УДК 681.3:658

Глибовець А. М.

ВИКОРИСТАННЯ МОБІЛЬНИХ ПРИСТРОЇВ У ДИСТАНЦІЙНІЙ ОСВІТІ

У статті розглянуто базові концепції та основні підходи до доцільного використання мобільних пристроїв у комп 'ютерних системах підтримки дистанційного навчання за допомогою спеціально розроблених на основі технології J2ME (конфігурації CDLC із застосуванням профілю MIDP) програмних комплексів клієнт -серверного типу. Проілюстровано реалізацію підходу на прикладі підсистеми тестування. Моделювання зв'язку «клієнт — сервер» проведено push-моделлю. Всі питання обміну повідомленнями винесено в окремі класи клієнтського і серверного конекторів. Ефективно розв'язано основну проблему передачі та отримання даних між мобільним пристроєм і системою.

Вступ

На сьогодні існує багато різнопланових комп'ютерних систем підтримки дистанційного навчання (КПД Н) з різним рівнем інтелектуалізації [1—3]. Визнаною перевагою КПДН є мобільність. Під мобільними пристроями (МП) слід розуміти два типи пристроїв [4]:

· персональні мобільні інформаційні пристрої, які здатні до нестаціонарних мережних

Але ця властивість не є повністю реалізованою, оскільки користувач системи залишається прив'язаним до комп'ютера та засобів зв'язку. Проте зрозуміло, що користувачу не завжди доступний комп'ютер з підключенням до Інтернету, тому це накладає певні обмеження на процес навчання. Ця проблема може бути вирішена завдяки використанню мобільних пристроїв.

комунікацій,- мобільні телефони, пейджери з двостороннім зв'язком, персональні інформаційні пристрої («кишенькові комп'ютери») та органайзери;

• інформаційні пристрої із загальним з'єднанням, приєднані фіксованим безперервним мережним з'єднанням,— комп'ютерні приставки до телевізорів, телевізори з виходом в Інтернет, телефони з дисплеєм і можливістю виходу в Інтернет, комунікатори високого рівня, розважальні і навігаційні автомобільні системи.

Сьогодні МП першого типу набули вже достатнього поширення. Зокрема, в Україні кількість користувачів МП першого типу збільшується з кожним місяцем і наближається до 6 млн користувачів. Надалі говоритимемо про МП лише першого типу, позаяк пристрої другого типу ще недостатньо поширені і не мають засобів, що можуть бути застосовані в КПДН.

Виокремимо певну групу пристроїв, які найкраще можуть бути застосовані в КПДН. Для МП необхідно мати середовище, яке дає змогу запускати певні програми (програмні додатки). На сьогодні єдиною платформою розробки і реалізації програмних додатків для всіх М П є Java 2 Micro Edition. Зрозуміло, що МП також повинен мати досить великий екран. З усіх виділених пристроїв цю властивість мають лише КПК та мобільні телефони останніх поколінь. Врешті-решт, мобільний пристрій мусить мати можливість незалежно від інших пристроїв з'єднуватися з КПДН для обміну даними. Такій характеристиці з-поміж усіх пристроїв повністю відповідають лише мобільні телефони, оскільки вони мають можливість у будь-який час отримати доступ до системи. КПК можуть використовуватися лише з якимось іншим пристроєм у сукупності. Тому вважатимемо, що на сьогодні про ефективне використання мобільних пристроїв у дистанційній освіті можна говорити лише стосовно застосування мобільних телефонів.

Незважаючи на те, що майже всі мобільні телефони останніх поколінь мають кольорові дисплеї, на яких вміщується від 8 до ЗО рядків тексту, підтримують можливість показу графічних зображень, програвання аудіо- і відеофайлів та можливість з'єднання з Інтернетом за допомогою протоколів Wap та GPRS, їх можна розглядати лише як повнофункціональні термінали КПДН з обмеженими можливостями. Тобто МП можна вважати лише додатковим засобом в КПДН, який доповнює існуючі стаціонарні пристрої. Однак сучасні темпи розвитку технологій МП свідчать про те, що найближчим часом моделі наступних поколінь наблизяться за своїми можливостями до того, щоб вважати їх цілком функціональними терміналами, які можуть повністю замінити використання персональних комп'ютерів у дистанційній освіті.

МП у КПДН можна використовувати двома способами.

