An Approach for Equalizing Huge Volume Data Sources with the Network

V. Topalova, R. Raychev

Key Words: *Multimedia data flow; multimedia data archive flow; attr-DB backup flow; network transfer; DBMS.*

Abstract. *All well known techniques for compression could be applied to multimedia data flows transferred via a network. But the transmission of associated with them database archives remains a challenge, because their volume increases on and on in the time. This paper describes a system approach, which manages the transfer of huge, growing in real-time, multimedia archives via the network. The analytical model and clarifying graphics demonstrate significant, over 20 times, diminution of the data volume transferred via the network, compared to the classic transmission practice. This allows reducing both the risk of redundant reservations of network resources and the number of session escapes on timeout. In this way, the sources of huge data volumes could be easier adapted to the network.*

Introduction

Nowadays a lot of users generate multimedia data flows (MDF) with continuously increasing volume and some rhythm (monthly, weekly, daily etc.) as in the case of video surveillance and access control systems, systems for archive management and network transfer and so on. Associated attribute databases (attr-DB) maintain the service of MDF. The transmission of the multimedia data archives (MDA) has to take into consideration the dependencies between MDF and attr-DB backup flow (DBF). That is, the appearance of a new element of MDF provokes the growth of two flows - the MDF containing the element itself and the DBF flow, updated after the appearance of the element in question. In consequence, a new retransmission of the whole attr-DB backup is inevitable to follow MDF changes in permanent [10] (*figure 1*).

There are a lot of schemes for transmission of MDF and DBF flows in common. Widespread are the transfer, scheduled immediately after a MDF data unit creation, as in the case of real-time data transfer and the postponed MDF transfer. Usually, video surveillance systems use scheduled (one-time or recurring) and continuous archiving for the MDF flows [3]. Full backup database strategy is appropriate for relatively small DBF flows and offers the best approach for efficient recovery process [5].

However, if all well known techniques of MDF compression based on the temporal locality properties, could be used before the MDF transmission the transfer of their associated DBF remains a challenge, since there is no temporal locality dependency in DBF. For this reason, the attention is paid at this time to operations over attr-DB flow, such as structuring, creation of attr-DB backups, DBF archiving and traffic management of context-oriented DBF, in sense of correlation with the MDF [9,1,2,7].

This work was supported by the Technical University of Gabrovo Science Research Foundation grant E-906/2009-2011.

Figure 1 depicts a classic approach for transmission of MDA flows, consisting of two associated data flows - MDF (circles) and DBF (rectangles). Although the time interval Ti between two network sessions for MDA transfer could vary in general as in *figure 1*, this work examines the case with even distribution of network sessions (Ti=const=T). It has to be noted, that the DBF flow volume increases continuously (the nested rectangles on the figure), as the DBF flow depends on the MDF flow. It follows the rhythm of the MDF flow growth. A full backup of ever increasing attr-DB is made every network session.

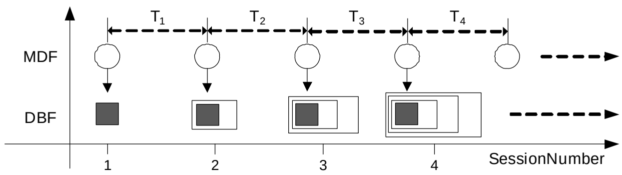


Figure 1. Classic approach for MDA transfer via the network

Hence, it is very important to look for approaches providing optimal network bandwidth usage for data sources proving to have incessantly increasing volumes [6,8,10]. The discussed one applies particular schemes for flow regulation of the attr-DB archives so that the full flow MDA = rMDF + rDBF arrives to the receiver with minimal data losses. The symbol r denotes a structuring and compacting operator over the corresponding flows, as discussed on *figure 4*.

Approach for Management and Transfer of Multimedia Data Archives

In the scope of the paper are the flow coordination problems at the network entrance. Methodical problems for determining the rhythm and volumes of the introduced in the network data arise to meet the end-user requirements [4].

The user requirements depend on the user tolerable level of data loss which can result from accidental deletion of data from the attr-DB, software problem, hardware problem and etc.

Most often, the rhythm of MDA archives generating is subordinated to the rhythm of MDF following calendar schedule - twenty-four-hour, weekly, monthly etc. It can be coordinated also with the constraints of competitive network processes in critical time periods.

The classic approach for transmission of rDBF consists of sending full backups FB of the attr-DB, neglecting potential database optimization. The proposed approach aims volume optimization of rDBF managing the transfer of MDA [9].

Instead of the systematic transmissions of full attr-DB archives, named FB-type archives, according to *figure 1*, the proposed approach, named F-D-T approach, offers transmissions of three different database archive types: FB-type (the same as in the classic approach), DB-type and TB-type, as marked on *figure 6*, zoomed on *figure 2* and *figure 3*. In any given instant, the classic approach transmits a FB-type archive, while the F-D-T approach can transmit a FB-type, a DB-type or a TB-type archive. The benefit in the F-D-T approach comes from the fact that the DB-type and TB-type archive volume are in general smaller than the FB-type archive volume [6,8,10].

