ІНСТРУМЕНТАЛЬНЕ ЗАБЕЗПЕЧЕННЯ КУРСУ КОМП'ЮТЕРНОГО МОДЕЛЮВАННЯ

В.М. Соловйов, С.О. Семеріков, І.О. Теплицький

Комп'ютерне моделювання як сучасний метод теоретичного пізнання неминуче ставить перед дослідником питання про вибір середовища для моделювання, адекватного досліджуваній проблемі. Навчання методам комп'ютерного моделювання є важливою складовою підготовки сучасного фахівця [1, 5]. Розуміючи, що така робота не повинна носити епізодичний характер, ми розробили відповідні спецкурси для старших школярів та студентів. Нами було створено методичну систему навчання основам комп'ютерного моделювання, до складу якої входять: 1) програмно-методичний комплекс (навчальний посібник, методичні рекомендації, ППЗ); 2) система задач – змістовних моделей, підібраних у відповідності до дидактичних принципів поступовості, наочності й наступності, – та вправ для самостійного опрацювання. Експериментальна апробація методичної системи показала, що при навчанні доцільно не обмежуватися якимось одним середовищем моделювання – раціональним є перехід від одного середовища моделювання до іншого у міру оволодіння школярами та студентами знаннями з інформатики [4].

В [7] розглянуто основні ідеї, цілі та можливості створеного нами факультативного курсу комп'ютерного моделювання. При цьому зазначається, що на початковому етапі цілком придатним середовищем моделювання є електронні таблиці. Але при всій їх простоті та наочності, а також розвиненим засобам графічної інтерпретації залежностей між досліджуваними величинами слід розуміти, що електронні таблиці є ефективними для досить обмеженого кола порівняно простих і спеціально підібраних задач. Так, вже при спробах розв'язання задач, що передбачають організацію циклів з великою кількістю повторень, звичайне копіювання формул у відповідні стовпці з метою утримання даних для подальшої побудови графіків (тобто організація великих масивів даних) викликає значні утруднення, лишаючи користувача змоги простежити динаміку зміни значень будь-якої величини переглядом відповід-

ного стовпця. Організація циклу в межах однієї клітки таблиці неможлива принципово. Нарешті, електронні таблиці стають зовсім непридатними для задач, що вимагають засобів візуального спостереження динаміки процесу, тобто візуалізація поточного стану об'єкту. Такі задачі традиційно розв'язують із використанням середовищ, створених на основі мов програмування високого рівня. Одним з критеріїв необхідності зміни середовища моделювання є ситуація, коли сама таблиця перестає використовуватися для аналізу даних, залишаючи користувачеві можливість лише їх графічної інтерпретації.

Проаналізуємо можливості, переваги та недоліки різних середовищ моделювання на конкретному прикладі розв'язання задачі побудови траєкторій руху планет Сонячної системи [8].

Почнемо з найпростішого випадку – побудови траєкторії руху однієї планети. Сила всесвітнього тяжіння, що діє на планету, надає їй доцентрового прискорення. За другим законом Ньютона

$$m\vec{a} = -G\frac{mM}{\left|\vec{r}\right|^3}\vec{r},$$

де G – стала всесвітнього тяжіння, m – маса планети, M – маса Сонця ($M \gg m$), $\vec{r}$ – радіус-вектор планети, $\vec{a}$ – прискорення планети.

Докладне розв'язання задачі на побудову траєкторії у середовищі електронних таблиць наведено в [7]. На рис. 1 показано результати моделювання – фрагмент таблиці та траєкторія. Повна таблиця містить 14 стовпців та близько 400 рядків. Такі розміри таблиці вже утруднюють аналіз даних, проте залишається можливість їх графічної інтерпретації засобами розглядуваного середовища. Але розгляд усіх дев'яти планет разом із Сонцем вимагатиме вже 62 стовпці та близько 90500 рядків, що зумовлює потребу у іншому середовищі.

Повний курс основ комп'ютерного моделювання після початкового етапу включає ще дві істотні частини: вивчення методів ефективного використання ресурсів комп'ютера для розв'язання задач моделювання та вивчення об'єктно-орієнтованого підходу до моделювання динамічних систем.

