УДК 800.92:681.3.06

ПРОГРАММИРОВАНИЕ ОПТИМИЗАЦИОННЫХ ЗАДАЧ УПРАВЛЕНИЯ. ВЗАИМОДЕЙСТВИЕ С ПОЛЬЗОВАТЕЛЕМ ПРИ ДЛИТЕЛЬНЫХ РАСЧЕТАХ

Ж.Э.КАЗЫБАЕВА, А.Д.АСКАРОВА
E.mail. ksucta@elcat.kg

Макалада оптимизациялык башкаруу маселелеринин автоматташтырылган маалымдоо системаларындагы жана колдонмо программаларынын талаптарына ылайык чечүү жолдору каралган. Жогоруда айтылган программаларды жазуу үчүн C# программалоо тилинин колдонулушу келтирилген. Узак убакта эсептелүүдөгү колдонуучу менен баарлашуунун ыкмалары кененирээк келтирилип каралды.

В статье сформулированы особенности решения оптимизационных задач управления в автоматизированных информационных системах и соответствующие требования к прикладным программам. Рассмотрено использование языка программирования C# для написания таких программ. Подробно рассмотрены способы построения диалога с пользователем во время выполнения длительных расчетов.

In article features of the solution of optimizing problems of management in the automated information systems and relevant requirements to applied programs are created. C# programming language use for writing of such programs is considered. Ways of creation of dialogue with the user are in detail considered during performance of long calculations.

В современных автоматизированных информационных системах управления (АСУ) часто необходимо решать оптимизационные задачи, такие как задача календарного планирования, задача коммивояжера, транспортная задача, «задача о ранце», задача о назначениях и другие.

В реальных системах высокая сложность этих задач и их большая размерность вынуждают применять для их решения различные эвристические методы (генетический алгоритм, метод роя частиц, муравьиные алгоритмы, нейронные сети и др.).

Поскольку сложность задач управления экономикой растет быстрее, чем сама экономика, и еще гораздо быстрее, чем число занятых в экономике людей /1/, постольку и размерность таких задач постоянно повышается. При этом поиск даже приближенных квазиоптимальных решений может занимать значительное для конечного пользователя время. Поэтому необходимо уделять внимание повышению скорости расчетов и удобству пользователя при их выполнении.

Особенностями решения задач управления (ЗУ) в современных информационных системах являются:

1. Высокая размерность.
2. Как правило, условия оптимизационных задач могут часто меняться, так как производственно-экономические условия даже на отдельном объекте динамично изменяются.
3. При использовании на предприятиях автоматизированных информационных систем все количественные исходные данные ЗУ, как правило, содержатся в централизованной базе данных, в нее же необходимо сохранять результаты решения.
4. Нередко полученные при решении ЗУ данные необходимо распечатывать и делать доступными для просмотра через браузеры в локальной или глобальной сети.
5. ЗУ запускаются на выполнение конечными пользователями, заинтересованными не только в быстром и качественном решении, но и в удобстве работы с программными приложениями.

Из анализа перечисленных особенностей можно вывести следующие требования к программам, выполняющим решение ЗУ.

1. Высокая надежность и безопасность приложений, поскольку при решении ЗУ в автоматизированных системах недопустимы сбои ПО и риск нанесения вреда системе.

2. Работа в вычислительных сетях. Предполагается использование сетевых ресурсов и возможность доступа к части функций через сеть Интернет.

3. Взаимодействие с базами данных. В информационных системах чаще всего используются реляционные базы данных.

4. Возможность интеграции программ. При построении сложных информационных систем необходимо стремиться к гибкости, которая достигается, в частности, разделением системы программ на модули, которые могут быть написаны на разных языках программирования и при этом должны без проблем взаимодействовать между собой.

5. Быстрое создание и изменение программ. Выполнение этого требования позволяет не отставать от развития предприятий, прогресса технологий управления и технических средств.

6. Продуманный интуитивно-понятный интерфейс, предоставляющий пользователю возможность легко вносить изменения в условия задач, выбирать режим решения и способы представления полученных результатов.

7. При выполнении расчетов, занимающих заметное для пользователя время, необходимо выводить информацию о ходе решения и предоставлять пользователю возможность досрочного корректного завершения работы программы, например, если ему нужно срочно покинуть свое рабочее место.

В данной работе в качестве инструмента, позволяющего создавать соответствующие требованиям приложения, предлагается использовать Visual C#.

Действительно, C# обладает всеми требуемыми свойствами. Использование платформы .NET обеспечивает высокий уровень безопасности ПО. C# имеет одновременно чистый, как в Java, понятный, как в Viual Basic, синтаксис и высокую гибкость и мощь, почти как C++ /2/. Это позволяет быстро создавать приложения для решения практически любых задач, значительно сокращая время программиста, в том числе на поиск ошибок в коде /3/.