Перший спосіб полягає у використанні МП як браузерів WAP-порталів систем КПДН. У більшість МП вмонтовано браузери, які можуть забезпечити перегляд WAP-порталів. Тому завдяки МП користувач може просто працювати з КПДН завдяки спеціально розробленому WAP-порталу КПДН. До переваг цього способу належать:

- вирішення питання мобільності (користувач завдяки своєму МП із будь-якого місця і в зручний для нього час може з'єднатися з системою КПДН і продовжити своє навчання);
- простота реалізації забезпечення безпеки певного рівня;
- використання двох протоколів зв'язку WAP та G PRS для зв'язку із сервером.

Недоліками першого способу є:

- обмеженість інтерфейсу користувача (обмеженість МП не дає змоги використовувати WAP-портал з функціональними можливостями звичайної системи);
- незначна адаптованість до конкретних моделей МП;
- порівняно великий об'єм трафіку, що призводить до високої вартості використання;
- значна залежність від якості з'єднання з WAP-порталом, що може бути критичним для масового застосування.

Другий спосіб застосування МП в КПДН полягає у використанні спеціально розроблених програмних комплексів підтримки взаємодії М П з системою клієнт-серверного типу.

Цей спосіб забезпечує основні можливості попереднього підходу та надає ще й такі додаткові переваги:

- спеціально розроблений інтерфейс користувача, який дасть змогу максимально використовувати можливості МП;
- можливість розробки клієнтської частини під конкретні моделі МП з урахуванням їх технічних характеристик;
- невеликий об'єм трафіку (клієнтська частина містить інтерфейсну частину на локальному пристрої, і непотрібно щоразу за-

вантажувати її: під час роботи завантажуються лише необхідні дані).

Але й останній підхід не позбавлений певних недоліків. Серед них можна виокремити такі:

- проблеми налаштування клієнтської частини програми на особливості технічної реалізації конкретної моделі МП;
- нестабільність каналів мобільного зв'язку.

1. Інструментарій розробки

1.1. Java2ME

J2ME (Java 2 Micro Edition) об'єднує під своєю назвою багато технологій, кожна з яких вирішує свою конкретну задачу: специфікація J2M E визначає так звані конфігурації (configuration). Кожна конфігурація описує середовище виконання ЛМЕ-додатків (JVM, набір доступних класів, деякі правила функціонування додатків). Для конфігурації, своєю чергою, може бути визначено декілька профілів (profile), кожний з яких «уточнює» середовище виконання, додаючи чи забороняючи використання певних класів, визначаючи нові правила функціонування додатків. Досить чітко концепцію використання різних Java-технологій подано в [4] (рисі).

Згідно з тематикою нашої статті найбільше нас цікавитиме конфігурація пристроїв CDLC [5] (Connected Limited Device Configuration) і один з її профілів — MI DP [6] (Mobile Information Device Profile).

1.2. CLDC

Повний опис конфігурації CLDC можна знайти в [5]. Зазначимо лише декілька основних характеристик, які вирізняють середовище реалізації CLDC (Connected Limited Device Configuration) J2ME від інших.

1. Для CLDC була розроблена своя JVM, яка в реалізації Sun має назву KVM (Kilo VM). Ця реалізація не має операцій з плаваючою комою, не допускає фіналізації у класах, не реалізує JNI, не допускає використання користувацьких завантажувачів класів, має дуже слабкі Security-механізми, не має механізму Reflection та ін. Але вона має дуже малий розмір — декілька сотень кілобайт, що дає змогу розмістити таку віртуальну машину, наприклад, у мобільний телефон з дуже малим розміром пам'яті.

2. Бібліотеки класів, доступні в CLDC, можна поділити на дві групи: перша є підмножиною бібліотек J2SE, друга є специфічною для CLDC. До першої групи входять класи з таких пакетів J2SE, flKJava.lang.*, java.util.*, java.io.* Усі класи, які належать цій групі, сумісні з відповідними класами з J2SE знизу вгору. До другої групи входять класи з пакету javax.microedition.* Власне, сама CLDC з цієї групи реалізує лише Generic Connection Framework.

Слід зазначити, що специфікація CLDC сама по собі не визначає кінцеве середовище виконання, тому в реалізацію CLDC від Sun було включено додатковий пакет com.sun.kjava, класи якого реалізують тестовий користувацький інтерфейс і деякі протоколи для Generic Connection Framework.

