УДК 681.3.06:519.876

Е. П. Моргунов

Система поддержки принятия решений при исследовании эффективности сложных систем: принципы разработки, требования и архитектура

Сформулированы принципы разработки и требования к системе поддержки принятия решений при исследовании эффективности сложных систем. Предложена архитектура такой системы и обоснован выбор программных средств ее реализации.

Управление сложными системами предполагает, в том числе, и управление их эффективностью. «Эффективность – это наиболее общее, определяющее свойство любой целенаправленной деятельности, которое … объективно выражается степенью достижения цели с учетом затрат ресурсов и времени» [5, с. 59]. В работе [6, с. 71–72] обобщенный показатель эффективности большой системы в наиболее общей форме предлагается строить как некоторую функцию или функционал

$$W = \Phi(Y_к, Y_н, U_к, U_н),$$

где $Y_к$ – возможный или фактически достигнутый полезный эффект (конечный результат) функционирования и развития системы; $Y_н$ – целевой полезный эффект (необходимый конечный результат) функционирования и развития системы; $U_к$ – возможные или фактические затраты количества труда (живого и прошлого) для получения $Y_к$; $U_н$ – минимальные необходимые затраты количества труда (живого и прошлого) для получения $Y_н$. Если ($Y_к$, $U_к$) рассматриваются как возможные величины, то речь идет о прогнозировании эффективности, а в случае, когда ($Y_к$, $U_к$) фактически полученные, то показатель W будет отражать фактически достигнутую эффективность за некоторый период функционирования системы.

Применяемые методы оценки эффективности систем отличаются большим разнообразием. Среди них есть как детерминированные методы, так и стохастические. В технических системах часто применяются стохастические методы. Например, в работе [6] предлагается обобщенный показатель эффективности:

$$W = (W_н, W_п, W_э),$$

где $W_н$, $W_п$, $W_э$ – комплексные показатели, соответственно, целевой надежности системы, целевой производительности системы и целевой экономичности системы. Определять, например, показатель эффективности $W_э$ предлагается по формуле:

$$W_э = \int_0^\infty u dF_к(u),$$

где u – переменная, выражающая возможные значения расхода ресурсов $U_к$ на получение конечного результата $Y_к$; $F_к(u)$ – функция распределения случайной величины $U_к$. Задача получения аналитических выражений для подобных функций решается на основе аналитических и статистических подходов. Однако выбор вида функциональной зависимости является неформализуемым шагом.

В экономических исследованиях часто используются производственные функции (см. например, [3]), характеризующие максимально возможный объем выпуска y продукта в зависимости от используемого объема ресурсов $X = (x_1, ..., x_m)$:

$$y = f(X).$$

Такая функция позволяет описывать только однопродуктовые технологии. В литературе предложены различные виды производственных функций (см. [3]).

Одним из наиболее распространенных методов оценки эффективности систем является метод, известный в России под названием «анализ среды функционирования» (АСФ) [1]. Он был разработан в 1978 г. в США [7] и с тех пор завоевал широкую популярность на Западе. Там он известен как Data Envelopment Analysis (DEA). Метод АСФ (DEA) применяется для оценки эффективности функционирования однородных объектов в таких областях, как экономика, социальная сфера, административное управление и даже спорт.

Метод АСФ (DEA) основан на построении так называемой границы эффективности в многомерном пространстве входных и выходных переменных, описывающих исследуемые объекты. Эта граница строится по реальным данным и представляет собой, по сути, оценку производственной функции для случая, когда выход является векторным. Степень эффективности зависит от расстояния между объектом и границей эффективности.

