STUDIA INFORMATICA 2011
Volume 32 Number 2B (97)

Kacper PABJAŃCZYK, Adam PELIKANT
Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych

IMPLEMENTACJA PARSERA PROTOKOŁU DICOM, PRZY UŻYCIU JĘZYKA T-SQL NA PLATFORMIE SQL SERVER 2008

Streszczenie. Artykuł dotyczy metody implementacji parsera pliku DICOM z wykorzystaniem języka T-SQL na platformie SQL Server 2008. Protokół DICOM – Digital Imaging and Communications in Medicine (Obrazowanie Cyfrowe i Wymiana Obrazów w Medycynie) – jest normą opracowaną przez ACR/NEMA (American College of Radiology/National Electrical Manufacturers Association) dla potrzeb ujednolicenia, wymiany [3] i interpretacji danych medycznych [9], reprezentujących lub związanych z obrazami diagnostycznymi w medycynie. Znajduje on zastosowanie przy przetwarzaniu cyfrowo zapisanych danych z urządzeń diagnostycznych [6], takich jak: tomograf komputerowy, pozytronowy tomograf emisyjny, ultrasonograf. Aktualnie implementowana jest trzecia wersja standardu, która została po raz pierwszy ogłoszona w 1992 roku i od tego czasu jest ciągle aktualizowana. Praca opisuje notację stosowaną w tym formacie. W zasadniczej części przedstawiono dwie metody dekompozycji plików do schematu relacyjnego z zastosowaniem rozszerzenia proceduralnego oraz omówiono ich wydajność.

Słowa kluczowe: DICOM, SQL Server 2008, T-SQL

IMPLEMENTATION OF DICOM PARSER USING T-SQL ON SQL SERVER 2008

Summary. This article applies the method of implementing DICOM parser file using T-SQL in SQL Server 2008. DICOM Protocol – Digital Imaging and Communications in Medicine (Digital Imaging and Exchange of Images in Medicine) is a standard developed by the ACR / NEMA (American College of Radiology / National Electrical Manufacturers Association) for the unification, exchange [3] and interpretation [9] of medical data representing or associated with diagnostic images in medicine. It is used in the processing of digitally stored data from diagnostic devices [6] such as a computer tomography, positron emission tomography, and diagnostic sonography. Currently implemented is the third version of the standard, which was announced for the first time in 1992 and since that time is constantly being updated.

Work describes notation used in this format. In the principle part two methods of decomposition of files to relational schema using procedural extension was introduced and discus their efficiency.

Keywords: DICOM, SQL Server 2008, T-SQL

1. Wstęp

Prace poświęcone protokołowi oraz formatowi DICOM są prowadzone od lat przez wiele ośrodków akademickich. Można w nich wyróżnić kilka istotnych nurtów. Pierwszym jest zastosowanie tego protokołu do integracji danych medycznych pochodzących z różnych skanerów [6, 7, 8]. Głównymi problemami poruszanymi w tych badaniach są wydajność procesu transmisji i przetwarzania, spójność i zgodność ze standardem danych pochodzących z różnych urządzeń oraz sposoby przechwytywania i inicjowania połączeń. Pochodną tego nurtu jest tworzenie narzędzi do dekompozycji danych w formacie danych do jednolitego schematu relacyjnego, głównie nastawionych na oddzielenie części zawierającej atrybuty objaśniające (alfanumeryczne) od części obrazowej [8, 9]. Kolejny z nurtów zajmuje się zastosowaniem narzędzi przetwarzania obrazów (sygnałów) do analizy części obrazowej pliku [10]. Istnieje wiele prac poświęconych wykorzystaniu narzędzi integracyjnych w służbie zdrowia, zajmujących się problematyką od strony zarządzania i ochrony informacji [12] oraz analiz czysto medycznych. Przytoczone odwołania do publikacji stanowią nieznaczną część prac poświęconych badaniom nad DICOM.