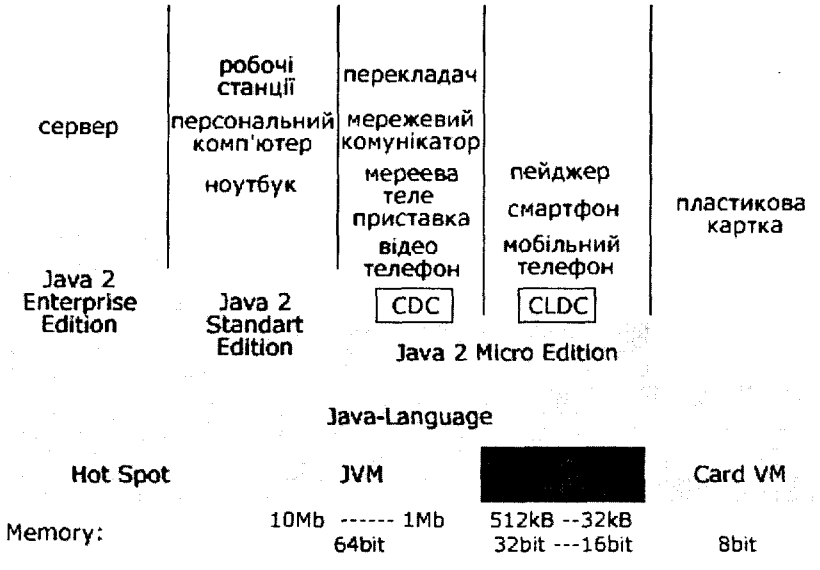

Рис 1. Еволюція використання Java-технологій

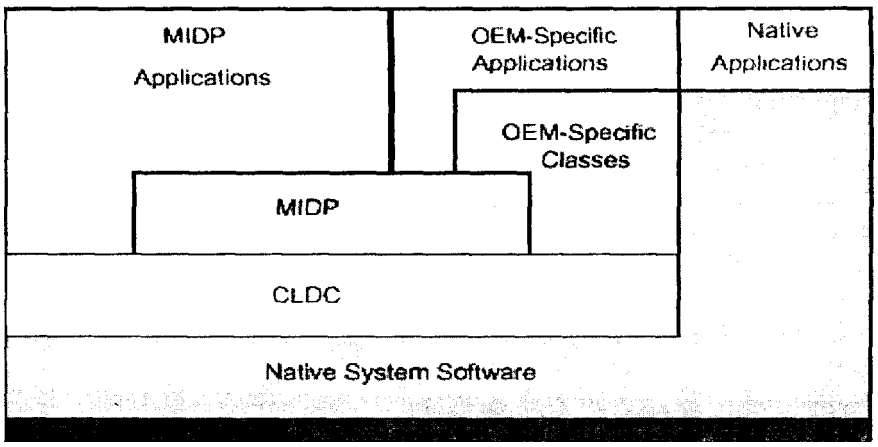

Рис. 2. Стек MIDP-додатка

1.3. MIDP

Профіль MIDP побудований на базі конфігурації CLDC і повністю визначає середовище виконання (але не всю інфраструктуру) додатка. Зазначений профіль націлений на створення додатків для таких мобільних пристроїв, як мобільні телефони, пейджери, PDA, Smart Phones. Місце, яке посідає профіль MIDP (Mobile Information Device Profile) в технології J2ME, показано на рис. 2.

M1DP визначає модель додатка, що дає змогу поділяти декільком додаткам обмежені ресурси МП,— цю модель називають MI Diet. Вона визначає, що таке MIDlet-додаток, як він має бути запакований, яке середовище виконання доступне для MI Diet і як має поводити себе додаток, щоб МП міг ним управляти [6].

2. Розробка додатка МП + КПД Н

Проілюструємо розробку програмного комплексу підтримки використання МП другого типу на прикладі підсистеми тестування.

Задля простоти розгляду зосередимося на такій схемі взаємодії МП з КПДН. Після аутентифікації МП сервер КПДН передає клієнтові тестове запитання з варіантами відповіді, які відображає клієнтська частина на екрані МП і надає можливість користувачеві вибрати правильну відповідь. Далі клієнт надсилає відповідь серверу. Сервер запам'ятовує відповідь і продовжує опитування до закінчення тесту. В кінці виставляється оцінка і надсилається користувачеві. Отже, структурно маємо три основні етапи реалізації взаємодії: формування й обробка даних у КПДН (сервер), відображення даних на МП і протокол взаємодії. Формування й обробка даних у КПДН істотно не відрізняються від традиційної реалізації серверної обробки. Відображення даних на МП має також традиційне вирішення, яке описано в багатьох джерелах, наприклад в [4]. Тому реалізацію цих частин не будемо докладно аналізувати в пропонованій статті. Головною проблемою при проектуванні цієї підсистеми є те, як MIDP-додатки мають спілкуватися з сервером. Отже, детальніше розглянемо розв'язок цієї задачі.