В качестве иллюстрации приведем описание одной из моделей метода. Пусть требуется определить показатель эффективности каждого из n объектов (это могут быть предприятия, организации, университеты, банки и т. д.). Для описания каждого объекта o_j, $j = \overline{1, n}$, служит пара векторов (x_j, y_j). При этом вектор $x_j = (x_{j1}, ..., x_{ji}, ..., x_{jm})^T$ содержит входные показатели (входы) для объекта o_j, а вектор $y_j = (y_{j1}, ..., y_{jr}, ..., y_{js})^T$ содержит выходные показатели (выходы) для объекта o_j. Тогда матрица $X = (x_j)$, имеющая размерность $m \times n$, содержит вектор-столбцы с входными данными для всех n объектов, а матрица $Y = (y_j)$, имеющая размерность $s \times n$, содержит вектор-столбцы с выходными данными для всех n объектов. Модель имеет такой вид [10, с. 43]:

$$\begin{aligned} &\min_{\theta,\lambda}(\theta), \\ &-y_j + Y\lambda \geq 0, \\ &\theta x_j - X\lambda \geq 0, \\ &\lambda \geq 0. \end{aligned} \qquad (1)$$

Скаляр θ и является мерой эффективности j-го объекта. Важно отметить, что $\theta \in (0; 1]$. Критерием эффективности объекта является условие $\theta = 1$. Объекты, имеющие такое значение показателя θ, считаются эффективными и находятся, как принято говорить, на *границе эффективности*. Аналогичная задача решается для каждого объекта, т. е. n раз.

Для объектов, имеющих $\theta < 1$, предлагаются рекомендуемые (целевые) значения показателей, достигнув которых, эти объекты также окажутся на границе эффективности. Определение целевых значений переменных для неэффективного объекта производится путем *проецирования* данного объекта на границу эффективности. Проецирование обеспечивается за счет присутствия в модели (1) коэффициента θ при векторе x_j и наличием ограничения $\lambda \geq 0$. Вектор констант $\lambda = (\lambda_1, ..., \lambda_j, ..., \lambda_n)^T$ позволяет сформировать неотрицательную линейную комбинацию объектов, которая и будет являться гипотетическим (и при этом – эффективным) целевым объектом для того реального объекта, который оказался неэффективным.

Граница эффективности в данном случае будет иметь вид выпуклого конуса в пространстве входных и выходных переменных R^{m+s}. Она определяется эффективными (крайними) точками. Модель (1) называется *ориентированной на вход*. Это объясняется тем, что коэффициент θ оказывает влияние на вектор входных переменных. Модель, *ориентированная на выход*, может быть построена аналогично [10, с. 58].

Еще одним граничным методом, но использующимся не столь часто, как метод АСФ (DEA), является метод оценки стохастической производственной границы – Stochastic Frontier Analysis (SFA). Его предложили D. J. Aigner и S. F. Chu [8, с. 184]. Отличительной чертой метода является вероятностный характер производственной границы, что позволяет учесть наличие случайных ошибок в значениях входных переменных. Однако метод позволяет работать только со скалярным выходом.

В реальных задачах зачастую исследуются сложные системы с большим числом подсистем и элементов (объектов). Для решения подобных задач необходимо программное обеспечение (ПО), реализующее те или иные методы исследования эффективности систем.

Большинство широко доступных программных продуктов, предназначенных для оценки эффективности систем, реализуют именно метод АСФ (DEA). Есть ПО, реализующее метод SFA [8]. Программных продуктов, реализующих другие методы оценки эффективности и спроектированных в расчете на широкое применение, нам не известно. Детальный обзор существующего зарубежного ПО, реализующего метод АСФ (DEA), представлен в работе [11]. Известны и две разработки российских специалистов: EffiVision (http://www.dea-21.ru) и KonSi-DEA (http://www.data-envelopment-analysis.ru). Основные инструменты, применяемые для разработки подобных программных продуктов, – это языки программирования Fortran, C/C++, Visual Basic.