The advantages of the F-D-T approach over the classic one could be estimated as follow.

First of all, the total number of sessions is equal for both classic and F-D-T approach, and it is

(1) $SessionNmb = i(M+1)(N+1)+j(N+1)+(k+1)$

where i stands for the number of FB-type transmissions, j - the number of DB-type transmissions in a $\mathbf{T_{FULL}(i)}$ interval and $\mathbf{k}$ - the number of TB-type transmissions in a $\mathbf{T_{DIFF}\ i,j+1)}$ interval (*figure 2*, *figure 3* and *figure 6*). The border values are $\mathbf{j=M}$ and $\mathbf{k=N}$. That is, the i-th $\mathbf{T_{FULL}}$ interval consists of $\mathbf{M}$ network sessions for DB-type archive transfer. The $\mathbf{T_{DIFF}(i,j)}$ interval consists of $\mathbf{N}$ network sessions for TB-type archive transfer. To simplify it is supposed that the size of $\mathbf{TB(i,j,k)}$ for every $\mathbf{(i,j,k)}$ is constant and equal to $\mathbf{TB_{sz}}$. Other scenarios are also possible.

Next, the ClassicVolume $\mathbf{(i,j,k)}$ is:

$$(2)\quad ClassicVolume(i,j,k) = SessionNmb(SessionNmb+1)\frac{TB_{SZ}}{2} = \\ = i(M+1)(N+1)+j(N+1) \\ +(k+1)\big((i(M+1)(N+1) \\ +j(N+1)+(k+1))+1\big)\frac{TB_{SZ}}{2}$$

Let us compare this volume with the F-D-T volume. Firstly, let us find the volume of FB-type archives.

The time interval ($\mathbf{T_{FULL}}$) contains only one FB-type archive, which is sent in the first network session of that interval. This archive in the i-th interval $\mathbf{T_{FULL}(i)}$ is $\mathbf{FB(i)}$ (*figure 2*). Bringing up to date of $\mathbf{FB(i)}$ is provided till the end of the interval $\mathbf{T_{FULL}(i)}$ with cumulative archive types - DB- and TB-type archives throughout the following network sessions of the same interval $\mathbf{T_{FULL}(i)}$.

The total volume Acc(i) accumulated between instances (i) and (i+1) for the border values $\mathbf{j=M}$ and $\mathbf{k=N}$ is:

$$(3)\quad Acc(i) = (M(M+1)(N+1) + 2MN + 2N)\frac{TB_{SZ}}{2}\ .$$

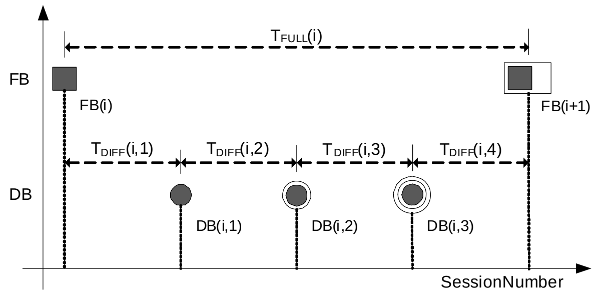
The difference $\mathbf{FB(i+1)}$ - $\mathbf{FB(i)}$, which is equal to

$$(4)\quad Acc(i)+TB_{SZ} = (M(M+1)(N+1) + 2MN + 2(N+1))\frac{TB_{SZ}}{2}$$

gives a recursive way to estimate the FB-type archive volume FB(i) in the moment i assuming an initial condition $\mathbf{FB(1)} = \mathbf{TB_{SZ}}$: FB(i) = FB(i-1) + $C_{MN}TB_{SZ}$, where

$$(5)\quad C_{MN} = \frac{1}{2}\ (\ (M(M+1)(N+1) + 2MN + 2(N+1))$$

Now, let us find the volume of DB-type archives. DB-type archives are related only to their preceding FB-type archive (*figure 2*). The DB-type archives $\mathbf{DB(i,j)}$, $\mathbf{j=[1,M]}$, contain the changes, made in the attr-DB after $\mathbf{FB(i)}$. Any following DB-type archive $\mathbf{DB(i,j)}$, $\mathbf{j=[2,M]}$, contains the preceding $\mathbf{DB(i,j-1)}$ and all TB-type archives after $\mathbf{DB(i,j-1)}$ after $\mathbf{FB(i)}$ (*figure 6*). This is depicted as nested circles on *figure 2*.



