Maciej Witos, Adam Pelikant
Wydział Informatyki i Zarządzania
Wyższa Szkoła Informatyki w Łodzi
e-mail: maciej.witos@gmail.com; apelikan@wsinf.edu.pl

PRZETWARZANIE GRAFIKI WEKTOROWEJ W ORACLE

Streszczenie – Artykuł zawiera prezentacje metod pozwalających na konwersje danych, zapisanych w postaci plików grafiki wektorowej do wewnętrznej reprezentacji w bazie danych, w postaci danych typu obiektowego – spatial. Został zrealizowany również mechanizm konwersji odwrotnej pozwalający na eksport danych zapisanych w bazie do postaci zewnętrznego pliku graficznego. Zapewniono wierną rekonstrukcje informacji graficznej.

1 Wprowadzenie

Grafika wektorowa jest rodzajem grafiki komputerowej, w której obraz przechowywany jest za pomocą figur geometrycznych umieszonych w układzie współrzędnych. Figury geometryczne są obiektami opisywanymi przy pomocy pewnych parametrów. W przypadku odcinka takimi parametrami mogą być współrzędne końców odcinka, a w przypadku łuku okręgu: współrzędne środka, długość promienia, kąt początki i kąt końca. Grafika wektorowa może być zapisywana na wiele różniących się sposobów. Jednym ze sposobów zapisywania rysunków grafiki wektorowej jest, stworzony przez firmę Autodesk, format DXF [1].

Grafika wektorowa jest również wykorzystywana w Systemach Informacji Geograficznej do celów analizy i prezentowania danych przestrzennych [3], [4]. W takich systemach do przechowywania informacji można zastosować bazę danych [4], [5], [6]. Jedną z baz danych wspierającą zapisywanie danych przestrzennych jest baza danych Oracle 11g wraz z modułem Oracle Spatial [8], [10].

2 Reprezentacja grafiki w plikach DXF

DXF (Data Exchange Format) jest formatem zapisu grafiki wektorowej przy pomocy pliku tekstowego w kodzie ASCII. Format został stworzony w 1982 roku przez firmę Autodesk w celu umożliwienia wymiany danych między aplikacją AutoCad a programami innych

producentów. Specyfikacja formatu została opublikowana i jest powszechnie dostępna [1].

Plik DXF składa się z par linii, przy czym w linii nieparzystej znajduje się kod opisujący znaczenie wartości w znajdującej się niżej linii parzystej. Używając tych par plik DXF jest złożony z sekcji, które składają się z kodów i wartości przez nieopisanych. Wyróżnia się następujące sekcje pliku DXF [1]:

- HEADER – zawiera zestaw metadanych opisujących rysunek np.: program, w którym utworzono plik.
- CLASSES – zawiera dane klas zdefiniowanych w aplikacji, których instancje występują w innych sekcjach pliku.
- TABLES – zawiera zestaw danych tabelarycznych opisujących domyślne elementy rysunku takie jak np.: typy linii czy warstwy.
- BLOCKS – definicje bloków rysunku.
- ENTITIES – opis wszystkich obiektów mających reprezentację graficzną np.: linia, polilinia, punkt.
- OBJECT – opis obiektów nieposiadających interpretacji graficznej.
- END OF FILE – koniec pliku DXF.

Tabela. 1. Plik opisujący okrąg z opisem znaczenia istotnych znaczników i ich wartości.

Zawartość pliku	**Opis pozycji w pliku**
0	Znacznik typu obiektu
CIRCLE	Nazwa typu obiektu
5	Unikalny uchwyt do obiektu
188	
330	
1F	
100	
AcDbEntity	
8	
0	
100	
AcDbCircle	
10	Znacznik współrzędna X środka
10.0	Współrzędna X środka
20	Znacznik współrzędna Y środka
15.0	Współrzędna Y środka
30	Znacznik współrzędna Z środka
0.0	Współrzędna Z środka
40	Znacznik promień
5.0	Wartość promienia

Najistotniejszą i jedyną obowiązkową jest sekcja ENTITIES. Zawiera ona opis elementów wchodzących w skład rysunku i podlega przetwarzaniu w programie będącym przedmiotem tego artykułu. Przykładowa struktura piku DXF jest przedstawiona w tabeli 1, na przykładzie pliku przechowującego informacje o okręgu o środku w punkcie (10,15) i promieniu r=5.