Существующие программные продукты имеют целый ряд достоинств в том, что касается разнообразия реализованных моделей метода АСФ (DEA) и интерфейса пользователя. Однако им присущи и недостатки. В частности, эти программные продукты:

– не являются комплексными в плане набора реализованных методов: все они реализуют только какой-то один метод исследования эффективности;

– не являются комплексными и в плане реализации всех *стадий* исследования эффективности. Такими стадиями можно считать: оценку достигнутого уровня эффективности, объяснение достигнутого уровня эффективности, прогнозирование будущего уровня, выдачу рекомендаций по способам достижения требуемого уровня эффективности (в частности, за счет перераспределение ресурсов между подсистемами). Существующие программные продукты ограничиваются, в основном, оценкой текущего уровня эффективности и выдачей рекомендуемых значений входных и выходных показателей системы;

– не позволяют описать структуру сложной системы и, проведя исследование эффективности всех подсистем, интегрировать полученные оценки в единую оценку (или группу оценок) эффективности сложной системы, что было бы наглядно для пользователя (системного аналитика или лица, принимающего решения);

– не используют «большие» системы управления базами данных (СУБД), такие, как Oracle, PostgreSQL, MySQL и др., для хранения данных и манипулирования ими. Это снижает надежность операций с данными, а также лишает пользователя всех других преимуществ, которые дает использование СУБД.

Таким образом, есть необходимость в разработке системы поддержки принятия решений (СППР), которая могла бы служить в качестве аналитического и советующего инструмента при проведении комплексных исследований эффективности сложных систем. На данном этапе нами предложены основные принципы, которых следует придерживаться при разработке СППР, а также сформулированы конкретные требования, которые должны предъявляться к СППР, чтобы эти принципы были реализованы на практике (отметим, что

четкой границы между принципами и требованиями нет). Предложена также архитектура программного продукта, в которой показаны основные подсистемы СППР и отражены взаимосвязи между ними.

К числу потенциальных пользователей СППР относятся [4]:

– руководитель предприятия (подразделения), заинтересованный в наличии инструментария, который может помочь в выявлении резервов повышения эффективности, в прогнозировании эффективности при проведении организационных изменений;

– исследователь-практик (системный аналитик), которому, в дополнение к функциям программного продукта, требующимся руководителю, может быть очень полезным также наличие в СППР типовых сценариев для исследования эффективности систем в типичных ситуациях. Такие сценарии – это совокупности процедур (функций) программного продукта, активизируемые минимальными действиями пользователя и позволяющие решать типовые задачи без больших затрат времени и усилий;

– ученый, которому необходима возможность выполнения исследовательских задач, например, проведения многовариантных расчетов на основе сгенерированных искусственных наборов данных;

– студент, которому нужен инструмент, позволяющий освоить теорию и практику методов исследования эффективности сложных систем.

Предлагаемые принципы разработки СППР служат в качестве главного регулятора всего процесса проектирования [4]. Следование им позволит, на наш взгляд, получить действительно полезный инструмент поддержки принятия решений при исследовании эффективности сложных систем.

1. Реализация подхода «описание – объяснение – предсказание – рекомендации». Описание означает получение оценок эффективности исследуемой системы. Для *объяснения* полученных результатов необходимо провести исследование эффективности на уровне подсистем. Прогнозирование эффективности – *предсказание* – предусматривает, в частности, анализ тенденций изменения эффективности в подсистемах. *Рекомендации* должны включать в себя не только указание целевых (конечных) значений показателей для неэффективных систем и подсистем, но также и возможную траекторию перевода системы из неэффективного состояния в эффективное.

2. Включение в СППР средств для структуризации процесса проведения исследования в соответствии со структурой исследуемой системы и задачами исследования. Очевидно, что эффективность сложной системы зависит от эффективности функционирования ее подсистем. При этом эффективность подсистемы может быть рассмотрена с различных позиций (например, как с точки зрения вышестоящей системы, так и с точки зрения самой исследуемой подсистемы) и в различных сферах ее деятельности (например, эффективность кафедр в вузе может быть оценена как в сфере научной деятельности, так и в сфере образовательной деятельности).