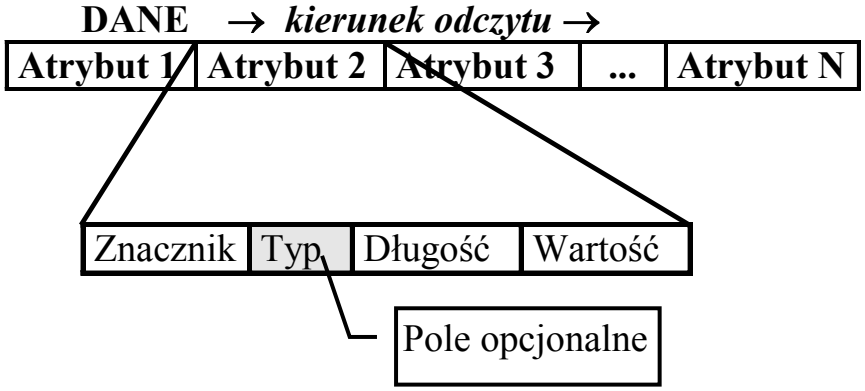
W naszym ośrodku były również prowadzone prace dotyczące w większości zagadnień integracji [4, 5, 11]. W wyniku których zbudowano środowisko z centralną bazą danych ORACLE, dla potrzeb zarządzania danymi medycznymi w szpitalu klinicznym w Aarhus. Głównym problemem było tam zapewnienie maksymalnej wydajności gromadzenia danych, ze względu na bardzo dużą liczbę obrazów pochodzących z jednej serii badań FMRI (Functional Magnetic Resonance Imaging). Zajmowano się również problematyką wyekstrahowania części graficznej plików DICOM i ich konwersji do zewnętrznych formatów graficznych. W innym projekcie zrealizowano szybki mechanizm diagnostyczny dla objawu Raynauda, korzystający z tego samego formatu i bazy ORACLE. Korzystanie z tego środowiska serwera bazy danych znacznie ułatwia pracę ze względu na to, że zawiera on wbudowane narzędzia Oracle Multimedia/InterMedia (ORDSYS.ORDDicom). Obiekt ten zawiera wiele klas pozwalających na dość zaawansowaną obsługę danych, jednak większość środowisk serwerów baz danych nie zawiera tego typu narzędzi – dotyczy to również MS SQL Server. Głównym zadaniem było więc opracowanie zestawu procedur napisanych po stronie serwera bazy danych z zastosowaniem rozszerzenia proceduralnego T-SQL, które

dawałoby podobne funkcjonalności. Podstawą tego systemu jest sprawna dekompozycja danych wejściowych DICOM do schematu relacyjnego. Prezentowane w dalszej części rozwiązanie jest fragmentem szerszego projektu, prowadzącego do budowy podobnych narzędzi w środowisku bazy Microsoft.

2. Opis protokołu DICOM

2.1. Budowa pliku DICOM

Z punktu widzenia formalnego pliki standardu DICOM są klasycznymi plikami znacznikowymi. W odróżnieniu od większości są zapisywane w notacji hexadecymalnej, a nie klasycznej, znakowej. Przykład budowy fragmentu takiego pliku zawiera rys. 1.



Rys. 1. Budowa ciągu informacji w pliku DICOM
Fig. 1. Construction of the information in the DICOM file

Widoczny na rys. 1 ciąg danych składa się z kolejnych atrybutów (Data Elements).
W skład opisu pojedynczego atrybutu wchodzą:

- Znacznik (Tag) – 8-cyfrowy kod zawartości elementu,
- Typ (VR – Value Representation) – reprezentacja wartości wg standardu użytego w pliku DICOM,
- Długość (Value Length) – liczba znaków, na których zapisano wartość pola (opcjonalnie),
- Wartość (Value Field) – dla typów danych prostych – wartość, dla obrazów – ich reprezentacja bitowa.

Tabela 1
Przykładowy pojedynczy Data Element w pliku DICOM

4 Bajty	2 Bajty	2 Bajty	12 Bajtów
0010 0010	**PN**	**0C**	**Jan Kowalski**
Tag	VR	VL	Value Field

W tabeli 1 znacznikiem (Tag) jest ciąg cyfr 0010 0010, którego pierwsze cztery cyfry (czyli 0010) oznaczają, że są to informacje dotyczące pacjenta. Kolejne cztery dają już pełny numer znacznika. Znacznik 0010 0010 oznacza imię i nazwisko pacjenta – zgodnie z opisem wszystkich znaczników, który można znaleźć w dokumentacji standardu DICOM w rozdziale 6 „Data Dictionary" [1, 2]. Tabela z opisem znaczników jest jedną z tabel słownikowych zastosowanych w stworzonej aplikacji do dekompozycji plików. Następną informacją jest reprezentacja wartości (Value Representation). W przykładowym ciągu jest to PN, czyli Patient Name. Kolejny fragment zawiera informacje o długość ciągu znaków zawierających wartość. W przykładzie 0C oznacza zapis informacji przy użyciu 12 znaków. Ostatnią częścią jest pole (Value Lenght), zawierające już konkretną wartość zapisanej w tym standardzie danej.