3 Reprezentacja grafiki w bazie danych Oracle

Zapis grafiki wektorowej w bazie danych Oracle 11g jest możliwy dzięki modułowi Oracle Spatial [9], [12]. Moduł ten umożliwia zarządzanie danymi geograficznymi i jednym z jego zadań jest udostępnianie struktury danych do przechowywania obiektów grafiki wektorowej. Obiekty te zapisywane są w tabeli bazy danych w kolumnie typu SDO_GEOMETRY. Tabele zawierające obiekty typu SDO_GEOMETRY muszą posiadać unikalne klucze główne [12].

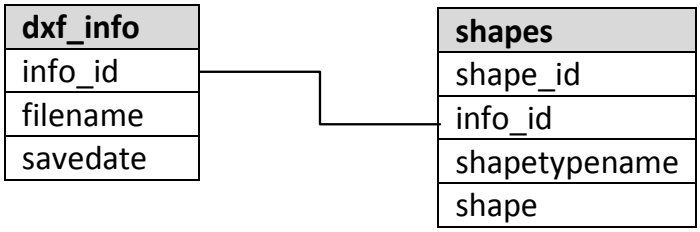
Oracle Spatial definiuje typ SDO_GEOMETRY w następujący sposób:

```
CREATE TYPE sdo_geometry AS OBJECT (
SDO_GTYPE NUMBER,
SDO_SRID NUMBER,
SDO_POINT SDO_POINT_TYPE,
SDO_ELEM_INFO SDO_ELEM_INFO_ARRAY,
SDO_ORDINATES SDO_ORDINATE_ARRAY
);
```

gdzie:

- SDO_GTYPE – typ obiektu
- SDO_SRID – rodzaj układu współrzędnych przypisanego do obiektu – w programie będącym przedmiotem pracy SDO_SRID jest zawsze równe null
- SDO_POINT – pozwala utworzyć obiekt typu punkt. Ponieważ niniejsza praca nie zakłada tworzenia pojedynczych punktów, wartość tego parametru zawsze wynosi null.
- SDO_ELEM_INFO – tablica zawierająca informacje o elemencie, określa, w jaki sposób należy interpretować wartości zawarte w tablicy SDO_ORDINATES
- SDO_ORDINATES – tablica współrzędnych elementu.

Baza danych, z którą komunikuje się program, składa się z dwóch tabel rys. 1: DXF_INFO oraz SHAPES. Pierwsza z nich przechowuje nazwę pliku, z którego zapisana została grafika oraz datę dokonania zapisu. Pozwala to rozróżniać zapisane rysunki w bazie danych. Tabela SHAPES zawiera informacje o obiektach graficznych.



Rys. 1. Schemat bazy danych

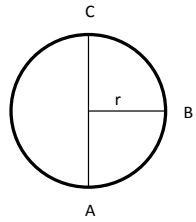
4 Reprezentacja obiektów grafiki w programie

Za reprezentację obiektów graficznych w programie odpowiedzialne są klasy z pakietu shapes. Klasa Shape jest klasą abstrakcyjną nie zawierającą żadnych metod ani atrybutów. Wszystkie klasy opisujące konkretne obiekty kształtów dziedziczą po klasie Shape [7]. Takie rozwiązanie umożliwia, dzięki wykorzystaniu mechanizmu polimorfizmu, przechowywanie wszystkich obiektów graficznych w jednym kontenerze typu ArrayList<Shape>. Dzięki temu rozwiązaniu obiekty graficzne są przekazywane między poszczególnymi modułami programu w przejrzysty i łatwy do zrozumienia sposób [7], [11]. Kontener ArrayList<Shape>, nazywany dalej w pracy „listą kształtów" jest tworzony przez obiekty odczytujące dane z bazy danych i z pliku, a następnie przekazywany do odpowiednich obiektów zapisujących. Zarówno w przypadku zapisu z pliku do bazy jak i z bazy do pliku komunikacja odbywa się w jednakowy sposób.

W skład pakietu shapes wchodzą następujące klasy:

- Arc – klasa opisująca łuk okręgu
- Line – klasa opisująca odcinek
- Polyline – klasa opisująca linię składającą się z wielu odcinków
- Circle – klasa opisująca okrąg.

Klasa Circle opisuje obiekty okręgu. Standard DXF zakłada opis okręgu przy użyciu współrzędnych środka i długości promienia baza danych, Oracle natomiast do zdefiniowania okręgu używa współrzędnych trzech punktów znajdujących się na łuku. Z powodu różnic w zapisie klasa implementuje metody umożliwiające przechodzenie między dwoma sposobami definicji. W programie definiowane są współrzędne trzech punktów należących do okręgu, wybrane w sposób pokazany na rysunku:



Rys. 2. Wybór punktów na okręgu

Klasa posiada trzy konstruktory:

- public Circle(double x, double y, double r)
- public Circle(double[coordinates)
- public Circle()

Pierwszy z konstruktorów inicjalizuje odpowiednimi wartościami współrzędne punktu środka okręgu oraz długość promienia.