Для розв'язку застосуємо Generic Connection Framework (GCF) [7], що є підкласом бібліотеки класів CLDC і надає можливість зручно визначити єдиний високорівневий інтерфейс протоколів передачі даних у будь-якому вигляді. Зазначимо, що CLDC, визначаючи Generic Connection Framework, не реалізує жодного протоколу, покладаючись на реалізацію протоколів у кожному конкретному профілі. Специфікація MIDP, своєю чергою, потребує обов'язкової реалізації лише протоколу HTTP. Тому було вирішено для обміну даними між клієнтом і сервером використовувати протокол datagram (UDP).

Наводимо діаграму реалізованих класів GCF (рис. 3).

Як бачимо, класів небагато, але достатньо для роботи з будь-якими типами з'єднань за будь-якими протоколами.

З'єднання створюється за допомогою виклику статичного методу класу Connector:

Connector.open("<protocol>:<addressXparameters>").

Синтаксис рядка має відповідати RFC2396 [9]: <protocol> — протокол, за допомогою якого буде виконана спроба встановити з'єднання, <address> — адреса ресурсу, формат адреси за-

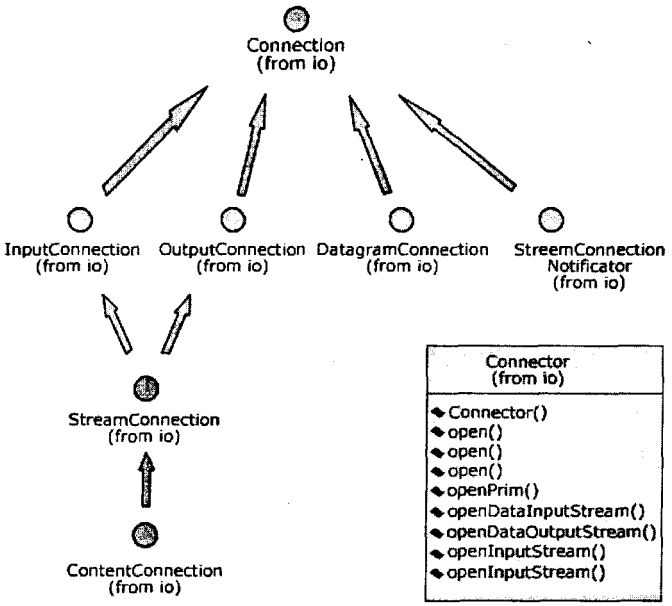

Рис. 3. Діаграма класів GCF

лежить від протоколу, який використовується, <parameters> — параметри з'єднання.

Приклади створення з'єднань:

- ***HTTP***

Connector.open("http://www.foo.com");

- ***Sockets***

*Connector.open("socket://129.**144**.111.222:9000");*

- ***Communication ports***

Connector.open("comm:0;baudrate=9600");

- ***Datagrams***

Connector.open("datagram://: 1234");

- ***Files***

Connector. open("flle:/foo. dat").

При реалізації зв'язку «клієнт — сервер» використовуватимемо push-модель [8]. Зазначимо, що основною функцією push-моделі в комунікаційній задачі є серверне «заштовхування» події в клієнта. Загальний вигляд push-моделі за допомогою sequence-діаграми показано на рис. 4.

Розробимо протокол обміну даними (повідомленнями). Наша специфікація протоколу складатиметься з декількох класів і деякої кількості узгоджень з використання цих класів.

Наведемо діаграму спільних для клієнта і серверу класів, які мають стосунок до протоколу передачі даних (рис. 5).

Message — клас, що інкапсулює усю семантику повідомлення (яке містить дані); Mobile Protocol — інтерфейс, який задає методи, що їх повинен реалізовувати клас, котрий хоче виступати в ролі протоколу кодування в масив байтів (декодування з масиву байтів); MessageReceivedListener— інтерфейс, який повинен реалізовувати клас, в якому повідомляється про надходження нового повідомлення.