2.2. Sposób zapisu

Sposób odczytywania informacji jest zależny od formatu zapisu danych. Może być zastosowany format Big Endian lub Little Endian. Sposób kodowania jest zapisany w znaczniku 0002 0010. Wartość 1.2.840.10008.1.2 wskazuje na Little Endian (domyślny sposób zapisu), natomiast 1.2.840.10008.1.2.2 wskazuje na Big Endian. Pierwszy z formatów powoduje zapisanie znaczników i długości pola w odwrotnej kolejności bajtów (pierwszy młodszy bajt). Przykład takiego kodowania znajduje się w tabeli 2.

Tabela 2

Różnice kodowania w formatach Little Endian i Big Endian

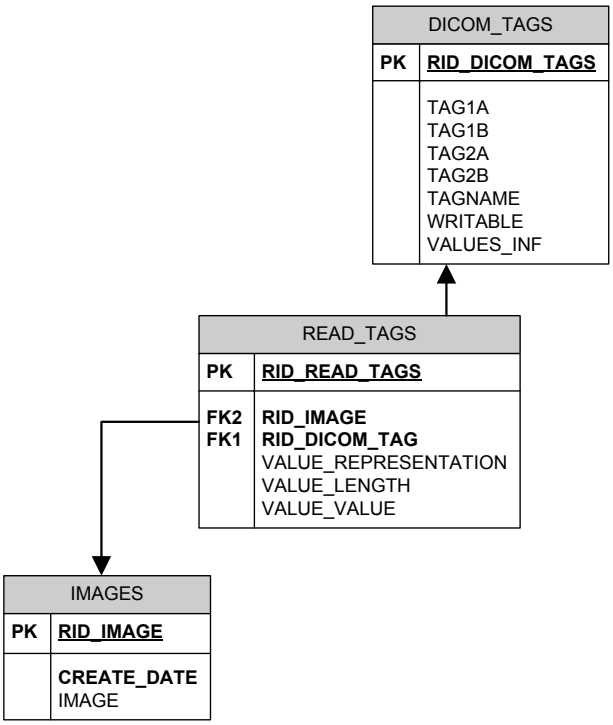
	Tag 1		Tag 2		VR		VL		Value Field																		
Little Endian	02	00	10	00	55	49	14	00	31	2E	32	2E	38	34	30	2e	31	30	30	30	38	2E	31	2E	32	2E	31	00
Big Endian	00	02	00	10	55	49	00	14	31	2E	32	2E	38	34	30	2e	31	30	30	30	38	2E	31	2E	32	2E	31	00
Ciąg do interpretacji	00	02	00	10	55	49	00	14	31	2E	32	2E	38	34	30	2e	31	30	30	30	38	2E	31	2E	32	2E	31	00
Interpretacja	00	02	00	10	U	I	20		1	.	2	.	8	4	0	.	1	0	0	0	8	.	1	.	2	.	1	

Jak widać w tabeli 2, wybór formatu zapisu jest zauważalny w trzech pierwszych znacznikach. Dokumentacja standardu zakłada również taką sytuacje, w której mimo zastosowania Little Endian część znaczników jest zapisana w Big Endian [2]. Fakt ten znacznie komplikuje budowę uniwersalnego algorytmu dekompozycji plików DICOM. Dodatkowy problem stanowi brak w niektórych ciągach wartości atrybutu opisującego długość pola zawierającego wartość.

3. Odczytywanie danych z pliku DICOM w SQL

3.1. Model bazy danych

Jako miejsce przechowywania danych została wybrana baza danych SQL Server 2008 w wersji Developer. Część bazy danych, która odpowiada za składowanie zarówno nieprzetworzonych, jak i przetworzonych plików, została przedstawiona na rys. 2 w formie diagramu tabel. Na diagramie znajdują się trzy tabele, DICOM_TAGS, READ_TAGS oraz IMAGES.