3. Реализация идеи многовариантных расчетов. Пользователь должен иметь возможность использовать различные методы (модели) для получения оценок эффективности исследуемых объектов, варьируя при этом наборы и значения переменных и параметров. Сопоставление полученных многовариантных результатов должно производиться в автоматизированном режиме.

4. Обеспечение возможности настройки СППР в соответствии с потребностями пользователя и уровнем его квалификации.

5. Реализация широкого спектра методов исследования эффективности, возможность расширения СППР путем добавления новых методов, методик, моделей. Важно отметить, что все дополнительные методы, методики и модели должны органично встраиваться в архитектуру СППР по единой схеме.

На основе анализа потребностей потенциальных пользователей и в соответствии с заявленными принципами можно сформулировать следующие требования к СППР [4].

1. Предоставление пользователю возможности описания структуры сложной системы. Структуру системы необходимо хранить в базе данных в виде матрицы, описывающей связи элементов. При этом необходимо хранить и информацию о характеристиках каждой такой связи: направленность связи (от первого элемента ко второму, от второго к первому, двусторонняя связь); сила связи (в обоих направлениях); тип связи (связь управления, информационная связь и др.); влияние этой связи на достижение глобальной цели всей системы и т. д.

2. Использование иерархического подхода при структуризации этапов исследования. За основу формирования логики работы с программным продуктом предлагается принять понятие «Исследование». Это понятие означает всю совокупность действий по обработке информации в процессе проведения исследования эффективности сложной системы. При этом необходимо учесть, что в рамках одного «Исследования» может исследоваться более одной системы. Аналогично, одна и та же система может быть объектом более одного «Исследования». Каждое «Исследование» может быть разделено на этапы, структура которых отражает структуру исследуемой системы и учитывает цели и задачи исследования. У каждого этапа может быть этап-родитель. Для объединения этапов в связанные группы следует использовать этапы-контейнеры. С этапом-контейнером непосредственно не связывается выполнение каких-либо вычислительных операций. Например, этап-контейнер «Исследование эффективности факультетов» может содержать рабочие этапы «Научная работа на факультетах» и «Учебно-методическая работа на факультетах».

Для реализации идеи многовариантных расчетов необходимо предоставить пользователю возможность формировать для каждого этапа различные варианты исполнения. Такие варианты могут различаться наборами объектов и переменных, а также параметрами моделей. Анализ влияния вариаций на оценку эффективности должен выполняться в автоматизированном режиме (это должно делать ядро программного продукта).

3. Следование принципу «один этап – один метод». Все варианты конкретного этапа должны выполняться на основе одного и того же метода. Если требуется исследовать какую-то подсистему различными методами, то необходимо создать этап-контейнер и включить в него рабочие этапы, реализованные на основе требуемых методов. Данный подход позволит упростить реализацию механизма автоматизированного сравнения результатов многовариантных расчетов.

4. Использование концепции репозитория. Репозиторий содержит всю информацию, необходимую для проведения конкретного «Исследования»: описания объектов (в том числе и гипотетических объектов, предложенных при реструктуризации системы); описания переменных; описания структуры системы (или систем, если их больше одной); исходные данные (возможно, за несколько временны*х* периодов); описания временны*х* периодов.

Предлагается следующая иерархия уровней репозитория по степени детализации: уровень «Исследования»; уровень этапа «Исследования»; уровень варианта этапа «Исследования».

С точки зрения степени обобществления информации должен существовать:

– глобальный репозиторий (доступ к нему должен быть открыт для всех пользователей, выполняющих другие «Исследования»);

– локальный репозиторий (доступен только для пользователей, проводящих конкретное «Исследование»).

Данные, содержащиеся в репозитории «Исследования», должны быть представлены в оригинальном виде, т. е. без каких-либо преобразований. Преобразования данных целесообразно выполнять уже непосредственно при работе с данными в рамках проведения этапа (варианта этапа) «Исследования».