Спецификация языка (CLS) и система поддержки общих типов (CTS) обеспечивают правильное взаимодействие любых языков программирования на платформе .NET /4/.

Платформа .NET имеет мощные средства для работы с базами данных (ADO.NET) и создания приложений под Интернет (ASP.NET).

Что касается скорости работы, следование некоторым правилам программирования повышает производительность до уровня, подходящего даже для научных вычислений /5/. Кроме того, программы, работающие в среде .NET, автоматически оптимизируются под конкретный компьютер /3/.

Из сказанного следует, что язык Visual C# очень хорошо подходит под указанные требования с первого по пятое. Что касается остальных требований, среды для создания программ на C# имеют все необходимое для создания интерфейса с необходимыми свойствами, нужно только иметь их использовать.

Как было отмечено, при выполнении длительных расчетов возникает проблема, как построить взаимодействия приложения с пользователем на время вычислений.

Можно предложить несколько вариантов.

1. Приложение не отвечает на действия пользователя и не выводит во время расчетов данных о ходе решения.

2. Приложение регулярно приостанавливает вычисления, выводит информацию и обрабатывает действия, которые успел совершить пользователь.

3. Разделение вычислений и организации диалога на два параллельно работающих потока.

Критерии оценки вариантов: быстрота, наличие вывода информации о ходе решения, возможность корректно прервать работу и простота реализации.

Первый вариант является самым быстрым, так как программа выполняет только вычисления, не тратя время на взаимодействие с пользователем, и этот вариант наиболее прост в реализации. Но пользователь не сможет видеть процент выполнения расчетов, а диспетчер задач операционной системы сообщит пользователю, что приложение «не отвечает», что может быть понято пользователем как зависание – тогда он просто закроет программу. Когда даже примерно не известно время ожидания, пользователю будет некомфортно.

Второй вариант значительно хуже по производительности, так как вычисления будут прерываться для реакции на внешние действия, зато в эти паузы можно выводить необходимые данные и позволить пользователю в любой момент остановить вычисления, не закрывая приложение и без утраты промежуточных результатов. В C# организовать обработку приложением внешних воздействий (сообщений) можно командой *Application.DoEvents()*.

Третий вариант ненамного уступает первому по скорости и второму по качеству интерфейса за счет разделения функций между потоками, недостаток способа – необходимость синхронизации потоков.

Рассмотрим в этом варианте два способа. Первый: создается поток для вычислений, а основной поток по сигналам от таймера выводит на экран процент выполнения расчетов и реагирует на действия пользователя. Второй: окно скрывается в область системных уведомлений, и создается поток для вывода информации о ходе решения и контекстного меню при наведении пользователем указателя на иконку в области уведомлений.

При реализации на платформе .NET в первом случае используется объект Windows.Forms.Timer, во втором – объекты Windows.Forms.ContexMenuStrip и Windows.Forms.NotifyIcon.

Достоинства второго способа: на время расчетов окно сворачивается и не мешает пользователю. Однако, если общее время расчетов мало (несколько секунд), второй вариант представляется неудачным, так как окно исчезает и сразу появляется, это может раздражать. По скорости второй способ в среднем лучше.

Приводим усредненные по нескольким типам расчетов данные о быстродействии различных вариантов (табл. 1).

Таблица 1

Сравнение быстродействия

Вариант	Время, с	Время относительно первого варианта
Без взаимодействия	4,163	1
С взаимодействием	5,308	1,275
Два потока, способ 1	4,356	1,046
Два потока, способ 2	4,214	1,0122

Решение задач оптимизации в автоматизированных системах имеет свои особенности, которые необходимо учитывать при разработке программного обеспечения. Язык программирования Visula C# позволяет создавать программное обеспечение, в полной мере соответствующее этим особенностям. Грамотное построение интерфейса с пользователем, соблюдение баланса между комфортом и производительностью является необходимым умением разработчика автоматизированных систем.

Список литературы

1. Глушков В.М. Что такое ОГАС?/ В.М. Глушков, В.Я. Валах. – М.: Наука, 1980. – 71 с.

2. Троелсен Э. C# и платформа .NET. Библиотека программиста/ Э.Троелсен; пер. с англ. Р.Михеев. – СПб.: Питер, 2004. – 796 с.

3. Гросс К. C# 2008/ К.Гросс; пер. с англ. – СПб.: БВХ-Петербург, 2009. – 576 с.

4 . Шилдт Г. Полный справочник по C#/ Г. Шилдт; пер. с англ. – М.: Издательский дом «Вильямс», 2004. – 752 с.

5. Фахад Г. C# и наука: применение языковых средств C# в проектах для научных вычислений [Электронный ресурс] // MSDN Magazine. Электрон.дан. URL:http://msdn.microsoft.com/ru-ru/library/dd353137.aspx, свободный. Яз.рус. (дата обращения 12.12.2011).