Принцип наступності реалізовано, з одного боку, як формування нових знань

на основі попередніх, а, з іншого – як органічний зв'язок у системі неперервного навчання (школа–вуз). Таким чином, у запропонованій методичній системі засоби й методи вивчення комп'ютерного моделювання в ході навчання зазнають змін, спрямованих на опанування учнями все більш досконалих прийомів роботи.

Наступне середовище для моделювання, яке ми розглянемо – це мова програмування. Перехід від електронних таблиць до мови програмування позбавляє зручності миттєвої зміни параметрів моделі та наочності процесу розрахунків, проте надає такі нові можливості:

- поточні значення змінних (роль яких відігравали комірки стовпців) в програмі можуть залежати від попередніх;
- групування подібних між собою змінних у масиви;
- зняття обмежень на кількість ітерацій (роль яких відігравали рядки таблиці) та послаблення обмежень на кількість змінних;
- графічна інтерпретація динаміки моделі в процесі розрахунків (а не по їх закінченню, як в електронних таблицях).

Існує можливість, залишаючись у середовищі електронних таблиць, використати вбудовану мову програмування, наприклад – Visual Basic, внутрішню мову електронних таблиць Excel. Такий крок дозволяє об'єднати переваги електронних таблиць та мов програмування високого рівня, проте орієнтація на мову програмування лише однієї фірми неминуче ставить у залежність від стратегії розвитку мови, а ця стратегія обирається переважно з комерційних міркувань. До того ж, швидкість еволюції такої мови (табл. 1), відсутність інваріантного ядра та переобтяженість додатковими можливостями, які підвищують вимоги до техніки, утруднюють її вивчення у закладах освіти.

Таблиця 1.

Еволюція мови програмування електронних таблиць Microsoft Excel

Рік	*Мова*	*Версія Excel*	*Операційна система*	*Апаратна платформа*
1994	ExcelBasic	Excel 5	Windows 3.1	386DX, 4 Мб
1997	Visual Basic for	Excel 97	Windows 95/NT 4.0	Pentium, 16 Мб

Рік	*Мова*	*Версія Excel*	*Операційна система*	*Апаратна платформа*
	Applications			
2000	Visual Basic	Excel 2000	Windows 98/2000	Pentium II, 64 Мб

Тому у якості наступного середовища для моделювання ми пропонуємо використовувати замість мови електронних таблиць процедурну мову високого рівня. Однією з найпоширеніших мов високого рівня, що користується великою популярністю у системі освіти США, Франції, Німеччини та інших країн, є мова С. Доступність некомерційних та ліцензованих для закладів освіти операційних систем і компіляторів цієї мови (табл. 2) дозволяє використовувати їх майже на всіх типах техніки, встановлених у наших школах.

Таблиця 2.

Деякі вільно поширювані компілятори мови С.

Компілятор	*Операційна система*	*Апаратна платформа*
C80 (Microsoft, 1979)	CP/M-80, MSX DOS	Корвет, Yamaha
Turbo C 2.0 (Borland, 1988)	MS DOS 3.3-6.22	Пошук-1, IBM-сумісні
djgpp C/C++ (DJ Delorie, 1999)	MS DOS 5.0-6.22	386DX та кращі
gcc (GNU/FSF, 1999)	FreeBSD, Linux	386DX та кращі

Перехід до моделювання у середовищі мови програмування так само, як і перехід від паперових розрахунків до електронних таблиць, є еволюційним кроком, спрямованим на підвищення ефективності процесу розв’язання задач моделювання.

Повертаючись до розглядуваної задачі, зазначимо, що на будь-яку планету діє не лише Сонце, а й кожна з решти планет. Враховуючи рух Сонця, математична модель руху кожної з планет набуває такого вигляду:

$$\vec{a}_i = -G \sum_{\substack{j=1 \\ (i \neq j)}}^{N} \frac{m_j}{\left|\vec{r}_{ij}\right|^3} \vec{r}_{ij} ,$$

$$\vec{\upsilon}_i^{cur} = \vec{\upsilon}_i^{prev} + \vec{a}_i \Delta t ,$$

$$\vec{r}_i^{\,cur} = \vec{r}_i^{\,prev} + \vec{\upsilon}_i^{\,cur}\Delta t,$$

де $\vec{a}_i$ – результуюче прискорення *i*-ої планети внаслідок дії на неї решти планет;

i=1…N – номер планети (9 планет та Сонце);

m_j – маса *j*-ої планети;

