов-задач будем использовать контейнеры вида cACsOfTsk (cAllContainersOfTask):

$$\begin{aligned}&\mathrm{FL} \to \ldots \to \mathrm{uSTsk} \leftarrow \mathrm{cACsOfTsk} \to\\ &\quad \{\mathrm{dQs} \to \ldots;\ \mathrm{dBgnSh} \to \ldots;\ \mathrm{dRsrs} \to \ldots;\ \mathrm{dAnss} \to \ldots;\\ &\quad \mathrm{dEndShs} \to \ldots;\ \mathrm{dSbTsks} \to \ldots;\ \mathrm{dMtds} \to \ldots;\ \ldots\}.\end{aligned}$$

Здесь контейнеры вида cACsOfTsk объединяют следующие схемы, доступные по соответствующим направлениям: dQs (dQuestions) — запросы задачи; dBgnSh

(dBeginScheme) — начальная схема задачи; dRsrs (dResources) — ресурсы задачи; dAnss (dAnswers) — ответы; dEndShs (dEndSchemes) — конечные схемы задачи; dSbTsks (dSubTasks) — подзадачи задачи; dMtds (dMethods) — методы, используемые при решении задачи.

Проблема оценки схем и разработки их классификации потребовала построения схем (их мы также не будем рассматривать подробно), с помощью которых представляются оценки схем и методы их получения, классы схем, ультраконтейнеры классификации. В результате применения методов решения задач жизненного цикла сложной системы схемы радикалов преобразуются. Рассмотрим некоторые классы преобразований схем. Будем рассматривать множества задач uSTsk для ультраоснащённых схем.

Пусть преобразования некоторой схемы uSS, осуществляемые в результате решения задачи uSTsk некоторого класса таковы, что конечные схемы dEndShs отличаются от начальной схемы dBgnSh только значениями временных индексов. Такие преобразования и схемы мы будем называть *тождественными* относительно рассматриваемого класса задач и методов их решения. Преобразования

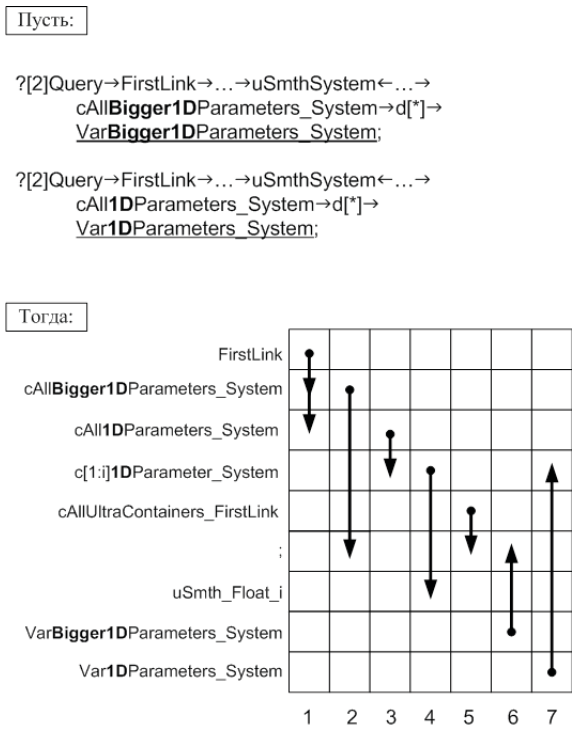


Рис. 10. Теорема о неразвёртываемых схемах

и схемы, не удовлетворяющие этому условию, мы будем называть *нетождественными*.

Если для некоторой схемы в результате решения задач некоторого класса мы получаем только тождественные преобразования, то схема называется *неразвёртываемой* относительно рассматриваемого класса задач и методов их решения. Если же преобразования не являются тождественными, а схемы dEndShs можно представить как результат объединения тождественно преобразованных схем dBgnSh и схем, полученных в результате активации ультраконтейнеров исходных схем dBgnSh, то схема называется *развёртываемой* относительно рассматриваемого класса задач и методов их решения.

В технологии решения задач важное практическое значение имеет выделение и исследование классов схем. Примеры теорем о неразвёртываемых и развёртываемых схемах с использованием типового геометрического отображения приведены на рисунках 10 и 11.