Drugi konstruktor, jako parametr przyjmuje tablicę współrzędnych następującej postaci:

{XA, YA, XB, YB, XC, YC}

Tablica ta zawiera współrzędne punktów należących do okręgu wybranych tak, jak przedstawiono na rysunku. Konstruktor na podstawie otrzymanej tablicy oblicza współrzędne punktu środka okręgu i długość promienia i inicjalizuje odpowiednie właściwości. Taka postać konstruktora ułatwia współpracę z obiektem klasy DbReader.

Trzeci nie przyjmuje żadnych parametrów i tworzy obiekt z niezainicjalizowanymi współrzędnymi. Taki sposób tworzenia obiektu zakłada późniejsze przypisanie wartości do poszczególnych współrzędnych.

Oprócz konstruktorów i standardowych metod dostępowych klasa posiada metodę getOrdinates(), która zwraca referencję do tablicy współrzędnych. Tablica jest tworzona na podstawie wartości współrzędnych punktu środka okręgu i długości promienie. Zwracana tablica jest takiej postaci, jaką wymaga baza danych Oracle i opisuje punkty przedstawione na rysunku.

5 Zapis grafiki wektorowej z pliku do bazy danych

Proces przepisania informacji opisującej grafikę wektorową do bazy danych podzieliłem na dwa niezależne etapy:

- odczytu danych z pliku DXF
- zapisu danych do bazy danych Oracle

Za etap pierwszy odpowiadają obiekty klas DxfFileReader i Interpreter, za etap drugi jeden obiekt klasy DbWriter. W przypadku odczytu danych z pliku obiekt klasy DxfFileReader odpowiada jedynie za otwarcie, odczytanie wskazanego pliku i przekazaniu jego zawartości do in-

terpretera. Za odpowiednią interpretację treści odczytanego pliku odpowiada obiekt klasy Interpreter.

Klasa DxfInterpreter posiada tylko jedną metodę go. Celem tej metody jest zinterpretowanie zawartości pliku DXF. Metoda ta, jako wynik działania zwraca kontener ArrayList zawierający elementy typu Shape. Obiekt klasy DxfInterpreter jest tworzony wewnątrz obiektu DxfFileReader. Najpierw fileReader odczytuje plik, a następnie odczytaną zawartość przekazuje do metody go() interpretera.

Pierwszą czynnością wykonywaną wewnątrz metody jest utworzenie kontenera ArrayList<Shape> do przechowywania odczytanych elementów graficznych. Następnie pusta pętla while odczytuje kolejne linijki z bufora tak długo, aż natknie się na słowo kluczowe ENTITIES oznaczające początek sekcji zawierającej informacje o obiektach graficznych. Od tego momentu zaczyna się właściwa interpretacja pliku. Odczytywana są dwie kolejne linie z sekcji ENTITIES. Pierwsza linia zawiera zawsze kod odczytywanego parametru oznaczający jego zastosowanie, druga linia zaś zawiera wartość tego parametru. Interpretacja trwa tak długo, aż osiągnięty zostanie znacznik końca sekcji ENDSEC. Pierwszym kodem odczytanym będzie 0, oznaczające, że w następnej linijce znajduje się informacja o rodzaju elementu graficznego. Ponieważ dla każdego elementu sekcji ENTITIES operacja odczytu wygląda analogicznie – zmieniają się jedynie kody parametrów – w niniejszej pracy opisany zostanie tylko przypadek odczytu obiektu odcinka. Jeżeli wartość odczytanego parametru jest ciągiem znaków LINE tworzony jest nowy obiekt klasy Line. Następnie odczytywane są kolejne kody parametrów i ich wartości. W zależności od kodu parametru dla nowego obiektu linii ustawiane są odpowiednie atrybuty.