$\vec{r}_{ij}$ – радіус-вектор, що сполучає *j*-ту та *i*-ту планети;

$\vec{\upsilon}_i^{\,prev}$, $\vec{\upsilon}_i^{\,cur}$ – швидкості *i*-ої планети у попередній та поточний моменти часу відповідно;

$\vec{r}_i^{\,prev}$, $\vec{r}_i^{\,cur}$ – радіус-вектори *i*-ої планети у попередній та поточний моменти часу.

У початковий момент часу вважатимемо, що всі планети розташовані на одному промені (вісь *Ox*) на середніх геліоцентричних відстанях та мають середні орбітальні швидкості:

$$\vec{\upsilon}_i^{\,start}=(0,\ \frac{2\pi}{T_i}a_{\odot i},\ 0),$$

$$\vec{r}_i^{\,start}=(a_{\odot i},\ 0,\ 0),$$

де $\vec{\upsilon}_i^{\,start}$ – швидкість *i*-ої планети у початковий момент часу;

$\vec{r}_i^{\,start}$ – радіус-вектор *i*-ої планети у початковий момент часу;

T_i – сидеричний період *i*-ої планети;

$a_{\odot i}$ – середня геліоцентрична відстань *i*-ої планети.

Змінні, які використовуються у програмі, пробігають значення, що зберігалися у стовпцях електронної таблиці. Оскільки для обчислення поточного значення деякої змінної необхідно знати лише її попереднє значення, зникає потреба у зберіганні всіх проміжних значень.

Результатом такого переходу є суттєва економія пам’яті та зниження вимог до використовуваних ресурсів. Якщо в електронних таблицях ми могли побудувати графік, лише закінчивши процес нагромадження всіх необхідних значень змінних, то в середовищі мови програмування ми маємо додаткову можливість графічної візуалізації результатів обчислень під час проведення розрахунків, що робить цю мо-

дель динамічною не лише за суттю, а й за зовнішніми проявами.

Усі описані середовища відображають не стільки саму модель, скільки формалізований засобами процедурної методології алгоритм роботи з нею. Кожну планету ми можемо охарактеризувати таким набором даних:

– статичні параметри: назва *name*; сидеричний період *T*; середня геліоцентрична відстань $a_{\odot}$; маса *m*.
– динамічні параметри: вектор прискорення $\vec{a}$; вектор швидкості $\vec{\upsilon}$; радіус-вектор $\vec{r}$.

Якщо визначення типів статичних параметрів не викликає утруднень (*name* – рядок, *T*, $a_{\odot}$ та *m* – дійсні змінні), то кожен динамічний параметр у електронних таблицях доводиться подавати через проекції на координатні вісі та зберігати у окремих стовпцях. У середовищі процедурної мови програмування необхідно заводити масиви проекцій, перетворюючи найпростіші векторні операції на операції з масивами, що включають допоміжні цикли, не передбачені алгоритмом. Це пов'язано з відсутністю векторного типу даних. Якби ми мали вбудований у мову програмування тип "арифметичний вектор", програмна реалізація моделі була б записана у компактній, згорнутій формі, максимально наближеній до запису алгоритму [6].

Процедурна мова С не має такого вбудованого типу, так само, як і її об'єктно-орієнтоване розширення С++. Проте мова С++ дає змогу його створити, оскільки легко пристосовується для використання у спеціальних областях шляхом створення на її базі складених понять та конструкцій, що слугують будівельними блоками при програмуванні у спеціальній області [2]. Автори першої об'єктно-орієнтованої мови програмування Simula-67 наголошували на тому, що об'єктно-орієнтоване середовище є природним середовищем для розв'язання задач моделювання (*simulation*) [3]. Об'єктно-орієнтований підхід (ООП) вносить якісні зміни у самий процес моделювання, надаючи потужні можливості щодо підвищення рівня абстракції даних – від масивів до векторних об'єктів.