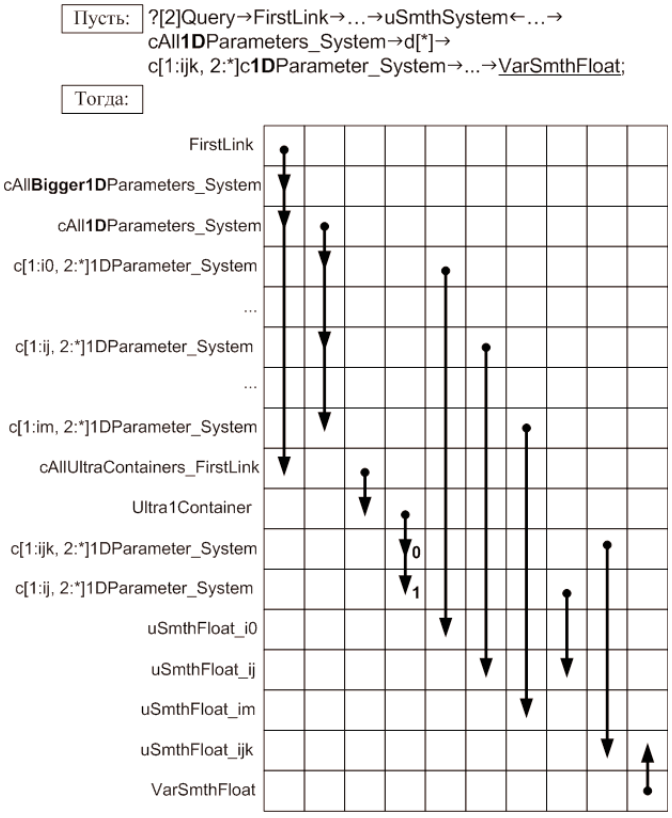
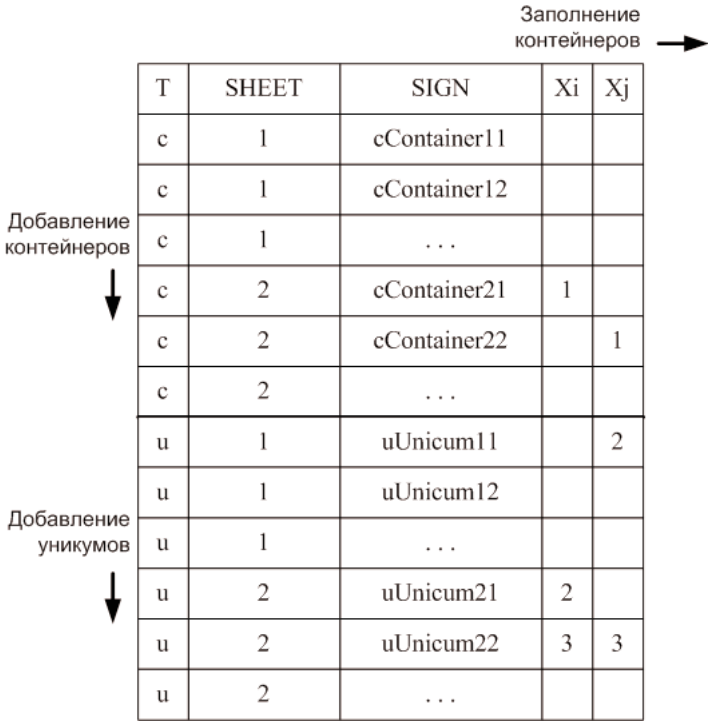


Рис. 11. Теорема о развёртываемых схемах

3. Конфликты в среде радикалов и их разрешение

Для информационно-системно-безопасного решения задач жизненного цикла сложной системы требуется создание «базового механизма». Его применение должно обеспечить информационно-безопасный рост среды радикалов (рис. 12).



T	SHEET	SIGN	Xi	Xj
c	1	cContainer11		
c	1	cContainer12		
c	1	. . .		
c	2	cContainer21	1	
c	2	cContainer22		1
c	2	. . .		
u	1	uUnicum11		2
u	1	uUnicum12		
u	1	. . .		
u	2	uUnicum21	2	
u	2	uUnicum22	3	3
u	2	. . .		