5. Разработка языка описания сценариев работы с СППР. Такой язык должен позволять пользователю описывать взаимосвязи элементов системы в терминах теории эффективности. Он должен использоваться для написания сценариев работы СППР в тех случаях, когда требуется проведение многовариантных расчетов, а также для реализации высокоуровневых методик, построенных на основе одного или нескольких методов оценки эффективности (подобные методики предлагались, например, в работах [2, 9]). В этом языке должны быть предусмотрены соответствующие типы данных и операторы. Пользователь должен иметь возможность сформировать сценарий работы согласно методике и сохранить его описание в базе данных с тем, чтобы затем воспроизводить написанный сценарий на различных наборах данных (на различных системах).

6. Организация коллективной работы. Коллективное использование программного продукта может быть организовано через локальную сеть или на одном компьютере. При этом необходимо обеспечить раздельный доступ к данным, принадлежащим различным пользователям.

7. Наличие средств гибкого конфигурирования СППР. При этом следует использовать иерархический подход: конфигурирование на уровне всей СППР, конфигурирование на уровне «Исследования», конфигурирование на уровне конкретного пользователя.

8. Различная степень детализации результатов и форм их представления. Должны быть предусмотрены возможности для представления как детализированной, так и агрегированной информации о результатах. Результаты должны сохраняться и визуализироваться в виде таблиц базы данных, в виде текстового файла-отчета, в виде графиков и диаграмм.

В соответствии с вышеизложенными принципами и требованиями предлагается *архитектура СППР* [4], представленная на рис. 1. Главной подсистемой, которая управляет работой СППР, является ядро (диспетчер). Оно принимает запросы от интерфейса пользователя и преобразует их в вызовы процедур, содержащихся в библиотеке методов и моделей. Интерфейсом между ядром и СУБД служит подсистема манипулирования данными. Модули библиотеки методов и моделей не должны содержать вызовов SQL-запросов. Ядро формирует символьные строки команд для работы с базой данных и переадресует их для выполнения в подсистему манипулирования данными. Эта подсистема «знает», где находится база данных: на локальной машине или на другом сервере в локальной сети. Подсистема манипулирования данными выполняет команды и сообщает ядру результат: выполнено успешно или с ошибкой. Результаты запросов на выборку данных подсистема манипулирования данными передает ядру. Ядро готовит данные для передачи их функциям (процедурам), реализующим математические методы. При этом оно формирует необходимые структуры данных в памяти в том виде, в каком требуется для библиотеки методов и моделей. Ядро принимает результаты выполнения математических процедур от библиотеки методов и моделей и передает их для визуализации или формирует строки команд для вставки записей в таблицы базы данных.

Ядро считывает файлы конфигурации и формирует необходимые глобальные структуры данных.

Важным вопросом является выбор средств *программной реализации* СППР. Используемые средства должны обеспечивать переносимость программного продукта в различные операционные среды: программный продукт (конечно, после перекомпиляции исходных текстов) должен работать не только в среде операционной системы Windows, но также и в среде операционной системы UNIX (в частности, Linux и FreeBSD). Исходя из этого, библиотека методов и моделей должна быть написана на языке C, а интерфейс пользователя представляется целесообразным разработать на языке C++ с использованием многоплатформенной библиотеки wxWidgets (http://www.wxwidgets.org).

Рис. 1. Предлагаемая архитектура программного продукта

От СУБД зависит удобство выполнения и надежность операций манипулирования данными. В качестве СУБД, на наш взгляд, целесообразно выбрать PostgreSQL (http://www.postgresql.org). Эта система управления базами данных поддерживает стандарт языка SQL, отличается высокой надежностью и быстродействием.

Немаловажным фактором при выборе инструментальных программных средств является их цена. В этом плане предлагаемые программные средства имеют преимущество: они являются бесплатными и доступны в исходных кодах.