* FB(i) stands for FB-type archive in the i-th interval $T_{FULL}(i)$

* DB(i,j) stands for DB-type archive in the j-th interval $T_{DIFF}(i,j)$

Figure 2. DB-type archives in a T_{FULL} interval

Hence, $DB_{SZ}(i,\ j)$ conforms to the following recursive equation:

$$(6)\quad DB_{SZ}(i,\ j) = DB_{SZ}\,(i,\ j\text{-}1) + (N\ +\ 1)TB_{SZ}$$

with the initial condition $DB_{SZ}(i,\ j) = (N\ +\ 1)TB_{SZ}$. In consequence, the total volume of all DB-type archives transferred after the interval FB(i) is:

$$(7)\quad \Sigma^{j}_{k\text{-}1}\, DB_{SZ}(i,\ k) = \frac{j(j+1)}{2}\ \ (N\ +\ 1)TB_{SZ}$$

Finally, find the volume of TB-type archives. Every TB-type archive $\boldsymbol{TB(i,\ j,\ k)}$, $\mathbf{k=[1,N]}$, contains the committed changes in the attr-DB in the interval T_{TRAN} coming after its preceding archive (independently from its type - FB-type, DB-type or TB-type). On the *figure 3* three cases of dependencies in the TB-type archive formation are seen:

- Case A: TB(i,0,1), made in the i-th interval $T_{FULL}(i)$ and the first interval $T_{DIFF}(i,1)$, accumulates the differences after FB(i).
- Case B: TB(i,0,2), made in the i-th interval $T_{FULL}(i)$ and the first interval $T_{DIFF}(i,1)$, accumulates the differences after TB(i,0,1).
- Case C: TB(i,1,1), made in the i-th interval $T_{FULL}(i)$ and the second interval $T_{DIFF}(i,2)$, accumulates the differences after DB(i,1).

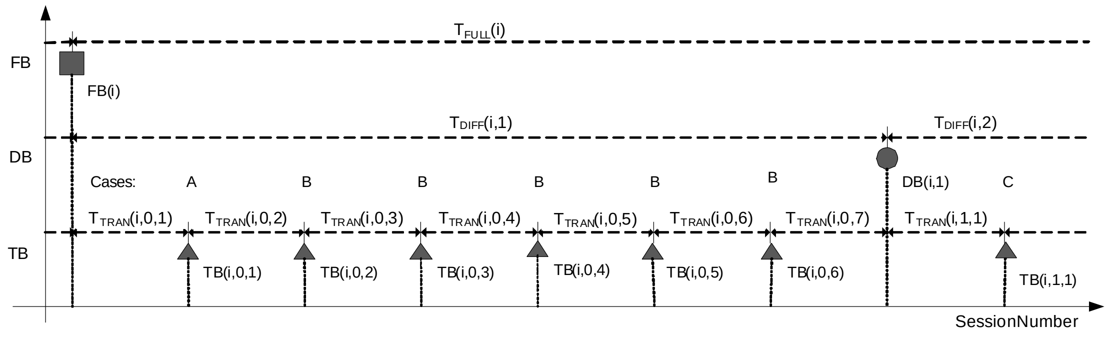
Therefore, the volume of all TB-type archives transferred after the last FB-type archive FB(i), is

$$(8)\quad j * N * TB_{SZ} + k * TB_{SZ}.$$

Added to the above volume, it gives the part V(i) of the volume TotalFDTVolume(i,j,k), transmitted after the last transferred FB(i) archive:

$$(9)\quad V(i) = (j(j+1)(N+1) + 2jN + 2k)\frac{TB_{SZ}}{2}\ .$$

It seems at first (*figure 6*), that the proposed approach increases the whole volume of the flow rDBF, rather than to decrease it. This is not true, because the interval $\mathbf{T_{FULL}}$ is obviously longer than $\mathbf{Ti=const=T}$ on *figure 1*, since $\mathbf{T_{TRAN}=T}$. As a drawback, the impossibility to transmit a DB-type or a TB-type archive after $\mathbf{FB(i)}$ breaks the successful restoring of the attr-DB till the interval $\mathbf{T_{TRAN}(i,j,k)}$ and restricts the restoring options either till the preceding interval $\mathbf{T_{TRAN}(i,j,k-1)}$ or till the



* TB(i,j,k) stands for TB-type archive in the k-th interval $T_{TRAN}(i,j,k)$ and the (j+1)-th interval $T_{DIFF}(i,j+1)$ for j?0

Figure 3. Sequence of TB-type archives in a TDIFF interval

moment of its successful transmission.

An equalizing arrangement is proposed on *figure 4* for the F-D-T approach implementation. Its functionalities are covered by a set of software components, forming the threads in the MDF and the associated DBF processing.