Наявність типу "арифметичний вектор" та визначених операцій над ним спрощує запис програми, наближуючи його до алгоритмічного. До того ж об'єктно-

орієнтоване середовище дозволяє від групування зазначених вище параметрів за ознакою спільності фізичного змісту перейти до групування за ознакою “бути планетою”. В такому разі ми розглядатимемо Сонячну систему як набір планет (вважаючи Сонце однією з них), а не як сукупність прискорень, швидкостей, координат тощо.

Для створення об’єкту типу “планета” проаналізуємо статичні дані, що його характеризують. Ім’я планети не впливає на процес обчислень, проте ми залишимо цей параметр, щоб відрізняти одне тіло від іншого. Стосовно середньої геліоцентричної відстані та сидеричного періоду зауважимо, що вони потрібні лише для визначення початкового радіус-вектора та вектора швидкості, тому не будемо відносити їх до даних об’єкту типу “планета”.

Групування даних за ознакою “бути планетою” не є прерогативою ООП, воно можливе і в процедурній методології. Але принциповою перевагою ООП є те, що об’єкти не лише зберігають свої дані, а й змінюють їх, обмінюються повідомленнями, взаємодіють тощо. В розглядуваному випадку взаємодія об’єктів типу “планета” є інформаційною моделлю фізичної взаємодії.

Побудова інформаційної моделі вимагає визначення операцій над об’єктами. Перш за все, будемо відрізняти об’єкти один від одного за іменем, визначивши операції перевірки на рівність та нерівність. Далі введемо операції обчислення відстані між об’єктами та взаємодії об’єкта з усіма іншими. Декларування класу, що породжує об’єкти типу “планета”, може бути таким:

```
typedef vector<double> dvector; /*тип "дійсний вектор"*/

class Planet /*клас "планета"*/
{
  /*дані, що характеризують планету*/
  string name; /*ім’я планети*/
  dvector r,v,a; /*радіус-вектор, вектори швидкості та прискорення*/
  double m; /*маса тіла*/
 public: /*визначення операцій над планетою*/
  Planet(){name=""; m=0; v=a=r=dvector(3); } /*створення об’єкту, про
який поки що немає даних*/
  Planet(string _name, dvector _r, dvector _v, dvector _a, double _m):
name(_name),r(_r),v(_v),a(_a),m(_m) {} /*створення об’єкта, про який
відомі всі дані*/
  Planet(Planet &p):name(p.name),r(p.r),v(p.v),a(p.a),m(p.m){ }
/*створення планети за даними про іншу планету*/
```

```
  void Iter(); /*взаємодія об’єкта з усіма іншими*/
  /*перевірка об’єктів на рівність та нерівність*/
  int operator ==(Planet);
  int operator !=(Planet);
  friend double Distance(Planet,Planet); /*обчислення відстані між
об’єктами*/
};
```

Для того, щоб виконати один крок обчислень (ітерацію), для кожної планети необхідно викликати метод Iter(), який реалізує взаємодію поточного об'єкта з усіма іншими. Тоді основна частина програми може бути такою:

```
  while(умова_завершення_програми)
  {
    for(long i=0;i<PCount;i++) /*перебір усіх тіл*/
      planets[i].Iter(); /*"включення" взаємодії і-го тіла*/
    візуалізація_результатів_обчислень
  }
```

Метод Iter(), наведений нижче, показує використання побудованих типів векторів та планет:

```
void Planet::Iter()
{
  a=dvector(3); /*запис в а вектора (0,0,0)*/
  for(long i=0;i<PCount;i++) /*цикл перебору планет*/
    if(planets[i]!=(*this)) /*перевірка на неспівпадання планет*/
    {
      double er=Distance(planets[i],*this); /*обчислення відстані між
і-ою планетою та поточною*/
      a=a-G*planets[i].m*(r-planets[i].r)*(1./(er*er*er));
/*обчислення нового значення вектора прискорення*/
    }
  v=v+a*dt; /*обчислення нового значення вектора швидкості*/
  r=r+v*dt; /*обчислення нового значення радіус-вектора*/
}
```

На рис. 2 зображено стан Сонячної системи у момент завершення Ураном повного оберту.