SimpleMobileProtocol — клас, який реалізує процес кодування-декодування повідомлень у найпростіший спосіб. Для того, щоб клієнт і сервер спілкувалися однією мовою, вони мають використовувати однаковий протокол обміну. Цьо-

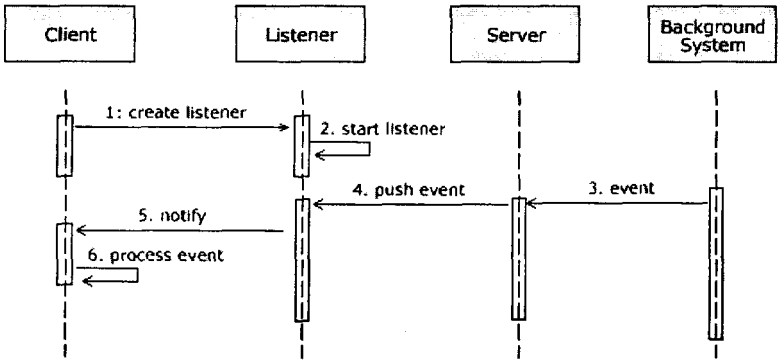

Рис 4. Push-модель

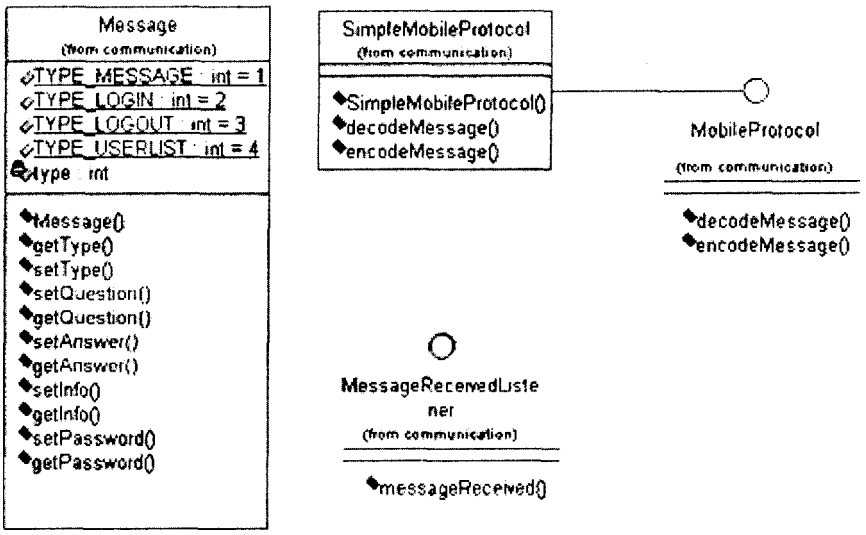

Рис. 5. Діаграма спільних для клієнта і серверу класів

то можна досягнути двома шляхами: клієнт і сервер використовують один і той самий клас, що реалізує протокол; класи різні, але використовують один і той самий формат кодування.

Застосуємо перший варіант. У цьому випадку Simple Mobile Protocol буде мати вигляд:

```
public class Simple MobileProtocol implements MobileProtocol

    public SimpleMobileProtocolO
    {
    }
    public Message decodeMessage(byte[] data) throws IOException
    {
        ByteArraylnputStream bais = new Byte Array InputStream(data);
        DatalnputStream dis = new Data InputStream(bais);
        Message message = new Message();
        message.setType(dis.readlntO);
        message.setQuestion(dis.readUTF());
```