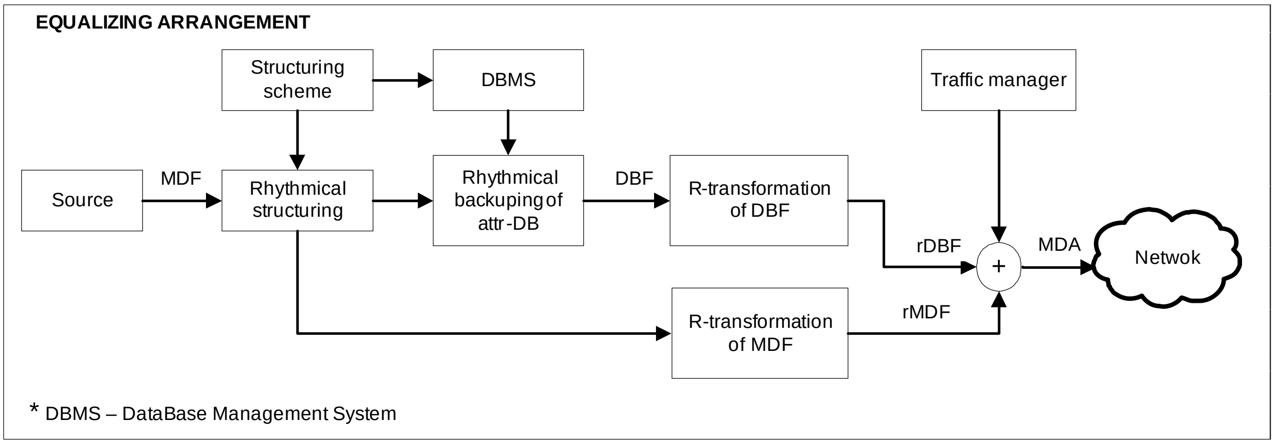


Figure 4. Equalizing arrangement between the source and the network

Although the threads serve MDF and DBF flows separately, they perform the operations structuring, archiving, scheduling and dispatching synchronously and produce at the end of the equalizing arrangement a common MDA flow. The component „Structuring scheme" structures the two associated flows - MDF and DBF. It operates with the flows under a common F-D-T scheme. But if the component „Rhythmical structuring" structures the MDF flow into directories and executes one level of structuring, the DBF flow is processed rather differently. First, the component „Structuring scheme" sends queries to the DBMS (component „DBMS") in order to organize the internal backuping of the attr-DB flow through DBMS commands. Next, the result DBF flow (attr-DB backups) is processed from the component „Rhythmical structuring", which now allows further synchronized network transfer of the MDF and DBF flows. The already structured flows pass separately through R-transformation respectively the components „R-transformation of DBF" and „R-transformation of MDF" with common functionality. The R-transformation represents identical archiving procedure with opportunities for setting maximum archive size and maximum transfer unit size. Finally, the component „Traffic manager" merges the MDF archive flow (rMDF) and the DBF archive flow (rDBF) into a combined MDA archive flow and performs scheduling and dispatching of the MDA archive flow at the entrance of the network.

On *figure 5* is depicted the software hierarchy of the equalizing arrangement. It includes object- and flow-oriented components (separate libraries), organized in application programming interface (API) layer with three levels of operation. The objects in the equalizing arrangement are multimedia data items (pictures, video, etc.), DBMS, attr-DB (associated with the MDF), disk storages, rMDF archives, rDBF archives and log files. The components „Queries toward the DBMS", „Disk Storages Assignment" and „Log Handler", placed at level „Execution", operate respectively with DBMS, disk storages and log files. The component „Archive Handler" placed on level „Management", manages the operations structuring and archiving of the MDF and DBF flows, and also merging, organizing, scheduling and dispatching of the resulting rMDF and rDBF archive flows. Its subordinated component „Archiving and Transferring", placed at level „Execution", is responsible for the operations, archiving of the MDF and DBF flows and network transfer of the MDA flow by means of. The working mode of the equalizing arrangement is set at the level of administration. The API layer supports both the automatic and manual mode for MDF processing.

The API layer has no requirements for human operations, although manual operations are also supported.

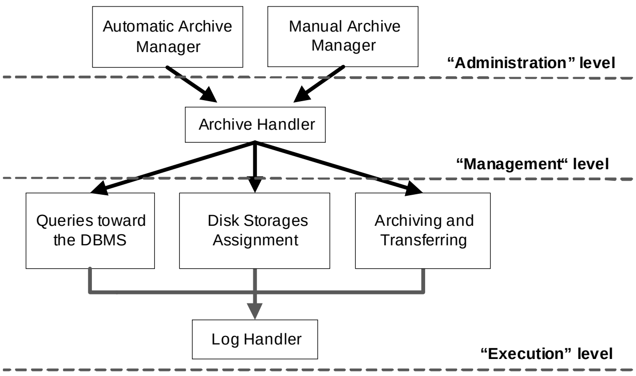


Figure 5. Software hierarchy of the equalizing arrangement

Method for Management of the Network Transfer of Huge Multimedia Data Volumes

The main moments in the proposed method are the following six:

Investigating MD flows

1. Analyzing the character of the MD flows, the dependencies between the different logical fragments, the network environment requirements - bandwidth, time intervals of heavy network loading and the priorities of the different MD types.