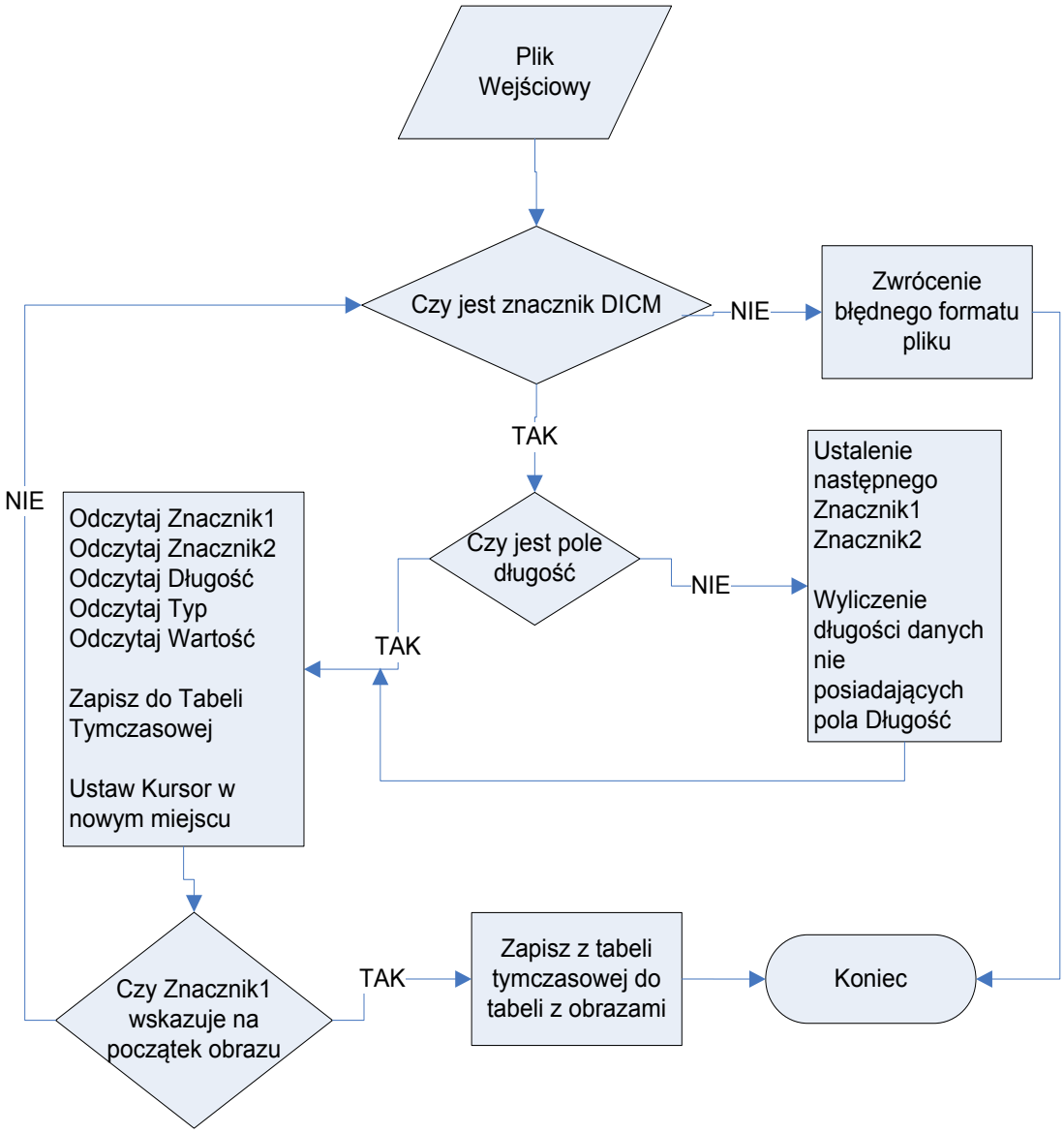
Rys. 2. Struktura bazy danych
Fig. 2. Database structure

- DICOM_TAGS zawiera wszystkie znaczniki występujące w protokole DICOM. Pola TAG1A, TAG1B, TAG2A oraz TAG2B są polami heksadecymalnymi, w każdym z nich jest przechowywana jedna liczba, dopiero konkatenacja wszystkich pól daje wgląd w pełny znacznik. Podział znaczników na pojedyncze pola jest podyktowany tym, że w zależności od wybranego systemu zapisu (Big Endian lub Little Endian), znaczniki w zdjęciu mogą być zapisanie w odwrotnej kolejności niż używany na co dzień sposób zapisu liczb. Umieszczenie całego znacznika w jednym polu utrudniłoby porównywanie z danymi przychodzącymi z pliku. Pole TAGNAME zawiera angielską nazwę znacznika.