Рис. 12. Рост среды радикалов

Рисунок основан на типовом геометрическом отображении среды радикалов, реализованном с помощью таблицы. Столбец T содержит символы, именующие основной тип радикала: c — контейнер, u — уникум. Столбец SHEET предназначен для структурирования множества имеющихся контейнеров и уникумов. В столбце SIGN представлены обозначения контейнеров и уникумов. В столбцах X∗ представлена реализация контейнеров cContainer∗. Символы «2», «3»,. . . в ячейках столбцов X∗ именуют направления в соответствующих контейнерах. Направления роста среды радикалов показаны на рисунке стрелками.

В течение жизненного цикла сложной системы контейнеры среды радикалов могут преобразовываться, что может привести к конфликтам между ними. Контейнеры могут быть изменены из-за непосредственных внешних воздействий

окружающей среды, а также в результате целенаправленных управляющих действий. К преобразованиям контейнеров приводит взаимодействия составляющих сложной системы. Контейнеры могут «стареть», расходуя ресурсы на обеспечение жизненного цикла, вплоть до разрушения. Фиксация (учёт) и устранение конфликта (уход от конфликта) — это главное в решении проблемы информационно-системно-безопасного функционирования сложной системы.

Конфликты могут возникать при решении практически любых задач жизненного цикла. Допустим, требуется построить уникум uGoal (с некоторой характеризующей его системой контейнеров и ультраконтейнеров). Это может быть подсистема или составляющая сложной системы, некоторое управляющее действие и т. п. При решении задачи должны использоваться библиотечные схемы, решаться штатные задачи, подзадачи и задачи связывания целевых схем с экземплярами библиотечных схем, заполнения соответствующих контейнеров. Так строится (реализуется) уникум uGoal. В итоге целевой уникум реализуется иерархией составляющих-уникумов, характеризуемых своими контейнерам и ультраконтейнерами. В процессе построения такой иерархии в неё добавляются все новые и новые уникумы, контейнеры которых могут, вообще говоря, конфликтовать с контейнерами уже имеющихся уникумов, и эти конфликты необходимо разрешать.

В течение жизненного цикла схемы, представляющие сложную систему, а также окружающую её среду, преобразуются. Это может привести к конфликтам. На этапе эксплуатации, например, может быть зарегистрирован уникум, соответствующий некоторому нештатному внешнему воздействию, способному вызвать конфликт контейнеров, которого необходимо избежать, преобразуя доступные для этого схемы среды радикалов.

Введём некоторые понятия, которые будем использовать при рассмотрении конфликтов в среде радикалов. Предположим, что некоторая схема среды радикалов находится в состоянии, которое может быть определено как конфликтное, причём принадлежащее некоторому определённому классу. Одна и та же схема одновременно может участвовать во множестве конфликтов различных классов. Для определения классов конфликтов будем использовать контейнеры вида cSmthConflictClass, входящие в контейнеры вида cClassOfScheme и имеющие вид

$$\begin{aligned}&\mathrm{cSmthConflictClass} \to \\ &\qquad \{\mathrm{dScheme} \to \ldots;\ \mathrm{dUltraContainersOfClass} \to \ldots;\ \mathrm{dEstimate} \to \ldots;\}.\end{aligned}$$

Рассматриваемый контейнер объединяет следующие схемы: dScheme — оцениваемая (классифицируемая) схема; dUltraContainersOfClass — ультраконтейнеры, используемые для классификации схемы; dEstimate — схема-оценка (соответствующая цепочка может оканчиваться звеном uFalse или звеном uTrue). Дадим теперь определение конфликтной схемы.

Схему среды радикалов будем называть *конфликтной* для заданной системы ультраконтейнеров, если в результате решения задачи классификации этой

схемы получена оценка вида

$$\text{cSmthConflictClass} \to \text{dEstimate} \to \ldots \to \text{uTrue;}.$$

Класс конфликта определяется контейнером cSmthConflictClass, а соответствующая система ультраконтейнеров должна быть доступна в контейнере cSmthConflictClass по направлению dUltraContainersOfClass.