Selecting procedure for rDBF formation.

2. Selecting a backup strategy for the attr-DB.

3. Selecting a scheme for formation of rDBF - a combination of the FB-, DB- and TB-type archives and of a calendar schedule for transmission of rDBF.

Selecting procedure for rMDF construction.

4. Selecting a scheme for construction of rMDF in accordance with their structural organization (most often built upon hourly and calendar directories).

Selecting mix procedure for the MDA flow.

5. Coordinating the archiving and dispatcher procedures for the resulting MDA flow.

Selecting mechanism for MDA flow retransmission.

6. Selecting a mechanism for retransmission of the resulting MDA flow based on the MDA transfer timestamp feedback.

Algorithm for Management of the Network Transfer of Huge Multimedia Data Volumes

The algorithm for management of the network transfer of huge multimedia data volumes consists of three key procedures: procedure for preparation of FDT schedule, procedure for automatic MDA formation and procedure for archive retransmission after signal received from timestamp feedback.

Procedure „Preparation of F-D-T schedule" - the preparation of a schedule for formation of rDBF with FB-, DB- and TB-type archives, named as F-D-T scheme, is performed at the „Administration" level (*figure 5*). This procedure sets a scenario for generation of the merged MDA flow. The selected scenario is implemented by the operations at the lower layers in the software hierarchy of the equalizing arrangement

```
Procedure Preparation of F-D-T schedule()
    Set initial configuration()
    if (FB does not exist() || Exists FB archive created before
        an interval >= T_FULL())
          return Procedure for automatic MDA formation(FB)
    else if ((DB does not exist () && Exists FB archive created
              before an interval >= T_DIFF()) ||
              Exists DB archive created before an interval >=
             T_DIFF())
               return Procedure for automatic MDA
                      formation(DB)
    else if (TB does not exist () || Exists TB archive created
             before an interval = T_TRAN())
              return Procedure for automatic MDA
                     formation(TB)
    else if (Exists TB archive created before an interval >=
           T_TRAN())
             return Procedure for automatic MDA
                    formation(DB)
   endif
End
```

Procedure for automatic MDA generation represents basic functionality at level „Management" (*figure 5*). The archive type, which has to be generated, depends on the offered and launched Procedure ***„Preparation of F-D-T schedule"*** and on the mechanism for timestamp monitoring of the rMDF and rDBF flows. In particular, the procedure for archive retransmission after feedback signal is vital functionality for the successful network transmission of the MDA flow.

```
Procedure Automatic MDA formation (parameters)
     Get current settings(parameters)
     if NOT(„Success")
          return „End of Procedure"
    endif
      if NOT(Active timestamp monitoring())
          return „End of Procedure"
     endif
      if (Second archive copy required())
            return Provide storage resources for saving second
                   archive copy()
     endif
      Set configuration for BACKUP creation of attr-DB()
      Create BACKUP of attr-DB()
      if (Succesful BACKUP creation of attr-DB())
            return Delete outdated BACKUPS of attr-DB
                   (parameters)
     else return „End of Procedure"
      endif
      Set configuration for DBF archiving()
      DBF archiving and transfer of rDBF(max archive size,
      max transfer unit size)
     Procedure for archive retransmission after feedback
    signal (rDBF)
      MDF archiving and rMDF transfer(max archive size,
     max transfer unit size)
    Procedure for archive retransmission after feedback
   signal (rMDF)
   Update older rDBF archives according to F-D-T
   scheme()
End
```