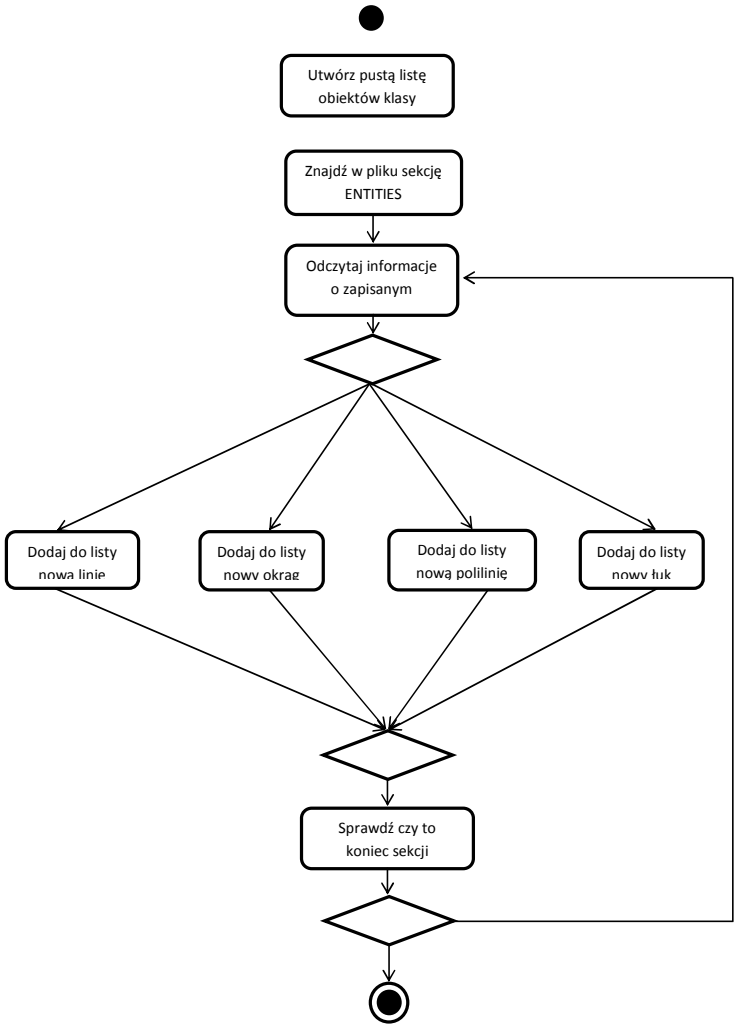
Kolejne informacje o elemencie w pliku są poszukiwane tak długo, aż znaleziony zostanie kod znacznika 0 oznaczający początek następnego elementu graficznego. Gdy już wszystkie atrybuty obiektu Line są ustawione, obiekt ten zostaje dodany do utworzonego kontenera. Kiedy wszystkie elementy zostaną odczytane, osiągnięty zostaje koniec sekcji i metoda zwraca zapełniony kontener.

Metoda write jest jedyną publiczną metodą klasy DbWriter (poza konstruktorem). Jest to główna metoda klasy, której zadaniem jest zapisanie listy kształtów do bazy danych. Metoda przyjmuje dwa argumenty: listę kształtów i nazwę pliku źródłowego. Na podstawie nazwy pliku i aktualnej daty metoda tworzy unikalny wpis w tabeli DXF_INFO. W tym celu najpierw tworzony jest ciąg znaków reprezentujących zapytanie, a następnie następuje próba wywołania zapytania.

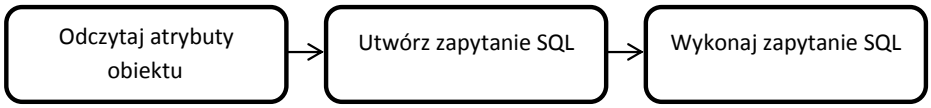
Gdy operacja nie powiedzie się, treść wyjątku zostaje wysłana na domyślne wyjście. Gdy zostanie utworzony odpowiedni wpis w DXF_INFO metoda przechodzi do najważniejszej czynności, czyli zapi-

sania obiektów do bazy. Dla każdego obiektu typu shape w tablicy sprawdzany jest typ obiektu. Po sprawdzeniu typu, obiekt jest rzutowany w dół na swój typ, a następnie zostaje wywołana jedna z metod prywatnych zapisująca kształt do bazy danych. Wszystkie metody zapisujące do bazy danych działają według schematu przedstawionego na rys. 4.

Podobnie jak zapis z pliku do bazy, zapis z bazy do pliku przebiega w dwóch etapach. Najpierw następuje odczyt z bazy danych, a następnie zapis do pliku. Za pierwszy etap odpowiada obiekt klasy *DbReader*, za drugi zaś obiekt klasy *DxfFileWriter*.