Конфликты в языке радикалов, фиксируемые с использованием схем вида cSmthConflictClass, соответствуют конфликтам в проблемной области.

Приведём определение, относящееся к «расположению» контейнеров друг относительно друга. Будем говорить, что контейнер c0 *охватывает* контейнер c1, если c0 и c1 однотипны и все уникумы контейнера c1 являются уникумами контейнера 0, но не наоборот.

В качестве примера можно привести схему, содержащую два контейнера c0 и c1, где c0 охватывает c1. Пусть преобразования контейнеров приводят к появлению уникумов, принадлежащих преобразованному контейнеру c1, но уже не принадлежащих преобразованному контейнеру c0. Пусть имеется система ультраконтейнеров, с использованием которой может быть получена оценка вида

$$\text{cSmthConflictClass} \to \text{dEstimate} \to \ldots \to \text{uTrue;}.$$

Таким образом, рассматриваемая схема становится конфликтной из-за конфликта контейнеров 0 и 1, вызванного их преобразованиями.

Мы видим, что преобразования контейнеров среды радикалов, описываемые схемами вида

$$\text{cChangeScheme} \to \{\text{dBeforeScheme} \to \ldots;\ \text{dAfterScheme} \to \ldots;\ \ldots\},$$

могут привести к конфликтам, принадлежащим некоторым классам. Именно в смысле преобразований и конфликтов контейнеров мы будем говорить о преобразованиях и конфликтах уникумов среды радикалов.

Для решения проблемы конфликтов и обеспечения бесконфликтности жизненного цикла сложной системы необходимо иметь возможность достаточно быстро определять все преобразованные контейнеры. Целесообразно использовать по возможности более простые, содержащие меньшее число звеньев контейнеры для приближения (более сложных) контейнеров. (Преобразования сложных контейнеров заменять преобразованиями простых контейнеров.) В качестве примеров можно привести контейнеры типа «отрезок на прямой», «прямоугольник на плоскости», «прямоугольник в n-мерном пространстве». Системы таких контейнеров используются для построения приближений различных множеств точек на прямой, в двумерном и многомерном пространствах.

Рассмотрим контейнер «отрезок». Такие контейнеры мы будем рассматривать как «проекции» соответствующего уникума на контейнеры «прямые». Левые и правые границы таких «отрезков» описываются контейнерами одномерных параметров вида c1DP. Будем использовать разложения контейнеров среды радикалов на контейнеры одномерных параметров. Рассмотрим для сравнения все

возможные пары контейнеров из множества всех контейнеров среды радикалов. Для этого будем использовать схемы вида

$$\mathrm{cSmthCPair} \to \{\mathrm{dFirstC} \to \ldots;\ \mathrm{dSecondC} \to \ldots;\}.$$

Для каждой пары рассматриваются все возможные преобразования её контейнеров. Пусть имеется разложение контейнеров некоторой пары на контейнеры одномерных параметров c1DP. В общем случае при преобразованиях этой пары множество всех таких контейнеров можно разбить на два подмножества: непреобразуемых (тождественно преобразуемых) и преобразуемых контейнеров c1DP. Будем предполагать, что для каждой рассматриваемой пары контейнеров задача uAllDistancesTsk (задача определения всех расстояний между контейнерами пары) имеет единственное решение. Таким образом, предполагается, что в среде радикалов имеются ультраконтейнеры, используя которые можно заполнить все контейнеры вида cDistanceOfCPair, входящие в контейнер cAllDistancesOfCPair:

$$\mathrm{cDistanceOfCPair} \to \{\mathrm{dCPair} \to \mathrm{cSmthCPair} \to \ldots;\ \mathrm{dFloatVar} \to \ldots \to \mathrm{uSmthFloat};\}.$$

Цепочка

$$\mathrm{dFloatVar} \to \ldots \to \mathrm{uSmthFloat};$$

оканчивается уникумом, являющимся конечной неотрицательной десятичной дробью. Для оценок могут использоваться также целые неотрицательные числа. Предположим, что для каждой рассматриваемой пары контейнеров задача определения всех её конфликтов uAllConflictsTsk имеет тоже единственное решение. В результате решения этой задачи заполняются контейнеры вида cSmthConflictClass. Пусть

$$\mathrm{cSmthConflictClass} \to \mathrm{dUltraContainersOfClass} \to \ldots;.$$