Выводы. Предлагаемые архитектурные решения позволят создать СППР, которая будет иметь ряд преимуществ по сравнению с программными продуктами, доступными в настоящее время. К главным преимуществам СППР следует отнести:

– возможность комплексного исследования эффективности сложной системы с учетом ее структуры;

– реализация идеи многовариантных исследований с автоматизированным сопоставлением полученных результатов;

– организация хранения и обработки данных в соответствии с теорией баз данных и с использованием профессиональной СУБД.

Таким образом, программный продукт, разработанный в соответствии с изложенными принципами и требованиями, мог бы стать полноценной системой поддержки принятия

решений в процессе исследования эффективности сложных систем в экономике, социальной сфере, административном управлении.

Библиографический список

1. Анализ эффективности функционирования сложных систем [Текст] / В. Е. Кривоножко, А. И. Пропой, Р. В. Сеньков, И. В. Родченков, П. М. Анохин // Автоматизация проектирования. – 1999. – № 1. – С. 2–7.

2. Антамошкин, А. Н. Методика исследования эффективности сложных иерархических систем [Текст] / А. Н. Антамошкин, О. Н. Моргунова, Е. П. Моргунов // Вестник Сиб. гос. аэрокосмич. ун-та. – 2006. – Вып. 2 (9). – С. 9–13.

3. Клейнер, Г. Б. Производственные функции: Теория, методы, применение [Текст] / Г. Б. Клейнер. – М. : Финансы и статистика, 1986. – 239 с.

4. Моргунов, Е. П. Архитектура и принципы разработки системы поддержки принятия решений для исследования эффективности сложных систем [Текст] / Е. П. Моргунов // XI Международная научно-практич. конф. «Системный анализ в проектировании и управлении», 28–30 июня 2007 г. (г. Санкт-Петербург) : труды. В 3 ч. Ч. 3 / Санкт-Петербургский гос. политехн. ун-т. – СПб. : Изд-во Политехн. ун-та, 2007. — С. 232–237.

5. Надежность и эффективность в технике [Текст] : справочник / Ред. совет: В. С. Авдуевский (пред.) и др. В 10 т. Т. 1. Методология. Организация. Терминология / Под ред. А. И. Рембезы. – М. : Машиностроение, 1986. – 224 с.

6. Соломонов, Ю. С. Большие системы: гарантийный надзор и эффективность [Текст] / Ю. С. Соломонов, Ф. К. Шахтарин. – М. : Машиностроение, 2003. – 368 с.

7. Charnes, A. Measuring the Efficiency of Decision Making Units [Text] / A. Charnes, W. W. Cooper, E. Rhodes // European Journal of Operational Research. – 1978. – Vol. 2. – P. 429–444.

8. Coelli, T. An Introduction to Efficiency and Productivity Analysis [Text] / T. Coelli, D. S. Prasada Rao, G. E. Battese. – Boston : Kluwer Academic Publishers, 1998.– 275 p.

9. Cook, W. D. Hierarchies and Groups in DEA [Text] / W. D. Cook, D. Chai, J. Doyle, R. Green // Journal of Productivity Analysis. – 1998. – Vol. 10. – P. 177–198.

10. Cooper, W. W. Data Envelopment Analysis [Text] : A Comprehensive Text with Models, Applications, References, and DEA-Solver Software / W. W. Cooper, L. M. Seiford, K. Tone.– Boston : Kluwer Academic Publishers, 2000. – 318 p.

11. Handbook on Data Envelopment Analysis [Text] / W. W. Cooper, L. M. Seiford, J. Zhu (Eds.). – Boston : Kluwer Academic Publishers, 2004. – 608 p.

E. P. Morgunov

Decision support system for efficiency assessment of complex systems: design principles, requirements, and architecture

Siberian State Aerospace University, Krasnoyarsk

Basic design principles and requirements for decision support system used for efficiency assessment of complex systems are formulated. Architecture of such a system is proposed and means for programming implementation of this system are justified