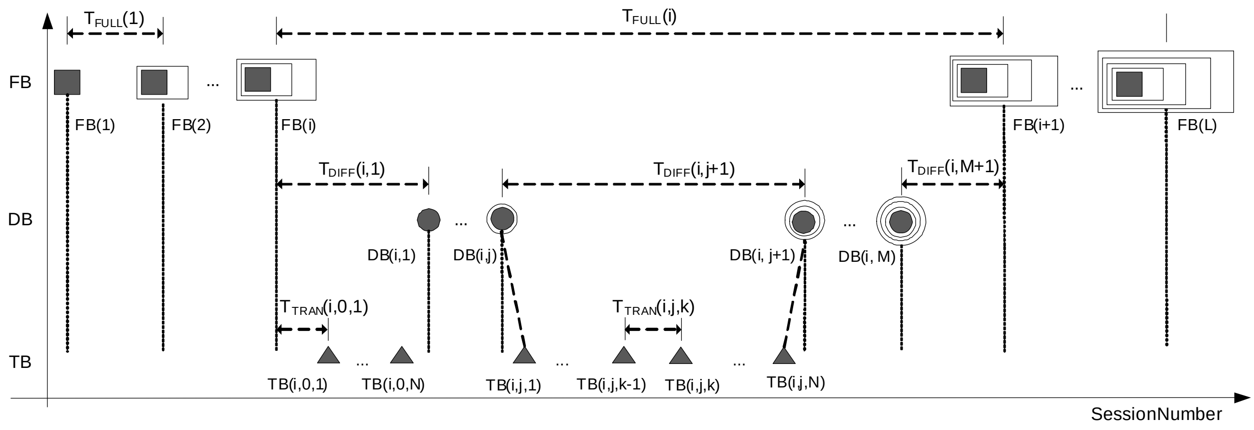


Figure 6. Proposed F-D-T approach for MDA transfer via the network

Procedure for archive retransmission after feedback signal (rMDF)
Update older rDBF archives according to F-D-T scheme()
End
Procedure Archive retransmission after feedback signal (parameters)
Get current settings(parameters)
Archive retransmisson after feedback signal()
if NOT(„Successful archive transmission or retransmission")
return „End of Procedure"
else
Update timestamps()
if (Second archive copy required() &&
Free disk storage resources available())
return Create second archive copy()
endif
End

Evaluation of the Approach Efficiency

It can be proved here that the proposed F-D-T approach provides the opportunity for restoring the attr-DB in every moment to transferring less volume of the rDBF. Let us consider the case of parity of the time intervals **Ti=const=T** (*figure 1*) and $\mathbf{T_{TRAN}}$ (*figure 6*). The other two cases for $\mathbf{T<T_{TRAN}}$ and $\mathbf{T>T_{TRAN}}$ will not be discussed here.

Actually, if the attr-DB has to be restored in the moment

(10) $t_0 = \sum_{s=1}^{i} t_{FULL}(s) + \sum_{u=0}^{j} t_{DIFF}(i,u) + \sum_{v=1}^{k} t_{TRAN}(i,j,v)$

then instead of the total volume TotalFDTVolume(t_0) of archives rDBF, transferred up to the moment t_0

(11)
$$TotalFDTVolume\ (t_0) = \sum_{s=1}^{i-1} FB(s) + FB(i) +$$
$$+ \sum_{s=1}^{i-1}\sum_{u=1}^{M} DB(s,u) + \sum_{u=1}^{j-1} DB(i,u) + DB(i,j) +$$
$$+ \sum_{s=1}^{i-1}\sum_{u=0}^{M}\sum_{v=1}^{N} TB(s,u,v) + \sum_{u=0}^{j-1}\sum_{v=1}^{N} TB(i,u,v) + \sum_{v=1}^{k} TB(i,j,v)$$

the restore function of the attr-DB will use only the volume FDTRestore(t_0), equal to

(12) $FDTRestore\ (t_0) = FB(i) + DB(i,j) + \sum_{v=1}^{k} TB(i,j,v),$

which is obviously smaller for every t_0, because FDTRestore(t_0) contains only three components - the 2[nd], 5[th] and 8[th] - of TotalFDTVolume(t_0) in the equation (11).

Let us find the numerical evaluation of the transferred volume TotalFDTVolume(t_0) up to the moment t_0. The total volume of all transmitted FB-type archives including **FB(i)**, is:

(13) $TB_{SZ}(i + \frac{i(i\text{-}1)}{2} C_{MN}) = i\ (1 + \frac{i(i\text{-}1)}{2} C_{MN})TB_{SZ},$

where C_{MN} is evaluated in equation (5).