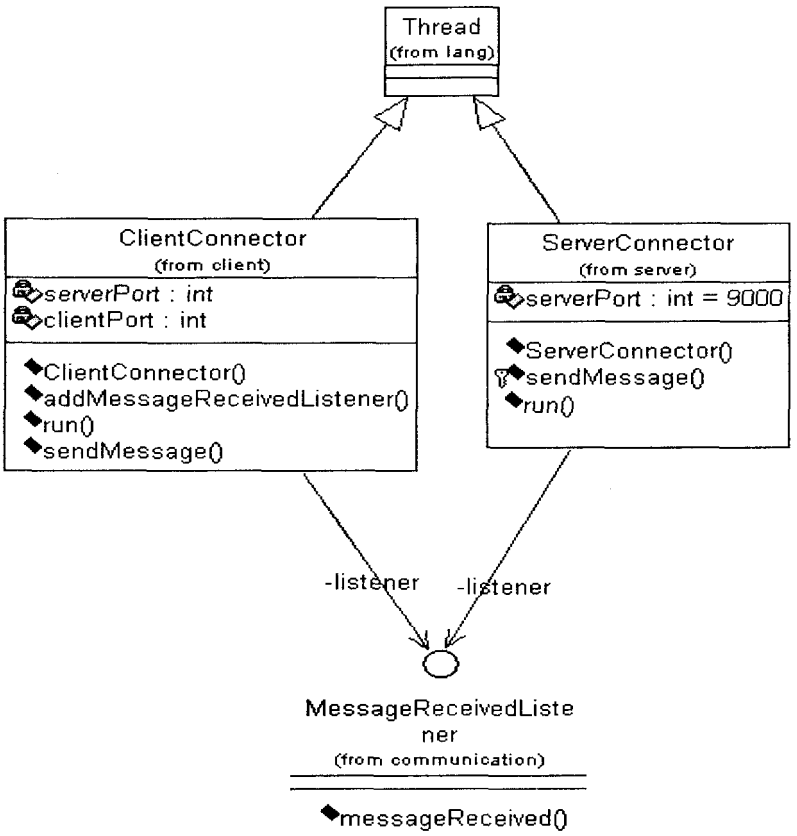

Рис. 6. Діаграма класів клієнтського і серверного конекторів

```
        message.setAnswer(dis.readUTFO);
        message.setPassword(dis.readUTFO);
        message.setlnfo(dis.readUTFO);
        return message;
    }
        public byte[] encodeMessage(Message
message) throws IOException
    {
        ByteArrayOutputStream baos = new
ByteArrayOutputStreamO;
        DataOutputStream das = new
DataOutputStream(baos);
        das.writeInt(message.getType());
        das. write UTF(message.getAnswer());
        das.writeUTF(message.getPassword());
        das.writeUTF(message.getlnfoO);
        return baos.toByteArray();
    }
}
```

Слід зазначити, що SimpleMobileProtocol є частиною всього протоколу обміну і відповідає лише за кодування-декодування повідомлень.

Для серверу і клієнта всі питання обміну повідомленнями винесемо в окремі класи клієнтського і серверного конекторів (рис. 6).

Обидва класи успадковуються від класу Thread (хоча на діаграмі це один і той самий клас, насправді в J2SE і J2ME ці класи відрізняються) і агрегують інтерфейс MessageRecievedListener. Класи мають метод для надсилання повідомлень sendMessage(Message).

Отже, ми вирішили основну проблему, яка постала при розробці системи тестування з використанням МП, а саме проблему передачі й отримання даних з МП в систему КПДН і навпаки. Наступним кроком є розробка повноцінної системи КПДН з використанням МП в ролі додатків, які допоможуть користувачам у роботі з системою.

1. Інститут дистанційного навчання.- http://www.ido.ru
2. Центр «Інформіка».- http://www.informika.ru
3. Євразійська асоціація дистанційної освіти.— http://www.dist-edu.ru
4. *Пирумян В.* Платформа программирования J2ME для портативных устройств,— М., 2003.
5. Connected Limited Device Configuration. Specification \fersion 1.0.- http://www.sun.com/software/communitysource/j2me/cldc/
6. Mobile Information Device Profile (JSR-37). JCP Specification Version 1.0a.— http://www.sun.com/software/communitysource/midp/
7. Java2 Micro Edition.- M.: http://www.javasoft.com/j2me/
8. *Аллен П. Р., Бамбара Д. Дж. и др.* J2EE разработка бизнес-приложений. -«ДиаСофтЮП», 2002.
9. Uniform Resource Identifiers (URI): Generic Syntax.— http://www.ietf.org/rfc/rfc2396.txt
10. Java 2 Platform Micro Edition, Wireless Toolkit.— http://www.javasoft.com/products/j2mewtoolkit/
11. Forte for Java.— http://www.sun.com/forte/ffj
12. Nokia Developer's Suite Beta 0.1 for the Java (TM) 2 Platform, Micro Edition.- http://forum.nokia.com/javaforum/main/1,6668,1_0_30,00.html
13. Bill Day J2ME Archive.— http://www.billday.com/j2me/

A. M. Glybovets

MOBILE DEVICES IN DISTANCE EDUCATION

The article considers base concepts and basic approaches to the necessity of using mobile devices in the remote training computer systems with specially developed, on J2ME (configurations CDLC with the use of structure MIDP), program complexes. Realization was displayed on a testing subsystem. Modeling of the "client-server" communication has been held on a push model. All the questions of the message exchange were inserted separately in the client and server connector classes. The basic problems of data transfer and reception between the mobile device and system have been effectively solved.