Rys. 3. Ogólny schemat klasy DxfInterpreter



Rys. 4. Ogólny schemat zapisu do bazy danych

Dla każdego rekordu w resultSet wykonywany jest szereg operacji, mający na celu utworzenie odpowiedniego obiektu kształtu jednego z typów dziedziczących po klasie Shape:

- z obiektu resultSet przy użyciu metody getObject odczytywany jest obiekt SDO_GEOMETRY i rzutowany na typ STRUCT
- z utworzonego obiektu STRUCT tworzony jest nowy obiekt klasy JGeometry zdefiniowany w programie, jako jGeo. Obiekt ten jest wykorzystywany do uzyskania informacji na temat obiektu graficznego. Pierwszą informacją jest informacja o typie obiektu mówiąca, czy należy go traktować jako linię, polilinię czy wielokąt. W drugiej kolejności interpretowana jest informacja o elemencie pozwalająca określić, czy linia jest prostą czy łukiem oraz czy wielokąt jest okręgiem gdyż okręgi w obiekcie SDO_GEOMETRY zapisywane są jako wielokąty.
- na podstawie uzyskanych informacji tworzony jest odołowieni obiekt kształtu i dodawany do listy kształtów

Gdy już wszystkie obiekty zostaną zapisane do listy, metoda zwraca zapełnioną listę kształtów.

Schemat ilustruje sposób, w jaki na podstawie informacji o typie SDO_GTYPE oraz informacji o elemencie SDO_ELEM_INFO podejmowane są decyzje o tworzeniu konkretnych typów obiektów kształtów.

6 Wnioski

Artykuł stanowi opis systemu umożliwiającego wymianę danych między formatem DXF i bazą danych Oracle 11g. Zaprezentowany został pokrótce sposób przechowywania informacji w obu przytoczonych formatach, jak i w programie będącym przedmiotem pracy. Przedstawiony został najpierw ogólny zarys zasady działania programu, a następnie szczegółowy opis poszczególnych modułów. Na schematach akcji została, na wysokim poziome abstrakcji, przedstawiona zasada działania modułów.

Działający i funkcjonalny program dowodzi realizacji celu pracy i spełnienia założeń. Dowodzi to faktu, że możliwa jest pełna automatyzacja procesu wymiany danych pomiędzy formatem DXF a sposobem zapisu stosowanym w bazie danych Oracle 11g. Pozwala to na przecho-

wywanie po stronie bazy danych nie tylko grafiki wektorowej w postaci odnośnika do pliku, ale również składowych kształtów. Daje to możliwość elastycznej wymiany zdefiniowanych elementów między różnymi plikami graficznymi. Korzystając z wbudowanych cech bazy danych dostępne są wszystkie operacje wyszukiwania. Stworzona aplikacja wiernie odtwarza odczytywane i zapisywane obiekty. W bazie danych obiekty formatowane są poprawnie i można na nich wykonywać wszystkie metody udostępnione przez typ Geometry.

7 Literatura

[1] DXF Reference" Autodesk 2011

[2] OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture

[3] OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option

[4] Rigaux P., Scholl M., Voisard A., *Representation Of Spatial Objects, Spatial Databases With Application to GIS* 2002 Elsevier Inc,

[5] van Oosterom P.J.M., Lemmen C.H.J., *Spatial data management on a very large cadastral database.* Computers, Environment and Urban Systems, Volume 25, Issues 4-5, July-September 2001, Pages 509-528

[6] Brent Hall G., *Spatial Database Systems: Design, Implementation and Project Management*.2009

[7] Eckel B., *Thinking In Java.* Fourth Edition, Helion, Gliwice 2006

[8] Rich K., Stern J., Fogel S., *McGregor C., Oracle Database 2 Day DBA 11g Release 2*, Oracle, 2010

[9] Pelikant A., *Bazy danych w zastosowaniach praktycznych – Monografia,* WSInf., 2007

[10] Lance A., Kyte T., *Oracle Database Concepts 11g Release 2*. WWW, 2010

[11] Kuassi M., *Oracle Database Programming Using Java and Web Services.* Elsevier Digital Press, 2006

[12] Pelikant A., *Programowanie serwera Oracle 11g SQL i PL/SQL*. Helion, 2009

VECTOR GRAPHICS PROCESSING IN ORACLE

Summary - The paper contains presentations of methods allowing for conversions of data saved in the form of vector graphic files to the internal representation of a database, spatial data type. Was realized also the mechanism of the reverse conversion which allows the export of the data stored in the database to an external image file. Ensured complete reconstruction of graphics information.