- READ_TAGS zawiera dane, które zostaną odczytane ze zdjęcia DICOM. Tabela zawiera klucze obce RID_Image (wskazuje, którego zdjęcia to są dane) oraz RID_DICOM_TAGS (wskazuje znacznik z tabeli DICOM_TAGS).
- IMAGES zawiera nieprzetworzone dane. Dane są zapisane w kolumnie Image i przechowywane jako ciąg bajtów VARBINARY.

3.2. Odczytywanie danych

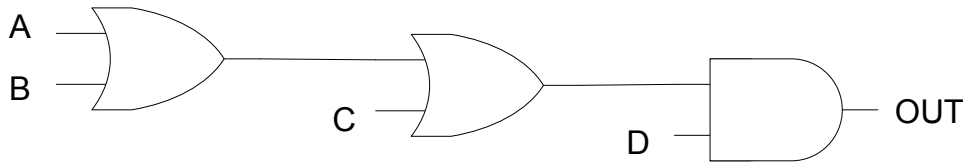
Algorytm pokazany na rys. 3 przedstawia w uproszczony sposób odczyt danych zapisanych w formacie DICOM.



Rys. 3. Algorytm odczytu danych z pliku DICOM
Fig. 3. The algorithm to read data from DICOM file

Pierwszym krokiem jest odnalezienie w pliku ciągu znaków „DICM", który wskazuje na zastosowanie formatu DICOM, jednocześnie wskazując koniec nagłówka i rozpoczynając zasadniczą część zawierającą dane. Bezpośrednio za nim znajduje się pierwszy znacznik. Po pobraniu lokalizacji pierwszego znacznika procedura odczytuje kolejne dane.

Odnalezienie i odczytanie wszystkich znaczników jest prostym zadaniem, jeśli w opisie pojedynczego atrybutu znajduje się pole określające długość wartości (VL). Znając tę wartość, można wyliczyć długość pojedynczego zbioru danych, a co za tym idzie określić położenie następnego znacznika, w przeciwnym przypadku analiza pliku znacznie się komplikuje. Sposób rozwiązania tego problemu zaprezentowano w postaci schematu blokowego na rys 3. Natomiast rys. 4 pokazuje schemat logiczny algorytmu realizującego ten fragment schematu.



Rys. 4. Logika zastosowana przy wyszukiwaniu „Value Length"
Fig. 4. The logic used when searching for "Value Length"

Od strony programistycznej rozwiązanie to zawiera kod implementujący logikę z rys. 4.

```
WHILE (((
(SELECT (SUBSTRING(Image,@C+1,1) + SUBSTRING(Image,@C,1)) AS a FROM
Images WHERE RID_IMAGE=@ID_OBRAZU)
!=
(SELECT (SUBSTRING(Image,@A+1,1) + SUBSTRING(Image,@A,1)) FROM Images
WHERE RID_IMAGE=@ID_OBRAZU)
OR
(SELECT CONVERT(VARBINARY(2), (SUBSTRING(Image,@C+1,1) +
 SUBSTRING(Image,@C,1))+7) FROM Images WHERE RID_IMAGE=@ID_OBRAZU)
<
(SELECT (SUBSTRING(Image,@A+1,1)+SUBSTRING(Image,@A,1)) FROM Images
WHERE RID_IMAGE=@ID_OBRAZU))
OR
(SELECT (SUBSTRING(Image,@C+3,1) + SUBSTRING(Image,@C+2,1)) AS b FROM
Images WHERE RID_IMAGE=@ID_OBRAZU)
<
(SELECT (SUBSTRING(Image,@A+3,1) + SUBSTRING(Image,@A+2,1)) FROM Images
WHERE RID_IMAGE=@ID_OBRAZU))
AND
(SELECT CONVERT(VARCHAR, SUBSTRING(Image,@C+4,2)) AS a FROM Images WHERE
RID_IMAGE=@ID_OBRAZU) != 'UL')
```

Powyższy kod jest fragmentem procedury składowanej, służącej do pobierania danych i dekapsulacji plików DICOM:

- Images – tabela przechowująca pliki DICOM,
- Image – kolumna VARBINARY, przechowująca pliki DICOM w formacie binarnym,
- RID_IMAGE – identyfikator wiersza w tabeli Images,

- @ID_OBRAZU – parametr procedury składowanej, określający identyfikator pliku do pobrania.