Therefore, the equation (12) becomes simply

(14) $FDTRestore\ (i,j,k) = (1+(i-1)C_{MN} + j(N+1)+k)TB_{SZ}$

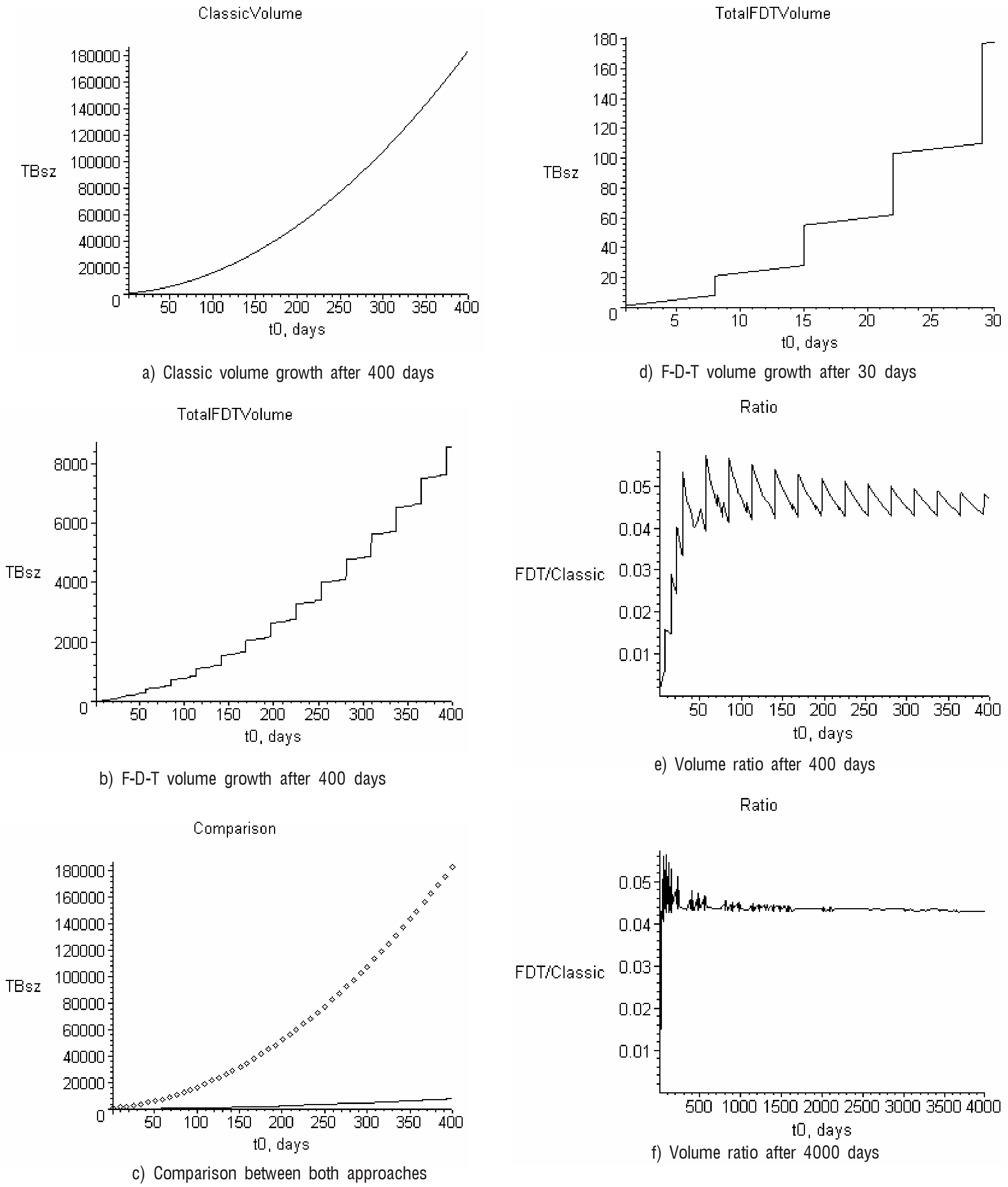
and the equation (11) is updated to

(15) $TotalFDTVolume\ (i,j,k) = \mathrm{i}\ (1 + \frac{i\text{-}1}{2} C_{MN})TB_{SZ} +$
$+ (N + 1)(M(M+1)(i-1) + j(j+1)\frac{TB_{SZ}}{2} +$
$+ ((i-1)N(M+1) + jN + k)TB_{SZ}.$

The curves in *figure 7* are plotted in the particular case for weekly-made TB-type and monthly-made DB-type archives. In this case N=6 and M=3 respectively. The diagrams show the efficiency enhancement of the proposed F-D-T approach over the classic one. The F-D-T algorithm outperforms more than twenty times the classic algorithm beginning from the 59-th day approximately (*figure 7e* and *figure 7f*).

The classic volume has a significantly different model of growth in comparison to the F-D-T volume (*figure 7c*). Whereas the first one increase is monotonous (*figure 7a*), the second has considerable cyclic delay of the volume increase (*figure 7b* and *figure 7d*). The F-D-T approach offers a perspective MDA transfer model, because it keeps the archive reservation up to date and a top priority at lesser network traffic price. This is achievable due to the usage of a scheme for transfer of associated and context-oriented multimedia flows, based on a calendar schedule, which takes into account the diversity of database backup management schemes.

Figure 7e and *figure 7f* depict the ratio between the classic volume and the F-D-T volume. The F-D-T algorithm has always volume savings: 167 times savings at the tenth day, 17 times (the absolute minimum) at the 59-th day, 20 to 23 times after



a) Classic volume growth after 400 days

d) F-D-T volume growth after 30 days

b) F-D-T volume growth after 400 days

e) Volume ratio after 400 days

c) Comparison between both approaches

f) Volume ratio after 4000 days

Figure 7. Experimental results on calendar schedule basis

one year. Approximately 700 days later, this algorithm achieves the asymptotic performance of twenty two times higher than the performance of the classic algorithm. This performance ratio is maintained until the exhaustion of one or more of the network or storage resources - network bandwidth, the database capacity, and the disk storage capacity.