Таким чином, разом із підвищенням рівня абстракції за рахунок створення засобами ООП нових типів даних “арифметичний вектор” та “планета” досягнуто високого рівня наочності та простоти програми. Отже, об’єктно-орієнтоване середовище моделювання найбільш природно відображає концепції сучасного комп’ютерного моделювання та є ефективним інструментом не лише при навчанні, а й у професійній діяльності.

Висновки.

1. Найчастіше з комп’ютерним моделюванням учнів знайомлять на заверша-

льному етапі вивчення програмування і рідше – разом із програмуванням. Застосування електронних таблиць дозволяє розпочати систематичний курс комп'ютерного моделювання помітно раніше.

2. Перехід від середовища електронних таблиць до мови програмування високого рівня викликаний не стільки тим, що з'являється можливість її використання, скільки можливістю суттєвого розширення спектру задач (моделей), таблична реалізація яких втрачає ефективність, хоч і залишається здійсненою.

3. У вищій школі вдосконалення техніки моделювання вимагає переходу до розгляду інформаційних моделей, що забезпечується об'єктно-орієнтованим середовищем. Таке середовище природним шляхом відображає досліджувані процеси на рівні інформаційних взаємодій.

Література

1. Белошапка В.К., Лесневский А.С. Основы информационного моделирования // Информатика и образование.– 1989.– № 3.– с. 17–25.
2. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. – 2-е изд. / Пер. с англ. – М.: Бином, СПб.: Невский диалект, 1999. – 560 с.
3. Дал У.-И., Мюрхауг Б., Нюгорд К. Симула–67. Универсальный язык программирования. – М.: Мир, 1969. – 100 с.
4. Жалдак М.І. Яким бути шкільному курсу "Основи інформатики" // Комп'ютер у школі та сім'ї. – 1998. – № 1.
5. Матюшкин-Герке А. Учебно-прикладные задачи в курсе информатики // Информатика и образование. – 1992. – № 3–6.
6. Полищук А.П., Семериков С.А. Методы вычислений в классах языка C++: Учебное пособие. – Кривой Рог: Издательский отдел КГПИ, 1999. – 350 с.
7. Теплицький І.О. Використання електронних таблиць у комп'ютерному моделюванні // Комп'ютер у школі та сім'ї. – 1999. – № 2. – с. 27–32.
8. Фейнман Р., Лейтон Р., Сэндс М. Фейнмановские лекции по физике. – М.: Мир, 1967. – Т. 1. – 272 с.

a_{1x}	a_{1y}	a_{2x}	a_{2y}	v_{1x}	v_{1y}	v_{2x}	v_{2y}	x_1	y_1	x_2	y_2	Дано:	Земля
1,779E-08	0,000E+00	-0,006	0,00	0,00E+00	0,00E+00	0	30124	0,000E+00	0,000E+00	1,5000E+11	0,0000E+00	G=	6,672E-11
1,779E-08	3,087E-10	-0,006	0,00	7,69E-04								Δt=	8,64E+04
1,778E-08	6,173E-10	-0,006	0,00	2,31E-03								m_1=	2,000E+30
1,777E-08	9,257E-10	-0,006	0,00	3,84E-03								m_2=	6,000E+24
1,775E-08	1,234E-09	-0,006	0,00	5,38E-03								R=	1,500E+11
1,772E-08	1,541E-09	-0,006	0,00	6,91E-03								v_{1x0}=	0
1,769E-08	1,849E-09	-0,006	0,00	8,44E-03								v_{1y0}=	0
1,766E-08	2,155E-09	-0,006	0,00	9,97E-03								v_{2x0}=	0
1,761E-08	2,461E-09	-0,006	0,00	1,15E-02								v_{2y0}=	3,01E+04
1,757E-08	2,766E-09	-0,006	0,00	1,30E-02								x_{10}=	0
1,751E-08	3,069E-09	-0,006	0,00	1,45E-02								y_{10}=	0
1,746E-08	3,372E-09	-0,006	0,00	1,60E-02								x_{20}=	1,500E+11
1,739E-08	3,674E-09	-0,006	0,00	1,76E-02								y_{20}=	0
1,732E-08	3,974E-09	-0,006	0,00	1,91E-02								a_2=	1,53E+11
1,725E-08	4,273E-09	-0,006	0,00	2,06E-02									
1,717E-08	4,571E-09	-0,006	0,00	2,20E-02									
1,709E-08	4,867E-09	-0,006	0,00	2,35E-02									
1,700E-08	5,161E-09	-0,006	0,00	2,50E-02									
1,690E-08	5,454E-09	-0,006	0,00	2,65E-02									
1,680E-08	5,744E-09	-0,006	0,00	2,79E-02									