Sprawdzane po kolei warunki logiczne można zapisać w poniższej postaci.

$$A \Rightarrow VL(poz(y+1)(y)) <> Tag1\_bież(poz(x+1)(x))$$
$$B \Rightarrow VL(poz(y+1)(y)) + 7 < Tag1\_bież(poz(x+1)(x))$$
$$C \Rightarrow Tag2\_nast(poz(y+3)(y+2)) < Tag2\_bież(poz(x+3)(x+2))$$
$$D \Rightarrow VR\_nowa(poz(y+4)(y+5)) <> 'UL'$$

Drugi sposób rozwiązania problemu zakłada stworzenie tablicy ze wszystkimi znacznikami oraz rodzajami reprezentacji wartości. Procedura zakłada następujące kroki:

- wykrycie braku długości wartości (VL),
- pobranie 6 kolejnych wartości,
- porównanie pierwszych 4 z tabelą znaczników,
- gdy znacznik się zgadza, sprawdzenie 5 i 6 wartości, czy są reprezentacją wartości,
- przy braku trafienia, przesunięcie sprawdzanych wartości o jedno miejsce.

Metoda ta wydaje się dosyć prosta i pewna, jednak w trakcie jej implementacji okazało się, że nie jest dostępny plik np. w formacie csv lub txt, zawierający wszystkie znaczniki DICOM, dlatego konieczne okazało się ręczne utworzenie tabeli na podstawie ogólnodostępnego spisu z domowej strony dla standardu DICOM [1].

Kolejnym problemem była zamiana pól dwóch pierwszych znaczników na pola heksadecymalne. Baza danych traktowała te pola jako pola tekstowe, a zmiana typu kolumny na varbinary kończyła się zamianą każdego znaku znacznika na odpowiednik heksadecymalny z tablicy ASCII. Rozwiązaniem tego problemu było stworzenie skryptu generującego tabelę, w której znajdowały się m.in. kolumny o wartościach od 0 do 255, gdzie jedna kolumna była wartością heksadecymalną w polu varbinary, a druga wartością heksadecymalną w polu tekstowym. Dzięki tej tabeli można było stworzyć główną tabelę ze wszystkimi znacznikami DICOM.

Drugi sposób różni się od pierwszego tylko i wyłącznie rozwiązaniem problemu wyszukiwania znacznika nieposiadającego wartości Value Lenght. Poniżej został zamieszony zmieniony fragment kodu:

```
SELECT @TAG=SUBSTRING(Image,@C+1,1) FROM Images WHERE RID_IMAGE=@ID_OBRAZU
SELECT @TAG2=SUBSTRING(Image,@C,1) FROM Images WHERE RID_IMAGE=@ID_OBRAZU
SELECT @TAG3=SUBSTRING(Image,@C+3,1) FROM Images WHERE RID_IMAGE=@ID_OBRAZU
SELECT @TAG4=SUBSTRING(Image,@C+2,1) FROM Images WHERE RID_IMAGE=@ID_OBRAZU
SELECT @TAGVR=SUBSTRING(Image,@C+4,2) FROM Images WHERE RID_IMAGE=@ID_OBRAZU
IF (EXISTS (SELECT * FROM DICOM_TAGS
WHERE TAG1A=@TAG AND TAG1B=@TAG2 AND TAG2A=@TAG3 AND TAG2B=@TAG4) AND
((@TAG>=@TAG5 AND @TAG2>@TAG6)
OR (@TAG>@TAG5 AND @TAG3<@TAG7)
OR (@TAG=@TAG5 AND @TAG2=@TAG6 ) ) )
OR (@TAG=0x7F AND @TAG2=0xE0)
```

gdzie `@C` jest parametrem określającym aktualne położenie w odczytywanym pliku.

W tym rozwiązaniu zamiast zastosowania logiki w wyszukiwaniu, warunek w klauzuli WHERE sprawdza każdy odczytany ciąg z tabelą słownikową, w której są zapisane znaczniki. W przypadku nieznalezienia znacznika, program pobiera kolejne porcje danych z pliku i porównuje pobrane dane ze słownikiem.

4. Testy wydajności

Badanie wydajności przeprowadzono, wykonując 100 wywołań procedury odczytującej znaczniki DICOM dla każdego zdjęcia, zmierzono czasy oraz wyliczono ich średnią. Zdjęcia w formacie DICOM znajdowały się w tabeli w bazie danych, nie były wczytywane z dysku. W tabeli 4.1 przedstawiono wyniki uzyskane dla trzech najbardziej reprezentatywnych zdaniem autorów plików, różniących się w sposób istotny rozmiarem, źródłem pochodzenia oraz w mniejszym zakresie liczbą zawartych w nich znaczników.