Conclusions and Future Work

While the video surveillance systems use usually scheduled and continuous archiving for the MDF flows only [3], the practice necessitates the transmission of associated with them database archives. One of the solutions for this challenge is the proposed here F-D-T approach.

As the full backup database strategy is appropriate for relatively small DBF flows and offers the best approach for the efficient recovery process [5], the F-D-T approach goes beyond this limit. It gives an answer to the more important and wide-spread cases, when an application demands transfer of huge, growing in real-time, multimedia archives via the network.

The proposed F-D-T approach for the equalizing arrangement and the corresponding algorithm for transfer management of huge, growing in real-time, multimedia archives reduce considerably the amount of MDA flow. In this way the sources of huge data volumes could be easier adapted to the network. Moreover, the reduced volumes diminish the risk of redundant reservations of network resources and the hazards of session escape on timeout.

The method applies particular schemes for flow regulation of the attr-DB archives so that the total flow arrives at the receiver with minimal data losses. The suggested equalizing arrangement implements a layer around DBMS to enhance the recovery technique performance of database archiving. Main advantages of this layer in the equalizing arrangement between the source and the network result from the fact, that although the FB-type archive of the attr-DB is self-sufficient for data restoring at the receiver side and makes useless all preceding FB-type archives, it is inefficient for network transfer, due to its continuously increasing volume.

References

1. A Tandberg Data white Paper on Backup and Archival Storage Best Practices. White Paper, 2010, http://www.exabyte.com/support/online/documentation/whitepapers/basicbackup.pdf.
2. Babineau, B. Separating Backup and Archive Processes to Meet New Challenges of Information Management. White Paper, Enterprise Strategy Group, September, 2006, http://www.ironmountain.com/resources/tape/infomanage.pdf.
3. Cisco Video Surveillance Manager User Guide. Release 6.3. Cisco Systems, 2010. http://www.cisco.com/en/US/docs/security/physical_security/video_surveillance/network/vsm/6_3/user_guide/vsmug_6_3_0.pdf.
4. Fernandez, G., J. Carrion, L. Aguilar, I. Collado. A New Approach to Dynamic Load Balancing Across Multimedia Servers. - *WSEAS Transactions on Computers*, 5, 2006, Issue 11.
5. IBM Tivoli Storage Manager for Databases, Data Protection for Microsoft SQL Server, Version 6.3. IBM Corporation, 2011. http://publib.boulder.ibm.com/infocenter/tsminfo/v6r3/index.jsp?topic=%2Fcom.ibm.itsm.db.sql.doc%2Fdps_con_bup_types.html.
6. Mistry, R., C. Amaris, A. Minty. SQL Server 2005 Management and Administration. Sams, 2007.
7. Optimizing Your Backup and Recovery Environment. White Paper, Symantec, 2010. http://viewer.media.bitpipe.com/939987896_418/1293479892_267/Your_Backup_is_Not_Your_Archive.pdf.
8. Rizzo, T., A. Machanic, J. Skinner et al. Pro SQL Server 2005. Apress, 2006.
9. Tucker, N. Disaster Recovery Techniques for SQL Server. GlobalKnowledge, 2009, http://images.globalknowledge.com/wwwimages/whitepaperpdf/WP_Tucker_DisasterRecovery_P.pdf.
10. Whalen, E., M. Garcia, B. Patel et al. Microsoft SQL Server 2005 Administrator's Companion. Microsoft Press, 2006.

Manuscript received on 30.06.2011

Vanya Topalova *(born 1981) has a M.Sc. degree in Computer Systems and Technologies. She is a PhD student. Her scientific interests are computer networks, critical network transfer in real-time and equalization of huge volume data sources with the network. She is MCTS and has active certificates at the area of .NET Framework 3.5, Windows Forms Applications and Microsoft SQL Server 2008, Database Development. Her professional skills include object-oriented programming (C# and C++), Windows- and Web-based programming and relational databases.*

Contacts:
Department of Computer Systems and Technologies
Technical University of Gabrovo
Gabrovo, 4 „H. Dimitar" Str.
Phone: +359 66 827 209
e-mail: vtopalova@tugab.bg

Radoslav Raychev *(born 1949) has a PhD degree in the area in microprocessor techniques from 1979. He is associate professor at the Technical University of Gabrovo and Head of the Computer Systems and Technologies Department at the same university. His scientific and research interests include computer architectures, industrial informatics and interfaces.*

Contacts:
Department of Computer Systems and Technologies
Technical University of Gabrovo
Gabrovo, 4 „H. Dimitar" Str.
Phone: +359 66 827 597
email: raychev@tugab.bg