При этом цепочки-оценки

$$\mathrm{cSmthConflictClass} \to \mathrm{dEstimate} \to \ldots;$$

завершаются звеньями uTrue тогда и только тогда, когда уникумы uSmthFloat из контейнеров cDistanceOfPair меньше (контейнер cSmaller) некоторых наперёд заданных уникумов uSmthConflictFloat. Зафиксируем тройку контейнеров cSmthCPair, cDistanceOfCPair и cSmthConflictClass (контейнер cSmthConflictClass фиксирует конфликт пары в смысле контейнера cDistanceOfCPair). Пусть к контейнерам пары cSmthCPair применяется некоторое преобразование, в результате чего расстояние

$$\mathrm{cDistanceOfCPair} \to \ldots \to \mathrm{uSmthFloat};$$

может либо уменьшиться, либо остаться неизменным, либо увеличиться. Определим, исходя из этого, некоторые классы преобразований пары контейнеров.

Преобразование пары контейнеров будем называть преобразованием типа «*плюс-конфликт*», если в результате его применения расстояние между контейнерами уменьшилось (и, таким образом, пара приблизилась к конфликту). Преобразование, в результате которого расстояние между контейнерами не изменилось (увеличилось), будем называть преобразованием типа «*нуль-конфликт*» (типа «*минус-конфликт*»). Будем рассматривать преобразования на промежутках времени, которые либо меньше, либо равны продолжительности жизненного цикла сложной системы.

В качестве примера преобразований типа «плюс-конфликт» приведём преобразования пары контейнеров-«отрезков» c0 и c1 на прямой. Пусть контейнер c0 охватывает контейнер c1. Пусть согласно ультраконтейнерам

$$\mathrm{cSmthConflictClass} \to \mathrm{dUltraContainersOfClass} \to \ldots;$$

конфликт пары наступит, если расстояния между правыми (левыми) граничными звеньями контейнеров-«отрезков» станут равными нулю. Пусть преобразование пары таково, что левые и правые граничные звенья контейнеров-«отрезков» сблизились (в смысле контейнера cDistanceOfCPair). Тогда рассматриваемое преобразование является преобразованием типа «плюс-конфликт».

Рассмотрим некоторое преобразование (последовательность преобразований) пары контейнеров cSmthCPair, в котором имеются преобразования всех классов: «плюс-конфликт», «нуль-конфликт» и «минус-конфликт». В результате такого преобразования (последовательности преобразований) конфликтное состояние пары, принадлежащее фиксированному классу cSmthConflictClass, может наступить, а может и не наступить. Выделим, основываясь на этих соображениях, ещё два класса преобразований пары контейнеров: безопасные и опасные.

Если преобразования пары контейнеров таковы, что цепочка

$$\mathrm{cSmthConflictClass} \to \mathrm{dEstimate} \to \ldots;$$

оканчивается звеном uFalse, то такие преобразования будем называть *безопасным* относительно класса конфликтов cSmthConflictClass. Если же в результате преобразований пара контейнеров становится конфликтной, то будем называть такие преобразования *опасными*. Отметим, что если пара контейнеров подвергается последовательным преобразованиям, содержащим только преобразования типа «минус-конфликт» и, может быть, типа «нуль-конфликт», то такие последовательности преобразований являются заведомо безопасными. Если последовательность преобразований пары контейнеров содержит только преобразования типа «плюс-конфликт», этого, вообще говоря, недостаточно для классификации её как опасной. Мы будем допускать представления (в пределах заданной точности) преобразований пары контейнеров последовательностями преобразований. Например, мы можем допускать представление преобразований пары контейнеров последовательностями преобразований, состоящими из безопасных, опасных и тождественных преобразований.

Если за промежуток времени, который либо меньше, либо равен продолжительности жизненного цикла сложной системы, конфликт пары контейнеров

cSmthCPair не наступил, то можно говорить о бесконфликтности жизненного цикла рассматриваемой пары контейнеров (на указанном промежутке времени и относительно определённого подмножества классов конфликтов).