Sposób pierwszy w dwóch próbach na trzy znalazł więcej znaczników, dla wszystkich prób czas był podobny (średnia wyniosła ok. 0,3 s). Sposób drugi w dwóch próbach na trzy znalazł ok. 75% znaczników wykrytych przez sposób pierwszy.

Tabela 3

Porównanie dwóch sposobów odczytywania danych z pliku DICOM

	Rozmiar pliku	Sposób pierwszy			Sposób drugi		
		Czas	Liczba znaczników	Liczba operacji SELECT dla jednego wywołania procedury	Czas	Liczba znaczników	Liczba operacji SELECT dla jednego wywołania procedury
Przykładowe zdjęcie z Internetu	16 kb	0,140s	59	1117	0,067s	44	403
Przykładowe zdjęcie z Internetu ze strony Osirix	1461 kb	0,037s	74	443	0,047s	74	490
Przykładowe zdjęcie z WSRM w Łodzi	8195 kb	0,133s	125	1162	1,833s	97	5192
Średnia		0,103s		907	0,649s		2028

Rozbieżność czasu przetwarzania pomiędzy poszczególnymi próbami jest znaczna. Dla przykładowych zdjęć pobranych z Internetu czas wyszukania znaczników wyniósł 0,067 s, a dla zdjęcia wykonanego w Wojewódzkiej Stacji Ratownictwa Medycznego w Łodzi czas wyniósł 1,833 s. W pierwszym sposobie czas przetwarzania wraz ze wzrostem rozmiaru pliku

waha się pomiędzy 0,037 s a 0,140 s, w drugim sposobie różnica już jest bardzo duża, czas zmienia się od 0,047 s do 1,833 s.

Najistotniejszy wpływ na średni czas wykonania pojedynczej procedury ma liczba zapytań, które są generowane podczas przetwarzania danych. Patrząc na wyniki w tabeli 3, można zauważyć, że bardziej optymalną metodą dekompozycji pliku DICOM jest pierwszy ze sposobów zaprezentowany w tym artykule. W tym podejściu średni czas wykonywania całego procesu był znacząco krótszy, a także liczba wykonywanych operacji jest znacznie mniejsza. Jak widać w tabeli 4.1, podczas analizy pliku dwoma różnymi metodami wykryta zostaje inna liczba znaczników. Różnice te wynikają z niejednoznacznej implementacji protokołu DICOM na różnych urządzeniach, co powoduje, że w prezentowanych danych znajdują się znaczniki, które nie są opisane w standardzie. Dlatego też pierwszy sposób odczytywania danych odnajduje zawsze większą liczbę znaczników niż sposób drugi. Różnica ta wynika z faktu, że sposób drugi korzysta ze zdefiniowanej wcześniej, zgodnej ze standardem, tabeli słownikowej, co automatycznie zawęża liczbę znalezionych atrybutów tylko do tych, które zawarte są w tym słowniku. Jeżeli implementacja parsera, przedstawiona w powyższym artykule, miałaby być uniwersalna, należy wykorzystać sposób ze wcześniej zdefiniowaną tabelą znaczników. Powoduje to, że niezależnie od urządzenia, z którego są pobierane pliki, wyniki dekompozycji będą do siebie zbliżone co do postaci formalnej. Natomiast, jeżeli taki parser byłby dedykowany dla systemu z wcześniej określonym urządzeniem, wtedy można brać pod uwagę sposób pierwszy. Można również opracować słownik rozszerzający standard na podstawie analiz przeprowadzonych pierwszą z metod.

5. Podsumowanie

W pracy przedstawiono dwa sposoby dekompozycji pliku zapisanego w formacie DICOM do tabel schematu relacyjnego. Zaproponowano hierarchiczne wykrywanie znaczników oraz sprawdzanie ze słownikami znaczników i typów atrybutów. Obydwie metody miały za zadanie odczytanie danych znajdujących się w pliku DICOM oraz rozwiązanie problemu niepodania długości pola wartości (Value Length). Obydwa sposoby spełniają swoją rolę, jednak ani jedno, ani drugie rozwiązanie nie jest stuprocentowo skuteczne. W trakcie prowadzenia prób na wielu różnych plikach, zapisanych w wyjściowym formacie, pojawiały się sporadycznie przypadki złej walidacji znaczników. Częściowym rozwiązaniem problemu było przekierowanie niezdekomponowanego fragmentu pliku do dodatkowego pola w tabeli. Powodowało to zarówno brak wystąpienia wyjątku (przerwanie przetwarzania), jak i brak utraty fragmentu danych. W każdym przypadku podstawowe