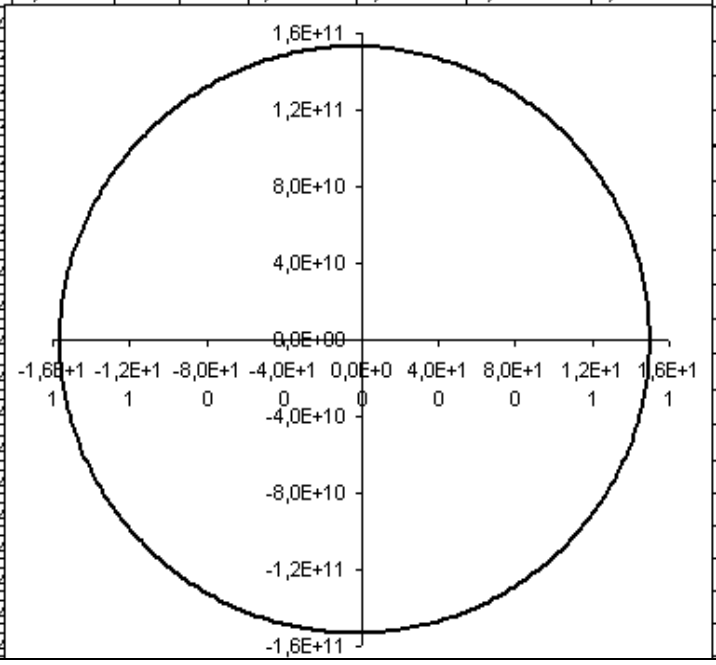

Рис. 1.

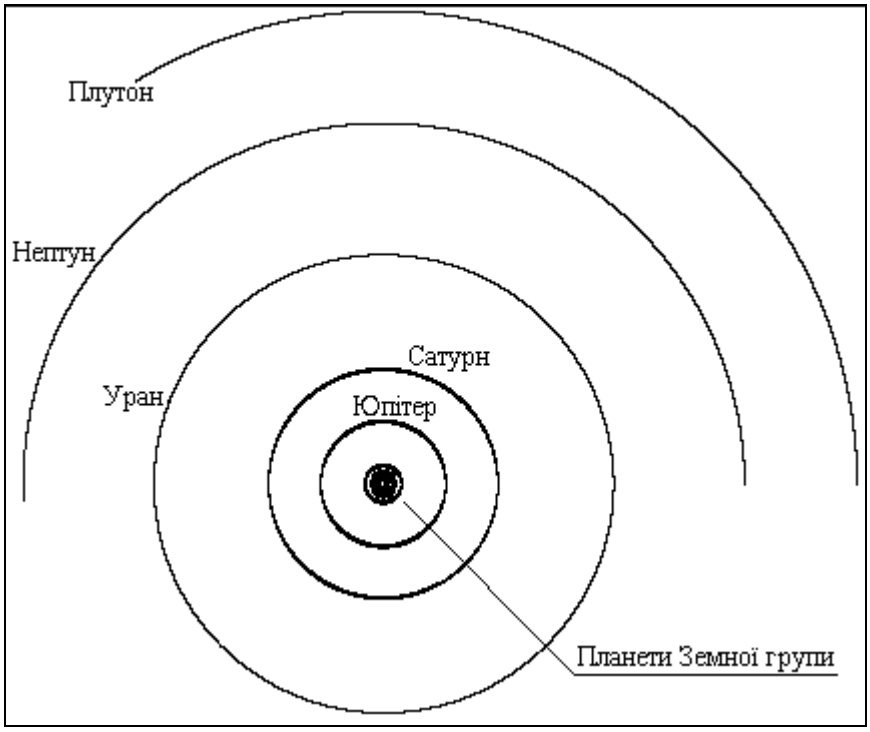

Рис. 2.