Переходя от пары контейнеров к множеству всех контейнеров среды радикалов, используемых для представления проблемной области, будем говорить о бесконфликтности жизненного цикла сложной системы в смысле бесконфликтности жизненных циклов всех возможных пар контейнеров.

Выделим ещё два класса преобразований контейнеров, связанных с проблемой контроля (управляемости).

Контейнеры, преобразования которых (на некотором временном промежутке жизненного цикла сложной системы) доступны для управления средой радикалов (с помощью замены звеньев схем), мы будем называть *управляемыми*. В противном случае преобразования будем называть *неуправляемыми*. Эти же термины будем использовать применительно к преобразованиям контейнеров (т. е. применительно к контейнерам вида cChangeScheme).

Управляемые контейнеры можно использовать для ухода от конфликтов при решении задач жизненного цикла сложной системы. Среда радикалов автоматически стремится «развести» контейнеры, приближающиеся к конфликту. Понятно, что чем больше в среде радикалов контейнеров, чем больше разнообразие типов этих контейнеров, чем больше преобразуемых схем и, одновременно, чем меньше управляемых контейнеров и активаторов, тем труднее решать задачи обеспечения информационно-системной безопасности сложной системы.

Изложенные соображения могут послужить основой для построения системы количественных оценок схем радикалов и использования этих оценок для обеспечения бесконфликтности жизненного цикла сложной системы.

Перейдём к проблеме выявления конфликтов. Пусть необходимо решить некоторую задачу анализа среды радикалов и прогнозирования возможных конфликтов с целью ухода от них. Пусть для приближения контейнеров среды радикалов используется система более простых контейнеров. Предположим, что прогнозируется конфликт двух таких приближающих контейнеров. Пусть используется система ультраконтейнеров, согласно которой конфликт имеет место, если контейнеры имеют один или более общих уникумов в один и тот же момент времени (на одном и том же временно́м промежутке).

Конфликт между приближающими контейнерами ещё не означает конфликта между приближаемыми контейнерами. Необходимо использовать более подробные разбиения области возможного конфликта, учесть гарантированный «зазор», который необходимо обеспечить между контейнерами, а также имеющиеся ресурсы. Контейнеры могут владеть одними и теми же уникумами, но в разное время, и конфликта (в смысле описанной системы ультраконтейнеров) не будет.

Преобразования контейнеров, переход уникумов от одного контейнера к другому могут происходить быстро, процессы могут иметь колебательный характер. Поэтому среда радикалов должна иметь достаточную «разрешающую способ-

ность», для того чтобы «видеть» опасные преобразования, прогнозировать конфликты и уходить от них, используя управляемые контейнеры.

4. Заключение

Мы провели краткое описание технологии информационно-системно-безопасного решения задач жизненного цикла сложной системы на основе среды радикалов. Перечислим некоторые перспективные задачи, решение которых необходимо для развития такой технологии.

- Развитие средств визуализации среды радикалов. Разработка типовых геометрических отображений различных видов, удобных при решении тех или иных классов задач. Средства визуализации должны быть ориентированы на применение современных программно-технических средств, на использование динамики, цвета и звука. «Геометризация» среды радикалов — использование геометрических объектов (например, векторов на плоскости) для представления схем радикалов и их преобразований.
- Разработка дополнительных принципов нормализации среды радикалов.
- Выделение и исследование классов схем радикалов, характерных для проблемной области сложной системы.
- Разработка схем, обеспечивающих представление конфликтов в среде радикалов и уход от них.
- Разработка системы количественных оценок среды радикалов и использование этих оценок для ухода от конфликтов.
- Постановка и исследование штатных задач о конфликтах.
- Построение развивающейся системы информационно-системно-безопасных карт — идеальных объектов, выраженных в схемах радикалов и подлежащих встраиванию в сложную систему. Такие карты должны реализовываться с помощью математического, программного, информационного, технического, методического, лингвистического, организационного, правового, эргономического и метрологического обеспечений сложных систем.

Литература

[1] Чечкин А. В., Пирогов М. В. Интеллектуализация сложной системы как средство обеспечения её информационно-системной безопасности // Фундамент. и прикл. мат. — 2009. — Т. 15, вып. 3. — С. 225—239.