znaczniki były wykrywane poprawnie. Nie występowały również błędy w odszukiwaniu danych obrazowych. Należy zaznaczyć, że podobne problemy występowały w przypadku weryfikacji dekompozycji źle zinterpretowanych plików przy zastosowaniu narzędzi ORACLE. Ponieważ wyjściowym formatem dla bibliotek ORACLE jest typ XML, spowodowało to, że nie zostały wygenerowane znaczniki docelowego formatu, a w części z wygenerowanych właściwa informacja była uzupełniona o nadmiarowe dane (dotyczyło to pól znakowych).

BIBLIOGRAFIA

1. http://www.sno.phy.queensu.ca/~phil/exiftool/TagNames/DICOM.html.
2. http://medical.nema.org/.
3. Revet B.: DICOM Cook Book for implementation in Modalties. January 1997.
4. Szalinski M., Pelikant A.: Medical data migration and access management platform. System Modeling Control – 2005, Zakopane 2005. Problemy współczesnej nauki teoria i zastosowania. Informatyka, Akademicka Oficyna Wydawnicza Exit, Warszawa 2005.
5. Wojciechowski K., Pelikant A.: Zastosowanie protokołu DICOM oraz technologii J2EE w komunikacji między urządzeniami medycznymi a bazą danych. Bazy danych, nowe technologie, bezpieczeństwo, wybrane technologie i zastosowanie. WKŁ, Warszawa 2007, s. 229÷237.
6. Wang Ch., Huang Y.: DICOM Communication Mechanism and Engineering Project Integration Based on ESB. 2010 WASE International Conference on Information Engineering, August 2010, s. 119÷122.
7. Tee S. H.: Integrating DICOM Information Model with Risk Management Process Area of CMMI for Radiotherapy Applications. 2010 Second International Conference on Computer Research and Development, May 2010, s. 73÷76.
8. Selvarani A. G., Annadurai S.: Medical Image Retrieval by Combining Low Level Features and DICOM Features. International Conference on Computational Intelligence and Multimedia Applications, December 2007, s. 587÷589.
9. Agostinho R., Pereira M., Freire M.: Advanced Querying Architecture for DICOM Systems. Second International Conference on Systems and Networks Communications (ICSNC 2007), August 2007, s. 53.

10. Stanescu L., Ion A., Burdescu D., Brezovan M.: Algorithms and Results in Content-Based Visual Query of the Image Databases Resulting from Dicom Files. Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'06), September 2006, s. 119÷124.
11. Pelikant A.: Integracja danych medycznych za pomocą plików znacznikowych XML i DICOM. Technologie informatyczne w administracji publicznej i służbie zdrowia. Wydawnictwo SGH.
12. Horzelski W.: System rejestracji danych Dyżurowych w Klinice Medycyny Matczyno-Płodowej ICZMP w Łodzi. Bazy danych, struktury, algorytmy, metody, WKŁ, Warszawa 2006.
13. Zwoliński G.: Communication with users in the computerized recruitment system of candidates for higher education studies (SMC). Zakopane, 12-14 October 2009.

Recenzenci: Dr inż. Michał Kawulok
Prof. dr hab. inż. Bolesław Pochopień

Wpłynęło do Redakcji 31 stycznia 2011 r.

Abstract

The topic of this article is the implementation of the parser for DICOM file using T-SQL in SQL Server 2008. The article is organized as follows. The first part introduces the DICOM protocol and shows the way in which DICOM file is built (section 1). It also presents the differences between Little Endian and Big Endian way of write data (section 2). The second part includes two different approaches of the decomposition of file by parsing DICOM tags. In first section there is presented simple algorithm of the DICOM file analysis which is used to implement parser. First way of parsing is made without using predefined dictionary table with DICOM tags, and second approach is using that table. The final conclusion about usage of both presented methods, are offered in third part.

Adresy

Kacper PABJAŃCZYK: Politechnika Łódzka, Instytut Mechatriniki i Systemów Informatycznych, ul. Stefanowskiego 18/22, 90-924 Łódź, Polska, kacper.pabjanczyk@gmail.com.

Adam PELIKANT: Politechnika Łódzka, Instytut Mechatriniki i Systemów Informatycznych, ul. Stefanowskiego 18/22, 90-924 Łódź, Polska, adam.pelikant@p.lodz.pl.